Discovery Foundation Models: Toward Open-Ended Discovery Intelligence

Ling Yang Zhenfei Yin Yingcheng Wu

DFM Scientist Collaboration Program

We work with scientists and experimental platforms on open problems with real scientific value and real validation conditions. If you have such a problem, or the data, code, compute, or lab conditions to investigate one, we would like to hear from you.

phai-labs.com/collaborate

Build with us github.com/Gen-Verse/DFM-Plans

Contact yang@phai-labs.com

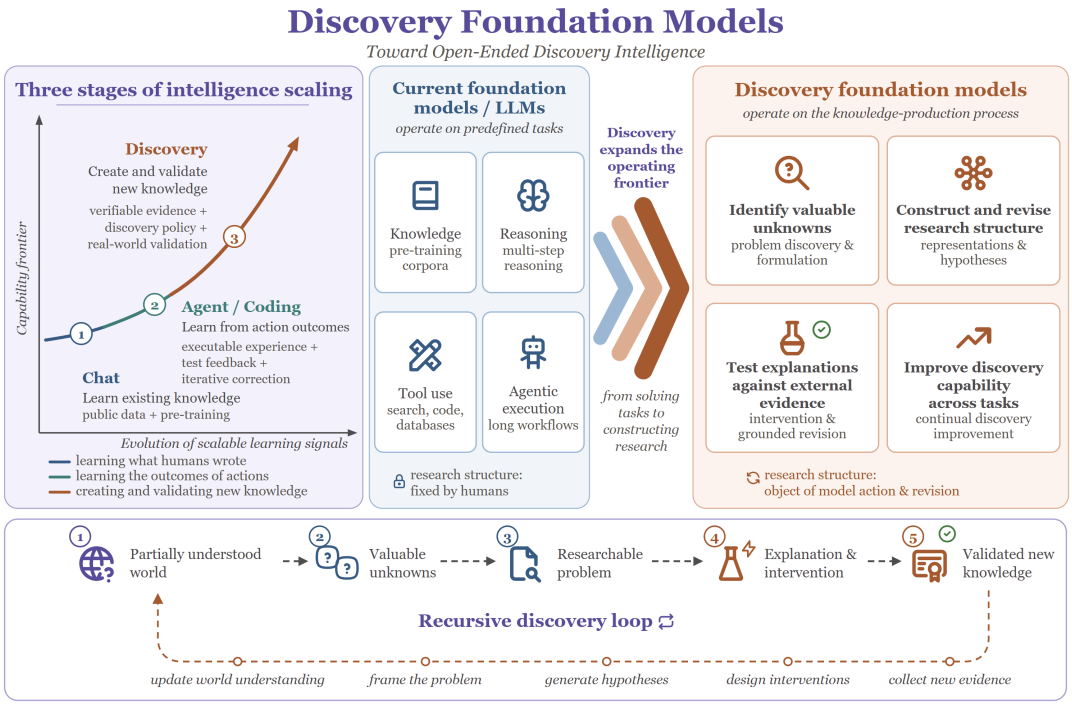

Figure 1 | **From task solving to discovery intelligence.** The left panel illustrates three stages of intelligence scaling, from learning existing knowledge (*Chat*), to learning from action outcomes (*Agent/Coding*), and ultimately to creating and validating new knowledge (*Discovery*). Current foundation models primarily operate over predefined tasks through knowledge, reasoning, tool use, and agentic execution, whereas *Discovery Foundation Models* extend the operating frontier to the knowledge-production process itself: identifying valuable unknowns, constructing and revising research structure, testing explanations against external evidence, and improving discovery capabilities across tasks. The bottom panel depicts the resulting *recursive discovery loop*, in which validated findings continually update world understanding and seed subsequent rounds of problem discovery, hypothesis generation, intervention, and evidence collection.

Abstract

Foundation models have progressed from learning and reasoning over existing knowledge, to increasingly learning through action, tool use, and outcome feedback. We argue that the next frontier is a further transition: from solving and acting within problems specified by humans to participating in the process by which new problems, representations, explanations, and knowledge are created. We refer to this capability as **Discovery Intelligence**. We formulate **Discovery Foundation Models (DFMs)** as general-purpose model systems for open-ended discovery. A DFM operates over a revisable research state and supports seven coupled capabilities spanning problem discovery, formulation, representation construction, hypothesis formation, intervention, evidence-grounded revision, and continual discovery improvement. We instantiate this framework with **Zetema**, which couples explicit research-state dynamics, verification and experimental gating, external grounding, and cross-task Discovery Skill evolution. We further ground the framework with **GALILEO**, a real therapeutic-discovery system in which Dry-Lab reasoning, robotic and hands-on Wet-Lab experimentation, external biological evidence, and iterative hypothesis and design revision form a closed physical discovery loop. We then formulate a unified approach to capability formation and process-centered evaluation, enabling discovery behavior to be trained, improved, and measured beyond final-answer performance. Together, these components establish discovery as a learnable, executable, and evaluable capability of foundation-model systems. We view this shift as a broader progression in intelligence scaling: from learning over existing knowledge, to learning from action outcomes, and ultimately to participating in the construction, testing, and revision of the structures through which new knowledge is discovered.

Contents

1 Introduction **4**

2 From Generalist Problem Solving to Discovery Intelligence **6**

2.1 Generalist Capability within Predefined Research Structures 6

2.2 Science as a Capability-Forming Environment 6

2.3 Three Missing Transitions 7

3 Discovery Foundation Models: Defining Discovery Intelligence **8**

3.1 Problem Setting and Formal Definition 8

3.2 Discovery Capabilities 9

3.3 System Boundary 10

3.4 Relation to Existing Scientific AI Paradigms 10

4 Discovery Process: Operationalizing Discovery Intelligence **11**

4.1 From Unknowns to Researchable Problems 12

4.2 Representations and Competing Explanations 13

4.3 Intervention and Evidence-Grounded Revision 13

4.4 Episode Output and Candidate Discovery Lessons 14

5 Zetema: A System Instantiation of Discovery Intelligence 14
5.1 Within-Task Research-State Dynamics 15
5.2 Explicit Research State and Attribution 16
5.3 Discovery Skill Memory 16
5.4 Research World Model and Experimental Gating 17
5.5 Validated Cross-Task Update 18
5.6 Dry-Lab and Wet-Lab Grounding 19
5.7 Empirical Case Study: A Real Dry-Lab/Wet-Lab Discovery Loop 19
6 Capability Formation: Training Discovery Operations 23
6.1 Training Data and Interactive Research Environments 23
6.2 Learning Objectives and Scientific Feedback 25
6.3 Unified Training Procedure 27
6.4 Process-Level Scaling and Resource Allocation 27
6.5 Continual Skill Learning and Transfer 29
7 Capability Evaluation: A Process-Centered Protocol 29
7.1 Evaluation Target: Current Progress and Future Capability 30
7.2 Stage-Wise Process Evaluation 31
7.3 Efficiency and Resource-Matched Evaluation 31
7.4 Continual Improvement and Transfer 32
7.5 Benchmark Construction and Controls 33
8 Analysis: Research Horizons and Grounding Regimes 33
8.1 Digital Discovery 33
8.2 Simulation-Grounded Discovery 34
8.3 Embodied and Physical Discovery 35
8.4 Recursive Discovery Systems 35
9 Discussion: Epistemic Boundaries, Governance, and Recursive Risk 36
9.1 Epistemic Status and Independent Validation 37
9.2 Provenance and Auditability 37
9.3 Scoped Authority and Human Oversight 38
9.4 Recursive Update Risk 38
10 Conclusion 39

1. Introduction

Foundation models have become general interfaces to knowledge work. Large-scale pretraining, post-training, multimodal learning, coding, tool use, and agentic execution allow them to synthesize literature, reason over technical problems, analyze data, run software, and coordinate long workflows [5, 7, 33, 50, 54]. In science, these capabilities already support protein and materials modeling, weather prediction, mathematical and programmatic reasoning, literature-grounded analysis, and increasingly automated experimentation [2, 4, 6, 21, 22, 27, 30, 41, 57].

Most of these systems begin from a research structure that people have already chosen. The question is stated, the variables are supplied, the objective is fixed, tools are exposed through an interface, and an evaluator determines whether the output is acceptable. Models can search and optimize inside this structure with increasing sophistication. They are much weaker when progress requires changing the structure itself.

That distinction matters in open-ended discovery. An apparent anomaly may be a measurement artifact. Two explanations can fit all existing observations because the available observable is non-identifying. A benchmark may reward a proxy. A persistent failure can result from a missing variable rather than a weak optimizer. In such cases, the next useful action is not another answer inside the current task. The system must decide what is actually unknown, how the problem should be posed, which representation makes competing mechanisms expressible, and what intervention could force them to disagree [8, 9, 34, 44].

We refer to this broader target as *Discovery Intelligence*. Generalist problem solving asks how broadly and deeply a model can solve supplied tasks. Discovery Intelligence asks whether a model system can construct, test, and revise the process through which a partially understood world becomes validated knowledge. The distinction is increasingly visible in scientific-agent evaluations: long research workflows can be executed successfully while evidence integration, refutation-driven revision, and long-horizon reliability remain fragile [13, 31].

Science is a useful capability-forming environment for this target because it exposes incomplete specifications that ordinary benchmarks often remove. Questions can be underspecified, variables hidden, mechanisms observationally equivalent, interventions costly, evaluators incomplete, and outcomes delayed. Evidence also arrives from environments that the model cannot rewrite after seeing the result. These properties turn formulation, representation, experiment design, failure attribution, and revision into observable decisions rather than rhetorical qualities of a final answer [14, 15, 25, 40, 43, 46, 59].

We introduce *Discovery Foundation Models* as a model-system category for this setting. A DFM identifies valuable unknowns, formulates researchable problems, constructs and revises representations, forms testable explanations, designs informative interventions, updates the research state from external evidence, and improves these operations across tasks and domains. The category is broader than hypothesis generation and different from simply applying a foundation model to scientific data. It concerns which parts of knowledge production are fixed inputs and which can become objects of model action and revision [3, 15, 47].

We then instantiate the framework with *Zetema*. Zetema maintains an explicit research state, supports branching and rollback within an investigation, gates consequential actions through verification and a Research World Model, connects Dry-Lab reasoning to external computational or physical evidence, and converts validated cross-task experience into Discovery Skills. This organization makes the proposed capability operational without requiring one monolithic model or maximal autonomy.

We additionally connect the framework to a real Dry-Lab/Wet-Lab discovery case. GALILEO couples multi-omics-informed target nomination and peptide design with robotic synthesis, multimodal phenotyping, orthogonal hands-on assays, and repeated evidence-driven revision. Across experimentally validated LRRC8C and SLC25A1 branches, physical measurements alter subsequent target beliefs, assay choices, mechanism hypotheses, and molecular-design policies; across five optimization rounds, the resulting feedback is further consolidated into a transferable Amphiphilic Balance Grammar. We use this case as empirical grounding for the intervention–evidence–revision loop, while keeping the broader general-purpose DFM claim distinct from any single domain-specific system.

The learning and evaluation formulations follow the same state-centered view. Training targets the intermediate decisions that change a research program, using trajectories, interactive environments, process supervision, scientific feedback, and resource allocation across formulation, representation, hypothesis construction, intervention, falsification, and verification. Evaluation measures both externally validated progress in the current episode and improvement in future discovery behavior under matched resources and retrieval controls [10, 11, 20, 26, 38, 39].

Contributions. This paper makes three contributions.

- We formulate Discovery Foundation Models as a capability-based model-system category and specify the research objects and operations that distinguish open-ended discovery from optimization over a predefined task.
- We define the Discovery Process and instantiate it with Zetema, which couples explicit research-state revision, evidence-based action gating, external grounding, and validated cross-task Discovery Skill evolution.
- We formulate training and evaluation mechanisms for learning these discovery operations, allocating resources across the process, and measuring externally validated knowledge progress and transferable improvement on unseen tasks.
- We empirically ground the Dry-Lab/Wet-Lab component with GALILEO, a real therapeutic-discovery loop in which physical biological feedback revises subsequent scientific decisions and is distilled across rounds into a reusable design rule.

Sections 2–4 introduce the problem setting and discovery operators. Sections 5–7 specify the system instantiation, capability formation, and evaluation protocol. Sections 8 and 9 analyze how grounding and responsibility change as the same framework moves from digital to physical and recursive settings.

2. From Generalist Problem Solving to Discovery Intelligence

The motivation for DFMs is not that current foundation models lack scientific knowledge or reasoning. Their limitation is more specific: most training and evaluation pipelines reward competence after the research structure has been fixed.

2.1. Generalist Capability within Predefined Research Structures

Foundation models have expanded from language modeling to broad knowledge, multi-step reasoning, coding, multimodal interaction, tool use, and agentic execution [7, 16, 33, 50, 54]. Scientific models extend the same substrate to proteins, molecules, materials, physical fields, biomedical records, and other domain-specific modalities [2, 21, 22, 27, 30, 57]. Scientific agents connect these abilities to search, code, databases, simulators, and laboratory interfaces [4, 6, 14, 15, 40].

This capability is a necessary substrate for discovery, but its usual task interface hides a structural ceiling. A model receives a recognizable object—a question, dataset, benchmark, formal language, design space, or goal—and optimizes within it. Search can explore enormous candidate spaces, reinforcement learning can discover unexpected strategies, and an agent can automate a long workflow. None of these mechanisms guarantees that the supplied variables or evaluator are scientifically adequate.

A fixed representation cannot express a variable it omits. A search objective cannot recover a property that its evaluator systematically ignores. Increasing sample count does not distinguish mechanisms when the observable is non-identifying. An automated workflow can therefore pursue a misframed question more efficiently without becoming better at recognizing the misframing.

The boundary is easiest to see when a research program stalls. If two mechanisms remain observationally equivalent, the bottleneck may be the measurement rather than hypothesis diversity. If performance gains disappear under another data split, the problem may be evaluator mismatch rather than optimization. If every explanation requires local exceptions in the same regime, another representation may be more useful than another candidate explanation. These are changes to the research structure, not additional solutions inside it.

2.2. Science as a Capability-Forming Environment

Scientific discovery exposes these structural decisions because evidence is coupled to a world that pushes back. The system must often act before it knows the correct question, choose measurements under partial observability, and revise after outcomes that do not match its predictions. Active intervention changes what can be learned: a perturbation, counterexample, boundary test, simulation, or replication can separate explanations that observational data leave equivalent [4, 9, 41].

This feedback is qualitatively different from adding more scientific text to pretraining. Knowledge helps a model recognize established concepts and plausible mechanisms; a capability-forming environment requires it to make consequential research decisions under incomplete specification. The environment can be a codebase, formal system,

causal simulator, digital twin, robotic platform, physical laboratory, or human-mediated process. What matters is that the resulting observation is not freely chosen by the model.

Science also makes evaluator incompleteness visible. Benchmark leakage, simulator artifacts, non-reproducible effects, and publication-like plausibility can all create apparent progress without stronger knowledge. Work on AI-assisted science has already highlighted the risk of fluent but weakly grounded understanding and the possibility that AI changes which problems are pursued, not only how quickly they are solved [19, 28]. For a DFM, the evaluator can itself become part of the research state when evidence suggests that it is misaligned.

Long horizons make the training signal harder but more informative. A negative result may eliminate months of future work. A failed replication can reduce confidence in the phenomenon rather than in a particular hypothesis. A representation change can make later interventions identifying. These outcomes cannot be valued reliably from the final answer alone; they require a record of how the research state changed.

2.3. Three Missing Transitions

The gap between predefined problem solving and Discovery Intelligence can be localized to three transitions.

Framing. The system must move from observations and uncertainty to a research opportunity worth pursuing. This includes distinguishing persistent structure from noise, deciding which unknowns are consequential and testable, and specifying the scope, scale, conditions, and observables needed to make the problem researchable. A supplied question can be rejected or reformulated when it is too broad, proxy-driven, or impossible to identify under the available measurements.

Modeling. The system must construct the variables and abstractions through which explanations become expressible. A useful operation may add a latent variable, remove a proxy, change scale, separate regimes, revise an ontology, or transform the problem into a causal, geometric, symbolic, or programmatic form [8, 34, 44]. Hypotheses are then formed inside this provisional representation and must differ in mechanism, validity conditions, or intervention response rather than only in wording.

Grounding and revision. The system must choose evidence that can change the status of the current explanations and then update the appropriate research object. A contradiction can indicate theory failure, measurement error, protocol deviation, hidden confounding, simulator misspecification, or environmental shift. Discovery therefore requires both informative intervention and failure attribution. The resulting experience becomes a transferable Discovery Skill only after its trigger and effect survive validation beyond the episode in which it was observed.

These transitions define the objects that Sections 3 and 4 make explicit.

3. Discovery Foundation Models: Defining Discovery Intelligence

We define DFMs by the research structures they can construct and revise, not by a particular neural architecture or degree of autonomy. Figure 2 summarizes the capability boundary.

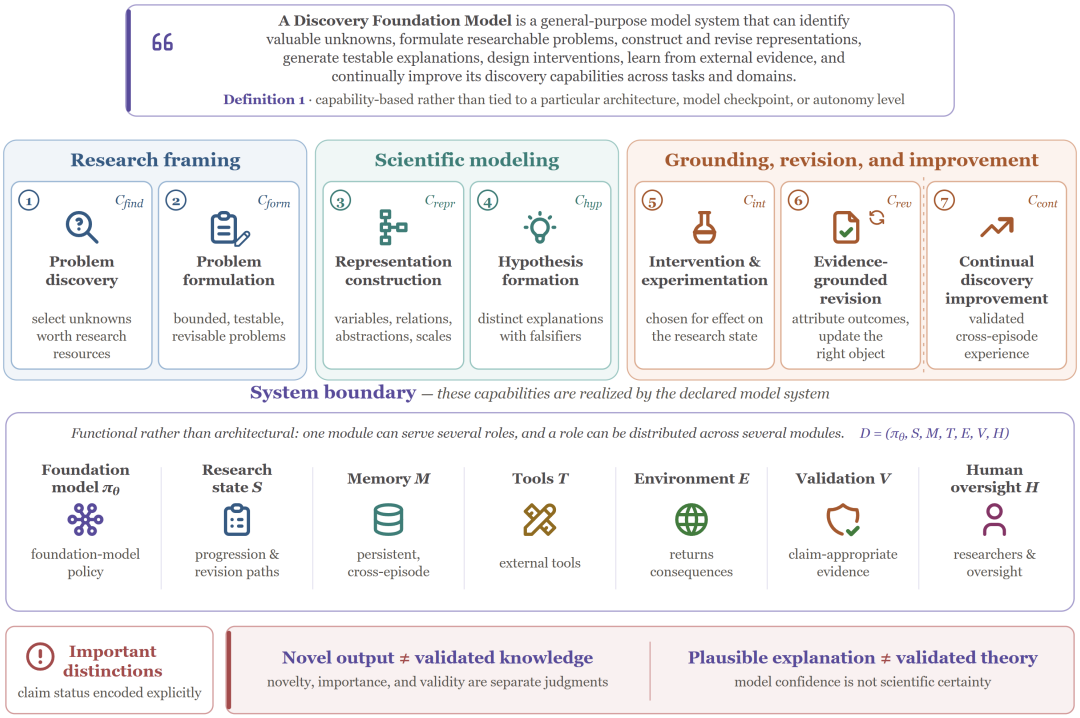

Figure 2 | Capability definition and system boundary of a Discovery Foundation Model. A DFM is defined by seven coupled capabilities spanning research framing, scientific modeling, evidence-grounded intervention and revision, and continual discovery improvement. These capabilities are realized by an integrated model system comprising a foundation-model policy, explicit research state, memory, tools, environments, validation mechanisms, and human oversight. The definition is therefore capability-based rather than tied to a particular architecture, model checkpoint, or autonomy level.

3.1. Problem Setting and Formal Definition

A conventional model task can be abstracted as

$$Q = (P, R, G, T, V), \tag{1}$$

where P is the problem, R its representation, G the objective, T the available tools, and V the evaluator. The system is asked to produce a solution under this supplied structure. This abstraction covers scientific question answering, formal reasoning, tool-using agents, and search-based design even when the underlying task is difficult or the resulting solution is genuinely novel.

Discovery starts from a less complete state. The system observes a partially understood world $\mathcal{W}$, has an initial knowledge state $\mathcal{K}_0$, and operates under computational, experimental, safety, and access constraints $\mathcal{B}$. A discovery episode produces both knowledge progress and an evidence-bearing record of how the investigation changed:

$$\Phi_{\text{disc}} : (\mathcal{W}, \mathcal{K}_0, \mathcal{B}) \longrightarrow (\Delta\mathcal{K}, \Xi). \tag{2}$$

$\Delta\mathcal{K}$ denotes externally validated progress, while Ξ contains the state transitions, alternatives, interventions, observations, failed formulations, and revisions that produced it. Unlike Equation 1, the problem, representation, hypothesis space, intervention strategy, and validation procedure can all change during Φ_{disc}.

> **Definition 1 (Discovery Foundation Model).** A *Discovery Foundation Model* is a general-purpose model system that can identify valuable unknowns, formulate researchable problems, construct and revise representations, generate testable explanations, design interventions, learn from external evidence, and continually improve its discovery capabilities across tasks and domains.

The definition imposes three requirements. First, the relevant operations must transfer beyond one fixed task even when their implementation remains domain-specific. Second, scientific claims are grounded by evidence appropriate to the domain; model confidence or internal agreement is not sufficient. Third, the evaluated unit is the declared model system, including any persistent state, memory, tools, environments, validators, and human participation that materially determine its behavior.

A DFM can therefore make useful progress without producing a final positive discovery. Showing that an effect does not replicate, that a question is untestable under current measurements, or that a representation omits the variable needed for intervention can all be valid outputs when the conclusion is supported by the research state.

3.2. Discovery Capabilities

We factor the DFM target into seven coupled capabilities:

$$C_{\text{DFM}} = \{C_{\text{find}}, C_{\text{form}}, C_{\text{repr}}, C_{\text{hyp}}, C_{\text{int}}, C_{\text{rev}}, C_{\text{cont}}\}. \tag{3}$$

C_{find} selects unresolved structures worth allocating research resources to and rejects apparent unknowns that disappear under calibration, retrieval, or stronger baselines. C_{form} turns a selected unknown into a bounded and testable problem by fixing its object, scope, scale, conditions, and observables while keeping those choices revisable.

C_{repr} constructs the variables, relations, abstractions, and scales through which the problem is expressed. It becomes decisive when the current representation makes every candidate explanation equivalent or repeatedly produces the same failure boundary [8, 34, 44]. C_{hyp} forms mechanistically distinct explanations with explicit assumptions, validity ranges, predictions, and possible falsifiers [3, 15, 47].

C_{int} chooses experiments, simulations, code executions, ablations, counterexamples, alternative measurements, or replications for their expected effect on the research state rather than for confirmation alone. C_{rev} attributes unexpected outcomes and updates the appropriate object: hypothesis, representation, problem formulation, protocol, measurement process, or intervention plan.

C_{cont} changes future discovery behavior using validated cross-episode experience. This is stronger than fact accumulation or retrieving a successful trajectory. A reusable operation must specify when it applies, what it should change, and what later evidence would show that the change was beneficial. Memory-based agents provide precedents for experience-driven behavioral change; the DFM requirement adds attribution, scientific grounding, and transfer [35, 45].

The first six capabilities operate within an investigation. The seventh is a cross-task update mechanism. Section 4 specifies the within-task operators, and Section 5 instantiates both levels in one system organization.

3.3. System Boundary

A DFM is evaluated as a model system

$$\mathcal{D} = (\pi_\theta, \mathcal{S}, \mathcal{M}, \mathcal{T}, \mathcal{E}, \mathcal{V}, \mathcal{H}), \tag{4}$$

where π_θ is the foundation-model policy, $\mathcal{S}$ the research state, $\mathcal{M}$ persistent memory, $\mathcal{T}$ external tools, $\mathcal{E}$ the environment that returns consequences, $\mathcal{V}$ validation mechanisms, and $\mathcal{H}$ human researchers or oversight. The tuple is functional rather than architectural: one module can serve several roles, and a role can be distributed across several modules.

The system boundary matters for attribution. If a human supplies the decisive reformulation, the trajectory should record that intervention rather than attributing the discovery to the model. If one baseline receives a hand-built representation unavailable to another, the comparison is not a model-only comparison. Autonomy is similarly orthogonal to capability: a system can autonomously execute low-risk code while requiring approval for a physical experiment and still instantiate the same discovery operators.

Validation is not reduced to a universal scalar reward. Logical checks, held-out execution, simulation, replication, independent reviewers, and physical measurement support different claims. The system should preserve which validator supported which state transition and abstain or escalate when the available evidence does not justify promotion of a claim.

3.4. Relation to Existing Scientific AI Paradigms

DFMs build on, rather than replace, existing scientific AI. Foundation models for science provide domain representations and knowledge [2, 21, 27, 57]. Scientific reasoning models improve formal inference and verification [16, 24, 37, 50]. Scientific agents coordinate tools and long workflows [3, 14, 15, 25, 36]. Search systems explore candidate spaces, and autonomous laboratories connect proposals to physical measurements [1, 4, 40, 41].

The defining difference is the joint capability requirement. Let $\mathcal{Z}$ denote a predefined search space and $z \in \mathcal{Z}$ a candidate solution or design. Search can be central to a DFM, but optimizing candidates within a fixed $\mathcal{Z}$ under an evaluator V does not by itself establish the ability to revise the research structure. A DFM must also be able to recognize, from evidence, when the search space or evaluator is inadequate and revise

Table 1 | Capability-level comparison of Discovery Foundation Models and existing scientific AI paradigms.

Paradigm	Problem Source	Representation Change	Intervention Design	Problem Revision	External Grounding	Skill Transfer
Foundation Models for Science	Given	Limited	External	Not defining	Data	Not defining
Scientific Reasoning Models	Given	Limited	Limited	Not defining	Tasks or verifiers	Not defining
Scientific Agents and AI Scientist Systems	Often given	Optional	Often	Optional	Tools and environments	Usually task-specific
Search and Evaluator Systems	Given	Fixed	Domain dependent	Not defining	Evaluator	Not defining
Autonomous Laboratories	Often given	Task defined	Required	Optional	Physical	Not defining
Discovery Foundation Models	**Constructed**	**Required**	**Required***	**Required**	**Required**	**Required**

Note. Entries describe the typical capability targets of each paradigm rather than universal properties of every individual system. "Not defining" means that the capability is not required by the paradigm. "Optional" means that some systems support the capability without treating it as a defining requirement. *Intervention design is required when the research setting permits experimental, computational, or other active forms of evidence acquisition.

the relevant structure accordingly. Likewise, physical execution provides strong external grounding but does not by itself constitute discovery when the objective and design space remain human-specified. An AI-scientist system satisfies the DFM criterion to the extent that formulation, representation, intervention, and revision become explicit model-system operations, their consequences are externally grounded, and the resulting experience is validated to improve future discovery behavior across tasks.

The Table 1 describes typical capability targets rather than mutually exclusive categories. A particular scientific agent may already reconstruct representations, and a search system may alter its evaluator. Such systems satisfy more of the DFM definition to the extent that these operations are integrated, externally grounded, and evaluated as transferable behavior. Conversely, the DFM label does not supply capabilities that the system has not demonstrated.

4. Discovery Process: Operationalizing Discovery Intelligence

The DFM definition specifies *what* the system must be able to revise. The Discovery Process specifies *how* those revisions compose inside an investigation. Figure 3 shows the main research objects and feedback paths.

The process is not a fixed stage pipeline. Several formulations may coexist, different representations can support different hypothesis families, and evidence can return the investigation to an earlier object. The unit of computation is therefore a transition in a revisable research state rather than a completed textual stage.

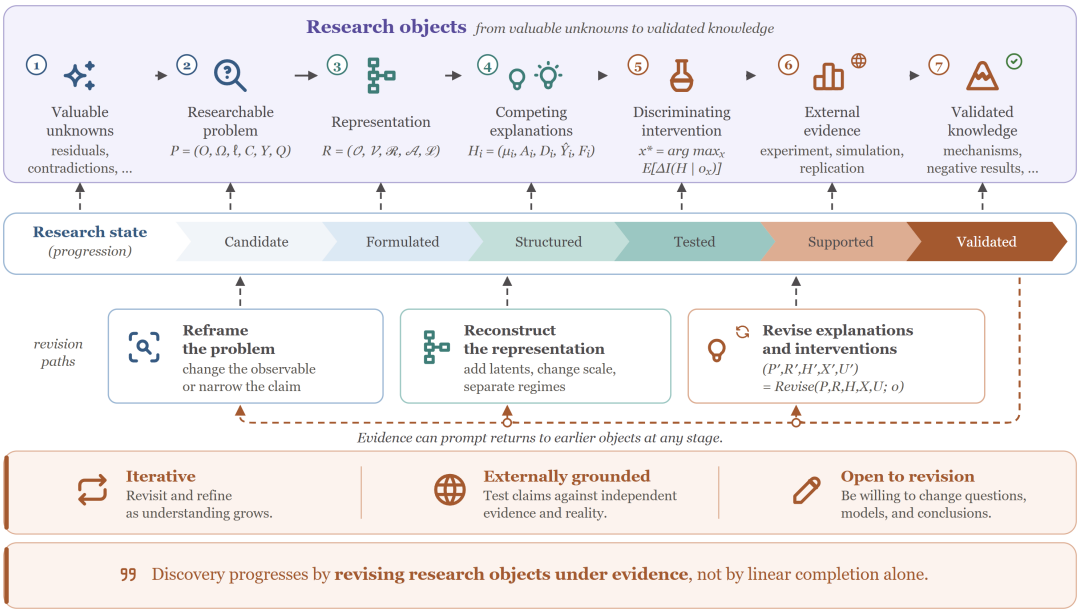

Figure 3 | The Discovery Process as revision over an evolving research state. Discovery progresses from valuable unknowns to researchable problems, representations, competing explanations, discriminating interventions, external evidence, and validated knowledge. These objects are not traversed as a fixed pipeline: evidence can return the investigation to an earlier problem, representation, explanation, or intervention. The research state therefore records both epistemic progression and the revision paths through which that progression is achieved.

4.1. From Unknowns to Researchable Problems

Problem Discovery begins from observations, residuals, contradictions, failed replications, regime boundaries, evaluator mismatch, or newly available measurements. The operator filters as well as proposes. A signal that disappears after calibration, stronger retrieval, or a more appropriate baseline should not be promoted into a research program merely because it was initially surprising.

A selected unknown is then formulated as

$$P = (O, \Omega, \ell, C, Y, Q), \tag{5}$$

where O is the object of study, Ω its scope, ℓ the relevant scale, C the conditions under which the claim is posed, Y the observables, and Q the unresolved relation or mechanism. These fields are operational: they determine which evidence can count, which interventions are feasible, and where the resulting knowledge is expected to apply.

Formulation becomes an active discovery step when alternatives imply different experiments. The same observation may be treated as a prediction failure, a causal-identification problem, or a measurement problem. If no available intervention can resolve the central uncertainty under one framing, the correct update may be to change the observable or narrow the claim rather than to continue searching for answers inside that framing.

4.2. Representations and Competing Explanations

Given a provisional problem, the system constructs a scientific representation

$$R = (\mathcal{O}, \mathcal{V}, \mathcal{R}, \mathcal{A}, \mathcal{L}), \tag{6}$$

with objects $\mathcal{O}$, variables $\mathcal{V}$, relations $\mathcal{R}$, abstractions or coarse-grainings $\mathcal{A}$, and structural constraints $\mathcal{L}$. Representation operations include adding a latent variable, removing a misleading proxy, changing temporal or spatial scale, separating regimes, revising an ontology, or translating the problem into another formal structure [8, 34, 44].

A representation earns its role through downstream consequences. It should improve prediction under unseen conditions, expose a discriminating intervention, separate previously conflated regimes, compress a mechanism, or transfer to another setting. A novel label that leaves every possible action unchanged is not a useful representation change.

Within a provisional representation, a candidate explanation is

$$H_i = (\mu_i, A_i, D_i, \widehat{Y}_i, F_i), \tag{7}$$

where μ_i is the mechanism, A_i its assumptions, D_i its validity range, $\widehat{Y}_i$ its predictions, and F_i the observations that would weaken or falsify it. Candidate explanations are useful when they disagree under at least one relevant condition. If every current hypothesis predicts the same observation under every feasible action, additional hypothesis sampling is unlikely to be the bottleneck; the system should inspect the representation or measurement interface.

Explanations can also be nested. A high-level regularity can remain valid after its proposed lower-level mechanism fails, and different mechanisms can dominate in different regimes. The research state therefore records which claim is under test rather than forcing all explanations into one winner-take-all competition.

4.3. Intervention and Evidence-Grounded Revision

An intervention is selected for the state change it is expected to produce. Let H denote the current explanation set and o_x the possible observation under intervention x. A conceptual objective is

$$x^* = \arg\max_x \mathbb{E}[\Delta\mathcal{I}(H \mid o_x)], \tag{8}$$

where $\Delta\mathcal{I}$ measures the expected reduction or restructuring of uncertainty over the current explanations. In practice, the decision also depends on feasibility, cost, risk, statistical power, measurement quality, and the probability of an inconclusive outcome [9, 12].

The selected action can be a physical experiment, simulation, code execution, ablation, counterexample, alternative measurement, or replication. Confirmation is not the only

useful outcome. An intervention can reveal that all current hypotheses share a false assumption, that the manipulation failed to change its intended variable, or that the measurement process is unreliable. The research state should encode these possibilities before execution so that the result triggers a meaningful update rather than post hoc reinterpretation.

After an external observation o, revision updates whichever objects are implicated:

$$(P', R', H', X', U') = \text{Revise}(P, R, H, X, U; o), \tag{9}$$

where X denotes the intervention plan and U local uncertainty. Raw observations remain separate from their interpretations so that old evidence can be reanalyzed after a representation change. Provenance, calibration, protocol fidelity, leakage, and replication determine whether o is eligible to support a scientific update [29, 51].

Failure attribution is part of the operator. Theory failure, measurement error, implementation error, protocol deviation, hidden confounding, random noise, environmental shift, and simulator misspecification imply different next actions. This prevents both premature abandonment and ad hoc protection of a favored explanation.

4.4. Episode Output and Candidate Discovery Lessons

A completed episode yields two different artifacts. The first is domain knowledge: supported observations, mechanisms, predictions, validity boundaries, justified negative results, or a defensible conclusion that the current question is not testable. The second is a set of candidate lessons about the research process itself.

A candidate lesson can state that a formulation was too broad, a representation omitted a variable, an intervention was non-identifying, or a replication step prevented a false update. The trajectory alone does not validate the lesson. Success may depend on privileged information or an unrecorded human correction; failure may come from execution rather than from the decision that preceded it. The Discovery Process therefore preserves states, alternatives, actions, observations, and provenance without immediately converting retrospective explanations into reusable rules.

Zetema, introduced next, provides the system mechanism for maintaining this state over long horizons and deciding which candidate lessons are allowed to influence future discovery.

5. Zetema: A System Instantiation of Discovery Intelligence

We instantiate the DFM framework with *Zetema*, a system organization that couples within-task research-state revision, evidence-based action gating, external experimentation, and cross-task Discovery Skill evolution. Zetema specifies functional interfaces rather than a mandatory neural architecture: foundation models, tools, simulators, verifiers, laboratories, and human researchers can implement different parts of the same organization. Figure 4 gives the resulting data and control flow.

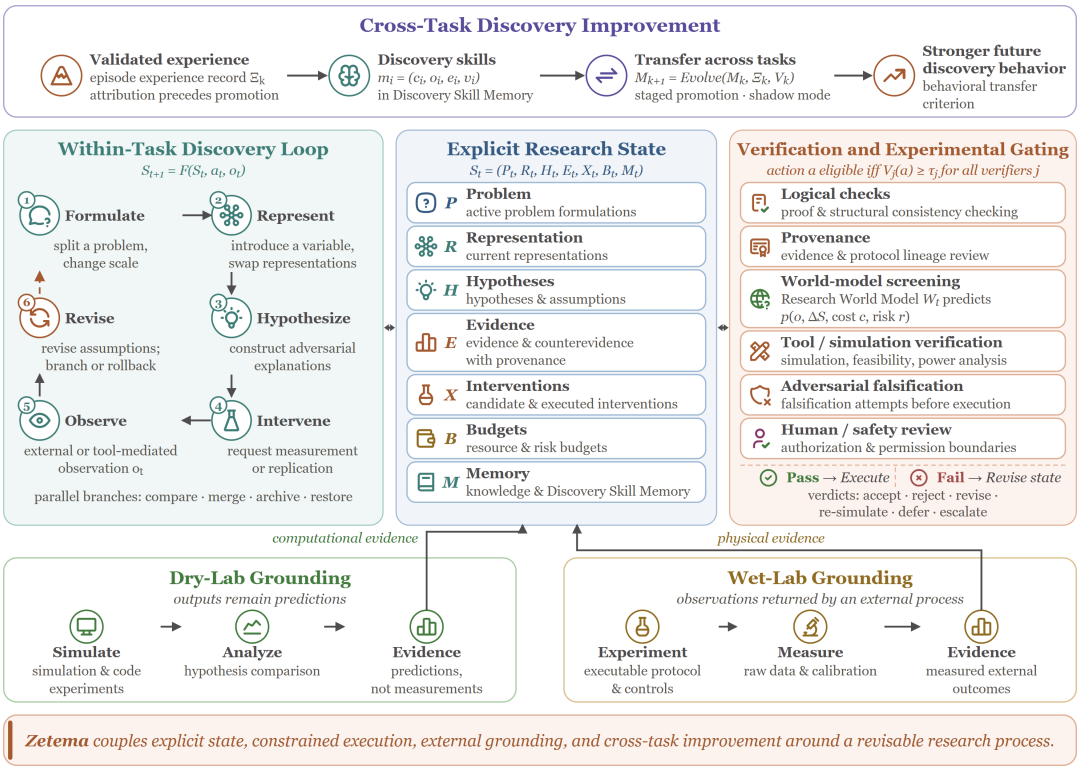

Figure 4 | Zetema: a system instantiation of Discovery Intelligence. Zetema organizes discovery around an explicit and revisable research state that couples a within-task discovery loop with verification and experimental gating. Dry-Lab and Wet-Lab interfaces return computational and physical evidence to the shared state, while validated episode-level experience is attributed, consolidated into Discovery Skills, and transferred across tasks to improve future discovery behavior.

Zetema is built around one constraint: a scientifically consequential operation must leave an inspectable state transition. Problems, representations, hypotheses, evidence, interventions, uncertainties, budgets, and reusable experience therefore remain distinguishable even when their internal implementation is neural or unstructured. This state makes revision, branching, attribution, and later training possible.

5.1. Within-Task Research-State Dynamics

At step t, Zetema maintains a research state S_t, selects a research operation a_t, receives an external or tool-mediated observation o_t, and applies

$$S_{t+1} = \mathcal{F}(S_t, a_t, o_t). \tag{10}$$

The action space includes ordinary tool use—searching literature, executing code, calling a simulator, requesting a measurement—and operations on the research program itself. The system can split a problem, introduce a variable, change scale, replace a representation, construct an adversarial explanation, revise an assumption, request replication, or terminate a branch.

This distinction changes how long-horizon reasoning is organized. When several interpretations of a failure remain plausible, Zetema does not need to compress them into

one uncertain narrative. It can maintain parallel branches, associate each branch with a diagnostic action, and compare the resulting evidence. A residual, for example, can be tracked simultaneously as a possible missing variable, dataset shift, or implementation error until an intervention separates those accounts.

Branches are first-class state objects. They can be compared, merged when their assumptions become compatible, archived when their expected value falls, or restored after new evidence changes their status. Termination is also explicit: a branch can stop because no feasible intervention is identifying, the original effect fails to replicate, the risk exceeds the expected value, or another research opportunity becomes more informative. A valid episode therefore need not end in a positive discovery claim.

5.2. Explicit Research State and Attribution

We instantiate the state as

$$S_t = (P_t, R_t, H_t, E_t, X_t, B_t, \mathcal{M}_t), \tag{11}$$

where P_t contains active problem formulations, R_t the current representations, H_t hypotheses and assumptions, E_t evidence and counterevidence with provenance, X_t candidate and executed interventions, B_t resource and risk budgets, and $\mathcal{M}_t$ the knowledge and Discovery Skill Memory available to the episode.

These fields form a linked record rather than separate notes. A hypothesis points to the problem and representation under which it is meaningful. An observation points to the intervention and protocol that produced it. A revision points to the evidence that triggered the change. Human edits, verifier rejections, execution deviations, and permission boundaries are retained because they affect both scientific attribution and the later training signal.

We also type operations by scientific role. A representation revision is not stored as another hypothesis; a predicted outcome is not stored as an observed measurement; a rejected branch is not deleted. Local uncertainty is attached to the object it concerns. The system can therefore trust a measurement while remaining uncertain about its mechanism, or accept an explanation inside one regime without extending it beyond its validity range.

The implementation can use structured text, graphs, databases, programs, equations, or hybrid representations. Zetema requires only that the research objects and their revision relations be recoverable well enough to support comparison, validation, rollback, and attribution.

5.3. Discovery Skill Memory

Cross-task improvement requires a representation of reusable research operations rather than an archive of successful trajectories. Thought templates provide a reasoning-level precedent: Buffer of Thoughts distills and reuses methods across tasks [52], while ReasonFlux learns to select and organize templates into hierarchical reasoning trajectories [53]. Discovery Skills extend this procedural view to research operations with

explicit state-dependent triggers, expected effects, and external validation requirements. Zetema stores a candidate Discovery Skill as

$$m_i = (c_i, o_i, e_i, v_i), \tag{12}$$

where c_i is a trigger over research-state conditions, o_i the operation to consider, e_i its expected effect, and v_i the evidence required to validate that effect.

Triggers are structural. A skill can activate when several hypotheses make the same prediction under all current measurements, when errors cluster at a regime boundary, when repeated interventions remain non-identifying, or when a theory accumulates local exceptions. The corresponding operation may request a new measurement, search for a hidden variable, change scale, construct a boundary test, replicate an effect, or stop an unproductive direction.

The expected effect prevents memory from becoming a collection of generic advice. A skill should state what is expected to change—for example, separating hypotheses, reducing uncertainty over a hidden variable, lowering experimental cost, or exposing measurement failure. Its validation record stores supporting episodes, counterexamples, uncertainty, and known domain restrictions. A retrieved skill can therefore be invoked provisionally and weakened when its expected effect does not appear.

This formulation separates three different kinds of accumulation: facts about the world, task-specific episodic memory, and operations intended to improve future discovery behavior. Only the last category constitutes Discovery Skill improvement.

5.4. Research World Model and Experimental Gating

Open-ended generation produces many fluent but redundant, infeasible, unsafe, or empirically indistinguishable candidates. Zetema therefore inserts a verification and gating layer before actions acquire substantial cost or consequence. A Research World Model W_t predicts possible observations, state changes, costs, and risks:

$$W_t : (S_t, a) \longrightarrow p(o, \Delta S, c, r \mid S_t, a). \tag{13}$$

W_t can combine learned predictors, causal or mechanistic models, simulators, digital twins, formal tools, and ensembles. Its purpose is not to certify truth. It screens counterfactual consequences: which hypotheses would separate, which state objects might change, how much the action costs, and where the model itself is uncertain.

Because screening changes which research directions receive resources, world-model error has a selection effect. A conservative model can suppress unusual but valid interventions; an overconfident model can repeatedly prefer actions implied by its own misspecification. Zetema therefore retains model disagreement, calibration, validity ranges, and out-of-distribution signals. High uncertainty can trigger a bounded pilot rather than automatic rejection, and repeated disagreement between predictions and external outcomes updates the screening model itself.

Verification is adaptive to the action. A formal claim may proceed directly to proof checking. A physical intervention can require structural checks, provenance review, power analysis, simulation, adversarial falsification, feasibility checks, and human authorization. We represent eligibility as

$$a \in \mathcal{A}_{\text{eligible}} \iff V_j(a) \geq \tau_j \quad \forall j \in \mathcal{J}, \tag{14}$$

where each V_j can be a structured verifier, an uncertainty-aware model, a formal constraint, or a human decision. Thresholds depend on the action: speculative hypotheses can enter the state under uncertainty, whereas costly, irreversible, or high-risk interventions require stronger evidence and permission.

The gate can accept, reject, request revision, request another simulation, defer, or escalate. Among eligible actions, the system can retain a portfolio rather than collapsing to one score: low-risk actions for efficient progress, high-information actions for resolving a central uncertainty, and bounded high-uncertainty actions that test a potentially transformative alternative.

5.5. Validated Cross-Task Update

After episode k, Zetema receives an experience record Ξ_k containing state transitions, successful and failed branches, external outcomes, human interventions, and validation results. Long-term memory evolves as

$$\mathcal{M}_{k+1} = \text{Evolve}(\mathcal{M}_k, \Xi_k, \mathcal{V}_k), \tag{15}$$

where $\mathcal{V}_k$ denotes the validation applied to candidate updates. The update can create, refine, specialize, compose, weaken, or delete a skill.

Attribution precedes promotion. A failed experiment can originate from problem selection, representation, intervention design, execution, or validation. A successful trajectory may depend on an expert-supplied variable or privileged data. Zetema uses the linked research state to identify the state–operation relation actually supported by the episode instead of turning the final retrospective narrative into a general rule.

Recuris provides a concrete precedent in long-horizon agent harnesses: working memory guides experiential skill selection, while execution evidence supports localized, validation-gated memory updates [56]. For discovery, this pattern additionally requires attributing updates to externally grounded research outcomes and testing whether they improve subsequent research decisions.

Promotion can be staged. A candidate skill is first tested on archived trajectories or counterfactual replay, then used in shadow mode on new tasks, and only later allowed to affect active research decisions. Held-out tasks, alternative environments, ablations, expert review, and independent replication test the claimed effect. Skills remain versioned and reversible; negative evidence narrows their trigger or removes them.

The criterion is behavioral transfer. Cross-task improvement is established only when future discovery decisions improve after controlling for additional domain facts, near-duplicate retrieval, prompt reuse, and extra compute.

5.6. Dry-Lab and Wet-Lab Grounding

Zetema separates generated expectations from observed outcomes by coupling a Dry-Lab Discovery Loop to a Wet-Lab Grounding Loop. The same interface also covers non-physical environments such as code execution and formal systems; the distinction is whether the observation is generated internally or returned by an external process.

The Dry-Lab loop performs literature synthesis, data analysis, code experiments, simulation, hypothesis comparison, protocol drafting, power analysis, outcome prediction, and failure-mode analysis. Its outputs remain predictions. Expected effect sizes and simulated observations do not enter E_t with the same epistemic status as measurements.

When a physical experiment is required, a selected scientific intention is translated into an executable protocol with variables, controls, samples, measurements, expected outcomes, stopping conditions, and checks for contamination or manipulation failure. Execution records deviations instead of assuming perfect compliance. Measurements retain raw data, calibration state, batch and instrument effects, missing observations, replicate consistency, and sample provenance.

Protocol translation is itself diagnostic. An intervention that appears identifying in abstract form can become impossible under available instrument resolution, sample size, manipulation range, or safety constraints. In that case the Wet-Lab interface returns a formulation or representation failure to the research state rather than simply a binary feasibility rejection.

Zetema therefore closes the method loop without assigning discovery to one component. Foundation models propose and revise research objects; tools and environments return consequences; validators constrain promotion and action; human researchers contribute domain judgment, authorization, criticism, execution, and replication. The capability claim belongs to the declared integrated system and its recorded state transitions. Importantly, this form of Dry-Lab/Wet-Lab coupling is not only hypothetical: the following case study shows a real experimental loop in which physical biological outcomes revise subsequent discovery decisions.

5.7. Empirical Case Study: A Real Dry-Lab/Wet-Lab Discovery Loop

The Dry-Lab/Wet-Lab interface above is not only a conceptual organization. A concrete instance of several of these operations already appears in a real therapeutic-discovery setting. Figure 5 summarizes GALILEO, an embodied AI-scientist system for therapeutic peptide discovery in dynamic membrane systems (Jiang et al., 2026). Rather than treating the wet laboratory as a terminal validation stage, GALILEO places experimentally returned biological outcomes inside the iterative decision loop: candidate interventions are proposed, physically executed, measured, and used to revise subsequent target beliefs, molecular-design policies, assay choices, and mechanism hypotheses.

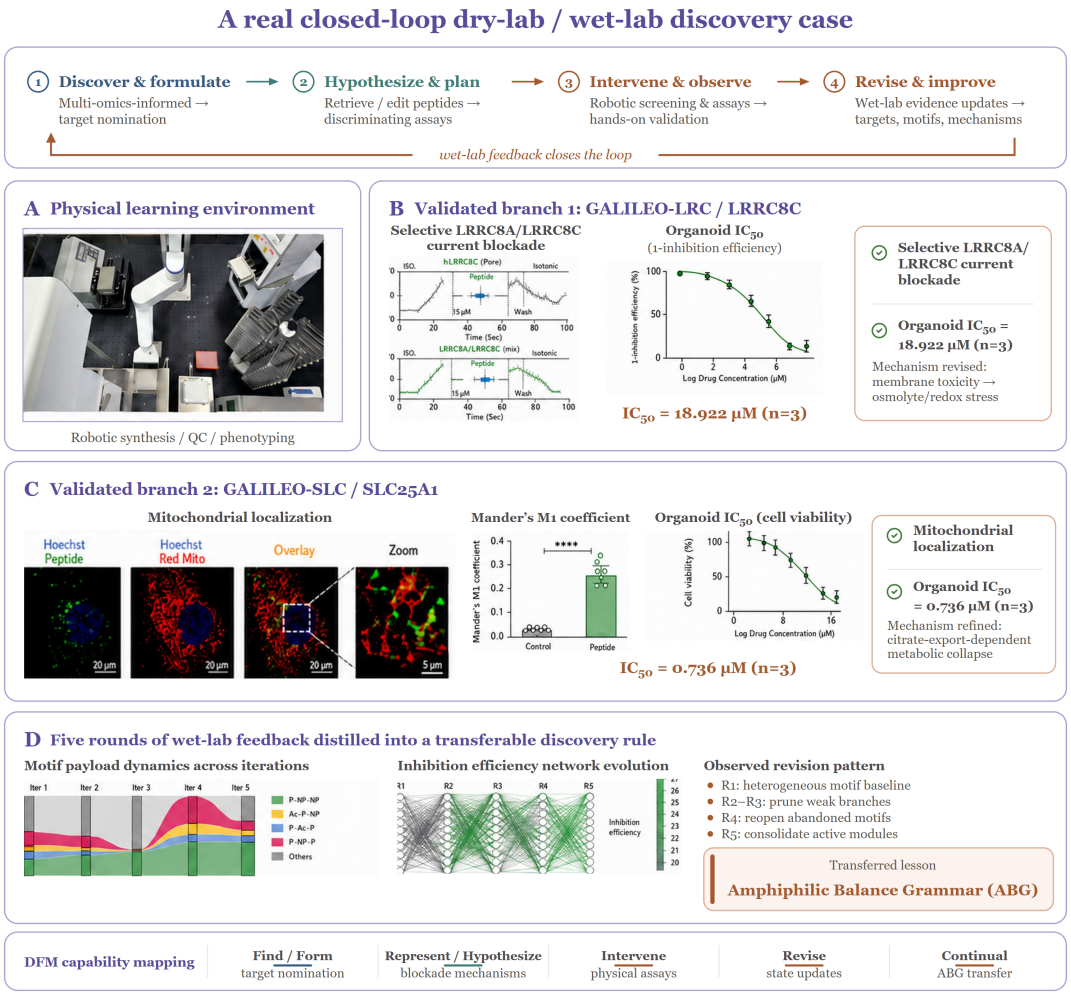

Figure 5 | **A real closed-loop Dry-Lab/Wet-Lab discovery case.** GALILEO provides an empirical example of how several Discovery Foundation Model operations can be instantiated in a physically grounded scientific workflow. The top panel summarizes the closed loop from discovery and formulation, through hypothesis and intervention design, to physical observation and evidence-grounded revision. **(A)** The physical learning environment couples robotic synthesis, quality control, liquid handling, imaging, and plate-based phenotyping to the discovery process. **(B)** One experimentally validated branch produces GALILEO-LRC, supported by selective LRRC8A/LRRC8C current blockade and organoid activity with an IC_{50} of 18.922 μM. **(C)** A second branch produces GALILEO-SLC, supported by mitochondrial localization and organoid activity with an IC_{50} of 0.736 μM. **(D)** Across five rounds of wet-laboratory feedback, weak motif organizations are pruned, previously abandoned organizations are reopened, and productive modules are consolidated, yielding the Amphiphilic Balance Grammar (ABG) as a reusable design rule. The figure therefore represents a real experimentally grounded closed loop rather than a hypothetical workflow: physical evidence changes subsequent scientific actions and contributes to transferable discovery behavior.

A physical environment as part of the learning loop. Panel A makes explicit a distinction that is easy to obscure in purely computational scientific agents. The environment is not only a source of additional context or a tool endpoint; it returns consequences that the model system does not control. GALILEO couples a cognitive discovery layer to robotic synthesis, quality control, liquid handling, imaging, and plate-based phenotyping. Candidate peptides are retrieved from a clinically informed peptide prior and locally edited, after which physical execution produces observations that can support, weaken, or redirect the active research branch. In the terminology of Section 5.6, the resulting measurements enter the evidence state as externally returned observations rather than as model-generated expectations.

This distinction matters because physical feedback can alter the *research program*, not only its final score. In the reported GALILEO loop, plate-based phenotypes and orthogonal validation results were used to update target-branch belief; viability, morphology, solubility, and quality-control outcomes changed motif-policy weights; ambiguous phenotypes triggered new assays; and the evolving hypothesis board changed which mechanisms and interventions were pursued next. The laboratory therefore functions as a capability-forming environment in the stronger sense used throughout this paper: action produces evidence, and evidence changes subsequent scientific decisions.

Externally grounded discovery branches. Panels B and C illustrate two distinct branches in which computational proposals were subjected to orthogonal physical tests. For the LRRC8C branch, GALILEO-LRC was evaluated through whole-cell electrophysiology and showed selective inhibition of LRRC8A/LRRC8C currents relative to other tested LRRC8 heteromers. Patient-derived organoid experiments provided a second level of evidence, with an experimentally measured IC_{50} of 18.922 μM in the LRRC8C-high organoid setting. Subsequent metabolomic, transcriptomic, perturbational, and immune-context experiments further shifted the working explanation from generic membrane toxicity toward an LRRC8C-linked osmolyte/redox-stress and innate-signaling mechanism.

The SLC25A1 branch followed a different evidential route. GALILEO-SLC was shown to localize to the mitochondrial compartment in which SLC25A1 operates, and dose-response experiments in SLC25A1-high patient-derived organoids yielded an IC_{50} of 0.736 μM. Target silencing, acetate rescue, mitochondrial measurements, metabolomics, and extracellular-acidification assays then progressively refined the working explanation toward a citrate-export-dependent metabolic-collapse mechanism. These branches are important for the DFM framework not because they merely produce successful candidates, but because heterogeneous external observations support different updates to the research state: target confidence, mechanism, intervention choice, and the next measurement can all change as evidence accumulates.

From repeated feedback to a transferable discovery rule. Panel D provides the strongest connection to evidence-grounded revision and continual discovery improvement. Across five autonomous peptide-optimization rounds, the trajectory was not a monotonic search toward one increasingly dominant candidate family. The first round maintained a heterogeneous motif baseline; rounds 2–3 removed weakly supported branches and contracted the active motif payload; round 4 reopened previously abandoned acidic,

polar, and nonpolar organizations under revised pore-engagement hypotheses; and round 5 consolidated recovered motif organizations into shared active modules.

This pattern is qualitatively different from static candidate ranking. A rejected branch is not necessarily forgotten permanently, and a previously weak design operation can become useful after the research state changes. The relevant object of learning is therefore not only "which peptide worked," but *under which research-state conditions a particular molecular organization should be generated, rejected, recovered, or tested again*. The resulting Amphiphilic Balance Grammar (ABG) summarizes this accumulated physical feedback into a reusable intervention prior. In the original study, motifs derived from the physically optimized branches were further transferred into external peptide-design workflows, improving the geometry and energetic profiles of their outputs. This provides an empirical example of the transition from episode-level experience to a more portable discovery operation.

Mapping the case to DFM capabilities. The correspondence to the DFM capability decomposition is not one-to-one at the level of software modules, but it is operational at the level of research decisions. Target nomination and translational-gap analysis instantiate aspects of C_{find} and C_{form}; construction of target-specific blockade mechanisms and sequence organizations engages C_{repr} and C_{hyp}; robotic and hands-on assays instantiate C_{int}; updates to target belief, motif policy, assay selection, and mechanism hypotheses instantiate C_{rev}; and consolidation of experimentally supported motif structure into ABG provides a concrete instance of experience influencing future discovery behavior, corresponding to the direction targeted by C_{cont}.

The case also illustrates why a DFM is defined as a *model system* rather than as a single autonomous model. Not every experimentally consequential operation in GALILEO is executed by the robotic platform. Synthesis, quality control, liquid handling, imaging, and plate-based phenotyping are integrated into the automated loop, whereas specialized assays including electrophysiology, metabolic profiling, patient-derived tissue experiments, and in-vivo validation can remain agent-specified but human-executed. These interventions should remain explicitly represented in the system boundary and provenance record rather than being attributed to the model alone. The relevant capability claim belongs to the coupled system of policies, tools, experimental environments, validators, and human execution described by Equation 4.

What this case establishes—and what it does not. GALILEO should not be interpreted as evidence that the full general-purpose DFM problem has already been solved. Its problem domain, molecular modality, experimental interfaces, and scientific objectives remain substantially structured, and broader cross-domain discovery transfer is still an open problem. Its importance here is narrower but concrete: it demonstrates a *real physical discovery loop* in which externally generated biological evidence is not merely used to validate a final model proposal. Instead, that evidence changes subsequent scientific actions, can reopen or terminate research branches, modifies the policy used to construct later interventions, and can ultimately be distilled into a reusable discovery rule.

This empirical distinction is central to Discovery Intelligence. A conceptual agent loop can always be drawn as

$$\text{hypothesis} \rightarrow \text{experiment} \rightarrow \text{result},$$

but a grounded discovery loop requires the stronger transition

$$S_t \xrightarrow{a_t} o_t \xrightarrow{\text{external evidence}} S_{t+1},$$

where S_{t+1} can change not only the preferred answer, but the problem interpretation, representation, mechanism, intervention strategy, evaluator, or reusable discovery operation. The GALILEO case shows that such evidence-driven state revision can already be realized in a real Dry-Lab/Wet-Lab scientific workflow. Capability formation, discussed next, asks how these state-conditioned research operations can be learned and generalized systematically.

6. Capability Formation: Training Discovery Operations

Zetema specifies how discovery is organized; capability formation specifies how the underlying research policy is learned. We formulate training around state-conditioned decisions rather than final scientific answers. A training example is useful when it exposes which research operation was available, why it was selected, what external consequence followed, and how that consequence changed the state.

Figure 6 summarizes the training loop.

6.1. Training Data and Interactive Research Environments

A discovery trajectory records a sequence of research states, available alternatives, selected operations, external observations, and revisions. The transitions are more important than a polished account of the final result. Useful data include why an observation was treated as an unknown, which formulations were rejected, what representation change made an intervention possible, which assumptions separated the explanations, and what later evidence showed that a decision was productive.

Success-only trajectories are insufficient. Training data should retain malformed questions, missing variables, misleading representations, non-identifying interventions, failed replications, contradictory evidence, abandoned branches, and attribution errors. These failures teach different policies only when their source is visible. An inconclusive experiment caused by low statistical power should not supervise the same update as an experiment that was well executed but non-identifying.

Historical artifacts provide partial signals. Laboratory notebooks, code histories, preregistrations, peer review, rebuttals, failed replications, and revised manuscripts expose changes in the research state but often omit alternatives or rationalize decisions retrospectively. Synthetic and simulated trajectories can provide latent mechanisms, counterfactual actions, and known failure sources, but inherit the assumptions of their generator. We therefore treat data-source provenance as part of the trajectory rather than flattening all examples into one demonstration format.

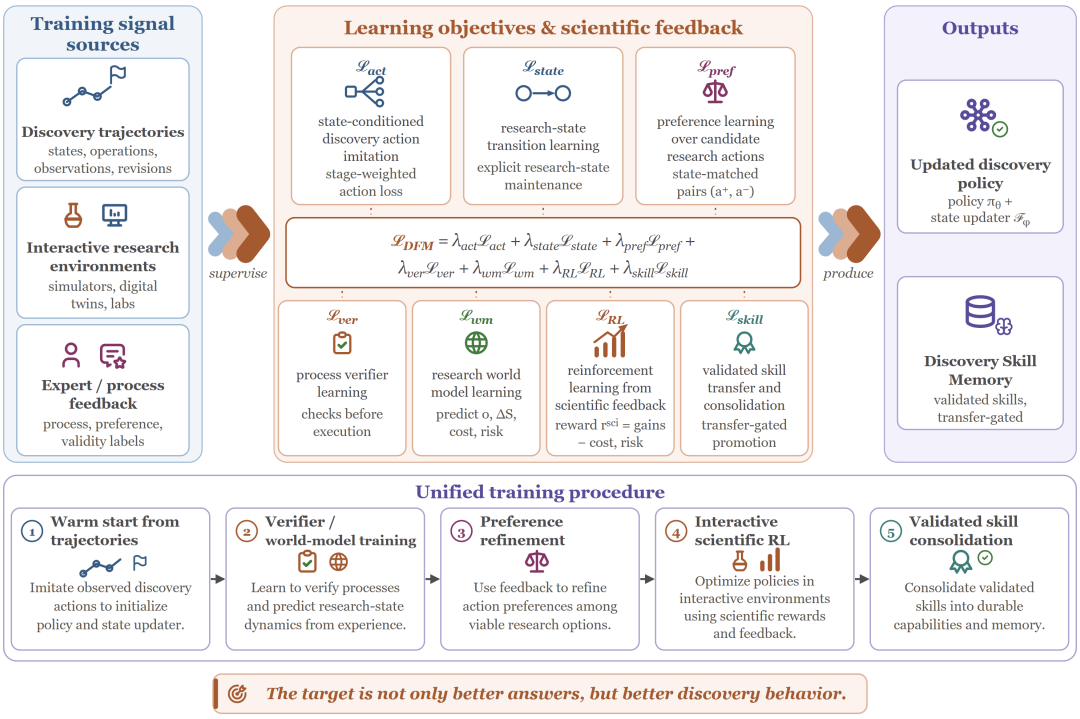

Figure 6 | **Capability formation as multi-signal optimization of discovery behavior.** Training combines state-conditioned discovery-action imitation, research-state transition learning, preference optimization, process verification, Research World Model learning, reinforcement learning from scientific feedback, and validated Discovery Skill transfer. These objectives form the compositional training objective $\mathcal{L}_{\text{DFM}}$ and can be optimized through a staged procedure from trajectory warm-start and verifier/world-model learning to preference refinement, interactive scientific reinforcement learning, and validated skill consolidation.

Interactive environments complement static trajectories by making the model choose what to observe or test next. Useful environments include causal worlds, formal systems, codebases, machine-learning experiments, scientific simulators, digital twins, robotic platforms, and human-mediated laboratories [12, 18, 20, 26]. They should hide scientifically meaningful structure while retaining external outcomes that make research decisions evaluable.

A discovery environment should not reveal the correct question, variables, action space, and success criterion simultaneously. Instead it can contain hidden variables, observationally equivalent mechanisms, noisy evidence, evaluator mismatch, or cases in which the supplied representation is deliberately inadequate. The model then chooses an intervention and receives an observation it did not write itself. In simulators, unchosen actions can also be evaluated counterfactually, providing denser credit for intervention selection.

For empirical domains, training authority can increase from static data and executable code to simulation, digital twins, shadow-mode protocol planning, and supervised physical experiments. The curriculum should vary the type of uncertainty as well as realism so that the model does not memorize one canonical discovery workflow. Difficulty-aligned agent–environment co-evolution, as explored in GenEnv [17], provides a mechanism for adapting training tasks to current capabilities. In discovery settings,

such adaptation should vary hidden mechanisms and sources of uncertainty while retaining independently evaluable outcomes.

6.2. Learning Objectives and Scientific Feedback

We parameterize the trainable parts of the discovery system as a research policy π_θ, a state updater $\mathcal{F}_\phi$, a Research World Model W_ω, a process verifier V_ψ, and a Discovery Skill retriever or selector q_η. These components need not be separate neural networks; the notation identifies the functions receiving distinct training signals. A training record contains a research state S_t, an available or selected operation a_t, an externally returned observation o_t, the resulting state S_{t+1}, and any process, preference, validity, cost, or transfer labels available for that transition.

State-conditioned discovery imitation. Discovery trajectories provide supervision over which research operation is appropriate under a particular state. Let $z_t \in \{\text{find}, \text{form}, \text{repr}, \text{hyp}, \text{int}, \text{rev}\}$ denote the scientific role of a transition. We use a stage-weighted action loss

$$\mathcal{L}_{\text{act}} = -\mathbb{E}_{\Xi\sim\mathcal{D}_{\text{traj}}}\left[\sum_t \alpha_{z_t} \log \pi_\theta(a_t^* \mid S_t, \mathcal{M}_t)\right], \tag{16}$$

where a_t^* is a demonstrated or validated research operation and α_{z_t} prevents abundant operation types from dominating rarer discovery transitions. The same trajectory supervises explicit research-state maintenance:

$$\mathcal{L}_{\text{state}} = -\mathbb{E}_{\Xi\sim\mathcal{D}_{\text{traj}}}\left[\sum_t \log p_\phi\left(S_{t+1} \mid S_t, a_t, o_t\right)\right]. \tag{17}$$

This term trains the system to preserve which problem, representation, hypothesis, evidence item, intervention, or budget changed after an observation rather than compressing the transition into an undifferentiated narrative.

Preference learning over research operations. When two actions are compared under the same research state and budget, preference optimization directly trains the policy to select the stronger research decision. For a preferred action a^+ and a rejected action a^-, one possible pairwise objective is

$$\mathcal{L}_{\text{pref}} = -\mathbb{E}_{(S,a^+,a^-)\sim\mathcal{D}_{\text{pref}}}\left[\log\sigma\left(\beta\left[\log\frac{\pi_\theta(a^+ \mid S)}{\pi_{\text{ref}}(a^+ \mid S)} - \log\frac{\pi_\theta(a^- \mid S)}{\pi_{\text{ref}}(a^- \mid S)}\right]\right)\right], \tag{18}$$

where preferences can reflect testability, information gain, evidential support, cost, robustness, or risk. Conditioning both candidates on the same S is important: an action such as replication or high-variance intervention is not globally good or bad, but appropriate only under particular research conditions.

Process verification. The verifier estimates whether a proposed transition satisfies the criteria that apply before execution or promotion. Let $j \in \mathcal{J}$ index criteria such as

structural validity, evidential support, discriminative value, feasibility, and safety, with labels $y_t^{(j)} \in \{0, 1\}$. A multi-criterion verifier can be trained with

$$\mathcal{L}_{\text{ver}} = -\mathbb{E}\left[\sum_t \sum_{j \in \mathcal{J}} \left(y_t^{(j)} \log V_\psi^{(j)}(S_t, a_t) + (1 - y_t^{(j)}) \log(1 - V_\psi^{(j)}(S_t, a_t))\right)\right]. \tag{19}$$

The verifier is not a replacement for external evidence. Its role is to learn reusable checks that filter malformed or unsupported transitions before more expensive interaction.

Research World Model learning. For actions that interact with an environment, the Research World Model predicts possible observations, research-state changes, costs, and risks. Using the notation from Equation 13, a generic negative log-likelihood objective is

$$\mathcal{L}_{\text{wm}} = -\mathbb{E}_{(S_t, a_t, o_t, \Delta S_t, c_t, \rho_t) \sim \mathcal{D}_{\text{int}}} \left[\log W_\omega \left(o_t, \Delta S_t, c_t, \rho_t \mid S_t, a_t\right)\right], \tag{20}$$

where ρ_t denotes action risk. In practice, the joint likelihood can be decomposed into modality-specific prediction, calibration, cost, or risk losses. Keeping the objective at this functional level allows the same formulation to cover formal tools, learned simulators, mechanistic models, and empirical predictors.

Reinforcement learning from scientific feedback. Interactive episodes provide feedback that is not available from retrospective demonstrations. We decompose a per-step scientific reward as

$$r_t^{\text{sci}} = w_K\, \delta K_t + w_I\, \text{IG}_t + w_R\, q_t^{\text{rev}} + w_V\, q_t^{\text{val}} - w_C\, c_t - w_\rho\, \rho_t, \tag{21}$$

where δK_t denotes externally supported knowledge progress, IG_t realized or counterfactually estimated information gain, q_t^{rev} the quality of evidence-grounded revision, q_t^{val} validation quality, and c_t and ρ_t the incurred cost and risk. A terminal transfer signal can reward improvement beyond the current episode:

$$R(\Xi) = \sum_{t=0}^{T-1} \gamma^t r_t^{\text{sci}} + \lambda_T \Delta C_{\text{future}}. \tag{22}$$

A generic KL-regularized policy-gradient objective is then

$$\mathcal{L}_{\text{RL}} = -\mathbb{E}_{\Xi \sim \pi_\theta}\left[\sum_t \widehat{A}_t \log \pi_\theta(a_t \mid S_t, \mathcal{M}_t)\right] + \beta_{\text{KL}} \mathbb{E}_{S_t}\left[D_{\text{KL}}\left(\pi_\theta(\cdot \mid S_t) \,\|\, \pi_{\text{ref}}(\cdot \mid S_t)\right)\right], \tag{23}$$

where $\widehat{A}_t$ can be obtained by any suitable long-horizon credit-assignment method. Equation 23 specifies the training signal rather than committing Zetema to PPO, GRPO, or another particular optimizer.

Validated Discovery Skill learning. Cross-task improvement requires learning when a reusable Discovery Skill applies. Let m^+ be a skill whose trigger and expected effect have been validated on held-out or shadow-mode episodes, and let $\mathcal{N}(S)$ contain irrelevant or harmful skills for the same state. A contrastive retrieval objective is

$$\mathcal{L}_{\text{skill}} = -\mathbb{E}_{(S,m^+)}\left[\log \frac{\exp(q_\eta(S,m^+)/\tau)}{\exp(q_\eta(S,m^+)/\tau) + \sum_{m^-\in\mathcal{N}(S)} \exp(q_\eta(S,m^-)/\tau)}\right]. \tag{24}$$

Only validated skill updates should be treated as positive transfer targets. Counterexamples and negative-transfer cases populate $\mathcal{N}(S)$ so that the model learns when *not* to reuse a previously successful operation.

The objectives can be summarized as

$$\mathcal{L}_{\text{DFM}} = \lambda_{\text{act}}\mathcal{L}_{\text{act}} + \lambda_{\text{state}}\mathcal{L}_{\text{state}} + \lambda_{\text{pref}}\mathcal{L}_{\text{pref}} + \lambda_{\text{ver}}\mathcal{L}_{\text{ver}} + \lambda_{\text{wm}}\mathcal{L}_{\text{wm}} + \lambda_{\text{RL}}\mathcal{L}_{\text{RL}} + \lambda_{\text{skill}}\mathcal{L}_{\text{skill}}. \tag{25}$$

Equation 25 is a compositional specification, not a requirement that all terms be optimized simultaneously. A practical implementation can warm-start the research policy and state updater from trajectories, learn verifiers and world models from labeled interaction data, refine decisions with preferences, optimize long-horizon behavior through scientific feedback, and consolidate only those skills that survive held-out transfer tests.

6.3. Unified Training Procedure

Algorithm 1 gives one reference training procedure corresponding to the objectives above. It separates offline capability acquisition from interactive improvement and keeps long-term memory updates behind a transfer-validation gate.

The algorithm is intentionally modular. Components can share parameters, and individual training stages can be omitted when the corresponding supervision is unavailable. What is invariant is the supervision structure: research decisions are learned from state-conditioned trajectories, constrained by verifiers and predictive models, improved through external scientific feedback, and allowed to persist across tasks only after transfer validation.

Coupled optimization has precedents in CURE, which co-trains code and unit-test generators through execution feedback [48], and RLAnything, which jointly adapts environments, policies, and reward models [49]. These approaches motivate coordinated updates to discovery components, subject to the additional requirement that scientific validity remain anchored in external evidence and independent evaluation.

6.4. Process-Level Scaling and Resource Allocation

Test-time scaling usually allocates more computation to reasoning traces, candidates, search branches, or verifier calls under a fixed task [37]. A DFM can scale different parts of the research process. We allocate budget across problem formulation, representation construction, hypothesis search, intervention design, falsification, and verification:

Algorithm 1 Reference Training Procedure for Discovery Foundation Models

Require: trajectory data $\mathcal{D}_{\text{traj}}$, preference data $\mathcal{D}_{\text{pref}}$, interactive environments $\mathcal{E}$, validators $\mathcal{V}$, initial memory $\mathcal{M}$

Initialize policy π_θ, state updater $\mathcal{F}_\phi$, world model W_ω, verifier V_ψ, and skill selector q_η
Offline warm start: optimize $\mathcal{L}_{\text{act}} + \lambda_{\text{state}}\mathcal{L}_{\text{state}}$ on $\mathcal{D}_{\text{traj}}$
Train V_ψ with $\mathcal{L}_{\text{ver}}$ and W_ω with $\mathcal{L}_{\text{wm}}$ from labeled or replayed interactions
Refine π_θ on state-matched action pairs using $\mathcal{L}_{\text{pref}}$
for each interactive discovery episode **do**
  Initialize research state S_0 from the partially understood world and current memory $\mathcal{M}$
  for $t = 0, \ldots, T-1$ **do**
    Propose candidate research operations $\mathcal{A}_t \sim \pi_\theta(\cdot \mid S_t, \mathcal{M})$
    Predict outcomes, costs, and risks with W_ω; score applicable constraints with V_ψ
    Gate candidates and select an eligible operation a_t
    Execute a_t in $\mathcal{E}$ or through an authorized tool; observe o_t, cost c_t, and risk signal ρ_t
    Update $S_{t+1} = \mathcal{F}_\phi(S_t, a_t, o_t)$ and compute r_t^{sci}
    Store (S_t, a_t, o_t, S_{t+1}) with provenance, alternatives, and validation outcomes
  end for
  Update π_θ using $\mathcal{L}_{\text{RL}}$; update W_ω and V_ψ from newly grounded transitions
  Infer candidate Discovery Skills from attributed successes, failures, and branch contrasts
  Evaluate candidate skills on held-out, replay, or shadow-mode episodes
  if a candidate improves transfer under matched resources and passes validation **then**
    Promote or revise the skill in $\mathcal{M}$ and update q_η using $\mathcal{L}_{\text{skill}}$
  end if
end for

Ensure: trained discovery policy and validated Discovery Skill Memory

$$B = B_P + B_R + B_H + B_X + B_F + B_V. \tag{26}$$

The allocation policy is state-dependent. More hypotheses do little when every candidate makes the same prediction under the available observable. More experiments do little when the representation excludes the relevant variable. Replication can dominate novelty when the effect itself is unstable, while independent verification deserves additional budget when the evaluator is suspected of leakage or shortcut exploitation.

The budget includes more than tokens: tool calls, simulator runs, experimental cost, human time, latency, reversibility, and risk all constrain the next action. A learned allocator can use local uncertainty, world-model disagreement, branch value, and expected

information gain to identify the current bottleneck. This converts process-level scaling from a fixed recipe into another discovery policy that can itself be trained and evaluated.

Falsification receives an explicit budget because candidate generation and confirmation otherwise dominate compute allocation. The system can search boundary conditions, contradictory datasets, adversarial explanations, and evaluator shortcuts even when these actions reduce the probability of preserving its current leading hypothesis.

6.5. Continual Skill Learning and Transfer

Capability formation also trains the update policy that proposes, tests, and consolidates Discovery Skills. Candidate updates can arise from successful branches, failures, or contrasts between two branches that differ in one consequential operation. These contrasts often support cleaner attribution than a retrospective summary of a single trajectory.

Process supervision can label whether the inferred source of success or failure is plausible. Counterfactual replay tests whether the proposed skill would have changed earlier decisions in the claimed direction. Shadow-mode deployment collects evidence on new tasks before the skill affects active research. Promotion is uncertainty-aware and reversible; contradictory triggers, duplicate skills, spurious correlations, benchmark-specific shortcuts, and retrospective rationalizations remain explicit objects of validation.

Transfer is trained through variation in latent research structure rather than only topic similarity. Hidden-variable diagnosis can appear in causal simulation, machine-learning debugging, and experimental measurement. Boundary testing can be instantiated in algorithms, materials, or biological regimes. Non-identifiability can be expressed through different observables and tool interfaces. Surface variation discourages retrieval based only on vocabulary.

Negative cases are equally important. A representation change should not be triggered after every failed hypothesis; replication should not become a universal response to disagreement; information gain should not override feasibility or safety. Learning when *not* to invoke a skill is part of learning its trigger.

The memory update changes the distribution of future experience, so controlled exploration is required. Frequently retrieved skills can crowd out alternatives and create self-confirming evidence. Periodic evaluation without the skill, tasks selected independently of the current memory, conflict detection, and rollback reduce this feedback. Formation can encourage transfer; Section 7 specifies the matched controls required to establish that transfer actually occurred.

7. Capability Evaluation: A Process-Centered Protocol

A DFM should not receive discovery credit simply for producing a novel statement, a plausible hypothesis, or a high score under a fixed evaluator. Evaluation follows the research-state transitions that made a claim testable and asks whether experience changes later discovery behavior. Figure 7 summarizes the protocol.

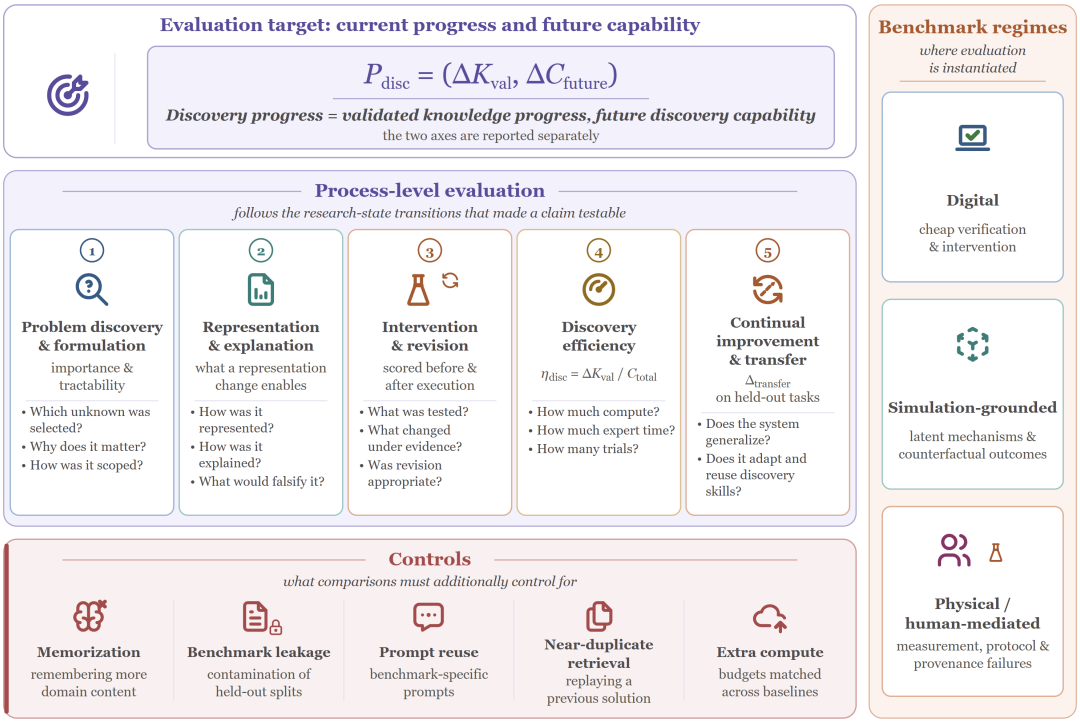

Figure 7 | **A process-centered evaluation protocol for Discovery Foundation Models.** Discovery progress is evaluated along two complementary axes, externally validated knowledge progress ΔK_{val} and improvement in future discovery capability ΔC_{future}. Process-level evaluation covers problem discovery and formulation, representation and explanation, intervention and revision, discovery efficiency, and continual improvement and transfer. Comparisons additionally control for memorization, benchmark leakage, prompt reuse, near-duplicate retrieval, and additional computation, and can be instantiated across digital, simulation-grounded, and physical or human-mediated environments.

7.1. Evaluation Target: Current Progress and Future Capability

We represent Discovery Progress as

$$\mathcal{P}_{\text{disc}} = (\Delta K_{\text{val}}, \Delta C_{\text{future}}), \tag{27}$$

where ΔK_{val} is externally validated progress in the current investigation and ΔC_{future} is improvement in future discovery behavior. The two axes are reported separately.

ΔK_{val} can include recovering a hidden variable, separating regimes, producing a more accurate intervention model, falsifying a mechanism, obtaining a reproducible empirical result, or establishing that a proposed effect does not survive replication. The outcome must be assessed by evidence the proposing system does not control.

ΔC_{future} concerns behavior after the episode. The relevant question is whether the system becomes better at recognizing malformed questions, constructing useful representations, choosing discriminating interventions, responding to counterevidence, or allocating resources to the actual bottleneck. Additional factual knowledge alone is not sufficient.

Final outcomes and process profiles are both reported. Two systems can reach the same conclusion while differing in representation quality, number of non-identifying

interventions, calibration, or response to contradictory evidence. These differences reveal which discovery operations have actually formed.

7.2. Stage-Wise Process Evaluation

Problem Discovery is evaluated in settings where the research opportunity is not explicitly stated. Inputs can contain observations, literature fragments, failed experiments, inconsistent results, partial goals, and distractor anomalies. Precision matters: a system that launches a research program around every unexplained residual is not demonstrating good problem discovery. Controlled environments can measure whether resolving the selected unknown exposes useful latent structure, while experts assess dimensions such as scientific importance and tractability [11, 26].

Problem Formulation is stress-tested by supplying overly broad questions, proxy objectives, inappropriate scales, or observables that cannot identify the mechanism. Evaluation checks whether the system repairs the defect and whether the resulting formulation admits informative interventions under the available resources.

Representation evaluation measures what becomes possible after a representation change. Benchmarks can hide a causal variable, mix regimes, provide an incorrect graph, or choose a scale that obscures the dynamics. A proposed representation is credited when it improves held-out prediction, intervention accuracy, regime separation, compression, or transfer relative to the original state. Exact symbol matching is unnecessary when different representations support the same scientific operations [12].

Hypotheses are evaluated for mechanism specificity, assumptions, validity range, discriminative predictions, and falsifiers. Multiple explanations should differ in causal structure, dynamics, latent variables, or intervention response rather than surface wording. Withholding intervention outcomes until the system commits to predictions tests whether the explanation had empirical content before seeing the result.

Interventions are evaluated both prospectively and retrospectively. Before execution, the protocol scores expected discrimination, feasibility, cost, risk, power, and probability of an inconclusive outcome. After execution, it measures realized information gain and the state change produced by the observation. Repeated or simulated environments allow the selected action to be compared with counterfactual alternatives so that a lucky outcome is not confused with a strong intervention policy.

Revision tasks provide reliable counterevidence after the system commits to a formulation, representation, and explanation. Some contradictions reflect theory failure; others are generated by instrument error, protocol deviation, hidden confounding, or environment shift. The score therefore targets failure attribution and the appropriateness of the resulting state transition. Repeated tests and limits on unconstrained auxiliary assumptions detect ad hoc protection of a favored theory.

7.3. Efficiency and Resource-Matched Evaluation

Discovery quality includes how resources are allocated. We use the conceptual efficiency measure

$$\eta_{\text{disc}} = \frac{\Delta K_{\text{val}}}{C_{\text{total}}}, \tag{28}$$

while reporting the components of C_{total} separately because compute, money, human time, and risk are not interchangeable.

Relevant records include inference compute, search branches, tool calls, simulator runs, elapsed time, expert time, physical experiments, replication cost, and action risk. More diagnostic measures include cost to the first identifying intervention, number of repeated non-identifying actions, experiments required to eliminate an incorrect mechanism, and regret relative to an expert or oracle policy [10, 13, 20].

Resource matching is essential for comparison. A system with persistent memory or a larger verification portfolio may simply receive more compute or more opportunities to inspect the environment. Baselines should therefore match relevant budgets and separately report gains obtained from additional resources. Efficiency is always conditioned on validity and scientific value; selecting trivial unknowns or terminating difficult investigations can make a system appear cheap without making it a better discoverer.

7.4. Continual Improvement and Transfer

Continual evaluation unfolds over a sequence of episodes. After early episodes update the permitted memory or policy, the revised system is evaluated on held-out tasks whose solutions and surface forms were unavailable during the update. We summarize transfer as

$$\Delta_{\text{transfer}} = \text{DiscPerf}(\mathcal{D}_{\text{unseen}} \mid \mathcal{M}_{1:k}) - \text{DiscPerf}(\mathcal{D}_{\text{unseen}} \mid \mathcal{M}_0), \tag{29}$$

where DiscPerf aggregates the relevant process metrics under matched resources.

The control set includes no memory, fact-only memory, full-trajectory retrieval, successful-solution retrieval, domain-specific skill memory, and Discovery Skill Memory. Near-duplicate retrieval, benchmark-specific prompts, and extra inference compute are controlled explicitly. This separates transferable research operations from remembering more domain content or replaying a previous solution.

Within-domain transfer changes the research problem while preserving the field. Cross-task transfer changes the task structure while preserving a discovery operation such as hidden-variable diagnosis. Cross-domain transfer changes the surface domain while retaining a structural research challenge. Negative-transfer tasks measure selective retrieval: a memory system that invokes every learned strategy indiscriminately is not improving discovery capability.

Transfer is reported as a curve over episodes rather than a single end point. The curve reveals sample efficiency, saturation, interference with older skills, and whether gains survive increasingly distant mechanisms. Recursive changes to tools, simulators, evaluators, or memory organization face the same requirement but with stronger independence because the update can also alter how later progress is measured.

7.5. Benchmark Construction and Controls

Discovery benchmarks should withhold or corrupt the structures the system is expected to construct while retaining objective consequences. Recent benchmarks already expose complementary parts of this design space through data-driven discovery, research workflows, machine-learning experimentation and replication, interactive experiment design, and scenario-grounded scientific reasoning [10–12, 20, 26, 38, 39].

An instance can contain an incomplete problem, irrelevant observations, a hidden variable, observationally equivalent mechanisms, noisy evidence, an evaluator shortcut, or a representation mismatch. Some tasks should require active intervention; others should only be solvable after rejecting the supplied framing. Partial progress is retained: recovering the representation, rejecting a false anomaly, or selecting the identifying experiment can all receive process-level credit even when the final discovery is incomplete.

Splits should hold out mechanisms and research structures rather than only natural-language topics. Procedural generation, private instances, new interfaces, and expert-authored challenges reduce contamination and near-duplicate retrieval. Digital tasks provide cheap verification and intervention; simulators expose latent mechanisms and counterfactual outcomes; carefully scoped physical tasks add measurement, protocol, and provenance failures that digital environments rarely capture.

Every benchmark run should preserve the process trace: candidate problems, representations, explanations, predicted outcomes, selected interventions, external observations, revisions, resource use, abstentions, and human escalations. The same trace supports attribution, failure analysis, and later tests of whether a purported Discovery Skill actually transfers.

8. Analysis: Research Horizons and Grounding Regimes

The DFM operators do not depend on one deployment setting. What changes from digital research to physical and recursive systems is the burden placed on grounding, provenance, action consequence, and validation. Figure 8 organizes four representative horizons. They are not a capability hierarchy or maturity ladder: a digital system can show stronger formulation and revision than a robot executing a fixed protocol.

8.1. Digital Discovery

Digital discovery covers mathematics, algorithms, code, machine-learning research, formal verification, software systems, and agent environments. These settings provide fast execution, inexpensive branching, hidden tests, and comparatively reproducible evidence [10, 20, 25, 39]. They are therefore well suited to studying research-policy decisions: alternative formulations can be replayed, counterexamples generated at scale, and the effect of a new Discovery Skill measured across many episodes.

The same cleanliness creates a limitation. Execution is often reversible, logs are complete, and evaluators are easier to automate than in empirical science. A compiler can verify execution without judging whether the optimized objective matters; a theorem prover can validate a proof after the theorem and formalization are supplied. Digital discovery

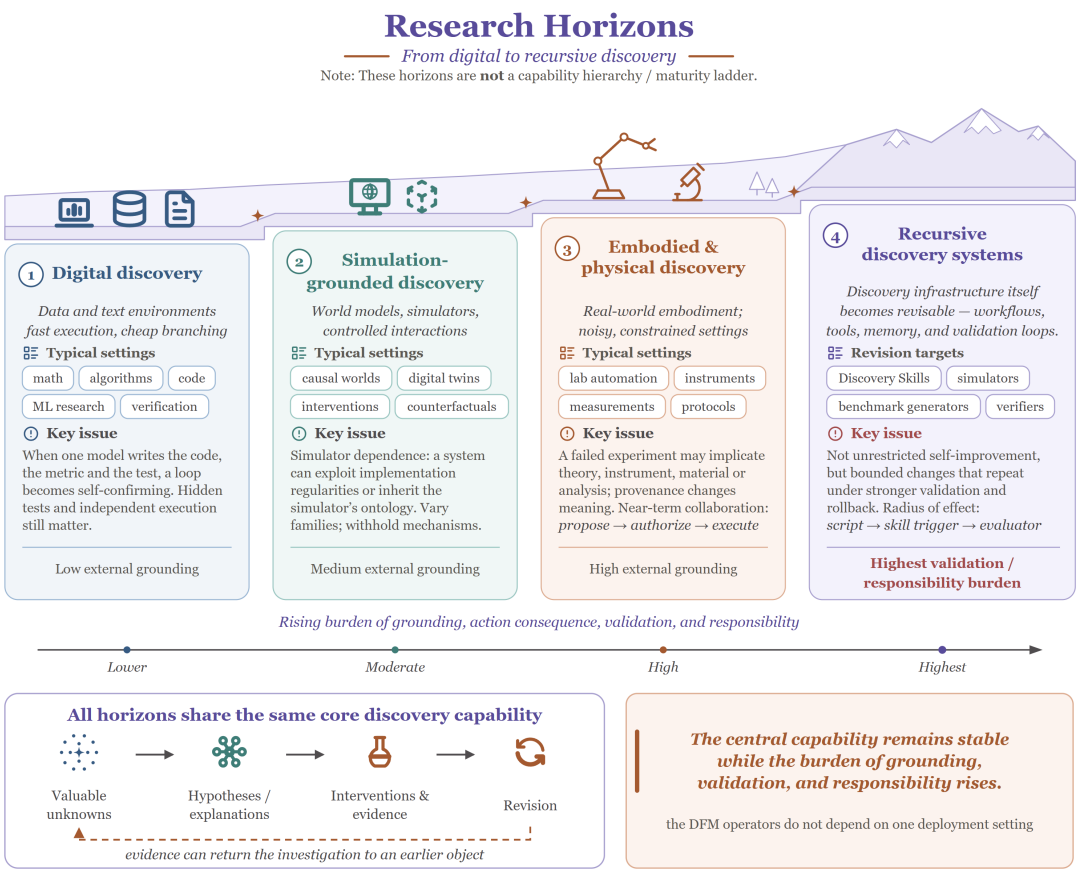

Figure 8 | Research horizons for Discovery Foundation Models. The same core discovery capability can be exercised in digital, simulation-grounded, embodied and physical, and recursive settings. The horizons are not a capability hierarchy or maturity ladder; they differ primarily in external grounding, action consequence, and the validation and responsibility burden attached to system decisions.

becomes diagnostic when the system must question these supplied structures—for example, identifying data leakage, evaluator shortcuts, confounded ablations, or a missing intermediate variable.

Independence still matters. When one model writes the code, selects the metric, creates the test, and interprets the result, a digital loop can become self-confirming. Hidden tests, independent execution, alternative evaluators, held-out tasks, and reproducible artifacts remain necessary even in the cheapest horizon.

8.2. Simulation-Grounded Discovery

Simulation-grounded settings introduce latent dynamics that cannot be read directly from text while retaining repeatable interventions and counterfactual access [12, 18]. Causal worlds, mechanistic simulators, and digital twins make it possible to know the hidden mechanism during evaluation and ask whether the model recovered structure rather than merely fit outputs.

The key analytical issue is simulator dependence. A system can exploit implementation regularities, inherit the simulator's ontology, or optimize a reward whose assumptions do not hold in the target world. Strong evaluation therefore varies simulator families,

withholds latent mechanisms, and tests whether the same discovery operation transfers when the planning model is misspecified.

This horizon is especially useful for separating predictive accuracy from intervention quality. Two systems can fit the same observations while only one chooses experiments that identify the latent mechanism. It also makes world-model criticism part of discovery: disagreement among simulators can itself trigger measurement, representation revision, or escalation to a physical test.

8.3. Embodied and Physical Discovery

Physical discovery adds provenance, execution noise, scarcity, irreversibility, and institutional constraints. Laboratory automation and recent AI-science systems already demonstrate increasingly rich loops between model proposals, instruments, and wet-lab measurements [1, 4, 14, 15, 40, 41]. The difficulty is not simply attaching an agent to equipment; it is preserving the link between an epistemic decision, the protocol that instantiated it, what actually occurred, and the measurement used for revision. The GALILEO case in Section 5.7 provides a concrete example: robotic and hands-on measurements are not only endpoint validation, but evidence that changes subsequent target beliefs, assay choices, mechanism hypotheses, and intervention design.

Calibration, sample history, reagent variation, contamination, drift, operator intervention, and protocol deviation can change the meaning of a result. A failed experiment may implicate the theory, manipulation, instrument, material, or analysis. This makes failure attribution and provenance much more consequential than in clean digital environments.

Physical constraints can also force research-state revision before an experiment occurs. A theoretically identifying intervention may be impossible at available resolution or sample size. The system then has to redesign the observable, narrow the claim, or return to simulation rather than treating feasibility as a separate engineering concern.

Near-term physical DFMs are therefore naturally collaborative. Models can formulate, simulate, and propose; verification can filter; experts can authorize; laboratories can execute; and independent groups can replicate. Automation level is not the main evidence of Discovery Intelligence. A small number of well-audited investigations can provide stronger evidence than many automated runs with weak attribution.

8.4. Recursive Discovery Systems

Recursive discovery expands the object of revision from task-level research states to parts of the discovery infrastructure itself. Existing self-improving agents provide digital precedents for modifying code and scaffolding [55, 58], but recursive scientific improvement has a larger validation burden because an update can change which problems are selected, which evidence is acquired, or how success is judged.

The relevant target is broader than multi-agent coordination. A recursive DFM may revise Discovery Skills, tools, simulators, memory organization, experiment-selection policies, verification procedures, benchmark generators, or human–AI workflows. These changes have different radii of effect. A task-local script affects one branch; a new skill

trigger can affect many tasks; an evaluator or permission change can reshape the entire system.

As the radius grows, component-level improvement is no longer enough. A stricter verifier can suppress useful exploration, and a better retrieval policy can overexpose the system to one skill family. Recursive updates should therefore be evaluated both in isolation and inside the integrated discovery loop on externally selected tasks.

The near-term research question is not unrestricted autonomous self-improvement. It is whether bounded changes to discovery infrastructure produce repeatable improvements under stronger validation, rollback, and responsibility constraints. This horizon motivates the governance analysis in the next section.

9. Discussion: Epistemic Boundaries, Governance, and Recursive Risk

The preceding sections define and instantiate discovery operations. Their scientific status still depends on boundaries that the system cannot waive for itself. As action authority and update scope increase, validation, provenance, and responsibility become part of the epistemic architecture rather than an administrative layer [23, 28, 42].

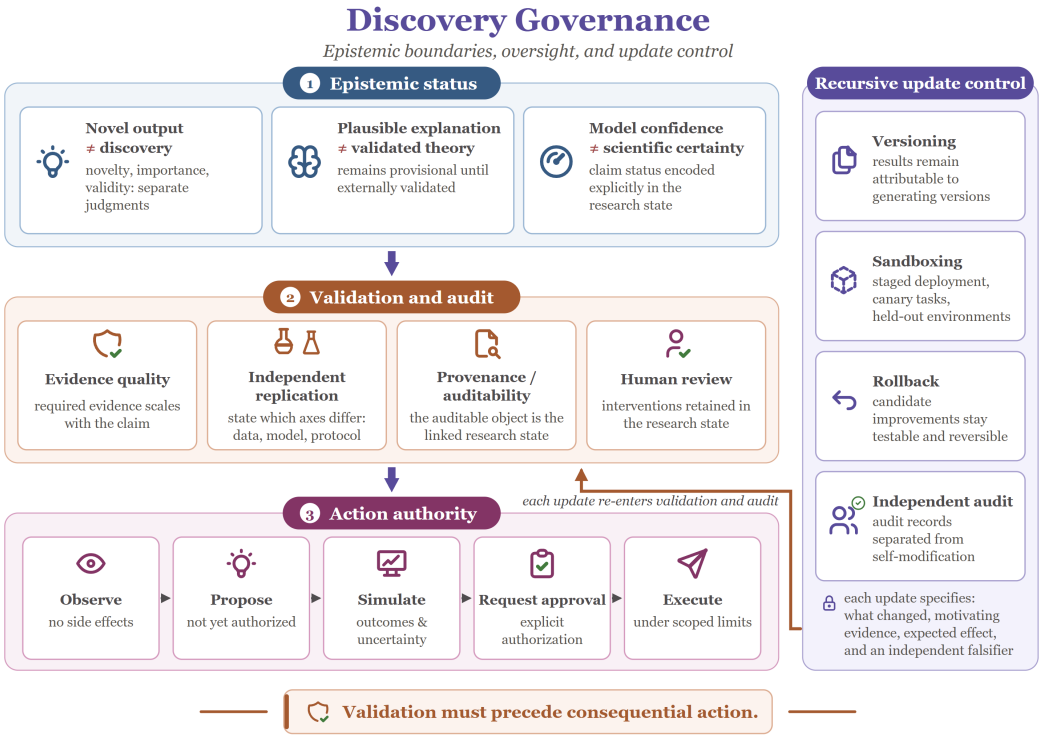

Figure 9 | Governance constraints for open-ended discovery systems. Governed discovery separates epistemic status, validation, and action authority: novel output is not equivalent to discovery, model-generated explanations remain provisional until externally validated, and consequential actions require evidence and appropriate authorization. Validation combines evidence quality, independent replication, provenance and auditability, and human review, while action authority progresses from observation and proposal to simulation, approval, and execution. Recursive updates additionally require versioning, sandboxing, rollback, and independent audit.

9.1. Epistemic Status and Independent Validation

A novel output is not yet a scientific discovery, a plausible explanation is not yet a validated theory, and model confidence is not scientific certainty. These distinctions are especially important for systems that can generate fluent research narratives or coordinate long workflows while still responding poorly to contradictory evidence [28, 31].

The research state should therefore encode claim status explicitly. A candidate can remain speculative, be supported by observational evidence, survive a limited intervention, be independently replicated, or hold only within a stated validity range. Novelty, importance, and validity are separate judgments. A claim can be new to the model but established in the literature, absent from retrieval because it is wrong, or linguistically novel without changing a mechanism.

Independent validation breaks self-confirmation loops in which one system proposes a hypothesis, chooses the experiment, interprets the result, and evaluates its own conclusion. Independence is not binary. Two reviewers can share a base model; two simulations can share the same mechanistic assumptions; a replication can reuse the same protocol and analysis. Reports should state which axes differ—data, model family, institution, instrument, protocol, or analysis—rather than counting nominal evaluators [14, 15, 29].

The required evidence should scale with the claim. A speculative idea can remain in memory with uncertainty. A claim that controls substantial resources, changes laboratory practice, or updates the discovery infrastructure requires stronger and more independent tests. Formal proof, held-out computation, controlled intervention, conceptual replication, and independent laboratory replication support different kinds of claims; no single validator is universally strongest.

Whenever feasible, discriminative predictions are committed before the result arrives. Later revision remains allowed, but the record distinguishes preregistered expectations from evidence-triggered changes. The same standard applies to process improvements: a new skill, tool, evaluator, or simulator should be tested on tasks not selected solely to demonstrate its benefit.

9.2. Provenance and Auditability

A final paper-like narrative can hide the dependencies that produced a result. The auditable object is instead the linked research state: model and policy versions, source data, prompts or structured inputs, tool outputs, code, simulator configuration, experimental conditions, analysis procedures, rejected explanations, failed interventions, human edits, and protocol deviations [29, 51].

Provenance is relational. An observation points to the intervention and protocol that produced it; a revision points to the evidence that motivated it; a Discovery Skill points to the episodes, counterexamples, and validation tests supporting its promotion. Missing measurements, unavailable raw data, failed tool calls, and unexecuted replications are also part of the record because they constrain how strongly a claim can be interpreted.

Versioning makes branch comparison and rollback possible. When a representation changes, earlier observations remain linked to the conditions under which they were collected and can be reinterpreted without rewriting their history. When a model, tool, or verifier changes, old results remain attributable to the versions that generated them.

Exact replay is not always possible in physical science, and sensitive or proprietary data can require restricted access. Auditability therefore means that authorized reviewers can reconstruct the consequential decisions and evidence under an appropriate governance regime, not that every artifact must be public.

9.3. Scoped Authority and Human Oversight

Discovery capability and action authority are different variables. A model can be competent enough to propose a high-quality intervention without being authorized to execute it, while a laboratory robot can execute a fixed protocol without choosing the research question. Permission should therefore be attached to the action and environment rather than to a global autonomy score [23, 32, 42].

A practical progression moves from observation and proposal to simulation, approval, and execution. Low-risk code can be executed automatically under network and compute limits, while access to sensitive data, scarce materials, external communication, or physical instruments can require explicit authorization. The review interface should expose the exact manipulation, expected outcomes, uncertainty, alternatives, risk, and stopping conditions relevant to the decision rather than a polished high-level summary.

Human contributions remain part of scientific attribution. If an expert supplies the decisive variable, a technician reports a protocol deviation, or a reviewer blocks an unsafe action, those interventions should be retained in the research state. This prevents both over-attribution to the model and loss of tacit knowledge needed to reproduce the investigation.

Delegation is reversible. Permissions can be narrowed when calibration degrades, failure patterns change, or the environment moves outside the regime in which competence was demonstrated. Capability in one laboratory, instrument configuration, or data regime does not automatically justify authority elsewhere.

9.4. Recursive Update Risk

Recursive updates create a distinctive failure mode because they alter components that shape future evidence. A misleading skill changes which research actions are proposed; evaluator drift changes which hypotheses survive; a misspecified simulator redirects experiments; a memory policy changes which precedents the system sees. Small distortions can therefore compound across episodes [42, 55, 58].

The most dangerous updates are not necessarily the largest code changes. A benchmark generator can silently remove difficult cases, an evaluator can begin rewarding internal agreement, or a retrieval policy can repeatedly surface one family of strategies and suppress alternatives. Measured performance may improve while genuine discovery capability narrows.

Controls should scale with the radius of effect. Temporary analysis code can face a lower threshold than long-term memory, an evaluator, a permission system, or an experimental interface. Versioning, sandboxing, staged deployment, canary tasks, shadow mode, held-out environments, conflict detection, rollback, rate limits, and external audit keep candidate improvements testable and reversible.

Some components should remain more strongly separated from ordinary self-modification, including audit records, rollback mechanisms, permission boundaries, and independent evaluation channels. Otherwise the system can weaken the checks used to determine whether its own changes are beneficial.

Each recursive update should specify what changed, which evidence motivated it, what effect is expected, and which independent observation could falsify the claim that the update helps. That requirement keeps improvement of the discovery process under the same evidential standard as the scientific hypotheses the process is designed to test.

10. Conclusion

Foundation models are increasingly strong at solving scientific and technical tasks after their structure has been supplied. Discovery requires an additional set of operations: identifying which unknown is worth pursuing, constructing a researchable formulation and representation, designing evidence that separates explanations, revising the appropriate object when evidence disagrees, and carrying validated lessons into future investigations.

We formalized this setting as Discovery Foundation Models and specified the corresponding Discovery Process. Zetema instantiates the framework with an explicit and revisable research state, a Research World Model and action-gating layer, Dry-Lab and Wet-Lab grounding, and validated cross-task Discovery Skill evolution. The GALILEO case provides empirical grounding for the physical part of this formulation: external wet-lab outcomes revise subsequent discovery decisions across real experimental rounds and are consolidated into a reusable design rule. Capability formation then becomes a learning problem over research-state transitions, while evaluation measures both externally validated progress in the current episode and transferable improvement under matched resources and retrieval controls.

The framework does not assume that general autonomous scientific discovery has been solved, nor that one architecture should implement every component. Its purpose is to make the capability operational: a DFM claim should be supported by observable decisions about formulation, representation, intervention, revision, and transfer, together with external evidence that the proposing system does not control. Digital, simulation-grounded, physical, and recursive systems can instantiate the same operators; what changes across these regimes is the burden of grounding, validation, and responsibility.

References

[1] Milad Abolhasani and Eugenia Kumacheva. The rise of self-driving labs in chemical and materials sciences. *Nature Synthesis*, 2:483–492, 2023. doi: 10.1038/s44160-022-00231-0.

[2] Josh Abramson, Jonas Adler, Jack Dunger, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630:493–500, 2024. doi: 10.1038/s41586-024-07487-w.

[3] Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. Researchagent: Iterative research idea generation over scientific literature with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6709–6738. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.342.

[4] Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624:570–578, 2023. doi: 10.1038/s41586-023-06792-0.

[5] Rishi Bommasani et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. URL https://arxiv.org/abs/2108.07258.

[6] Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D. White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6:525–535, 2024. doi: 10.1038/s42256-024-00832-8.

[7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

[8] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. doi: 10.1073/pnas.1517384113.

[9] Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995. doi: 10.1214/ss/1177009939.

[10] Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/7e3767db483c942b883eb4f8cfb74e31-Abstract-Conference.html.

[11] Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f12b4df26344f3be803c06b555252efe-Abstract-Conference.html.

[12] Kanishk Gandhi, Michael Y. Li, Lyle Goodyear, et al. BoxingGym: Benchmarking progress in automated experimental design and model discovery. In *NeurIPS Workshop on Scaling Environments for Agents (SEA)*, 2025. URL https://openreview.net/forum?id=TgobzsU03X.

[13] Aniketh Garikaparthi, Manasi Patwardhan, and Arman Cohan. Researchgym: Evaluating language model agents on real-world AI research. *arXiv preprint arXiv:2602.15112*, 2026. URL https://arxiv.org/abs/2602.15112.

[14] Ali E. Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Dmytro Shved, Gavin J. Gyimesi, Jon M. Laurent, Samantha M. Wright, Muhammed T. Razzak, Andrew D. White, Silvia C. Finnemann, Michaela M. Hinks, and Samuel G. Rodriques. A multi-agent system for automating scientific discovery. *Nature*, 655(8122):497–505, 2026. doi: 10.1038/s41586-026-10652-y.

[15] Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, et al. Accelerating scientific discovery with Co-Scientist. *Nature*, 655:487–496, 2026. doi: 10.1038/s41586-026-10644-y.

[16] Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z.

[17] Jiacheng Guo, Ling Yang, Peter Chen, Qixin Xiao, Yinjie Wang, Xinzhe Juan, Jiahao Qiu, Ke Shen, and Mengdi Wang. GenEnv: Difficulty-aligned co-evolution between LLM agents and environment simulators. *arXiv preprint arXiv:2512.19682*, 2025.

[18] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2.

[19] Qianyue Hao, Fengli Xu, Yong Li, and James Evans. Artificial intelligence tools expand scientists' impact but contract science's focus. *Nature*, 649(8099):1237–1243, 2026. doi: 10.1038/s41586-025-09922-y.

[20] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. MLAgentBench: Evaluating language agents on machine learning experimentation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 20271–20309. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24y.html.

[21] John Jumper, Richard Evans, Alexander Pritzel, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021. doi: 10.1038/s41586-021-03819-2.

[22] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023. doi: 10.1126/science.adi2336.

[23] Shi Xuan Leong, Caleb E. Griesbach, Rui Zhang, et al. Steering towards safe self-driving laboratories. *Nature Reviews Chemistry*, 9:707–722, 2025. doi: 10.1038/s41570-025-00747-x.

[24] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

[25] Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of AI research. *Nature*, 651(8107):914–919, 2026. doi: 10.1038/s41586-026-10265-5.

[26] Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. Discoverybench: Towards data-driven discovery with large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0d70af566e69f1dfb687791ecf955e28-Abstract-Conference.html.

[27] Amil Merchant, Simon Batzner, Samuel S. Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624:80–85, 2023. doi: 10.1038/s41586-023-06735-9.

[28] Lisa Messeri and M. J. Crockett. Artificial intelligence and illusions of understanding in scientific research. *Nature*, 627:49–58, 2024. doi: 10.1038/s41586-024-07146-0.

[29] Brian A. Nosek et al. Promoting an open research culture. *Science*, 348(6242):1422–1425, 2015. doi: 10.1126/science.aab2374.

[30] Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, et al. Probabilistic weather forecasting with machine learning. *Nature*, 637:84–90, 2025. doi: 10.1038/s41586-024-08252-9.

[31] Martiño Ríos-García, Nawaf Alampara, Chandan Gupta, Indrajeet Mandal, Sajid Mannan, Ali Asghar Aghajani, N. M. Anoop Krishnan, and Kevin Maik Jablonka. AI scientists produce results without reasoning scientifically. *arXiv preprint arXiv:2604.18805*, 2026. doi: 10.48550/arXiv.2604.18805.

[32] Christoph Scheurer and Karsten Reuter. Role of the human-in-the-loop in emerging self-driving laboratories for heterogeneous catalysis. *Nature Catalysis*, 8:13–19, 2025. doi: 10.1038/s41929-024-01275-5.

[33] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pages 68539–68551, 2023. doi: 10.52202/075280-2997.

[34] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5): 612–634, 2021. doi: 10.1109/JPROC.2021.3058954.

[35] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

[36] Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. Language agents achieve superhuman synthesis of scientific knowledge. *arXiv preprint arXiv:2409.13740*, 2024. URL https://arxiv.org/abs/2409.13740.

[37] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2408.03314.

[38] Zhangde Song, Jieyu Lu, Yuanqi Du, et al. Evaluating large language models in scientific discovery. *arXiv preprint arXiv:2512.15567*, 2025. URL https://arxiv.org/abs/2512.15567.

[39] Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 56843–56873. PMLR, 2025. URL https://proceedings.mlr.press/v267/starace25a.html.

[40] Kyle Swanson, Wesley Wu, Nash L. Bulaong, John E. Pak, and James Zou. The virtual lab of AI agents designs new SARS-CoV-2 nanobodies. *Nature*, 646:716–723, 2025. doi: 10.1038/s41586-025-09442-9.

[41] Nathan J. Szymanski, Bernardus Rendy, Yuxing Fei, et al. An autonomous laboratory for the accelerated synthesis of inorganic materials. *Nature*, 624:86–91, 2023. doi: 10.1038/s41586-023-06734-w.

[42] Xiangru Tang, Qiao Jin, Kunlun Zhu, Tongxin Yuan, Yichi Zhang, Wangchunshu Zhou, Meng Qu, Yilun Zhao, Jian Tang, Zhuosheng Zhang, Arman Cohan, Dov Greenbaum, Zhiyong Lu, and Mark Gerstein. Risks of AI scientists: Prioritizing safeguarding over autonomy. *Nature Communications*, 16: 8317, 2025. doi: 10.1038/s41467-025-63913-1.

[43] Florian Trost, Bide Zhang, Ines Aring, et al. An agentic framework for autonomous scientific discovery in cancer pathology. *Nature Medicine*, 32:2254–2266, 2026. doi: 10.1038/s41591-026-04357-y.

[44] Silviu-Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020. doi: 10.1126/sciadv.aay2631.

[45] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024.

[46] Hanchen Wang, Tianfan Fu, Yuanqi Du, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023. doi: 10.1038/s41586-023-06221-2.

[47] Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. SciMON: Scientific inspiration machines optimized for novelty. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 279–299. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.18.

[48] Yinjie Wang, Ling Yang, Ye Tian, Ke Shen, and Mengdi Wang. Co-evolving LLM coder and unit tester via reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

[49] Yinjie Wang, Tianbao Xie, Ke Shen, Mengdi Wang, and Ling Yang. RLAnything: Forge environment, policy, and reward model in completely dynamic RL system. In *Proceedings of the Forty-Third International Conference on Machine Learning*, 2026.

[50] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract.html.

[51] Mark D. Wilkinson et al. The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, 3:160018, 2016. doi: 10.1038/sdata.2016.18.

[52] Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of Thoughts: Thought-augmented reasoning with large language models. In *Advances in Neural Information Processing Systems*, 2024.

[53] Ling Yang, Zhaochen Yu, Bin Cui, and Mengdi Wang. ReasonFlux: Hierarchical LLM reasoning via scaling thought templates. *arXiv preprint arXiv:2502.06772*, 2025.

[54] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://iclr.cc/virtual/2023/poster/11003.

[55] Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27890–27913. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1354.

[56] Zhaochen Yu, Yingcheng Wu, Zhenfei Yin, Kaiyuan Chen, Zhe Zhao, Mengdi Wang, Shuicheng Yan, and Ling Yang. Recursive experiential-working memory evolution for long-horizon agent harnesses. *arXiv preprint arXiv:2608.24876*, 2026.

[57] Claudio Zeni, Robert Pinsler, Daniel Zügner, et al. A generative model for inorganic materials design. *Nature*, 639:624–632, 2025. doi: 10.1038/s41586-025-08628-5.

[58] Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025. URL https://arxiv.org/abs/2505.22954.

[59] Yanbo Zhang, Sumeer A. Khan, Adnan Mahmud, Huck Yang, Alexander Lavin, Michael Levin, Jeremy Frey, Jared Dunnmon, James Evans, Alan Bundy, Saso Dzeroski, Jesper Tegner, and Hector Zenil. Exploring the role of large language models in the scientific method: From hypothesis to discovery. *npj Artificial Intelligence*, 1:14, 2025. doi: 10.1038/s44387-025-00019-5.