Advanced Materials

WILEY

REVIEW

From Execution to Discovery: Physical Verification as the Key Gap in AI Agents for Science

Yixuan Zhang[1] | Bin Cao[2] | Tong-Yi Zhang[1,2]

[1]Materials Genome Institute, Shanghai University, Shanghai, China | [2]Guangzhou Municipal Key Laboratory of Materials Informatics, Advanced Materials Thrust, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China |

Correspondence: Bin Cao (bincao@hkust-gz.edu.cn); Tong-Yi Zhang (mezhangt@hkust-gz.edu.cn)

Keywords: AI agents | capability externalization | scientific discovery | materials discovery | environmental verifiability | research automation

ABSTRACT

AI agents operate reliably in digital environments such as software development, yet they still struggle to achieve autonomous scientific discovery. This gap reflects not only model capability but also how agent capabilities are externalized and how reliably an environment verifies the consequences of agent actions. A three-stage framework of capability externalization is proposed: reasoning externalization makes knowledge, memory, and intermediate inference inspectable; execution externalization makes tool use and long-horizon workflows governable; and world-interaction externalization integrates environmental evidence into hypothesis revision and action selection. Materials discovery serves as a demanding testbed spanning literature-grounded hypothesis generation, computational design, automated experimentation, and iterative knowledge updating. Completing a predefined research workflow, however, does not constitute scientific discovery. Sustained autonomy requires agents to assess whether new observations support or challenge a hypothesis and to select subsequent experiments accordingly. Environmental verifiability is therefore identified as a central systems constraint on agent autonomy: the extent to which an environment translates actions into timely, attributable, reproducible, and affordable evidence. Physical verification capacity is accordingly positioned as a key scaling dimension for autonomous discovery.

1 | Introduction

Agents now plan over long horizons, write and repair software, invoke diverse tools, and search difficult reasoning spaces. LLMs supply the general cognitive substrate [10, 16]; systems such as ReAct [11], along with commercial agents like Claude Code, Codex and Devin, show how this substrate can be turned into sustained digital work. This progress has raised expectations for materials research: if agents can tackle long-horizon coding tasks and formal reasoning problems, why can they not autonomously advance a materials research project from an initial hypothesis to a reproducible result? The answer cannot lie solely in the need for larger models or more tools. Our position is that the bottleneck is *environmental verifiability*. Code offers fast, low-cost, repeatable feedback through compilers and tests; formal reasoning offers proof checking and counterexamples. Materials research instead requires an agent to infer hidden states from synthesis histories and noisy characterization [6], attribute measured properties to the right causes, and establish reproducibility across experiments. An agent can generate candidates, run simulations, plan experiments, or operate instruments without being able to determine whether its scientific conclusions are correct. The prediction–synthesis–characterization–interpretation–property–reproduction chain is therefore constrained by its least verifiable link. This constraint distinguishes *tool-driven task completion* from *autonomous scientific discovery*. It also helps explain why changing the harness can alter long-horizon performance even when the underlying model remains the same [17], and why robotic closed-loop execution alone does not constitute discovery [14]. We argue that model capability and tool availability are necessary but insufficient: the practical ceiling on sustained autonomy is set by how reliably, rapidly, and affordably the environment can verify an agent's decisions.

Existing reviews offer valuable but partial perspectives on agent autonomy. Component-oriented surveys catalogue reasoning,

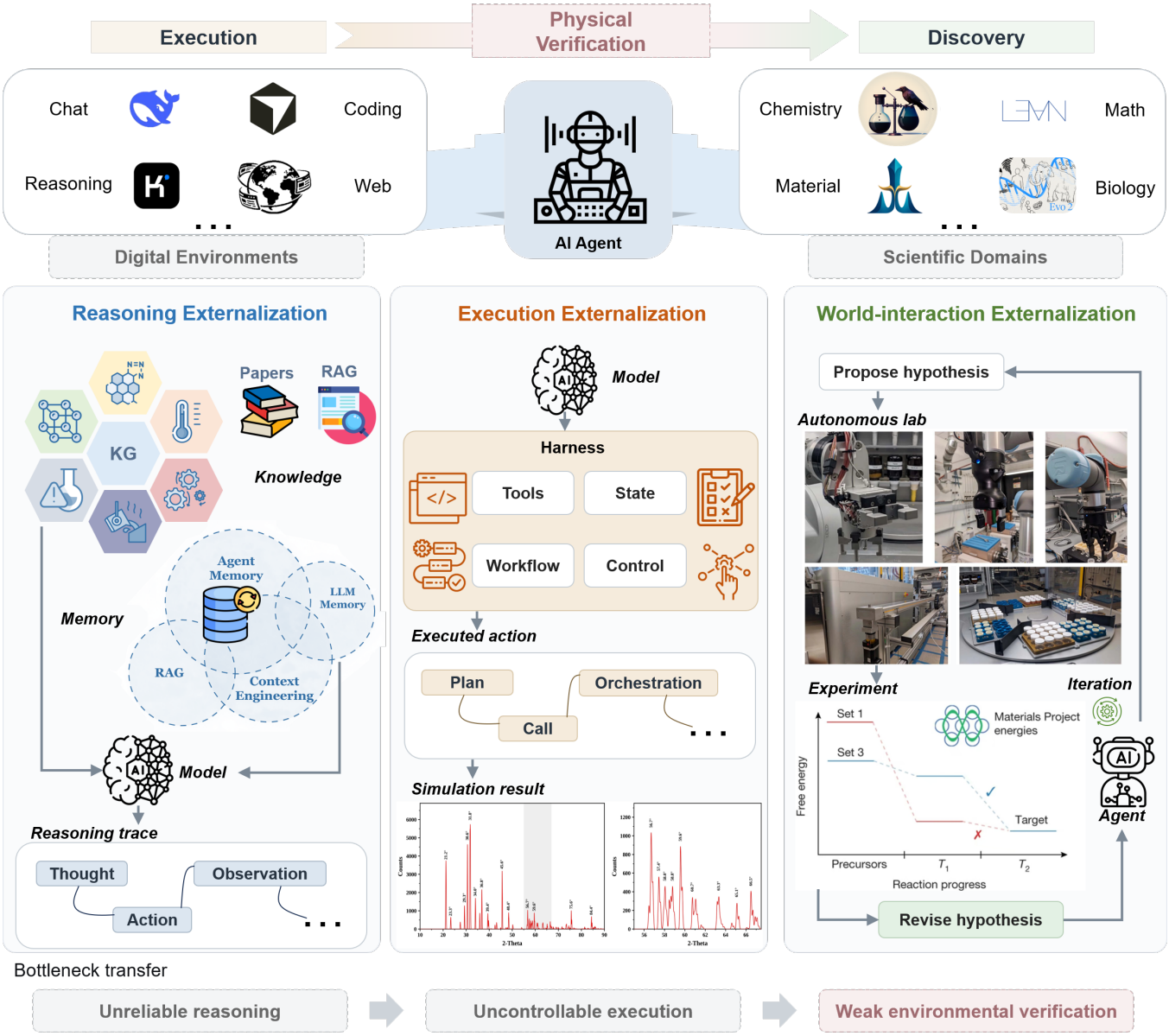


FIGURE 1 | **The three-stage externalization of agent and the associated bottleneck transfer.** The upper panel depicts the AI Agent bridging **Digital Environments** and **Scientific Domains**, with **Physical Verification** as the critical gap for true scientific discovery. [1–7] The lower panel details three stages: **Reasoning Externalization** (injecting knowledge and memory to emit inspectable reasoning traces [8–11]); **Execution Externalization** (delegating actions to a controllable harness for testable and bounded execution [12, 13]); and **World-interaction Externalization** (using act/revise cycles to drive hypothesis revision toward epistemic closure [14, 15]). The bottom arrows trace the bottleneck transfer from unreliable reasoning through uncontrollable execution to weak environmental verification, defining the **verifiability ceiling** (Supporting Information, Note 1) that limits sustained AI autonomy in scientific discovery.

planning, memory, and multi-agent modules[18–20], yet offer limited insight into why autonomy varies across domains. Engineering accounts explain how harnesses, tools, and model optimization co-evolve [21–23], but rarely examine what the target environment can reliably verify. Application benchmarks assess task performance[24–27], while science-focused reviews document adoption within the domain [28–30]; neither alone explains why a capable digital agent stack does not necessarily translate into an end-to-end scientific discovery system. A critical cross-disciplinary synthesis is therefore needed to connect agent capabilities, engineering architectures, and environmental feedback.

Here, we organize this review around *three-stage capability externalization*, tracing how agent capabilities extend from model-internal cognition, through engineered execution, to world-grounded interaction. Reasoning externalization makes knowledge, memory, and multi-step inference observable and amenable to intervention, without establishing their validity. Execution externalization embeds tool use, state management, and control in testable harnesses and workflows, making digital work governable and recoverable. World-interaction externalization extends agent operation into scientific and physical environments, where progress must be grounded in observation, verification, and correction. These stages are overlapping analytical categories rather than a strict chronology. Together, they reveal a transfer of bottlenecks from unreliable reasoning, through poorly controlled execution, to limited access to trustworthy environmental feedback. We identify this final constraint as the *verifiability ceiling* and examine its implications through an L0–L4 ladder, plus a separate self-modification criterion, which maps the autonomy frontier across the materials discovery chain. By analyzing prediction, synthesis, characterization, property evaluation, and reproduction, we argue that verification infrastructure is the missing link between local automation and reproducible autonomous discovery. Figure 1 maps this framework and its central argument.

2 | Agent autonomy depends on externalized capabilities and environmental verification

The notion of agent spans philosophy and artificial intelligence. In AI, Russell and Norvig defined an agent as a computational entity that perceives its environment and acts within it [31], and Wooldridge and Jennings abstracted four core attributes of autonomy, reactivity, pro-activeness, and social ability [32]. This classical paradigm assumed that the decision mechanism is deterministic symbolic reasoning or a task-specific policy, which constrained cross-scenario generalization for over half a century. A glossary of frequently mixed terms is provided in Supporting Information, Note 6 (Table S11). Large language models broke this bottleneck. Pre-trained on massive corpora, LLMs exhibit general capabilities including instruction following, in-context learning, chain-of-thought reasoning, planning, and tool use [10, 16]. The brain of an LLM agent is no longer symbolic rules or a task-specific policy network but a pre-trained model with general knowledge that enables strong zero- and few-shot generalization, which gave rise to the widely cited formulation that an LLM agent consists of an LLM together with planning, memory, and tool use [33]. We treat this formulation not as a static architecture, but as a starting point from which capabilities are progressively externalized.

We use three-stage capability externalization as an analytical framework rather than a strict chronology. Reasoning externalization moves knowledge, intermediate inference, and state beyond model parameters into inspectable traces, retrieval systems, and memory. Execution externalization places tool use, workflow control, permissions, recovery, and termination in an independent engineering layer. World-interaction externalization incorporates environmental observations into hypothesis revision and action selection. Each stage addresses a prior reliability problem while exposing a new bottleneck. The stages are summarized in Supporting Information, Note 1 (Table S1).

Materials discovery makes this bottleneck transfer particularly clear. Agents can retrieve documented materials knowledge, generate hypotheses, query databases, invoke computational tools, and coordinate experimental workflows; yet completing these tasks does not by itself constitute scientific discovery [34]. The coupled synthesis–processing–structure–property relationships in materials research mean that relevant material states are only partially observable and that evidence is distributed across heterogeneous measurements. An agent must therefore assess whether new observations support or falsify a hypothesis, distinguish proposed causal mechanisms from plausible alternatives, and use the resulting evidence to select subsequent experiments. When verification is slow, costly, or difficult to reproduce, environmental verifiability becomes a central constraint on sustained autonomy: successful experimental execution alone does not establish a causal explanation or a reproducible scientific claim.

The limits of agent capability become particularly apparent when agents move from closed digital environments to open physical ones. GNoME reported 2.2 million candidate crystal structures predicted to be stable using graph neural networks, whereas independent experimental realization was reported for only 736 structures [35]. Berkeley's A-Lab combined literature-derived synthesis plans, robotic execution, and X-ray diffraction feedback, reporting a 63 percent success rate across 57 target compositions over 17 days. However, this achievement demonstrates autonomous synthesis within a predefined target set, while reliable automated phase identification remains a critical challenge [14, 36–38]. ResearchClawBench evaluates agents on 40 scientific rediscovery tasks and reports that even frontier models achieve an average score of just 26.5 out of 100, with failures concentrated in protocol mismatches and omissions of core scientific content [15]. Together, these cases reveal a recurring systems constraint. In coding and formal mathematics, tests and proof checkers can often provide rapid, repeatable feedback, supporting increasingly long-horizon autonomous operation. In materials discovery, relevant physical states are only partially observable, and verification frequently requires costly synthesis and characterization. Agents must therefore do more than invoke tools and interpret their outputs: they must establish whether the resulting evidence supports a scientific claim and justifies the next action. This difference in environmental verifiability motivates the analytical framework developed in the following sections. Extended analyses of these cases are provided in Supporting Information, Note 3.

3 | Reasoning externalization makes evidence and inference inspectable

Reasoning externalization is the first response to the reliability problem of LLM-based agents. LLMs can produce a plausible answer without exposing the evidence, state, or intermediate decisions that produced it. This makes errors difficult to diagnose and hallucinations hard to correct. The central move of this paradigm is therefore not simply to make models reason more: it is to turn black-box inference into a traceable and controllable process. Knowledge components provide inspectable evidence, memory systems retain relevant state, and reasoning modules expose alternatives, critiques, and revisions. Together, they make the cognitive process an engineering object rather than an unobservable model behavior. Figure 2 maps these architectures, their engineered knowledge and memory components, and their extension into layered materials agents. This shift is foundational for autonomy but is not a solution to autonomy in itself. An explicit reasoning trace can still contain an unsupported premise, retrieved evidence can still be irrelevant, and a self-critique can still repeat the original error. The value of externalization is that such failures can be located and acted upon.

Reasoning traces expose decisions without guaranteeing correctness. Early LLM reasoning methods made intermediate thought visible through natural-language chains. Their lasting contribution was not the particular prompt format, but the recognition that a model's answer could be decomposed into inspectable intermediate decisions [10]. ReAct made this insight operational for agents by interleaving a reasoning trace with actions and observations [11]. The resulting loop allows an agent to formulate a plan, query an external source, revise the plan, and expose the trajectory for inspection. It established a basic architecture for connecting deliberation to information gathering rather than treating generation as a one-shot response. However, exposing a trace does not guarantee that the trace is faithful or correct. A single mistaken premise can propagate through a chain, and a static prompt cannot reliably maintain state, select evidence, recover from a failed action, or determine when a task is complete. Sampling and search methods such as self-consistency and Tree of Thoughts improve robustness by comparing or exploring alternative trajectories [39, 40]; reflection methods add an explicit critique-and-revision loop [41]. These methods reveal the same limitation: without reliable external feedback, a model is not a dependable judge of its own reasoning [42]. The response was to engineer the inputs, state, and evaluation points around the model.

Knowledge, memory, and feedback make reasoning traceable and revisable. Knowledge externalization addresses the fact that model parameters are neither an auditable database nor an updatable scientific record. Retrieval-augmented generation separates evidence from generation, allowing a claim to be associated with retrievable sources rather than relying only on parametric recall [8]. Later systems make retrieval conditional and subject retrieved evidence to self-assessment or hierarchical organization [43, 44]. The important shift is epistemic as well as computational: evidence becomes a component that can be updated, inspected, and challenged. Retrieval reduces one source of hallucination, but it cannot ensure that retrieved information is relevant, complete, or correctly interpreted.

Memory externalization addresses a related limitation: a long-horizon agent cannot depend on a transient context window as its only state. MemGPT frames this problem as explicit movement between memory tiers [9], while subsequent systems couple stored observations with reflection and retrieval for future planning [45, 46]. In this formulation, memory is not merely a larger prompt. It records what the agent observed, what it inferred, and which prior decisions require reconsideration. This provides the continuity required for a multi-step investigation, while also creating an audit trail for the decisions that follow.

Reasoning modules complete the engineering layer by making alternatives and critiques explicit objects of search. Multiple candidate trajectories, program execution, and self-refinement provide ways to compare a proposed inference with other inferences or with deterministic computation [39–41]. Their common contribution is to convert an opaque generation into a process with identifiable failure points: missing evidence, stale memory, inconsistent paths, or an unsupported conclusion. Knowledge, memory, and reasoning modules therefore improve the observability, controllability, and revisability of agent cognition. However, they do not replace an external verifier, they prepare the agent to use one.

Materials agents connect scientific evidence to testable hypotheses. The same transition reshapes the role of computation in materials research. Materials discovery is often described through four overlapping paradigms: empirical trial-and-error, theory-guided inquiry, computational modeling and high-throughput simulation, and data-driven discovery [47]. The fourth paradigm expanded the prospects for inverse and on-demand design by learning structure–property and process–structure–property relationships from accumulated data [48]. Its effectiveness, however, depends on what has been measured, how consistently those measurements have been reported, and whether the learned relationships can guide new experimental decisions. LLM-based agents extend this paradigm by connecting materials theory, machine learning, and the distributed scientific record within a goal-directed reasoning workflow [49].

The immediate opportunity lies upstream of laboratory execution. Materials knowledge is dispersed across prose, tables, images, databases, and simulation outputs. ChemicalTagger and Mat2Vec demonstrated complementary ways to extract structured chemical information and latent scientific relationships from the literature [50, 51], while knowledge graphs such as MatKG organize these relationships for large-scale querying [52]. LLM-based agents can connect these resources within a sequence of scientific tasks: retrieving relevant evidence, interpreting it in context, reasoning about candidate structure–property relationships, and formulating testable hypotheses. Interpretable language-model predictions of materials properties and agentic materials-design systems illustrate this progression from knowledge extraction toward scientific reasoning and design [53–55].

The significance of this shift lies in how scientific evidence is assembled and used. Agents can shorten the path from scattered findings to a design proposal constrained by available evidence, while making the intermediate steps of cross-document synthesis available for inspection. Comparable patterns appear in chemistry, biology, and other data-rich sciences, where agents combine literature retrieval, domain tools, and iterative reasoning to formulate and prioritize hypotheses. These advances nevertheless support a bounded claim: they accelerate literature-grounded

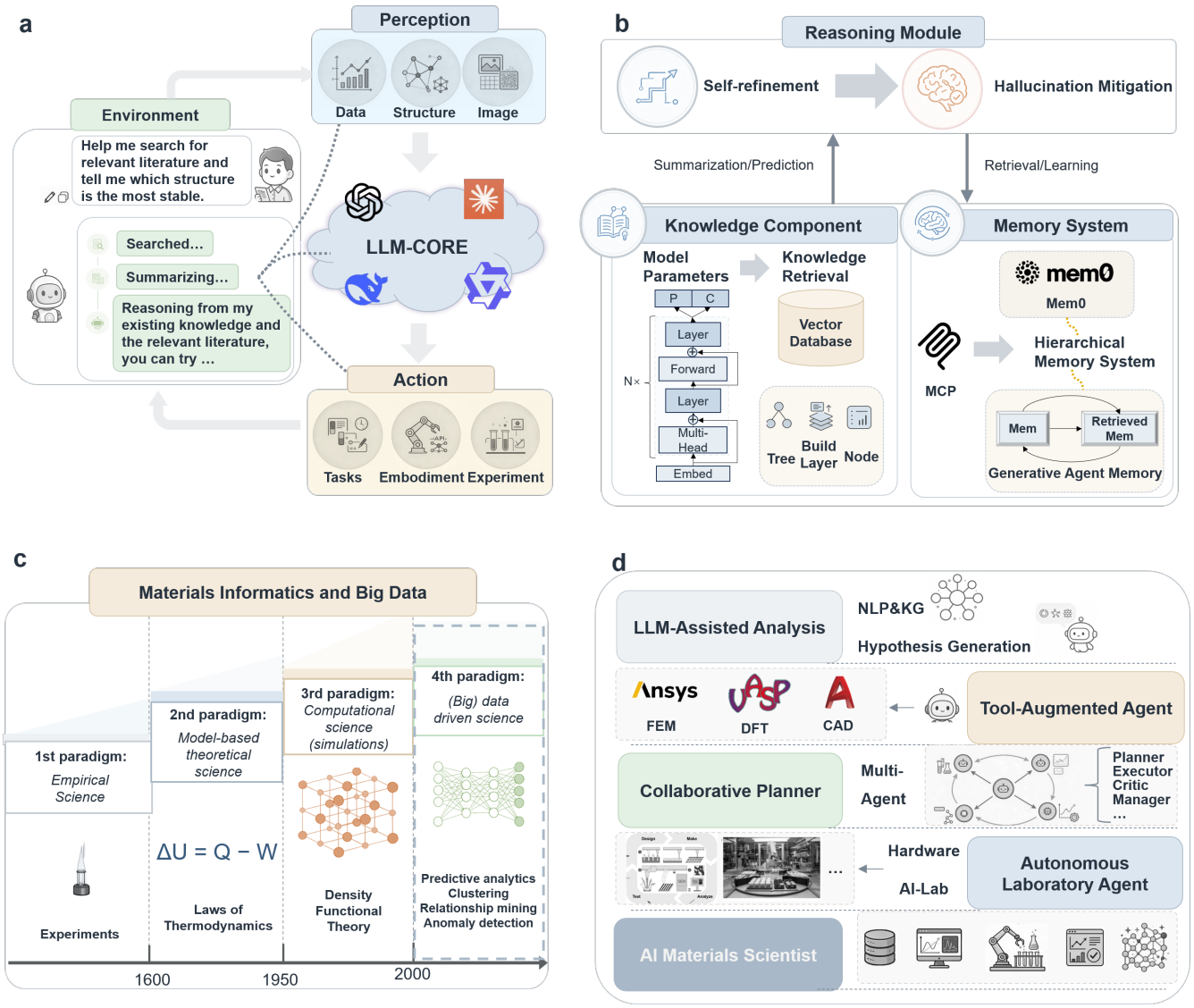


FIGURE 2 | Reasoning externalization, from the agent loop to layered materials agents. The figure decomposes reasoning externalization into four levels. **a**, the LLM-centered perception–action loop that interleaves environment, model core, and tool-mediated action [10, 11]. **b**, the engineered components that make cognition inspectable: a Reasoning Module supporting self-refinement [41, 42], a Knowledge Component that retrieves and organizes external evidence [8, 43, 44], and a Memory System that combines tiered memory [9], hierarchical and production memory such as mem0 [45, 46], and the Model Context Protocol as a common interface [13]. **c**, the four paradigms of materials informatics, from empirical and theory-guided science (thermodynamic balance $\Delta U = Q - W$) through computational simulation (density-functional theory and molecular dynamics) to data-driven science [47]. **d**, the progression of materials informatics towards a "fifth paradigm" driven by artificial intelligence [28], illustrated as a five-tier hierarchy of LLM-based materials agents. The first tier, *LLM-Assisted Analysis*, focuses on NLP, knowledge graphs (KG), and hypothesis generation, functioning primarily as a conversational interface. This is followed by the *Tool-Augmented Agent*, which integrates external execution tools such as FEM, DFT, and CAD (e.g., Ansys, VASP, AutoCAD) to ground reasoning in physical simulations. The third tier introduces a *Collaborative Planner*, enabling multi-agent workflows with defined engineering roles (Planner, Executor, Critic, Manager) [56, 57]. The fourth tier extends into the physical world through the *Autonomous Laboratory Agent*, leveraging hardware AI-Labs to conduct real-world experiments [14, 58]. Ultimately, these layers culminate in the prospective *AI Materials Scientist*, a full-chain system designed to autonomously manage the entire research lifecycle [59]. While these layers externalize cognition and render it inspectable, they do not inherently verify scientific validity, motivating the subsequent externalization of execution and world interaction.

reasoning and data-driven design, but do not by themselves establish an autonomous discovery loop. A generated hypothesis remains provisional until synthesis, characterization, property measurement, and independent reproduction provide the evidence needed to evaluate it. Reasoning externalization thus establishes important cognitive preconditions for autonomy. In materials research, it connects literature extraction with scientific interpretation, structure–property reasoning, and hypothesis generation. By exposing the evidence and intermediate inferences behind a proposal, it also makes disagreements easier to investigate: a researcher can trace a questionable design to its sources, retrieved context, reasoning steps, or applied constraints. The systems discussed in this section are catalogued in Table S6 by the capability each of them externalizes.

Its limits become apparent when a proposal must be carried through a sustained sequence of actions. The agent must still select and invoke tools, maintain execution state, detect failures, determine when recovery is needed, and assess whether the task's success criteria have been met. These are operational-control challenges whose consequences accumulate over long task horizons. An inspectable reasoning trace may help explain a failed action, but it does not itself provide the mechanisms needed to execute, monitor, or recover from that action. The next stage of capability externalization therefore places execution and control in a dedicated engineering layer, where tool interfaces, workflows, state management, and evaluation mechanisms can be tested and improved independently of the model.

4 | Execution externalization makes tools and workflows governable

Reasoning externalization makes evidence, memory, and intermediate inference available for inspection and revision, but does not by itself ensure reliable execution. When the model alone decides which tools to call, when to call them, how long to continue, and whether a task has succeeded, even a plausible plan can lead to redundant tool calls, goal drift, or premature termination. The bottleneck thus shifts from how the agent reasons to how its actions are governed throughout a task. We call the response to this challenge *execution externalization*: the transfer of execution and control from the model's moment-to-moment judgment to an explicit, independently inspectable engineering layer. This layer defines tool interfaces, manages state, enforces permission boundaries, represents workflows, tracks progress, and provides mechanisms for recovery and termination. The model continues to propose actions and revise plans, while the engineering layer governs how those actions are carried out. The aim is to transform a sequence of generated instructions into a coherent workflow that can be monitored, constrained, resumed, and evaluated.

This distinction is important for scientific agents. Access to more databases, simulators, instruments, and specialist models lets an agent do more things, but it does not tell the agent which action to take next, when to stop, or whether its output is reliable enough to count as a scientific finding. Table 1 collects representative systems as evidence for these developments. A complete catalogue of execution-externalization systems is given in Table S7.

Reliable tool use requires explicit interfaces and controls. Tool calling is the smallest unit of execution externalization. Toolformer showed that a language model can learn to select an API, decide when to invoke it, formulate arguments, and incorporate the result into subsequent reasoning [12]. This is a necessary capability, but it also reveals the governance problem: a model that can call a tool must still be constrained by what the tool accepts, what it is permitted to change, and what counts as a usable result. Increasing the number of accessible tools therefore enlarges the control problem; it does not automatically increase autonomous task completion. The engineering response is to make tools first-class operational objects. ToolLLM and Gorilla broaden the space of callable APIs, whereas MCP supplies a common interface through which tools can declare their capabilities and input-output contracts [13, 73, 74]. Active-discovery systems such as AnyTool, RAG-MCP, and MCP-Zero further address how an agent finds a relevant capability in a large tool space [63, 64, 75]. These systems move part of tool use from informal prompt behavior into explicit schemas, retrieval mechanisms, and runtime interfaces. The resulting action can be logged, validated against an interface, denied by a permission rule, or retried after a known failure. This separation matters especially in scientific computing. LLaMP, for example, combines Materials Project retrieval with atomistic simulation tools, while modular materials agents connect distinct computational tasks through specified interfaces [76, 77]. Tool access establishes an action space. Reliable autonomy requires the engineering layer to delimit and organize that space.

Stateful workflows coordinate actions and support recovery. Correctly calling one tool does not ensure that a task with many dependent actions will finish correctly. Long-horizon execution needs an explicit account of dependencies, intermediate state, branching conditions, failures, and termination. A workflow supplies this account: it converts an otherwise transient sequence of model decisions into a process that can be inspected, interrupted, resumed, and revised. Orchestration specifies who or what may act at each point, including an agent, a deterministic service, or a human reviewer. Early plan-execute and reflection methods separate deliberation from action and introduce checkpoints at which an agent can revise its course [66, 78]. Stateful workflow systems such as LangGraph make control flow and persistent state explicit, while AutoGen and MetaGPT use structured messages, roles, and shared artifacts to coordinate multiple agents [56, 57]. Magentic-One illustrates the same principle at the system level: an orchestrator tracks progress, dispatches specialized browser, file, and coding agents, and replans after failure [69]. Their common contribution is not simply that several agents collaborate, but that task progress ceases to exist only in the model's context window.

In scientific research, AutoDFT coordinates structure generation, job submission, error handling, and result analysis within a DFT workflow [68], while MechAgents organizes mechanical analysis through specialized roles and interaction protocols [79]. Ganjiang [6] extends this approach to powder X-ray diffraction analysis by coordinating phase identification, quantitative analysis, and whole-pattern fitting within an integrated workflow. These systems illustrate how execution externalization can make research workflows more structured, traceable, and repeatable at the operational level.

Task agents automate workflows toward predefined outcomes. Once tools and workflows are externalized, the relevant unit of evaluation becomes the task rather than the call. A task-level agent must deliver an artifact or state change within defined permissions, resources, and completion criteria. This is a stronger

TABLE 1 | Execution externalization as a reliability stack in materials discovery.

Execution layer	Externalized capability	Representative work	Role
Action interfaces	Tool schemas, contracts, permissions, and logs.	Toolformer[12]; MCP[60]; BFCL, τ-bench[61, 62]	Reliable database, simulation, and instrument access.
Capability discovery	Tool retrieval, skill selection, and reusable procedures.	AnyTool[63]; RAG-MCP[64]; Voyager[65]	Finds relevant research capabilities.
Stateful workflows	Plans, persistent state, checkpoints, and recovery.	ReWOO[66]; Sherlock[67]; AutoDFT[68]	Enables reproducible computational workflows.
Multi-agent orchestration	Roles, shared artifacts, scheduling, and coordination.	AutoGen[56]; Magentic-One[69]; A-Lab[14]	Coordinates computational and experimental resources.
Adaptive harnesses	Optimizable prompts, policies, memory, and agent code.	STOP[70]; AFlow[71]; Darwin Gödel Machine[72]	Improves research workflows.

Boundary. Execution reliability does not establish scientific validity, motivating world-interaction externalization.

claim than the ability to invoke tools: it requires the runtime to preserve state over time, detect failures, decide when escalation is required, and distinguish an intermediate result from a completed outcome. It nevertheless remains a bounded form of autonomy, because its success is judged against an operational target specified in advance. Task-level environments and their feedback strength are compared in Table S8.

Coding provides the clearest example because its execution boundary is comparatively well specified. Agents such as Devin, OpenHands, and Claude Code combine repository access, terminal execution, stateful runtimes, and test feedback to move from issue interpretation through code editing to validation and repair [80]. A build, test suite, or review requirement does not guarantee that every implementation decision is correct, but it provides a comparatively fast and repeatable criterion for whether a proposed change has met the task boundary. This strong feedback is a property of the coding environment as much as of the model or the harness. Web agents operate under looser boundaries. Systems and benchmarks such as WebArena, WebVoyager, SeeAct, and STEP frame browsing as a sequence of actions over visual and textual interfaces [25, 81–83]. Here completion depends on a changing external state, ambiguous interface semantics, and side effects such as submitting a form or making a purchase. The browser is therefore not merely a large callable tool: it is an environment in which the runtime must track credentials, permissions, page state, and the distinction between reversible navigation and consequential action. The shift from coding to web use shows why more powerful tool use alone cannot define safe termination or reliable completion.

Research agents expose the most consequential distinction between task completion and discovery. The AI Scientist can generate ideas, write and execute code, visualize results, compose papers, and simulate review in machine-learning settings [59, 84]. It shows that a sizable research *workflow* can be automated when experiments and evaluators reside in a digitally tractable environment. In materials science, a full research loop spans hypothesis formation, synthesis, characterization, structural interpretation, performance measurement, and reproduction. Computational tools, laboratory robots, and data services can each be incorporated into an engineering workflow [14, 28, 77]. But a completed procedure, a generated report, or a successful instrument call does not alone determine whether the observed result supports a hypothesis, whether hidden experimental variables have been controlled, or whether the finding will reproduce.

Harness self-improvement depends on reliable external evaluation. Once the harness is an independent layer it becomes an object of optimization. Self-improving-harness work searches over prompts, tool policies, memory rules, workflow graphs and executable agent code: STOP established the early self-referential loop, ADAS and AFlow automate the construction or search of procedures, and Self-Harness and Meta-Harness extend the target to the mechanisms deciding what is stored, retrieved and presented to the model [6, 70, 71, 85–87]. This is recursive self-improvement (RSI) in its narrow engineering sense: the agent edits a component of itself, and an external signal decides whether the edit survives. Recent systems widen the target from prompts to code and weights. Gödel Agent and SICA rewrite their implementation, SEAL and self-rewarding models their parameters, and AlphaEvolve and Absolute Zero replace the human-supplied task set with evaluator-driven search [88–93]. The Darwin Gödel Machine shows where the boundary lies. It retains changes to a coding agent's own code only when coding benchmarks improve [72], so its gain is defined by that benchmark and not by unrestricted improvement or scientific understanding. Self-improvement is therefore only as trustworthy as its evaluator, and the question shifts from whether an agent can change itself to whether the environment can judge the change.

The loop has also moved from harness to research strategy. AI Co-Scientist evolves hypotheses against literary and expert review [94], AlphaEvolve and MLEvolve evolve programs against executable evaluators [92, 95], and closed-loop laboratories evolve experimental selection against diffraction and property measurements [14, 48]. The editing mechanism is shared across these cases; the evaluator is not. A checker tolerates thousands of queries per search, whereas a synthesis campaign admits a handful of interventions per claim, which is what separates self-improvement in software from self-improvement in science.

Execution externalization transforms model-proposed actions into operations that can be constrained, monitored, recovered, and evaluated. Tool interfaces make individual actions governable; workflows expose dependencies and track progress; task-level systems align execution with intended outcomes; and self-improving harnesses allow the execution infrastructure itself to be refined. Together, these capabilities provide the engineering foundation for sustained, long-horizon operation. Whether such operations yield evidence that supports or challenges a scientific claim, however, depends on environmental feedback. The next stage therefore treats this feedback as a constraint on scientific inference and subsequent action.

5 | World-interaction externalization grounds hypothesis revision in environmental evidence

Figure 3 gives the loop an operational form: the decision structure that turns a workflow into a closed loop (Figure 3a) and a profile of what three representative environments can verify (Figure 3b). A workflow may complete, an instrument may return a measurement, and a report may be delivered, while the agent still cannot determine whether an observation supports its hypothesis or whether an unobserved variable invalidates the result. World-interaction externalization begins at this boundary. Its concern is not another class of callable tool, but the incorporation of environmental feedback into the agent's process of forming and revising beliefs. The central shift is from task closure to epistemic closure. The agent must formulate a hypothesis, select an informative action, obtain an observation, distinguish competing explanations, and decide whether further verification is required. This is necessarily an interaction with an environment that may be open, partially observable, delayed, and costly to probe. The degree to which the environment can verify these decisions determines how far sustained autonomy can extend. Figure 3b compares three environments along exactly these dimensions.

Environmental feedback determines how far agents can act autonomously. An agent interacts with the world whenever its actions change an external state and that state returns information relevant to later decisions. This includes digital environments such as code repositories and proof assistants, web environments with changing interfaces, and physical environments accessed through sensors, instruments, and robots. The distinction from Section 4 is functional: in execution externalization, the environment is primarily the target of a controlled procedure; in world-interaction externalization, it is also the source of evidence that can overturn the procedure's premise. This distinction explains why autonomy is uneven across domains. Compilers, test suites, and proof checkers offer fast, deterministic feedback. They make it possible to reject many erroneous actions cheaply and to continue search over long horizons. At the other end of the gradient, empirical environments contain hidden variables, observation noise, delayed effects, and irreversible or expensive interventions. The same model and harness therefore face fundamentally different prospects for reliable autonomy. The challenge is not only to act, but to learn what the action has established. The dimensions of environmental verifiability are detailed in Table S2; a world-interaction system catalogue is given in Table S9.

Scientific discovery requires evidence to test and revise hypotheses. Scientific discovery is an epistemic loop rather than a longer workflow. A workflow can execute the sequence hypothesis → experiment → result. An epistemic loop must additionally determine whether the result distinguishes the hypothesis from alternatives, whether the evidence is adequate, and how the next hypothesis or experiment should change. This difference separates an agent that automates research operations from one whose claims are constrained by the world. Formal research environments provide the clearest case because their feedback is unusually strong. AlphaGeometry and AlphaProof demonstrate how search can be coupled to formal or near-formal checking [96, 97]; FunSearch similarly uses executable evaluation to search for improved constructions [98]. These systems show how reliable feedback can support substantial autonomy. Their performance does not imply that all scientific environments offer the same conditions: the success of a proof checker rests on a specified formal language and a determinate validity criterion. Empirical research reverses these conditions. ResearchClawBench finds that current agents often fail through protocol mismatch, evidence mismatch, and missing scientific core rather than through a mere inability to call tools [15]. Work on intrinsically verifiable environments, such as AutoWebWorld, and on synthesized tasks and trajectories attempts to create cheaper settings for training and evaluation [3, 99–101]. Such infrastructure can strengthen feedback, but it does not eliminate the gap between a synthetic evaluator and a scientific environment whose relevant state is only partially observed.

Materials discovery requires interpretable and reproducible physical evidence. Materials discovery concentrates the difficulty of world grounding. The relevant state includes composition, structure, defects, processing history, and measurement conditions; characterization returns only indirect and often noisy evidence about that state. A prediction, a completed synthesis, or an XRD pattern is therefore not itself a verified discovery. The agent must relate those observations to alternative explanations, identify which variables remain uncontrolled, and decide whether replication or a discriminating follow-up experiment is needed.

Existing systems demonstrate important capabilities within specific research workflows. A-Lab integrates literature-derived synthesis plans, robotic execution, and XRD feedback [14]; ChemCrow combines planning with expert chemistry tools [4]; modular materials agents connect databases with computational solvers [77]; and Ganjiang integrates phase identification, quantitative analysis, and full-pattern fitting within a powder XRD analysis workflow [6]. These works show that individual segments of the loop can be automated. They also clarify the remaining gap: automating prediction, synthesis, or characterization does not by itself establish a causal interpretation or reproducible result. Therefore, materials agents research requires not only more tools, but also a verification infrastructure that links experimental observations back to claims about material behavior [28].

This is why materials discovery is a harder case than digitally evaluated research. GNoME can generate a large space of predicted stable structures, yet prediction must still be tested against synthesis and characterization in the physical world [35]. Each such test is slower, more expensive, and less complete than running a program or checking a proof. The bottleneck is consequently the strength of environmental verification, not the raw number of models, tools, or automated instruments available to the agent.

Embodied agents and world models support physical exploration. Embodied agents and world models offer complementary ways to strengthen this interaction. Vision-language-action systems connect high-level instruction to perception and physical action: SayCan uses affordance information to connect planning with execution, while RT-2 and OpenVLA learn transferable visual-action policies from large-scale data [102–104]. They supply an interface through which an agent can observe and affect a physical environment; they do not by themselves tell the agent what a resulting observation means scientifically. World models supply a second resource: an inexpensive, predictive surrogate in which the agent can anticipate action consequences, plan, and generate experience. Genie and related models demonstrate this prospect in interactive visual settings [105]. For materials science, an adequate world model would need to predict the consequences of untried processing choices, infer latent material state from

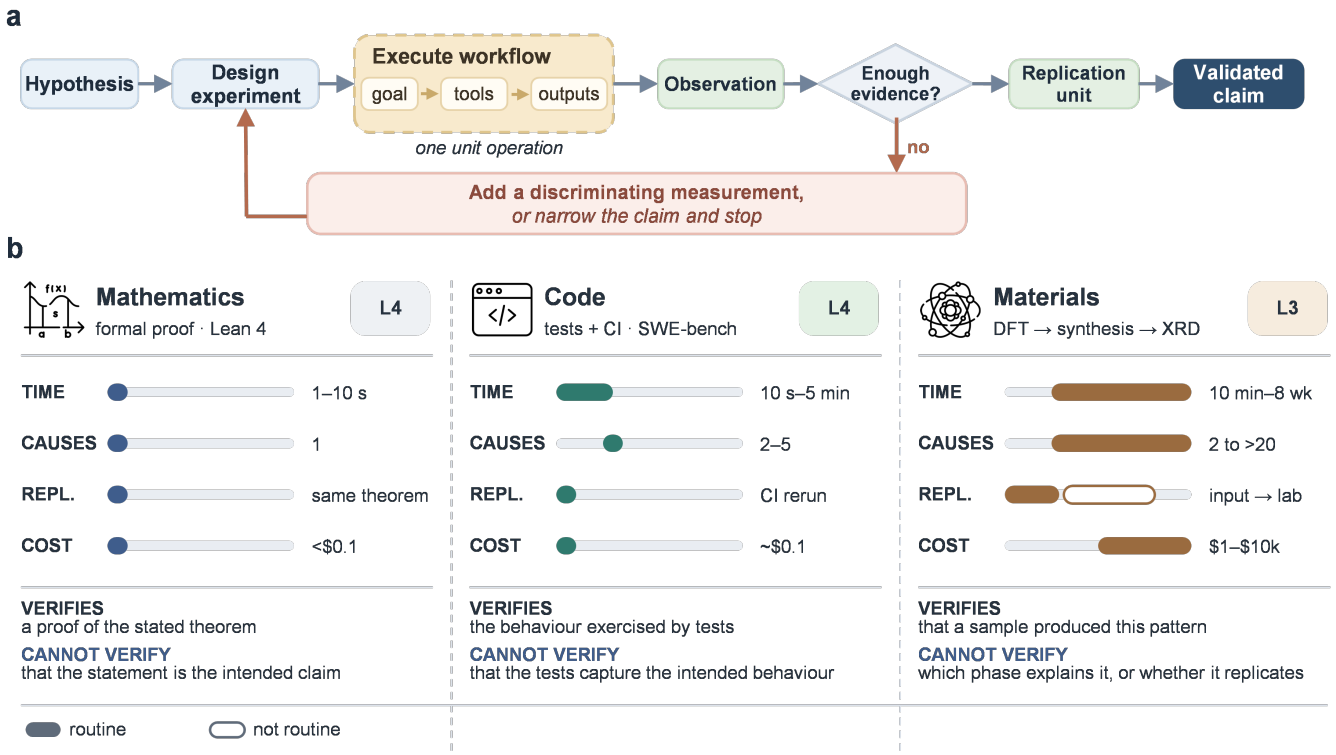


FIGURE 3 | **Environmental verifiability is a property of the task, not of the agent. a**, One iteration of a closed-loop investigation. A prescribed workflow (goal → tools → outputs) is the unit operation inside the loop, not an alternative to it: the loop wraps that operation in hypothesis formation, observation and a sufficiency test, and admits two exits, add a discriminating measurement, or narrow the claim and stop. A workflow that merely returns outputs therefore terminates in an observation, not in a validated claim, unless the sufficiency test is passed and the result survives a replication unit; the execution, observation, sufficiency-test and replication stages instantiate the four verification levels of Table S10. **b**, Three research environments, formal mathematics (Lean 4), code (tests + CI, SWE-bench) and materials (DFT → synthesis → XRD), profiled on the four dimensions of Table S2: time to evidence, attributability (the number of candidate explanations consistent with the observation), replication unit, and cost per evidence. Environments are ordered from strongest (left) to weakest (right) feedback; bar length encodes magnitude and the numerals give the range across the examples compiled in Table S12, with the filled and open segments of the REPL. row marking replication that is, respectively, routinely and not routinely performed. The badge gives the highest autonomy level demonstrated to date, a lower bound on what the environment can support. The three environments are exemplars chosen to span the range, not a sample. Each column separates what the environment verifies from what it cannot, which is why neither a stronger model nor a richer harness moves the badge.

characterization, quantify uncertainty, and direct the next experiment toward unresolved alternatives. Such a model would reduce the cost of exploration, but it could not replace validation against real synthesis and measurement. The boundary conditions of materials world models are discussed in Supporting Information, Note 4.

Reasoning externalization makes deliberation inspectable; execution externalization makes procedures governable; world-interaction externalization incorporates environmental feedback into the agent's decision-making process. However, the observability, attributability, repeatability, and affordability of that feedback set the practical ceiling on autonomy.

6 | The least verifiable link limits autonomous materials discovery

Task capability and operational autonomy are related but distinct. A system may perform a difficult task once, or execute a prescribed pipeline flawlessly, while still depending on humans to define its objective, authorize consequential actions, recover from exceptions, or decide whether an outcome is valid. Operational autonomy concerns the extent to which an agent can sustain goal-directed operation within those boundaries.

Operational autonomy and self-modification are scored on separate axes. L0–L4 form a single ordinal ladder of operational autonomy: each rung is defined by a further withdrawal of human decision authority, and a system at Ln has satisfied L($n-1$). Figure 4 applies this ladder to the chain: each row is scored at the highest level demonstrated by published systems, and levels are assigned against the rubric before the rows are ordered. L0 denotes human-only operation. At L1, an LLM assists a human who retains decision authority. At L2, the agent can invoke predefined tools, but a human or fixed workflow retains responsibility for objectives and consequential decisions. At L3, the agent can plan and execute multiple dependent steps with persistent state, while human oversight remains necessary at key checkpoints. At L4, an autonomous operator can sustain a design–execute–observe–recover loop in a bounded environment with only limited intervention. Separately, L5 denotes a self-modification criterion: the agent improves aspects of its own harness, policy, or weights against an external evaluator that remains stable across modifications, and retains the improvement across tasks. L5 is not a rung above L4; it can co-occur with any L0–L4 level and is scored on a

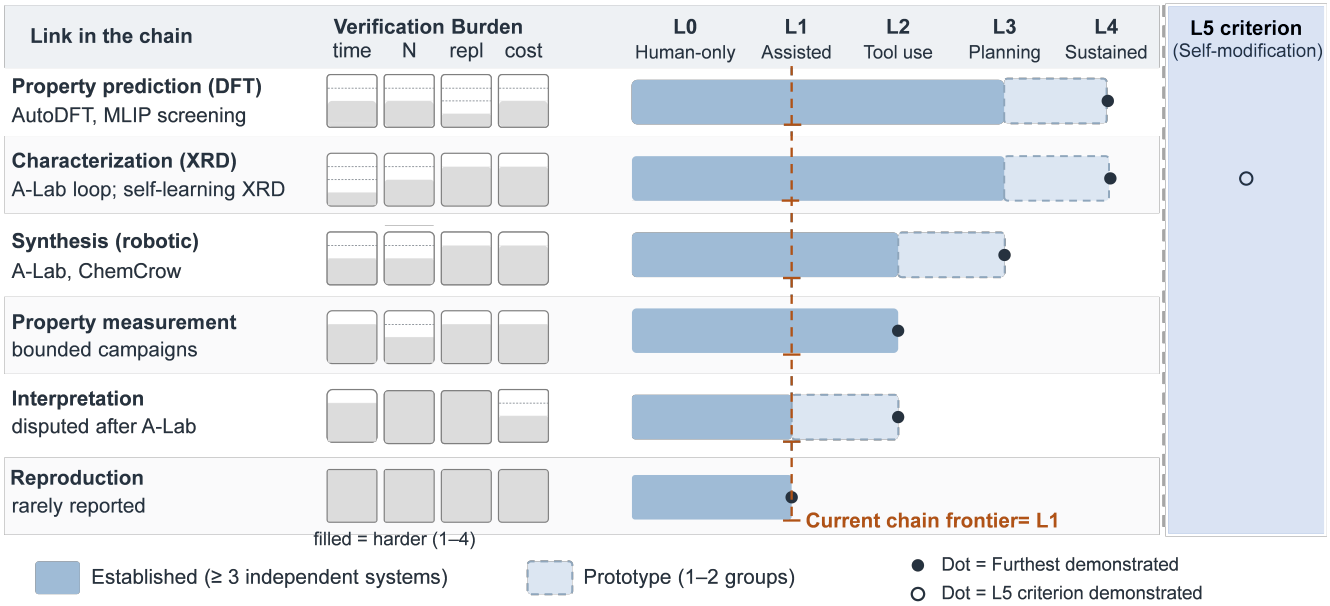


FIGURE 4 | **The autonomy frontier is set by the least verifiable link, not by the model.** Each row is one link of the materials discovery chain (prediction, synthesis, characterization, property measurement, interpretation, reproduction), ordered from most to least environmentally verifiable. The left block gives the four verification burdens—time to evidence, number of competing explanations (N), replication unit, and cost per evidence—on an ordinal 1–4 scale; higher fill indicates a harder burden, and the dimension with the highest fill is the binding constraint for that link. Solid blue bars mark levels demonstrated by at least three independent systems, including one third-party replication; pale blue bars with dashed outlines mark prototypes from one or two groups. The black dot at each bar end marks the furthest L0–L4 level demonstrated by any published system, named beneath the row label. The dashed orange line is the current chain frontier: the minimum over links of the level demonstrated by published systems. It sits at L1 because reproduction is demonstrated only at L1 and interpretation is established only at L1 (with an L2 prototype), while prediction and characterization individually reach L4. Links above the frontier identify where an end-to-end campaign cannot terminate; they do not weaken the claim. The rightmost column is a criterion column, not a sixth level: a hollow dot marks a link on which a published system satisfies the L5 self-modification criterion—the system modifies its own harness, policy, or weights against an evaluator it cannot alter, and retains the gain. The marked entry is Ganjiang on characterization[6]. L5 is system-level, may co-occur with any L0–L4 level, and is excluded from the chain frontier. Levels follow Table S3, evidence encoding follows Table S4, and the per-link scoring sheet is Table S13.

different axis. Definitions and scoring criteria are given in Tables S3 and S4.

The levels evaluate *operational autonomy*, not model intelligence or the breadth of tools. A model may display advanced reasoning in a single interaction yet remain at L1 if a human directs every consequential step. Conversely, a modest model may support L3 operation when a well-designed harness supplies state, tools, checkpoints, and clear termination criteria. This distinction also prevents a category error in scientific applications: completing a workflow belongs to task capability, whereas independently testing and revising a scientific claim is an L4-level operational property only when environmental feedback supports it. The three stages of externalization provide the prerequisites for this ascent without determining it mechanically. Reasoning externalization supports planning and revisability; execution externalization supports governable long-horizon operation; and world-interaction externalization supplies feedback that can constrain decisions in an environment. The autonomy achieved in a domain depends on the combination of these capabilities with its verification conditions.

Environmental verifiability explains autonomy gaps across domains. Coding is currently closest to L4 because tests, compilers, version control, and rollback provide unusually strong feedback for long-horizon repair and development tasks. Systems such as Devin, OpenHands, Claude Code, and SWE-agent demonstrate the required combination of planning, tool use, execution, and recovery [80, 106]. Web and computer-use agents have reached comparable multi-step planning in bounded interfaces, but changing page state, ambiguous task completion, and consequential side effects keep them largely at L3 [3, 25, 81, 107]. Per-domain assignments are given in Table S14. Research agents likewise show L2–L3 capabilities. The AI Scientist can organize a digital machine-learning research workflow, while materials agents and automated laboratories can execute selected computational or experimental segments [14, 59, 77]. They do not yet reliably achieve L4 discovery because scientific validity and reproducibility remain external, costly, and partially observable. Embodied agents are earlier still: RT-2 and OpenVLA demonstrate transferable perception-action policies, but the sim-to-real gap and physical safety costs make sustained autonomous operation exploratory [103, 104, 108].

These differences are explained principally by environmental verifiability, which determines whether a failure can be cheaply diagnosed, whether an agent can continue without supervision, and whether a local success can support a broader claim. L5 is no longer visionary in form, but it remains domain-local. Self-referential agents now rewrite their own prompts, control flow, tool policies and code, and keep a modification when an external evaluator confirms a gain: Gödel Agent and the Darwin Gödel Machine search over their own implementation [72, 88], SICA extends the loop to the agent's own codebase [89], and

SEAL, self-rewarding models, RISE and Absolute Zero move it inward to weights, reward models and self-proposed curricula [90, 91, 93, 109]. AlphaEvolve and MLEvolve close the same loop over program search whenever the evaluator is a fast, deterministic scorer [92, 95]. In materials research, L5 has been demonstrated at the level of analysis skills: Ganjiang revises and versions its XRD procedures against diffraction feedback while the base model stays frozen [6]. Adaptive experimental campaigns such as A-Lab revise which synthesis target to execute next against X-ray outcomes [14, 48], but this is adaptation of the campaign rather than modification of the agent, so it illustrates the boundary rather than satisfying L5. Neither system can independently determine whether the evidence is sufficient to distinguish between possible phases. The L5 systems discussed here improve their performance using a given evaluator, but they do not judge whether the evidence behind its feedback supports the scientific claim. This judgment remains a missing part of the materials-discovery workflow. If the evaluator is unreliable, or the system creates its own evaluator without independent checks, the system may appear to reach L5 while producing scientifically unreliable results. The safeguards in Table S5 are therefore necessary for L5 to be scientifically meaningful.

Autonomous materials research requires traceable workflows and verifiable evidence. Materials research is an unusually demanding execution environment. A scientific plan must cross heterogeneous resources: literature and materials databases, simulation codes, laboratory information systems, synthesis equipment, characterization instruments, and data-analysis pipelines. Each component has its own interface, state representation, unit conventions, failure modes, and permissions. A plan that is scientifically reasonable can therefore fail operationally because a sample is unavailable, an instrument is miscalibrated, a job terminates unexpectedly, or an experimental parameter is not recorded. The problem is not simply tool access, but preserving a traceable state across a chain of dependent physical and digital operations. Current systems illustrate how research workflows can be coordinated and progressively improved. AutoDFT integrates calculation setup, job submission, error handling, and result analysis, while modular materials agents connect databases and computational solvers through specialized components [68, 77]. Ganjiang extends workflow coordination to self-evolution in powder X-ray diffraction analysis: it uses execution traces and diffraction feedback to revise reusable skills, retaining validated procedural improvements in an external skill library without updating the underlying language model's parameters [6]. A-Lab connects literature-derived synthesis plans, robotic execution, and X-ray diffraction feedback within an automated experimental workflow [14]. Scaling these examples into sustained research operation requires interoperable tool interfaces, machine-readable experimental protocols, and provenance records that capture not only outputs but also sample histories, calibration conditions, failed runs, and human interventions. For self-evolving systems, these records must also track skill versions and the evidence supporting each revision, so that procedural improvements can be evaluated, reproduced, and rolled back when necessary.

Cost and safety are part of this engineering problem rather than secondary deployment concerns. Re-running a simulation is often inexpensive; repeating a synthesis or characterization may consume scarce samples, instrument time, reagents, and operator attention. Some actions are irreversible or hazardous, so agents need explicit authorization boundaries, resource budgets, and escalation rules before operating equipment. Reliability in agentic materials science consequently means more than completing a workflow: it means completing it within documented experimental, resource, and safety constraints.

The fundamental limitation is whether a materials research environment can turn an action into trustworthy evidence, which requires four forms of verification. *Action verification* asks whether a procedure or instrument ran as intended. *Measurement verification* asks whether the signal reliably represents the material state. *Interpretive verification* asks whether the observation supports one explanation over plausible alternatives. *Reproducibility verification* asks whether the claim survives changes in sample, batch, laboratory, and operating conditions. Each level is necessary but not sufficient for the next. The four levels map onto the checkpoints of Figure 3a: execution, observation, the sufficiency test, and the replication unit, and Table 2 makes the protocol operational for the powder/XRD example.

The prediction–synthesis–characterization–interpretation–property–reproduction chain is therefore constrained by its least verifiable link. A predicted structure may be thermodynamically stable within a given theoretical framework yet remain inaccessible through available synthesis routes; a successful synthesis may contain an unintended phase; and a measured property may reflect defects or processing conditions rather than the proposed structure. A completed experiment, an instrument readout, or an automated report is not by itself a scientific discovery.

Three interconnected forms of infrastructure define the frontier of long-horizon autonomy. The first is *verification infrastructure*, encompassing standardized, machine-readable protocols, provenance-aware experimental records, cross-modal characterization, automated anomaly detection, and evaluation that explicitly tests reproducibility. Its purpose is to make evidence accessible to agents in forms that support hypothesis testing and causal interpretation, with sufficient context to assess how observations were obtained and what they establish. The second is *materials world models and digital twins*. As discussed in Supporting Information, Note 4, these systems should model process–state–property relationships and quantify predictive uncertainty to guide the selection of informative experiments. Their value lies in reducing the cost of exploration and identifying where additional measurements are needed, while their predictions remain subject to validation against physical observations. The third is *guarded self-improvement*. Agents can refine retrieval policies, experiment-selection strategies, workflow graphs, and potentially the harness code that governs execution. Ganjiang provides a concrete example at the level of reusable analysis skills, using diffraction feedback to revise procedures and evaluating frozen skill versions on held-out samples [6]. Extending such improvement across materials research requires fixed evaluation protocols, versioned modifications, simulation-based testing where appropriate, independent validation, and clearly defined boundaries for human oversight. These safeguards make it possible to distinguish transferable improvements in scientific reliability from gains confined to convenient tasks or proxy metrics.

Together, these infrastructures should make scientific claims easier to test: measurements should show what is happening in a material, experiments should help distinguish between possible

TABLE 2 | Claim-specific verification protocol for an inorganic powder/XRD task.

Level	Evidence and acceptance record	Unresolved result and permitted agent action
Action	Furnace log, precursor lot, atmosphere and heating profile; all within pre-registered limits	Quarantine the batch; log the deviation; re-run
Measurement	Raw XRD, calibration/QC record and detection limit; pattern passes QC	Recalibrate or remeasure; do not interpret
Interpretive	Candidate-phase enumeration and discriminating reflections; assignment excludes alternatives	Add complementary characterization; withhold the phase-purity claim
Reproducibility	Second batch from a different precursor lot, with independent-lab record; discriminating reflections reproduced	Repeat; narrow the claim; report the result as unresolved

Note. Thresholds are pre-registered for the material, instrument, and claim.

explanations, and repeated tests should establish whether findings are reliable enough to guide further research. Autonomous materials discovery therefore requires systems that can turn actions into clear, reproducible evidence and use that evidence to update hypotheses and improve research procedures. Building such systems is a central challenge for AI agents in materials science and essential for sustained autonomy.

7 | Discussion

The central message of this review is that moving from task execution to scientific discovery requires agents to judge what their results actually establish. Human scientific practice offers a useful comparison. Researchers use existing knowledge to form explanations, but they also need ways to check and revise those explanations. They write down assumptions, use instruments to examine what they cannot directly observe, and design experiments to distinguish between possible causes. Their judgments can be wrong. Calibration, independent measurements, and replication help identify errors that reasoning alone may leave unnoticed.

The three forms of capability externalization described here can be understood from this perspective. Reasoning externalization makes knowledge, assumptions, and intermediate steps available for inspection and revision, much as notes and calculations support human thinking. Execution externalization connects a plan to tools and procedures that can carry it out reliably. World-interaction externalization brings observations back into the process so that an agent can reconsider its explanation and choose its next action. These functions overlap and support one another. Together, they show why an agent's scientific capability depends on the larger system through which it reasons, acts, and obtains evidence. Materials discovery makes the remaining difficulty especially clear. A synthesis can run as planned, and a diffraction measurement can produce a usable pattern, while the identity or purity of the product remains uncertain. Several explanations may still fit the observations. A researcher must then consider which explanations remain plausible and what additional measurement could distinguish them. For an agent to make this decision independently, it needs access to the relevant evidence, an account of how that evidence was obtained, and methods for assessing its limits. This is the practical meaning of environmental verifiability: whether the available observations can support the scientific decision the agent needs to make.

Verification is therefore a shared challenge for models, workflows, and experimental infrastructure. Better models may identify overlooked explanations, interpret existing data more effectively, or propose more informative experiments. Better instruments and protocols may provide evidence that was previously unavailable. The resulting limits on autonomy can change as these capabilities improve. World models can help select promising experiments, while independent physical measurements remain necessary to assess whether their predictions hold. Progress should consequently be evaluated by how agents handle both successful and inconclusive experiments. Useful measures include whether an agent detects conflicting evidence, requests an informative follow-up measurement, revises an unsupported claim, and produces results that survive independent replication. The same principle applies to self-improvement: a higher evaluation score matters only if it reflects more reliable scientific work. Physical verification connects the agent's reasoning to what the material actually does. The key step from execution to discovery is the ability to determine what the evidence supports, recognize what remains uncertain, and decide what must be tested next.

AUTHOR CONTRIBUTIONS

Yixuan Zhang: investigation, writing – original draft, writing – review, visualization. **Bin Cao**: conceptualization, methodology, writing – review and editing, project administration, supervision. **Tong-Yi Zhang**: conceptualization, funding acquisition, writing – review and editing, project administration, supervision, resources, methodology.

CONFLICT OF INTEREST

The authors declare no competing financial or commercial interests.

DATA AVAILABILITY STATEMENT

This review does not report primary experimental data. All systems, benchmarks, and case studies discussed are documented in the cited literature. The Supporting Information (Tables S12–S14) records the provenance and the evidential basis of every numerical estimate used in Figures 3 and 4, so that each value can be traced to its source or flagged as a stated assumption.

References

1. Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. https://doi.org/10.48550/arXiv.2501.12948.
2. Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025. https://doi.org/10.48550/arXiv.2507.20534.

3. Yifan Wu, Yiran Peng, Yiyu Chen, Jianhao Ruan, Zijie Zhuang, Cheng Yang, Jiayi Zhang, Man Chen, Yenchi Tseng, Zhaoyang Yu, et al. Autowebworld: Synthesizing infinite verifiable web environments via finite state machines. *arXiv preprint arXiv:2602.14296*, 2026. https://doi.org/10.48550/arXiv.2602.14296.
4. 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(5):525–535, 2024. https://doi.org/10.1038/s42256-024-00832-8.
5. Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan J Prenger, and Animashree Anandkumar. Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems*, 36:21573–21612, 2023. https://doi.org/10.52202/075280-0944.
6. Bin Cao, Huichi Zhou, Runyu Yang, Jingsong Li, Shuchen Sun, Yan Song, Hanyu Gao, Zhongwei Yu, Tong-Yi Zhang, and Jun Wang. A self-learning scientific agent for x-ray diffraction. *arXiv preprint arXiv:2610.07862*, 2026. https://doi.org/10.48550/arXiv.2610.07862.
7. Garyk Brixi, Matthew G Durrant, Jerome Ku, Mohsen Naghipourfar, Michael Poli, Gwanggyu Sun, Greg Brockman, Daniel Chang, Alison Fanton, Gabriel A Gonzalez, et al. Genome modelling and design across all domains of life with evo 2. *Nature*, 652(8112):1349–1361, 2026. https://doi.org/10.1038/s41586-026-10176-5.
8. Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.
9. Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. https://doi.org/10.48550/arXiv.2310.08560.
10. Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. https://doi.org/10.52202/068431-1800.
11. Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022. https://doi.org/10.48550/arXiv.2210.03629.
12. 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. *Advances in neural information processing systems*, 36:68539–68551, 2023. https://doi.org/10.52202/075280-2997.
13. Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model context protocol (mcp): Landscape, security threats, and future research directions. *ACM Transactions on Software Engineering and Methodology*, 35(10):1–37, 2026. https://doi.org/10.1145/3796519.
14. Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of inorganic materials. *Nature*, 624(7990):86–91, 2023. https://doi.org/10.1038/s41586-023-06734-w.
15. Wanghan Xu, Shuo Li, Tianlin Ye, Qinglong Cao, Yixin Chen, Hengjian Gao, Yiheng Wang, Qi Li, Kun Li, Sheng Xu, et al. Researchclawbench: A benchmark for end-to-end autonomous scientific research. *arXiv preprint arXiv:2606.07591*, 2026. https://doi.org/10.48550/arXiv.2606.07591.
16. Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
17. The OpenJudge Team. Pawbench: A benchmark for evaluating llm × harness performance, 2026. https://github.com/agentscope-ai/PawBench.
18. Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of computer science*, 18(6):186345, 2024. https://doi.org/10.1007/s11704-024-40231-1.
19. Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China information sciences*, 68 (2):121101, 2025. https://doi.org/10.1007/s11432-024-4222-0.
20. Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*, 2024. https://doi.org/10.48550/arXiv.2402.01680.
21. Guanting Dong, Xiaoshuai Song, Yuyang Hu, Jiajie Jin, Chenghao Zhang, Yifei Chen, Xiaoxi Li, Huaying Yuan, Xinyu Yang, Tongyu Wen, et al. Towards long-horizon agents: A survey. 2026.
22. Chaoyue He, Xin Zhou, Di Wang, Hong Xu, Wei Liu, and Chunyan Miao. Harness engineering for language agents: The harness layer as control, agency, and runtime. 2026. https://doi.org/10.20944/preprints202603.1756.v1.
23. Qianyu Meng, Yanan Wang, Liyi Chen, Yihang Li, Wei Wu, Wenyuan Jiang, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, et al. Agent harness for large language model agents: A survey. 2026. https://doi.org/10.20944/preprints202604.0428.v1.
24. Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pages 52989–53046, 2024.
25. Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pages 15585–15606, 2024.
26. Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pages 54107–54157, 2024.
27. Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, et al. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, volume 2025, pages 96934–96990, 2025.
28. Jiayi Zhu, Longhan Zhang, Yizhang Zhu, Xiaotian Lin, Yifan Wu, Shimin Di, Bang Liu, Yuyu Luo, and Tongyi Zhang. A survey of agentic materials science and engineering: where are we and where are we going? *Journal of Materials Informatics*, 6(2), 2026. https://doi.org/10.20517/jmi.2026.07.
29. Keith T Butler, Daniel W Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, 2018. https://doi.org/10.1038/s41586-018-0337-2.

30. Kamal Choudhary, Brian DeCost, Chi Chen, Anubhav Jain, Francesca Tavazza, Ryan Cohn, Cheol Woo Park, Alok Choudhary, Ankit Agrawal, Simon JL Billinge, et al. Recent advances and applications of deep learning methods in materials science. *npj Computational Materials*, 8(1):59, 2022. https://doi.org/10.1038/s41524-022-00734-6.
31. Stuart Jonathan Russell, Peter Norvig, John F Canny, Jitendra M Malik, and Douglas D Edwards. *Artificial intelligence: a modern approach*, volume 2. Prentice hall Englewood Cliffs, NJ, 1995.
32. Michael Wooldridge and Nicholas R Jennings. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152, 1995. https://doi.org/10.1017/s0269888900008122.
33. Lilian Weng. Llm-powered autonomous agents. *lilianweng.github.io*, Jun 2023. https://lilianweng.github.io/posts/2023-06-23-agent/.
34. Bin Cao. *Physics-Constrained Learning of Crystal Structures and Properties from Powder Diffraction*. PhD thesis, PhD thesis, The Hong Kong University of Science and Technology, 2026.
35. Amil Merchant, Simon Batzner, Samuel S Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, 2023. https://doi.org/10.1038/s41586-023-06735-9.
36. Bin Cao, Zinan Zheng, Yang Liu, Longhan Zhang, Lawrence WY Wong, Lu-Tao Weng, Jia Li, Haoxiang Li, and Tong-Yi Zhang. Xqueryer: an intelligent crystal structure identifier for powder x-ray diffraction. *National Science Review*, 12 (12):nwaf421, 2025. https://doi.org/10.1093/nsr/nwaf421.
37. Hanyu Gao, Bin Cao, Yunyue Su, Tong-Yi Zhang, and Qiang Liu. Xdecomposer: Learning prior-free set decomposition for multiphase x-ray diffraction. *arXiv preprint arXiv:2605.05866*, 2026. https://doi.org/10.48550/arXiv.2605.05866.
38. Bin Cao, Qian Zhang, Zhenjie Feng, Taolue Zhang, Jiaqiang Huang, Lu-Tao Weng, and Tong-Yi Zhang. Ai-driven structure refinement of x-ray diffraction. *arXiv preprint arXiv:2602.16372*, 2026. https://doi.org/10.48550/arXiv.2602.16372.
39. Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022. https://doi.org/10.48550/arXiv.2203.11171.
40. Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023. https://doi.org/10.52202/075280-0517.
41. Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023. https://doi.org/10.52202/075280-2019.
42. Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *International conference on learning representations*, volume 2024, pages 32808–32824, 2024.
43. Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pages 9112–9141, 2024.
44. Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, volume 2024, pages 32628–32649, 2024.
45. Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22, 2023. https://doi.org/10.1145/3586183.3606763.
46. Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025. https://doi.org/10.48550/arXiv.2504.19413.
47. Ankit Agrawal and Alok Choudhary. Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science. *APL materials*, 4(5), 2016. https://doi.org/10.1063/1.4946894.
48. Bin Cao, Jie Xiong, Jiaxuan Ma, Yuan Tian, Yirui Hu, Mengwei He, Longhan Zhang, Jiayu Wang, Jian Hui, Li Liu, et al. Bgolearn: a unified bayesian optimization framework for accelerating materials discovery. *npj Computational Materials*, 2026. https://doi.org/10.1038/s41524-026-02226-3.
49. Zhongwei Yu, Sourabh Roy, Bin Cao, Xue Yan, Anjie Liu, and Jun Wang. Harnessing large language models to compile task-relevant context into bayesian optimisation. *arXiv preprint arXiv:2609.36788*, 2026. https://doi.org/10.48550/arXiv.2609.36788.
50. Lezan Hawizy, David M Jessop, Nico Adams, and Peter Murray-Rust. Chemicaltagger: A tool for semantic text-mining in chemistry. *Journal of cheminformatics*, 3(1):17, 2011. https://doi.org/10.1186/1758-2946-3-17.
51. Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga Kononova, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature*, 571(7763):95–98, 2019. https://doi.org/10.1038/s41586-019-1335-8.
52. Vineeth Venugopal and Elsa Olivetti. Matkg: An autonomously generated knowledge graph in material science. *Scientific Data*, 11(1):217, 2024. https://doi.org/10.1038/s41597-024-03039-z.
53. Vadim Korolev and Pavel Protsenko. Accurate, interpretable predictions of materials properties within transformer language models. *Patterns*, 4(10), 2023. https://doi.org/10.1016/j.patter.2023.100803.
54. Markus J Buehler. Generative retrieval-augmented ontologic graph and multiagent strategies for interpretive large language model-based materials design. *ACS Engineering Au*, 4(2):241, 2024. https://doi.org/10.1021/acsengineeringau.3c00058.
55. Shuyi Jia, Chao Zhang, and Victor Fung. Llmatdesign: Autonomous materials discovery with large language models. *arXiv preprint arXiv:2406.13163*, 2024. https://doi.org/10.48550/arXiv.2406.13163.
56. Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023. https://doi.org/10.48550/arXiv.2308.08155.
57. Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Steven Yau, Zijuan Lin, Liyang Zhou, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations*, volume 2024, pages 23247–23275, 2024.
58. Zhen Zhang, Zhichu Ren, Chia-Wei Hsu, Weibin Chen, Zhang-Wei Hong, Chi-Feng Lee, Aubrey Penn, Hongbin Xu, Daniel J Zheng, Shuhan Miao, et al. A multimodal robotic platform for multi-element electrocatalyst discovery. *Nature*, 647(8089): 390–396, 2025. https://doi.org/10.1038/s41586-025-09640-5.
59. Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint*

arXiv:2408.06292, 2024. https://doi.org/10.48550/arXiv.2408.06292.

60. Model Context Protocol. Model Context Protocol specification, 2026. https://modelcontextprotocol.io/specification/2026-07-28 (accessed: September 2026).
61. Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.
62. Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. taubench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. https://doi.org/10.48550/arXiv.2406.12045.
63. Yu Du, Fangyun Wei, and Hongyang Zhang. Anytool: Self-reflective, hierarchical agents for large-scale api calls. *arXiv preprint arXiv:2402.04253*, 2024. https://doi.org/10.48550/arXiv.2402.04253.
64. Tiantian Gan and Qiyao Sun. Rag-mcp: Mitigating prompt bloat in llm tool selection via retrieval-augmented generation. *arXiv preprint arXiv:2505.03275*, 2025. https://doi.org/10.48550/arXiv.2505.03275.
65. 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. *arXiv preprint arXiv:2305.16291*, 2023. https://doi.org/10.48550/arXiv.2305.16291.
66. Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. Rewoo: Decoupling reasoning from observations for efficient augmented language models. *arXiv preprint arXiv:2305.18323*, 2023. https://doi.org/10.48550/arXiv.2305.18323.
67. Yeonju Ro, Haoran Qiu, Íñigo Goiri, Rodrigo Fonseca, Ricardo Bianchini, Aditya Akella, Zhangyang Wang, Mattan Erez, and Esha Choukse. Sherlock: reliable and efficient agentic workflow execution. *arXiv preprint arXiv:2511.00330*, 2025. https://doi.org/10.48550/arXiv.2511.00330.
68. Penghui Yang, Zhonghan Zhang, Yue Li, Xinrun Wang, Yanchen Deng, Yuhao Lu, Bijun Tang, Zheng Liu, and Bo An. Autodft: A closed-loop multi-agent framework for autonomous dft calculations. *arXiv preprint arXiv:2605.26179*, 2026. https://doi.org/10.48550/arXiv.2605.26179.
69. Adam Fourney, Gagan Bansal, Hussein Mozannar, Cheng Tan, Eduardo Salinas, Friederike Niedtner, Grace Proebsting, Griffin Bassman, Jack Gerrits, Jacob Alber, et al. Magentic-one: A generalist multi-agent system for solving complex tasks. *arXiv preprint arXiv:2411.04468*, 2024. https://doi.org/10.48550/arXiv.2411.04468.
70. Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023. https://doi.org/10.48550/arXiv.2310.02304.
71. Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pages 34040–34077, 2025.
72. Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pages 104223–104294, 2026.
73. Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, volume 2024, pages 9695–9717, 2024.
74. Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024. https://doi.org/10.52202/079017-4020.
75. Xiang Fei, Xiawu Zheng, and Hao Feng. Mcp-zero: Active tool discovery for autonomous llm agents. *arXiv preprint arXiv:2506.01056*, 2025. https://doi.org/10.48550/arXiv.2506.01056.
76. Yuan Chiang, Elvis Hsieh, Chia-Hong Chou, and Janosh Riebesell. Llamp: Large language model made powerful for high-fidelity materials knowledge retrieval. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 25200–25232, 2025. https://doi.org/10.18653/v1/2025.emnlp-main.1280.
77. Akshat Chaudhari, Janghoon Ock, and Amir Barati Farimani. Modular large language model agents for multi-task computational materials science. *Communications Materials*, 7(1):131, 2026. https://doi.org/10.1038/s43246-025-00994-x.
78. Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 2609–2634, 2023. https://doi.org/10.18653/v1/2023.acl-long.147.
79. Bo Ni and Markus J Buehler. Mechagents: Large language model multi-agent collaborations can solve mechanics problems, generate new data, and integrate knowledge. *Extreme Mechanics Letters*, 67:102131, 2024. https://doi.org/10.1016/j.eml.2024.102131.
80. Xingyao Wang, Boxuan Li, Yufan Song, Frank F Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. Openhands: An open platform for ai software developers as generalist agents. In *International Conference on Learning Representations*, volume 2025, pages 65882–65919, 2025.
81. Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6864–6890, 2024. https://doi.org/10.18653/v1/2024.acl-long.371.
82. Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024. https://doi.org/10.48550/arXiv.2401.01614.
83. Tianxing Zhou, Zhirui Wang, Haojia Ao, Guangyan Chen, Boyang Xing, Jingwen Cheng, Yi Yang, and Yufeng Yue. Step planner: Constructing cross-hierarchical subgoal tree as an embodied long-horizon task planner. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16731–16738. IEEE, 2025. https://doi.org/10.1109/iros60139.2025.11246104.
84. 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. https://doi.org/10.1038/s41586-026-10265-5.
85. Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pages 21344–21377, 2025.
86. Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-harness: Harnesses that improve themselves. *arXiv preprint arXiv:2606.09498*, 2026. https://doi.org/10.48550/arXiv.2606.09498.
87. Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026. https://doi.org/10.48550/arXiv.2603.28052.

88. 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, 2025. https://doi.org/10.18653/v1/2025.acl-long.1354.
89. Maxime Robeyns, Martin Szummer, and Laurence Aitchison. A self-improving coding agent. *arXiv preprint arXiv:2504.15228*, 2025. https://doi.org/10.48550/arXiv.2504.15228.
90. Adam Zweiger, Jyo Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. *Advances in Neural Information Processing Systems*, 38:74084–74115, 2026. https://doi.org/10.52202/085713-2483.
91. Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024. https://doi.org/10.48550/arXiv.2401.10020.
92. Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025. https://doi.org/10.48550/arXiv.2506.13131.
93. Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *Advances in Neural Information Processing Systems*, 38:105816–105879, 2026. https://doi.org/10.52202/085713-3534.
94. Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, et al. Accelerating scientific discovery with co-scientist. *Nature*, pages 1–3, 2026.
95. Shangheng Du, Xiangchao Yan, Jinxin Shi, Zongsheng Cao, Shiyang Feng, Zichen Liang, Boyuan Sun, Tianshuo Peng, Yifan Zhou, Xin Li, et al. Mlevolve: A self-evolving framework for automated machine learning algorithm discovery. *arXiv preprint arXiv:2606.06473*, 2026. https://doi.org/10.48550/arXiv.2606.06473.
96. Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024. https://doi.org/10.1038/s41586-023-06747-5.
97. AlphaProof Team and AlphaGeometry Team. Ai achieves silver-medal standard solving international mathematical olympiad problems. Google DeepMind Blog, 2024.
98. Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995): 468–475, 2024. https://doi.org/10.1038/s41586-023-06924-6.
99. Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, Minghao Liu, Yuchen Jiang, et al. Taskcraft: Automated generation of agentic tasks. In *International Conference on Learning Representations*, volume 2026, pages 43714–43734, 2026.
100. Zhengwei Tao, Jialong Wu, Wenbiao Yin, Pu Wu, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, et al. Webshaper: Agentically data synthesizing via information-seeking formalization. In *International Conference on Learning Representations*, volume 2026, pages 101872–101889, 2026.
101. Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. Toucan: Synthesizing 1.5 m tool-agentic data from real-world mcp environments. *arXiv preprint arXiv:2510.01179*, 2025. https://doi.org/10.48550/arXiv.2510.01179.
102. Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pages 287–318. Pmlr, 2023.
103. Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. https://doi.org/10.48550/arXiv.2307.15818.
104. Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. https://doi.org/10.48550/arXiv.2406.09246.
105. Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.
106. John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024. https://doi.org/10.52202/079017-1601.
107. Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024. https://doi.org/10.52202/079017-1650.
108. Zhonghao Zhan, Yefan Zhang, Krinos Li, and Hamed Haddadi. Simverity: When does simulated agent success survive physical deployment? *arXiv preprint arXiv:2608.25067*, 2026. https://doi.org/10.48550/arXiv.2608.25067.
109. Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. *Advances in Neural Information Processing Systems*, 37:55249–55285, 2024. https://doi.org/10.52202/079017-1754.

SUPPORTING INFORMATION

Supporting Information accompanies this manuscript as a separate document, containing Notes 1–8 and Tables S1–S15 (including Table S3b).