SCIPRODBENCH: EVALUATING AND IMPROVING SCIENTIFIC RESEARCH BY AI AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

AI agents can now carry out substantial parts of scientific research, from surveying the literature to running experiments and reporting conclusions. Whether those conclusions rest on synthesized evidence and correct computation, however, remains hard to assess, and automated research can improve only on what it can verify. We introduce SCIPRODBENCH, a benchmark of 184 research tasks in Biology, Chemistry, Computer Science, and Physics that span the research cycle, from problem formulation to experiment design, analysis, and reporting. Each task pairs a research objective with required deliverables and a task-specific environment. Report rubrics assess evidence synthesis and scientific reasoning, and for tasks that require computation, hidden executable tests check the underlying artifacts. Tasks and their verifiers are derived from the same graph of scientific evidence, so that every requirement and scoring criterion can be traced to its scientific basis. Evaluating nine models and four harnesses, we find that agents meet 90–97% of instruction-following criteria but only 49–57% of evidence-synthesis criteria and that Codex and Claude Code raise scores on reasoning tasks by 10–23 points while lowering scores on execution tasks by 6–9 points. We further introduce EULERFLOW, a multi-agent harness that checks reported findings against artifacts and execution records and returns unsupported findings for revision. EULERFLOW improves both reasoning and execution tasks with two different underlying models, raising the overall score of Claude-Opus-4.8 from 44.7 to 53.1.

1 INTRODUCTION

AI agents can now search the literature, propose methods, and run experiments (Gottweis et al., 2025; Schmidgall et al., 2025; Lu et al., 2024; Yamada et al., 2025). Completing these activities, however, is not the same as producing sound research. A research deliverable is only as useful as the support behind its conclusions. That support requires synthesizing evidence across sources, reconciling conflicting findings (Shi et al., 2025), and stating the conditions under which each conclusion holds (Peters & Chin-Yee, 2025); when the question calls for it, the underlying computations must also be carried out correctly. Existing benchmarks typically assess one of these requirements in isolation: scientific coding benchmarks verify programs with deterministic tests (Tian et al., 2024; Siegel et al., 2024; Chen et al., 2025), while research-report benchmarks grade written analyses with rubrics (Sharma et al., 2025; Li et al., 2026). Suites such as AstaBench (Bragg et al., 2026) cover both kinds of activity, but assemble them from sub-benchmarks whose tasks and scorers are constructed separately.

As a result, task requirements and scoring criteria are rarely traced to a common scientific source, and it remains unclear whether an agent that writes a convincing report has also done the work the report describes. This matters because automated research (Tie et al., 2025; Lu et al., 2024) can improve only on what it can verify (Wang et al., 2026; Starace et al., 2025; Wadden et al., 2020): a verifier that scores a convincing report without checking the work behind it rewards persuasive writing rather than sound research. A sound evaluation must therefore check both the reasoning in the report and the computation it relies on, against criteria grounded in the same scientific source as the task.

We introduce SCIPRODBENCH, a benchmark of 184 research tasks across Biology, Chemistry, Computer Science, and Physics, in two modes: *Retrieval and Reasoning* (RR) and *Retrieval, Reasoning,*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

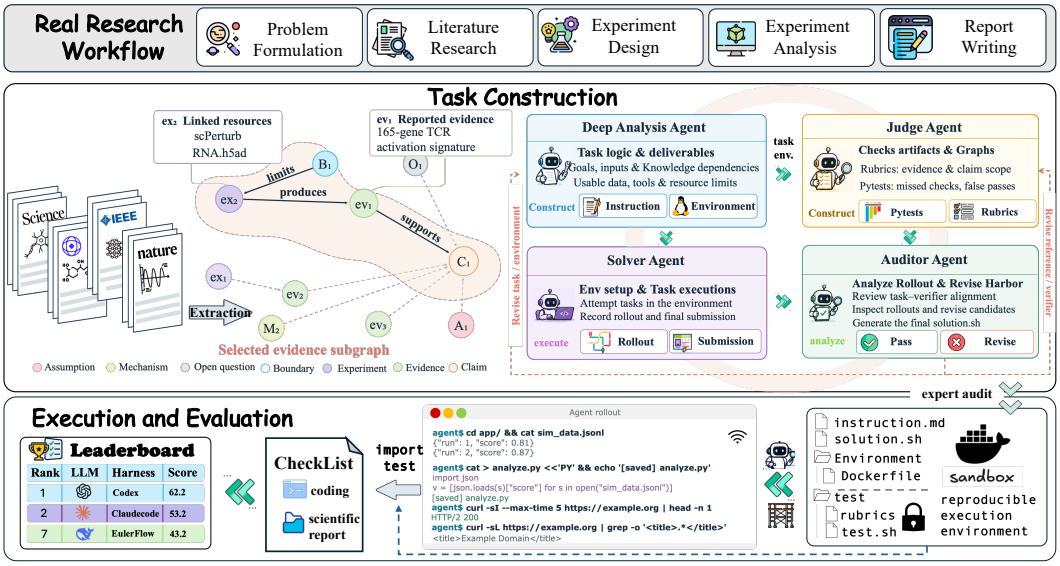


Figure 1: **Overview of SciProdBench.** SciProdBench uses evidence subgraphs extracted from papers to guide four agents in constructing and refining tasks and verifiers. After expert review, evaluated agents execute tasks in Harbor environments, with report rubrics assessing evidence-supported conclusions and executable tests checking required computations.

and Execution (RRE), which additionally requires computation (§3.1). Rubrics score the reasoning in each report and hidden executable tests check the computation in each RRE task; both are derived from the same scientific evidence graph as the task itself, so every requirement and scoring condition can be traced to its scientific basis.

Evaluating nine models and four harnesses, we find that current agents reliably satisfy the form of a research task but often fail at its substance: Instruction Following criteria pass at 90–97% across disciplines, whereas Synthesis of Information criteria pass at only 49–57%. Existing harnesses do not close this gap: Codex and Claude Code raise their backbones' RR scores by 10–23 points but lower RRE by 6–9 points, trading computation for better-written reports.

These findings motivate EULERFLOW, a multi-agent research harness (Wu et al., 2023; Schmidgall et al., 2025) that brings verification into the research loop: it accepts a finding only after checking it against the artifacts that should support it. A Coordinator Agent delegates work to Research and Coding Agents, opens the files they submit, and compares reported values with saved results and execution records; missing evidence or inconsistent results are returned for revision. EULERFLOW improves both task modes on two backbones, raising Overall from 44.7 to 53.1 on Claude-Opus-4.8 and from 36.9 to 42.3 on DeepSeek-V4-Pro, and achieves the highest accuracy among harnesses sharing its backbone on two of three external search benchmarks.

Main Contributions.

- **SCIPRODBENCH**, 184 research tasks across four disciplines in which report rubrics and executable tests separately measure evidence synthesis and computational correctness on the same tasks.
- **Scientific task–verifier construction**, a multi-agent pipeline that derives tasks and traceable verifiers from shared evidence graphs, with auditing, solver rollouts, and expert review.
- **EULERFLOW**, a multi-agent harness that delegates by deliverable and verifies findings against artifacts before acceptance, improving both RR and RRE on two backbones.

Table 1: Comparison with related benchmarks by task type and scoring method. *Joint scoring* indicates that the same task is scored by both a rubric and a deterministic verifier. ✓/✗: present/absent.

Benchmark	Retrieval & synthesis	Coding & execution	Rubric	Determ. verifier	Joint scoring
SciCode (Tian et al., 2024)	✗	✓	✗	✓	✗
CORE-Bench (Siegel et al., 2024)	✗	✓	✗	✓	✗
ScienceAgentBench (Chen et al., 2025)	✗	✓	✗	✓	✗
DiscoveryBench (Majumder et al., 2025)	✗	✓	✗	✗	✗
LABBench2 (Laurent et al., 2026)	✗	✓	✗	✓	✗
TB-Science (TB-Science Team, 2026)	✗	✓	✗	✓	✗
PaperBench (Starace et al., 2025)	✗	✓	✓	✗	✗
DeepResearch Bench II (Li et al., 2026)	✓	✗	✓	✗	✗
AstaBench (Bragg et al., 2026)	✓	✓	✓	✓	✗
SCIPRODBENCH (ours)	✓	✓	✓	✓	✓

2 RELATED WORK

Scientific benchmarks and task construction. We now compare SCIPRODBENCH with recent scientific benchmarks, as detailed in Table 1. Existing benchmarks evaluate scientific coding and research engineering (Tian et al., 2024; Wijk et al., 2025), computational reproducibility (Siegel et al., 2024), and domain-specific research tasks (Chen et al., 2025; Majumder et al., 2025; Laurent et al., 2026). Their tasks draw on published discoveries (Majumder et al., 2025), replication rubrics co-developed with original paper authors (Starace et al., 2025), and expert authoring or extraction from research reports (Sharma et al., 2025; Li et al., 2026). AstaBench (Bragg et al., 2026) is the closest prior work, assembling sub-benchmarks that span literature search, coding, data analysis, and end-to-end discovery under a shared environment, with tasks and scorers defined per sub-benchmark. SCIPRODBENCH instead derives each task and its verifier from the same scientific evidence graph, so that task requirements and scoring criteria share a traceable source.

Evaluating research outputs. Evaluation protocols range from programmatic checks of task outputs (Merrill et al., 2026; Terminal-Bench-Science Team, 2026) to LLM judges that assess reasoning, synthesis, and evidence support in research reports (Du et al., 2025; Sharma et al., 2025). PaperBench (Starace et al., 2025) executes replication submissions and grades code, execution, and results with a single LLM judge using hierarchical rubrics. SCIPRODBENCH keeps these signals separate on the same task: a rubric judge scores the reasoning in the report text, and hidden executable tests independently check computational artifacts, so the two can be compared directly.

Automated research systems. Automated research systems support literature synthesis (Asai et al., 2024), experimentation and manuscript generation (Lu et al., 2024; Yamada et al., 2025), and hypothesis refinement (Gottweis et al., 2025), and general-purpose coding harnesses are increasingly applied to research tasks (Anthropic, 2026c; OpenAI). EULERFLOW targets a failure mode that SCIPRODBENCH exposes: it accepts a reported finding only after checking it against saved artifacts and execution records.

3 SCIPRODUCTIVITY BENCHMARK

3.1 OVERVIEW

SCIPRODBENCH evaluates research tasks spanning problem formulation, literature review, experimental design, analysis, and reporting. Tasks are divided into two modes according to whether scientific computation is required. *Retrieval and Reasoning* (RR) tasks require agents to retrieve evidence, synthesize findings, and present conclusions in a research report. *Retrieval, Reasoning, and Execution* (RRE) tasks additionally require agents to perform scientific computation, submit code and verifiable artifacts, and integrate the results into their reports.

Each task specifies a research question, required deliverables, and a resource budget. Tasks are packaged in the Harbor format (Merrill et al., 2026), with a containerized environment, a working directory, and a hidden verifier. A rubric-based judge evaluates report text in both modes; hidden executable tests additionally verify computational outputs in RRE (§3.5).

3.2 Scientific Evidence Graph

Each anchor paper is structured as a *scientific evidence graph* that hierarchically layers claims, evidence, experiments, and their supporting relationships. Both task instructions and evaluation criteria are sampled from these graphs, ensuring that task content and scoring standards share the same source material.

Source collection. We construct epistemic graphs from a corpus of 868 papers spanning four scientific domains. Of these papers, 139 have associated code repositories. The corpus also includes 1,048 data artifacts supporting reported results or their reproduction.

Graph construction. For each anchor paper a, an LLM-based pipeline extracts scientific content from the full text and organizes it into a directed property multigraph (Angles, 2018):

$$G_a = (N,\, E,\, \rho,\, \lambda,\, \sigma), \tag{1}$$

where N and E are the node and edge sets, $\rho\colon E \to N \times N$ maps each edge to its ordered source–target pair, and λ, σ assign labels and properties to nodes and edges.

(1) Nodes represent nine epistemic roles: experiments, evidence, claims, mechanisms, assumptions, boundaries, formulations, contrasts with prior work, and open questions. **(2) Edges** encode scientific relationships with attributes such as support polarity, entailment strength, and assumption updates. **(3) Filtering.** We retain extracted elements with confidence $c(x) > 0.85$, $x \in N \cup E$, and discard edges whose endpoints fall below this threshold; confidence reflects extraction reliability, not scientific validity. Appendix A.1 describes the graph schema.

Web enrichment. Retrieval guided by the retained graph neighborhood identifies additional web evidence. An LLM assesses relevance and relationships to existing content; accepted additions form an enriched graph G_a^+ that preserves all original elements while adding source-attributed content.

3.3 Task Formulation and Validation

As shown in Figure 1, four specialized agents coordinate the construction: a Deep Analysis Agent drafts tasks, a Judge Agent constructs reference solutions and verifiers, a Solver Agent attempts the tasks, and an Auditor Agent reviews results and iterates on defects.

From scientific graphs to tasks. A subgraph $G_s \subseteq G_a^+$ is sampled using one of six strategies in Appendix A.2, balancing coverage across node types so that tasks do not collapse into claim recall, evidence lookup, or isolated code execution. Task generation maps each G_s to four components:

$$\Phi(G_s) = (I,\, \mathcal{E},\, S^\star,\, V). \tag{2}$$

The Deep Analysis Agent uses the full graph structure, associated repository, and data artifacts to produce the task instruction I and execution environment $\mathcal{E}$, including a runnable Dockerfile and working directory. Tasks target scientific objectives such as extending an analysis, comparing methods, or testing a conclusion under specified conditions. Both construction agents operate on the full subgraph G_s, but the resulting task exposes only a subset of each node's attributes to the agent under test. An exposed view $M^+(G_s)$ and a withheld view $M^-(G_s)$ partition the attributes of each node: for example, an experiment node exposes its setup and input data but withholds the reference outcome. The task instruction is generated from M^+; the verification targets $S^\star$ and verifier V are grounded in M^-. All task requirements must appear in exposed attributes; withheld attributes must not introduce additional obligations.

Constructing reference solutions and verifiers. The Judge Agent receives I, $\mathcal{E}$, $M^-(G_s)$, and supporting sources. It derives verification targets $S^\star$ specifying reference quantities, constraints, and

evidence dependencies without prescribing a unique answer. For both modes, it develops a reference solution and hierarchical rubrics. For RRE tasks, it executes repository code or a generated reference implementation and checks outputs against source material. The resulting verifier comprises two layers:

$$V = (V^{\text{out}}, V^{\text{exec}}), \qquad V^{\text{exec}} = \varnothing \quad \text{for RR tasks.} \tag{3}$$

V^{out} derives rubric criteria from graph content to assess evidence use, assumptions, and support for conclusions. V^{exec} checks results, code behavior, and code–data invariants including intermediate artifacts absent from reports.

Auditing and rollout-based refinement. The Auditor Agent checks task–environment–verifier alignment using G_s and solver trajectories: whether the instruction matches the verifier's expectations, whether the environment provides required dependencies, and whether the reference solution is reproducible. It also checks for reference leakage in the exposed view. The Solver Agent attempts the task in independent runs using only I and permitted resources in $\mathcal{E}$. When a run fails, the Auditor traces the failure to its source: instruction ambiguities are routed to the Deep Analysis Agent, verifier errors to the Judge Agent, and environment issues to whichever agent configured them. Each responsible agent revises its outputs and the Auditor rechecks. This cycle repeats until all checks pass or the task is rejected. Solver failure alone does not justify relaxing correctness criteria.

Expert adjudication At least two domain experts with doctoral degrees review each candidate against three requirements. *Grounded*: the instruction must be scientifically sound with conclusions traceable to the evidence graph; Ill-posed tasks are rejected, leaving 67% of candidates. *Challenging*: the task must demand evidence synthesis, not direct lookup; experts verify that rubric criteria and their positive/negative polarity are correctly assigned. *Verifiable*: scoring criteria must match stated requirements and executable tests must not exceed scope. Defects in V^{out} are revised, concentrating on EC and SI; defects in V^{exec} are revised to remove scope violations. Review averages 1.5 expert-hours per task.

3.4 Benchmark Construction

Domains and areas. The benchmark contains 184 tasks spanning 4 domains, 23 subdomains, and 15 standard areas: Computer Science (50), Biology (45), Chemistry (45), and Physics (44). We use domains for reporting, subdomains for sampling balance, and standard areas for taxonomy alignment. Appendix A.4 lists the subdomain-to-area mappings and classification sources. Figure 2 shows the task distribution. Each subdomain contains 3–15 tasks. The benchmark contains 153 RR and 31 RRE tasks. RRE tasks cover 11 of the 23 subdomains, where source papers provide runnable code and reference results against which computational outputs can be checked.

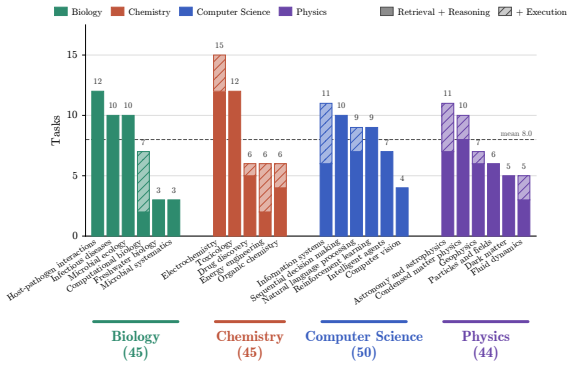


Figure 2: Tasks per subdomain, grouped by domain and split by track.

Cost. RR tasks take a median of 23.8 wall-clock minutes, with an interquartile range of 18.2–30.6 minutes. The distribution is right-skewed, with a 90th percentile of 38.6 minutes and a maximum of 102.2 minutes. Mean runtime is 10.2 minutes longer for RRE tasks than for RR tasks.

3.5 Evaluation Protocol

Each run ends upon submission or budget exhaustion. Rubrics and executable tests are hidden from the agent. Every task is scored on its research report; RRE tasks are additionally scored on computational outputs verified by hidden executable tests.

Scoring. (1) **Report scoring.** Reports are evaluated along five dimensions: Explicit Criteria (EC), Synthesis of Information (SI), References and Citation (RC), Instruction Following (IF), and Communication Quality (CQ), with weights $w_d = 0.25, 0.35, 0.25, 0.05, 0.10$, respectively. A judge assigns binary verdicts to rubric items: positive items pass when requirements are met, and negative items pass when specified errors are avoided. Each item is labeled *must* if mandatory or *nice* if supplementary. Any failed must item sets its dimension score $J_{i,d}$ to zero. Otherwise, positive must and nice pass rates are weighted 0.75 and 0.25, minus 0.15 times the negative nice failure rate, with the result clipped to $[0, 1]$. Appendix A.5 gives the complete definition. (2) **Task scoring.** The weighted report score J_i determines the RR score; RRE combines it equally with executable verification:

$$J_i = \sum_d w_d J_{i,d}, \qquad S_i = \begin{cases} J_i, & \text{RR}, \\ \frac{1}{2} J_i + \frac{1}{2} V_i, & \text{RRE}. \end{cases} \tag{4}$$

Here, $V_i = 1$ if all required executable tests pass, and $V_i = 0$ otherwise. (3) **Aggregation.** Within each mode, we average task scores within each domain, then average the four domain means equally. Overall is the equally weighted mean of RR and RRE.

4 Research Harness

During development, we observed that agents routinely produce well-structured reports whose conclusions lack computational support: methods sections describe plausible analyses, yet the underlying code either fails silently or is never executed. This gap between surface compliance and evidential grounding motivates EulerFlow, a multi-agent research harness that enforces artifact-level verification before accepting any claim. Its design rests on two principles: separating worker-reported progress from coordinator acceptance, and connecting scientific claims to inspectable execution artifacts.

- **Architecture.** A Coordinator Agent decomposes each task and delegates subtasks to Research Agents for evidence gathering and synthesis, or to Coding Agents for sandboxed computation. Workers of the same type execute concurrently within batches. Each worker returns a structured report and deposits artifacts on a shared filesystem; full interaction histories are not propagated.
- **Shared task plan.** A shared `plan.json` records task scopes, expected outputs, statuses, and delivery notes. This plan is injected into the context before each model call, giving every agent a consistent view of progress. The Coordinator manages top-level tasks; workers may create and update child items but cannot close parent tasks.
- **Review and revision.** The Coordinator evaluates each submission against three requirements: *Coverage* requires that assigned questions are addressed and promised artifacts delivered. *Artifact validity* requires that file contents satisfy the requested analysis rather than merely exist. *Evidence consistency* requires that reported findings agree with saved artifacts and execution logs. If any check fails, the Coordinator records the deficiency in the shared plan and reassigns the work. This cycle repeats until all checks pass or the resource budget is exhausted. The final report is written to `/app/report.md`, with computational findings traceable to `/app/results/` and unresolved gaps explicitly disclosed.

5 Experiments

5.1 Evaluation Setup

Models and Harness. We evaluate the following models on SciProdBench under a shared base harness in Appendix C.1: GPT-5.5 (OpenAI, 2026), Grok-4.6 (xAI, 2026), GLM-5.1 and GLM-5.2 (GLM-5 Team et al., 2026), Claude-Opus-4.7 (Anthropic, 2026a), Claude-Opus-4.8 (Anthropic, 2026b), Kimi-K3 (Moonshot AI, 2026), DeepSeek-V4-Pro (DeepSeek-AI, 2026), and Qwen3.6-35B-A3B (Qwen Team, 2026).

To isolate harness effects, we also run Claude Code CLI v2.1.267 (Anthropic, 2026c) with Claude-Opus-4.8, Codex CLI v0.118.0 (OpenAI) with GPT-5.5, and EulerFlow with both Claude-Opus-4.8 and DeepSeek-V4-Pro.

Table 2: SCIPRODBENCH results across four scientific disciplines with base, specialized, and EulerFlow harnesses. Scores are reported on a 0–100 scale.

Agent	Bio.		Chem.		CS		Phys.		Overall
	RRE	RR	RRE	RR	RRE	RR	RRE	RR	
LLM with base harness									
`GPT-5.5`	**69.8**	56.8	**53.0**	59.4	**77.4**	62.7	**60.6**	51.7	**61.4**
`Claude-Opus-4.7`	35.2	34.2	50.9	53.4	57.5	53.3	57.7	36.6	47.4
`GLM-5.1`	45.0	44.7	38.8	55.5	51.2	45.9	53.0	39.2	46.6
`Claude-Opus-4.8`	52.9	30.8	37.0	50.9	57.8	46.1	49.2	32.8	44.7
`GLM-5.2`	56.4	42.5	35.2	42.5	39.2	49.3	54.7	34.4	44.3
`Grok-4.6`	24.6	**62.2**	17.7	**68.8**	13.2	**81.7**	9.6	**62.9**	42.6
`DeepSeek-V4-Pro`	27.4	39.4	43.1	48.6	27.1	44.5	33.1	31.7	36.9
`Qwen3.6-35B-A3B`	39.9	32.3	28.2	41.7	42.5	37.0	35.5	28.9	35.8
`Kimi-K3`	9.5	47.6	33.1	52.3	37.9	52.8	6.9	43.4	35.4
LLM with specialized harness									
Claude Code `Opus-4.8`	43.8	47.5	44.0	71.3	33.6	**75.8**	52.0	**57.5**	53.2
Codex `GPT-5.5`	**56.6**	**69.9**	**45.8**	**72.4**	**68.6**	75.3	**53.8**	55.3	**62.2**
LLM with our harness									
EulerFlow `DeepSeek-V4-Pro`	$26.4^{-0.9}$	$39.7^{+0.4}$	$44.9^{+1.9}$	$53.9^{+5.3}$	$44.0^{+16.9}$	$43.5^{-1.0}$	$41.5^{+8.4}$	$\mathbf{44.0}^{+12.4}$	$42.3^{+5.4}$
EulerFlow `Claude-Opus-4.8`	$\mathbf{56.5}^{+3.6}$	$\mathbf{46.1}^{+15.3}$	$\mathbf{45.1}^{+8.1}$	$\mathbf{63.7}^{+12.7}$	$\mathbf{63.0}^{+5.2}$	$\mathbf{57.7}^{+11.6}$	$\mathbf{48.8}^{-0.4}$	$43.7^{+10.9}$	$\mathbf{53.1}^{+8.4}$

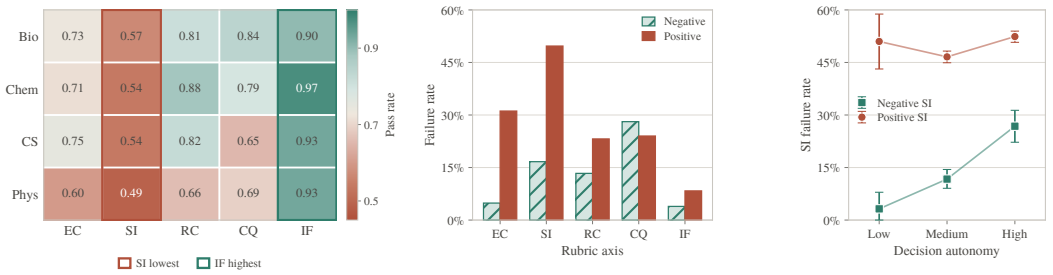


Figure 3: Rubric-level performance across disciplines, criterion polarities, and levels of decision autonomy. **Left:** SI has the lowest pass rate and IF the highest in all four disciplines. **Middle:** Positive criteria specify required behaviors, whereas negative criteria identify prohibited errors; the failure-rate gap is largest for SI. **Right:** Positive SI failure rates remain near 50% across autonomy levels, while negative SI failures increase as more research decisions are left to the agent.

5.2 Main Results

Agents show uneven performance across task modes and weaker evidence synthesis than instruction following, while harness design and targeted fine-tuning improve distinct aspects of research performance.

Performance varies across models and disciplines. In Table 2, GPT-5.5 achieves the highest Overall score at 61.4 and leads RRE in all four disciplines, whereas Grok-4.6 leads every RR column. The contrast is most pronounced in CS, where Grok-4.6 scores 81.7 on RR but only 13.2 on RRE, compared with 62.7 and 77.4 for GPT-5.5. Similar Overall scores can mask opposite profiles: Grok-4.6 and GLM-5.2 score 42.6 and 44.3 Overall, yet Grok-4.6 dominates every RR column while GLM-5.2 dominates every RRE column. Disciplinary variation is equally sharp. Chemistry and CS tie for the highest average RR at 52.6, yet Chemistry records the lowest average RRE at 37.4. Even GPT-5.5's RRE ranges from 53.0 in Chemistry to 77.4 in CS.

Following instructions does not ensure evidence synthesis. Across all four disciplines, IF pass rates range from 0.90 to 0.97 and SI pass rates from 0.49 to 0.57, the highest and lowest among the



Figure 4: External benchmark results: accuracy on BrowseComp-Plus and Item F1 on WideSearch and DeepWideSearch. EulerFlow and WebSwarm* are evaluated by us using DeepSeek-V4-Pro. See Appendix C.5 for baseline sources and evaluation settings.

five dimensions shown in Figure 3. High IF scores therefore provide little assurance that an agent has performed the substantive analytical work a task demands.

Criterion polarity identifies the source of this deficit. Positive SI criteria, which require agents to *produce* a correct analysis, fail in roughly half of cases; negative criteria, which penalize specific errors, fail in only one-sixth. Agents thus avoid prohibited mistakes far more reliably than they carry out the required evidential reasoning. This polarity gap appears across all rubric dimensions but is largest for SI; it reverses only slightly for CQ, where negative criteria are marginally harder.

The asymmetry also varies with decision autonomy. Positive SI failure rates hold near 50% at low, medium, and high autonomy, suggesting a stable capability ceiling. Negative SI failures, by contrast, rise from roughly 3% at low autonomy to 12% at medium and 27% at high. Greater autonomy is thus associated with additional error modes but not with improved synthesis ability.

These patterns are compatible with several explanations. The simplest is that evidential synthesis is intrinsically more demanding than instruction compliance. However, the polarity structure of the gap offers additional diagnostic value: if the difficulty were purely intrinsic, one would expect positive and negative criteria to fail at comparable rates; instead, positive criteria fail at roughly three times the rate of negative criteria.

Performance improves with harness design. Table 2 shows that Codex and Claude Code raise RR by 10–23 points but lower RRE by 6–9 points: stronger report writing at the cost of weaker verified computation. EULERFLOW improves both modes on two backbones, raising Overall from 44.7 to 53.1 on Claude-Opus-4.8 and from 36.9 to 42.3 on DeepSeek-V4-Pro. Although EULERFLOW and Claude Code reach similar Overall scores on Claude-Opus-4.8, their profiles differ by roughly 10 points in opposite directions on RR and RRE.

Paired within-task comparisons in Table 3 reveal which rubric dimensions each harness improves. Claude Code and Codex yield the largest gains on SI, the dimension with the lowest base-harness pass rates. EULERFLOW on Claude-Opus-4.8 instead improves EC, RC, and CQ significantly. No configuration significantly improves IF, already near ceiling.

As Figure 4 shows, with DeepSeek-V4-Pro fixed, EULERFLOW leads all compared harnesses on BrowseComp-Plus, at 72.05% accuracy, and on WideSearch, at 78.3% Item F1, and trails the best DeepWideSearch result by 0.3 points.

Table 3: Report-dimension gains (score points). Top: harness gains over the corresponding base harness; bold: $p < 0.05$. Bottom: SFT gains over Qwen3.6 base.

Configuration	SI	EC	CQ	RC	IF
Specialized harness (Δ vs. base harness)					
Claude Code	**+31.2**	**+28.6**	**+23.3**	**+16.0**	+6.4
Codex	**+17.9**	**+11.3**	**+10.0**	+1.4	+2.7
EulerFlow (Δ vs. base harness)					
Claude-Opus-4.8	+7.1	**+10.2**	**+12.6**	**+15.7**	+5.5
DeepSeek-V4-Pro	+5.2	+5.2	+3.5	+2.4	+2.6
Post-training (Δ vs. Qwen3.6 base)					
Qwen3.6-SFT	+3.0	+4.8	+18.5	+14.3	−8.3

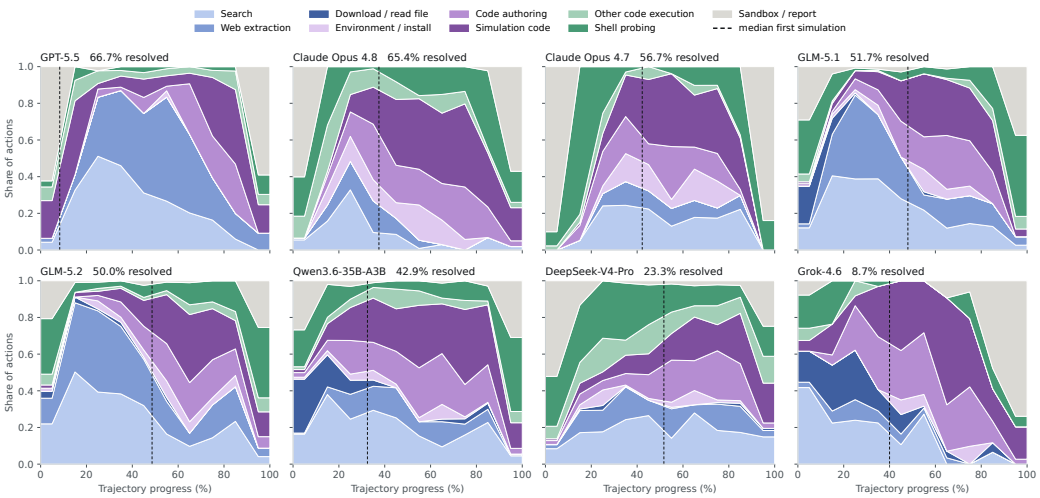


Figure 5: Action composition over normalized trajectory progress for eight models, ordered by task resolution rate. Colored areas show the share of actions in nine categories at each position. The dashed line marks the median position of the first simulation action. Progress is normalized within each run; the panels therefore show relative action allocation rather than absolute trajectory length.

In a preliminary experiment, supervised fine-tuning of Qwen3.6-35B-A3B on rubric-validated solver trajectories raises Overall score by 7.5 points, as shown in Table 3. The largest changes are in CQ ($+18.5$) and RC ($+14.3$), with a decline in IF (-8.3). This provides initial evidence that SCIPRODBENCH's task data carries learnable signal for research task performance; training details appear in Appendix C.4.

5.3 AGENT BEHAVIOR

Diverse workflows converge on a shared execution requirement. Figure 5 shows that models adopt markedly different research strategies: some compute before searching, others retrieve before coding, and action counts per run vary by an order of magnitude. Despite this diversity, successful runs almost always complete a *closed computational chain*: reading the supplied data, writing and executing a solver without runtime errors, and obtaining a clean result before reporting. Of 104 resolved runs, 103 close this chain; only one of 23 unclosed runs resolves. Under task and model fixed effects, never obtaining a clean computation lowers the probability of resolution by 0.36 ($p = .006$), and ending on one raises it by 0.25 ($p = .029$).

Chain closure is associated with greater report traceability. When the chain closes, 72–75% of precise numbers in the report match artifacts the agent saved, compared with 36% when it does not; reports cite a nonexistent result file in about 1% of closed runs versus 50% of unclosed ones. Under the same fixed effects, closure raises the share of traceable numbers by 0.30 ($p = .003$).

6 CONCLUSION

We introduced SCIPRODBENCH, which uses scientific evidence graphs to jointly construct research tasks and their verifiers, assessing both the reasoning in agents' reports and, where a task requires it, the correctness of their computation. Our evaluation shows that current agents follow instructions reliably but struggle to synthesize evidence and to complete the required computation. EULERFLOW, our multi-agent research harness, improves mean performance in both task modes on two backbones. Together, these results suggest that harness design can partly compensate for current model limitations, and that verifiable evaluation is what makes these gaps, and the extent of that compensation, measurable.

AI USE STATEMENT

We used generative AI tools to assist with architecture design and code implementation, support expert review of benchmark cases, polish the manuscript, and provide feedback on our methods and analyses. We also used LLMs for benchmark construction and report evaluation, as described in the paper. The authors reviewed the AI-assisted outputs and made the final research and editorial decisions.

ETHICS STATEMENT

This work aims to support responsible scientific research through evidence-based evaluation of AI agents. We emphasize human oversight and independent validation when applying these systems to scientific research.

REPRODUCIBILITY STATEMENT

The evaluation protocol and experimental setup are described in Sections 3.5 and 5.1. Appendices A, B, and C provide the evidence-graph schema, scoring rules, agent prompt excerpts, and additional experimental settings.

REFERENCES

Renzo Angles. The Property Graph Database Model. In *Alberto Mendelzon International Workshop on Foundations of Data Management (AMW)*, volume 2100 of *CEUR Workshop Proceedings*, 2018. URL https://ceur-ws.org/Vol-2100/paper26.pdf.

Anthropic. Introducing Claude Opus 4.7, 2026a. URL https://www.anthropic.com/news/claude-opus-4-7.

Anthropic. Introducing Claude Opus 4.8, 2026b. URL https://www.anthropic.com/news/claude-opus-4-8.

Anthropic. Claude Code, Version 2.1.267, 2026c. URL https://github.com/anthropics/claude-code/releases/tag/v2.1.267. Software release.

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'Arcy, David Wadden, Matt Latzke, Minyang Tian, Pan Ji, Shengyan Liu, Hao Tong, Bohao Wu, Yanyu Xiong, Luke Zettlemoyer, Graham Neubig, Daniel S. Weld, Doug Downey, Wen-tau Yih, Pang Wei Koh, and Hannaneh Hajishirzi. OpenScholar: Synthesizing Scientific Literature with Retrieval-augmented LMs. *CoRR*, abs/2411.14199, 2024. doi: 10.48550/arXiv.2411.14199. URL https://doi.org/10.48550/arXiv.2411.14199.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi Mishra, Sergey Feldman, Dany Haddad, Jena D. Hwang, Peter Jansen, Varsha Kishore, Bodhisattwa Prasad Majumder, Aakanksha Naik, Sigal Rahamimov, Kyle Richardson, Amanpreet Singh, Harshit Surana, Aryeh Tiktinsky, Rosni Vasu, Guy Wiener, Chloe Anastasiades, Stefanus Candra, Jason Dunkelberger, Daniel Emery, Rob Evans, Malachi Hamada, Regan Huff, Rodney Kinney, Matt Latzke, Jaron Lochner, Ruben Lozano-Aguilera, Ngoc-Uyen Nguyen, Smita Rao, Amber Tanaka, Brooke Vlahos, Peter Clark, Doug Downey, Yoav Goldberg, Ashish Sabharwal, and Daniel S. Weld. AstaBench: Rigorous Benchmarking of AI Agents with a Scientific Research Suite. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=M7TNf5J26u.

Mingju Chen, Guibin Zhang, Heng Chang, Yuchen Guo, and Shiji Zhou. A-mapreduce: Executing wide search via agentic mapreduce. *CoRR*, abs/2602.01331, 2026a. doi: 10.48550/ARXIV.2602.01331. URL https://doi.org/10.48550/arXiv.2602.01331.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Jimmy Lin, Akari Asai, and Victor Zhong. Agentir: Reasoning-aware retrieval for deep research agents. *CoRR*, abs/2603.04384, 2026b. doi: 10.48550/ARXIV.2603.04384. URL https://doi.org/10.48550/arXiv.2603.04384.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Sahel Sharifymoghaddam, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Hosna Oyarhoseini, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. Browsecomp-plus: A fair and disentangled evaluation benchmark for deep search agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 22349–22370. Association for Computational Linguistics, 2026c. doi: 10.18653/V1/2026.ACL-LONG.1023. URL https://doi.org/10.18653/v1/2026.acl-long.1023.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=6z4YKr0GK6.

DeepSeek-AI. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence, 2026.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. DeepResearch Bench: A Comprehensive Benchmark for Deep Research Agents. *CoRR*, abs/2506.11763, 2025. doi: 10.48550/arXiv.2506.11763. URL https://doi.org/10.48550/arXiv.2506.11763.

GLM-5 Team, Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, Chenzheng Zhu, Congfeng Yin, Cunxiang Wang, Gengzheng Pan, Hao Zeng, Haoke Zhang, Haoran Wang, Huilong Chen, Jiajie Zhang, Jian Jiao, Jiaqi Guo, Jingsen Wang, Jingzhao Du, Jinzhu Wu, Kedong Wang, Lei Li, Lin Fan, Lucen Zhong, Mingdao Liu, Mingming Zhao, Pengfan Du, Qian Dong, Rui Lu, Shuang-Li, Shulin Cao, Song Liu, Ting Jiang, Xiaodong Chen, Xiaohan Zhang, Xuancheng Huang, Xuezhen Dong, Yabo Xu, Yao Wei, Yifan An, Yilin Niu, Yitong Zhu, Yuanhao Wen, Yukuo Cen, Yushi Bai, Zhongpei Qiao, Zihan Wang, Zikang Wang, Zilin Zhu, Ziqiang Liu, Zixuan Li, Bojie Wang, Bosi Wen, Can Huang, Changpeng Cai, Chao Yu, Chen Li, Chengwei Hu, Chenhui Zhang, Dan Zhang, Daoyan Lin, Dayong Yang, Di Wang, Ding Ai, Erle Zhu, Fangzhou Yi, Feiyu Chen, Guohong Wen, Hailong Sun, Haisha Zhao, Haiyi Hu, Hanchen Zhang, Hanrui Liu, Hanyu Zhang, Hao Peng, Hao Tai, Haobo Zhang, He Liu, Hongwei Wang, Hongxi Yan, Hongyu Ge, Huan Liu, Huanpeng Chu, Jia'ni Zhao, Jiachen Wang, Jiajing Zhao, Jiamin Ren, Jiapeng Wang, Jiaxin Zhang, Jiayi Gui, Jiayue Zhao, Jijie Li, Jing An, Jing Li, Jingwei Yuan, Jinhua Du, Jinxin Liu, Junkai Zhi, Junwen Duan, Kaiyue Zhou, Kangjian Wei, Ke Wang, Keyun Luo, Laiqiang Zhang, Leigang Sha, Liang Xu, Lindong Wu, Lintao Ding, Lu Chen, Minghao Li, Nianyi Lin, Pan Ta, Qiang Zou, Rongjun Song, Ruiqi Yang, Shangqing Tu, Shangtong Yang, Shaoxiang Wu, Shengyan Zhang, Shijie Li, Shuang Li, Shuyi Fan, Wei Qin, Wei Tian, Weining Zhang, Wenbo Yu, Wenjie Liang, Xiang Kuang, Xiangmeng Cheng, Xiangyang Li, Xiaoquan Yan, Xiaowei Hu, Xiaoying Ling, Xing Fan, Xingye Xia, Xinyuan Zhang, Xinze Zhang, Xirui Pan, Xu Zou, Xunkai Zhang, Yadi Liu, Yandong Wu, Yanfu Li, Yidong Wang, Yifan Zhu, Yijun Tan, Yilin Zhou, Yiming Pan, Ying Zhang, Yinpei Su, Yipeng Geng, Yong Yan, Yonglin Tan, Yuean Bi, Yuhan Shen, Yuhao Yang, Yujiang Li, Yunan Liu, Yunqing Wang, Yuntao Li, Yurong Wu, Yutao Zhang, Yuxi Duan, Yuxuan Zhang, Zezhen Liu, Zhengtao Jiang, Zhenhe Yan, Zheyu Zhang, Zhixiang Wei, Zhuo Chen, Zhuoer Feng, Zijun Yao, Ziwei Chai, Ziyuan Wang, Zuzhou Zhang, Bin Xu, Minlie Huang, Hongning Wang, Juanzi Li, Yuxiao Dong, and Jie Tang. GLM-5: from Vibe Coding to Agentic Engineering, 2026. URL https://arxiv.org/abs/2602.15763.

Juraj Gottweis, Wei-Hung Weng, Alexander N. Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an AI co-scientist. *CoRR*, abs/2502.18864, 2025. doi: 10.48550/ARXIV.2502.18864. URL https://doi.org/10.48550/arXiv.2502.18864.

Tz-Huan Hsu, Jheng-Hong Yang, and Jimmy Lin. Rethinking agentic search with pi-serini: Is lexical retrieval sufficient? *CoRR*, abs/2605.10848, 2026. doi: 10.48550/ARXIV.2605.10848. URL https://doi.org/10.48550/arXiv.2605.10848.

Tian Lan, Bin Zhu, Qianghuai Jia, Junyang Ren, Haijun Li, Longyue Wang, Zhao Xu, Weihua Luo, and Kaifu Zhang. Deepwidesearch: Benchmarking depth and width in agentic information seeking. *CoRR*, abs/2510.20168, 2025. doi: 10.48550/ARXIV.2510.20168. URL https://doi.org/10.48550/arXiv.2510.20168.

Jon M. Laurent, Albert Bou, Michael Pieler, Conor Igoe, Alex Andonian, Siddharth Narayanan, James D. Braza, Alexandros Sanchez Vassopoulos, Jacob L. Steenwyk, Blake Lash, Andrew D. White, and Samuel G. Rodriques. Labbench2: An improved benchmark for AI systems performing biology research. *CoRR*, abs/2604.09554, 2026. doi: 10.48550/ARXIV.2604.09554. URL https://doi.org/10.48550/arXiv.2604.09554.

Ruizhe Li, Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. DeepResearch Bench II: Diagnosing Deep Research Agents via Rubrics from Expert Reports. *CoRR*, abs/2601.08536, 2026. doi: 10.48550/arXiv.2601.08536. URL https://doi.org/10.48550/arXiv.2601.08536.

Qi Liu, Yiqun Chen, Zidan Chen, Yan Gao, Yi Wu, Yao Hu, Jiaxin Mao, Fengbin Zhu, and Tat-Seng Chua. Fetch-then-explore: Decoupling selection from extraction over a persistent workspace for search agents. *CoRR*, abs/2608.02097, 2026. doi: 10.48550/ARXIV.2608.02097. URL https://doi.org/10.48550/arXiv.2608.02097.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob N. Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. *CoRR*, abs/2408.06292, 2024. doi: 10.48550/ARXIV.2408.06292. URL https://doi.org/10.48550/arXiv.2408.06292.

Yuheng Lu, Qingcheng Zeng, Heli Qi, Puxuan Yu, Fuheng Zhao, Rui Yang, Hitomi Yanaka, Naoto Yokoya, and Weihao Xuan. Beyond monolingual deep research: Evaluating agents and retrievers with cross-lingual browsecomp-plus. *CoRR*, abs/2606.15345, 2026. doi: 10.48550/ARXIV.2606.15345. URL https://doi.org/10.48550/arXiv.2606.15345.

Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. DiscoveryBench: Towards Data-Driven Discovery with Large Language Models. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=vyflgpwfJW.

Mike A Merrill, Alexander Glenn Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, Junhong Shen, Guanghao Ye, Haowei Lin, Jason Poulos, Maoyu Wang, Marianna Nezhurina, Di Lu, Orfeas Menis Mastromichalakis, Zhiwei Xu, Zizhao Chen, Yue Liu, Robert Zhang, Leon Liangyu Chen, Anurag Kashyap, Jan-Lucas Uslu, Jeffrey Li, Jianbo Wu, Minghao Yan, Song Bian, Vedang Sharma, Ke Sun, Steven Dillmann, Akshay Anand, Andrew Lanpouthakoun, Bardia Koopah, Changran Hu, Etash Kumar Guha, Gabriel H. S. Dreiman, Jiacheng Zhu, Karl Krauth, Li Zhong, Niklas Muennighoff, Robert Kwesi Amanfu, Shangyin Tan, Shreyas Pimpalgaonkar, Tushar Aggarwal, Xiangning Lin, Xin Lan, Xuandong Zhao, Yiqing Liang, Yuanli Wang, Zilong Wang, Changzhi Zhou, David Heineman, Hange Liu, Harsh Trivedi, John Yang, Junhong Lin, Manish Shetty, Michael Yang, Nabil Omi, Negin Raoof, Shanda Li, Terry Yue Zhuo, Wuwei Lin, Yiwei Dai, Yuxin Wang, Wenhao Chai, Shang Zhou, Dariush Wahdany, Ziyu She, Jiaming Hu, Zhikang Dong, Yuxuan Zhu, Sasha Cui, Ahson Saiyed, Arinbjörn Kolbeinsson, Christopher Michael Rytting, Ryan Marten, Yixin Wang, Jenia Jitsev, Alex Dimakis, Andy Konwinski, and Ludwig Schmidt. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=a7Qa4CcHak.

Moonshot AI. Kimi-K3: Model Card, 2026. URL https://huggingface.co/moonshotai/Kimi-K3.

OpenAI. Codex CLI. URL https://github.com/openai/codex. Software. Accessed September 20, 2026.

OpenAI. Introducing GPT-5.5, 2026. URL https://openai.com/index/introducing-gpt-5-5/.

Uwe Peters and Benjamin Chin-Yee. Generalization bias in large language model summarization of scientific research. *Royal Society Open Science*, 12(4):241776, 2025. doi: 10.1098/rsos.241776. URL https://doi.org/10.1098/rsos.241776.

Qwen Team. Qwen3.6-35B-A3B: Agentic Coding Power, Now Open to All, April 2026. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent Laboratory: Using LLM Agents as Research Assistants. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 5977–6043. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-EMNLP.320. URL https://doi.org/10.18653/v1/2025.findings-emnlp.320.

Manasi Sharma, Chen Bo Calvin Zhang, Chaithanya Bandi, Clinton Wang, Ankit Aich, Huy Nghiem, Tahseen Rabbani, Ye Htet, Brian Jang, Sumana Basu, Aishwarya Balwani, Denis Peskoff, Marcos Ayestaran, Sean M. Hendryx, Brad Kenstler, and Bing Liu. ResearchRubrics: A Benchmark of Prompts and Rubrics For Evaluating Deep Research Agents. *CoRR*, abs/2511.07685, 2025. doi: 10.48550/arXiv.2511.07685. URL https://doi.org/10.48550/arXiv.2511.07685.

Xuanyu Shi, Wenjing Zhao, Ting Chen, Chao Yang, and Jian Du. Evidence triangulator: using large language models to extract and synthesize causal evidence across study designs. *Nature Communications*, 16(1):7355, 2025. doi: 10.1038/s41467-025-62783-x. URL https://doi.org/10.1038/s41467-025-62783-x.

Zachary S. Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl, and Arvind Narayanan. CORE-Bench: Fostering the Credibility of Published Research Through a Computational Reproducibility Agent Benchmark. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=BsMMc4MEGS.

Xiaoshuai Song, Liancheng Zhang, Kangzhi Zhao, Yutao Zhu, Zhongyuan Wang, Guanting Dong, Jinghan Yang, Han Li, Kun Gai, Ji-Rong Wen, and Zhicheng Dou. Webswarm: Recursive multi-agent orchestration for deep-and-wide web search. *CoRR*, abs/2607.08662, 2026. doi: 10.48550/ARXIV.2607.08662. URL https://doi.org/10.48550/arXiv.2607.08662.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's Ability to Replicate AI Research. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/starace25a.html.

Terminal-Bench-Science Team. Terminal-Bench-Science: Evaluating AI Agents on Research Workflows across Scientific Domains, 2026. URL https://doi.org/10.5281/zenodo.22110254. Version 0.1.0.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu A. Huerta, and Hao Peng. SciCode: A Research Coding Benchmark Curated by Scientists. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/36850592258c8c41cecdaa3dea5ff7de-Abstract-Datasets_and_Benchmarks_Track.html.

Guiyao Tie, Pan Zhou, and Lichao Sun. A survey of AI scientists. *CoRR*, abs/2510.23045, 2025. doi: 10.48550/ARXIV.2510.23045. URL https://doi.org/10.48550/arXiv.2510.23045.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. Fact or fiction: Verifying scientific claims. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 7534–7550. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.609. URL https://doi.org/10.18653/v1/2020.emnlp-main.609.

Xuehai Wang, Haowei Qin, Tongxin Liu, Junkai Li, Buqiang Xu, Jintian Zhang, Yijun Chen, Zirui Xue, and Shumin Deng. Learning to evaluate before improving: Automatic rubric induction for automatic research agents. *CoRR*, abs/2608.31076, 2026. doi: 10.48550/ARXIV.2608.31076. URL https://doi.org/10.48550/arXiv.2608.31076.

Hjalmar Wijk, Tao Roa Lin, Joel Becker, Sami Jawhar, Neev Parikh, Thomas Broadley, Lawrence Chan, Michael Chen, Joshua Clymer, Jai Dhyani, Elena Ericheva, Katharyn Garcia, Brian Goodrich, Nikola Jurkovic, Megan Kinniment, Aron Lajko, Seraphina Nix, Lucas Jun Koba Sato, William Saunders, Maksym Taran, Ben West, and Elizabeth Barnes. RE-Bench: Evaluating Frontier AI R&D Capabilities of Language Model Agents against Human Experts. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/wijk25a.html.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. Widesearch: Benchmarking agentic broad info-seeking. *CoRR*, abs/2508.07999, 2025. doi: 10.48550/ARXIV.2508.07999. URL https://doi.org/10.48550/arXiv.2508.07999.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen LLM applications via multi-agent conversation framework. *CoRR*, abs/2308.08155, 2023. doi: 10.48550/ARXIV.2308.08155. URL https://doi.org/10.48550/arXiv.2308.08155.

xAI. Introducing Grok 4.6, 2026. URL https://x.ai/news/grok-4-6.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob N. Foerster, Jeff Clune, and David Ha. The AI scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *CoRR*, abs/2504.08066, 2025. doi: 10.48550/ARXIV.2504.08066. URL https://doi.org/10.48550/arXiv.2504.08066.

A Benchmark Construction and Evaluation

A.1 Scientific Evidence Graph

The graph schema supplements §3.2. The graph is internal construction state and is not provided to the evaluated agent.

Representation and scope. For an anchor paper a, the directed property multigraph is

$$G_a = (N, E, \rho, \lambda, \sigma), \qquad \rho : E \to N \times N.$$

N and E are node and edge sets; distinct edges may share endpoints. The maps λ and σ assign labels and key–value properties, including scientific attributes and provenance metadata. Nodes represent the anchor paper, epistemic units, and cited-paper records. Source locations and supporting text connect graph elements and derived verification criteria to their underlying evidence. Extraction confidence measures how reliably content was recovered from the source. As in §3.2, only extracted nodes and edges with confidence $c(x) > 0.85$ are retained, and an edge is retained only when both endpoints survive filtering.

Validation and freezing. A valid graph has existing edge endpoints, source–target types consistent with the relation signatures, and all required node and edge attributes within their declared domains. Enrichment adds source-attributed content while preserving existing nodes, edges, and properties. The enriched graph is then frozen: task construction may revise a task to match the evidence, but cannot revise the evidence to accommodate the task.

Node types. Table 4 lists the node types and their attributes.

Table 4: Node types, type-specific attributes, and operational definitions. All content nodes additionally carry a source location, supporting text, and an extraction confidence $c(x)$ (§3.2). Categorical attribute domains are given in braces.

Type	Type-specific attributes	Operational definition
Experiment	setup, conditions, sample or data, dataset link, repository link, reference outcome	A concrete procedure, including its conditions and sample or data.
Evidence	result type {numerical, statistical, qualitative}, value, unit	An observed numerical, statistical, or qualitative result, without an added causal explanation.
Claim	truth value {true, false}	A falsifiable proposition that relevant evidence can support or oppose.
Mechanism	completeness {complete, incomplete, incorrect}; entity, activity, state change	A causal account connecting an entity, an activity, and a resulting state change.
Assumption	scope {method, conclusion}	A premise required by a method or conclusion but not established locally.
Boundary	kind {limitation, failure condition, applicability restriction}	A limitation, failure condition, or restriction on applicability.
Formulation	form {mathematical, computational}; role {summarize, predict}	A mathematical or computational representation used to summarize observations or derive predictions.
Contrast	target {external work, theory}	A difference with external work or theory; within-study group comparisons are Evidence.
Open	kind {unresolved question, future direction}	An unresolved question or future research direction identified in the source.
Paper	title, authors, venue, year, identifier; repository and dataset links	The anchor paper's bibliographic record.
CitedPaper	title, authors, venue, year, identifier	A referenced work represented bibliographically and linked to the content citing it.

Relations and attributes. Table 5 specifies scientific relations and their attributes. A Claim *contradicts* a Contrast when it conflicts with the external position recorded there, and *rebuts* it when an explicit argument against that position is given.

Table 5: Scientific-relation signatures and type-specific attributes. All edges additionally carry a source location, supporting text, and an extraction confidence $c(x)$ (§3.2). *formalizes* links a claim to a formulation that summarizes it empirically; *derives* links a formulation to a claim obtained from it by deduction. A dash indicates no type-specific attribute.

Relation	Source type	Target type	Type-specific attributes
produces	Experiment	Evidence	—
requires	Experiment, Mechanism	Assumption	—
supports	Evidence	Claim, Mechanism	polarity $\in \{+, -\}$; causal step (Mechanism targets)
explains	Mechanism	Claim	—
composes	Mechanism	Mechanism	—
entails	Assumption	Claim	strength $\in \{\text{weak}, \text{strong}\}$
limits	Boundary	Experiment	—
constrains	Boundary	Assumption	—
refines	Claim	Claim	—
contradicts	Claim	Claim, Contrast	—
rebuts	Claim	Claim, Contrast	—
grounds	Claim	Assumption	—
formalizes	Claim	Formulation	—
derives	Formulation	Claim	—
guides	Formulation	Experiment	—
updates	Evidence	Assumption	effect $\in \{\text{confirm}, \text{revise}\}$
replicates	Evidence	Evidence	outcome $\in \{\text{agree}, \text{conflict}\}$
supersedes	Evidence	Evidence	—

Bibliographic and citation relations. Citations link the relevant epistemic nodes to the referenced works, recording their roles as support, dependence on prior methods or results, or contrast.

Figures 6–7 show the extraction and relation-construction prompts.

Epistemic Unit Extraction Prompt

```
SYSTEM:
You are a bioinformatics expert extracting epistemic units from a paper.
Node definitions and target sections:
{dimension_definitions_section_targets_and_trigger_phrases}

PROCEDURE:
1. Read the assigned sections and their context through tools.
2. Extract units using the supplied node types. Keep observations separate
   from interpretations.
3. Keep one minimal verifiable proposition per unit. Avoid duplicate
   paraphrases and do not merge distinct propositions.
4. Record each unit's source location and extraction confidence.

USER:
Paper index: {inventory_preview}
Target dimensions: {target_dimensions}
Extract the units and return a JSON array containing each unit's ID,
type, text, source location, citations, and extraction confidence.
```

Figure 6: Selected English instructions for bioinformatics unit extraction. Braced fields denote inputs or omitted definitions.

Epistemic Relation Construction Prompt

```
SYSTEM:
Identify evidence-supported relations among the supplied epistemic units.
Permitted source-target types, relations, and required attributes:
{relation_signatures_and_required_attributes}

PROCEDURE:
1. Examine candidate node pairs for scientific dependencies.
2. Create a relation only when a quotation from the unit text supports it.
3. Select the most precise permitted relation and check its endpoint types
   and required attributes.
4. Record the supporting quotation and extraction confidence.

USER:
Epistemic units: {units_json}
Return a JSON array containing each relation's source and target IDs,
relation type, required attributes, supporting quotation, and confidence.
```

Figure 7: Selected English instructions for identifying and attributing candidate relations. The retained graph follows the representation and confidence filter above. Braced fields denote inputs or omitted definitions.

A.2 Task Sampling

We sample epistemic nodes from the frozen, enriched graph G_a^+ to provide the scientific context for each candidate task. The resulting subgraph G_s retains all scientific edges between selected nodes, their attributes, and associated citation links and bibliographic records. Except for directed path sampling, neighborhood expansion considers both incoming and outgoing edges while preserving their original directions in G_s.

Sampling strategies. Six strategies provide complementary views of the source graph:

1. **Breadth-first.** Expand a seed's neighborhood level by level within node and depth budgets, preferring Claim, Formulation, or Assumption seeds when available.
2. **Top-down.** Rank seeds by incoming degree and node-type priority, favoring Claim, Formulation, and Assumption. Expand from a randomly chosen top-ranked seed, prioritizing Evidence and Experiment neighbors to connect propositions to empirical context.
3. **Citation-anchored.** Rank seeds primarily by citation-edge count, prefer those with scientific neighbors, and expand their neighborhoods to support questions about dependence on prior work.
4. **Dimension-cover.** Sample one node per requested epistemic type, then add nodes and neighbors within the remaining budget. Requested types begin with Claim and Evidence when present and extend to Mechanism, Assumption, Experiment, or Boundary.
5. **Heuristic-ranked.** Rank nodes using a heuristic combining extraction confidence, resolution status, incoming degree, and optional citation weighting, then expand around high-scoring nodes.
6. **Directed path.** Repeatedly sample an outgoing edge uniformly, optionally constrained by a relation sequence, until a dead end or length limit. This follows dependencies such as Experiment $\rightarrow$ Evidence $\rightarrow$ Claim.

Scale and task coverage. For the first five strategies, the automatic planner sets a nominal budget of six to eight epistemic nodes, excluding bibliographic records. Depth limits for breadth-first, top-down, and citation-anchored expansion range from two to five hops; directed paths are capped at five or six steps. These are planning limits, not observed subgraph sizes. Strategy selection and question quotas depend on graph structure, citation availability, and construction feedback. Research-stage weights derived from node and relation types guide coverage across literature review, research plan-

ning, experimental design, experimental analysis, and paper writing. Each slice retains its source, strategy, and seed metadata and is passed to question generation.

A.3 Scientific Question Generation

Figure 8 gives the question-generation prompt.

Scientific Question Generation Prompt

```
SYSTEM:
Generate focused scientific questions from the supplied evidence subgraph.
Require causal, counterfactual, or robustness reasoning rather than recall.

PROCEDURE:
1. Examine the graph relations and supporting evidence, guided by the
   requested research stages.
2. Ground each question in at least one graph relation and cover different
   combinations of epistemic types.
3. Make each question understandable without the source paper and state
   all task requirements explicitly. Withhold reference outcomes and
   conclusions; withheld information must not introduce extra requirements.
4. For computational questions, identify the dataset or data source.
   If the nodes contain dataset_ref, include at least one question requiring
   computation or code verification.

USER:
Generate {max_questions} scientific questions.
Nodes: {nodes_json}
Edges: {edges_json}
Quantitative resources: {quantitative_context}
Stage preferences and weights: {phase_guidance}
Return a JSON array containing each question, its target dimensions, and
research stage. For computational questions, also include
requires_computation: true and data_source.
```

Figure 8: Selected English instructions. Braced fields denote inputs or omitted definitions.

Figure 9 gives the Deep Analysis Agent prompt for constructing computational tasks and their execution environments.

Task and Environment Construction Prompt

```
SYSTEM:
Construct one executable research task from an evidence subgraph with a
linked code repository and input data. Draft the task instruction and
execution environment only; do not write reference solutions or verifiers.

PROCEDURE:
1. Confirm that the subgraph locates a repository implementing the analysis
   and input data for the same scientific work; otherwise return ineligible.
2. Partition each node's attributes into an exposed and a withheld view.
   Experiment nodes expose setup and input data and withhold reference
   outcomes.
3. Choose one objective: extend an analysis, compare methods, or test a
   conclusion under specified conditions. Reject tasks that only rerun a
   precomputed result or require withheld values.
4. Write the instruction from exposed attributes only. Specify inputs,
   protocol parameters, and deliverable paths and formats; state no
   reference results or conclusions. Withheld attributes must not add
   obligations.
5. Write a buildable Dockerfile with pinned dependencies that stages the
   repository and exposed inputs, leaves the output directory empty, and
   excludes reference results and verification scripts.
```

Task and Environment Construction Prompt (continued)

```
6. Statically check path closure, entry points, staged inputs, answer
   leakage, empty outputs, and dependency traceability; record runtime
   check commands for the Auditor Agent.

USER:
Paper: {paper_id}; sampling strategy: {sampling_strategy}; seed: {seed_node}
Nodes: {nodes_json}
Edges: {edges_json}
Repository: {github_refs_json}; data: {data_refs_json}
Return JSON with eligibility, the attribute partition, the task instruction,
and the environment (Dockerfile, files, excluded paths, usability review).
```

Figure 9: Selected English instructions for constructing computational tasks. Braced fields denote inputs or omitted definitions.

A.4 Subdomain-to-Area Mappings

Table 6: Computer Science: subdomains, area assignments, and classification sources.

Subdomain	Area	Classification source and code
Computer vision	Artificial intelligence	ACM CCS 2012 Computer vision; Scopus ASJC 1707 Computer Vision and Pattern Recognition
Intelligent agents	Artificial intelligence	ACM CCS 2012 Intelligent agents; Scopus ASJC 1702 Artificial Intelligence
Natural language processing	Artificial intelligence	ACM CCS 2012 Natural language processing
Information systems	Information systems	ACM CCS 2012 Information systems; Scopus ASJC 1710 Information Systems
Reinforcement learning	Reinforcement learning	ACM CCS 2012 Reinforcement learning; Scopus ASJC 1702 Artificial Intelligence
Sequential decision making	Reinforcement learning	ACM CCS 2012 Sequential decision making (under Reinforcement learning)

Table 7: Biology: subdomains, area assignments, and classification sources.

Subdomain	Area	Classification source and code
Host–pathogen interactions	Host–pathogen interactions	MeSH D054884 Host-Pathogen Interactions; OECD FOS 1.6 Genetics / 3.1 Immunology
Infectious diseases	Infectious diseases	Scopus ASJC 2725 Infectious Diseases; OECD FOS 3.3 Infectious diseases; MeSH Drug Resistance, Microbial
Computational biology	Microbiology	OECD FOS 1.2 Bioinformatics; WoS Mathematical & Computational Biology
Freshwater biology	Microbiology	OECD FOS 1.6 Freshwater biology, limnology
Microbial ecology	Microbiology	MeSH Microbiota; Scopus ASJC 2404 Microbiology; OECD FOS 1.6 Microbiology, Ecology
Microbial systematics	Microbiology	Scopus ASJC 1105 Ecology, Evolution, Behavior and Systematics

Table 8: Chemistry: subdomains, area assignments, and classification sources.

Subdomain	Area	Classification source and code
Electrochemistry	Electrochemistry	Scopus ASJC 1603 Electrochemistry; OECD FOS 1.4 Electrochemistry; IUPAC electrocatalysis
Energy engineering	Energy engineering	Scopus ASJC 2102 Energy Engineering and Power Technology; OECD FOS 2.7 Energy and fuels
Organic chemistry	Organic chemistry	Scopus ASJC 1605 Organic Chemistry; OECD FOS 1.4 Organic chemistry
Drug discovery	Pharmacology and toxicology	Scopus ASJC 3002 Drug Discovery; OECD FOS 3.1 Pharmacology and pharmacy
Toxicology	Pharmacology and toxicology	Scopus ASJC 3005 Toxicology; OECD FOS 3.1 Toxicology

Table 9: Physics: subdomains, area assignments, and classification sources.

Subdomain	Area	Classification source and code
Astronomy and astrophysics	Astronomy and astrophysics	Scopus ASJC 3103 Astronomy and Astrophysics; OECD FOS 1.3 Astronomy (including astrophysics)
Condensed matter physics	Condensed matter physics	Scopus ASJC 3104 Condensed Matter Physics; OECD FOS 1.3 Condensed matter physics; PhySH Condensed Matter & Materials Physics
Geophysics	Earth and environmental sciences	Scopus ASJC 1908 Geophysics; OECD FOS 1.5 Earth and related environmental sciences
Fluid dynamics	Fluid dynamics	PhySH Fluid Dynamics; Scopus ASJC 1507 Fluid Flow and Transfer Processes; OECD FOS 1.3 Fluids and plasma physics
Dark matter	Nuclear and high energy physics	PACS 95.35.+d Dark matter; PhySH Gravitation, Cosmology & Astrophysics
Particles and fields	Nuclear and high energy physics	Scopus ASJC 3106 Nuclear and High Energy Physics; OECD FOS 1.3 Particles and fields physics; PhySH Particles & Fields

A.5 SCORING RULES

Benchmark aggregation. Within each of the RR and RRE modes, we first average task scores within each of the four scientific domains, then average the four domain means with equal weight. Overall is the equally weighted mean of the resulting RR and RRE scores.

Item structure. Each rubric item carries three fields. The *dimension* assigns it to one of the five dimensions in Table 11. The *polarity* is positive if the item describes a requirement the report should meet and negative if it describes an error the report should avoid. The *level* is `must` if the item is necessary and `nice` if it is supplementary. Items are scored as binary judgments: a positive item passes when the report meets the requirement, and a negative item passes when the report does not commit the described error.

The 184 evaluated tasks carry 3,595 rubric items, distributed as EC 1,198, SI 1,101, CQ 465, IF 419 and RC 412. Of these, 68.2% are `must` items and 87.1% are positive.

Dimension scoring. Let $y_{ik} \in \{0, 1\}$ indicate whether criterion k passes. For dimension d, let $B_{i,d}$ be the weighted mean of positive must and positive nice pass rates, with weights 0.75 and 0.25. If only one group is present, its weight becomes one; if neither is present, $B_{i,d} = 1$. Let $F_{i,d}^{-,\mathrm{nice}}$ be the fraction of negative nice criteria that fail, or zero if none are present. The dimension score is

$$J_{i,d} = \begin{cases} 0, & \text{if any must criterion fails,} \\ \mathrm{clip}\big(B_{i,d} - 0.15F_{i,d}^{-,\mathrm{nice}}, 0, 1\big), & \text{otherwise.} \end{cases}$$

The gate applies to both positive and negative must criteria; when no must criteria are present, it does not apply. Here, $\mathrm{clip}(x, 0, 1) = \min(1, \max(0, x))$.

Dimension semantics. EC asks whether the report addresses the specific content the task demands. SI asks whether separate pieces of evidence are combined into a defensible inference. RC asks whether the report keeps the provenance of its claims legible. IF asks whether the report respects the constraints of the instruction. CQ asks whether the report communicates its findings clearly. Table 10 gives one example item per dimension.

Table 10: One example rubric item per dimension.

Dim.	Pol./Level	Criterion
EC	positive, must	Identifies several of the masking mechanisms named in the query as concrete ways in which execution-based gains can be inflated by structurally wrong SQL, rather than discussing execution equivalence only in the abstract.
SI	negative, must	Penalized for reading a prevalence pattern as a true gut-phage signal without ruling out batch or run clustering, blank-control profiles, and index-set or barcode structure: writing that taxa are prevalent across many samples and are therefore real takes co-occurrence for proof of authenticity.
RC	positive, must	Marks which numerical inputs are given by the query or measurement context (139 fb^{-1}; ggF 7%; VBF 12%; luminosity 1.7%) and which are inferred, assumed, or drawn from general LHC knowledge, such as the qualitative size of forward JES and pile-up contributions, so that a reader can tell anchored numbers from reasoning.
IF	positive, must	Keeps the comparison within the scope the query sets: with hardware, NUMA and storage configuration, buffer pool, concurrency, data size and TPC-C transaction mix held fixed, covers DPTC against uncompressed tables, LZ4 page compression and Zstd page compression on throughput, p95/p99 latency and CPU and I/O cost.
CQ	positive, nice	Where the judgment forks on whether the product likelihood already mitigates correlations through shared nuisance parameters, sets the mitigated and unmitigated cases side by side, so that a reader sees which conclusion follows from which, instead of arguing one case and leaving the other in a closing parenthesis.

Table 11: Report rubric dimensions and their weights in J_i.

Dimension	Assesses	Weight
Explicit Criteria (EC)	Task-specific requirements	0.25
Synthesis of Information (SI)	Integrating evidence into conclusions	0.35
References & Citation (RC)	Accurate use of sources	0.25
Instruction Following (IF)	Compliance with instruction constraints	0.05
Communication Quality (CQ)	Clarity and organization	0.10

B Research Harness

The following prompt excerpts retain the core instructions and workflow for delegation, evidence gathering, computation, and synthesis. Research and Coding Agents report findings within their assigned scope; final synthesis is performed by the Coordinator Agent.

B.1 Coordinator Agent

EulerFlow Coordinator Prompt Excerpt

```
ROLE
Delegate research, independently assess coverage, and synthesize findings.
Do not search the web yourself.

WORKFLOW
1. Initialize top-level plan items. Delegate scoped assignments with the
   original question and Plan item ID; parallelize independent subtasks.
2. Wait for all reports. Classify each key dimension as covered, thin,
   or missing; do not rely solely on worker-reported gaps.
3. Record findings and follow up on significant gaps or resolvable critical
   conflicts within the worker-call budget. Include prior findings in
   follow-ups to avoid redundant searches.
4. When coverage is sufficient, close the plan and synthesize findings.
   If limits are reached, use the available findings and preserve
   uncertainties and unresolved conflicts.

COMPUTE TASKS
Create one shared sandbox and name the files each compute item must produce
under /app/results/. Accept a compute report only if its sanity checks pass,
its limitations leave the results usable, and its outputs are saved rather
than only described; trace every computed number in /app/report.md to
/app/results/.

EVIDENCE
Factual claims must trace to worker-reported sources. Use inline [[n]]
citations and source URLs. Distinguish task facts, outside knowledge,
and inference where needed; check citations after merging references.
```

Figure 10: Coordinator prompt excerpt.

B.2 Research Agent

EulerFlow Research Agent Prompt Excerpt

```
ROLE
Research only the assigned scope and return findings; do not perform
final synthesis.

WORKFLOW
1. Map the topic and pursue specific angles. Manage only your child plan
   items, never the parent; skip planning for single-fact lookups.
```

```
2. Search and read primary sources. Try multiple queries before declaring
   an angle exhausted. Cross-check significant claims against at least
   two independent sources; flag those not corroborated.
3. Stop based on coverage. Return findings with inline [[n]] citations
   and source URLs, retaining necessary quantitative details, uncertain
   interpretations, and both sides of unresolved conflicts.
4. If limits are reached, report partial findings and describe remaining
   gaps and attempted searches precisely enough for follow-up.

EVIDENCE
Specific numbers, names, and attributions stated as facts must come from
actual tool results. Distinguish task facts, outside knowledge, and
inference where needed.
```

Figure 11: Research Agent prompt excerpt.

B.3 CODING AGENT

EulerFlow Coding Agent Prompt Excerpt

```
ROLE
Execute only the assigned computational scope in the shared sandbox and
return a computation report; do not perform final synthesis. Reuse the
given sandbox ID; do not create a new sandbox.

WORKFLOW
1. Identify the explicit outputs and the companion quantities a reviewer
   would expect, such as units, uncertainty, and sample size. Manage only
   your child plan items, never the parent.
2. Validate inputs in small steps before running the full computation.
3. Sanity-check significant results by a second method, dimensional
   analysis, or an order-of-magnitude estimate.
4. Save machine-checkable outputs under /app/results/ before reporting;
   save partial outputs if the computation is incomplete.

REPORT
Return Scope, Method, Results with the code behind each key number,
Sanity Checks, Limitations, Artifacts, and Open Questions. A report
without executed sandbox code is invalid; do not present hand-calculated
numbers as verified. Surface errors, warnings, and failed attempts.
```

Figure 12: Coding Agent prompt excerpt.

Figure 13 illustrates how EulerFlow coordinates research and computation on a representative task.

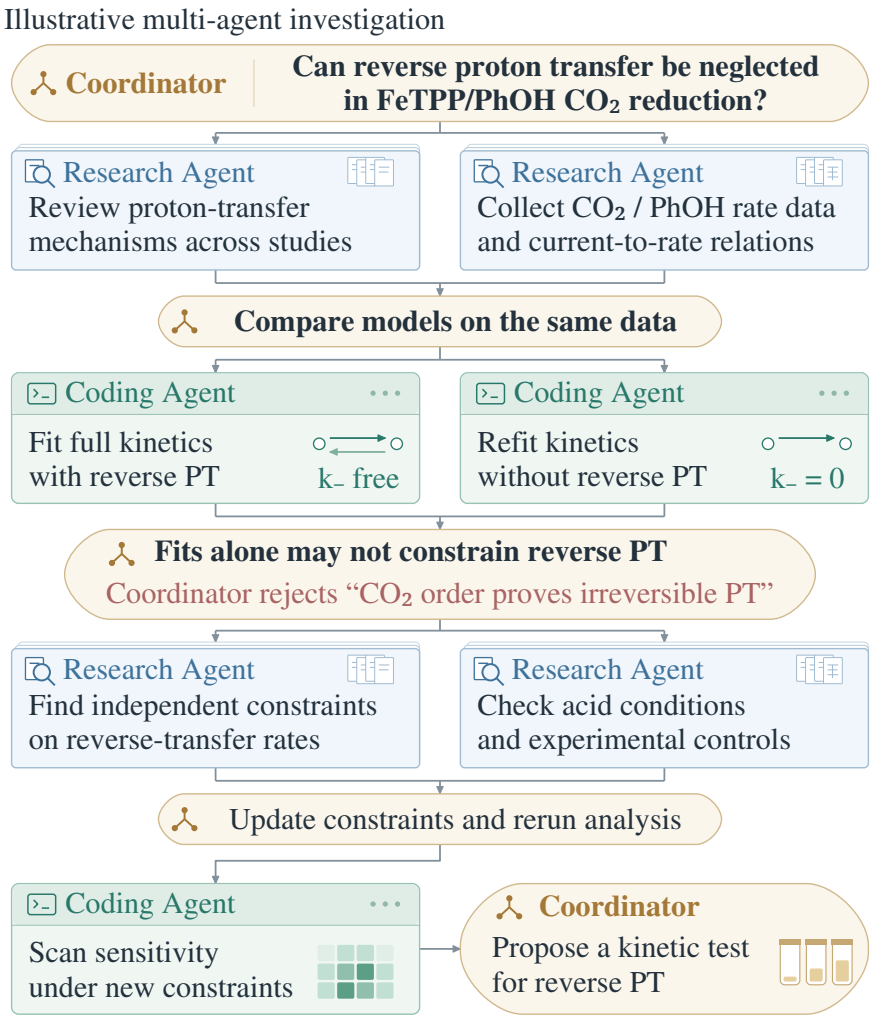


Figure 13: Illustrative EulerFlow workflow for investigating reverse proton transfer in FeTPP/PhOH-catalyzed CO_2 reduction.

C Experimental Details and Additional Results

C.1 Evaluation Setup

Figure 14 gives the baseline search prompt.

Baseline Search Prompt

```
SYSTEM:
Answer the user's question using the provided tools.
Today: {formatted_date}
{tool_definitions_and_call_format}

Start with a concise, numbered plan of distinct subgoals. Work through
them sequentially. After every result, extract relevant evidence,
partial clues, constraints, and keywords, even if it does not directly
answer the question. Revise the plan and revisit overlooked subgoals.

Issue exactly one tool call per message, then stop and wait for the
actual result. Before calling, briefly state what is known, missing,
or unreliable and why the tool is needed. Use self-contained queries;
tools do not retain memory. Verify required parameters and do not
invent missing inputs. Avoid vague or repetitive queries. Use archived
sources when the question concerns information at a past date.

The final answer must be a complete research report with inline [[n]]
citations immediately after sourced factual claims and the actual URLs visited. Do
not end
with process commentary or offers of further help.

USER:
{task_description}
{associated_file_information}

Investigate the question comprehensively using reliable web sources.
Preserve candidate answers, supporting facts, reasoning, intermediate
findings, uncertainties, and conflicting evidence. Do not prematurely
exclude uncertain possibilities or silently resolve disagreements.
Use the most reasonable interpretation of the question. If explicit
requirements are ambiguous or inconsistent, document plausible
interpretations and corresponding findings rather than silently
correcting the task.
{task_hint}

USER (when a final summary is requested):
Do not use further tools. Write the best complete research report
supported by the information collected for:
{task_description}
If the turn or context limit was reached, use the findings available.
Cover the major findings, preserve important details, flag uncertainty,
and present conflicting viewpoints. Include inline [[n]] citations and source
URLs. Omit tool instructions and process commentary.
Write in the language of the original question.
```

Figure 14: Selected English instructions. Braced fields denote inputs or omitted definitions.

Additional runtime settings. Each SciProdBench episode has a budget of five hours and 500 turns. It ends upon submission or when either limit is reached. Models and harness versions are listed in §5.1. For search tasks, the base harness provides Serper search, Jina webpage extraction, and document-reading tools. EulerFlow's search configuration exposes these tools to the Research Agent; the Coordinator receives worker findings and manages the plan.

For the EulerFlow–Claude search runs, the Coordinator is configured for at most 30 turns and each worker execution for at most 50 turns; the Coordinator can dispatch at most four workers per task.

C.2 REPORT JUDGE

Figure 15 gives the report-judging prompt. Rubric semantics and scoring weights are specified in Appendix A.5; the validation experiments follow in Appendix C.3.

Inference settings. For benchmark report judging, we use Gemini 3.1 Pro, accessed as `gemini-3.1-pro-preview`, with temperature 0.2 and `thinking_level=low`.

Inputs and outputs. The judge receives the submitted report and the frozen rubric for that task. The reference solution and construction-time rubric metadata are excluded, and the evaluated system's identity is not explicitly supplied. Rubric items are grouped by dimension, and each dimension is judged in one call: the report is sent once per dimension, and all items of that dimension are decided together. For each item the judge returns a binary decision with a short written reason; the reasons are retained for audit and do not enter the score. The `must`/`nice` level is withheld from the judge and used only when aggregating scores.

Report Judge Prompt

```
SYSTEM:
Audit each report criterion independently with a binary pass/fail verdict.

CRITERION TEXT WINS
Enforce only what the criterion states; add no thresholds or dimension
theories. Dimension and polarity are metadata; if they conflict with the
wording, follow the wording: requirements must be met, and prohibitions
must not be violated.

SPAN AUDIT
1. Split the criterion into its minimal necessary conditions.
2. For each condition, find an explicit supporting span in the report.
   Shared keywords, topics, method names, or citations do not suffice.
3. Required relations, such as distinctions, comparisons, causal links,
   and attributions, must be stated; naming both objects is not enough.
   Enforce literal qualifiers such as "at least" and "simultaneously".
4. Fail if any condition is missing, only implied, or requires the judge
   to supply an inference. There is no partial credit, and strengths
   elsewhere do not compensate. Accept semantic equivalents; conditions
   may be met across passages when their connection is explicit.

EVIDENCE
Citation markers, bibliographies, and URLs alone are not verifiable
support. When a criterion requires it, the report body must distinguish
task facts, outside knowledge, inference, and hypotheses. When unsupported
claims are prohibited, specific numbers, results, mechanisms, and
literature conclusions must trace to cited or task material or be labeled
as hypotheses. Hedging counts only if it limits epistemic status;
professional appearance and relevant terms are not grounds to pass.

OUTPUT
For each item, return pass, confidence (0-1), and a one- or two-sentence
reason naming the supporting span or the missing condition.

USER:
Report: {report_text}
Dimension: {dimension}; items: {item_count}
{criterion_blocks_with_id_dimension_type_and_criterion}
Return only JSON, one result per item. Preserve exact IDs; never omit, merge,
or add items. Order is unrestricted.
{"verdicts": [{"id": "<item ID>", "pass": false, "confidence": 0.9,
  "reason": "<1-2 sentences>"}]}
```

Figure 15: Selected English instructions for criterion-level, evidence-focused report judging. Braced fields denote inputs or omitted definitions.

C.3 JUDGE VALIDATION

For judge-model selection, we compare a standard prompt with an *evidence-focused* variant, which asks whether the reasoning still holds once citations are set aside, and which states that formatting, citation count and confident phrasing are not by themselves grounds to pass. The validation below uses both variants.

Design. We evaluate Gemini 3.1 Pro, GPT-5.4 and Claude Opus 4.8 on 20 matched question pairs: six from physics, eight from chemistry and six from computer science. Each question carries two reports sharing one rubric. The L2 report is designed to preserve scientific content while reducing formatting, hedging its claims, and allowing conversational prose. The L3 report retains polished organization, citations, equations and confident language while introducing scientific defects such as conceptual confusion, unsupported precision, or missing controls. The judge sees neither the L2/L3 labels nor the construction pipeline's own item labels. The 40 reports yield 760 report–criterion decisions per judge and prompt setting, with 380 at each level. Reports and rubrics are held fixed throughout.

The pairs test whether a judge recovers the intended quality ordering despite L3 having more polished presentation. For each question we record the separation gap between the L2 and L3 scores and count an *inversion* when that gap is zero or negative. The two reports in a pair share one rubric, so we summarize each report by its pass rate over that rubric.

Separation. All six judge and prompt settings preserve the intended ordering on all 20 pairs (Table 12). The evidence-focused prompt widens the mean gap for every judge, chiefly by lowering L3, while L2 pass rates stay above 93%. Under this prompt the three judges' gaps lie within 0.45 pp of one another.

Table 12: Matched-pair audit on 20 question pairs. L2 and L3 are rubric pass rates in percent; the gap is L2 minus L3 in percentage points.

Judge	Prompt	L2	L3	Gap	Inversions
Gemini 3.1 Pro	standard	97.34	40.10	57.24	0/20
GPT-5.4	standard	93.01	36.07	56.94	0/20
Claude Opus 4.8	standard	95.90	31.55	64.35	0/20
Gemini 3.1 Pro	evidence-focused	95.99	27.97	68.03	0/20
GPT-5.4	evidence-focused	94.16	26.16	67.99	0/20
Claude Opus 4.8	evidence-focused	93.94	25.50	68.44	0/20

Repeatability. Table 13 summarizes three evaluations of the same 40 reports with the evidence-focused prompt for Gemini and Claude. Both preserve all 20 pair orderings in every run. Across-run SDs of the mean L2 and L3 pass rates are below one percentage point for both judges. At the report–criterion level, 90.7% of the 760 decisions are identical across all three Gemini runs, 95.5% on L2 and 85.8% on L3; the corresponding rate for Claude is 93.4%. Mean within-report score SDs are larger on L3: 2.54 pp on L2 and 7.16 pp on L3 for Gemini, and 1.75 and 4.07 pp for Claude.

Table 13: Three repeated runs with the evidence-focused prompt. Scores and gaps are averages of run means, with across-run SDs in parentheses. Same items denotes unanimous decisions across all three runs.

Judge	L2	L3	Gap	Same items	Inversions
Gemini	95.28 (0.64)	28.86 (0.98)	66.42 (0.79)	90.7%	0/20, 0/20, 0/20
Claude	95.57 (0.40)	24.07 (0.85)	71.49 (1.22)	93.4%	0/20, 0/20, 0/20

Agreement across judges and domains. Aligning the initial evidence-focused decisions by question, report and item, Gemini agrees with GPT-5.4 on 93.0% of the 760 report–criterion decisions and with Claude on 90.4%, and all three order every pair identically (Table 14). Gemini produces no

inversion in any domain in any repeat, covering 6/6 physics, 8/8 chemistry and 6/6 computer science pairs. Domain mean gaps across runs range from 70.07 to 71.87 pp in physics, 73.03 to 78.51 pp in chemistry, and 46.25 to 53.12 pp in computer science.

Table 14: Agreement with Gemini in the initial evidence-focused experiment. Each level contains 380 item decisions; pair ordering uses 20 questions.

Comparison	L2 items	L3 items	All items	Pair ordering
Gemini vs. GPT-5.4	94.7%	91.3%	93.0%	20/20
Gemini vs. Claude	95.0%	85.8%	90.4%	20/20

Judge selection. All three judges recover the intended ordering on these 20 constructed pairs; Gemini and Claude also show stable aggregate scores across repeated evaluations. We use Gemini 3.1 Pro as the report judge. Figure 15 specifies the criterion-level span checks used for benchmark report judging.

C.4 Post-training Details

Supervised fine-tuning uses sequence parallelism, a 5% warm-up ratio followed by cosine learning-rate decay, and weight decay of 0.05. The maximum sequence length is 200K tokens; one epoch corresponds to 1,519 optimization steps.

C.5 External Benchmark Experiments

EulerFlow and WebSwarm* in Figure 4 are our DeepSeek-V4-Pro runs; other results come from the studies cited below. Additional comparisons appear in Appendix C.5.

We use the official evaluation procedures for each benchmark Chen et al. (2026c); Wong et al. (2025); Lan et al. (2025).

Additional comparisons. Tables 15–17 provide additional comparisons of models and agent methods. The WideSearch and DeepWideSearch tables also report additional metrics. All scores are percentages.

Table 15: Additional comparisons on BrowseComp-Plus.

Method / configuration	Model	Accuracy
EulerFlow	DeepSeek-V4-Pro	72.05
Pi-Serini Hsu et al. (2026)	DeepSeek-V4-Pro	71.40
Pi-Serini	GPT-5	74.60
Pi-Serini	GPT-5.5	83.10
Original agent Chen et al. (2026c)	GPT-5	55.90
Original agent	o3	49.28
Original agent	gpt-oss-120B (high)	28.67
XBCP Lu et al. (2026)	DeepSeek-V4-Pro	45.06
AgentIR reference + visit Chen et al. (2026b)	Tongyi-DR	36.87

D Research Task Case Studies

The task, trajectory, and report excerpts below are condensed; rubric judgments and executable checks are selected excerpts.

D.1 RR: designing controls for genetic compensation

Task ID: `q_478fe4e0` **Label:** `biology` **Mode:** RR

Table 16: Additional comparisons on WideSearch.

Method	Model	Row F1	Item F1	SR
This work				
EulerFlow	DeepSeek-V4-Pro	59.91	78.28	14.65
WebSwarm	DeepSeek-V4-Pro	56.47	76.49	8.47
Original benchmark Wong et al. (2025)				
Original single-agent	o3 (high)	34.00	52.61	4.50
Original multi-agent	o3	37.80	57.30	5.10
Fetch-then-Explore Liu et al. (2026)				
Fetch-then-Explore	DeepSeek-V4-Pro	51.40	72.20	—
WebSwarm study Song et al. (2026)				
ReAct	GLM-4.5	33.23	64.61	4.00
InfoSeeker	GLM-4.5	39.98	71.91	4.00
WebSwarm	GLM-4.5	44.14	74.37	7.00
WebSwarm	Qwen3.5-35B	47.54	75.91	—

Table 17: Additional comparisons on DeepWideSearch.

Method	Model	CE Acc.	Col. F1	Row F1	Item F1	SR
This work						
EulerFlow	DeepSeek-V4-Pro	83.90	56.74	25.81	46.69	2.44
WebSwarm	DeepSeek-V4-Pro	87.50	56.52	26.62	47.02	1.56
Search agents Chen et al. (2026a)						
WebSailor	Claude Sonnet 4	70.91	42.01	16.88	32.90	2.39
Flash-Searcher	GPT-5-mini	75.57	42.07	19.36	34.97	2.61
A-MapReduce	GPT-5-mini	79.09	51.78	26.44	42.11	4.43
WebSwarm study Song et al. (2026)						
ReAct	GLM-4.5	—	—	20.08	46.63	3.95
WebSwarm	GLM-4.5	—	—	29.64	58.40	6.58
WebSwarm	Qwen3.5-35B	—	—	35.90	58.34	—
Original benchmark Lan et al. (2025)						
Base LLM (no tools)	DeepSeek-R1	66.93	38.42	10.72	25.01	0.28

Environment

Agent: GPT-5.5, search harness.
Input: the research question below.
Tools: `google_search`, `scrape_and_extract_info`, and `read_file`.
Output: a Markdown research report with inline citations and a reference list.

Task

After Ddx3x deletion, male mice retain a reduced but functional B-cell population, whereas Ddx3y did not provide similar compensation in the reported NK-cell study. Before attributing this contrast to cell lineage, how should control experiments first exclude differences in Cre-mediated deletion timing and residual gene dosage?

Trajectory and report

Trajectory / locating the experimental systems

1. B-cell evidence `google_search`
Query: Ddx3x deletion B cell development male mice Ddx3y compensation functional population Cre

Return: The returned B-cell study describes the male phenotype as consistent with partial compensation by Ddx3y.

2. NK-cell evidence `scrape_and_extract_info`
Source: DDX3X and NK-cell survival. Request genotype, Cre driver, deletion timing, and Ddx3y evidence.

Return: The study uses Ncr1-iCre, with deletion accompanying NKp46/Ncr1 expression in committed NK cells. Ddx3y transcripts are detected, but male Ddx3x-deficient NK cells still show a survival defect.

Trajectory / resolving the timing comparison

3. Check the B-cell driver `scrape_and_extract_info`
Source: Mb1-Cre methods paper. Request the onset and specificity of Cre activity.

Return: Mb1/Cd79a expression begins at the very early pro-B-cell stage, and Cre-mediated recombination extends through subsequent B-cell development.

4. Check the revised B-cell study `scrape_and_extract_info`
Source: Ddx3x B-cell study, revised abstract. Request the Cre models and male phenotype.

Return: The revised study includes Vav1-Cre and Mb1-Cre. It proposes Ddx3y or other male-associated factors as possible explanations for the less severe male phenotype.

Trajectory and report (continued)

Report / proposed experimental design

Question	Proposed control	Readouts	Interpretation
Deletion timing	Use a shared early or inducible Cre system across B/NK cells; compare early and late deletion within each lineage.	Actual recombination timing; survival and function at fixed post-induction intervals.	Loss of the residual B-cell phenotype after late deletion would support a timing or developmental-selection explanation.
Incomplete deletion	Sort Cre-reporter-positive B/NK cells and measure the target locus directly.	Ddx3x recombination PCR/ddPCR, residual RNA, and DDX3X/DDX3Y protein.	Residual DDX3X would leave incomplete deletion as an explanation for surviving cells.
Ddx3y dependence	Compare Ddx3x-deficient male B cells with and without Ddx3y depletion.	Survival and function of the residual B-cell population.	Loss of that population after depletion would support a functional dependence on Ddx3y.
DDX3Y dosage	Test low, intermediate, and high DDX3Y levels in Ddx3x-deficient NK cells; include DDX3X rescue.	Protein levels together with NK-cell survival, maturation, and function.	Rescue at higher DDX3Y levels would support a dosage explanation for the original contrast.

Controls and interpretation. The report includes Cre-negative floxed littermates, Cre-positive Ddx3x-wild-type controls, and a recombination reporter. Whole-tissue RNA reduction is distinguished from deletion in the surviving target cells. Persistence of the B/NK contrast under comparable deletion timing and functional gene dosage would more directly support a lineage-specific explanation.

Selected rubric judgments — LLM judge

Criterion	Judge	Recorded reason
SI-1. Separate lineage effects from deletion timing using comparable timing across lineages or early/late deletion within a lineage.	Met	The report proposes shared early or inducible Cre systems and within-lineage timing comparisons.
SI-2. Separate dosage effects through matched DDX3X/DDX3Y levels or dose–rescue tests.	Met	Dose matching and graded Ddx3y rescue are proposed.
EC-3. Quantify recombination or residual DNA/RNA in target cells at matched stages.	Met	Sorted-cell recombination PCR and RNA measurements are specified.
EC-4. Measure residual DDX3X protein and consider turnover or half-life in interpreting compensation.	Not met	Protein measurements are proposed; turnover and half-life are not discussed.

Scientific significance. This task requires turning a biological contrast into experiments that distinguish competing explanations. The research product is a control design linking deletion timing, functional gene dosage, and cell survival, with explicit readouts for each comparison.

D.2 RRE: INVESTIGATING THE SOURCE OF GRAPH-UPDATE SPEEDUPS

Task ID: `q_0c3df0f7` **Label:** `computerscience` **Mode:** RRE

Environment

Agent: EulerFlow with Claude Opus 4.8.
Execution: Docker sandbox; shared workspace at `/app`.
Tools: plan manager, Research Agent and Coding Agent delegation, and sandbox tools for reading, writing, and executing code.

Inputs and saved outputs / selected paths.

`/app/`	
`+-- data/`	
`| +-- paper_data.json`	Supplied input
`+-- coding/`	
`| +-- analyze_real_data.py`	Item 1 script
`| +-- synthetic_certificate_experiment.py`	Item 2 script
`+-- results/`	
`| +-- mechanism_checks.csv`	Item 1 results
`| +-- operation_timing.csv`	Item 2 timings
`| +-- synthetic_ba_timing.csv`	Item 2 results
`| +-- figures/`	Saved plots
`+-- report.md`	Coordinator Agent report

Task

SC-Insert accelerates edge insertion when maintaining a sparse k-connectivity certificate. Does the gain primarily arise from union-find and single-edge updates avoiding repeated reconstruction, or do graph structure and update patterns amplify the baseline's work?

Align certificate representation, k, insertion sequence, batch size, update order, and execution conditions. Inspect certificate-layer changes, edge movements, union-find operations, candidate scans, memory access, and certificate quality. Deliver executable analysis, a report explaining when its conclusion holds, and three result tables: `operation_timing.csv`, `mechanism_checks.csv`, and `synthetic_ba_timing.csv`.

EulerFlow plan and assignments

Item 0 / Literature grounding. Investigate certificate construction, incremental maintenance, baseline reconstruction, and quality criteria.

Item 1 / Supplied-data analysis. Inspect the schema of `paper_data.json`; distinguish insertion from deletion; pair baseline and dynamic timings; compare speedups with graph properties. Save `mechanism_checks.csv` with graph, speedup, and power-law-exponent fields, and a figure.

Item 2 / Instrumented experiment. Implement forest reconstruction and incremental union-find maintenance. Compare Barabási–Albert (BA) and Erdős–Rényi (ER) graphs at matched vertex counts and target mean degree. Match k, graph, insertion sequence, and execution settings between methods; record operations and timings, and compare final certificate edge sets. Save `operation_timing.csv` and `synthetic_ba_timing.csv`.

Multi-agent trajectory — search and dispatch

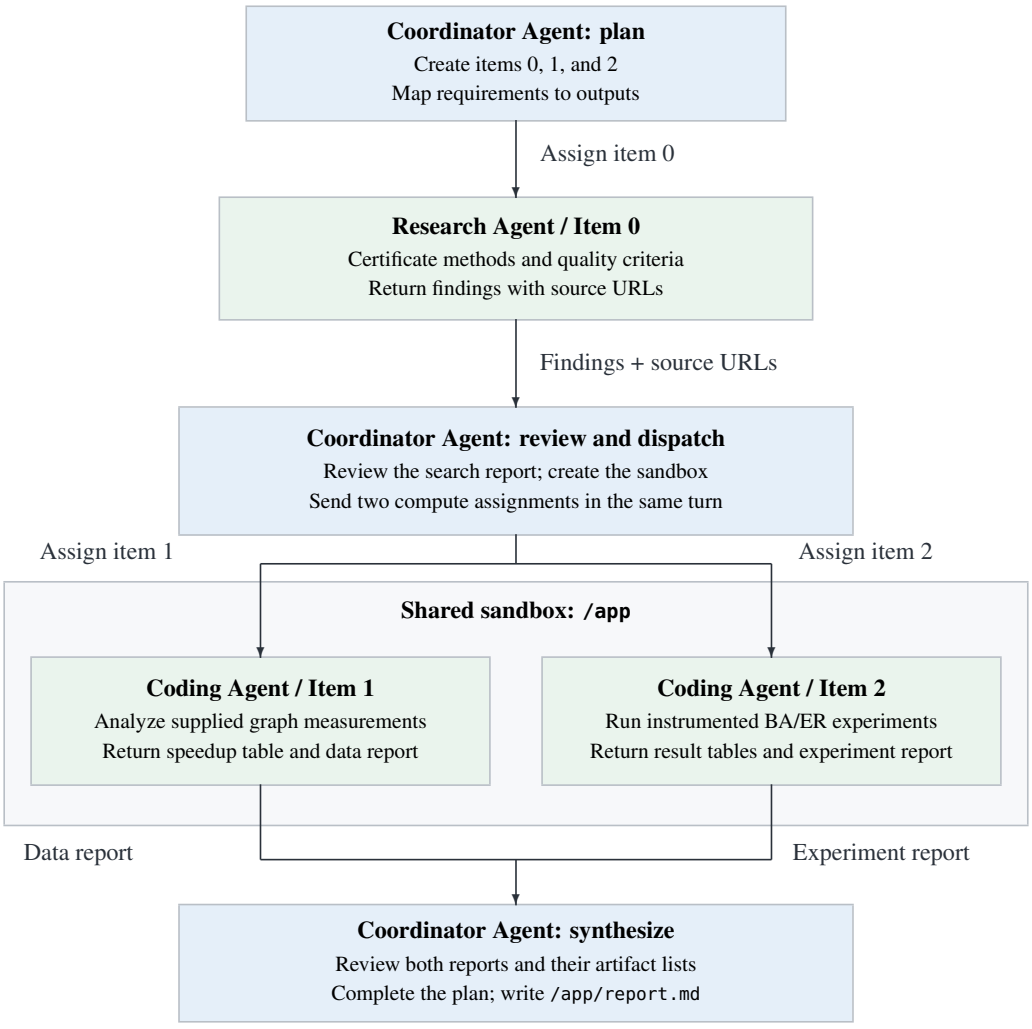
1. Coordinator Agent → Research Agent `agent-worker.execute_subtask`, plan item 0. Request algorithm references, the role of union-find, certificate-quality criteria, and source URLs.

Return: The Research Agent supplies literature on sparse certificates and dynamic maintenance. It also reports that the full primary SC-Insert paper was not accessible through the sources it retrieved.

2. Coordinator Agent → two Coding Agents Create a shared sandbox, mark items 1 and 2 `in_progress`, and issue two `compute-worker.execute_subtask` calls in the same turn.

Return: Separate worker runs start for supplied-data analysis and synthetic experiments. Both receive the research question, retrieved findings, their plan-item ID, the shared sandbox ID, and explicit script and output paths. The Coordinator Agent requests scope, method, results, sanity checks, limitations, artifacts, and open questions, with outputs saved before each worker writes its report.

EulerFlow coordination — assignments and returned reports



Reading the trajectory. Blue boxes show successive stages of the same Coordinator Agent. Green boxes show separate worker runs.

Multi-agent trajectory — Coding Agent / supplied data

Tool calls. `run_python_code` reads `paper_data.json`; `run_command` executes:
`python coding/analyze_real_data.py`

Coding Agent return to the Coordinator Agent. The input has eight records for four graphs. PK, SNAP_email, and KONECT describe insertion; SK describes deletion. The Coding Agent saves `mechanism_checks.csv`, `correlations.csv`, and a figure. Its summary distinguishes the supplied aggregate timings from per-insertion instrumentation.

Saved result excerpt / `mechanism_checks.csv`. Ratios are recomputed from the task-supplied timings in `paper_data.json`.

Graph	Baseline (s)	Insert (s)	Ratio
PK	3.6	0.00048	7500.0
SNAP_email	0.17	0.00021	809.524
KONECT_100pct	0.17	0.00021	809.524

Multi-agent trajectory — Coding Agent / synthetic experiment

Coding Agent's local plan. The Coding Agent adds four children under item 2 for implementation, experiments, result tables, and artifact checks.

Tool calls. `write_file` creates `coding/synthetic_certificate_experiment.py`; `run_command` executes:
`python coding/synthetic_certificate_experiment.py`

Coding Agent return to the Coordinator Agent. The Coding Agent reports 18 configurations across $n \in \{1000, 2000, 4000\}$ and $k \in \{2, 3, 4\}$ for BA and ER graphs, and saves the required timing and synthetic-result tables. All recorded `cert_equal` fields are true.

Saved result excerpt / `synthetic_ba_timing.csv`.

Graph	Certificate edges		Equal sets
	Insert	Rebuild	`cert_equal`
BA_n1000_k2	1992	1992	True
ER_n1000_k2	1965	1965	True

Coordinator Agent review — from worker reports to the final report

The Coordinator Agent reviews the baseline/dynamic pairing, separates the SK deletion workload from the insertion cases, and notes that the supplied file contains aggregate timings rather than internal operation traces.

It reviews the Coding Agent's BA/ER comparisons, certificate-set checks, and saved tables. The Coordinator Agent judges the combined coverage sufficient, marks both compute items complete, and calls `write_report` to produce `/app/report.md`.

Report — Coordinator Agent synthesis

Reported interpretation. The report attributes the observed gains primarily to avoiding repeated forest reconstruction through persistent union-find and localized insertion. It combines the supplied-data speedup table with the synthetic comparison and the recorded certificate-set checks.

Conditions discussed in the report. The implemented comparison uses forest-union certificates, reconstruction after each insertion, and single-threaded execution. The report identifies batched reconstruction as a further comparison and notes that hardware cache misses were not directly measured. Deletion workloads are treated separately from insertion.

Selected rubric judgments — LLM judge

Criterion	Judge	Recorded reason
EC-4. Compare both graph-structure strata and update-sequence strata, such as hub-related, random, temporal, or adversarial updates.	Not met	The report compares graph families but does not propose stratifying the comparison by update sequence.
SI-2. Link internal work metrics to the interpretation of speedup, rather than relying on runtime alone.	Met	The report uses scans, union-find operations, and edge movements to interpret the timing results.
EC-5. Assess certificate quality through required connectivity properties or agreement with trusted recomputation.	Met	The judge credits the report's comparison of final certificate edge sets against reconstruction.
SI-1. Compare mechanism savings and workload amplification, with evidence patterns for each and room for a mixed explanation.	Met	The report sets out the competing explanations and discusses possible additional cache effects on wall-clock time.

Deterministic verifier — selected archived checks

Check	Result	Recorded outcome
`DIM-1-timing`, `DIM-2-mech`, `DIM-3-synth`: required result-table checks.	Pass	The archived verifier passes all three table checks.
`NUM-1-pk-speedup`: PK baseline/insert ratio equals 7,500.	Fail	PK baseline/insert rows are missing from `operation_timing.csv`.
`NUM-2-paper-speedup-min`: paper-reported speedups remain in the hundreds-to-thousands.	Fail	No rows match `source=paper_reported` in `mechanism_checks.csv`.

Artifact correspondence. The Coding Agent for supplied data records the PK ratio in `mechanism_checks.csv`; the synthetic Coding Agent's `operation_timing.csv` contains synthetic-graph timings. The latter lacks the PK rows requested by the ratio check.

Scientific significance. This task makes performance attribution an executable research question: the agent must connect timing gains to internal work while checking the maintained graph object. The plan separates literature, supplied measurements, and controlled computation, while the two evaluation channels inspect the argument and its submitted artifacts.