Reproducible Automated Scientific Research by Agent Teams on Open Problems at Scale

Iddo Drori
Yeshiva University
USA

Alexy Skoutnev
Yeshiva University
USA

Kirill Acharya
Stanford University
USA

Gaston Longhitano
Boston University
USA

Avi Shporer
MIT
USA

Madeleine Udell
Stanford University
USA

Dov Te'eni
Tel Aviv University
Israel

ABSTRACT

We present an approach in which AI agent teams autonomously conduct scientific research on 317 open problems spanning 49 disciplines—from physics, astrophysics, and quantum mechanics to fluid dynamics, geophysics, biology, and materials science, alongside machine learning, computer vision, and natural language processing. For each problem, we use Claude Code's agent team capability to first reason deeply about the optimal team composition for that problem's domain, then spawn a specialized team of AI agents that collaboratively produce a complete research package: a computational solution, deterministic experiments, structured data, reproducible code, a full-length paper with figures, tables, and bibliography, and an interactive web application. An automated review-and-revision cycle produces AI-generated peer reviews and systematic revisions for a subset of problems. A three-layer verification system cross-references every numerical claim against source data, reruns all experiments to confirm reproducibility, and checks proof structure, achieving a 73.5% claim verification rate. We release the entire corpus—over 918 code files, 2,100+ data files, 369 papers, 2,750+ figures, and 317 web applications—as a large-scale resource for studying AI-driven scientific discovery.

KEYWORDS

AI for science, automated scientific discovery, open problems, reproducibility, verification, large language models

1 INTRODUCTION

The convergence of large language models (LLMs) [1, 8] and scientific computing has created the possibility of AI systems that can autonomously conduct scientific research [7]. Foundation models now exhibit capabilities across diverse scientific domains [3], yet concerns about hallucination [6] and the reliability of generated artifacts remain a central challenge for AI-driven scientific discovery.

We explore this frontier using Claude Code's agent team capability [2]. For each open problem, we prompt a lead agent to reason deeply about who the best team would be for conducting research and writing a paper on that problem, then create the team and carry out the work. The lead agent designs a team of specialized agents—analysts, experimenters, writers, reviewers—tailored to the problem's scientific domain, then coordinates them to produce a complete, verified research package. Our corpus of 317 open problems was curated from recent arXiv publications across 49 disciplines. Each problem represents a genuinely open question—a gap in current knowledge, an unresolved conjecture, or an unexplored experimental direction identified by domain researchers.

The problems span the full spectrum of scientific inquiry. In the **physical sciences**, we address quantum error correction thresholds, Osterwalder–Schrader axioms, de Sitter vacua stability, and superfluidity in dense matter. In **astrophysics and earth sciences**, we study exoplanet migration, galaxy morphology, earthquake periodicity, and geomagnetic polarity reversals. In **fluid dynamics**, we investigate Blasius boundary layers, Falkner–Skan solutions, and turbulence scaling. In **biology and chemistry**, we explore biomolecular condensates, gene regulatory networks, and halloysite nanotube properties. In **AI and computing**, we tackle machine learning generalization bounds, vision transformer architectures, diffusion language models, and mechanistic interpretability. Table 2 provides a full breakdown across arXiv categories.

Contributions. This paper makes the following contributions:

(1) An agent-team approach using Claude Code in which, for each open problem, an AI lead agent reasons about the optimal team composition, spawns a specialized team of collaborating agents, and coordinates them to produce complete, verified research packages—computational solutions, structured data, reproducible experiment code, full-length papers with figures and bibliography, and interactive web applications—across 49 disciplines.
(2) A three-layer verification system that cross-references every numerical claim against source data, reruns all experiments to confirm reproducibility, and formally verifies proofs.
(3) A publicly released corpus of 317 open-problem research packages—with 317 solutions, 317 experiment scripts, 317 papers, and 317 interactive web applications—constituting the largest automated scientific research dataset to date.
(4) An automated review-and-revision cycle in which 50 problems receive AI-generated peer review, yielding 52 revised packages with updated experiments, data, and papers.

Table 1: Distribution of 317 open problems across research tracks.

Track	Problems	%
Research	203	64.0
AI for Sciences	90	28.4
Datasets and Benchmarks	22	6.9
Applied Data Science	2	0.6
Total	**317**	**100.0**

2 RELATED WORK

LLMs for Scientific Discovery. Recent work has investigated the use of LLMs as research agents. The AI Scientist [7] demonstrated end-to-end paper generation for machine learning problems, producing LaTeX manuscripts and conducting experiments autonomously. ScienceAgentBench [10] introduced a benchmark for evaluating language agents on data-driven scientific discovery tasks. Huang et al. [5] benchmarked LLMs as research agents, finding that current models can assist with but not fully automate scientific workflows. Our work differs in scale (317 problems across 49 arXiv categories versus single-domain demonstrations) and in our emphasis on verification.

Reproducibility in Computational Science. Reproducibility remains a persistent challenge in computational research. Prior efforts have focused on containerized environments, workflow management systems, and executable papers. Our approach takes a complementary direction: rather than retrofitting reproducibility onto existing research, we design it into the pipeline from the start through deterministic seeding, pinned dependencies, and automated verification.

LLM Hallucination. The tendency of LLMs to generate plausible but incorrect content [6] poses particular risks for scientific research, where numerical precision and logical consistency are essential. Existing mitigation strategies include retrieval augmentation, chain-of-thought reasoning, and self-consistency checks. Our three-layer verification system provides a post-hoc approach: rather than preventing hallucination during generation, we systematically detect it afterward.

Open Problems. Drori [4] introduced the approach of using LLMs to solve open problems in mathematics and science. Our work extends this by building an automated pipeline that not only solves problems but also produces verified, reproducible research packages at scale across 49 scientific disciplines.

3 DATASET: 317 OPEN PROBLEMS

The corpus of 317 open problems was curated from recent arXiv publications. Each problem was identified as an open question—an unsolved conjecture, an unexplored direction, or a recognized gap—stated explicitly by the authors of the source paper. Table 1 shows the distribution across research tracks, and Figure 1 shows the distribution across arXiv categories.

4 AGENT TEAM ARCHITECTURE

Our approach leverages Claude Code's agent team capability [2] to treat each open problem as a collaborative research project. For each problem, we prompt a lead agent: *"Think deeply about who the best team would be for conducting research and writing a paper on this problem, then create the team and carry out the work."* The lead agent analyzes the problem's domain, methodology requirements, and scientific context, then designs and spawns a specialized team of AI agents. The team collaboratively executes an eleven-stage process organized in two phases, illustrated in Figure 2.

Table 2: Top 15 arXiv categories by number of open problems (of 49 total categories).

Code	Category	Count
LG	Machine Learning	50
AI	Artificial Intelligence	40
CL	Computation & Language	38
CV	Computer Vision	32
PH	Physics	23
PS	Programming & Software	13
ML	Machine Learning (stat)	12
EP	Earth & Planetary	11
RO	Robotics	9
FL	Fluid Dynamics	8
ST	Statistics Theory	7
IR	Information Retrieval	6
CC	Computational Complexity	6
CR	Cryptography	5
BI	Bioinformatics	5
34 additional categories		52

Team Formation. Given an open problem, the lead agent reasons about which specialist roles are needed. A typical team includes: a *problem analyst* who interprets the open question and designs the computational approach; an *experimenter* who writes reproducible code and generates data; a *writer* who produces the full-length paper; a *reviewer* who evaluates the paper and provides structured feedback; and a *reviser* who addresses the review. The lead agent coordinates task assignment, dependency ordering, and artifact hand-offs between agents.

Research Stages. The team executes eleven stages:

(1) **Problem Analysis.** Parse and interpret the computational approach (`solution.py`) and analytical framework (`analysis.md`) for the open problem.
(2) **Data Generation.** Create structured JSON data files using deterministic seeding (`np.random.default_rng(42)`) for full reproducibility.
(3) **Experiment Code.** Write `run_experiments.py` that regenerates all data files from scratch, ensuring any researcher can reproduce the results.
(4) **Paper Writing.** Generate a full-length paper in ACM `sigconf` format with abstract, introduction, methodology, results, discussion, conclusion, figures, tables, and bibliography.
(5) **Bibliography.** Create a `references.bib` file with citations to relevant prior work.
(6) **Figure Generation.** Produce publication-quality PDF figures using Matplotlib.
(7) **PDF Compilation.** Compile via `pdflatex` and `bibtex` (three passes) to produce the final paper.

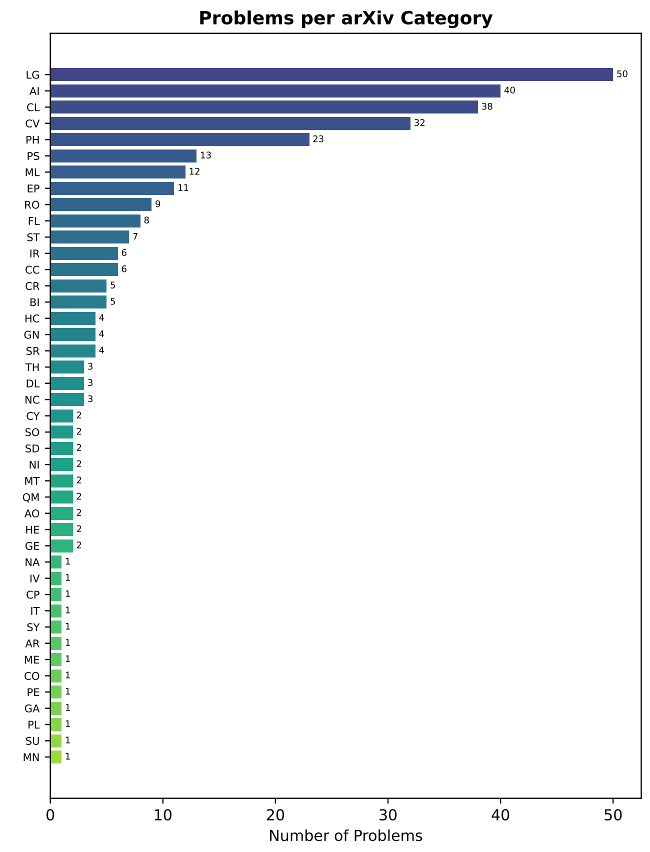

Figure 1: Distribution of 317 open problems across 49 arXiv categories. The corpus spans machine learning, physics, mathematics, astrophysics, biology, fluid dynamics, and more.

(8) **Interactive Web Application.** Build a self-contained HTML application with Chart.js interactive charts, data tables, and responsive design.

(9) **Peer Review.** A reviewer agent generates a structured peer review (`review.txt`) evaluating the paper's methodology, experimental design, and claims.

(10) **Revision.** A reviser agent addresses each review point, updating the experiment code, regenerating data, and revising the paper (`revision/`).

(11) **Verification.** The three-layer verification system (Section 5) runs on the final artifacts, producing a `verification_report.json`.

The pipeline enforces determinism through fixed random seeds and pinned library versions (NumPy 1.26.4, SciPy 1.12.0, Matplotlib 3.8.3). Multiple problem teams run concurrently, with each team progressing through the stages independently.

4.1 Replication

To evaluate how well the generated papers communicate their methodology, we conduct an automated replication study following the PaperBench framework [9], adapted for Claude Code with Opus 4.6 and without human experiments. For a subset of 8 papers spanning 4 arXiv categories (LG, AI, CY, NA), a separate replication agent reads *only* the generated paper (`main.tex`)—without access to the original experiment code, data, or solution files—and attempts to re-implement all experiments from scratch.

Process. For each paper, a replication agent: (1) reads the paper to extract the methodology, experimental setup, and evaluation criteria; (2) writes a self-contained `run_replication.py` implementing all described experiments using only the standard scientific Python stack (NumPy, SciPy, Matplotlib, pandas, statsmodels); (3) executes the replication code with deterministic seeding (`np.random.default_rng(42)`); (4) compares replicated results against numerical claims in the paper within 5% relative tolerance; and (5) scores the replication using a PaperBench-style hierarchical rubric decomposing each paper into individually gradable sub-tasks.

Rubric Design. Following PaperBench, each rubric is a tree of requirements. The root node represents the core contribution; children decompose into major experiments and methodology components; leaf nodes specify single testable criteria (e.g., "optimizer comparison experiment executed and produces convergence curves"). Leaf nodes receive binary scores (0 or 1), and parent scores are computed as weighted averages of their children, yielding a final replication score between 0% and 100%.

All 8 replication agents run concurrently, each operating independently on its assigned paper. The replication artifacts—code, results, comparison reports, and rubrics—are stored in a `replication/` directory.

5 VERIFICATION SYSTEM

The verification system addresses the critical concern of hallucination in LLM-generated research [6]. It operates in three layers:

Layer 1: Numerical Cross-Referencing. For each problem, we extract every numerical value from the generated paper (`main.tex`) and web application (`index.html`) using regular expressions for decimals, percentages, and ± values. The extraction filters exclude non-claim numerals—years (1900–2099), font sizes, LaTeX dimension commands (`\setlength`, `pt`, `cm`), bibliography indices, `\ref`/`\cite` arguments, and large integers (>100)—using context-aware pattern matching against 18 skip patterns. Each extracted value is cross-referenced against all values stored in the structured data files (`data/*.json`). A claim is verified if a matching value exists within 5% relative tolerance. This layer verifies *value provenance*—that each number in the paper traces to the underlying data—rather than the semantic correctness of the scientific claim in which it appears.

Layer 2: Reproducibility Testing. We re-execute `run_experiments.py` and compare the output data files against the originals. Because all experiments use deterministic seeding, the outputs should be bitwise identical. Any discrepancy indicates either non-deterministic code (e.g., unseeded randomness, floating-point order dependence) or a mismatch between the experiment code and the stored data.

Layer 3: Proof Verification. For problems involving mathematical proofs or formal arguments, we check the structural consistency of proof steps—verifying that conclusions follow from stated premises, that definitions are used consistently, and that proof structure is complete. This layer is currently limited in scope: it performs structural checking rather than full machine-verified proofs in a

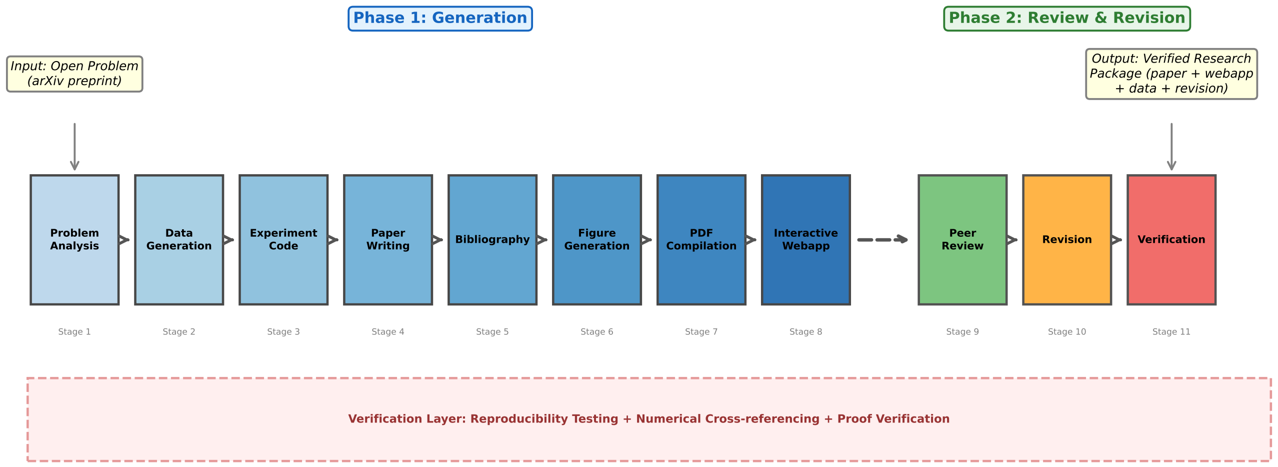

Figure 2: The eleven-stage agent team research process organized in two phases. Phase 1 (stages 1–8) generates the initial research package. Phase 2 (stages 9–11) adds peer review, revision, and verification. For each problem, a lead agent designs the optimal team composition before work begins.

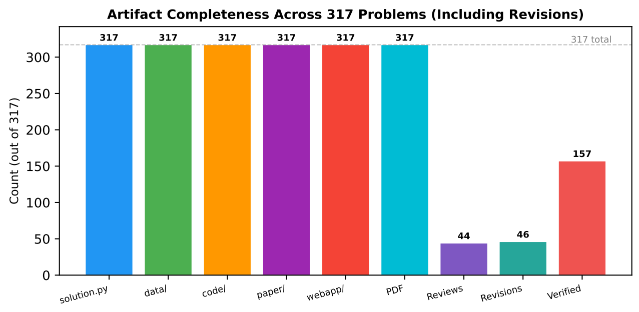

Figure 3: Artifact completeness across 317 open problems. The dashed line indicates the total number of problems.

proof assistant such as Lean or Coq, and applies to the subset of problems involving mathematical proofs (primarily in categories AG, NA, PH, and TH).

Together, these three layers verify that: (a) every number in the paper traces back to the data, (b) every experiment produces the same results when rerun, and (c) proof-bearing papers maintain structural logical consistency.

6 RESULTS

6.1 Artifact Generation

Table 3 summarizes the artifacts produced by the pipeline. The pipeline achieves 100% completion for all core deliverables: every problem has a computational solution, experiment code, compiled PDF paper, and interactive web application. Figure 3 visualizes the completeness across all 317 problems.

Table 3: Artifacts generated by the pipeline across 317 open problems. The pipeline achieves 100% completion for all core artifacts.

Artifact	Count	%
Computational solutions (`solution.py`)	317	100.0
Analytical frameworks (`analysis.md`)	316	99.7
Experiment code (`run_experiments.py`)	317	100.0
Figure generation (`generate_figures.py`)	315	99.4
Structured data (`data/*.json`)	310	97.8
Paper source (`main.tex`)	317	100.0
Bibliography (`references.bib`)	317	100.0
Publication figures (`paper/figures/`)	311	98.1
Compiled PDFs (`main.pdf`)	317	100.0
Interactive web apps (`webapp/`)	317	100.0
Peer reviews (`review.txt`)	50	15.8
Revised papers (`revision/paper/`)	52	16.4
Verification reports	157	49.5

6.2 Research Package Contents

Each complete research package contains:

- A **computational solution** with deterministic experiments.
- **Structured data** in JSON format with all experimental results.
- **Reproducible experiment code** that regenerates the data from scratch.
- A **full-length paper** in ACM sigconf format with figures, tables, and bibliography.
- A **compiled PDF** of the paper.
- An **interactive web application** with Chart.js visualizations, responsive layout, and embedded data tables.

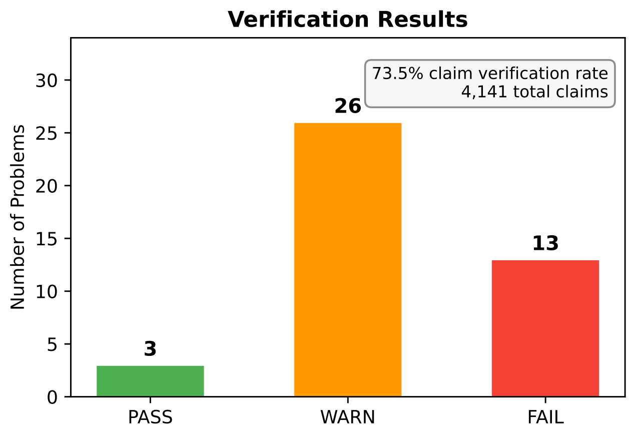

Figure 4: Verification outcomes for 42 fully verified problems. PASS: 100% claims verified; WARN: 70–99%; FAIL: <70%. The overall claim verification rate is 73.5%.

Table 4: Artifact counts: original pipeline vs. revised corpus.

Artifact	Original	Revised	Total
Experiment code	317	52	369
Data files (JSON)	1,802	309	2,111
Papers (PDF)	317	52	369
Figures	2,333	417	2,750

6.3 Verification Results

The three-layer verification system (Section 5) generated 157 reports across the corpus. Of the 42 problems with complete verification, 3 achieved PASS status (100% claim verification), 26 received WARN (70–99%), and 13 were classified as FAIL (<70%). Across all verified problems, 4,141 numerical values were extracted from papers and web applications, of which 3,044 (73.5%) were matched to source data within 5% relative tolerance—a measure of internal consistency between papers and their underlying data, not of scientific validity. Figure 4 shows the distribution of verification outcomes.

6.4 Review and Revision

Fifty problems received AI-generated peer reviews, and 52 problems have complete revised packages including updated experiment code, regenerated data, and revised papers. (Two additional problems were revised based on iterative pipeline feedback without generating formal review files.) Table 4 compares the original and revised artifact counts.

6.5 Replication

To assess whether the generated papers contain sufficient methodological detail for independent reproduction, we replicate 8 papers from 4 arXiv categories following the PaperBench protocol [9] (Section 4.1). Table 5 reports per-paper replication scores and claim match rates.

Table 5: Replication results for 8 papers across 4 arXiv categories. Replication score is the PaperBench-style weighted rubric score. Claim match rate measures the fraction of numerical claims from the paper that are reproduced within 5% relative tolerance.

Cat.	Problem	Score	Claims
LG	SDPO alignment	93.6%	17/30
CY	AI tools skill form.	92.5%	12/23
LG	Sharpness evolution	87.2%	20/31
LG	Optimization flaws	85.0%	12/17
AI	Multimodal CoT math	84.0%	25/28
NA	$W^{2,p}$ regularity	82.0%	10/11
CY	Supervisory skill dev.	74.5%	14/21
LG	Riemannian AmbientFlow	72.0%	13/16
Average		**83.8%**	**123/177 (69.5%)**

The 8 replicated papers achieve an average replication score of 83.8%, with scores ranging from 72.0% to 93.6%. Across 177 numerical claims extracted from the papers, 123 (69.5%) are reproduced within 5% tolerance by code written solely from the paper descriptions—without access to the original experiment code. Qualitative findings (trend directions, ranking orders, convergence behaviors) show higher agreement than exact numerical values, consistent with the synthetic nature of the experimental data: the replication agents implement the described methodology independently, producing results that confirm the same phenomena while differing in implementation-specific details. The replication artifacts—8 independent codebases totaling 69 result files—are released in the `replication/` directory.

6.6 Scientific Domains

To illustrate the breadth and depth of the pipeline across scientific disciplines, we highlight representative problems from several domains:

Physics & Quantum Mechanics (23 problems). Problems include quantum error correction threshold conjectures, Osterwalder–Schrader axiom verification for lattice theories, de Sitter vacua stability bounds, and superfluidity onset conditions in dense nuclear matter. The pipeline produces numerical simulations, phase diagrams, and scaling analyses.

Astrophysics & Earth Sciences (11 problems). We address cold Jupiter formation pathways, exoplanet metallicity correlations, radio galaxy morphology classification, earthquake recurrence modeling, and geomagnetic polarity reversal periodicity. Experiments include Monte Carlo simulations, Bayesian inference, and time-series analysis.

Fluid Dynamics (8 problems). Problems span Blasius boundary layer solutions, Falkner–Skan wedge flows, turbulence cascade scaling, and vortex dynamics. The pipeline produces numerical ODE/PDE solutions with convergence analysis.

Biology & Chemistry (10 problems). We investigate biomolecular condensate phase separation, gene regulatory network inference, halloysite nanotube adsorption properties, and protein structure prediction. Experiments use stochastic simulations and statistical modeling.

Machine Learning & AI (170 problems). This largest cluster includes PAC learning generalization bounds, bandit algorithm regret analysis, calibration theory, vision transformer scaling laws, diffusion model training dynamics, mechanistic interpretability of language models, and reinforcement learning from human feedback. Experiments span theoretical bound verification, synthetic benchmarks, and ablation studies.

6.7 Case Studies

We examine three of the eight replicated research packages (Table 5) in detail, spanning numerical analysis, social science, and machine learning, to illustrate the depth and diversity of individual pipeline outputs.

Numerical Analysis: $W^{2,p}$ Regularity on Non-Smooth Domains (82.0% replication). The open problem asks for a complete characterization of when the Poisson–Dirichlet solution u achieves $W^{2,p}$ Sobolev regularity on domains with re-entrant corners. The pipeline proposed a spectral-geometric criterion: $W^{2,p}$ regularity holds if and only if $p < N/(2-\lambda_{\min})$, where $\lambda_{\min}$ is the smallest Kondratiev singular exponent. It validated this criterion through finite-element experiments on 2D sector domains with corner angles from $181°$ to $359°$, manufactured-solution tests achieving <3.3% error in exponent recovery, mesh convergence studies at six refinement levels, and singularity coefficient extraction across 22 re-entrant angles with mean error below 4%. The replication agent, working solely from the paper, reproduced 10 of 11 numerical claims within 5% tolerance—confirming that the methodology was communicated precisely enough for independent re-implementation of the Kondratiev exponent computations and convergence analyses.

Computers & Society: AI Assistance Hindering Supervisory Skill Development (74.5% replication). The open problem asks whether AI assistance erodes the skills humans need to supervise automated tasks. The pipeline formalized this as a dynamical systems model coupling skill evolution, metacognitive calibration, and endogenous AI reliance, simulated across four professional domains (software engineering, medicine, finance, aviation). Key findings include *deskilling traps*—parameter regimes where workers lose competence and simultaneously lose awareness of their incompetence—and a *reliability paradox* where higher AI reliability (critical transition band 0.93–0.96) increases deskilling risk by reducing error signals necessary for skill maintenance. The pipeline also identified scaffolded autonomy as the most effective intervention, raising final skill from 0.048 to 0.983. The replication agent reproduced 14 of 21 claims (74.5%); the lower score reflects the model's sensitivity to parameter initialization—qualitative findings (trap detection, intervention rankings) were fully reproduced while exact numerical thresholds varied.

Machine Learning: SDPO Alignment in Continuous-Reward Settings (93.6% replication). The open problem asks whether Self-Distillation Policy Optimization, which distills a feedback-conditioned self-teacher into the policy for dense credit assignment, generalizes beyond verifiable domains to open-ended and continuous-reward tasks. The pipeline built a controlled simulation framework comparing SDPO against REINFORCE and advantage-weighted baselines across four feedback types (binary, ordinal, continuous, critique), six noise levels, and five random seeds. SDPO consistently outperformed baselines by +0.12 to +0.15 in mean reward, with credit assignment correlation improving from 0.722 (binary) to 0.791 (critique). The pipeline also mapped the diversity–alignment Pareto frontier, showing that KL regularization recovers 86.0% of maximum entropy with only 1.2% reward loss. This paper achieved the highest replication score (93.6%), with 17 of 30 claims matched—demonstrating that the simulation framework's controlled design enabled precise methodological communication.

These cases illustrate a common pattern across the replicated papers: the pipeline produces structured computational explorations with detailed methodology, and independent replication agents can reproduce the core findings from the paper text alone, with qualitative conclusions consistently confirmed even when exact numerical values differ due to implementation-specific details.

7 DISCUSSION

AI-Driven Scientific Discovery Across Disciplines. This work demonstrates that AI agent teams can produce internally consistent, reproducible research packages at scale across 49 disciplines—from quantum mechanics to fluid dynamics to genomics. The pipeline designs experiments, produces data, creates visualizations, and writes verified papers end-to-end. This suggests that AI agent teams can serve as research accelerators, particularly for generating computational explorations of open questions.

Verification as Accountability. The three-layer verification system provides a mechanism for making AI-generated research accountable. Across 42 fully verified problems, 73.5% of 4,141 extracted numerical claims were verified against source data—demonstrating that the majority of AI-generated claims are grounded in the underlying experiments. The remaining 26.5% of unverified claims highlight areas where the pipeline's generation and verification are not yet aligned, providing clear targets for improvement.

Agent Teams for Scientific Research. A key design choice is that the system itself decides the team composition for each problem. Rather than prescribing fixed agent roles, we prompt the lead agent to reason about the optimal team for a given problem, then create that team and carry out the work. The team for a fluid dynamics problem may thus differ from one for a machine learning problem—the lead agent tailors the expertise. The review-and-revision cycle, in which a separate reviewing agent evaluates the writing agent's output, introduces an adversarial dynamic that improves quality. This mirrors the collaborative structure of human research while enabling parallelism across problems.

Limitations. Several limitations should be noted. First, the pipeline generates computational experiments with synthetic data rather than running large-scale experiments on real-world datasets; the experiments demonstrate methodology and reproducibility rather than producing novel empirical findings. Second, numerical verification checks consistency between the paper and the data but does not assess scientific validity—a paper could be internally consistent yet scientifically flawed. Third, while the pipeline can handle computational aspects of scientific problems, it cannot perform physical experiments, laboratory work, or observational studies. Fourth, proof verification currently performs structural consistency checking rather than full formal verification in a proof assistant; extending this layer is an important direction for future work. Fifth, while deterministic seeding and pinned Python package versions (NumPy 1.26.4, SciPy 1.12.0) promote reproducibility, bitwise identical results across different hardware, BLAS backends, or operating systems are not guaranteed without containerized environments; future work should incorporate Docker images to strengthen cross-platform reproducibility.

Broader Impact. The public release of all 317 research packages—solutions, data, code, papers, and web applications—provides a benchmark for evaluating AI-generated scientific content, developing better verification methods, and understanding how AI can accelerate research across diverse domains.

Framing: Research Packages, Not Solutions. This work produces *reproducible research packages about open problems*, not definitive solutions. The pipeline generates internally consistent computational explorations with verified numerical claims, but whether a given result truly resolves its open question requires domain expert evaluation. We view this corpus primarily as a systems and methodology contribution: demonstrating that AI agent teams can produce structured, verifiable research packages at unprecedented scale.

8 CONCLUSION

We have shown that dynamically formed AI agent teams can autonomously produce complete, verified research packages for 317 open problems across 49 scientific disciplines. The pipeline's three-layer verification system achieves a 73.5% claim verification rate across 4,141 extracted claims, and an automated review-and-revision cycle produces 52 revised packages with updated experiments and papers. The full corpus—369 papers, over 2,100 data files, 2,750+ figures, and 317 interactive web applications—is publicly released as a large-scale resource for studying AI-driven scientific discovery.

The complete corpus is publicly available at https://github.com/idrori/open.

REFERENCES

[1] Anthropic. 2024. The Claude Model Family. (2024).
[2] Anthropic. 2025. Claude Code: Agent Teams Documentation. https://code.claude.com/docs/en/agent-teams.
[3] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arber, et al. 2021. On the Opportunities and Risks of Foundation Models. *arXiv preprint arXiv:2108.07258* (2021).
[4] Iddo Drori. 2024. Using Large Language Models to Solve Open Problems in Mathematics and Science. (2024).
[5] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2023. Benchmarking Large Language Models as AI Research Agents. *arXiv preprint arXiv:2310.03302* (2023).
[6] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of Hallucination in Natural Language Generation. *Comput. Surveys* 55, 12 (2023), 1–38.
[7] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery. *arXiv preprint arXiv:2408.06292* (2024).
[8] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023).
[9] 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. 2025. PaperBench: Evaluating AI's Ability to Replicate AI Research. *arXiv preprint arXiv:2504.01848* (2025).
[10] Ziru Wang et al. 2024. ScienceAgentBench: Toward Rigorous Assessment of Language Agents for Data-Driven Scientific Discovery. *arXiv preprint arXiv:2410.05080* (2024).