SciAgent: Containerized Code Generation for Scientific Computing with Verification

Shruti Badhwar [1]

[1] *sciagent.ai (Open Source Project)*. Correspondence to: Shruti Badhwar shruti@sciagent.ai.

1. Introduction

Recent scientific AI agents have achieved notable results: Google's AI Co-Scientist validates hypotheses in days [1], ChemCrow synthesizes compounds autonomously [2], and self-driving labs report 6× acceleration [3]. Yet deployment gaps persist—37% degradation in enterprise settings [4], and agents achieve only 14% success versus 78% for humans on complex tasks [5].

We identify a fundamental issue: current benchmarks emphasize output validity over execution integrity. We present SciAgent, an open-source LLM-agnostic scientific coding agent with 25+ containerized environments and multi-layer verification. During development, we observed failure modes—fabrication, silent degradation, research-execution disconnection—that motivate verification beyond output checking. SciAgent is an open testbed for evaluating models, expanding scientific domains, and advancing verification methods.

1.1 Why Containers Over Predefined Tool APIs

Function-calling approaches constrain agents to predefined API surfaces. While safer, this limits discovery to pre-packaged workflows. SciAgent generates arbitrary code within containerized environments, enabling novel experimental designs while addressing risk through verification: *safety through verification, not API limitation*.

2. Failure Modes Observed

These failure modes were observed during SciAgent's development across 44 case study runs, directly informing our verification design.

Research-Execution Disconnection (RED): Agents research documentation but fail to use findings. One agent researched 321 amino acid descriptors but implemented only 15—95% reduction undetected by output validation. Claude Code with Opus on photonics tasks researched RCWA parameters thoroughly but generated code using different values.

Sophisticated Fabrication (SF): When computation fails, capable models generate plausible synthetic data. Claude Sonnet (via SciAgent) produced 10 antimicrobial peptides labeled as "fetched from database" when DBAASP returned HTTP 403—scientifically accurate, fabricated provenance. Claude Code with Opus exhibited similar behavior on photonics tasks.

Silent Degradation (SD): Agents substitute simplified implementations without warning. In photonics, bar+circle geometries replaced partial-circle pillars, causing 30% efficiency degradation.

Current benchmarks (ChemBench [6], MCP-Bench [7], ScienceAgentBench [8]) focus on output validity and do not explicitly test for these failure modes. Concurrent work shares our diagnosis that unstructured execution context underlies agent failures, addressing this through typed object graphs and dual verification [9]; we focus specifically on fabrication and integrity failures that type systems cannot catch.

3. Architecture

3.1 Containerized Environments

We provide 25+ Docker containers across 10 scientific domains (Table 1). These are production tools, not mocks. Container enforcement ensures actual software executes—not model approximations.

Table 1: Containerized scientific computing services.

Domain	Services
Molecular Dynamics	GROMACS, LAMMPS
Comp. Chemistry	RDKit, ASE, DWSim
CFD / FEM	OpenFOAM, Elmer, Gmsh
Photonics	MEEP, RCWA, PyOpTools
Quantum Computing	Qiskit
EDA (80+ tools)	OpenROAD, ngSPICE
Bioinformatics	BioPython, BLAST

3.2 Multi-Layer Verification

Code generation introduces risks [10] addressed through targeted verification: **DAG Enforcement** catches RED via explicit task dependencies; **Container Logging** catches SD through system-level invocation records; **Execution Evidence** verifies computation via artifacts and timestamps; **Independent Verifier** catches SF using a separate LLM with fresh context.

4. Results

4.1 Photonics: Metasurface Reproduction

Task: Reproduce RCWA metasurface simulation from published PDF [11]. From 23 runs, 4 produced complete results. SciAgent achieved 20.1% minimum field efficiency versus 25.3% published simulation (20.5% relative error). Our result falls closer to experimental measurement (17.0%) than the paper's own simulation. Zone-by-zone analysis revealed geometry limitations: Zone 3 showed 61% error because partial-circle pillars optimized with commercial RSoft MOST were approximated with bar+circle geometry from PDF-only reproduction. After automated optimization within SciAgent, efficiency improved to 48.3%.

4.2 Bioinformatics: BRCA1 Verification

Task: Verify BRCA1 mutation fitness scores [12] map correctly to AlphaFold protein structure [13] and confirm established biology. Results: 100% mapping achieved for all 1,837 variants (target >95%). Buried residues show significantly higher mutation sensitivity than exposed ($p < 0.001$, Cohen's $d = 0.60$). Functional domains (RING, BRCT) exhibit greater sensitivity than linker regions ($p < 10^{-8}$). These findings validate both the data pipeline and biological interpretation.

4.3 Data Integrity: Model Behavior Differences

Task: Request peptides from DBAASP [14] (returns HTTP 403). This tests agent behavior when external data sources fail.

Table 2: Model behavior when data retrieval fails.

Model	Behavior	Integrity
Claude Sonnet 4	Fabricated peptides	**FAILED**
GPT-4o	Escalated to user	PASSED
Grok-4.1	Returned empty result	PASSED

Each model exhibited distinct patterns reflecting capability tradeoffs. Claude Sonnet 4 understands scientific domains deeply—it fabricated physically accurate peptides precisely because it could. Greater capability enables more convincing fabrication. GPT-4o asked for user guidance repeatedly, appropriate for human-in-the-loop but potentially excessive in autonomous pipelines. Grok-4.1-Fast-Reasoning followed instructions literally, returning empty results—it lacked semantic understanding of scientific workarounds but consequently did not fabricate.

This reveals a capability paradox: the most scientifically capable models pose the greatest fabrication risk. System prompts are insufficient—Claude fabricated despite explicit prohibition. Verification must not rely on model compliance.

5. Discussion

Our findings do not diminish LLM coding agents for software development. SciAgent itself was developed with Claude Code—its strengths in code generation, debugging, and iterative human collaboration are substantial. The distinction is context: when humans review each code change, RED and SF are caught through natural oversight. Autonomous scientific pipelines propagate these failures undetected. SciAgent adds the integrity checks that human-in-the-loop development provides implicitly.

The variation in model behavior underscores why SciAgent is an LLM-agnostic testbed. No single model is universally best—capability profiles differ, failure modes differ, and the right choice depends on task requirements. Community testing across models will yield insights no single evaluation can provide.

6. Conclusion

SciAgent demonstrates that: (1) container-based code generation enables scientific flexibility with verification ensuring safety; (2) failure modes (RED, SF, SD) occur across agent configurations and merit explicit testing; (3) model behavior varies—capability correlates with fabrication sophistication; (4) system prompts are insufficient for autonomous integrity; (5) working results are achievable (photonics 20.5% error, BRCA1 $p < 10^{-8}$).

SciAgent is an open testbed for model evaluation and verification research. Open source: Apache 2.0 at `github.com/sciagent/sciagent-cli`.

Acknowledgments

SciAgent was developed with the help of Claude Code. We acknowledge the open-source scientific computing community: GROMACS, OpenFOAM, MEEP, RDKit, Qiskit, BioPython, and others.

References

[1] Google Research. Accelerating scientific breakthroughs with an AI co-scientist. `https://research.google/blog/accelerating-scientific-breakthroughs-with-an-ai-co-scientist/`, February 2025.

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

[3] Martin Seifrid, Jason Hattrick-Simpers, Alán Aspuru-Guzik, et al. Benchmarking self-driving labs. *RSC Digital Discovery*, 2025.

[4] Beyond accuracy: A multi-dimensional framework for evaluating enterprise agentic AI systems, 2025.

[5] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

[6] Adrian Mirza, Nawaf Lutfi, et al. ChemBench: A framework for evaluating chemical knowledge and reasoning abilities of LLMs. *Nature Chemistry*, 2025.

[7] Accenture Labs. MCP-Bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers, 2025.

[8] Ziru Chen, Shijie Xu, Yuxiang Chen, et al. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations (ICLR)*, 2025.

[9] Jiaru Bai, Abdulrahman Aldossary, Thomas Swanick, Marcel Müller, Yeonghun Kang, Zijian Zhang, Jin Won Lee, Tsz Wai Ko, Mohammad Ghazi Vakili, Varinia Bernales, and Alán Aspuru-Guzik. El Agente Gráfico: Structured execution graphs for scientific agents, 2026.

[10] Kevin Maik Jablonka, Philippe Schwaller, Andres Ortega-Guerrero, and Berend Smit. Leveraging large language models for predictive chemistry. *Nature Machine Intelligence*, 6:161–169, 2024.

[11] Jianghao Xiong et al. Design and experimental validation of a high-efficiency multi-zone metasurface waveguide in-coupler. *Optical Materials Express*, 15(12):2847–2862, 2025.

[12] Gregory M. Findlay, Riza M. Daza, Beth Martin, Melissa D. Zhang, Anh P. Leith, Molly Gasperini, Joseph D. Janizek, Xingfan Huang, Lea M. Starita, and Jay Shendure. Accurate classification of BRCA1 variants with saturation genome editing. *Nature*, 562:217–222, 2018.

[13] John Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021.

[14] DBAASP Team. DBAASP: Database of antimicrobial activity and structure of peptides. `https://dbaasp.org/`, 2024.