Test-Time Self-Evolution in Multi-Agent Systems for Materials Discovery

Siyu Liu [1,2] * **Bo Hu** [1] * **Beilin Ye** [1] * **He Cao** [3] **David J. Srolovitz** [1,2] **Tongqi Wen** [1,2]

*Equal contribution [1]*Center for Structural Materials, Department of Mechanical Engineering, The University of Hong Kong, Hong Kong, China* [2]*Materials Innovation Institute for Life Sciences and Energy (MILES), HKU-SIRI, Shenzhen, China* [3]*International Digital Economy Academy (IDEA), Shenzhen, China*. Correspondence to: Tongqi Wen tongqwen@hku.hk.

1. Introduction

The integration of large language models (LLMs) with materials science shows great potential, but existing materials-science agent frameworks mainly rely on short-term conversational memory and static tool use to complete tasks. They still face challenges in answering questions that require new, unseen tools and in reusing previous knowledge across different research tasks. In this study, we propose a multi-agent framework that enables materials-science agent systems to *self-evolve at test time* through the evolution of agent tools and memory, continuously accumulating usable new skills. Our central innovation is a training-free mechanism that self-evolves at inference time. By combining test-time reasoning optimization with feedback from a sandbox environment, our agents can dynamically master new tools, summarize tools into skills text, and fix code scripts in real time, all without updating model parameters. Furthermore, we introduce a cross-session memory architecture that encapsulates successful research plans, tool configurations, and executable code snippets, enabling retrieval and adaptation of prior solutions in later, unrelated studies. Experiments on two real-world materials simulation benchmarks, MatTools [1] and Sol27LC [2] demonstrate that our system can successfully accumulate knowledge and reuse prior experience across tasks, and that the acquired memories and skills improve task completion accuracy. This framework paves the way for truly autonomous scientific assistants that improve continuously through use, supporting lifelong learning without retraining.

2. Substantial section

We introduce a self-evolving multi-agent architecture that forms a closed loop among reasoning, tool execution, and memory consolidation. In Fig. 1, a coordinator agent receives the user query, decomposes it, and retrieves relevant tools and previously learned *skills* from an agent memory that stores reusable procedures and prior insights. Specialized agents conduct deep research via web/network access and run code or simulations in a sandboxed environment, enabling safe experimentation, error correction, and tool refinement. The coordinator agent invokes specialized agents to execute decomposed tasks through a dedicated task tool to keep the coordinator context compact. Finally, execution traces and validated outcomes are distilled into new skills and memories and written back to memory, while key findings are returned to the coordinator for response synthesis. This design replaces static tool calling with a cumulative skill and memory loop, so the system can reuse and adapt knowledge across tasks and sessions.

We evaluate our system on MatTools (49 tasks, 138 subtasks), which tests whether an LLM can *write*, *execute*, and *verify* sandboxed `pymatgen`-based Python code for realistic materials property calculations. We run three rounds: Round 1 starts with an empty memory; Round 2 reruns after updating memory from Round 1 sandbox feedback (e.g., output-vs-ground-truth comparisons without revealing the correct code); Round 3 repeats this update-and-rerun procedure from Round 2. As shown in Fig 2, accuracy improves from 51.02% to 65.31% (tasks) and 62.32% to 75.36% (subtasks), while the runnable rate increases from 97.96% to 100%. Table 1 shows memory updates also reduce cost: total token usage drops by 30.8% in Round 2 (even accounting for memory writing), then rises slightly in Round 3 (+11.0%) due to richer memory, reflecting a maintenance–efficiency trade-off while retaining net gains. Runtime decreases monotonically (-10.8% from Round 1 to 2 and -13.6% from Round 2 to 3), yielding a cumulative -22.9%. Overall, test-time self-evolution yields consistent improvements in both correctness and efficiency (up to -30.8% tokens and -22.9% runtime).

We further evaluate the framework on Sol27LC, a benchmark of 27 elemental solids with experimental lattice constants spanning FCC, BCC, and diamond structures. Each task computes the equilibrium lattice constant via DFT EOS fitting with ABACUS [3], an open-source first-principles package typically unfamiliar to general-purpose LLMs. Tasks with the same crystal structure share a memory system; the first task in each family is a cold start (FCC Cu, BCC Li, diamond C), and later tasks reuse accumulated memories/skills. In Fig. 3, the first round yields 22 *Correct*, 1 *Partially Correct*, 4 *Error*. *Correct* denotes <5% deviation from experiments; *Partially Correct* indicates correct EOS but a unit inconsistency (ABACUS uses Bohr but experiments are in Å); and *Error* indicates failed runs or large deviations. After rerunning only the failed cases in Round 2, performance improves to 25 *Correct*, 2 *Partially Correct*, 0 *Error*. Trajectory analysis shows that memory reduces repeated failures. In the three cold-start tasks, SCF frequently fails with default wavefunction initialization with SG15 ONCV pseudopotentials [4], while setting init_wfc=random improves convergence. Once distilled into memory as a reusable skill, this fix prevents the same failure mode in >90% of subsequent tasks across the families

(Table 2). Overall, the results in Table 2 and Fig. 3 indicate robust solving of real-world simulations and test-time self-evolution generalizes execution-level knowledge across chemically distinct materials, substantially reducing repeated failures without external intervention.

2.1 Related work

Recent LLM-driven materials agents automate DFT/MD by coupling high-level planners with domain tools. Representative systems translate natural-language goals into executable DFT workflows and coordinate sub-agents for structure generation, convergence checks, HPC scheduling, and error handling (e.g., MASTER [5], DREAMS [6]). For MD, role-based agents generate, run, and iteratively refine LAMMPS inputs (MDAgent [7]). Other efforts integrate retrieval/reasoning/tool use for first-principles calculations or shared-memory expert teams for end-to-end atomistic simulations [8, 9], while platforms like Bohrium+SciMaster focus on robust orchestration [10]. However, most frameworks still rely on static tool sets and workflow-scoped memory, limiting on-the-fly tool creation and reusable skills across tasks.

Beyond static orchestration, self-evolving agents adapt their internal state at test time from interaction trajectories and feedback to improve future performance [11]. Prior surveys summarize self-evolution along several dimensions, including updates to prompts/memory, tool creation/refinement, and agent architectures or collaboration patterns [12]. In materials science, CASCADE explores consolidated memory retrieval for long-horizon tasks [13], and Lu et al. study test-time scientific tool generation from capability gaps [14]. Yet few systems unify these dimensions in a single materials agent that can manage context, create tools on the fly, and accumulate transferable skills across sessions. We address this gap by jointly evolving context, tools, and agent architecture during inference, enabling continual improvement without parameter updates.

2.2 Figures and tables

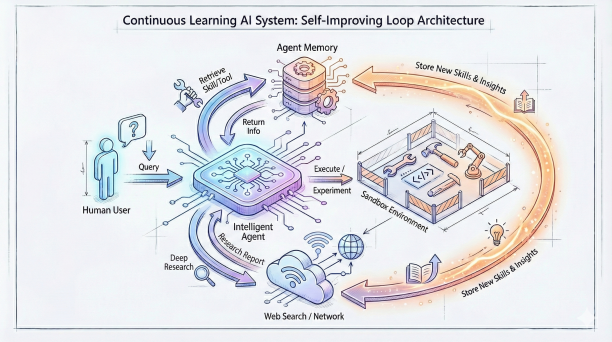

Fig. 1: Schematic diagram of our multi-agent system architecture.

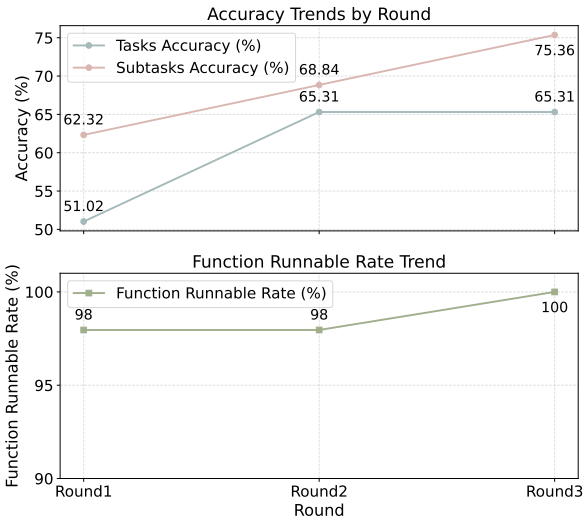

Fig. 2: Accuracy and function runnable rate across three rounds on the MatTools benchmark.

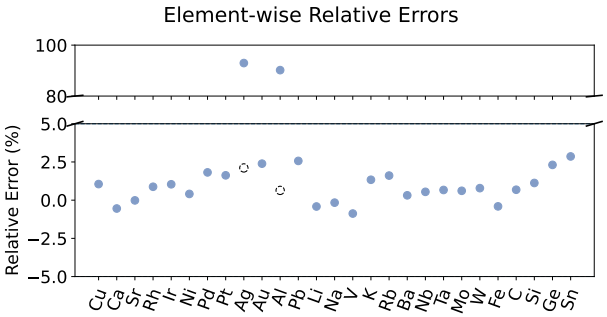

Fig. 3: Element-wise relative errors on the Sol27LC benchmark: dashed points represent the values with correct units.

Table 1: Token usage and runtime statistics across three rounds. Total tokens refer to the combined usage incurred by testing the MatTools benchmark and performing memory updates.

Round	Total Tokens	Memory Update	Δ Tokens (%)	Time (s)	Δ Time (%)
1	719,050	0	–	4583	–
2	497,475	140,435	-30.8	4089	-10.8
3	552,438	103,367	+11.0	3533	-13.6

Table 2: Task outcomes and memory-based error prevention across crystal structures on the Sol27LC benchmark.

Structure	#Tasks	Round 1 (C/P/E)	Round 2 (C/P/E)	Error Avoidance (%)
FCC	12	8/1/3	10/2/0	90.9
BCC	11	10/0/1	11/0/0	90.0
Diamond	4	4/0/0	4/0/0	100.0
Total	27	22/1/4	25/2/0	91.7

Acknowledgments

The work described is partially supported by a grant from the NSFC/RGC Joint Research Scheme sponsored by the Research Grants Council of the Hong Kong Special Administrative Region, China and the National Natural Science Foundation of China (Project No. N_HKU767/25). The authors would like to thank for startup funding from Materials Innovation Institute for Life Sciences and Energy (MILES), HKU-SIRI in Shenzhen for support of this manuscript. This work is partially supported by Research Grants Council, Hong Kong SAR through the General Research Fund (17210723, 17200424). T. W. acknowledges additional support by The University of Hong Kong (HKU) via seed funds (2509100468) and Guangdong Natural Science Fund (2025A1515012129).

References

[1] Siyu Liu, Bo Hu, Beilin Ye, Jiamin Xu, David J. Srolovitz, and Tongqi Wen. Mattools: Benchmarking large language models for materials science tools. *arXiv:2505.10852*, 2025.

[2] Jess Wellendorff, Keld T. Lundgaard, Andreas Møgelhøj, Vivien Petzold, David D. Landis, Jens K. Nørskov, Thomas Bligaard, and Karsten W. Jacobsen. Density functionals for surface science: Exchange-correlation model development with bayesian error estimation. *Phys. Rev. B*, 85:235149, 2012.

[3] Mohan Chen, G-C Guo, and Lixin He. Systematically improvable optimized atomic basis sets for ab initio calculations. *Journal of Physics: Condensed Matter*, 22(44):445501, 2010.

[4] D. R. Hamann. Optimized norm-conserving vanderbilt pseudopotentials. *Phys. Rev. B*, 88:085117, 2013.

[5] Samuel Rothfarb, Megan C. Davis, Ivana Matanovic, Baikun Li, Edward F. Holby, and Wilton J. M. Kort-Kamp. Hierarchical multi-agent large language model reasoning for autonomous functional materials discovery. *arXiv:2512.13930*, 2025.

[6] Ziqi Wang, Hongshuo Huang, Hancheng Zhao, Changwen Xu, Shang Zhu, Jan Janssen, and Venkatasubramanian Viswanathan. Dreams: Density functional theory based research engine for agentic materials simulation. *arXiv:2507.14267*, 2025.

[7] Zhuofan Shi, Chunxiao Xin, Tong Huo, Yuntao Jiang, Bowen Wu, Xingyue Chen, Wei Qin, Xinjian Ma, Gang Huang, Zhenyu Wang, and Xiang Jing. A fine-tuned large language model based molecular dynamics agent for code generation to obtain material thermodynamic parameters. *Scientific Reports*, 15(1):10295, 2025.

[8] Zeyu Xia, Jinzhe Ma, Congjie Zheng, Shufei Zhang, Yuqiang Li, Hang Su, P. Hu, Changshui Zhang, Xingao Gong, Wanli Ouyang, Lei Bai, Dongzhan Zhou, and Mao Su. An agentic framework for autonomous materials computation. *arXiv:2512.19458*, 2025.

[9] Aikaterini Vriza, Uma Kornu, Aditya Koneru, Henry Chan, and Subramanian K. R. S. Sankaranarayanan. Multi-agentic ai framework for end-to-end atomistic simulations. *Digital Discovery*, 5:440–452, 2025.

[10] Linfeng Zhang, Siheng Chen, Yuzhu Cai, Jingyi Chai, Junhan Chang, Kun Chen, Zhi X. Chen, Zhaohan Ding, Yuwen Du, Yuanpeng Gao, Yuan Gao, Jing Gao, Zhifeng Gao, Qiangqiang Gu, Yanhui Hong, Yuan Huang, Xi Fang, Xiaohong Ji, Guolin Ke, Zixing Lei, Xinyu Li, Yongge Li, Ruoxue Liao, Hang Lin, Xiaolu Lin, Yuxiang Liu, Xinzijian Liu, Zexi Liu, Jintan Lu, Tingjia Miao, Haohui Que, Weijie Sun, Yanfeng Wang, Bingyang Wu, Tianju Xue, Rui Ye, Jinzhe Zeng, Duo Zhang, Jiahui Zhang, Linfeng Zhang, Tianhan Zhang, Wenchang Zhang, Yuzhi Zhang, Zezhong Zhang, Hang Zheng, Hui Zhou, Tong Zhu, Xinyu Zhu, Qingguo Zhou, and Weinan E. Bohrium + scimaster: Building the infrastructure and ecosystem for agentic science at scale. *arXiv:2512.20469*, 2025.

[11] Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, Zhaochun Ren, Nikos Aletras, Xi Wang, Han Zhou, and Zaiqiao Meng. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv:2508.07407*, 2025.

[12] Huan ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Qihan Ren, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *arXiv:2507.21046*, 2026.

[13] Xu Huang, Junwu Chen, Yuxing Fei, Zhuohan Li, Philippe Schwaller, and Gerbrand Ceder. Cascade: Cumulative agentic skill creation through autonomous development and evolution. *arXiv:2512.23880*, 2025.

[14] Jiaxuan Lu, Ziyu Kong, Yemin Wang, Rong Fu, Haiyuan Wan, Cheng Yang, Wenjie Lou, Haoran Sun, Lilong Wang, Yankai Jiang, Xiaosong Wang, Xiao Sun, and Dongzhan Zhou. Beyond static tools: Test-time tool evolution for scientific reasoning. *arXiv:2601.07641*, 2026.