CAN AI SCIENTISTS COORDINATE AT RUNTIME?

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-agent AI scientists have shown improving performance across a diverse range of tasks. Yet a common approach is design-time agentic orchestration, which typically relies on fixed workflows. In contrast, human scientists coordinate and adjust their division of labor at runtime. We therefore ask: can AI scientists also coordinate at runtime? To this end, we introduce Runtime Agent Coordination (RAC), which selects agents from existing AI-scientist hosts during execution, assigns scoped work contracts, and provides artifact-grounded verification. Verification informs subsequent agents without blocking transitions or discarding artifacts. We evaluate one shared implementation across Agent Laboratory, EvoScientist, and ARK on ResearchClawBench under matched models, tools, permissions, and run-level budgets. Four cumulative conditions separate native execution, runtime communication, runtime selection, and the combined addition of contracts and verification. Our results highlight the potential of runtime coordination to improve research quality, while revealing host-dependent trade-offs between coordination overhead and performance under matched budgets.

1 INTRODUCTION

AI scientists automate increasingly large parts of the research lifecycle, including idea generation, literature synthesis, experiment design, implementation, analysis, and paper writing (Lu et al., 2024; Yamada et al., 2025; Schmidgall et al., 2025; Goel et al., 2025; Gottweis et al., 2025; Yang et al., 2026). Multi-agent systems distribute these activities across specialized roles that exchange prompts, critiques, code, and research artifacts. This communication determines what agents share. Coordination is a separate decision: which agent should act next, what it should produce, whether its output is valid, and how the run should continue.

Many prominent AI-scientist systems organize agent interactions through predefined stages, loops, and fallback rules (Lu et al., 2024; Yamada et al., 2025; Tang et al., 2025; Schmidgall et al., 2025; Xin et al., 2026). Search, repair, and debate add flexibility within these workflows, but do not necessarily let the system select the most relevant agent across stages (Yamada et al., 2025; Liu et al., 2026). Other systems, such as EvoScientist (Lyu et al., 2026), already support supervisor-directed delegation. These differences motivate us to investigate whether a shared runtime coordination approach can improve research performance across hosts with varying degrees of native flexibility.

Selecting the next agent based on the current research state matters because long-horizon scientific failures are state dependent. A missing result calls for an experimenter, a broken bibliography calls for a deterministic check or writer, or a contradictory figure may require code rather than another prose revision. Recent evaluations locate substantial AI-scientist failures in experimental design, grounding, and scientific reasoning rather than only in software crashes (Xu et al., 2026; Fei et al., 2026). A fixed workflow cannot remove these failures merely by adding more agents. It must either execute all contingencies, which increases cost, or omit contingencies that later turn out to be necessary.

We therefore ask whether runtime coordination is a useful alternative to fixed workflows for multi-agent AI scientists. To address this question, we propose Runtime Agent Coordination (RAC), illustrated in Figure 1, which uses current artifacts, open problems, execution history, and remaining budget to select an authorized existing agent. Scoped contracts define what each selected agent should accomplish, while artifact-grounded verification provides feedback on its output for subsequent work. Every verdict retains the workspace and host transition. We apply the same framework

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.

across different hosts under matched budgets, keeping their underlying agents and tools fixed to assess the benefits of communication and coordination.

Our core contributions are threefold. First, we formally study runtime coordination in multi-agent AI scientists and formulate next-agent selection as a state-dependent decision, distinguishing it from communication under native workflow control. Second, we instantiate the RAC framework in multi-agent AI scientists such as Agent Laboratory, EvoScientist, and ARK without replacing their native agents, tools, or permissions. Third, we empirically test whether runtime coordination improves research performance under matched budgets, reporting research scores, token usage, and estimated inference costs across a range of tasks.

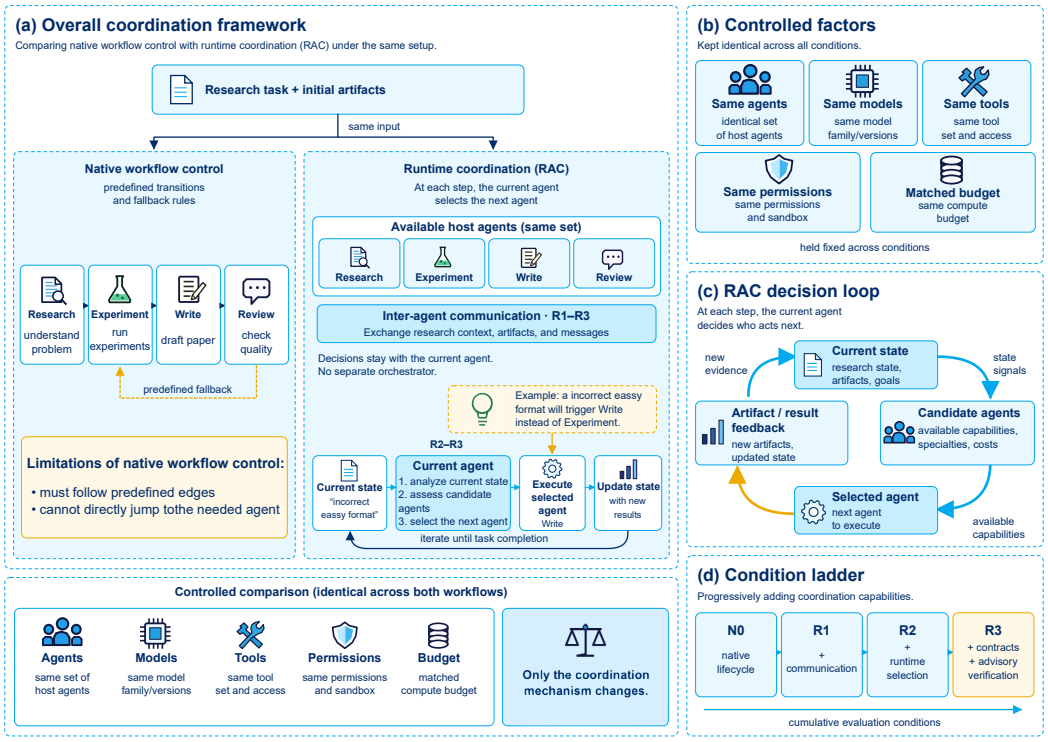


Figure 1: **Same agents, different coordination.** (a) Native transitions versus state-dependent selection by the current agent, with inter-agent communication for exchanging context, artifacts, and messages; this schematic does not imply that all hosts lack runtime delegation. (b) Agents, models, tools, permissions, and run-level budget are held fixed. (c) At each step, the current agent selects the next available agent based on current state, without a separate orchestrator; returned artifacts update the state for the next decision. (d) The cumulative conditions add communication (R1), runtime selection (R2), and contracts plus verification (R3) to the native lifecycle (N0).

2 Related Work

2.1 AI Scientists

AI scientists aim to automate research activities ranging from idea generation and experimentation to analysis and scientific writing (Lu et al., 2024; Schmidgall et al., 2025; Tang et al., 2025). The AI Scientist organizes these activities into an end-to-end workflow, while AI Scientist-v2 introduces agentic tree search within experimentation (Lu et al., 2024; Yamada et al., 2025). Agent Laboratory assigns specialized agents to research phases, AI-Researcher combines literature exploration with experimental development, and ARK provides a human-steered research harness (Schmidgall et al., 2025; Tang et al., 2025; Xin et al., 2026). These systems demonstrate different ways to organize specialized capabilities across the research lifecycle.

Other systems emphasize iterative feedback and adaptation. Co-scientist combines hypothesis generation, debate, and ranking (Gottweis et al., 2025); AutoResearchClaw integrates multi-agent debate, execution repair, and verifiable reporting (Liu et al., 2026); and ARIS uses cross-model review and evidence checks to assess research claims (Yang et al., 2026). EvoScientist accumulates reusable research experience through persistent memory and self-evolution (Lyu et al., 2026). Moving beyond predefined pipelines, freephdlabor supports workflows determined by real-time agent reasoning, while AutoScientists organizes decentralized teams around evolving experimental evidence (Li et al., 2025; Gao et al., 2026). Our work examines whether a shared runtime coordination approach can improve existing hosts with different native control mechanisms.

Scientific output also requires evaluation beyond successful execution. ResearchClawBench evaluates end-to-end research on real-paper-derived tasks, while diagnostic studies examine failures in experimental design, grounding, and scientific reasoning (Xu et al., 2026; Fei et al., 2026). Automated review introduces further uncertainty because assessments can vary across evaluator models (Ravideshik & Kejriwal, 2026). We therefore distinguish feedback used during research from external benchmark evaluation of the final outputs.

2.2 Multi-Agent Coordination

Multi-agent systems combine specialized roles through communication and structured collaboration. AutoGen supports programmable agent conversations, MetaGPT combines role-specific workflows with shared structured messages, and multiagent debate iteratively exchanges candidate answers and critiques (Wu et al., 2023; Hong et al., 2023; Du et al., 2023). These mechanisms shape the information available to agents, while coordination determines how their contributions are organized and which agent acts next.

Several approaches explicitly optimize or adapt this organization. GPTSwarm represents agents as computational graphs and optimizes prompts and inter-agent connectivity (Zhuge et al., 2024). DyLAN selects agent teams and supports dynamic interactions during task solving (Liu et al., 2023). In scientific applications, freephdlabor and AutoScientists further demonstrate reasoning-driven workflows and self-organizing collaboration (Li et al., 2025; Gao et al., 2026). Building on these directions, RAC studies runtime selection across existing AI-scientist hosts while preserving their underlying agents, tools, and permissions. Its cumulative conditions distinguish communication under native control from runtime selection and the combined addition of contracts and verification.

Feedback and resource use are integral to this comparison. MetaGPT incorporates executable feedback, while analyses of multi-agent failures identify inter-agent misalignment and verification or termination problems (Hong et al., 2023; Cemri et al., 2025). In RAC, scoped contracts define the selected agent's task, and artifact-grounded verification informs subsequent work without vetoing transitions. Because additional coordination can increase inference costs, we follow cost-aware agent evaluation principles (Kapoor et al., 2024; 2025) and compare research scores, token usage, and estimated costs under matched run-level budgets.

3 Runtime Agent Coordination for AI Scientists

3.1 Problem Formulation

At decision step t, an AI scientist has a research state s_t comprising its current objective, saved artifacts, unresolved problems, execution history, and remaining budget. Let $\mathcal{A}_t$ denote the agents currently permitted by the host. Communication adds information to this state, while runtime coordination uses it to select the next agent and specify its task. At a nonterminal handoff, the next agent is selected as

$$a_{t+1} = \begin{cases} \pi_{\text{host}}(s_t), & \text{native control (N0, R1),} \\ \pi_{\text{RAC}}(s_t, \mathcal{A}_t), & \text{runtime coordination (R2, R3),} \end{cases} \qquad a_{t+1} \in \mathcal{A}_t. \tag{1}$$

Here, π_{host} is the host's native successor policy, and π_{RAC} is the shared runtime selection policy. Native policies may already include conditional branches, fallback rules, or adaptive delegation. Our comparison examines the effect of replacing this native selection policy with a shared state-dependent policy while retaining the host's permitted agents. Each condition generates its own research trajectory, so s_t need not be identical across conditions.

3.2 Engineering Implementation

Figure 2 summarizes the execution cycle. We use the same RAC package and condition configuration across Agent Laboratory, EvoScientist, and ARK. Each host bridge serializes native state, invokes an existing capability, and returns artifacts and usage records through a shared interface. Bridges contain no host-specific runtime selection rules, verification criteria, or benchmark hints. This design preserves each host's agents, tools, and permissions without requiring identical host internals.

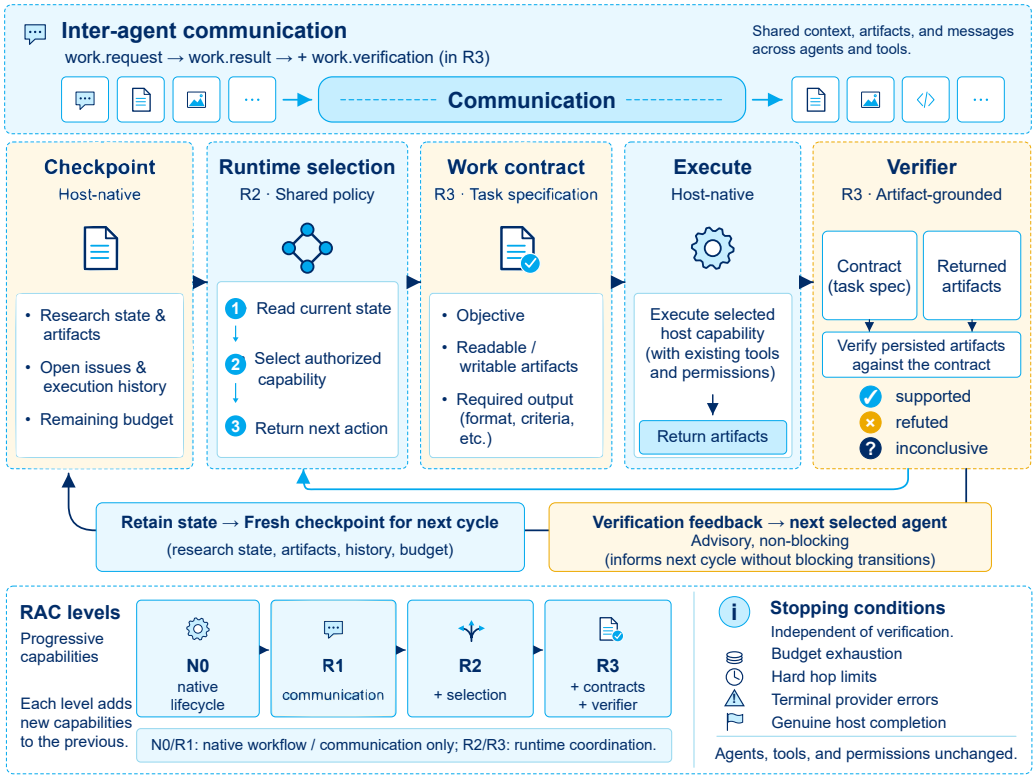


Figure 2: **The RAC execution cycle.** At each handoff, agents exchange requests and results. In R2–R3, the current agent selects the next authorized host capability from a fresh checkpoint, without a separate orchestrator. R3 additionally specifies a work contract and verifies the returned artifacts, passing feedback to the next selected agent.

N0 executes the host-native lifecycle without additional communication mechanisms or a RAC phase loop. R1 introduces inter-agent communication while preserving the native successor at every handoff. R2 retains this communication mechanism and enables the current agent to select the next authorized capability using artifacts, execution history, unresolved problems, and remaining budget. Agents exchange structured requests and results through a common communication channel. Thus, R1–R2 changes next-agent selection while retaining the same communication mechanism. Communication state is isolated between runs, and cross-run memory is reset across all conditions.

R3 adds scoped work contracts and artifact-grounded verification. Each contract specifies the objective, readable and writable artifacts, and required output. The verifier checks persisted artifacts against the contract and records a `supported`, `refuted`, or `inconclusive` verdict. All verdicts preserve the workspace and host transition; feedback enters the next selected agent's prompt rather than directly triggering retries, rerouting, rollback, or stopping. Runs terminate on budget exhaustion, hard hop limits, terminal provider errors, or native host completion. Checkpoints and event records document coordination decisions, while external benchmark evaluation assesses final outputs separately from in-loop verification.

4 EXPERIMENTAL SETUP

4.1 BASELINES

We evaluated RAC with Agent Laboratory (Schmidgall et al., 2025), EvoScientist (Lyu et al., 2026), and ARK (Xin et al., 2026). These hosts span predefined control loops, fixed research phases, and adaptive delegation, allowing us to examine runtime coordination across different native control mechanisms. Each host's native lifecycle serves as its N0 baseline.

4.2 BENCHMARK AND CONDITIONS

ResearchClawBench provides real-paper-derived tasks across scientific domains. Each task supplies a research question, related literature, raw data, and an executable environment while withholding the target paper (Xu et al., 2026). Our evaluation covers mathematics, neuroscience, information science, energy, life science, and physics. We evaluate Agent Laboratory and EvoScientist on ten tasks each and ARK on five tasks. All runs use seed 0.

Table 1 defines the cumulative conditions used to examine communication, runtime selection, and contracts with verification. R3 versus N0 compares the full framework with native execution; R2–R3 treats contracts and verification as a joint addition rather than isolating their individual effects.

Table 1: Four cumulative conditions. R1 preserves native successors, R2 adds shared runtime selection, and R3 jointly adds scoped contracts and verification.

ID	Configuration	Newly enabled mechanism
N0	Native lifecycle	Native execution without additional communication mechanisms or a RAC phase loop.
R1	Communication	Inter-agent communication with the native successor at each handoff.
R2	+ Runtime selection	Select a host capability from a fresh checkpoint using the shared policy.
R3	+ Contracts and verification	Scope the task and check artifacts; pass supported, refuted, or inconclusive feedback to the next selected agent. Retain state and transitions for all verdicts.

4.3 CONTROLS AND ANALYSIS

Within each host, we hold base models, native role prompts, tools, permissions, artifact interfaces, task inputs, and maximum run-level budgets fixed across conditions. Routing and verification costs are included in the same budget. Cross-run memory and evolution stores are reset before each run to isolate within-run coordination. Model choices and inference-cost accounting are detailed in Appendix B.

The primary outcome is the official ResearchClawBench weighted score. We report input and output token usage and estimated inference costs separately. Score summaries equally weight the final evaluation score for each task–host–condition run, including recorded zeros. Within-task differences from N0 are averaged over the same evaluated tasks within each host. Table 4 provides task-level scores.

5 RESULTS

5.1 FULL RUNTIME COORDINATION ACROSS HOST SYSTEMS

Table 2 summarizes all tasks with recorded scores for each host and condition. Each task–host–condition cell contributes its final evaluation score, with equal weighting across tasks. Mean N0 and R3 scores were 5.47 and 10.07 for Agent Laboratory, 15.99 and 15.96 for EvoScientist, and 16.40 and 17.98 for ARK. All four conditions use the same task set within each host. Input-token and cost means summarize the runs with recorded token usage.

Table 2: Means over recorded results, with equal task weighting of final evaluation scores. Mean total input tokens are shown in millions (M). Costs are standardized estimates in USD from recorded input and output tokens using the cost-accounting convention in Appendix B.

Host	Condition	Mean score	Mean total input tokens (M)	Est. cost (USD)
ARK	Native (N0)	16.40	48.85	69.31
ARK	Full RAC (R3)	17.98	68.76	96.69
Agent Laboratory	Native (N0)	5.47	4.26	7.09
Agent Laboratory	Full RAC (R3)	10.07	2.56	4.48
EvoScientist	Native (N0)	15.99	11.49	15.44
EvoScientist	Full RAC (R3)	15.96	38.58	51.50

5.2 Component ablations

Table 3 summarizes the cumulative conditions using the task-weighted aggregation in Table 2. R2 had the highest observed mean for all three hosts: 12.08 for Agent Laboratory, 18.53 for EvoScientist, and 18.42 for ARK. R1 and R3 exceeded N0 for ARK and Agent Laboratory. For EvoScientist, R1 fell below N0, whereas R3 was nearly unchanged. All four conditions are compared on the same tasks within each host. R2–R3 represents the contracts-and-verification bundle rather than either component independently.

Table 3: Mean scores across all available tasks for each host and condition. Tasks are equally weighted using their final evaluation scores; recorded zeros are retained.

Shared configuration	ARK	Agent Lab.	EvoScientist
N0: Native lifecycle	16.40	5.47	15.99
R1: Runtime communication	17.32	9.63	12.07
R2: + Runtime selection	18.42	12.08	18.53
R3: + Contracts and verification	17.98	10.07	15.96

Table 4 reports the task-level scores underlying these summaries. Overall, improvements over N0 outnumber declines in each of R1–R3, with R2 showing the broadest pattern of gains across task–host comparisons. R3 improves on Math 000 and Math 001 for both Agent Laboratory and EvoScientist, as well as on Physics 002 for ARK and Energy 001 for EvoScientist. Several gains also persist across all three runtime conditions, including ARK on Physics 002 and Agent Laboratory on Life 001. Improvements are not uniform, however: EvoScientist scores lower under R3 on Physics 002 and Math 002–003, while other comparisons show only small changes.

Figure 3 complements these summaries. Bars show mean within-task score differences from N0 for each host and condition. Agent Laboratory improves under R1, R2, and R3 by 4.16, 6.61, and 4.61 points, respectively. EvoScientist's corresponding differences are -3.92, 2.55, and -0.03 points; ARK's are 0.92, 2.02, and 1.58 points. R2's mean difference from N0 is positive for all three hosts. These descriptive averages do not quantify variability across independent runs.

Detailed statistics of token usage and estimated inference costs are reported in Appendix C.

5.3 Transferability Test

We additionally evaluated RAC on four DiscoveryBench (Majumder et al., 2025) tasks using Agent Laboratory and EvoScientist. Figure 4 shows positive mean improvements over N0 under R1–R3. Runtime selection further increases the mean improvement from R1 to R2, whereas adding contracts and verification in R3 slightly reduces it. R2 achieves the largest overall gain, providing initial evidence that runtime coordination can improve outcomes beyond ResearchClawBench.

Table 4: Task-level evaluation scores at seed 0. The upper block reports the five tasks evaluated with all three hosts, using recorded median evaluation scores. The lower block reports five additional tasks evaluated with Agent Laboratory and EvoScientist. Each cell gives the score; R1–R3 additionally show the change from the same host's N0 in brackets. Dark green and red denote gains and declines exceeding 3 points, respectively; smaller changes remain uncolored. Colors describe magnitude, not statistical significance.

Host	Condition	Neuroscience 000	Energy 001	Life 001	Neuroscience 002	Physics 002
ARK	N0	10.00	29.30	8.50	6.60	27.60
	R1	7.20 [-2.80]	27.60 [-1.70]	9.40 [+0.90]	5.70 [-0.90]	**36.70 [+9.10]**
	R2	8.40 [-1.60]	31.90 [+2.60]	**14.10 [+5.60]**	5.70 [-0.90]	**32.00 [+4.40]**
	R3	**13.80 [+3.80]**	27.90 [-1.40]	10.45 [+1.95]	3.15 [-3.45]	**34.60 [+7.00]**
Agent Lab.	N0	9.00	9.20	1.30	3.30	0.00
	R1	8.00 [-1.00]	10.50 [+1.30]	**4.65 [+3.35]**	5.40 [+2.10]	**6.60 [+6.60]**
	R2	3.40 [-5.60]	12.20 [+3.00]	**6.90 [+5.60]**	5.70 [+2.40]	**24.55 [+24.55]**
	R3	10.40 [+1.40]	5.90 [-3.30]	2.90 [+1.60]	5.05 [+1.75]	**4.30 [+4.30]**
EvoScientist	N0	8.40	17.70	16.20	7.80	39.20
	R1	6.60 [-1.80]	19.90 [+2.20]	13.50 [-2.70]	4.60 [-3.20]	10.40 [-28.80]
	R2	6.00 [-2.40]	**25.80 [+8.10]**	13.75 [-2.45]	8.30 [+0.50]	**43.10 [+3.90]**
	R3	**13.60 [+5.20]**	**28.50 [+10.80]**	14.10 [-2.10]	7.05 [-0.75]	16.45 [-22.75]

Host	Condition	Math 000	Math 001	Math 002	Math 003	Information 003
Agent Lab.	N0	17.15	11.10	0.00	0.00	3.60
	R1	18.90 [+1.75]	**23.10 [+12.00]**	**3.60 [+3.60]**	**10.05 [+10.05]**	5.45 [+1.85]
	R2	**22.50 [+5.35]**	**27.50 [+16.40]**	**8.40 [+8.40]**	**6.60 [+6.60]**	3.00 [-0.60]
	R3	**20.25 [+3.10]**	**29.90 [+18.80]**	**4.80 [+4.80]**	**9.00 [+9.00]**	**8.20 [+4.60]**
EvoScientist	N0	16.05	16.80	10.60	16.30	10.80
	R1	9.60 [-6.45]	**32.60 [+15.80]**	5.30 [-5.30]	9.45 [-6.85]	8.70 [-2.10]
	R2	**21.60 [+5.55]**	**36.20 [+19.40]**	11.00 [+0.40]	11.25 [-5.05]	8.30 [-2.50]
	R3	**22.35 [+6.30]**	**37.70 [+20.90]**	3.20 [-7.40]	9.70 [-6.60]	6.90 [-3.90]

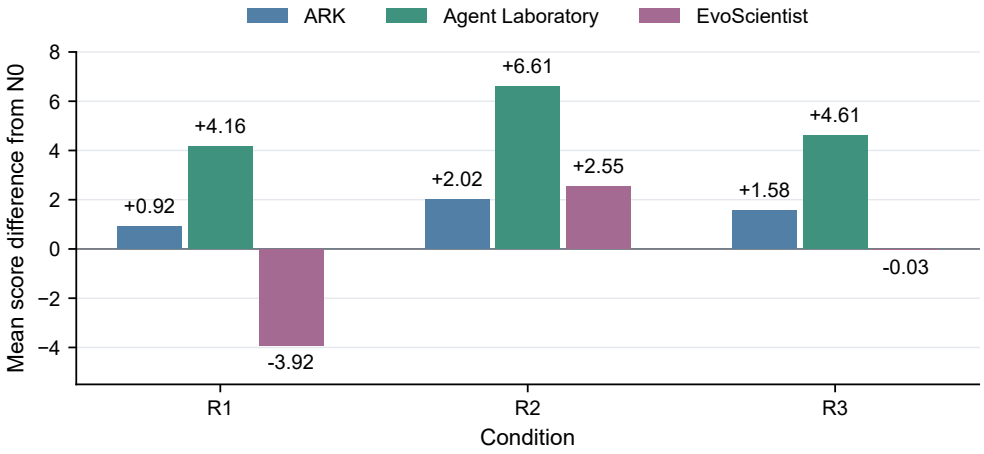


Figure 3: **Within-task score improvements.** Bars show the mean score difference from N0 for each host under R1–R3, computed on tasks with scores for both the condition and the same host's N0. All conditions use the same task set within each host; bars represent descriptive means, not uncertainty estimates.

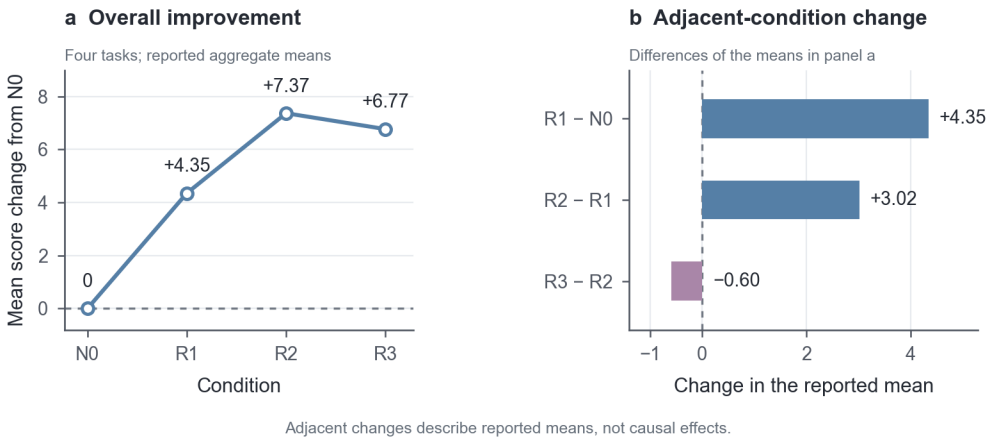


Figure 4: **Transfer to DiscoveryBench.** (a) Mean score improvements over N0, computed within each host and task, averaged across available hosts within each task, and then equally across the four tasks. Scores are rescaled to 0–100; N0 is zero by construction. (b) Differences between adjacent condition means in (a). These changes are descriptive comparisons, not estimates of independent component effects. Connecting lines in (a) are visual guides.

5.4 Trace-level analysis: three coordination effects

To examine how coordination helps or fails, we inspected records from 60 task–host–condition cells in the main experiments, comprising 1,181 hop excerpts and 180 external scoring receipts. We use selected trajectories for qualitative analysis, not to estimate the prevalence of each outcome. The excerpts are not complete conversations, and their scientific claims were not independently recomputed. Hop indices are zero-based.

Evidence–action closure effect. In Physics_002 with EvoScientist R2, the native reviewer reported at hop 6 that the report contained only 39 bytes despite existing code, analyses, and figures. At hop 7, the selected debug capability reported producing a 337-line report of approximately 22 KB. Ten recorded scientific-artifact changes and the final inventory support the conclusion that a report was produced. This handoff connected a specific defect to corrective work, although subsequent debug calls made no recorded scientific-artifact changes before the hop limit. The feedback came from the host reviewer, not R3's verifier; delivery repair does not establish scientific correctness.

Contract-satisfied stagnation effect. In Physics_002 with EvoScientist R3, the reviewer identified a standard-error discrepancy: 0.011 in the report versus 0.015 in the data. At hop 8, the planner attributed it to duplicated observations and proposed code changes followed by regeneration of results. However, hops 8–13 remained planner calls with no recorded scientific-artifact changes, ending at the 14-hop limit. The trajectory illustrates a failure to move from diagnosis to implementation. Claims of repeated verification appeared in agent messages but were not independently confirmed from verifier events.

Self-consistent drift effect. In Neuroscience_000 with EvoScientist R3, the native reviewer reported that numerical and code–report consistency checks passed and assigned a score of 8/10. External evaluation nevertheless identified missing cross-laboratory, sex, and environment SHAP comparisons, yielding a three-round median score of 13.6. The two scores use different criteria and scales and are not directly comparable. Similarly, Life_001 with Agent Laboratory R2 analyzed ten synthetic patients rather than the supplied seven-patient dataset. These cases distinguish internal consistency from coverage of the research objective, without identifying whether divergence originated in task interpretation, data limitations, or coordination.

These cases illustrate possible coordination mechanisms and failure modes; they do not establish the causes of the aggregate R2–R3 score differences.

6 DISCUSSION

Figure 3 shows positive mean differences from N0 for R2 across all three hosts and for every condition in ARK and Agent Laboratory. R2 has the highest observed mean on the common task set within each host, although a single run per condition does not establish a robust performance advantage. Host structure and coordination overhead are plausible influences on these outcomes. DiscoveryBench shows a similar aggregate pattern (see Figure 4), with positive gains across R1–R3 and the largest gain under R2, providing limited evidence of applicability beyond ResearchClawBench.

Under matched run-level budgets, R1 adds message context without changing native transitions, so its extra input may not be offset by avoiding unnecessary steps. R2 can redirect work toward a relevant agent, whereas R3 adds contract context and verification calls that can further increase per-hop cost. This overhead could cause earlier budget exhaustion and limit useful revisions. However, the aggregate results do not establish whether R3 completes fewer steps than R2 on common tasks. Distracting instructions or inaccurate verifier feedback are alternative explanations for weaker outcomes. R2 has the lowest recorded mean inference costs for ARK and Agent Laboratory, but not EvoScientist; the results do not establish a universal efficiency advantage. A modest budget increase might allow R3 to complete useful revisions displaced by its overhead. However, the repeated planning trajectory in Section 5.4 suggests that additional budget alone may not resolve ineffective delegation: progress also requires moving from diagnosis to implementation. Testing all conditions over the same budget range, with per-hop token use and completed research steps, would distinguish this possibility from ineffective additional coordination.

Agent Laboratory's lower recorded token consumption may leave more room for coordination within the budget. Its fixed phase structure may also benefit from information carried between stages, while EvoScientist's native supervisor already provides some runtime delegation. Native role prompts may differ in how effectively they use the added context. These explanations remain hypotheses: aggregate consumption does not establish remaining budget at each decision, and baseline performance and task difficulty may also contribute.

The evaluation has four boundaries. First, a single run per task–host–condition at seed 0 does not measure variability across independent runs. Second, ResearchClawBench covers bounded problems rather than open-ended scientific discovery. Third, inference costs use a common token tariff rather than provider invoices. Fourth, R3 bundles contracts and verification, leaving their individual effects unidentified.

7 CONCLUSION

We presented RAC as an alternative coordination approach for multi-agent AI scientists. It selects existing agents from current state and, in R3, adds scoped contracts and artifact-grounded verification feedback. Across all available ResearchClawBench scores, R2 had the highest task-weighted mean for each host, whereas R3's performance relative to N0 varied across hosts. These rankings are descriptive and do not establish robustness across independent runs. Standardized token-based costs describe resource use without establishing a general performance or cost advantage.

ETHICS STATEMENT

Adaptive scientific workflows may accelerate research but also amplify unsupported claims or optimize for fallible reviewer signals. RAC preserves host permissions, scopes tasks through contracts, and records verification evidence. Verification does not prevent unsupported outputs from propagating and is not a safety gate. Accountable human review remains necessary, particularly in safety-critical, dual-use, or regulated domains.

Reproducibility statement

The host adaptation boundary and experimental conditions are described in Sections 3 and 4. Table 4 provides task-level evaluation scores, and Appendix B specifies the execution and judge models and token-based cost calculation. The analysis below defines task weighting and the paired comparisons used in Figure 3.

References

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, et al. Why do multi-agent LLM systems fail? *arXiv preprint arXiv:2503.13657*, 2025. URL https://arxiv.org/abs/2503.13657.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023. URL https://arxiv.org/abs/2305.14325.

Yanlin Fei, Nazhou Liu, Xinmiao Yu, Shaolong Chen, Lei Li, Rahul Thapa, et al. How Do Agents Fail on AutoResearch: End-to-End Diagnostic Evaluation on 100 Real-World Frontier Research Tasks. *arXiv preprint arXiv:2608.14905*, 2026. URL https://arxiv.org/abs/2608.14905.

Shanghua Gao, Ada Fang, and Marinka Zitnik. AutoScientists: Self-Organizing Agent Teams for Long-Running Scientific Experimentation. *arXiv preprint arXiv:2605.28655*, 2026. URL https://arxiv.org/abs/2605.28655.

Shashwat Goel, Rishi Hazra, Dulhan Jayalath, Timon Willi, Parag Jain, William F Shen, Ilias Leontiadis, Francesco Barbieri, Yoram Bachrach, Jonas Geiping, et al. Training ai co-scientists using rubric rewards. *arXiv preprint arXiv:2512.23707*, 2025.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, et al. Accelerating Scientific Discovery with Co-Scientist. *arXiv preprint arXiv:2502.18864*, 2025. URL https://arxiv.org/abs/2502.18864.

Sirui Hong, Mingchen Zhuge, Jiaqi Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. Metagpt: Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*, 2023. URL https://arxiv.org/abs/2308.00352.

Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. AI Agents That Matter. *arXiv preprint arXiv:2407.01502*, 2024. URL https://arxiv.org/abs/2407.01502.

Sayash Kapoor, Benedikt Ströbl, Peter Kirgis, Nitya Nadgir, Zachary S. Siegel, et al. Holistic Agent Leaderboard (HAL): Cost-Controlled Third-Party Evaluation of AI Agents. https://hal.cs.princeton.edu, 2025.

Ed Li, Junyu Ren, Xintian Pan, Cat Yan, Chuanhao Li, Dirk Bergemann, and Zhuoran Yang. Build Your Personalized Research Group: A Multiagent Framework for Continual and Interactive Science Automation. *arXiv preprint arXiv:2510.15624*, 2025. URL https://arxiv.org/abs/2510.15624.

Jiaqi Liu, Shi Qiu, Mairui Li, Bingzhou Li, Haonian Ji, Siwei Han, Xinyu Ye, Peng Xia, Zihan Dong, Congyu Zhang, et al. Autoresearchclaw: Self-reinforcing autonomous research with human-ai collaboration. *arXiv preprint arXiv:2605.20025*, 2026. URL https://arxiv.org/abs/2605.20025.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic LLM-powered agent network for task-oriented agent collaboration. *arXiv preprint arXiv:2310.02170*, 2023. URL https://arxiv.org/abs/2310.02170.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery. *arXiv preprint arXiv:2408.06292*, 2024. URL https://arxiv.org/abs/2408.06292.

Yougang Lyu, Xi Zhang, Xinhao Yi, Yuyue Zhao, Shuyu Guo, Wenxiang Hu, Jan Piotrowski, Jakub Kaliski, Jacopo Urbani, Zaiqiao Meng, Lun Zhou, and Xiaohui Yan. Evoscientist: Towards multi-agent evolving ai scientists for end-to-end scientific discovery. *arXiv preprint arXiv:2603.08127*, 2026. URL https://arxiv.org/abs/2603.08127.

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 *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0d70af566e69f1dfb687791ecf955e28-Abstract-Conference.html.

Vaibhava Lakshmi Ravideshik and Mayank Kejriwal. Can AI Evaluate AI Scientists? A Benchmarking Study of Autonomous Research Generation Systems Using Automated Multi-Model Review. *arXiv preprint arXiv:2607.28631*, 2026. URL https://arxiv.org/abs/2607.28631.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, et al. Agent Laboratory: Using LLM Agents as Research Assistants. *arXiv preprint arXiv:2501.04227*, 2025. URL https://arxiv.org/abs/2501.04227.

Jiabin Tang, Lianghao Xia, Zhonghang Li, and Chao Huang. AI-Researcher: Autonomous scientific innovation. *arXiv preprint arXiv:2505.18705*, 2025. URL https://arxiv.org/abs/2505.18705.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023. URL https://arxiv.org/abs/2308.08155.

Jihao Xin, Yangzhixin Luo, Muhammad Bilal, and Marco Canini. ARK: A human-steered research harness for AI-assisted scientific discovery. In *ICML 2026 AI for Science Workshop*, 2026. URL https://openreview.net/forum?id=fSAiWmpk0O.

Wanghan Xu, Shuo Li, Tianlin Ye, Qinglong Cao, Yixin Chen, Hengjian Gao, et al. ResearchClawBench: A Benchmark for End-to-End Autonomous Scientific Research. *arXiv preprint arXiv:2606.07591*, 2026. URL https://arxiv.org/abs/2606.07591.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist-v2: Workshop-Level Automated Scientific Discovery via Agentic Tree Search. *arXiv preprint arXiv:2504.08066*, 2025. URL https://arxiv.org/abs/2504.08066.

Ruofeng Yang, Yongcan Li, and Shuai Li. ARIS: Autonomous Research via Adversarial Multi-Agent Collaboration. *arXiv preprint arXiv:2605.03042*, 2026. URL https://arxiv.org/abs/2605.03042.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. GPTSwarm: Language agents as optimizable graphs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 62743–62767. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhuge24a.html.

A Contract and Trace Schema

Checkpoints record run and hop identifiers, objectives, native state, artifacts, unresolved problems, budget, and available capabilities. Requests and results are typed `work.request` and `work.result` events. R3 adds contracts and `work.verification` records containing verdicts, checks, reasons, and feedback status. Every verdict is retained and passed to the next selected agent. Provenance records the communication backend and non-secret room identifier, never invite tokens.

B Models and Cost Accounting

Models. We use DeepSeek V4 Pro as the execution model for the AI-scientist hosts and GPT-5.5 as the external ResearchClawBench judge. External judging is separate from the in-loop verifier and does not control agent selection or host transitions.

Cost accounting. We estimate inference costs from recorded input and output token usage using a common DeepSeek V4 Pro tariff. We apply the official peak rates of US$1.32 per million non-cached input tokens and US$3.96 per million output tokens, as listed in the DeepSeek API pricing documentation on 23 September 2026. For each episode, the estimated cost is $(1.32\,T_{\mathrm{in}} + 3.96\,T_{\mathrm{out}})/10^6$ USD. Here, T_{in} and T_{out} are its recorded input and output token counts. We apply no cache-hit or off-peak discounts. Table entries average episode estimates before rounding. These are standardized inference-cost estimates, not Azure invoices, and exclude external GPT-5.5 judging charges.

C Token Usage and Estimated Cost

Table 5 summarizes mean input and output tokens and standardized inference costs for the evaluated tasks. Agent Laboratory has its lowest input-token and cost means in R2, but its lowest output-token mean in R1. EvoScientist has its lowest token and cost means in N0. ARK has its lowest input-token, output-token, and cost means in R2. These are within-host resource summaries; the hosts use different task sets.

Table 5: Mean recorded token usage and estimated inference cost. Input tokens are in millions (M); output tokens are in thousands (K). Costs use the uniform DeepSeek V4 Pro peak, non-cached tariff described in Appendix B and exclude external judging. Unequal task coverage prevents interpreting these means as matched-task efficiency effects.

Host	Condition	Input (M)	Output (K)	Est. cost (USD)
ARK	N0	48.85	1,219.69	69.31
ARK	R1	63.06	1,425.76	88.88
ARK	R2	45.47	1,113.58	64.42
ARK	R3	68.76	1,496.85	96.69
Agent Laboratory	N0	4.26	369.42	7.09
Agent Laboratory	R1	2.90	269.91	4.89
Agent Laboratory	R2	2.36	308.18	4.33
Agent Laboratory	R3	2.56	279.24	4.48
EvoScientist	N0	11.49	70.34	15.44
EvoScientist	R1	18.31	126.37	24.67
EvoScientist	R2	37.74	146.97	50.40
EvoScientist	R3	38.58	145.30	51.50

D Analysis Specification

The primary comparison is R3 versus N0 under matched budgets. Condition means equally weight final task–host–condition scores. For each host and each of R1–R3, we subtract the N0 score on the same task and average these differences over the same evaluated task set within that host. Figure 3 reports these descriptive paired means. The bars do not measure variability across independent runs. Adjacent conditions are compared on the same tasks, but R2–R3 does not separate contracts from verification.

E Generative AI use disclosure

Generative AI systems are intrinsic experimental components of the evaluated AI-scientist workflows. They generate plans, code, experimental analyses, reviews, and manuscript drafts inside

each evaluated system. Human authors are responsible for the research question, system design, evaluation protocol, verification of reported values, literature checking, interpretation, and final manuscript. All AI-assisted content was checked against persisted artifacts and source records before submission.