WHERE THE THREAD BREAKS: EVALUATING LLM AUDITORS IN LONG-HORIZON SCIENTIFIC WORKFLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Advances in large language models (LLMs) have enabled autonomous agents to perform complex, long-horizon tasks. These tasks require reliable output verification, as errors can propagate across steps and compromise the final results. LLM-based auditing provides a practical approach to verifying the output of each step. However, its performance in long-horizon workflows, where later steps depend on earlier outcomes, remains underexplored. We introduce AuditSpan, a benchmark for evaluating LLM-based auditing in long-horizon scientific workflows. Specifically, we (1) examine whether and how the benefits and limitations of auditing individual outputs carry over to long-horizon workflows and (2) investigate challenges arising from interactions among workflow steps, including difficulties in assessing overall scientific validity and tracking error propagation and correction. Our results show that, when auditing extends from individual outputs to long-horizon workflows, appropriate abstention decreases and overconfidence increases. Beyond these amplified limitations, auditors may approve scientifically flawed workflows when most individual steps are correct and may treat errors as resolved even when their effects persist in later outputs. These findings can guide the development of more reliable LLM-based auditors and support progress toward fully automated long-horizon workflows.

1 INTRODUCTION

Large language model (LLM)-based research agents are rapidly gaining the ability to autonomously execute end-to-end scientific workflows (Lu et al., 2024; Yamada et al., 2025; Bragg et al., 2026). For instance, in *autoresearch*, an agent repeatedly edits model-training code, runs a fixed-budget experiment, and retains only changes that improve performance, allowing the loop to continue autonomously overnight (Karpathy, 2026). Similarly, Ricursive Intelligence uses AI to design better chips, which in turn support more capable AI systems (Ricursive Intelligence, 2026). Because such systems can operate independently for long periods, their experimental output can far exceed human experts' capacity to review it. At that scale, their effectiveness depends on how reliably and efficiently their outputs can be evaluated.

Long-horizon scientific workflows typically require auditing both the final conclusion and the output of each intermediate step (Chen et al., 2024; Starace et al., 2025; Bragg et al., 2026). Errors introduced early in a workflow can propagate through later stages and lead to costly laboratory validation of unsupported conclusions. The stakes are especially high in drug development, where a program may take decades and cost billions of dollars (Cheng et al., 2025). However, autonomous agents can run continuously and generate large volumes of intermediate outputs. Reviewing all these outputs is impractical for human experts.

LLM-based auditing offers a practical approach to this evaluation by using a reviewer model to check a candidate output against task requirements and available evidence. Studies of individual responses, summaries, and code show that LLM judges can approximate human preference judgments and that model-generated critiques can help human reviewers identify errors (Zheng et al., 2023; Saunders et al., 2022; McAleese et al., 2024). However, in long-horizon scientific workflows, evidence accumulates and later steps depend on earlier results. For example, a calculation may be correct but

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.

rely on an invalid earlier result. Auditing the calculation alone can therefore miss an error that affects the final conclusion (Zheng et al., 2025; Verma et al., 2026). Such failures can accumulate across steps, even when average audit accuracy is high. These risks make it necessary to examine whether LLM auditors can reliably assess the validity of complete workflows.

We introduce **AuditSpan**, a benchmark for evaluating LLM auditors in scientific workflows with explicit dependencies. AuditSpan is designed around two questions. First, how do correctness, abstention, calibration, and resistance to pressure behave as auditing extends from individual outputs to a sequence of dependent steps? Second, can auditors assess overall scientific validity, trace the consequences of upstream errors, and recognize incomplete corrections? To make these questions testable, we construct controlled workflows with independently established reference verdicts. Errors in candidate outputs and gaps in accessible evidence are varied separately, and final conclusions are checked against explicit scientific validity conditions.

Our experiments reveal three key findings. First, audit reliability and confidence calibration tend to worsen as more dependent steps are included. For instance, Gemini 2.5 Pro without tools correctly audits the first step in 73.3% of workflows, but correctly audits every step in only 15.8%. Moreover, calibration error increases over the same comparison in 19 of 24 model–tool settings. Second, auditors frequently fail to abstain and can yield to unsupported generator pressure. Kimi K2.6 with tools achieves only 2.3% abstention recall. In addition, paired follow-up tests after correct initial rejections show that pressure induces harmful verdict changes at 14.8% of checkpoints for GPT-5 mini and 2.8% for Claude Sonnet 4.5, both with tools. Third, auditors struggle to assess overall scientific validity and recognize incomplete corrections. GPT-5.5 and Gemini 2.5 Pro without tools approve 35.7% and 68.5% of invalid final conclusions, respectively. After correct initial rejections, these models also incorrectly declare 23.3% and 66.7% of partial repairs fully resolved despite remaining downstream errors.

Our findings reveal a gap between judging individual outputs and assessing the scientific validity of a complete workflow. We argue that future work should prioritize reasoning across steps, so auditors can trace how errors affect later results and check whether those effects remain after corrections. Progress also requires better confidence calibration, appropriate abstention, and resistance to unsupported pressure. AuditSpan provides a controlled setting for investigating these challenges and testing methods for more reliable scientific auditing.

1.1 Main Contributions

Our main contributions are summarized below.

- We introduce AuditSpan, a benchmark for evaluating LLM auditors in long-horizon scientific workflows, covering ten synthetic base workflows across five fields with four to ten auditable steps per workflow.
- We construct controlled workflows with explicit dependencies and independently established reference verdicts. Output errors and missing evidence are varied separately to evaluate how auditors handle incorrect results and insufficient information.
- We develop an evaluation protocol for correctness, abstention, calibration, and resistance to pressure at individual steps and across complete workflows, together with tests of scientific validity, error propagation, and correction tracking.
- We evaluate twelve models with and without tools, showing that audit errors accumulate across steps and that auditors frequently approve invalid conclusions or mistake partial repairs for complete corrections.

1.2 Related Work

Autonomous long-horizon research. Advances in LLMs have enabled research agents to automate connected stages of scientific work, including idea generation, experimentation, and reporting (Lu et al., 2024; Schmidgall et al., 2025). To support research over multiple stages, the AI Scientist-v2 uses tree search to explore experimental choices (Yamada et al., 2025), while Kosmos maintains shared findings across literature search and data analysis (Mitchener et al., 2025b). The *autoresearch* loop similarly uses experiment results to guide repeated changes to training code (Karpathy, 2026).

Alongside these systems, ScienceAgentBench and AstaBench evaluate agents' ability to complete scientific tasks (Chen et al., 2024; Bragg et al., 2026). As these workflows progress, later decisions depend on accumulated results, so an undetected error can affect subsequent analyses and conclusions. This creates a need to evaluate the auditors responsible for checking those results.

Auditing individual outputs. LLM-based evaluation offers a way to review model outputs at a scale that is difficult to achieve through human review alone. One approach uses LLM judges to assign scores or compare candidate responses (Zheng et al., 2023). Since agreement with human preferences does not necessarily establish correctness, JudgeBench evaluates judges using response pairs with objectively verifiable labels (Tan et al., 2025). Models can also assist evaluation by generating critiques that help human reviewers identify errors in summaries and code (Saunders et al., 2022; McAleese et al., 2024). These approaches assess an individual answer or artifact against its task and available evidence. In scientific workflows, we examine whether auditors can determine if an output is supported by earlier records that may themselves contain errors.

Auditor behavior on individual outputs. Beyond identifying errors, reliable auditing requires appropriate handling of uncertainty. Work on unanswerable questions shows that models often fail to abstain when information is insufficient (Kirichenko et al., 2025), while studies of judge calibration find that stated confidence can exceed actual accuracy (Tian et al., 2025). Judgments can also be influenced by the interaction itself: studies of *sycophancy* show that feedback may shift toward users' stated beliefs (Sharma et al., 2024), including under repeated pressure in multi-turn dialogue (Hong et al., 2025). Even a low error rate at each step can produce frequent failures somewhere in a long workflow. We evaluate these behaviors at both individual steps and across complete audit trajectories to distinguish changes in judgment quality from the accumulation of failures as more steps are audited.

Auditing long-horizon tasks. Long-horizon auditing requires checking how intermediate results support later conclusions. Prior work studies verification of reasoning steps (Lightman et al., 2024; Zheng et al., 2025) and agent workflows (Zhuge et al., 2025; Lù et al., 2025), including scientific replications (Starace et al., 2025) and tool-call dependencies (Verma et al., 2026). AuditSpan focuses on *verifiable steps in chained scientific analyses*, where a result can be consistent with its immediate inputs yet scientifically invalid because of an earlier error. Each step has an independently established reference verdict and explicit links to the earlier outputs it uses. We introduce errors and regenerate later outputs to determine which downstream results remain valid. This allows us to test whether auditors can verify final conclusions, trace the consequences of upstream errors, and recognize corrections that leave invalid downstream results.

2 Benchmark Curation

AuditSpan examines how the reliability of LLM auditing changes from individual outputs to long-horizon scientific workflows. It also examines additional challenges arising from dependencies between workflow steps, including assessing overall scientific validity, tracing error propagation, and recognizing incomplete corrections. To study these questions, we construct workflows with explicit dependencies and verifiable simulated outputs, then create controlled cases containing output errors or missing evidence. The following subsections describe the base workflows, verifiable steps, and perturbation strategy.

2.1 Base scientific workflows

AuditSpan comprises ten base workflows spanning five scientific fields, as listed in Table 3(a). Drawing on task types from BixBench (Mitchener et al., 2025a), ScienceAgentBench (Chen et al., 2024), SciCode (Tian et al., 2024), DiscoveryBench (Majumder et al., 2024), and BLADE (Gu et al., 2024), we construct synthetic measurements and study records with known source facts so that workflow outputs can be independently verified. Each workflow links scientific steps through explicit dependencies and assigns a verification rule to every output. For example, the materials-screening workflow (C2) aims to identify candidate materials whose predicted bulk modulus meets a specified threshold. Its eight steps harmonize measurement units, separate training and test material families, normalize features using training data, fit a property predictor, predict held-out properties, validate prediction error, screen candidate materials, and report recommendations. We characterize each

workflow by its horizon T, the number of auditable scientific steps, and dependency depth D, the number of steps along the longest dependency chain. All designed workflows contain four to ten steps, and eight of the ten contain at least six.

2.2 VERIFIABLE STEPS

A workflow step performs a specific scientific task and produces an output to audit. Each step has a predefined verification rule and three components:

Components of a verifiable step

Input. Source data, selected output fields from earlier steps, or both, used to perform the step. We record these dependencies, which may skip steps.

Output. The simulated result to be audited, organized into named fields, such as a mean and a sample count. Later steps use only the fields specified in their inputs.

Reference label. The expected verdict for the candidate output given the supplied evidence: ACCEPT (correct), REJECT (incorrect), or ABSTAIN (insufficient evidence to decide).

Each step combines one or more of the eight basic actions in Table 3(b), such as retrieving documents, preparing data, writing or executing code, and analyzing results. We define verification rules for these actions using source facts, independent calculations, or explicit scientific conditions. In C2, for example, step 6 compares the predictions from step 5 with measured properties of held-out materials and reports the mean absolute error and a validation decision. We verify these outputs by independently recomputing the error and checking that the decision follows the specified criterion: validation passes if and only if the error is at most 2 GPa.

2.3 PERTURBATION STRATEGY

We introduce controlled perturbations to evaluate LLM auditors' ability to detect errors and recognize insufficient evidence. We distinguish two categories: *output perturbations*, which introduce errors into candidate outputs, and *evidence perturbations*, which remove supporting information from the auditor's input while keeping the candidate outputs unchanged.

Output perturbations. Starting from the correct baseline, we apply one of three actions to one output field at one selected step: *replace with incorrect content*, *remove required content*, or *add invalid or unsupported content*. Examples appear in Table 3(c) in Appendix A. The field must be used by a later step so that we can test how errors propagate through the workflow. We select perturbations that make the chosen output incorrect while allowing subsequent steps to produce their specified outputs. To test another step or action, we start again from the correct baseline. We then regenerate later outputs using their specified inputs, including changed values passed through intermediate steps. The original task inputs remain fixed. Each resulting output is checked against its verification rule to assign a reference label.

Evidence perturbations. We hide one source value from the auditor while keeping the underlying data and candidate outputs unchanged. Using the checks detailed in Appendix B.3, we assign ACCEPT or REJECT when the remaining evidence determines whether the output is correct, and ABSTAIN otherwise. For example, an output claims that Method A has higher test accuracy than Method B, with recorded accuracies of 90% and 80%, respectively. The reference changes from ACCEPT to ABSTAIN when B's result is hidden.

Benchmark composition and validation. Each base workflow contributes two baselines, six output-error cases, and four incomplete-evidence cases, yielding 120 workflows and 828 step targets. The references comprise 406 ACCEPT, 294 REJECT, and 128 ABSTAIN labels. The 60 output-error cases contain 49 replacements, six removals, and five additions. Replacements are used in ten cases for which removal or addition has no eligible target, so operator coverage is uneven. Separate numerical implementations produce outputs and compute references. We check analytical cases, reference labels, accessible evidence, and artifact histories, with independent AI review as an additional construction check. Reference labels are stored separately from auditor inputs; variants of a base workflow are kept together in any data split.

3 EXPERIMENTAL SETUP

We evaluate individual judgments, complete workflow audits, and the recognition of error propagation and correction. The main evaluation compares twelve models on the same workflows. Controlled comparisons vary history, tools, workflow structure, evidence, and interaction conditions. Their protocols appear in Appendix C, and the reporting rules appear in Appendix E.

3.1 MODELS AND MAIN EVALUATION

We evaluate GPT-5.5, GPT-5.4, GPT-5, GPT-5 mini, GPT-5 nano, GPT-4.1, Claude Opus, Sonnet, and Haiku 4.5, Gemini 2.5 Pro and Flash, and Kimi K2.6. All requests use the providers' direct APIs. Model identifiers and decoding settings are fixed and reported in Table 4. Each model audits all 828 step targets in the 120 benchmark workflows, with and without tools.

3.2 AUDITING PROTOCOL

At each workflow step, the auditor receives an audit request and returns a judgment of whether the candidate output satisfies the task requirements. The main evaluation permits one follow-up per condition after a correct initial rejection to test responses to pressure and repairs.

Auditor input. The auditor receives the current step's task, its candidate output, and the evidence available for verification. The request also includes audit instructions, tool permissions, workflow history, and audit dialogue. History records earlier tasks, inputs, outputs, and their dependencies; dialogue records the previous verdict and latest simulated generator reply and is empty for an initial audit. Each audit uses a fresh conversation and workspace, with reference labels and solutions hidden from both the prompt and tool environment. The examples below illustrate an initial audit of C2's validation step, which checks mean absolute error (MAE).

Example auditor input

Auditor instruction.	`Audit output; return verdict, confidence, justification.`
Current step task.	`Step 6: compute MAE; pass if MAE <= 2 GPa, else fail.`
Candidate output.	`Mean absolute error: 3 GPa; validation: pass.`
Available evidence.	`Predictions=[10,20] GPa; measurements=[10,26] GPa; ...`
Workflow history.	`Step 5: predict from model(...), features(...): [10,20] GPa.`
Audit dialogue.	`Empty (initial audit).`
Tool permissions.	`Run Python on supplied files; at most 8 tool calls; ...`

Auditor output. Every response contains a verdict, confidence, and brief justification. The verdict is ACCEPT if the evidence establishes correctness, REJECT if it establishes an error, and ABSTAIN if it is insufficient to decide. Confidence estimates the probability that a definitive verdict is correct, expressed in $[0, 1]$; it is `null` for ABSTAIN. Propagation checks additionally request links from originating errors to affected outputs, and repair reviews request a status of RESOLVED, UNRESOLVED, or UNCERTAIN. The complete input and output specifications appear in Appendix D. An example response is REJECT with confidence 0.95 and the justification "MAE is 3 GPa, exceeding the 2 GPa limit; validation should fail."

Tool access. We evaluate auditors with and without tools using the same source evidence. With tools, auditors may inspect supplied files, execute candidate code, and run independent checks within a fixed budget.

Pressure follow-ups. We compare a neutral request to reconsider with a reply insisting that the output is correct. Both retain the same initial verdict, candidate output, evidence, and history, so differences in subsequent verdicts measure susceptibility to unsupported pressure.

Repair follow-ups. We present unchanged, partially repaired, and completely repaired workflow versions in separate follow-ups. A partial repair corrects the original error but leaves at least one affected later output incorrect; a complete repair also corrects all affected later outputs. We score

the verdict and repair status separately. Selection rules, additional controls, and resource limits are detailed in Appendix D.

3.3 Evaluation metrics

The evaluation compares each auditor verdict with the reference assigned for the accessible evidence. Four measures extend this assessment from individual outputs to sequences of steps, while three further checks address challenges arising from step dependencies. Formal definitions and denominators for both groups appear in Appendix E.

Auditing across steps. *(1) Correctness.* Step accuracy measures agreement with reference verdicts, while whole-workflow success requires every step to be audited correctly. False approval and false rejection further characterize judgment errors, while success over observed prefixes tracks their accumulation across steps. However, lower whole-workflow success alone does not establish that individual judgments deteriorate with longer context. *(2) Abstention.* When evidence is insufficient, abstention recall measures how often the auditor abstains; for decidable outputs, over-abstention measures unnecessary abstention (Kirichenko et al., 2025). At the workflow level, this assessment records whether any definitive verdict is issued where abstention is required. *(3) Calibration.* For valid definitive verdicts, expected calibration error (ECE) measures agreement between confidence and accuracy using ten fixed, equal-width confidence bins (Guo et al., 2017). Complementary measures include mean confidence minus accuracy and definitive-verdict coverage, reported by step position and workflow prefix. *(4) Resistance to pressure.* Under unsupported pressure, a harmful change occurs when a correct neutral verdict becomes an incorrect definitive verdict matching the expressed preference. Paired checkpoint rates measure this change locally; the fraction of workflows with any harmful change and the cumulative occurrence of the first change extend the assessment across steps (Hong et al., 2025).

Challenges arising from step dependencies. The remaining checks assess whether auditors recognize how linked steps affect scientific conclusions. *(1) Scientific validity.* Approval rates for invalid and valid final conclusions are measured against the workflow's explicit validity conditions. *(2) Error propagation.* For each invalid downstream output, correct tracing requires both rejection and a link to the originating error. Alongside correct tracing, false attribution measures links incorrectly assigned to unaffected outputs. *(3) Correction tracking.* Once corrections are introduced, premature resolution measures the fraction of partial repairs declared fully resolved, while joint verdict and repair-status accuracy evaluates complete repairs. To account for missed initial rejections, conditional scores are accompanied by original-cohort scores that retain these cases.

4 Results and Analysis

The results show that errors in individual judgments accumulate across dependent steps and that auditors often miss invalid conclusions and incomplete repairs. Controlled comparisons further show how workflow structure, available evidence, and auditing capabilities influence these outcomes.

4.1 Auditing reliability across steps

Audit failures accumulate across dependent steps. In Table 1, GPT-5.5 without tools reaches 71.1% step accuracy but correctly audits every step in only 26.3% of workflows, the highest whole-workflow score under the prescribed response format. Its all-correct rate falls from 75.0% at the first step to 41.7% over four steps and 26.3% over the complete workflow. Gemini 2.5 Pro without tools similarly falls from 73.3% to 15.8% in Table 5(a). These results show how imperfect judgments accumulate into unreliable workflow audits.

Insufficient evidence rarely leads to appropriate abstention. GPT-5.4 and Gemini 2.5 Pro never correctly abstain in either tool setting. GPT-5.5's abstention recall is only 6.7% without tools and 3.8% with tools. Both GPT-5.4 and GPT-5.5 issue at least one unsupported definitive verdict in every workflow requiring abstention in Table 1. Confidence does not reliably signal these limitations. In Table 5(b), ECE increases from the first step to the full workflow in 19 of 24 model–tool settings. For GPT-5 mini with tools, mean confidence exceeds accuracy by 5.2 percentage points at the first step and 34.2 points over the full workflow, among valid definitive judgments. Thus, poor uncertainty

Table 1: **Auditing from steps to dependent workflows.** Whole correct requires every step to be audited correctly. Any unjust. measures workflows containing an unsupported definitive verdict among those requiring abstention; Cov. denotes coverage by valid definitive verdicts. Pressure rates use all eligible correct initial rejections. Valid/eligible reports schema-valid reply pairs; N/A means none. Means ± 1 SE; rates in %, ECE on 0–1. Shading indicates tool use. Definitions appear in Appendix E.

Auditor	Tools	Correctness		Abstention		Calibration		Pressure		Response
		All-step acc. ↑	Whole correct ↑	Recall↑	Any unjust. ↓	ECE↓	Cov.	Harmful change ↓	Valid/ eligible	Valid format ↑
GPT-5.5	No	71.1 ±3.5	26.3 ±6.3	6.7 ±5.1	100.0 ±0.0	0.28 ±0.03	99.4 ±0.3	2.4 ±1.7	82/83	99.8 ±0.1
	Yes	69.2 ±3.7	25.4 ±5.3	3.8 ±3.8	100.0 ±0.0	0.30 ±0.04	99.7 ±0.2	2.9 ±1.9	82/82	100.0 ±0.0
GPT-5.4	No	64.0 ±3.5	19.2 ±5.6	0.0 ±0.0	100.0 ±0.0	0.35 ±0.03	99.9 ±0.1	2.4 ±1.6	69/69	99.9 ±0.1
	Yes	62.8 ±3.2	17.1 ±5.1	0.0 ±0.0	100.0 ±0.0	0.36 ±0.03	99.9 ±0.1	3.9 ±1.6	85/85	99.9 ±0.1
GPT-5	No	43.8 ±2.7	2.9 ±1.8	24.2 ±4.6	89.2 ±4.2	0.46 ±0.03	80.8 ±3.2	0.0 ±0.0	53/54	97.1 ±0.7
	Yes	53.2 ±2.8	9.2 ±2.6	5.8 ±2.8	100.0 ±0.0	0.41 ±0.03	93.3 ±1.2	3.3 ±2.2	56/58	96.1 ±0.7
GPT-5 mini	No	48.6 ±3.2	12.1 ±2.0	0.0 ±0.0	100.0 ±0.0	0.38 ±0.03	97.0 ±0.7	14.3 ±14.3	12/13	97.0 ±0.7
	Yes	51.8 ±2.9	8.8 ±2.0	0.0 ±0.0	100.0 ±0.0	0.34 ±0.03	96.6 ±0.8	14.8 ±7.6	25/27	96.6 ±0.8
GPT-5 nano	No	26.0 ±1.9	0.0 ±0.0	2.5 ±1.5	86.7 ±4.7	0.18 ±0.02	56.1 ±2.7	0.0 ±0.0	2/15	59.6 ±2.4
	Yes	27.2 ±1.7	1.3 ±0.6	20.8 ±5.2	86.7 ±5.0	0.16 ±0.03	50.5 ±4.8	N/A	0/3	72.2 ±2.4
GPT-4.1	No	52.9 ±3.1	6.7 ±2.7	11.2 ±6.6	100.0 ±0.0	0.47 ±0.03	98.3 ±0.7	0.0 ±0.0	43/44	99.7 ±0.2
	Yes	51.7 ±3.4	3.3 ±1.5	14.3 ±6.8	100.0 ±0.0	0.47 ±0.03	95.9 ±1.3	0.0 ±0.0	41/41	99.9 ±0.1
Opus 4.5	No	14.8 ±2.2	0.0 ±0.0	10.6 ±4.2	12.5 ±10.0	0.18 ±0.08	16.5 ±2.6	N/A	0/10	18.5 ±2.6
	Yes	0.6 ±0.3	0.0 ±0.0	0.0 ±0.0	0.0 ±0.0	0.03 ±0.01	0.6 ±0.3	N/A	0/3	0.6 ±0.3
Sonnet 4.5	No	15.3 ±1.9	0.0 ±0.0	9.8 ±7.4	58.8 ±10.7	0.38 ±0.06	24.7 ±2.6	N/A	0/18	25.5 ±2.5
	Yes	18.2 ±1.5	0.0 ±0.0	5.0 ±1.5	60.8 ±9.6	0.29 ±0.04	26.0 ±2.0	2.8 ±2.8	6/34	26.9 ±2.0
Haiku 4.5	No	47.0 ±4.4	2.9 ±2.2	0.0 ±0.0	92.5 ±4.0	0.37 ±0.04	80.0 ±3.4	0.0 ±0.0	45/58	80.3 ±3.4
	Yes	23.7 ±3.3	0.0 ±0.0	0.8 ±0.8	90.4 ±6.7	0.32 ±0.04	35.6 ±4.1	3.1 ±3.1	15/39	35.8 ±4.0
Gemini 2.5 Pro	No	65.0 ±2.9	15.8 ±3.7	0.0 ±0.0	100.0 ±0.0	0.34 ±0.03	99.1 ±0.4	1.7 ±1.7	59/59	99.1 ±0.4
	Yes	25.3 ±1.5	0.0 ±0.0	0.0 ±0.0	72.9 ±8.5	0.33 ±0.03	38.3 ±2.1	0.0 ±0.0	5/22	38.3 ±2.1
Gemini 2.5 Flash	No	28.3 ±3.2	0.4 ±0.4	0.0 ±0.0	73.8 ±10.6	0.36 ±0.04	43.8 ±3.9	0.0 ±0.0	3/22	43.8 ±3.9
	Yes	45.5 ±2.8	0.8 ±0.6	0.6 ±0.6	97.5 ±2.5	0.44 ±0.03	82.1 ±1.7	0.0 ±0.0	27/33	82.3 ±1.6
Kimi K2.6	No	40.0 ±4.6	5.8 ±3.0	0.8 ±0.8	100.0 ±0.0	0.46 ±0.04	76.8 ±3.5	1.1 ±1.1	72/91	76.9 ±3.5
	Yes	68.3 ±3.3	20.4 ±5.5	2.3 ±1.2	95.4 ±2.4	0.23 ±0.03	89.2 ±1.8	0.0 ±0.0	84/96	89.6 ±1.7

recognition is accompanied by greater overconfidence in some settings. These increases accompany changes in the steps being audited and in the fraction of responses that contain a definitive verdict.

Unsupported pressure can overturn a correct rejection. Paired follow-ups preserve the evidence but vary whether a reply pressures the auditor to approve. With tools, GPT-5 mini makes a harmful change at 14.8% of eligible checkpoints, whereas Kimi K2.6 makes none, as shown in Table 9(a). Only correctly rejected outputs receive these follow-ups: GPT-5 mini and Kimi initially detect errors at 27 and 96 of 115 prepared checkpoints, respectively. Across the workflows selected for this test, GPT-5.4's whole-workflow success with tools falls from 15.0% under neutral follow-ups to 11.7% under pressure. Thus, pressure can reverse an initially correct judgment, while a low harmful-change rate can coexist with missed initial errors.

4.2 Scientific validity and error tracking

Auditors approve invalid conclusions and miss their upstream causes. Without tools, GPT-5.5 approves 35.7% of scientifically invalid final conclusions, and Gemini 2.5 Pro approves 68.5%, as reported in Table 2. They also reject and correctly trace only 46.4% and 23.4% of invalid downstream outputs, respectively. The challenge therefore extends beyond checking the final answer to identifying the earlier errors that invalidate it. Kimi K2.6 with tools distinguishes valid and invalid conclusions more successfully, approving 90.7% of the former and 9.6% of the latter. Yet it correctly rejects and

Table 2: **Scientific validity, error propagation and repair recognition.** Rates are mean ± 1 SE (%); eligibility is a count. Shading indicates tool use. Validity and tracing use complete evidence. Repair scores include only correct initial rejections. N/A denotes an unavailable estimate. Definitions appear in Appendix E.

Auditor	Tools	Scientific validity		Error tracing			Repair recognition		
		Invalid accepted ↓	Valid accepted ↑	Origin rejected ↑	Invalid traced ↑	False links ↓	Partial resolved ↓	Complete joint acc. ↑	Eligible cases
GPT-5.5	No	35.7 ±9.7	81.5 ±7.0	75.0 ±4.5	46.4 ±6.7	11.8 ±6.8	23.3 ±7.6	95.8 ±4.2	32/48
	Yes	38.4 ±10.5	83.9 ±7.1	80.0 ±4.3	39.1 ±5.4	10.6 ±4.4	16.0 ±6.8	100.0 ±0.0	29/48
GPT-5.4	No	43.5 ±11.1	82.5 ±10.6	62.5 ±5.2	34.1 ±4.6	33.0 ±9.1	32.3 ±15.5	97.9 ±2.1	26/48
	Yes	47.3 ±13.0	84.4 ±6.8	84.2 ±5.2	30.8 ±5.4	22.4 ±6.8	17.9 ±14.1	100.0 ±0.0	25/48
GPT-5	No	68.0 ±10.3	70.9 ±11.5	62.5 ±6.8	8.3 ±3.0	27.8 ±8.8	33.3 ±23.6	41.7 ±21.0	9/48
	Yes	80.1 ±10.8	88.8 ±8.8	72.5 ±4.8	8.0 ±3.6	26.9 ±13.6	61.1 ±20.0	100.0 ±0.0	8/48
GPT-5 mini	No	96.3 ±3.7	98.0 ±2.0	17.5 ±6.0	7.4 ±2.7	11.0 ±4.1	0.0 ±N/A	0.0 ±N/A	1/48
	Yes	95.4 ±3.7	95.1 ±2.6	43.3 ±5.2	6.3 ±2.1	9.9 ±4.0	0.0 ±N/A	0.0 ±N/A	1/48
GPT-5 nano	No	45.2 ±8.7	57.5 ±8.4	18.3 ±5.7	0.3 ±0.3	2.4 ±2.4	12.5 ±12.5	37.5 ±23.9	6/48
	Yes	48.3 ±12.3	66.3 ±9.9	4.2 ±2.6	0.0 ±0.0	0.0 ±0.0	N/A	N/A	0/48
GPT-4.1	No	91.7 ±4.4	98.1 ±1.3	60.8 ±9.9	6.7 ±2.8	8.2 ±7.0	33.3 ±33.3	66.7 ±33.3	3/48
	Yes	90.7 ±7.4	100.0 ±0.0	55.8 ±9.9	4.6 ±2.4	0.5 ±0.5	25.0 ±25.0	100.0 ±0.0	3/48
Opus 4.5	No	24.4 ±9.2	15.0 ±7.6	15.0 ±3.9	2.9 ±1.7	0.0 ±0.0	0.0 ±N/A	0.0 ±N/A	1/48
	Yes	0.0 ±0.0	0.0 ±0.0	2.5 ±1.8	0.5 ±0.4	0.0 ±0.0	N/A	N/A	0/48
Sonnet 4.5	No	28.0 ±10.5	28.5 ±11.2	12.5 ±3.1	4.9 ±1.8	6.5 ±3.8	16.7 ±16.7	33.3 ±21.1	8/48
	Yes	13.7 ±5.1	26.3 ±5.6	26.7 ±6.4	13.7 ±3.1	5.3 ±3.5	0.0 ±0.0	65.4 ±14.0	20/48
Haiku 4.5	No	45.9 ±9.8	88.0 ±6.6	55.0 ±6.0	11.8 ±3.2	27.1 ±13.9	0.0 ±0.0	25.7 ±9.2	22/48
	Yes	14.3 ±8.0	18.9 ±9.7	28.3 ±6.0	16.5 ±4.5	14.3 ±8.3	14.3 ±14.3	63.6 ±12.9	21/48
Gemini 2.5 Pro	No	68.5 ±10.5	96.3 ±2.7	71.7 ±3.6	23.4 ±3.9	11.5 ±7.7	66.7 ±17.8	100.0 ±0.0	13/48
	Yes	28.1 ±7.2	39.2 ±10.4	39.2 ±3.9	5.2 ±1.3	0.0 ±0.0	0.0 ±N/A	0.0 ±N/A	1/48
Gemini 2.5 Flash	No	16.9 ±6.8	28.8 ±7.6	22.5 ±5.7	3.0 ±1.2	6.3 ±4.1	0.0 ±0.0	62.5 ±23.9	6/48
	Yes	79.5 ±7.2	85.0 ±6.7	40.8 ±8.6	6.0 ±2.2	0.5 ±0.5	37.5 ±23.9	62.5 ±23.9	5/48
Kimi K2.6	No	4.8 ±2.5	20.2 ±10.5	89.2 ±2.8	27.5 ±3.2	24.1 ±7.6	0.0 ±0.0	17.6 ±7.6	30/48
	Yes	9.6 ±7.5	90.7 ±4.2	84.2 ±4.2	55.2 ±4.8	17.9 ±9.5	0.0 ±0.0	91.9 ±4.7	40/48

traces only 55.2% of invalid downstream outputs. Thus, accurate final judgments can coexist with failures to identify how upstream errors affect later outputs.

Auditors can mistake partial repairs for complete corrections. A partial repair restores the modified output but leaves at least one invalid downstream output. Nevertheless, in Table 2, GPT-5.5 without tools declares 23.3% of these repairs resolved, and Gemini 2.5 Pro declares 66.7% resolved. Although GPT-5.5 recognizes 95.8% of complete repairs with the correct verdict and status, this conditional score covers only 32 of 48 prepared cases. Across all prepared cases, including those where the original error was missed, complete-repair success is only 67.2% in Table 9(c). Kimi K2.6 with tools makes no premature resolution claims across its 40 eligible cases. Auditors can therefore recognize complete repairs accurately while still missing original errors and accepting incomplete repairs.

4.3 Ablation studies

The ablations involve many combinations of models, experimental settings, and test cases. To keep their computational cost manageable, comparisons of workflow structure, available information, and auditing capabilities focus on GPT-5.4 and Claude Sonnet 4.5. The primary pressure and repair comparisons cover all twelve models. Appendix F provides the full results.

The effect of workflow length depends on the auditing setting. Across five base workflows, extending the horizon from 8 to 32 steps while preserving the shared target calculations and dependency depth reduces GPT-5.4's tool-enabled accuracy on shared operations from 72.7% to 66.2%.

The paired change is -6.5 percentage points. Without tools, accuracy remains similar at 65.2% and 65.4%, although whole-workflow success falls from 20.0% to 6.7% in Tables 7(a) and 7(c). Whole-workflow success therefore falls even where accuracy on shared operations remains stable. In Table 10(d), Sonnet's no-tools accuracy under the required response format rises from 11.3% to 21.1%, whereas accuracy after extracting valid JSON from completed responses falls from 72.5% to 66.9%. This reversal shows how response formatting can alter the apparent trend in judgment quality. The error-lag effects vary across settings in Table 7(c).

Useful history depends on its contents, not simply its length. Removing all history reduces step accuracy by 6.9 points for both GPT-5.4 without tools and Sonnet with tools. GPT-5.4 with tools achieves 63.3% with either full or ancestor-only history in Table 6(a). The separate comparison in Table 6(b) shows that omitting earlier task descriptions from full history raises its accuracy to 68.0%, a paired gain of 4.7 points. History can therefore support auditing, but including every available field does not consistently improve judgments.

Auditors struggle both to recognize missing evidence and to use accessible evidence. For 128 step outputs, withholding evidence removes the information needed to determine whether the output is correct. GPT-5.4 with tools correctly judges 78.8% of these outputs with complete evidence but abstains on none after withholding. Accuracy on the 148 targets that remain decidable stays nearly unchanged, at 77.8% and 77.6% in Table 8(a). Moving complete evidence from the prompt to a permitted file creates a separate difficulty: accuracy falls from 74.2% to 48.3% on 39 selected targets in Table 8(b). Providing tools to retrieve evidence does not ensure that the auditor uses it effectively.

Tool access and larger budgets do not consistently improve auditing. In Table 1, the accuracy change from enabling tools is $+28.4$ points for Kimi K2.6 and -1.9 for GPT-5.5. These changes reflect both verdict correctness and whether the auditor returns a complete response in the required format. In Table 6(e), budgets of two, eight, and sixteen calls yield GPT-5.4 accuracies of 66.0%, 63.3%, and 65.3%. These comparisons provide no clear evidence of a benefit from either reducing or increasing the budget. The script-capability comparison in Table 6(d) covers one base workflow; its findings may not extend to other workflows.

More conspicuous errors do not necessarily yield a correct workflow audit. In Table 7(d), increasing numerical error magnitude raises GPT-5.4's tool-enabled step accuracy from 56.2% to 68.5%, yet whole-workflow success remains 10.0%. Correct rejection and tracing of invalid downstream outputs improve from 18.3% to 39.2%, but false attributions to valid outputs also rise from 19.0% to 52.4%. Making the initial error easier to see therefore does not ensure accurate identification of all its consequences. Comparisons across error types and locations also involve different tasks, so score differences may reflect task composition as well as the perturbation.

Response compliance can obscure the quality of scientific judgments. Claude Opus 4.5 with tools produces only 9 schema-valid responses in 1,656 audits. In Table 10(c), extracting an unchanged valid JSON object from completed responses raises its step accuracy from 0.6% to 83.6% and whole-workflow success from 0% to 46.7%. Similarly, removing justification reduces Sonnet's tool-enabled strict accuracy from 19.3% to 0.5%, alongside format compliance falling from 26.8% to 0.5%. After the same JSON extraction, however, accuracy rises from 73.3% to 76.1% in Tables 6(c) and 10(d). These gaps show that many responses fail the required format even when they contain a correct verdict.

5 Conclusion

AuditSpan evaluates LLM auditors on scientific workflows with explicit step dependencies, independently verified outputs, and controlled errors and evidence gaps. The results show that success on individual outputs can coexist with incorrect workflow audits, insufficient abstention, and failures to track the consequences of errors and corrections. The controlled comparisons further show that history, tools, evidence access, and response requirements affect observed performance. These findings motivate auditing methods that assess both the available evidence and the dependencies supporting a scientific conclusion.

LIMITATIONS

The benchmark covers ten synthetic base workflows with explicit validity conditions. Its main workflows contain four to ten steps and mostly sequential dependencies; the matched length extension covers five base workflows, and the depth control uses one base workflow. Error-operator coverage is uneven. Generalization to open-ended research, novelty judgments, and real scientific projects remains untested. Model comparisons depend on the specified decoding settings and response requirements. Pressure and repair scores condition on correct initial rejection, so eligible subsets differ across models; the accompanying original-cohort scores retain missed detections. These constraints limit claims about universal effects of workflow length, model capability, or repeated interaction.

AI USE STATEMENT

An AI assistant supported literature search and retrieval, code execution, synthetic data generation, and manuscript editing.

REFERENCES

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D. Hwang, Peter Jansen, Varsha Kishore, et al. AstaBench: Rigorous benchmarking of AI agents with a scientific research suite. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.21652.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, et al. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery, 2024. URL https://arxiv.org/abs/2410.05080.

Haoran Cheng, Joy Le Yi Wong, Chrystie Wan Ning Quek, Jeffrey L. Goldberg, Vinit B. Mahajan, Tien Yin Wong, Jodhbir S. Mehta, Daniel S. W. Ting, and Darren S. J. Ting. Ophthalmic drug discovery and development using artificial intelligence and digital health technologies. *npj Digital Medicine*, 8:573, 2025. doi: 10.1038/s41746-025-01954-y. URL https://doi.org/10.1038/s41746-025-01954-y.

Ken Gu, Ruoxi Shang, Ruien Jiang, Keying Kuang, et al. BLADE: Benchmarking language model agents for data-driven science, 2024. URL https://arxiv.org/abs/2408.09667.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

Jiseung Hong, Grace Byun, Seungone Kim, and Kai Shu. Measuring sycophancy of language models in multi-turn dialogues. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 2239–2259. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.121. URL https://aclanthology.org/2025.findings-emnlp.121/.

Andrej Karpathy. autoresearch: AI agents running research on single-GPU nanochat training automatically. GitHub repository, 2026. URL https://github.com/karpathy/autoresearch. Accessed 2026-09-24.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. AbstentionBench: Reasoning LLMs fail on unanswerable questions. *arXiv preprint arXiv:2506.09038*, 2025. URL https://arxiv.org/abs/2506.09038.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

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.

Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher J. Pal, and Siva Reddy. AgentRewardBench: Evaluating automatic evaluations of web agent trajectories. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=fQcUZMPIvu.

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, 2024. URL https://arxiv.org/abs/2407.01725.

Nat McAleese, Rai Michael Pokorny, Juan Felipe Ceron Uribe, Evgenia Nitishinskaya, Maja Trebacz, and Jan Leike. LLM critics help catch LLM bugs. *arXiv preprint arXiv:2407.00215*, 2024. URL https://arxiv.org/abs/2407.00215.

Ludovico Mitchener, Jon M. Laurent, Alex Andonian, Benjamin Tenmann, Siddharth Narayanan, Geemi P. Wellawatte, Andrew White, Lorenzo Sani, and Samuel G. Rodriques. BixBench: a comprehensive benchmark for LLM-based agents in computational biology, 2025a. URL https://arxiv.org/abs/2503.00096.

Ludovico Mitchener, Angela Yiu, Benjamin Chang, Mathieu Bourdenx, Tyler Nadolski, Arvis Sulovari, Eric C. Landsness, Daniel L. Barabasi, Siddharth Narayanan, Nicky Evans, Shriya Reddy, Martha Foiani, Aizad Kamal, Leah P. Shriver, Fang Cao, Asmamaw T. Wassie, Jon M. Laurent, Edwin Melville-Green, Mayk Caldas, Albert Bou, Kaleigh F. Roberts, Sladjana Zagorac, Timothy C. Orr, Miranda E. Orr, Kevin J. Zwezdaryk, Ali E. Ghareeb, Laurie McCoy, Bruna Gomes, Euan A. Ashley, Karen E. Duff, Tonio Buonassisi, Tom Rainforth, Randall J. Bateman, Michael Skarlinski, Samuel G. Rodriques, Michaela M. Hinks, and Andrew D. White. Kosmos: An AI scientist for autonomous discovery. *arXiv preprint arXiv:2511.02824*, 2025b. URL https://arxiv.org/abs/2511.02824.

Ricursive Intelligence. Recursive self-improvement via AI for chip design and chip design for AI, 2026. URL https://www.ricursive.com/. Accessed 2026-09-24.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*, 2022. URL https://arxiv.org/abs/2206.05802.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using LLM agents as research assistants. *arXiv preprint arXiv:2501.04227*, 2025. URL https://arxiv.org/abs/2501.04227.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=tvhaxkMKAn.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. *arXiv preprint arXiv:2504.01848*, 2025. URL https://arxiv.org/abs/2504.01848.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica. JudgeBench: A benchmark for evaluating LLM-based judges. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.12784.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, et al. SciCode: A research coding benchmark curated by scientists, 2024. URL https://arxiv.org/abs/2407.13168.

Zailong Tian, Zhuoheng Han, Yanzhe Chen, Haozhe Xu, Xi Yang, Richeng Xuan, Houfeng Wang, and Lizi Liao. Overconfidence in LLM-as-a-judge: Diagnosis and confidence-driven solution. *arXiv preprint arXiv:2508.06225*, 2025. URL https://arxiv.org/abs/2508.06225.

Abhigya Verma, Amit Kumar Saha, Seganrasan Subramanian, and Sai Harshitha Aluru. Agent-JudgeBench: A multi-difficulty benchmark for evaluating LLM judges on agentic tool-calling. *arXiv preprint arXiv:2608.26623*, 2026. URL https://arxiv.org/abs/2608.26623.

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.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pp. 1009–1024. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.50. URL https://aclanthology.org/2025.acl-long.50/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36, 2023. URL https://arxiv.org/abs/2306.05685.

Mingchen Zhuge, Changsheng Zhao, Dylan R. Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-judge: Evaluate agents with agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80569–80611. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhuge25a.html.