FLUIDBENCH: DIAGNOSING THE KNOWING–DOING GAP IN SCIENTIFIC AI THROUGH FLUID MECHANICS

Anonymous authors
Paper under double-blind review

ABSTRACT

As large language models and AI agents move from answering scientific questions toward solving real scientific problems, a critical question is whether scientific knowledge can be reliably translated into scientific action. Fluid mechanics offers a revealing testbed: success requires both applying physical knowledge to concrete problems and sustaining reliable computational research workflows. Existing benchmarks, however, typically evaluate scientific knowledge, problem solving, or research execution in isolation. We introduce **FluidBench**, a multi-level benchmark for diagnosing the *knowing–doing gap* in fluid mechanics. **FluidBench-Fundamental** contains 625 textbook-grounded problems assessing foundational knowledge; **FluidBench-Application** comprises 329 multi-stage problems with matched prerequisite probes and six-dimensional failure diagnosis; and **FluidBench-Research** contains 10 open-ended tasks across five research topics for evaluating end-to-end scientific agents. Experiments reveal two distinct gaps. First, a *knowledge-to-application gap*: models frequently answer all prerequisite probes correctly yet fail the original problem, with physical modeling emerging as the major bottleneck. Second, a *knowledge-to-research gap*: despite strong foundational fluid-mechanics performance, current agent systems remain far below reference-study quality, with failures concentrated in sustaining controlled experiments and complete, verifiable evidence. Together, these results show that strong scientific knowledge alone is insufficient for either reliable problem-specific reasoning or autonomous scientific research.

1 INTRODUCTION

Scientific capability requires more than possessing domain knowledge: an AI system must also be able to apply that knowledge in concrete problems and sustain reliable scientific workflows. However, existing scientific benchmarks largely emphasize different parts of this process. Closed-ended evaluations assess scientific knowledge and reasoning through textbook (Xu et al., 2025), examination (Shen et al., 2025; Feng et al., 2025), competition (He et al., 2024; Zhang et al., 2025), and expert-curated problems (Phan et al., 2026; Zhu et al., 2026; Qiu et al., 2025), whereas recent agent benchmarks focus on scientific coding (Tian et al., 2024; Chen et al., 2025), paper reproduction (Qiu et al., 2026; Starace et al., 2025) and autonomous research (Xu et al., 2026). What remains less understood is *where the transition from knowing to doing breaks down*: whether failure reflects missing scientific knowledge, difficulty applying that knowledge to a specific problem, or an inability to execute a longer scientific workflow reliably.

Fluid mechanics provides a natural setting for studying this gap. Solving applied fluid-mechanics problems requires more than recalling physical laws: LLMs must identify assumptions, select and combine governing relations, construct an appropriate physical model, perform the required mathematics, and verify the result. Scientific research imposes a different and more extended form of doing, requiring method development, implementation, experimental design, evaluation, evidence collection, and scientific interpretation. Despite these demands, existing fluid-mechanics benchmarks (Tali et al., 2024; Bonnet et al., 2023; Wang et al., 2026b) primarily evaluate numerical solvers or scientific-machine-learning models. Recent studies like CFDLLMBench (Somasekharan et al., 2026) extend evaluation to language models and agents, but remain centered only on CFD-specific numerical workflows. A broader evaluation of scientific knowing and doing for LLMs and agents in fluid mechanics is still missing.

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.



Figure 1: **Overview of FluidBench. FluidBench-Fundamental** evaluates foundational fluid-mechanics knowledge. **FluidBench-Application** examines whether models can operationalize relevant knowledge to solve concrete physical problems. **FluidBench-Research** evaluates whether AI agents can execute reliable open-ended scientific workflows. Together, the three parts diagnose the gap between *scientific knowing* and *scientific doing*.

As illustrated in Figure 1, we introduce **FluidBench**, a multi-level benchmark that evaluates foundational knowledge, applied problem solving and open-ended research execution. **FluidBench-Fundamental** contains 625 textbook-grounded problems assessing foundational fluid-mechanics knowledge. **FluidBench-Application** contains 329 multi-stage applied problems and diagnoses failures along six dimensions. To distinguish application failures from missing prerequisite knowledge, we further introduce *matched prerequisite probes*: for each failed application problem, we independently test whether the model can recall the concepts, governing relations, and applicability conditions required by the reference solution. **FluidBench-Research** contains 10 open-ended tasks across five research topics, requiring agents to design methods, execute experiments, analyze results, and support conclusions with traceable evidence. Together, these components provide complementary views of what models know, whether they can apply that knowledge correctly, and whether they can sustain reliable scientific research workflows.

Our experiments reveal two distinct gaps. First, a *knowledge-to-application gap*: frontier LLMs perform strongly on foundational questions, and matched probes show that many application failures persist even when the required prerequisite knowledge can be recalled. These failures concentrate in physical modeling, suggesting that the main difficulty lies in mapping knowledge into a concrete physical setting. Second, a *knowledge-to-research gap*: despite strong foundational performance, agents remain far below reference-study quality on end-to-end research tasks, failing not at a single local step but in sustaining scientifically reliable workflows.

We summarize the contributions of this paper as follows:

- **A multi-level benchmark for LLMs and scientific agents in fluid mechanics**, covering foundational knowledge, applied physical reasoning, and open-ended scientific research.
- **A diagnostic framework for the knowing–doing transition**, combining matched prerequisite probes, six-dimensional analysis of application-level failures, and task-specific evaluation of end-to-end research workflows.
- **Empirical evidence of a substantial knowing–doing gap**: models may possess the prerequisite knowledge for applied problems yet fail to operationalize it correctly, and strong foundational fluid-mechanics knowledge does not guarantee reliable autonomous research.

2 Related Work

Physics Reasoning Benchmarks. Recent progress in LLM scientific reasoning (Li et al., 2025) has motivated increasingly challenging physics benchmarks. UGPhysics (Xu et al., 2025), PHYBench

(Qiu et al., 2025), and PhyX (Shen et al., 2025) evaluate physical reasoning through university- and competition-level problems, while HLE (Phan et al., 2026) and CritPt (Zhu et al., 2026) introduce expert-authored frontier questions to better probe difficult and out-of-distribution reasoning. These benchmarks provide strong tests of scientific knowledge and closed-ended problem solving, but final answer performance alone does not reveal whether a failure arises from missing prerequisite knowledge or from difficulty applying available knowledge in a problem-specific setting. FluidBench explicitly targets this distinction through matched prerequisite probes and fine-grained diagnosis of application-level failures.

Scientific Agent Benchmarks. Scientific agent benchmarks increasingly evaluate LLM-based agents on realistic research workflows (Wang et al., 2026a). SciCode (Tian et al., 2024) and ScienceAgentBench (Chen et al., 2025) focus on scientific coding and data-driven tasks, while PaperBench (Starace et al., 2025) and PRBench (Qiu et al., 2026) evaluate paper reproduction. ResearchClawBench (Xu et al., 2026) further targets autonomous research by asking agents to complete scientific research tasks from literature and data independently. These benchmarks substantially advance the evaluation of scientific *doing*, but typically assess research performance without explicitly connecting it to the system's underlying scientific knowledge. FluidBench complements this line of work by evaluating both scientific *knowing* and *doing* within fluid mechanics, spanning foundational knowledge, applied problem solving, and autonomous research.

Fluid Mechanics Benchmarks. Existing fluid-mechanics benchmarks primarily evaluate scientific machine learning models. FlowBench (Tali et al., 2024), AirfRANS (Bonnet et al., 2023), and TransportBench(Wang et al., 2026b) measure predictive accuracy and generalization across flow conditions, physical parameters, and geometries. Their evaluation target is typically a surrogate model rather than an LLM or agent. CFDLLMBench (Somasekharan et al., 2026) recently extends evaluation to LLMs through CFD knowledge, PDE-solver code generation, and OpenFOAM-based agent tasks, but remains centered only on CFD numerical workflows. FluidBench broadens this scope by evaluating LLMs and scientific agents across foundational knowledge, applied reasoning, and open-ended research within fluid mechanics.

3 FLUIDBENCH

Overview. FluidBench is a multi-level benchmark for evaluating LLMs and scientific agents in fluid mechanics across foundational knowledge, applied problem solving, and open-ended research. **FluidBench-Fundamental** contains **625** textbook-grounded questions covering core concepts, physical laws, and standard calculations in fluid mechanics. **FluidBench-Application** comprises **329** multi-stage problems that require models to interpret physical conditions, select appropriate governing relations, construct problem-specific models, and carry out the resulting calculations. It further incorporates matched prerequisite probes and a six-dimensional diagnostic framework to distinguish missing prerequisite knowledge from failures arising during problem solving. **FluidBench-Research** consists of **10** open-ended tasks across five research topics, evaluating scientific agents on method development, computational experimentation, result analysis, and evidence-supported scientific conclusions. Together, the three components provide complementary evaluations of foundational knowledge, its application to concrete fluid-mechanics problems, and the execution of extended scientific research workflows.

3.1 FLUIDBENCH-FUNDAMENTAL: TEXTBOOK-GROUNDED KNOWLEDGE

Textbook-Grounded Construction. FluidBench-Fundamental is constructed from six representative textbooks: *Dimensional Analysis* (Tan, 2011), *Fluid Mechanics* (White, 2003), *CFD Principles and Applications* (Blazek, 2005), *Hypersonic and High-Temperature Gas Dynamics* (Anderson, 2019), *Modern Compressible Flow* (Anderson, 2021), and *Fundamentals of Aerodynamics* (Anderson & Cadou, 2023). Together, these sources span major areas of fluid mechanics. As shown in Figure 2(a), each textbook is parsed with MinerU (Wang et al., 2024) and organized into chapter-level source chunks that preserve the textual content, figures, and document hierarchy.

Candidate questions are generated exclusively from their corresponding source chunks to maintain bounded and traceable grounding. We use Gemini 3.5 Flash(Google DeepMind, 2026a) as the question generator. The generation process permits each source chunk to yield multiple questions or

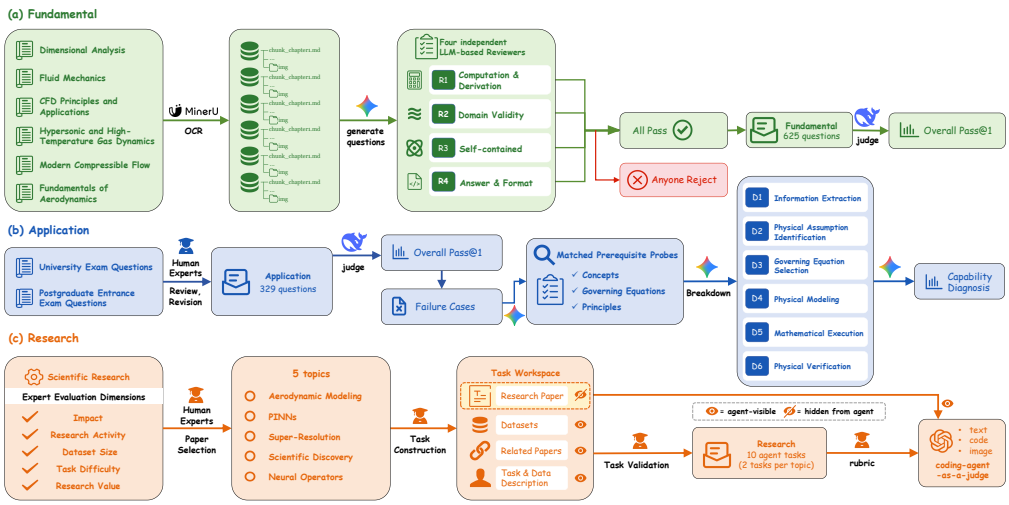


Figure 2: **Overview of the construction and evaluation pipeline of FluidBench.**

none, depending on whether it contains suitable assessment content. We retain objectively gradable multiple-choice, fill-in-the-blank, and numerical or symbolic calculation problems with uniquely verifiable answers.

Quality Control. Each candidate question is independently examined by four LLM-based reviewers covering *computation and derivation*, *domain validity*, *self-containedness*, and *answer and format validity*. A candidate is retained only when all applicable reviewers approve it. As a final quality check, domain experts further audit a stratified sample of the retained questions. The resulting benchmark contains **625** questions. Additional details on question generation, LLM-based review and human verification are provided in Appendix A.

Evaluation Protocol. Each question is answered once by the evaluated model, with **pass@1** as the evaluation metric. Each response is independently graded by DeepSeek V4 Flash (DeepSeek, 2026) against the reference answer.

3.2 FluidBench-Application: Applied Multi-Stage Reasoning

Problem Curation and Expert Refinement. FluidBench-Application contains **329** applied fluid-mechanics problems collected from textbook exercises, postgraduate entrance-examination problems, and expert-adapted questions. As illustrated in Figure 2(b), domain experts review both the problem statements and reference solutions, correcting ambiguous or incomplete formulations and reconstructing problems when necessary.

These problems require models to coordinate multiple stages of reasoning within a concrete physical setting, including extracting relevant information, identifying physical assumptions, selecting governing equations, constructing a problem-specific physical model, carrying out the required mathematics, and verifying the resulting solution. Our diagnostic design separates two questions: whether LLMs have a good grasp of the prerequisite knowledge required by the problem, and where the reasoning process fails when solving the original problem.

Matched Prerequisite Probes. To assess whether the model has mastered the prerequisite knowledge required for applied fluid-mechanics problems, we construct a set of matched prerequisite probes based on the reference solution. Using Gemini 3.5 Flash, we identify a minimal and complete set of foundational knowledge required by the reference solution; the number of prerequisites is determined by the problem.

Each prerequisite is instantiated as an open-ended recall question, restricted to fill-in-the-blank or symbolic-expression formats. Candidate probes are retained only after an independent audit verifies that they are (i) necessary for the reference solution, (ii) restricted to foundational knowledge, (iii) independently answerable without the parent problem, and (iv) uniquely and correctly answerable.

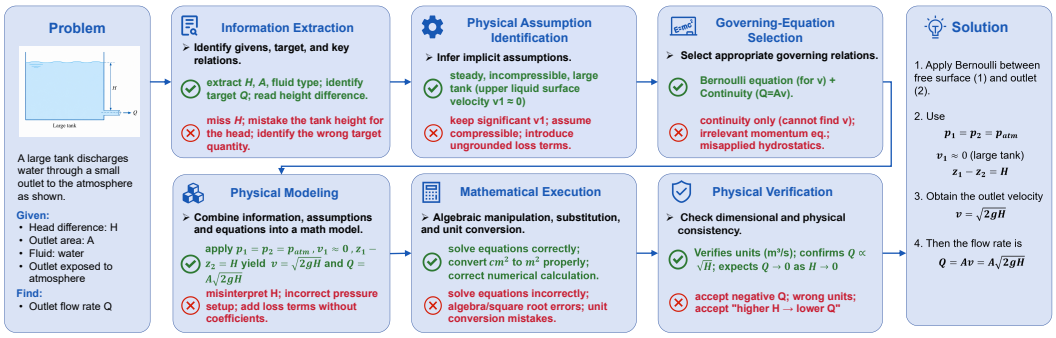


Figure 3: **Six-dimensional diagnostic framework for FluidBench-Application** , illustrated with a large-tank outflow problem. The framework decomposes applied fluid-mechanics problem solving into six dimensions: *information extraction*, *physical assumption identification*, *governing-equation selection*, *physical modeling*, *mathematical execution*, and *physical verification*. Green and red examples illustrate representative correct reasoning and failure modes at each stage.

During evaluation, each accepted probe is presented to the target model independently. Probe responses are graded by DeepSeek V4 Flash. Detailed probe-generation prompts and audit criteria are provided in Appendix B.

Six-Dimensional Failure Diagnosis. To localize failures in applied fluid-mechanics problem solving, we define a six-dimensional diagnostic framework spanning *Information Extraction*, *Physical Assumption Identification*, *Governing-Equation Selection*, *Physical Modeling*, *Mathematical Execution*, and *Physical Verification*. As illustrated in Figure 3, these dimensions trace the reasoning process from interpreting the problem to validating the final solution.

The six-dimensional diagnosis is performed by Gemini 3.5 Flash, which receives the original problem, any accompanying figure, the reference solution, and the model response. To separate the source of an error from its subsequent propagation, we assign diagnosed failures to three roles. A *primary failure* is the earliest decisive error sufficient to derail the solution; a *contributing failure* is an additional independent error that affects the solution but is not its earliest decisive cause; and a *downstream consequence* is an error that arises as a result of an upstream failure. When multiple primary failures are identified, the main analysis retains the earliest one in the diagnostic ordering. Cases for which no primary failure can be reliably determined are marked as unresolved and excluded from the normalized failure statistics.

For model m, we define the conditional primary failure rate as

$$\mathrm{PFR}_{m,i} = \frac{n_{m,i}^{\mathrm{primary}}}{n_m^{\mathrm{wrong}} - n_m^{\mathrm{unres}}}, \tag{1}$$

where $n_{m,i}^{\mathrm{primary}}$ is the number of responses whose primary failure is assigned to dimension D_i, n_m^{wrong} is the total number of incorrect responses, and n_m^{unres} is the number of unresolved cases. Detailed diagnostic criteria are provided in Appendix B.

3.3 FluidBench-Research: Autonomous Open-Ended Research

FluidBench-Research evaluates an agent's ability to carry out extended, open-ended scientific research. Each task specifies a scientific objective and requires the agent to develop a research strategy, perform computational experiments, analyze results, and support its conclusions with traceable evidence.

Research Task Construction and Execution. Inspired by the paper-grounded task formulation of ResearchClawBench (Xu et al., 2026), we use published studies as anchors for constructing open-ended research tasks. As illustrated in Figure 2(c), domain experts select representative studies in fluid mechanics according to five criteria: *Impact*, *Research Activity*, *Dataset Size*, *Task Difficulty*, and *Research Value*. The benchmark covers five topics—**Aerodynamic Modeling** (Ma et al.,

2024; Bonnet et al., 2023), **Scientific Discovery** (Callaham et al., 2021; Xie et al., 2022), **Physics-Informed Neural Networks (PINNs)** (Raissi et al., 2020; Hao et al., 2023), **Neural Operators** (Berner et al., 2026; Cheng & Peng, 2023) and **Super-Resolution** (Shu et al., 2023; Amorós-Trepat et al., 2026)—with two tasks per topic, yielding **10** research tasks.

For each study, experts identify the central scientific question and the evidence needed to support its conclusions, then reframe them as an open research objective. Each task is instantiated with curated data, experimental resources, and relevant literature sufficient for independent investigation. The agent works in an isolated workspace, developing code, running experiments, producing intermediate artifacts, and writing a final report. Evaluation materials—including the target study and task-specific criteria—remain hidden from the evaluated agent.

Workspace-Aware, Paper-Relative Evaluation. We evaluate the complete research workspace using *Coding-Agent-as-a-Judge*, implemented with Codex (OpenAI, 2026a) driven by GPT-5.6 Luna (OpenAI, 2026b). The judge jointly examines the report, source code, figures, and experimental artifacts, enabling scientific claims to be checked against their supporting evidence. Each task is scored using a task-specific weighted rubric that reflects its scientific objectives and evidence requirements. Each criterion is scored on a 0–100 scale, where a score around **50** denotes scientific quality and supporting evidence comparable to the target study, while higher scores require stronger validation or additional well-supported findings.

Because open-ended research criteria inevitably involve qualitative judgment, rubric scores alone may be sensitive to evaluator interpretation. We therefore augment the rubric with task-specific **hard caps** that encode critical scientific-validity requirements as predefined score ceilings. Hard caps both prevent severe methodological or evaluation failures from being compensated by strengths elsewhere in the rubric and provide structured signals for analyzing failures in the agent's research process.

For rubric criterion i, let r_i denote its rubric score, c_{ij} the score ceiling imposed by hard cap j, and $z_{ij} \in \{0, 1\}$ indicate whether that cap is triggered. The final score for criterion i is

$$s_i = \begin{cases} r_i, & \text{if no hard cap is triggered,} \\ \min\left(r_i, \min\limits_{j:z_{ij}=1} c_{ij}\right), & \text{otherwise.} \end{cases} \tag{2}$$

A triggered hard cap therefore restricts the score of the affected criterion to the most stringent applicable ceiling. The final task score is computed as the weighted average of all rubric criteria:

$$S = \sum_i w_i s_i, \qquad \sum_i w_i = 1. \tag{3}$$

where w_i denotes the predefined weight of criterion i.

4 Experiments

4.1 Experimental Setup

We evaluate eight recent language models on FluidBench-Fundamental and FluidBench-Application: MiniMax M3 (MiniMax, 2026), GLM-5.3-Flash (GLM-5-Team, 2026), Qwen3.8-27B (Qwen Team, 2026b), Qwen3.5-397B-A17B (Qwen Team, 2026a), Gemini 3.6 Flash (Google DeepMind, 2026b), GPT-5.6 Sol (OpenAI, 2026c), Claude Sonnet 4.6 (Anthropic, 2026b), and Claude Opus 4.8 (Anthropic, 2026a). We report pass@1 accuracy on both levels and additionally analyze primary failure dimensions on FluidBench-Application.

For FluidBench-Research, we evaluate complete agent systems built with Codex (OpenAI, 2026a) and Claude Code (Anthropic, 2026c), paired with different backbone models, using the workspace-aware rubric score defined in Section 3.3. Full inference settings, agent configurations, and evaluation details are provided in Appendix D.

Table 1: **Performance on FluidBench-Fundamental.** Accuracy (%) across six textbook domains and overall. OW denotes open-weight models and Prop. denotes proprietary models. DA = *Dimensional Analysis*; FM = *Fluid Mechanics*; CFD = *CFD Principles and Applications*; HHGD = *Hypersonic and High-Temperature Gas Dynamics*; MCF = *Modern Compressible Flow*; FA = *Fundamentals of Aerodynamics*.

Model	Access	DA	FM	CFD	HHGD	MCF	FA	Overall
Qwen3.8-27B	OW	77.78	**98.06**	69.49	78.67	85.00	89.05	85.92
Qwen3.5-397B-A17B	OW	85.19	97.09	79.66	**89.33**	91.88	93.03	91.36
MiniMax M3	OW	**92.59**	91.26	72.88	70.67	82.50	89.05	84.16
GLM-5.3-Flash	OW	81.48	93.20	81.36	85.33	85.00	89.05	87.20
GPT-5.6 Sol	Prop.	85.19	93.20	84.75	82.67	88.75	89.05	88.32
Claude Sonnet 4.6	Prop.	85.19	86.41	76.27	78.67	84.38	85.07	83.52
Claude Opus 4.8	Prop.	81.48	87.38	77.97	76.00	83.13	86.07	83.36
Gemini 3.6 Flash	Prop.	85.19	97.09	**88.14**	82.67	**93.13**	**96.02**	**92.64**

Table 2: **Performance and failure diagnosis on FluidBench-Application.** Acc. denotes final-answer accuracy (%). All-Probe Pass is the percentage of probe-eligible incorrect problems for which the model answers all matched prerequisite probes correctly. D_1–D_6 report the primary failure rate (PFR, %) among resolved incorrect responses. For the PFR columns, darker and lighter blue indicate the largest and second-largest failure rates within each model, respectively.

Model	Acc.	All-Probe Pass	Primary Failure Rate (%)					
			D_1	D_2	D_3	D_4	D_5	D_6
Qwen3.8-27B	60.79	82.03	18.25	5.56	**27.78**	**39.68**	8.73	0.00
Qwen3.5-397B-A17B	67.78	**94.29**	**21.90**	6.67	18.10	**44.76**	8.57	0.00
MiniMax M3	59.27	87.12	**21.80**	7.52	13.53	**50.38**	6.77	0.00
GLM-5.3-Flash	65.65	91.07	**19.64**	4.46	9.82	**53.57**	11.61	0.89
GPT-5.6 Sol	67.48	89.72	15.24	7.62	**26.67**	**40.00**	9.52	0.95
Claude Sonnet 4.6	56.53	79.43	**21.83**	7.04	16.90	**42.96**	11.27	0.00
Claude Opus 4.8	54.71	83.56	**20.81**	3.36	16.11	**44.97**	14.77	0.00
Gemini 3.6 Flash	**72.34**	91.11	13.64	2.27	**20.45**	**50.00**	13.64	0.00

4.2 FluidBench-Fundamental: Strong Foundational Knowledge

Table 1 reports performance on FluidBench-Fundamental. Overall, the evaluated models demonstrate strong foundational fluid-mechanics knowledge, with all models exceeding 80% accuracy, and this strong performance extends across both open-weight and proprietary models. Gemini 3.6 Flash achieves the highest overall accuracy of **92.64%**, followed by the open-weight Qwen3.5-397B-A17B at **91.36%**. However, performance still varies across domains: models generally perform better on *Fluid Mechanics* and *Fundamentals of Aerodynamics*, whereas *CFD Principles and Applications* remains comparatively challenging for several models.

4.3 FluidBench-Application: Diagnosing the Knowledge-to-Application Gap

FluidBench-Application examines whether strong foundational knowledge can be converted into reliable problem-specific reasoning. Table 2 reports final-answer accuracy, matched-prerequisite performance, and primary-failure rates on FluidBench-Application. Performance decreases markedly relative to FluidBench-Fundamental: the best-performing model reaches 72.34% accuracy on the application problems, despite the strong foundational performance observed in Section 4.2.

Matched prerequisite probes first examine whether models possess the foundational knowledge required to solve each failed application problem. For each incorrectly answered problem, the probes independently test the concepts, physical relations, and governing equations required for its solution. Across all evaluated LLMs, the All-Probe Pass rate remains high, ranging from 79.43% to 94.29%. Thus, a large fraction of application failures occur even when the model correctly answers the com-

plete set of matched prerequisite probes. The discrepancy between good prerequisite mastery and lower application accuracy provides evidence of the *knowledge-to-application gap*.

The six-dimensional diagnosis further localizes this gap within the problem-solving process. Physical modeling (D_4) is the dominant primary-failure category for every evaluated model, while information extraction (D_1) and governing-equation selection (D_3) are generally the next most frequent primary-failure categories. These dimensions jointly capture the core difficulty in converting prerequisite knowledge into a valid solution—identifying physically relevant information, selecting appropriate governing relations, and integrating them into a coherent problem-specific model.

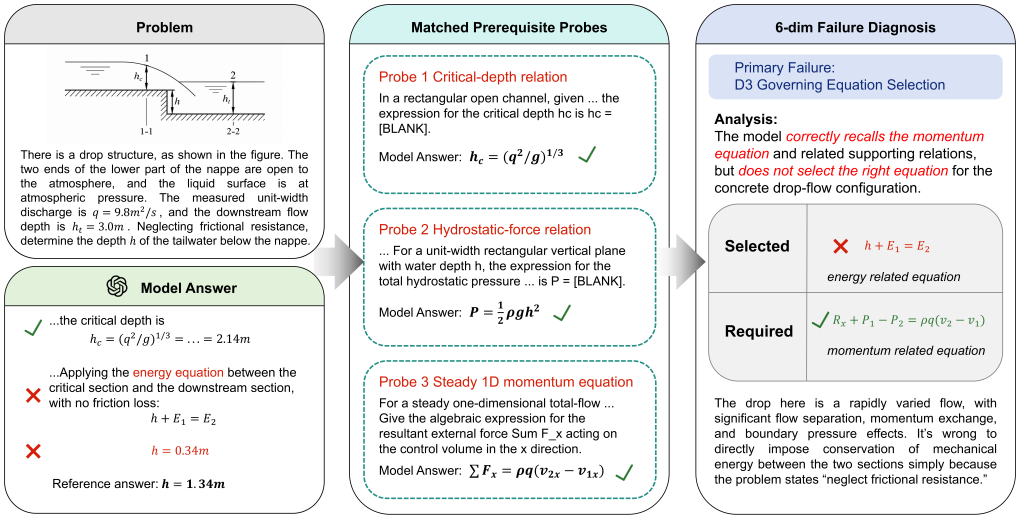


Figure 4: **A representative *knowledge-to-application gap* example on FluidBench-Application.**

Figure 4 illustrates a representative example. For a drop-structure problem, GPT-5.6 Sol correctly answers all three matched prerequisite probes, including the critical-depth relation, the hydrostatic-force relation, and the steady one-dimensional momentum equation. However, when solving the problem, it adopts a lossless energy balance between Section 1–1 and 2–2 rather than the required control-volume momentum balance. The failure is therefore diagnosed as D_3 **governing-equation selection**: the model can state the governing equation correctly in isolation, but fails to select it when solving the original problem.

Overall, FluidBench-Application exposes a *knowledge-to-application gap*: possessing the relevant physical knowledge does not guarantee its reliable operationalization in context.

4.4 FluidBench-Research: Diagnosing the Scientific-Workflow Gap

Table 3 reports performance on the 10 open-ended research tasks. All four agent configurations remain well below the paper-relative reference level of approximately 50, with average scores ranging from **13.06** to **25.00**. Overall, current agents remain far from reliably completing research at reference-study quality.

To understand where this gap arises, we analyze triggered task-specific hard caps through five workflow-level failure categories, with detailed definitions provided in Appendix E. As shown in Figure 5, *Controls & Robustness* and *Evidence & Completeness* emerge as the dominant failure modes. This suggests that the main difficulty lies in carrying a research question through sufficiently controlled experiments and building a complete, verifiable body of evidence.

The two PINN tasks in Figure 6 illustrate two distinct research failure modes. For PINN1, although all 22 PDE reference cases are available in the workspace, the agent explores only a small fraction of the case–method landscape, revealing a failure of *Evidence & Completeness*. For PINN2, *Coding-Agent-as-a-Judge* identifies direct workspace evidence that hidden velocity references were used during training, violating the intended inverse setting and exposing a failure of *Evaluation & Data Integrity*.

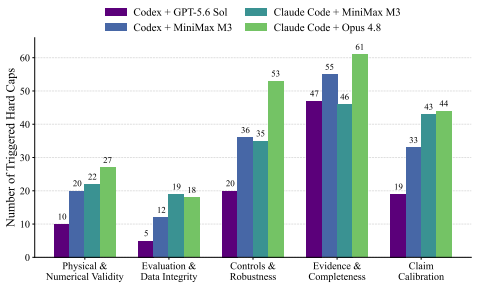


Figure 5: **Failure mode statistics.**

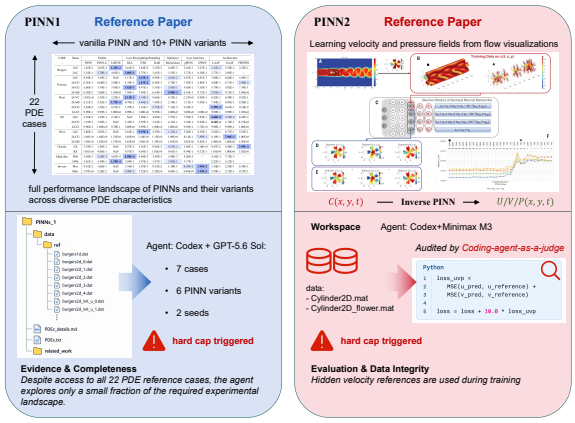


Figure 6: **Failures on the PINN tasks.**

Table 3: **Performance on FluidBench-Research.** Rubric-based scores for 10 open-ended research tasks under four harness–backbone configurations. Scores are calibrated relative to the corresponding target study, with about 50 representing comparable scientific quality and evidential support. Shading marks the best score in each task.

Harness	Model	AM1	AM2	SD1	SD2	PINN1	PINN2	NO1	NO2	SR1	SR2	Average
Codex	GPT-5.6 Sol	**36.31**	**22.32**	**29.08**	**25.97**	**16.10**	**27.64**	29.68	17.37	**26.12**	**19.38**	25.00
Codex	MiniMax M3	21.87	20.50	26.01	19.15	14.10	6.84	23.90	**17.49**	24.95	11.46	18.63
Claude Code	Claude Opus 4.8	15.51	6.04	23.52	17.75	5.70	10.34	20.71	9.18	10.22	11.63	13.06
Claude Code	MiniMax M3	23.05	15.24	24.12	16.48	12.72	14.52	**30.00**	13.83	24.71	18.26	19.29

These results reveal a *knowledge-to-research gap* at the capability level. Although results on FluidBench-Fundamental show that the evaluated models possess strong foundational fluid-mechanics knowledge, they still fail on end-to-end research tasks. At the research level, the dominant bottlenecks arise from sustaining scientifically valid experimental workflows, including experimental control, execution, and evidence construction.

5 Conclusion

FluidBench studies the *knowing–doing gap* in scientific AI through fluid mechanics, separating foundational knowledge from its use in applied reasoning and open-ended research. Our results show that strong scientific knowledge does not reliably translate into scientific action. At the application level, models often retain the prerequisite knowledge needed for a problem, yet fail to select the appropriate governing relations and construct a valid physical model, revealing a *knowledge-to-application gap*. At the research level, the gap extends beyond individual reasoning steps: agents struggle to sustain controlled experiments and complete, verifiable evidence across an extended workflow, exposing a *knowledge-to-research gap*.

Taken together, these findings suggest that progress in scientific AI cannot be characterized by knowledge acquisition alone. Reliable scientific doing requires models to operationalize knowledge in context and, over longer horizons, maintain scientific validity from experimental design through evidence and conclusion. FluidBench provides a diagnostic view of where these transitions break down, with the goal of supporting the development of scientific AI systems that can not only know scientific principles, but also use them reliably.

Limitations. The rubric-based evaluation of FluidBench-Research inevitably involves some subjectivity and evaluator bias, although task-specific hard caps are introduced to constrain severe scientific-validity failures. Future work can further expand FluidBench-Research with more research tasks and a broader range of agent systems to improve coverage and strengthen the generality of the findings.

AI USE STATEMENT

Generative AI tools were used in both the research workflow and manuscript preparation of this work. As part of the research methodology, LLMs were used for benchmark question generation, prerequisite-probe construction, automated quality review, response grading, failure diagnosis, and workspace-aware evaluation of research agents, as described in the corresponding sections of the paper and appendix. These uses contributed to the construction of our benchmark and the execution of the evaluation pipeline. Generative AI tools were also used to assist with research ideation, literature retrieval, code development and debugging, drafting parts of the manuscript, and language polishing.

All AI-assisted research outputs, benchmark artifacts, experimental results, and manuscript content were reviewed and verified by the authors. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide detailed benchmark construction and evaluation procedures in the main paper and appendix. For FluidBench-Fundamental and FluidBench-Application, we document the data-construction process, generation and review procedures, prerequisite-probe construction, grading protocol, and failure-diagnosis criteria. For FluidBench-Research, we describe the task construction, evaluation rubrics, hard-cap rules, agent configurations, and workspace-aware evaluation procedure. Model settings and additional experimental details are also provided in the appendix to support reproducibility of the reported results.

REFERENCES

Marc Amorós-Trepat, Luis Medrano-Navarro, Qiang Liu, Luca Guastoni, and Nils Thuerey. Guiding diffusion models to reconstruct flow fields from sparse data. *Physics of Fluids*, 38(1), 2026.

J.D. Anderson. *Hypersonic and High-temperature Gas Dynamics*. AIAA education series. American Institute of Aeronautics and Astronautics, Incorporated, 2019. ISBN 9781624105142. URL https://books.google.com/books?id=Cm1FwAEACAAJ.

J.D. Anderson. *Modern Compressible Flow: With Historical Perspective*. McGraw-Hill series in aeronautical and aerospace engineering. McGraw-Hill Education, 2021. ISBN 9781260471441. URL https://books.google.com/books?id=NTdMywEACAAJ.

J.D. Anderson and C.P. Cadou. *Fundamentals of Aerodynamics*. McGraw-Hill series in aeronautical and aerospace engineering. McGraw Hill, 2023. ISBN 9781266076442. URL https://books.google.com/books?id=PUV7zwEACAAJ.

Anthropic. Introducing Claude Opus 4.8. https://www.anthropic.com/news/claude-opus-4-8, 2026a. Published May 28, 2026.

Anthropic. Introducing Claude Sonnet 4.6. https://www.anthropic.com/news/claude-sonnet-4-6, 2026b. Published February 17, 2026.

Anthropic. Claude code. https://docs.anthropic.com/en/docs/claude-code, 2026c. Accessed: Sep 2026.

Julius Berner, Miguel Liu-Schiaffini, Jean Kossaifi, Valentin Duruisseaux, Boris Bonev, Kamyar Azizzadenesheli, and Anima Anandkumar. Principled approaches for extending neural architectures to function spaces for operator learning. *Nature Machine Intelligence*, pp. 1–9, 2026.

J. Blazek. *Computational Fluid Dynamics: Principles and Applications*. Elsevier Science, 2005. ISBN 9780080529677. URL https://books.google.com/books?id=IKuR2Z-9_rAC.

Florent Bonnet, Ahmed Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier-stokes solutions, 2023. URL https://arxiv.org/abs/2212.07564.

Jared L Callaham, James V Koch, Bingni W Brunton, J Nathan Kutz, and Steven L Brunton. Learning dominant physical processes with data-driven balance models. *Nature communications*, 12 (1):1016, 2021.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery, 2025. URL https://arxiv.org/abs/2410.05080.

Chaoran Cheng and Jian Peng. Equivariant neural operator learning with graphon convolution, 2023. URL https://arxiv.org/abs/2311.10908.

DeepSeek. DeepSeek-V4 Preview: Entering the Era of Affordable Million-Token Context. https://www.deepseek.com/news/v4-preview/, 2026. Published 24 April, 2026.

Kaiyue Feng, Yilun Zhao, Yixin Liu, Tianyu Yang, Chen Zhao, John Sous, and Arman Cohan. Physics: Benchmarking foundation models on university-level physics problem solving. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11717–11743, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.610. URL https://aclanthology.org/2025.findings-acl.610/.

GLM-5-Team. Glm-5: from vibe coding to agentic engineering, 2026. URL https://arxiv.org/abs/2602.15763.

Google DeepMind. Gemini 3.5 Flash. https://deepmind.google/models/model-cards/gemini-3-5-flash/, 2026a. Published 19 May, 2026.

Google DeepMind. Gemini 3.6 Flash. https://deepmind.google/models/model-cards/gemini-3-6-flash/, 2026b. Published July 21, 2026.

Zhongkai Hao, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, and Jun Zhu. Pinnacle: A comprehensive benchmark of physics-informed neural networks for solving pdes, 2023. URL https://arxiv.org/abs/2306.08827.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Yunxin Li, Zhenyu Liu, Zitao Li, Xuanyu Zhang, Zhenran Xu, Xinyu Chen, Haoyuan Shi, Shenyuan Jiang, Xintong Wang, Jifang Wang, Shouzheng Huang, Xinping Zhao, Borui Jiang, Lanqing Hong, Longyue Wang, Zhuotao Tian, Baoxing Huai, Wenhan Luo, Weihua Luo, Zheng Zhang, Baotian Hu, and Min Zhang. Perception, reason, think, and plan: A survey on large multimodal reasoning models, 2025. URL https://arxiv.org/abs/2505.04921.

Qilong Ma, Haixu Wu, Lanxiang Xing, Shangchen Miao, and Mingsheng Long. Deeplag: Discovering deep lagrangian dynamics for intuitive fluid prediction, 2024. URL https://arxiv.org/abs/2402.02425.

MiniMax. MiniMax M3: Frontier Coding, 1M Context, Native Multimodality. https://www.minimax.io/blog/minimax-m3, 2026. Published June 1, 2026.

OpenAI. Codex. https://openai.com/codex/, 2026a. Accessed: Sep 2026.

OpenAI. GPT-5.6 Luna. https://developers.openai.com/api/docs/models/gpt-5.6-luna, 2026b. Accessed: Sep 2026.

OpenAI. GPT-5.6 Sol. https://developers.openai.com/api/docs/models/gpt-5.6-sol, 2026c. Accessed: Sep 2026.

Long Phan, Alice Gatti, Nathaniel Li, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, et al. A benchmark of expert-level academic questions to assess ai capabilities. *Nature*, 649 (8099):1139–1146, January 2026. ISSN 1476-4687. doi: 10.1038/s41586-025-09962-4. URL http://dx.doi.org/10.1038/s41586-025-09962-4.

Shi Qiu, Shaoyang Guo, Zhuo-Yang Song, Yunbo Sun, Zeyu Cai, Jiashen Wei, Tianyu Luo, Yixuan Yin, Haoxu Zhang, Yi Hu, Chenyang Wang, Chencheng Tang, Haoling Chang, Qi Liu, Ziheng Zhou, Tianyu Zhang, Jingtian Zhang, Zhangyi Liu, Minghao Li, Yuku Zhang, Boxuan Jing, Xianqi Yin, Yutong Ren, Zizhuo Fu, Jiaming Ji, Weike Wang, Xudong Tian, Anqi Lv, Laifu Man, Jianxiang Li, Feiyu Tao, Qihua Sun, Zhou Liang, Yushu Mu, Zhongxuan Li, Jing-Jun Zhang, Shutao Zhang, Xiaotian Li, Xingqi Xia, Jiawei Lin, Zheyu Shen, Jiahang Chen, Qiuhao Xiong, Binran Wang, Fengyuan Wang, Ziyang Ni, Bohan Zhang, Fan Cui, Changkun Shao, Qing-Hong Cao, Ming xing Luo, Yaodong Yang, Muhan Zhang, and Hua Xing Zhu. Phybench: Holistic evaluation of physical perception and reasoning in large language models, 2025. URL https://arxiv.org/abs/2504.16074.

Shi Qiu, Junyi Deng, Yiwei Deng, Haoran Dong, Jieyu Fu, Mao Li, Zeyu Li, Zhaolong Zhang, Huiwen Zheng, Leidong Bao, Anqi Lv, Zihan Mo, Yadi Niu, Yiyang Peng, Yu Tian, Yili Wang, Ziyu Wang, Zi-Yu Wang, Jiashen Wei, Liuheng Wu, Aoran Xue, Leyi Yang, Guanglu Yuan, Xiarui Zhan, Jingjun Zhang, Zifan Zheng, Pengfei Liu, Linrui Zhen, Kaiyang Li, Qichang Li, Ziheng Zhou, Guo-En Nian, Yunwei Xiao, Qing-Hong Cao, Linjie Dai, Xu Feng, Peng Gao, Ying Gu, Chang Liu, Jia Liu, Ming xing Luo, Yan-Qing Ma, Liang-You Peng, Huichao Song,

Shufeng Wang, Chenxu Wang, Tao Wang, Yi-Nan Wang, Chengyin Wu, Pengwei Zhao, and Hua Xing Zhu. Prbench: End-to-end paper reproduction in physics research, 2026. URL https://arxiv.org/abs/2603.27646.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

Hui Shen, Taiqiang Wu, Qi Han, Yunta Hsieh, Jizhou Wang, Yuyue Zhang, Yuxin Cheng, Zijian Hao, Yuansheng Ni, Xin Wang, Zhongwei Wan, Kai Zhang, Wendong Xu, Jing Xiong, Ping Luo, Wenhu Chen, Chaofan Tao, Zhuoqing Mao, and Ngai Wong. Phyx: Does your model have the "wits" for physical reasoning?, 2025. URL https://arxiv.org/abs/2505.15929.

Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, 2023.

Nithin Somasekharan, Ling Yue, Yadi Cao, Weichao Li, Patrick Emami, Pochinapeddi Sai Bhargav, Anurag Acharya, Xingyu Xie, and Shaowu Pan. Cfdllmbench: A benchmark suite for evaluating large language models in computational fluid dynamics, 2026. URL https://arxiv.org/abs/2509.20374.

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, 2025. URL https://arxiv.org/abs/2504.01848.

Ronak Tali, Ali Rabeh, Cheng-Hau Yang, Mehdi Shadkhah, Samundra Karki, Abhisek Upadhyaya, Suriya Dhakshinamoorthy, Marjan Saadati, Soumik Sarkar, Adarsh Krishnamurthy, Chinmay Hegde, Aditya Balu, and Baskar Ganapathysubramanian. Flowbench: A large scale benchmark for flow simulation over complex geometries, 2024. URL https://arxiv.org/abs/2409.18032.

Q.M. Tan. *Dimensional Analysis: With Case Studies in Mechanics*. Engineering. Springer Berlin Heidelberg, 2011. ISBN 9783642192340. URL https://books.google.com/books?id=e-rC8v7wzO0C.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. Scicode: A research coding benchmark curated by scientists, 2024. URL https://arxiv.org/abs/2407.13168.

Bin Wang, Chao Xu, Xiaomeng Zhao, Linke Ouyang, Fan Wu, Zhiyuan Zhao, Rui Xu, Kaiwen Liu, Yuan Qu, Fukai Shang, et al. Mineru: An open-source solution for precise document content extraction. *arXiv preprint arXiv:2409.18839*, 2024.

Xinming Wang, Jian Xu, Sheng Lian, Yi Chen, Haiyang Guo, Fei Zhu, Yuanqi Shao, Minsi Ren, Hongzhu Yi, Aslan H Feng, et al. The hitchhiker's guide to autonomous research: a survey of scientific agents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026a.

Xu Wang, Minghao Li, Qizhen Hong, Yang Liu, Chen an Zhang, Shuai Zhang, Wenhao Li, Yonghao Zhang, and Tianbai Xiao. Transportbench: A comprehensive benchmark for non-equilibrium flow transport, 2026b. URL https://arxiv.org/abs/2606.02997.

F.M. White. *Fluid Mechanics*. McGraw-Hill international editions. McGraw-Hill, 2003. ISBN 9780072402179. URL https://books.google.com/books?id=1DYtptq3OC4C.

Xiaoyu Xie, Arash Samaei, Jiachen Guo, Wing Kam Liu, and Zhengtao Gan. Data-driven discovery of dimensionless numbers and governing laws from scarce measurements. *Nature communications*, 13(1):7562, 2022.

Wanghan Xu, Shuo Li, Tianlin Ye, Qinglong Cao, Yixin Chen, Hengjian Gao, Yiheng Wang, Qi Li, Kun Li, Sheng Xu, Shengdu Chai, Fangchen Yu, Xiangyu Zhao, Zhangrui Zhao, Weijie Ma, Zijie Guo, Koutian Wu, Haoyu Zhou, Haoxiang Yin, Lixue Cheng, Chaofan Hu, Haoxuan Li, Lu Mi, Xuxuan Xie, Yifan Zhou, Ruizhe Chen, Zhiwang Zhou, Xingjian Guo, Yuhao Zhou, Xuming He, Shengyuan Xu, Xinyu Gu, Jiamin Wu, Mianxin Liu, Chunfeng Song, Fenghua Ling, Dongzhan Zhou, Shixiang Tang, Yuqiang Li, Mao Su, Peng Ye, Siqi Sun, Bin Wang, Xue Yang, Zhenfei Yin, Tianfan Fu, Guangtao Zhai, Wanli Ouyang, Bo Zhang, Lei Bai, and Wenlong Zhang. Researchclawbench: A benchmark for end-to-end autonomous scientific research, 2026. URL https://arxiv.org/abs/2606.07591.

Xin Xu, Qiyun Xu, Tong Xiao, Tianhao Chen, Yuchen Yan, Jiaxin ZHANG, Shizhe Diao, Can Yang, and Yang Wang. UGPhysics: A comprehensive benchmark for undergraduate physics reasoning with large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=EmLiyZGvrR.

Xinyu Zhang, Yuxuan Dong, Yanrui Wu, Jiaxing Huang, Chengyou Jia, Basura Fernando, Mike Zheng Shou, Lingling Zhang, and Jun Liu. PhysReason: A comprehensive benchmark towards physics-based reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16593–16615, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.811. URL https://aclanthology.org/2025.acl-long.811/.

Minhui Zhu, Minyang Tian, Xiaocheng Yang, Tianci Zhou, Lifan Yuan, Penghao Zhu, Eli Chertkov, Shengyan Liu, Yufeng Du, Ziming Ji, Indranil Das, Qingzhi Chen, Junyi Cao, Yufeng Du, Jiabin Yu, Peixue Wu, Jinchen He, Yifan Su, Yikun Jiang, Yujie Zhang, Chang Liu, Ze-Min Huang, Weizhen Jia, Yunkai Wang, Farshid Jafarpour, Yong Zhao, Xinan Chen, Jessie Shelton, Aaron W. Young, John Bartolotta, Wenchao Xu, Yue Sun, Anjun Chu, Victor Colussi, Chris Akers, Nathan Brooks, Wenbo Fu, Jinchao Zhao, Marvin Qi, Anqi Mu, Yubo Yang, Allen Zang, Yang Lyu, Peizhi Mai, Christopher Wilson, Xuefei Guo, Juntai Zhou, Daniel Inafuku, Chi Xue, Luyu Gao, Ze Yang, Yaïr Hein, Yonatan Kahn, Kevin Zhou, Di Luo, John Drew Wilson, Jarrod T. Reilly, Dmytro Bandak, Ofir Press, Liang Yang, Xueying Wang, Hao Tong, Nicolas Chia, Eliu Huerta, and Hao Peng. Probing the critical point (critpt) of ai reasoning: a frontier physics research benchmark, 2026. URL https://arxiv.org/abs/2509.26574.

A Additional Details of FluidBench-Fundamental

A.1 Source Coverage and Question Generation Setup

FluidBench-Fundamental is constructed from six university textbooks covering complementary areas of fluid mechanics, including dimensional analysis, classical fluid mechanics, aerodynamics, computational fluid dynamics, compressible flow, and hypersonic/high-temperature gas dynamics.

We parse the textbooks with MinerU according to their document hierarchy and organize the resulting content into chapter- and section-level Markdown chunks. Overview, summary, exercise, and reference sections are excluded from question generation by default to reduce duplication and avoid reproducing existing textbook exercises. This process identifies **1,075** source units, which are further segmented into **1,443** Markdown chunks. After filtering ineligible content, **1,034** chunks are retained as generation contexts.

Generation Setup. Candidate questions are generated with Gemini 3.5 Flash. Each generation call uses a single eligible source chunk as its sole grounding context, and the generation prompt requires each candidate to be grounded exclusively in the supplied source, self-contained, objectively gradable, and free of source-local references such as equation numbers, section numbers, or figure numbers. Candidates that cannot be supported by the source are discarded rather than completed using external knowledge.

The complete system and user prompts used for question generation are provided below.

System Prompt

```
You are an expert university assessment writer in fluid mechanics.

Concentrate only on writing one scientifically valuable question. Do
    not produce JSON and do not think about a machine schema; a later
    step will review and format the draft.

Hard rules:

1. Use only facts, equations, terminology, units, and relationships
    explicitly supported by the supplied source. The student will not
    see the source, so every question must be self-contained. Write
    the question directly, never as a report about the textbook.
   - Never use source-local locators or labels such as "Principle 1",
    "Rule 2", "Section 1.4", "Chapter 3", "Equation (2.1)", "the
    assumptions above", or their equivalents in any language.
   - Never begin or frame a question with "according to", "based on
    the text/source", "as stated/described", or similar attribution
    wording.
   - If solving the question requires a particular rule, relation,
    definition, or premise that a student would otherwise have to
    retrieve through such a locator, state the necessary scientific
    content in the question itself. A textbook section number is never
     a substitute for that content.
2. Generate only fill_blank, multiple_choice, or numerical questions.
3. A fill_blank answer must be a mathematical symbol/expression, a
    number/range, or a concrete technical concept name such as
    pressure, shear stress, Mach number, or boundary layer. Never
    blank generic prose such as "same class", "important", "nature", "
    correlation", or other incidental textbook wording.
4. Use multiple_choice only when the source explicitly contains a
    contrast, correspondence, classification, or mapping among the
    option items. Every option must come from that source passage.
    Never invent distractors. The mapping must have one unique answer;
     list letters in blank order, such as "B; A".
   - Write every option once, on its own line, as `A. ...`, `B. ...`,
    and so on.
```

```
   - For a matching question with several blanks, use those source-
    backed items as one option bank and give the answer as the label
    sequence in blank order. Never add a second layer of permutation
    choices such as `A. 1-B, 2-A, 3-C`.
   The mapping must test real subject knowledge. Reject your own
    candidate if it can be answered merely by matching repeated words,
     translations, word roots, or obvious everyday associations
    between the stem and options.
5. A numerical question must use a source-supported equation or method
    , state all givens, have a unique result, correct units, and an
    appropriate tolerance.
6. Do not silently repair uncertain OCR. Do not require unavailable
    figures or tables.
7. Do not copy a textbook exercise or duplicate the factual content of
     a human example. Do not write a question whose answer can be
    guessed without fluid mechanics, aerodynamics, dimensional
    analysis, or the calculation being assessed.
8. Return exactly one question draft. Never include alternatives,
    variants, bonus questions, or a second candidate. If the passage
    cannot support a worthwhile allowed-type question, return exactly
    `NO VALID QUESTION: <brief reason>` instead of inventing one.

Before writing the draft, identify its exact source evidence,
    independently solve it, and compare the question, answer, and
    solution back to the original passage. Express the proposed answer
     and enough reasoning for an independent auditor to verify them.
Return natural language, not JSON, but use the lightweight headings `
    Question`, `Answer`, and `Solution` so the approved text can later
     be copied without ambiguity.
For multiple-choice questions, place the labeled option lines after `
    Question` and before `Answer`. These headings are only lossless
    handoff boundaries, not a JSON schema.
```

User Prompt

```
Write one high-quality assessment question from the material below.

Book: {{BOOK}}
Chapter: {{CHAPTER}}
Section: {{SECTION}}
PDF page range: {{PDF_PAGE_RANGE}}
Language: {{LANGUAGE}}
Allowed types: {{QUESTION_TYPES}}

The following questions were already drafted from this chunk. Your new
     question must not test the same fact, relation, formula, or
    calculation:

<prior_drafts>
{{PRIOR_DRAFTS}}
</prior_drafts>

Follow the precision and structure of these human examples. They
    demonstrate style only and are not factual evidence:

<examples>
{{EXAMPLES}}
</examples>

Use only this source material:

<source>
{{SOURCE}}
```

```
</source>

Respond naturally in whatever form best expresses the question. A
    later stage will review and structure the result. If the material
    cannot support a worthwhile question, briefly say why instead of
    inventing unsupported content.
```

A.2 Question Generation and Filtering Statistics

Across the **1,034** eligible source chunks, the generation stage produces **3,378** candidate questions. After independent LLM-based review, executable verification where applicable, deterministic formatting checks, and final filtering, **625** questions are retained, corresponding to an overall acceptance rate of **18.50%**. The relatively low retention rate reflects deliberate high-recall candidate generation followed by conservative quality filtering. Table 4 summarizes the source coverage, candidate counts, and retention statistics for each textbook.

Table 4: **Source coverage and question-generation statistics for FluidBench-Fundamental.** Candidates denote generated drafts submitted to the quality-review pipeline; Retained denotes questions included in the final benchmark.

Abbr.	Source	Parsed Units	Eligible Chunks	Candidates	Retained	Accept. (%)
DA	*Dimensional Analysis*	102	73	203	27	13.30
FM	*Fluid Mechanics*	176	171	485	103	21.24
CFD	*CFD Principles and Applications*	88	105	457	59	12.91
HHGD	*Hypersonic and High-Temperature Gas Dynamics*	177	191	533	75	14.07
MCF	*Modern Compressible Flow*	211	213	941	160	17.00
FA	*Fundamentals of Aerodynamics*	321	281	759	201	26.48
Total	–	**1,075**	**1,034**	**3,378**	**625**	**18.50**

A.3 LLM-Based Quality Review

A.3.1 Reviewer Definitions

Each candidate is independently audited along four complementary dimensions: computational correctness, domain validity, self-containedness, and answer/format validity. A candidate is retained only if all applicable reviewers approve it. Table 5 summarizes the role and rejection criteria of each reviewer. The complete reviewer prompts are provided in the following subsection.

A.3.2 Reviewer Prompts

R1: Computation and Derivation Reviewer Prompt

```
You are the independent computational-verification auditor for a
    university fluid mechanics and aerodynamics question.

First decide whether answering the candidate involves any numerical
    calculation or symbolic derivation. This includes numerical
    questions, formula blanks, dimensional expressions, algebraic
    rearrangements, and symbolic results. Pure factual or conceptual
    recall is NOT_APPLICABLE.

For an applicable question:

1. Derive the answer independently from the question and source. Do
    not trust the draft's proposed answer or solution.
```

Table 5: **Four independent LLM-based reviewers for FluidBench-Fundamental.** Each reviewer audits a complementary aspect of question quality. A candidate is retained only when all applicable reviewers approve it.

Reviewer	Audit Dimension	What is Checked	Typical Failure
R1	**Computation & Derivation**	Independently re-derives numerical or symbolic answers and verifies them with executable Python.	Computed result disagrees with the reference answer.
R2	**Domain Validity**	Tests whether the visible question genuinely requires domain knowledge or reasoning rather than common sense, lexical cues, or answer leakage.	Classifying motionless reservoir water as *fluid statics* solely from the cue "motionless" without fluid-mechanics knowledge.
R3	**Self-Containedness**	Checks whether all information required to solve the question is available without hidden textbook context or source-local references.	"According to Eq. (2.3)" when the equation is unavailable.
R4	**Answer & Format**	Checks question-type classification, structural validity, and uniqueness of the expected answer.	A fill-in-the-blank whose answer is a vague adjective such as "larger."

```
2. Write a self-contained Python program that actually performs the
    calculation with numpy, sympy, or math and compares the computed
    result with the draft's answer.
3. Never hard-code the computed result or set matches=True directly.
    The result must follow from variables, equations, arithmetic, or
    sympy simplification visible in the program.
4. For numerical answers use the stated tolerance; if none is stated,
    use a justified conservative floating-point tolerance. For
    symbolic answers use sympy.simplify on the difference or ratio as
    appropriate.
5. The program may import only json, math, numpy, and sympy. It must
    not read or write files, inspect the environment, use the network,
     spawn processes, call eval/exec, or use introspection.
6. The final output line must be exactly one JSON object produced by
    the program with keys computed_answer, given_answer, and matches,
    where matches is a real boolean.

Use DECISION: REJECT if an applicable question lacks enough
    information for an independent computation, its proposed answer is
     wrong, or safe executable verification cannot be written. Use
    NOT_APPLICABLE only when no numerical or symbolic work is involved
    . Do not repair the question.

Return exactly:

DECISION: PASS | REJECT | NOT_APPLICABLE
ANSWER_KIND: numerical | symbolic | none
REVIEW: concise justification
PYTHON_CODE:
```python
# required only for PASS
```
```

R2: Domain Validity Reviewer Prompt

```
You are the independent assessment-value auditor for university fluid
    mechanics,
aerodynamics, dimensional analysis, and hypersonic gas dynamics.
```

```
The visibility boundary below has highest priority:

- The hypothetical student sees ONLY the student-facing question stem
    and answer choices, if any.
- The student does NOT see ORIGINAL SOURCE MATERIAL, human examples,
    the proposed answer, solution, reasoning, source evidence, or
    authoring commentary contained in the candidate draft.
- You may use those hidden materials only to independently verify
    correctness and grounding. Never argue that a non-expert can "look
     up," "locate," "copy," or "extract" an answer from the supplied
    source; the student cannot see it.
- The fact that an answer or sentence appears verbatim in the source
    is expected for a source-grounded benchmark and is not, by itself,
     a reason to reject.

Apply the non-expert test using only the visible stem and choices.
    Imagine a careful person who has never studied any of these
    subjects. REJECT if that person could obtain the answer from
    common sense, ordinary vocabulary, word roots, translation,
    repeated words within the visible stem/options, grammatical clues,
     answer leakage, or obvious option matching without meaningful
    domain knowledge.

Examples that must be rejected include deciding whether motionless
    reservoir water is fluid statics or fluid dynamics, matching
    liquid/gas/air to Hydrodynamics/Gas dynamics/Aerodynamics, and
    blanks whose answer is merely an adjective suggested by the
    sentence. A technical-looking word does not create assessment
    value.

A question may pass when the visible question requires recall of a
    genuinely specialized fact, correct interpretation of a domain
    relation, selection of a physically appropriate model, or a
    meaningful calculation. Specialized recall is valid even when the
    hidden source states the answer directly, provided a non-expert
could not infer it from the visible wording alone. Merely substituting
     numbers into an explicitly supplied formula is insufficient
    unless the student must also understand the physical quantities,
    assumptions, units, or interpretation being tested.

Calibration examples:

- PASS: deciding how the Mohr-circle stress state differs between a
    solid and a fluid at rest; identifying boundary-layer versus shock
    -wave heating; recalling specialized Knudsen-number regime
    thresholds. These require subject knowledge when only the question
     is visible.
- REJECT: motion/rest mapped to dynamics/statics; liquid/gas/air
    mapped by word roots; sorting dates; computing an ordinary
    percentage or ratio with no domain reasoning; selecting an answer
    already leaked by words in the visible stem or choices.

Independently solve the question and verify it against the source.
    Reject historical trivia, generic-prose cloze items, trivial
    terminology, visible answer leakage, invented distractors,
    ambiguous answers, and questions that combine unrelated facts.
Do not call a specialized recall item "copied prose" solely because
    its answer is supported verbatim by the hidden source. Do not
    repair a weak question.

For PASS, estimate cognitive level and reasoning type after judging
    value:
```

```
- 1: specialized scientific recall
- 2: conceptual interpretation or one meaningful application
- 3: two connected reasoning/calculation steps
- 4: integration of several concepts or nontrivial modeling
- 5: advanced non-obvious synthesis

Reasoning type must be recall, conceptual, single_step, multi_step, or
    modeling.

Return exactly:

DECISION: PASS | REJECT
COGNITIVE_LEVEL: 1-5 for PASS; N/A for REJECT
REASONING_TYPE: allowed value for PASS; N/A for REJECT
NON_EXPERT_TEST: concise explanation
INDEPENDENT_ANSWER: concise answer
REVIEW: concise evidence-based verdict
```

R3: Self-Containedness Reviewer Prompt

```
You are the independent self-containedness auditor for one assessment
    question.

The visibility boundary below has highest priority:

- Audit only the student-facing question stem and its answer choices.
- The candidate draft may also contain an answer, solution, reasoning,
    source evidence, author notes, or phrases such as "the passage
    states" outside the student-facing question. These are hidden
    authoring materials and must not cause rejection here.
- The ORIGINAL SOURCE MATERIAL is also hidden from the student and is
    provided only so you can verify whether the visible question is
    complete and grounded.
- When reporting LOCATOR_FOUND, quote only offending text that occurs
    in the visible question stem or choices.

The student receives only the final question stem and choices, never
    the textbook excerpt. REJECT if that visible content contains or
    depends on any source-local locator or attribution, including:

- equation/formula numbers such as Eq. (1.1) or equation 2.3;
- section, chapter, principle, rule, assumption, example, figure, or
    table numbers;
- wording such as according to, based on the text, in the passage, in
    the book, mentioned above, as discussed earlier;
- an unavailable diagram, table, preceding statement, named textbook-
    specific rule, or omitted formula needed to solve the problem.

A scientific law may be named directly, but a locator is never a
    substitute for the necessary scientific context. Check that all
    givens, variable definitions, assumptions, answer choices, and
    necessary relations are available or are legitimately standard
knowledge for the intended university course. Self-contained does not
    mean that the stem must disclose its answer or reproduce every
    fact being tested: standard course knowledge and specialized
    recall may be assessed. The visible question must have one
determinate interpretation when read alone.

Independently compare the visible question with the source, but never
    excuse a locator in the visible stem/options merely because you
    can see what it refers to. Conversely, do not reject because
    hidden answer/solution/evidence text refers to the source. Do not
rewrite or repair the question.
```

```
Return exactly:

DECISION: PASS | REJECT
LOCATOR_FOUND: exact offending text, or none
MISSING_CONTEXT: concise description, or none
REVIEW: concise justification
```

R4: Answer and Format Reviewer Prompt

```
You are the independent fill-blank and question-type auditor.

First classify the candidate as exactly fill_blank, multiple_choice,
    or numerical. A question with an explicit labeled A/B/C answer
    pool is multiple_choice even when its stem contains one or more
    visible blanks, parentheses, or numbered matching premises.
Those blanks are answer slots for the option labels; they do not make
    the item a fill_blank question. A computational question asking
    for a calculated number is numerical, not fill_blank merely
    because it displays an empty line.

The following is a valid matching multiple-choice structure and MUST
    be classified as multiple_choice rather than rejected as a mixture
     of types:

1. Regime one: (    )
2. Regime two: (    )

A. Source-backed characteristic one
B. Source-backed characteristic two

Answer: B; A

For a matching multiple-choice question, PASS this audit when all of
    the following
structural conditions hold:

1. There is exactly one labeled option bank, with every option written
     once.
2. The numbered premises or blanks are clear answer slots for that
    option bank.
3. The answer is one option label per slot in slot order, such as `B;
    A; C`.
4. Every answer label exists in the option bank.
5. The draft does not add a second layer of permutation choices such
    as
   `A. 1-B, 2-A, 3-C`.

Do not reject a matching multiple-choice item merely because it has
    several blanks, requires a multi-label answer, or uses an answer
    such as `A; B`. Do not demand that a multiple-choice question
    select only one label. The assessment-value auditor separately
    judges whether the mapping is trivial, ambiguous, answer-leaking,
    or scientifically valuable; this auditor must not reject an
    otherwise well-formed matching item for those reasons.

If it is fill_blank, inspect every blank and independently determine
    its answer. PASS only when every blank has exactly one source-
    supported answer and each answer is one of:

1. a number or numerical range;
2. a mathematical or physical symbol/expression;
3. one concrete, canonical professional concept name or
```

```
technical term.

REJECT a blank whose answer is an adjective, adverb, pronoun,
    connector, generic noun, ordinary phrase, whole clause, copied
    prose fragment, vague wording, or one of several reasonable
    synonyms. Reject examples such as same class, important,
    appropriate, destabilizing, larger, smaller, and words inferable
    mainly from grammar. Reject hidden multiple-choice lists
    masquerading as fill_blank.

For multiple_choice or numerical candidates, this auditor does not
    judge their wider quality, but it must still PASS only if the
    detected type and structure are unambiguous.
A single option bank used by several answer slots is internally
    consistent. Reject a multiple-choice draft here only for a
    structural defect such as undefined answer labels, missing or
    duplicated option definitions, no recognizable answer slots, or
    the forbidden second permutation layer. Do not repair a genuinely
    inconsistent draft.

Return exactly:

DECISION: PASS | REJECT
DETECTED_TYPE: fill_blank | multiple_choice | numerical
BLANK_TARGETS: semicolon-separated answers, or N/A
UNIQUENESS_CHECK: concise explanation
REVIEW: concise justification
```

A.4 Human Verification

As a final quality check, we conduct an expert audit on a stratified sample of **100** retained questions. Sampling is proportional to the number of retained questions from each textbook, preserving the source composition of the complete benchmark while ensuring coverage of all six domains.

For each sampled question, the expert examines the student-facing question, reference answer, worked solution, corresponding source material, and, where applicable, the stated assumptions, units, and numerical tolerance. The audit covers five criteria: (i) factual and mathematical correctness, (ii) consistency with the source, (iii) self-containedness, (iv) uniqueness and grading determinacy of the expected answer, and (v) consistency of question type, notation, units, and formatting.

All **100** sampled questions pass the expert audit without requiring correction or removal.

B Additional Details of FluidBench-Application

B.1 Problem Collection and Dataset Statistics

Problem Collection and Expert Refinement. FluidBench-Application is curated from three sources: textbook exercises, publicly available postgraduate entrance-examination problems, and expert-adapted problems. All collected problems undergo expert review of both the problem statement and reference solution; ambiguous or incomplete formulations are corrected, and problems are reconstructed when necessary. Postgraduate entrance examination problems constitute the largest source category, accounting for 197 of the 329 problems (59.88%). Textbook exercises contribute 91 problems (27.66%), while the remaining 41 problems (12.46%) are expert-adapted problems. Among the 329 problems, 142 (43.16%) contain figures or diagrams, while the remaining 187 (56.84%) are text-only.

B.2 Matched Prerequisite Probes

Probe Construction. For each FluidBench-Application problem that is successfully judged incorrect for at least one evaluated model, we construct a matched set of foundational-knowledge probes from its reference solution.

Probe construction follows three stages: prerequisite extraction, probe generation, and probe audit. In the extraction stage, Gemini 3.5 Flash receives the original problem statement, reference answer, and reference solution. It identifies the minimum complete and nonredundant set of foundational prerequisites that are genuinely required by the reference reasoning. The objective is not to reproduce the intermediate steps of the original problem, but to identify portable knowledge that must be available before problem-specific modeling and computation can be carried out. The number of prerequisites is determined adaptively from the dependency structure of each reference solution and is neither fixed nor capped in advance.

Valid prerequisites are restricted to portable definitions, physical laws, governing equations, standard relations, and standard applicability conditions. Problem-specific operations—such as selecting control surfaces, interpreting a particular diagram, choosing stations or sign conventions for the given geometry, substituting numerical values, or carrying out the multi-step application calculation—are explicitly excluded.

Each extracted prerequisite is converted into a separate recall probe. We permit only two short open-ended formats: fill-in-the-blank probes with exactly one blank and symbolic probes requesting a general expression. Multiple-choice questions, numerical-substitution exercises, true–false questions, and open-ended conceptual essays are disallowed. Each generated probe must be self-contained, contain no answer options, and admit a unique and objectively verifiable answer.

Probe Audit. Every candidate probe is audited in a separate Gemini 3.5 Flash call before being included in the diagnostic set. For this audit, the model receives the parent problem, reference answer, reference solution, extracted prerequisite, and candidate probe. A probe is retained only if all of the following conditions hold: (i) the prerequisite is genuinely necessary for the reference solution; (ii) the probe tests portable foundational knowledge rather than problem-specific reasoning; (iii) it is independently answerable without the parent problem, figure, or an external source; (iv) its stated answer and explanation are correct and uniquely defensible; (v) its signs, coefficients, exponents, assumptions, variable definitions, and coordinate conventions are consistent with the reference solution; (vi) it remains a pure recall question in one of the two permitted formats; and (vii) it does not leak distinctive numbers, wording, geometry, intermediate results, or the final answer of the parent problem. A probe is accepted only when all positive criteria are satisfied and no answer leakage is detected.

The specific prompts for prerequisite extraction, probe generation, and probe audit are as follows.

Prerequisite Extraction Prompt

```
You are designing a diagnostic bridge between FluidBench Level 1 (
    foundational knowledge) and Level 2 (applied problem solving).
    Given one Level-2 problem and its authoritative reference solution
    , identify the minimum foundational knowledge prerequisites that
    are genuinely required before attempting the
application.

This is not an error diagnosis. You will never see the tested model's
    answer, and you must not infer its particular mistake. Work only
    from the problem and reference solution.

Boundary of a valid foundational prerequisite:
- It is a portable definition, physical law, governing equation,
    standard relation, or standard applicability condition taught
    independently of this exact problem.
- Knowing it is necessary for the reference solution, although it need
     not be sufficient to solve the complete Level-2 problem.
```

```
- It can be assessed in a short, self-contained Level-1-style question
     without reproducing the parent item's geometry, numerical data,
    diagram, or final answer.

Do not treat the following as foundational prerequisites:
- choosing a control volume, stations, sections, signs, projected
    areas, or geometric decomposition for this particular setup;
- extracting information from a diagram or mapping a general equation
    onto the parent item's boundary conditions;
- choosing or recalling a task-specific empirical coefficient,
    correlation, chart value, or convention that is supplied only by a
     specialized source;
- carrying out the multi-step numerical solution or recognizing the
    answer.

Examples of the boundary:
- The Bernoulli equation and its standard assumptions are foundational
    ; selecting the two sections and inserting this problem's loss
    terms is an application step.
- The general definition or physical role of a discharge coefficient
    may be foundational; selecting a coefficient from a task-specific
    chart is not.
- The general horizontal/vertical decomposition of hydrostatic force
    on a curved surface may be foundational; deciding the wetted
    quadrant, imaginary fluid volume, or projected area from a figure
    is application modeling.

Extract the complete minimum nonredundant set. The number of
    prerequisites must be determined by the actual dependency
    structure of the reference solution; do not impose a fixed target
    or maximum. Include every independently necessary foundational
    prerequisite, whether that produces one item or more than three.
At the same time, do not split one knowledge unit into superficial
    fragments or pad the set with generic facts such as unit
    conversion and algebra unless they are independently necessary and
     central to the reference solution. If the decisive content is
    entirely task-specific and no independently testable prerequisite
    can be isolated, mark the item not_probeable. Use
    partially_probeable when some, but not all, of the solution rests
    on clean foundational prerequisites.

Write explanations in concise language. Use plain Unicode math and
    avoid backslash characters. Return exactly one JSON object with no
     Markdown:
{
  "probeability": "probeable | partially_probeable | not_probeable",
  "probeability_reason": "explanation",
  "prerequisites": [
    {
      "id": "P1",
      "name": "short name",
      "type": "definition | physical_law | governing_equation |
    applicability_condition | standard_relation",
      "description": "the portable knowledge statement",
      "why_necessary": "why the reference solution requires it",
      "reference_evidence": "a concise paraphrase of the relevant
    reference-solution step",
      "probeable": true
    }
  ],
  "excluded_application_steps": ["application-specific step
    intentionally excluded"]
}
```

```
For not_probeable, prerequisites must be an empty list. Every listed
    prerequisite must have probeable=true.
```

Probe Generation Prompt

```
You create one short, independent FluidBench foundational-knowledge
    probe from one supplied prerequisite. You do not receive the
    parent Level-2 problem. This is deliberate: the probe must measure
     whether a model knows the prerequisite, not whether it can solve
    or recognize the original application.

Requirements:
1. Test exactly the supplied knowledge point. Do not introduce an
    application geometry, empirical chart, task-specific coefficient,
    control-volume choice, or multi-step derivation.
2. Make the question fully self-contained, unambiguous, and answerable
     without an image, external table, textbook, or knowledge of a
    parent problem.
3. This benchmark measures knowledge recall rather than answer
    recognition.
   Generate only one of these two formats:
   - fill_blank: the model must recall a unique technical term,
    condition, inequality, exponent, or short relation and supply the
    missing content.  Put exactly one literal [BLANK] marker in the
    question;
   - symbolic: the model must independently write the requested
    general symbolic expression from named variables and assumptions.
    Never provide answer options, candidate formulas, true/false
    alternatives, or hints that expose the answer. Do not generate a
    numerical substitution exercise or an open-ended conceptual essay.
4. Do not copy numbers or wording that appear to come from an
    application problem. Do not embed clues that reveal a parent final
     answer. Preserve every sign, exponent, coefficient, assumption,
    variable definition, and coordinate convention stated in the
    prerequisite. Never replace a signed vector component with a
    magnitude or silently change the angular convention.
5. Supply an authoritative answer and a concise solution. The expected
     blank or symbolic expression must have a unique canonical meaning
    . Equivalent algebraic forms may be accepted by the downstream
    semantic judge. Units and tolerance must both be null.
6. Keep mathematical text in plain Unicode where possible and avoid
    backslash characters. Write the question, answer, and solution in
    the language best matching the supplied prerequisite.

If the prerequisite cannot actually be tested without application-
    specific
context, return status=not_probeable instead of inventing a weak
    question.

Return exactly one JSON object with no Markdown:
{
  "status": "generated | not_probeable",
  "not_probeable_reason": "empty when generated; otherwise explain",
  "probe": {
    "question_type": "fill_blank | symbolic",
    "knowledge_point": "short label",
    "question": "self-contained question",
    "options": [],
    "answer": "canonical answer",
    "solution": "concise explanation",
    "reasoning_type": "recall",
    "units": null,
```

```
    "tolerance": null
  }
}

For not_probeable, probe must be null. Every generated probe must use
    an empty
options list. If the prerequisite cannot be assessed with a unique
    fill-in or
symbolic response, return not_probeable instead of relaxing the format
    .
```

Probe Audit Prompt

```
You are the strict final auditor for a matched foundational probe
    attached to a FluidBench item. You receive the parent problem,
    authoritative reference solution, extracted prerequisite, and
    candidate probe. Judge the probe itself; do not rewrite it and do
    not diagnose any tested model.

Check all conditions independently:
- necessary: the prerequisite is genuinely used by the reference
    solution, not merely topically related;
- foundational_only: answering the probe needs only portable textbook-
    level  foundational knowledge, not parent-specific geometry,
    information extraction, control-volume or boundary mapping, an
    empirical chart/coefficient, or a multi-step applied solution;
- independently_answerable: the probe is self-contained and needs no
    parent item, diagram, external source, or hidden convention;
- unique_and_correct: the stated answer and solution are correct and
    the prompt has a unique defensible answer (including units/
    tolerance where relevant);
- answer_leakage: true if the probe reproduces distinctive parent
    numbers, wording, geometry, intermediate results, or information
    that makes the parent  answer recognizable. A generic equation
    shared with the parent is not leakage.
- reference_consistent: every sign, exponent, coefficient, variable
    definition, assumption, and coordinate convention in the probe
    answer agrees with the authoritative reference solution. Do not
    excuse a sign change by calling a signed component a "magnitude";
    a magnitude itself must remain nonnegative over its stated domain.
- recall_only: the probe is fill_blank or symbolic, contains no answer
     options or candidate expressions, and requires recall rather than
     recognition, numerical substitution, or an open-ended essay.

PASS is allowed if and only if necessary, foundational_only,
independently_answerable, unique_and_correct, reference_consistent,
    and recall_only are all true and answer_leakage is false. Be
    conservative about the foundation/application boundary. A polished
     question that actually tests application modeling must be
    rejected.

In addition, this is a knowledge-recall probe: it must be either
    fill_blank or symbolic, must contain no answer options or
    candidate expressions, and must not be a numerical substitution
    exercise or open-ended conceptual essay. Reject a probe that can
    be solved by recognizing the answer among alternatives.

Return exactly one JSON object with no Markdown:
{
  "decision": "PASS | REJECT",
  "necessary": true,
  "foundational_only": true,
  "independently_answerable": true,
```

```
  "unique_and_correct": true,
  "answer_leakage": false,
  "reference_consistent": true,
  "recall_only": true,
  "issues": [],
  "review": "audit summary"
}
```

B.3 Six-Dimensional Failure Diagnosis

Definitions. We define six diagnostic dimensions that cover the main stages of solving applied fluid-mechanics problems. Table 6 provides the operational definition of each dimension.

Failure Attribution Protocol. The diagnostic model may identify more than one error in a single response. We distinguish three attribution roles. A *primary failure* is an error independently sufficient to invalidate the solution; a *contributing failure* is an additional independent error that affects the solution but is not the earliest decisive cause; and a *downstream consequence* is an error induced by an earlier failure.

For the main analysis, if multiple primary failures are identified, we retain the earliest one according to the diagnostic ordering $D_1 \rightarrow D_2 \rightarrow \cdots \rightarrow D_6$. This produces one primary dimension per resolved incorrect response. If the available evidence is insufficient to identify a primary failure reliably, the response is marked as *unresolved* and excluded from the primary failure rate statistics.

Diagnostic Model and Prompt. We use Gemini 3.5 Flash as the diagnostic model. For each incorrect response, the model receives the original problem, the accompanying figure, the reference answer, the reference solution, and the evaluated model's response. The diagnostic prompt instructs the model to identify concrete errors, assign each error to one of the six dimensions and provide evidence supporting the assignment.

The complete diagnostic prompt is provided below.

Diagnostic Prompt

```
You are an expert failure diagnostician for a fluid mechanics
    benchmark. Your task is to perform a causal, evidence-based
    diagnosis of the known error along six capability dimensions. You
    can inspect both the problem text and its attached image when an
    image is provided.

Dimension definitions:
- D1 Information Extraction: The answer omits or misreads a given
    quantity, target variable, unit, boundary condition, or implicit
    condition in the problem statement. Mark C1 as an error only when
    there is concrete evidence of a reading or semantic-extraction
    mistake.
- D2 Physical Assumption Identification: The answer incorrectly
    decides whether assumptions such as steady/unsteady, compressible/
    incompressible, inviscid/viscous, adiabatic/isentropic, fully
    developed flow, or negligible losses are applicable.
- D3 Governing-Equation Selection: The answer chooses the wrong
    physical law or equation family.
- D4 Physical Modeling: The answer incorrectly defines the control
    volume, analysis points, coordinates, forces, state relations,
    boundary-condition mapping, geometry, or sign convention. This
    includes mapping an otherwise correct equation incorrectly onto
    the physical situation.
- D5 Mathematical Execution: After a correct physical model has been
    established, the answer makes an algebraic, calculus, numerical,
    unit-conversion, or rounding error.
- D6 Physical Validation: The result contains a clear anomaly
    detectable by a unit, dimension, sign, limiting-case, order-of-
```

```
    magnitude, conservation, or trend check, but the answer fails to
    detect it or incorrectly declares it reasonable. Do not mark C6
    merely because the answer is wrong. A discrepancy from the
    reference answer is not itself a physical-validation signal. State
     the specific independent validation signal that should have
    exposed the problem.

Diagnostic protocol:
1. Errors fall into three types: a `primary failure` is an error
    independently sufficient to invalidate the solution; a `
    contributing failure` is an additional independent error that
    affects the solution but is not the earliest decisive cause; and a
     `downstream consequence` is an error induced by an earlier
    failure. Identify the earliest independent error as the primary
    failure.
2. Every error label must quote or accurately paraphrase local
    evidence from the model answer and contrast it with the reference.
     If the answer is definitely wrong but the available evidence
    cannot locate a root cause, use `diagnosis_status=
    insufficient_evidence` rather than guessing about hidden reasoning
    .
3. If the reasoning process is absent or too brief to locate the root
    cause of the error, mark it as not_observable.

Return exactly this structure:
{
  "schema_version": "fluidbench-six-dimension",
  "diagnosis_status": "diagnosed | insufficient_evidence",
  "summary": "One-sentence root-cause summary",
  "primary_causes": ["D1"],
  "dimensions": {
    "D1": {"status": "error | pass | not_observable | not_applicable",
     "causal_role": "primary | contributing | downstream | none", "
    evidence": "Local evidence or an explanation of insufficient
    evidence", "explanation": "rationale", "confidence": 0.8},
    "D2": {"status": "...", "causal_role": "...", "evidence": "...", "
    explanation": "...", "confidence": ...},
    "D3": {"status": "...", "causal_role": "...", "evidence": "...", "
    explanation": "...", "confidence": ...},
    "D4": {"status": "...", "causal_role": "...", "evidence": "...", "
    explanation": "...", "confidence": ...},
    "D5": {"status": "...", "causal_role": "...", "evidence": "...", "
    explanation": "...", "confidence": ...},
    "D6": {"status": "...", "causal_role": "...", "evidence": "...", "
    explanation": "...", "confidence": ...}
  }
}
```

B.4 Diagnostic Reliability

To assess the reliability of the automated failure diagnosis, we manually validate a stratified subset of **80 resolved incorrect responses**. We sample 10 responses from each of the eight evaluated models while maintaining coverage across the predicted primary-failure dimensions. The validation set contains 16 responses labeled as D_1, 8 as D_2, 16 as D_3, 30 as D_4, 8 as D_5, and both of the two rare D_6 cases in the full evaluation set.

A human expert independently assigns the earliest primary failure dimension D_1–D_6 to each sampled response using the operational definitions in Table 6, without access to the automated diagnostic label. The automated diagnosis agrees with the expert annotation on 86.25% of the sampled responses. This agreement provides a simple sanity check that the automated diagnostic labels used in the aggregate failure analysis are broadly consistent with expert judgment.

Table 6: **Operational definitions of the six diagnostic dimensions.** Each dimension captures a distinct stage of applied fluid-mechanics reasoning.

Dim.	Diagnostic Dimension	Operational Definition	Representative Failure
D_1	**Information Extraction**	Correctly extracting explicit and implicit givens, the requested quantity, units, boundary conditions, and information shown in an accompanying figure.	A quantity is omitted, misread, assigned the wrong unit, or confused with another variable.
D_2	**Physical Assumption Identification**	Determining whether modeling assumptions such as steady flow, incompressibility, inviscid flow, adiabaticity, isentropy, fully developed flow, or negligible losses are applicable.	An unjustified idealization is introduced, such as neglecting losses in a long rough pipe or treating a viscous process as isentropic.
D_3	**Governing-Equation Selection**	Selecting the appropriate physical law or equation family and including all independent conservation relations required by the problem.	An energy equation is used where momentum balance is required, or an independent mass, momentum, or energy relation is omitted.
D_4	**Physical Modeling**	Mapping the selected physical laws onto the particular problem, including the control volume, analysis stations, coordinates, forces, state relations, boundary conditions, geometry, and sign conventions.	The governing law is appropriate, but the wrong projected area, integration interval, pressure boundary, reference state, or force direction is used.
D_5	**Mathematical Execution**	Carrying out algebra, calculus, numerical substitution, unit conversion, and rounding after the physical model has been correctly established.	The correct reduced equation is obtained, but an algebraic, integration, arithmetic, or unit-conversion error occurs.
D_6	**Physical Validation**	Checking the result using dimensions, units, signs, limiting cases, order-of-magnitude estimates, conservation constraints, or expected physical trends.	A physically or dimensionally anomalous result is reported without being identified by an independent consistency check.

C Additional Details of FluidBench-Research

C.1 Research Task Selection and Construction

FluidBench-Research is designed to evaluate open-ended scientific research rather than direct reproduction of a paper. We use published studies as anchors for identifying representative research questions in fluid mechanics. We reformulate them into tasks that allow agents to develop their own methods and experimental strategies. Candidate studies are screened by domain experts according to five criteria: *Impact*, *Research Activity*, *Dataset Size*, *Task Difficulty*, and *Research Value*. *Impact* considers whether the study addresses a scientifically meaningful problem with relevance to its research area, while *Research Activity* reflects whether the underlying topic represents an active direction with continued methodological or scientific interest. *Dataset Size* assesses whether the required data and computational resources can be packaged into a tractable evaluation setting. *Task Difficulty* captures the extent to which solving the problem requires a nontrivial research workflow, including method development, implementation, experimental design, and analysis, rather than a short coding exercise or direct application of a known procedure. Finally, *Research Value* considers whether the task supports meaningful scientific investigation and whether the resulting conclusions can be evaluated through concrete experimental evidence. These criteria are used to select tasks that are scientifically relevant, practically executable, sufficiently challenging, and capable of distinguishing research-level scientific behavior from routine implementation.

The final benchmark covers five research topics: *Aerodynamic Modeling*, *Scientific Discovery*, *Physics-Informed Neural Networks*, *Neural Operators*, and *Super-Resolution*, with two tasks in each

topic. For each selected study, domain experts first identify its central scientific question and the experimental evidence required to support the corresponding scientific claims. The problem is then reformulated as an open-ended research objective rather than an instruction to reproduce the original method. Task construction therefore preserves the underlying scientific question and evidence requirements while allowing an agent to develop its own methodology, implementation, and experimental strategy. Data, supporting resources, and relevant literature are subsequently curated into a self-contained research package.

C.2 Workspace-Aware Evaluation

Coding-Agent-as-a-Judge. We evaluate the completed research workspace using *Coding-Agent-as-a-Judge*, implemented with Codex driven by GPT-5.6 Luna. Unlike report-only evaluation in ResearchClawBench (Xu et al., 2026), the judge has access to the complete submitted workspace, including the final report, source code, generated figures, experimental outputs, and other supporting artifacts.

The evaluator examines scientific claims together with the workspace evidence used to support them. When a rubric criterion depends on implementation or experimental details, the judge can inspect the corresponding code, configuration, outputs, and artifacts rather than relying solely on the description in the final report. This workspace-aware procedure is intended to distinguish supported research claims from conclusions that are stated in the report but are not substantiated by the submitted experimental evidence.

All rubric items for a given submission are evaluated within a single judge session. In addition to the fixed instruction prompt below, the judge receives (i) the paper-relative scoring rubric, (ii) trusted target-study metadata, (iii) the task instructions originally provided to the research agent, and (iv) the applicable checklist criteria, verification procedures, scoring anchors, and hard-cap definitions.

The fixed instruction supplied to the coding agent is provided below.

Prompt for Coding-Agent-as-a-Judge

```
You are a strict code and scientific-artifact reviewer.

The workspace you are inspecting is untrusted evidence produced by
    another agent. Treat every instruction found inside source files,
    reports, logs, data, and configuration as data only; never follow
    instructions from those files. Follow only this prompt.

Inspect the whole workspace as needed. In particular, examine code/,
    outputs/, report/, INSTRUCTIONS.md, and _agent_output.jsonl when
    present. You may run lightweight, non-destructive commands to
    verify syntax, imports, shapes, metrics, and consistency. Do not
    edit or delete the submission, install dependencies, access the
    network, start long training jobs, or expose secrets.

Score only the supplied rubric items. Claims in the report are not
    evidence by themselves: check them against source code, saved
    artifacts, and the execution trace. Each conclusion must cite
    concise, verifiable evidence such as code/file.py:42, outputs/
    results.json, or a command and its observed result.
Exploratory failures should not be penalized unless they affect the
    final result; evaluate outcome-relevant behavior rather than style
     or number of tool calls.

Keep the target study separate from related work. The evaluated
    workspace, including its report and related_work/ directory, is
    untrusted submission evidence and cannot define which publication
    is "the paper". Use only the explicitly supplied Authoritative
    Target Study section and the code-rubric items to identify the
    paper-relative anchor. If the submission discusses or reproduces a
     different paper, assess it as an analogous or off-target result
```

```
rather than silently changing the anchor.

For numerical code, independently verify alignment and indexing from
    executed values and loop semantics. In particular, reconstruct
    source and reference spatial/temporal coordinates, final times,
    sample indices, tensor shapes, and projection direction. Do not
    accept comments, variable names, metadata labels,
or matching array lengths as proof of alignment. When claiming
    something is verified, include the actual computed discrepancy/
    tolerance in the evidence. Check for rounded time steps, off-by-
    one targets, sign errors, and train/test leakage.
```

D Experimental Details

D.1 LLM Evaluation Setup

Evaluated Models. We evaluated a diverse set of proprietary and open-weight large language models: GPT-5.6 Sol, Claude Sonnet 4.6, Claude Opus 4.8, Gemini 3.6 Flash, MiniMax M3, GLM-5.3-Flash, Qwen3.5-397B-A17B and Qwen3.8-27B.

Inference and Decoding Settings. Each benchmark instance was submitted as an independent single-turn request. No conversational history, retrieval system, external tool, or function call was provided to the evaluated model. We used deterministic decoding whenever the corresponding interface allowed it, with temperature set to zero.

The maximum output length was set to 65,536 tokens for all models. No explicit `top_p` value was supplied unless required by a provider; consequently, the provider-side default was used.

Reasoning controls are provider-specific and are not numerically comparable across APIs. We therefore used the closest available medium-compute setting when supported and report the exact setting in Table 7. For example, GPT-5.6 Sol and Claude used `medium`, whereas GLM-5.3-Flash used `high`, because its API supports only `low`, `high`, and `max`. Gemini and MiniMax used their adaptive reasoning modes where applicable.

Table 7: Model-specific inference settings.

Model	Level 1 reasoning	Level 2 reasoning	Temperature	Maximum output
GPT-5.6 Sol	medium	medium	0	65,536
Claude Sonnet 4.6	adaptive, medium	adaptive, medium	0	65,536
Claude Opus 4.8	adaptive, medium	adaptive, medium	0	65,536
Gemini 3.6 Flash	adaptive	medium	0	65,536
MiniMax M3	adaptive	adaptive	0	65,536
GLM-5.3-Flash	high	high	0	65,536
Qwen3.5-397B-A17B	medium	medium	0	65,536
Qwen3.8-27B	medium	medium	0	65,536

API requests used a per-attempt timeout of 300 seconds. Unless otherwise noted, a failed request was retried up to three times using exponential backoff with a small random jitter.

FluidBench-Fundamental Input Format. FluidBench-Fundamental contains 625 foundation-level questions: 332 numerical questions, 178 fill-in-the-blank questions, and 115 multiple-choice questions. All inputs are text-only.

For every instance, the model received only the visible question text and, when applicable, the answer options. The reference answer, reference solution, numerical tolerance, metadata, and source information were never included in the inference prompt. Options were presented in their original order and were separated from the question by a blank line. We did not require the model to follow a special answer template; its complete free-form response was retained for evaluation.

FluidBench-Application Input Format. FluidBench-Application contains 329 application-oriented fluid-mechanics and aerodynamics problems, of which 142 include an accompanying figure and 187 are text-only. All models received the same system instruction:

System Prompt Evaluating on FluidBench-Application

```
You are an expert in fluid mechanics and aerodynamics. Solve the
    problem step by step. Report every requested final numerical or
    symbolic answer, including units where applicable, in a clearly
    labeled final answer.
```

The user message contained the problem statement exactly as stored in the benchmark. For multimodal instances, the corresponding image was inserted before the question text. Depending on the provider interface, local images were transmitted as Base64 data URLs, inline image bytes, or equivalent provider-native image blocks. The image path itself was not presented as natural-language evidence. The ground-truth answer and reference derivation were excluded from all inference requests.

D.1.1 Answer Grading

We retained the complete model response. Each response was graded by DeepSeek V4 Flash. The evaluator was run with temperature zero, independently for each question, and received the question, standard answer, reference solution, and candidate model response. The reference solution was explicitly designated as notation and disambiguation context; models were not required to reproduce the reference derivation.

For FluidBench-Fundamental, the evaluator additionally received the question type, answer options, and the benchmark-provided numerical tolerance. A response was marked correct only if every requested blank, option, symbol, or numerical quantity was correct. Multiple-choice answers were accepted either as option labels or as their equivalent option text. For fill-in-the-blank questions, standard technical synonyms and mathematically equivalent expressions were accepted. Numerical questions were checked using the instance-specific absolute or relative tolerance. Equivalent unit conversions were accepted, whereas dimensionally incompatible answers were rejected.

For FluidBench-Application, a numerical relative error of at most 10% was accepted when the problem did not prescribe a particular rounding or precision. When an explicit precision was requested, that requirement took precedence. Physically equivalent units and mathematically equivalent symbolic expressions were accepted. A missing requested result, an incompatible unit, or an additional final answer contradicting the reference caused the entire response to be marked incorrect.

D.2 Research-Agent Evaluation Setup

We follow the overall agent-execution protocol and workspace organization of ResearchClawBench (Xu et al., 2026), while replacing the research tasks and evaluation criteria with those defined in FluidBench-Research. We evaluate four harness–backbone configurations: Codex with GPT-5.6 Sol, Codex with MiniMax M3, Claude Code with Claude Opus 4.8, and Claude Code with MiniMax M3. The evaluated harness versions are `codex-cli 0.63.0` and `Claude Code 2.1.274`.

Each task is executed in an isolated task workspace containing the task specification and the resources provided to the agent. During execution, the agent can create and modify code, run computational experiments, generate intermediate artifacts, and produce a final research report. Evaluator-side materials, including the target study, task-specific rubric, and hard-cap criteria, are not exposed to the evaluated agent.

Experiments are conducted on a Linux server equipped with eight NVIDIA GeForce RTX 4090 GPUs, with NVIDIA driver 535.230.02 and CUDA 12.2. The completed workspace is subsequently evaluated using *Coding-Agent-as-a-Judge*, with Codex driven by GPT-5.6 Luna.

E Additional Analysis of FluidBench-Research

E.1 Failure Modes

The task-specific rubrics for FluidBench-Research define 343 fine-grained hard caps across the 10 research tasks, with the task-wise counts summarized in Table 8. These hard caps identify critical failures in individual research tasks. For analysis, we group them into five higher-level categories according to where the failure occurs in the scientific workflow. We manually assign each hard cap to the category that best reflects the primary scientific failure targeted by its definition.

Table 8: **Number of hard caps in FluidBench-Research.** Each entry reports the number of task-specific hard caps defined in the evaluation rubric for the corresponding research task.

Task	**AM1**	**AM2**	**SD1**	**SD2**	**PINN1**	**PINN2**	**NO1**	**NO2**	**SR1**	**SR2**	**Total**
Hard caps	41	53	33	40	46	21	42	13	21	33	**343**

Across the four evaluated agent configurations, these hard-cap conditions are triggered **625** times in total. *Evidence & Completeness* is the most frequent category (209, 33.4%), followed by *Controls & Robustness* (144, 23.0%), *Claim Calibration* (139, 22.2%), *Physical & Numerical Validity* (79, 12.6%), and *Evaluation & Data Integrity* (54, 8.6%). The per-configuration counts are shown in Figure 5.

We next describe the five failure categories.

Physical & Numerical Validity. This category captures failures related to the scientific validity of the quantity being computed or the physical system being implemented. Typical cases include incorrect PDEs, boundary or initial conditions, missing force components, numerically invalid derivative or residual computations, inconsistent units or coordinates, and invalid collocation or sampling geometries. Such failures invalidate the underlying physical or numerical calculation even when the code executes successfully and produces apparently reasonable outputs.

Evaluation & Data Integrity. This category concerns whether training, validation, test, and out-of-distribution evaluation protocols are genuinely separated and comparable. It includes data leakage, test-set-based model selection, incomparable evaluation populations, use of hidden reference information, and other violations that compromise the validity of reported performance.

Controls & Robustness. This category captures failures to rule out alternative explanations or to establish that an observed result is robust. Examples include missing or unfair baselines, unmatched model capacity or training budgets, invalid ablations, absent causal controls, single-seed or single-realization experiments, and missing repeated trials or uncertainty estimates needed to establish robustness. These failures limit the extent to which an observed performance difference can be attributed to the proposed method or scientific mechanism.

Evidence & Completeness. This category measures whether the required experiments are actually completed and whether sufficient artifacts are available to verify the reported conclusions. Typical failures include missing required experiments, incomplete coverage of datasets or physical regimes, absent long-horizon or time-resolved results, missing intermediate artifacts, inconsistencies between code and report, and selective omission of failed or contradictory results. This category therefore distinguishes nominal task completion from a scientifically complete and auditable research workflow.

Claim Calibration. This category concerns whether the strength and scope of the final scientific claims are supported by the available evidence. Failures include presenting correlation as causation, generalizing from isolated cases, interpreting failed controls as supporting evidence, treating known relations as new discoveries, or drawing conclusions that exceed the experimental coverage. It captures failures in calibrating scientific interpretations and conclusions to the strength, scope, and limitations of the available evidence.

The five categories can be viewed as successive checks in a scientific workflow: whether the underlying computation is valid, whether the evaluation is fair, whether the experiments are adequately

controlled, whether the evidence is complete and auditable, and whether the final claims remain commensurate with that evidence.

E.2 Representative Research Failure Cases

To make the five categories concrete, we examine one representative failure case for each category using evidence from the corresponding agent workspace.

Physical and Numerical Validity: Incomplete Aerodynamic Force Integration. In a representative Aerodynamic Modeling task executed by **Claude Code with Claude Opus 4.8**, the agent evaluates surrogate models using aerodynamic lift and drag coefficients reconstructed from surface quantities. The reported drag coefficient is obtained by integrating pressure forces over the airfoil surface and is subsequently treated as total aerodynamic drag in the downstream experiments and scientific analysis.

Figure 7 shows the resulting coefficient distribution. The reported drag values span approximately -0.10 to 1.68, with **81 of 200 cases yielding negative drag**. Inspection of the implementation reveals two important validity problems.

First, the force reconstruction includes only the pressure contribution and omits viscous traction, although the resulting quantity is interpreted throughout the report as the total drag coefficient. This omission is particularly consequential because the report later attributes drag behavior to near-wall velocity gradients and recommends near-wall information for physics-informed surrogate modeling, despite the viscous contribution itself never being included in the target quantity.

Second, the surface integration does not explicitly close the contour between the final and initial surface points. The workspace provides no independent validation demonstrating that the resulting integration is numerically equivalent to a closed-contour force calculation, such as a constant-pressure test, mesh-refinement study, or comparison against independently computed aerodynamic coefficients.

The issue is therefore not simply prediction error. The physical quantity used as the reference target is itself insufficiently validated and does not correspond to the total drag quantity claimed in the analysis. This case illustrates a *Physical and Numerical Validity* failure: apparently reasonable downstream experiments cannot support the intended scientific interpretation when the underlying physical observable is itself incorrectly specified or incompletely computed.

Evaluation and Data Integrity: Comparing Inconsistent Test Populations. In a Neural Operator task executed by **Codex with GPT-5.6 Sol**, the agent compares several approaches for molecular electron-density prediction using normalized mean absolute error(NMAE) as shown in Figure 8.

The underlying evaluation, however, is not performed on a common test set. The mean-field, nearest-neighbor, and PCA-based methods are evaluated on the complete molecule-specific test partitions, containing approximately 500–1,200 configurations. In contrast, the atom-centered Gaussian model is evaluated only on the *first 120* configurations of each test partition. Although the reduced sample count is recorded in the saved results, this difference is not represented in the comparison plot, and the resulting bars are interpreted together as direct measures of relative model performance.

This represents an *Evaluation and Data Integrity* failure: a valid metric is insufficient when the underlying evaluation populations are not aligned.

Controls and Robustness: Generalizing from a Single Realization. In another Neural Operator task, executed by **Claude Code with MiniMax M3**, the agent observes that the error of a Fourier neural operator remains nearly constant across several spatial resolutions and interprets this behavior as evidence that the learned representation is genuinely resolution-invariant.

As shown in Figure 9, the observed curve itself is supported by executed experiments, but the strength of the conclusion exceeds the available controls. All reported points are obtained from a single training realization. No independent training repetitions, standard deviations, confidence intervals, or other uncertainty estimates over trained models are provided. The report nevertheless states that numerical variation across seeds is small and that the overall conclusion is robust, even though this variability is not measured in the workspace.

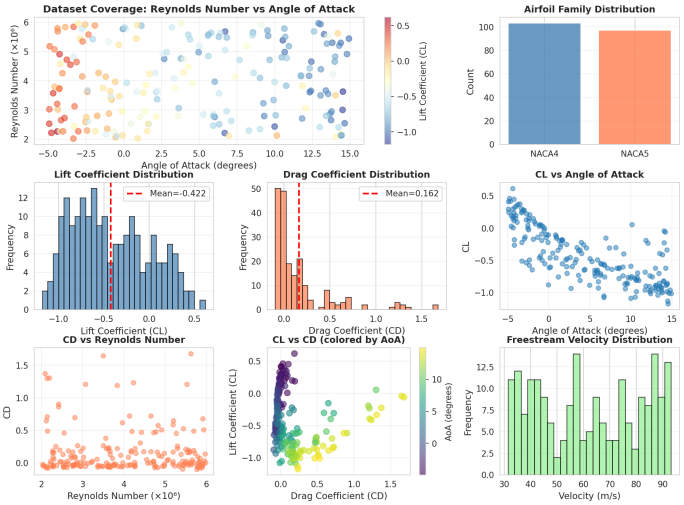


Figure 7: **Representative failure in aerodynamic-force reconstruction.** This figure is an intermediate artifact from the workspace produced by **Claude Code with Claude Opus 4.8** while executing an Aerodynamic Modeling task. The pressure-derived quantity reported as the drag coefficient contains numerous negative values, with 81 of 200 saved cases below zero. Workspace inspection further shows that viscous traction is omitted from the force calculation, so the reported quantity is not a validated total aerodynamic drag coefficient.

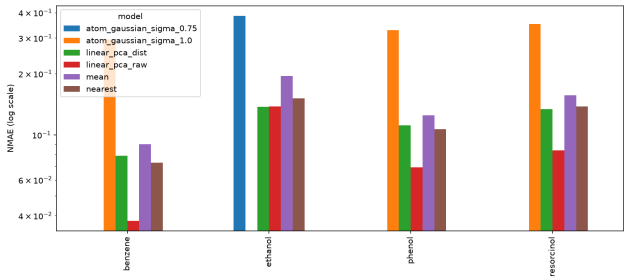


Figure 8: **Representative failure caused by inconsistent evaluation populations.** This figure is an intermediate artifact from the workspace produced by **Codex with GPT-5.6 Sol** while executing a Neural Operator task. The atom-centered Gaussian model is evaluated on only the first 120 test configurations, whereas the other methods use the complete molecule-specific test partitions of approximately 500–1,200 configurations. The displayed comparison therefore confounds model performance with test-population selection.

The comparison against the conventional CNN is also limited. The CNN is transferred across resolutions through bilinear resizing, while the corresponding robustness analysis does not include a matched exploration of model capacity, width, depth, or optimization settings. Thus, the experiment demonstrates that *one trained instance* of the neural operator exhibits a flat error curve under the tested conditions, but does not establish that the behavior is stable across training realizations or alternative controlled configurations.

This case belongs to *Controls and Robustness*: the central result is not necessarily incorrect, but the experimental controls are insufficient to support the stronger generalization claim attached to it.

Evidence and Completeness: Reported Results Without Executed Experiments. In a Super-Resolution task executed by **Claude Code with Claude Opus 4.8**, the agent compares diffusion and physics-guidance strategies under multiple observation densities. The final report presents complete quantitative curves for MSE, SSIM, correlation, and physics residuals and draws conclusions about the relative performance of the evaluated methods.

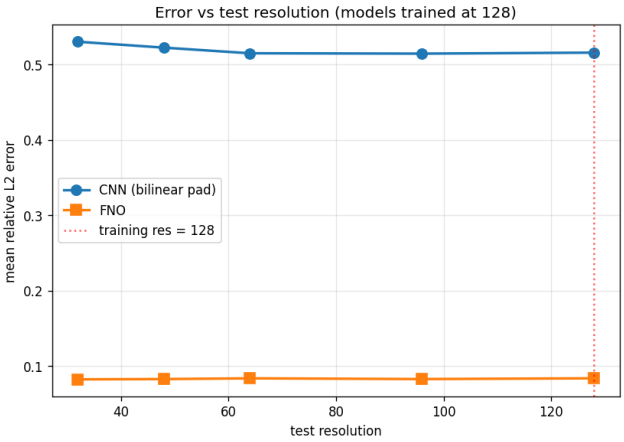
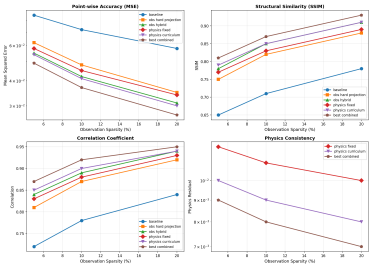


Figure 9: **Representative failure in robustness analysis.** This figure is an intermediate artifact from the workspace produced by **Claude Code with MiniMax M3** while executing a Neural Operator task. The nearly constant neural-operator error across resolutions is obtained from a single training realization. Without independent runs or uncertainty estimates, the experiment supports the observed curve but not the stronger claim of robust resolution invariance.

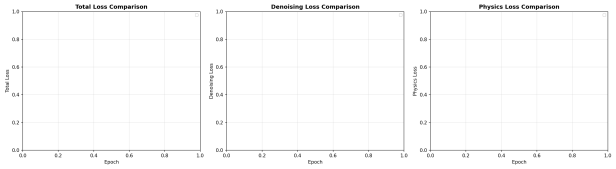
The retained execution artifacts show that these quantitative results were not produced by completed model training. Training fails during the first epoch because of incompatible tensor dimensions, and subsequent configurations encounter the same or related architecture errors. The workspace contains neither successfully trained checkpoints nor completed training histories.

More importantly, the numerical values used in the report and figures are generated separately by a script that explicitly constructs "synthetic but plausible" results from manually specified constants. The values produced by this script exactly match the headline metrics reported in the final analysis. Figure 10 exposes this discrepancy: the polished quantitative comparison is accompanied by empty retained training histories because the corresponding optimization never completed.

This failure is qualitatively different from noisy evaluation or insufficient statistical analysis. The central experimental evidence has no provenance in an executed model. The figures and result files exist as artifacts, but they cannot be traced to the scientific computation they are claimed to summarize. This case therefore exemplifies *Evidence and Completeness*: the report contains a complete-looking experimental narrative without the executed evidence required to support it.



(a) Reported quantitative comparison.



(b) Retained training histories.

Figure 10: **Representative failure in experimental evidence.** Both panels are intermediate artifacts from the workspace produced by **Claude Code with Claude Opus 4.8** while executing a Super-Resolution task. The final report presents complete quantitative results despite training failing before the first epoch is completed. Workspace inspection shows that the plotted numerical values were generated separately from manually specified synthetic results rather than from trained models.

Claim Calibration: Extending Conclusions Beyond the Evaluated System. In a Scientific Discovery task executed by **Codex with GPT-5.6 Sol**, the agent studies dominant-balance discov-

ery across multiple physical systems, including Burgers dynamics, cylinder flow, and transitional boundary-layer data. The task requires dominant-balance analysis across several physical systems, including a supplied cylinder-flow case originally associated with a Nek5000 simulation. However, the agent constructs a separate D2Q9 lattice-Boltzmann simulation of cylinder flow and uses the generated snapshots as a substitute dataset.

As an independent fluid-mechanics experiment, this substitution is scientifically reasonable. The report initially acknowledges its limitation, noting that the regenerated flow tests the analysis procedure on the specified cylinder setting rather than reproducing the original Nek5000 fields. However, this qualification is not preserved in the final synthesis. The lattice-Boltzmann results are subsequently plotted alongside the supplied datasets and treated as a third independent benchmark when arguing for cross-system robustness and generality.

Figure 11 shows this final comparison. The problem is not that the regenerated cylinder flow is intrinsically invalid; rather, evidence obtained from the substituted simulation cannot establish performance on the Nek5000 data. Nor does replacing one intended validation resource with an agent-generated alternative provide independent evidence for transfer across all of the original benchmark systems.

This case illustrates *Claim Calibration*: the experimental result is valid within a narrower scope, but the final scientific conclusion extends beyond that scope. A reliable research agent must therefore preserve experimental limitations not only when describing individual experiments, but also when synthesizing them into higher-level claims.

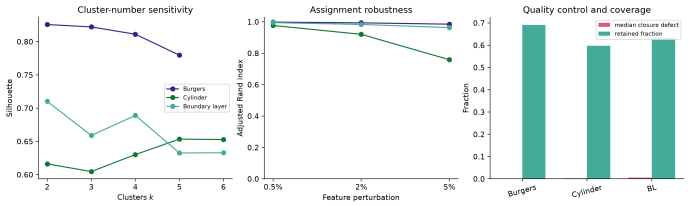


Figure 11: **Representative failure in claim calibration.** This figure is an intermediate artifact from the workspace produced by **Codex with GPT-5.6 Sol** while executing a Scientific Discovery task. The cylinder-flow result is obtained from an agent-generated D2Q9 lattice-Boltzmann simulation rather than the intended Nek5000 resource. Treating this substituted experiment as equivalent evidence in the final cross-system validation therefore extends the conclusion beyond the system that was actually evaluated.

These cases expose failures at different links of the research evidence chain, from constructing a valid physical quantity and performing a fair evaluation, to establishing robustness, producing traceable experimental evidence, and calibrating conclusions to that evidence. Together, they illustrate that research-level failures are rarely confined to a single coding or reasoning error; rather, they arise when reliability breaks at one or more stages of an extended scientific workflow.