

IMPROOFBENCH: BENCHMARKING AI ON RESEARCH-LEVEL MATHEMATICAL PROOF GENERATION

Johannes Schmitt¹, Gergely Bérczi², Jasper Dekoninck¹, Jeremy Feusi¹, Tim Gehringer¹

¹ETH Zurich ²Aarhus University

Benchmark Contributors: Raphael Appenzeller, Jim Bryan, Niklas Canova, Timo de Wolff, Filippo Gaia, Michel van Garrel, Baran Hashemi, David Holmes, Aitor Iribar Lopez, Victor Jaeck, Martina Jørgensen, Steven Kelk, Stefan Kuhlmann, Adam Kurpisz, Chiara Meroni, Ingmar Metzler, Martin Möller, Samuel Muñoz-Echániz, Robert Nowak, Georg Oberdieck, Daniel Platt, Dylan Possamaï, Gabriel Ribeiro, Raúl Sánchez Galán, Zheming Sun, Josef Teichmann, Richard P Thomas, Charles Vial

🌐 <https://improofbench.math.ethz.ch>

ABSTRACT

As the mathematical capabilities of large language models (LLMs) improve, it becomes increasingly important to evaluate their performance on research-level tasks at the frontier of mathematical knowledge. However, existing benchmarks are limited, as they focus solely on final-answer questions or high-school competition problems. To address this gap, we introduce IMProofBench, a private benchmark consisting of 39 peer-reviewed problems developed by expert mathematicians. Each problem requires a detailed proof and is paired with subproblems that have final answers, supporting both an evaluation of mathematical reasoning capabilities by human experts and a large-scale quantitative analysis through automated grading. Furthermore, unlike prior benchmarks, the evaluation setup simulates a realistic research environment: models operate in an agentic framework with tools like web search for literature review and mathematical software such as SageMath. Our results show that current LLMs can succeed at the more accessible research-level questions, but still encounter significant difficulties on more challenging problems. Quantitatively, GROK-4 achieves the highest accuracy of 52% on final-answer subproblems, while GPT-5 obtains the best performance for proof generation, achieving a fully correct solution for 22% of problems. IMProofBench will continue to evolve as a dynamic benchmark in collaboration with the mathematical community, ensuring its relevance for evaluating the next generation of LLMs.

1 INTRODUCTION

Large language models (LLMs) are making rapid progress on mathematical tasks, achieving strong results on challenging benchmarks like AIME (Balunovic et al., 2025) and FrontierMath (Glazer et al., 2024). These improvements suggest that LLMs may soon support mathematical research by collaborating with professional mathematicians on open problems. However, to determine whether current systems are capable of contributing in such settings, benchmarks are needed that test capabilities at the frontier of mathematical research.

Limitations of existing benchmarks Existing benchmarks fall short of this objective: most focus on high-school or undergraduate-level mathematics (Balunovic et al., 2025; Frieder et al., 2023), due to the difficulty associated with designing rigorous, research-level problems. The few benchmarks that do target more advanced mathematics, like FrontierMath (Glazer et al., 2024) and HLE (Phan et al., 2025), focus exclusively on final-answer problems. As a result, they overlook proof-writing capabilities and allow models to apply shortcuts to reach the correct final answer without fully solving the problem (EpochAI, 2025).









Question: Isomorphism Classes of Stable Graphs	
<p>Given an integer $g \geq 2$, let N_g be the number of isomorphism classes of stable graphs of genus g with precisely 3 edges. Give a closed formula for N_g valid for all $g \geq 2$.</p> <p>Follow-up subquestions: What are N_3, N_8, and N_{10000}?</p>	
GPT-5 Reasoning Summary	GROK-4 Reasoning Summary
N_g is a sum of connected multigraph types limited by 3 edges, considering partitions of genera.	Finds a non-closed formula for N_g , requiring to compute a sum from 1 to g .
 Obtain experimental data for some values.	 Attempts to look up the sequence in OEIS.
Submits a closed formula for N_g as a degree-3 quasi-polynomial with a period of 6.	Submits a very concise sketch of the answer, with the non-closed formula for N_g .
Full Grade: 3/3 Subquestions:   	Full Grade: 2/3 Subquestions:   

Figure 1: Example IMProofBench problem. Models are tested on research-level questions in an agentic framework with tool access. Grading of the main reasoning is done by a human expert, while follow-up subquestions are evaluated using an automated parser. For the question above, models other than GPT-5 and GROK-4 only made minor progress. For full details, see App. C.

This work: IMProofBench To fill this gap, we introduce IMProofBench, a private benchmark developed in collaboration with the mathematical research community to evaluate LLMs on research-level proof writing. IMProofBench is built on a custom platform and supported by initiatives that actively involve professional mathematicians. It includes tasks ranging from challenging oral exam questions in a graduate course to open research questions based on the contributors’ own work. Unlike static benchmarks, IMProofBench is designed as a platform for continuous evaluation: problems are added on a rolling basis, ensuring its continued relevance for evaluating the next generation of frontier LLMs. Currently, IMProofBench consists of 39 problems developed in collaboration with over 23 mathematicians, with 30 more questions in the latest stages of the problem creation pipeline.

Problem creation pipeline Each problem in IMProofBench is authored by a research mathematician within their area of expertise. Submissions undergo a rigorous review process by a core team member and an additional mathematician with expertise in the relevant field. Reviewers provide feedback that allows authors to refine their problems before finalization. Alongside the main proof-writing tasks, authors are encouraged to add follow-up subquestions with final answers that can be automatically graded. These follow-ups enable a comparison between proof-writing and final-answer performance, while also supporting lower-cost evaluation across a broader range of models.

Evaluation process Evaluation is conducted in an agentic framework designed to mirror a research environment. Models have access to computational tools such as Python and SageMath (sag, 2025), as well as web search and multi-turn reasoning. Each model is first tasked with solving the main problem, followed by the associated follow-up subquestions. The main solution is graded by the problem’s author, who assigns a score from zero to three. Graders also annotate the types of errors, such as logical mistakes, and identify specific areas of partial progress, such as correct intermediate insights. Follow-up answers are automatically evaluated by comparing them with ground-truth solutions. We illustrate this process in Fig. 1, which shows a sample problem with two model solutions.

Key results We evaluate 10 state-of-the-art LLMs on the current version of IMProofBench. Our results show that models can already solve a small but meaningful fraction of research-level problems: the best model, GPT-5, produces complete solutions for 22% of tasks, closely followed by GROK-4 at 19%. Notably, GROK-4 achieves the highest final-answer accuracy at 52%, surpassing GPT-5’s 42%. Other models lag further behind, with CLAUDE-OPUS-4.1 scoring particularly poorly, only providing complete solutions in 3% of tasks.

Qualitative analysis Beyond aggregate scores, our analysis reveals that many models are prone to reasoning errors, ranging from simple logical mistakes to deep misconceptions unlikely to be exhibited by any professional mathematician in the relevant area. Indeed, almost half of the model solutions contain arguments revealing fundamental misunderstandings of mathematical concepts, as judged by human graders. Moreover, models frequently hallucinate existing results to obtain a (flawed) answer. Finally, models almost never abstain from providing a solution attempt, preferring to present convincing but incorrect proofs rather than admit they are stuck. At the same time, they also show a wide-ranging familiarity with existing literature and can often provide insights that could meaningfully support mathematicians on a substantial number of problems. These results indicate that state-of-the-art LLMs can already aid mathematicians in their research, but also still need significant supervision to avoid simple mistakes.

Core contributions The core contributions of this work are:

- IMProofBench, a private and evolving benchmark for research-level problems, developed in collaboration with the mathematical community.
- A systematic analysis of proof generation capabilities across state-of-the-art LLMs, demonstrating that GPT-5 provides fully justified solutions for a small but non-trivial fraction of our research-level problems, as judged by human expert graders.
- A qualitative analysis discussing both the difficulties and strengths of current state-of-the-art models and their potential application to research-level mathematics.

2 RELATED WORK

We briefly review existing benchmarks that evaluate LLMs on mathematical reasoning tasks.

Competition mathematics benchmarks High-school and undergraduate problems are the most common source of mathematical benchmarks due to their wide availability. Examples include GSM8k (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), along with more recent efforts such as OmniMath (Gao et al., 2024) and MathArena (Balunovic et al., 2025). However, these benchmarks fail to measure model performance on realistic, research-level tasks. Furthermore, even the most challenging competition problems are increasingly tractable for state-of-the-art LLMs (Balunovic et al., 2025), meaning that these benchmarks are reaching their saturation point.

Research-level benchmarks To move beyond competition problems, several benchmarks aim to capture research-level mathematical reasoning, though each has notable limitations, and none provide systematic proof evaluation. FrontierMath (Glazer et al., 2024) offers extremely challenging private problems, though privileged access by OpenAI raises concerns about evaluation fairness (AI, 2025). Humanity’s Last Exam (Phan et al., 2025) crowd-sources expert-level questions across domains, including mathematics, but suffers from contamination risks due to its open nature and reports of substantial noise in the benchmark (Skarlinski et al., 2025). RealMath (Zhang et al., 2025) sources problems from arXiv papers, enabling dynamic evaluation of research-level problems, but it is currently not being maintained. Finally, the UQ-Dataset (Nie et al., 2025) collects unsolved StackExchange questions, many of them mathematical. While promising, it lacks systematic human evaluation of proof validity, making consistent cross-model comparisons difficult.

Proof-based benchmarking efforts The importance of evaluating proof-generation capabilities has recently gained attention, leading to a range of benchmarking efforts. For example, Mahdavi et al. (2025) showed that models trained with reinforcement-style methods such as GRPO (Shao et al., 2024) perform poorly at proof writing. However, more recent evaluations on the USAMO and IMO 2025 demonstrated substantial progress in the ability of frontier models to construct rigorous mathematical arguments (Petrov et al., 2025; Balunovic et al., 2025). At the same time, other studies highlighted a persistent gap between final-answer accuracy and genuine proof-writing ability, indicating that final-answer benchmarks are not sufficient to measure mathematical capabilities (Guo et al., 2025; Dekoninck et al., 2025). Despite these advances, benchmarks remain focused on high-school and undergraduate mathematics, leaving research-level proof generation unexplored.

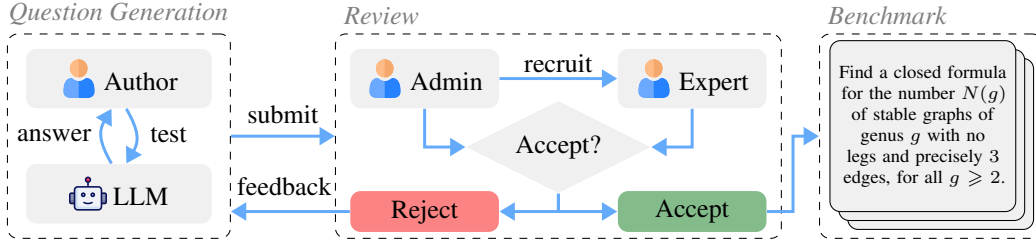


Figure 2: Workflow for question creation with peer review. Authors iteratively refine questions based on expert review. A problem is only accepted once the reviewers have no further comments.

Formal math benchmarks A complementary line of work evaluates LLMs on their ability to generate proofs in formal systems such as Lean (de Moura and Ullrich, 2021). Success in this setting typically requires fine-tuning frontier models for this particular task (Ren et al., 2025; Lin et al., 2025), as off-the-shelf LLMs perform poorly. Formal proofs offer the advantage of automatic verification and scalable evaluation, but current models still lag significantly behind their natural language proof counterparts (Dekoninck et al., 2025). Benchmarks in this space include PutnamBench (Tsoukalas et al., 2024) and MiniF2F (Zheng et al., 2022), which formalize problems from well-known mathematics competitions into Lean or Isabelle. Work on a Lean-based benchmark with research-level problems is currently in progress with the *ProofBench* initiative (Bowler and Carnesin).

3 BENCHMARK METHODOLOGY

In this section, we present the creation and evaluation process of IMProofBench. We begin by outlining our community outreach efforts (§3.1), followed by a description of the problem creation pipeline (§3.2) and the evaluation methodology (§3.3). Finally, we discuss the current state of the benchmark and our plans to maintain and extend it as a platform for continuous evaluation (§3.4).

3.1 COMMUNITY OUTREACH

Creating a novel and diverse collection of research-level problems is a challenging task, requiring professional mathematicians from a wide range of fields. To facilitate this, we undertook several initiatives to engage the community:

- **Workshops:** We organized several problem-creation sessions as satellite events at mathematical research conferences.
- **Posters and flyers:** We distributed informational materials in math common rooms and conference venues to reach graduate students, postdocs, and faculty.
- **Personal outreach:** Organizers and motivated contributors actively contacted their academic networks to invite participation.

These efforts are ongoing as we continue to expand the benchmark. Informal surveys of contributors indicate that key motivating factors to participate include convenient access to frontier models via the platform, curiosity about AI-generated responses to submitted questions, and the opportunity for co-authorship on resulting publications for contributors whose questions are accepted.

3.2 PROBLEM CREATION PIPELINE

Question creation As shown in Fig. 2, authors draft questions through a dedicated web interface and can immediately test them on an instance of GPT-5 configured with high reasoning effort, built-in web search and code interpreter tools, and safeguards such as a 30-minute timeout and a cap of 20 evaluations per day to prevent abuse. This LLM interaction allows quick, optional feedback on both difficulty and potential ambiguities. Importantly, problem selection criteria are independent of the model’s performance on the draft question. Where possible, authors are asked to include follow-up subquestions with unique, automatically gradable answers, with the option to assign point weights

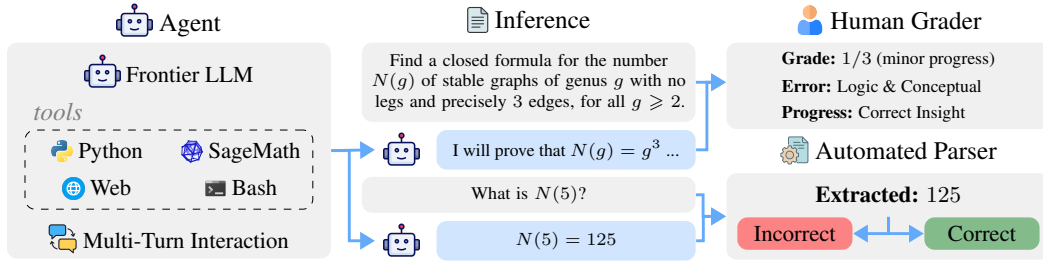


Figure 3: Evaluation workflow in a multi-turn environment with research tools. The main solution is graded by a human expert, while follow-up questions are automatically evaluated.

for the solution of different subquestions to reflect their difficulty or importance. This facilitates broader evaluation of more models by reducing reliance on human grading, while also supporting comparisons between final-answer accuracy and proof-generation capability. To guide contributions, authors receive detailed instructions that include illustrative examples and emphasize that questions should require PhD-level insight, while avoiding standard textbook exercises or computational problems. A complete description of the author instructions is provided in App. B.2.

Question peer-review process Once a question is submitted, an administrator recruits a reviewer whose expertise aligns with the problem’s subject area. Reviewers are invited via email, with invitations extended to both existing benchmark participants and external experts if necessary. The review process follows an academic peer-review model, with the administrator and reviewer providing detailed feedback, asking for revisions where necessary. While the reviewer concentrates on verifying mathematical correctness and difficulty, the administrator ensures that the submission adheres to the guidelines. Authors are then invited to revise their problem and respond to comments with clarifications or adjustments. A problem is accepted only after both the administrator and reviewer have no remaining concerns. A full description of the reviewer instructions is given in App. B.3.

3.3 MODEL EVALUATION

Evaluation environment As shown in Fig. 3, models are evaluated within an agentic framework designed to approximate real research conditions. We use the Inspect framework (AI Security Institute, 2024) and give models access to a diverse set of tools:

- **Python**: a full scientific environment with NumPy, SciPy, SymPy, and related libraries.
- **Bash**: an Arch Linux console with persistent filesystem and computer algebra systems like GAP (GAP, 2024), and Maxima (Maxima, 2025).
- **SageMath**: open-source mathematical software with specialized packages and mathematical databases (sag, 2025).
- **Web search**: a tool for retrieving literature and external references.

A full description of these tools is provided in App. E. To submit an answer, models must use a dedicated submit tool, which ensures a clear distinction between intermediate reasoning steps and the final output. The submitted answer is either presented to the human grader for main questions or compared with ground-truth answers for follow-up subquestions. Each model is allocated up to 300,000 tokens for main questions, with an additional 100,000 tokens available for each follow-up, supporting extended interaction and tool use.

Model selection and tiers To ensure scalability, we adopt a tiered evaluation system. Each model is assigned to a tier that reflects its priority for human grading, allowing question authors to focus on the most important submissions when their time is limited. The highest-priority tier includes state-of-the-art models that demonstrate strong performance on existing benchmarks: GPT-5 (OpenAI, 2025b), GEMINI-2.5-PRO (DeepMind, 2025), GROK-4 (xAI, 2025), and CLAUDE-OPUS-4.1 (Anthropic, 2025a). Lower tiers currently include O3 and O4-MINI (OpenAI, 2025a), GPT-4o (OpenAI, 2024), GEMINI-2.5-FLASH (DeepMind, 2025), GROK-3 (xAI, 2025), and CLAUDE SONNET 4 (Anthropic, 2025b). A complete description of the tiers is provided in App. D.

Grading process Scoring of model answers takes place in two separate stages. First, follow-up subquestions are automatically graded by comparing the model’s output with the ground-truth reference. Currently, this automated evaluation is also manually verified by an administrator, who can correct parsing errors and update the grading script if necessary. In the second stage, human grading is conducted through our dedicated web interface. The question’s author serves as grader and provides three types of feedback:

- **Error classification:** identifying reasoning mistakes caused by incorrect logic, hallucinations, calculation errors, or conceptual misunderstandings.
- **Achievement indicators:** marking whether the model demonstrated understanding, reached correct conclusions, identified key insights, or produced useful reasoning.
- **Overall progress:** assigning a score of no (0/3), minor (1/3), major (2/3), or full (3/3) progress.

Error classification and achievement indicators are recorded as eight binary marks and enable a more fine-grained analysis of model performance. In particular, this structure allows us to identify both the areas where models can already assist research mathematicians and the areas where they remain most prone to errors. To avoid bias, model identities remain hidden until grading is completed.

3.4 BENCHMARK STATISTICS AND FUTURE DEVELOPMENT

State of the benchmark IMProofBench is under active development, with this paper presenting its first pilot phase. This initial version consists of 39 questions and 79 follow-up subquestions. Topics range from areas of pure mathematics, such as algebraic geometry, combinatorics, and graph theory, to applied subjects such as stochastic analysis and bioinformatics. In Fig. 13 of App. A, we include a word cloud of question tags, weighted by frequency. Of the 39 benchmark problems, authors characterize 7 as open research questions. A total of 23 mathematical researchers have contributed at least one question in their area of expertise.

Continuous development With models showing rapid progress in mathematics, benchmarks are being saturated at an accelerating pace. For example, the USAMO 2025 benchmark moved from a solve rate below 5% to more than 60% in only a few months (Petrov et al., 2025; xAI, 2025). To ensure that IMProofBench remains both unsaturated and challenging, we are committed to its continuous development along several dimensions. First, we will maintain our problem creation pipeline and accept problems on a rolling basis, while forming new strategic partnerships with leading mathematical institutions to keep problem difficulty aligned with the capabilities of future models. Second, to prevent contamination, we will employ a dynamic problem management system in which authors are encouraged to revisit and possibly retire their problems once new publications or techniques make them significantly easier. Third, we plan to create a transparent interface that allows major companies or research labs to configure and provide their own agents for solving problems in IMProofBench. This creates an opportunity to provide objective and equitable evaluations of internal research models, ensuring that the benchmark reflects the latest state-of-the-art models and agents in realistic settings. Other ideas for future work are given in App. F.

4 EXPERIMENTAL RESULTS

In this section, we give quantitative and qualitative summaries of model performance on IMProofBench. In §4.1, we compare final-answer correctness and proof-generation capabilities of several frontier LLMs. Then, in §4.2, we present a detailed analysis of errors and achievements made by these models. In §4.3, we analyze token and tool usage. We conclude in §4.4 with a qualitative discussion of several notable examples and overall results.

4.1 MAIN RESULTS

Proof-based evaluation As illustrated in Fig. 5, GPT-5 achieves the strongest performance, producing a complete solution in 22% of cases. It fails to make any progress on only 17% of the questions, showing that the model can engage meaningfully with most problems in the benchmark. These results highlight both the impressive capabilities of current systems and the difficulty of IMProofBench, as substantial progress remains possible. Importantly, none of the 7 open problems were solved.

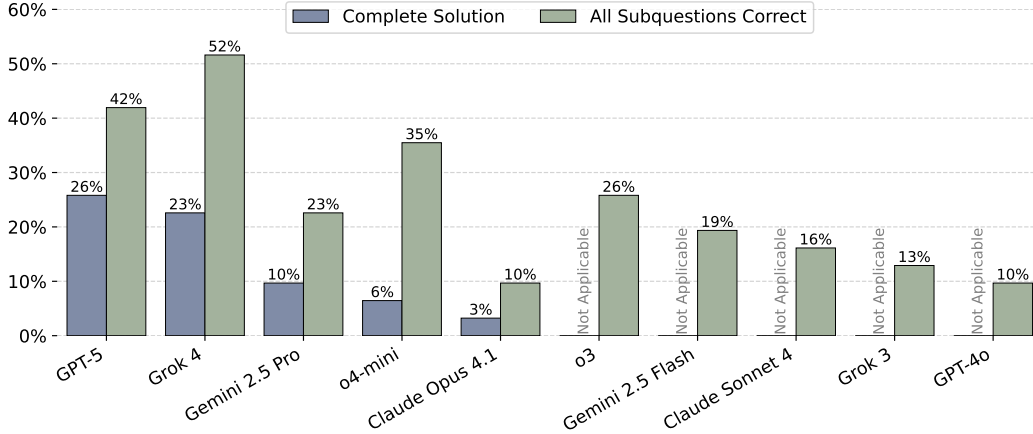


Figure 4: Results on the 31 questions that include follow-up subquestions and human grading.

Final-answer evaluation In Fig. 4, we compare performance between final-answer accuracy and full proof-based evaluation on the 31 questions that include both. While GROK-4 is slightly worse compared to GPT-5 on proof-based evaluation, it performs best on final-answer accuracy, obtaining 52%. The ranking of other models is consistent between the two evaluation modes. Furthermore, the correlation coefficient between author-weighted subquestion scores and the 0–3 progress scores assigned by human graders is 0.45. This suggests that final-answer evaluation is a useful proxy for model ability, but human grading provides essential nuance and a more refined view of performance. In App. A, we analyze final-answer accuracy over the full set of questions, including partial progress.

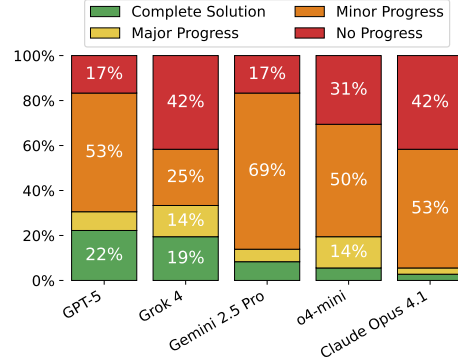


Figure 5: Results on IMProofBench.

4.2 ERROR AND PROGRESS ANALYSIS

We now analyze the error and achievement indicators classified by the question authors. This provides a clearer picture of where models fail and where they already provide meaningful help.

Error indicators As shown in Fig. 6, models make a wide variety of errors. Logical errors are the most common, with models frequently introducing unfounded assumptions or claiming incorrect implications. CLAUDE-OPUS-4.1 is particularly weak in this respect, having logical errors in nearly 80% of its responses. Conceptual errors are also widespread. Importantly, these errors are described as fundamental misunderstandings of mathematical concepts in the grader guidelines, showing that models do not fully understand some advanced mathematical concepts. Furthermore, hallucinations are surprisingly frequent, with GEMINI-2.5-PRO hallucinating results in 50% of its answers. In contrast, calculation mistakes are rare, which is expected since problems are proof-oriented and models can rely on tools to perform calculations. A notable outlier among all models is GROK-4: it often produces extremely short answers that only contain a final answer attempt without supporting arguments. This leads graders to be unsure about the precise mistakes or achievements in its reasoning.

Achievement indicators As shown in Fig. 7, most models demonstrate general familiarity with the background knowledge needed to understand the problems, which is an impressive achievement given that many of these questions reference highly specialized mathematical concepts. Creative ideas are rarer, but GPT-5 still displays non-trivial creativity in almost half its solutions. This indicates that the model can already make remarkable progress on difficult problems. Finally, in some cases, models provide insights that could be helpful to expert mathematicians, with GPT-5 offering meaningful contributions in about half its attempts. This is a significant achievement for any automated system.

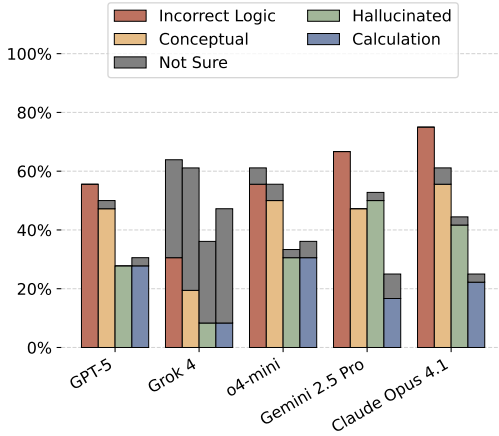


Figure 6: Error indicators

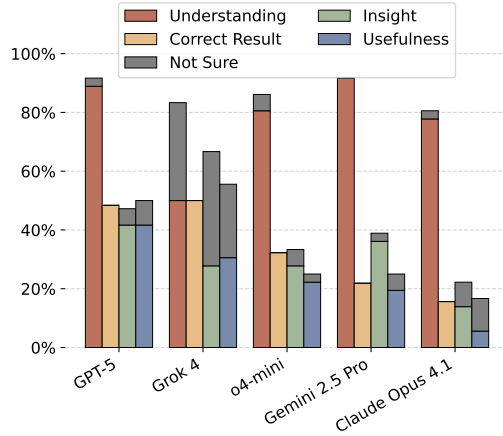


Figure 7: Achievement Indicators

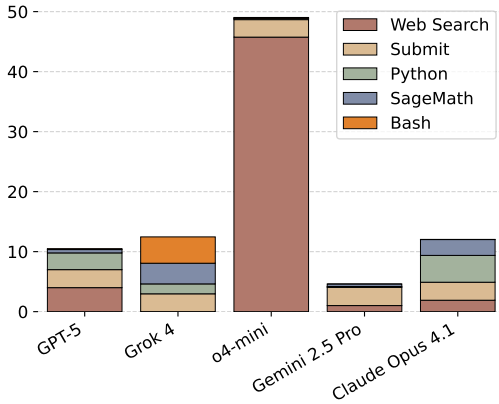


Figure 8: Average tool usage per question.

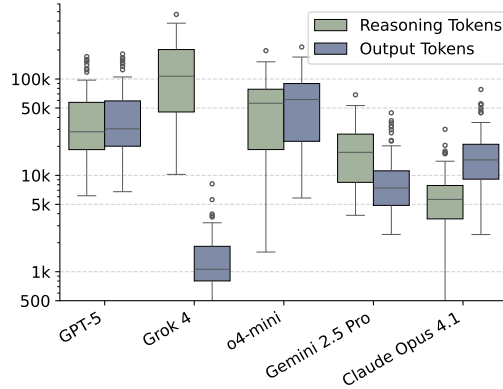


Figure 9: Token usage per question.

4.3 TOOL AND TOKEN USAGE

As illustrated in Fig. 8 and Fig. 9, models vary widely in their resource usage, both in tool selection and token consumption. GROK-4 spends almost three times as many reasoning tokens as other models while producing relatively few output tokens. It is also the only model to make heavy use of the bash tool. Inspection of its logs shows frequent use of the command line to download research papers from arXiv (via `wget` or `curl`) and to convert them using utilities like `strings` or `gs`. This sometimes gives GROK-4 an edge over models that rely only on internal search tools. Another pattern is that O4-MINI relies heavily on the web search tool, averaging over 40 searches per problem, while CLAUDE-OPUS-4.1 makes frequent use of Python, occasionally misusing it as a scratchpad with many comments or static `print` statements. Usage plots for all models, including those only evaluated on final-answer questions, are shown in App. A.

4.4 QUALITATIVE ANALYSIS

We now describe qualitative observations drawn from manual inspection of logs and grader comments.

Broad and deep literature knowledge Leading models such as GPT-5 show strong familiarity with the mathematical literature and are often able to identify specialized results in published work. However, they struggle to locate more obscure sources, such as private lecture notes, which human experts often use.

Use of specialized tools When confronted with complex computations, models frequently employ tools like SageMath. However, more specialized packages that are accessible through the bash tool pose challenges, with models often producing syntactically invalid code. After repeated failures, they sometimes revert to more common libraries, e.g., by using a manual Python re-implementation.

Mistakes are often hidden Models are typically quite economical with their mistakes, adding just a single simplifying assumption or incorrect claim. This one mistake often makes the problem significantly easier but leads to incorrect conclusions. Importantly, they are usually presented with confidence and framed rhetorically, for example, by stating that a “well-known result” implies a key step. Sometimes, different models even independently converge on the *same* shortcut, leading to parallel arguments that can create a false sense of consensus for the user. Although reasoning traces are often not accessible, we did not find evidence of *deliberate deception* where models were aware of their own mistakes and presented the flawed argument nonetheless.

Models rarely abstain Models rarely abstain from claiming a solution to the presented IMProofBench questions. Even on the extremely challenging open problems in the benchmark, models almost always make an attempt at a definite answer. This happens despite user preferences strongly favoring an abstention over a mistaken but convincing proof.

GROK-4 gives short responses As noted earlier, GROK-4 often provides only a final answer, particularly when the question allows for a short response. This occurs despite repeated instructions to provide full proofs (see App. H). Combined with the hidden reasoning tokens in the GROK-4 API, this made evaluations difficult and led to frequent “Not Sure” grades on our binary categories.

User testimonials For many contributors, this benchmark was their first hands-on experience with state-of-the-art LLMs in an agentic setup. Participants at outreach events expressed surprise at the level of performance (“*Quite impressive, especially the case of degree 3 where one has to argue a little bit...*”). During grading, we found that some models applied new approaches to known problems, surprising the expert graders (“*Interestingly, I was not familiar with the correct solution from the models, even though it is relatively fundamental.*”). Although no open problems were solved, some attempts received positive feedback (“*Still I am amazed by the quality of the one-shot answers.*”).

5 LIMITATIONS

The main limitation of IMProofBench is its current scale, with only 39 questions included so far. However, we are continuously expanding the benchmark, with an additional 17 problems at an advanced draft stage and 30 problems in the final stages of review. Even at this point, our analysis already provides detailed and valuable insights into the potential of LLMs for research-level mathematics, and these findings will become even more compelling as the benchmark develops further. Much smaller-scale evaluations of proof-based problems, such as those conducted on the USAMO and IMO 2025 (Petrov et al., 2025; Balunovic et al., 2025), have already produced meaningful conclusions, which underscores the value of such efforts even when the number of problems is small.

6 CONCLUSION

In this paper, we introduced IMProofBench, a benchmark designed to evaluate research-level proof-writing capabilities in LLMs. Unlike prior datasets that focus primarily on final answers, IMProofBench evaluates whether models can produce logically sound arguments that meet the standards of mathematical research. Each problem is authored and peer-reviewed by professional mathematicians, and evaluation takes place in an agentic framework that mirrors a real research environment. Our experiments with state-of-the-art LLMs show that models can already solve a meaningful subset of research-level problems, with GPT-5 achieving complete solutions on 22% of tasks. These findings highlight that while current models remain imperfect and prone to errors, they are already capable of providing valuable support to working mathematicians for some problems.

REPRODUCIBILITY STATEMENT

While the IMProofBench dataset remains private, we take several measures to ensure transparency of the resulting evaluations: we give detailed descriptions of tested models and their API configuration (App. D), an account of the tools available to them within the Inspect framework (App. E), and the used evaluation prompts (App. H). We plan to release the code base of our web platform and evaluation framework under a suitable open-source license before November 30, 2025, and to actively encourage scrutiny, feedback, and participation by outside developers. This release will include a continuously expanding collection of open sample problems that allow users to test the relevant systems and reproduce our data analysis on this sample set.

Moreover, we are open to scientific collaborations with outside parties to conduct specific investigations using our problem and grading dataset. Such requests will be evaluated on a case-by-case basis with the aim of ensuring the privacy commitments we make to our contributors.

ETHICS STATEMENT

We acknowledge that our project received support in the form of free credits for both the xAI and the Gemini APIs, for which we thank the teams at the respective companies. These contributions did not have an influence on our scientific evaluation of the respective models, which happens via a model-agnostic framework.

We address several further ethical considerations:

- **Contributor protection:** Problems remain private to protect contributors’ intellectual property, with generous withdrawal policies if questions lead to publishable insights. Contributors maintain rights to their content and receive co-authorship on benchmark publications.
- **Responsible AI evaluation:** By keeping the dataset private and focusing on evaluation rather than training data provision, we aim to measure capabilities without directly improving them.

ACKNOWLEDGMENTS

Acknowledgments We would like to thank Honglu Fan for his help and support throughout the project, including advice and feedback at the early planning stages. We are also deeply grateful to the IT Support group of the D-MATH at ETH, and in particular to Michele Marcionelli, who provided competent, patient and prompt support in setting up our benchmarking website.

We also thank the organizers of the 2025 Helvetic Algebraic Geometry Seminar, Oberwolfach Workshop “Recent Trends in Algebraic Geometry” and the ITS workshop “Computations in Algebraic Geometry: Complex, Real, and Tropical” for providing outreach platforms to advertise the benchmark in the form of satellite events and during poster sessions.

Financial acknowledgments (Project) We acknowledge that our project received support in the form of free credits for both the xAI and the Gemini APIs, for which we thank the teams at the respective companies.

Financial acknowledgments (Contributors) Various contributors were supported by grants:

- David Holmes is supported by ERC grant EAGL.
- Baran Hashemi is supported by the Excellence Cluster ORIGINS, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy-EXC2094-390783311.
- Chiara Meroni is supported by Dr. Max Rössler, the Walter Haefner Foundation, and the ETH Zürich Foundation.
- Johannes Schmitt was supported by SNSF grant 200020-219369 and SwissMAP.
- Charles Vial is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 491392403 – TRR 358.
- Gergely Bérczi was supported by DFF Grant 40296.

REFERENCES

- Sagemath — open-source mathematical software system. <https://www.sagemath.org/>, 2025. Accessed: 2025-09-15.
- Epoch AI. Clarifying the creation and use of the frontiermath benchmark. <https://epoch.ai/blog/openai-and-frontiermath>, January 2025. Accessed: 2025-09-25.
- UK AI Security Institute. Inspect AI: Framework for Large Language Model Evaluations, 2024. URL https://github.com/UKGovernmentBEIS/inspect_ai.
- Anthropic. Claude 4.1 system card (addendum). Technical report, Anthropic, August 2025a. Last updated September 15, 2025. Available at <https://assets.anthropic.com/m/4c024b86c698d3d4/original/Claude-4-1-System-Card.pdf>.
- Anthropic. Claude opus 4 & claude sonnet 4 system card. Technical report, Anthropic, May 2025b. Last updated September 2, 2025. Available at <https://www-cdn.anthropic.com/6d8a8055020700718b0c49369f60816ba2a7c285.pdf>.
- Benjamin Assarf, Ewgenij Gawrilow, Katrin Herr, Michael Joswig, Benjamin Lorenz, Andreas Paffenholz, and Thomas Rehn. Computing convex hulls and counting integer points with polymake. *Math. Program. Comput.*, 9(1):1–38, 2017. doi: 10.1007/s12532-016-0104-z. URL <http://dx.doi.org/10.1007/s12532-016-0104-z>.
- V. Baldoni, N. Berline, J. A. De Loera, B. Dutra, M. Köppe, S. Moreinis, G. Pinto, M. Vergne, and J. Wu. *A User’s Guide for LattE integrale v1.7.2*, 2013. URL <https://www.math.ucdavis.edu/~latte/>.
- Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanovic, and Martin T. Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *CoRR*, abs/2505.23281, 2025. doi: 10.48550/ARXIV.2505.23281. URL <https://doi.org/10.48550/arXiv.2505.23281>.
- Peter Belcak, Ard Kastrati, Flavio Schenker, and Roger Wattenhofer. Fact: Learning governing abstractions behind integer sequences. *Advances in Neural Information Processing Systems*, 35: 17968–17980, 2022.
- Suresh Bolusani, Mathieu Besançon, Ksenia Bestuzheva, Antonia Chmiela, João Dionísio, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Mohammed Ghannam, Ambros Gleixner, Christoph Graczyk, Katrin Halbig, Ivo Hedtke, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Dominik Kamp, Thorsten Koch, Kevin Kofler, Jurgen Lentz, Julian Manns, Gioni Mexi, Erik Mühmer, Marc E. Pfetsch, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Mark Turner, Stefan Vigerske, Dieter Weninger, and Lixing Xu. The SCIP Optimization Suite 9.0. ZIB-Report 24-02-29, Zuse Institute Berlin, February 2024. URL <https://nbn-resolving.org/urn:nbn:de:0297-zib-95528>.
- Nathan Bowler and Johannes Carmesin. Proofbench: a benchmark suite for mathematical proof verification and generation. in preparation.
- W. Bruns, B. Ichim, C. Söger, and U. von der Ohe. Normaliz. algorithms for rational cones and affine monoids. Available at <https://www.normaliz.uni-osnabrueck.de>.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021.
- Stéphane D’Ascoli, Pierre-Alexandre Kamienny, Guillaume Lample, and Francois Charton. Deep symbolic regression for recurrence prediction. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 4520–4536. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/d-ascoli22a.html>.

- Leonardo de Moura and Sebastian Ullrich. The lean 4 theorem prover and programming language. In André Platzer and Geoff Sutcliffe, editors, *Automated Deduction - CADE 28 - 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings*, volume 12699 of *Lecture Notes in Computer Science*, pages 625–635. Springer, 2021. doi: 10.1007/978-3-030-79876-5_37. URL https://doi.org/10.1007/978-3-030-79876-5_37.
- Wolfram Decker, Gert-Martin Greuel, Gerhard Pfister, and Hans Schönemann. SINGULAR 4-4-0 — A computer algebra system for polynomial computations. <http://www.singular.uni-kl.de>, 2024.
- Google DeepMind. Gemini 2.5 pro model card. Technical report, Google DeepMind, June 2025. URL <https://storage.googleapis.com/model-cards/documents/gemini-2.5-pro.pdf>. Last updated: June 27, 2025; Accessed: 2025-09-21.
- Jasper Dekoninck, Ivo Petrov, Kristian Minchev, Mislav Balunovic, Martin T. Vechev, Miroslav Marinov, Maria Drencheva, Lyuba Konova, Milen Shumanov, Kaloyan Tsvetkov, Nikolay Drenchev, Lazar Todorov, Kalina Nikolova, Nikolay Georgiev, Vanesa Kalinkova, and Margulan Ismoldayev. The open proof corpus: A large-scale study of llm-generated mathematical proofs. *CoRR*, abs/2506.21621, 2025. doi: 10.48550/ARXIV.2506.21621. URL <https://doi.org/10.48550/arXiv.2506.21621>.
- EpochAI. Introducing FrontierMath Tier 4, 2025. URL <https://x.com/EpochAIResearch/status/1943744468781289543>.
- Simon Frieder, Luca Pinchetti, Alexis Chevalier, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Petersen, and Julius Berner. Mathematical capabilities of chatgpt. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/58168e8a92994655d6da3939e7cc0918-Abstract-Datasets_and_Benchmarks.html.
- Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. Omni-math: A universal olympiad level mathematic benchmark for large language models. *CoRR*, abs/2410.07985, 2024.
- GAP – Groups, Algorithms, and Programming, Version 4.14.0. The GAP Group, 2024. URL <https://www.gap-system.org>.
- Thibault Gauthier and Josef Urban. Learning program synthesis for integer sequences from scratch. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7670–7677, 2023.
- Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järvinen, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, Natalie Stewart, Bogdan Grechuk, Tetiana Grechuk, Shreepranav Varma Enugandla, and Mark Wildon. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in AI. *arXiv*, 2024.
- Jiaying Guo, Wenjie Yang, Shengzhong Zhang, Tongshan Xu, Lun Du, Da Zheng, and Zengfeng Huang. Right is not enough: The pitfalls of outcome supervision in training llms for math reasoning. *CoRR*, abs/2506.06877, 2025. doi: 10.48550/ARXIV.2506.06877. URL <https://doi.org/10.48550/arXiv.2506.06877>.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*, 2021.

- Yong Lin, Shange Tang, Bohan Lyu, Ziran Yang, Jui-Hui Chung, Haoyu Zhao, Lai Jiang, Yihan Geng, Jiawei Ge, Jingruo Sun, Jiayun Wu, Jiri Gesi, Ximing Lu, David Acuna, Kaiyu Yang, Hongzhou Lin, Yejin Choi, Danqi Chen, Sanjeev Arora, and Chi Jin. Goedel-prover-v2: Scaling formal theorem proving with scaffolded data synthesis and self-correction. *CoRR*, abs/2508.03613, 2025. doi: 10.48550/ARXIV.2508.03613. URL <https://doi.org/10.48550/arXiv.2508.03613>.
- Hamed Mahdavi, Alireza Hashemi, Majid Daliri, Pegah Mohammadipour, Alireza Farhadi, Samira Malek, Yekta Yazdanifard, Amir Khasahmadi, and Vasant Honavar. Brains vs. bytes: Evaluating llm proficiency in olympiad mathematics. *arXiv preprint arXiv:2504.01995*, 2025.
- Maxima. Maxima, a computer algebra system. version 5.48.0, 2025. URL <https://maxima.sourceforge.io/>.
- Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60:94–112, 2014. ISSN 0747-7171. doi: 10.1016/j.jsc.2013.09.003. URL <https://www.sciencedirect.com/science/article/pii/S0747717113001193>.
- Fan Nie, Ken Ziyu Liu, Zihao Wang, Rui Sun, Wei Liu, Weijia Shi, Huaxiu Yao, Linjun Zhang, Andrew Y. Ng, James Zou, Sanmi Koyejo, Yejin Choi, Percy Liang, and Niklas Muennighoff. Uq: Assessing language models on unsolved questions, 2025. URL <https://arxiv.org/abs/2508.17580>.
- OEIS Foundation Inc. The On-Line Encyclopedia of Integer Sequences, 2025. Published electronically at <http://oeis.org>.
- OpenAI. Gpt-4o system card. Technical report, OpenAI, August 2024. Available at <https://cdn.openai.com/gpt-4o-system-card.pdf>.
- OpenAI. Openai o3 and o4-mini system card, April 2025a. URL <https://cdn.openai.com/pdf/2221c875-02dc-4789-800b-e7758f3722c1/o3-and-o4-mini-system-card.pdf>.
- OpenAI. Gpt-5 system card. Technical report, OpenAI, August 2025b. URL <https://cdn.openai.com/gpt-5-system-card.pdf>. Accessed: 2025-09-21.
- Ivo Petrov, Jasper Dekoninck, Lyuben Baltadzhiev, Maria Drencheva, Kristian Minchev, Mislav Balunović, Nikola Jovanović, and Martin Vechev. Proof or bluff? evaluating llms on 2025 usa math olympiad. *arXiv preprint arXiv:2503.21934*, 2025.
- Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Adam Khoja, Ryan Kim, Jason Hausenloy, Oliver Zhang, et al. Humanity’s last exam. *arXiv*, 2025.
- Z. Z. Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, Z. F. Wu, Zhibin Gou, Shirong Ma, Hongxuan Tang, Yuxuan Liu, Wenjun Gao, Daya Guo, and Chong Ruan. Deepseek-prover-v2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *CoRR*, abs/2504.21801, 2025. doi: 10.48550/ARXIV.2504.21801. URL <https://doi.org/10.48550/arXiv.2504.21801>.
- Johannes Schmitt and Jason van Zelm. Intersections of loci of admissible covers with tautological classes. *Selecta Math. (N.S.)*, 26(5):Paper No. 79, 69, 2020. ISSN 1022-1824. doi: 10.1007/s00029-020-00603-4. URL <https://doi.org/10.1007/s00029-020-00603-4>.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL <https://doi.org/10.48550/arXiv.2402.03300>.
- Michael Skarlinski, Jonathan Laurent, Andy Bou, and Adam White. About 30% of humanity’s last exam chemistry/biology answers are likely wrong. <https://www.futurehouse.org/research-announcements/hle-exam>, July 23 2025. FutureHouse Research. Accessed: 2025-09-01.

PARI/GP version 2.17.2. The PARI Group, Univ. Bordeaux, 2024. available from <http://pari.math.u-bordeaux.fr/>.

George Tsoukalas, Jasper Lee, John Jennings, Jimmy Xin, Michelle Ding, Michael Jennings, Amityayush Thakur, and Swarat Chaudhuri. Putnambench: Evaluating neural theorem-provers on the putnam mathematical competition. *CoRR*, abs/2407.11214, 2024.

xAI. Grok 3 beta — the age of reasoning agents, February 2025. URL <https://x.ai/news/grok-3>. Accessed: 2025-04-03.

xAI. Grok 4 model card. Technical report, xAI, August 2025. Last updated August 20, 2025. Available at <https://data.x.ai/2025-08-20-grok-4-model-card.pdf>.

Jie Zhang, Cezara Petrucci, Kristina Nikolic, and Florian Tramèr. Realmath: A continuous benchmark for evaluating language models on research-level mathematics. *CoRR*, abs/2505.12575, 2025. doi: 10.48550/ARXIV.2505.12575. URL <https://doi.org/10.48550/arXiv.2505.12575>.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. minif2f: a cross-system benchmark for formal olympiad-level mathematics. In *ICLR*. OpenReview.net, 2022.

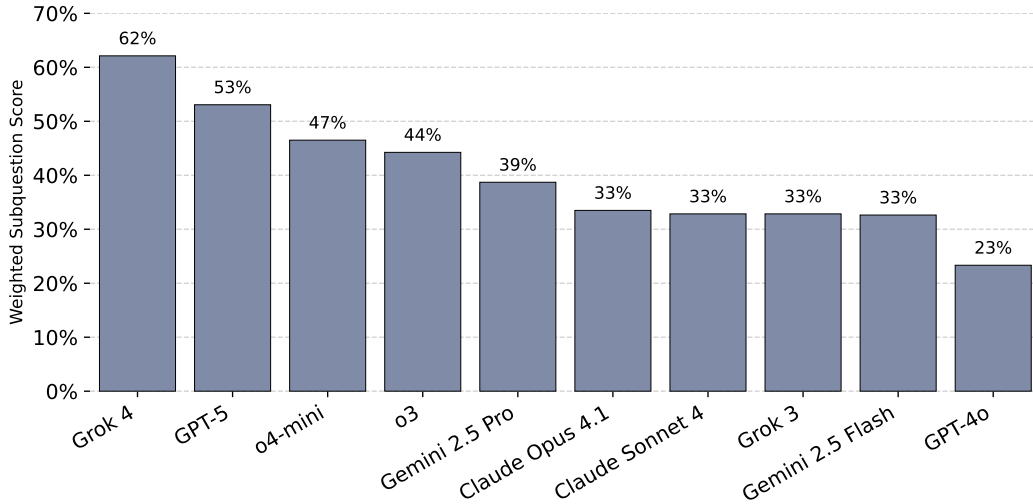


Figure 10: Average percentage of points for subquestion evaluation. Here, performance on any individual question is weighted by the point rewards determined by the problem author.

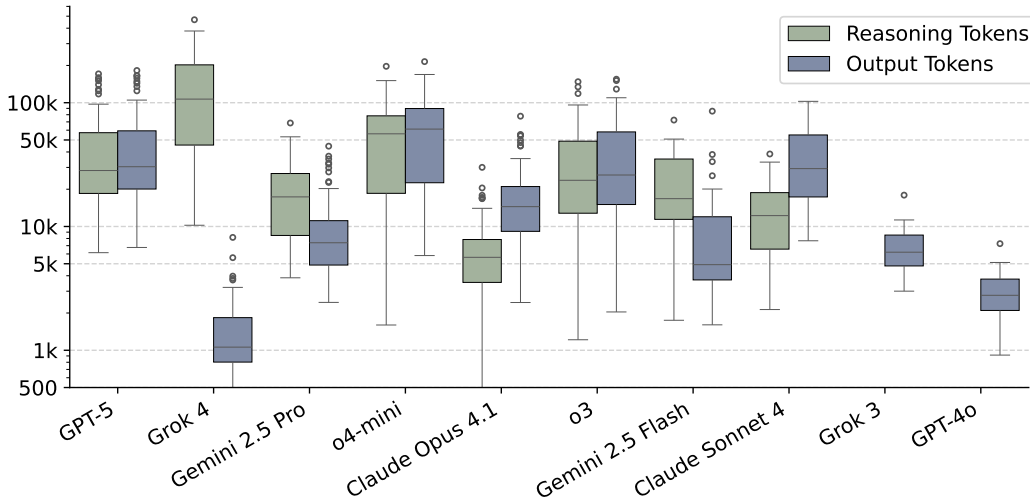


Figure 11: Token usage distribution for problem evaluation (main question and subquestions) for all tested models.

A BENCHMARK COMPOSITION AND ADDITIONAL EVALUATION RESULTS

Performance on final-answer subquestions In Fig. 10, we present the average scores obtained by all 10 evaluated models on the final-answer subquestions, using the author-appointed weights that reflect importance or difficulty. As shown in the figure, GROK-4 achieves the highest performance, with almost a 10% margin over the second-ranked model, GPT-5.

Token usage In Fig. 11, we show the distribution of reasoning and output tokens across the evaluated questions. GROK-4 produces the longest reasoning traces but the shortest outputs among all models in the benchmark, consistent with the trend described in §4.4. In contrast, the OpenAI models show a more balanced ratio of reasoning to output tokens. The Gemini models use slightly more reasoning tokens, while the Claude models generate more verbose outputs. With respect to token limits, which allow 300k tokens for the main question and 100k tokens for each subquestion, models almost always remain well below these thresholds.

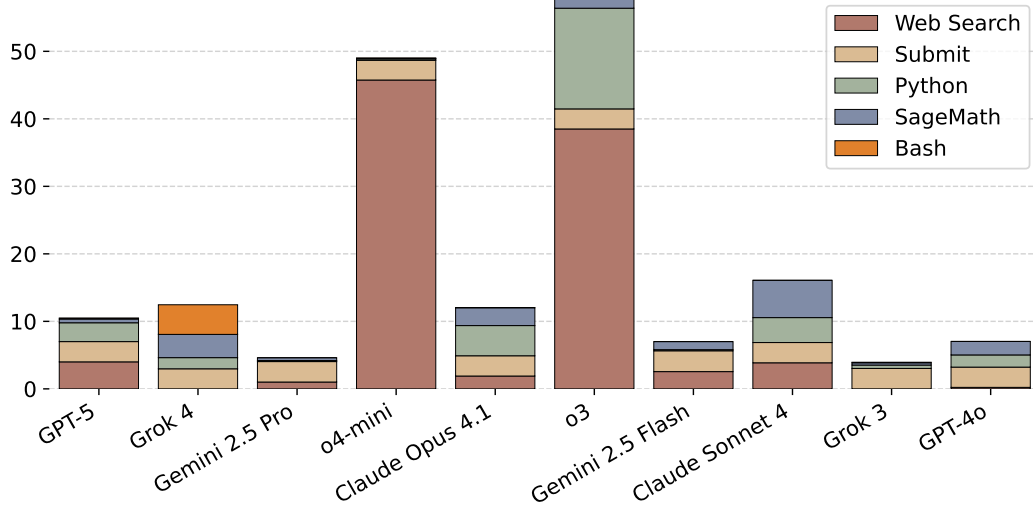


Figure 12: Average tool usage for all tested models.

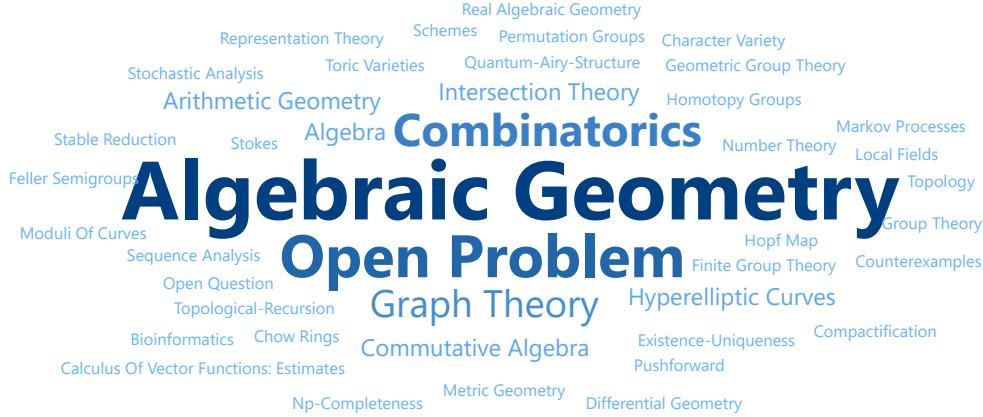


Figure 13: Word cloud of tags assigned to IMProofBench problems.

Tool usage In Fig. 12, we show the average tool usage across models. The patterns differ substantially. O4-MINI and O3 each make around 50 tool calls per problem, relying more heavily on the web search tool than any other model in the leaderboard. Further, GROK-4 is the only model that makes frequent use of the bash tool. Other models display broadly similar usage patterns, distributing their calls among web search, Python, and SageMath.

Topics in IMProofBench In Fig. 13, we display the distribution of problem tags in IMProofBench. The topic of "Algebraic Geometry" currently dominates, reflecting the research focus of the benchmark organizers. These organizers both contributed problems themselves and solicited input primarily from colleagues in their own academic networks. Future development of the benchmark will aim to broaden its coverage to include a wider range of topics in pure and applied mathematics, as outlined in App. F.

B HUMAN INTERFACE AND INSTRUCTIONS

In this appendix, we discuss how contributors and benchmark administrators interact with IMProofBench, including the instructions and interface for different steps of the submission process (question generation, review, and grading). In App. B.1, we give a brief overview of the main pages on the web interface. Then, in App. B.2, we provide details on how questions are created and edited. In App. B.3, we explain the review process. Finally, in App. B.4, we discuss the grading interface.

B.1 SUBMISSION WEBSITE

Contributors submit problems via a secure website designed for submitting and reviewing questions, and grading AI answers (see Fig. 14). Features include:

- **User accounts and permissions:** Contributors can create an account tied to a (verified) email, which allows them to author questions and use website features like the free AI solution previews for these questions. Benchmark administrators have additional access to manage model evaluations, review requests, and access a live view of benchmark results.
- **Community features:** The website shows a list of contributors (ordered by the number of accepted questions or similar parameters) to encourage active participation, and links to a project Zulip with further news and an opportunity to provide feedback.
- **Benchmark dashboard:** Total numbers of contributors and questions in different stages of the submission process are displayed to show project progress. An overview page with both live results and archived snapshots of the benchmark state will be added in the future.
- **About the project:** Information about the IMProofBench is provided. This information contains the initial whitepaper, an overview of core team members, a timeline of planned steps, and a page with frequently asked questions. A privacy policy detailing our handling of user data is linked in the footer of the page.

B.2 QUESTION CREATION AND EDITING

Benchmark problems are created through a structured interface that guides contributors through the submission requirements. The system provides comprehensive guidelines (see Figure 15) emphasizing the key characteristics of suitable benchmark problems.

Problem guidelines Effective benchmark problems must meet several criteria:

- **PhD-level difficulty:** Problems should be suitable for oral exams of graduate courses, research papers, or advanced seminars, representing mathematics close to or at research-level.
- **Genuine mathematical insight:** Solutions must require non-routine approaches that cannot be solved through pattern matching or standard algorithm application.
- **Clear proof-based main question:** The primary answer should consist of a complete mathematical argument rather than merely a numerical result.
- **Auto-gradable subquestions:** Each problem requires 2–3 subquestions with unique answers (e.g., “Is the statement true for $n = 5$?” or “What is the rank of this group?”), enabling automated evaluation.

Contributors should avoid problems solvable by lucky guessing, standard textbook exercises (even from graduate texts), or purely computational problems that mathematical software can solve directly.

Question editing interface The question creation and editing window (see Figure 16) provides a comprehensive authoring environment with the following components:

- **Main question editor:** A text area supporting Markdown with LaTeX mathematics, featuring a live preview pane that renders the formatted content in real-time. Contributors can use standard LaTeX delimiters (\dots for inline and \dots for display mathematics).

- **Problem metadata:** A tags field allows contributors to categorize problems by area (e.g., “group theory”, “representation theory”, or “permutation groups”) and special characteristics (e.g., “open problem” for questions where the author seeks but does not know the answer).
- **AI solution preview:** Contributors can test their questions against a frontier AI model (currently GPT-5 with high reasoning effort) using up to 20 free attempts per day. This feature helps authors evaluate whether their problem has appropriate difficulty and clarity.
- **Sample solution:** A dedicated editor for the complete solution, which serves as the reference for reviewers and graders. The solution should demonstrate the expected level of rigor and detail to allow expert review to verify correctness and serve as a reference for grading model answers.
- **Subquestions management:** A dynamic form system for adding multiple subquestions, where each subquestion consists of:
 - Question text (supporting Markdown and LaTeX)
 - Expected answer field for the unique answer
 - Evaluation method selector (e.g., exact match)
 - Optional points value (defaulting to 1) for weighting subquestions by difficulty or importance
 - Rationale field for explaining the correct answer

Question detail view Once submitted, questions are displayed in a detail view (see Figure 17) that presents all components in their rendered form. This view shows:

- The question status in the submission pipeline (Draft → Under Review → Approved → Active)
- Rendered the main question and sample solution with properly formatted mathematics
- List of subquestions with their expected answers
- AI solution attempt preview when available
- Review comments from expert reviewers (when in review stage)
- Response interface allowing authors to address reviewer feedback and revise their submission

The detail view serves as the central hub for tracking a question’s progress through the review process and facilitating communication between authors and reviewers.

B.3 REVIEW PROCESS AND INSTRUCTIONS

Each question is reviewed by at least one expert before being included in the benchmark. These experts are invited to submit a review via email. An example of such an email is included below.

Reviewer invitation email

Dear [invited_user],

My name is [inviting_user] and I am part of a small team of mathematicians studying the question of how good today’s AI models are at solving research-level math questions. As part of this IMProofBench project, we are building a collection of challenging mathematical problems to use for testing the AI performance.

We would like to ask for your help in verifying the mathematical correctness of one such question. If you are interested to learn more about the project, further information is available at <https://improofbench.math.ethz.ch/faq/>

The following question was submitted for inclusion in the IMProofBench dataset:

Title: Permutation representation
Author: Example Participant

Would you be willing to review this question and:

- Verify that the phrasing is well-defined and unambiguous
- Confirm the provided solution is mathematically correct
- Make any suggestions for improvements (e.g., additional unique-answer subquestions)

We estimate that for most problems, this should take between 10 and 30 minutes.

You can view the full submitted problem and write a review at:

[ACCEPT_URL]

There, you will also have the option to decline this review request after viewing the question.

Alternatively, you can decline immediately by clicking:

[DECLINE_URL]

If you provide a review, the question’s author will be notified and have the chance to revise the question and compose a response. After seeing the response, you have the option to submit a further review or recommend the question for acceptance in the benchmark.

Thank you for considering this request!

Best regards,
[inviting_user]

Note: To track your review and allow you to see the author’s replies, accepting the review request will create a user account for you on our website. You can optionally set a password after submitting your review to log back in and e.g., contribute a question to the benchmark yourself.

When the reviewer accepts the review invitation by clicking on the link, they are forwarded to a webpage displaying the problem to be reviewed, along with a form for review submission and further information (see Figure 18). The reviewer may also view the full review guidelines displayed in Figure 19. The review consists of a short comment by the reviewer indicating improvements and/or mistakes in the question statement. Before submitting the review, the reviewer decides on a recommended action among the following: “Recommended for acceptance”, “Needs revision” and “Not suitable”. The site admins are notified when a review is complete and can take action accordingly. If the reviewer selects “Not suitable”, the question is automatically reset to the “draft” status. Independent of the outcome, the author is permitted to submit an answer to the reviewer’s comments and change the question if necessary. The reviewer may then either submit a new review taking into account the changes, or a new reviewer may be invited.

B.4 GRADING INTERFACES

The grading system provides a structured interface for human evaluation of model-generated proofs through a dedicated web page.

Human grading interface The main grading interface (see Figure 20) employs a three-column layout designed to facilitate easy access to relevant information and the feedback form:

- **Left column:** Displays the question statement and sample solution for reference
- **Center column:** Shows the model’s complete response with mathematical rendering
- **Right column:** Contains the interactive grading panel with scoring controls

To prevent bias, model identities are concealed behind randomized aliases (Answer A, B, C, etc.) that remain hidden until all answers for a question have been graded. The system maintains independent grading sessions for each evaluator, with aliases shuffled differently to ensure blind evaluation.

Grading categories The scoring form consists of three main components providing multifaceted evaluation, with relevant information available via concise tooltips:

AI mistake indicators: Four binary categories identifying common failure modes:

1. **Incorrect Logic:** Flawed logical steps or invalid reasoning
2. **Hallucinated:** References to non-existent theorems, papers, or results
3. **Calculation:** Arithmetic or algebraic errors
4. **Conceptual:** Fundamental misunderstanding of mathematical concepts

AI achievement indicators Four binary categories recognizing positive aspects:

5. **Understanding:** Correctly identifies what needs to be proven or calculated

6. **Correct Result:** Arrives at the correct final answer (with N/A option for open-ended problems or when the correct answer is unknown)
7. **Insight:** Shows creative problem-solving or novel approaches
8. **Usefulness:** Solution would be helpful to someone learning this topic

Each binary category offers three response options: “True”, “False”, or “Not Sure”, allowing graders to indicate uncertainty when evaluation is ambiguous.

Overall progress A four-point scale (0–3) rating overall solution progress:

- **0/3:** No progress toward solution
- **1/3:** Minor progress with limited advancement
- **2/3:** Major progress with substantial work completed
- **3/3:** Complete solution achieved

This overall progress score serves as the primary metric for model ranking and comparison.

Additional grading features The interface includes several supporting elements to ensure grading consistency and quality:

- **Grading notes:** A persistent text area where graders record their evaluation criteria and decision patterns across all answers (e.g., “Matrix errors count as Calculation, Theory errors as Logic”). These notes help maintain consistency when grading multiple model responses and facilitate reproducibility in future grading sessions.
- **Comments field:** Answer-specific observations about edge cases or explanations for grading decisions.
- **Auto-save functionality:** Grading selections are automatically preserved with a 2-second debounce to prevent data loss.
- **Focus mode:** An optional distraction-free interface that maximizes screen space by hiding navigation elements and allowing collapsible panels, enabling graders to concentrate on detailed evaluation.
- **Flag for organizers:** Option to mark responses requiring special attention due to serious issues or technical problems.

The grading workflow supports iterative evaluation, allowing graders to mark answers as complete, incomplete, or given up (for responses that cannot be meaningfully evaluated). Once all model answers for a question are marked complete, the system reveals the true model identities, enabling post-hoc analysis of performance patterns.

C SAMPLE PROBLEM

Below, we present an example of a problem from the benchmark and discuss model performance and solution strategies from our evaluation.

Background for reader (not included in benchmark question) A *stable graph* is a connected graph $\hat{\Gamma}$, multi-edges and loops allowed, together with a vertex-labeling by non-negative integers $(g_v)_{v \in V(\hat{\Gamma})}$ satisfying that each vertex v with $g_v = 0$ has valence at least 3. These combinatorial objects appear in algebraic geometry in the study of moduli spaces of stable curves, see e.g. (Schmitt and van Zelm, 2020, Section 2). The *genus* of $\hat{\Gamma}$ is defined as $g = b_1(\hat{\Gamma}) + \sum_{v \in V(\hat{\Gamma})} g_v$, with b_1 the first Betti number (or cyclomatic number) of $\hat{\Gamma}$.

Question Given an integer $g \geq 2$, let N_g be the number of isomorphism classes of stable graphs of genus g with precisely 3 edges. Give a closed formula for N_g valid for all $g \geq 2$.

Solution To compute N_g , we note that each stable graph $\hat{\Gamma}$ has an undecorated underlying graph Γ , which is one of the 10 connected multi-graphs with precisely 3 edges. Then N_g can be calculated by summing over those graphs Γ and counting the number of assignments g_v to the vertices of Γ , avoiding double-counting by taking into account symmetries of Γ .

The final answer is that for $g = 2$ we have $N_2 = 2$ and for $g \geq 3$, we have

$$N_g = \begin{cases} \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - 2 & \text{if } g \equiv 0 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{155}{72} & \text{if } g \equiv 1 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - \frac{20}{9} & \text{if } g \equiv 2 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{19}{8} & \text{if } g \equiv 3 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - \frac{16}{9} & \text{if } g \equiv 4 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{187}{72} & \text{if } g \equiv 5 \pmod{6} \end{cases}$$

Subquestions What is N_3 ? (Answer: 9) What is N_8 ? (Answer: 114) What is N_{10000} ? (Answer: 111198615276)

Model approaches and performance

- GPT-5 instantly identifies the solution strategy in its first reasoning step, writing *”Essentially, I’m computing N_g as a sum of connected multigraph types limited by 3 edges and considering partitions of genera”*. It performs a Python calculation to obtain the first experimental data. From theoretical considerations, it correctly identifies the shape of the final answer, writing *”Ultimately, I want a final closed formula for N_g as a degree-3 quasi-polynomial with a period of 6.”*. After a few attempts, it calculates this polynomial via Lagrange interpolation on datapoints with fixed residue modulo 6, discovering that the case $g = 2$ needs separate treatment. This not only represents a perfect solution to the given problem, but also mirrors precisely the approach of the human question author to solving the problem.
- GROK-4 obtains an expression for N_g in a single reasoning step, though no further details are available as the GROK-4 API does not expose reasoning summaries. The model then uses a Python tool to calculate the first values and the SageMath tool to look up the resulting integer sequence in the OEIS database OEIS Foundation Inc. (2025). This being unsuccessful, it submits a very concise sketch of its answer, which is slightly less simple than the formula for N_g above, as it still features a summation over $g - 2$ terms.
In a second evaluation, GROK-4 uses the bash tool to download textbooks on algebraic graph theory and moduli spaces of curves and convert them to text. Lacking the software tools for the latter, it tries and fails to install new packages on the sandboxed Docker container, receiving an error for attempting to use sudo rights. Finally, it abandons these attempts and just submits a solution that is mostly correct, but has some small errors in one of the terms.
- CLAUDE-OPUS-4.1 also tries to combine combinatorial arguments with computer calculations in SageMath, but fails to find even the contribution from 2-vertex graphs, forgetting some topological possibilities for Γ . One noteworthy pattern is that the model includes very verbose reasoning in the form of comments and static print statements within the SageMath code.
- GEMINI-2.5-PRO starts with a correct calculation of N_2, N_3, N_4 . However, then it makes the completely unfounded claim that *”This implies that N_g is a quadratic polynomial in g .”*, whereas in reality it is a *cubic quasi-polynomial*. It then submits an answer based on that wrong assumption. It does get partial credit in the subquestions for calculating $N_3 = 9$ correctly.

D MODEL TIERS

We evaluate models across four tiers based on their capabilities and release timeline. Tier 1 comprises current frontier models with state-of-the-art mathematical reasoning capabilities. Tier 3 includes previous-generation models that have demonstrated strong mathematical performance. Tier 4 contains legacy models included for historical comparison and baseline establishment. Currently, only models

in Tiers 1–3 are included in human grading to focus evaluation resources on the most relevant comparisons.¹

Table 1: Models evaluated in IMProofBench, organized by tier

Tier	Model	API Endpoint	Parameters
1	CLAUDE-OPUS-4.1	claude-opus-4-1-20250805	cache_prompt="auto" max_tokens=32000 reasoning_tokens=31000
	GPT-5	gpt-5	reasoning_effort="high" reasoning_summary="auto"
	GEMINI-2.5-PRO	gemini-2.5-pro	reasoning_tokens=32768
	GROK-4	grok-4-0709	—
3	O4-MINI	o4-mini-2025-04-16	reasoning_effort="high" reasoning_summary="auto"
4	CLAUDE SONNET 4	claude-sonnet-4-20250514	cache_prompt="auto" max_tokens=64000 reasoning_tokens=63000
	GPT-4o	gpt-4o-2024-11-20	—
	GEMINI-2.5-FLASH	gemini-2.5-flash	reasoning_tokens=24576
	GROK-3	grok-3	—
	O3	o3-2025-04-16	reasoning_effort="high" reasoning_history="auto" reasoning_summary="auto" reasoning_tokens=100000

All models are evaluated using the Inspect framework with standardized prompting and tool access, including Python execution, web search, and SageMath for advanced mathematical computation (see App. E). The `reasoning_effort` parameter, when specified as "high", enables enhanced reasoning capabilities for models that support it. The `reasoning_tokens` parameter controls the maximum length of the model’s internal reasoning process, while `max_tokens` limits the total response length, including both reasoning and final answer.

E DETAILED TOOL DESCRIPTIONS

The evaluation environment for IMProofBench was designed to emulate the computational resources available to research mathematicians when solving complex problems. Rather than restricting models to basic arithmetic operations, we provide access to the same sophisticated mathematical software that researchers routinely use in their work. This approach reflects the reality that modern mathematical research frequently involves computational exploration, symbolic manipulation, and verification of conjectures through extensive calculation.

E.1 TECHNICAL SPECIFICATIONS

All tools operate within the following constraints to balance computational power with practical limitations:

- **Timeout:** 15 minutes per tool invocation
- **Memory limit:** 8 GB RAM per execution
- **Environment:** Isolated Docker container running Arch Linux
- **Execution model:** Independent tool calls (no variables persist between calls), but files written to the filesystem remain accessible throughout the evaluation session

¹Tier 2 is reserved for testing Command Line Interface models such as Claude Code, but implementation has been deferred to a future version of the benchmark.

E.2 CORE COMPUTATIONAL TOOLS

E.2.1 PYTHON ENVIRONMENT

The Python tool provides access to a comprehensive scientific computing environment (Python 3.13.7). This language was chosen for its prevalence in scientific computing and the extensive familiarity that language models demonstrate with its syntax and libraries. The environment includes standard numerical and symbolic computation packages:

- **Numerical computing:** NumPy, SciPy, pandas
- **Symbolic mathematics:** SymPy, SymEngine
- **Visualization:** Matplotlib (though output is text-based)
- **Graph theory:** NetworkX, igraph, graph-tool
- **Optimization:** CVXPY with multiple backend solvers (GLPK, ECOS, OSQP, SCS, CSDP)
- **Machine learning:** Basic scikit-learn functionality

Each Python execution runs independently with no variables or imports preserved between invocations, though files written to disk remain accessible for subsequent tool calls.

E.2.2 BASH SHELL ACCESS

The bash tool provides command-line access to the evaluation environment, enabling models to leverage specialized mathematical software that operates through command-line interfaces. This tool serves as the gateway to domain-specific mathematical systems detailed in Section E.3.

E.2.3 SAGEMATH

SageMath (sag, 2025) (version 10.6) serves as the primary computer algebra system, providing a unified Python-based interface to numerous mathematical software packages. Its significance in the research community stems from its comprehensive coverage of mathematical domains and its philosophy of combining the best open-source mathematics software into a coherent system.

Key features available through the `sage_computation` tool include:

- Natural mathematical syntax through automatic preparsing (e.g., x^2 for exponentiation, $K.<a>$ for field extensions)
- Extensive algebraic capabilities: polynomial rings, number fields, elliptic curves, modular forms
- Combinatorial structures: graphs, matroids, posets, designs
- Specialized packages: `adm_cycles` for moduli spaces of curves, `ore_algebra` for D-finite functions and recurrence operators, `pari_jupyter` for enhanced PARI/GP integration
- Integration with external systems: automatic interfacing with GAP, Maxima, PARI/GP, Singular

E.3 SPECIALIZED MATHEMATICAL SOFTWARE

The evaluation environment includes a comprehensive suite of specialized mathematical software, accessible through the bash tool:

E.3.1 COMPUTER ALGEBRA SYSTEMS

- **GAP** (Groups, Algorithms, Programming): Specialized system for computational discrete algebra, particularly group theory and combinatorics GAP (2024)
- **Maxima**: General-purpose computer algebra system for symbolic computation, descended from MIT's Macsyma Maxima (2025)
- **PARI/GP** (version 2.17.2): High-performance system focused on number theory computations The (2024)
- **Singular**: Specialized system for polynomial computations, commutative algebra, and algebraic geometry Decker et al. (2024)

- **Polymake** (version 4.14): System for research in polyhedral geometry and related areas Assarf et al. (2017)

E.3.2 ALGEBRAIC AND GEOMETRIC COMPUTATION

- **Normaliz**: Computation of normalizations of affine semigroups and rational cones Bruns et al.
- **LattE integrale**: Lattice point enumeration and integration over convex polytopes Baldoni et al. (2013)
- **Gfan**: Gröbner fans and tropical varieties computation
- **4ti2**: Algebraic, geometric, and combinatorial problems on linear spaces
- **msolve**: Polynomial system solving over finite fields and rational numbers

E.3.3 GRAPH THEORY AND COMBINATORICS

- **nauty and Traces**: Graph automorphism and canonical labeling McKay and Piperno (2014)
- **bliss**: Another efficient graph automorphism tool
- **igraph**: Network analysis and graph algorithms library

E.3.4 OPTIMIZATION SOLVERS

- **Linear Programming**: GLPK (GNU Linear Programming Kit), Gurobi-compatible interfaces
- **Mixed-Integer Programming**: SCIP (Solving Constraint Integer Programs) Bolusani et al. (2024)
- **Semidefinite Programming**: CSDP, DSDP for SDP problems
- **SAT Solvers**: glucose, kissat, cryptominisat for Boolean satisfiability

E.3.5 PROOF ASSISTANTS AND VERIFICATION

- **Lean** (de Moura and Ullrich, 2021): Interactive theorem prover and functional programming language
- **Mathics**: Open-source alternative to Mathematica for symbolic computation

E.3.6 NUMERICAL AND SCIENTIFIC COMPUTING

- **Julia**: High-performance language for numerical computing
- **SciLab**: Numerical computational package similar to MATLAB
- **FLINT**: Fast Library for Number Theory
- **NTL**: High-performance number theory library

E.4 DATA RESOURCES

The environment includes numerous mathematical databases accessible through SageMath:

- Stein-Watkins database of elliptic curves
- Jones' database of number fields
- Kohel database for elliptic curves and modular polynomials
- Cunningham tables for factorizations
- OEIS (Online Encyclopedia of Integer Sequences) integration
- Various polytope databases and mutation class data

E.5 WEB SEARCH CAPABILITIES

The `web_search` tool provides access to current mathematical literature and online resources. The implementation follows a provider-based architecture:

- **Internal providers:** Models from OpenAI, Anthropic, and Grok utilize their respective built-in web search capabilities, requiring no additional API keys
- **External provider:** Tavily is configured as a fallback for models without internal search capabilities (e.g., Gemini), providing AI-optimized search results

Some models, notably GROK-4, combine web search capabilities with the `wget` bash command to download full research papers for detailed analysis.

E.6 EXAMPLE TOOL USES FROM BENCHMARK EVALUATION

Below, we list some example tool applications that occurred during our model evaluations. In each case, the full log file of the multi-turn evaluation reveals that the respective calculation played a decisive role in allowing the model to find the correct answer. To preserve benchmark privacy, we describe the relevant tool uses in general terms while leaving out the details of the specific benchmark problem.

- **Generating functions** (Model: GROK-4, Tool: SageMath)
Solved combinatorics problem by calculating a generating function $F(x)$ and forming the exponential $G(x) = \exp(F(x))$ to extract a specific coefficient from G
- **Modular forms** (Model: GROK-4, Tool: SageMath)
Compute q -expansion of the weight 12 cusp form Δ
- **Group theory** (Model: GPT-5, Tool: GAP (2024) via Bash Shell)
Accessed entries of the character table of a sporadic group
- **Literature access** (Model: GROK-4, Tool: Bash Shell)
Model uses `curl` to download PDF of paper from arXiv, installs the `PyPDR2` package via `pip`, and converts the PDF to text to obtain relevant information for the benchmark problem. Note: after an initial failed attempt at installing the `PyPDR2` package, the model uses the `pip` argument `--break-system-packages` to force a user installation in the externally managed Python environment of our sandboxed evaluation environment.

F PLANS FOR FUTURE DEVELOPMENT

Below, we give further details on our plans for the continuous development of IMProofBench.

- **Scale and outreach:** We aim to expand the benchmark to 150–300 problems, e.g. through strategic partnerships with leading mathematical institutions (e.g., MFO Oberwolfach, IAS, Fields Institute) and by recruiting domain-specific ambassadors who can promote participation at conferences and within their research networks.
- **Quality assurance and grading:** To strengthen the scientific validity of our evaluations, we will study inter-rater reliability by comparing expert gradings on the same problems. We will support graders via AI-assisted pre-screening of model answers and refine our error classification system to localize specific mistakes within solution texts rather than applying only global categories.
- **Dynamic problem management:** As mathematical knowledge evolves, problems may become easier due to new publications or techniques. We will implement a generous retirement policy allowing authors to withdraw problems affected by recent research, while regularly adding fresh problems to maintain benchmark difficulty. We also plan to release small sets of sample problems to provide the community with concrete reference points for gauging AI progress.
- **Technical innovation:** We plan to develop automated difficulty classifiers to predict which problems challenge current AI systems, explore alternative evaluation formats (such as formula reconstruction tasks and interactive problem-solving sessions), and implement bring-your-own-agent interfaces to enable companies to test internal models against the benchmark.
- **Model coverage:** Beyond proprietary frontier models, we will evaluate leading open-source reasoning systems like DeepSeek-V3.1-Terminus and Qwen3-235B-Think, promoting the strongest to Tier 1 status for human grading, ensuring long-term comparison baselines even as commercial models are deprecated. *Note:* One reason why these models were not included in this initial version of the benchmark is the ongoing challenges with enabling tool use for these models

- a requirement to put them on equal footing with other models within the inspect evaluation framework of IMProofBench.
- **Evaluation modalities:** Building on the existing IMProofBench platform and contributor network, we plan to explore further problem types and evaluation methodology. This includes:
 - combinations of informal and formalized questions and solutions (e.g., in collaboration with the ProofBench project Bowler and Carmesin),
 - specialized task formats with wide importance to mathematical research, such as formula reconstruction for sequence data of natural/rational numbers, polynomials, ... (see e.g. Gauthier and Urban (2023); Belcak et al. (2022); D’Ascoli et al. (2022)),
 - interactive or collaborative proof attempts, including provision of hints or feedback to the model during evaluation time, more closely mimicking the setting of a researcher using commercially available AI systems.

G USE OF LARGE LANGUAGE MODELS

We report our use of LLMs throughout this research project. The authors take full responsibility for all content in this paper, including any LLM-assisted portions.

G.1 WRITING AND PRESENTATION

Claude Opus 4.1 was used to generate an initial draft of Sections 3 and Appendices B, D, E, and G, and provided feedback and suggestions for our Reproducibility and Ethics Statements. Additionally, Claude Opus 4.1 provided proofreading assistance and offered stylistic and structural suggestions. GPT-5 was used to collect bibliography entries of software packages in Appendix D. All LLM-generated content was thoroughly reviewed, fact-checked, and edited by the authors.

G.2 LITERATURE DISCOVERY AND RELATED WORK

During the ideation phase and preparation of the benchmark whitepaper, we used ChatGPT o3 and Claude Opus Research to conduct comprehensive searches of the benchmarking literature and identify related projects. These tools helped surface relevant prior work and ensure thorough coverage of the existing landscape, though all citations were independently verified by the authors.

G.3 RESEARCH IMPLEMENTATION AND DEVELOPMENT

LLMs played a substantial role in implementing the benchmark infrastructure. Claude Code, supplemented by ChatGPT’s Codex CLI tool, assisted with:

- Development of the benchmark website and database architecture
- Adaptation of the Inspect framework for model evaluation
- Extraction and visualization of quantitative results

Additionally, Claude Opus provided support for organizational tasks, including meeting summaries and the creation of promotional materials. The background image for our benchmark poster was generated using GPT-5’s multimodal capabilities.

All code and implementations were tested, validated, and debugged by the authors to ensure correctness and functionality.

H EVALUATION PROMPTS

Main Question Prompt

```
# Background
The IMProofBench project is a mathematical reasoning benchmark for AI systems, testing
their ability to solve research level math problems. Each such problem consists of one **
main question**, where the expected answer is a longform mathematical proof, and several
```

related **subquestions** which have short, unique answers (e.g. a natural number). The main answer will be graded by both human expert mathematicians (often the author of the question) and AI evaluators, whereas subquestion answers are checked automatically using a Python script.

Structure of the evaluation

In the following we would like to evaluate your mathematical reasoning abilities on one such problem. The overall structure of the conversation below is that we iterate through the questions in order (main question, sub-question 1, sub-question 2, ...) and in each step, you can:

- Read the current question
- Think about it in a multi-turn environment with tool use (see below)
- Submit the answer to the current question

At each point in the conversation, you have the context of the entire previous conversation including your outputs in the thinking steps and the record of any tool uses. Note that you will *not necessarily* have access to records of your internal reasoning traces and internal tool uses, so any helpful information from these should be documented in your (external) thinking outputs.

Multi-turn reasoning environment

To help you solve the problem, you will have access to a multi-turn conversation environment with optional tool use, based on the Inspect AI framework. At each step, you can:

- Think out loud to analyze the problem, devise a solution approach, think through the steps of mathematical arguments, etc.
- Use the 'python' tool to run self-contained experiments in a standard python environment
- Use the 'bash' tool to execute commands inside a docker container (running ArchLinux with some open-source mathematical software installed)
- Use the 'web_search' tool to search for current information, mathematical definitions, theorems, or recent research
- Use the 'sage_computation' tool for conducting an experiment in a self-contained SageMath terminal session
- Use the 'submit' tool to provide your final answer to the current question (main or sub-question)

All tools have a timeout of 15 minutes, maximal memory usage (RAM) of 8 GB and run on standard 2025 hardware.

Token constraints

You have {main_question_token_limit:,} tokens to solve the main question, and {subquestion_token_limit:,} tokens for each of the following sub-questions. This counts both your output tokens (including in tool calls) and your reasoning tokens. You are informed about your current usage after each conversation turn.

Answer format for main question

Below you will see the text of the main question. Once you finished reasoning about it, you can register your answer using the 'submit' tool. The answer for the main question should be a detailed mathematical argument, formatted in Markdown with LaTeX formulas using $...$ for inline mathematical expressions and $...$ for equations. Use Markdown [link formatting] (<https://www.markdownguide.org/basic-syntax/#links>) for including online references, *not* any internal web-referencing system.

Problem:

{question_text}

Please work through this problem step by step. When you have your final answer, use the submit() tool to submit it.

Subquestion Prompt

Great work on the previous part!

You have successfully completed the previous question. Now please solve the following subquestion while keeping the context of your previous work:

Subquestion {subquestion_order}:
{subquestion_text}

Instructions:

- You can reference your work from previous parts
- Use the same mathematical tools available to you

- When you have your final answer, use the `submit()` tool to submit it
- Be precise and specific in your answer format

Please proceed with solving this subquestion.

Conversation Status Update

Please continue working on the current question. To formally register your answer, use the 'submit' tool as per the original instructions above. Note: the conversation will only proceed to the next stage once you use the 'submit' tool.

Token usage: {current_tokens:,} of {token_limit:,} tokens used for this stage.

Python tool description

Use the python function to execute Python code.

The Python tool executes single-run Python scripts. Important notes:

1. Each execution is independent - no state is preserved between runs
2. You must explicitly use `print()` statements to see any output
3. Simply writing expressions (like in notebooks) will not display results
4. The script cannot accept interactive input during execution
5. Return statements alone won't produce visible output
6. All variables and imports are cleared between executions
7. Standard output (via `print()`) is the only way to see results
8. This tool has a timeout of 15 minutes and maximal memory usage (RAM) of 8 GB

Bash tool description

Use this function to execute bash commands. Underlying system is ArchLinux with many standard open-source computer algebra systems (like GAP) pre-installed. This tool has a timeout of 15 minutes and maximal memory usage (RAM) of 8 GB.

Web search tool description

Use this function to search the web for current information, mathematical definitions, theorems, or recent research.

This tool gives you access to up-to-date information that can help with:

- Looking up mathematical definitions and theorems
- Finding recent research papers or results
- Verifying computational results against known databases
- Checking current mathematical conventions or notation
- Finding examples of similar problems or techniques

The search results will include titles, URLs, and relevant excerpts from web pages. Use this tool when you need information that might not be in your training data or when you want to verify facts.

Sage tool description

Use the `sage_computation` function to run calculations in the open-source mathematics software system SageMath.

The `sage_computation` tool executes single-run SageMath scripts. Important notes:

1. Each execution is independent - no state is preserved between runs
2. You must explicitly use `print()` statements to see any output
3. Simply writing expressions (like in notebooks) will not display results
4. The script cannot accept interactive input during execution
5. Return statements alone won't produce visible output
6. All variables and imports are cleared between executions
7. Standard output (via `print()`) is the only way to see results
8. This tool has a timeout of 15 minutes and maximal memory usage (RAM) of 8 GB

All standard SageMath functions are pre-imported and available. The SageMath parser is applied, so you can use natural mathematical syntax.

Key Features:

- Natural syntax: Use x^2 for powers, $K.\langle a \rangle$ for field extensions

- All mathematical objects pre-imported: Matrix, EllipticCurve, PolynomialRing, etc.
- Advanced packages available: admcycles for moduli spaces, and many more

Examples:

```
# Factor a polynomial
factor(x^100 - 1)

# Define a number field
K.<a> = NumberField(x^3 - 2)

# Work with elliptic curves
E = EllipticCurve([0, 1])
print(E.rank())

# Use specialized packages (example with admcycles)
from admcycles import *
G = StableGraph([1,1],[[1,3],[2,4]],[(1,2),(3,4)])
print(f"Automorphisms^2: {G.automorphism_number()^2}")
```

IMPORTANT: Like the python() tool, you must use print() to see any output. Nothing is returned automatically - always print your results!

Submit tool description

Submit your final answer for the current question or subquestion. Use Markdown + LaTeX formatting.

The answer for the main question should be a detailed mathematical argument.

Your answer should be formatted as natural Markdown text with LaTeX formulas.

Use \$ for inline math and \$\$ for display math, or \begin{equation} environments.


Use standard [Markdown link syntax](<https://www.markdownguide.org/basic-syntax/#links>) for online references.

RECOMMENDED: Use raw strings (r''' or r"") to write LaTeX naturally without escaping.

Important formatting notes:

- Write your answer exactly as you would in a math document
- Use raw triple quotes r''' for multiline answers with LaTeX
- This lets you write \frac, \sqrt, \int naturally (no escaping needed)
- Include full mathematical reasoning with the final answer clearly stated
- Do not use custom macros (e.g., \Z, \Q, \RR, etc.). Only use valid standard LaTeX commands


IMProofBench Mathematical Reasoning Benchmark
participant@example.com
Admin
Profile
Sign Out



IMProofBench

Informal Mathematical Proof Benchmark


IMProofBench evaluates the ability of AI systems to create research-level mathematical proofs. We maintain a curated, private repository of PhD-level problems across pure mathematics to measure genuine mathematical reasoning capabilities while preventing data contamination and benchmark overfitting.



Questions

Create and review mathematical proof problems to test frontier AI models.


- Browse Questions
- Problem Guidelines
- Create New Question



Community

Connect with mathematical researchers and track contributions.

- Participants & Leaderboard
- Zulip Chatroom



Dashboard

Real-time statistics and benchmark performance metrics.

PARTICIPANTS


138

Total Participants

QUESTIONS

147 Drafts	27 Under Review
37 Accepted	28 Graded


[all Benchmark Results](#)



About

Learn about the benchmark's goals, methodology, and team.

- White Paper
- Team
- Timeline
- FAQ




Administration

Manage and monitor model evaluations, view results, and control the evaluation queue.


- Model Evaluations
- Model Scores
- Review Invitations
- Database Administration

Key Features




AI Model Testing

Test problems against frontier models with immediate feedback




Peer Review System

Expert review ensures problem quality and appropriate difficulty



Automated Grading

Subquestions enable objective evaluation alongside proof assessment




Privacy Preservation

Majority private dataset prevents overfitting and gaming

IMProofBench © 2025 | Mathematical Reasoning Benchmark for AI Research | Privacy Policy


Figure 14: Landing and overview page of IMProofBench website.


IMProofBench Mathematical Reasoning Benchmark


participant@example.com
Admin
Profile
Sign Out


Problem Guidelines

Creating high-quality benchmark problems for mathematical AI evaluation

 Quick Start


Effective benchmark problems require **PhD-level difficulty**, **genuine mathematical insight**, and **2-3 auto-gradable subquestions**. Think about recent calculations from your research that required a clever insight or non-obvious proof techniques.

 Required Characteristics

 What to Avoid

- **PhD-level difficulty:** Suitable for qualifying exams, research papers, or advanced seminars
- **Requires genuine insight:** Not solvable by routine application of known algorithms
- **Clear proof-based main question:** Answer should be a complete mathematical argument, not just a number
- **2-3 unique-answer subquestions:** Enable automated evaluation (e.g., "Is the statement true for $n=5$ ", "What is the rank of this group?")

- Problems solvable by pattern matching or lucky guessing
- Standard textbook exercises (even from graduate texts)
- Purely computational problems that Mathematica/SageMath can solve directly
- Problems without clear subquestions for automated evaluation

 Problem Templates

Intersection Theory

Main: Let X be [variety]. Compute the class of [specific cycle] in the Chow ring $A^*(X)$.

Subquestions: What is the degree of this class? Does it vanish in $A^2(X)$?

Main: For the moduli space M of [objects], compute a closed formula for the intersection number $\int_M \alpha_1 \cup \alpha_2 \cup \dots \cup \alpha_n$.

Subquestions: What is this number for specific parameter values?


Classification Problems

Main: Classify all [objects] with [property]. Give explicit representatives for each isomorphism class.

Subquestions: How many classes are there? Which have additional property P ?

Main: What is the rank of the cohomology group $H^d(M)$ for [variety/moduli space M]?

Subquestions: What is $\dim H^0(M)$? Is $H^n(M) = 0$ for $n > d$?

 Example Problems

Example 1: Stable Graphs

Main question: Find a closed formula for the number $N(g)$ of stable graphs of genus g with no legs and precisely 3 edges, for all $g \geq 2$.

Subquestions:

- What is $N(3)$?
- What is $N(8)$?
- What is $N(1000)$?

Example 2: Permutation Representations

Main question: Let G be a finite group. Is the functor $\text{Perm} : G\text{-sets} \rightarrow \text{Rep}_{\mathbb{C}}(G)$ sending X to its permutation representation fully faithful? Prove or provide a counterexample.

Subquestions:

- Is the statement true for all finite groups?
- Is the statement true for all finite cyclic groups?
- Is the statement true for all finite abelian groups?

Brainstorming Tips

→ A tricky calculation from your recent work that required a clever insight

→ An "obvious" statement that actually needs a non-trivial proof

→ A self-contained lemma that came up in a research project

→ An oral exam question for an advanced course

Ready to Contribute?

Start creating your problem using our editor with LaTeX support and AI testing.

Create New Problem

IMProofBench © 2025 | Mathematical Reasoning Benchmark for AI Research | Privacy Policy

Figure 15: Guidelines for authoring benchmark problems.

[illegible]

Figure 16: Window for editing questions, solutions, and their associated subquestions; via the blue button, the user can request up to 20 free AI solution previews per day to check the suitability of the question.

IMProofBench Mathematical Reasoning Benchmark
participant@example.com Admin Profile Sign Out

Questions · Permutation representation

Permutation representation

Example Participant September 24, 2025

Question

Given a finite group G acting on a finite set X , let $\text{Perm}(X)$ be the complex permutation, whose underlying vector space is \mathbb{C}^X . Give a proof or a counter-example of the following statement:

Claim (1):

In the setting above, two finite G -sets X_1, X_2 admit a G -equivariant bijection $\phi: X_1 \rightarrow X_2$ if and only if the representations $\text{Perm}(X_1)$ and $\text{Perm}(X_2)$ are isomorphic as complex G -representations.

Solution

The claim is false. A counter-example is $G = (\mathbb{Z}/2\mathbb{Z})^{\otimes 3}$. Let

$$Y_1 = \bigoplus_{|H|=4} G/H, Y_2 = \bigoplus_{|H|=2} G/H, Y_3 = G,$$

where the sums are over subgroups $H \subseteq G$ of the given cardinalities and all actions carry their natural transitive G -action by translation. Then for

$$X_1 = Y_1 \sqcup Y_2^{-7}, X_2 = Y_2^{-3}$$

we have: $\ast X_1$ is not G -equivariantly bijective to X_2 since X_1 has 8 orbits and X_2 has 3 orbits under the G -action. We have an equality of permutation representations $\text{Perm}(X_1) \cong \text{Perm}(X_2)$ since the characters χ of these two permutations satisfy $\chi(0) = |X_1| = 84$ and $\chi(a) = |X_1^a| = 3$ for all $a \in G \setminus \{0\}$, where X_1^a is the fixed set of a acting on X_1 . Here we use that each a is contained in precisely 3 subgroups H of G of order 4 and precisely one such subgroup of order 2.

Latest AI Solution Attempt (Sep 24, 2025 at 8:08 AM)

Claim (1) is false in general.

"Only if" direction

If there is a G -equivariant bijection $\phi: X_1 \rightarrow X_2$, the linear map $\mathbb{C}^{X_1} \rightarrow \mathbb{C}^{X_2}$ sending the basis vector δ_x to $\delta_{\phi(x)}$ is a G -equivariant linear isomorphism. Hence $\text{Perm}(X_1) \cong \text{Perm}(X_2)$ as complex G -representations.

Counterexample to the converse

Let $G = S_4$. For a subgroup $H \leq G$, write G/H for the transitive G -set of left cosets and denote its representations. The converse is false: in S_4 , the non-isomorphic G -sets

$$X_1 = G/((12)(34)) \sqcup 2 \cdot G/S_3, \quad X_2 = G/((123)) \sqcup 2 \cdot G/((12), (34))$$

have identical permutation characters (hence isomorphic permutation representations) but different orbit decompositions, hence no equivariant bijection.

500568ms | \$ 50.4792 | gpt-5 with web search and code interpreter

Tags: group theory representation theory permutation groups

Subquestions

Subquestion a 1 pts

Is Claim (1) from the main question above true?

Answer: No

Rationale: See counter-example in main solution.

Subquestion b 1 pts

Is Claim (1) from the main question above true under the additional assumption that G is Abelian?

Answer: No

Rationale: The counter-example from the main solution applies here as well.

Waiting for Reviews

Your question is currently under review. You'll see feedback here once reviewers submit their comments.

Difficulty Ratings

Background: 2 Reasoning: 2

Insight: 3 Compute: 2

Scale: 1 (Easy) to 5 (Very Hard)

Metadata

Question ID: #238

Created: September 2, 2025 4:13 PM

Last Modified: September 24, 2025 8:20 AM

Subquestions: 4 items

Revert to Draft

Edit Question

Assign Reviewer

Approve Question

Grade Model Answers

Retract Question

Delete Question

Back to Questions

Figure 17: Overview page of question data (with main question, sample solution, AI answer preview, and subquestions).

IMProofBench Mathematical Reasoning Benchmark
participant@example.com Admin Profile Sign Out

Questions > Permutation representation > Review

Question to Review

Permutation representation

Given a finite group G acting on a finite set X , let $\text{Perm}(X)$ be the complex permutation, whose underlying vector space is \mathbb{C}^X . Give a proof or a counter-example of the following statement:

Claim (1):

In the setting above, two finite G -sets X_1, X_2 admit a G -equivariant bijection $\phi: X_1 \rightarrow X_2$ if and only if the representations $\text{Perm}(X_1)$ and $\text{Perm}(X_2)$ are isomorphic as complex G -representations.

Author's Solution

The claim is false. A counter-example is $G = (\mathbb{Z}/2\mathbb{Z})^{\oplus 3}$. Let

$$Y_1 = \bigoplus_{|H|=4} G/H, Y_2 = \bigoplus_{|H|=2} G/H, Y_3 = G,$$

where the sums are over subgroups $H \subseteq G$ of the given cardinalities and all actions carry their natural transitive G -action by translation. Then for

$$X_1 = Y_1 \sqcup Y_3^{*7}, X_2 = Y_2^{*3}$$

we have: X_1 is not G -equivariantly bijective to X_2 since X_1 has 8 orbits and X_2 has 3 orbits under the G -action. We have an equality of permutation representations $\text{Perm}(X_1) \cong \text{Perm}(X_2)$ since the characters χ of these two permutations satisfy $\chi(0) = |X_1| = 84$ and $\chi(a) = |X_1^a| = 3$ for all $a \in G \setminus \{0\}$, where X_1^a is the fixed set of a acting on X_1 . Here we use that each such a is contained in precisely 3 subgroups H of G of order 4 and precisely one such subgroup of order 2.

Author: Johannes Schmitt Created: Sep 2, 2025
Tags: group theory, representation theory, permutation groups

Difficulty Ratings (1-5)

Background Reasoning
Insight Compute

Subquestions

Subquestion a 5 pts

Is Claim (1) from the main question above true?

Answer: No

Rationale: See counter-example in main solution.

Subquestion b 1 pts

Is Claim (1) from the main question above true under the additional assumption that G is Abelian?

Answer: No

Rationale: The counter-example uses an Abelian group $G = (\mathbb{Z}/2\mathbb{Z})^{\oplus 3}$.

Subquestion c 2 pts

Is Claim (1) from the main question above true under the additional assumption that G is cyclic?

Answer: Yes

Rationale: A G -set is determined by its mark, i.e. by the cardinalities $(X^H)_{H \subseteq G \text{ subgroup}}$. As every subgroup $H = \langle h \rangle$ is cyclic, and we have $X^H = X^h$ we can reconstruct these numbers since the cardinality of X^h equals the trace of the permutation matrix associated to the element $h \in G$. That trace is the character of $\text{Perm}(X)$, evaluated at h .

Subquestion d 2 pts

Is Claim (1) from the main question above true under the relaxed assumption that G is a compact Lie group acting on a compact manifold X , replacing $\text{Perm}(X)$ with the smooth functions $C^\infty(X)$ seen in the category of Fréchet spaces with continuous G -action?

Answer: No

Rationale: The counter-example from the main solution applies here as well.

Submit Review

Decision* ☒ Submit anonymously

You must select a decision for your review. Check to submit review anonymously (your name will not be shown to the author)

Comment*

Good question with correct answer!

Formulation of question could ask more explicitly which of the claimed directions \implies or \impliedby holds.

Suggest adding tag: "group actions"

Your review feedback using Markdown + LaTeX. Be constructive and specific about issues or improvements needed.

[Review Guidelines](#)

What makes a good benchmark question?

- Clear problem statement:** Unambiguous mathematical notation and well-defined objectives
- Appropriate difficulty:** Challenging but solvable within the target domain
- Complete solution:** Author should provide a correct, detailed solution
- Proper formatting:** Good use of LaTeX and clear mathematical presentation

Common issues to look for:

- Typos or grammatical errors
- Unclear or ambiguous wording
- Missing or incorrect mathematical notation
- Solution errors or incomplete reasoning
- Inappropriate difficulty rating

Decision guidelines:

- Accept:** Ready for benchmark inclusion with minimal or no changes
- Needs revision:** Good question but requires specific improvements
- Not suitable:** Fundamental issues that make it inappropriate for the benchmark

[View Full Guidelines](#)

Cancel

IMProofBench © 2025 | Mathematical Reasoning Benchmark for AI Research | Privacy Policy

Figure 18: Question review window showing text box for feedback and review instruction summary.

IMProofBench Mathematical Reasoning Benchmark
participant@example.com Admin Profile Sign Out

Questions > Review Guidelines

Review Guidelines

Standards and best practices for reviewing mathematical proof questions

☆ What Makes a Good Benchmark Question?

Mathematical Content

- Clear problem statement:** Unambiguous mathematical notation and well-defined objectives
- Appropriate difficulty:** Challenging but solvable within the target domain
- Mathematical rigor:** Precise definitions and logically sound reasoning
- Benchmark relevance:** Tests important mathematical reasoning skills

Presentation Quality

- Complete solution:** Author provides correct, detailed solution with clear reasoning steps
- Proper formatting:** Good use of LaTeX and clear mathematical presentation
- Professional language:** Grammar, spelling, and mathematical terminology
- Appropriate metadata:** Accurate difficulty ratings and relevant tags

⚠ Common Issues to Look For

Content Issues

- Ambiguous or unclear problem statements
- Missing or incorrect mathematical notation
- Solution errors or incomplete reasoning
- Inappropriate difficulty rating for the content
- Questions that are too easy or impossibly hard

Presentation Issues

- Typos, grammatical errors, or unclear wording
- Poor LaTeX formatting or rendering issues
- Missing tags or inappropriate categorization
- Inconsistent mathematical notation
- Unprofessional language or tone

✓ Review Decision Guidelines

Accept
Accept for Benchmark
Ready for benchmark inclusion with minimal or no changes. High quality content and presentation.

Needs Revision
Needs Revision
Good question but requires specific improvements. Provide clear, actionable feedback.

Not Suitable
Not Suitable
Fundamental issues that make it inappropriate for the benchmark. Explain why clearly.

📝 How to Write Constructive Feedback

✓ Good Feedback

- Be specific:** Point out exact locations of issues
- Be constructive:** Suggest how to improve, not just what's wrong
- Be respectful:** Professional tone, acknowledge effort
- Be complete:** Address all major issues you notice
- Use examples:** Show corrected notation or phrasing

Example:

"In line 3, the notation $f : X \rightarrow Y$ should be $f : \mathbb{R} \rightarrow \mathbb{R}$ to be more specific about the domain. Consider rephrasing the conclusion to be more precise about the uniqueness condition."

✗ Avoid This

- Vague criticism:** "This is wrong" without explanation
- Personal attacks:** Comments about the author rather than the work
- Overwhelming details:** Listing every minor typo without priorities
- Unhelpful rejection:** "Not suitable" without explaining why
- Contradictory advice:** Conflicting suggestions

Bad Example:

"This question is terrible and has lots of errors. The math is wrong and the formatting is bad. You should rewrite the whole thing."

⚙ Review Process

Expected Timeline

- Review submission:** Within 1-2 weeks of assignment
- Thorough review:** Allow 30-60 minutes per question
- Complex questions:** May require additional time for verification

Anonymity Options

- Named reviews:** Default, promotes accountability
- Anonymous reviews:** Use when concerned about conflicts
- Admin visibility:** Admins can always see reviewer identity

[← Back to Questions](#)
[🔍 Find Questions to Review](#)

IMProofBench © 2025 | Mathematical Reasoning Benchmark for AI Research | Privacy Policy

Figure 19: Detailed explainer of review instructions and process.

IMProofBench Mathematical Reasoning Benchmark

participant@example.com
 [Admin](#)
[Profile](#)
[Sign Out](#)

[Questions](#) / [Counting stable graphs](#) / [Grade Model Answers](#) / Answer B

Answer A T1 ✓
 Answer B T1 ✗
 Answer C T1
 Answer D T1
 Answer E
 [Focus Mode](#)

☒ Grading Notes ⓘ

Question & Solution

Question

Given an integer $g \geq 2$ let N_g be the number of isomorphism classes of stable graphs of genus g with $n = 0$ legs and precisely 3 edges. Here stable graphs are the decorated graphs classifying boundary strata of the moduli space $\overline{\mathcal{M}}_g$ of stable curves.

Give a closed formula for N_g valid for all $g \geq 2$.

Solution

For $g = 2$ we have $N_2 = 2$. For g at least 3, we claim that the number of isomorphism classes of stable graphs of genus g with precisely 3 edges follows a pattern dependent on the value of g modulo 6. More precisely, for $g \geq 3$, the formula for N_g is given by:

$$N_g = \begin{cases} \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - 2 & \text{if } g \equiv 0 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{155}{72} & \text{if } g \equiv 1 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - \frac{20}{9} & \text{if } g \equiv 2 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{19}{8} & \text{if } g \equiv 3 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{5}{12}g - \frac{16}{9} & \text{if } g \equiv 4 \pmod{6} \\ \frac{1}{9}g^3 + \frac{7}{8}g^2 + \frac{1}{6}g - \frac{187}{72} & \text{if } g \equiv 5 \pmod{6} \end{cases}$$

This result can be verified using the `admcycles` package for calculating the numbers N_g in many cases, and calculating the Lagrange interpolation polynomial for different remainder classes modulo 6. Below we provide some code for performing these calculations.

```
from admcycles import *
L = {g : len(list_strata(g,0,3)) for g in range(2,4
R,4g) = PolynomialRing(QQ)
pdict = dict()
for o in range(3,9):
    pdict[o%6] = R.lagrange_polynomial([g,L[g]] for g i
for o in range(6):
    print(o, pdict[o])
```

Output:

Model B Response

$b_1 = E - V + 1 = 4 - V$. The genus formula gives:

$$g = b_1 + \sum_{v \in V} g(v)$$

where $g(v)$ is the genus of vertex v . The stability condition requires $2g(v) - 2 + \text{val}(v) > 0$ for each vertex.

Case-by-Case Enumeration

Case 1: $V = 1$ (one vertex)

- First Betti number: $b_1 = 3$
- The vertex has genus $g = 3$ with valence 6 (three self-loops)
- Stability: $2(g - 3) - 2 + 6 = 2g - 2 > 0$ (always satisfied for $g \geq 2$)
- Requirement: $g - 3 \geq 0$, so this contributes 1 **graph for $g \geq 3$**

Case 2: $V = 2$ (two vertices)

- First Betti number: $b_1 = 2$
- Need $g_1 + g_2 = g - 2$

Subcase 2a: Three parallel edges (valences 3, 3) - Stability for each vertex: $2g_i - 2 + 3 > 0$, requiring $g_i \geq 0$ - Number of non-negative integer solutions to $g_1 + g_2 = g - 2$: **$(g - 1)$ graphs for $g \geq 2$**

Subcase 2b: One self-loop and one edge (valences 4, 2) - Vertex with valence 2: needs $2g - 2 + 2 > 0$, so $g \geq 1$ - Vertex with valence 4: needs $2g - 2 + 4 > 0$, so $g \geq 0$ - Solutions with $g_1 + g_2 = g - 2$ where the valence-2 vertex has genus ≥ 1 : **$(g - 2)$ graphs for $g \geq 3$**

Case 3: $V = 3$ (three vertices)

- First Betti number: $b_1 = 1$
- Forms a triangle with each vertex having valence 2
- Need $g_1 + g_2 + g_3 = g - 1$
- Stability: each vertex needs $2g_i - 2 + 2 > 0$, so $g_i \geq 1$

Grading Panel

Auto-save enabled

AI Mistake Indicators

- Incorrect Logic or Reasoning ⓘ

True False Not Sure
- Hallucinated Results or Literature ⓘ

True False Not Sure
- Calculation Mistakes ⓘ

True False Not Sure
- Conceptual Misunderstanding ⓘ

True False Not Sure

AI Achievement Indicators

- Problem Understanding ⓘ

True False Not Sure
- Correct End Result ⓘ

True False Not Sure
- Insight and Creativity ⓘ

True False Not Sure
- Practical Usefulness ⓘ

True False Not Sure

Overall Progress Grade ⓘ

0	1	2	3
None	Minor	Major	Complete

Comments (Optional) ⓘ

Reasonable calculation strategy but formula for $V=2$ vertices is false already

☐ Flag for organizers ⓘ

[Mark as Complete](#)

IMProofBench © 2025 | Mathematical Reasoning Benchmark for AI Research | Privacy Policy

Figure 20: Grading form, displaying sample solution, model answer, and scoring form side by side. Model identities (A–E) at the top are randomized on starting the grading, and only revealed when grading is complete, to avoid bias.