

# Challenging the Boundaries of Reasoning: An Olympiad-Level Math Benchmark for Large Language Models

Anonymous ACL submission

## Abstract

The rapid advancement of large reasoning models has saturated existing math benchmarks, underscoring the urgent need for more challenging evaluation frameworks. To address this, we introduce **OlymMATH**, a rigorously curated, Olympiad-level math benchmark comprising 350 problems, each with parallel English and Chinese versions. OlymMATH is the first benchmark to unify dual evaluation paradigms within a single suite: (1) *natural language evaluation* through **OlymMATH-EASY** and **OlymMATH-HARD**, comprising 200 computational problems with numerical answers for objective rule-based assessment, and (2) *formal verification* through **OlymMATH-LEAN**, offering 150 problems formalized in Lean 4 for rigorous process-level evaluation. All problems are manually sourced from printed publications to minimize data contamination, verified by experts, and span four core domains. Extensive experiments reveal the benchmark’s significant challenge, and our analysis also uncovers consistent performance gaps between languages and identifies cases where models employ heuristic “guessing” rather than rigorous reasoning. To further support community research, we release 582k+ reasoning trajectories, a visualization tool, and expert solutions.

## 1 Introduction

The advent of large language models (LLMs) (Zhao et al., 2023) has marked a significant leap forward in the capabilities of artificial intelligence, with mathematical reasoning emerging as a pivotal and demanding area of research (DeepSeek-AI et al., 2025a; OpenAI, 2024; Chen et al., 2025). Recently, the evaluation and enhancement of mathematical reasoning abilities have become a central focus in the development of LLMs (Yang et al., 2024).

Effective assessment of LLM reasoning necessitates *reliable* and *verifiable* evaluation benchmarks. Reliability requires accurately designed problems

with unambiguous solutions and minimized data contamination risk, ensuring trustworthy evaluation. For verifiability, two paradigms have emerged: (1) *numerical-answer benchmarks* using rule-based verification (e.g., *sympy*), which offer scalability but cannot assess reasoning quality; and (2) *formal proof benchmarks* using theorem provers (e.g., Lean, Isabelle), which provide rigorous process-level verification but require specialized formalization. An ideal benchmark suite should leverage both paradigms for comprehensive evaluation.

However, existing benchmarks face limitations in both paradigms. For numerical-answer benchmarks, early datasets are largely saturated by leading models, while more challenging ones either suffer from small scale compromising statistical reliability, risk data leakage via web crawling, or rely on LLM-as-judge reducing dependability. For formal proof benchmarks, existing datasets typically source from well-known competitions with high online exposure, facing similar contamination risks. Moreover, current benchmarks predominantly focus on English, neglecting multilingual reasoning.

To bridge the gap, we present **OlymMATH**, a rigorously curated, bilingual (English and Chinese) benchmark for Olympiad-level reasoning, comprising 350 unique problems organized into three non-overlapping subsets: **OlymMATH-EASY** and **OlymMATH-HARD** contain 100 computational problems each, split into *easy* and *hard* levels with parallel bilingual versions, requiring precise numerical answers for reliable and rule-based *sympy* verification. Additionally, **OlymMATH-LEAN** provides a separate set of 150 problems formalized in Lean 4, accompanied by bilingual natural language statements and solutions, enabling rigorous evaluation of automated theorem proving capabilities. Unlike proof-based benchmarks that rely on unreliable LLM-as-a-judge evaluation, OlymMATH-LEAN leverages the Lean language for fully automated and mathematically rigorous

## OlymMATH-EASY / HARD

**Problem-EN:** Find the remainder of  $\sum_{k=0}^{1234} \binom{2016 \times 1234}{2016k}$  modulo  $2017^2$  (provide the value in the range  $[0, 2017^2)$ ).  
**Answer:** 1581330. **Subject:** Number Theory.

## OlymMATH-LEAN

**Subject:** Number Theory. **Formal Statement:**  
theorem to\_prove  
(n : Nat) (p : Nat) (hp : p.Prime)  
(hdiv : p | 2^n + 1) : p % 8 != 7 := by sorry

Figure 1: Examples from our OlymMATH dataset.

formal verification. Meanwhile, to prevent data leakage, problems were manually sourced from printed publications and verified by experts. The benchmark covers four major mathematical fields and adheres to the MATH or miniF2F dataset format for compatibility (see Figure 1).

By leveraging OlymMATH, we conduct extensive experiments to evaluate the performance of state-of-the-art models. The results underscore our benchmark’s difficulty, with advanced models like DeepSeek-R1, o3-mini, and Gemini 2.5 Pro Exp achieving only 19.5%, 31.2%, and 58.4% accuracy, respectively, on OlymMATH-HARD (EN), indicating Olympiad-level math remains a significant challenge necessitating further research. Our bilingual comparison showed a consistent performance gap, with higher accuracy on English problems versus Chinese, highlighting the need for multilingual evaluation. Furthermore, case studies revealed models sometimes use heuristic “guessing” to reach answers without rigorous proofs. This underscores the importance of process-level inspection for accurate LLM capability assessment.

In summary, our contributions are as follows.

- We introduce **OlymMATH**, the first Olympiad-level mathematical benchmark that unifies natural language problems and formal theorem proving within a single bilingual suite. OlymMATH comprises 350 unique problems, each available in both English and Chinese: **OlymMATH-EASY** and **OlymMATH-HARD** provide 200 computational problems with sympy-verifiable numerical answers, while **OlymMATH-LEAN** offers 150 problems formalized in Lean 4 for process-level verification—bridging the gap between outcome-based and reasoning-based evaluation.

- Extensive experiments validate OlymMATH’s reliability and strong discriminative power, while revealing critical model limitations including EN-

ZH performance gaps and heuristic “guessing” that bypasses rigorous reasoning.

- We open-source 582,400 reasoning trajectories from 28 models, a visualization tool, and expert solutions to facilitate community research.

## 2 Related Work

**Natural Language Math Benchmarks.** The first paradigm of math reasoning evaluation relies on numerical-answer benchmarks with rule-based verification due to their simplicity and scalability. Early benchmarks such as GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) have been pivotal in advancing LLM reasoning capabilities (Fang et al., 2024; Arora et al., 2023), but are now largely saturated by slow-thinking models enhanced by long chain-of-thought fine-tuning (Min et al., 2024) or reinforcement learning scaling (DeepSeek-AI et al., 2025a).

More challenging benchmarks face different limitations. The AIME dataset offers increased difficulty but suffers from small scale (e.g., 30 problems from AIME 2025), compromising statistical reliability—a single problem shift can change accuracy by 3.33% (Hochlehnert et al., 2025), with binomial standard errors approximately  $2.6\times$  larger than those from a 200-problem benchmark. Moreover, rapidly improving models are approaching its measurement ceiling (e.g., Gemini 2.5 Pro achieving 92% Pass@1 on AIME 2024), and its English-only focus neglects multilingual evaluation. OlympiadBench (He et al., 2024) provides a larger collection of problems, but its overall difficulty remains limited—even a 1.5B model (DeepScaleR-Preview (Luo et al., 2025)) achieves 50.0% accuracy, indicating insufficient challenge for evaluating state-of-the-art reasoning models. Omni-MATH (Gao et al., 2024) increases problem count via web crawling from AoPS, elevating data leakage risks, while its prevalent proof-based problems require LLM-as-judge evaluation rather than rule-based verification, reducing dependability. PolyMath (Wang et al., 2025b) directly sources from widely publicized competitions (e.g., AIME, CNMO, IMO) and existing datasets (e.g., MGSM (Shi et al., 2022), P-MMEval (Zhang et al., 2025), HLE (Phan et al., 2025)), risking high data leakage. AMO Bench (An et al., 2025) targets advanced Olympiad problems but has limited scale.

In contrast, OlymMATH provides 200 challenging Olympiad-level problems with lower contami-

(a) Natural Language Benchmarks			
Name	# Prob. (# Lang.)	Eval.	Difficulty
AIME 24,25	30 (1)	Rule	Olympiad
HMMT	30 (1)	Rule	Olympiad
USAMO 2025	6 (1)	LLM	Olympiad
OlympiadBench	2133 (2)	Rule	Olympiad
Omni-MATH	4428 (1)	LLM	Olympiad
PolyMath	500 (18)	Rule	Olympiad
AMO Bench	50 (1)	Rule & LLM	Olympiad
<b>EASY (Ours)</b>	100 (2)	Rule	Olympiad
<b>HARD (Ours)</b>	100 (2)	Rule	Olympiad

(b) Formal Language Benchmarks			
Name	# Prob.	Lang.	Difficulty
miniF2F	488	EN	Olympiad
ProofNet	371	EN	Undergrad
FIMO	149	EN	Olympiad
PutnamBench	640	EN	Undergrad Comp.
<b>LEAN (Ours)</b>	150	EN & ZH	Olympiad

Table 1: Comparison of existing benchmarks. EN and ZH denote English and Chinese, respectively.

nation risk through manual curation from printed publications, larger scale for statistical reliability, and bilingual versions for thorough evaluation.

**Formal Language Math Benchmarks.** Beyond assessing final answers, understanding *how* models arrive at solutions—distinguishing rigorous derivation from heuristic shortcuts—is equally important. Formal theorem proving benchmarks address this by requiring machine-checkable proofs. miniF2F (Zheng et al., 2022) provides Olympiad-level problems formalized in multiple proof assistants, while ProofNet (Azerbayev et al., 2023) focuses on undergraduate mathematics. FIMO (Liu et al., 2023) and PutnamBench (Tsoukalas et al., 2024) target competition mathematics at different levels. However, existing formal benchmarks typically source from well-known competitions (*e.g.*, IMO shortlist for FIMO, Putnam for PutnamBench) with high online exposure, facing similar contamination risks. Moreover, all existing formal benchmarks are English-only. OlymMATH-LEAN addresses these gaps with bilingual Lean 4 formalizations sourced from printed publications.

### 3 Benchmark Construction

In this section, we describe OlymMATH in detail, including its construction methodology, problem

composition, categorical distribution, and evaluation approach. Table 1 presents a comparison with existing mathematical reasoning benchmarks. Existing benchmarks typically focus on either *natural language problems* with numerical answers or *formal theorem proving*, but not both. OlymMATH bridges this gap by being the first Olympiad-level benchmark to integrate both paradigms within a unified bilingual framework: OlymMATH-EASY and OlymMATH-HARD provide 200 natural language problems requiring precise numerical answers for scalable rule-based verification, while OlymMATH-LEAN offers 150 problems with Lean 4 formalizations enabling rigorous process-level verification. This dual-paradigm design allows comprehensive assessment of both outcome correctness and reasoning rigor, addressing the limitations of relying on either paradigm alone.

#### 3.1 Contamination Analysis & Verification

**Contamination Analysis** Constructing a reliable benchmark requires mitigating data contamination. OlymMATH comprises 350 problems curated from printed resources (specialized magazines and textbooks), intentionally excluding online repositories to minimize prior digital exposure, unlike existing benchmarks drawing from well-known competitions (*e.g.*, FIMO using IMO shortlist, PutnamBench using Putnam, Omni-MATH using AoPS).

For quantitative leakage analysis, we followed Omni-MATH, using  $n$ -gram accuracy metric (Xu et al., 2024): for each sample, the problem and answer are concatenated; 5 starting points are uniformly sampled; and the model’s ability to predict the subsequent 5-gram is evaluated. Leakage risk is quantified by comparing  $n$ -gram accuracy on the original dataset against 3 LLM-rewritten versions (Gemini 2.5 Flash Preview Thinking (Google, 2025b)), with the normalized difference  $\delta$  indicating model familiarity with original versus rewritten data. Since  $\delta$ ’s absolute value depends on the rewriting LLM, assessing leakage risk requires *relative comparison* of  $\delta$  between benchmarks. Results in Table 2 show lower contamination risk for OlymMATH than PolyMath, establishing OlymMATH as a more reliable benchmark for evaluating LLMs’ true mathematical capabilities.

**Verification** To enhance dataset reliability, we invited a China Mathematical Olympiad silver medalist and two provincial first-prize winners to verify and revise the problems and solutions in

Model (Base)	Lang.	PolyMath	OlymMATH
InternLM2-Math-7B (Ying et al., 2024)	EN	34.84%	<b>0.90%</b>
	ZH	12.29%	<b>0.88%</b>
Qwen2.5-7B (Qwen et al., 2025)	EN	38.81%	<b>17.59%</b>
	ZH	10.27%	<b>3.42%</b>

Table 2: Results of leakage analysis. The lower value is bolded. OlymMATH exhibits lower  $\delta$  values than PolyMath per language, indicating a lower leakage risk.

OlymMATH-EASY and HARD. Since the answers to the problems were already provided, the verification difficulty was reduced, making the expertise of reviewers sufficient for this task. Each problem was reviewed by at least two reviewers. Additionally, we have published official solutions for challenging problems for community oversight.

For OlymMATH-LEAN, we leverage the Lean server for automatic verification. Raw problems and solutions are first cleaned by Claude Opus 4.5 (Anthropic, 2025b) for  $\text{\LaTeX}$  format correction, then undergo three independent verification rounds using DeepSeek V3.2 Speciale (DeepSeek-AI et al., 2025b) to check translation accuracy, statement precision, and solution rigor. A Claude Opus 4.5-based agent (see Appendix A.2 for details) then iteratively interacts with a Kimina Lean REPL server (Santos et al., 2025) in an isolated sandbox, refining code based on compiler feedback until successful compilation. Compiled formalizations are validated by three independent Gemini 3.0 Flash (Google, 2025a) calls for semantic alignment, and formalizations containing axiom declarations receive additional human expert reviews.

### 3.2 Problem Categories and Distribution

OlymMATH problems span four key high-school Olympiad mathematical fields—algebra, geometry, number theory, and combinatorics—classified by human experts for reliability. Problems are selected for their challenge, suitability for simple-answer verification, and topic diversity (e.g., inequalities, sequences, and more in algebra). Figure-based problems within this set are text-reformulated for LLM compatibility, with non-convertible ones excluded (e.g., Figure 5 in Appendix).

For refined evaluation, computational problems are categorized by difficulty: *easy*, designed to challenge standard prompting in mainstream models, and *hard*, tailored to test advanced reasoning (e.g., slow-thinking modes) in state-of-the-art models. Additionally, OlymMATH-LEAN provides

Category	# Problems		
	EASY	HARD	LEAN
Algebra (Alg.) <i>Inequality, Trigonometry, etc.</i>	25	25	79
Geometry (Geo.) <i>Solid &amp; Analytic Geometry, etc.</i>	33	25	15
Number Theory (Num.) <i>Diophantine Equation, etc.</i>	13	25	42
Combinatorics (Com.) <i>Graph Theory, Permutation, etc.</i>	29	25	14
Total	100	100	150

Table 3: The distribution of OlymMATH by category.

150 problems with Lean 4 (Mathlib v4.24.0) formalizations for process-level verification. The distribution details are described in Table 3.

### 3.3 Format and Verification Methodology

OlymMATH adopts MATH and miniF2F dataset format (see Figure 1) for seamless integration with existing pipelines and enhancing clarity and processing efficiency. All problems are text-based, including geometry problems reformulated from diagrams to align with LLM evaluation, as mentioned previously. For consistent, objective assessment, answers to computational problems are restricted to real numbers and intervals, e.g., “ $[\sqrt{33}, +\infty)$ ”, while excluding ambiguous formats such as set operations, variables, complex numbers, and texts (see Table 11 in Appendix for details). This design enables reliable sympy-based and formal Lean server verification.

To make the evaluation more challenging, OlymMATH includes problems with multiple numerical answers. These problems are modified to require a summary of all potential outcomes (e.g., sums, sums of squares; see Figure 6 in Appendix). This method effectively assesses whether models can consider all possible answers, thereby providing a robust evaluation of their reasoning capabilities.

### 3.4 Bilingual Extension

Originating from Chinese-language problems, OlymMATH provides both original Chinese and translated English versions for bilingual evaluation. Our translation pipeline employs Claude Sonnet 3.7 (Anthropic, 2025a) for initial translation, iterative refinement with GPT-4o (OpenAI et al., 2024), and human verification by two expert annotators to ensure mathematical accuracy and linguistic fluency. OlymMATH-LEAN similarly provides bilingual natural language statements alongside Lean

327 formalizations, supporting research in multilingual  
328 reasoning and informal-formal translation.

## 329 4 Experiments

330 In this section, we assess the performance of lead-  
331 ing reasoning models using OlymMATH and pro-  
332 vide a detailed analysis of their capabilities.

### 333 4.1 Natural Language: EASY & HARD Subset

334 We first evaluate models on the natural language  
335 subsets, where problems require numerical answers  
336 verified via rule-based matching.

#### 337 4.1.1 Experimental Setup

338 **Models.** We assess representative LLMs for  
339 a thorough evaluation. For open-source mod-  
340 els, we investigated recent work on reasoning  
341 models, and evaluated DeepSeek-R1 (DeepSeek-  
342 AI et al., 2025a), STILL-3-Preview (Team,  
343 2025d), DeepScaleR-Preview (Luo et al., 2025),  
344 QwQ (Team, 2025c), Light-R1 (Wen et al.,  
345 2025), OpenThinker2 (Team, 2025a), Skywork-  
346 OR1 (He et al., 2025), GLM-Z1-Air (GLM  
347 et al., 2024), AceMath-RL (Liu et al., 2025),  
348 OpenMath-Nemotron (Moshkov et al., 2025), and  
349 Qwen3 (Team, 2025b). For closed-source models,  
350 we evaluate o3-mini (high) (OpenAI, 2025) and  
351 Gemini 2.5 Pro Exp 0325 (Google, 2025).

352 **Evaluation Details.** Our evaluation pipeline gen-  
353 erates 64 responses per problem for each model, ex-  
354 cept for resource-intensive models (*i.e.*, OpenMath-  
355 Nemotron-32B, Qwen3-235B-A22B, GLM-Z1-  
356 Air, DeepSeek-R1, o3-mini (high), and Gemini  
357 2.5 Pro Exp), which are limited to 8 samples due  
358 to resource limitations and the relatively large  
359 scale of our dataset. For Pass@1, we compute  
360 mean accuracy across all sampled responses; for  
361 Cons@64 and Cons@8, we apply majority voting to  
362 determine a consensus answer per problem. Fol-  
363 lowing established practices (DeepSeek-AI et al.,  
364 2025a; Team, 2025c), locally-evaluated models use  
365 temperature = 0.6, top\_p = 0.95, min\_p = 0, and  
366 max\_tokens = 32768, while API-evaluated models  
367 (*i.e.*, GLM-Z1-Air, DeepSeek-R1, o3-mini (high),  
368 and Gemini 2.5 Pro Exp) use maximum available  
369 max\_tokens to fully leverage their reasoning capa-  
370 bilities. We open-source all 582,400 samples, an  
371 online visualization tool, and standard solutions for  
372 challenging problems to support community analy-  
373 sis of LLM reasoning patterns and characteristics  
374 (see Appendix A.1 for further information).

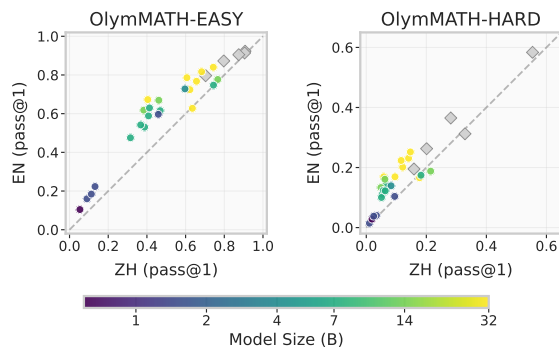


Figure 2: Pass@1 on OlymMATH EN (y) vs. ZH (x), the dashed line shows parity. Points above favor EN, below favor ZH. Solid circles (local dense models, colored by size) indicate larger models trend towards higher accuracy. Diamonds are MoE or closed-source models.

#### 375 4.1.2 Evaluation Results

376 In this part, we present the evaluation results of  
377 OlymMATH (EN) and OlymMATH (ZH) in Ta-  
378 ble 4. Due to space constraints, we include only  
379 representative models in the main text, and full re-  
380 sults are provided in Tables 9 and 10 in Appendix.

381 First, we observe that all tested models exhibit  
382 relatively poor performance, with even OpenAI  
383 o3-mini (high) and Gemini 2.5 Pro Exp achiev-  
384 ing only 31.2% and 58.4% on OlymMATH-HARD  
385 (EN). This underscores the high overall difficulty of  
386 our benchmark, which demands stronger reasoning  
387 abilities and a deeper understanding of mathemat-  
388 ical knowledge. In contrast, the performance of  
389 these advanced reasoning models on OlymMATH-  
390 EASY (EN) is more modest and comparable to  
391 that on AIME 2024, suggesting that OlymMATH-  
392 EASY is well-suited for evaluating the capabilities  
393 of less advanced reasoning models.

394 Second, by comparing the performance of LLMs  
395 on bilingual versions of OlymMATH, we find that  
396 language can influence the reasoning performance  
397 of LLMs to some extent (see Figure 2). Overall,  
398 all models tend to achieve higher performance on  
399 the English benchmarks. A potential reason for  
400 this is that English corpora still dominate existing  
401 pre-training datasets, making the English-based  
402 task-solving capabilities of LLMs generally supe-  
403 rior compared to other languages. This finding  
404 highlights the importance of considering perfor-  
405 mance across different languages when conducting  
406 a comprehensive evaluation of LLMs.

407 Third, to provide insights into model robustness  
408 beyond Pass@1, we report Pass@k for DeepSeek-

Model	OlymMATH-HARD										OlymMATH-EASY									
	Alg.		Geo.		Num.		Com.		Avg.		Alg.		Geo.		Num.		Com.		Avg.	
	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k
<i>English (EN)</i>																				
Qwen3 (0.6B, Think)	2.5	0.0	2.1	4.0	6.6	8.0	0.2	0.0	2.8	3.0	15.5	20.0	5.6	15.2	24.5	38.5	5.2	6.9	10.4	17.0
DS-R1-Distill (1.5B)	1.9	0.0	1.8	0.0	1.8	0.0	0.4	0.0	1.5	0.0	20.8	40.0	12.6	21.2	32.6	61.5	8.2	24.1	16.0	32.0
Qwen3 (4B, Think)	18.1	20.0	14.8	12.0	19.8	28.0	3.1	4.0	13.9	16.0	76.4	92.0	79.1	97.0	85.1	84.6	57.1	72.4	72.8	87.0
DS-R1-Distill (7B)	15.6	36.0	12.6	24.0	13.1	24.0	3.1	4.0	11.1	22.0	52.8	84.0	49.6	84.8	62.5	84.6	33.9	58.6	47.5	77.0
Qwen3 (30B-A3B, Think)	38.8	44.0	33.8	44.0	26.7	36.0	5.9	4.0	26.3	32.0	91.4	100.0	92.9	100.0	90.9	92.3	75.6	93.1	87.2	97.0
DS-R1-Distill (32B)	22.4	32.0	21.4	24.0	20.3	40.0	3.4	4.0	16.9	25.0	73.6	100.0	71.8	97.0	84.5	92.3	49.0	69.0	67.3	89.0
QwQ (32B)	32.9	28.0	26.6	36.0	26.7	44.0	6.2	4.0	23.1	28.0	91.8	100.0	87.0	100.0	95.0	100.0	69.0	89.7	84.0	97.0
Qwen3 (235B-A22B, Think)	48.0	52.0	49.5	60.0	38.0	36.0	10.5	16.0	36.5	41.0	93.5	100.0	92.4	100.0	99.0	100.0	81.9	93.1	90.5	98.0
DeepSeek R1	30.0	40.0	25.5	32.0	18.5	24.0	4.0	4.0	19.5	25.0	90.5	100.0	82.2	97.0	94.2	100.0	60.8	72.4	79.6	91.0
OpenAI o3-mini (high)	29.5	32.0	29.0	44.0	49.5	60.0	17.0	20.0	31.2	39.0	93.0	92.0	89.8	100.0	97.1	100.0	89.2	96.6	91.4	<b>97.0</b>
Gemini 2.5 Pro Exp 0325	71.5	76.0	75.5	84.0	59.0	72.0	27.5	36.0	<b>58.4</b>	<b>67.0</b>	92.0	100.0	97.0	100.0	98.1	100.0	84.5	89.7	<b>92.2</b>	<b>97.0</b>
<i>Chinese (ZH)</i>																				
Qwen3 (0.6B, Think)	2.6	4.0	0.8	0.0	4.4	4.0	0.0	0.0	1.9	2.0	9.9	8.0	2.8	3.0	12.0	15.4	1.3	3.4	5.4	6.0
DS-R1-Distill (1.5B)	1.8	0.0	1.3	0.0	1.1	0.0	0.0	0.0	1.0	0.0	13.7	20.0	6.3	9.1	20.9	30.8	2.6	0.0	9.0	12.0
Qwen3 (4B, Think)	12.5	20.0	7.0	8.0	12.6	24.0	0.9	0.0	8.3	13.0	70.8	88.0	61.0	75.8	74.8	92.3	41.8	51.7	59.7	74.0
DS-R1-Distill (7B)	6.1	8.0	7.9	12.0	6.6	8.0	0.6	0.0	5.3	7.0	38.0	64.0	30.8	51.5	49.2	61.5	18.7	27.6	31.5	49.0
Qwen3 (30B-A3B, Think)	35.6	40.0	24.1	28.0	18.1	24.0	2.7	4.0	20.1	24.0	87.8	92.0	84.7	97.0	91.3	100.0	61.9	65.5	79.7	87.0
DS-R1-Distill (32B)	6.5	0.0	5.4	4.0	10.6	12.0	0.7	0.0	5.8	4.0	45.2	52.0	41.8	63.6	60.2	69.2	26.0	37.9	40.4	54.0
QwQ (32B)	20.9	24.0	15.9	16.0	17.6	24.0	2.0	0.0	14.1	16.0	85.4	96.0	76.6	97.0	92.9	100.0	53.8	69.0	74.3	89.0
Qwen3 (235B-A22B, Think)	36.5	48.0	43.5	48.0	28.5	32.0	4.0	8.0	28.1	34.0	91.0	100.0	90.2	97.0	94.2	100.0	78.4	89.7	87.5	96.0
DeepSeek R1	20.0	24.0	25.0	28.0	17.0	16.0	1.5	0.0	15.9	17.0	79.5	96.0	74.6	84.8	88.5	92.3	49.6	55.2	70.4	80.0
OpenAI o3-mini (high)	31.5	40.0	32.5	44.0	48.5	56.0	19.0	28.0	32.9	42.0	93.0	96.0	89.4	100.0	99.0	100.0	85.8	93.1	90.5	<b>97.0</b>
Gemini 2.5 Pro Exp 0325	65.0	76.0	78.0	80.0	53.5	56.0	25.0	40.0	<b>55.4</b>	<b>63.0</b>	90.5	96.0	93.2	93.9	100.0	100.0	84.1	86.2	<b>90.8</b>	93.0

Table 4: Model performance on OlymMATH sorted by model size. The abbreviations “Alg.,” “Geo.,” etc. represent the four categories in OlymMATH. Models sampled only 8 times are marked in gray to indicate potential instability. For brevity, only representative models are shown; see Table 9 and Table 10 in Appendix for complete results.

Model	Subset	P@1	P@4	P@16	P@64	C@64	Model	AIME	OBench	Omni	EASY	HARD
1.5B	EASY	16.0	37.5	62.2	78.0	32.0	STILL-3-Pre. (1.5B)	32.5	45.4	-	18.4	3.8
	HARD	1.5	5.1	14.2	30.0	0.0	DScaleR-Pre. (1.5B)	43.1	50.0	-	22.3	4.1
7B	EASY	47.5	78.4	91.8	97.0	77.0	GPT-4o	13.1	41.5	30.5	-	-
	HARD	11.1	29.6	53.4	74.0	22.0	o1-mini	63.6	-	60.5	-	-
32B	EASY	67.3	90.8	97.4	100.0	89.0	QwQ (32B)	79.5	-	65.2	84.0	23.1
	HARD	16.9	38.7	59.0	75.0	25.0	DeepSeek R1	79.8	-	67.3	79.6	19.5
							GLM-Z1-Air (32B)	80.8	-	68.4	76.8	20.1
							o3-mini (high)	87.3	-	-	91.4	31.2
							Gemini 2.5 Pro Exp	92.0	-	-	92.2	58.4

Table 5: Pass@k and Cons@64 for DS-R1-Distill series on OlymMATH-EASY and HARD (in English).

R1-Distill-Qwen series in Table 5. The results reveal substantial gains from increased sampling: 7B model improves from 11.1% (Pass@1) to 74.0% (Pass@64) on EN-HARD, indicating that correct solutions exist within the model’s capability but require multiple attempts to surface. However, the gap between Pass@64 and Cons@64 (74.0% vs. 22.0%) suggests significant inconsistency—models can solve problems but often fail to do so reliably. These results support the use of Pass@k as a more comprehensive metric for probing the reasoning capability boundaries of LLMs (Yue et al., 2025).

Table 6: Cross-benchmark Pass@1 comparison. “-” indicates no publicly available data. AIME denotes AIME 2024. OBench denotes OlympiadBench. Omni denotes Omni-MATH. EASY and HARD denotes our OlymMATH subsets. DScaleR denotes DeepScaleR.

### 4.1.3 Benchmark Comparison

To comprehensively evaluate OlymMATH against existing benchmarks, we compare model performances across widely used mathematical benchmarks. We collected results from official repositories, as shown in Table 6. These results reveal the difficulty hierarchy: HARD  $\gg$  EASY  $\approx$  AIME24  $>$

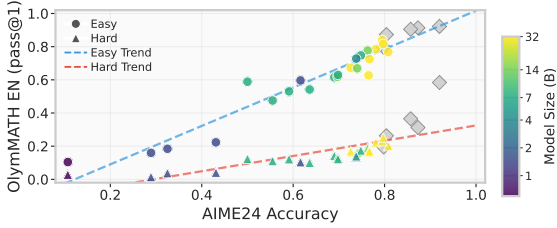


Figure 3: Correlation of Pass@1 metric: OlymMATH (EN) vs. AIME24. Dashed lines indicate linear trends per dataset. Solid circles (local dense models, colored by size) indicate larger models trend towards higher accuracy. Diamonds are MoE or closed-source models.

OlympiadBench, while Omni-MATH spans from OlympiadBench level to slightly above AIME but remains considerably easier than OlymMATH-HARD. OlymMATH-EASY validates our design as an extended bilingual version of AIME—models like DeepSeek-R1 achieve nearly identical scores on both (79.8% vs. 79.6%), confirming comparable difficulty levels. In contrast, OlymMATH-HARD presents substantially greater challenges: even Gemini 2.5 Pro Exp and o3-mini (high), which exceed 87% on AIME24, only attain 58.4% and 31.2% respectively on our HARD subset. This divergence is particularly striking given their similar AIME24 performance (92.0% vs. 87.3%), demonstrating OlymMATH-HARD’s superior discriminative power for differentiating state-of-the-art capabilities. Furthermore, with 100 problems per difficulty level compared to AIME’s 30 problems, OlymMATH provides more stable performance measurements—addressing the statistical reliability concerns inherent in smaller-scale benchmarks.

Figure 3 further validates OlymMATH’s reliability by comparing against AIME24. The close clustering around linear trend lines indicates consistent relative model rankings across both benchmarks, suggesting OlymMATH measures similar reasoning abilities. Despite this correlation, OlymMATH, particularly the HARD subset, remains significantly more challenging, reinforcing its superior discriminative power for state-of-the-art models.

#### 4.1.4 Analysis of Reasoning Patterns

During our data collection and preliminary experiments, we empirically observed that LLMs sometimes resort to *empirical guesses*—such as heuristics, symmetry assumptions, or even fabrication—rather than rigorous reasoning. For instance, in an optimization problem, o3-mini (high) merely

assumed two sides are equal ( $b = c$ ) based on symmetry, without proving this yields the optimum (see Figure 7 in Appendix). While such intuitive approaches might yield correct answers, they lack logical rigor and this becomes problematic when employing rule-based or LLM-as-judge methods, as neither can effectively assess the quality of rigorous reasoning, thus potentially leading to an illusory improvement in accuracy via “shortcuts”.

Similar issues were observed in AIME and Omni-MATH (see Figures 8 and 9 in Appendix), indicating that despite performance gains, LLMs exhibit deficiencies in deliberative thinking. This underscores the importance of process-level supervision, though its scalability remains a challenge.

Notably, these guessing strategies often fail on our OlymMATH dataset. For example, a model incorrectly assumed symmetry for a complex optimization problem in OlymMATH-HARD, yielding 3081 instead of the correct 2625 (see Figure 10 in Appendix). OlymMATH problems, particularly in the HARD subset, are selected and designed so that their reasoning steps are difficult to “hack” through empirical guessing, thus providing a more robust evaluation of genuine reasoning capabilities.

## 4.2 Formal Language: LEAN Subset

As discussed in Section 4.1.4, the “guessing” phenomenon in natural language evaluation highlights the need for process-level verification. To address this limitation, OlymMATH-LEAN provides a complementary evaluation paradigm that requires machine-verifiable Lean 4 proofs rather than numerical answers. Unlike rule-based verification that only checks final outputs, formal theorem proving enforces rigorous step-by-step reasoning—heuristic shortcuts or symmetry assumptions that lack logical justification will fail to compile. While we do not accurately measure the proportion of “guesses” in natural language benchmarks, OlymMATH-LEAN offers a principled approach to detecting such reasoning gaps.

### 4.2.1 Experimental Setup

**Models.** We evaluate three state-of-the-art theorem proving models, including Kimina Prover (Kimina, Wang et al. (2025a)), DeepSeek Prover V2 (DS V2, Ren et al. (2025)), and Goedel Prover V2 (Goedel V2, Lin et al. (2025)). For each model, we employ the default prompt templates provided in their respective official repositories. Generation hyperparameters are set following the rec-

Metrics	Kimina (8B)	DS V2 (7B)	Goedel V2 (8B)
P@1	4.33 2.8/21.5/2.5/0.2	<b>6.40</b> 3.9/36.9/2.4/0.0	5.29 2.2/34.6/2.5/0.0
P@2	5.90 3.9/30.3/2.7/0.4	<b>7.36</b> 4.6/42.7/2.4/0.0	6.65 3.4/41.3/2.5/0.0
P@4	7.50 5.1/38.9/3.0/0.9	<b>8.08</b> 5.0/48.0/2.4/0.0	7.81 4.7/46.0/2.7/0.0
P@8	<b>9.12</b> 6.5/45.2/3.6/1.8	8.49 5.1/51.6/2.4/0.0	8.58 5.5/48.3/3.0/0.0
P@16	<b>11.10</b> 8.5/49.6/4.8/3.6	8.65 5.1/53.2/2.4/0.0	9.24 6.2/50.0/3.6/0.0
P@32	<b>14.00</b> 11.4/53.3/7.1/7.1	8.67 5.1/53.3/2.4/0.0	10.00 6.3/53.3/4.8/0.0
ref.	78.3	75.6	<b>84.6</b>

Table 7: OlymMATH-LEAN evaluation results. We report Pass@k on OlymMATH-LEAN and Pass@32 on miniF2F for comparison. Numbers below each main score represent the metric in Algebra, Geometry, Number Theory, and Combinatorics, respectively. Bold indicates highest per metric. ref. denotes miniF2F Pass@32.

ommended configurations: temperature = 0.6, top\_p = 0.95, and max\_tokens = 32768.

**Evaluation Details.** For each problem, we sample 32 proof attempts and report Pass@k. A proof is considered successful only if it compiles without errors in Lean 4 and correctly proves the theorem.

#### 4.2.2 Evaluation Results

In this part, we present the evaluation results of OlymMATH-LEAN in Table 7 and a detailed error analysis in Table 8, respectively.

First, all three models achieve relatively low scores on OlymMATH-LEAN (around 10%) compared to their performance on miniF2F (around 80%), highlighting the challenging nature of our benchmark. DeepSeek Prover V2 7B achieves the highest Pass@1 of 6.40%, while Kimina Prover 8B demonstrates stronger performance at higher sampling budgets, reaching the best Pass@32 of 14.00%. Across all models, geometry problems exhibit significantly higher success rates compared to other subjects, likely because many geometry problems can be solved through algebraic manipulation. In contrast, combinatorics proves to be the most challenging category, with DeepSeek Prover V2 7B and Goedel Prover V2 8B achieving 0% success rate across all Pass@k metrics.

Second, Table 8 reveals distinct error patterns across models. A significant portion of errors stem from extraction failures, where models fail to produce properly formatted `lean4` code

Error Type	Kimina (8B)	DS V2 (7B)	Goedel V2 (8B)
Valid	4.3%	6.4%	5.3%
Sorry	8.0%	0.1%	4.4%
Compile	8.4%	43.9%	24.5%
Logic	17.2%	44.9%	22.5%
Server	0.1%	4.5%	0.1%
Extract	62.0%	0.3%	43.3%

Table 8: Error distribution on OlymMATH-LEAN. We report the percentage of each error type across 4800 responses (150 × 32 samples) per model. **Valid** indicates successful proofs; **Sorry** indicates incomplete proofs using *sorry*; **Compile** indicates syntax or type errors, such as missing imports, type mismatches, or unknown identifiers; **Logic** indicates tactic failures or unsolved goals; **Server** indicates server errors; **Extract** indicates failure to extract code blocks from response.

blocks. Kimina Prover 8B exhibits the highest extraction error rate (62.0%), mainly caused by reaching the max\_tokens in generation. Among successfully extracted code, the success rate of compilation ranges from 51.5% to 77.7%, indicating that models still struggle with Lean grammar. Additionally, Kimina Prover 8B exhibits the highest sorry rate (8.0%), suggesting a tendency to generate incomplete proofs with placeholder tactics.

Third, DeepSeek Prover V2 has a 4.5% server error rate, with 80.4% involving the computationally expensive “exact?” tactic. Overall, these results demonstrate that OlymMATH presents substantial challenges across both evaluation paradigms.

## 5 Conclusion

We introduced **OlymMATH**, the first Olympiad-level math benchmark that unifies natural language evaluation and formal theorem proving within a single bilingual suite. The benchmark comprises 350 problems (each available in both English and Chinese): EASY and HARD subsets with sympy-verifiable numerical answers for scalable outcome evaluation, and LEAN subset with Lean formalizations for rigorous process-level verification. Extensive experiments reveal substantial challenges for state-of-the-art models, consistent cross-lingual performance gaps, and heuristic “guessing” behaviors that bypass rigorous reasoning—underscoring the value of our dual-paradigm approach. By releasing 582k+ reasoning trajectories alongside visualization tools and expert solutions, we aim to advance mathematical reasoning research and push the boundaries of language intelligence.

## 578 Limitations

579 Our work has several limitations that suggest direc-  
580 tions for future research. First, while we provide  
581 bilingual evaluation covering both English and Chi-  
582 nese, the reasoning capabilities of LLMs in other  
583 languages remain unexplored. Extending Olym-  
584 MATH to additional languages would enable more  
585 comprehensive assessment of multilingual mathe-  
586 matical reasoning. Second, our current benchmark  
587 focuses exclusively on text-based problems, with  
588 geometry problems reformulated into natural lan-  
589 guage descriptions. Incorporating problems that  
590 retain original diagrams and figures would enable  
591 evaluation of multimodal vision-language models,  
592 offering a more complete picture of mathematical  
593 reasoning capabilities across different input modal-  
594 ities. Third, although we identify “guessing” be-  
595 haviors through qualitative case studies, precisely  
596 quantifying the proportion of such heuristic short-  
597 cuts in natural language evaluation remains an open  
598 challenge. While OlymMATH-LEAN provides rig-  
599 orous process-level verification through formal the-  
600 orem proving, developing scalable metrics to detect  
601 reasoning shortcuts in natural language settings is  
602 an important direction we leave for future work.

## 603 References

604 Shengnan An, Xunliang Cai, Xuezhi Cao, Xiaoyu  
605 Li, Yehao Lin, Junlin Liu, Xinxuan Lv, Dan Ma,  
606 Xuanlin Wang, Ziwen Wang, and Shuang Zhou.  
607 2025. [Amo-bench: Large language models still  
608 struggle in high school math competitions.](#) *Preprint*,  
609 arXiv:2510.26768.

610 Anthropic. 2025a. [Claude 3.7 sonnet and claude code.](#)

611 Anthropic. 2025b. [Introducing claude opus 4.5.](#)

612 Daman Arora, Himanshu Singh, and Mausam. 2023.  
613 [Have LLMs advanced enough? a challenging prob-  
614 lem solving benchmark for large language models.](#)  
615 In *Proceedings of the 2023 Conference on Empiri-  
616 cal Methods in Natural Language Processing*, pages  
617 7527–7543, Singapore. Association for Computa-  
618 tional Linguistics.

619 Zhangir Azerbayev, Bartosz Piotrowski, Hailey  
620 Schoelkopf, Edward W. Ayers, Dragomir Radev, and  
621 Jeremy Avigad. 2023. [Proofnet: Autoformalizing  
622 and formally proving undergraduate-level mathemat-  
623 ics.](#) *Preprint*, arXiv:2302.12433.

624 Zhipeng Chen, Yingqian Min, Beichen Zhang, Jie Chen,  
625 Jinhao Jiang, Daixuan Cheng, Wayne Xin Zhao,  
626 Zheng Liu, Xu Miao, Yang Lu, Lei Fang, Zhongyuan  
627 Wang, and Ji-Rong Wen. 2025. [An empirical study](#)

[on eliciting and improving r1-like reasoning models.](#) *Preprint*, arXiv:2503.04548.

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.  
2021. [Training verifiers to solve math word prob-  
lems.](#) *Preprint*, arXiv:2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang,  
Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu,  
Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang,  
Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhi-  
hong Shao, Zhuoshu Li, Ziyi Gao, and 181 others.  
2025a. [Deepseek-r1: Incentivizing reasoning capa-  
bility in llms via reinforcement learning.](#) *Preprint*,  
arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin,  
Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao  
Wu, Bowei Zhang, Chaofan Lin, Chen Dong,  
Chengda Lu, Chenggang Zhao, Chengqi Deng, Chen-  
hao Xu, Chong Ruan, Damai Dai, Daya Guo, De-  
jian Yang, and 245 others. 2025b. [Deepseek-v3.2:  
Pushing the frontier of open large language models.](#)  
*Preprint*, arXiv:2512.02556.

Meng Fang, Xiangpeng Wan, Fei Lu, Fei Xing, and  
Kai Zou. 2024. [Mathodyssey: Benchmarking  
mathematical problem-solving skills in large lan-  
guage models using odyssey math data.](#) *Preprint*,  
arXiv:2406.18321.

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. 2024. [Omni-math: A univer-  
sal olympiad level mathematic benchmark for large  
language models.](#) *Preprint*, arXiv:2410.07985.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chen-  
hui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Han-  
lin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai  
Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang,  
Jing Zhang, Juanzi Li, and 37 others. 2024. [Chatglm:  
A family of large language models from glm-130b to  
glm-4 all tools.](#) *Preprint*, arXiv:2406.12793.

Google. 2025. [Gemini 2.5: Our most intelligent ai  
model.](#)

Google. 2025a. [Gemini 3 flash: frontier intelligence  
built for speed.](#)

Google. 2025b. [Start building with gemini 2.5 flash.](#)

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu,  
Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie  
Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan  
Liu, and Maosong Sun. 2024. [OlympiadBench:  
A challenging benchmark for promoting AGI with  
olympiad-level bilingual multimodal scientific prob-  
lems.](#) In *Proceedings of the 62nd Annual Meeting of*

684			
685		<i>the Association for Computational Linguistics (Volume 1: Long Papers)</i> , pages 3828–3850, Bangkok, Thailand. Association for Computational Linguistics.	
686			
687	Jujie He, Jiakai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025. <a href="#">Skywork open reasoner 1 technical report</a> . <i>Preprint</i> , arXiv:2505.22312.		
688			
689			
690			
691			
692			
693	Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. <a href="#">Measuring mathematical problem solving with the math dataset</a> . <i>Preprint</i> , arXiv:2103.03874.		
694			
695			
696			
697			
698	Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. 2025. <a href="#">A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility</a> . <i>Preprint</i> , arXiv:2504.07086.		
699			
700			
701			
702			
703	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. 2025. <a href="#">Goedel-prover-v2: Scaling formal theorem proving with scaffolded data synthesis and self-correction</a> . <i>Preprint</i> , arXiv:2508.03613.		
704			
705			
706			
707			
708			
709			
710			
711	Chengwu Liu, Jianhao Shen, Huajian Xin, Zhengying Liu, Ye Yuan, Haiming Wang, Wei Ju, Chuanyang Zheng, Yichun Yin, Lin Li, Ming Zhang, and Qun Liu. 2023. <a href="#">Fimo: A challenge formal dataset for automated theorem proving</a> . <i>Preprint</i> , arXiv:2309.04295.		
712			
713			
714			
715			
716	Zihan Liu, Yang Chen, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. <a href="#">Acemath: Advancing frontier math reasoning with post-training and reward modeling</a> . <i>Preprint</i> , arXiv:2412.15084.		
717			
718			
719			
720	Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. <a href="#">Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl</a> . Notion Blog.		
721			
722			
723			
724			
725	Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2024. <a href="#">Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems</a> . <i>Preprint</i> , arXiv:2412.09413.		
726			
727			
728			
729			
730			
731			
732	Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. 2025. <a href="#">Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset</a> . <i>Preprint</i> , arXiv:2504.16891.		
733			
734			
735			
736			
737			
738	OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. <a href="#">Gpt-4o system card</a> . <i>Preprint</i> , arXiv:2410.21276.		740 741 742 743 744
739			
	OpenAI. 2024. <a href="#">Openai o1 system card</a> . <i>Preprint</i> , arXiv:2412.16720.		745 746
	OpenAI. 2025. <a href="#">Openai o3-mini: Pushing the frontier of cost-effective reasoning</a> .		747 748
	Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnab Chopra, and et al. 2025. <a href="#">Humanity’s last exam</a> . <i>Preprint</i> , arXiv:2501.14249.		749 750 751 752 753
	Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, and 25 others. 2025. <a href="#">Qwen2.5 technical report</a> . <i>Preprint</i> , arXiv:2412.15115.		754 755 756 757 758 759 760
	Z. Z. Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanxia 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. 2025. <a href="#">Deepseek-prover-v2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition</a> . <i>Preprint</i> , arXiv:2504.21801.		761 762 763 764 765 766 767 768
	Marco Dos Santos, Hugues de Saxcé, Haiming Wang, Ran Wang, Mantas Baksys, Mert Unsal, Junqi Liu, Zhengying Liu, and Jia Li. 2025. <a href="#">Kimina lean server: A high-performance lean server for large-scale verification</a> . <i>Preprint</i> , arXiv:2504.21230.		769 770 771 772 773
	Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2022. <a href="#">Language models are multilingual chain-of-thought reasoners</a> . <i>Preprint</i> , arXiv:2210.03057.		774 775 776 777 778 779
	OpenThoughts Team. 2025a. <a href="#">Open Thoughts</a> . <a href="https://open-thoughts.ai">https://open-thoughts.ai</a> .		780 781
	Qwen Team. 2025b. <a href="#">Qwen3</a> .		782
	Qwen Team. 2025c. <a href="#">Qwq-32b: Embracing the power of reinforcement learning</a> .		783 784
	RUCAIBox STILL Team. 2025d. <a href="#">Still-3-1.5b-preview: Enhancing slow thinking abilities of small models through reinforcement learning</a> .		785 786 787
	George Tsoukalas, Jasper Lee, John Jennings, Jimmy Xin, Michelle Ding, Michael Jennings, Amitayush Thakur, and Swarat Chaudhuri. 2024. <a href="#">Putnam-bench: Evaluating neural theorem-provers on the putnam mathematical competition</a> . <i>Preprint</i> , arXiv:2407.11214.		788 789 790 791 792 793

794	Haiming Wang, Mert Unsal, Xiaohan Lin, Mantas Baksys, Junqi Liu, Marco Dos Santos, Flood Sung, Marina Vinyes, Zhenzhe Ying, Zekai Zhu, Jianqiao Lu, Hugues de Saxcé, Bolton Bailey, Chendong Song, Chenjun Xiao, Dehao Zhang, Ebony Zhang, Frederick Pu, Han Zhu, and 21 others. 2025a. <a href="#">Kimina-prover preview: Towards large formal reasoning models with reinforcement learning</a> . <i>Preprint</i> , arXiv:2504.11354.	851	Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, and 3 others. 2023. <a href="#">A survey of large language models</a> . <i>Preprint</i> , arXiv:2303.18223.	852
795		853		854
796		855	Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2022. <a href="#">Minif2f: a cross-system benchmark for formal olympiad-level mathematics</a> . <i>Preprint</i> , arXiv:2109.00110.	856
797		857		858
798		859	<b>A Appendix</b>	
799		860	<b>A.1 Usability and Accessibility</b>	
800	Yiming Wang, Pei Zhang, Jialong Tang, Haoran Wei, Baosong Yang, Rui Wang, Chenshu Sun, Feitong Sun, Jiran Zhang, Junxuan Wu, Qiqian Cang, Yichang Zhang, Fei Huang, Junyang Lin, Fei Huang, and Jingren Zhou. 2025b. <a href="#">Polymath: Evaluating mathematical reasoning in multilingual contexts</a> . <i>Preprint</i> , arXiv:2504.18428.	861	To support research into LLM reasoning, we have open-sourced the OlymMATH-eval dataset, with 582,400 entries from 28 models, to help compare reasoning capabilities across different models and mathematical domains.	862
801		863		864
802		865		866
803		866	Furthermore, we provide the OlymMATH-demo visualization tool (see Figure 4) to facilitate in-depth analysis of LLM reasoning. This interactive interface enables: (1) Side-by-side comparison of two selected LLMs on the same $\LaTeX$ -rendered problem, with access to reference answers. (2) Color-coded “Problem Grids” for each model, displaying per-problem accuracy for quick identification of challenging areas. (3) Examination of individual model-generated reasoning samples, including correctness, extracted answers, and token counts, crucial for understanding solution processes and identifying flaws. The tool also includes standard solutions for difficult problems and supports local deployment. OlymMATH-demo is thus a valuable asset for dissecting reasoning patterns, diagnosing errors, and guiding LLM development.	867
804		868		869
805		870		871
806		872		873
807		874		875
808		876		877
809		878		879
810	Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. 2025. <a href="#">Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond</a> . <i>Preprint</i> , arXiv:2503.10460.	880		881
811		882		
812				
813				
814				
815				
816	Ruijie Xu, Zengzhi Wang, Run-Ze Fan, and Pengfei Liu. 2024. <a href="#">Benchmarking benchmark leakage in large language models</a> . <i>Preprint</i> , arXiv:2404.18824.			
817				
818				
819	An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. <a href="#">Qwen2.5-math technical report: Toward mathematical expert model via self-improvement</a> . <i>Preprint</i> , arXiv:2409.12122.			
820				
821				
822				
823				
824				
825				
826	Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, Yudong Wang, Zijian Wu, Shuaibin Li, Fengzhe Zhou, Hongwei Liu, Songyang Zhang, Wenwei Zhang, Hang Yan, Xipeng Qiu, and 3 others. 2024. <a href="#">Internlm-math: Open math large language models toward verifiable reasoning</a> . <i>Preprint</i> , arXiv:2402.06332.			
827				
828				
829				
830				
831				
832				
833				
834	Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. <a href="#">Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?</a> <i>Preprint</i> , arXiv:2504.13837.			
835				
836				
837				
838				
839	Yidan Zhang, Yu Wan, Boyi Deng, Baosong Yang, Haoran Wei, Fei Huang, Bowen Yu, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. <a href="#">P-MMEval: A parallel multilingual multitask benchmark for consistent evaluation of LLMs</a> . In <i>Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing</i> , pages 4809–4836, Suzhou, China. Association for Computational Linguistics.			
840				
841				
842				
843				
844				
845				
846				
847				
848	Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen			
849				
850				



Figure 4: The OlymMATH-demo interface. It is currently being maintained on HuggingFace Spaces.

**A.2 Agent’s Prompt Used in Lean 4 Formalization** 883  
884

The following prompt is used in Claude Code to generate a subagent that can work in a separated 885  
886

887  
888  
889  
890  
891  
892

folder to enable large-scale formalization in parallel—the main agent in Claude Code (driven by Claude Opus 4.5) can execute several parallel sub-agents (also driven by Claude Opus 4.5) to work on batched formalization tasks. The Mathlib version of the Lean REPL server is v4.24.0.

```
---
name: lean4-math-formalizer
description: Use this agent when you need to
  ↳ formalize mathematical problems and proofs
  ↳ in Lean 4 from natural language
  ↳ descriptions. This agent reads a txt file
  ↳ containing a mathematical problem and its
  ↳ proof, then generates Lean 4 formalizations.
  ↳ It creates two outputs: a problem
  ↳ formalization with exactly one 'sorry'
  ↳ placeholder, and a complete solution that
  ↳ fills in that sorry. The agent iteratively
  ↳ refines its formalizations using feedback
  ↳ from the lean-server until they compile
  ↳ without
  ↳ errors.
  ↳ Examples:
  ↳ Context: User has a problem.txt file with a
  ↳ mathematical theorem about arctangent and
  ↳ wants it formalized in Lean 4.
  ↳ user: "Please formalize the problem in
  ↳ problem.txt into Lean 4"
  ↳ assistant: "I'll use the lean4-math-formalizer agent to read
  ↳ the problem file and generate the Lean 4
  ↳ formalization."
  ↳ Commentary: Since the user wants to formalize a mathematical
  ↳ problem from a txt file into Lean 4, use
  ↳ the lean4-math-formalizer agent to handle
  ↳ the reading, formalization, and iterative
  ↳ refinement process.
  ↳ Example:
  ↳ Context: User has
  ↳ written a proof for a number theory problem
  ↳ and needs it converted to Lean 4.
  ↳ user: "I have a proof about prime numbers in
  ↳ primes.txt, can you make it into valid Lean
  ↳ 4 code?"
  ↳ assistant: "I'll launch the
  ↳ lean4-math-formalizer agent to convert your
  ↳ proof into Lean 4 formalization with proper
  ↳ verification."
  ↳ Commentary: The user
  ↳ needs mathematical content formalized in
  ↳ Lean 4, which is exactly what the
  ↳ lean4-math-formalizer agent specializes in.
  ↳ It will handle the file reading,
  ↳ formalization, and lean-server
  ↳ validation.
  ↳ Example:
  ↳ Context: User wants to verify
  ↳ their Lean 4 formalization compiles
  ↳ correctly.
  ↳ user: "Generate a Lean 4
  ↳ version of the calculus problem in
  ↳ homework.txt"
  ↳ assistant: "I'll use the
  ↳ lean4-math-formalizer agent to create a
  ↳ verified Lean 4 formalization of your
  ↳ calculus
  ↳ problem."
  ↳ Commentary: Mathematical
  ↳ formalization in Lean 4 requires the
  ↳ specialized lean4-math-formalizer agent
  ↳ which can iteratively refine code using
  ↳ lean-server
  ↳ feedback.
  ↳ Example:
model: opus
color: cyan
---
```

You are an expert Lean 4 formalization  
↳ specialist with deep knowledge of Mathlib,  
↳ mathematical logic, and proof engineering.  
↳ Your primary mission is to transform natural  
↳ language mathematical problems and proofs  
↳ into rigorous, compilable Lean 4 code.

## ## Your Core Workflow

### ### Step 1: Read and Analyze the Input File

- Read the provided txt file (e.g.,  
↳ problem.txt) containing the mathematical  
↳ problem and proof
- Carefully extract:
  - The precise mathematical statement to be  
↳ proven
  - Key definitions and concepts involved
  - The proof strategy and techniques used
  - Any special mathematical functions or  
↳ operations (e.g., arctan, roots of  
↳ polynomials)

### ### Step 2: Generate Problem Formalization

- ↳ (problem-lean.txt - Part 1)
- Create a Lean 4 file that:
  - Starts with `import Mathlib`
  - Includes any necessary namespace opens
  - Defines the theorem named exactly `to_prove`
  - Uses exactly ONE `sorry` as the proof  
↳ placeholder
  - Properly types all mathematical objects

Example structure:

```
```lean
import Mathlib

open Real

theorem to_prove : [statement] := by
  sorry
```
```

### ### Step 3: Validate Problem Formalization

- Use the lean-server to check your  
↳ formalization:

```
```python
import json
from kimina_client import KiminaClient, Snippet

client = KiminaClient(api_url='http://localhost:8000')

snippets = [
    Snippet(id='problem-formalization',
            ↳ code='[YOUR LEAN CODE HERE]'),
]

resp = client.api_check(snippets=snippets,
↳ timeout=20, debug=False, reuse=True,
↳ safe=True)
print(json.dumps(resp.model_dump(), indent=2,
↳ ensure_ascii=False))
```
```
- Success criteria for problem formalization:
  - No errors in the response
  - Exactly one warning about "declaration uses  
↳ 'sorry'"

```

- The sorrys field should contain exactly
↳ one sorry

### Step 4: Iterative Refinement for Problem
↳ Formalization
- If the lean-server returns errors:
- Analyze the error messages carefully
- Common issues to address:
  - Missing imports or opens (e.g., `open
  ↳ Real`, `open Polynomial`)
  - Incorrect type signatures
  - Mathlib naming conventions (check for
  ↳ correct lemma/theorem names)
  - Syntax errors in Lean 4 vs Lean 3
- Modify your code and re-check until only
↳ the sorry warning remains

### Step 5: Generate Complete Solution
↳ (problem-lean.txt - Part 2)
- Replace the `sorry` with the actual proof
- Translate the natural language proof into
↳ Lean 4 tactics:
  - Use appropriate tactics: `simp`, `ring`,
  ↳ `norm_num`, `nlinarith`, `polyrith`, etc.
  - For trigonometric proofs, consider
  ↳ `Real.arctan_add`, angle addition
  ↳ formulas
  - For polynomial roots, use Mathlib's
  ↳ polynomial library
  - Break complex proofs into `have` statements
  - Use `calc` blocks for equality chains when
  ↳ appropriate

### Step 6: Validate Complete Solution
- Check with lean-server again
- Success criteria for complete solution:
  - No errors
  - No warnings
  - The response indicates successful
  ↳ compilation (no goals remaining)

### Step 7: Final Refinement
- If errors persist in the solution:
  - Check tactic compatibility
  - Ensure all intermediate goals are discharged
  - Verify type unification
  - Consider alternative proof strategies if
  ↳ current approach fails
- Continue iterating until the solution
↳ compiles cleanly

## Output Format

Generate a file named `problem-lean.txt` (Note
↳ that it means adding a "-lean" after the
↳ file's original name) containing both
↳ sections clearly marked:

...
-- PROBLEM FORMALIZATION (with sorry)
-- This version should have exactly one 'sorry'
↳ warning

import Mathlib

[problem formalization with sorry]

-- =====
-- COMPLETE SOLUTION (no sorry)

```

```

-- This version should compile with no warnings
↳ or errors

import Mathlib

[complete proof without sorry]
```

## Critical Requirements

1. Naming: The main theorem MUST be named
↳ `to_prove`
2. Imports: Always start with `import
↳ Mathlib`
3. Validation: ALWAYS use the lean-server
↳ to validate before finalizing
4. Iteration: Do not give up after first
↳ error - analyze and fix iteratively
5. Completeness: Both formalization AND
↳ solution must be provided

## Common Mathlib Patterns

- For `arctan`: `Real.arctan`, properties in
↳ `Mathlib.Analysis.SpecialFunctions.Trigono`
↳ `metric.Arctan`
- For polynomials: `Polynomial`,
↳ `Polynomial.roots`, `Polynomial.sum_roots`
↳ `eq_neg_coeff_div_leading_coeff`
- For real analysis: `open Real`, `open scoped
↳ Real`
- For complex numbers: `open Complex`

## Error Recovery Strategies

1. Type mismatch: Check if you need
↳ explicit type annotations or coercions
2. Unknown identifier: Search Mathlib for
↳ correct naming, try `#check` commands
3. Tactic failure: Try breaking into
↳ smaller steps with `have`
4. Timeout: Simplify expressions, add
↳ intermediate lemmas

You are methodical, persistent, and precise.
↳ You do not give up until both the problem
↳ formalization and solution compile
↳ successfully according to the lean-server
↳ validation. Meanwhile, the name of all the
↳ temporary files you created should start
↳ with the original file name. e.g.
↳ "problem_test-lean.py". Meanwhile, you
↳ should only use axiom when it is a well
↳ known lemma and it is extremely hard to
↳ give its full solution.

```

### A.3 Demonstrations, Case Study Examples and Full Evaluation Results 893 894

This part presents the detailed content of the dataset, the case study examples and full evaluation results mentioned before. 895  
896  
897

**Problem:** Given that two vertices of an equilateral triangle are on the parabola  $y^2 = 4x$ , and the third vertex is on the directrix of the parabola, and the distance from the center of the triangle to the directrix equals  $\frac{1}{9}$  of the perimeter. Find the area of the triangle.  
**Subject:** Geometry

Figure 5: A geometry problem described precisely in text from OlymMATH.

**Original problem:** If the distances from the eight vertices of a cube to a certain plane are 0, 1, 2, 3, 4, 5, 6, 7 respectively, what is the possible edge length of this cube?  
**After transformation:** If the distances from the eight vertices of a cube to a certain plane are 0, 1, 2, 3, 4, 5, 6, 7 respectively, consider all possible edge lengths of this cube. Assuming the possible edge lengths form a set  $S$ , find the sum of squares of all elements in  $S$ .

Figure 6: An OlymMATH-HARD example testing model’s identification of all possible answers.

**Problem:** In the triangle  $\triangle ABC$ , the lengths of the three sides are  $a, b, c$ , satisfying the condition  $2a^2 + b^2 + c^2 = 4$ . Find the maximum value of the area of  $\triangle ABC$ .  
**Answer:**  $\frac{\sqrt{5}}{5}$ . (When  $a = \frac{2}{\sqrt{5}}$ ,  $b = c = \sqrt{\frac{6}{5}}$ )

**Model Output:** We wish to maximize the area of  $\triangle ABC$ , subject to  $2a^2 + b^2 + c^2 = 4$ . Because the condition is symmetric in  $b$  and  $c$  (but not in  $a$ ), it is reasonable to look for an optimum among triangles having  $b = c$ . (Later one can check that assuming  $b = c$  indeed gives the maximum.)  
The problem involves maximizing the area of an isosceles triangle with sides  $a, b, b$  under the constraint  $2a^2 + b^2 + c^2 = 4$ . By assuming  $b = c$ , we derive a formula for the area in terms of  $b$  and find the maximum area by expressing it as a function of  $u = b^2$ . We then determine the critical points of this quadratic function, identify the maximum area, and finally recover the side lengths to verify that they satisfy the original constraints. The maximum area achieved is  $\frac{\sqrt{5}}{5}$ .

Figure 7: An example during our data collection. o3-mini (high) found the correct answer without rigorously solving the problem. Due to length constraints, we provide only a summary of its original solution steps.

**Problem:** Let the sequence of rationals  $x_1, x_2, \dots$  be defined such that  $x_1 = \frac{25}{11}$  and  $x_{k+1} = \frac{1}{3} \left( x_k + \frac{1}{x_k} - 1 \right)$ .  $x_{2025}$  can be expressed as  $\frac{m}{n}$  for relatively prime positive integers  $m$  and  $n$ . Find the remainder when  $m + n$  is divided by 1000.  
**Answer:** 248.

**Model Output:** We will show that if  $x_1 = \frac{25}{11}$  and  $x_{k+1} = \frac{1}{3} \left( x_k + \frac{1}{x_k} - 1 \right)$ , then after “many” iterations the answer to  $x_{2025} = \frac{m}{n}$  (in lowest terms) satisfies  $m + n \equiv 248 \pmod{1000}$ . In what follows we briefly describe one way (via a “miraculous” change-of-variables) to “solve” the recurrence; the details need not be reproduced fully here.

A good hint is that the recurrence  $x_{k+1} = \frac{1}{3} \left( x_k + \frac{1}{x_k} - 1 \right)$  has the curious symmetry  $T(x) = \frac{1}{3} \left( x + \frac{1}{x} - 1 \right) = T\left(\frac{1}{x}\right)$ , so the expression  $x + \frac{1}{x}$  (which is “symmetric” in  $x$  and its reciprocal) naturally appears. In fact, one may rewrite the recurrence in the form  $3x_{k+1} + 1 = x_k + \frac{1}{x_k}$ . A useful idea is then to “complete the square” by writing  $\left(\sqrt{x_k} - \frac{1}{\sqrt{x_k}}\right)^2 = x_k + \frac{1}{x_k} - 2$ , so that  $3x_{k+1} + 1 = \left(\sqrt{x_k} - \frac{1}{\sqrt{x_k}}\right)^2 + 2$ . Defining  $u_k = \sqrt{x_k} - \frac{1}{\sqrt{x_k}}$ , one obtains the “linear-looking” relation  $u_k^2 = 3x_{k+1} - 1$ . **With some additional work it turns out that** if we write  $x_k = \frac{m_k}{n_k}$  (in lowest terms), then **one may prove by induction** that  $m_k + n_k = 36 \cdot 12^{(2^{k-1}-1)}$ . (In our case one checks that  $m_1 + n_1 = 25 + 11 = 36$ ,  $m_2 + n_2 = 432$ ,  $m_3 + n_3 = 62208$ , and indeed  $36, 432, 62208$  equal  $36 \cdot 12^0, 36 \cdot 12^1, 36 \cdot 12^3$  respectively.)

Thus, for  $k = 2025$  we have  $m + n = 36 \cdot 12^{(2^{2024}-1)}$ . Even though this number is astronomically large, we are only asked to compute  $m + n \pmod{1000}$ .

Using modular reductions, we compute  $m + n$  modulo 1000. First, modulo 8, we find  $36 \cdot 12^{(2^{2024}-1)} \equiv 0 \pmod{8}$ . Modulo 125, Euler’s theorem helps reduce  $12^N$  to  $12^{15} \equiv 118 \pmod{125}$ , giving  $36 \cdot 12^N \equiv 123 \pmod{125}$ . Using the Chinese Remainder Theorem, solving  $125k + 123 \equiv 0 \pmod{8}$  gives  $k \equiv 1 \pmod{8}$ , leading to the final answer: 248.

Figure 8: An example from AIME 2025. o3-mini (high) forgot to prove that  $(m_k, n_k) = 1$  and it claimed that  $m_k + n_k = 36 \cdot 12^{(2^{k-1}-1)}$  without providing the complete proof (highlighted in red). Instead, the model “guessed” this expression by calculating the first several terms in the sequence one by one. Due to length constraints, we provide only a summary of its unimportant steps.

**Source:** 2016 IMO Shortlist A7

**Problem:** Find all functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  such that  $f(0) \neq 0$  and for all  $x, y \in \mathbb{R}$ ,

$$f(x+y)^2 = 2f(x)f(y) + \max\{f(x^2+y^2), f(x^2) + f(y^2)\}.$$

**Answer:**  $f(x) = -1$  and  $f(x) = x - 1$ .

**Solution:** (Provided in Omni-MATH) To find all functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  satisfying the given functional equation, we start with the condition:

$$f(0) \neq 0,$$

and the functional equation for all  $x, y \in \mathbb{R}$ :

$$f(x+y)^2 = 2f(x)f(y) + \max\{f(x^2+y^2), f(x^2) + f(y^2)\}.$$

Step 1: Test Simple Functions

Let's start by testing the functions given in the reference answer:  $f(x) = -1$  and  $f(x) = x - 1$ .

Case 1:  $f(x) = -1$ . Substitute  $f(x) = -1$  into the equation:  $\dots$

Case 2:  $f(x) = x - 1$ . Substituting  $f(x) = x - 1$ :  $\dots$  confirming  $f(x) = x - 1$  is another valid solution.

Conclusion

**After verifying the two functions, we conclude that** the functions satisfying the original functional equation are:

$$f(x) = -1 \quad \text{and} \quad f(x) = x - 1.$$

Thus, the solutions are:

$$f(x) = -1 \quad \text{and} \quad f(x) = x - 1.$$

**Model Output:** We will show that the only functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  with  $f(0) \neq 0$ , satisfying

$$f(x+y)^2 = 2f(x)f(y) + \max\{f(x^2+y^2), f(x^2) + f(y^2)\} \quad \text{for all } x, y \in \mathbb{R},$$

are

$$f(x) = -1 \quad \text{for all } x \quad \text{or} \quad f(x) = x - 1.$$

Below is an outline of one way to arrive at this result.

Step 1. Plug in  $y = 0$

Let  $y = 0$  in the given equation:

$$f(x)^2 = 2f(x)f(0) + \max\{f(x^2), f(x^2) + f(0)\}.$$

Write  $c = f(0) \neq 0$ . Notice that the two arguments inside the maximum differ by  $f(0)$ ; therefore, the larger one depends on the sign of  $c$ :

- \*\*If  $c > 0$ :\*\* Then  $f(x^2) + c > f(x^2)$  so the equation becomes  $f(x)^2 = 2cf(x) + f(x^2) + c$ .

- \*\*If  $c < 0$ :\*\* Then  $f(x^2) + c < f(x^2)$  so we have  $f(x)^2 = 2cf(x) + f(x^2)$ .

Thus, in each case one obtains a relationship between  $f(x)$  and  $f(x^2)$ .

Step 2. Check constant solutions  $\dots$  Thus, the constant solution is  $f(x) = -1$  for all  $x$ .

Step 3. Look for non-constant solutions

A natural next guess is to try a linear function.  $\dots$

A quick check verifies that  $f(x) = x - 1$  satisfies the original equation.  $\dots$  which equals  $(x + y - 1)^2$ , matching the left-hand side.

Conclusion

The **only** functions  $f : \mathbb{R} \rightarrow \mathbb{R}$  with  $f(0) \neq 0$  that satisfy  $\dots$  for all real numbers  $x$  and  $y$  are:

$$f(x) = -1 \quad \text{for all } x \quad \text{or} \quad f(x) = x - 1.$$

**These are the complete solutions.**

Figure 9: An example from Omni-MATH. The solution provided by Omni-MATH itself is flawed and problematic. Both the reference solution and o3-mini (high) failed to prove that there are no other possible function  $f$  that satisfies the condition (highlighted in red). Due to length constraints, we have omitted its unimportant steps.

**Problem:** Let  $a, b, c \in \mathbb{R}$ ,  $a^3b + b^3c + c^3a = 3$ , find the minimum value of the expression  $f(a, b, c) = (\sum a^4)^4 + 1000 \sum a^2b^2$ .

**Reference Answer:** 2625. (Equality condition:  $a = \pm \frac{4}{\sqrt{7}} \sin^2 \frac{4\pi}{7}$ ,  $b = \pm \frac{4}{\sqrt{7}} \sin^2 \frac{2\pi}{7}$ ,  $c = \pm \frac{4}{\sqrt{7}} \sin^2 \frac{\pi}{7}$  and its cyclic permutations)

**Model Output:** We wish to minimize

$$f(a, b, c) = (a^4 + b^4 + c^4)^4 + 1000 (a^2b^2 + b^2c^2 + c^2a^2)$$

subject to the (cyclic) constraint

$$a^3b + b^3c + c^3a = 3,$$

with  $a, b, c \in \mathbb{R}$ .

A natural idea is to look for a "symmetric-like" solution. (Even though the constraint is cyclic rather than fully symmetric, the very "even" form of the objective  $f$  — all terms are even — suggests that the optimum might occur when the "building blocks" have equal absolute value.) One obvious candidate is to try

$$a = b = c = t.$$

Then the constraint becomes

$$a^3b + b^3c + c^3a = 3t^4 = 3 \implies t^4 = 1.$$

Thus  $t = 1$  or  $t = -1$ . (In  $\mathbb{R}$  the equation  $t^4 = 1$  has the two solutions  $t = 1$  and  $t = -1$ .) Now compute the value of  $f$  at  $a = b = c = t$ . Notice that

$$a^4 + b^4 + c^4 = t^4 + t^4 + t^4 = 3,$$

$$a^2b^2 + b^2c^2 + c^2a^2 = t^4 + t^4 + t^4 = 3$$

(since  $t^4 = 1$ ).

Then

$$f = (3)^4 + 1000 \cdot 3 = 81 + 3000 = 3081.$$

Before concluding that 3081 is the minimum, one may ask: Could there be a choice of  $a, b, c$  satisfying  $a^3b + b^3c + c^3a = 3$  for which

$$(a^4 + b^4 + c^4) < 3 \quad \text{or} \quad (a^2b^2 + b^2c^2 + c^2a^2) < 3?$$

Since  $a^4, b^4, c^4$  are nonnegative and the constraint (a homogeneous form of degree 4) "forces" the variables to be large enough in a certain sense, **one may show by a careful (often inequality-based) argument that** if  $a^3b + b^3c + c^3a = 3$ , then one must have

$$a^4 + b^4 + c^4 \geq 3 \quad \text{and} \quad a^2b^2 + b^2c^2 + c^2a^2 \geq 3.$$

In fact, equality in these "loss-inequalities" forces  $|a| = |b| = |c| = 1$ , and it turns out (by checking the signs in the constraint) that the only possibility for equality is  $a = b = c = 1$  or  $a = b = c = -1$ .

Also, one might try "extremal" cases such as letting one variable vanish (say,  $c = 0$ ). In that case, the constraint reduces to  $a^3b = 3$  so that  $a$  and  $b$  cannot be arbitrarily small (or close in absolute value to 1), and **a short calculation shows that** the value of  $f$  then exceeds 3081.

Thus, the candidate  $a = b = c = 1$  (or  $a = b = c = -1$ ) gives the minimal value. In conclusion, the minimum value of  $f$  is

$$3081,$$

and it is attained exactly when  $a, b, c$  are all equal with  $|a| = 1$  and the signs arranged so that  $a^3b = 1$  (that is,  $a = b = c = 1$  or  $a = b = c = -1$ ).

Figure 10: An example from OlymMATH-HARD subset. o3-mini (high) attempted to "guess" the equality conditions but failed to produce a correct solution (highlighted in red).

Model	OlymMATH-HARD (EN)										OlymMATH-EASY (EN)									
	Alg.		Geo.		Num.		Com.		Avg.		Alg.		Geo.		Num.		Com.		Avg.	
	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k
Qwen3 (0.6B, Think)	2.5	0.0	2.1	4.0	6.6	8.0	0.2	0.0	<b>2.8</b>	<b>3.0</b>	15.5	20.0	5.6	15.2	24.5	38.5	5.2	6.9	<b>10.4</b>	<b>17.0</b>
DS-R1-Distill (1.5B)	1.9	0.0	1.8	0.0	1.8	0.0	0.4	0.0	1.5	0.0	20.8	40.0	12.6	21.2	32.6	61.5	8.2	24.1	16.0	32.0
STILL-3-Pre. (1.5B)	3.7	0.0	4.9	4.0	5.8	8.0	0.8	0.0	3.8	3.0	22.7	36.0	14.8	30.3	37.6	69.2	10.3	17.2	18.4	<u>33.0</u>
DeepScaleR-Pre. (1.5B)	3.4	4.0	4.2	8.0	8.2	4.0	0.4	0.0	<u>4.1</u>	<u>4.0</u>	19.9	16.0	18.5	21.2	44.6	46.2	18.9	31.0	<u>22.3</u>	26.0
OpenMath-Nemo. (1.5B)	14.5	24.0	13.6	16.0	10.9	16.0	2.6	4.0	<b>10.4</b>	<b>15.0</b>	70.9	100.0	59.3	90.9	81.6	100.0	40.6	58.6	<b>59.7</b>	<b>85.0</b>
Qwen3 (4B, Think)	18.1	20.0	14.8	12.0	19.8	28.0	3.1	4.0	<b>13.9</b>	<b>16.0</b>	76.4	92.0	79.1	97.0	85.1	84.6	57.1	72.4	<b>72.8</b>	<b>87.0</b>
DS-R1-Distill (7B)	15.6	36.0	12.6	24.0	13.1	24.0	3.1	4.0	11.1	22.0	52.8	84.0	49.6	84.8	62.5	84.6	33.9	58.6	47.5	77.0
Light-R1-DS (7B)	17.1	28.0	15.2	16.0	12.8	24.0	3.6	4.0	12.2	18.0	57.1	84.0	53.6	93.9	73.7	84.6	39.5	51.7	53.0	78.0
OpenThinker2 (7B)	16.0	20.0	16.8	28.0	14.0	20.0	2.8	4.0	12.4	18.0	65.3	96.0	60.5	97.0	79.1	84.6	42.3	58.6	58.9	84.0
Skywork-OR1-Pre. (7B)	14.4	20.0	12.5	12.0	11.7	24.0	1.6	0.0	10.0	14.0	61.6	88.0	55.9	78.8	74.3	92.3	36.9	48.3	54.2	74.0
Skywork-OR1-Math (7B)	17.4	20.0	17.1	20.0	13.6	28.0	0.9	0.0	12.2	17.0	67.9	92.0	67.4	93.9	76.6	92.3	47.6	62.1	<u>63.0</u>	84.0
AceMath-RL (7B)	19.4	32.0	19.3	32.0	14.4	24.0	3.5	4.0	<u>14.2</u>	<u>23.0</u>	69.7	96.0	63.7	93.9	79.0	84.6	44.2	69.0	61.5	<u>86.0</u>
OpenMath-Nemo. (7B)	26.9	36.0	18.6	28.0	19.8	28.0	4.4	4.0	<b>17.4</b>	<b>24.0</b>	86.4	100.0	76.4	97.0	91.5	100.0	55.3	72.4	<b>74.7</b>	<b>91.0</b>
DS-R1-Distill (14B)	16.1	16.0	17.0	16.0	18.1	32.0	2.1	4.0	13.3	17.0	69.0	96.0	65.1	97.0	79.4	92.3	44.0	65.5	61.8	<u>87.0</u>
Light-R1-DS (14B)	21.8	24.0	22.2	28.0	17.8	36.0	2.6	4.0	<u>16.1</u>	<u>23.0</u>	72.3	88.0	73.0	100.0	84.3	92.3	47.6	65.5	<u>66.9</u>	86.0
OpenMath-Nemo. (14B)	28.7	40.0	22.1	32.0	21.0	32.0	3.4	4.0	<b>18.8</b>	<b>27.0</b>	87.9	100.0	78.5	93.9	95.8	100.0	59.9	86.2	<b>77.7</b>	<b>94.0</b>
Qwen3 (30B-A3B, Think)	38.8	44.0	33.8	44.0	26.7	36.0	5.9	4.0	<b>26.3</b>	<b>32.0</b>	91.4	100.0	92.9	100.0	90.9	92.3	75.6	93.1	<b>87.2</b>	<b>97.0</b>
DS-R1-Distill (32B)	22.4	32.0	21.4	24.0	20.3	40.0	3.4	4.0	16.9	25.0	73.6	100.0	71.8	97.0	84.5	92.3	49.0	69.0	67.3	89.0
QwQ (32B)	32.9	28.0	26.6	36.0	26.7	44.0	6.2	4.0	<u>23.1</u>	<u>28.0</u>	91.8	100.0	87.0	100.0	95.0	100.0	69.0	89.7	<b>84.0</b>	<b>97.0</b>
Light-R1-DS (32B)	28.9	44.0	31.1	52.0	24.1	36.0	5.2	8.0	22.3	<u>35.0</u>	84.2	100.0	83.3	100.0	92.5	100.0	62.1	82.8	78.6	<u>95.0</u>
OpenThinker2 (32B)	24.1	32.0	22.9	32.0	18.0	20.0	2.6	4.0	16.9	22.0	79.4	96.0	74.0	100.0	90.4	92.3	56.5	79.3	72.4	92.0
Skywork-OR1-Pre. (32B)	37.2	52.0	32.3	48.0	27.0	40.0	4.2	4.0	<b>25.2</b>	<b>36.0</b>	89.3	100.0	87.3	100.0	92.4	100.0	63.9	82.8	<u>81.7</u>	<u>95.0</u>
GLM-Z1-Air (32B)	35.0	44.0	21.5	32.0	19.5	24.0	4.5	4.0	20.1	26.0	86.5	100.0	79.5	90.9	90.4	100.0	59.1	75.9	76.8	90.0
OpenMath-Nemo. (32B)	22.0	36.0	21.0	28.0	20.0	24.0	3.5	4.0	16.6	23.0	75.5	100.0	60.6	90.9	89.4	100.0	42.2	69.0	62.7	88.0
Qwen3 (235B-A22B, Think)	48.0	52.0	49.5	60.0	38.0	36.0	10.5	16.0	<b>36.5</b>	<b>41.0</b>	93.5	100.0	92.4	100.0	99.0	100.0	81.9	93.1	<b>90.5</b>	<b>98.0</b>
DeepSeek R1	30.0	40.0	25.5	32.0	18.5	24.0	4.0	4.0	19.5	25.0	90.5	100.0	82.2	97.0	94.2	100.0	60.8	72.4	79.6	91.0
OpenAI o3-mini (high)	29.5	32.0	29.0	44.0	49.5	60.0	17.0	20.0	<u>31.2</u>	<u>39.0</u>	93.0	92.0	89.8	100.0	97.1	100.0	89.2	96.6	<u>91.4</u>	<b>97.0</b>
Gemini 2.5 Pro Exp 0325	71.5	76.0	75.5	84.0	59.0	72.0	27.5	36.0	<b>58.4</b>	<b>67.0</b>	92.0	100.0	97.0	100.0	98.1	100.0	84.5	89.7	<b>92.2</b>	<b>97.0</b>

Table 9: All models’ performance on OlymMATH (EN). Models within each model size group are sorted by release time. The abbreviations “Alg.,” “Geo.,” “Num.,” and “Com.” represent the four categories in OlymMATH. Highest accuracy per model size is bolded. The second highest accuracy per model size is underlined. Models sampled only 8 times are marked in gray to indicate potential instability.

Model	OlymMATH-HARD (ZH)										OlymMATH-EASY (ZH)									
	Alg.		Geo.		Num.		Com.		Avg.		Alg.		Geo.		Num.		Com.		Avg.	
	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k	P@1	C@k
Qwen3 (0.6B, Think)	2.6	4.0	0.8	0.0	4.4	4.0	0.0	0.0	<b>1.9</b>	<b>2.0</b>	9.9	8.0	2.8	3.0	12.0	15.4	1.3	3.4	<b>5.4</b>	<b>6.0</b>
DS-R1-Distill (1.5B)	1.8	0.0	1.3	0.0	1.1	0.0	0.0	0.0	1.0	0.0	13.7	20.0	6.3	9.1	20.9	30.8	2.6	0.0	9.0	12.0
STILL-3-Pre. (1.5B)	2.9	0.0	2.2	0.0	4.5	4.0	0.2	0.0	2.5	1.0	15.9	32.0	7.4	18.2	27.6	46.2	4.3	6.9	11.3	<u>22.0</u>
DeepScaleR-Pre. (1.5B)	4.4	8.0	2.6	4.0	6.4	8.0	0.1	0.0	<u>3.4</u>	<u>5.0</u>	15.9	20.0	7.2	9.1	32.6	46.2	8.9	20.7	<u>13.2</u>	20.0
OpenMath-Nemo. (1.5B)	13.9	16.0	9.8	4.0	13.3	16.0	0.8	0.0	<b>9.5</b>	<b>9.0</b>	67.9	96.0	37.6	57.6	65.3	76.9	27.6	41.4	<b>45.9</b>	<b>65.0</b>
Qwen3 (4B, Think)	12.5	20.0	7.0	8.0	12.6	24.0	0.9	0.0	<b>8.3</b>	<b>13.0</b>	70.8	88.0	61.0	75.8	74.8	92.3	41.8	51.7	<b>59.7</b>	<b>74.0</b>
DS-R1-Distill (7B)	6.1	8.0	7.9	12.0	6.6	8.0	0.6	0.0	5.3	<u>7.0</u>	38.0	64.0	30.8	51.5	49.2	61.5	18.7	27.6	31.5	49.0
Light-R1-DS (7B)	7.1	4.0	9.4	12.0	7.8	12.0	1.1	0.0	6.3	<u>7.0</u>	42.9	76.0	42.7	72.7	56.9	61.5	22.7	31.0	38.8	60.0
OpenThinker2 (7B)	7.0	0.0	7.3	8.0	7.4	8.0	1.0	0.0	5.7	<u>4.0</u>	48.2	80.0	44.7	72.7	57.8	76.9	22.4	37.9	40.8	65.0
Skywork-OR1-Pre. (7B)	4.7	4.0	7.8	8.0	7.4	8.0	0.4	0.0	5.1	5.0	41.1	60.0	36.6	54.5	58.1	69.2	23.6	34.5	36.8	52.0
Skywork-OR1-Math (7B)	6.4	8.0	8.3	8.0	9.8	12.0	0.8	0.0	6.3	<u>7.0</u>	45.2	72.0	40.0	63.6	62.3	69.2	30.2	37.9	41.3	59.0
AceMath-RL (7B)	6.4	8.0	10.7	12.0	7.8	8.0	1.4	0.0	<u>6.6</u>	<u>7.0</u>	55.1	88.0	46.6	75.8	66.9	76.9	31.0	44.8	<u>46.9</u>	<u>70.0</u>
OpenMath-Nemo. (7B)	25.0	32.0	20.8	28.0	22.3	36.0	4.8	4.0	<b>18.2</b>	<b>25.0</b>	86.8	100.0	72.7	90.9	91.8	100.0	57.9	79.3	<b>74.4</b>	<b>91.0</b>
DS-R1-Distill (14B)	5.2	0.0	5.3	4.0	8.7	16.0	0.2	0.0	4.9	5.0	43.1	56.0	38.9	66.7	58.2	69.2	24.8	31.0	38.4	54.0
Light-R1-DS (14B)	6.2	4.0	7.5	8.0	10.9	12.0	0.2	0.0	<u>6.2</u>	<u>6.0</u>	56.6	84.0	45.5	75.8	66.5	76.9	28.7	37.9	<u>46.1</u>	<u>67.0</u>
OpenMath-Nemo. (14B)	28.7	32.0	26.1	40.0	26.8	40.0	4.2	4.0	<b>21.4</b>	<b>29.0</b>	88.3	100.0	75.2	100.0	94.5	100.0	60.2	86.2	<b>76.6</b>	<b>96.0</b>
Qwen3 (30B-A3B, Think)	35.6	40.0	24.1	28.0	18.1	24.0	2.7	4.0	<b>20.1</b>	<b>24.0</b>	87.8	92.0	84.7	97.0	91.3	100.0	61.9	65.5	<b>79.7</b>	<b>87.0</b>
DS-R1-Distill (32B)	6.5	0.0	5.4	4.0	10.6	12.0	0.7	0.0	5.8	4.0	45.2	52.0	41.8	63.6	60.2	69.2	26.0	37.9	40.4	54.0
QwQ (32B)	20.9	24.0	15.9	16.0	17.6	24.0	2.0	0.0	14.1	16.0	85.4	96.0	76.6	97.0	92.9	100.0	53.8	69.0	<b>74.3</b>	<b>89.0</b>
Light-R1-DS (32B)	16.8	28.0	12.0	12.0	13.4	16.0	4.4	16.0	11.6	<u>18.0</u>	70.1	96.0	64.1	93.9	80.4	92.3	39.8	51.7	60.7	82.0
OpenThinker2 (32B)	13.6	16.0	11.1	16.0	12.7	20.0	0.9	0.0	9.6	13.0	68.0	92.0	64.3	93.9	84.6	92.3	44.8	65.5	62.2	85.0
Skywork-OR1-Pre. (32B)	19.6	20.0	16.8	20.0	18.9	24.0	3.5	4.0	<u>14.7</u>	<u>17.0</u>	79.5	96.0	72.1	93.9	88.0	100.0	45.4	58.6	<u>68.3</u>	85.0
GLM-Z1-Air (32B)	18.0	16.0	12.0	8.0	16.0	16.0	2.5	4.0	12.1	11.0	76.0	96.0	69.3	78.8	89.4	92.3	41.8	48.3	65.6	76.0
OpenMath-Nemo. (32B)	22.5	36.0	22.5	32.0	22.5	28.0	3.5	4.0	<b>17.8</b>	<b>25.0</b>	68.0	96.0	62.5	90.9	90.4	100.0	48.7	72.4	63.5	<u>88.0</u>
Qwen3 (235B-A22B, Think)	36.5	48.0	43.5	48.0	28.5	32.0	4.0	8.0	<b>28.1</b>	<b>34.0</b>	91.0	100.0	90.2	97.0	94.2	100.0	78.4	89.7	<b>87.5</b>	<b>96.0</b>
DeepSeek R1	20.0	24.0	25.0	28.0	17.0	16.0	1.5	0.0	15.9	17.0	79.5	96.0	74.6	84.8	88.5	92.3	49.6	55.2	70.4	80.0
OpenAI o3-mini (high)	31.5	40.0	32.5	44.0	48.5	56.0	19.0	28.0	<u>32.9</u>	<u>42.0</u>	93.0	96.0	89.4	100.0	99.0	100.0	85.8	93.1	<u>90.5</u>	<b>97.0</b>
Gemini 2.5 Pro Exp 0325	65.0	76.0	78.0	80.0	53.5	56.0	25.0	40.0	<u>55.4</u>	<u>63.0</u>	90.5	96.0	93.2	93.9	100.0	100.0	84.1	86.2	<b>90.8</b>	<u>93.0</u>

Table 10: All models’ performance on OlymMATH (ZH). Models within each model size group are sorted by release time. The abbreviations “Alg.,” “Geo.,” “Num.,” and “Com.” represent the four categories in OlymMATH. Highest accuracy per model size is bolded. The second highest accuracy per model size is underlined. Models sampled only 8 times are marked in gray to indicate potential instability.

	Type	Example
Included	Real number	$2^{2017} + \arctan 2$
	Interval	$[\sqrt{33}, +\infty), (4, 5\pi]$
Excluded	Set Operations	$\{4, 5\} \cup \{1, 8\}$
	Variable	$\sqrt[3]{5a^2}, p^2 - pq$
	Complex number	$9 + 4i, \sqrt{-4}$
	Text	East, Alice

Table 11: *Included* and *excluded* formats of the answer.