From Atomic to Agentic: Towards Interpretable Evaluation of LLMs' Agentic Mathematical Capabilities

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are evolving from performing end-to-end mathematical reasoning to integrating *agentic intelligence*. However, most existing math benchmarks evaluate only final answers. This outcome-oriented evaluation provides limited diagnostic value for identifying process-level failures or rigorous logic, failing to guide the transformation of LLMs into robust agents. To bridge this gap, we present a process-level benchmark designed to evaluate the *inherent agentic mathematical reasoning abilities of LLMs*. Our framework aligns problem-solving **agentic** behaviors with a structured taxonomy of reusable mathematical **atomic** capabilities. We design a comprehensive suite of planning, action, and feedback tasks across both textual and multimodal contexts, supported by an automated pipeline that synthesizes high-quality trajectories and produces fine-grained annotations via controlled LLM rewriting. Experiments reveal that models with stronger end-to-end accuracy can exhibit markedly different agentic capability profiles. This demonstrates that process-level evaluation is crucial for interpreting the true potential of LLMs and guiding the development of next-generation mathematical agents [1].

1 Introduction

In recent years, Large Language Models (LLMs) have achieved remarkable progress on complex reasoning tasks, particularly in the domain of mathematics (Lewkowycz et al., 2022; Achiam et al., 2023). As tasks become increasingly diverse and complex, LLM-driven approaches are evolving beyond simple chain-of-thought generation toward *agentic paradigms* (Wei et al., 2022; Wang et al., 2024a). By dynamically incorporating planning, tool execution, and self-reflection, these paradigms deliver more structured reasoning processes while boosting both stability and interpretability (Yao et al., 2023a; Schick et al., 2023; Shinn et al., 2023; Yao et al., 2023b). However, before deploying complex agentic systems, a critical question remains: *Do foundational LLMs possess the inherent agentic capabilities required to apply effectively within these frameworks?*

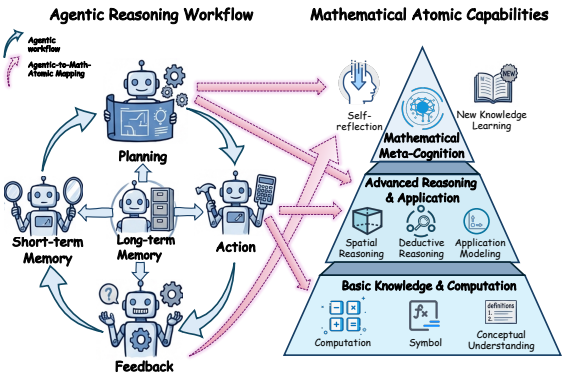


Figure 1: Illustration of the alignment between agentic behaviors and hierarchical mathematical atomic capabilities. This mapping enables interpretable, process-level evaluation of an LLM's agentic intelligence.

From both cognitive and structural perspectives, mathematical reasoning and agentic behavior exhibit a shared property: they can be decomposed into atomic and reusable units (Figure 1). In mathematics, solving a complex problem typically requires multiple such atomic capabilities across different levels (Zhang et al., 2025b; Kuang et al., 2025). For example, solving a geometry problem requires translating the natural language and visual diagrams into variables (Modeling), choosing a strategy (Planning), and performing calculations (Execution). This mirrors the agentic workflow, where high-level goals are realized through sequences of atomic behaviors. This atomistic perspective naturally motivates process-level evaluation: instead of judging a model solely by its final answer, we examine whether it performs the

[1] We will open-source all data and code after the anonymity review process ends.

right intermediate steps and how these steps interact throughout the reasoning process.

Based on this structural alignment, existing benchmarks are insufficient for evaluating the agentic potential of LLMs. First, most benchmarks emphasize final correctness, making it difficult to diagnose errors during problem-solving and whether the overall logic is correct (Xia et al., 2024). Second, current evaluations provide only limited coverage of mathematical capabilities (Liu et al., 2025c). While some datasets attempt to model more skills, they often cover a relatively narrow set of abilities or lack large-scale, systematic assessment (Zhang et al., 2025b; Lu et al., 2024b). More critically, existing benchmarks rarely establish alignment between mathematical atomic capabilities and agentic atomic behaviors, resulting in a lack of agentic-oriented evaluation with a mathematical perspective, hindering the understanding of which agentic capability is the bottleneck (Kuang et al., 2025).

To address these limitations, we introduce an interpretable process-level benchmark, Agentic-MathBench(AMB), that systematically evaluates the agentic capabilities of LLMs in mathematical reasoning. We first define a structured taxonomy of mathematical atomic capabilities and align them with core agentic functions: *Planning*, *Action*, and *Feedback*. Based on this, we design diverse evaluation tasks covering both multimodal and text-only scenarios. To support these tasks, we construct an automated pipeline that integrates large-scale problem collection, high-quality trajectory synthesis, and fine-grained annotation. Extensive experiments demonstrate that our benchmark reveals substantial differences in agentic abilities among models with similar end-to-end performance. Our main contributions are:

- We propose a process-level benchmark, that goes beyond final accuracy, enabling fine-grained diagnosis of the inherent agentic behaviors in LLMs.
- We introduce a structured taxonomy of mathematical atomic capabilities and systematically align with core agentic abilities.
- We design diverse planning, action, and feedback tasks, supported by an automatic data engineering pipeline with large-scale, high-quality data collection and annotations.
- Extensive experiments show that models exhibit interestingly different agentic profiles, highlighting the necessity of process-level evaluation for future agent development.

2 Related Work

2.1 Benchmarks for Mathematical Reasoning

Early benchmarks for mathematical reasoning predominantly evaluate models in simple question answering (Koncel-Kedziorski et al., 2016). Synthetic collections such as the DeepMind Mathematics dataset (Saxton et al., 2019) probe generalization on algebra, calculus, and related topics, while GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021) have become standard for grade-school reasoning. To move beyond elementary problems, competition-level benchmarks such as OlympiadBench (He et al., 2024) evaluate advanced reasoning, but typically still report aggregate accuracy. More recent work incorporates more structured skills, and process-level signals (Lu et al., 2024b; Sun et al., 2024; Wang et al., 2025a) benchmarks visual mathematical reasoning with figures and diagrams. GAUSS (Zhang et al., 2025b) organizes evaluation along structured skill dimensions to obtain interpretable profiles. However, evaluation remains driven by problem-level, with limited visibility into which atomic skills fail and how they interact (Ahn et al., 2024; Wang et al., 2025b; Forootani, 2025). Instead of defining new problem, we operationalize mathematical atomic capabilities and link them to agent behaviours, yielding an interpretable benchmark to LLM's agentic intelligence.

2.2 Agentic Mathematical Reasoning

In parallel with benchmark development, many works activate LLMs with agentic capabilities, including tools, memory, or multi-agent structures. Representative systems such as ToRA (Gou et al., 2023), MathChat (Wu et al., 2023), and MathAgent-PRER (Liao et al., 2024) use tool integration and explicit Planner–Reasoner–Executor–style decompositions to tackle challenging problems. Some recent researches focus on domain-specific tasks and formalize mathematical problem solving as pipelines. MM-Agent (Liu et al., 2025b) targets real-world modeling, and K-12 systems such as MathAgent (Yan et al., 2025) use mixtures of specialized agents for multimodal error detection. At the same time, Math-Shepherd (Wang et al., 2024b) and PRM800K-style process reward models (Lightman et al., 2023; Ma et al., 2025; Zhang et al., 2025c; Sun et al., 2025b) score individ-

ual reasoning steps and support reranking, reinforcement learning, and multi-agent debate frameworks (Zhang et al., 2025a). Overall, the role of agentic ability in mathematical reasoning is becoming increasingly prominent. However, evaluation remains outcome-centric. There is still no unified benchmark that formulates dedicated planning, action, and feedback tasks with fine-grained process-level metrics. AgenticMathBench is designed to fill this gap as an evaluation framework that can provide a multidimensional diagnosis of agentic ability from a mathematical perspective.

3 Method

3.1 Task Definition

Atomic thinking decomposes complex mathematical reasoning into a sequence of minimal, indivisible cognitive operations. This paradigm closely parallels the operational structure of agentic systems. Motivated by this alignment, we design our benchmark, AgenticMathBench, AMB, to integrate *mathematical atomic capabilities* with *agentic behaviors*, forming a unified interpretable evaluation framework that jointly covers a wide range of mathematical reasoning and agentic abilities.

3.1.1 Atomic Capability System

We first systematically establish the atomic capabilities required for mathematical reasoning. Drawing upon empirical studies (Zhang et al., 2025b; Kuang et al., 2025) and consulting with math research experts, we construct a hierarchical **Atomic System**:

- **Level 1: Foundational Concept and Calculation.** Focuses on basic knowledge comprehension and calculation execution ability, including *Symbol Recognition*, *Concept Understanding*, and *Calculation*.
- **Level 2: Advanced Reasoning and Application.** Addresses complex reasoning and application, comprising *Spatial Perception*, *Formalization*, *Deductive and Inductive Reason*, and *Mathematics Modeling*.
- **Level 3: Mathematical Meta-Cognitive.** Targets high-level meta-cognition, specifically *Theorem Application*, *Self-Reflection* and *New Knowledge Acquisition*.

These atomic capabilities can be naturally aligned with different agentic modules. Level 1 and 2 capabilities are primarily exercised during step-wise execution and thus correspond to *action* capability. In contrast, higher-level capabilities emphasize meta-cognition and are associated with *feedback*, facilitating learning and reflection from the current state. The *planning* capability operates at a global level, requiring a holistic understanding of the problem structure and the coordinated deployment of multiple atomic capabilities.

3.1.2 Agentic Task Formulation

Based on the interaction patterns between agentic modules and mathematical atomic capabilities, we design a set of evaluation tasks targeting three core agentic abilities: *planning*, *action*, and *feedback*. We do not consider *memory* task because memory is hard to map to specific mathematical atomic capabilities, and difficult to evaluate directly in a controlled and comparable way.

Planning Planning is formulated as an ordered decision sequence generation problem. Given an input problem with initial state s_0, the planner generates an ordered sequence

$$\pi = \big[(a_1, g_1), (a_2, g_2), \ldots, (a_T, g_T)\big], \quad (1)$$

where $a_t \in \mathcal{A}$ denotes the atomic capability, and g_t specifies the concrete sub-goal to be solved at step t. The sequence must satisfy causal and logical dependency constraints. At the planning level, we focus exclusively on whether the model identifies the correct atomic capabilities and decomposes the problem, while excluding execution correctness.

Feedback. Feedback is formalized as a state evaluation and correction. It analyzes partial or complete trajectories to guide subsequent actions:

$$f : (\text{problem}, \tau_{\leq t}) \rightarrow \{\text{status}, \text{type}, \text{loc}, \text{sugg}\} \quad (2)$$

This capability includes correctness judgment, error localization, and repair. At an advanced level, feedback may further extract transferable learning signals from failure cases and update the memory.

Action Action corresponds to executing the atomic sub-tasks specified by the planning. Unlike planning and feedback, which require multiple atomic capabilities comprehension, action focuses on the model's performance on decoupled, single-capability tasks. This design enables fine-grained assessment of each atomic capability in isolation.

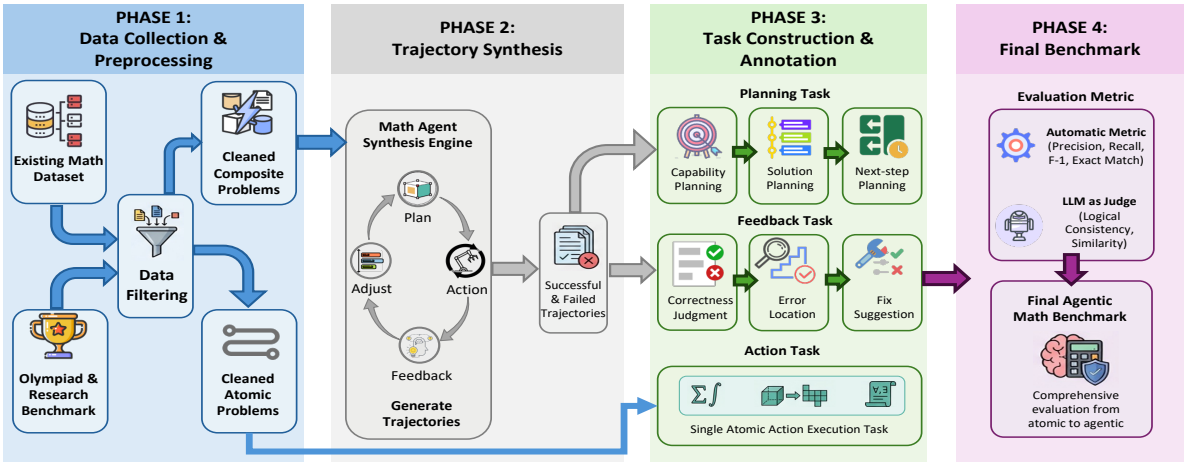


Figure 2: The overall benchmark construction pipeline of our AgenticMathBench.

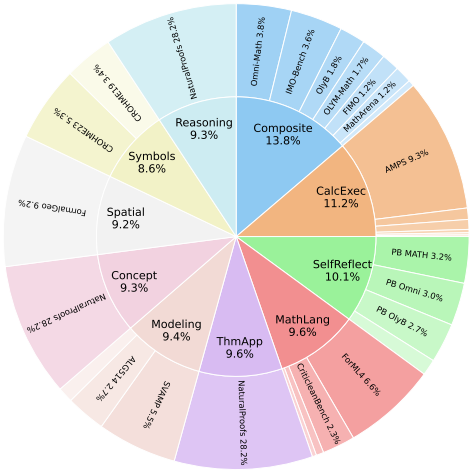


Figure 3: Data distribution of collected datasets. CalcExec: calculation, MathLang: formal mathematical language, Reasoning: Deductive and Inductive Reasoning, SelfReflect: self-reflection, ThmApp: theorem application, Composite: composite ability.

3.2 Benchmark Construction

3.2.1 Data Collection

We collect raw data from a large number of existing authoritative math benchmarks. By reviewing over 150 math benchmarks, including easier high school level and more challenging competition and research level difficulties, we filter and obtain 27 datasets covering various mathematical abilities. The data distribution is shown in Figure 3.

For single atomic abilities, we primarily filter existing benchmarks covering handwritten symbol recognition, elementary and advanced problem solving, geometry, and formal theorem proving, including CROHME (Xie et al., 2023), NaturalProofs (Welleck et al., 2021), LILA/AMPS (Mishra et al., 2022), FormalGeo (Zhang et al., 2023b), CriticLeanBench (Peng et al., 2025), FormL4 (Lu et al., 2024a), MiniF2F (Zheng et al., 2021), ProofNet (Azerbayev et al., 2023), and ProcessBench (Zheng et al., 2025). These datasets are rewritten to a unified schema, with a shared pipeline for filtering, de-duplication, and downsampling. Two abilities, forward reasoning and modeling transformation, require intermediate structures that are generated by an LLM from theorem proof pairs and algebraic problems, with only automatically validated instances retained. Details are given in Appendix A and C.

Composite problems are drawn from recent competition-level and Olympiad-style benchmarks, including AMO-Bench (An et al., 2025), IMO-Bench (Guo et al., 2025), OlymMATH (Sun et al., 2025a), Omni-MATH (Gao et al., 2024), MathArena (Balunović et al., 2025), FIMO (Liu et al., 2023), and OlympiadBench (He et al., 2024). Each problem is converted to the same unified format and annotated with a multi-label over the atomic abilities, trying to fully cover all capabilities. These composite instances underpin our planning and feedback tasks, and additional statistics are reported in Appendix A.

3.2.2 Trajectory Synthesis and Preprocessing

While single atomic Action tasks can be derived from raw data, directly annotating planning and feedback behaviors at the problem level is insufficient, as such annotations fail to reflect authentic step-wise reasoning and dynamic decision-making. To bridge this gap, we synthesize a dataset of high-quality mathematical trajectories following

Table 1: Metrics are abbreviated as: ExpAcc/SymAcc=CROHME-style expression/symbol accuracy, CAS-Eq=CAS-verified equivalence, Lean-Cmp=Lean compilation success rate, Lean-Align=semantic alignment, Judge(Cov/Cons)=LLM-judge coverage/consistency, Sim=semantic similarity. (Data cases are in Appendix E).

Task	#Num	Input	Target Output	Metric(s)
			Planning	
Capability Planning	221	T&I	Selected Capability set $\{a_i\}$	P/R/F1, EM
Solution Planning	237	T&I	Ordered plan $(a_1, g_1) \rightarrow \cdots \rightarrow (a_T, g_T)$	Judge(Cov/Cons)
Next-step Planning	609	T&I	Next step (a_{t+1}, g_{t+1})	Acc(a_{t+1}), Sim(g_{t+1})
			Action	
Symbol Recognition	1,022	T&I	Canonical LaTeX expression	ExpRate, SymRate
Concept Understanding	1,099	T	Required Concept set	P/R/F1
Computation Execution	1,328	T	Numeric/symbolic calculation result	EM (num), CAS-Eq (sym)
Spatial Reasoning	1,139	T&I	Spatial relation predicate set	P/R/F1 (+ Acc)
Formal Language	1,140	T	Lean4 theorem declaration	Lean-Cmp, Lean-Align
Deductive and Inductive	1,096	T	Ordered proof-outline steps	Judge(Cov/Cons)
Modeling Transformation	1,106	T	Variables + constraints + objective	F1(vars), Judge (cons,obj)
Theorem Application	1,135	T	Theorem-selection & instantiation trace	Judge(Cov/Cons)
			Feedback	
Correctness Judgment	506	T&I	Binary label (correct/incorrect)	Acc
Error Localization	347	T&I	First-error step index + error type	Acc(step), Acc(type)
Fix Suggestion	280	T&I	Repair proposal (a_{t+1}, g_{t+1}) + rationale	Judge(rationale), Judge(Align)

a canonical *plan–action–feedback* agent paradigm.

We implement a mathematical agent that reasons explicitly over atomic capabilities. Given a problem, the agent first generates a global plan specifying the required atomic capabilities, corresponding sub-tasks, and their execution order. It then iteratively executes each step using the selected atomic capability, evaluates intermediate results, and dynamically adjusts subsequent actions based on feedback. Final answers and trajectory quality are automatically verified by LLM and human sampling inspection using reference solutions and consistency checks. This process yields a rich corpus of both successful and failed trajectories, which enables reliable extraction of planning and feedback annotations in subsequent stages.

3.3 Task Annotation and Evaluation

After obtaining the filtered and classified math data and the synthesis of trajectory data, we design each task and evaluation metrics, and complete the construction and annotation, as shown in Table 1.

Planning. We evaluate *Planning* as the ability to (i) select the atomic capabilities required to solve a problem, (ii) synthesize a coherent solution roadmap, and (iii) dynamically generate the next step under executed trajectories. Concretely, given a problem and its synthesized ground-truth trajectory, we extract: a *ground truth capability set*, an *ordered solution sequence* with steps, atomic capabilities, and sub-goals, and *next-step targets* obtained by truncating trajectories at completion ratios 20%, 50%, 80%. We score capability selection with metrics P/R/F1, while full-plan is judged by a strong LLM for step coverage, sub-goal and logical consistency against the reference plan. Next-step planning is evaluated by capability accuracy and semantic similarity of the predicted subgoal. Details of statistics are in Appendix B.

Action. We operationalize *Action* as directly executable atomic mathematical operations, covering symbol and concept understanding, calculation and reasoning, and structured transformations including formalization, modeling, and theorem application. For tasks such as symbol recognition and calculation, annotations are obtained mainly via lightweight rewriting and normalization from existing datasets into standard formats. For *process-structured* targets that require abstraction (e.g., forward-reasoning outlines, modeling templates), we prompt a strong LLM to rewrite original solutions into our intermediate representations, followed by normalization and spot-checking, with full prompts and methods in Appendix C. Evalua-

Table 2: Planning performance of models across capability planning, solution planning, and next-step planning. The results on multimodal tasks are in Table 12.

Model	Capability Planning			Solution Planning				Next-step Planning		
	Pre	Rec	F1	Logic	Sub-goal	Step	Overall	Capability	Subgoal	Overall
General Open-sourced Models										
Llama-4-Scout-17B	47.6	**90.5**	60.8	37.5	38.0	33.0	36.2	27.1	39.9	33.5
Llama-4-Maverick-17B	**54.7**	83.4	**64.1**	44.5	44.5	44.0	44.3	43.8	**48.5**	46.1
DeepSeek-V3.2	48.2	59.3	52.1	**77.1**	**77.7**	**73.5**	**76.1**	**45.8**	46.8	**46.3**
Qwen3-32B	43.5	44.0	41.5	40.0	40.4	39.1	40.2	34.1	38.5	36.3
Qwen3-235B-A22B-Thinking	46.1	77.4	54.1	7.4	7.2	6.9	7.2	26.9	30.4	28.7
GLM-4.7	47.7	80.3	56.0	8.5	8.1	8.2	8.3	27.6	30.3	29.0
Math Open-sourced Models										
Qwen2.5-Math-72B-Instruct	**50.6**	**97.5**	**65.4**	11.0	12.2	10.4	11.2	23.7	**36.8**	**30.2**
Deepseek-Math-V2	46.6	82.4	57.9	**14.1**	**13.7**	**13.0**	**13.6**	**31.6**	27.5	29.5
Commercial Models										
GPT-5.2	**76.0**	75.4	74.0	**86.3**	**85.4**	**83.6**	**85.1**	39.2	42.8	41.0
Claude-4.5-Sonnet-Thinking	70.6	**93.0**	**78.8**	85.4	**85.4**	81.3	84.0	**48.5**	**50.3**	**49.4**
Gemini-3-Pro	65.3	86.9	72.9	67.7	67.3	63.6	66.2	30.2	38.2	34.2

tion follows the target type: exact match/structure-aware accuracy for recognition and transcription, set-based scores for concept/relation extraction, Lean compilation and automated semantic alignment for formalization, and LLM-judge scoring for plan/trace alignment when equivalence cannot be reliably decided by rules (details in Appendix D).

Feedback. We evaluate *Feedback* as process judgment and correction over full or partial trajectories. From mixed correct/incorrect trajectories, we construct: (i) *correctness judgment* as a binary classification; (ii) *error localization* by labeling the earliest erroneous step and its error type (annotations generated with a strong LLM and validated via sampling); and (iii) *fix suggestion* by truncating immediately before an modified step and asking for a repair action (capability + subgoal) consistent with a valid correction strategy. Accordingly, we evaluate correctness judgment with accuracy, error localization with step-index accuracy and error-type accuracy, and fix suggestion with the modified reason consistency, and the effectiveness of the modified strategy includes the next step's capabilities and sub-task by LLM.

4 Experiment

4.1 Experimental Settings

We evaluate 3 families of models: (i) general-purpose open models, (ii) math-specialized open models, (iii) closed models. The general models are Llama-4-Scout-17B-16E-Instruct and Llama-4-Maverick-17B-128E-Instruct (Meta AI, 2025), DeepSeek-V3.2 (Liu et al., 2025a), Qwen3-32B and Qwen3-235B-A22B-Thinking-2507 (Qwen Team, 2025a), and GLM-4.7 (Zhipu AI, 2025). For multimodal open models, we consider Qwen3-VL-32B-Instruct and Qwen3-VL-235B-A22B-Thinking (Qwen Team, 2025b), InternVL3.5-38B-Instruct and InternVL3.5-241B-A28B-Instruct (Wang et al., 2025c), and Deepseek-VL2 (Wu et al., 2024). The math-specialized models comprise Qwen2.5-Math-72B-Instruct (Yang et al., 2024) and DeepSeek-Math-V2 (Shao et al., 2025). For comparison with the strongest proprietary models, we include GPT-5.2 (OpenAI, 2025), Claude-Sonnet-4.5-thinking (Anthropic, 2025), Gemini-3-Pro(-Preview) (Google DeepMind, 2025). Details about the inference hyperparameters are provided in Appendix F.1.

4.2 Results of Planning Ability

We evaluate planning ability and Table 2 reports performance. Closed-source models consistently outperform open-source models in capability planning, with Claude-4.5 achieving the highest F1 scores. For full solution planning, several strong open-source models (e.g., DeepSeek-V3.2) achieve results comparable to or even exceeding some closed-source models. Additionally, compared to solution-level planning, nearly all models exhibit a significant performance drop in next-step planning. This gap indicates that while many models can reason about plans offline, they struggle to perform *dynamic planning*. Such degradation exposes

Table 3: Feedback performance of models, evaluating correctness judgement, error localization, and fix suggestion. The results on multimodal tasks are in Table 13.

Model	Correctness	Error Localization			Fix Suggestion	
	Accuracy	Step Judge	Type Classification	Overall	Reason Consistency	Overall
General Open-source Models						
Llama-4-Scout-17B	50.0	30.8	30.8	30.8	62.2	28.0
Llama-4-Maverick-17B	44.4	30.8	7.7	19.2	53.9	24.3
DeepSeek-V3.2	49.0	**54.2**	**66.8**	**60.5**	**70.1**	**31.5**
Qwen3-32B	56.1	44.6	58.1	51.4	54.9	24.7
Qwen3-235B-Thinking	**56.5**	28.4	16.2	22.3	5.2	2.4
GLM-4.7	47.3	13.8	18.9	16.3	28.8	13.0
Math Open-source Models						
Qwen2.5-Math-72B	**51.7**	19.5	12.3	15.9	**18.4**	**11.4**
Deepseek-Math-V2	48.8	**39.8**	**29.3**	**34.6**	18.2	9.5
Commercial Models						
GPT-5.2	49.3	**50.9**	43.4	**47.2**	**77.2**	**34.7**
Claude-4.5	46.3	49.7	42.8	46.3	72.9	32.8
Gemini-3-Pro	**52.8**	36.4	**50.0**	43.2	63.5	28.6

a core weakness in current LLMs, as adaptive next-step decision making is central to agentic reasoning in real execution environments. Overall, our benchmark exposes failure modes that are entirely hidden under end-to-end accuracy, which provides guidance for future agentic development, emphasizing the need for training on step-wise planning rather than static reasoning alone.

4.3 Results of Feedback Ability

Results in Table 3 reflect the abilities to assess outcomes, diagnose failures, and propose repairs. Most models achieve moderate accuracy in distinguishing correct from incorrect trajectories. Even commercial models remain below 65% accuracy, indicating that **outcome assessment is non-trivial when reasoning traces are complex**. Error localization proves substantially more challenging. While some models achieve reasonable accuracy on locating the error step, sometimes models cannot accurately determine the specific cause. For the highest-level feedback capability, Fix suggestion shows significantly weaker performance. Models only provide natural language explanations for why we need to modify, but they fail to translate feedback into **actionable next-step suggestions**. Our benchmark exposes that current models can partially assess and localize errors, but struggle to convert feedback into effective corrective actions.

4.4 Results of Action Ability

Table 4 evaluates models on *action-level* execution. Across all tasks, commercial models consistently

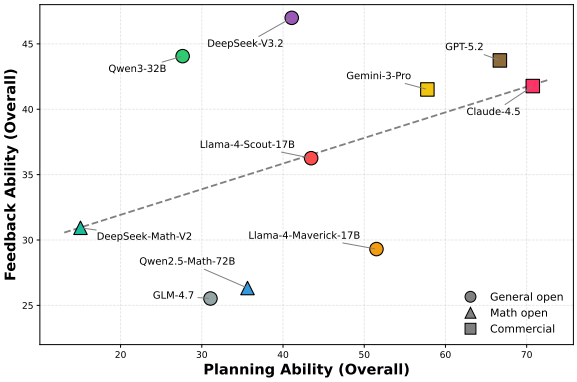


Figure 4: This figure illustrates the different performance of models on planning and feedback tasks.

achieve strong and balanced performance across, particularly in formal mathematical language usage and mathematical modeling. In contrast, while several open-source models perform competitively on calculation and reasoning, they often struggle with conceptual understanding, and degrade significantly on tasks requiring structured action execution, such as modeling. This suggests that training focused on answer correctness alone is insufficient for robust action-level competence. Our benchmark exposes a critical bottleneck, revealing that even **models with strong planning and feedback capabilities can fail at precise, structured mathematical execution**.

4.5 Analysis of Planning vs Feedback Ability

Figure 4 visualizes the overall planning ability against feedback ability. While commercial

Table 4: Action performance of models across execution-oriented mathematical capabilities. The results on multimodal tasks are in Table 14.

Model	Calc.	Concept		Formal Lang.		Fwd. Reason	Modeling			Theorem
	EM	Set-F1	Acc	Compile	Align	Prec.	Var-F1	Constr	Obj	Pass@k
General Open-source Models										
Llama-4-Scout-17B	55.0	10.0	70.0	45.0	30.0	29.0	**85.7**	76.8	85.0	50.0
Llama-4-Maverick-17B	50.0	29.1	85.0	70.0	65.0	53.3	77.7	78.7	**100.0**	65.0
DeepSeek-V3.2	80.0	**44.5**	90.0	**100.0**	90.0	91.8	79.3	**83.8**	95.0	**100.0**
Qwen3-32B	80.0	41.8	**95.0**	**100.0**	**100.0**	40.4	83.3	72.4	**100.0**	85.0
Qwen3-235B-A22B-Thinking	**85.0**	39.7	90.0	**100.0**	95.0	**92.0**	72.3	71.1	**100.0**	**100.0**
GLM-4.7	**85.0**	28.3	80.0	25.0	25.0	57.4	74.3	74.9	**100.0**	95.0
Math Open-source Models										
Qwen2.5-Math-72B-Instruct	50.0	**10.0**	**75.0**	5.0	5.0	9.0	35.7	19.9	**80.0**	15.0
DeepSeek-Math-V2	**75.0**	5.6	70.0	**25.0**	**25.0**	**35.4**	**76.4**	**78.1**	95.0	**25.0**
Commercial Models										
GPT-5.2	77.4	51.9	99.8	**100.0**	**93.8**	92.6	90.9	**88.9**	98.6	98.6
Claude-4.5-Sonnet-Thinking	83.5	52.7	99.8	99.8	84.4	93.1	86.6	81.7	**100.0**	99.0
Gemini-3-Pro	**88.7**	**56.0**	**72.0**	**100.0**	19.3	**94.7**	**92.2**	84.7	99.7	**99.0**

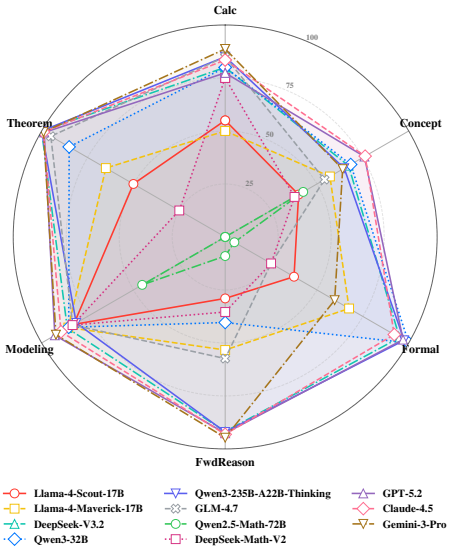


Figure 5: Distribution of performance of different models on action tasks.

models such as Claude-4.5-sonnet achieve high scores in both planning and feedback, many open-source models exhibit a noticeable divergence. DeepSeek-V3.2 attains relatively high feedback performance despite moderate planning scores. Math-specialized open models such as Qwen2.5-Math-72B tend to cluster in the lower-left quadrant, highlighting that training solely on problem-solving datasets does not guarantee comprehensive agentic abilities. This analysis demonstrates that our benchmark can disentangle different facets of agentic competence, providing guidance for future model development to balance agentic adaptability.

4.6 Analysis of Action Performance

Figure 5 shows the overall performance on different action tasks. Across the six text-only atomic dimensions, frontier closed models and large reasoning-enhanced models form a high, relatively balanced envelope, while several smaller open models exhibit "spiky" profiles with pronounced weak axes. CALC is broadly near-saturated, whereas **CONCEPT is the primary bottleneck** with the largest variance. Overall, high scores typically co-occur with balanced fundamentals, consistent with these tasks requiring coordinated competence, concept alignment → formalization → complex reasoning.

5 Conclusion

We propose AgenticMathBench for evaluating the *agentic mathematical capabilities* of LLMs. Our key innovation lies in decomposing mathematical reasoning into a structured taxonomy of atomic capabilities and aligning them with the agentic abilities of planning, action, and feedback. Our evaluation targets not only *whether* a model solves a problem, but specifically assesses its potential regarding *how* it plans, executes, reflects, and repairs its solution in both textual and multimodal contexts. Empirically, our results reveal substantial disparities in agentic proficiency among models that appear similar under final-answer accuracy metrics. These findings highlight the interpretable evaluation, paving the way for better eliciting and utilizing the latent agentic abilities of LLMs.

Limitations

While we align agentic functions with mathematical atomic capabilities, the taxonomy and task instantiations may not cover all forms of mathematical agency (e.g., long-horizon memory, or interactive theorem proving beyond single-statement formalization). In addition, for tasks with a high degree of freedom in feedback and planning, we can consider designing more evaluation dimensions.

Ethics Statement

All datasets used in this work are collected from publicly available and open-source resources. We apply the data that complies with the original licenses and terms of use. Our benchmark is intended for evaluating mathematical reasoning behaviors and does not involve sensitive personal data. We have no potential risks, and we document data sources and licensing.

LLM Usage Statement

We use large language models to assist with non-substantive writing support, including grammatical checking, wording refinement, and improving clarity of exposition, as well as ideas for figure/table presentation. All technical content, benchmark design decisions, experimental results, and claims were conducted by the human authors. Any LLM-generated suggestions were reviewed, edited, and validated by the authors to ensure correctness, and compliance with publication norms.

References

Josh Achiam and 1 others. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the ACL (Student Research Workshop)*, pages 225–237.

Shengnan An, Xunliang Cai, Xuezhi Cao, Xiaoyu Li, Yehao Lin, Junlin Liu, Xinxuan Lv, Dan Ma, Xuanlin Wang, Ziwen Wang, and Shuang Zhou. 2025. AMO-bench: Large language models still struggle in high school math competitions. *arXiv preprint arXiv:2510.26768*.

Anthropic. 2025. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5.

Zhangir Azerbayev, Bartosz Piotrowski, Hailey Schoelkopf, Edward W. Ayers, Dragomir Radev, and Jeremy Avigad. 2023. Proofnet: Autoformalizing and formally proving undergraduate-level mathematics. *arXiv preprint arXiv:2302.12433*.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating LLMs on uncontaminated math competitions. *NeurIPS Datasets and Benchmarks Track*.

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 problems. In *Advances in Neural Information Processing Systems*.

Ali Forootani. 2025. A survey on mathematical reasoning and optimization with large language models. *arXiv preprint arXiv:2503.17726*.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, and 1 others. 2024. Omni-MATH: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Google DeepMind. 2025. A new era of intelligence with gemini 3. https://blog.google/products/gemini/gemini-3/.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.

Zining Guo, Zhi Huang, Yuzhuo Zhang, and 1 others. 2025. IMO-bench: Towards robust mathematical reasoning. Dataset available at https://imobench.github.io/.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 2024. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Advances in Neural Information Processing Systems*.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of NAACL-HLT*, pages 1152–1157.

Jiayi Kuang, Haojing Huang, Yinghui Li, Xinnian Liang, Zhikun Xu, Yangning Li, Xiaoyu Tan, Chao Qu, Meishan Zhang, Ying Shen, and Philip S. Yu. 2025. Atomic thinking of llms: Decoupling and exploring mathematical reasoning abilities. *Preprint*, arXiv:2509.25725.

Nate Kushman, Luke Zettlemoyer, Regina Barzilay, and Yoav Artzi. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Haoran Liao, Qinyi Du, Shaohua Hu, Hao He, Yanyan Xu, Jidong Tian, and Yaohui Jin. 2024. Modeling complex mathematical reasoning via large language model based mathagent. In *Proceedings of ICLR*.

Sam Lightman, Aitor Lewkowycz, Alex Slone, and 1 others. 2023. Let's verify step by step: PRM800K, a process supervision dataset for math reasoning. *arXiv preprint arXiv:2305.20050*.

Aixin Liu and 1 others. 2025a. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2505.13655*.

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. FIMO: A challenge formal dataset for automated theorem proving. *arXiv preprint arXiv:2309.04295*.

Fan Liu, Zherui Yang, Cancheng Liu, Tianrui Song, Xiaofeng Gao, and Hao Liu. 2025b. MM-Agent: LLM as agents for real-world mathematical modeling problem. *arXiv preprint arXiv:2505.14148*.

Tianqiao Liu, Zui Chen, Zhensheng Fang, Weiqi Luo, Mi Tian, and Zitao Liu. 2025c. Matheval: A comprehensive benchmark for evaluating large language models on mathematical reasoning capabilities. *Frontiers of Digital Education*, 2(16).

Jianqiao Lu, Yingjia Wan, Zhengying Liu, Yinya Huang, Jing Xiong, Chengwu Liu, Jianhao Shen, Hui Jin, Jipeng Zhang, Haiming Wang, and 1 others. 2024a. Process-driven autoformalization in lean 4. *arXiv preprint arXiv:2406.01940*.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024b. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *Proceedings of ICLR*.

Yunsheng Ma, Beidi Chen, Fuchun Yang, and 1 others. 2025. What are step-level reward models? a framework for structured reasoning supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Meta AI. 2025. The llama 4 herd: The beginning of a new era of natively multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Øyvind Tafjord, Ashish Sabharwal, Peter Clark, and Ashwin Kalyan. 2022. LILA: A unified benchmark for mathematical reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

OpenAI. 2025. Introducing GPT-5.2. https://openai.com/index/introducing-gpt-5-2/.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094.

Zheng Peng, Zhengying Liu, Jianqiao Lu, Haiming Wang, Wei Ju, Jianhao Shen, and 1 others. 2025. Criticlean: Critic-guided reinforcement learning for mathematical autoformalization in lean 4. *arXiv preprint arXiv:2507.06181*. Introduces the CriticLeanBench benchmark.

Qwen Team. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.01703*.

Qwen Team. 2025b. Qwen3-VL technical report. *arXiv preprint arXiv:2509.01898*.

Sandip Sarkar, Dipankar Das, Partha Pakray, and David Eduardo Pinto-Avendaño. 2023. Math word problem solving: Operator and template techniques with multi-head attention. *Computación y Sistemas*, 27(4):1075–1088.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations (ICLR)*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhihong Shao and 1 others. 2025. Deepseekmath-v2: Towards self-verifiable mathematical reasoning. *arXiv preprint arXiv:2511.22570*.

Shuming Shi, Danqing Huang, Chin-Yew Lin, and Wei-Ying Ma. 2015. Datasets for math word problem solving (sigmadolphin project). Technical report, Microsoft Research; includes the Dolphin1878 dataset.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Haoxiang Sun, Yingqian Min, Zhipeng Chen, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, Lei Fang, and Ji-Rong Wen. 2025a. Challenging the boundaries of reasoning: An olympiad-level math benchmark for large language models. *arXiv preprint arXiv:2503.21380*.

Kai Sun, Yushi Bai, Ji Qi, Lei Hou, and Juanzi Li. 2024. Mm-math: Advancing multimodal math evaluation with process evaluation and fine-grained classification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1358–1375.

Xiaoyu Sun, Peng Li, and 1 others. 2025b. Retrieval-augmented process reward model for mathematical reasoning. *arXiv preprint arXiv:2502.14361*.

Bintao Tang, Xin Yang, Yuhao Wang, Zixuan Qiu, Zimo Ji, and Wenyuan Jiang. 2025. INTEGRAL-BENCH: Benchmarking LLMs with definite integral problems. *NeurIPS Datasets and Benchmarks Track*. ArXiv:2507.21130.

Shyam Upadhyay and Kai-Wei Chang. 2017. Annotating derivations: A new evaluation strategy and dataset for algebra word problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 916–926.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18:186345.

Peijie Wang, Zhong-Zhi Li, Fei Yin, Dekang Ran, and Cheng-Lin Liu. 2025a. Mv-math: Evaluating multimodal math reasoning in multi-visual contexts. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19541–19551.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yixuan Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of ACL*.

Peng-Yuan Wang, Tian-Shuo Liu, Chenyang Wang, Yi-Di Wang, Shu Yan, Cheng-Xing Jia, Xu-Hui Liu, Xin-Wei Chen, Jia-Cheng Xu, Ziniu Li, and Yang Yu. 2025b. A survey on large language models for mathematical reasoning. *ACM Computing Surveys*. Early access.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, and 1 others. 2025c. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. 2021. Naturalproofs: Mathematical theorem proving in natural language. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.

Yiran Wu, Feiran Jia, Shaokun Zhang, Hangyu Li, Erkang Zhu, Yue Wang, Yin Tat Lee, Richard Peng, Qingyun Wu, and Chi Wang. 2023. Mathchat: Converse to tackle challenging math problems with LLM agents. *arXiv preprint arXiv:2306.01337*.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, and 1 others. 2024. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2024. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*.

Yejing Xie, Harold Mouchère, Christian Viard-Gaudin, Joan-Andreu Sánchez, Andrés Garz, and 1 others. 2023. Icdar 2023 CROHME: Competition on recognition of handwritten mathematical expressions. In *Proceedings of the 17th International Conference on Document Analysis and Recognition (ICDAR)*.

Yibo Yan, Shen Wang, Jiahao Huo, Philip S. Yu, Xuming Hu, and Qingsong Wen. 2025. Mathagent: Leveraging a mixture-of-math-agent framework for real-world multimodal mathematical error detection. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Industry Track)*.

An Yang and 1 others. 2024. Qwen2.5-math: A family of finetuned LLMs for mathematical reasoning. *arXiv preprint arXiv:2409.12122*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. ReAct: Synergizing reasoning and acting in language

models. In *International Conference on Learning Representations (ICLR)*.

Beichen Zhang, Kun Zhou, Xilin Wei, Wayne Xin Zhao, Jing Sha, Shijin Wang, and Ji-Rong Wen. 2023a. Evaluating and improving tool-augmented computation-intensive math reasoning. *NeurIPS Datasets and Benchmarks Track*. ArXiv:2306.02408.

Shuo Zhang, Bowen Li, Tianyu Sun, and 1 others. 2025a. Multi-agent debate for fine-grained reasoning in math. In *Findings of ACL*.

Xiaokai Zhang, Na Zhu, Yiming He, Jia Zou, Qike Huang, Xiaoxiao Jin, Yanjun Guo, Chenyang Mao, Yang Li, Zhe Zhu, and 1 others. 2023b. FormalGeo: An extensible formalized framework for olympiad geometric problem solving. *arXiv preprint arXiv:2310.18021*.

Yue Zhang, Jiaxin Zhang, Qiuyu Ren, Tahsin Saffat, Xiaoxuan Liu, Zitong Yang, Banghua Zhu, and Yi Ma. 2025b. GAUSS: Benchmarking structured mathematical skills for large language models. *arXiv preprint arXiv:2509.18122*.

Zhen Zhang, Huajie Chen, and 1 others. 2025c. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Processbench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2021. Minif2f: A cross-system benchmark for formal olympiad-level mathematics. *arXiv preprint arXiv:2109.00110*.

Zhipu AI. 2025. Glm-4.7: Advancing the coding capability of general large models. https://zhipuai.cn.

A Dataset Sources and Curation

This appendix provides additional details on the datasets underlying AgenticMathBench. We first give an overview of all external sources and their role in our atomic abilities, then describe how we curate the atomic split from existing benchmarks, and finally explain how we select and annotate competition problems for the composite split.

A.1 Source Datasets Overview

AgenticMathBench is constructed by systematically reorganizing a wide range of existing mathematical benchmarks. For the nine atomic abilities we draw from handwritten expression recognition, natural mathematical text, symbolic and numeric problem sets, formal proof corpora, algebra word problems, and process-level error annotations. For the composite split we rely on recent Olympiad and competition benchmarks that emphasize high-level problem solving.

Table 5 summarizes, for each atomic ability, the number of curated examples and the external datasets from which they are derived. Most abilities are supported by multiple sources, which reduces the risk of overfitting to the quirks of any single benchmark. The composite split reuses only competition-style benchmarks and is discussed in more detail in Section A.3; Table 6 provides a qualitative view of how different competition benchmarks exercise the atomic abilities.

A.2 Curated Atomic Abilities from Existing Benchmarks

Seven of the nine atomic abilities—symbol recognition, concept understanding, computation execution, spatial reasoning, formal mathematical language, theorem application, and self-reflection—are obtained by curating and reorganizing existing datasets, without using LLM-based rewriting. All such examples are converted into a unified schema with a question field, an answer field, and minimal metadata (source, split, and ability tag). We apply light de-duplication by normalizing and exact-matching the question text, discard instances that are malformed or fall outside the intended ability, and then sample to achieve the balanced counts in Table 5.

Symbol recognition. For symbol recognition we reuse handwritten mathematical expression data from the CROHME 2019 and 2023 competitions (Xie et al., 2023). We keep only single-expression instances with complete LaTeX annotations, remove multi-line or multi-equation layouts, and normalize the LaTeX strings to a canonical form (e.g., stripping stylistic commands and enforcing a consistent macro set). Each example is then represented as an image and its target LaTeX, testing an agent's ability to recognize symbolic structure from visual input.

Concept understanding and theorem application. Concept understanding and theorem application are built from the NaturalProofs corpus (Welleck et al., 2021), which contains mathematical statements and accompanying natural-language proofs. We select statements that are self-

Table 5: Statistics of the sources of the 8 atomic abilities in AgenticMathBench.

Atomic ability	Data sources
Symbol recognition	CROHME19/23 (Xie et al., 2023)
Concept understanding	NaturalProofs (Welleck et al., 2021)
Computation execution	LILA/AMPS (Mishra et al., 2022), CARP (Zhang et al., 2023a), DeepMind Mathematics (Saxton et al., 2019), Dolphin1878 (Shi et al., 2015), operator–template MWP data (Sarkar et al., 2023), INTEGRALBENCH (Tang et al., 2025)
Spatial reasoning	FormalGeo v2 (Zhang et al., 2023b)
Formal mathematical language	CriticLeanBench (Peng et al., 2025), FormL4 (Lu et al., 2024a), MiniF2F (Zheng et al., 2021), ProofNet (Azerbayev et al., 2023)
Forward reasoning	NaturalProofs (Welleck et al., 2021)
Modeling transformation	ALG514 (Kushman et al., 2014), DRAW-1K (Upadhyay and Chang, 2017), SVAMP (Patel et al., 2021)
Theorem application	NaturalProofs (Welleck et al., 2021)

contained and of moderate length, and we derive two complementary views. For concept understanding, we construct short question–answer pairs that probe definitions, assumptions, or immediate consequences of a statement. For theorem application, we extract instances where a theorem is invoked within a proof and reformulate them as problems that require identifying or applying the appropriate result. In both cases we retain only items whose input can be understood without external context beyond the provided statement.

Computation execution. The computation execution ability targets symbolic and numeric calculation. We combine several sources: the LILA/AMPS unified benchmark (Mishra et al., 2022), CARP for computation-intensive algebra reasoning (Zhang et al., 2023a), the DeepMind Mathematics dataset (Saxton et al., 2019), the Dolphin1878 family of math word problems (Shi et al., 2015), operator–template based MWP data that emphasize arithmetic operations (Sarkar et al., 2023), and definite integral problems from INTEGRALBENCH (Tang et al., 2025). From each source we retain only problems whose main requirement is to carry out a well-specified calculation or symbolic manipulation, and we convert them to a short-answer format with a normalized numeric or algebraic target.

Spatial reasoning. Spatial reasoning is supported by geometry problems from the FormalGeo framework (Zhang et al., 2023b). We focus on Euclidean geometry questions with a clearly defined diagram and goal, such as proving a relation among lengths or angles. Problems that require extensive combinatorial or algebraic reasoning beyond geometry are excluded. We represent each instance by a textual description of the configuration and the target statement, leaving explicit diagram generation to downstream tasks.

Formal mathematical language. For the formal mathematical language ability we use Lean-based formalization corpora, including CriticLeanBench (Peng et al., 2025), FormL4 (Lu et al., 2024a), MiniF2F (Zheng et al., 2021), and ProofNet (Azerbayev et al., 2023). We extract parallel pairs of natural-language text (e.g., theorem statements or proof sketches) and corresponding Lean fragments, as well as small checking tasks such as filling in missing arguments or verifying that a proposed formal statement matches its natural-language version. All examples are normalized to a simple input–output format where the answer is either a Lean expression or a discrete decision about the correctness of a formalization.

The two remaining atomic abilities—forward reasoning and modeling transformation—require substantial restructuring of the original problems and are therefore constructed with LLM-based generation pipelines. Their data construction procedures are described in detail in Appendix C.

A.3 Composite Competition Problems and Ability Labels

Composite problems are drawn from recent Olympiad and competition benchmarks and are defined as questions that require a non-trivial combination of several atomic abilities. We use

seven sources: AMO-Bench (An et al., 2025), IMO-Bench (Guo et al., 2025), OlymMATH (Sun et al., 2025a), Omni-MATH (Gao et al., 2024), MathArena (Balunović et al., 2025), FIMO (Liu et al., 2023), and OlympiadBench (He et al., 2024). From each benchmark we start from the official training or evaluation splits, remove duplicated or near-duplicated problems across datasets, and discard items that are purely descriptive, non-mathematical, or excessively dependent on external context. Multi-part problems are retained only when the parts form a coherent whole that can be treated as a single composite question; otherwise they are split or dropped depending on the extent of entanglement.

Each retained problem is then annotated with a multi-label vector over the nine atomic abilities. Annotators are given concise guidelines for each ability (e.g., when a problem should be considered to involve spatial reasoning or formal mathematical language) and are asked to mark all abilities that are genuinely required for a complete solution rather than merely mentioned. Initial labels are assigned independently by two annotators; disagreements are resolved through discussion, and problems for which consensus cannot be reached are excluded from the benchmark. This process yields 1,627 composite problems with reliable ability annotations.

To provide a high-level view of how the competition benchmarks exercise different abilities, Table 6 reports a qualitative coverage matrix. We mark an ability as ✓ when it is a primary focus of many problems in the corresponding benchmark, as ○ when it appears in a non-trivial but secondary role, and as × when it is rarely or never required. These labels are not used in evaluation but help characterize the diversity of composite problems across sources.

B Additional Details of Data Statistics

Table 7 presents the task volume distribution of planning/feedback tasks across text-only and multimodal inputs, showing the total number of tasks for each subtask (e.g., 221 for Atomic Ability Selection in total).

To analyze the capability coverage of agentic tasks, we first evaluate the atomic ability distribution across text-only and multimodal settings for the planning task 1 "Capability Planning". Table 8 summarizes the coverage statistics of core atomic abilities in two scenarios: the first corresponds to text-only planning tasks, while the second corresponds to multimodal planning tasks.

We further analyze the trajectory statistics and capability usage of Planning Task 2 "Solution Planning", which also includes text-only and multimodal scenarios. Table 9 summarizes the trajectory details, step distribution, and capability usage frequency, complementing the capability coverage analysis of Planning Task 1.

C LLM-based Generation Pipelines

C.1 Modeling Conversion

The modeling conversion subset is built on top of classical algebra word-problem datasets Alg514 (Kushman et al., 2014), DRAW-1K (Upadhyay and Chang, 2017) and SVAMP (Patel et al., 2021). We keep only the original question text as the question field and perform light de-duplication by normalizing and exact-matching this field. On top of these questions, we run a three-stage LLM pipeline, and the final verified JSON model becomes the answer field in our benchmark.

Stage 1: Modelability filtering. The first stage discards problems that cannot be reasonably captured by a small algebraic model. For each question, an LLM is prompted as a binary classifier that must answer strictly "true" or "false" to the following decision: whether the problem can be represented using a small number of variables and algebraic constraints without actually solving it. Only problems classified as modelable are passed to the next stage.

Stage 2: JSON model generation. In the second stage, a separate LLM is instructed to act as a mathematical modeler and to convert each remaining question into a minimal algebraic model. The model is asked to output a JSON object with three fields named variables, constraints and objective. Variables are described by short names and natural-language descriptions, constraints are written as algebraic equalities or inequalities over these variables using simple operators and optional domain restrictions, and the objective briefly states what quantity should be solved for or compared. The prompt explicitly forbids inventing information that is not present in the text and forbids performing any numeric computation. We parse the output as JSON and discard responses that are not syntactically valid.

Atomic ability	AMO-Bench	IMO-Bench	OlymMATH	Omni-MATH	MathArena	FIMO	OlympiadBench
Symbol recognition	×	×	×	×	×	×	✓
Concept understanding	○	✓	○	○	○	✓	○
Computation execution	✓	✓	✓	✓	✓	×	✓
Spatial reasoning	○	○	○	○	○	○	✓
Formal mathematical language	×	×	×	×	×	✓	×
Forward reasoning	○	✓	○	○	○	✓	○
Proof construction	○	○	○	○	○	○	○
Modeling transformation	✓	✓	✓	✓	✓	✓	✓
Theorem application	○	✓	○	○	○	✓	○

Table 6: Qualitative coverage of atomic abilities across the competition benchmarks used to construct the composite split. Each cell indicates how frequently the benchmark exercises a given ability: ✓ denotes that the ability is a primary focus of many problems, ○ denotes that it appears in a non-trivial but secondary role, and × denotes that it is rarely or never required. Benchmarks: AMO-Bench (An et al., 2025), IMO-Bench (Guo et al., 2025), OlymMATH (Sun et al., 2025a), Omni-MATH (Gao et al., 2024), MathArena (Balunović et al., 2025), FIMO (Liu et al., 2023), and OlympiadBench (He et al., 2024).

Table 7: Task Volume Distribution of Planning/Feedback Tasks (Text-only vs. Multimodal)

Agentic Ability	Task	Text-only	Multimodal	Total
Planning	Atomic Ability Selection	190	31	221
	Plan Generation	217	20	237
	Generate the Next Step	561	48	609
Feedback	Verification and Judgement	486	20	506
	Error Location	334	13	347
	Correction and Fix	262	18	280

Stage 3: Consistency verification. The final stage uses a verifier LLM to check whether the generated JSON model is faithful to the original question and satisfies a minimal reasonable representation criterion. The verifier is asked to judge whether every variable corresponds to a quantity mentioned in the text, whether the constraints only use given information and remain compatible with the narrative, and whether the objective matches the question being asked. It returns a small JSON verdict with a boolean pass flag and a short textual justification. We keep only examples with the pass flag set to true. For these items, the verified JSON model is stored as the gold answer for the modeling conversion ability.

C.2 Forward Reasoning

The forward reasoning subset is constructed from NaturalProofs (Welleck et al., 2021), which provides natural-language theorems and proofs. For each sample we start with a theorem statement and its full proof and aim to produce a sequence of steps of the form (index, proof segment, plan). In our benchmark, the theorem statement is used as the question field and the ordered list of such triples serves as the answer field.

We employ a four-stage LLM pipeline.

Stage 1: Prior proof checking. In the first stage, an LLM reads the theorem statement together with the proof and decides whether, as written, the proof correctly proves the theorem without major gaps or contradictions. The model is required to output a compact JSON verdict with a boolean pass flag and brief reasons. Only proofs that pass this global sanity check are retained, and their statements become the question field.

Stage 2: Proof segmentation. In the second stage, another LLM partitions each accepted proof into a small number of conceptual segments, typically between three and eight. The instructions require the model to split the proof into contiguous blocks, each representing a meaningful reasoning step rather than a single algebraic manipulation, and to output a JSON list containing for each segment a running index and the corresponding proof text. Segmentations that are not valid JSON or that fall outside the desired length range are discarded.

Table 8: Atomic Ability Coverage Statistics of Planning Task (Text-only vs. Multimodal)

Atomic Ability	**Text-only (Count/Percentage)**	**Multimodal (Count/Percentage)**
Symbol Recognition	54 (28.4%)	6 (19.4%)
Concept Understanding	185 (97.4%)	15 (48.4%)
Calculation Execution	147 (77.4%)	26 (83.9%)
Spatial Perception	33 (17.4%)	22 (71.0%)
Formal Math Language	6 (3.2%)	3 (9.7%)
Reasoning	184 (96.8%)	23 (74.2%)
Proof Construction	135 (71.1%)	1 (3.2%)
Theorem Application	65 (34.2%)	19 (61.3%)
Modeling Transformation	11 (5.8%)	10 (32.3%)

Table 9: Trajectory & Capability Statistics of Planning Task "Solution Planning" (Text-only vs. Multimodal)

Metric	**Text-only**	**Multimodal**
Trajectory Details		
Total Trajectories	243	25
Successfully Written	217	20
Skipped (Insufficient Steps)	26	5
Multimodal Samples	-	20 (100.0%)
Step Count Distribution		
2 Steps	16 (7.4%)	3 (15.0%)
3 Steps	42 (19.4%)	3 (15.0%)
4 Steps	89 (41.0%)	9 (45.0%)
5 Steps	62 (28.6%)	4 (20.0%)
6 Steps	6 (2.8%)	1 (5.0%)
10 Steps	2 (0.9%)	-
Tool Usage Frequency		
Symbol Recognition	76	2
Concept Understanding	229	9
Calculation Execution	137	17
Spatial Perception	35	12
Formal Math Language	3	2
Reasoning	223	14
Proof Construction	108	0
Theorem Application	59	13
Modeling Transformation	10	6

Stage 3: Plan extraction. In the third stage, a third LLM takes the theorem and the segmented proof as input and generates a high-level plan sentence for each segment. The prompt emphasizes that each plan should describe the main goal or proof action of the segment, such as setting up induction, rewriting a sum in a different form or applying a particular inequality, without reproducing detailed calculations. It also explicitly forbids claiming that the theorem has already been proved or that the proof is complete. The model outputs a JSON list that pairs each segment index with a short plan sentence, which yields an aligned sequence of proof segments and plans.

Stage 4: Posterior plan verification. The final stage uses a judge LLM to examine the theorem, the full proof and the entire list of segment–plan pairs and to decide whether the plans collectively form a coherent forward reasoning guide. The judge is instructed to check that the plans cover the key ideas of the proof in the correct order, accurately describe what each segment is doing, and together provide a reasonable high-level roadmap to reprove the theorem. It again outputs a JSON verdict with a boolean pass flag and brief reasons. We keep only samples with the pass flag set to true. For those samples, the ordered list of triples (index, proof segment, plan) is stored as the gold answer for the forward reasoning ability.

D Evaluation Metrics

We describe the metrics used to evaluate each atomic action capability. For each ability a, let $\mathcal{D}^a = \{(x_i^a, y_i^a)\}_{i=1}^{N^a}$ denote the evaluation set, where x_i^a is the input (problem statement and, when applicable, diagram) and y_i^a is the ground-truth structured answer. Given a model f, we write $\hat{y}_i^a = f(x_i^a)$ for the prediction on instance i. We denote by $\mathbb{I}[\cdot]$ the indicator function, which equals 1 if its argument is true and 0 otherwise.

Many metrics rely on an auxiliary LLM-as-a-judge J that compares the gold answer y_i^a and the model prediction $\hat{y}_i^a$ (together with the original question) and returns a per-instance score $s_i^{(m,a)} \in [0, 1]$ for metric m. Unless otherwise specified, the reported score for metric m on ability a is the average

$$M^{(m,a)} = \frac{1}{N^a} \sum_{i=1}^{N^a} s_i^{(m,a)}, \tag{3}$$

where $N^a = |\mathcal{D}^a|$ is the number of evaluation instances of ability a.

For several metrics (e.g., ConceptSet_F1, RelationF1, VariableF1), the judge first determines semantic matches between gold and predicted items and then computes a standard F_1 score. For a given

instance i, let t_i be the number of "true positive" matches, p_i the total number of predicted items, and g_i the total number of gold items. We define

$$\text{precision}_i = \frac{t_i}{\max(1,\, p_i)}, \qquad \text{recall}_i = \frac{t_i}{\max(1,\, g_i)}, \tag{4}$$

and the per-instance F_1 score

$$\mathrm{F1}_i = \begin{cases} \dfrac{2\,\text{precision}_i\,\text{recall}_i}{\text{precision}_i + \text{recall}_i}, & \text{if precision}_i + \text{recall}_i > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

In practice, the counts t_i, p_i, g_i are obtained from the judge based on semantic matching rules (e.g., treating synonyms and minor paraphrases as the same concept or fact), while the formula above specifies how the final score is computed from these counts.

Symbol recognition. For symbol recognition ($a =$ SYMBOL_RECOGNITION), each gold answer y_i^a is a canonical LaTeX expression and the model prediction $\hat{y}_i^a$ is taken from the `latex` field in the structured output.

Expression-level accuracy (ExpRate). For each instance i, the judge J compares y_i^a and $\hat{y}_i^a$ and returns a binary decision $r_i^{\text{exp}} \in \{0, 1\}$ indicating whether the entire expression is a correct transcription (up to mathematical equivalence, e.g., ignoring harmless LaTeX formatting differences). The per-instance score for ExpRate is $s_i^{(\text{ExpRate},a)} = r_i^{\text{exp}}$, and the dataset-level metric is

$$\text{ExpRate} = M^{(\text{ExpRate},a)} = \frac{1}{N^a} \sum_{i=1}^{N^a} r_i^{\text{exp}}. \tag{6}$$

Symbol-level similarity (SymRate). To capture graded symbol-level correctness, we define a similarity score between the gold and predicted expressions. For each instance i, let G_i and $\hat{G}_i$ denote the sequences (or multisets) of mathematical symbols and operators extracted from y_i^a and $\hat{y}_i^a$, respectively (e.g., variables, constants, relation symbols, arithmetic operators, function symbols, and brackets that affect structure). We define a symbol-level correctness score

$$s_i^{\text{sym}} = \phi(G_i, \hat{G}_i) \in [0, 1], \tag{7}$$

where ϕ is a similarity function that assigns $s_i^{\text{sym}} \approx 1$ when G_i and $\hat{G}_i$ encode essentially the same symbol inventory and structure, values around 0.7 when the main structure is preserved but with minor missing/extra symbols, values around 0.4 for partially similar expressions, and $s_i^{\text{sym}} \approx 0$ when the expressions are unrelated. In our implementation, ϕ is instantiated by the judge J, which inspects $(y_i^a, \hat{y}_i^a)$ and outputs such a score. The per-instance score for SymRate is $s_i^{(\text{SymRate},a)} = s_i^{\text{sym}}$, and the dataset-level metric is

$$\text{SymRate} = M^{(\text{SymRate},a)} = \frac{1}{N^a} \sum_{i=1}^{N^a} s_i^{\text{sym}}. \tag{8}$$

Calculation execution. For calculation execution ($a =$ CALCULATION_EXECUTION), each gold answer y_i^a is a short algebraic or numeric result, and the model prediction $\hat{y}_i^a$ is taken from the `answer` field.

Exact match (ExactMatch / EM). The judge J compares the gold answer y_i^a and the predicted answer $\hat{y}_i^a$ and outputs a binary score $r_i^{\text{em}} \in \{0, 1\}$ indicating whether the two are mathematically equivalent (allowing for standard algebraic rewrites and simple representation differences). The per-instance score is $s_i^{(\text{ExactMatch},a)} = r_i^{\text{em}}$, and

$$\text{ExactMatch} = \frac{1}{N^a} \sum_{i=1}^{N^a} r_i^{\text{em}}. \tag{9}$$

Concept understanding. For concept understanding ($a =$ CONCEPT_UNDERSTANDING), each gold answer y_i^a is a list of key mathematical concepts required to prove the given statement, while the model prediction $\hat{y}_i^a$ contains a list of predicted concepts (field `concepts`) and a free-text explanation (field `understanding`).

Concept set F_1 (ConceptSet_F1 / Set-F1). For each instance i, let C_i be the set of unique gold concept names extracted from y_i^a and $\hat{C}_i$ the set of unique predicted concept names from $\hat{y}_i^a$. The judge J marks which gold concepts in C_i are "covered" by at least one predicted concept in $\hat{C}_i$ (allowing synonyms, paraphrases, and naming variations) and thereby determines:

- t_i: the number of covered gold concepts (true positives),
- $g_i = |C_i|$: the number of unique gold concepts,
- $p_i = |\hat{C}_i|$: the number of unique predicted concepts.

Using these counts, we compute per-instance precision, recall, and F_1 as in the general definition

above, and set

$$s_i^{(\text{ConceptSet\_F1},a)} = \text{F1}_i. \quad (10)$$

The dataset-level ConceptSet_F1 is then

$$\text{ConceptSet\_F1} = \frac{1}{N^a}\sum_{i=1}^{N^a}\text{F1}_i. \quad (11)$$

Conceptual understanding accuracy (ConceptACC / Acc). For each instance i, let q_i^a denote the natural-language statement to be proved and let u_i be the model's explanation from the `understanding` field. The judge J decides whether u_i captures the main objects and the overall claim direction reasonably well and returns $r_i^{\text{acc}} \in \{0,1\}$. The per-instance score is $s_i^{(\text{ConceptACC},a)} = r_i^{\text{acc}}$, and

$$\text{ConceptACC} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{acc}}. \quad (12)$$

Counterexample construction. For counterexample construction (a = COUNTER_CONSTRUCTION), the input specifies a mathematical statement and the task is to construct a concrete counterexample. The gold answer y_i^a encodes one valid counterexample, and the prediction $\hat{y}_i^a$ is taken from the `answer` field.

Exact match (ExactMatch / EM). We reuse the ExactMatch metric: the judge checks whether the predicted counterexample $\hat{y}_i^a$ indeed satisfies all premises and violates the target conclusion. It returns $r_i^{\text{ctr}} \in \{0,1\}$; the per-instance score is $s_i^{(\text{ExactMatch},a)} = r_i^{\text{ctr}}$, and

$$\text{ExactMatch} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{ctr}}. \quad (13)$$

Spatial reasoning. For spatial reasoning (a = SPATIAL_AWARENESS), each gold answer y_i^a encodes a list of geometric facts (objects, relations, numeric attributes), and the prediction $\hat{y}_i^a$ contains a list of predicted facts in the `facts` field and, when applicable, numeric values.

Geometric relation F_1 (RelationF1). For each instance i, let R_i be the set of unique gold relational facts (e.g., incidence, parallelism, perpendicularity, equality of angles/segments) and $\hat{R}_i$ the set of unique predicted relational facts. The judge J decides which facts match semantically (e.g., treating AB and BA as the same segment and allowing symmetric relations) and thereby determines:

- t_i: the number of matched fact pairs (true positives),
- $g_i = |R_i|$: the number of unique gold facts,
- $p_i = |\hat{R}_i|$: the number of unique predicted facts.

We then compute per-instance precision, recall, and F_1 as before and define

$$s_i^{(\text{RelationF1},a)} = \text{F1}_i, \qquad \text{RelationF1} = \frac{1}{N^a}\sum_{i=1}^{N^a}\text{F1}_i. \quad (14)$$

Numeric attribute accuracy (ValueAcc). For examples that include numeric attributes, let $v_i = (v_{i,1},\ldots,v_{i,d_i})$ denote the vector of gold numeric values (e.g., lengths, angles) and $\hat{v}_i = (\hat{v}_{i,1},\ldots,\hat{v}_{i,d_i})$ the predicted values. We define a per-instance correctness indicator $r_i^{\text{val}} \in \{0,1\}$ that is 1 when all numeric attributes are judged correct (up to a small tolerance in continuous values) and 0 otherwise. The dataset-level metric is

$$\text{ValueAcc} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{val}}. \quad (15)$$

Formal mathematical language. For formal mathematical language (a = FORMAL_MATH_LANGUAGE), the task is to translate an informal statement into a Lean4-style theorem. For each instance i, the gold formalization is y_i^a, and the model outputs a list of k candidate Lean snippets $(\hat{\ell}_{i,1},\ldots,\hat{\ell}_{i,k})$ from the `lean` field; in our experiments $k = 5$.

Compilation pass@k (LeanCompilePass_at_k / Compile). We approximate top-k syntactic well-formedness. The judge inspects the first k candidates and returns $r_i^{\text{comp}} \in \{0,1\}$, where $r_i^{\text{comp}} = 1$ if at least one $\hat{\ell}_{i,j}$ looks like a plausible top-level Lean declaration (e.g., a `theorem`/`lemma`/`def` header together with a body starter such as `:=` or `by`), and 0 otherwise. The dataset-level metric is

$$\text{LeanCompilePass\_at\_k} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{comp}}. \quad (16)$$

Semantic alignment pass@k (LeanSemAlign_at_k / Align). Beyond syntactic plausibility, we require that the formal statement is semantically aligned with the gold theorem y_i^a. The judge compares y_i^a and the k candidates and returns $r_i^{\text{align}} \in \{0,1\}$ indicating whether there exists a candidate whose main proposition/goal matches

that of y_i^a (up to harmless equivalences such as reordering of conjuncts or renaming of bound variables). The metric is

$$\text{LeanSemAlign\_at\_k} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{align}}. \quad (17)$$

Forward reasoning. For forward reasoning ($a =$ FORWARD_REASONING), each instance has a gold high-level proof plan P_i represented as an ordered list of step annotations, and the model prediction $\hat{P}_i$ is provided in the `plans` field.

Plan precision (PlanPrecision / Prec.). Let $\hat{S}_i$ be the multiset of predicted plan steps for instance i, after removing empty steps and deduplicating near-duplicates by intent. The judge decides which steps in $\hat{S}_i$ are "useful" for proving the target statement (i.e., relevant and realistically helpful). Let $p_i = |\hat{S}_i|$ be the number of predicted steps and t_i the number of useful steps among them. The per-instance PlanPrecision score is

$$s_i^{(\text{PlanPrecision},a)} = \frac{t_i}{\max(1,\, p_i)}, \quad (18)$$

and the dataset-level metric is

$$\text{PlanPrecision} = \frac{1}{N^a}\sum_{i=1}^{N^a} s_i^{(\text{PlanPrecision},a)}. \quad (19)$$

Modeling transformation. For modeling transformation ($a =$ MODELING), each gold answer y_i^a is a structured mathematical model containing a set of variables, a set of constraints, and an objective function. We denote by V_i the gold variable set, by C_i the gold constraint set, and by o_i the gold objective. The model prediction $\hat{y}_i^a$ provides corresponding fields $\hat{V}_i$, $\hat{C}_i$, and $\hat{o}_i$ via the keys `variables`, `constraints`, and `objective`.

Variable F_1 (VariableF1 / Var-F1). For each instance i, let V_i be the set of unique gold variables and $\hat{V}_i$ the set of unique predicted variables. The judge marks which gold variables are correctly recovered (allowing name changes, missing units, and partial but clearly identifiable descriptions) and thereby determines:

- t_i: the number of matched variables,
- $g_i = |V_i|$: the number of unique gold variables,
- $p_i = |\hat{V}_i|$: the number of unique predicted variables.

We compute per-instance precision, recall, and F_1 as before and set

$$s_i^{(\text{VariableF1},a)} = \text{F1}_i, \qquad \text{VariableF1} = \frac{1}{N^a}\sum_{i=1}^{N^a}\text{F1}_i. \quad (20)$$

Constraint equivalence rate (ConstrEqvRate / Constr). For each instance i, the judge compares the gold constraint set C_i and the predicted constraint set $\hat{C}_i$. A gold constraint is counted as "covered" if some predicted constraint is an equivalent or clearly paraphrased version of it (possibly allowing minor relaxations). Let h_i be the number of covered gold constraints and $g_i = |C_i|$ the total number of gold constraints. The per-instance constraint coverage is

$$s_i^{(\text{ConstrEqvRate},a)} = \frac{h_i}{\max(1,\, g_i)}, \quad (21)$$

and the dataset-level metric is

$$\text{ConstrEqvRate} = \frac{1}{N^a}\sum_{i=1}^{N^a} s_i^{(\text{ConstrEqvRate},a)}. \quad (22)$$

Objective equivalence rate (ObjEqvRate / Obj). Similarly, for each instance i, the judge compares the gold objective o_i and the predicted objective $\hat{o}_i$ (given the original question) and returns $r_i^{\text{obj}} \in \{0, 1\}$ indicating whether they target the same core quantity with the same direction (e.g., minimize vs maximize). The dataset-level metric is

$$\text{ObjEqvRate} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{obj}}. \quad (23)$$

Theorem application. For theorem application ($a =$ THEOREM_APPLICATION), the input specifies a goal statement and hints about usable theorems. The model must select and instantiate appropriate theorems and produce a proof. For each instance i, we obtain k proof candidates $(\hat{p}_{i,1}, \ldots, \hat{p}_{i,k})$ from the `proof` field, with $k = 5$ in our experiments.

Pass@k (Pass_at_k). The judge J checks whether at least one of the k proofs $(\hat{p}_{i,1}, \ldots, \hat{p}_{i,k})$ successfully and coherently proves the target statement. It returns $r_i^{\text{pass}} \in \{0, 1\}$, where $r_i^{\text{pass}} = 1$ if some $\hat{p}_{i,j}$ is accepted as a valid proof and 0 otherwise. The empirical pass@k is

$$\text{Pass\_at\_k} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{pass}}. \quad (24)$$

E Illustrative Data Cases

We provide a series of sample data to further illustrate our task design and to help readers better understand what capabilities of the model we evaluated. See Table 10 and 11.

F Additional Experiment Results

F.1 Inference Hyperparameters.

All models are queried with low randomness and a single sample per instance ($n = 1$). For planning and feedback tasks we fix the temperature to 0.0 and top-p to 1.0, and cap the generation length at 256 tokens for capability planning, 512 tokens for solution planning, and 1,024 tokens for next-step planning. For atomic action tasks we use temperature 0.7 and top-$p = 1.0$ with a maximum of 2,048 output tokens. For formal mathematical language and theorem application we draw five samples per instance ($n = 5$) and aggregate predictions by majority vote; all other action tasks use a single sample.

F.2 Results on Multimodal Planning Tasks

Planning performance of multimodal models across capability planning, solution planning, and next-step planning in Table 12.

F.3 Results on Multimodal Feedback Tasks

Feedback performance of multimodal models, evaluating correctness judgment, error localization, and fix suggestion, in Table 13.

F.4 Results on Multimodal Action Tasks

Additional results of Action performance of Multimodal Models, in Table 14.

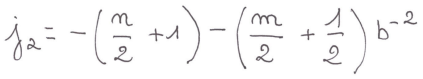
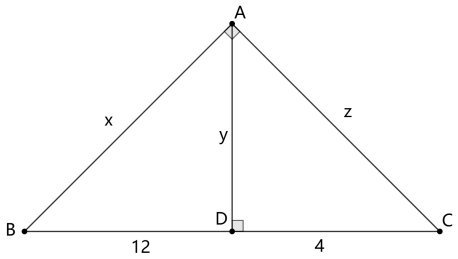
Atomic ability	Case
Symbol recognition	**Input:**  **Output:** $j_2 = -\left(\frac{n}{2}+1\right) - \left(\frac{m}{2}+\frac{1}{2}\right) b^{-2}.$
Concept understanding	**Input:** **Irrational Numbers form G-Delta Set in Reals:** Let $R \setminus Q$ denote the set of irrational numbers. Let (R, τ) denote the real number line with the usual (Euclidean) topology. Then $R \setminus Q$ forms a G_δ set in R. **Output:** 1. **Rational Numbers are Countably Infinite:** The set Q of rational numbers is countably infinite. 2. **Rational Numbers form F-Sigma Set in Reals:** Let Q be the set of rational numbers. Let (R, τ) be the real number line with the usual (Euclidean) topology. Then Q is a F_σ set in R. 3. **De Morgan's Laws (Set Theory)/Set Difference/General Case/Difference with Union:** $S \setminus \bigcup \mathcal{T} = \bigcap_{T' \in \mathcal{T}} (S \setminus T')$, where $\bigcup \mathcal{T} := \{x : \exists T' \in \mathcal{T} \text{ such that } x \in T'\}$, that is, the union of $\mathcal{T}$.
Computation execution	**Input:** Problem: Find the smallest x such that $x \equiv 16 \pmod 3$, $x \equiv 4 \pmod 3$, and $x \equiv 7 \pmod{18}$. **Output:** 7.
Spatial awareness	**Input:**  **Output:** Equal(LengthOfLine(AB), x) Equal(LengthOfLine(AC), z) Equal(LengthOfLine(AD), y) Equal(LengthOfLine(BD), 12) Equal(LengthOfLine(CD), 4) PerpendicularBetweenLine(AD, CD) PerpendicularBetweenLine(BD, AD) PerpendicularBetweenLine(CA, BA)
Formal mathematical language	**Input:** The function f, defined on the set of ordered pairs of positive integers, satisfies the following properties: $f(x, x) = x$, $f(x, y) = f(y, x)$, and $(x + y)f(x, y) = yf(x, x + y)$. Calculate $f(14, 52)$. Show that it is 364. **Output:** theorem aime_1988_p8 (f : $\mathbb{N} \to \mathbb{N} \to \mathbb{R}$) (h_0 : $\forall x,\ 0 < x \to f\, x\, x = x$) (h_1 : $\forall x\, y,\ 0 < x \wedge 0 < y \to f\, x\, y = f\, y\, x$) (h_2 : $\forall x\, y,\ 0 < x \wedge 0 < y \to (\uparrow x + \uparrow y) * f\, x\, y = y * f\, x\, (x + y)$) : $f\, 14\, 52 = 364$:= by sorry

Table 10

Atomic ability	Case
Forward reasoning	**Input:** **Hermitian Matrix has Real Eigenvalues:** Every Hermitian matrix has eigenvalues which are all real numbers. **Output:** 1. Set up the eigenpair for the Hermitian matrix and record the eigen-equation relating A, v, and λ. 2. Form a scalar inner-product relation by left-multiplying the eigen-equation with v^*, relating v^*Av to $\lambda \cdot v^*v$. 3. Treat these quantities as 1×1 matrices and express the conjugate-transpose of the quadratic form in terms of conjugate-transposes of its factors. 4. Apply the Hermitian property of A and the conjugate-transpose involution to relate the conjugates of v^*Av and v^*v back to themselves, preparing to view them as real scalars. 5. Introduce scalar representatives a and b for the 1×1 forms, note b is nonzero for an eigenvector, and connect their self-adjointness to real-valuedness. 6. Use the scalar equality to express λ as the quotient a/b and combine this with the real-valuedness of a and b to relate λ to the real numbers.
Modeling	**Input:** 26 children were riding on the bus. At the bus stop 38 more children got on the bus. How many children are on the bus now? **Output:** **Variables:** name: initial_children, desc: number of children riding on the bus initially, unit: children name: boarded_children, desc: number of children who got on at the bus stop, unit: children name: total_children, desc: number of children on the bus now, unit: children **Constraints:** initial_children = 26 boarded_children = 38 total_children = initial_children + boarded_children initial_children $\in Z$ boarded_children $\in Z$ total_children $\in Z$ initial_children ≥ 0 boarded_children ≥ 0 total_children ≥ 0 **Objective:** solve total_children
Theorem application	**Input:** **Target: Irrational Numbers form G-Delta Set in Reals:** Let $R \setminus Q$ denote the set of irrational numbers. Let (R, τ) denote the real number line with the usual (Euclidean) topology. Then $R \setminus Q$ forms a G_δ set in R. **Hints:** 1. **Rational Numbers are Countably Infinite:** The set Q of rational numbers is countably infinite. 2. **Rational Numbers form F-Sigma Set in Reals:** Let Q be the set of rational numbers. Let (R, τ) be the real number line with the usual (Euclidean) topology. Then Q is a F_σ set in R. 3. **De Morgan's Laws (Set Theory)/Set Difference/General Case/Difference with Union:** $S \setminus \bigcup \mathcal{T} = \bigcap_{T' \in \mathcal{T}} (S \setminus T')$, where $\bigcup \mathcal{T} := \{x : \exists T' \in \mathcal{T} \text{ such that } x \in T'\}$, that is, the union of $\mathcal{T}$. **Output:** {{begin-eqn}} $Q = \bigcup_{\alpha \in Q} \{\alpha\}$ (Rational Numbers form F-Sigma Set in Reals) $\leadsto R \setminus Q = R \setminus \bigcup_{\alpha \in Q} \{\alpha\}$ $= \bigcap_{\alpha \in Q} (R \setminus \{\alpha\})$ (De Morgan's Laws: Difference with Union) {{end-eqn}} The result follows from Rational Numbers are Countably Infinite. {{qed}}

Table 11

Table 12: Planning performance of multimodal models across capability planning, solution planning, and next-step planning.

Model	Capability Planning			Solution Planning				Next-step Planning		
	Pre	Rec	F1	Logic	Sub-goal	Step	Overall	Capability	Subgoal	Overall
Multimodal Tasks										
General Open-sourced Models										
DeepSeek-VL2	47.33	79.14	53.48	54.50	56.50	48.50	53.17	25.00	38.03	31.52
Qwen3-VL-32B-Instruct	52.15	49.73	49.98	67.00	64.00	66.00	65.67	52.08	47.64	49.86
Qwen3-VL-235B-A22B-Instruct	45.62	94.46	60.63	52.00	49.50	50.50	50.67	54.17	55.09	54.63
InternVL3.5-38B-Instruct	53.13	72.53	58.50	63.00	64.50	61.00	62.83	37.50	45.99	41.74
InternVL3.5-241B-A28B-Instruct	33.47	42.15	36.30	53.50	52.50	52.00	52.67	43.75	45.72	44.74
Commercial Models										
GPT-5.2	63.66	71.40	66.31	79.00	80.00	77.50	78.83	60.42	56.29	58.35
Claude-4.5-Sonnet-Thinking	65.47	88.87	74.16	79.00	76.50	76.00	77.17	56.25	56.11	56.18
Gemini-3-Pro	61.32	89.57	71.99	74.50	73.00	71.50	73.00	56.25	59.70	57.98

Table 13: Feedback performance of multimodal models, evaluating correctness judgment, error localization, and fix suggestion.

Model	Correctness	Error Localization			Fix Suggestion	
	Accuracy	Step Judge	Type Classification	Overall	Reason Consistency	Overall
Multimodal Tasks						
General Models						
DeepSeek-VL2	50.00	30.77	61.54	46.15	34.44	15.50
Qwen3-VL-32B	55.56	69.23	61.54	65.38	66.11	29.75
InternVL3.5-38B	55.56	84.62	38.46	61.54	50.00	22.50
Commercial Models						
GPT-5.2	61.11	61.54	23.08	42.31	53.56	23.87
Claude-4.5	50.00	30.77	30.77	30.77	67.22	30.25
Gemini-3-Pro	50.00	57.14	71.43	64.29	61.94	27.88

Table 14: Action performance of Multimodal Models.

Models	**Symbol**		**Spatial**
	ExpACC	SymACC	RelationF1
General Models			
InternVL3_5-241B-A28B	80.00	94.50	57.28
InternVL3_5-38B	75.00	75.50	66.74
Qwen3-VL-235B-A22B	60.00	85.00	70.76
Qwen3-VL-32B	60.00	84.50	77.13
Deepseek-VL2	65.00	64.00	36.15
Commercial Models			
gpt-5.2	63.80	86.50	75.40
claude-4.5-sonnet	82.60	82.50	12.30
Gemini-3-pro	96.10	97.70	87.30