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

Jiayi Kuang[1]*, Yinghui Li[2], Yunze Song[2], Keyu Chen[2], Zhifeng Shen[2], Yangning Li[2], Yidong Wang[2], Di Yin[2], Ruizhi Qiao[2], Xing Sun[2], Kai Jin[1,†], Ying Shen[1,4,†], Liang Lin[1], Philip S. Yu[3]
[1]Sun Yat-sen University, [2]Tencent Youtu Lab
[3]University of Illinois Chicago, [4]Pengcheng Laboratory
kuangjy6@mail2.sysu.edu.cn, lebronyhli@tencent.com

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 [1] 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 similar 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 Introduction

In recent years, Large Language Models (LLMs) have achieved remarkable progress on complex reasoning tasks (Lu et al., 2025), particularly in mathematics (Lewkowycz et al., 2022; Achiam et al., 2023). As tasks become increasingly diverse and complex, LLM-driven approaches are evolving beyond chain-of-thought toward *agentic reasoning* (Wei et al., 2022; Wang et al., 2024b). By dynamically incorporating planning, action 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?*

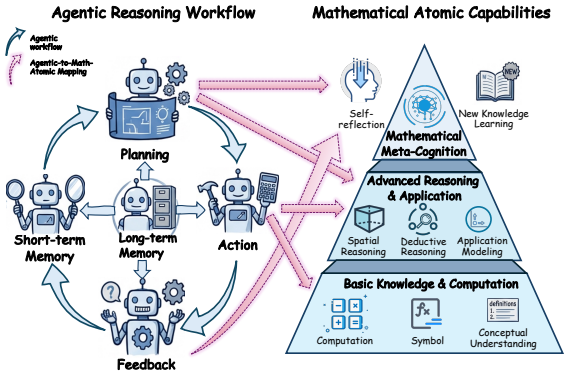


Figure 1: It illustrates alignment between agentic behaviors and mathematical atomic capabilities, enabling interpretable evaluation of LLMs' agentic intelligence.

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 (Zhang et al., 2025a; Kuang et al., 2025). For example, solving a geometry problem requires translating the visual diagrams into variables (Spatial Perception), choosing a strategy (Modeling), and calculating (calculation). 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 agentic intelligence in every step throughout the reasoning.

*Work done during the internship at Tencent Youtu Lab.
† Corresponding authors.
[1]https://github.com/Eternity-gaga/Agentic-Math-Bench

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 4437–4465
October 24-29, 2026 ©2026 Association for Computational Linguistics

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., 2025a; 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 benchmark, AgenticMathBench(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*, which novelly decompose agentic mathematical reasoning. Based on this, we formulate new process-level evaluation tasks covering both multimodal and text-only scenarios. To support these tasks, we construct an automated pipeline that integrates data collection, high-quality trajectory synthesis, multi-stage filtering, 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 Method

2.1 Task Definition

Atomic thinking decomposes complex mathematical reasoning into a sequence of atomic cognitive units, closely paralleling the 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. AMB targets the *intrinsic agentic intelligence* of LLMs rather than a fully deployed autonomous agent. Specifically, we (i) align core agentic behaviors (*Planning*, *Action*, *Feedback*) with mathematical atomic capabilities, (ii) evaluate the reasoning process at the process level, and (iii) decouple intrinsic evaluation from end-to-end agent execution so that each capability can be diagnosed in isolation, with a detailed agentic discussion in Appendix B.4. Concretely, our framework is organized along two aligned axes: a *mathematical atomic-capability* axis and an *agentic-function* axis. AMB evaluates the *intersection* of the two axes rather than either axis alone, so that the taxonomy is not a re-labeling of mathematical skills: planning is assessed as capability selection, ordering, and next-step decision making, action as isolated execution of a single atomic capability, and feedback as monitoring and revision over an existing trajectory.

2.1.1 Atomic Capability System

We first collected definitions of mathematical atomic skills from numerous literature sources, and compared them with existing data. We then further merged or broke down several mathematical atomic abilities, while removing some abilities that have rarely been focused on within existing research. Finally, after consulting with several mathematics experts, we finalized a complete system of atomic abilities, with more information in Appendix B.

- **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*, *For-*

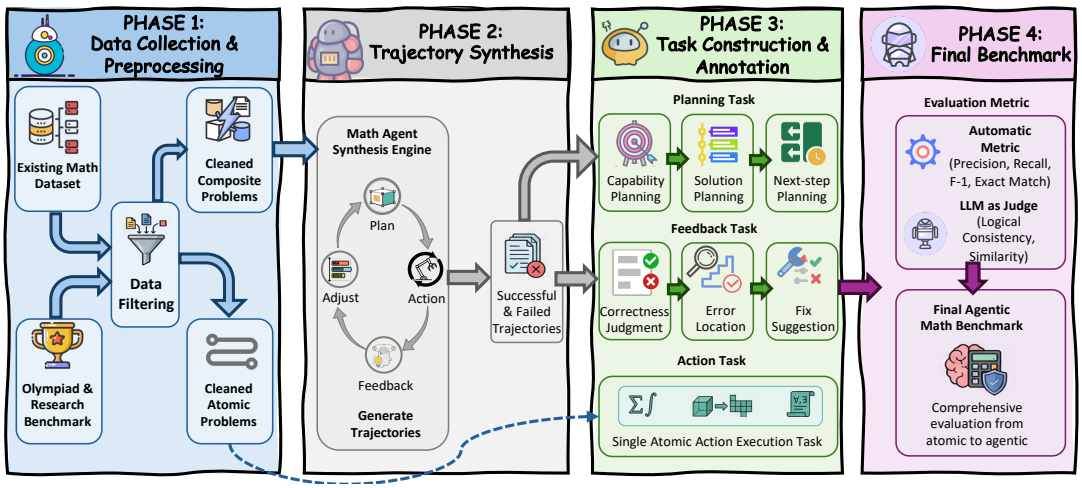


Figure 2: The overall benchmark construction pipeline of our AgenticMathBench.

malization, *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 Learning*.

These atomic capabilities can be naturally aligned with different agentic modules. Level 1 and Level 2 capabilities primarily support the *Action*, enabling the execution of decomposed sub-tasks. In contrast, higher-level capabilities emphasizing meta-cognition 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 and the coordinated deployment of multiple atomic capabilities. *New knowledge learning* is closely related to memory, which is difficult to evaluate in a single test, so we have not included it yet.

2.1.2 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 at step t. The sequence must satisfy logical dependency. In planning, we focus on whether the model identifies the correct atomic capabilities and decomposes the problem, while excluding execution correctness.

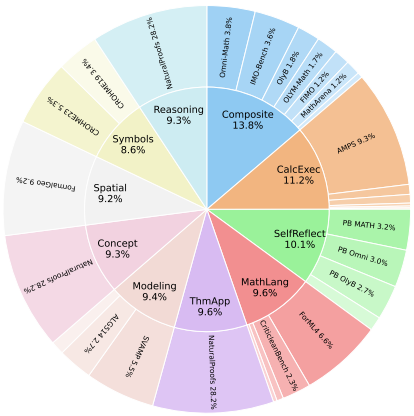


Figure 3: Data distribution of collected datasets.

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

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

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.

2.2 Benchmark Construction

2.2.1 Data Collection

By manually reviewing over 150 math benchmarks, including easier high school level and more challenging competition and research level difficulties, we collect and filter 27 datasets covering various mathematical abilities, as distribution in Figure 3.

Single Atoms We primarily filter existing benchmarks covering handwritten symbol, elementary and advanced problem solving, geometry, and theorem proving, including CROHME (Mouchère et al., 2016), 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), and ProofNet (Azerbayev et al., 2023). These datasets are transferred to a unified schema, with a pipeline of filtering, de-duplication, and down-sampling.

Composite Atoms Recent competition-level and Olympiad-style benchmarks are collected, including AMO-Bench (An et al., 2025), IMO-Bench (Luong et al., 2025), OlymMATH (Sun et al., 2025), 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, with more process details and statistics in Appendix C.

2.2.2 Trajectory Synthesis and Filtering

While single atomic Action tasks can be derived from raw data, directly annotating planning and feedback behaviors at the problem level is difficult, 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 a canonical *plan–action–feedback* agent paradigm. This agent 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,verifies intermediate results, and dynamically adjusts subsequent actions based on feedback. After execution, we perform automated final answer checks, manual reasoning process quality filtering, and diversity filtering on coverage steps and atomic capabilities on the 1,236 and 136 text and multimodal trajectories obtained. Ultimately, 17.3% of the trajectory data was retained, with more filtering details in Appendix C.5.

2.3 Task Annotation and Evaluation Metrics

While obtaining the filtered math data and the trajectory data, we design tasks and metrics for agentic abilities in Table 1, with statistics in Appendix D. The specific calculation of every metric and the implementation details of LLM-as-judge are provided in Appendix F, with information that ensures the effectiveness of the convincing LLM evaluation, and the experiment with human validation. To further verify that our conclusions are not sensitive to the judge LLM, we additionally conduct a robustness check in Appendix F.1.3, Table 14).

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.

Action. We operationalize *Action* as directly executable atomic mathematical tasks, covering 8 atomic capabilities. We do not evaluate *Self-reflection* because this ability is more suitable for *Feedback*. For tasks such as *Symbol Recognition* and *Calculation*, we rewrite and normalize from existing datasets into standard formats. For *process-structured* targets such as *Deductive and Inductive reasoning* and *Mathematics Modeling*, we prompt GPT-4o to rewrite original solutions into our intermediate representations, followed by normalization and process checking, with detailed methods in Appendix E. Evaluation follows the target type: exact match/structure-aware accuracy for recognition and transcription, set-based scores for concept/relation

Table 1: ExpAcc/SymAcc=expression/symbol accuracy, CAS-Eq=CAS equivalence, Lean-Cmp=Lean compilation, Judge(Cov/Cons)=LLM-judge coverage/consistency, Sim=semantic similarity, with data cases in Appendix C.4).

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
Calculation	1,328	T	Numeric/symbolic calculation result	EM (num), CAS-Eq (sym)
Spatial Perception	1,139	T&I	Spatial relation predicate set	P/R/F1 (+ Acc)
Formalization	1,140	T	Lean4 theorem declaration	Lean-Cmp, Lean-Align
Deductive and Inductive	1,096	T	Ordered proof-outline steps	Judge(Cov/Cons)
Mathematics Modeling	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)

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.

Feedback. We evaluate *Feedback* as judgment and correction over executed 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 GPT-4o); and (iii) *fix suggestion* by truncating before a modified step and asking for a repair action 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 consistency of modified reason, next step's capabilities, and sub-task.

3 Experiment

3.1 Experimental Settings

We evaluate 3 families of models: (i) general open-source models, (ii) math open-source models, and (iii) commercial 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 and Qwen3-VL-235B-A22B (Qwen Team, 2025b), InternVL3.5-38B and InternVL3.5-241B-A28B (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). Commercial models include GPT-5.2 (OpenAI, 2025), Claude-Sonnet-4.5-thinking (Anthropic, 2025), Gemini-3-Pro-Preview (Google DeepMind, 2025). Details about the implementation are provided in Appendix G.1.

3.2 Results of Planning Ability

We evaluate planning ability, and Table 2 reports that commercial 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 a core weakness, as adaptive next-step decision making is cen-

Table 2: Planning performance of models across capability planning, solution planning, and next-step planning. The results on multimodal tasks are in Table 15.

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

tral 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 and the need for training on planning rather than static reasoning alone.

3.3 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. The highest-level fix suggestion shows significantly weaker performance. Models can explain why a modification is needed, but they fail to translate such explanations into **actionable next-step suggestions**. Our benchmark exposes that current models can partially localize errors, but struggle to convert feedback into effective corrective actions. We further provide case analysis (Appendix G.3) indicating that many models can flag a nearby erroneous step but still fail to attribute the *causal* origin of the failure, with a fully instantiated example in Appendix G.4.

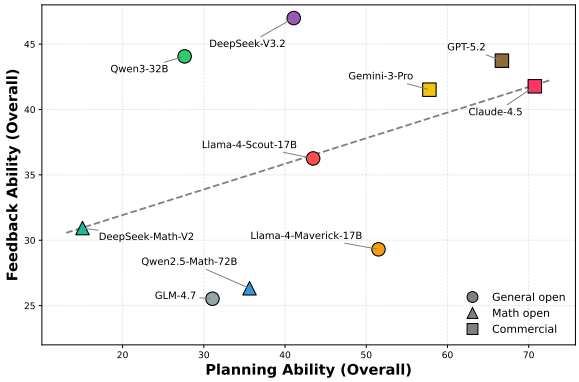


Figure 4: This figure illustrates the different performance of models on planning and feedback tasks.

3.4 Results of Action Ability

Table 4 evaluates models on *action*. Across all tasks, commercial models consistently achieve strong and balanced performance, particularly in formal mathematical language and mathematical modeling. In contrast, while several open-source models perform competitively on calculation, 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**.

Table 3: Feedback performance of models, evaluating correctness judgment, error localization, and fix suggestion. The results on multimodal tasks are in Table 16.

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

4 Analysis and Discussion

4.1 Analysis of the Performance between End-to-end and Agentic Abilities

We select some typical models, and the Table 5 shows the difference between end-to-end and agentic abilities. Interestingly, strong performance on end-to-end math benchmarks does not necessarily show superior agentic abilities. The results clearly confirm our key observation: some models with excellent end-to-end math scores, especially those focused on math-specialized or problem-solving training, perform poorly in core agentic capabilities such as GLM-4.7. Meanwhile, there are clearly multiple pairs of models that have similar E2E math scores but significant differences in agentic capabilities, for example, GPT-5.2 and Gemini-3-pro. This result directly supports the core conclusion of our paper that similar end-to-end accuracy does not correspond to similar agentic capability profiles, and also suggests that specialized training for math tasks may cause a certain trade-off in the interactive reasoning and agentic capabilities.

4.2 Analysis of Planning vs Feedback Ability

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

4.3 Analysis of Action Performance

Figure 6 shows the overall performance on different action tasks. Across the six atomic dimensions, commercial models and large reasoning-enhanced models form a high, relatively balanced envelope, while several smaller open models exhibit "spiky" profiles with pronounced weak axes. *Concept* is the primary bottleneck, suggesting many Action failures stem from concept misalignment rather than arithmetic itself. Overall, high scores typically co-occur with balanced fundamentals, consistent with these tasks requiring coordinated competence.

4.4 Case Study

We provide representative cases, illustrating typical breakdowns in Planning and Feedback capabilities, with more analysis in Appendix G.3. In Figure 5, these cases reflect recurring patterns observed across multiple models, especially those with strong end-to-end accuracy but weaker agentic profiles. We observe that models tend to collapse multi-step reasoning into final answers, undermining step-level planning and have weak state tracking. To make these qualitative findings auditable, we further provide a complete *case card* in Ap-

Table 4: Action performance of models across execution-oriented mathematical capabilities. The results on multimodal tasks are in Table 17.

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	59.3	13.5	55.2	49.4	39.4	20.7	86.0	78.4	90.8	39.5
Llama-4-Maverick-17B	54.5	21.9	65.4	89.4	80.0	57.4	82.9	77.3	97.9	52.1
DeepSeek-V3.2	**84.7**	**52.4**	**97.7**	99.4	91.4	**93.1**	**88.3**	**83.0**	99.0	**100.0**
Qwen3-32B	73.2	42.0	90.0	**99.6**	**93.4**	48.1	83.6	75.1	**99.2**	85.6
Qwen3-235B-A22B-Thinking	78.8	44.7	92.3	98.9	91.4	86.9	77.8	73.8	98.8	98.2
GLM-4.7	43.5	16.6	59.7	4.4	2.7	38.8	74.3	64.9	98.2	41.6
Math Open-source Models										
Qwen2.5-Math-72B-Instruct	6.0	6.2	48.9	2.5	2.0	7.8	36.0	23.1	90.8	5.3
DeepSeek-Math-V2	**83.1**	**33.2**	**72.3**	**56.1**	**50.1**	**65.8**	**79.5**	**81.6**	**99.3**	**80.6**
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**	93.7	**94.7**	**92.2**	84.7	99.7	**99.0**

Case Study of Planning Failure

Planning Failure: Inability to Dynamically Adjust the Next Step

Task setting. The model is given a partially completed solution trajectory and must generate the correct next reasoning step.
Problem (simplified). Solve for integer solutions of a Diophantine equation after partial simplification.
Given trajectory:
1. Transform the equation into standard form.
2. Factor the expression into two linear components.
3. Conclude that the product equals zero.

At this stage, the correct next action should be:
Analyze each factor separately and enumerate possible integer solutions under the given constraints.

Observed failure pattern.
Several models instead:
- Re-expand the factorization (redundant step),
- Jump directly to a final answer without enumerating constraints,
- Or apply an unrelated algebraic manipulation (e.g., completing the square).

Behavioral issue.
The failure is not mathematical incompetence. Instead, it **fails to adapt to the local trajectory state.** This type of failure is particularly common in models optimized for final-answer accuracy. They shortcut toward conclusions rather than maintaining step-level structural coherence.

Figure 5: Case studies of typical breakdowns in Planning capabilities.

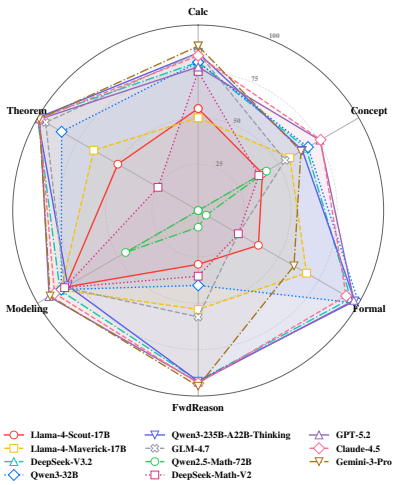


Figure 6: Performance on action tasks.

Table 5: Comparison between end-to-end mathematical benchmarks and agentic mathematical capabilities.

Model	End-to-end Math		Agentic		
	MATH	AIME25	Plan.	Feed.	Act.
Llama-4-Scout-17B	82.6	10.0	33.5	46.7	53.7
Llama-4-Maverick-17B	90.6	15.9	**46.1**	40.9	67.4
Qwen3-235B-A22B	98.0	81.5	28.7	21.4	84.5
GLM-4.7	98.8	95.7	29.0	29.7	64.5
Qwen2.5-Math-72B	85.9	20.0	30.2	27.2	30.5
GPT-5.2	**100.0**	**99.0**	41.0	**53.7**	**89.3**
Gemini-3-Pro	**100.0**	95.7	34.2	48.3	80.6

pendix G.4 that lists the complete input problem, the trajectory shown to the model, the expected output format, the raw model output, and the corresponding diagnosis results.

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 intelligence 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 in final-answer accuracy. Since mathematical reasoning encompasses many behaviors of agent reasoning, our future work will expand to more multimodal scenarios, paving the way for better eliciting the latent agentic intelligence.

Limitations

While we align agentic functions with mathematical atomic capabilities, the taxonomy and task instantiations may not cover all forms of mathematical agentic reasoning, especially for long-horizon memory and iterative learning of new knowledge. In future work, we plan to extend the framework toward long-horizon evaluation settings, including explicit memory perturbation tests and tool-interaction loops. We also note that the trajectory-based multimodal subset is smaller than the text-only subset, mainly because high-difficulty multimodal mathematical problems with diagrams are scarce. Expanding diagram-based and visual-reasoning trajectories is a direction of future versions of AMB.Finally, since AMB measures intrinsic agentic capability rather than a deployed agent system, it reports capability scores without fine-grained token and latency accounting; a preliminary cost discussion is given in Appendix I, and cost-aware evaluation that jointly considers capability gains and inference budgets remains an important extension.

Acknowledgement

This work was supported in part by the New Generation Artificial Intelligence-National Science and Technology Major Project (2025ZD0123003), and the National Natural Science Foundation of China Enterprise Innovation and Development Joint Fund (Artificial Intelligence Field) Key Support Projects (U25B2072).

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, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and et al. 2023. Gpt-4 technical report. *arXiv 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 et al. 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*.

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

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, and et al. 2025. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*.

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)*.

Yinghui Li, Jiayi Kuang, Haojing Huang, Zhikun Xu, Xinnian Liang, Yi Yu, Wenlian Lu, Yangning Li, Xiaoyu Tan, Chao Qu, and 1 others. 2025. One example shown, many concepts known! counterexample-driven conceptual reasoning in mathematical llms. *arXiv preprint arXiv:2502.10454*.

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*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and et al. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

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 et al. 2024a. Process-driven autoformalization in lean 4. *arXiv preprint arXiv:2406.01940*.

Junru Lu, Jiarui Qin, Lingfeng Qiao, Yinghui Li, Xinyi Dai, Bo Ke, Jianfeng He, Ruizhi Qiao, Di Yin, Xing Sun, and 1 others. 2025. Youtu-llm: Unlocking the native agentic potential for lightweight large language models. *arXiv preprint arXiv:2512.24618*.

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*.

Minh-Thang Luong, Dawsen Hwang, Hoang H Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, and et al. 2025. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 35406–35430.

Yiran Ma, Zui Chen, Tianqiao Liu, Mi Tian, Zhuo Liu, Zitao Liu, and Weiqi Luo. 2025a. What are step-level reward models rewarding? counterintuitive findings from mcts-boosted mathematical reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24812–24820.

Ziyang Ma, Qingyue Yuan, Zhenglin Wang, and Deyu Zhou. 2025b. Large language models have intrinsic meta-cognition, but need a good lens. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.

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)*.

Harold Mouchère, Christian Viard-Gaudin, Richard Zanibbi, and Utpal Garain. 2016. Icfhr2016 crohme: Competition on recognition of online handwritten mathematical expressions. In *2016 15th international conference on frontiers in handwriting recognition (ICFHR)*, pages 607–612. IEEE.

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 et al. 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, Yuxiang Luo, Chengda Lu, ZZ Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, and Xiaokang Zhang. 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. 2025. 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.

Bintao Tang, Xin Yang, Yuhao Wang, Zixuan Qiu, Zimo Ji, and Wenyuan Jiang. 2025. INTEGRALBENCH: 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.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li.

2024a. Measuring multimodal mathematical reasoning with math-vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169.

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. 2024b. 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. 2024c. 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 et al. 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 et al. 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*.

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, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and et al. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *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.

Shaowei Zhang and Deyi Xiong. 2025. Debate4math: Multi-agent debate for fine-grained reasoning in math. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16810–16824.

Xiaokai Zhang, Na Zhu, Yiming He, Jia Zou, Qike Huang, Xiaoxiao Jin, Yanjun Guo, Chenyang Mao, Yang Li, Zhe Zhu, and et al. 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. 2025a. GAUSS: Benchmarking structured mathematical skills for large language models. *arXiv preprint arXiv:2509.18122*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025b. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10495–10516.

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.

Jiachen Zhu, Congmin Zheng, Jianghao Lin, Kounianhua Du, Ying Wen, Yong Yu, Jun Wang, and Weinan Zhang. 2025. Retrieval-augmented process reward model for generalizable mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8453–8468.

A Related Work

A.1 Benchmarks for Mathematical Reasoning

Early benchmarks for mathematical reasoning predominantly evaluate models in simple question answering (Koncel-Kedziorski et al., 2016; Li et al., 2025). 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., 2025a) 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.

A.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., 2024c) and PRM800K-style process reward models (Lightman et al., 2024; Ma et al., 2025a; Zhang et al., 2025b; Zhu et al., 2025) score individual reasoning steps and support reranking, reinforcement learning, and multi-agent debate frameworks (Zhang and Xiong, 2025). 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.

B Additional Details of Math Atomic Capability Taxonomy

For the math atomic capability taxonomy, we clarify our design process.

B.1 Literature Survey and Theoretical Foundations

We reviewed the existing related work and noted that the current focus on mathematical ability can be roughly divided into two categories. The first category focuses on a data perspective, paying more attention to mathematical corpora from different fields (He et al., 2025; Wang et al., 2024a). These works cover various areas of mathematics, including algebra, geometry, calculus, analysis, topology, combinatorics, and so on. However, this classification method differs from agent capabilities and the specific reasoning patterns of the model. Another major category focuses on different abilities in mathematical reasoning or problem-solving. Some tasks concentrate on a specific ability, such as formal language proofs (Peng et al., 2025) and mathematical modeling (Liu et al., 2025b). These abilities are closer to our conception of various agentic abilities in mathematical reasoning. We further collected a large number of mathematical task definitions and integrated literature from empirical studies or surveys. For example, atomic thinking (Kuang et al., 2025) defines three basic mathematical atomic abilities from a logical reasoning perspective, while GAUSS (Zhang et al., 2025a) defines mathematical problem-solving abilities in three levels and twelve dimensions. Based on the above analysis, we integrated a large number

of mathematical atomic abilities.

B.2 Dataset Mapping and Practical Validation

Furthermore, we collected a large number of existing mathematical datasets and attempted to map them to the mathematical atomic capabilities we integrated in the previous step. While some mathematical atomic capabilities are defined in existing work, there may be issues such as inappropriate classification or insufficient research. We removed some rarely studied mathematical atomic capabilities, because these scarce capabilities are often not important or frequently occurring core atomic capabilities in mathematical problem-solving.

B.3 Expert Consultation

Finally, we discussed the atomic capability classification definition and related existing datasets with mathematical experts, including three researchers with PhDs degrees in mathematics-related fields and two senior instructors with more than 10 years of experience in higher education in mathematics. We iteratively refined the taxonomy to ensure Diversity (covering major reasoning patterns), Near-completeness (broad coverage of common math problem-solving scenarios), Relative independence (minimizing overlap), Evaluability (amenable to structured assessment), and Agent alignment (clear mapping to planning, action, and feedback capabilities). The final taxonomy reflects a balance between theoretical grounding and empirical operability.

B.4 Scope of the Agentic Formulation

We provide a more detailed account of the scope of the "agentic" formulation used in AMB. Our agentic framing comes from three explicit design choices: (i) *mapping agentic functions to mathematical atomic capabilities*, where Planning, Action, Feedback, and Memory are aligned with reusable mathematical sub-skills (Figure 1 and Section 2.1); Action is instantiated as executable mathematical sub-skills (e.g., symbol recognition, calculation, formalization) that correspond to operations an agent would delegate to tools or specialized modules; (ii) *evaluating the interactive reasoning loop at the process level*: planning tasks include both global solution planning and *next-step planning conditioned on a partially executed trajectory*, while feedback tasks require correctness judgement, first-error localization, and repair suggestion conditioned on previous trajectory states held in memory; (iii) *separating intrinsic capability evaluation from deployed-agent evaluation*: instead of executing a complete agency task and analyzing the entire trajectory, we break down agentic behavior into finer-grained sub-tasks that can examine one capability at a time. Thus, although AMB does not invoke external tools at evaluation time and intentionally factors out memory updates, its decomposition is designed specifically around agentic mathematical reasoning capabilities, and the resulting scores indicate whether an LLM can effectively leverage its agentic capabilities within an agent scaffold.

C 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.

C.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 6 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 C.3; Table 7 provides a qualitative view of how different competition benchmarks exercise the atomic abilities.

C.2 Curated Atomic Abilities from Existing Benchmarks

Seven of the nine atomic abilities—symbol recognition, concept understanding, computation ex-

Table 6: Statistics of the sources of the 9 atomic abilities in AgenticMathBench.

Atomic ability	Data sources
Symbol recognition	CROHME19/23 (Mouchère et al., 2016)
Concept understanding	NaturalProofs (Welleck et al., 2021)
Calculation	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 perception	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)
Deductive and inductive reasoning	NaturalProofs (Welleck et al., 2021)
Mathematics Modeling	ALG514 (Kushman et al., 2014), DRAW-1K (Upadhyay and Chang, 2017), SVAMP (Patel et al., 2021)
Theorem application	NaturalProofs (Welleck et al., 2021)

ecution, 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 6.

Symbol recognition. For symbol recognition we reuse handwritten mathematical expression data from the CROHME 2019 and 2023 competitions (Mouchère et al., 2016). 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-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.

Calculation The calculation 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 perception. 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 tar-

get statement, leaving explicit diagram generation to downstream tasks.

Formal mathematical language. For the formal mathematical language ability we use Lean-based formalization corpora, including CriticLean-Bench (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, Deductive and Inductive Reasoning and Mathematics Modeling, 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 E.

C.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 (Luong et al., 2025), OlymMATH (Sun et al., 2025), 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 7 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.

C.4 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 8 and 9.

C.5 Trajectory Filtering Details

we conducted multi-stage evaluation and filtering to ensure high-quality trajectories before using them as ground truth.

Final answer correctness filtering. From 1,627 collected complex math problems, we removed proof-only problems without final answers. We retained 1,236 pure-text and 136 multimodal problems. After generating solution trajectories with the Math Agent, we first filtered by final answer correctness. Only trajectories with correct final answers were retained, resulting in 283 pure-text and 36 multimodal trajectories.

Human evaluation of reasoning quality. Two human annotators evaluated each trajectory with reference solutions provided. They assessed Step completeness, Logical consistency, and Reasoning validity. Each criterion was scored in a binary manner (0/1). Only trajectories receiving positive scores on all three criteria were retained. This resulted in 243 pure-text and 25 multimodal trajectories.

Diversity filtering. We further removed trajectories with fewer than two reasoning steps or involving fewer than two atomic abilities. This ensures

Table 7: 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 (Luong et al., 2025), OlymMATH (Sun et al., 2025), Omni-MATH (Gao et al., 2024), MathArena (Balunović et al., 2025), FIMO (Liu et al., 2023), and OlympiadBench (He et al., 2024).

Atomic ability	**AMO-Bench**	**IMO-Bench**	**OlymMATH**	**Omni-MATH**	**MathArena**	**FIMO**	**OlympiadBench**
Symbol recognition	×	×	×	×	×	×	✓
Concept understanding	○	✓	○	○	○	✓	○
Calculation	✓	✓	✓	✓	✓	×	✓
Spatial perception	○	○	○	○	○	○	✓
Formal mathematical language	×	×	×	×	×	✓	×
Deductive and inductive reasoning	○	✓	○	○	○	✓	○
Mathematics modeling	✓	✓	✓	✓	✓	✓	✓
Theorem application	○	✓	○	○	○	✓	○

structural richness and capability diversity. The final dataset contains 217 pure-text and 20 multimodal trajectories. After this rigorous filtering pipeline, 17.3% of the original trajectories were retained, ensuring correctness, logical rigor, and capability diversity.

Analysis of Retention Rate The relatively low retention rate is by design: we deliberately select competition-level and advanced problems so that retained trajectories cover diverse atomic capabilities and non-trivial reasoning, with ∼76.8% removed by final-answer correctness and a further ∼6% by step-coverage and capability-diversity filters. Importantly, LLMs are used in restricted, validated roles in AMB—e.g., schema normalization for action tasks, trajectory generation for planning/feedback, and judge-based scoring—rather than for free-form generation of new mathematical content. A component-level summary of human-curated, LLM-rewritten, LLM-synthesized, and LLM-judged parts is provided in Appendix H (Table 18).

D Additional Details of Data Statistics

Table 10 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 11 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 12 summarizes the trajectory details, step distribution, and capability usage frequency, complementing the capability coverage analysis of Planning Task 1.

E Additional Details of Task Construction and Annotation

E.1 Methematics Modeling

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.

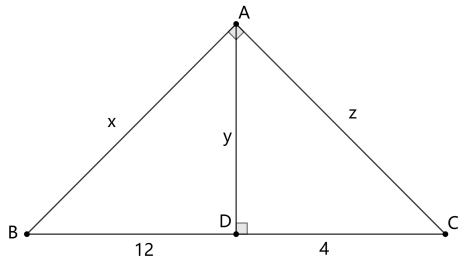
Atomic ability	Case
Symbol recognition	**Input:** $j_2 = -\left(\frac{n}{2}+1\right) - \left(\frac{m}{2}+\frac{1}{2}\right) b^{-2}$ **Output:** $j_2 = -\left(\frac{n}{2}+1\right) - \left(\frac{m}{2}+\frac{1}{2}\right) b^{-2}.$
Concept understanding	**Input:** **Irrational Numbers form G-Delta Set in 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 $\bigcup \mathcal{T} := \{x : \exists T' \in \mathcal{T}$ such that $x \in T'\}$, that is, the union of $\mathcal{T}$.
Calculation	**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 perception	**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 8

Atomic ability	Case
Deductive and Inductive 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.
Mathematics 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 $\bigcup \mathcal{T} := \{x : \exists T' \in \mathcal{T}$ 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) $\rightsquigarrow R \setminus Q = R \setminus \bigcup_{\alpha \in Q} \{\alpha\}$ $= \bigcap_{\alpha \in Q} (R \setminus \{\alpha\})$ (De Morgan's Laws: Difference with Union) {{end-eqn}} The result follows from Rational Numbers are Countably Infinite. {{qed}}

Table 9

Table 10: 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

Table 11: 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	147 (77.4%)	26 (83.9%)
Spatial Perception	33 (17.4%)	22 (71.0%)
Formal Math Language	6 (3.2%)	3 (9.7%)
Deductive and Inductive Reasoning	184 (96.8%)	23 (74.2%)
Proof Construction and Counter-proof	135 (71.1%)	1 (3.2%)
Theorem Application	65 (34.2%)	19 (61.3%)
Mathematics Modeling	11 (5.8%)	10 (32.3%)

Table 12: 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	137	17
Spatial Perception	35	12
Formal Math Language	3	2
Deductive and Inductive Reasoning	223	14
Proof Construction and Counter-proof	108	0
Theorem Application	59	13
Modeling Transformation	10	6

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.

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.

E.2 Deductive and Inductive Reasoning

The Deductive and Inductive 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.

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 Deductive and Inductive Reasoning ability.

F Addition Details of LLM Usage of Evaluation and Metrics

F.1 Overall Details of LLM-as-judge

For process-level tasks such as solution planning and fix suggestion, evaluation inherently involves flexibility. In realistic agent trajectories, multiple reasoning paths may be logically sound, efficient, and ultimately correct. Rigid string matching or automatic metrics would penalize legitimate reasoning variations and fail to capture nuanced differences in coherence, efficiency, and logical structure. For this reason, an LLM-based evaluator is particularly suitable for assessing process-level agentic behaviors. Moreover, LLM-as-a-judge has become increasingly popular in recent research, particularly for open-ended and structured reasoning evaluation. Our use of this methodology therefore aligns with established practice.

F.1.1 Details of LLM Evaluation Pipeline

Concretely, we use DeepSeek-V3, as the evaluation model, with temperature fixed at 0 to ensure deterministic outputs. To ensure reliability and consistency, we implemented several safeguards.

- First, each evaluation prompt includes 3–5 human-authored scoring criteria designed by domain experts. These criteria guide the model to assess reasoning from multiple perspectives rather than performing superficial answer matching. Ground-truth reference solutions are provided to anchor evaluation.

- Second, particularly for Planning and Feedback tasks, we explicitly instruct the judge to assess outputs along multiple dimensions, including correctness, logical consistency, coherence, and efficiency. The model is encouraged to compare against the reference solution without requiring strict step-by-step identity. A weighted scoring scheme is then used to produce the final score.

- Third, we conducted human validation. We randomly sampled 10% of evaluation cases for manual review. The agreement between human annotations and LLM-based scores exceeded 96%, indicating strong consistency.

- Fourth, we implemented strict output normalization procedures. Beyond enforcing standardized output formats in inference prompts,

we designed a post-processing stage that extracts relevant answers from partially formatted responses. This ensures that formatting inconsistencies do not bias evaluation results.

- Finally, for near-zero or anomalous scores, we conducted additional manual inspections to exclude artifacts such as empty outputs or formatting failures. Human re-evaluation confirmed that these low scores reflected genuine capability limitations rather than evaluation noise.

Taken together, these measures ensure that the LLM-based evaluation is structured, reproducible, and empirically validated.

F.1.2 Verification of LLM-based Evaluation

To ensure the reliability of LLM-based scoring in our benchmark, we conduct a human-aligned verification study on a subset of model outputs. Specifically, we focus on four key agentic capabilities: *Solution Planning*, *Next-step Planning*, *Error Localization*, and *Fix Suggestion*. These dimensions correspond to the core competencies required for mathematical agentic reasoning.

Human Evaluation. We randomly sample a subset of 100 evaluated results and collect human annotations. Each sample is independently scored by both the LLM judge and human annotators following the same evaluation rubric. We then compute standard classification metrics based on the confusion matrix between LLM predictions and human labels, including Accuracy, Precision, Recall, and F1-score.

Results. The alignment between LLM-based evaluation and human judgment is summarized in Table 13. Overall, the LLM judge demonstrates strong agreement with human annotations across all dimensions, supporting its effectiveness as a scalable evaluation proxy.

Overall, the combined evaluation achieves an F1 score of 0.86, demonstrating strong alignment with human judgment. Compared to single-criterion evaluation strategies, our multi-dimensional LLM-as-a-judge framework provides a more fine-grained and comprehensive assessment of agentic reasoning capabilities.

We note that the high recall bias of the LLM judge may lead to a slight overestimation of performance, as it tends to accept borderline cases. Nevertheless, this design choice ensures that valid reasoning trajectories are less likely to be incorrectly discarded, making it suitable for large-scale evaluation scenarios where coverage is critical.

Table 13: Agreement between LLM-based evaluation and human annotations across four agentic capability dimensions. Metrics are computed from confusion matrices on a human-annotated subset.

Capability	Acc.	Prec.	Recall	F1
Solution Planning	0.87	0.83	0.91	0.87
Next-step Planning	0.84	0.79	0.90	0.84
Error Localization	0.89	0.85	0.94	0.89
Fix Suggestion	0.81	0.76	0.92	0.83
Overall	0.85	0.81	0.92	0.86

F.1.3 Second-Judge Robustness Check

To examine whether our conclusions depend on the choice of the LLM judge, we replicate the human-aligned verification study with GPT-5-mini as an alternative judge, using the same sampled outputs and the same rubric. Table 14 reports the resulting agreement with human annotations as well as the cross-judge agreement against our default DeepSeek-V3 judge. The two judges yield highly consistent absolute scores (Acc. 0.85 vs. 0.86, F1 0.86 vs. 0.85) and a 0.96 cross-judge agreement, indicating that the LLM-as-judge results in our benchmark are robust to the specific judge used. Since the same judge and rubric are applied uniformly to all evaluated models, any residual bias is unlikely to invalidate relative comparisons across models, although it may slightly overestimate absolute scores for verbose outputs.

Table 14: Second-judge robustness check using GPT-5-mini as an alternative judge on the same human-annotated subset. "Agreement" denotes the cross-judge agreement with our default DeepSeek-V3 judge.

Judge	Acc.	Prec.	Rec.	F1	Agreement
DeepSeek-V3	0.85	0.81	0.92	0.86	1.00
GPT-5-mini	0.86	0.83	0.88	0.85	0.96

F.2 Metrics of Action Ability

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. 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

$$s_i^{(\text{ConceptSet_F1},a)} = \text{F1}_i. \tag{10}$$

The dataset-level ConceptSet_F1 is then

$$\text{ConceptSet_F1} = \frac{1}{N^a}\sum_{i=1}^{N^a} \text{F1}_i. \tag{11}$$

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

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

Spatial perception. 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

$$s_i^{(\text{RelationF1},a)} = \text{F1}_i, \qquad \text{RelationF1} = \frac{1}{N^a}\sum_{i=1}^{N^a} \text{F1}_i. \tag{13}$$

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

$$\text{ValueAcc} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{val}}. \tag{14}$$

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

$$\text{LeanCompilePass_at_k} = \frac{1}{N^a}\sum_{i=1}^{N^a} r_i^{\text{comp}}. \tag{15}$$

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

$$\text{LeanSemAlign_at_k} = \frac{1}{N^a} \sum_{i=1}^{N^a} r_i^{\text{align}}. \quad (16)$$

Deductive and Inductive Reasoning. For Deductive and Inductive 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

$$s_i^{(\text{PlanPrecision},a)} = \frac{t_i}{\max(1,\, p_i)}, \quad (17)$$

and the dataset-level metric is

$$\text{PlanPrecision} = \frac{1}{N^a} \sum_{i=1}^{N^a} s_i^{(\text{PlanPrecision},a)}. \quad (18)$$

Mathematics Modeling. 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

$$s_i^{(\text{VariableF1},a)} = \text{F1}_i, \qquad \text{VariableF1} = \frac{1}{N^a} \sum_{i=1}^{N^a} \text{F1}_i. \quad (19)$$

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

$$s_i^{(\text{ConstrEqvRate},a)} = \frac{h_i}{\max(1,\, g_i)}, \quad (20)$$

and the dataset-level metric is

$$\text{ConstrEqvRate} = \frac{1}{N^a} \sum_{i=1}^{N^a} s_i^{(\text{ConstrEqvRate},a)}. \quad (21)$$

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

$$\text{ObjEqvRate} = \frac{1}{N^a} \sum_{i=1}^{N^a} r_i^{\text{obj}}. \quad (22)$$

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

$$\text{Pass_at_k} = \frac{1}{N^a} \sum_{i=1}^{N^a} r_i^{\text{pass}}. \tag{23}$$

G Additional Experiment Setting and Results

G.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.

G.2 Results on Multimodal Tasks

Multimodal Planning Tasks Planning performance of multimodal models across capability planning, solution planning, and next-step planning in Table 15.

Multimodal Feedback Tasks Feedback performance of multimodal models, evaluating correctness judgment, error localization, and fix suggestion, in Table 16.

Multimodal Action Tasks Additional results of Action performance of Multimodal Models, in Table 17.

G.3 Case Study

Below we provide three representative failure cases in Figure 7, illustrating typical breakdowns in Planning and Feedback capabilities. These examples reflect recurring patterns observed across multiple models, especially those with strong end-to-end accuracy but weaker agentic profiles. Across these cases, we observe several recurring behavioral patterns:

Shortcut bias. Models optimized for strong end-to-end performance tend to collapse multi-step reasoning into final answers, undermining step-level planning.

Weak state tracking. Failures often stem from insufficient sensitivity to the current trajectory state—either repeating previous steps or skipping necessary intermediate reasoning.

Repair inconsistency. Even when errors are correctly identified, corrective suggestions are often vague, generic, or inconsistent with the existing trajectory.

G.4 A Complete Case Card

To make the qualitative analysis fully auditable, we present one representative case in a compact *case card* format that exposes the input problem, the trajectory shown to the model, the expected output, the raw model output, and our diagnosis. The case targets the *Feedback–Error Localization* task, whose evaluated capability is to trace the earliest causal error in a completed trajectory.

Input problem. Calculate the limit $\lim_{x\to 0}(e^x - 1 - x)/x^2$.

Trajectory shown to the model. (1) Direct substitution gives $0/0$, so L'Hopital's rule can be applied (correct). (2) Differentiate the numerator: $e^x - 1 - x \to e^x - 1$ (correct). (3) Differentiate the denominator: $x^2 \to 2x$ (correct). (4) Apply L'Hopital's rule again, but differentiate the denominator as $2x \to 1$ (*incorrect; earliest error*). (5) Substitute $x = 0$ and conclude that the limit is 1 (consequence of step 4).

Expected output. The earliest incorrect step is step 4; the error type is a differentiation/calculation error; the correct repair differentiates $2x$ as 2 and then computes $\lim_{x\to 0} e^x/2 = 1/2$.

Actual model output (one representative model). "The first error occurs in Step 5. The previous applications of L'Hopital's rule are valid, but the final substitution gives an incorrect final value. The solution should recompute the final limit instead of concluding 1."

Why this case is diagnostic. An outcome-oriented benchmark only observes that the final answer is wrong and cannot attribute the failure. AMB instead reveals that the model does sense the final-answer inconsistency but localizes the error to step 5 rather than to the earliest faulty transformation in step 4, i.e., a process-level feedback failure in causal error localization rather than a

Table 15: 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 16: 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 17: Action performance of Multimodal Models.

Models	Symbol		Spatial
	ExpACC	SymACC	RelationF1
General Models			
InternVL3_5-241B-A28B	74.5	90.3	60.6
InternVL3_5-38B	72.6	80.2	62.6
Qwen3-VL-235B-A22B	61.8	85.9	76.5
Qwen3-VL-32B	61.1	84.6	75.4
Deepseek-VL2	72.0	74.3	24.4
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

mere arithmetic slip. This distinction is actionable: a feedback module should repair step 4 before recomputing the final value, whereas final-answer signals cannot tell whether the agent should recalculate, replan, or repair a prior step.

H Roles of Human check and LLM as judge

Table 18 summarizes, at a component level, the source, LLM role, and downstream usage of each part of AMB. The table makes explicit that LLM rewriting is not used to freely create new mathematical content for most of the benchmark: in many cases it is used for controlled schema conversion (normalizing existing problems into a unified format) or for producing structured annotations that are subsequently checked by rules, validators, or humans. Trajectory synthesis follows a math-agent pipeline whose outputs go through automated final-answer checks, two-annotator reasoning-quality filtering, and diversity filtering before being used as ground truth (Appendix C.5).

Table 18: Component-level summary of human-curated, LLM-rewritten, LLM-synthesized, and LLM-judged parts of AMB.

Component	Source	LLM role	Used for
Raw problem pool	150+ reviewed math benchmarks; 27 retained datasets	No free-form generation	Base problem collection
Atomic action tasks	Existing datasets and structured conversions	Mostly schema normalization or controlled rewriting (e.g., GPT-4o for modeling/deductive-inductive reasoning)	Action evaluation
Trajectory synthesis	1,236 text and 136 multimodal candidates	Math-agent trajectory generation (DeepSeek-V3.2, Gemini-3-Pro-Preview) followed by automated and human filtering	Planning and feedback tasks
Feedback labels	Correct/incorrect trajectories	LLM-assisted error labeling, manually validated	Error localization and fix suggestion
Evaluation (judge)	All open-ended planning/feedback/action subtasks	DeepSeek-V3 as default judge; GPT-5-mini as robustness check (Table 14)	Scoring and human-aligned verification

I Analysis of Computational Cost and Contamination Risk

I.1 Cost-effectiveness of agentic decomposition.

Although our paper primarily benchmarks intrinsic agentic capabilities rather than proposing a deployment-time agent, we briefly discuss the cost trade-off. Conceptually, agentic decomposition does not always imply higher total cost. For difficult problems, direct long-chain reasoning can require very large generation budgets and may still suffer from truncation, overthinking, or local optima. In contrast, an atomized agentic process can decompose the problem into shorter subgoals, use specialized tools or validators for certain steps, and terminate earlier when feedback detects an error. As a preliminary qualitative observation on difficult AIME25-style problems, direct reasoning with a strong model such as Gemini-3-Pro often requires a maximum token budget above 20K and can still be truncated, while in an atomized agentic setting, many problems can be completed in roughly 2–4 steps with each step capped around 4K tokens (Table 19). This is not a universal cost guarantee; the cost-effectiveness of agentic workflows depends on the controller, number of steps, tool overhead, and stopping criteria, and we view fine-grained token/latency accounting as an important direction for future benchmark extensions.

Table 19: Qualitative cost observation on difficult AIME25-style problems: direct long-chain reasoning vs. atomized agentic reasoning.

Setting	Usual Steps	Token Cost
Gemini-3-Pro (direct)	>5 steps	>20,000
Gemini-3-Pro-based agent	2–4 steps	<4,000 / step

I.2 Contamination risk.

Data contamination is a known concern for any math benchmark built from public problems. Following common practice in the math-benchmark literature, we mitigate this risk in three ways: (i) we review more than 150 datasets and prioritize *recent* competition-level and Olympiad-style benchmarks; (ii) we evaluate planning, feedback, and action *processes* rather than only final answers, so that simply memorizing a final solution is not sufficient to succeed on AMB; and (iii) we will release metadata about data sources and timestamps wherever possible to enable users to analyze contamination risk. Going forward, we plan to maintain a refresh protocol that periodically incorporates newly released contest problems and to provide a strictly held-out split built after the release dates of the evaluated models.

I.3 Beyond the current evaluation paradigm.

We further view multi-agent debate and meta-cognitive evaluation as natural complements to AMB (Zhang and Xiong, 2025; Ma et al., 2025b): AMB's taxonomy already includes meta-cognitive capabilities (self-reflection, new knowledge learning) at the conceptual level, and future versions can evaluate whether multiple agents can debate intermediate plans, identify conflicting reasoning paths, and converge to a better repair strategy. We will consider these perspectives to future-work. A natural extension of AMB would be to evaluate whether multiple agents can debate intermediate plans and identify conflicting reasoning paths.

Case Study of Failure Behavior

Planning Failure: Inability to Dynamically Adjust the Next Step

Task setting. The model is given a partially completed solution trajectory and must generate the correct next reasoning step.
Problem (simplified). Solve for integer solutions of a Diophantine equation after partial simplification.
Given trajectory:
1. Transform the equation into standard form.
2. Factor the expression into two linear components.
3. Conclude that the product equals zero.

At this stage, the correct next action should be:
Analyze each factor separately and enumerate possible integer solutions under the given constraints.

Observed failure pattern.
Several models instead:
- Re-expand the factorization (redundant step),
- Jump directly to a final answer without enumerating constraints,
- Or apply an unrelated algebraic manipulation (e.g., completing the square).

Behavioral issue.
The failure is not mathematical incompetence. Instead, it **fails to adapt to the local trajectory state.** This type of failure is particularly common in models optimized for final-answer accuracy. They shortcut toward conclusions rather than maintaining step-level structural coherence.

Feedback Failure: Incorrect Localization of the First Error

Task setting. The model is given a complete solution trajectory containing and must identify the earliest incorrect step without reference to ground truth.
Problem (simplified). Evaluate a limit involving L'Hôpital's Rule.
Given trajectory:
1. Substitute directly and obtain 0/0 form.
2. Apply L'Hôpital's Rule.
3. Differentiate numerator correctly.
4. Differentiate denominator incorrectly (algebraic slip).
5. Compute final value.

Ground-truth earliest error: Step 4.

Observed failure pattern.
Many models identify Step 5 (incorrect final answer) or cannot pinpoint the earliest faulty transformation, thus fail to fix it in the next step.

Behavioral issue.
The model detects that something is wrong but fails to trace the causal origin of the error. This indicates weak **causal backward reasoning over trajectories.** Instead of reconstructing the dependency chain, it focuses on surface discrepancies (incorrect result) rather than structural failure points.

Feedback Failure: Incorrect Fix Suggestion After Identifying an Error

Task setting. After identifying an incorrect step, the model must propose the correct next action that repairs the reasoning, including the right atomic ability and the subgoal in next step..
Problem (simplified). The model correctly identifies that Step 3 incorrectly applies the quadratic formula due to a sign error.
Ground-truth earliest error:
Recompute the discriminant carefully and apply the quadratic formula with the corrected sign.

Observed failure pattern.
Models often:
- Restate the formula generically without recomputing,
- Or introduce an entirely new solving strategy inconsistent with previous steps.

Behavioral issue.
Here, the model demonstrates partial diagnostic ability but fails at **trajectory-consistent repair planning**. It either produces a vague meta-level suggestion or resets the reasoning path.

Figure 7: The example of 3 representative failure cases, illustrating typical breakdowns in Planning and Feedback capabilities.