Evaluating Mathematical Reasoning in Large Language Models: A Comprehensive Analysis of Prompting Strategies on the Google DeepMind Mathematics Dataset

Saanidhya Vats[1], Rui Min[1], Subramanyam Sahoo[2], Siddharth Mohapatra[3], Aman Chadha[4], Vinija Jain[5] and Divya Chaudhary[1,*]

[1]*Northeastern University, Seattle, WA, USA*
[2]*Berkeley AI Safety Initiative (BASIS), University of California, Berkeley, USA*
[3]*Northeastern University, Boston, MA, USA*
[4]*Amazon GenAI*
[5]*Google*

Abstract

Mathematical reasoning represents a fundamental benchmark for evaluating whether large language models can perform systematic logical operations beyond pattern recognition. This work presents a comprehensive evaluation of GPT-4o's mathematical reasoning across the complete Google DeepMind Mathematics Dataset, encompassing 28,000 problems spanning 56 subtopics in eight mathematical domains. We compare two distinct prompting strategies: meta-prompts with explicit formatting constraints and chain-of-thought (CoT) prompts requiring step-by-step reasoning demonstration. Chain-of-thought prompting achieved 39.02% overall accuracy compared to 28.83% for meta-prompting, representing an improvement of 10.19 percentage points. However, this improvement varied dramatically across domains. Our analysis reveals three mechanisms underlying meta-prompt underperformance: instruction over-constraint from extensive formatting rules, performance degradation under explicit evaluation criteria, and suppression of intermediate reasoning processes. Notably, 13 subtopics achieved 0% accuracy under chain-of-thought prompting despite nonzero meta-prompt performance. These findings demonstrate that prompting strategy effectiveness depends critically on mathematical domain characteristics, with sequential operations benefiting from explicit reasoning while pattern-based tasks may require alternative approaches. Our results challenge assumptions about uniform prompting effectiveness and highlight the need for domain-adaptive evaluation methodologies in mathematical reasoning assessment.

Keywords

Large language models, mathematical reasoning, prompting strategies, chain-of-thought, evaluation benchmarks

1. Introduction

While recent evaluations of large language models on mathematical reasoning have demonstrated impressive performance on high-level problems like mathematical olympiads, a critical gap remains in understanding how different prompting strategies affect performance across the full spectrum of school-level mathematics. Previous studies have typically evaluated single prompting approaches on limited problem sets, providing overall accuracy metrics without systematic analysis of domain-specific effectiveness patterns or underlying failure mechanisms. This work addresses this gap by conducting the first comprehensive comparison of meta-prompting and chain-of-thought strategies across the complete Google DeepMind Mathematics Dataset [1], encompassing 28,000 problems spanning 56 subtopics in eight mathematical domains including algebra, arithmetic, calculus, and probability. Our systematic evaluation reveals that prompting strategy effectiveness depends critically on mathematical

[PromptEng] Third International Workshop on Prompt Engineering for Pre-Trained Language Models co-located with the ACM WebConf, April.13, 2026, Dubai, United Arab Emirates
*Corresponding author.
vats.sa@northeastern.edu (S. Vats); min.ru@northeastern.edu (R. Min); subramanyam.sahoo@berkeley.edu (S. Sahoo); mohapatra.si@northeastern.edu (S. Mohapatra); aman@amanchadha.com (A. Chadha); vinija@google.com (V. Jain); chaudhary.div@northeastern.edu (D. Chaudhary)

domain characteristics, challenging assumptions about uniform prompting effectiveness and providing insights essential for developing robust mathematical reasoning systems.

The mathematical reasoning capabilities of large language models (LLMs) remain a critical frontier in artificial intelligence, representing a fundamental test of whether models can perform genuine logical inference beyond mere pattern matching. Unlike previous evaluations that focus primarily on accuracy metrics, we investigate the impact of prompting strategies on both performance and failure modes, progressing from meta-prompts with explicit format constraints to chain-of-thought (CoT) prompts that require step-by-step reasoning.

Our investigation addresses three key research questions: (1) How does prompting strategy affect mathematical reasoning performance across different domains? (2) What systematic failure patterns emerge when models encounter specific mathematical problem types? (3) Why do structured meta-prompts, despite their explicit formatting requirements, still fail to elicit correct responses in certain mathematical contexts?

The contributions of this work are threefold: (1) a systematic comparison of prompting strategies across the full spectrum of school-level mathematics, (2) detailed failure mode analysis revealing domain-specific weaknesses in mathematical reasoning, and (3) insights into the limitations of current prompting techniques for mathematical problem-solving. These findings have important implications for developing more robust mathematical reasoning systems and highlight critical areas for future research in LLM evaluation methodologies.

2. Related Work

The evaluation of mathematical reasoning capabilities in large language models has expanded rapidly, with benchmarks designed to test discrete compositional reasoning, algebraic generalization, and multi-step problem-solving. The Google DeepMind Mathematics Dataset [1] serves as a foundational benchmark for assessing neural models' abilities to handle procedurally generated mathematical problems across eight domains: algebra, arithmetic, calculus, comparisons, measurement, numbers, polynomials, and probability, divided into 56 subdomains. Recent benchmarks have emphasized uncontaminated evaluations to address data leakage. MathArena [2] presents a dynamic benchmark using real-time math competitions to evaluate 30 LLMs on 149 problems, finding scores below 25% for most models. Proof or Bluff? [3] assesses reasoning models on USAMO 2025, highlighting failures in rigorous proof generation, with only Gemini-2.5 Pro achieving 25% while other models scored less than 5%. Brains vs Bytes [4] develops an automatic schema for Olympiad proof evaluation, critiquing reliance on heuristics over true understanding. MathBench [5] covers arithmetic to college-level topics in bilingual settings, while Mathify [6] introduces MathQuest for high-school problems.

Prompting techniques have emerged as key to enhancing LLMs' mathematical performance. Chandra et al. [7] employ Structured CoT to evaluate GPT-4o, DeepSeek-V3, and Gemini-2.0 on GSM8K and MATH500, noting GPT-4o's strength in high-level tasks but issues with missing steps. Xu et al. [8] investigate Context Length Generalizability on extended word problems, proposing prompts akin to meta-prompting for improved reasoning efficacy. Vidal [9] compares GPT-4 variants on middle-school word problems, achieving 67-93% accuracy. Satpute et al. [10] evaluates GPT-4 on Math Stack Exchange, showing inconsistent accuracy in complex scenarios. Baral [11] introduces the MAPLE score for assessing reasoning misalignment in multi-step logic. However, these studies focus on individual prompting strategies or limited mathematical domains, lacking systematic comparison across comprehensive mathematical benchmarks. In contrast to prior evaluations, our work specifically examines GPT-4o on the DeepMind Mathematics Dataset using CoT for reasoning transparency and meta-prompting for structured outputs, addressing gaps in multi-domain generalization and prompting efficacy.

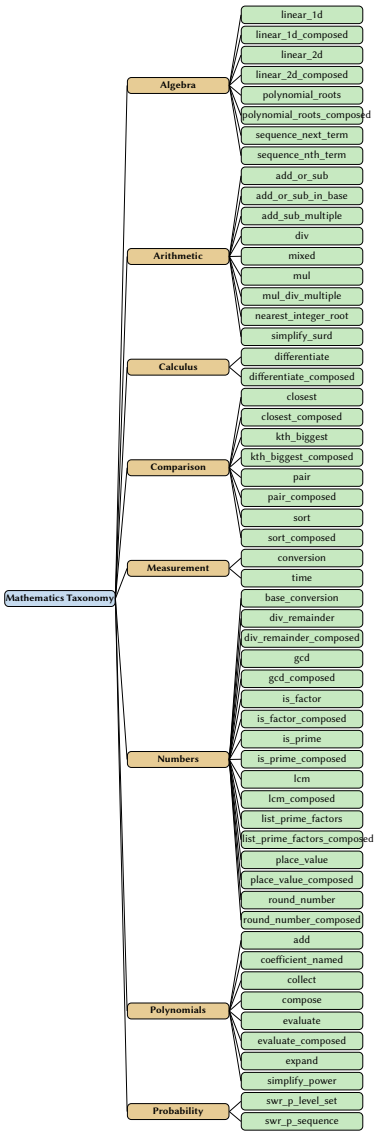

Figure 1: Taxonomy of mathematical topics, organized into major categories including Algebra, Arithmetic, Calculus, Comparison, Measurement, Numbers, Polynomials, and Probability, with their respective subtopics.

3. Methodology

Dataset: The Google DeepMind Mathematics Dataset encompasses eight major mathematical domains: algebra, arithmetic, calculus, comparisons, measurement, number theory, polynomial manipulation, and probability. The 8 domains are further divided into 56 subdomains (Figure 1).

Prompting Strategies: We implemented two distinct prompting strategies: (1) *Meta-Prompt Strategy* employs structured output constraints designed to standardize model responses and eliminate formatting-related evaluation errors, incorporating specific formatting rules for different mathematical problem types. The key principle is that mathematical reasoning failures may stem from output formatting inconsistencies rather than fundamental reasoning deficits. (2) *Chain-of-Thought Strategy* encourages explicit step-by-step reasoning demonstration before final answer generation, testing whether mathematical accuracy improves when models externalize their reasoning processes.

Meta Prompt

```
You are a mathematical problem solver. You will be given a single math question to solve. Your task is
to return only the final answer, without showing any intermediate steps, explanations, or reasoning.
Here is the math question: <question> {question} </question>
Solve this problem internally and return ONLY the final answer.
```

```
CRITICAL FORMAT RULES:
1. For "Solve for [variable]" questions: Return ONLY the numerical value, NOT the equation - Example:
If asked "Solve for x", answer "5" not "x = 5"
2. For "What is the nth term" sequence questions: Return the algebraic expression - Example: "2n + 3"
or "-3n*2 + 5n - 1"
3. For "Simplify" questions: Return the simplified mathematical expression - Keep mathematical notation:
"2*sqrt(3)" not decimal approximations
4. For "List the prime factors" questions: Return a comma-separated list in brackets - Example: [2, 3,
5] for factors 2, 3, and 5
5. For True/False questions: Return exactly "True" or "False"
6. For multiple choice: Return only the letter (A, B, C, D)
7. For numerical calculations: Return the exact number - Use fractions when appropriate: "1/2" not
"0.5"
NEVER include variable names in solutions to "Solve for [variable]" problems.
Provide your final answer now:
```

Chain-of-Thought Prompt

```
You are a mathematical problem solver. Solve the following problem step by step, showing your reasoning
and calculations, then provide your final answer in the specified format.
<question> question </question>
Instructions: 1. Think through this problem step by step 2. Show your mathematical reasoning and
calculations 3. Explain your approach clearly 4. End with your final answer in the exact format
specified below
CRITICAL: You must end your response with: FINAL ANSWER: [your answer here]
CRITICAL FORMAT RULES for FINAL ANSWER: 1. For "Solve for [variable]" questions: Return ONLY the
numerical value, NOT the equation - Example: If asked "Solve for x", answer "FINAL ANSWER: 5" not "FINAL
ANSWER: x = 5" - Example: If asked "Solve for n", answer "FINAL ANSWER: -3" not "FINAL ANSWER: n = -3"
2. For "What is the nth term" sequence questions: Return the algebraic expression - Example: "FINAL
ANSWER: 2*n + 3" or "FINAL ANSWER: -3*n**2 + 5*n - 1"
3. For "Simplify" questions: Return the simplified mathematical expression - Keep mathematical notation:
"FINAL ANSWER: 2*sqrt(3)" not decimal approximations - Preserve exact forms: "FINAL ANSWER: 1/3" not
"FINAL ANSWER: 0.333..."
4. For "List the prime factors" questions: Return a comma-separated list in brackets - Example: "FINAL
ANSWER: [2, 3, 5]" for factors 2, 3, and 5
5. For True/False questions: Return exactly "True" or "False" - Example: "FINAL ANSWER: True" or "FINAL
ANSWER: False"
6. For multiple choice: Return only the letter (a, b, c, d) - Example: "FINAL ANSWER: a"
7. For numerical calculations: Return the exact number - Use fractions when appropriate: "FINAL ANSWER:
1/2" not "FINAL ANSWER: 0.5" - Use integers when the answer is a whole number: "FINAL ANSWER: 7"
NEVER include variable names in solutions to "Solve for [variable]" problems. The question already
tells you what variable to solve for.
Example format: Question: Solve 2x + 5 = 13 for x.
Step 1: I need to isolate x in the equation 2x + 5 = 13 Step 2: Subtract 5 from both sides: 2x + 5 - 5
= 13 - 5 Step 3: Simplify: 2x = 8 Step 4: Divide both sides by 2: x = 8/2 = 4
FINAL ANSWER: 4
Now solve the given problem, showing your work step by step:
```

Experimental Design: Our evaluation compares these prompting strategies across all 56 subtopics of the dataset, with 500 problems per subtopic totaling 28,000 mathematical problems. Each prompting condition was evaluated independently to isolate the effects of prompt design on mathematical reasoning performance. To understand mechanisms underlying prompt effectiveness, we conducted controlled experiments testing three specific hypotheses about meta-prompt underperformance: instruction over-constraint, format anxiety, and internal reasoning suppression across six representative subtopics with varying baseline performance levels.

These subtopics (algebra_polynomial_roots, algebra_sequence_nth_term, arithmetic_mul_div_multiple, arithmetic_simplify_surd, comparison_sort_composed, and numbers_base_conversion) were selected to span different mathematical domains while exhibiting relatively poor meta-prompt performance, allowing us to isolate the effects of prompt design modifications.

4. Results

Our comprehensive evaluation reveals significant differences between prompting strategies. Meta-prompting achieved an overall accuracy of 28.83% (8,073/28,000), while chain-of-thought prompting

achieved 39.02% (10,926/28,000), representing a 10.19 percentage point improvement.

Table 1 presents the direct comparison between prompting strategies across all eight mathematical domains. Chain-of-thought prompting produced substantial improvements in five domains: algebra (+33.10pp), measurement (+29.80pp), polynomials (+18.30pp), arithmetic (+14.25pp), and numbers (+13.73pp). However, it significantly degraded performance in comparison tasks (-32.52pp) and showed minimal change in calculus (-0.80pp) and probability (0.00pp).

Table 1
Comparative Performance: Meta-Prompt vs Chain-of-Thought by Domain

Domain	Meta-Prompt (%)	CoT (%)	Difference (pp)
Measurement	36.90	66.70	+29.80
Arithmetic	35.04	49.29	+14.25
Algebra	15.28	48.38	+33.10
Numbers	33.07	46.80	+13.73
Polynomials	12.35	30.65	+18.30
Comparison	50.40	17.88	-32.52
Calculus	15.90	15.10	-0.80
Probability	3.60	3.60	0.00

Chain-of-thought prompting resulted in 13 subtopics achieving 0% accuracy (see Appendix A Tables 3 and 4), including polynomials__add, polynomials__collect, comparison__sort, and all boolean classification tasks in the numbers domain (is_factor, is_prime, list_prime_factors). In contrast, meta-prompting showed no complete failures. The universal failure of boolean classification tasks under chain-of-thought prompting, despite showing moderate performance under meta-prompting, represents a critical failure mode suggesting that explicit reasoning requirements may interfere with the model's ability to perform direct classification judgments.

5. Analysis and Discussion

5.1. Domain-Specific Effectiveness Patterns

The dramatic improvement in algebra (+33.10pp) and measurement (+29.80pp) suggests that step-by-step reasoning is particularly beneficial for problems requiring sequential logical operations and unit transformations. Conversely, the catastrophic decline in comparison tasks (-32.52pp) indicates that explicit reasoning requirements can actively impair performance on problems involving pattern recognition or relative ordering. The complete failure of sorting tasks (0% accuracy) under chain-of-thought prompting suggests that these operations may require holistic processing rather than sequential decomposition.

5.2. Mechanisms of Meta-Prompt Underperformance

Our experimental results reveal three hypotheses explaining why structured meta-prompts consistently underperform: **(1) Instruction Over-Constraint (2) Format Anxiety (3) Internal Reasoning Suppression**

5.2.1. Hypothesis A- Instruction Over-Constraint

We compared meta-prompts against simplified prompts containing only essential problem-solving instructions without formatting constraints. This tests whether extensive formatting rules create cognitive load that interferes with mathematical reasoning.

Results: Systematic performance degradation appears across all tested subtopics when transitioning from simple prompts to meta-prompts. The most dramatic examples include:

- `arithmetic_mul_div_multiple`: 74% (simple) to 7.2% (meta-prompt) - 66.8pp decline
- `numbers_base_conversion`: 78% (simple) to 8.6% (meta-prompt) - 69.4pp decline

- `arithmetic_simplify_surd`: 46% (simple) to 3.6% (meta-prompt) - 42.4pp decline

Analysis: This consistent pattern across diverse problem types suggests that the performance degradation stems from cognitive load management issues rather than domain-specific difficulties. The dual-task interference effect appears when models must simultaneously solve mathematical problems while adhering to complex formatting requirements, overwhelming available cognitive resources.

5.2.2. Hypothesis B- Format Anxiety

We implemented a three-way comparison testing: (1) no formatting constraints, (2) simple formatting guidance, and (3) full meta-prompt formatting. This isolates the impact of format awareness itself on mathematical performance.

Results: Even minimal formatting guidance proves detrimental across multiple subtopics:
`arithmetic_simplify_surd`:

- No formatting constraints: 42%
- Simple formatting: 4%
- Meta-prompts: 3.6%

`numbers_base_conversion`:

- No formatting constraints: 64%
- Simple formatting: 8%
- Meta-prompts: 8.6%

Analysis: The dramatic performance drop from no-formatting to simple-formatting conditions (38pp and 56pp respectively) demonstrates that format awareness itself impairs performance independent of formatting complexity. This suggests a psychological effect where awareness of evaluation criteria paradoxically degrades natural problem-solving processes.

5.2.3. Hypothesis C- Internal Reasoning Suppression

We tested three reasoning conditions: (1) no reasoning allowed (solve internally), (2) brief reasoning permitted, and (3) full reasoning encouraged. This directly tests whether the meta-prompt directive to "solve internally" suppresses necessary cognitive processes.

Results: Dramatic improvements occur when reasoning is permitted:
`arithmetic_simplify_surd`:

- No reasoning: 2%
- Brief reasoning: 48% (+46pp)
- Full reasoning: 52% (+50pp)

`comparison_sort_composed`:

- No reasoning: 28%
- Brief reasoning: 76% (+48pp)
- Full reasoning: 76% (+48pp)

Analysis: The substantial improvements when reasoning is permitted demonstrate that mathematical problem-solving in current language models requires explicit externalization of intermediate steps. The instruction to solve internally actively suppresses this natural process, leading to performance degradation. This suggests fundamental differences between human and machine cognitive architectures for mathematical reasoning.

Table 2
Complete Hypothesis Testing Results

Subtopic	Hyp. A	Hyp. B		Hyp. C			Meta-Prompt
	Simple	No-Fmt	Fmt	No-Rsn	Brief	Full	Baseline
algebra_polynomial_roots	24%	20%	4%	6%	20%	20%	4.4%
algebra_sequence_nth_term	60%	62%	36%	36%	46%	50%	15.8%
arithmetic_mul_div_multiple	74%	66%	16%	14%	54%	56%	7.2%
arithmetic_simplify_surd	46%	42%	4%	2%	48%	52%	3.6%
comparison_sort_composed	2%	6%	14%	28%	76%	76%	12.4%
numbers_base_conversion	78%	64%	8%	8%	56%	62%	8.6%

5.3. Cross-Hypothesis Validation

The experimental results provide strong quantitative support for all three hypotheses:

Instruction Over-Constraint (Hypothesis A): Meta-prompts consistently underperform simpler approaches across 5/6 subtopics, with an average degradation of 52.7 percentage points.

Format Anxiety (Hypothesis B): Even minimal formatting constraints cause substantial performance drops, with an average decline of 41.8 percentage points from no-formatting to simple-formatting conditions.

Internal Reasoning Suppression (Hypothesis C): Allowing reasoning steps provides substantial improvements, with an average gain of 42.3 percentage points from no-reasoning to full-reasoning conditions.

These findings show that meta-prompting interferes with the cognitive processes needed for mathematical reasoning in language models. We propose two design principles for prompts, and the consistency of these effects across diverse problem types suggests they reflect fundamental mechanisms of how current models process mathematical information, rather than domain-specific artifacts.

1. **Minimize Cognitive Load:** Reduce formatting constraints to essential requirements only
2. **Enable Explicit Reasoning:** Allow or encourage externalization of intermediate reasoning steps

5.4. Implications for Mathematical Reasoning Systems

Our findings reveal a fundamental tension between evaluation convenience and reasoning effectiveness. Structured prompts designed to standardize outputs for automated evaluation consistently impair mathematical reasoning performance, suggesting that evaluation methodology may need to accommodate natural reasoning processes rather than constraining them. The domain-dependent effectiveness patterns indicate that optimal prompting strategies must be tailored to specific mathematical problem characteristics. Sequential operations (algebra, arithmetic) benefit from explicit reasoning, while holistic operations (comparison, pattern recognition) may require different approaches. The inability to separate thinking from showing work suggests that current language models' mathematical reasoning capabilities may be fundamentally different from human cognitive architecture.

6. Conclusion

This comprehensive evaluation of GPT-4o's mathematical reasoning capabilities across 28,000 problems spanning 56 mathematical subtopics demonstrates that prompting effectiveness depends critically on problem domain characteristics. The 10.19 percentage point overall improvement from chain-of-thought prompting masks dramatic domain-dependent variations, with algebra showing exceptional gains (+33.10pp) while comparison tasks experienced substantial degradation (-32.52pp). Our analysis identifies three mechanisms underlying meta-prompt underperformance: cognitive resource allocation conflicts, performance degradation under explicit evaluation criteria, and suppression of intermediate reasoning processes. The emergence of complete failure modes under chain-of-thought prompting with 13 subtopics achieving 0% accuracy despite nonzero meta-prompt performance highlights the need for adaptive prompting strategies. These results have important implications for developing mathematical reasoning systems that can perform consistently across diverse mathematical problem-solving domains.

7. Limitations and Future Directions

This study focuses exclusively on GPT-4o performance on school-level mathematics problems. Future research should investigate whether these patterns generalize to other language models, advanced mathematical topics, and alternative prompting strategies. The complete failure of certain subtopics under chain-of-thought prompting highlights the need for hybrid approaches that can adaptively select prompting strategies based on problem characteristics.

Acknowledgments

Thanks to the developers of ACM consolidated LaTeX styles and to the developers of Elsevier updated LaTeX templates.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] D. Saxton, E. Grefenstette, F. Hill, P. Kohli, Analysing mathematical reasoning abilities of neural models, arXiv preprint arXiv:1904.01557 (2019). URL: https://arxiv.org/abs/1904.01557.

[2] J. Dekoninck, MathArena: Evaluating LLMs on uncontaminated math competitions, arXiv preprint (2025). URL: https://arxiv.org/abs/2505.23281, arXiv:2505.23281.

[3] J. Dekoninck, Proof or bluff? evaluating LLMs on 2025 USA math olympiad, arXiv preprint (2025). URL: https://arxiv.org/abs/2506.00309, arXiv:2506.00309.

[4] H. Mahdavi, A. Hashemi, M. Daliri, P. Mohammadipour, A. Farhadi, S. Malek, Y. Yazdanifard, A. Khasahmadi, V. Honavar, Brains vs. bytes: Evaluating llm proficiency in olympiad mathematics, 2025. URL: https://arxiv.org/abs/2504.01995. arXiv:2504.01995.

[5] S. Liu, MathBench: Evaluating the theory and application proficiency of LLMs with a hierarchical mathematics benchmark, arXiv preprint (2024). URL: https://arxiv.org/abs/2503.21934, arXiv:2503.21934.

[6] M. Gupta, Mathify: Evaluating large language models on mathematical problem solving tasks, arXiv preprint (2024). URL: https://arxiv.org/abs/2405.14804, arXiv:2405.14804.

[7] R. Wang, R. Wang, Y. Shen, C. Wu, Q. Zhou, R. Chandra, Evaluation of llms for mathematical problem solving, 2025. URL: https://arxiv.org/abs/2506.00309. arXiv:2506.00309.

[8] X. Xu, Can LLMs solve longer math word problems better?, arXiv preprint (2024). URL: https://arxiv.org/abs/2504.01995, arXiv:2504.01995.

[9] J. Vidal, Evaluation of the performance of state-of-the-art large language models (LLMs) in solving math word problems, SSRN (2024). URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4902960, sSRN:4902960.

[10] A. Satpute, N. Giessing, A. Greiner-Petter, M. Schubotz, O. Teschke, A. Aizawa, B. Gipp, Can llms master math? investigating large language models on math stack exchange, 2024. URL: https://arxiv.org/abs/2404.00344. arXiv:2404.00344.

[11] A. Baral, Can LLMs understand math? – exploring the pitfalls in mathematical reasoning, arXiv preprint (2024). URL: https://arxiv.org/abs/2405.12209, arXiv:2405.12209.

A. Complete Results

Table 3
Complete Meta-Prompting Performance Results for All 56 Subtopics

Domain	Subtopic	Accuracy (%)
Algebra	algebra__linear_1d	23%
	algebra__linear_1d_composed	15.4%
	algebra__linear_2d	9.4%
	algebra__linear_2d_composed	15%
	algebra__polynomial_roots	4.4%
	algebra__polynomial_roots_composed	3.8%
	algebra__sequence_next_term	35.4%
	algebra__sequence_nth_term	15.8%
Arithmetic	arithmetic__add_or_sub	86.6%
	arithmetic__add_or_sub_in_base	31.8%
	arithmetic__add_sub_multiple	26.4%
	arithmetic__div	60.4%
	arithmetic__mixed	8.6%
	arithmetic__mul	64.8%
	arithmetic__mul_div_multiple	7.2%
	arithmetic__nearest_integer_root	26%
	arithmetic__simplify_surd	3.6%
Calculus	calculus__differentiate	28.4%
	calculus__differentiate_composed	3.4%
Comparison	comparison__closest	67.6%
	comparison__closest_composed	33.8%
	comparison__kth_biggest	49.6%
	comparison__kth_biggest_composed	30.2%
	comparison__pair	81%
	comparison__pair_composed	44%
	comparison__sort	84.6%
	comparison__sort_composed	12.4%
Measurement	measurement__conversion	47.2%
	measurement__time	26.6%
Numbers	numbers__base_conversion	8.6%
	numbers__div_remainder	28.8%
	numbers__div_remainder_composed	8.2%
	numbers__gcd	34.4%
	numbers__gcd_composed	18.4%
	numbers__is_factor	67%
	numbers__is_factor_composed	52.6%
	numbers__is_prime	52.2%
	numbers__is_prime_composed	51.6%
	numbers__lcm	20.8%
	numbers__lcm_composed	9.2%
	numbers__list_prime_factors	7.8%
	numbers__list_prime_factors_composed	6%
	numbers__place_value	69.2%
	numbers__place_value_composed	17.4%
	numbers__round_number	93.2%
	numbers__round_number_composed	16.8%
Polynomials	polynomials__add	0.4%
	polynomials__coefficient_named	25.4%
	polynomials__collect	42%
	polynomials__compose	3.8%
	polynomials__evaluate	15.8%
	polynomials__evaluate_composed	6.6%
	polynomials__expand	0.6%
	polynomials__simplify_power	4.2%
Probability	probability__swr_p_level_set	2.8%
	probability__swr_p_sequence	4.4%

Table 4
Complete Chain-of-Thought Prompting Performance Results for All 56 Subtopics

Domain	Subtopic	Accuracy (%)
Algebra	algebra__linear_1d	83.0%
	algebra__linear_1d_composed	77.2%
	algebra__linear_2d	87.6%
	algebra__linear_2d_composed	54.4%
	algebra__polynomial_roots	0.8%
	algebra__polynomial_roots_composed	3.2%
	algebra__sequence_next_term	80.8%
	algebra__sequence_nth_term	0.0%
Arithmetic	arithmetic__add_or_sub	88.4%
	arithmetic__add_or_sub_in_base	22.4%
	arithmetic__add_sub_multiple	85.8%
	arithmetic__div	44.8%
	arithmetic__mixed	35.0%
	arithmetic__mul	74.6%
	arithmetic__mul_div_multiple	34.6%
	arithmetic__nearest_integer_root	49.8%
	arithmetic__simplify_surd	8.2%
Calculus	calculus__differentiate	11.4%
	calculus__differentiate_composed	18.8%
Comparison	comparison__closest	32.2%
	comparison__closest_composed	14.0%
	comparison__kth_biggest	36.4%
	comparison__kth_biggest_composed	18.6%
	comparison__pair	34.4%
	comparison__pair_composed	7.4%
	comparison__sort	0.0%
	comparison__sort_composed	0.0%
Measurement	measurement__conversion	89.8%
	measurement__time	43.6%
Numbers	numbers__base_conversion	39.0%
	numbers__div_remainder	73.2%
	numbers__div_remainder_composed	87.2%
	numbers__gcd	65.2%
	numbers__gcd_composed	81.0%
	numbers__is_factor	0.0%
	numbers__is_factor_composed	0.0%
	numbers__is_prime	0.0%
	numbers__is_prime_composed	0.0%
	numbers__lcm	42.4%
	numbers__lcm_composed	51.4%
	numbers__list_prime_factors	0.0%
	numbers__list_prime_factors_composed	0.0%
	numbers__place_value	95.8%
	numbers__place_value_composed	89.4%
	numbers__round_number	84.6%
	numbers__round_number_composed	86.4%
Polynomials	polynomials__add	0.0%
	polynomials__coefficient_named	91.2%
	polynomials__collect	0.0%
	polynomials__compose	0.0%
	polynomials__evaluate	84.4%
	polynomials__evaluate_composed	69.2%
	polynomials__expand	0.0%
	polynomials__simplify_power	0.4%
Probability	probability__swr_p_level_set	4.4%
	probability__swr_p_sequence	2.8%