JOURNAL OF
EMERGING INVESTIGATORS

Article

Evaluating the impact of prompting styles on LLM accuracy for AIME math questions

Vasu Ganesa[1], Lars Holdijk[1]
[1] American High School, Fremont, California

SUMMARY

Researchers are beginning to use LLMs to solve math problems but their ability to perform multi-step reasoning remains unclear. In this study, we investigated whether various prompting styles could enhance the problem-solving abilities of Large Language Models (LLMs) for difficult math problems such as those from the American Invitational Mathematics Exam (AIME). We hypothesized that, of the five prompting styles we tested, providing detailed AIME solutions (previous AIME problems with step-by-step solutions would lead to the highest LLM accuracy on new AIME problems. To test this hypothesis, we used three LLMs (ChatGPT Free, ChatGPT Plus, and Gemini) as tools to examine the effects of five prompting styles: a baseline with the problem alone, a study guide, past AIME problems with the answer alone, past AIME problems with detailed solutions, and Olympiad problems with detailed solutions. Our dataset included four AIME exams (2024 AIME I, 2024 AIME II, 2025 AIME I, and 2025 AIME II), totaling 60 problems. Although performance varied, particularly for Gemini, the detailed solution style did not significantly improve accuracy over the baseline or other prompting styles. This suggests that, compared to other prompting styles, detailed AIME solutions do not significantly improve LLM performance on multi-step math problems. Our work provides one approach for testing LLM performance in math and shows the necessity of deeper approaches beyond prompting to improve LLM performance.

INTRODUCTION

Large Language Models (LLMs) are a form of artificial intelligence that have been trained on enormous text datasets to recognize language patterns and generate coherent responses to user prompts (1). They represent a major technological advancement of the 21st century, and are revolutionizing fields such as computer science, by assisting with software development (1). LLMs have evolved based on transformer-based architecture, which allows them to comprehend the input a user gives and provide a coherent response to the question (1,2). A transformer is a type of neural network that uses an "attention" system to determine which words in a sentence matter most, helping the model understand and respond more accurately (1). The architecture of LLMs is built upon billions of parameters, enabling them to provide responses and gather information from many sources (1). Researchers argue that LLMs have three key abilities which enable them to do well in their tasks (1). The first is context learning, in which they learn a new task from examples which are formed during the inference time (1). The second is instruction following, in which LLMs can execute tasks without the necessity of examples (1). Finally, the third is multi-step reasoning, in which they solve a difficult task by breaking down the task into different intermediate steps (1). This threefold method is the way that LLMs perform tasks that require multi-step reasoning such as solving complex math (2). These methods work because of LLM transformer architecture (2,3). The transformer can perform parallel processing in comparison to the sequential processing of a Recurrent Neural Network (RNN) (2). This feature allows transformers to understand basic units of text called "tokens" and identify key relationships between them (2). This breakthrough has allowed LLMs to revolutionize the field of Natural Language Processing (NLP) which focuses on allowing computers to understand and generate human language (2).

Even though LLMs excel in NLP, they struggle when it comes to complex math problems (3). For example, models trained on the Mathematics Aptitude Test of Heuristics (MATH) dataset, which contained 12,500 problems at American Mathematics Competition (AMC) and American Invitational Mathematics Exam (AIME) difficulty levels, frequently made logical mistakes or gave nonsensical answers (3). The AMC is a 25-question multiple choice test that gets progressively harder and covers counting and probability, number theory, geometry, and algebra (3). The AIME builds on these same topics but is more challenging, requiring three-digit numerical answers instead of multiple choice (3). Unlike basic math problems that can be solved with formulas, AMC and AIME problems require creativity and reasoning (3). Previous research has explored ways to improve LLM performance on such reasoning tasks (4). Minerva, a specialized math model, was shown to perform better on math tests (4). However, Minerva still struggles compared to the ability of many students who excel in competitive math (4).

One of the main problems with LLMs is that they cause systematic errors, specifically when problems require multi-step reasoning (5). LLMs are not able to check their steps at each stage, unlike mathematicians, which causes LLMs to make simple logic mistakes that deviate from the correct solution (5). This is referred to as "hallucination" in mathematical reasoning and shows the limits of LLMs' problem-solving capabilities (5). This limitation was noted in a study which showed that LLMs often hallucinated plausible but incorrect solutions to problems (6). Additionally, this study shows the importance of prompting styles and how they can be used to make LLMs perform better (6). Consequently, prompt engineering has emerged as an important field, aimed at improving LLM performance on complex math problems (7). A widely studied technique has been Chain of Thought prompting (CoT), which is when the LLM is told to

https://doi.org/10.59720/25-137

provide intermediate reasoning steps (7). This method has significantly improved LLM performance on word and logic problems (8). However, it is unclear whether these prompting strategies will improve LLMs performance on complex math problems. We predicted that detailed AIME solutions would function similarly by offering structured examples of step-by-step reasoning. Therefore, we hypothesized that, of the five prompting styles tested, providing detailed AIME solutions would lead to the highest LLM accuracy on new AIME problems. The five prompting styles were chosen as they gradually provided more guidance to the LLM. The baseline condition was provided only the problem, while other styles provided additional context such as problem-solving insights, Olympiad problems with detailed solutions, example problems with answers and example problems with full solutions. In this study, we found that providing detailed AIME solutions did not significantly improve LLM accuracy compared to the baseline or other prompting styles. Although some models showed variation across prompting styles, these differences were not statistically significant after correction for multiple comparisons. These findings suggest that prompt engineering alone may not be enough to improve LLM performance on complex, multi-step math problems, and that future progress may require deeper model-level improvements rather than only changes in how problems are presented.

RESULTS

To evaluate whether detailed AIME solutions led to better LLM performance compared to other prompting styles, we tested three models using problems from four different AIME exams (2024 AIME I, 2024 AIME II, 2025 AIME I, and 2025 AIME II), totaling 60 problems. The three models we tested were ChatGPT Free (GPT-3.5), ChatGPT Plus (GPT-4-turbo), and Gemini (Gemini 1.5 Flash), using five prompting styles. The five prompting styles that we used were a baseline with the problem alone, a study guide, past AIME problems with the answer alone, past AIME problems with detailed solutions, and Olympiad problems with detailed solutions. We used *Cochran's Q* Test to determine if there were significant differences in accuracy across the five prompting styles for each model. To check individual prompting styles against the baseline (Just the Problem prompting style), we ran McNemar's Test for each style and adjusted the *p*-values using the Benjamini–Hochberg procedure. After running the McNemar tests and adjusting the *p*-values using the Benjamini–Hochberg procedure, all adjusted *p*-values were greater than 0.05. Thus, no prompting style significantly outperformed the baseline (**Figure 1**). Although Gemini showed more variation across styles (*Cochran's Q*=10.623, *p*=0.031), these differences did not hold up statistically once multiple comparison adjustments were applied (adjusted *p*-values ≥ 0.053).

Since LLMs can produce slightly different answers each time, we repeated the experiments in three sessions to check for reproducibility. The results were nearly identical in every run (**Figure 2**) (Original run: ChatGPT Free: $Q = 5.760$, $p = 0.218$;

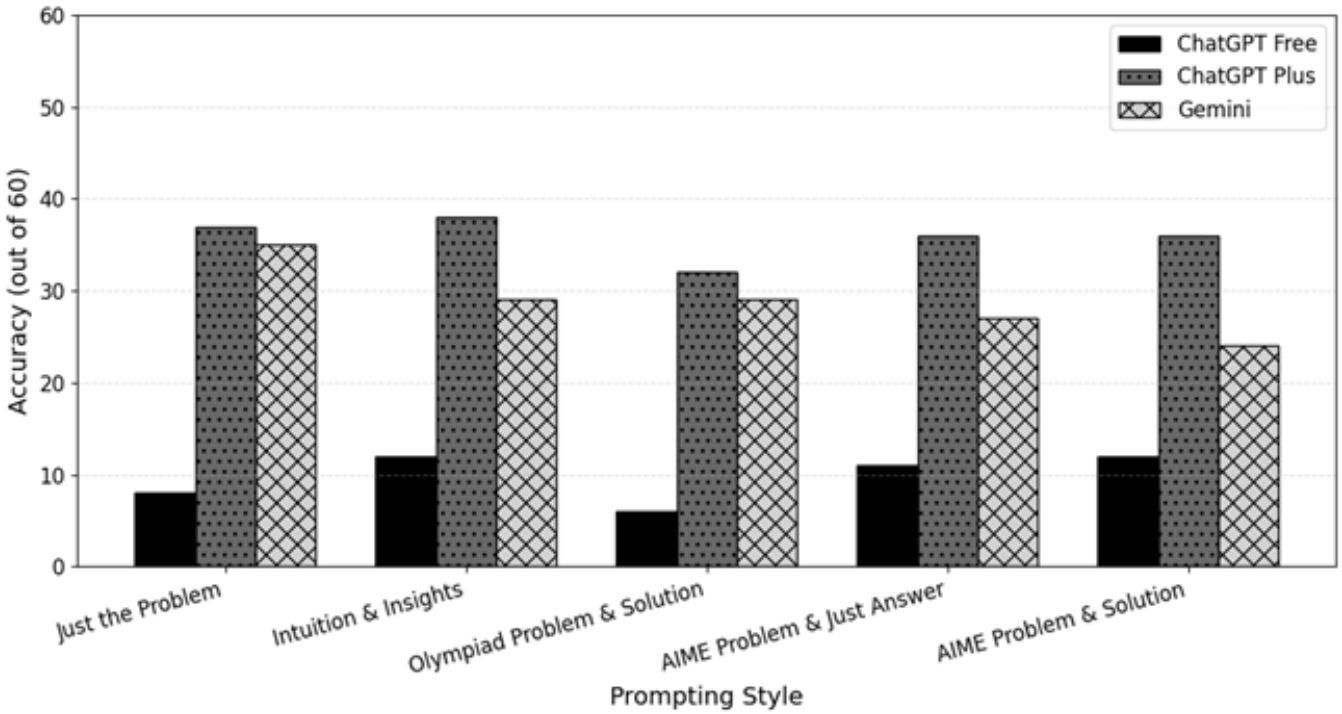

Figure 1: Comparison of LLM accuracy by prompting style across three models in the original experimental run. This bar graph represents the scores that the three models (ChatGPT Free, ChatGPT Plus, and Gemini) received on all AIME questions (n=60) given various prompting styles in the original experimental run. The styles included giving Just the Problem, Intuition and Insights, Olympiad Problem and Solution, AIME Problem and Just Answer, and AIME Problem and Solution. Statistical significance was evaluated using the Cochran Q test and McNemar's test and was defined as $p < 0.05$. ChatGPT Free showed little variation across prompting styles (*Cochran's Q*=5.760, *p*=0.218) and ChatGPT Plus also showed vary little variation (Q=4.522, *p*=0.340), and Gemini showed the greatest variation across styles (Q=10.623, *p*=0.031), though none of these differences were statistically significant after Benjamini-Hochberg correction (adjusted $p \geq 0.053$).

https://doi.org/10.59720/25-137

ChatGPT Plus: $Q = 4.522$, $p = 0.340$; Gemini: $Q = 10.623$, $p = 0.031$, Run 1: ChatGPT Free: $Q = 5.652$, $p = 0.227$; ChatGPT Plus: $Q = 4.627$, $p = 0.328$; Gemini: $Q = 10.587$, $p = 0.032$. Run 2: ChatGPT Free: $Q = 5.812$, $p = 0.214$; ChatGPT Plus: $Q = 4.712$, $p = 0.318$; Gemini: $Q = 10.419$, $p = 0.034$; all adjusted p-values ≥ 0.053). Variation occurred when models answered a problem incorrectly; sometimes they gave a different wrong answer for a particular problem, but their total exam accuracy remained nearly the exact same across all prompting styles. The models' performance patterns were highly consistent across runs, even with their natural randomness (**Figure 2**).

All three models answered the earlier questions more accurately and became progressively worse as we presented the later questions, which was expected since the later questions were meant to be more complex (**Figure 1**). When looking at each model individually, ChatGPT Free showed little variation across prompting styles (*Cochran's Q*=5.760, p=0.218) and ChatGPT Plus also showed vary little variation (Q=4.522, p=0.340), and Gemini showed the greatest variation across styles, though none of these differences were statistically significant (Q=10.623, p=0.031) (**Figure 1**).

Finally, we noticed differences in the reasoning and solutions that each model provided. Even in incorrect responses, ChatGPT Plus provided more detailed explanations on how it solved a problem. However, Gemini tended to provide shorter and less complete answers. For example, when solving geometry problems on the AIME, ChatGPT plus generated a multi-step solution that involved constructing a figure in the coordinate plane and decomposing it, while Gemini produced a much shorter solution that omitted several important steps and jumped quickly to a final expression. While none of these explanations impacted the final scores, it illustrated that both models operated quite differently and presented a unique type of reasoning.

DISCUSSION

Our results suggest that prompting LLMs with detailed AIME solutions does not lead to improved performance on AIME math problems compared to other prompting styles. After multiple comparison correction, none of the differences between prompting styles remained statistically significant. Therefore, contrary to our original hypothesis that the detailed AIME solutions would perform best, we found no consistent improvement.

AIME problems are extremely complicated, and they cannot be solved using a simple formula. Instead, the vast majority of these problems make students think "outside the box." LLMs rely on statistical language pattern prediction rather than true critical thinking, which limits their ability to problem solve (1). Previous work has shown that providing examples to train LLMs helps in solving simple problems such as recalling a single fact (e.g. "What year was Justin Bieber born?"), but may not be sufficient for more challenging problems that require multiple facts (e.g. "Who won the master's tournament the year Justin Bieber was born?") (9). As seen in our study, even with inclusion of helpful example problems and detailed solutions, we did not observe an improvement in performance. This is similar to the way humans learn; just giving a few example problems does not improve one's mathematical prowess. Instead, mastery requires repeated

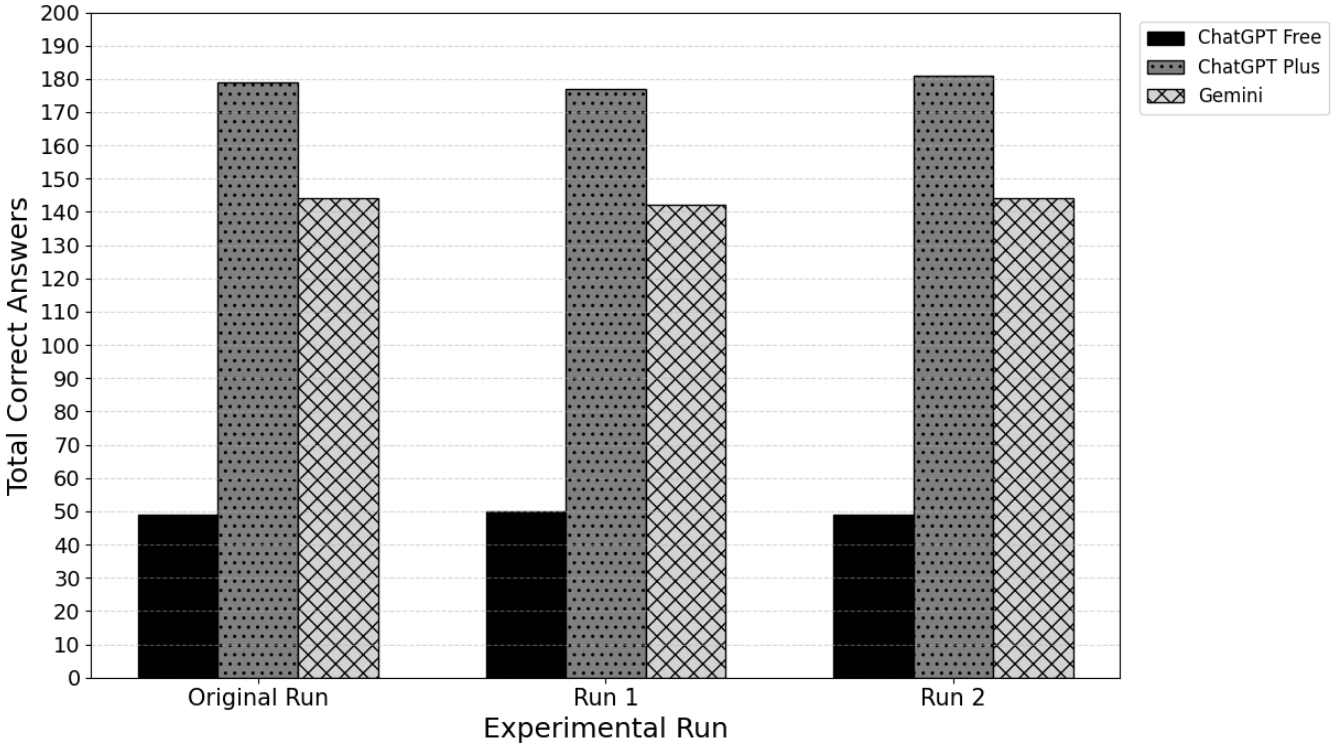

Figure 2: Reproducibility of LLM accuracy across three experimental runs and all five prompting styles. This bar graph represents the total number of correct responses that the three models (ChatGPT Free, ChatGPT Plus and Gemini) produced across all AIME questions during three separate experimental runs. The runs (Original Run, Run 1, and Run 2) were the full evaluation across all five prompting styles.

https://doi.org/10.59720/25-137

practice with hundreds, perhaps thousands, of problems. An unexpected observation was that Gemini's performance appeared lower when using the detailed solution prompting style (**Figure 1**). A plausible explanation could be that long prompts burden the model with additional information, causing it to lose focus on the actual question. This is called "context window overload", where the model simply cannot retain all parts of the input prompt and is unable to prioritize the important information (10). Another reason could be that the prompts with examples confused the model rather than explaining how to arrive at the solution. Previous research has shown that when examples are too far from the original problem domain, they can mislead the model (11).

Our study also has limitations. This study assessed final accuracy only and did not evaluate reasoning quality. A model may generate valid intermediate steps yet provide a wrong solution at the end. In a binary accuracy measure, it would be impossible to see these nuances. Therefore, future studies should use scoring systems to assess the quality of the reasoning by giving partial credit for correct steps. This way, we could observe if prompting styles are genuinely improving the reasoning, rather than just seeing if different prompting styles improve accuracy.

Our findings might also reflect the deeper shortcomings of LLMs in math. Multi-step logic, abstraction, or analogical reasoning might be difficult for LLMs due to architectural or training designs, and prompt engineering strategies may not be sufficient to overcome this difficulty. This suggests that further developments in LLM reasoning and skill will be less dependent on prompt engineering and more on model-level advances.

These observations also are relevant in the field of education. Many people have expressed concern that students will learn how to develop sophisticated prompting techniques that could give them an advantage on exams. We found that current LLMs did poorly on more advanced areas of math, especially on AIME questions. Therefore, the risk of cheating at this level is currently minimal; however, as LLMs develop, this may become something to monitor. Ultimately, our results illustrate the present limitations of prompt engineering and present the future directions for both enhancing the ability of LLMs in performing high-order reasoning problems.

MATERIALS AND METHODS

LLM Selection and Testing

This study used three LLMs: ChatGPT Free (GPT-3.5, OpenAI), ChatGPT Plus (GPT-4-turbo, OpenAI), and Gemini (Gemini 1.5 Flash, Google DeepMind). All LLMs were accessed through their respective websites: ChatGPT Free and Plus via chat.openai.com and Gemini via gemini.google.com. The dataset consisted of four AIME exams (2024 I, 2024 II, 2025 I, 2025 II), for a total of 60 problems. Each problem was presented separately to avoid memory effects. After completing all questions for one prompting style, the session was cleared before starting the next.

Because LLM outputs can vary between sessions, the full evaluation was repeated for each model and prompting style in three independent chat sessions. Across these repeated runs, the total accuracy for each model–style combination was the same or differed by at most one problem, indicating that the results were stable despite normal stochastic variation. Therefore, one representative run was used for the statistical analysis.

Prompting Styles and Problem Selection

To eliminate carryover effects or bias introduced by prior prompts, each style was tested in a separate chat session. For every problem, the LLM first received the prompt for that style and then immediately received the AIME problem. This means the prompting material (for example, detailed solutions, numerical answers, or study-guide insights) was shown before each question, not only once at the beginning. After the LLM produced an answer, the response was recorded and checked for accuracy before moving to the next problem.

After completing all problems for a prompting style, a new chat session was opened and the LLM was instructed not to remember anything from earlier chats. This process was repeated for all five prompting styles and the baseline condition.

The five prompting styles were as follows. Just the Problem was the statement of the problem and served as the baseline. Insights and Intuition was a compilation of AIME strategies and background information on the four main topics (Counting and Probability, Number Theory, Geometry, and Algebra) tested on the AIME (13). Olympiad Problem and Solution used United States of America Junior Mathematical Olympiad (USAJMO) problems that included full detailed solutions, with one problem chosen per topic. Olympiad problems, such as those from the USAJMO, are proof-based and require full written solutions rather than the three-digit numerical answers used on the AIME. AIME Problem and Answer included one past AIME problem from each topic paired only with its numerical answer. AIME Problem and Solution used the same problems as the previous condition but included the full solutions rather than just the answers.

Prompt Formatting and Statistical Testing Code

For each problem, the LLM received a mark of 1 for a correct answer and 0 for an incorrect answer. The number of correct responses for each prompting style and each LLM was then totaled.

All statistical calculations were completed in Python (v 3.11). The statistical analyses were implemented using the statsmodels package (14). Visualizations were created using Matplotlib and Seaborn (15), and NumPy was used for array manipulation and construction of the binary accuracy matrices (16). All analyses were run in Python with statsmodels 0.14.1, NumPy 1.26.4, Matplotlib 3.8.2, and Seaborn 0.13.2. The full Python script used for these analyses can be found at:https://github.com/einstefer/aime-llm-prompting/blob/main/code%20for%20my%20project

Statistical Methods

In this study, each response was coded as correct (1) or incorrect (0), making the outcomes binary. *Cochran's Q* Test was chosen because it is designed for repeated measures with binary outcomes across multiple related conditions. McNemar's Test was used to compare each prompting

https://doi.org/10.59720/25-137

style directly to the baseline (Just the Problem), because it measures whether two paired binary outcomes differ. The McNemar p-values were adjusted using the Benjamini–Hochberg procedure to account for the false-positive risk associated with multiple comparisons.

p-values were used to determine whether observed differences in accuracy were greater than would be expected by chance. A threshold of $p < 0.05$ was used for statistical significance. The inclusion of multiple-comparison corrections ensured that statistical conclusions were not driven by inflation of Type I error due to repeated pairwise tests.

All data were visualized using Python's Matplotlib and Seaborn libraries, and summary statistics were compiled for all prompting styles and LLMs (15).

Received: May 19, 2025
Accepted: January 08, 2026
Published: July 26, 2026

REFERENCES

1. Minaee, Shervin, *et al.* "Large Language Models: A Survey." *arXiv*, 20 Feb. 2024, https://doi.org/10.48550/arXiv.2402.06196
2. Vaswani, Ashish, *et al.* "Attention Is All You Need." *arXiv*, 2 Aug. 2023, https://doi.org/10.48550/arXiv.1706.03762
3. Hendrycks, Dan, *et al.* "Measuring Mathematical Problem Solving with the MATH Dataset." *arXiv*, 8 Nov. 2021, https://doi.org/10.48550/arXiv.2103.03874
4. Lewkowycz, Aitor, *et al.* "Solving Quantitative Reasoning Problems with Language Models." *arXiv*, 1 July 2022, https://doi.org/10.48550/arXiv.2206.14858
5. Frieder, Simon, *et al.* "Mathematical Capabilities of ChatGPT." *arXiv*, 20 July 2023, https://doi.org/10.48550/arXiv.2301.13867
6. Sun, Yuhong, *et al*. "Benchmarking Hallucination in Large Language Models Based on Unanswerable Math Word Problem." *arXiv*, 6 Mar. 2024, https://doi.org/10.48550/arXiv.2403.03558
7. Wei, Jason, *et al.* "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models." *arXiv*, 10 Jan. 2023, https://doi.org/10.48550/arXiv.2201.11903
8. Kojima, Takeshi, *et al.* "Large Language Models Are Zero-Shot Reasoners." *arXiv*, 29 Jan. 2023, https://doi.org/10.48550/arXiv.2205.11916
9. Press, Ofir, *et al.* "Measuring and Narrowing the Compositionality Gap in Language Models." *arXiv*, 17 Oct. 2023, https://doi.org/10.48550/arXiv.2210.03350
10. Liu, Nelson F., *et al.* "Lost in the Middle: How Language Models Use Long Contexts." *arXiv*, 20 Nov. 2023, https://doi.org/10.48550/arXiv.2307.03172
11. Min, Sewon, *et al.* "Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?" *arXiv*, 20 Oct. 2022, https://doi.org/10.48550/arXiv.2202.12837
12. Zhang, Yukun. "Unraveling Text Generation in LLMs: A Stochastic Differential Equation Approach."*arXiv*, 17 Aug. 2024, https://doi.org/10.48550/arXiv.2408.11863
13. BOGTRO. AIME Need-to-Know Study Guide. AoPS, www.aops.com. Accessed 15 Mar. 2025
14. Seabold, Skipper, and Josef Perktold. "Statsmodels: Econometric and Statistical Modeling with Python." *Proceedings of the 9th Python in Science Conference*, 2010, https://doi.org/10.25080/Majora-92bf1922-011
15. Hunter, John D. "Matplotlib: A 2D Graphics Environment." *Computing in Science and Engineering*, vol. 9, no. 3, 2007, pp. 90–95, https://doi.org/10.1109/MCSE.2007.55
16. Harris, Charles R., *et al.* "Array Programming with NumPy." *Nature*, vol. 585, no. 7825, 2020, pp. 357–362, https://doi.org/10.1038/s41586-020-2649-2