CHAMP: A Competition-level Dataset for Fine-Grained Analyses of LLMs' Mathematical Reasoning Capabilities

Yujun Mao
Boston University
amao@bu.edu

Yoon Kim
MIT CSAIL
yoonkim@mit.edu

Yilun Zhou
Salesforce Research
yilun.zhou@salesforce.com

https://yujunmao1.github.io/CHAMP/

Abstract

Recent large language models (LLMs) have shown indications of mathematical reasoning ability on challenging competition-level problems, especially with self-generated verbalizations of intermediate reasoning steps (i.e., chain-of-thought prompting). However, current evaluations mainly focus on the end-to-end final answer correctness, and it is unclear whether LLMs can make use of helpful side information such as problem-specific hints. In this paper, we propose a challenging benchmark dataset for enabling such analyses. The Concept and Hint-Annotated Math Problems (CHAMP) consists of high school math competition problems, annotated with concepts, or general math facts, and hints, or problem-specific tricks. These annotations allow us to explore the effects of additional information, such as relevant hints, misleading concepts, or related problems. This benchmark is difficult, with the best model only scoring 58.1% in standard settings. With concepts and hints, performance sometimes improves, indicating that some models can make use of such side information. Furthermore, we annotate model-generated solutions for their correctness. Using this corpus, we find that models often arrive at the correct final answer through wrong reasoning steps. In addition, we test whether models are able to verify these solutions, and find that most models struggle.

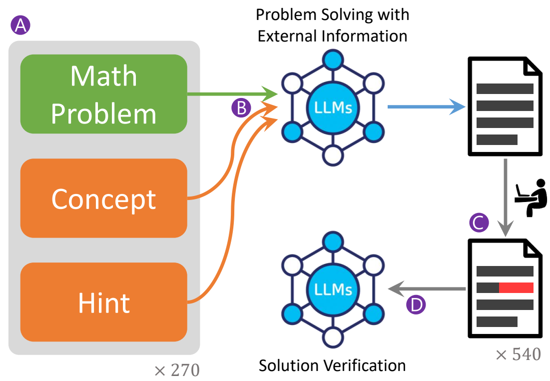

Figure 1: Overview of our dataset and experiment contribution. **A**: We collect 270 challenging, high-school math competition problems (e.g., *Find all positive integer solutions to the equation* $x^3 + 3 = 4y(y + 1)$). For each problem, we write the relevant and helpful **C**oncepts (e.g., $a^3 \pm b^3 = (a + b)(a^2 \pm ab + b^2)$), and **H**ints (e.g, *Express* x^3 *as the product of two factors involving* y). **B**: In our experiments, to investigate a model's ability to understand and use the additional C & H information, we design 17 prompts to evaluate ten models: GPT-3.5 / 4 / 4 Turbo, PaLM 2 Medium, Llama 2 7B / 70B, Llama 3 8B / 70B, Mistral 7B and Mixtral 8x22B. **C**: For each problem, we manually judge two model-generated solutions on their correctness, and further annotate the first wrong step of the reasoning (red highlights), if present. **D**: This corpus thus serves as a novel dataset for benchmarking and evaluating the solution verification ability of LLMs.

1 Introduction

Recent large language models (LLMs) have demonstrated impressive performance on many tasks that previously required specialized models or were thought to be out of reach of conventional neural networks. One such capability is mathematical reasoning: LLMs can often solve simple math problems and make reasonable attempts at challenging, competition-level problems. In addition to model scaling (Kaplan et al., 2020), there are two key factors behind the progress: sophisticated prompting methods such as chain-of-thought (Wei et al., 2022; Kojima et al., 2022) and self-consistency (Wang et al., 2022), which provide useful heuristics for generating and selecting better reasoning paths; and access to calculators or code interpreters which offloads some of the symbolic computation to external tools (Gao et al., 2023; Zhou et al., 2023). However, a direction which remains less explored is how external concepts and hints impact LLMs' reasoning abilities. This is difficult to address with existing datasets which typically only contain problem statements and their solutions, and do not pro-

Findings of the Association for Computational Linguistics ACL 2024, pages 13256–13274
August 11-16, 2024

vide annotated concepts or hints that would be helpful for the problem at hand.

To enable such analyses, we introduce the Concept and Hint-Annotated Math Problems (CHAMP) dataset, which consists of 270 diverse high school competition-level math problems (Fig. 1). In addition to problem statements and full solutions, we annotate each problem with two key pieces of information: concepts and hints. Concepts are general math theorems or formulas, while hints are problem-specific tricks or strategies. The design of CHAMP enables previously under-explored evaluations of multi-step reasoning abilities of LLMs. For example, can LLMs make use of these concepts and hints? How should these be provided to the LLM? Could a model infer useful information from studying sample problems using the same concepts? What happens if the model is provided with irrelevant/misleading concepts?

Using this dataset, we design 17 different prompts, and evaluate various proprietary and open-source models including GPT-3.5 / 4 / 4 Turbo (OpenAI, 2023), PaLM 2 Medium (Anil et al., 2023)[1], Llama 2 7B / 70B (Touvron et al., 2023), Llama 3 8B / 70B (AI, 2024), Mistral 7B (Jiang et al., 2023) and Mixtral 8x22B (Team, 2024). While we observe a diverse range of behaviors across different models and prompts, we find the accuracy of the best setting to be 67.0% only (measured by the final answer correctness, ignoring any possible errors in intermediate reasoning steps), and gains from the additional concept and hint information vary across models. The results indicate a large room for improvement with competition-level math for LLMs, and moreover highlight the utility of CHAMP for developing and benchmarking future models.

For each problem, we further analyze solutions generated by these models by manually annotating the first wrong step in the reasoning process, such as an arithmetic error or question misunderstanding, or validating that the solution is fully correct. This annotation serves two purposes. First, it concretely identifies how much the final answer accuracy (which is the predominant practice for math problem evaluations) over-estimates a model's ability to generate fully correct solutions: indeed, we find that in many instances, a model gets the answer "right" despite generating wrong reasoning steps, indicating that these models are potentially relying on shortcut heuristics. Second, we can evaluate the verification ability of any model, i.e., how well it can reason about a given solution and identify any errors, where we find that most models struggle. The above evaluations suggest key deficiencies in current LLMs and highlight the value of these annotations as an additional benchmarking resource.

In summary, we evaluate four model capabilities using our dataset: generating correct final answer, generating correct full solution, using the contextual C & H information, and verifying a given solution. Our findings uncover new strengths and limitations of current models, and give directions for future work on improving them.

[1]The API access for PaLM 2 Large was not publicly available at the time of experiments.

2 Background and Related Work

Math datasets and benchmarks. Large language models have seen significant improvement in understanding and solving math problems, with GPT-4 (OpenAI, 2023) being able to tackle most math problems that require grade-level knowledge and direct applications of formulas, even for problems with diverse formats and wordings (Cobbe et al., 2021). Nonetheless, they still struggle with competition-level problems, such as those found in the MATH dataset (Hendrycks et al., 2021). Competition-level problems—for which applications of formulas are not straightforward—are therefore the focus of our CHAMP dataset.

A key distinction of CHAMP compared to other math datasets is the information associated with each problem. In addition to the problem text and its solution—which are common components of such datasets—we annotate relevant concepts and hints and further label them on each solution step, with problems relating to each other via common math concepts (e.g., Fermat's little theorem). In this way, CHAMP enables fine-grained evaluations of mathematical problem solving abilities of LLMs that are not possible with other datasets, for example allowing for different types of additional information to be made available in the context through prompting.

We observe that many techniques seek to improve an LLM's mathematical reasoning ability, such as encouraging chain-of-thought generation (Wei et al., 2022), selecting a final result from multiple sampled outputs (Wang et al., 2022), and using external tools such as a calculator or Python interpreter (Gao et al., 2023) to eliminate arithmetic errors. These directions can be combined with our experimental setup.

Solution verification ability of LLMs. Another distinguishing factor of our dataset is the first wrong step annotations on model solutions, which enables more fine-grained model analyses and, more importantly, evaluations of how well models can *verify* a given answer.

There have been recent attempts at crafting such datasets. For example, Lightman et al. (2023) collected PRM800K, containing 800K steps of 75K solutions to 12K problems in the MATH dataset (Hendrycks et al., 2021), with each step labeled as correct, incorrect or neutral. Chen et al. (2023) curated FELM, a factuality benchmark, including annotations of solutions to 208 GSM8K (Cobbe et al., 2021) and 194 MATH problems. Compared to CHAMP, where annotations are made exclusively by the paper authors, both PRM800K and FELM are labeled via crowdsourcing. Moreover, solutions in PRM800K are selected to maximally confuse a reward model being developed in the project, while FELM uses only GPT-3.5 as the solution generator. In contrast, our 540 annotated solutions are generated by a mix of GPT-3.5, 4, 4 Turbo and PaLM 2 Medium, each with varying capabilities.

Roles of contexts. Our dataset and experiments are similar in spirit to works that explore how well LLMs understand different contexts, which have yielded surprising findings. For example, models can be insensitive to label correctness (Min et al., 2022) but sensitive to label distribution (Zhao et al., 2021) and exemplar ordering (Lu et al., 2021). McKenzie et al. (2023) find that larger LLMs resist absorbing context information inconsistent with world knowledge acquired during training (e.g., redefining $\pi = 432$). Similarly, Wu et al. (2023) find that LLMs perform worse in atypical setups for common tasks (e.g., base-9 integer addition). With CHAMP, we can explore how different information supplied in various ways affect LLMs' behavior.

3 The CHAMP Dataset

This section describes the dataset structure and construction. Due to the high level of math expertise required, the dataset curation is carried out exclusively by the paper authors.

Problems. We select problems from the book *Problem-Solving Strategies* by Engel (2008), a classic piece of material for high-school math competitions. All problems require specific tricks or creative strategies, rather than routine knowledge applications. We require problems to have final check-able answers for easy evaluation, and thus rewrite proof problems where possible (e.g., "Prove $f(x) \geq 1$ for $x \geq 0$" is transformed to "What is the smallest value of $f(x)$ for $x \geq 0$"). A total of 270 problems span five categories: number theory (80), polynomial (50), sequence (50), inequality (50) and combinatorics (40). We make some adaptations to lessen the impact of "trivial limitations" of LLMs, such as weakness in precise arithmetics. See App. A for further details.

Problem ID: P_Inequality_36

Problem: For non-negative a, b, c, d, what is the smallest value of $\sqrt{(a+c)(b+d)} - \sqrt{ab} - \sqrt{cd} - 1$?

Concepts and **H**ints:

H1. Compare $\sqrt{(a+c)(b+d)}$ with $\sqrt{ab} + \sqrt{cd}$ by squaring both terms.

C1. $(x \pm y)^2 = x^2 \pm 2xy + y^2$.

C2. For non-negative x, y, we have $(x+y) \geq 2\sqrt{xy}$ and $(x+y)/2 \geq \sqrt{xy}$, with equality if and only if $x = y$.

C3. For non-negative x, y, $\sqrt{x} \geq \sqrt{y}$ if and only if $x \geq y$.

Answer: -1

Solution Step:

1. We have $\sqrt{(a+c)(b+d)}^2 = (a+c)(b+d) = ab + ad + bc + cd$. [H1]
2. We have $(\sqrt{ab} + \sqrt{cd})^2 = ab + cd + 2\sqrt{abcd}$. [H1, C1]
3. Thus, $\sqrt{(a+c)(b+d)}^2 - (\sqrt{ab} + \sqrt{cd})^2 = ad + bc - 2\sqrt{abcd}$, which is non-negative because $ad + bc \geq 2\sqrt{abcd}$. [H1, C2]
4. Thus, $\sqrt{(a+c)(b+d)}^2 \geq sqrt(ab) + \sqrt{(cd)}^2$.
5. Since a, b, c, d are all non-negative, we have $\sqrt{(a+c)(b+d)} \geq \sqrt{ab} + \sqrt{cd}$. [C3]
6. So the smallest value of $\sqrt{(a+c)(b+d)} - \sqrt{ab} - \sqrt{cd} - 1$ is -1, achieved when $a = b = c = d$.

Table 1: A sample from the CHAMP dataset, which shows the problem (top), the concepts and hints (middle), and the full solution (bottom).

For each problem, we manually verify and write out the full detailed step-wise solution in natural language, as the solution manual often skips steps and occasionally contains typographical errors. We also provide an explicit final answer that can be checked against.

Concepts and hints. We additionally annotate relevant *concepts* and *hints*, which provide helpful information to solve a problem. For the purposes of this paper, we define concepts to mean general math knowledge, such as an equation or a theorem, for example, "$x^2 - y^2 = (x+y)(x-y)$". We de-

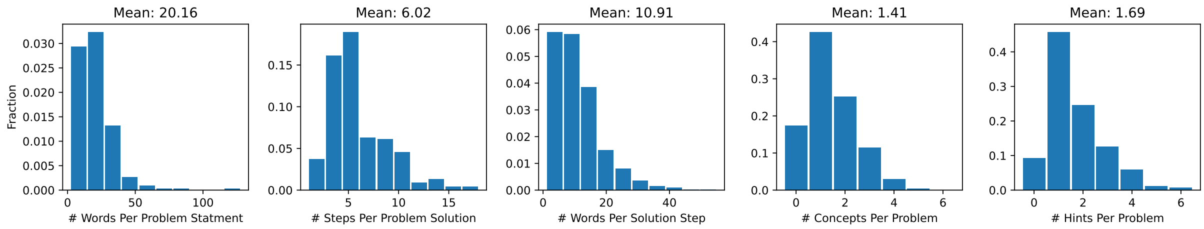

Figure 2: The distribution of dataset statistics in CHAMP. Problems in CHAMP require a nontrivial number of reasoning steps (6.0 on average). Each problem is linked to an average of 1.4 concepts and 1.7 hints.

fine hints, on the other hand, to be problem-specific tricks or strategies, such as "Add 1 to both sides." (to prepare for further simplification). These concepts and hints are also labeled to the relevant solution steps in the ground truth solution.

Each concept is additionally annotated with three metadata fields: the *category*, such as "number theory" or "polynomial"; the *name*, such as "difference of squares formula" for the concept $x^2 - y^2 = (x+y)(x-y)$; and the *parent concept*, which is a general form of the concept. For example, both $(x+y)^2 = x^2 + 2xy + y^2$ and $(x+y)^3 = x^3 + 3x^2y + 3xy^2 + y^3$ have the parent concept of the binomial theorem, which states $(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{(n-k)} y^k$. While every concept is assigned a category, not all concepts have names or parent concepts. One CHAMP problem with its solution and concept & hint annotation is shown in Tab. 1; additional examples are given in Tab. 10 of App. B.

First wrong step (FWS) annotations. For every problem, we further analyze the LLM-generated solutions to identify mistakes in reasoning steps. Specifically, we randomly assign a "generator model," one of GPT-3.5, GPT-4, GPT-4 Turbo and PaLM 2 M, to each problem, stratified by the problem's category (i.e., number theory, polynomial, etc.), and collect the model-generated solutions for two prompts: one with the problem statement only (the "No C + w/o H" prompt in Tab. 2 of Sec. 4.2) and another with the problem statement supplemented by the list of relevant concepts and hints (the "Direct + w/H" in Tab. 2 of Sec. 4.2). We assess whether the *full solution* (including the reasoning steps) is correct, and if not, annotate the occurrence of the first wrong step (FWS) as a span in the text. The FWS refers to an objective mistake, such as an arithmetic error or question misunderstanding, and does not contain subjective assessments, such as a seemingly implausible strategy to the problem.

Dataset statistics. We collect 270 problems, 54 concepts, 330 hints and 540 FWS annotations. Every problem has at least 1 concept or hint, with an average of 1.4 concepts and 1.7 hints, and an average of 6.0 solution steps. Each problem statement has an average of 20.2 words, and each solution step 10.9 words. Fig. 2 plots the distribution histograms of these statistics.

4 Experiment 1: Problem Solving

4.1 Experimental Setup

We evaluate ten models: GPT-3.5 Turbo (16k context version), GPT-4, GPT-4 Turbo, PaLM 2 Medium, Llama 2 7B, Llama 2 70B, Llama 3 8B, Llama 3 70B, Mistral 7B and Mixtral 8x22B (exact model versions in Tab. 2). We set temperature to 0 for all experiments, and allow at least 1024 tokens to be generated for each model depending on the specific context window size, which is more than enough to output the correct reasoning, with unfinished generations treated as incorrect. Inference for proprietary models is done via the respective API, and that for open-source models is done via the `together.ai` API.

Baseline. Sometimes models can output correct final answers with incorrect reasoning, especially when those answer values appear often in the training corpus. To contextualize the final answer accuracy metrics, we construct the majority guess baseline as follows. For each of the four answer types in our dataset—numeric answers (e.g., 0), expression answers (e.g., n), yes/no answers (e.g., no) and enumeration answers (e.g., $x = 1$ or $x = -2$)—the baseline guesses the most frequently appearing answer values in the ground truth. This baseline accuracy is 33.0%. See Tab. 11 of App. B for more details.

Automatic evaluation. As is the case with many generation tasks, the correct final answer can be expressed in multiple ways: "no solution" is equiv-

Model	Standard Evaluations 0-Shot	5-Shot	1/3 Soln	2/3 Soln		CHAMP-Enabled Evaluations (7 Ways to Provide Concepts With or Without Hints) 1. No C	2. Direct	3. Root	4. Name	5. Example	6. Problem	7. Misleading
GPT-3.5	28.5	34.8	33.7	40.7	w/o H	28.5	28.5	28.1	**33.0**	30.4	30.0	27.8
gpt-3.5-turbo-16k-0613					w/ H	31.9	33.0	33.0	31.9	31.9	**34.4**	31.5
GPT-4	41.9	38.1	53.7	65.6	w/o H	41.9	**43.0**	42.6	40.0	42.2	43.7	39.3
gpt-4-0613					w/ H	51.9	**53.0**	49.6	49.3	52.2	51.1	48.9
GPT-4 Turbo	58.1	53.0	61.5	67.0	w/o H	**58.1**	57.0	55.6	51.1	55.9	55.2	51.9
gpt-4-1106-preview					w/ H	62.2	**65.9**	64.8	63.0	63.3	64.4	55.6
PaLM 2 Medium	14.1	17.4	15.9	23.3	w/o H	14.1	15.6	15.6	14.1	**17.0**	16.7	19.3
chat-bison-001					w/ H	14.4	15.2	15.2	18.1	16.7	**20.7**	19.3
Llama 2 7B	8.5	9.6	7.4	10.7	w/o H	8.5	7.8	8.1	6.7	9.3	**10.0**	9.3
Llama-2-7b-chat-hf					w/ H	9.6	7.4	9.3	7.4	10.0	**11.1**	8.9
Llama 2 70B	11.9	13.0	14.8	21.5	w/o H	11.9	13.3	11.1	**14.1**	13.7	13.3	12.6
Llama-2-70b-chat-hf					w/ H	15.6	15.6	15.9	15.9	**16.7**	15.6	16.3
Llama 3 8B	20.7	22.2	25.2	35.9	w/o H	21.1	23.0	24.1	**24.8**	20.0	**24.8**	20.7
Meta-Llama-3-8B-Instruct					w/ H	23.7	25.6	25.6	**29.3**	26.3	25.9	21.5
Llama 3 70B	37.8	35.9	47.0	59.6	w/o H	37.8	**40.0**	37.0	**40.0**	38.1	37.0	35.6
Meta-Llama-3-70B-Instruct					w/ H	47.0	49.6	48.9	47.4	**50.0**	48.9	42.2
Mistral 7B	20.7	16.3	18.1	24.8	w/o H	**20.7**	20.4	20.0	**20.7**	19.3	19.6	18.1
Mistral-7B-Instruct-v0.3					w/ H	18.1	19.3	17.0	20.0	18.9	**22.2**	16.7
Mixtral 8x22B	36.7	36.7	47.0	60.7	w/o H	36.7	38.1	**39.3**	32.6	34.1	38.5	32.2
Mixtral-8x22B-Instruct-v0.1					w/ H	47.4	48.5	49.6	47.0	46.3	**50.0**	45.2
Majority Guess Baseline: 33.0												

Table 2: Final answer accuracy (in percentage) with the different prompt settings.

alent to "none", "unsolvable" is equivalent to "impossible to solve", etc. Therefore a simple criteria-based exact or sub-string match to a manually constructed set of valid answers is prone to false negatives. We thus propose to use GPT-4 as an automatic grader, and use a three-stage procedure for the solution generation and evaluation. First, we prompt the model for its step-by-step solution. Then, we ask the model to produce a one-sentence summarization of its answer. Finally, we use GPT-4 to grade the answer summary, given the ground truth final answer, which essentially checks for semantic equivalence between the two. The prompts are listed in Tab. 12 and 13 of App. C.

To assess the validity of this automatic evaluation procedure, we manually checked 500 examples for (1) whether the one-sentence summarization was correct and (2) whether GPT-4 was able to correctly grade the summarized solution given the ground truth final answer. While not perfect, we found GPT-4 to be quite good at this, with the accuracies on both tasks being $\geq 97\%$.

4.2 Model Analyses

Our experiments are aimed at evaluating four different aspects: raw model performance, effectiveness of different ways to provide concepts, relative importance of concepts and hints, and impact of irrelevant (i.e., misleading) concepts. The quantitative results are summarized in Tab. 2.

Model performance. We first study the model performance with both zero-shot (Kojima et al., 2022) and few-shot (Wei et al., 2022) chain-of-thought prompting. Following the experiments by (Hendrycks et al., 2021), we also study models' performance when given partial solution (1/3 and 2/3 of solution steps) under the zero-shot setup. The prompts are listed in Tab. 14-16 of App. C.

The blue cells of Tab. 2 summarize these results, with some full model-generated solutions presented in Tab. 22 and 23 of App. D. Generally, larger and more recent models perform better than their smaller and earlier versions, and partial solutions are mostly helpful in guiding the model to correct solutions, largely consistent with the findings by Hendrycks et al. (2021). Overall, the best performing models, GPT-4 and GPT-4 Turbo, are still proprietary ones and there is a gap to close for (even the latest) open-source models. In addition, five-shot prompting is often not beneficial, suggesting that such instruction-tuned models, especially high-performing ones, may not need in-context exemplars to activate the "problem-solving mode," as long as the instruction is sufficiently clear.

Concept provision method. As concepts are general facts or formulas in math, they are likely already learned by the model during pre-training. However, different ways to provide the knowledge in the context may affect how well the model can understand and use it. We designed six concept provision approaches, each in two versions where the hint is withheld or provided, corresponding to the light green cells and dark green cells of Tab. 2:

1. Prompt with no concepts (No C).
2. Directly provide the concept in the prompt (Direct).
3. Provide the root concept up the parent-child chain (i.e., most general form) in the prompt (Root).
4. Ask the model to retrieve the concept by its name in a separate round of conversation (Name).
5. Ask the model to provide an example for the concept in a separate round of conversation (Example).
6. Provide a sample problem that uses the concept and its step-by-step solution (Problem).

The specific prompts are listed in Tab. 17-21 of App. C. The best performing concept provision method for each model with and without hints is **bolded**. No single method performs the best across the board. Furthermore, concepts may sometimes even be detrimental, potentially because they contradict with the model's "initial planned approach."

Importance of hints. Compared to concepts, hints are arguably more valuable as they often require creativity from the test-takers. The performance contrast without and with the hint under each prompt are shown by the light green rows vs. dark green rows of Tab. 2. While providing hints helps, the performance increase is correlated with the model's "base capability": GPT-4, 4 Turbo, Llama 3 70B and Mixtral 8x22B, which have the best zero-shot accuracy, are able to score 10% higher on average with hints, while the accuracy increase is much less notable for other models.

Impact of misleading concepts. How well could a model deal with misleading concepts? In this experiment, for each useful concept, we replace it with a random one of the same category (to ensure maximal confusion) but not on the path to its root concept (to avoid accidentally providing useful information).[2] The results are summarized in the orange cells of Tab. 2. Compared to the "No C" setup, misleading concepts have different impacts on different models: while most of the models show slight drop of accuracy with misleading concepts, GPT-4 Turbo suffers the most, with over 10% relative decrease of accuracy compared with the "No C" setup. On the other hand, PaLM 2 Medium and both Llama 2 models even show some improvement, indicating that they are unlikely to understand and act on provided (mis)information.

[2]We do not experiment with misleading hints, as they would appear nonsensical due to problem-specificity.

	Problem Only		Problem + C & H	
	Final Ans	Full Soln	Final Ans	Full Soln
GPT-3.5	34.3%	6.0%	32.8%	16.4%
GPT-4	54.4%	17.6%	52.9%	33.8%
GPT-4 T	56.7%	22.4%	68.7%	44.8%
PaLM 2 M	19.1%	1.5%	14.7%	0.0%

Table 3: Final answer vs. full solution accuracy for four models under two prompts.

4.3 Full Solution Accuracy

The above analyses are based on an important assumption: that the final answer accuracy is a faithful reflection of the model's mathematical ability. However, focusing on final answer alone could inflate the models' performance as incorrect reasoning could lead to correct final answers, especially for questions with yes/no answers (see examples in Tab. 22 and 23 of App. D). As a result, we proceed to examine the full solutions generated by four models: GPT-3.5, GPT-4, GPT-4 Turbo and PaLM 2 Medium, based on the first wrong step (FWS) annotation.

Tab. 3 displays the final answer accuracy (FAA) and full solution accuracy (FSA) of model outputs from the 0-shot problem-only prompt and the problem + concept & hint list prompt.[3] Full solution accuracy (FSA) is significantly lower than final answer accuracy (FAA), suggesting that the latter is indeed an inflated measurement of the models' true reasoning ability. As an extreme case, PaLM 2 M produces only one fully correct solution out of 136 attempts on 68 problems, despite still achieving 14.7% FAA. Given that almost all benchmarks (e.g. Cobbe et al., 2021; Hendrycks et al., 2021) are based on FAA, true model performance may not be as high as previous benchmarking results suggest. Nonetheless, performance of the GPT models increases under both prompts, regardless of the evaluation metrics of FSA or FAA, suggesting that FAA could likely be a proxy for FSA.

For all GPT models, providing the C & H list significantly helps with FSA, even when FAA stays at a similar level (among the respective problem sub-

[3]Note that the FAA statistics in Tab. 3 are based on model-generated solutions of 25% sampled problems for each model, and hence different from those in Tab. 2.

You need to grade a student's answer to a math problem and determine if it contains any objective error, including but not limited to mistakes in logical deductions, algebraic manipulations, arithmetic calculations or question understanding. You should not make any subjective judgment, such as marking a strategy that seems unsuccessful as incorrect, unless you have objective evidence of an error. *For your convenience, a reference solution is also given. However, the student answer could differ from it significantly but still be correct, by, for example, using a different strategy.* Ⓐ

Feel free to think step by step through the student answer. On the last line, write "Judgment: ", followed by your judgment, in one of two cases:

1. If you think that the answer is wrong, copy the sentence containing the first error (i.e.: "Judgment: <verbatim sentence containing the first error>").
2. If you think that the solution is fully correct, write "No mistake" as your judgment (i.e.: "Judgment: No mistake").

Question Statement:

(“No C + w/o H” or “Direct + w/ H” Prompt) Ⓑ

Reference Solution: Ⓐ

(Ground Truth Step-by-Step Solution)

Student Answer:

(Solution Under Judgment) Ⓒ

Figure 3: The prompt for model verification evaluation. Text in black is given as the default, and we experiment with several variations. Ⓐ: we choose to give or withhold the reference solution in the prompt, where the *blue italic texts* are not provided in the latter case. Ⓑ: we evaluate model solutions for two prompts – problem only and problem with concept and hint list. Ⓒ: the corresponding solution is given as the “Student Answer”.

set). By comparison, PaLM 2 M could not benefit from additional information. These results imply the necessity for the finer-grained evaluations as we test LLMs on more challenging benchmarks.

5 Experiment 2: Solution Verification

The first wrong step (FWS) annotation allows for evaluating whether LLMs can read a solution to a problem and *verify* its correctness, which is a much more difficult task than comparing the final answer against the ground-truth answer. In this set of experiments, we study how well LLMs can judge model-generated solutions, as well as the (error-free) ground truth solution.

5.1 Experimental Setup

As noted in Sec. 3, the FWS dataset is obtained by (1) randomly assigning a model (out of GPT-3.5, GPT-4, GPT-4 Turbo and PaLM 2 M) to each problem and collecting its solutions for two prompts—Problem Only (i.e., “No C + w/o H” in Tab. 2) and Problem + C & H (i.e., “Direct + w/H” in Tab. 2)—and then (2) manually annotating the first wrong step in each solution (or label it to be fully correct if there are no wrong steps). We first evaluate the ten models on each annotated solution, using the prompt shown in Fig. 3. Two variants are explored: one where the reference solution is not given (by removing all *blue italic texts*) and the other where the reference solution is given (by using the complete prompt). This prompt both allows the model to engage in chain-of-thought reasoning and moreover enables easy parsing of the output, via the line starting with “Judgment:”.

5.2 Results

Output judgment type. We first study the judgment types from models (regardless of judgment correctness), with three categories: *has-FWS*, where the model identifies a FWS, *no-mistake*, where the model outputs “Judgment: No mistake”, and *invalid*, where the sentence following “Judgment:” is not in the solution or this line is not found.

Tab. 4 lists the count breakdown for each of the three judgment types, in that respective order. Compared to the ground truth distribution, only GPT-4 and 4 Turbo behave reasonably, with other models either producing a large number of invalid judgments (e.g., Llama 2 7B and Mistral 7B), or failing to identify mistakes in most solutions (e.g., PaLM 2 Medium and Llama 2 70B). The results indicate that despite the ability to get correct final answers on challenging math problems, most models lack strong verification abilities or even present difficulty understanding and following the verification task prompt. For more detailed analyses, we focus on GPT-4 and 4 Turbo, which produce few invalid answers and best resemble the ground truth judgement patterns.

Judgment correctness analysis. For syntactically valid judgements (*has-FWS* and *no-mistake*), we evaluate their outputs in more depth. For each of the verifier model of GPT-4 and 4 Turbo, we consider 4 setups by varying two factors: the Problem Only or Problem + C & H prompt, and with or without reference solution. Tab. 5 shows the counts of different model vs. ground truth judgments.

We compute sensitivity and specificity to quantify the two verifiers’ FWS identification accuracy. Recall that true-positive (TP) is the case where the

	Problem Only		Problem + C & H	
	w/o Ref Soln	w/ Ref Soln	w/o Ref Soln	w/ Ref Soln
GPT-3.5	6 | 228 | 36	5 | 170 | 95	2 | 243 | 25	1 | 186 | 83
GPT-4	180 | 86 | 4	178 | 87 | 5	171 | 94 | 5	177 | 87 | 6
GPT-4 T	203 | 56 | 11	200 | 55 | 15	205 | 49 | 16	193 | 67 | 10
PaLM 2 M	0 | 233 | 37	0 | 225 | 45	0 | 221 | 49	0 | 236 | 34
Llama 2 7B	124 | 20 | 126	106 | 54 | 110	149 | 11 | 110	130 | 27 | 113
Llama 2 70B	0 | 264 | 6	0 | 255 | 15	0 | 264 | 6	0 | 266 | 4
Llama 3 8B	3 | 252 | 15	13 | 249 | 8	3 | 252 | 15	18 | 241 | 11
Llama 3 70B	93 | 128 | 49	46 | 199 | 25	69 | 138 | 63	47 | 197 | 26
Mistral 7B	0 | 73 | 197	0 | 13 | 257	0 | 65 | 205	0 | 17 | 253
Mixtral 8x22B	57 | 123 | 90	40 | 182 | 48	70 | 136 | 64	29 | 200 | 41
Ground truth	238 | 32 | 0		206 | 64 | 0	

Table 4: Count breakdown of judgment types produced by each model. In each cell, the three numbers represent the number of *has-FWS*, *no-mistake* and *invalid* judgments. Each triplet sums up to 270, the total number of problems (and annotated solutions for the prompt). The ground truth statistics are shown on the last line.

Setup			Model vs. Ground Truth Judgment					
Verifier	C & H	Ref Soln	early	TP	late	spur.	miss	TN
GPT-4	✗	✗	33	61	76	10	64	22
	✗	✓	31	66	78	3	58	29
	✓	✗	33	50	69	19	49	45
	✓	✓	33	73	66	7	30	57
GPT-4 T	✗	✗	33	71	86	13	37	19
	✗	✓	37	77	80	6	29	26
	✓	✗	31	54	87	33	19	30
	✓	✓	29	69	81	14	18	49

Table 5: Detailed analysis of FWS identification. "Verifier" is the model under evaluation. A cross mark under "C & H" uses the Problem Only ("No C + w/o H") prompt and its solution, and a check mark uses the Problem + C & H ("Direct + w/ H") prompt. A cross mark under "Ref Soln" withholds the reference solution, and a check mark reveals it. For the six judgment types, red highlighting marks the ground truth (GT) FWS span, a green check mark means that the full solution is error-free, and yellow crosses mark the verifier's identification (VI), if present. The six judgments are: *early* (where VI is before GT), *TP* (true positive, where VI overlaps with GT), *late* (where VI is after GP), *spurious* (where GT is error-free but the verifier makes an identification), *miss* (where verifier misses a GT FWS) and *TN* (true negative, where model makes a correct judgment of error-free). Green and red cell background colors indicates correct and incorrect judgments, respectively. The number in each cell counts the specific model judgments under the prompt. Invalid responses are excluded.

model correctly identifies the ground truth FWS (with any overlapping) in a wrong solution, and true-negative (TN) is the case where the model correctly reports a no-mistake judgment for an error-free solution. Sensitivity is defined as the fraction of TP among all wrong solutions, and specificity as the fraction of TN among all error-free solutions. The former measures the verifiers' ability of accurately locating a FWS, while the latter measures that of recognizing correct solutions.

Prompt	Verifier	Sensitivity		Specificity	
		w/o Ref	w/ Ref	w/o Ref	w/ Ref
Prob Only	GPT-4	16.3	*19.9*	71.0	*90.3*
	GPT-4 T	**21.6**	***28.1***	**58.1**	***80.6***
Prob + C & H	GPT-4	18.1	*32.6*	70.3	*89.1*
	GPT-4 T	**28.1**	***33.8***	**47.6**	***77.8***

Table 6: Sensitivity and specificity for different verifiers and experimental setups.

Tab. 6 shows these statistics. This table, and the raw counts in Tab. 5, reveal two trends. First, giving the verifier access to reference solutions (i.e., italicizing) helps its performance, as evidenced by the increase in both sensitivity and specificity. Second, GPT-4 has lower sensitivity but higher specificity than GPT-4 Turbo (in bold), meaning that it is less capable of identifying FWSs, but also less prone to hallucinating errors in error-free solutions.

Verification of reference solutions. Finally, we evaluate whether models can verify the reference ground-truth solution (pretending it to be the student answer). Here the correct response is always "no-mistake". The results are summarized in Tab. 7. The best verifier models in the last experiment, GPT-4 and GPT-4 Turbo, make the most number of wrong "has-FWS" identifications. GPT-4 Turbo performs better than GPT-4 but still hallucinates a mistake on more than half solutions. Many models other than GPT-4 and GPT-4 Turbo ostensibly perform well on this task. However, this is due to their inability to identify *any* FWS (c.f., Tab. 4),

Judgment	Has-FWS	No-Mistake	Invalid
GPT-3.5	4	186	80
GPT-4	190	79	1
GPT-4 T	161	102	7
PaLM 2 M	0	187	83
Llama 2 7B	76	123	71
Llama 2 70B	2	235	15
Llama 3 8B	30	222	18
Llama 3 70B	38	223	9
Mistral 7B	0	4	266
Mixtral 8x22B	97	139	74

Table 7: Model verification of (error-free) reference solutions. Only "No-Mistake" judgments are correct.

as shown in the previous experiment. These results collectively indicate that solution verification is beyond the current capabilities of many LLMs.

6 Discussion

This paper presents CHAMP, the Concept and Hint-Annotated Math Problems dataset, along with annotations of logical correctness of model-generated solutions to each problem. The unique construction of CHAMP enables previously under-explored studies regarding context-related reasoning abilities as well as verification abilities of LLMs.

We investigate the mathematical reasoning abilities of 10 proprietary and open-source LLMs of various sizes and release dates through CHAMP. Even the best models are far from perfect, and many models are unable to incorporate useful concepts and problem-specific hints. Even when the final answer accuracy is high, a closer inspection of the reasoning steps reveals that many models may be accidentally arriving at the correct answer.

Furthermore, all models struggle at solution verification, indicating that these models often generate but do not understand their solutions, similar to some of the findings of recent work (West et al., 2023; Qiu et al., 2023). These results advocate for a more fine-grained and multi-faceted investigation of LLMs' mathematical reasoning capabilities.

7 Limitations

While CHAMP reveals intriguing behaviors of different models, there are several limitations. For one, we relied on extensive manual annotation for high-quality labeling, and hence the dataset currently contains only 270 problems. There is also the risk of dataset contamination; i.e., the models we evaluate may have been trained on the original versions of the problems from CHAMP. However, to mitigate this we rewrote the problems to fit the design of the dataset, minimizing the impact of memorization. Finally, while we rely on automatic evaluation for final answer accuracy to enable scalable evaluation, this may not be perfect. Our manual grading results suggest, though, that GPT-4's automatic grading has a high accuracy of 97%.

Acknowledgments

We thank the reviewers for their suggestions. This study was supported in part by funds from MIT-IBM Watson AI.

References

Meta AI. 2024. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. arXiv preprint arXiv:2305.10403.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I Chern, Siyang Gao, Pengfei Liu, Junxian He, et al. 2023. Felm: Benchmarking factuality evaluation of large language models. arXiv preprint arXiv:2310.00741.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

Arthur Engel. 2008. Problem-solving strategies. Springer Science & Business Media.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In International Conference on Machine Learning, pages 10764–10799. PMLR.

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. arXiv preprint arXiv:2103.03874.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. arXiv preprint arXiv:2310.06825.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. Advances in neural information processing systems, 35:22199–22213.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. arXiv preprint arXiv:2305.20050.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. arXiv preprint arXiv:2104.08786.

Ian R McKenzie, Alexander Lyzhov, Michael Pieler, Alicia Parrish, Aaron Mueller, Ameya Prabhu, Euan McLean, Aaron Kirtland, Alexis Ross, Alisa Liu, et al. 2023. Inverse scaling: When bigger isn't better. arXiv preprint arXiv:2306.09479.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? arXiv preprint arXiv:2202.12837.

OpenAI. 2023. Gpt-4 technical report. ArXiv.

Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, et al. 2023. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. arXiv preprint arXiv:2310.08559.

Mistral AI Team. 2024. Cheaper, better, faster, stronger. https://mistral.ai/news/mixtral-8x22b/.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. arXiv preprint arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35:24824–24837.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, et al. 2023. The generative ai paradox:" what it can create, it may not understand". arXiv preprint arXiv:2311.00059.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2023. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. arXiv preprint arXiv:2307.02477.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In International Conference on Machine Learning, pages 12697–12706. PMLR.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, et al. 2023. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. arXiv preprint arXiv:2308.07921.

A Problem Collection and Annotation Considerations

A.1 Number Theory

A notable feature of number theory problems is that most of them are proof problems. We manage to convert most of them into problems asking for an answer, with examples listed in Table 8. In addition, there are some questions which require non-trivial factorization. Since LLMs are often bad at arithmetics above 100, we provide them directly as hints, such as $1971 = 27 \times 73$.

Before	After
Prove that $n^4 + 4^n$ can never be a prime number for integer $n > 1$.	For how many integers n in $\{1, 2, ..., 99\}$ is $n^4 + 4^n$ a prime number? (Answer: 1)
Prove that $x^2 + y^2 + z^2 = 2xyz$ has no positive integer solutions.	Find all positive integer solutions to the equation $x^2 + y^2 + z^2 = 2xyz$. (Answer: No positive integer solutions)
Prove that $323 \mid 20^n + 16^n - 3^n - 1$ for even n.	What are possible values of $20^n + 16^n - 3^n - 1 \bmod 323$ for even n? (Answer: 0 is the only possible value)

Table 8: Conversion of proof problems into those with check-able answers.

A.2 Polynomial

Some polynomial problems require factorization or root finding involving nontrivial arithmetics, similar to number theory problems. To reduce errors in this process, we provide the relevant arithmetic calculation as hints, such as $264 = 6 \times 44$ when factoring $v^2 - 50v + 264 = (v - 6)(v - 44)$.

In addition, there are several polynomial division and remainder problems, for which we provide the concrete definition as a concept (although all models could easily retrieve and explain this definition with a straightforward query of "What is polynomial division and remainder?"):

> When a polynomial $f(x)$ is divided by a polynomial $g(x)$, the quotient $q(x)$ and the remainder $r(x)$ are polynomials such that $f(x) = g(x)q(x) + r(x)$ and the remainder $r(x)$ has degree less than that of $g(x)$.

A.3 Sequence

A common type of problems in sequence is to find its limit. However, a prerequisite is to prove that the limit exists. Thus, we frame such questions explicitly, using wording such as "Determine if the limit exists, and if so, find its value." We also annotate these questions with concepts stating the theorem that establish the existence of the limit, most commonly the monotone convergence theorem:

> A sequence that is monotonic and bounded has a limit. Specifically, a sequence that is monotonically increasing and bounded from above, or monotonically decreasing and bounded from below, has a limit.

In addition, a common strategy is induction, which shows that a property holds for all a_n by showing that it holds for a_n if it holds for all of $a_1, ..., a_{n-1}$. Because the instantiation of the strategy, especially the property to show, is problem-specific, we provide it as a hint, rather than a concept.

A.4 Inequality

Just like with the category of number theory problems, many problems in inequality are written as proofs of inequality identity. We manage to convert them into questions requiring numerical answers with approaches such as asking for the extremum (i.e., maximum or minimum depending on the original inequality) value (while making sure that the value can indeed be attained by some variable value assignment). Some sample conversions are listed in Tab. 9.

Before	After
Prove that, for $a, b, c > 0$, $\sqrt[3]{abc} \leq \sqrt{(ab+bc+ca)/3}$?	For positive a, b, c, what is the smallest value of $\sqrt{ab+bc+ac}/\sqrt[3]{abc}$? (Answer: $\sqrt{3}$)
If $n > 1$, proof that $1/(n+1) + 1/(n+2) ... + 1/(2n) > 1/2$.	For how many values of n in $\{101, ..., 1000\}$ is $1/(n+1)+1/(n+2) + ... + 1/(2n) > 1/2$? (Answer: 900)
The product of three positive reals is 1. Their sum is greater than the sum of their reciprocals. Prove that exactly one of these numbers is > 1.	The product of three positive real numbers is 1, and their sum is greater than the sum of their reciprocals. How many of them can be greater than 1? (Answer: 1)

Table 9: Conversion of inequality proof problems into those requiring answers.

A.5 Combinatorics

Most combinatorics problems describe real-world scenarios. Where applicable, we provide any unmentioned commonsense knowledge (e.g., "*On a chess board, two rooks are placed peacefully if they are not on the same row or column.*") before the problem (e.g., "*For an $n \times n$ chess board, find the number of ways that n rooks can be placed peacefully (i.e., any two are placed peacefully), as an expression of n.*").

In addition, many combinatorics problems ask for the number of ways in a setup size n (e.g., the number of ways that n horses can finish in a race with the possibility of ties), and it is solved in the following manner:

1. Find a recurrence relationship to express $P(n)$ in terms of $P(n-1)$ and $P(n-2)$ (and possibly more terms), where $P(n)$ is the quantity asked in the question.
2. Find the initial values $P(1), P(2)$ (and possibly more terms).
3. Set up a characteristic equation (which is a polynomial) and find its root.
4. Use the roots to express $P(n)$ as a function of n.

The key difficulty is the root-finding part, so instead of asking for the general expression of $P(n)$ in terms of n, we ask for a specific value, such as $P(7)$, which could be worked out by repeatedly applying the recurrence relationship from the initial values. We also make sure that the asked $P(n)$ value is relatively small, usually less than 200, to minimize the chance of arithmetic errors.

B Dataset Details

Tab. 10 shows one problem from each category, with problem, concepts and hints on the left column, and solution on the right column.

Number Theory (Problem ID: P_Number-Theory_1)	
Problem: Are there integer solutions to the equation $(x^2-1)(y^2-1)+1985 = z^2$? **C**oncepts and **H**ints: H1. Consider the equation modulo 9. C1. For integer x, x^2 mod 9 can take values of 0, 1, 4 and 7. C2. $(a+b) \bmod m = ((a \bmod m)+(b \bmod m) \bmod m).(a-b) \bmod m = ((a \bmod m) - (b \bmod m) \bmod m).ab \bmod m = ((a \bmod m)(b \bmod m) \bmod m).a^k \bmod m = ((a \bmod m)^k \bmod m)$. H2. $1985 \bmod 9 = 5$.	Answer: No Solution Steps [Concepts and Hints Used]: 1. If the equation has a solution, we have $(x^2-1)(y^2-1) = z^2 - 1985$. 2. Since $u^2 \bmod 9 \in \{0, 1, 4, 7\}$, we have $u^2 - 1 \bmod 9 \in \{0, 3, 6, 8\}$ and $(x^2-1)(y^2-1) \bmod 9 \in \{0, 1, 3, 6\}$. [H1, C1, C2] 3. However, since $1985 \bmod 9 = 5$, $z^2 - 1985 \bmod 9 \in \{2, 4, 5, 8\}$. [H1, H2, C2] 4. Since there is no overlapping values, we conclude that the equation has no solution.

Polynomial (Problem ID: P_Polynomial_1)	
Problem: What is the remainder of $nx^{n+1} - (n+1)x^n + 1$ divided by $(x-1)^2$? **C**oncepts and **H**ints: C1. When a polynomial $f(x)$ is divided by a polynomial $g(x)$, the quotient $q(x)$ and the remainder $r(x)$ are polynomials such that $f(x) = g(x)q(x)+r(x)$ and the remainder $r(x)$ has degree less than that of $g(x)$. H1. Let $f(x) = nx^{n+1} - (n+1)x^n + 1$ and study $f(1)$ and $f'(1)$.	Answer: 0 Solution Steps [Concepts and Hints Used]: 1. We have $f(x) = (x-1)^2 * q(x) + r(x)$ where $r(x)$ is a polynomial of degree at most 1 (i.e., $r(x) = ax + b$). [C1] 2. Thus, $f(1) = 0 = r(1)$. [H1] 3. We have $f'(x) = 2(x-1) * q(x) + (x-1)^2 * q(x) + r'(x), sof'(1) = r'(1) = 0$. [H1] 4. Since $r(x)$ has the form of $ax + b$, we have $a + b = 0$, $a = 0$, so $b = 0$. 5. Thus, $r(x) = 0$ is the remainder.

Sequence (Problem ID: P_Sequence_2)	
Problem: Let $\{x_n\}, \{y_n\}, \{z_n\}$ be three sequences with positive initial terms x_1, y_1, z_1, defined as $x_{n+1} = y_n + 1/z_n, y_{n+1} = z_n + 1/x_n, z_{n+1} = x_n + 1/y_n$. Let w_n be the maximum value of x_n, y_n, z_n. For different values of x_1, y_1, z_1, do we have w_{200} always greater than 20, always smaller than 20, or sometimes greater and sometimes smaller than 20? **C**oncepts and **H**ints: H1. Let $a_n = x_n + y_n + z_n$. H2. Derive a lower bound on a_2. C1. For positive x, $x + 1/x \geq 2$, with equality if and only if $x = 1$. H3. Compare a_n with $18n$ for all n. C2. $(x \pm y)^2 = x^2 \pm 2xy + y^2$. C3. For real numbers $a_1, ..., a_n$ and $b_1, ..., b_n$, $(a_1b_1 + ... + a_nb_n)^2 \leq (a_1^2 + ... + a_n^2)(b_1^2 + ... + b_n^2)$.	Answer: Always greater than 20 Solution Steps [Concepts and Hints Used]: 1. Let $a_n = x_n + y_n + z_n$. [H1] 2. We have $a_2^2 = (x_1 + 1/x_1 + y_1 + 1/y_1 + z_1 + 1/z_1)^2 \geq (2+2+2)^2 = 36 = 2 \cdot 18$. [H2, C1] 3. If $a_n^2 \geq 18n$, then we have $a_{n+1}^2 = (x_n + 1/x_n + y_n + 1/y_n + z_n + 1/z_n)^2 \geq a_n^2 + 2(x_n + y_n + z_n)(1/x_n + 1/y_n + 1/z_n) \geq a_n^2 + 2 * 9 \geq 18n + 18 = 18(n+1)$. [H3, C2, C3] 4. So we have $a_n^2 \geq 18n$. [H3] 5. Thus, $a_{200}^2 \geq 18 \cdot 200 = 3600$, which means that $a_{200} = x_{200} + y_{200} + z_{200} \geq 60$. 6. So one of $x_{200}, y_{200}, z_{200}$ must be at least 20. 7. Thus, w_{200} cannot be smaller than 20.

Inequality (Problem ID: P_Inequality_2)	
Problem: For positive a, b, what is the smallest value of $(a^2 + b^2)/(a+b)^2$? **C**oncepts and **H**ints: C1. For non-negative x, y, we have $(x+y)/2 \leq \sqrt{(x^2+y^2)/2}$, with equality if and only if $x = y$.	Answer: $1/2$ Solution Steps [Concepts and Hints Used]: 1. Since $(a+b)/2 \leq \sqrt{(a^2+b^2)/2}$, we have $(a+b)^2/4 \leq (a^2+b^2)/2$. [C1] 2. This means that $(a^2+b^2)/(a+b)^2 \geq 1/2$. 3. So the smallest value is $1/2$, achieved at $a = b$.

Combinatorics (Problem ID: P_Combinatorics_1)	
Problem: Let a string consist of digit 1, 2, 3. How many such strings of length 6 have adjacent digit differing by less than or equal to 1? **C**oncepts and **H**ints: H1. Let x_n, y_n, z_n be the number of length-n strings that end with digit 1, 2, 3 respectively. H2. What are x_1, y_1, z_1? H3. By appending a digit to the existing string, derive the formula for x_{n+1}, y_{n+1}, z_{n+1} from x_n, y_n, z_n. C1. If there are n actions, with p_i ways to perform the i-th action, and no two actions can be performed at the same time, then there are $p_1 + p_2 + ... + p_n$ ways to perform the action in total.	Answer: 239 Solution Steps [Concepts and Hints Used]: 1. Let x_n, y_n, z_n be the number of length-n strings that end with digit 1, 2, 3 respectively. [H1] 2. Thus, we have $x_1 = y_1 = z_1 = 1$. [H2] 3. For a string ending with 1, we can append 1 and 2; for a string ending with 2, we can append 1, 2 and 3; for a string ending with 3, we can append 2 and 3. [H3] 4. Thus, we have $x_{n+1} = x_n + y_n, y_{n+1} = x_n + y_n + z_n, andz_{n+1} = y_n + z_n$. [H3, C1] 5. Starting from (1, 1, 1), we have the sequence of (x_n, y_n, z_n) to be (1, 1, 1), (2, 3, 2), (5, 7, 5), (12, 17, 12), (29, 41, 29), (70, 99, 70). 6. Thus, in total, there are $x_6 + y_6 + z_6 = 70 + 99 + 70 = 239$ such strings.

Table 10: One example problem per category from the dataset. The left column presents the problem, along with relevant concepts and hints. The right column gives the solution, both the final answer and the full step-wise solution with concept and hint labels.

Table 11 presents a breakdown of problems by answer formats and the corresponding baseline answer design. This baseline simulates the best performance of a dummy model that has no math reasoning ability but can answer the question in a semantically sensible manner (e.g., answering Yes or No to a question asking for a Boolean answer).

Answer Format	# Probs	Example	Baseline Answer
Boolean	42	Is $4^{545} + 545^4$ a prime number?	No
Numeric	162	In how many ways can 4 horses go through the finish (with possibility of ties)?	0
Expression	45	Among all sequences of positive integer numbers have sum n, for integer $k < n - 1$, how many times does the number k appear, as an expression of n and k?	Sum of all variables (i.e., $n + k$ for the example)
Enumeration	21	Find all integer solutions to the equation $15x^2 - 7y^2 = 9$.	None

Table 11: The construction of baseline answers based on four answer formats.

C Full Prompt Texts

This section lists all the prompts used in the experiments. Texts of normal fonts are provided literally. Parenthesized italic *(texts)* are provided to the model, and parenthesized bold **(texts)** are model outputs.

Tab. 12 shows the main prompt setup for the model's problem solving capability evaluation. The final answer summary is evaluated by GPT-4 with the prompt listed in Tab. 13. We include "Partially correct" as one grading verdict for the model to use in ambiguous situations (e.g., the solver model finds one of two solutions to the equation) but treat it as incorrect for accuracy calculation (e.g., in Tab. 2).

Role	Message
System	You are an expert on mathematics.
User Solver	***(One or more rounds of user-solver conversation that end in the solver generating the full solution as the message of the last round. The specific conversation contents are presented in Tab. 14-21.)***
User	Now, summarize the answer above in one sentence, without any intermediate steps or explanations.
Solver	**(Model-generated final answer summary)**

Table 12: Prompt for eliciting full solution and final answer summary from the model under evaluation (i.e., solver).

Role	Message
System	You are a math teacher and need to grade student's homework. For each question, you need to determine the correctness of the student answer, given the reference answer. You only need to judge the correctness of the final answer, and should not consider any reasoning or explanations given in the answer. Note that if the student gives an obviously equivalent solution to the reference answer (e.g., 1.5 vs $3/2$ or $a^2 - b^2$ vs $(a + b)(a - b)$), the answer should be judged as correct. Your decision should be one of "Correct", "Incorrect" or "Partially correct". There is no need to explain your decision.
User	The question is: *(Problem statement)* The reference answer is: *(Ground truth final answer from the dataset)* The student answer is: *(Model-generated final answer summary)* Is the student answer correct, incorrect, or partially correct?
GPT-4	**(Grading verdict)**

Table 13: Prompt for grading the solver's final answer summary using GPT-4.

Tab. 14-16 present the prompts for evaluating models under current practices, including zero-shot, few-shot (5-shot) and zero-shot with partial solution. These conversations are to be swapped into the orange cell of Tab. 12. Note that in the few-shot prompt of tab. 15, only the last round is actually generated by the model. The "solver output" in the earlier rounds are directly fed to the model as the context (pretending to be earlier model generations).

Role	Message
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)*
Solver	**(Model-generated full solution)**

Table 14: Zero-shot prompt.

Role	Message
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Statement of sample problem 1)*
Solver	*(Ground truth solution steps for problem 1)*
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Statement of sample problem 2)*
Solver	*(Ground truth solution steps for problem 2)*
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Statement of sample problem 3)*
Solver	*(Ground truth solution steps for problem 3)*
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Statement of sample problem 4)*
Solver	*(Ground truth solution steps for problem 4)*
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Statement of sample problem 5)*
Solver	*(Ground truth solution steps for problem 5)*
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)*
Solver	**(Model-generated full solution)**

Table 15: Few-shot prompt.

Role	Message
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* Below is a partial solution to the problem that may be helpful: *(List of steps revealed in the partial solution)*
Solver	**(Model-generated full solution)**

Table 16: Zero-shot prompt with partial solution provided.

Tab. 17-21 presents the prompts for different concept provision methods covered in Tab. 2. The light green texts and dark green texts are used for the w/o H and w/ H prompts respectively, consistent with the color-coding of Tab. 2. Any corner cases are discussed in table captions.

Role	Message
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* You may find the following information useful: *(List of all hints.)*
Solver	**(Model-generated full solution)**

Table 17: The "No C" concept provision prompt (i.e. not providing any concept). The w/o H version (and w H version when the problem does not have any hint) is the same as the zero-shot prompt of Tab. 14.

Role	Message
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* You may find the following information useful: *(List of relevant concepts.)* / *(List of relevant concepts and hints.)*
Solver	**(Model-generated full solution)**

Table 18: The "Direct" concept provision prompt (also used for "Root" and "Misleading" with respective concepts). If there are no concepts in the w/o H version, then the last paragraph is removed entirely and the prompt reduces to the zero-shot prompt of Tab. 14.

Role	Message
User	Please explain the following concept: *(name of concept 1, skip this round if unnamed)*.
Solver	**(Model-generated concept explanation)**
User	Please explain the following concept: *(name of concept 2, skip this round if unnamed)*.
Solver	**(Model-generated concept explanation)**
	(One round of conversation for each named concept)
User	Please explain the following concept: *(name of concept n, skip this round if unnamed)*.
Solver	**(Model-generated concept explanation)**
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* Besides the concepts above, you may also find the following information useful: *(List of remaining unnamed concepts.)* / *(List of remaining unnamed concepts and all hints.)*
Solver	**(Model-generated full solution)**

Table 19: The "Name" concept provision prompt. If there are no unnamed concepts (and hints), i.e., an empty list, then the sentence "Besides the concepts above..." is replaced with "You may find the above concepts helpful.", and the prompt is terminated.

Role	Message
User	Please give an example that applies the following concept: *(Text of concept 1)*.
Solver	**(Model-generated example)**
User	Please give an example that applies the following concept: *(Text of concept 2)*.
Solver	**(Model-generated example)**
	(One round of conversation for each concept)
User	Please give an example that applies the following concept: *(Text of concept n)*.
Solver	**(Model-generated concept explanation)**
User	Solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* You may find the above concepts helpful. Besides the concepts above, you may also find the following information useful: *(List of all hints.)*
Solver	**(Model-generated full solution)**

Table 20: The "Example" concept provision prompt. If there are no hints, then we use the w/o H version.

Role	Message
User	First, study the following problems and their solutions as they may contain helpful information. *(Sample problem that uses concept 1 and its step-wise solution)* *(Sample problem that uses concept 2 and its step-wise solution)* *(...)* *(Sample problem that uses concept n and its step-wise solution)* With what you have learned, solve the following problem. Make sure to show your work before giving the final answer. *(Problem statement)* In addition, you may also find the following information helpful: *(List of all hints.)*
Solver	**(Model-generated full solution)**

Table 21: The "Problem" concept provision prompt. If there are no hints, then the colored text is omitted.

D Model-Generated Solutions and First Wrong Step Annotations

Tab. 22 and 23 show one solution generated by each model. The final answer (FA) correctness and full solution (FS) correctness are separately noted. When the FS is incorrect (regardless of the FA correctness), the first wrong step (FWS) is highlighted in red, which consists the FWS corpus annotation.

Solution Author: GPT-3.5	Final Answer: Correct	Full Solution: Incorrect

Problem ID: P_Inequality_10
Problem: For $x, y, z > 0$, what is the smallest value of $x^2/y^2 + y^2/z^2 + z^2/x^2 - (y/x + z/y + x/z)$?
Answer: 0

Model-generated solution:
To find the smallest value of the expression, we can use the AM-GM inequality.
By AM-GM, we have:
$x^2/y^2 + y^2/z^2 + z^2/x^2 \geq 3\sqrt{(x^2/y^2)(y^2/z^2)(z^2/x^2)}$
Simplifying the right side, we get:
$3\sqrt{(x^2/y^2)(y^2/z^2)(z^2/x^2)} = 3\sqrt{(x^2y^2z^2)/(x^2y^2z^2)} = 3\sqrt{1} = 3$
Now, let's consider the second term, $-(y/x + z/y + x/z)$.
By AM-GM, we have:
$y/x + z/y + x/z \geq 3\sqrt{(y/x)(z/y)(x/z)}$
Simplifying the right side, we get:
$3\sqrt{(y/x)(z/y)(x/z)} = 3\sqrt{(xyz)/(xyz)} = 3\sqrt{1} = 3$
Therefore, the smallest value of the expression is:
$3 - 3 = 0$
So, the smallest value of $x^2/y^2 + y^2/z^2 + z^2/x^2 - (y/x + z/y + x/z)$ is 0.

Solution Author: GPT-4	Final Answer: Correct	Full Solution: Incorrect

Problem ID: P_Number-Theory_13
Problem: What is the smallest value of $|36^m - 5^n|$ for positive integers m and n?
Answer: 11

Model-generated solution:
To find the smallest value of $|36^m - 5^n|$ for positive integers m and n, we need to find the smallest values of m and n such that 36^m is as close as possible to 5^n.
Let's start with $m = 1$ and $n = 1$. We have:
$|36^1 - 5^1| = |36 - 5| = 31$
If we increase m to 2, we get:
$|36^2 - 5^1| = |1296 - 5| = 1291$
This is much larger, so increasing m is not a good strategy. Let's try increasing n to 2:
$|36^1 - 5^2| = |36 - 25| = 11$
This is smaller, so increasing n seems to be a good strategy. Let's try $n = 3$:
$|36^1 - 5^3| = |36 - 125| = 89$
This is larger again, so $n = 2$ seems to be the best choice. Let's check $m = 2$ just to be sure:
$|36^2 - 5^2| = |1296 - 25| = 1271$
This is much larger, so $m = 1$ and $n = 2$ is the best choice.
So, the smallest value of $|36^m - 5^n|$ for positive integers m and n is 11.

Table 22: Sample solutions generated by different models and their correctness. If the full solution is not correct, the first wrong step (FWS) is marked in red. Table continued in Tab. 23.

Solution Author: GPT-4 T	Final Answer: Correct	Full Solution: Correct

Problem ID: P_Combinatorics_40
Problem: Find the number of ways to fill a 2 x 11 rectangle with 2 x 1 tiles.
Answer: 144

Model-generated solution:
To solve this problem, we can use a recursive approach. Let's denote the number of ways to fill a 2 x n rectangle with 2 x 1 tiles as $f(n)$. We are looking to find $f(11)$.
Let's consider the first column of the 2 x 11 rectangle. We have two choices for placing the tiles:
1. We place two tiles
vertically in the first column. If we do this, we are left with a 2 x (11-1) = 2 x 10 rectangle to fill. The number of ways to fill the rest of the rectangle is $f(10)$.
2. We place one tile horizontally at the top of the first column. If we do this, we must also place another tile horizontally below it to fill the first two columns. This leaves us with a 2 x (11-2) = 2 x 9 rectangle to fill. The number of ways to fill the rest of the rectangle is $f(9)$.
Therefore, we can express $f(n)$ as a recursive function: $f(n) = f(n-1) + f(n-2)$
We also need to establish the base cases. For a 2 x 1 rectangle, there is only one way to fill it (vertically), so $f(1) = 1$.
For a 2 x 2 rectangle, there are two ways to fill it (either two vertical tiles or two horizontal tiles), so $f(2) = 2$.
Now we can compute $f(11)$ using the recursive formula:
$f(1) = 1$
$f(2) = 2$
$f(3) = f(2) + f(1) = 2 + 1 = 3$
$f(4) = f(3) + f(2) = 3 + 2 = 5$
$f(5) = f(4) + f(3) = 5 + 3 = 8$
$f(6) = f(5) + f(4) = 8 + 5 = 13$
$f(7) = f(6) + f(5) = 13 + 8 = 21$
$f(8) = f(7) + f(6) = 21 + 13 = 34$
$f(9) = f(8) + f(7) = 34 + 21 = 55$
$f(10) = f(9) + f(8) = 55 + 34 = 89$
$f(11) = f(10) + f(9) = 89 + 55 = 144$
So, there are 144 ways to fill a 2 x 11 rectangle with 2 x 1 tiles.

Solution Author: PaLM 2 M	Final Answer: Incorrect	Full Solution: Incorrect

Problem ID: P_Polynomial_11
Problem: If x_1, x_2 are the two roots of the polynomial $x^2 - 6x + 1$, for how many integer n in $\{61, 62, ..., 120\}$ does $x_1^n + x_2^n$ divided by 5 give a remainder of 4?
Answer: 10

Model-generated solution:
The two roots of the polynomial $x^2 - 6x + 1$ are $x_1 = \frac{3+\sqrt{5}}{2}$ and $x_2 = \frac{3-\sqrt{5}}{2}$.
We can write $x_1^n + x_2^n$ as follows:
$x_1^n + x_2^n = \left(\frac{3+\sqrt{5}}{2}\right)^n + \left(\frac{3-\sqrt{5}}{2}\right)^n$
We can factor the expression in the parentheses as follows:
$\left(\frac{3+\sqrt{5}}{2}\right)^n + \left(\frac{3-\sqrt{5}}{2}\right)^n = \left(\frac{3+\sqrt{5}+3-\sqrt{5}}{2}\right)^n + \left(\frac{3+\sqrt{5}(3-\sqrt{5})}{2}\right)^n$
Simplifying, we get:
$x_1^n + x_2^n = \left(\frac{6}{2}\right)^n + \left(\frac{-2\sqrt{5}}{2}\right)^n$
We can then write $x_1^n + x_2^n$ as follows:
$x_1^n + x_2^n = 3^n - 2^n\sqrt{5}^n$
We know that 3^n is always divisible by 5, since 3 is a factor of 5. We also know that $\sqrt{5}$ is an irrational number, so $\sqrt{5}^n$ is never divisible by 5. Therefore, $x_1^n + x_2^n$ is never divisible by 5.
In the range $61 \leq n \leq 120$, there are 60 integers. Since $x_1^n + x_2^n$ is never divisible by 5, there are 0 integers in this range for which $x_1^n + x_2^n$ divided by 5 gives a remainder of 4.
Therefore, the answer is $\boxed{0}$.

Table 23: Sample solutions generated by different models and their correctness. If the full solution is not correct, the first wrong step (FWS) is marked in red. Table continued from Tab. 22.