MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™ Part of the MACHINE LEARNING Series

BENCHMARK • OPEN ACCESS

Theoretical physics benchmark (TPBench)—a dataset and study of AI reasoning capabilities in theoretical physics

To cite this article: Daniel J H Chung *et al* 2025 *Mach. Learn.: Sci. Technol.* **6** 030505

View the article online for updates and enhancements.

You may also like

- First-class AI? Assessing LLMs on undergraduate physics exams
Ben Dowsett, Martin P Rey and Michele Pizzochero
- Physics simulation capabilities of LLMs
Mohamad Ali-Dib and Kristen Menou
- Using large language models for grading in education: an applied test for physics
Ryan Mok, Faraaz Akhtar, Louis Clare et al.

This content was downloaded from IP address 73.56.87.51 on 29/10/2026 at 15:49

IOP Publishing Mach. Learn.: Sci. Technol. 6 (2025) 030505 https://doi.org/10.1088/2632-2153/adfcb0

MACHINE LEARNING
Science and Technology

BENCHMARK

Theoretical physics benchmark (TPBench)—a dataset and study of AI reasoning capabilities in theoretical physics

Daniel J H Chung[1], Zhiqi Gao[2], Yurii Kvasiuk[1], Tianyi Li[1], Moritz Münchmeyer[1,5,*], Maja Rudolph[3], Frederic Sala[2] and Sai Chaitanya Tadepalli[4]

[1] Department of Physics, University of Wisconsin-Madison, Madison, WI, United States of America
[2] Department of Computer Science, University of Wisconsin-Madison, Madison, WI, United States of America
[3] Data Science Institute (DSI), University of Wisconsin-Madison, Madison, WI, United States of America
[4] Department of Physics, Indiana University, Bloomington, IN, United States of America
[5] NSF-Simons AI Institute for the Sky (SkAI), Chicago, IL, United States of America
[*] Author to whom any correspondence should be addressed.

E-mail: muenchmeyer@wisc.edu

Keywords: theoretical physics, reasoning, large language models

OPEN ACCESS

RECEIVED
6 April 2025

REVISED
7 July 2025

ACCEPTED FOR PUBLICATION
18 August 2025

PUBLISHED
2 September 2025

Original Content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence.

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

Abstract

We introduce a benchmark to evaluate the capability of AI to solve problems in theoretical physics (TP), focusing on high-energy theory and cosmology. The first iteration of our benchmark consists of 57 problems of varying difficulty, from undergraduate to research level. These problems are novel in the sense that they do not come from public problem collections. We evaluate our data set on various open and closed language models, including o3-mini, o1, DeepSeek-R1, GPT-4o and versions of Llama and Qwen. While we find impressive progress in model performance with the most recent models, our research-level difficulty problems are mostly unsolved. We address challenges of auto-verifiability and grading, and discuss common failure modes. While currently state-of-the art models are still of limited use for researchers, our results show that AI assisted TP research may become possible in the near future. We discuss the main obstacles towards this goal and possible strategies to overcome them. The public problems and solutions, results for various models, and updates to the data set and score distribution, are available on the website of the dataset tpbench.org.

© 2025 The Author(s). Published by IOP Publishing Ltd

Contents

1. Introduction 2
2. Properties of TPBench 5
2.1. Overview 5
2.2. Problem statistics 5
2.3. Auto-verification of solutions 6
2.4. AI-based holistic grading of the entire solution 8
2.5. Novelty and difficulty of our problems 8
2.6. Public and private data set and data leakage concerns 9
3. Model performance evaluation 9
3.1. Results for auto-verified solutions 10
3.2. Results for holistic AI-based grading 10
3.3. Augmenting inference with python to reduce algebraic mistakes 13
4. Failure mode analysis 13
4.1. Background knowledge of the model 13
4.2. Algebraic mistakes 14
4.3. Logical mistakes 15
4.4. Hallucinations 16
4.5. Performance of pre-o-series models 17
4.6. Performance of o1, o3-mini, and DS reasoning models 18
5. Related work 19
5.1. Mathematical reasoning benchmarks 19
5.2. Reasoning capabilities of LLMs 20
6. Discussion 21
Data availability statement 22
Acknowledgments 23
Appendix A. Summary of problem data 23
Appendix B. Prompts 23
B.1. Prompts to query problem solutions 23
B.2. Prompts to query grading of solutions 25
Appendix C. Public problems and solutions 25
C.1. Level 5—one-pole problem 25
C.2. Level 5—bias of a sampled halo field 30
C.3. Level 4—SHO vacuum entanglement 32
C.4. Level 4—SUSY-symmetry 37
C.5. Level 3—slow-roll inflation 38
C.6. Level 3—scalar particle scattering 39
C.7. Level 2—dark matter capture as a function of time 40
C.8. Level 2—a 3-state QM problem 41
C.9. Level 1—blackbody in d dimensions 42
C.10. Level 1—boosted parabolic trajectory 42
References 42

1. Introduction

Automated mathematical reasoning at research level with AI in theoretical physics (TP) may now be within reach. Novel large language model (LLM)-based AI systems, powered by improved AI reasoning techniques at training and inference time, are potentially powerful tools for the TP community. If substantial parts of the theoretical research process could be performed by AI, this would allow to significantly accelerate progress in TP. If AI could act as a fast, reliable and skilled research assistant that can perform theoretical calculations and solve mathematical problems, human researchers could cover substantially more theoretical ground, evaluate more ideas for their promise, and thus make more theoretical discoveries. Even without super-human intelligence, an AI 'craftsman' would allow humans to outsource tedious calculation work and to focus more on creative aspects of the theoretical research process.

Recent advancements in LLMs have allowed models to solve progressively more difficult tasks that require abstract mathematical reasoning. While high-school level math competition benchmarks like `MATH` [1] are almost saturated by current models, the focus has recently turned to graduate level and research level mathematics. A main data set in this domain, the recently introduced `FrontierMath` [2], which contains

research level difficulty problems, is still mostly unsolved by frontier models. In TP, which also requires extensive abstract mathematical reasoning, there has been comparatively less work than in mathematics. Existing benchmarks which include physics such as `JEEBench` [3], `OlympiadBench` [4] and `PhysicsQA` [5], cover mostly high-school-level problems from college entrance exams or competitions. There is little existing work on mathematical reasoning for TP at graduate or research level. An exception is [6], where the authors evaluate the performance of LLMs for symbolic calculations in quantum many-body physics, however in the narrow context of a specific physical setting. Very recently, the `Humanity's Last Exam` dataset [7] (HLE) appeared as a multi-domain benchmark that includes problems from TP. We provide a more complete list of available data sets in section 5.1.

In the present work, we build a data set to test TP reasoning skill over a broad range of difficulty. We aim to answer the following questions:

- How good is the current state-of-the-art AI for problem-solving in TP? Are existing models useful for research-level reasoning?
- What are the most common failure modes? For example, are models performing correct reasoning but fail mostly at algebra (at which LLMs are known to perform poorly)?

To answer these questions, we created a new benchmark data set `TPBench` of TP problems of varying degree of difficulty, from advanced undergraduate to research level. Our problems are novel, in the sense that they do not come from public problem collections (see section 2.5 for detailed comments). For graduate level and research problems we focus in particular on problems from high-energy physics and cosmology. An important property of our data set is that it provides a *continuum* of problem difficulty, from easy to research level, which few mathematical data sets do. This allows us to compare the performance of different models over a wide spectrum of difficulty. We invite the reader to skip ahead to appendix C to get an impression of the difficulty of these problems. Before discussing our data set in detail, we begin with some general remarks about reasoning for TP and its relation to AI models.

Differences between reasoning in math and TP. Because TP is extremely broad and math is arguably even broader, any summary discussion of the differences between mathematical and physics reasoning is unlikely to be accurate in many examples in a generic comparison set. Nevertheless, in terms of modern graduate level and higher physics and mathematics comparisons, several aspects typically stand out.

- Mathematical reasoning tends to focus on establishing exact broad statements constructed within a rigid logical framework, while TP reasoning mostly deals with approximate narrower statements constructed within a logical framework in which some of the less quantitatively relevant details are left unspecified but 'most likely' can be filled in such that the statements can be made arbitrarily precisely if desired[6]. This difference naturally stems from the different approximate goals of each discipline: a commonly accepted goal of TP is to model nature while a commonly accepted goal of mathematics to construct nontrivial, beautiful true statements connecting surprisingly disparate ideas [15]. The emphasis on rigidity is what naturally leads to the format of theorems and proofs in mathematics while the emphasis on quantitative modeling has allowed the Standard Model of particle physics to make successful predictions despite the evolving nature of its underlying mathematical structure.
- TP reasoning primarily relies on techniques of direct computations, while mathematical reasoning tends to use more often indirect techniques such as contradiction and induction. More explicitly, TP computations often utilize algorithmic methods in calculus, linear algebra, complex analysis, differential equations, differential geometry, and group representation theory.
- TP reasoning often focuses on derivations of formulas whose parametric dependences as well as the overall normalization are implicitly defined in a narrow domain of physical relevance. For example, if one writes down a quantum field theory (QFT) Lagrangian and computes observables, the coupling constants with conventional normalization cannot be a large number such as 1000 since such theories are expected to have the field degrees of freedom reorganize into a different effective theory. However, the exact parametric range of validity for the coupling constant is left implicit. This is in contrast with much of mathematical reasoning, where parametric ranges are precisely defined. This makes TP reasoning quite efficient at the expense of imprecision in the domain of validity.

[6] Certain corners of TP such as formal general relativity and string theory come very close to the reasoning style of mathematics (e.g. [8–14]). This will not be treated here, and this in some sense is covered by the LLM literature dealing with mathematics.

- TP typically focuses on approximations whose quantitative uncertainties are often left unspecified. For example, one of the most popular computational techniques in TP is perturbation theory, a type of asymptotic expansion, which often has a zero radius of convergence, and because there is often no exact computation to compare to, there is no rigorous quantitative estimate of uncertainties in most cases. One typically understands the estimate of the uncertainty to be the next order contribution in perturbation theory. Researchers also implicitly understand that there are non-perturbative contributions such as instantons which have an exact representation of zero in perturbation theory that can become important in certain instances.

These properties make TP an exciting testbed for AI reasoning models, which has not been extensively explored, perhaps because models were not powerful enough to do so, until very recently.

Generating novel research ideas/problems in TP. Novel research in TP, as in all fields of science, is usually incremental, and novel research ideas are combinations or further developments of prior work. For example, once a novel method has been invented, it can often be applied to many different problems. Indeed, Feynman advised to keep a list of favorite problems, and to check whether any newly learned technique could be useful for one of these problems [16]. Experienced researchers have an advantage over students at generating interesting research because their knowledge base is much larger and more interconnected. Indeed, what makes a research level question different from a classroom question is often the novelty and connection with existing knowledge and not the reasoning difficulty. It seems very plausible that machine learning models, with their ability to ingest vast amounts of knowledge during training or inference, could be particularly strong at finding promising combinations of novel results and techniques. A recent study in NLP research [17] found that LLM research ideas are rated more novel (but slightly less feasible) by human experts than human expert ideas. Experienced researchers are also able to judge whether a mathematical result is interesting or surprising and deserves further investigation. Such 'theoretical taste' may be beyond existing AI models. With our data set, we are not currently aiming to test these aspects of theoretical research.

Reasoning abilities required to solve research problems in TP. Researchers (consciously or unconsciously) have a number of techniques or heuristics to solve theoretical problems. A famous collection of problem solving techniques and advise is George Polya's book *How to solve it* [18] which lists about 50 heuristics with suitable examples in mathematics. Techniques include decomposing the problem, finding a related problem, generalization, and many less obvious ones. Most researchers have a more limited toolkit than Polya and many novel papers are somewhat straight forward combinations of reasoning steps contained in previous works. A main difficulty in this case is to understand this prior work and be able to recall and connect it when needed. Of course, insights are also often re-discovered independently. When solving a hard problem, researchers may try many different paths or heuristics, jump back and forth in their reasoning chain, analyze examples, answer subquestions, clear up their misunderstandings, read related literature, etc. In principle, given a large enough context window for prior thoughts and unlimited inference time, LLMs may be able to perform such very long thought processes, but currently available models (with a public reasoning chain) do not show very deep thought processes in our experience.

Technical (calculation) abilities required to solve research problems in TP. Once a mathematical reasoning step has been proposed, it needs to be executed correctly. This step is in principle straightforward but error-prone for most humans. For example, one may decide to Taylor expand an expression to third order, perform a Gaussian integral, re-arrange terms, or even just multiply numbers. LLMs are well known to perform poorly at such tasks, but this problem can in principle be fixed by using computer algebra systems, if they can work with the required mathematical objects (which however is often not the case in TP).

Observations from our evaluation. We list some observations from our experiments, which we discuss in more details in the following sections.

- Progress has been very rapid with the most recent models. When we initiated this project, GPT-4o [19] (released on May 2024) was state-of-the-art and unable to solve almost any TP problem beyond undergraduate level. When the o1-preview model [20] (released on Sep 2024) appeared, it could solve many easy graduate level problems, but rarely any harder ones. The o3-mini series [21] (released on January 2025), is able to solve about half of our advanced graduate level problems and even a few research problems. Nevertheless, as we will see, research problems involving long mathematical arguments are generally unsolved.
- Symbolic calculation mistakes. Existing models are known to perform poorly at mathematical calculations (see e.g. [22]), which could be performed correctly with a computer algebra system such as `SymPy`

or `Mathematica`. Such wrong intermediate results then lead to incorrect followup reasoning. It should be noted that humans tend to make similar mistakes in calculations, but are often able to spot them on revisiting. We made an initial attempt to encourage symbolic verification with python, which we describe in section 3.3, but found that it barely improved results. Better symbolic tool integration would be very beneficial for TP reasoning.
- Logical mistakes and lack of information about uncertainty. LLMs are generally poor at self-correcting [23] and typically cannot provide very useful information of where they are uncertain [24]. Many techniques have been proposed to mark mistakes (such as asking a different model to verify) [25–28], and for mathematical reasoning it would be particularly important to improve and include them. For lengthy reasoning chains, logical errors are a significant problem because human experts often need to perform solutions in detail themselves before being able to spot errors. Humans are often aware where in a derivation they are uncertain and can ask for help, or investigate further themselves.

The paper is organized as follows. In section 2 we discuss the properties of our data set, including the origin of problems and our approach to verification and grading. In section 3 we benchmark popular closed source and open source models on this data set. In section 4 we analyze the output of these models in more detail, and categorize their failure modes. In section 5 we discuss related work. Finally in section 6 we discuss future directions to improve AI-based reasoning in TP.

2. Properties of TPBench

2.1. Overview

We have curated a dataset of problems and associated solutions in main areas of TP. For research level problems we currently focus on high-energy theory and cosmology, the main expertise of the authors. Problems in our collection should have the following properties (similar to `FrontierMath` [2]):

- The problem is well-posed and the solution to the problem is unambiguous. An expert in the field, after reading the solution, should not have any objections.
- The problem is original. The solution to the problem cannot be easily found in the existing literature.
- The answer should be auto-verifiable. This is easily achieved for numerical answers or simple algebraic expressions, but more difficult for tensor expressions. We discuss this property further below.
- It should not be possible to guess the answer or remember it from the literature, despite a wrong reasoning chain.

It is hard to strictly enforce all these conditions in TP, as we discuss further below. Problem originality and the possibility to guess the answer can be judged differently by different researchers. For this reason we also provide metadata for each problem individually. We point out potential shortcomings in instances where we are aware of them. We include problems of varying degrees of difficulty, from undergraduate to graduate and to research problems. Naturally, research problems are more difficult to create, especially when requiring the answers to be novel and unpublished. Furthermore, more difficult problems are often more novel than easier problems (since the space of possible problems grows rapidly with their complexity). We discuss the aspect of novelty of our problems in more detail below, as well as individually in the problem metadata. We also make sure that our problems do not contain steps where a human would need a calculator to solve them (e.g. no floating point operations).

We now discuss the attributes of our data set in more detail, including their statistical distribution. We aim to enlarge and diversify the data set further in the future. We also provide ten sample problems in appendix C and we encourage the reader to browse the problems to get an impression of the whole data set.

2.2. Problem statistics

The dataset is categorized into five difficulty levels: *1—easy undergrad*, *2—undergrad*, *3—easy grad*, *4—grad*, and *5—research*. This classification ensures that the dataset can accommodate a wide range of use cases, from introductory studies to cutting-edge research challenges. The distribution of problems across these difficulty levels is detailed in table 1. For difficulty level 1–4 this means that the problem could appear in a homework problem or exam for students. For level 5, this problem could appear as a nontrivial step in a publication: i.e. our research level problems are sub-problems that would constitute a part of a publication, and are not by themselves large enough to constitute an entire publication. Solving level 4 and 5 problems would make models useful for theoretical research, but would not mean that models could write their own publishable papers (by a significant margin). Indeed, one of the most important steps in TP research is establishing why a particular question is important and organizing a string of level 5 type of steps to answer that question.

Table 1. Distribution of problems by difficulty level.

Difficulty level	Number of problems	Percentage
1—Easy undergrad	8	14.0%
2—Undergrad	13	22.8%
3—Easy grad	11	19.3%
4—Grad/easy research	14	24.6%
5—Research	11	19.3%

Table 2. Distribution of problems by domain. The 'Other' category includes astrophysics, electromagnetism, quantum mechanics, statistical mechanics, and classical mechanics. Many problems are in between areas. For example some Cosmology problems could also be classified as High Energy Theory.

Domain	Number of problems	Percentage
Cosmology	19	33.3%
High energy theory	18	31.6%
General relativity	4	7.0%
Other	16	28.1%

Future iterations of this data set could include more open-ended research problems, more reminiscent of a research publication.

The problems in the dataset span specialized domains, including *cosmology*, *high energy theory*, and *general relativity*. The less difficult problems span a wide area including astrophysics, electromagnetism, quantum mechanics, statistical mechanics, and classical mechanics. This domain-specific focus ensures the dataset's relevance to theoretical research related to the fundamental laws of nature, while the less difficult problems allow us to establish as a baseline what a successful AI performance looks like. Table 2 provides an overview of the distribution of problems by domain. In the future, we aim to include problems from other domains of TP, such as condensed matter theory.

The dataset includes problems from various sources, in particular unpublished research, private coursework, and recently published research papers. Almost half of the problems are novel (e.g. most of the level 3, 4, and 5 problems), having been created specifically for this dataset, while others draw on course-related material of the authors. A small number of problems have been taken from very recent publications (e.g. [29]).

2.3. Auto-verification of solutions

To automate the evaluation pipeline, we developed a system inspired by how coding competitions validate their results. We introduced the requirement that the final answer to each problem be provided as a `Python` callable with the specified signature. We then developed a simple automatic (not LLM-based) grading agent that, given the model's answer and the correct solution, extracts the code, creates, and executes a consistency-check script. This approach allows for efficient evaluation of algebraic answers and automatically ensures that equivalent correct answers are classified as such. Additionally, it is flexible enough to verify answers involving a variety of special functions or answers that involve several outputs. In some problems, the natural system of units ($c = \hbar = 1$) is specified in the prompt, while in other cases we pass constants of nature as function arguments to be unit agnostic. Alternatively, we could have adopted other automatic verification strategies. We could have provided numerical test cases in the prompt, but this would have led to lengthy problem statements, floating point operations, and much less flexibility. Another option is to consider multiple-choice answers, but this would make it easier to guess the answer without detailed understanding. Yet another possibility is to use another LLM as a grading agent and instruct it to compare the given solution to the true one. However we found that this approach is very error prone and LLMs are often not able to check mathematical equivalence of expressions (see below).

Our proposed scheme gives the flexibility to check the variety of classes of answers exactly. The verification process consists of three components:

1. **Code extraction:** The system extracts Python functions from both the model's solution and the expert solution.
2. **Test case execution:** Both functions are executed with identical test inputs across multiple parameter combinations.
3. **Output comparison:** Results are compared numerically with appropriate tolerances for floating-point arithmetic.

Each problem in our dataset is accompanied by a comprehensive set of test cases, carefully designed to probe both the physical validity and mathematical correctness of solutions. These test cases span different parameter ranges (e.g. negative or complex arguments where appropriate), to ensure thorough verification.

To illustrate this approach, consider the following undergraduate-level example:

Problem statement: A photon with the energy E scatters on an electron at rest at angle θ in the electron's reference frame. Find the angular frequency ω of the scattered photon.

Answer requirements: Provide the answer in the form of a `verbatim` function with the following signature:

```
#let c be the speed of light, m_e - electron mass, h_bar - reduced Planck constant
def omega_scattered(E: float, m_e:float, theta:float, c:float, h_bar:float) -> float:
    pass
```

Model answer:

$$\omega = \frac{1}{\frac{\hbar}{E} + \frac{\hbar}{mc^2}(1-\cos\theta)}$$

```
import math
def omega_scattered(E: float, m_e:float, theta:float, c:float, h_bar:float) -> float:
    return 1/(h_bar/E + h_bar/(m_e*c**2)*(1-math.cos(theta)))
```

This example demonstrates several key aspects of our auto-verification approach. First, the problem statement is clear and unambiguous, requiring a specific physical quantity (ω) to be calculated. Second, the answer requirements explicitly specify the expected format of the solution, including the function signature and parameter types. This standardization enables automated testing across different parameter regimes. Third, the model answer provides both the analytical expression and its implementation in Python code, allowing for direct numerical verification.

Furthermore, our verification system incorporates several safeguards to ensure reliable evaluation:

- **Timeout mechanisms:** Each function execution is limited to a maximum runtime of 30 s. This prevents infinite loops based on the model's incorrect reasoning while allowing sufficient time for complex calculations.
- **Error handling:** The system catches and classifies runtime exceptions, including syntax errors and memory issues. Invalid solutions are automatically flagged incorrect.
- **Parameter space coverage:** Test cases are generated to cover different regimes of the parameter space while maintaining numerical stability.

While our verification system works well for many problems, certain TP problems present challenges:

- **Tensor expressions:** Problems involving abstract tensor expressions (e.g. $R_\mu{}^\nu = \mathrm{d}\omega_\mu{}^\nu + \omega_\mu{}^\alpha \wedge \omega_\alpha{}^\nu$, $\mathcal{L} = \epsilon_{\mu\nu\rho\theta}F^{\mu\nu}F^{\rho\theta}$, or $\nabla_\mu T^{\mu\nu} = 0$) often have multiple equivalent representations due to symmetries. For instance, the Riemann tensor $R_{\mu\nu\alpha\beta}$ exhibits several symmetries including certain index permutations and the Bianchi identity.
- **Differential expressions:** Verification of expressions involving derivatives, especially of fields, presents special challenges. Derivative expressions must satisfy constraints such as the product rule, chain rule, metric compatibility, and recognize the group representation of the field the derivative acts on: e.g. $D_\mu\phi$ has a different elementary calculus expression than $D_\mu\psi$ even for the same gauge group and symbol D_μ if ϕ and ψ have different representations of the group. Indeed in situations where the intermediate result is a differential equation in a system with gauge invariances, knowing whether the two sets of differential equations (here the differential equation itself being the solution to the physics related problem) are physically equivalent can become nontrivial.
- **Integral expressions:** Integral expressions would be even more difficult to check numerically than differential expressions. Furthermore, they have many of the same challenges for verification as the differential expressions in terms of equivalence classes, as can be seen in the expression $\int_{\mathcal{M}} \mathrm{d}(H \wedge B) = \int_{\partial\mathcal{M}} H \wedge B$. For

the typical case of vanishing fields at infinity, there are also equivalences up to total derivative terms: e.g. $\int [\mathrm{d}\phi \wedge *\mathrm{d}\phi + \mathrm{d}C] = \int \mathrm{d}^4x\sqrt{-g}\partial_\mu\phi\,\partial^\mu\phi$.

- **Manifolds:** Furthermore, in cases where the solution to the problem is a manifold (often expressed as a metric), there is an infinite number of different equivalent algebraic expressions depending upon the coordinates used. An example of this can be seen in asking for a non-compact, static, spherically symmetric, asymptotically flat vacuum solution to the Einstein equations which has a Komar mass of M. A more abstract related situation of difficult-to-identify equivalence class is when two quantum field theories can be mapped to one another by integrating in and out different degrees of freedom (which abstractly covers the situation of renormalization group equivalence as well).

Although this list is common for TP problems in the literature, it can be extended depending on the classes of mathematical objects that need to be covered. The obvious common theme is the wealth of equivalence classes that the verification system needs to be aware of if it were to be generally applicable.

In our current data set, we only include problems where the above issues do not occur, i.e. where the final answer is an algebraic expression without tensors, derivatives, integrals, or manifolds. Of course, these objects do occur in the solution, but not in the final answer. In the future, it would be interesting to develop auto-verifiers for expressions involving these more general mathematical objects listed above. We have reserved a number of such problems for future iterations of the dataset that would be useful for testing dedicated more general verification codes.

2.4. AI-based holistic grading of the entire solution

In addition to auto-verification, we also employ AI-based grading. In this process, the grader model has access to both the expert-labeled solution and the LLM-generated solutions from a separate model, and is tasked with assigning grades (A–D). This approach mirrors how a human teaching assistant grades homework, where partial credit is given for correct reasoning steps, even if the final solution is incorrect. Moreover, holistic grading can identify instances where a solution arrives at the correct answer using incorrect reasoning, which occurs in a small number of our problems. While holistic grading is conceptually preferred, we observe significant disagreement between different grader models as well as humans.

2.5. Novelty and difficulty of our problems

Most of the problems presented here are constructed based on those given in standard courses as well as unpublished research related notes. For example, the solution to the research-level problem 'One pole problem' (see appendix C.1), without steps explained, is given in a footnote of [30]. Most of the research level problems would be readily doable by a good TP graduate student, and some of these are not much different from hard problems in graduate courses whose problems and solutions can be found publicly. However, we have made significant efforts to construct or modify problem statements so that the answers cannot be found by web search. Most of the research-level problems use typical or not-too-atypical notation to simulate a research setting, although this may facilitate literature recall (rather than reasoning) by the model[7].

The difficulty of a problem can vary along different axes, i.e. problems may be easy or hard for different reasons. We aimed to provide a sampling of this space:

- Some of the problems are difficult for a human researcher because they may not know that a similar problem has already been solved in the literature. Indeed, almost all solution techniques used in literature evolve over time incrementally as people build upon results of previous related computations. This gives LLMs an advantage for many problems, especially if the problem statement makes it clear what literature knowledge is required (which we try to avoid). Fortunately, publications often omit minor reasoning steps, and asking the model for detailed mathematical derivation can thus reveal such literature memory. For examples of models solving difficult problems by using 'superhuman literature knowledge' see section 4.6. Indeed, a key challenge in constructing this data set was to avoid this phenomenon as much as possible, to reveal true reasoning.
- Another obvious often encountered difficulty in research is simply the accuracy of routine calculus/algebraic manipulations. The probability of errors increases with the number of steps needed to reach the answer as well as the number of variables that are involved. LLMs are not currently performing very well with such long calculations.
- More truly physical setting (e.g. experimental setting) related TP problems contain larger number of seemingly-disorganized set of variables, in contrast with more formal setting TP problems that contain a

[7] An interesting followup study would be to vary variable naming and other notation to evaluate this point.

well-organized set of variables (typically using group theoretic structure). Some of our problems have been designed specifically to test whether the AI can reason using a seemingly-disorganized set of variables.
- Some of the problems have been given with a great deal of contextual information (such as the 'One pole problem' in appendix C.1), but others require a much more contextual interpretation (e.g. appendix C.2). In some sense, such 'less specific' problems are similar in difficulty as the problems requiring literature recall. If the LLM pattern matches the words in the problem to solution patterns in the literature, the LLM can be deemed to have understood the context.
- The 'One pole problem' in appendix C.1) also tests diagrammatic reasoning skills which are slightly more abstract than Feynman rules. This is part of a small number of problems in our data set where humans would use the help of diagrams to reason through them, and their expert solutions sometimes contain diagrams, usually in the `TikZ` LaTeX format. More generally, graphical languages such as TikZ (particularly with its Feynman diagrammatic extension TikZ-Feynman [31] and other such extensions) might be a good language with which to develop an LLM's graphical reasoning skills because of its efficiency in capturing the mathematical content of the diagrams.

2.6. Public and private data set and data leakage concerns

We make 10 of our problems and solutions public (see appendix C and tpbench.org), two for each difficulty level, such that they can be used to understand the data set, develop inference algorithms and examine failure modes. Naturally these problems will be part of future training data. To deal with this challenge, we also keep a large part of our data set private, currently about 50 problems. If you would like to evaluate your model on our private data set, please contact the authors directly.

Guaranteeing that private data does not end up in future training data is challenging. OpenAI, which we have used extensively, adds user interface chats to its training data but does not add API calls. Correspondingly, we have generally used API calls for querying problem solutions. However, in early phases of this projects, some problems were run in the user interface. In future iterations of this project, we will emphasize data leakage control further, especially for research level problems. We note that a small number of research problems is sufficient to evaluate significant model progress, as long as for these problems data set leakage control and originality of the problem are flawless. For our current problem set, we only enforce that problems (and especially solutions) do not appear publicly accessible online. Furthermore, we took particular care that expert solutions to problems were never passed to the ChatGPT user interface, where they could be added to future training data.

3. Model performance evaluation

In this section, we evaluate the performance of several leading models on our dataset, TPBench, across five different difficulty levels, ranging from undergraduate to research-level problems. Closed-source models include OpenAI GPT-4o, o1, and o3-mini [19–21]. Open-source models which we were able to run locally on our hardware include small and intermediate sized Llama 3.1, Qwen 2.5, and Qwen-QwQ, which is an experimental LLM that focused on advancing reasoning developed by the Qwen Team [32–34]. We also include the recent open-source reasoning model DeepSeek (DS) R1 [35] and its base-model DS V3 [36] which we ran on `Together AI` API. Finally, we tried to solve a subset of our research problems with OpenAI's Deep Research, including the problem in appendix C.1, primarily to spot solutions that could be found online. Deep Research was not able to solve any of these research problems. We believe our subset of models is representative of the spectrum of current LLM capabilities.

We provide the prompts for inference in the appendix B. The complete model answers from all models, for the public problems, can be found on the tpbench.org website. The evaluation considers two grading schemes: *answer-only* and *holistic*.

- **Answer-only (auto-verified) evaluation**. In the answer-only evaluation, models are tasked with producing a final answer to the problem, where correctness is assessed based on whether the model's answer matches the expected correct solution. This evaluation process is fully automated as decribed in section 2.3, with the correctness of the answer validated through numeric verification by the program.
- **Holistic AI-based grading.** In the holistic grading approach, we assess the reasoning process and the steps taken by the models. A separate LLM is provided with the problem statement, the expert solution, and the model's solution. It then evaluates the model's answer on a grading scale ranging from *A* to *D*. This grading scale accounts not only for the correctness of the final answer but also for the quality of reasoning, intermediate steps, and overall approach. Holistic grading is more lenient with minor errors or missing intermediate steps, and it provides partial credit for well-reasoned solutions even if the final answer is incorrect.

Table 3. Fraction of problems solved for each difficulty for each model.

	1-Easy undergrad		2-Undergrad		3-Easy grad		4-Grad		5-Research	
Model	avg@5	best@5	avg@5	best@5	avg@5	best@5	avg@5	best@5	avg@5	best@5
GPT-4o	0.75 (0.12)	0.88	0.86 (0.17)	1.00	0.25 (0.16)	0.45	0.09 (0.13)	0.29	0.00 (0.00)	0.00
o1 (high)	0.85 (0.05)	0.88	0.97 (0.04)	1.00	0.76 (0.24)	1.00	0.34 (0.13)	0.50	0.18 (0.07)	0.27
o3-mini (high)	0.97 (0.05)	1.00	1.00 (0.00)	1.00	0.87 (0.13)	1.00	0.57 (0.09)	0.64	0.15 (0.12)	0.27
DeepSeek-R1	0.95 (0.06)	1.00	0.98 (0.03)	1.00	0.76 (0.23)	0.91	0.49 (0.20)	0.64	0.07 (0.08)	0.18
DeepSeek-V3	0.72 (0.15)	0.88	0.80 (0.23)	1.00	0.29 (0.29)	0.64	0.11 (0.06)	0.21	0.00 (0.00)	0.00
Llama-3.1-8B	0.30 (0.06)	0.38	0.18 (0.20)	0.46	0.02 (0.04)	0.09	0.00 (0.00)	0.00	0.00 (0.00)	0.00
Llama-3.1-70B	0.45 (0.36)	0.88	0.52 (0.22)	0.77	0.11 (0.11)	0.27	0.04 (0.06)	0.14	0.00 (0.00)	0.00
Qwen2.5-7B	0.10 (0.11)	0.25	0.40 (0.21)	0.62	0.04 (0.07)	0.18	0.00 (0.00)	0.00	0.00 (0.00)	0.00
Qwen2.5-72B	0.60 (0.11)	0.75	0.42 (0.23)	0.77	0.24 (0.16)	0.36	0.04 (0.06)	0.14	0.00 (0.00)	0.00
QwQ-32B	0.62 (0.21)	0.75	0.60 (0.27)	0.92	0.07 (0.15)	0.36	0.01 (0.03)	0.07	0.00 (0.00)	0.00

Note: The number in the bracket is the average of model attempts' standard deviation per problems.

These two choices on their own are imperfect. The first one might consider a solution as correct which has two or more self-annihilating mistakes, or a solution that arrived at the correct answer with inconsistent or false reasoning. If the task is to evaluate the reasoning in challenging problem-solving, the binary grading system might not be representative to a satisfactory level. The second has the disadvantage of being somewhat arbitrary on assignments of grades for partial correctness. Our core results will use answer-only solutions.

3.1. Results for auto-verified solutions

We begin by discussing the answer-only results, which are the key empirical results of this paper. Our results are obtained using zero-shot reasoning where the model is given the problem statement and expected to reason through it without any prior examples. In fact, few-shot learning can degrade general performance in reasoning models [35]. We have experimented with prompt optimization, but found no significant differences (see appendix B for our prompts).

Table 3 presents the performance of each model across various difficulty levels, ranging from easy undergraduate problems (Level 1) to research-level problems (Level 5). The table reports the percentage of problems solved by each model. The columns labeled 'avg@5' represent the average score across five attempts, while the 'best@5' columns correspond to the average score of the best attempt out of five attempts. We visualize the 'average of five' solution percentage in figure 1(strong models) and figure 2. Finally, for our public problems, the individual results of models are given in appendix C. For example, we include one level 5 research problem that top models can solve and one that they cannot.

For the top models, o1, o3-mini and DS R1, undergraduate problems (levels 1 and 2) are now essentially solved, with performance of 95% to 100% for the oX models. For easy graduate problems (level 3), the performance is around 80%. For our level 4 graduate problems, some of which could appear in research investigations, the best models o1 and o3-mini solve around 50%, with o3-mini slightly beating o1. Research problems are mostly unsolved at this stage with a score around 15%. o1 slightly beats o3-mini here, which may be due to it having a larger literature knowledge to draw on.

Among mid-range models, GPT-4o and DS-V3 perform similary. They are between one and two levels of difficulty less powerful than the top models. Midrange models are essentially unable to solve problems above easy graduate level. Finally, lower parameter public models, which have the advantage that researchers can run them on invididual GPUs, cannot solve problems above undergraduate level. We also provide further model evaluation statistics of the data set on the website, including a unified model score over all difficulties.

3.2. Results for holistic AI-based grading

Table 4 presents the results for the holistic AI-based grading, which involves assigning letter grades (*A* to *D*) based on the quality of reasoning and correctness of the solution. This grading is not limited to the final answer but considers the overall approach taken by the model in solving the problem. We have used GPT-4o as a grader, as a currently mid-range model. We chose this model for cost efficiency reasons, and in the future we intend to use the most powerful model as a grader. The model was provided the grading prompt (appendix B), the expert solution, and the model solution to grade, similar to the way a human teaching assistant would work.

The models' performances are shown across the five difficulty levels. The letter grades represent the models' ability to produce correct solutions while demonstrating sound reasoning. An '*A*' indicates an excellent solution with minimal to no errors, a '*B*' suggests a good solution with minor mistakes, '*C*' indicates a solution with significant flaws, and '*D*' represents a fundamentally incorrect solution.

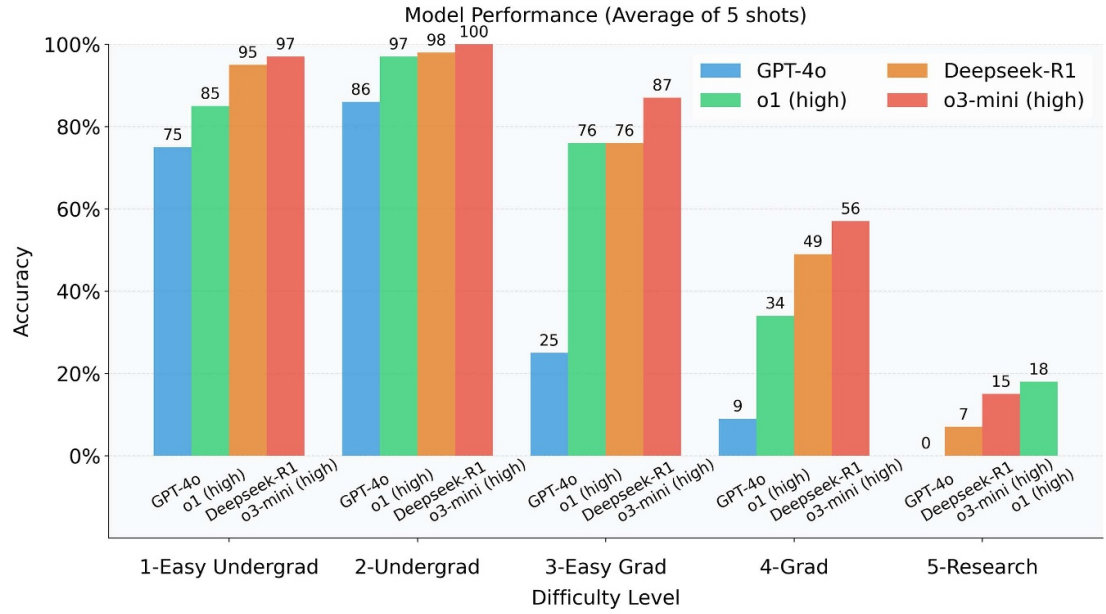


Figure 1. Accuracy of SOTA models by difficulty level.
Note: 'high' in brackets indicates reasoning effort.

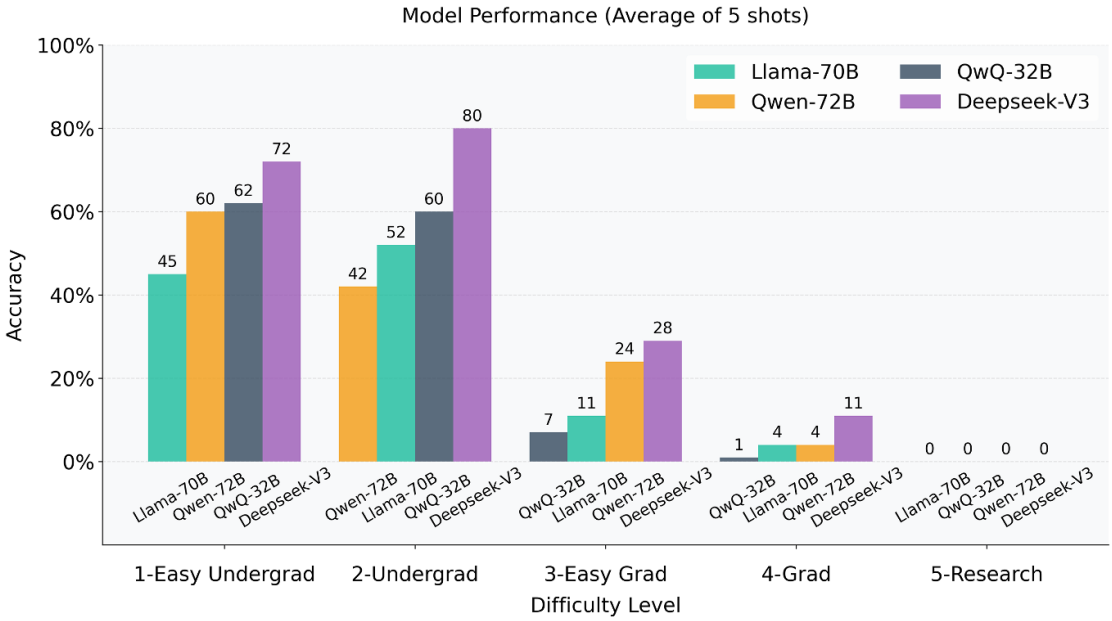


Figure 2. Accuracy of common open-source models by difficulty level.

In principle, the holistic grading system provides insights into the models' reasoning capabilities beyond just final correctness. However, we find some difficulties with holistic grading as we now describe. This is consistent with results showing that LLM-as-a-judge approaches have considerable bias [37].

Table 5 and the corresponding bar chart in figure 3 summarize how the automatically verified results (*Correct* vs. *Incorrect*) align with the letter grades (*A*, *B*, *C*, *D*) assigned by the AI-based holistic grading. For *A*-graded solutions, a large fraction (80.1%) aligns with the auto-grader's correct verification. By contrast, *B*- and *C*-graded solutions show substantially lower correctness rates (16.3% and 4.9% respectively). In the *D* category, an overwhelming 99.5% fail the auto-grader's check, indicating that both holistic assessment and numeric verification typically reject these solutions.

Overall, there is a strong correlation between higher letter grades and positive verification outcomes, which validates that the AI-based grading system's assessed quality generally corresponds to the auto-grader's numeric correctness checks. At the same time, deviations exist in each category. For example, nearly 20% of

Table 4. Letter grade received for different models.

	1-Easy undergrad				2-Undergrad				3-Easy grad				4-Grad				5-Research			
Model	*A*	*B*	*C*	*D*	*A*	*B*	*C*	*D*	*A*	*B*	C	*D*	*A*	*B*	*C*	*D*	*A*	*B*	*C*	*D*
GPT-4o	28	0	11	1	50	6	8	1	20	4	29	2	8	5	51	6	1	4	50	0
o1 (high)	36	0	4	0	60	5	0	0	48	3	4	0	41	4	22	3	23	9	23	0
o3-mini (high)	39	0	1	0	60	5	0	0	49	3	3	0	51	1	18	0	36	3	16	0
DeepSeek-R1	31	2	7	0	58	6	1	0	41	2	10	2	25	3	23	19	6	0	26	23
DeepSeek-V3	26	0	12	2	50	6	9	0	15	7	29	4	11	6	47	6	0	0	49	6
Llama-3.1-8B	11	1	15	13	4	7	25	29	0	1	13	41	0	0	15	55	0	0	8	47
Llama-3.1-70B	19	0	17	4	31	1	29	4	5	6	36	8	2	3	50	15	0	0	44	11
Qwen2.5-7B	3	2	24	11	22	4	25	14	3	1	22	29	0	1	34	35	0	0	32	23
Qwen2.5-72B	25	0	13	2	35	5	23	2	11	5	30	9	8	2	47	13	1	1	46	7
QwQ-32B	25	3	11	1	38	9	18	0	11	1	33	10	2	2	49	17	1	1	37	16

Note: the number of attempts per each level equals 5 shots times the number of problems in the level (see table 1).

Table 5. Grade verification results. Percentages in parentheses indicate the distribution of verification outcomes within each grade category.

Grade	Correct	Incorrect	Total
A	880 (82.2%)	190 (17.8%)	1070
B	61 (43.6%)	79 (56.4%)	140
C	74 (6.4%)	1075 (93.6%)	1149
D	5 (1.0%)	486 (99.0%)	491
Total	972 (34.1%)	1878 (65.9%)	2850

Note: The total number 2850 results from 5 attempts for each of the 57 problems in the data set across 10 models.

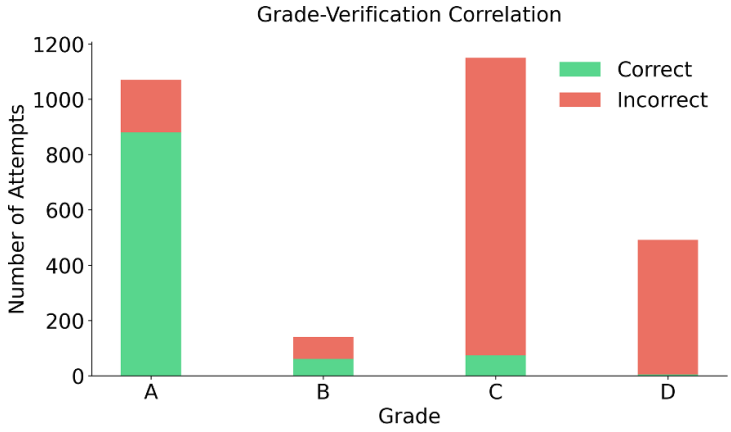


Figure 3. Stacked bar chart showing the number of solutions verified as correct (green) versus incorrect (red) across each letter grade.

A-graded solutions fail the numeric check, often because the AI holistic grader failed to correctly determine whether two answer expressions are equivalent. This is typically due to the expressions being overly complex. Conversely, a small fraction of lower-graded (*C* or *D*) responses may be mathematically correct in final form, yet insufficiently justified in intermediate steps, causing the holistic grader to assign a low grade despite correct numerical output.

Our findings illustrate that automatic verification and holistic AI-based grading are generally consistent: higher-quality solutions are confirmed as correct more frequently, while lower-quality solutions often fail numeric checks. Our current GPT-4o grader however has significant shortcomings. By cross-checking the grader with human grading, we find that LLM grading works reasonably well when grading solutions of low difficulty 1 to 2, but is not reliable at level 4 or 5. It seems likely that 4o is not strong enough to understand the logic of these higher difficulty problem solutions. Even when the grading model is as strong as the solver model, the success of holistic grading could be limited: LLMs are generally not very good at correcting their own results, as has been studied for example in [23]. In the present work, we thus focus on the auto-verifier results, and leave detailed exploration of holistic grading to future work.

3.3. Augmenting inference with python to reduce algebraic mistakes

We experimented with instructing models to break down calculations into smaller steps and verify these with python. Using a code interpreter was previously found to be beneficial in reducing algebraic mistakes in calculations (e.g. [38]) Our approach was based on the `MathChat` [39] framework and prompt tuning. We instructed the model to write python (particularly `SymPy`) code for each calculation step and verify its result using this code. In a few cases, for low difficulty problems, our approach was able to spot and correct mistakes. However, more often the approach disrupted the reasoning chain and led to worse results. For complicated problems, models struggled to identify steps that can be checked with `SymPy`. We note that our problems do not include floating point calculations, where verification would be straightforward, but require more complicated algebraic operations. Recently, the FrontierMath paper [2] included a set of prompts to encourage LLMs to verify with python, but noted that advanced models barely made use of this possibility. While human theorists do sometimes check their results with computer algebra systems, especially `Mathematica`, this process is not straightforward, and there is likely limited existing training data for this approach. We aim to experiment with few-shot inference or fine-tuning in the future, showing the model handcrafted examples of `SymPy` or `Mathematica` verification in the prompt. Since our current `MathChat`-based results are not stable we chose to defer this direction to future work[8].

4. Failure mode analysis

We now discuss common classes of mistakes. We present a few examples highlighting the various types of errors that the LLMs make while attempting to solve problems in TP. We broadly classify these errors into four classes as shown below. Our examples mostly draw from GPT-4o and o1 model results.

4.1. Background knowledge of the model

Background knowledge is a strength of LLMs. Problem authors were impressed by models' ability to recall relevant mathematical definitions that were not included in the problem but are known to practicing researchers. This ability makes it much easier in principle to solve problems than with a computer algebra system like Mathematica. For example, consider the level 5 cosmology problem from appendix C.2:

> **User:**
> In cosmology, large-scale cosmological dark-matter halo fields are biased tracers of the underlying Gaussian matter density δ_m. Assume we have a sample δ_m. We simulate a halo number density field by taking $n(\mathbf{x}) = \bar{n}\max(0, 1 + b\delta_m(\mathbf{x}))$, where bare number density $\bar{n}$ and bare bias b are specified constants. What is the bias of the sampled halo field? Derive an equation to evaluate the bias which depends on the bare bias and the variance in each pixel.

While well-defined for a cosmologist, the problem does not define the mathematical quantities in detail, and would be hard to interpret by a non-cosmologist. Advanced models correctly recalled the required definitions and generally set up the problem correctly.

However, while LLMs generally recall key definitions of various sub-fields of TP, they frequently encounter difficulties in accurately recalling more detailed mathematical information, as illustrated by the following two examples.

In one of the solutions to an undergraduate QM problem, the QwQ model incorrectly retrieves information about the Clebsch–Gordan coefficients. Specifically, it claims

> From standard tables or textbooks, the Clebsch–Gordan coefficients are:
>
> $$\langle 1\ m_1\ 1\ m_2 | j\ m \rangle$$
>
> For $j = 1$, $m = -1$:
>
> $$|1\ -1\rangle = \sqrt{\frac{2}{3}}|1\ -1\ 1\ 0\rangle + \sqrt{\frac{1}{3}}|1\ 0\ 1\ -1\rangle\,.$$

The correct value of these coefficients are $\mp 1/\sqrt{2}$ for $|1\ -1\ 1\ 0\rangle$ and $|1\ 0\ 1\ -1\rangle$ states respectively.

[8] We note that in followup work, we were able to leverage symbolic verification to improve model performance by employing an agent framework and parallel test-time scaling [40], however only for a limited set of mathematical operations.

In the following snippet, generated from a model answer, the GPT-4o model incorrectly identifies the standard eigenstates of a particle in a 1-D infinite potential well ($|x| \leqslant L/2$) from existing results[9]

$$\psi_n(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right) \quad \text{for } n = 1, 2, 3, \ldots$$

and

$$\psi_n(x) = \sqrt{\frac{2}{L}} \cos\left(\frac{n\pi x}{L}\right) \quad \text{for } n = 2, 4, 6, \ldots.$$

4.2. Algebraic mistakes

A major challenge for models is to perform correct algebraic calculations. Consider the following relatively easy math problem that appears as an individual step in one of our problem solutions.

User:
Determine the leading real term of the expression

$$F(k) = -1 + \left(\frac{59a^2k^2}{15} + \mathrm{i}ak + 1\right)^5 \exp\left(\frac{-1}{6}ak(85ak + 6i)\right)$$

for real $a, k \in \mathrm{R}$ and $k \ll 1$.

Expert solution:
The correct series expansion up to leading real and imaginary terms is

$$\lim_{k \ll 1} F(k) \approx \frac{793a^4k^4}{180} + 4\mathrm{i}ak. \tag{1}$$

We attempted this problem multiple times with o1 and o3-mini. In most of its responses, the LLMs did not expand the exponential term beyond the second order in k, falsely assuming that the leading real term must be proportional to k^2. In its best attempt, it expands up to quartic order in k, but fails to accurately combine the various terms to compute the coefficient of k^4. This is a good example of the promise of combining with computer algebra systems. If we add the prompt 'Write and execute `SymPy` code to evaluate the expression.' models can generate Python code that calculates the correct expression. We have therefore tried to encourage python usage as discussed in section 3.3, however with limited initial success.

Algebraic mistakes are numerous, and often occur even in very simple calculations. Models often simply forget mathematical terms in an expression from one calculation step to the next. For example, the following is an arithmetic evaluation by GPT-4o, in which it spuriously drops a factor of the imaginary number i:

$$\nabla^2\vec{E} = \mathrm{i}\omega(\sigma - \mathrm{i}\omega)\vec{E} = \omega(\sigma + \mathrm{i}\omega)\vec{E}.$$

Similar cases of forgotten i factors, minus signs or constants occur frequently in many problem solution attempts.

Mathematical identities are also often applied incorrectly. For example, in the following case GPT-4o fails to implement the vector triple product $\left((\vec{a} \times \vec{b}) \times \vec{c} = (\vec{a} \cdot \vec{c})\vec{b} - (\vec{b} \cdot \vec{c})\vec{a}\right)$ correctly and writes

$$\left(\vec{E} \times \hat{z}\right) \times \hat{z} = \left[\left(\vec{E} \times \hat{z}\right) \cdot \hat{z}\right]\hat{z} - \left(\vec{E} \times \hat{z}\right)(\hat{z} \cdot \hat{z}).$$

[9] The correct set of eigenvalues are

$$\psi_n(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right) \quad \text{for } n = 2, 4, 6, \ldots, \qquad \psi_n(x) = \sqrt{\frac{2}{L}} \cos\left(\frac{n\pi x}{L}\right) \quad \text{for } n = 1, 3, 5, \ldots.$$

More powerful reasoning models tend to make less frequent 'simple' math mistakes such as the following:

o3-mini: After performing the $\eta-$integrals (using the standard $i\epsilon-$prescription so that

$$\int_{-\infty}^{0} \mathrm{d}\eta\, \mathrm{e}^{\mathrm{i}K\eta} = \frac{1}{iK}, \qquad \int_{-\infty}^{0} \mathrm{d}\eta\, \eta\, \mathrm{e}^{\mathrm{i}K\eta} = -\frac{1}{K^2},$$

where the second integral erroneously contains a negative sign. It would be interesting to compare model performance on a set of automatically generated simple calculations typical for TP.

4.3. Logical mistakes

We frequently observed that LLMs struggle to accurately account for the validity and applicability of advanced mathematical concepts, such as incorrectly applying theorems, misinterpreting definitions, or failing to recognize the limitations of certain mathematical techniques. Consider the following mathematical problem, which we will use to discuss logical errors made by the oX series models in their attempted solution.

User:

By Taylor expanding the integrand, find a b cubic polynomial approximation to the integral

$$I(b) = \int_b^1 \left(\frac{\sqrt{\pi}}{x} - \frac{\pi\, \mathrm{erfc}\left(\frac{1}{\sqrt{x}}\right) \exp\left(\frac{1}{x}\right)}{x^{3/2} + x^{5/2}} \right) \mathrm{d}x,$$

that achieves a 90% or better accuracy when b lies in the interval $[0, 1]$.

Expert solution:

First, we note that the integrand remains finite in the limit $x \to 0$, as the singular term proportional to $1/x$ cancels out. To assess the validity of a series expansion, we estimate the radius of convergence near the boundary points $x = 0$ and $x = 1$. This analysis shows that the integrand is convergent within the interval $[0, 1]$ when expanded about $x = 1$. Consequently, we perform a Taylor expansion around $x = 1$ retaining terms up to $(x-1)^2$:

$$\frac{\sqrt{\pi}}{x} - \frac{\pi\, \mathrm{erfc}\left(\frac{1}{\sqrt{x}}\right) \exp\left(\frac{1}{x}\right)}{x^{3/2} + x^{5/2}} \approx \left(\sqrt{\pi} - \frac{1}{2} e\pi\, \mathrm{erfc}\,(1)\right) - \left(\frac{27\sqrt{\pi}}{4} - \frac{63}{8} e\pi\, \mathrm{erfc}\,(1)\right)(x-1) + \left(\frac{21\sqrt{\pi}}{8} - \frac{51}{16} e\pi\, \mathrm{erfc}\,(1)\right)(x^2-1).$$

Integrating over the interval $[b, 1]$, we obtain the approximate solution:

$$I(b) \approx (b^3-1)\left(\frac{17}{16} e\pi\, \mathrm{erfc}\,(1) - \frac{7\sqrt{\pi}}{8}\right) + (b^2-1)\left(\frac{27\sqrt{\pi}}{8} - \frac{63}{16} e\pi\, \mathrm{erfc}\,(1)\right) + (b-1)\left(\frac{83}{16} e\pi\, \mathrm{erfc}\,(1) - \frac{41\sqrt{\pi}}{8}\right).$$

In the limit $b \to 0$, our approximate expression evaluates to

$$I(0) \approx 1.546\,33,$$

which achieves approximately 93.6% accuracy compared to the numerical result 1.652 21. We note that this integral cannot be evaluated exactly using Mathematica or Maple software.

When the above problem was given to the o1 and o3-mini models, they demonstrated the following logical errors in their reasoning:

1. The model begins by identifying that the integrand is finite at $x = 0$. However, it fails to recognize that the radius of convergence for the integrand around $x = 0$ is 0. This oversight leads to an improper application

of approximations beyond the valid domain. Subsequently, the model factorizes the integral as

$$I(b) = \int_b^1 f(x)\,\mathrm{d}x = \int_0^1 f(x)\,\mathrm{d}x - \int_0^b f(x)\,\mathrm{d}x.$$

Assuming $b \ll 1$, the model proceeds to perform a Taylor expansion of the integrand around $x = 0$ and evaluates the second integral $\int_0^b f(x)\,\mathrm{d}x$ up to cubic order in b. However, the Taylor expansion is only valid within the radius of convergence, and this restriction is not respected, rendering the approximation potentially invalid.

2. To estimate the constant value of the first integral $\int_0^1 f(x)\,\mathrm{d}x$, the model imposes the boundary condition at $b = 1$, equating

$$\int_0^1 f(x)\,\mathrm{d}x = \lim_{b\to 1}\int_0^b f(x)\,\mathrm{d}x.$$

The model substitutes the cubic-order Taylor expansion solution for $\int_0^b f(x)\,\mathrm{d}x$ derived in the previous step into the right-hand side of this equation. This substitution constitutes a significant logical error in its reasoning, as the cubic-order approximation was determined only for $b \ll 1$, a fact that the model seemed to know but failed to implement. Extending this local approximation to $b = 1$, far beyond its domain of validity, leads to an erroneous evaluation of $\int_0^1 f(x)\,\mathrm{d}x$.

Interestingly, the o3-mini model demonstrates two critical flaws: it not only arrives at logically inconsistent conclusions but occasionally also confidently hallucinates the claim $I(0) = \pi/2$, failing to furnish a coherent proof despite repeated prompting.

In a different problem involving particle physics, the GPT-4o model was asked to determine the effective mass of a spin 1/2 particle with action

$$S = \int \mathrm{d}^4x\bar{\psi}\left(\mathrm{i}a\gamma^\mu\partial_\mu - c - \mathrm{i}\frac{b}{\sqrt{3}}\gamma^5\right)\psi.$$

The models did not understand and failed to reason out that the parameter c alone does not define the physical mass. The pseudoscalar γ^5-term must be included, corresponding to a chiral contribution to the mass.

For a much more basic example of failed logic, consider an example from QwQ. In one of our undergraduate Electrodynamics problems it produced the following expression followed by a faulty and rather incomplete reasoning:

$$\left(\vec{E}\times\hat{z}\right) = b\vec{E}.$$

But $\vec{E}\times\hat{z}$ is perpendicular to both $\vec{E}$ and $\hat{z}$, which suggests that $\vec{E}$ must be perpendicular to $\hat{z}$ for this equation to hold.

We found that advanced reasoning models such as o3-mini generally do not make such easy mistakes on undergraduate level physics problems. However, for difficult problems, they often oversimplify the problem due to a lack of detailed understanding. For instance, while solving the Level-5 problem detailed in appendix C.1, o3-mini and other advanced reasoning models approximate scale factor $a(\eta)$ by expanding it linearly around the transition point, η_e, not realizing that the pole is far from the transition point and thus one needs to apply $a(\eta) \sim \eta^2$. For further details, we refer the readers to the expert solution detailed in appendix C.1.

4.4. Hallucinations

Lastly, we present two instances where the LLM models generated new rules to obtain solutions that match with existing results in the literature. The following expression generated by GPT-4o represents an arithmetical hallucination error:

$$k = \sqrt{\omega\sigma} \approx \sqrt{\omega\sigma}\left(\frac{1}{\sqrt{2}} + \mathrm{i}\frac{1}{\sqrt{2}}\right).$$

The model performed the above arithmetic steps since it needed to determine the imaginary component of k. With this goal in mind, it carried out the above 'illogical' mathematical step inventing new arithmetic rules to justify its approach to obtaining the imaginary component from a wrong answer.

Another example, from our problems, of o1 hallucinating non-existent rules to justify its approach is the following excerpt from its solution:

> We can write
>
> $$T = \mathrm{Tr}\left[\gamma^{\nu}\ \not{p}_1\gamma^{\mu}\ \not{p}_2\,(1+\gamma_5)\,(1-\gamma_5)\right].$$
>
> Note that
>
> $$(1+\gamma_5)\,(1-\gamma_5) = 0.$$
>
> Therefore, to avoid vanishing of the trace, we need to consider that the γ_5 matrices need to be kept separate. Instead, we should expand the trace without combining the projectors.

There is no such mathematical rule that an apparent zero can (must) be avoided by separating the terms and adding them later. The LLM invented this 'rule' since it was working with incorrect expressions to nevertheless arrive at a correct solution, which in this case it was able to guess or recall (in only one of several attempts).

4.5. Performance of pre-o-series models

In our experience, models that are not explicitly trained for reasoning (i.e. before the oX series) can be used to assist researchers that reason through a simple problem, but with significant shortcomings. Consider the following easy mathematical subproblem that appeared in one of our recent works [29] in the context of cosmology, which we show here in simplified notation.

> **User:**
> Assume **a**, **b**, and **c** are vectors, and **N** is a symmetric positive-definite matrix. Let x and y be real numbers. I want to minimize $\mathbf{a}^{\top}\mathbf{N}\mathbf{a}$ under the constraints $\mathbf{a}^{\top}\mathbf{b} = x$ and $\mathbf{a}^{\top}\mathbf{c} = y$. Solve this for **a**, if possible.
>
> **Expert solution:**
> We minimize $\mathbf{a}^{\top}\mathbf{N}\mathbf{a}$ subject to the constraints $\mathbf{a}^{\top}\mathbf{b} = x$ and $\mathbf{a}^{\top}\mathbf{c} = y$. The Lagrangian $\mathcal{L}$ for this optimization problem is defined as:
>
> $$\mathcal{L}(\mathbf{a},\lambda,\mu) = \mathbf{a}^{\mathrm{T}}N\mathbf{a} + \lambda\left(\mathbf{a}^{\mathrm{T}}\mathbf{b} - x\right) + \mu\left(\mathbf{a}^{\mathrm{T}}\mathbf{c} - y\right).$$
>
> Taking the gradient of $\mathcal{L}$ with respect to **a** and setting it to zero yields:
>
> $$\nabla_{\mathbf{a}}\mathcal{L} = 2N\mathbf{a} + \lambda\mathbf{b} + \mu\mathbf{c} = 0$$
>
> which we can solve for **a** as:
>
> $$\mathbf{a} = -\frac{1}{2}N^{-1}(\lambda\mathbf{b} + \mu\mathbf{c}). \tag{2}$$
>
> The constraint equations are:
>
> $$\mathbf{a}^{\mathrm{T}}\mathbf{b} = x \quad \text{and} \quad \mathbf{a}^{\mathrm{T}}\mathbf{c} = y.$$
>
> Plugging the solution for **a** into the constraint equations gives
>
> $$\begin{bmatrix}\mathbf{b}^{\mathrm{T}}N^{-1}\mathbf{b} & \mathbf{b}^{\mathrm{T}}N^{-1}\mathbf{c}\\ \mathbf{c}^{\mathrm{T}}N^{-1}\mathbf{b} & \mathbf{c}^{\mathrm{T}}N^{-1}\mathbf{c}\end{bmatrix}\begin{bmatrix}\lambda\\ \mu\end{bmatrix} = \begin{bmatrix}-2x\\ -2y\end{bmatrix}$$

which is of form

$$\mathbf{M}\begin{bmatrix}\lambda\\ \mu\end{bmatrix} = -2\begin{bmatrix}x\\ y\end{bmatrix}.$$

The above linear system is solved by (assuming the inverse exists, e.g. the two bias vectors are not co-linear):

$$\begin{bmatrix}\lambda\\ \mu\end{bmatrix} = \frac{1}{\det(M)}\begin{bmatrix}\mathbf{c}^{\mathrm{T}}N^{-1}\mathbf{c} & -\mathbf{b}^{\mathrm{T}}N^{-1}\mathbf{c}\\ -\mathbf{c}^{\mathrm{T}}N^{-1}\mathbf{b} & \mathbf{b}^{\mathrm{T}}N^{-1}\mathbf{b}\end{bmatrix}\begin{bmatrix}-2x\\ -2y\end{bmatrix}.$$

We then substitute the solution for λ and μ back into **a** using equation (2).

That is a typical problem that GPT-4o and Llama-3 generally solve correctly, with correct mathematical derivation, although sometimes with a wrong numerical factor. It seems certain that this problem was in the training data of the model. Nevertheless, it is already time-saving for researchers to get answers to similar problems without manual labor. In particular for matrix algebra problems, existing computer algebra systems are not very strong or user friendly in our experience. However, the fact that models are very error prone limits their usefulness significantly. If every step needs to be checked in detail, the time saving can be minimal, or a wrong result can even confuse the user. Of course, human solutions can also have this property, depending on the skill and carefulness of the researcher.

4.6. Performance of o1, o3-mini, and DS reasoning models

From evaluating the output solutions generated by advanced LLM models such as o1, o3-mini and DS, we observe that these models exhibit significantly stronger reasoning capabilities compared to other LLMs tested in our study. Notably, these models can perform more difficult algebraic manipulations, identify different components of a problem, and connect them with established concepts in the literature. This ability allows them to make meaningful progress in research-level problems, including those from topics such as QFT and String theory, by pinpointing key aspects of the question and recalling relevant background knowledge.

However, these models still struggle with detailed and systematic logical reasoning. When tasked with solving our Level-4 and Level-5 problems, these models often perform well in the initial phase of problem solving, demonstrating promising insights. Yet, for problems requiring extensive calculations combined with step-by-step logical rigor (e.g. loop integrals in QFT, tensor manipulations in general relativity) and systematic justfication of the assumptions, their performance deteriorates significantly. Our analysis of multiple solutions suggests that when intermediate steps become too complex, the models (including DS) often resort to literature memory from pre-training rather than performing detailed calculations. Rather than explicitly detailing intermediate steps, the models often present only their final answer, recalling related literature knowledge without references or resorting to vague assertions such as 'after a lengthy (but straightforward) calculation' or 'a short calculation shows'. While the full CoT of the o-series models is not public, we have no evidence that the models genuinely perform relevant calculations internally in these cases.

As an illustrative example, when asked to compute the one loop anomalous magnetic moment of a fermion (e.g. [41]) including a contribution from a heavy scalar coupling, the model resorted to recalling existing solutions seen during pre-training rather than explicitly solving. However, it failed to recognize that the Yukawa interaction Lagrangian provided in our problem statement contained an additional factor of $1/\sqrt{2}$, which may deviate from the conventions in the literature. Consequently, its final answer overlooked this crucial modification. In a similar manner, when presented with the task of solving the Level-4 problem in appendix C.4, all advanced models (oX, DS-R1) initiate their response by articulating their interpretation of the problem statement and correctly identifying its connection to the standard supersymmetric transformations within the free Wess–Zumino model, as extensively documented in the literature. Subsequently, these models produce their final solution from memory. However, a consistent error emerges across all responses: the absence of the critical 'negative sign' as seen in the solution given in equation (108).

o3-mini: The well-known and consistent choice is

$$\delta_\eta\phi = \sqrt{2}\,\eta^\alpha\,\xi_\alpha,$$

with the Hermitian conjugate

$$(\delta_\eta \phi)^\dagger = \sqrt{2}\,\bar{\eta}_{\dot{\alpha}}\,\bar{\xi}^{\dot{\alpha}}.$$

This is verified by checking that the variations of all terms in $\mathcal{L}$ under the full set of SUSY transformations (including the ones for ξ and F) cancel (up to a total derivative).

While this might appear to be a minor discrepancy, it originates from a fundamental aspect of the problem. Specifically, the sign convention utilized in our given problem statement likely differed from the convention commonly adopted in the literature (for instance refer to section 5.2 in [42]) used within the models' training samples, thereby necessitating a corresponding modification in the final transformation rule. This seemingly subtle yet conceptually significant detail indicates a potential cognitive limitation in these AI models, reflecting an over-reliance on memorized patterns rather than a systematic, first-principles approach, as well as a failure to validate the appropriateness of the retrieved solution within the context of the specific problem statement.

As another example of literature memory, one of the problems in TPBench involves solving a nonlinear differential equation in a manner similar to how Chandrasekhar presents the Kerr solution in [43]. The number of steps to reach the answer is long and complicated. Such (complicated) recall problems are expected to be solvable by an AI due to its vast knowledge of the literature. Indeed, on one of the attempts, the AI can recognize the literature and write an answer to this problem, but even in that instance, it does not reason through the problem but just states:

o3-mini: In fact, after a (lengthy) calculation one finds that the only solution (consistent with the field equations and the asymptotic condition) is

$$e^X = \frac{r^2 + C_2\mu^2}{\sqrt{\Delta(r)}}.$$

These inconsistencies suggest that models' solutions often fluctuate based on how their internal sampling mechanism recalls (pre-)training data, rather than adhering to a logically coherent problem-solving strategy. This underscores a fundamental issue: unlike a proficient researcher who would maintain logical consistency across different attempts, these models exhibit uncertainty in their outputs, lacking a clear measure of confidence in their solutions. Such limitations and the opaque structure of the training and inference process (especially of closed-source models) present obstacles to their applicability in research settings. It appears that successfully solved high-difficulty problems often benefit from the very deep and interconnected literature memory of these models, in addition with their ability to translate this knowledge to the problem setting. While this ability is useful for research, it may not be sufficient to create novel TP results without human assistance. In summary, current model performance perhaps resembles a student with superhuman literature knowledge but low intellectual rigor and technical expertise.

5. Related work

Despite significant advances in the mathematical reasoning capabilities of LLMs, accurately solving reasoning problems in specialized domains, such as TP remains a persistent challenge. In math reasoning, the landscape of existing benchmarks has been instrumental for the evaluation of LLM reasoning capabilities and the development of more robust and interpretable reasoning strategies. We review related benchmarks in section 5.1 as well as common strategies for eliciting more accurate reasoning from LLMs in section 5.2.

5.1. Mathematical reasoning benchmarks

Recent progress in LLMs has enabled these models to tackle increasingly complex tasks that demand high-level abstract mathematical reasoning. A significant body of work has focused on datasets for mathematical reasoning at the middle-school (e.g. [44]), high-school (e.g. [1]), or undergraduate level (e.g. [45]), which often cover arithmetic, geometry, or math word problems. Other benchmarks are focused on theorem proving [46–48]. For example, the recently introduced `PutnamBench` [46] provides a collection of formalized theorems from the Putnam competition, while `MiniF2F` [47] and `FIMO` [48] offer datasets of formalized proof problems drawn from competitions like the IMO, AIME, and AMC. In addition, `ProofNet` [49] comprises both natural language and formalized theorem statements and proofs at the undergraduate

level. Complementary to these are natural language datasets that feature problems of varying difficulty [1, 44], as well as benchmarks like GPQA diamond [50], which are designed to be hard. Even more recently, the HLE dataset [7] is an industry-curated, multi-domain benchmark that includes very challenging problems, among them some from TP. However, problems in HLE are constrained to numerical answers or multiple choice formats, there is no spectrum of difficulty, and it is not specifically designed to probe reasoning capabilities in TP.

While lower difficulty math benchmarks such as MATH [1] have nearly been mastered by current LLMs, the FrontierMath [2] dataset, which includes research-level problems curated by working mathematicians, remain largely unsolved. FrontierMath spans a range of difficulties from high-school to research level and features properties like auto-verifiability and rich metadata, design principles we have also incorporated into TPBench. However, Glazer *et al* [2] provide limited information about the difficulty distribution and the specifics of the problems that have been solved by advanced models.

In the realm of physics, which also demands extensive abstract mathematical reasoning, the focus has been predominantly on high-school level challenges as seen in datasets such as JEEBench [3], OlympiadBench [4], and PhysicsQA [5]. Beyond undergraduate-level problems, very little work has addressed mathematical reasoning for TP. One notable exception is [6], which examines symbolic calculations, albeit within the narrow context of a specific class of quantum many-body physics problems.

Our new dataset, TPBench addresses the gap in TP reasoning benchmarks beyond the undergraduate level. TPBench encompasses problems ranging from undergraduate to research level, with research problems reflecting challenges typical of those found in TP publications (rather than representing entire publications in themselves). Importantly, TPBench is designed to be independent of industry control, ensuring that the TP research community has access to a reasoning benchmark that is not susceptible to data leakage from future training data. We look forward to sharing this dataset with collaborators under appropriate data leakage controls.

5.2. Reasoning capabilities of LLMs

Despite the remarkable fluency of LLMs in generating human-like text, their capacity to perform reliable multi-step reasoning remains a challenge [51]. Many LLMs still struggle with complex arithmetic and logical inference tasks. In this section, we review state-of-the-art methods, spanning both training-time and inference-time techniques that have been developed to boost the reasoning capabilities of LLMs.

Training-time methods for improved reasoning. Training-time methods encompass all strategies where pre-trained language models are fine-tuned or otherwise modified to improve their reasoning capabilities. The most popular approaches in this category rely on either supervised fine tuning [52–54], or reinforcement learning [35, 52] (or both [35]). In supervised fine-tuning [55–58], high-quality reasoning chains are curated and used to fine-tune models to display more accurate reasoning behavior. Chen *et al* [59] demonstrate that self-play fine-tuning can improve model reasoning.

Inference-time methods for improved reasoning. Test-time methods aim at improving reasoning capabilities by either designing prompts that elicit good reasoning behavior or by building reasoning systems which prompt the LLM over and over to arrive at a solution in a systematic way. The most popular strategy for prompting LLMs to reason is chain-of-thought [60], where the prompt includes instructions to 'think step-by-step'. This is a type of test-time approach [61–63], as it typically leads to longer token sequences generated by the LLM. The default prompt (see appendix B) we use to evaluate various LLMs on TPBench is a customized variation of chain-of-though—it includes the tips from Polya's famous manual 'How to solve it' [18] which was originally intended to teach students how to solve mathematical problems. Related advances include prompting the model to break down the problem into simpler subproblems [64–66], or seeking abstractions [67]. Other prompting strategies encourage models to self-verify [68, 69], self-improve [59, 70], or iteratively refine their answer [71, 72].

Other strategies to elicit reasoning behavior involve the generation of multiple reasoning chains which can then be sampled from (as in best-of-n [73]) or combined via majority voting or by ensuring self-consistency [74]. Methods that improve reasoning through planning [66, 75–78] roll out multiple reasoning chains hierarchically and explore the space with Monte-Carlo Tree Search [79]. The success of these methods depends on how the different reasoning chains are evaluated and can be achieved either through other language models [76] or through external tools, e.g. [77]. Tool usage in reasoning is explored next.

Verifiers and tool usage. Another avenue for boosting the performance of LLMs is by allowing tool usage [80, 81] either during the reasoning phase [82], or to verify intermediate reasoning steps [77] and solutions [22]. Verifiers and tools are compatible both with training-time and test-time methods. Since each of the

problems in TPBench has an auto-verifier, one could consider giving the LLM under evaluation access to the auto-verifier to test if, by using it, it can achieve better results.

6. Discussion

We developed the dataset TPBench to test TP reasoning capabilities of AI models. The core of our work is a set of novel uncontaminated problems, to detect true reasoning rather than memorization. However, as we discussed, scientific reasoning is always based on existing works and methods, and there is no sharp transition between true reasoning and memorization (for example of logically equivalent problems, or logically similar problems with minor modifications). It is clear from our benchmark results that reasoning models vastly outperform non-reasoning models, and that these models are capable of some degree of reasoning. We note that our problems were not constructed to match a particular target error rate (o3-mini and DS R1 appeared after most problems were finalized), but rather to reflect real problems encountered by theoretical physicists at each career level. Our TP reasoning results are consistent with studies from more general benchmarks, and illustrate the speed of progress in AI. The most advanced models are able to solve some problems at graduate level, but are not yet capable of solving most research level problems. While advanced models demonstrate remarkable proficiency in algebraic and conceptual problem-solving, they struggle with structured logical reasoning and transparent step-by-step calculations, particularly in complex, research-level problems. Their reliance on literature recall without verification or referencing and their lack of consistency in detailed reasoning remain key limitations in their problem-solving capabilities. We discussed these shortcomings and summarized common failure modes.

Progress has been rapid, even during the creation of this data set. If models could solve level 5 research problems consistently, their impact on TP would be substantial. However, even then, AI models could not perform independent research without further developments. We now discuss some future directions related to our work, that could make LLMs more powerful for TP research.

Updates to the TPBench data set and score board. We will update the score board for novel SOTA models. Results will be published on the website of the data set tpbench.org. The website also contains additional model evaluation metrics, which assign a unified model score over all difficulties. We aim to add more problems to the data set in the future, both public and private problems. It would be particularly interesting to design more research problems which are clearly outside of the training data. This could be achieved by curating research problems specifically from the newest arxiv publications, before the current knowledge cutoff. We invite interested researchers to contribute new problems and collaborate on future TPBench updates (see website for details).

Automatic problem scraping from publication archives. To improve inference methods specifically for TP, for example by reinforcement learning of reasoning chains (e.g. Deepeek R1 [35]), it would be important to have a large collection of verifiable problems. If problems could be extracted automatically from publications, perhaps after a training data cutoff, this would allow generation of training data without human labor at industrial scale. An initial exploration of LLM-based problem extraction from papers has revealed that this is difficult in TP because calculations are often spread over the paper and it is not clear to the model what information is needed to state the problem and what the answer is. This is more obvious in mathematical papers that clearly mark theorems and proofs (e.g. with latex tags), however those are more difficult to auto-verify. Nevertheless, this is an exciting direction for future work, especially since large industry labs keep their training data for reasoning models private.

Automatic verification for non-algebraic expressions. We were somewhat constrained in our choice of problems by the criteria of auto-verifiability. Many TP results can be written in inequivalent ways, and models are not currently good at judging equivalence of expressions. Large collections of verifiable problems are also important for reinforcement learning-based training of reasoning models, see e.g. the recent DS R1 [35]. Generating stronger verifiers that work for a wider class of problems is a very interesting direction for future work, where theoretical and computational physics domain expertise is valuable. Some challenges were listed in section 2.3. We note that results in TP are often symbolic expressions, which are more suited for auto-verification than mathematical proofs (which need to be checked by proof assistants).

Improving reasoning methods for TP. We have reviewed methods to improve reasoning capabilities of LLMs in section 5.2. It is clear from our experiments that a significant gain could be obtained if tools such as `SymPy` or `Mathematica` would be used consistently to check symbolic calculations where this is possible. Few shot learning or fine-tuning could be used to improve models ability to call symbolic software packages.

However, many TP calculations require specific packages and do not come with a lot of training data. Further, the human TP research process involves reading publications, and looking up results or methods when needed. References are also used to spot mistakes in calculations by comparing to known published results where possible. While LLMs can parse literature with techniques such as RAG [83], to our knowledge this has not been demonstrated to lead to performance gains in mathematical reasoning. The fact that models cannot point to a specific source for mathematical statements lowers their trustworthiness. Finally, inference methods that provide more information about uncertainty in individual steps would be particularly beneficial for difficult TP problems. This would pave the way for trustworthy, automated TP research assistants that reliably solve some aspects of a problem, but then ask for help for the parts they are uncertain about.

Diagrammatic and spatial reasoning. Theoretical physicists like to reason using spatial diagrams such as Feynman diagrams or drawing integration contours. In principle, such diagrams can be encoded in some formal language and multi-modality for spatial reasoning may not be necessary. For example, some of our problem solutions include Feynman diagrams or integration contours encoded with the `TikZ` LaTeX library (e.g. figure 4). For some of our problems humans would have trouble reasoning through them without the ability to draw on some scratchpad. It would be interesting to see whether multi-modal language and spatial reasoning models could make models stronger. Visualizing the problem (e.g. 'running an example in your head') is a common strategy and could be particularly powerful for models to develop truly novel ideas.

Training reasoning models on TPBench. While we designed TPBench for the evaluation of the reasoning capabilities of LLMs, it would also be very interesting to curate a dataset for supervised fine-tuning or for reinforcement learning purposes. While we expect fine-tuning to increase the TP specific reasoning capabilities of LLMs, it is equally important to avoid data leakage to avoid problems that are later used for evaluation to seep into the training data. For this reason we choose not to publish all of our problems in TPBench at this time. Instead we encourage researchers who wish to have their models evaluated on TPBench to reach out to us.

Open-ended research problems. If models could solve well-posed problems such as the research problems in our collection reliably, this would speed up TP research projects considerably. However, a large part of research consists of arriving at well-posed problems, which are interesting to answer and can be answered. It could be possible to design more open-ended tasks, where the goal is to 'derive interesting results' based on some set of initial constraints or observations. The AI model could suggest assumptions to include or drop, design its own problem statements, and attempt to judge the importance of its results (develop 'theoretical taste'). It would be exciting and challenging to set up such a more open-ended benchmark.

Community efforts by the TP community. With reasoning models being developed primarily by industry, usually with proprietary and closed data sets, it is important to consider how the open research community can contribute to AI driven TP reasoning. It now seems possible that AI models will be able to do significant theoretical research within a few years. The TP community should work towards the goal that such research remains open and accessible, rather than being performed exclusively at a few select industry labs. While pre-training may be financially inaccessible to publicly funded research, supervised fine-tuning, reinforcement learning, and algorithm development require more moderate resources. As an example, the community could build data sets for both TP reasoning training and benchmarking that are available to both the community and AI labs (with some data leakage control). These could also include examples of tool usage such as Mathematica. A large community-curated data set of verifiable TP problems would in particular allow supervised fine-tuning and Reinforcement Learning specifically for TP. Our data set is a first step in that direction. We hope that this work will contribute to engaging theoretical physicists in this exciting research direction.

Data availability statement

Some of the data we use is available here: https://tpbench.org/. The private data set needs to stay private to avoid leakage into pre-training data of future reasoning models. The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

We thank Kendrick Smith and Matthew Johnson for discussions. M M and D J H C acknowledge the support by the U.S. Department of Energy, Office of Science, Office of High Energy Physics under Award Number DE-SC0017647. M M also acknowledges the support by the National Science Foundation (NSF) under Grant Number 2307109 and the Wisconsin Alumni Research Foundation (WARF). F S is grateful for the support of the NSF under CCF2106707 and the Wisconsin Alumni Research Foundation (WARF).

Appendix A. Summary of problem data

For each problem we collect the following data.

- **Problem title**: Up to one sentence describing the problem.
- **Problem statement**: The problem statement in LaTeX.
- **Problem solution**: The full solution to the problem in LaTeX.
- **Public problem**: yes/no.
- **Auto-verifiable**: yes/no. All problems in the data set for this publication are auto-verifiable.
- **Auto-verifier instructions**: Instructions how to output the solution for the auto-verifier. See section 2.3.
- **Domain of TP**: e.g. High energy theory.
- **Difficulty level**: 1–5
- **Authors**: The contributors of the problem and solution.
- **Reviewers**: The reviewers of the problem and solution.
- **Problem origin and novelty**: How closely existing published work contains the solution (only above undergraduate level).
- **Problem ID**: Unique problem ID in our catalog.
- **Problem version**: In some cases there may be errors or ambiguities in a problem. For this case we track a version number.
- **Variation of a different problem**: In the future, we aim to provide minor modifications of existing problems to check stability of the reasoning chain (as opposed to memorization). Standard: No
- **Date problem was added to the data set**: Allows us to track new problems. Format: 01/31/2025.
- **Author comments**: Any additional comments the author has about the problem.

Appendix B. Prompts

B.1. Prompts to query problem solutions

We used two different system prompts to initialize the LLMs, as well as a unique user prompt to query individual solutions.

Simple system prompt

Our simple system prompt only specifies the required output format and encourages complete calculations.

> **System:** You are a mathematical problem-solving assistant specializing in TP.
> Input problems will be provided in LaTeX format, and you must provide your solutions in LaTeX format as well.
> Please provide detailed step-by-step solutions and clearly mark your final answer with 'Final answer:' at the end.
> When writing equations, ensure proper LaTeX formatting including appropriate equation environments and mathematical notation.

Extended system prompt with CoT advise

Our extended system prompt includes additional problem solving advice inspired by Polya's book 'How to Solve It'. [18]. We have used this system prompt as our default. However, we did not find a systematic difference between these prompts as illustrated in table 6 for a subset of problems.

Table 6. Performance comparison for different system prompts, using the GPT-4o model, on a subset of problems.

	A		*B*		*C*		*D*		avg@5		best@5	
Difficulty level	Ext	Std	Ext	Std	Ext	Std	Ext	Std	Ext	Std	Ext	Std
1-Easy undergrad	20	23	3	1	2	1	0	0	0.72	0.76	0.8	0.8
2-Undergrad	22	17	1	0	2	8	0	0	0.88	0.68	1.0	1.0
3-Easy grad	8	10	3	3	14	12	0	0	0.16	0.24	0.4	0.6

System: You are a mathematical problem-solving assistant specializing in TP.
Input problems will be provided in LaTeX format, and you must provide your solutions in LaTeX format as well.
Please provide detailed step-by-step solutions and clearly mark your final answer with 'Final answer:' at the end.
When writing equations, ensure proper LaTeX formatting including appropriate equation environments and mathematical notation.
Please follow a structured and logical approach. Here are your key steps for solving any problem:
1. Understand the problem:
- Identify the unknown, the given data, and the conditions.
- Evaluate if the conditions are sufficient, redundant, or contradictory.
- Break down and analyze the different parts of the condition.
2. Devise a plan:
- Explore connections between the data and the unknown.
- If necessary, consider auxiliary problems to bridge gaps.
- Reflect on whether you have encountered similar problems or solutions before.
- Look for related problems, theorems, or methods that might apply.
- Consider simplifying or reformulating the problem to make it more accessible.
- Use definitions and explore analogous, general, or special cases.
3. Carry out the plan:
- Execute your solution step by step, ensuring each step is clear and logically valid.
- Confirm the correctness of each step and justify your reasoning.
For each problem, ensure clarity, logical rigor, and consistency. You may iterate to refine and improve your solution.

User prompt

User: Problem:
problem["problem_details"]["Problem Statement"]

IMPORTANT SOLUTION REQUIREMENTS:
1. You MUST FIRST solve this problem using mathematical reasoning and symbolic calculations:
- Use proper mathematical notation and symbols
- Arrive at a final symbolic mathematical expression

2. ONLY AFTER completing the mathematical solution:
- Convert your final mathematical expression into Python code
- The code must satisfy these requirements:
problem["problem_details"]["Answer Requirements"]

Code Format Requirements:
1. Your solution MUST include the final executable Python code as required by the 'Answer Requirements'
2. You MUST wrap the final Python code between ```python and ``` tags
3. Ensure the code is complete and can run independently
4. The code should NOT contain ANY externally defined variables, including physical constants.

B.2. Prompts to query grading of solutions

System prompt

System: You are a grader for machine learning model solutions of TP problems. I will provide you with a correct expert solution to the problem for your reference, and a model solution for you to grade. Grade solutions using *A*/*B*/*C*/*D* grades where:
A = Excellent: The solution is mathematically equivalent to the expert solution, even if the symbolic expression differs (e.g. terms are arranged differently). The solution includes all necessary steps and the reasoning in each step is correct. Different but valid solution methods are acceptable.
B = Good with minor issues: Generally correct solution with small errors such as: arithmetic mistakes that do not affect the main approach, missing intermediate steps, or minor notation issues. The problem was correctly understood and the reasoning of the solution is generally correct.
C = Significant issues but partially correct: Shows basic understanding but has major flaws such as: incorrect application of formulas, missing crucial steps, or computational errors that lead to wrong final answer. The approach has some merit despite errors.
D = Incorrect or major issues: Fundamentally flawed approach, completely incorrect calculations, or missing essential components. Shows little to no understanding of the mathematical concepts involved. When comparing final answers, verify that the equations or expressions are mathematically equivalent (e.g. $2x+2$ is equivalent to $2(x+1)$). Always format your notes using LaTeX notation for mathematical expressions. Provide evaluation in compact JSON format with only 'grade' and 'notes' fields. Format all mathematical expressions in your notes using LaTeX notation (e.g. x^2, $ \frac{1}{2}$, $\sqrt{x}$).

User prompt

User: Compare the following model solution detailed steps with the expert solution, along with the code verification result which check the equivalence of 2 expression numerically, and evaluate its correctness.
Expert Solution: {expert_solution}
Model Solution {model_solution}
Code Verification result: {code_verification_result}
Format your response as JSON with the following structure:
{
"grade": "*A*/*B*/*C*/*D*",
"notes": "your notes here with LaTeX math notation"
}

Appendix C. Public problems and solutions

We list ten public sample problems along with their solutions. AI model results for these problems are available on the dataset website tpbench.org. Table 7 summarizes the performance of different AI models on these problems, covering a range of topics and difficulty levels from Level 1 (L1) to Level 5 (L5). The scores indicate the average accuracy of the 5 attempts of each model.

C.1. Level 5—one-pole problem

Problem statement

Consider the conformally coupled scalar field ϕ

$$\mathcal{L} = \frac{1}{2}\left[g^{\mu\nu}\partial_\mu\phi\partial_\nu\phi - \left(m^2 - \frac{1}{6}R\right)\phi^2\right] \tag{3}$$

in curved spacetime

$$\mathrm{d}s^2 = a^2(\eta)\left(\mathrm{d}\eta^2 - |\mathrm{d}\vec{x}|^2\right)$$

where the Ricci scalar is

$$R = -6\frac{a''(\eta)}{a(\eta)} \tag{4}$$

Table 7. Model average scores by problem.

Problem ID	Llama-70B	GPT-4o	R1	o1	o3-mini
Boosted parabolic trajectory (L1)	0.60	1.00	1.00	1.00	1.00
Blackbody in d dimensions (L1)	0.20	0.40	1.00	1.00	1.00
A 3-State QM Problem (L2)	0.40	0.80	1.00	1.00	1.00
Dark matter capture as a function of time (L2)	0.60	1.00	1.00	1.00	1.00
Slow-roll inflation (L3)	0.00	0.00	1.00	1.00	1.00
Scalar particle scattering (L3)	0.00	0.40	0.80	0.40	0.40
SHO vacuum entanglement (L4)	0.00	0.00	0.80	0.00	1.00
SUSY-symmetry (L4)	0.00	0.00	0.00	0.00	0.00
Bias of a sampled halo field (L5)	0.00	0.00	0.60	1.00	0.80
One-pole problem (L5)	0.00	0.00	0.00	0.00	0.00

and a satisfies the differential equation

$$\frac{\mathrm{d}}{\mathrm{d}t}\ln a = \Theta\left(t_e - t\right) H_I + \Theta\left(t - t_e\right) \frac{H_I}{1 + \frac{3}{2} H_I \left(t - t_e\right)} \tag{5}$$

with t_e a finite positive number, the Θ function having the steplike behavior

$$\Theta\left(t - t_e\right) \equiv \begin{cases} 1 & t \geqslant t_e \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

and t being the comoving proper time related to η through

$$t = t_e + \int_{\eta_e}^{\eta} a\left(y\right) \mathrm{d}y. \tag{7}$$

The boundary condition for the differential equation (in comoving proper time) is $a|_{t=t_e} = a_e$.

In the limit that $k/(a_e H_I) \to \infty$, using the steepest descent approximation starting from the dominant pole $\tilde{\eta}$ (with $\Re\tilde{\eta} > 0$) of the integrand factor $\omega_k'(\eta)/\left(2\omega_k(\eta)\right)$, compute the Bogoliubov coefficient magnitude $|\beta(k)|$ approximated as

$$\left|\beta\left(k\right)\right| \approx \left| \int_{-\infty}^{\infty} \mathrm{d}\eta \frac{\omega_k'\left(\eta\right)}{2\omega_k\left(\eta\right)} \mathrm{e}^{-2i\int_{\eta_e}^{\eta} d\eta' \omega_k\left(\eta'\right)} \right| \tag{8}$$

for particle production where the dispersion relationship given by

$$\omega_k^2\left(\eta\right) = k^2 + m^2 a^2\left(\eta\right) \tag{9}$$

with $0 < m \lesssim H_I$. Use a one pole approximation which dominates in this limit.

Answer requirements

Provide the answer in the form of the `verbatim` code. Implement the following function.

```
def abs_beta(k:float, a_e:float, m:float, H_I:float) -> float:
    pass
```

Comments about the problem

This is an example of a difficult problem from QFT in curved spacetime, dealing with gravitational particle production, that appears out of reach of current models. This is part of a published research work and the solution, without steps explained, is given in a footnote of [30], but would be difficult to locate (in fact we tried, without success, with OpenAI's Deep Research).

Solution

To find the pole of $\omega_k'(\eta)/\omega_k(\eta)$, we need $a(\eta)$ from the given differential equation

$$\frac{\mathrm{d}\ln a}{\mathrm{d}t} = \Theta\left(t_e - t\right) H_I + \Theta\left(t - t_e\right) \frac{H_I}{1 + \frac{3}{2} H_I \left(t - t_e\right)}. \tag{10}$$

Integrating from time $t = t_e$, we find

$$\ln \frac{a}{a_e} = \int_{t_e}^{t} \mathrm{d}T \frac{H_I}{1 + \frac{3}{2} H_I (T - t_e)} \tag{11}$$

$$= \frac{2}{3} \ln \left[1 + \frac{3}{2} H_I (T - t_e) \right]_{t_e}^{t} \tag{12}$$

$$= \frac{2}{3} \ln \left[1 + \frac{3}{2} H_I (t - t_e) \right] \tag{13}$$

for $t \geqslant t_e$. In other words, this scale factor

$$\frac{a}{a_e} = \left[1 + \frac{3}{2} H_I (t - t_e) \right]^{2/3} \tag{14}$$

behaves as a typical coherent oscillations spacetime minus the oscillatory effects. Hence, note that for $t \gg t_e$, the scale factor can be approximated as

$$a(\eta) \approx c_1 \eta^2 \tag{15}$$

for $\eta \gg \eta_e$ (where η_e is the corresponding conformal time for t_e) where we see by matching

$$\int_{\eta_i}^{\eta} a(\eta) \, \mathrm{d}\eta = t - t_i \tag{16}$$

with $\eta_i \gg \eta_e$ and $t_i \gg t_e$, we can write

$$\frac{1}{3} c_1 \eta^3 \approx t \tag{17}$$

for times much larger than η_i. This means that at time $\eta_i \gg \eta_e$, we have

$$c_1 \approx \frac{2}{H(\eta_i) \eta_i^3} \tag{18}$$

(where the Hubble expansion rate is $H(\eta) = a'(\eta)/a^2(\eta)$) which gives

$$a(\eta) \approx \frac{2\eta^2}{H(\eta_i) \eta_i^3} \tag{19}$$

for $\eta > \eta_i$ where the choice of η_i controls the approximation error proportional to positive power of η_e/η_i. Since $\eta_i \gg \eta_e > 0$, we can approximate $\eta = 0$ to be equivalent to $\eta - \eta_i \to -\infty$. In other words, when we analytically continue and consider the poles of the integrand, we will consider only the region with $\Re \eta > 0$.

Next, note the pole of

$$\frac{\omega'}{2\omega} = \frac{m^2 \partial_\eta a^2}{4(k^2 + m^2 a^2)} \tag{20}$$

is at $\tilde{\eta}$ defined by

$$k^2 = -m^2 a^2(\tilde{\eta}) \tag{21}$$

which means

$$\tilde{\eta} = \sqrt{\frac{H(\eta_i) \eta_i^3}{2}} \left(\frac{-k^2}{m^2} \right)^{1/4}$$

$$= \eta_i \sqrt{\frac{1}{a(\eta_i)}} \left(\frac{-k^2}{m^2} \right)^{1/4}$$

$$= \eta_i \mathrm{e}^{\mathrm{i}(2l+1)\pi/4} \frac{\sqrt{k/a(\eta_i)}}{\sqrt{m}} \tag{22}$$

where l is an integer. We see that $\Re \tilde{\eta} \gg \eta_i$ for $k/a(\eta_e) \gg k/a(\eta_i) \gg m$. We also see that $l \in \{1, 2\}$ have negative $\Re \tilde{\eta}$ which are in the region that we excised with the $\eta - \eta_i \to -\infty$ discussed above. That means we can consider either $l \in \{3, 4\}$. We will see below that one of these poles is irrelevant.

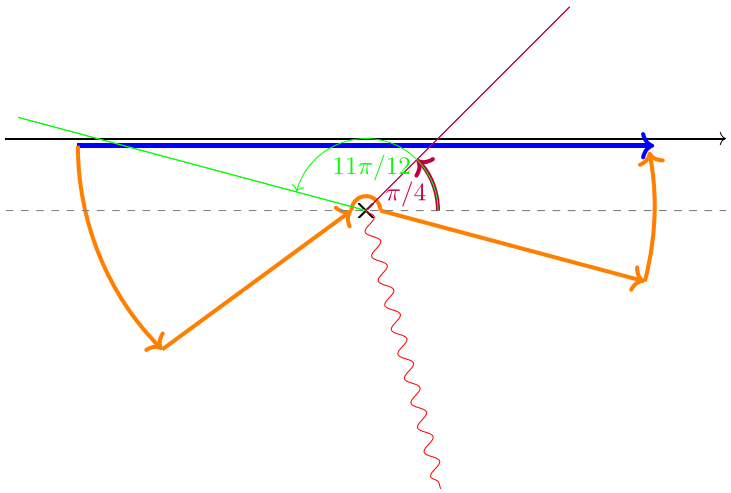


Figure 4. The original contour in blue is deformed into the orange contour in the lower half complex plane of η. The large radius arcs have vanishing contributions, and one-pole approximation has been taken. The upper green and purple boundaries correspond to where integrations over any arcs extended beyond this boundary would not converge. The dashed horizontal curve is parallel to the real axis. The red squiggly line is the branch cut at $-5\pi/12$.

Equation (8) tells us that

$$\begin{aligned}|\beta(\eta)| &= \left|\int_{-\infty}^{\infty} \mathrm{d}\eta \frac{\omega_k'(\eta)}{2\omega_k(\eta)} \mathrm{e}^{-2i\int_{\eta_e}^{\eta} \mathrm{d}\eta' \omega_k(\eta')}\right| \\ &= \left|\int_{-\infty}^{\infty} \mathrm{d}\eta \frac{\omega_k'(\eta)}{2\omega_k(\eta)} \mathrm{e}^{-2i\int_{\eta_i}^{\eta} \mathrm{d}\eta' \omega_k(\eta')} \mathrm{e}^{-2i\int_{\eta_e}^{\eta_i} \mathrm{d}\eta' \omega_k(\eta')}\right| \\ &= \left|\int_{-\infty}^{\infty} \mathrm{d}\eta \frac{\omega_k'(\eta)}{2\omega_k(\eta)} \mathrm{e}^{-2i\int_{\eta_i}^{\eta} \mathrm{d}\eta' \omega_k(\eta')}\right|. \end{aligned} \tag{23}$$

With the steepest descent technique starting from the pole of ω_k'/ω_k, we write after analytically continuing η

$$\begin{aligned}|\beta| &= \left|\int_{-\infty}^{\infty} \mathrm{d}\eta \frac{\omega_k'(\eta)}{2\omega_k(\eta)} \mathrm{e}^{-2i\left[\int_{\eta_i}^{\tilde{\eta}} \mathrm{d}\eta' \omega_k(\eta') + \int_{\tilde{\eta}}^{\eta} \mathrm{d}\eta' \omega_k(\eta')\right]}\right| \\ &= \left|\mathrm{e}^{-2i\int_{\eta_i}^{\tilde{\eta}} \mathrm{d}\eta' \omega_k(\eta')} \nu\right| \end{aligned} \tag{24}$$

where $\tilde{\eta}$ is the pole of $\omega_k'(\eta)/\omega_k(\eta)$ and ν is the part obtained from the steepest descent. The factor in the integrand of equation (8) is therefore

$$\frac{\omega'}{2\omega} \approx \frac{1}{4(\eta-\tilde{\eta})} \tag{25}$$

which implies ν in equation (24) is

$$\nu = \int_{-\infty}^{\infty} \frac{\mathrm{d}\eta}{4(\eta-\tilde{\eta})} \mathrm{e}^{-\frac{4}{3}im\sqrt{C'(\tilde{\eta})}(\eta-\tilde{\eta})^{3/2}} \tag{26}$$

where

$$C(\eta) \equiv a^2(\eta). \tag{27}$$

Deforming the integration contour as shown in figure 4 allows us to rewrite this as

$$\nu = \int_{\mathcal{C}} \frac{\mathrm{d}\eta}{4(\eta-\tilde{\eta})} \mathrm{e}^{-\frac{4}{3}im\sqrt{C'(\tilde{\eta})}(\eta-\tilde{\eta})^{3/2}} \tag{28}$$

where the $\mathcal{C}$ is the orange part of the contour in the lower half plane.

To define the contour, one must understand the complex values of $C'(\tilde{\eta})$. To this end, let

$$-\mathrm{i}\sqrt{C'(\tilde{\eta})} = U + \mathrm{i}W \tag{29}$$

where the imaginary part generically is nonvanishing. The branch points are given by equations (22) which gives

$$C'(\tilde{\eta}) = \frac{4a^2(\eta_i)}{\eta_i} e^{\frac{3}{4}i(2l+1)\pi} \left(\frac{k/a(\eta_i)}{m}\right)^{3/2} \tag{30}$$

which says

$$\begin{aligned} U + iW &= \frac{2a(\eta_i)}{\sqrt{\eta_i}} e^{\frac{1}{8}i(6l-1)\pi} \left(\frac{k/a(\eta_i)}{m}\right)^{3/4} \\ &= a^{3/2}(\eta_i)\sqrt{2H(\eta_i)} e^{\frac{1}{8}i(6l-1)\pi} \left(\frac{k/a(\eta_i)}{m}\right)^{3/4}. \end{aligned} \tag{31}$$

To deform the contour, we need regions where the arcs with large radius does not contribute to the integral. Note that if we define $\delta \equiv \eta - \tilde{\eta} = Re^{i\theta}$, we have

$$\delta^{3/2} = R^{3/2} e^{i3\theta/2} = R^{3/2}\left(\cos\frac{3\theta}{2} + i\sin\frac{3\theta}{2}\right) \tag{32}$$

making the exponent in v

$$-\frac{4}{3} im\sqrt{C'(\tilde{\eta})}(\eta - \tilde{\eta})^{3/2} = \frac{4}{3} mR^{3/2}(U + iW)\left(\cos\frac{3\theta}{2} + i\sin\frac{3\theta}{2}\right) \tag{33}$$

which is damped only if

$$U\cos(3\theta/2) - W\sin(3\theta/2) < 0. \tag{34}$$

For the case of equation (21), we need

$$\cos\left[\frac{\pi}{8}(6l-1)\right]\cos(3\theta/2) - \sin\left[\frac{\pi}{8}(6l-1)\right]\sin(3\theta/2) < 0 \tag{35}$$

for one choice of l. For the choice of $l = 3$, we can choose the arc regions to be $\theta \in [\frac{-5\pi}{12}, \frac{\pi}{4}]$ and another arc region to be $\theta \in [\frac{11\pi}{12}, \frac{19\pi}{12}]$ with a branch cut at $-5\pi/12$.

Choosing $l = 3$, we find the steepest descent contour shown in orange in figure 4. The left contour is $5\pi/4$ and the right contour is at $-\pi/12$, along which

$$-\frac{4}{3} im\sqrt{C'(\tilde{\eta})}(\eta - \tilde{\eta})^{3/2} = -\frac{4}{3} mR^{3/2} a^{3/2}(\eta_i)\sqrt{2H(\eta_i)}\left(\frac{k/a(\eta_i)}{m}\right)^{3/4}$$

gives a damped exponential in equation (26). Hence, the integral is

$$\begin{aligned} v = &\frac{1}{4}\int_{\infty}^{\epsilon} \frac{dR}{R} e^{-\frac{4}{3}mR^{3/2}a^{3/2}(\eta_i)\sqrt{2H_e}\left(\frac{k/a(\eta_i)}{m}\right)^{3/4}} + \frac{1}{4}\int_{\epsilon}^{\infty} \frac{dR}{R} e^{-\frac{4}{3}mR^{3/2}a^{3/2}(\eta_i)\sqrt{2H_e}\left(\frac{k/a(\eta_i)}{m}\right)^{3/4}} \\ &+ \frac{1}{4}\int_{5\pi/4}^{-\pi/12} id\theta \exp\left[-\frac{4}{3} im\sqrt{C'(\tilde{\eta})}\left(\epsilon e^{i\theta}\right)^{3/2}\right] \\ = &\frac{i}{4}\left[\frac{-\pi}{12} - \frac{15\pi}{12}\right] = \frac{-i\pi}{3} \end{aligned} \tag{36}$$

where in the first line we have introduced a regulator $\epsilon \to 0$.

The final piece in equation (24) is

$$I = e^{-2i\int_{\eta_i}^{\tilde{\eta}} d\eta' \omega_k(\eta')}. \tag{37}$$

Use the expansion

$$\begin{aligned} I &= e^{-2i\int_{\eta_i}^{\tilde{\eta}} d\eta' \omega_k(\eta')} \\ &= \exp(-2i[\Phi + J]) \end{aligned} \tag{38}$$

where Φ is real and J is purely imaginary. We take the path to be along the real axis until $\eta = \Re\tilde{\eta}$ and then integrate in the imaginary η direction:

$$J = \mathrm{i}\Im \int_{\Re\tilde{\eta}}^{\Re\tilde{\eta}+\mathrm{i}\Im\tilde{\eta}} \mathrm{d}\eta' \omega_k(\eta'). \tag{39}$$

This gives

$$J \approx -i\frac{2}{3}\sqrt{2\pi}\frac{\Gamma(5/4)}{\Gamma(3/4)}\frac{(k/a(\eta_i))^{3/2}}{H(\eta_i)\sqrt{m}}. \tag{40}$$

Now, note from equation (14), we can compute

$$\begin{aligned}\frac{1}{a_e^{3/2}} &= \frac{1}{a^{3/2}(\eta_i)}\left[1+\frac{3}{2}H_I(t_i-t_e)\right] \\ &\approx \frac{1}{a^{3/2}(\eta_i)}\frac{3}{2}H_I t_i \\ &\approx \frac{1}{a^{3/2}(\eta_i)}\frac{H_I}{H(\eta_i)}\end{aligned} \tag{41}$$

where we used equation (10). Equation (24) then becomes

$$\boxed{|\beta| \approx \frac{\pi}{3}\exp\left(-\frac{4}{3}\sqrt{2\pi}\frac{\Gamma(5/4)}{\Gamma(3/4)}\frac{(k/a_e)^{3/2}}{H_I\sqrt{m}}\right)}. \tag{42}$$

C.2. Level 5—bias of a sampled halo field

Problem statement

In cosmology, large-scale cosmological dark-matter halo fields are biased tracers of the underlying Gaussian matter density δ_m. Assume we have a sample δ_m. We simulate a halo number density field by taking $n(\mathbf{x}) = \bar{n}\max(0, 1 + b\delta_m(\mathbf{x}))$, where bare number density $\bar{n}$ and bare bias b are specified constants. What is the bias of the sampled halo field? Derive an equation to evaluate the bias which depends on the bare bias and the variance in each pixel.

Answer requirements

Provide the answer in the form of the `verbatim` code. Implement the following function.

```
#let b_in stand for bare bias
def b_eff(sigma: float, b_in:float) -> float:
    pass
```

Comments about the problem

This is an example of a cosmology research problem that is being solved correctly by advanced reasoning models. This may be because the calculation is similar to existing calculations in the literature. However, this is a genuine research problem, which we solved independently, for an upcoming cosmology publication. The problem requires to retrieve some background knowledge, such as the definition of the matter power spectrum in cosmology.

Solution

The solution to this question involves some domain knowledge, parts of which were given in the problem's statement, some approximations sourced by the domain knowledge, and some mathematical calculations. The domain knowledge is very basic and should be known to anyone in the field. Approximations are intuitive and also, mostly, inspired by the domain knowledge. Following Polya, we can organize it as follows:

Understand the problem. The number density of halos $n_h(\mathbf{x})$ is defined as

$$N_h = \int_V n_h(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{43}$$

The overdensity is defined as

$$\delta_h(\mathbf{x}) = \frac{n_h(\mathbf{x}) - \langle n_h(\mathbf{x})\rangle}{\langle n_h(\mathbf{x})\rangle}. \tag{44}$$

Linear bias is defined in terms of Fourier-transformed quantities:

$$\delta_h(\mathbf{k}) = b\delta_m(\mathbf{k}). \tag{45}$$

This is an approximation that holds on sufficiently large scales (small k). $\delta_m(\mathbf{k})$ and $\delta_h(\mathbf{k})$ are Gaussian random fields with zero mean and their variance depends only on the magnitude of the wave-vector $k = |\mathbf{k}|$:

$$\delta_m \sim \mathcal{N}(0, P_{mm}(k)),\ \delta_h \sim \mathcal{N}(0, P_{hh}(k)). \tag{46}$$

The quantity $P(k)$ is called the power spectrum and is defined as

$$\langle \delta(\mathbf{k})\, \delta(\mathbf{k'}) \rangle = (2\pi)^3 \delta^D(\mathbf{k} + \mathbf{k'})\, P(k). \tag{47}$$

It immediately follows that

$$P_{hh}(k) = b^2 P_{mm}(k). \tag{48}$$

We are given the expression in real space. In real space, the quantity $\delta_m(\mathbf{x})$ is also a Gaussian random field:

$$\delta_m(\mathbf{x}) \sim \mathcal{N}(0, \xi_m),\ \delta_h(\mathbf{x}) \sim \mathcal{N}(0, \xi_h). \tag{49}$$

Quantity ξ is called a two-point (real-space) correlation function and is defined as

$$\langle \delta(\mathbf{x})\, \delta(\mathbf{x'}) \rangle = \xi(|\mathbf{x} - \mathbf{x'}|). \tag{50}$$

This quantity is sufficiently small when $|\mathbf{x} - \mathbf{x'}| \gg 1$. We are asked to find what is the expression for b in the equation $\delta_h(k) = b\delta_m(\mathbf{k})$, given the real-space expression for the number density $n_h(\mathbf{x})$ in terms of real-space sample of $\delta_m(\mathbf{x})$.

Devise a plan. The key point to solve this problem should be that real-space correlation function for halos ξ_h should also be equal to $b^2\xi_m$. We want to calculate that correlation function. It should be expressed in terms of $\langle n(\mathbf{x}) \rangle$ and $\langle n_h(\mathbf{x}) n_h(\mathbf{x'}) \rangle$. We expect to be able to calculate these expectations since they are the expectations of functions of the Gaussian random variables. We are given the pixel variance σ. How does it connect to the other quantities we know? In principle, that's also the part of domain knowledge but it also can be deducted from the definitions already given. A discretized version of the correlation function is

$$\xi_{ij} = \langle \delta_{\mathbf{x_i}} \delta_{\mathbf{x_j}} \rangle. \tag{51}$$

When $i = j$, it becomes the pixel variance σ. *Aside, we could have given instead of* σ, *the quantity* $P_{mm}(k)$, *that is a common description of a cosmological dark-matter field. In that case, from the definitions of* $\xi(r)$ and $P_{mm}(k)$, *we could have deduced that* $\sigma = \frac{1}{V}\sum_k P_{mm}(k)$. Then we pick the ensemble of all the pixels at given fixed large distance $r = |\mathbf{x_i} - \mathbf{x_j}|$. The key is to recognize that it is fully described by a correlated bivariate Gaussian distribution.

$$\left(\delta_i^m, \delta_j^m\right) \sim \mathcal{N}(0, \Sigma) \tag{52}$$

with a covariance

$$\Sigma = \begin{pmatrix} \sigma^2 & \xi_r^m \\ \xi_r^m & \sigma^2 \end{pmatrix}. \tag{53}$$

In general, the integrals from the expectation values are cumbersome, but we should expect some simplifications from the fact that ξ is small and we can Taylor-expand the pdf.

Carry out the plan. It's more convenient to define $\hat{\delta}_i = \delta_i^m/\sigma$ and $\hat{\xi} = \xi_r^m/\sigma^2$, and ϕ_2—a correlated bivariate Gaussian pdf—then

$$\left(\hat{\delta}_i, \hat{\delta}_j\right) \sim \frac{\mathrm{e}^{-\frac{1}{2(1-\hat{\xi}^2)}\left[\hat{\delta}_i^2 + \hat{\delta}_j^2 - 2\hat{\xi}\hat{\delta}_i\hat{\delta}_j\right]}}{2\pi\sqrt{1-\hat{\xi}^2}} \equiv \phi_2\left(\hat{\delta}_i, \hat{\delta}_j | \hat{\xi}\right). \tag{54}$$

We note that

$$\xi_r^n = \frac{\langle n_i n_j \rangle}{\langle n \rangle^2} - 1. \tag{55}$$

The quantity $\langle n\rangle$ is the actual mean number density:

$$\bar{n}^{'} = \langle n\rangle = \langle n_i\rangle = \int n^{\rm loc}\left(\delta_i, b, \bar{n}\right)\phi_2\left(\hat{\delta}_i, \hat{\delta}_j|\hat{\xi}\right)\mathrm{d}\hat{\delta}_i\mathrm{d}\hat{\delta}_j = \int n_i^{\rm loc}\,\phi_1\left(\hat{\delta}_i\right)\mathrm{d}\hat{\delta}_i.$$

Here, ϕ_1—is a standard normal pdf. It is expected that it is not dependent on the correlation $\hat{\xi}$, but only on b and σ, just as the marginal of 2D correlated Gaussian distribution is 1D Gaussian that's not dependent on the cross-correlation. To the linear order in $\hat{\xi}$,

$$\phi_2\left(x, y|\hat{\xi}\right) \approx \phi_1\left(x\right)\phi_1\left(y\right)\left(1+\hat{\xi}xy\right). \tag{56}$$

So that the two-point function neatly factorizes:

$$\begin{aligned}\langle n_i n_j\rangle &= \int n^{\rm loc}\left(\delta_i, b, \bar{n}\right) n^{\rm loc}\left(\delta_j, b, \bar{n}\right)\phi_2\left(\hat{\delta}_i, \hat{\delta}_j|\hat{\xi}\right)\mathrm{d}\hat{\delta}_i\mathrm{d}\hat{\delta}_j\\ &\approx \int n_i^{\rm loc}\,\phi_1\left(\hat{\delta}_i\right)\mathrm{d}\hat{\delta}_i\int n_j^{\rm loc}\phi_1\left(\hat{\delta}_j\right)\mathrm{d}\hat{\delta}_j + \hat{\xi}\int n_i^{\rm loc}\,\phi_1\left(\hat{\delta}_i\right)\hat{\delta}_i\mathrm{d}\hat{\delta}_i\int n_j^{\rm loc}\phi_1\left(\hat{\delta}_j\right)\hat{\delta}_j\mathrm{d}\hat{\delta}_j\\ &\equiv \langle n\rangle^2 + \hat{\xi}\langle n\hat{\delta}\rangle^2.\end{aligned} \tag{57}$$

Substituting the results for $\langle n\rangle$ and $\langle n_i n_j\rangle$ in the equation for ξ_r^n, we can read off the bias:

$$b^{'2} = \frac{\xi_r^n}{\sigma^2\hat{\xi}} = \frac{\langle n\hat{\delta}\rangle^2}{\sigma^2\langle n\rangle^2}. \tag{58}$$

All that is left is to calculate the expectations. One can evaluate for $b \geqslant 0$

$$\begin{aligned}\langle n\rangle &= \int n_i^{\rm loc}\,\phi_1\left(\hat{\delta}_i\right)\mathrm{d}\hat{\delta}_i = \int \bar{n}\max\left(0, 1+b\sigma x\right)\phi_1\left(x\right)\mathrm{d}x\\ &= \bar{n}\int_{-\frac{1}{b\sigma}}^{+\infty}\left(1+b\sigma x\right)\phi_1\left(x\right)\mathrm{d}x = \bar{n}\left[\Phi_1\left(\frac{1}{b\sigma}\right) + b\sigma\phi_1\left(\frac{1}{b\sigma}\right)\right].\end{aligned} \tag{59}$$

For $b<0$ it is, however,

$$\begin{aligned}\langle n\rangle &= \bar{n}\int_{-\infty}^{+\frac{1}{|b|\sigma}}\left(1-|b|\sigma x\right)\phi_1\left(x\right)\mathrm{d}x\\ &= \bar{n}\left[\Phi_1\left(\frac{1}{|b|\sigma}\right) + |b|\sigma\phi_1\left(\frac{1}{|b|\sigma}\right)\right].\end{aligned} \tag{60}$$

So we conclude that the latter expression is valid for all b. Similarly, one can show that

$$\langle n\hat{\delta}\rangle = \bar{n}\int \max\left(0, 1+b\sigma x\right)x\phi_1\left(x\right)\mathrm{d}x = \bar{n}b\sigma\Phi_1\left(\frac{1}{|b|\sigma}\right) \tag{61}$$

where $\Phi_1(x) = \int_{-\infty}^{x}\phi_1(x)\mathrm{d}x$—normal cdf. Finally, one can get

$$\boxed{b^{'} = \frac{b\Phi_1\left(\frac{1}{|b|\sigma}\right)}{\Phi_1\left(\frac{1}{|b|\sigma}\right) + |b|\sigma\phi_1\left(\frac{1}{|b|\sigma}\right)}}. \tag{62}$$

Note: We also accept solutions as correct if they omit the | | around the bias, since halo bias is usually positive.

C.3. Level 4—SHO vacuum entanglement

Problem statement

Consider a coupled simple harmonic oscillator governed by the Hamiltonian

$$H = \sum_{i=1}^{2}\frac{1}{2}\left(\frac{p_i^2}{m} + kx_i^2\right) + g\frac{\left(x_1-x_2\right)^2}{2}. \tag{63}$$

If the ground state is $|\Omega\rangle$ and the operator $\hat{\rho}$ is the vacuum density matrix partially traced over the $|w\rangle_{x_2}$ components (satisfying $\hat{x}_2|w\rangle_{x_2} = w|w\rangle_{x_2}$), i.e.

$$\hat{\rho} \equiv \int \mathrm{d}x_1'' \int \mathrm{d}x_1' \int \mathrm{d}w \left(|x_1''\rangle_{x_1\, x_1}\langle x_1''| \otimes {}_{x_2}\langle w|\right)\left(|\Omega\rangle\langle\Omega|\right)\left(|x_1'\rangle_{x_1} \otimes |w\rangle_{x_2\, x_1}\langle x_1'|\right) \tag{64}$$

which is an operator acting on a reduced Hilbert space, compute

$$S \equiv -\mathrm{Tr}_{x_1}\left[\hat{\rho}\ln\hat{\rho}\right] \tag{65}$$

which involves the trace over x_1 states.

Answer requirements
Provide the answer in the form of the `verbatim` code. Implement the following function

```
def entropy(k:float,g:float,m:float)->float:
    pass
```

Comments about the problem
This problem, whose solution can be found in [84] (with less detailed reasoning steps), has been rephrased in a pedagogical manner for graduate-level physics courses. It is a well-known question in quantum entanglement research, and the best performing LLMs are capable of solving it accurately, perhaps at least partially due to memorization.

Solution
Diagonalize the original Hamiltonian

$$H = \begin{pmatrix} x_1 & x_2 & p_1 & p_2 \end{pmatrix} \begin{pmatrix} \frac{k+g}{2} & -\frac{g}{2} & & \\ -\frac{g}{2} & \frac{k+g}{2} & & \\ & & \frac{1}{2m} & \\ & & & \frac{1}{2m} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ p_1 \\ p_2 \end{pmatrix}. \tag{66}$$

One easily finds

$$x_1 = \frac{y_1 + y_2}{\sqrt{2}} \tag{67}$$

$$x_2 = \frac{y_1 - y_2}{\sqrt{2}} \tag{68}$$

diagonalizes the Hamiltonian such that in the $(y_1, y_2, q_1 \equiv m\dot{y}_1, q_2 \equiv m\dot{y}_2)$ basis, it is

$$H = \begin{pmatrix} y_1 & y_2 & q_1 & q_2 \end{pmatrix} \begin{pmatrix} \frac{k}{2} & 0 & & \\ 0 & \frac{k}{2}+g & & \\ & & \frac{1}{2m} & \\ & & & \frac{1}{2m} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ q_1 \\ q_2 \end{pmatrix}. \tag{69}$$

The ladder operators are

$$a_j = \frac{1}{\sqrt{2}}\left(\sqrt{m\omega_j}y_j + \frac{i}{\sqrt{m\omega_j}}q_j\right) \tag{70}$$

$$\omega_1^2 = \frac{k}{m} \quad \omega_2^2 = \frac{k+2g}{m} \tag{71}$$

which allows one to rewrite the Hamiltonian as

$$H = \sum_{j=1}^{2} a_j^\dagger a_j \omega_j + \frac{\omega_1 + \omega_2}{2}. \tag{72}$$

In this basis, we denote the ground state as

$$a_1|00\rangle_{\vec{n}_y} = 0 = a_2|00\rangle_{\vec{n}_y}. \tag{73}$$

Hence we have found $|\Omega\rangle = |00\rangle_{\vec{n}_y}$. We know that the wave function in the $\vec{y}$ coordinates is the product of well known simple harmonic oscillator solutions:

$$\langle y_1', y_2'|00\rangle_{\vec{n}_y} = \frac{1}{(\pi b_1^2)^{1/4}} \exp\left[\frac{-(y_1')^2}{2b_1^2}\right] \frac{1}{(\pi b_2^2)^{1/4}} \exp\left[\frac{-(y_2')^2}{2b_2^2}\right] \tag{74}$$

where

$$b_n \equiv \frac{1}{\sqrt{m\omega_n}} \tag{75}$$

making this a convenient basis to work with. Note

$$\begin{aligned}
\hat{y}_1\left(|a\rangle_{x_1} \otimes |b\rangle_{x_2}\right) &= \int \mathrm{d}y_1'\mathrm{d}y_2'\hat{y}_1|y_1'y_2'\rangle\langle y_1'y_2'|\left(|a\rangle_{x_1} \otimes |b\rangle_{x_2}\right) \\
&= \int \mathrm{d}y_1'\mathrm{d}y_2'y_1'|y_1'y_2'\rangle\langle y_1'y_2'|\left(|a\rangle_{x_1} \otimes |b\rangle_{x_2}\right) \\
&= \int \mathrm{d}x_1'\mathrm{d}x_2'y_1'\left(|x_1'\rangle_{x_1} \otimes |x_2'\rangle_{x_2}\right)\left({}_{x_1}\langle x_1'| \otimes {}_{x_2}\langle x_2'|\right)\left(|a\rangle_{x_1} \otimes |b\rangle_{x_2}\right) \\
&= \frac{a+b}{\sqrt{2}}\left(|a\rangle_{x_1} \otimes |b\rangle_{x_2}\right)
\end{aligned} \tag{76}$$

where we used the completeness of the basis, equations (67) and (68), and the usual delta function normalization of the position basis. This and a similar relation for $\hat{y}_2$ imply

$$|a\rangle_{x_1} \otimes |b\rangle_{x_2} = \left|\frac{a+b}{\sqrt{2}}, \frac{a-b}{\sqrt{2}}\right\rangle. \tag{77}$$

This means

$$\begin{aligned}
{}_{\vec{n}_y}\langle 00|\left(|x_1'\rangle_{x_1} \otimes |w\rangle_{x_2}\right) &= {}_{\vec{n}_y}\left\langle 00\left|\frac{x_1'+w}{\sqrt{2}}, \frac{x_1'-w}{\sqrt{2}}\right.\right\rangle \\
&= \frac{1}{(\pi b_1^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1'+w}{\sqrt{2}}\right)^2}{2b_1^2}\right] \frac{1}{(\pi b_2^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1'-w}{\sqrt{2}}\right)^2}{2b_2^2}\right].
\end{aligned} \tag{78}$$

The partial trace is defined through the following contraction of $(2,2)$ tensor to a $(1,1)$ tensor:

$$\begin{aligned}
\hat{\rho} &= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' \int \mathrm{d}w\left(|x_1''\rangle_{x_1}\,{}_{x_1}\langle x_1''| \otimes {}_{x_2}\langle w|\right)\left(|00\rangle_{\vec{n}_y}\,{}_{\vec{n}_y}\langle 00|\right)\left(|x_1'\rangle_{x_1} \otimes |w\rangle_{x_2}\,{}_{x_1}\langle x_1'|\right) \\
&= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' \int \mathrm{d}w|x_1''\rangle_{x_1}\,{}_{x_1}\langle x_1'| \frac{1}{(\pi b_1^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1''+w}{\sqrt{2}}\right)^2}{2b_1^2}\right] \frac{1}{(\pi b_2^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1''-w}{\sqrt{2}}\right)^2}{2b_2^2}\right] \\
&\quad \times \frac{1}{(\pi b_1^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1'+w}{\sqrt{2}}\right)^2}{2b_1^2}\right] \frac{1}{(\pi b_2^2)^{1/4}} \exp\left[\frac{-\left(\frac{x_1'-w}{\sqrt{2}}\right)^2}{2b_2^2}\right].
\end{aligned} \tag{79}$$

Integrate over w, we find

$$\begin{aligned}\hat{\rho} &= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' |x_1''\rangle_{x_1\,x_1} \langle x_1'| \frac{1}{(\pi b_1^2)^{1/2}} \frac{1}{(\pi b_2^2)^{1/2}} \exp\left[-\frac{m}{4}(\omega_1+\omega_2)\left([x_1']^2+[x_1'']^2\right)\right] \\ &\quad \times \frac{\sqrt{2\pi}}{\sqrt{m[\omega_1+\omega_2]}} \exp\left[\frac{\left(\frac{\sqrt{\omega_2}}{\sqrt{\omega_1}}-\frac{\sqrt{\omega_1}}{\sqrt{\omega_2}}\right)^2 (x_1'+x_1'')^2}{8\frac{1}{m}\left(\frac{1}{\omega_1}+\frac{1}{\omega_2}\right)}\right] \\ &= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' |x_1''\rangle_{x_1\,x_1} \langle x_1'| \frac{1}{(\pi b_1^2)^{1/2}} \frac{1}{(\pi b_2^2)^{1/2}} \\ &\quad \times \frac{\sqrt{2\pi}}{\sqrt{m[\omega_1+\omega_2]}} \exp\left[\frac{m(\omega_2-\omega_1)^2 2x_1'x_1'' - m\left[8\omega_1\omega_2+(\omega_1-\omega_2)^2\right]\left([x_1']^2+[x_1'']^2\right)}{8(\omega_1+\omega_2)}\right]. \end{aligned} \tag{80}$$

Next, to identify the matrix, use

$$\begin{aligned}&\frac{m(\omega_2-\omega_1)^2 2x_1'x_1'' - m\left[8\omega_1\omega_2+(\omega_1-\omega_2)^2\right]\left([x_1']^2+[x_1'']^2\right)}{8(\omega_1+\omega_2)} \\ &= -\frac{1}{2b^2}\left[\left([x_1']^2+[x_1'']^2\right) - 2\frac{(\omega_2-\omega_1)^2}{\gamma}x_1'x_1''\right] \end{aligned} \tag{81}$$

$$\gamma \equiv 8\omega_1\omega_2 + (\omega_1-\omega_2)^2 \tag{82}$$

$$\frac{1}{2b^2} \equiv \frac{m\gamma}{8(\omega_1+\omega_2)} \tag{83}$$

$$b = 2\sqrt{\frac{\omega_1+\omega_2}{m\left[8\omega_1\omega_2+(\omega_1-\omega_2)^2\right]}} \tag{84}$$

to write

$$\begin{aligned}\hat{\rho} &= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' |x_1''\rangle_{x_1\,x_1} \langle x_1'| \frac{1}{(\pi b_1^2)^{1/2}} \frac{1}{(\pi b_2^2)^{1/2}} \\ &\quad \times \frac{\sqrt{2\pi}}{\sqrt{m[\omega_1+\omega_2]}} \exp\left[-\frac{1}{2b^2}\left([x_1']^2+[x_1'']^2\right)\right] \exp\left(\frac{(\omega_2-\omega_1)^2}{\gamma b^2}x_1'x_1''\right). \end{aligned} \tag{85}$$

Change basis to energy with a new effective frequency

$$b_3 = \frac{1}{\sqrt{m\omega_3}} \tag{86}$$

$$\hat{\rho} = \sum_{nv} |v\rangle\langle v|\hat{\rho}|n\rangle\langle n| \tag{87}$$

$$\begin{aligned}\langle v|\hat{\rho}|n\rangle &= \int \mathrm{d}x_1'' \int \mathrm{d}x_1' \langle v|x_1''\rangle_{x_1\,x_1} \langle x_1'|n\rangle \frac{1}{(\pi b_1^2)^{1/2}} \frac{1}{(\pi b_2^2)^{1/2}} \\ &\quad \times \frac{\sqrt{2\pi}}{\sqrt{m[\omega_1+\omega_2]}} \exp\left[-\frac{1}{2b^2}\left([x_1']^2+[x_1'']^2\right)\right] \exp\left(\frac{(\omega_2-\omega_1)^2}{\gamma b^2}x_1'x_1''\right) \end{aligned} \tag{88}$$

where

$$\langle x_1'|n\rangle = \frac{1}{\sqrt{n!b_3\sqrt{\pi}2^n}} \mathrm{e}^{\frac{-(x_1')^2}{2b_3^2}} H_n\left(\frac{x_1'}{b_3}\right) \tag{89}$$

are the well known oscillator wave functions and b_3 still has to be chosen. One can show by carrying out the integrals that the matrix is diagonalized if

$$b_3 = \frac{b}{\left(1 - b^4\left[\frac{(\omega_2-\omega_1)^2}{\gamma b^2}\right]^2\right)^{1/4}} = \frac{1}{\sqrt{m}\omega_1^{1/4}\omega_2^{1/4}}. \tag{90}$$

This gives

$$\langle \nu|\hat{\rho}|n\rangle = \lambda_n \delta_{\nu n}$$

where

$$\lambda_n = \frac{\sqrt{2\pi}}{\sqrt{m\left[\omega_1+\omega_2\right]}}\frac{1}{\left(\pi b_1^2\right)^{1/2}}\frac{1}{\left(\pi b_2^2\right)^{1/2}} m_{11}\left(\frac{b^2\frac{(\omega_2-\omega_1)^2}{\gamma b^2}}{1+\sqrt{1-b^4\left[\frac{(\omega_2-\omega_1)^2}{\gamma b^2}\right]^2}}\right)^{n-1}$$
$$= \frac{\pi\sqrt{m}}{2\left[\omega_1+\omega_2\right]^{3/2}}\frac{1}{\left(\pi b_1^2\right)^{1/2}}\frac{1}{\left(\pi b_2^2\right)^{1/2}}\frac{\left(\omega_2-\omega_1\right)^2}{\left(\sqrt{m}\omega_1^{1/4}\omega_2^{1/4}\right)^3\left(\frac{b_3^2}{b^2}+1\right)^{3/2}}\left(\frac{\frac{(\omega_2-\omega_1)^2}{8\omega_1\omega_2+(\omega_1-\omega_2)^2}}{1+\frac{b^2}{b_3^2}}\right)^{n-1} \tag{91}$$

where we used

$$m_{11} = \frac{b^3\frac{(\omega_2-\omega_1)^2}{\gamma b^2}\sqrt{2\pi}}{\left(1+\sqrt{1-b^4\left(\frac{(\omega_2-\omega_1)^2}{\gamma b^2}\right)^2}\right)^{3/2}} = \frac{m\left(\omega_2-\omega_1\right)^2\sqrt{2\pi}}{4\left(\omega_1+\omega_2\right)\left(\frac{1}{b^2}+\frac{1}{b_3^2}\right)^{3/2}} \tag{92}$$

$$\left(\frac{b_3}{b}\right)^2 = \frac{1}{m\omega_1^{1/2}\omega_2^{1/2}}\frac{1}{4\frac{\omega_1+\omega_2}{m\left[8\omega_1\omega_2+(\omega_1-\omega_2)^2\right]}} = \frac{1}{\omega_1^{1/2}\omega_2^{1/2}}\frac{8\omega_1\omega_2+\left(\omega_1-\omega_2\right)^2}{4\left(\omega_1+\omega_2\right)}. \tag{93}$$

Simplify:

$$\lambda_n = \frac{4\sqrt{\omega_1\omega_2}}{\sqrt{8\omega_1\omega_2+\left(\omega_1-\omega_2\right)^2+4\omega_1^{1/2}\omega_2^{1/2}\left(\omega_1+\omega_2\right)}}\left(\frac{\left(\omega_2-\omega_1\right)^2}{8\omega_1\omega_2+\left(\omega_1-\omega_2\right)^2+\omega_1^{1/2}\omega_2^{1/2}4\left(\omega_1+\omega_2\right)}\right)^n$$
$$= \frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]^n. \tag{94}$$

Since we want to evaluate

$$-\operatorname{Tr}\left[\hat{\rho}\ln\hat{\rho}\right] = -\partial_n \ln \operatorname{tr}\hat{\rho}^n|_{n=1} \tag{95}$$

we compute

$$\ln \operatorname{tr}\rho^n = \ln\left(\sum_{j=0}^{\infty}\lambda_j^n\right) = \ln\left(\sum_j\left[\frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]^j\right]^n\right)$$

$$= n\ln\left[\frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\right] + \ln\left(\sum_j\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]^{nj}\right)$$
$$= n\ln\left[\frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\right] - \ln\left(1-\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]^{n}\right). \tag{96}$$

Hence, we arrive at

$$S = -\left\{\ln\left[\frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\right] - \frac{\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]\ln\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]}{\left(1-\left[\frac{\left(\omega_1-\omega_2\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right]\right)}\right\}$$
$$= \boxed{-\ln\left(\frac{4\sqrt{\omega_1\omega_2}}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\right) - \left(\frac{\left(\omega_2-\omega_1\right)^2}{4\sqrt{\omega_1\omega_2}\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^2}\right)\ln\left(\frac{\left(\omega_2-\omega_1\right)^2}{\left(\sqrt{\omega_1}+\sqrt{\omega_2}\right)^4}\right)} \tag{97}$$

where

$$\boxed{\omega_1 = \sqrt{\frac{k}{m}} \quad \omega_2 = \sqrt{\frac{k+2g}{m}}}. \tag{98}$$

C.4. Level 4—SUSY-symmetry

Problem statement

Consider the theory

$$\mathcal{L} = \mathrm{i}\bar{\xi}\bar{\sigma}^\mu\partial_\mu\xi + |\partial\phi|^2 - |F|^2 \tag{99}$$

where ξ is a 2-component Weyl spinor while ϕ and F are complex scalar fields. Suppose you want to make the following infinitesimal transformation a symmetry of this theory:

$$\delta_\eta\xi_\alpha = \mathrm{i}\sqrt{2}\sigma^\mu_{\alpha\dot{\alpha}}\bar{\eta}^{\dot{\alpha}}\partial_\mu\phi + \sqrt{2}\eta_\alpha F \tag{100}$$

$$\begin{aligned}\delta_\eta\bar{\xi}_{\dot{\beta}} &= \left[\mathrm{i}\sqrt{2}\sigma^\mu_{\beta\dot{\alpha}}\bar{\eta}^{\dot{\alpha}}\partial_\mu\phi + \sqrt{2}\eta_\beta F\right]^\dagger \\ &= -\mathrm{i}\sqrt{2}\left(\bar{\eta}^{\dot{\alpha}}\sigma^{\mu *}_{\dot{\alpha}\beta}\right)^*\partial_\mu\bar{\phi} + \sqrt{2}\bar{\eta}_{\dot{\beta}}\bar{F} \\ &= -\mathrm{i}\sqrt{2}\eta^\alpha\sigma^\mu_{\alpha\dot{\beta}}\partial_\mu\bar{\phi} + \sqrt{2}\bar{\eta}_{\dot{\beta}}\bar{F}\end{aligned} \tag{101}$$

$$\delta_\eta F = \mathrm{i}\sqrt{2}\bar{\eta}_{\dot{\alpha}}\bar{\sigma}^{\mu\dot{\alpha}\alpha}\partial_\mu\xi_\alpha = \mathrm{i}\sqrt{2}\bar{\eta}\bar{\sigma}^\mu\partial_\mu\xi \tag{102}$$

$$\begin{aligned}\delta_\eta\bar{F} &= -\mathrm{i}\sqrt{2}\left(\bar{\eta}\bar{\sigma}^\mu\partial_\mu\xi\right)^\dagger \\ &= -\mathrm{i}\sqrt{2}\left(\partial_\mu\xi\right)^\dagger\left(\bar{\sigma}^\mu\right)^\dagger\left(\bar{\eta}\right)^\dagger \\ &= -\mathrm{i}\sqrt{2}\partial_\mu\bar{\xi}\bar{\sigma}^\mu\eta\end{aligned} \tag{103}$$

along with $\delta_\eta\phi$ and $(\delta_\eta\phi)^\dagger$ where η is a spacetime-independent infinitesimal fermionic parameter inducing the transformation. Find the transformation rule $\delta_\eta\phi$ and $(\delta_\eta\phi)^\dagger$ for the action associated with $\mathcal{L}$ to remain invariant.

Answer requirements

Provide the answer in the form of the `verbatim` code. Implement the following function

```
from math import sqrt
def find_delta_phi(eta:float, xi:float, bar_eta:float, bar_xi:float) -> Tuple[float, float]:
    """
    Returns the SUSY transformation rules for phi and its Hermitian conjugate:
    a tuple (delta_phi, delta_phi_dagger)
    """
    pass
```

Comments about the problem
This problem is situated in advanced QFT within the framework of supersymmetry (SUSY). It involves analyzing how bosonic and fermionic fields transform under an infinitesimal SUSY transformation and requires knowledge and careful application of Grassmann variables and the associated algebra. Such topics are typically encountered in advanced graduate-level physics courses. Note that the Hermiticity of σ^μ matrix convention as well as the metric convention of $(1,-1,-1,-1)$ is implicit in the statement of the problem (the latter inherent in the kinetic minus the potential form of the Lagrangian).

Solution
Denoting the variation $(\delta_\eta\phi)^\dagger$ as $\delta_\eta\bar{\phi}$, we write

$$\begin{aligned}\delta_\eta\mathcal{L} &= \mathrm{i}\delta_\eta\bar{\xi}\bar{\sigma}^\mu\partial_\mu\xi + \mathrm{i}\bar{\xi}\bar{\sigma}^\mu\partial_\mu\delta_\eta\xi + \partial_\mu\delta_\eta\bar{\phi}\partial^\mu\phi + \partial_\mu\bar{\phi}\partial^\mu\delta_\eta\phi - \delta_\eta\bar{F}F - \bar{F}\delta_\eta F\\ &= \mathrm{i}\left[-\mathrm{i}\sqrt{2}\eta\sigma^\beta\partial_\beta\bar{\phi} + \cancel{\sqrt{2}\eta\bar{F}}\right]\bar{\sigma}^\mu\partial_\mu\xi + \mathrm{i}\bar{\xi}\bar{\sigma}^\mu\partial_\mu\left[\mathrm{i}\sqrt{2}\sigma^\beta\bar{\eta}\partial_\beta\phi + \sqrt{2}\eta F\right]\\ &\quad + \partial_\mu\delta_\eta\bar{\phi}\partial^\mu\phi + \partial_\mu\bar{\phi}\partial^\mu\delta_\eta\phi - \left[-\mathrm{i}\sqrt{2}\partial_\mu\bar{\xi}\bar{\sigma}^\mu\eta\right]F - \bar{F}\left[\mathrm{i}\sqrt{2}\bar{\eta}\bar{\sigma}^\mu\partial_\mu\xi\right].\end{aligned} \tag{104}$$

Integrating by parts, we find (denoting with equality an equivalence up to total derivative terms)

$$\begin{aligned}\delta_\eta\mathcal{L} &= \sqrt{2}\eta\sigma^\beta\partial_\beta\bar{\phi}\bar{\sigma}^\mu\partial_\mu\xi + \partial_\mu\bar{\xi}\bar{\sigma}^\mu\left[\sqrt{2}\sigma^\beta\bar{\eta}\partial_\beta\phi - \mathrm{i}\cancel{\sqrt{2}\eta F}\right]\\ &\quad + \partial_\mu\delta_\eta\bar{\phi}\partial^\mu\phi + \partial_\mu\bar{\phi}\partial^\mu\delta_\eta\phi + \cancel{\mathrm{i}\sqrt{2}\partial_\mu\bar{\xi}\bar{\sigma}^\mu\eta F}.\end{aligned} \tag{105}$$

Integrate by parts the first two terms to eliminate the the σ matrices using the identity $\bar{\sigma}^\mu\sigma^\nu + \bar{\sigma}^\nu\sigma^\mu = 2g^{\mu\nu}$:

$$\delta_\eta\mathcal{L} = \sqrt{2}\left(\eta\partial_\mu\bar{\phi}\partial^\mu\xi + \partial^\mu\bar{\xi}\bar{\eta}\partial_\mu\phi\right) + \partial_\mu\delta_\eta\bar{\phi}\partial^\mu\phi + \partial_\mu\bar{\phi}\partial^\mu\delta_\eta\phi \tag{106}$$

again denoting with equality an equivalence up to total derivative terms, and we are using the standard notation $\eta\xi \equiv \eta^\alpha\xi_\alpha$ and $\bar{\xi}_{\dot{\alpha}}\bar{\eta}^{\dot{\alpha}} \equiv \bar{\xi}\bar{\eta}$. To make the remainder cancel, we solve

$$\sqrt{2}\eta\partial_\mu\bar{\phi}\partial^\mu\xi + \partial_\mu\bar{\phi}\partial^\mu\delta_\eta\phi = 0 \tag{107}$$

yielding

$$\boxed{\delta_\eta\phi = -\sqrt{2}\eta\xi, \quad (\delta_\eta\phi)^\dagger = -\sqrt{2}\bar{\xi}\bar{\eta}}. \tag{108}$$

C.5. Level 3—slow-roll inflation

Problem statement
For the action

$$S = \int \mathrm{d}t a^3(t)\left\{\frac{1}{2}\dot{\phi}^2 - V_0\exp\left[-\sqrt{\frac{2}{q}}\left(\frac{\phi}{M_P}\right)\right]\right\} \tag{109}$$

where q and V_0 are constants, derive and solve (integrate) the equation of motion for the field ϕ assuming slow-roll inflation and initial condition $\phi(t=0) = \phi_0$.

Answer requirements
Provide the answer in the form of the `verbatim` code. Implement the following function

```
import numpy as np
def phi(q: float, M_p: float, phi_0: float, V_0: float, t: np.ndarray)->np.ndarray:
    pass
```

Comments about the problem
This problem lies in the field of cosmology, particularly in inflationary cosmology, and involves studying the dynamics of a scalar field (inflaton) driving the accelerated expansion of the early Universe, before the 'hot Big Bang'. It is typically encountered in specialized graduate-level courses in cosmology and requires familiarity with field theory in an expanding spacetime.

Solution
The equation of motion is

$$\ddot{\phi} + 3H\dot{\phi} - \sqrt{\frac{2}{q}}\left(\frac{1}{M_P}\right) V_0 \exp\left[-\sqrt{\frac{2}{q}}\left(\frac{\phi}{M_P}\right)\right] = 0. \tag{110}$$

For the slow-roll inflation, the following must hold:

$$\ddot{\phi} \ll 3H\dot{\phi}. \tag{111}$$

Hence, we have

$$3H\dot{\phi} = \sqrt{\frac{2}{q}}\left(\frac{1}{M_P}\right) V_0 \exp\left[-\sqrt{\frac{2}{q}}\left(\frac{\phi}{M_P}\right)\right]. \tag{112}$$

Slow-roll approximation also implies

$$H^2 \approx \frac{V(\phi)}{3M_P^2} \tag{113}$$

so we need to solve the following ODE:

$$3\sqrt{\frac{V_0 \exp\left[-\sqrt{\frac{2}{q}}\left(\frac{\phi}{M_P}\right)\right]}{3M_P^2}}\frac{\mathrm{d}\phi}{\mathrm{d}t} = \sqrt{\frac{2}{q}}\left(\frac{1}{M_P}\right) V_0 \exp\left[-\sqrt{\frac{2}{q}}\left(\frac{\phi}{M_P}\right)\right] \tag{114}$$

$$\int \frac{\mathrm{d}\phi}{\sqrt{V_0}} \exp\left[\sqrt{\frac{1}{2q}}\left(\frac{\phi}{M_P}\right)\right] = \sqrt{\frac{2}{3q}}t. \tag{115}$$

Performing the integration and solving for $\phi(t)$ we get

$$\frac{1}{\sqrt{V_0}} M_P \sqrt{2q}\left(\exp\left[\sqrt{\frac{1}{2q}}\left(\frac{\phi}{M_P}\right)\right] - \exp\left[\sqrt{\frac{1}{2q}}\left(\frac{\phi_0}{M_P}\right)\right]\right) = \sqrt{\frac{2}{3q}}t \tag{116}$$

$$\boxed{\phi = \sqrt{2q} M_P \ln\left\{\exp\left[\sqrt{\frac{1}{2q}}\left(\frac{\phi_0}{M_P}\right)\right] + \frac{1}{M_P q}\sqrt{\frac{V_0}{3}}t\right\}}. \tag{117}$$

C.6. Level 3—scalar particle scattering

Problem statement
Consider

$$\mathcal{L} = \left\{\sum_{i=1}^{2}\left[\frac{1}{2}\left(\partial_\mu \phi_i\right)\left(\partial^\mu \phi_i\right) - \frac{m_i^2}{2}\phi_i \phi_i\right] - \frac{\lambda}{4}\phi_1^2 \phi_2^2\right\} \tag{118}$$

What is the differential cross section $\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}$ for $\phi_1(\vec{k}_1)\phi_1(-\vec{k}_1) \to \phi_2(\vec{k}_1')\phi_2(-\vec{k}_1')$ in the CM frame accurate to $O(\lambda^2)$? Express your final answer in terms of Mandelstam variables.

Answer requirements
Provide the answer in the form of the `verbatim` code. Implement the following function.

```
def dsigma_domega(lam: float, s_m: float, p_m: float, u_m: float,
                  m1: float, m2: float) -> float:
    pass
```

Comments about the problem
This is a question from QFT. It involves calculating the differential cross section for a process where two ϕ_1 particles annihilate into two ϕ_2 particles. Such problems are typically encountered in graduate-level particle physics courses and require familiarity with perturbative field theory and the use of Mandelstam variables to express scattering amplitudes.

Solution
The amplitude for this process is

$$\mathrm{i}\mathcal{M} = -4\mathrm{i}\frac{\lambda}{4} = -\mathrm{i}\lambda \tag{119}$$

In the CM frame, energy conservation gives

$$2\sqrt{|\vec{k}_1|^2 + m_1^2} = 2\sqrt{|\vec{k}_1'|^2 + m_2^2} \tag{120}$$

A standard formula for differential cross section gives

$$\begin{aligned}\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{\mathrm{CM}} &= \frac{1}{64\pi^2 s}\frac{k_1'}{k_1}|\mathcal{M}|^2 \\ &= \frac{\lambda^2}{64\pi^2 s}\frac{\sqrt{|\vec{k}_1|^2 + (m_1^2 - m_2^2)}}{k_1}\end{aligned} \tag{121}$$

Since in the CM frame, we know

$$k_1 = \frac{1}{2\sqrt{s}}\sqrt{s^2 - 4m_1^2 s} \tag{122}$$

$$\begin{aligned}\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{\mathrm{CM}} &= \frac{2\sqrt{s}}{64\pi^2 s}\sqrt{\frac{1}{4s}\left[s^2 - 4m_1^2 s\right] + (m_1^2 - m_2^2)}\frac{\lambda^2}{\sqrt{s^2 - 4m_1^2 s}} \\ &= \frac{\lambda^2}{64\pi^2 s}\frac{\sqrt{s^2 - 4m_1^2 s + 4s(m_1^2 - m_2^2)}}{\sqrt{s^2 - 4m_1^2 s}}.\end{aligned} \tag{123}$$

The final result is

$$\boxed{\left(\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega}\right)_{\mathrm{CM}} = \frac{\lambda^2}{64\pi^2 s}\frac{\sqrt{s - 4m_2^2}}{\sqrt{s - 4m_1^2}}}. \tag{124}$$

C.7. Level 2—dark matter capture as a function of time

Problem statement
Suppose C is the capture rate of dark matter in an astrophysical body. Let C_A be the dark matter annihilation rate per effective volume. Then an approximate Boltzmann equation governing the number N of dark matter particles in the astrophysical body is

$$\frac{\mathrm{d}N}{\mathrm{d}t} = C - C_A N^2.$$

If initially, $N(0) = 0$, what is $N(t)$ as a function of time?

Answer requirements
Provide the answer in the form of the `verbatim` code. Implement the following function.

```
def answer(C: float, C_A: float, t: float) -> float:
    pass
```

Comments about the problem
This problem mainly belongs to astrophysics, specifically involving dark matter dynamics in celestial bodies. It is typically encountered in advanced undergraduate or graduate-level courses and requires knowledge of differential equations and kinetic theory. This type of analysis is also important for understanding dark matter detection and its astrophysical implications.

Solution
We can integrate by quadrature.

$$\int \frac{\mathrm{d}N}{C - C_A N^2} = t. \tag{125}$$

We can express the integrand as a sum of two fractions:

$$\begin{aligned}\frac{1}{C - C_A N^2} &= \frac{1}{\sqrt{C} - \sqrt{C_A} N} \frac{1}{\sqrt{C} + \sqrt{C_A} N} \\ &= \frac{1}{2\sqrt{C}} \left[\frac{1}{\sqrt{C} - \sqrt{C_A} N} + \frac{1}{\sqrt{C} + \sqrt{C_A} N} \right].\end{aligned} \tag{126}$$

Integrating, we find

$$\begin{aligned}t + K &= \frac{1}{2\sqrt{C}} \left[\frac{-1}{\sqrt{C_A}} \ln\left(\sqrt{C} - \sqrt{C_A} N\right) + \frac{1}{\sqrt{C_A}} \ln\left(\sqrt{C} + \sqrt{C_A} N\right) \right] \\ &= \frac{1}{2\sqrt{C_A C}} \ln\left(\frac{\sqrt{C} + \sqrt{C_A} N}{\sqrt{C} - \sqrt{C_A} N} \right)\end{aligned} \tag{127}$$

where K is an integration constant. Setting the boundary condition $N = 0$ at $t = 0$, we find

$$K = 0.$$

We find the solution

$$\boxed{N = \frac{\sqrt{C}}{\sqrt{C_A}} \frac{\left(\mathrm{e}^{2\sqrt{CC_A}t} - 1\right)}{\left(\mathrm{e}^{2\sqrt{CC_A}t} + 1\right)}}. \tag{128}$$

Note that it is easy to check that it reaches the obvious steady state in the limit $t \to \infty$.

C.8. Level 2—a 3-state QM problem

Problem statement
The Hamiltonian of a three-level system is given as $H = \begin{pmatrix} E_a & 0 & A \\ 0 & E_b & 0 \\ A & 0 & E_a \end{pmatrix}$ where A is real. The state of the system at time $t = 0$ is (in this basis) $\psi(t = 0) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ What is the expectation value of the energy at time t?

Answer requirements
Provide the answer in the form of `verbatim` code. Implement the following function

```
def expectation_value(A: float, E_a:float, E_b:float, t:float) -> float:
    pass
```

Comments about the problem
This problem belongs to quantum mechanics, focusing on multi-level quantum systems found in areas like quantum optics or molecular physics. It is typically encountered in advanced undergraduate or early graduate-level courses and requires knowledge of linear algebra, time evolution, and the calculation of expectation values in quantum mechanics.

Solution
The eigenstates are easily found to be $\frac{1}{\sqrt{2}}(1, 0, \pm 1)^{\mathrm{T}}$ and $(0, 1, 0)^{\mathrm{T}}$ with corresponding energies $E_a \pm A$, E_b. Let us denote them as $|1\rangle$, $|2\rangle$ and $|3\rangle$. Given state ψ is decomposed as $\frac{1}{2}(|1\rangle + |2\rangle) + \frac{1}{\sqrt{2}}|3\rangle$, the expectation of energy stays constant:

$$\langle E \rangle = \frac{1}{4}\left((E_a + A) + (E_a - A)\right) + \frac{1}{2}E_b = \boxed{\frac{1}{2}\left(E_a + E_b\right)}. \tag{129}$$

C.9. Level 1—blackbody in *d* dimensions

Problem statement

Assume we live in a 4+1 dimensional spacetime. How does the total energy density of a black body scale with temperature T. Find the exponent n in the expression $u \propto T^n$.

Answer requirements

Provide the answer in the form of `verbatim` code. Implement the following function

```
def answer() -> float:
    pass
```

Comments about the problem

This problem lies in the realm of statistical mechanics and thermodynamics applied to higher-dimensional spacetimes, a topic typically encountered at the undergraduate level in TP.

Solution

The density of states scales as $k^{D-1}dk$ in D spatial dimensions giving T^{D+1} scaling for the total energy density. Hence, $\boxed{n = 5}$.

C.10. Level 1—boosted parabolic trajectory

Problem statement

Consider a situation where a space-probe very briefly fires its rockets while passing a planet of mass M at periapsis, its nearest point to the planet. Suppose that the probe is on a parabolic trajectory and at periapsis, when travelling at velocity v_e, it results in a boost of δv. What will be its speed once it escapes the planet's gravitational field only in terms of v_e and δv?

Answer requirements

Provide the answer in the form of `verbatim` code. Implement the following function

```
def speed(v_e: float, delta_v:float) -> float:
    pass
```

Comments about the problem

This problem is part of orbital mechanics, typically covered at the undergraduate or advanced high school level in physics. It involves principle of energy conservation in Newtonian gravity.

Solution

Conservation of energy gives $\frac{1}{2}m(v_e+\delta v)^2 - \frac{mMG}{r_p} = \frac{1}{2}mv_\infty^2$. We also know that $\frac{1}{2}m(v_e)^2 - \frac{mMG}{r_p} = E = 0$ for the parabolic trajectory. We can solve for v_e: $v_e = \sqrt{\frac{2MG}{r_p}}$. Then we can substitute it in the first equation and get:

$$\boxed{v_\infty = \delta v\sqrt{1+\frac{2v_e}{\delta v}}}. \tag{130}$$

ORCID iDs

Zhiqi Gao 0009-0006-4989-3753
Yurii Kvasiuk 0009-0002-4720-1320
Tianyi Li 0000-0001-9545-8556
Moritz Münchmeyer 0000-0002-3777-7791
Sai Chaitanya Tadepalli 0000-0001-9947-4748

References

[1] Hendrycks D, Burns C, Kadavath S, Arora A, Basart S, Tang E, Song D and Steinhardt J 2021 Measuring mathematical problem solving with the math dataset (arXiv:2103.03874)
[2] Glazer E *et al* 2024 FrontierMath: a benchmark for evaluating advanced mathematical reasoning in AI (arXiv:2411.04872)
[3] Arora D, Singh H G and Mausam 2023 Have LLMs advanced enough? A challenging problem solving benchmark for large language models (arXiv:2305.15074)

[4] He C et al 2024 OlympiadBench: a challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems (arXiv:2402.14008)
[5] Jaiswal R, Jain D, Popat H P, Anand A, Dharmadhikari A, Marathe A and Shah R R 2024 Improving physics reasoning in large language models using mixture of refinement agents (arXiv:2412.00821)
[6] Pan H, Mudur N, Taranto W, Tikhanovskaya M, Venugopalan S, Bahri Y, Brenner M P and Kim E-A 2024 Quantum many-body physics calculations with large language models (arXiv:2403.03154)
[7] Phan L et al 2025 Humanity's last exam (arXiv:2501.14249)
[8] Iyer V and Wald R M 1994 Some properties of Noether charge and a proposal for dynamical black hole entropy *Phys. Rev. D* **50** 846–64
[9] Geroch R P 1968 Spinor structure of space-times in general relativity. I *J. Math. Phys.* **9** 1739–44
[10] Kontsevich M 1992 Intersection theory on the moduli space of curves and the matrix Airy function *Commun. Math. Phys.* **147** 1–23
[11] Schon R and Yau S-T 1981 Proof of the positive mass theorem. 2 *Commun. Math. Phys.* **79** 231–60
[12] Aganagic M, Danilenko I, Li Y, Shende V and Zhou P 2024 Quiver Hecke algebras from Floer homology in Couloumb branches (arXiv:2406.04258)
[13] Parker T and Taubes C H 1982 On Witten's proof of the positive energy theorem *Commun. Math. Phys.* **84** 223
[14] Hausel T and Thaddeus M 2003 Mirror symmetry, Langlands duality and the Hitchin system *Invent. Math.* **153** 197
[15] Hardy G H 1940 *A Mathematician's Apology* (Cambridge University Press) (available at: www.cambridge.org/core/books/mathematicians-apology/A344F9D097F5AFF45BDA21B57B54BDCA) (Foreword by C P Snow)
[16] Rota G-C 1997 Ten lessons i wish i had been taught *Not. Am. Math. Soc.* **44** 22–25 (available at: www.ams.org/notices/199701/comm-rota.pdf)
[17] Si C, Yang D and Hashimoto T 2024 Can LLMs generate novel research ideas? A large-scale human study with 100+ NLP researchers (arXiv:2409.04109)
[18] Pólya G 1945 *How to Solve It: A New Aspect of Mathematical Method* (Princeton University Press)
[19] OpenAI 2024 GPT-4o (available at: https://openai.com/index/hello-gpt-4o/)
[20] OpenAI 2024 Introducing OpenAI o1-preview (available at: https://openai.com/index/introducing-openai-o1-preview/)
[21] OpenAI 2025 o3-mini (available at: https://openai.com/index/openai-o3-mini/)
[22] Imani S, Du L and Shrivastava H 2023 Mathprompter: mathematical reasoning using large language models (arXiv:2303.05398)
[23] Huang J, Chen X, Mishra S, Zheng H S, Yu A W, Song X and Zhou D 2023 Large language models cannot self-correct reasoning yet (arXiv:2310.01798)
[24] Yin Z, Sun Q, Guo Q, Wu J, Qiu X and Huang X 2023 Do large language models know what they don't know? (arXiv:2305.18153)
[25] Kamoi R, Zhang Y, Zhang N, Han J and Zhang R 2024 When can LLMs actually correct their own mistakes? A critical survey of self-correction of LLMs (arXiv:2406.01297)
[26] Dhuliawala S, Komeili M, Xu J, Raileanu R, Li X, Celikyilmaz A and Weston J 2023 Chain-of-verification reduces hallucination in large language models (arXiv:2309.11495)
[27] Zhang J, Li Z, Das K, Malin B and Kumar S 2023 SAC3: reliable hallucination detection in black-box language models via semantic-aware cross-check consistency (arXiv:2311.01740)
[28] Jiang Z, Peng H, Feng S, Li F and Li D 2024 Llms can find mathematical reasoning mistakes by pedagogical chain-of-thought (arXiv:2405.06705)
[29] Kvasiuk Y, Münchmeyer M and Smith K 2024 A tale of two fields: neural network-enhanced non-gaussianity search with halos (arXiv:2410.01007)
[30] Basso E, Chung D J H, Kolb E W and Long A J 2022 Quantum interference in gravitational particle production *J. High Energy Phys.* JHEP12(2022)108
[31] Ellis J 2017 TikZ-Feynman: Feynman diagrams with TikZ *Comput. Phys. Commun.* **210** 103–23
[32] Meta AI 2024 Meta Llama 3.1 (available at: https://ai.meta.com/blog/meta-llama-3-1/)
[33] Qwen Team 2024 Qwen2.5 (available at: https://qwenlm.github.io/blog/qwen2.5/)
[34] Qwen Team 2024 Qwen QwQ 32B preview (available at: https://qwenlm.github.io/blog/qwq-32b-preview/)
[35] Guo D et al 2025 DeepSeek-R1: incentivizing reasoning capability in LLMs via reinforcement learning (arXiv:2501.12948)
[36] Bi X et al 2024 DeepSeek LLM: scaling open-source language models with longtermism (arXiv:2401.02954)
[37] Chen G H, Chen S, Liu Z, Jiang F and Wang B 2024 Humans or LLMs as the judge? A study on judgement biases (arXiv:2402.10669)
[38] Kumar T and Kats M A 2023 ChatGPT-4 with code interpreter can be used to solve introductory college-level vector calculus and electromagnetism problems (arXiv:2309.08881)
[39] Wu Y, Jia F, Zhang S, Li H, Zhu E, Wang Y, Lee Y T, Peng R, Wu Q and Wang C 2023 MathChat: converse to tackle challenging math problems with LLM agents (arXiv:2306.01337)
[40] Gao Z, Li T, Kvasiuk Y, Tadepalli S C, Rudolph M, Chung D J H, Sala F and Münchmeyer M 2025 Test-time scaling techniques in theoretical physics—a comparison of methods on the TPBench dataset (arXiv:2506.20729)
[41] Peskin M E and Schroeder D V 1995 *An Introduction to Quantum Field Theory* (Addison-Wesley) (https://doi.org/10.1201/9780429503559)
[42] Bailin D and Love A 1994 *Supersymmetric Gauge Field Theory and String Theory* (CRC Press) (https://doi.org/10.1201/9780367805807)
[43] Chandrasekhar S 1985 The mathematical theory of black holes *General Relativity and Gravitation* (Springer)
[44] Cobbe K et al 2021 Training verifiers to solve math word problems (arXiv:2110.14168)
[45] Ling W, Yogatama D, Dyer C and Blunsom P 2017 Program induction by rationale generation: learning to solve and explain algebraic word problems *Proc. 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* pp 158–67
[46] Tsoukalas G, Lee J, Jennings J, Xin J, Ding M, Jennings M, Thakur A and Chaudhuri S 2024 PutnamBench: evaluating neural theorem-provers on the Putnam mathematical competition (arXiv:2407.11214)
[47] Zheng K, Han J M and Polu S 2022 MiniF2F: a cross-system benchmark for formal olympiad-level mathematics (arXiv:2109.00110)
[48] Liu C et al 2023 FIMO: a challenge formal dataset for automated theorem proving (arXiv:2309.04295)
[49] Azerbayev Z, Piotrowski B, Schoelkopf H, Ayers E W, Radev D and Avigad J 2023 ProofNet: autoformalizing and formally proving undergraduate-level mathematics (arXiv:2302.12433)
[50] Rein D, Hou B Li, Stickland A C, Petty J, Pang R Y, Dirani J, Michael J and Bowman S R 2023 GPQA: a graduate-level Google-proof Q&A benchmark (arXiv:2311.12022)

[51] Mirzadeh I, Alizadeh K, Shahrokhi H, Tuzel O, Bengio S and Farajtabar M 2024 GSM-symbolic: understanding the limitations of mathematical reasoning in large language models (arXiv:2410.05229)
[52] Team K *et al* 2025 Kimi k1. 5: scaling reinforcement learning with LLMs (arXiv:2501.12599)
[53] Xu H *et al* 2025 RedStar: does scaling long-CoT data unlock better slow-reasoning systems? (arXiv:2501.11284)
[54] Bespoke Labs 2025 Bespoke-Stratos: the unreasonable effectiveness of reasoning distillation (available at: https://hf.co/bespokelabs/Bespoke-Stratos-32B) (Accessed 22 January 2025)
[55] Yu L, Jiang W, Shi H, Yu J, Liu Z, Zhang Y, Kwok J T, Li Z, Weller A and Liu W 2024 MetaMath: bootstrap your own mathematical questions for large language models (arXiv:2309.12284)
[56] Zelikman E, Wu Y, Mu J and Goodman N 2022 STaR: bootstrapping reasoning with reasoning *Advances in Neural Information Processing Systems* vol 35 pp 15476–88
[57] Zelikman E, Harik G, Shao Y, Jayasiri V, Haber N and Goodman N D 2024 Quiet-STaR: language models can teach themselves to think before speaking (arXiv:2403.09629)
[58] Shao Z, Wang P, Zhu Q, Xu R, Song J, Bi X, Zhang H, Zhang M, Li Y K, Wu Y and Guo D 2024 DeepSeekMath: pushing the limits of mathematical reasoning in open language models (arXiv:2402.03300)
[59] Chen Z, Deng Y, Yuan H, Ji K and Gu. Q 2024 Self-play fine-tuning converts weak language models to strong language models (arXiv:2401.01335)
[60] Wei J, Wang X, Schuurmans D, Bosma M, Ichter B, Xia F, Chi E, Le Q and Zhou D 2023 Chain-of-thought prompting elicits reasoning in large language models (arXiv:2201.11903)
[61] Snell C, Lee J, Xu K and Kumar A 2024 Scaling LLM test-time compute optimally can be more effective than scaling model parameters (arXiv:2408.03314)
[62] Welleck S, Bertsch A, Finlayson M, Schoelkopf H, Xie A, Neubig G, Kulikov I and Harchaoui Z 2024 From decoding to meta-generation: inference-time algorithms for large language models (arXiv:2406.16838)
[63] Muennighoff N, Yang Z, Shi W, Li X L, Fei-Fei L, Hajishirzi H, Zettlemoyer L, Liang P, Candès E and Hashimoto T 2025 s1: simple test-time scaling (arXiv:2501.19393)
[64] Khot T, Trivedi H, Finlayson M, Fu Y, Richardson K, Clark P and Sabharwal A 2022 Decomposed prompting: a modular approach for solving complex tasks (arXiv:2210.02406)
[65] Zhou D, Schärli N, Hou L, Wei J, Scales N, Wang X, Schuurmans D, Bousquet O, Le Q and Chi E 2022 Least-to-most prompting enables complex reasoning in large language models (arXiv:2205.10625)
[66] Hao S, Gu Y, Ma H, Hong J J, Wang Z, Wang D Z and Hu Z 2023 Reasoning with language model is planning with world model (arXiv:2305.14992)
[67] Zheng H S, Mishra S, Chen X, Cheng H-T, Chi H, Le Q V and Zhou D 2023 Take a step back: evoking reasoning via abstraction in large language models (arXiv:2310.06117)
[68] Lightman H, Kosaraju V, Burda Y, Edwards H, Baker B, Lee T, Leike J, Schulman J, Sutskever I and Cobbe K 2023 Let's verify step by step (arXiv:2305.20050)
[69] Ren J, Zhao Y, Vu T, Liu P J and Lakshminarayanan B 2023 Self-evaluation improves selective generation in large language models (arXiv:2312.09300)
[70] Chen S, Li B and Niu D 2024 Boosting of thoughts: trial-and-error problem solving with large language models (arXiv:2402.11140)
[71] Madaan A *et al* 2024 Self-refine: iterative refinement with self-feedback *Advances in Neural Information Processing Systems* vol 36
[72] Forsman A 2024 Analyzing the performance of self-refine on different large language models (available at: https://github.com/anforsm/self-refine/blob/main/report.pdf)
[73] Beirami A, Agarwal A, Berant J, D'Amour A, Eisenstein J, Nagpal C and Suresh A T 2025 Theoretical guarantees on the best-of-n alignment policy (arXiv:2401.01879)
[74] Wang X, Wei J, Schuurmans D, Le Q V, Chi E H, Narang S, Chowdhery A and Zhou D 2023 Self-consistency improves chain of thought reasoning in language models *11th Int. Conf. on Learning Representations* (available at: https://openreview.net/forum?id=1PL1NIMMrw)
[75] Yao S, Yu D, Zhao J, Shafran I, Griffiths T, Cao Y and Narasimhan K 2023 Tree of thoughts: deliberate problem solving with large language models *Advances in Neural Information Processing Systems* vol 36 pp 11809–22
[76] Qi Z, Ma M, Xu J, Zhang Li L, Yang F and Yang M 2024 Mutual reasoning makes smaller LLMs stronger problem-solvers (arXiv:2408.06195)
[77] Zhang D *et al* 2024 LLaMA-Berry: pairwise optimization for O1-like olympiad-level mathematical reasoning (arXiv:2410.02884)
[78] Kang J *et al* 2024 MindStar: enhancing math reasoning in pre-trained LLMs at inference time (arXiv:2405.16265)
[79] Kocsis L and Szepesvári C 2006 Bandit based monte-carlo planning *European Conf. on Machine Learning* (Springer) pp 282–93
[80] Schick T, Dwivedi-Yu J, Dessì) R, Raileanu R, Lomeli M, Hambro E, Zettlemoyer L, Cancedda N and Scialom T 2024 Toolformer: language models can teach themselves to use tools *Advances in Neural Information Processing Systems* vol 36
[81] Saad-Falcon J *et al* 2024 Archon: an architecture search framework for inference-time techniques (arXiv:2409.15254)
[82] Chen W, Ma X, Wang X and Cohen W W 2022 Program of thoughts prompting: disentangling computation from reasoning for numerical reasoning tasks (arXiv:2211.12588)
[83] Gao Y, Xiong Y, Gao X, Jia K, Pan J, Bi Y, Dai Y, Sun J and Wang H 2023 Retrieval-augmented generation for large language models: a survey (arXiv:2312.10997)
[84] Srednicki M 1993 Entropy and area *Phys. Rev. Lett.* **71** 666–9