SciCode: A Research Coding Benchmark Curated by Scientists

Minyang Tian[1,2*‡], Luyu Gao[3*], Shizhuo Dylan Zhang[1], Xinan Chen[1†], Cunwei Fan[1†], Xuefei Guo[1†], Roland Haas[1†], Pan Ji[4†], Kittithat Krongchon[1†],Yao Li[1†], Shengyan Liu[1†], Di Luo[5,6,11†],Yutao Ma[7†], Hao Tong[1†], Kha Trinh[7†], Chenyu Tian[8†], Zihan Wang[1†], Bohao Wu[1†], Yanyu Xiong[9†], Shengzhu Yin[1†], Minhui Zhu[1†], Kilian Lieret[10], Yanxin Lu[1], Genglin Liu[1], Yufeng Du[1], Tianhua Tao[1], Ofir Press[10], Jamie Callan[3], Eliu Huerta[1,2,7‡], Hao Peng[1‡]

[1]University of Illinois Urbana-Champaign [2]Argonne National Laboratory
[3]Carnegie Mellon University [4]University of North Carolina at Chapel Hill
[5]Massachusetts Institute of Technology [6]Harvard University [7] University of Chicago
[8]University of Texas at Austin [9]Stanford University [10] Princeton University
[11]The NSF AI Institute for Artificial Intelligence and Fundamental Interactions
[*] Equal contribution lead authors. [†] Data curation, alphabetical order.
‡ Correspondence to {mtian8, haopeng}@illinois.edu, elihu@anl.gov

Abstract

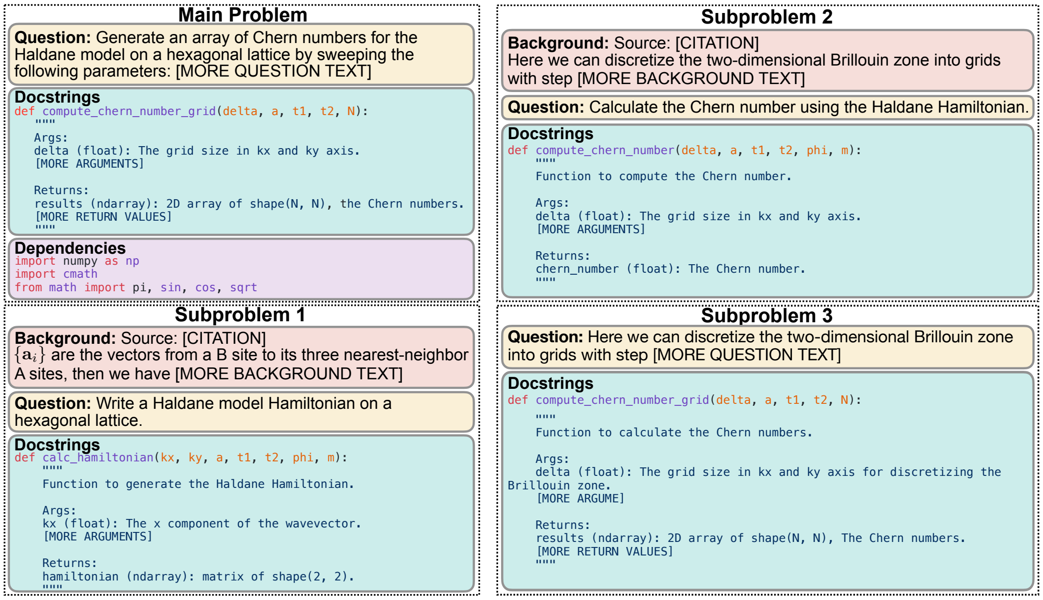
Since language models (LMs) now outperform average humans on many challenging tasks, it is becoming increasingly difficult to develop challenging, high-quality, and realistic evaluations. We address this by examining LM capabilities to generate code for solving real scientific research problems. Incorporating input from scientists and AI researchers in 16 diverse natural science sub-fields, including mathematics, physics, chemistry, biology, and materials science, we create a scientist-curated coding benchmark, **SciCode**. The problems naturally factorize into multiple subproblems, each involving knowledge recall, reasoning, and code synthesis. In total, SciCode contains 338 subproblems decomposed from 80 challenging main problems, and it offers optional descriptions specifying useful scientific background information and scientist-annotated gold-standard solutions and test cases for evaluation. Claude3.5-Sonnet, the best-performing model among those tested, can solve only 4.6% of the problems in the most realistic setting. We believe that SciCode demonstrates both contemporary LMs' progress towards realizing helpful scientific assistants and sheds light on the building and evaluation of scientific AI in the future. [1]

1 Introduction

The development of evaluations in tandem with language models (LMs) has substantially contributed to the rapid advancement of these models [29, 11, 7, 25, 81, 27, 72]. Because LMs now surpass the performance of most humans except domain experts, evaluating them becomes increasingly challenging. Many established benchmarks struggle to keep pace with the advancements in LM performance and have quickly become saturated [91, 14, 70, 58], leading to discrepancies between the models' perceived and actual capabilities [36]. As a consequence, researchers are developing synthetic challenging benchmarks, often involving models in the construction of evaluation instances. For example, some subsample instances from existing benchmarks that cannot be solved by current models [93, 82], or augment them to construct more challenging evaluations [21, 44, 49]. However, it is unclear whether such efforts accurately reflect real-world applications and the models' performance in practical scenarios. Realistic, high-quality, and challenging evaluations are crucial for the continued advancement of LMs.

[1] Data, code, and leaderboard available at `https://scicode-bench.github.io/`

Preprint. Under review.



Question: Generate an array of Chern numbers for the Haldane model on a hexagonal lattice by sweeping the following parameters: [MORE QUESTION TEXT]

Docstrings

```
def compute_chern_number_grid(delta, a, t1, t2, N):
   """
   Args:
   delta (float): The grid size in kx and ky axis.
   [MORE ARGUMENTS]

   Returns:
   results (ndarray): 2D array of shape(N, N), the Chern numbers.
   [MORE RETURN VALUES]
   """
```

Dependencies

```
import numpy as np
import cmath
from math import pi, sin, cos, sqrt
```

Background: Source: [CITATION]
$\{\mathbf{a}_i\}$ are the vectors from a B site to its three nearest-neighbor A sites, then we have [MORE BACKGROUND TEXT]

Question: Write a Haldane model Hamiltonian on a hexagonal lattice.

Docstrings

```
def calc_hamiltonian(kx, ky, a, t1, t2, phi, m):
    """
    Function to generate the Haldane Hamiltonian.

    Args:
    kx (float): The x component of the wavevector.
    [MORE ARGUMENTS]

    Returns:
    hamiltonian (ndarray): matrix of shape(2, 2).
    """
```

Background: Source: [CITATION]
Here we can discretize the two-dimensional Brillouin zone into grids with step [MORE BACKGROUND TEXT]

Question: Calculate the Chern number using the Haldane Hamiltonian.

Docstrings

```
def compute_chern_number(delta, a, t1, t2, phi, m):
    """
    Function to compute the Chern number.

    Args:
    delta (float): The grid size in kx and ky axis.
    [MORE ARGUMENTS]

    Returns:
    chern_number (float): The Chern number.
    """
```

Question: Here we can discretize the two-dimensional Brillouin zone into grids with step [MORE QUESTION TEXT]

Docstrings

```
def compute_chern_number_grid(delta, a, t1, t2, N):
    """
    Function to calculate the Chern numbers.

    Args:
    delta (float): The grid size in kx and ky axis for discretizing the
Brillouin zone.
    [MORE ARGUME]

    Returns:
    results (ndarray): 2D array of shape(N, N), The Chern numbers.
    [MORE RETURN VALUES]
    """
```

Figure 1: A SciCode main problem is decomposed into multiple smaller and easier subproblems. Docstrings specify the requirements and input-output formats. When necessary, scientific background knowledge is provided, written by our scientist annotators. The full problem is shown in subsection A.3

We therefore propose SciCode, a benchmark containing code generation problems drawn from diverse natural science fields, including mathematics, physics, chemistry, biology, and materials science. SciCode contains 80 main problems, each decomposed into multiple subproblems, totaling 338. Each problem provides the scientific background when necessary as well as detailed instructions. To solve it, the model must implement multiple Python functions—one for each subproblem—and then integrate them into a complete solution for the main problem. For every main problem and subproblem, SciCode provides gold-standard solutions and multiple test cases, facilitating easy and reliable automatic evaluation. Figure 1 shows an example.

SciCode aims to overcome the challenges of current LM evaluations by introducing the following value-added design choices.

- *Intentional focus on natural science fields*, such as computational mechanics, quantum information and computing, quantum chemistry, ecology, and molecular modeling.
- *Abundant high-quality data not usually made available to current LMs* [92, 1, 13], enabling a more robust evaluation of the models' ability to generalize to less familiar scenarios.
- *High annotation quality*, with all problems, including gold solutions and test cases, annotated, revised, and verified by at least two senior researchers (PhD student level or above) in represented scientific domains.
- *Realistic and current problems* sourced from scientists' everyday research tasks or influential papers. This ensures SciCode's relevance to real-world applications.
- *Problems curated to have zero overlap with publicly available datasets* to prevent potential data contamination.[2]
- *Problems that test LM's comprehensive and all-around capabilities*. Solving the main problems requires deep scientific background knowledge, strong analytical capabilities to decompose complex problems into simpler ones and correctly solve each, and the ability to integrate partial into complete solutions.
- *Opportunities to evaluate various model capabilities in varied setups* by toggling options, e.g., whether to provide scientific background information or to condition on gold or generated solutions to previous subproblems.

[2] In addition to addressing data contamination, we find that most problems are too challenging for even the best models. Therefore, we often simplify problem settings and provide more background during revisions.

Further, we believe that the availability of this well-designed benchmark can *motivate research into developing new AI methods for accelerating scientific research*, an area that has thus far benefited less from recent LM advancements partly due to a lack of commercial incentive.

We use SciCode to evaluate state-of-the-art proprietary and open models. Results show that SciCode is a very challenging benchmark: in the most realist evaluation setup, Claude3.5-Sonnet, the best-performing model in our experiments, can solve only 4.6% of the main problems, while other strong models, such as Claude3-Opus and GPT-4o, solve only 1.5%. Similarly, the best open source model under test, Deepseek-Coder-v2, can only solve 3.1% of the problems. The other open-source LLMs under test (e.g., Llama-3-70B-Instruct and Mixtral-8x22B-Inst) fail to complete any problems despite successfully solving some subproblems correctly. Our analysis finds that all models can benefit from the background knowledge written by our scientist annotators, achieving substantial and consistent improvements. However, even with background, the best model can solve only 12.3% of the main problems.

2 SciCode

This section examines the design principles and annotation process we chose for SciCode, describing: research-level coding problems from various natural science fields (§2.1); how we decomposed main problems into multiple, simpler subproblems (§2.2); our design choices for the annotation process (§2.3); and various evaluation setups that SciCode facilitates (§2.4).

2.1 Challenging and Realistic Scientific Coding Problems

SciCode sources challenging and realistic research-level coding problems across natural science disciplines, including mathematics, physics, chemistry, biology, and material science, covering a total of 16 subfields. This diverse selection ensures a comprehensive representation of the natural sciences, where extensive code development is essential.

SciCode is mainly drawn from the scripts that scientists use in their everyday workflow. Many of these have been used in one or more publications, demonstrating their robustness and correctness. However, they are primarily for internal use, which means that they are seldomly open-sourced and often poorly annotated. Consequently, unlike general-domain coding problems, natural science problems have less exposure in most current LMs' training data. This offers a unique opportunity to evaluate the models' ability to generalize to less familiar contexts. In total, SciCode consists of 80 main problems, decomposed into 338 subproblems.

Table 1 lists the subfields SciCode covers along with the number of main problems in each. Each main problem has a median of 3 subproblems, with a maximum of 15. We reserve 15 main problems (50 subproblems) for the development split and use the remaining 65 main problems (288 subproblems) as the test data. The 15 main development problems cover all five domains; over half of these have less than 4 subproblems each for easier few-shot settings.

Fields	**Subfields**
Mathematics	Numerical linear Algebra (8), Computational Mechanics (5), Computational Finance (1)
Physics	Condensed Matter Physics (13), Optics (10), Quantum Information/Computing (6), Computational Physics (5), Astrophysics (2), Particle Physics (1)
Chemistry	Quantum Chemistry (5), Computational Chemistry (3)
Biology	Ecology (6), Biochemistry (1), Genetics (1)
Material Science	Semiconductor Materials (7), Molecular Modeling (6)

Table 1: SciCode fields and subfields, with the number of main problems in each.

2.2 A Main Problem with Multiple Subproblems

In their everyday workflow, scientists often decompose a complex problem into multiple smaller, more manageable parts. They may write relatively independent code for each part and then integrate these parts into a complete solution to the main problem. In developing our dataset, we leverage

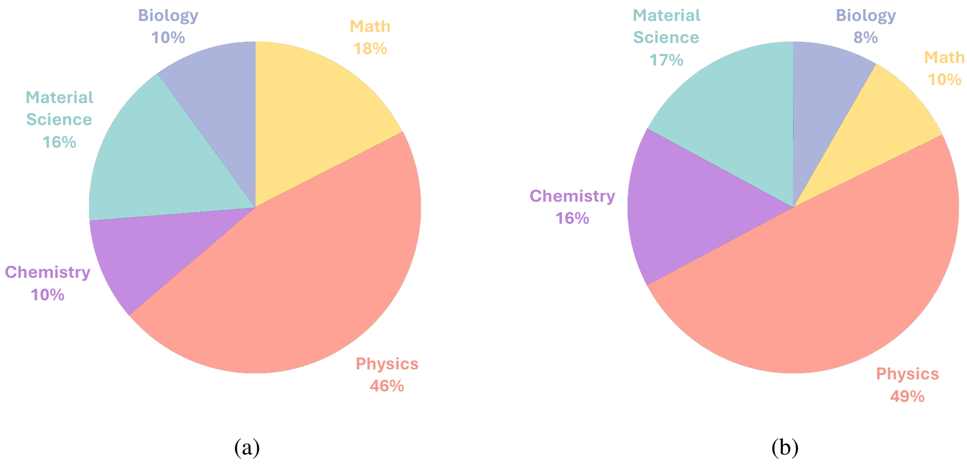


Figure 2: Distributions of (a) main problems and (b) subproblems.

this natural and intuitive structure and further standardize our dataset by instructing the scientists to adhere to the following format.

Main Problem A *main problem* is a primary task that needs to be addressed. It defines the overall objective of the research and guides the direction of the study. The main problem encompasses all subproblems, with detailed instructions on required inputs and expected outputs articulated in a docstring block. With the main problem defined, scientists have sufficient guidance to solve the task.

Subproblem Decomposition *Subproblems* focus on questions derived from the main problem. They decompose the complex main problem into smaller, more manageable parts, enabling a more detailed and systematic investigation. Detailed docstrings for each subproblem describe the required input and expected output, ensuring clarity and aiding in accurate code generation. This structured decomposition simplifies problem-solving and facilitates a more granular evaluation of the models' scientific coding capabilities.

2.3 Data Annotation

This process consists of three main stages:

(1) **Problem selection:** Deciding on question topics related to the research domain (§2.3.1).
(2) **Evaluation design:** Designing both numerical and domain-specific test cases to ensure the problem's validity (§2.3.2).
(3) **Problem validation:** Iterating on the problems through three rounds of revisions to further enhance question design (§2.3.3).

We now examine the design choices for each stage.

2.3.1 Problem Selection

Throughout the research project cycle, various coding needs arise, such as data processing, fitting, and plotting. To use SciCode, scientists select the problems that require intense scientific knowledge and reasoning to optimally test LM's science capability. This approach ensures that both the breadth and depth of frontier research are addressed. We focus on:

- **Numerical methods.** Analytical forms are usually impossible to achieve for very complicated systems. Therefore, scientists must derive numerical models and algorithms that describe physical phenomena [9, 74, 35, 77, 32], chemical reactions [22, 23, 87], biological systems [96, 97, 95, 83, 16, 15, 51, 89], or statistical behaviors[79, 69, 57, 55, 50, 24, 47].
- **Simulation of systems.** In fields of natural science, scientists write code to simulate systems and processes. These simulations are based on theoretical principles and empirical data,

reflecting deep scientific insights into the system being studied [76, 99, 62, 56, 73, 20, 90, 98, 38, 68, 41, 59, 71].
- **Scientific calculation.** During data post-processing and visualization, scientists often perform many transformations based on scientific formulas to get physical observable of interest instead of raw experimental data [10, 88, 30, 39, 40, 67, 5, 31].

We also include several research problems that are built upon or reproduce methods used in Nobel Prize-winning studies to highlight current trends in scientific research: the self-consistent field (SCF) method for density functional theory (DFT) calculations [37] **(The Nobel Prize in Chemistry 1998)**, the PMNS matrix for neutrino oscillation in matter [54, 61] **(The Nobel Prize in Physics 2015)**, the Haldane model for the anomalous quantum Hall effect [26] **(The Nobel Prize in Physics 2016)**, optical tweezer [46, 6] simulations for microscopic thermodynamics [50, 24, 47] **(The Nobel Prize in Physics 2018)**, and the replica method for spin glasses [79, 69, 57, 55] **(The Nobel Prize in Physics 2021)**.

2.3.2 Evaluation Design

To facilitate evaluation, we have scientist annotators use only widely adopted and well-documented packages such as NumPy, SciPy, and SymPy when writing the solution code for their problems, as shown in Figure 4.

Our test suite involves two key components. (1) **Numerical tests** list input-output pairs to check if the generated code produces the same outputs as ground truth. (2) **Domain-specific test cases**, introduced as an additional stage, evaluate whether model-generated solutions align with scientists' practical needs and further ensure the correctness and applicability of each solution within its specific field. These tests are extracted from real scientific workflows: scientists must design domain-specific test cases to verify code accuracy by reproducing results published in academic papers or matching analytical solutions derived from theoretical models. For example, we reproduce the phase transition at around $kT/J = 2.269$ for the 2D square Ising model problem [63], derive the surface plasmon mode in a 2D layered electron gas [10, 32], verify the ballistic Brownian motion in optical tweezer [46], etc. By doing so, we validate that the code not only functions correctly but also accurately represents the underlying scientific problem.

Overall, the evaluation design aims to balance the fidelity of the scientific problem with the practicality of the evaluation process, ensuring that the solutions are both accurate and accessible.

2.3.3 Problem Validation for Quality Control

We conduct three rounds of validation and revision for each problem:

(1) **In-domain scientist validation.** At least two scientists in the same research domain cross-check the *question design*, *solution code*, and *domain-specific test cases*, providing detailed feedback. The scientists who design the workflows iterate on them based on this feedback to ensure the problems are scientifically accurate.

(2) **Out-of-domain scientist validation.** One scientist from a different domain reviews the *question design* to ensure it is clear and that the information provided is precise and sufficient to solve the problem (e.g., all scientific constants are given). This helps to identify any assumptions that might be unclear to those outside the immediate field of study.

(3) **GPT-4 validation.** GPT-4 assists with the final review round. The previously validated sub-questions are input to GPT-4 to generate code solutions. Scientists perform error analysis for the generated solutions and redesign the *numerical test cases* if necessary to prevent false positives.Based on the code solutions from GPT-4, the scientist may also revise the *entire workflow* a third time to addressany potential ambiguity.

This multi-round validation approach ensures that the problems are scientifically rigorous, clear, and unambiguous, facilitating accurate and effective evaluation.

2.4 Various Types of Evaluations

SciCode offers unique opportunities for evaluating LMs across diverse settings, comprehensively testing their coding capabilities.

- **Without vs. with scientific background.** A subproblem can provide scientific background knowledge to guide LMs in solving the coding task. SciCode's scientific background for each problem offers two modes of evaluation. (1) When models are evaluated *without* scientific background, it tests their inherent scientific knowledge and reasoning along with their coding capability. (2) For models not designed to handle scientific problems, background provides the necessary knowledge and reasoning steps to solve the problems, shifting the evaluation's focus towards the models' coding and instruction-following capabilities. As we show in the experiments (§3), all models substantially improve performance when background is provided, indicating their lack of knowledge and reasoning capability in these natural science fields.
- **Gold vs. generated solutions to previous subproblems.** Each main problem in SciCode factorizes into multiple subproblems, and solutions to previous problems provide vital information for solving the current one. SciCode enables use of *gold* or *generated* solutions to previous subproblems. Gold solutions focus only on the current problem, while generated ones provide a more realistic evaluation setting and are more challenging due to error accumulation.
- **Main vs. subproblem levels.** (1) The LM is considered to have successfully solved the main problem when all subproblem solutions are correct and the integrated solution to the main problem is correct. (2) Alternatively, SciCode can assess at a subproblem level, evaluating a subproblem independently of other subproblems or its main problem.

Among these setups, evaluation *without background* carrying over *generated* solutions to previous problems is the closest to scientists' real use case of LMs. Therefore, we dub this the *standard setup*. Our experiments indicate that this setup is very challenging for even the best models available today: Claude3.5-Sonnet, the best performing one, can solve only 4.6% of the main problems.

To make SciCode useful for evaluating less capable or developing models, we also consider less challenging settings in our experiments.

3 Experiments

Prompts. We evaluate our model using zero-shot prompts. We keep the prompts general and design different ones for different evaluation setups only to inform the model about the tasks. We keep prompts the same across models and fields, and they contain the model's main and sub-problem instructions and code for previous subproblems. We also instruct the model to recall useful knowledge when gold background knowledge is not provided. §A.1 presents an example.

3.1 Evaluated Models

Since SciCode is a challenging benchmark, we mainly consider strong language models.[3]

- **GPT-4o** [65]: An optimized version of GPT-4 [64] by OpenAI with multi-modal capability.
- **GPT-4-Turbo**: A faster and more cost-effective variant of GPT-4 [64]. We use the 'gpt-4-turbo-2024-04-09' snapshot.
- **Claude3.5-Sonnet** [4]: The latest model from the Claude 3.5 family from Anthropic.
- **Claude3-Opus** [3]: The most capable model from the Claude 3 family from Anthropic.
- **Claude3-Sonnet** [3]: The second most capable model from the Claude 3 family.
- **Gemini 1.5 Pro** [85]: A model from the Gemini 1.5 family by Google and the largest with open access at the time of writing.
- **Llama-3-70B-Instruct** [2]: The instruction-tuned version of the largest available model from the Llama-3 family.
- **Mixtral-8x22B-Instruct** [33]: The instruction-tuned version of Mistral AI's largest publicly accessible Mixture-of-Expert Model.
- **Deepseek-Coder-v2** [102]: Mixture-of-Experts (MoE) code language model continue pre-trained on DeepSeek-V2
- **Qwen2-72B-Instruct** [86]: The largest instruction-tuned Qwen-2 model.

Models	Subproblem	Main Problem
Proprietary Models		
GPT-4o	25.0	1.5
GPT-4-Turbo	22.9	1.5
Claude3.5-Sonnet	**26.0**	**4.6**
Claude3-Opus	21.5	1.5
Claude3-Sonnet	17.0	1.5
Gemini 1.5 Pro	21.9	1.5
Open Models		
Llama-3-70B-Chat	14.6	0.0
Mixtral-8x22B-Instruct	16.3	0.0
Deepseek-Coder-v2	21.2	3.1
Qwen2-72B-Instruct	17.0	1.5

Table 2: Model performance in pass@1 rate on SciCode under the standard setup: without background knowledge and carrying over generated solutions to previous subproblems.

3.2 Main Results

Table 2 presents results under the *standard setup*.[4] For the easier subproblem-level evaluation, the state-of-the-art models we test solve 14%-26% of the subproblems. Among them, Claude3.5-Sonnet achieves the best performance, with a 26.0% pass@1 rate. However, all models perform much worse on the more realistic and challenging main problem evaluation. Claude3.5-Sonnet still performs the best in this setting, but with only a 4.6% pass@1 rate.

These results show that SciCode is a difficult benchmark for current LMs. Consistent with our observations on proprietary models, open-weight LMs under test also showed their lack of capabilities in solving any main problem despite being able to solve a number of sub-problems correctly.

3.3 Additional Results with Other Evaluation Settings

Providing gold scientific background knowledge. Table 3 presents results when background text authored by scientists is provided to the LMs and generated solutions to previous subproblems are used. This setting evaluates both the models' capabilities to faithfully follow the instructions provided in the background as well as their code-generation performance. The Δ columns indicate performance differences compared to the standard setup.

All models substantially improve performance for both subproblem and main problem evaluations when given scientific background knowledge. For the subproblem evaluation, Claude3.5-Sonnet and GPT-4o perform the best, both with a 35.4% pass@1 rate. GPT-4-Turbo benefits the most from the provided scientific background and reasoning with an increase of 10.8 %. Open models improve less compared to proprietary models which might indicate weaker Instruction following capability. Interestingly, the comparison between Llama-3-70B-Instruct and Mixtral-8x22B-Instruct reveals a trend that differs from the standard setup: Llama-3-70B-Instruct benefits more from the scientific background knowledge and reaches the performance of Mixtral-8x22B-Instruct in this setting.

For the main problem evaluation, the trend remains similar to the standard setup. Claude3.5-Sonnet performs best, with a 12.3% pass@1 rate, followed closely by GPT-4o and GPT-4-Turbo at 9.2%. GPT-4o, GPT-4-Turbo, and Claude3.5-Sonnet improve most from background content, at 7.7%. Nonetheless, all models still fall short of satisfactory performance even with the background knowledge provided. This reaffirms that SciCode is challenging even when focusing on code generation rather than testing the models' scientific knowledge.

[3] For instance, CodeLlama-7B-Instruct achieves only 0.4% pass@1 in our main setting.

[4] Without background and carrying over generated subproblem solutions. See §2.4 for a more detailed discussion.

Model	Subproblem Pass@1	 Δ	Main Problem Pass@1	 Δ
Proprietary Models				
GPT-4o	**35.4**	10.4	9.2	**7.7**
GPT-4-Turbo-2024-04-09	33.7	**10.8**	9.2	**7.7**
Claude3.5-Sonnet	**35.4**	9.4	**12.3**	**7.7**
Claude3-Opus	26.7	5.2	4.7	3.0
Claude3-Sonnet	25.7	8.7	4.7	3.0
Gemini 1.5 Pro	30.6	8.7	7.7	6.2
Open Models				
Llama-3-70B-Instruct	20.8	6.3	1.5	1.5
Mixtral-8x22B-Instruct	20.8	4.5	3.1	3.1
Deepseek-Coder-v2	27.1	5.9	4.6	1.5
Qwen2-72B-Instruct	22.2	5.2	4.6	3.1

Table 3: Pass@1 with generated solutions for previous subproblems and scientific background texts provided. The Δ columns show the performance differences compared to the *standard setting* in Table 2, i.e., where background content is *not* provided.

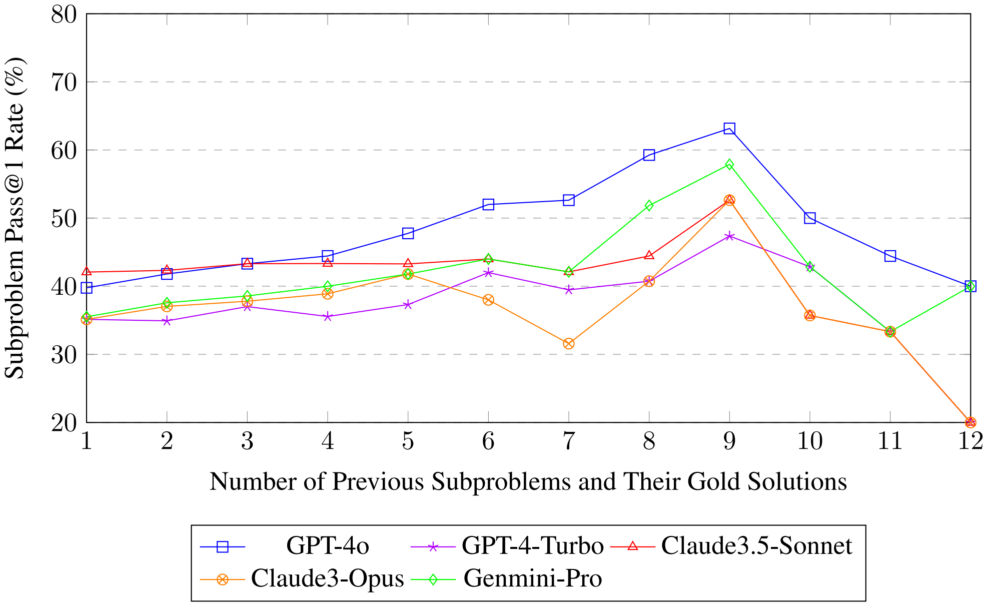


Figure 3: Subproblem pass@1 rate when conditioning on various numbers of previous subproblems and their gold solutions. Background knowledge is *not* provided.

With gold subproblem solutions. Figure 3 plots the subproblem pass@1 rates conditioning on various numbers of previous subproblems and their gold solutions. Background knowledge is *not* provided. The intuition behind this analysis is that later steps can leverage gold solutions from previous steps to gain a richer understanding of the problem. Instructions and solutions from earlier steps serve as in-context demonstrations, enabling the model to rely less on its instruction-following capability. By focusing on later steps, we can more precisely assess the models' inherent capabilities.

Overall, all three models show similar trends, with their performance generally improving as they condition on more gold solutions from previous steps. However, there is a notable exception when conditioning on 7 previous gold subproblem solutions. Additionally, performance starts to decline when models condition on more than 9 previous solutions, possibly due to the increased difficulty of managing long contexts.

4 Related Work

Language models for code. Code has long been an active field of research, and code LMs have co-evolved with foundation LMs since the era of BERT [17]. Earlier works include CodeBert [19] and CodeT5 [94], while Codex [12] arguably kick-started the LLM era for code-generation models. Since Codex, the field has experienced rapid growth in quantity and quality of large code generation models, including specially trained models like Codegen [60], StarCoder models [45, 52], and generalist models with code adapation [12] such as CodeLlama [75], CodeQwen [8], and DeepSeek-Coder [25]. As code generation gains more attention and becomes increasingly useful, contemporary generalist models often include non-trivial coding capabilities [66, 85].

Evaluating code generation. Before the emergence of very capable code synthesis models, when most models struggled to produce executable code, datasets like CoNaLa typically included n-gram-based metrics [100]. Soon after model capabilities improved, execution-based evaluation gained in popularity [28, 7, 11]. While n-gram or general text-based evaluation still exists, we opted to omit them from SciCode due to obvious limitations of surface form matching in scientific coding.

Code generation benchmarks now take various forms. For simple function completion, MBPP [7] and HumanEval [11] are two widely used benchmarks that contain basic programming questions, mainly evaluating LMs' ability to turn natural language instructions into Python programs. Other benchmarks assess the models' competence in real-world programming scenarios, such as writing data science code [42, 101], repository-level code completion [18], and more complex tasks in real-world software engineering [34]. Though our work is similar to MTPB [60] in terms of using a multi-turn setup, our subproblem instructions correspond to a high-level task, while theirs correspond to specific code actions (e.g., replace X with Y in the string).

Language models for science. Scientific tasks are complex due to their demands for reasoning and knowledge. However, Recent advances in general and specialized language models have revolutionized the processing of text and other data modalities, such as molecules and proteins, in scientific fields. Galactica [84], a general-purpose scientific model, can perform tasks like citation prediction, scientific reasoning, document generation, and molecular property prediction. Many models focus on one single domain or task, like math (e.g., Minerva [43] and Deepseek-Math [78]), protein structure prediction (e.g., ESM-2 [48]), medical reasoning (e.g., Med-PaLM [80], BioGPT [53]), and others.

5 Conclusion

We introduce SciCode, a scientific research benchmark curated by professional natural scientists. We designed SciCode for scientific problem evaluation and collected problems representing 16 diverse domains. By assessing SciCode with ten contemporary state-of-the-art AI models, we demonstrated that our benchmark is within reach but remains very challenging. We believe SciCode will serve as a helpful guideline for building future code language models for varied scientific applications.

References

[1] M. R. AI4Science and M. A. Quantum. The impact of large language models on scientific discovery: a preliminary study using gpt-4, 2023.

[2] AI@Meta. Llama 3 model card. 2024.

[3] Anthropic. Introducing the next generation of claude, 2024. https://www.anthropic.com/news/claude-3-family.

[4] Anthropic. Introducing the next generation of claude 3.5, 2024. https://www.anthropic.com/news/claude-3-5-sonnet.

[5] G. Ashiotis, A. Deschildre, Z. Nawaz, J. P. Wright, D. Karkoulis, F. E. Picca, and J. Kieffer. The fast azimuthal integration python library: pyfai. *Journal of applied crystallography*, 48(2):510–519, 2015.

[6] A. Ashkin. Acceleration and trapping of particles by radiation pressure. *Physical review letters*, 24(4):156, 1970.

[7] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, and C. Sutton. Program synthesis with large language models, 2021.

[8] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji, M. Li, J. Lin, R. Lin, D. Liu, G. Liu, C. Lu, K. Lu, J. Ma, R. Men, X. Ren, X. Ren, C. Tan, S. Tan, J. Tu, P. Wang, S. Wang, W. Wang, S. Wu, B. Xu, J. Xu, A. Yang, H. Yang, J. Yang, S. Yang, Y. Yao, B. Yu, H. Yuan, Z. Yuan, J. Zhang, X. Zhang, Y. Zhang, Z. Zhang, C. Zhou, J. Zhou, X. Zhou, and T. Zhu. Qwen technical report, 2023.

[9] Y. Bazilevs and T. J. R. Hughes. Weak imposition of dirichlet boundary conditions in fluid mechanics. *Computers & Fluids*, 36:12–26, 2007.

[10] J. Chen, X. Guo, C. Boyd, S. Bettler, C. Kengle, D. Chaudhuri, F. Hoveyda, A. Husain, J. Schneeloch, G. Gu, et al. Consistency between reflection momentum-resolved electron energy loss spectroscopy and optical spectroscopy measurements of the long-wavelength density response of bi 2 sr 2 cacu 2 o 8+ x. *Physical Review B*, 109(4):045108, 2024.

[11] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code, 2021.

[12] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code, 2021.

[13] Z. Chen, A. H. Cano, A. Romanou, A. Bonnet, K. Matoba, F. Salvi, M. Pagliardini, S. Fan, A. Köpf, A. Mohtashami, A. Sallinen, A. Sakhaeirad, V. Swamy, I. Krawczuk, D. Bayazit, A. Marmet, S. Montariol, M.-A. Hartley, M. Jaggi, and A. Bosselut. Meditron-70b: Scaling medical pretraining for large language models, 2023.

[14] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. volume abs/2110.14168, 2021.

[15] M. Cross and H. Greenside. *Pattern Formation and Dynamics in Nonequilibrium Systems*. Cambridge University Press, 2009.

[16] M. Cross and P. Hohenberg. Pattern formation outside of equilibrium. *Reviews of Modern Physics*, 65:851–1112, 1993.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[18] Y. Ding, Z. Wang, W. U. Ahmad, H. Ding, M. Tan, N. Jain, M. K. Ramanathan, R. Nallapati, P. Bhatia, D. Roth, and B. Xiang. Crosscodeeval: A diverse and multilingual benchmark for cross-file code completion. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[19] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, and M. Zhou. Codebert: A pre-trained model for programming and natural languages, 2020.

[20] W. M. Foulkes, L. Mitas, R. Needs, and G. Rajagopal. Quantum monte carlo simulations of solids. *Reviews of Modern Physics*, 73(1):33, 2001.

[21] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, and G. Neubig. Pal: program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

[22] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4):403–434, 1976.

[23] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25), 1977.

[24] N. A. Gumerov and R. Duraiswami. Recursions for the computation of multipole translation and rotation coefficients for the 3-d helmholtz equation. *SIAM Journal on Scientific Computing*, 25(4):1344–1381, 2004.

[25] D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. K. Li, F. Luo, Y. Xiong, and W. Liang. Deepseek-coder: When the large language model meets programming – the rise of code intelligence, 2024.

[26] F. D. M. Haldane. Model for a quantum hall effect without landau levels: Condensed-matter realization of the" parity anomaly". *Physical review letters*, 61(18):2015, 1988.

[27] D. Hendrycks, S. Basart, S. Kadavath, M. Mazeika, A. Arora, E. Guo, C. Burns, S. Puranik, H. He, D. Song, et al. Measuring coding challenge competence with apps. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[28] D. Hendrycks, S. Basart, S. Kadavath, M. Mazeika, A. Arora, E. Guo, C. Burns, S. Puranik, H. He, D. Song, and J. Steinhardt. Measuring coding challenge competence with apps. *NeurIPS*, 2021.

[29] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[30] A. A. Husain, E. W. Huang, M. Mitrano, M. S. Rak, S. I. Rubeck, X. Guo, H. Yang, C. Sow, Y. Maeno, B. Uchoa, et al. Pines' demon observed as a 3d acoustic plasmon in sr2ruo4. *Nature*, 621(7977):66–70, 2023.

[31] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[32] J. K. Jain and P. B. Allen. Dielectric response of a semi-infinite layered electron gas and raman scattering from its bulk and surface plasmons. *Physical Review B*, 32(2):997, 1985.

[33] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts, 2024.

[34] C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan. Swe-bench: Can language models resolve real-world github issues?, 2024.

[35] S. Khatri, K. Sharma, and M. M. Wilde. Information-theoretic aspects of the generalized amplitude-damping channel. *Physical Review A*, 102(1):012401, 2020.

[36] D. Kiela, M. Bartolo, Y. Nie, D. Kaushik, A. Geiger, Z. Wu, B. Vidgen, G. Prasad, A. Singh, P. Ringshia, Z. Ma, T. Thrush, S. Riedel, Z. Waseem, P. Stenetorp, R. Jia, M. Bansal, C. Potts, and A. Williams. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[37] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. *Physical review*, 140(4A):A1133, 1965.

[38] A. N. Kolmogorov and V. H. Crespi. Registry-dependent interlayer potential for graphitic systems. *Physical Review B*, 71(23):235415, 2005.

[39] M. J. Krogstad. *Diffuse Scattering and Local Order in Lead-Based Relaxor Ferroelectrics*. Northern Illinois University, 2018.

[40] M. J. Krogstad, S. Rosenkranz, J. M. Wozniak, G. Jennings, J. P. Ruff, J. T. Vaughey, and R. Osborn. Reciprocal space imaging of ionic correlations in intercalation compounds. *Nature materials*, 19(1):63–68, 2020.

[41] K. Krongchon, T. Rakib, S. Pathak, E. Ertekin, H. T. Johnson, and L. K. Wagner. Registry-dependent potential energy and lattice corrugation of twisted bilayer graphene from quantum monte carlo. *Physical Review B*, 108(23):235403, 2023.

[42] Y. Lai, C. Li, Y. Wang, T. Zhang, R. Zhong, L. Zettlemoyer, S. W. tau Yih, D. Fried, S. Wang, and T. Yu. Ds-1000: A natural and reliable benchmark for data science code generation, 2022.

[43] A. Lewkowycz, A. Andreassen, D. Dohan, E. Dyer, H. Michalewski, V. Ramasesh, A. Slone, C. Anil, I. Schlag, T. Gutman-Solo, Y. Wu, B. Neyshabur, G. Gur-Ari, and V. Misra. Solving quantitative reasoning problems with language models, 2022.

[44] Q. Li, L. Cui, X. Zhao, L. Kong, and W. Bi. Gsm-plus: A comprehensive benchmark for evaluating the robustness of llms as mathematical problem solvers. *ArXiv preprint*, abs/2402.19255, 2024.

[45] R. Li, L. B. allal, Y. Zi, N. Muennighoff, D. Kocetkov, C. Mou, M. Marone, C. Akiki, J. LI, J. Chim, Q. Liu, E. Zheltonozhskii, T. Y. Zhuo, T. Wang, O. Dehaene, J. Lamy-Poirier, J. Monteiro, N. Gontier, M.-H. Yee, L. K. Umapathi, J. Zhu, B. Lipkin, M. Oblokulov, Z. Wang, R. Murthy, J. T. Stillerman, S. S. Patel, D. Abulkhanov, M. Zocca, M. Dey, Z. Zhang, U. Bhattacharyya, W. Yu, S. Luccioni, P. Villegas, F. Zhdanov, T. Lee, N. Timor, J. Ding, C. S. Schlesinger, H. Schoelkopf, J. Ebert, T. Dao, M. Mishra, A. Gu, C. J. Anderson, B. Dolan-Gavitt, D. Contractor, S. Reddy, D. Fried, D. Bahdanau, Y. Jernite, C. M. Ferrandis, S. Hughes, T. Wolf, A. Guha, L. V. Werra, and H. de Vries. Starcoder: may the source be with you! *Transactions on Machine Learning Research*, 2023. Reproducibility Certification.

[46] T. Li, S. Kheifets, D. Medellin, and M. G. Raizen. Measurement of the instantaneous velocity of a brownian particle. *Science*, 328(5986):1673–1675, 2010.

[47] T. Li and M. G. Raizen. Brownian motion at short time scales. *Annalen der Physik*, 525(4):281–295, 2013.

[48] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, A. dos Santos Costa, M. Fazel-Zarandi, T. Sercu, S. Candido, et al. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022.

[49] J. Liu, C. S. Xia, Y. Wang, and L. Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation, 2023.

[50] S. Liu, Z.-q. Yin, and T. Li. Prethermalization and nonreciprocal phonon transport in a levitated optomechanical array. *Advanced Quantum Technologies*, 3(3):1900099, 2020.

[51] A. J. Lotka. *Elements of physical biology*. Williams & Wilkins, 1925.

[52] A. Lozhkov, R. Li, L. B. Allal, F. Cassano, J. Lamy-Poirier, N. Tazi, A. Tang, D. Pykhtar, J. Liu, Y. Wei, T. Liu, M. Tian, D. Kocetkov, A. Zucker, Y. Belkada, Z. Wang, Q. Liu, D. Abulkhanov, I. Paul, Z. Li, W.-D. Li, M. Risdal, J. Li, J. Zhu, T. Y. Zhuo, E. Zheltonozhskii, N. O. O. Dade, W. Yu, L. Krauß, N. Jain, Y. Su, X. He, M. Dey, E. Abati, Y. Chai, N. Muennighoff, X. Tang, M. Oblokulov, C. Akiki, M. Marone, C. Mou, M. Mishra, A. Gu, B. Hui, T. Dao, A. Zebaze, O. Dehaene, N. Patry, C. Xu, J. McAuley, H. Hu, T. Scholak, S. Paquet, J. Robinson, C. J. Anderson, N. Chapados, M. Patwary, N. Tajbakhsh, Y. Jernite, C. M. Ferrandis, L. Zhang, S. Hughes, T. Wolf, A. Guha, L. von Werra, and H. de Vries. Starcoder 2 and the stack v2: The next generation, 2024.

[53] R. Luo, L. Sun, Y. Xia, T. Qin, S. Zhang, H. Poon, and T.-Y. Liu. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23, 09 2022.

[54] Z. Maki, M. Nakagawa, and S. Sakata. Remarks on the unified model of elementary particles. *Progress of Theoretical Physics*, 28(5):870–880, 1962.

[55] E. Marinari, G. Parisi, and J. Ruiz-Lorenzo. Numerical simulations of spin glass systems. In *Spin Glasses and Random Fields*, pages 59–98. World Scientific, 1998.

[56] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[57] M. Mézard, G. Parisi, and M. A. Virasoro. *Spin glass theory and beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987.

[58] S.-y. Miao, C.-C. Liang, and K.-Y. Su. A diverse corpus for evaluating and developing English math word problem solvers. In *Proc. of ACL*, 2020.

[59] P. Moon and M. Koshino. Energy spectrum and quantum hall effect in twisted bilayer graphene. *Physical Review B*, 85(19):195458, 2012.

[60] E. Nijkamp, B. Pang, H. Hayashi, L. Tu, H. Wang, Y. Zhou, S. Savarese, and C. Xiong. Codegen: An open large language model for code with multi-turn program synthesis, 2023.

[61] T. Ohlsson and H. Snellman. Three flavor neutrino oscillations in matter. *Journal of Mathematical Physics*, 41(5):2768–2788, 2000.

[62] L. Onsager. Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Physical Review*, 65(3-4):117, 1944.

[63] L. Onsager. Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Phys. Rev.*, 65:117–149, Feb 1944.

[64] OpenAI. Gpt-4 technical report, 2023.

[65] OpenAI. Hello gpt-4o, 2024. https://openai.com/index/hello-gpt-4o/.

[66] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess,

C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph. Gpt-4 technical report, 2024.

[67] R. Osborn. Mapping structural correlations in real space. *Acta Crystallographica Section B*, 79(2):99–100, Apr 2023.

[68] W. Ouyang, D. Mandelli, M. Urbakh, and O. Hod. Nanoserpents: Graphene nanoribbon motion on two-dimensional hexagonal materials. *Nano letters*, 18(9):6009–6016, 2018.

[69] G. Parisi. A sequence of approximated solutions to the sk model for spin glasses. *Journal of Physics A Mathematical General*, 13(4):L115–L121, 1980.

[70] A. Patel, S. Bhattamishra, and N. Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[71] S. Pathak, T. Rakib, R. Hou, A. Nevidomskyy, E. Ertekin, H. T. Johnson, and L. K. Wagner. Accurate tight-binding model for twisted bilayer graphene describes topological flat bands without geometric relaxation. *Physical Review B*, 105(11):115141, 2022.

[72] D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman. Gpqa: A graduate-level google-proof qa benchmark, 2023.

[73] P. J. Reynolds, D. M. Ceperley, B. J. Alder, and W. A. Lester Jr. Fixed-node quantum monte carlo for molecules. *The Journal of Chemical Physics*, 77(11):5593–5603, 1982.

[74] M. Rezaeian and A. Grant. Computation of total capacity for discrete memoryless multiple-access channels. *IEEE transactions on information theory*, 50(11):2779–2784, 2004.

[75] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. C. Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom, and G. Synnaeve. Code llama: Open foundation models for code, 2024.

[76] I. Ruff, A. Baranyai, G. Pálinkás, and K. Heinzinger. Grand canonical Monte Carlo simulation of liquid argon. *The Journal of Chemical Physics*, 85(4):2169–2177, 08 1986.

[77] L. Schatzki, L. Ma, E. Solomonik, and E. Chitambar. Tensor rank and other multipartite entanglement measures of graph states. *arXiv preprint arXiv:2209.06320*, 2022.

[78] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. Li, Y. Wu, and D. Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.

[79] D. Sherrington and S. Kirkpatrick. Solvable model of a spin-glass. *Physical Review Letters*, 35(26):1792–1796, 1975.

[80] K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, P. Payne, M. Seneviratne, P. Gamble, C. Kelly, A. Babiker, N. Schärli, A. Chowdhery, P. Mansfield, D. Demner-Fushman, B. Agüera y Arcas, D. Webster, G. S. Corrado, Y. Matias, K. Chou, J. Gottweis, N. Tomasev, Y. Liu, A. Rajkomar, J. Barral, C. Semturs, A. Karthikesalingam, and V. Natarajan. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, Aug 2023.

[81] A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, A. Kluska, A. Lewkowycz, A. Agarwal, A. Power, A. Ray, A. Warstadt, A. W. Kocurek, A. Safaya, A. Tazarv, A. Xiang, A. Parrish, A. Nie, A. Hussain, A. Askell, A. Dsouza, A. Slone, A. Rahane, A. S. Iyer, A. Andreassen, A. Madotto, A. Santilli, A. Stuhlmüller, A. Dai, A. La, A. Lampinen, A. Zou, A. Jiang, A. Chen, A. Vuong, A. Gupta, A. Gottardi, A. Norelli, A. Venkatesh, A. Gholamidavoodi, A. Tabassum, A. Menezes, A. Kirubarajan, A. Mullokandov, A. Sabharwal, A. Herrick, A. Efrat, A. Erdem, A. Karakaş, B. R. Roberts, B. S. Loe, B. Zoph, B. Bojanowski, B. Özyurt, B. Hedayatnia, B. Neyshabur, B. Inden, B. Stein, B. Ekmekci, B. Y. Lin, B. Howald, B. Orinion, C. Diao, C. Dour, C. Stinson, C. Argueta, C. F. Ramírez, C. Singh, C. Rathkopf, C. Meng, C. Baral, C. Wu, C. Callison-Burch, C. Waites, C. Voigt, C. D. Manning, C. Potts, C. Ramirez, C. E. Rivera, C. Siro, C. Raffel, C. Ashcraft, C. Garbacea, D. Sileo, D. Garrette, D. Hendrycks, D. Kilman, D. Roth, D. Freeman, D. Khashabi, D. Levy, D. M. González, D. Perszyk, D. Hernandez, D. Chen, D. Ippolito, D. Gilboa, D. Dohan, D. Drakard, D. Jurgens, D. Datta, D. Ganguli, D. Emelin, D. Kleyko, D. Yuret, D. Chen, D. Tam, D. Hupkes, D. Misra, D. Buzan, D. C. Mollo, D. Yang, D.-H. Lee, D. Schrader, E. Shutova, E. D. Cubuk, E. Segal, E. Hagerman, E. Barnes, E. Donoway, E. Pavlick, E. Rodola, E. Lam, E. Chu, E. Tang, E. Erdem, E. Chang, E. A. Chi, E. Dyer, E. Jerzak, E. Kim, E. E. Manyasi, E. Zheltonozhskii, F. Xia, F. Siar, F. Martínez-Plumed, F. Happé, F. Chollet, F. Rong, G. Mishra, G. I. Winata, G. de Melo, G. Kruszewski, G. Parascandolo, G. Mariani, G. Wang, G. Jaimovitch-López, G. Betz, G. Gur-Ari, H. Galijasevic, H. Kim, H. Rashkin, H. Hajishirzi, H. Mehta, H. Bogar, H. Shevlin, H. Schütze, H. Yakura, H. Zhang, H. M. Wong, I. Ng, I. Noble, J. Jumelet, J. Geissinger, J. Kernion, J. Hilton, J. Lee, J. F. Fisac, J. B. Simon, J. Koppel, J. Zheng, J. Zou, J. Kocoń, J. Thompson, J. Wingfield, J. Kaplan, J. Radom, J. Sohl-Dickstein, J. Phang, J. Wei, J. Yosinski, J. Novikova, J. Bosscher, J. Marsh, J. Kim, J. Taal, J. Engel, J. Alabi, J. Xu, J. Song, J. Tang, J. Waweru, J. Burden, J. Miller, J. U. Balis, J. Batchelder, J. Berant, J. Frohberg, J. Rozen, J. Hernandez-Orallo, J. Boudeman, J. Guerr, J. Jones, J. B. Tenenbaum, J. S. Rule, J. Chua, K. Kanclerz, K. Livescu, K. Krauth, K. Gopalakrishnan, K. Ignatyeva, K. Markert, K. D. Dhole, K. Gimpel, K. Omondi, K. Mathewson, K. Chiafullo, K. Shkaruta, K. Shridhar, K. McDonell, K. Richardson, L. Reynolds, L. Gao, L. Zhang, L. Dugan, L. Qin, L. Contreras-Ochando, L.-P. Morency, L. Moschella, L. Lam, L. Noble, L. Schmidt, L. He, L. O. Colón, L. Metz, L. K. Şenel, M. Bosma, M. Sap, M. ter Hoeve, M. Farooqi, M. Faruqui, M. Mazeika, M. Baturan, M. Marelli, M. Maru, M. J. R. Quintana, M. Tolkiehn, M. Giulianelli, M. Lewis, M. Potthast, M. L. Leavitt, M. Hagen, M. Schubert, M. O. Baitemirova, M. Arnaud, M. McElrath, M. A. Yee, M. Cohen, M. Gu, M. Ivanitskiy, M. Starritt, M. Strube, M. Swędrowski, M. Bevilacqua, M. Yasunaga, M. Kale, M. Cain, M. Xu, M. Suzgun, M. Walker, M. Tiwari, M. Bansal, M. Aminnaseri, M. Geva, M. Gheini, M. V. T, N. Peng, N. A. Chi, N. Lee, N. G.-A. Krakover, N. Cameron, N. Roberts, N. Doiron, N. Martinez, N. Nangia, N. Deckers, N. Muennighoff, N. S. Keskar, N. S. Iyer, N. Constant, N. Fiedel, N. Wen, O. Zhang, O. Agha, O. Elbaghdadi, O. Levy, O. Evans, P. A. M. Casares, P. Doshi, P. Fung, P. P. Liang, P. Vicol, P. Alipoormolabashi, P. Liao, P. Liang, P. Chang, P. Eckersley, P. M. Htut, P. Hwang, P. Miłkowski, P. Patil, P. Pezeshkpour, P. Oli,

Q. Mei, Q. Lyu, Q. Chen, R. Banjade, R. E. Rudolph, R. Gabriel, R. Habacker, R. Risco, R. Millière, R. Garg, R. Barnes, R. A. Saurous, R. Arakawa, R. Raymaekers, R. Frank, R. Sikand, R. Novak, R. Sitelew, R. LeBras, R. Liu, R. Jacobs, R. Zhang, R. Salakhutdinov, R. Chi, R. Lee, R. Stovall, R. Teehan, R. Yang, S. Singh, S. M. Mohammad, S. Anand, S. Dillavou, S. Shleifer, S. Wiseman, S. Gruetter, S. R. Bowman, S. S. Schoenholz, S. Han, S. Kwatra, S. A. Rous, S. Ghazarian, S. Ghosh, S. Casey, S. Bischoff, S. Gehrmann, S. Schuster, S. Sadeghi, S. Hamdan, S. Zhou, S. Srivastava, S. Shi, S. Singh, S. Asaadi, S. S. Gu, S. Pachchigar, S. Toshniwal, S. Upadhyay, Shyamolima, Debnath, S. Shakeri, S. Thormeyer, S. Melzi, S. Reddy, S. P. Makini, S.-H. Lee, S. Torene, S. Hatwar, S. Dehaene, S. Divic, S. Ermon, S. Biderman, S. Lin, S. Prasad, S. T. Piantadosi, S. M. Shieber, S. Misherghi, S. Kiritchenko, S. Mishra, T. Linzen, T. Schuster, T. Li, T. Yu, T. Ali, T. Hashimoto, T.-L. Wu, T. Desbordes, T. Rothschild, T. Phan, T. Wang, T. Nkinyili, T. Schick, T. Kornev, T. Tunduny, T. Gerstenberg, T. Chang, T. Neeraj, T. Khot, T. Shultz, U. Shaham, V. Misra, V. Demberg, V. Nyamai, V. Raunak, V. Ramasesh, V. U. Prabhu, V. Padmakumar, V. Srikumar, W. Fedus, W. Saunders, W. Zhang, W. Vossen, X. Ren, X. Tong, X. Zhao, X. Wu, X. Shen, Y. Yaghoobzadeh, Y. Lakretz, Y. Song, Y. Bahri, Y. Choi, Y. Yang, Y. Hao, Y. Chen, Y. Belinkov, Y. Hou, Y. Hou, Y. Bai, Z. Seid, Z. Zhao, Z. Wang, Z. J. Wang, Z. Wang, and Z. Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023.

[82] M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, , and J. Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. *ArXiv preprint*, abs/2210.09261, 2022.

[83] J. Swift and P. Hohenberg. Hydrodynamic fluctuations at the convective instability. *Physical Review A*, 15:319–328, 1977.

[84] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic. Galactica: A large language model for science, 2022.

[85] G. Team, M. Reid, N. Savinov, D. Teplyashin, Dmitry, Lepikhin, T. Lillicrap, J. baptiste Alayrac, R. Soricut, A. Lazaridou, O. Firat, J. Schrittwieser, I. Antonoglou, R. Anil, S. Borgeaud, A. Dai, K. Millican, E. Dyer, M. Glaese, T. Sottiaux, B. Lee, F. Viola, M. Reynolds, Y. Xu, J. Molloy, J. Chen, M. Isard, P. Barham, T. Hennigan, R. McIlroy, M. Johnson, J. Schalkwyk, E. Collins, E. Rutherford, E. Moreira, K. Ayoub, M. Goel, C. Meyer, G. Thornton, Z. Yang, H. Michalewski, Z. Abbas, N. Schucher, A. Anand, R. Ives, J. Keeling, K. Lenc, S. Haykal, S. Shakeri, P. Shyam, A. Chowdhery, R. Ring, S. Spencer, E. Sezener, L. Vilnis, O. Chang, N. Morioka, G. Tucker, C. Zheng, O. Woodman, N. Attaluri, T. Kocisky, E. Eltyshev, X. Chen, T. Chung, V. Selo, S. Brahma, P. Georgiev, A. Slone, Z. Zhu, J. Lottes, S. Qiao, B. Caine, S. Riedel, A. Tomala, M. Chadwick, J. Love, P. Choy, S. Mittal, N. Houlsby, Y. Tang, M. Lamm, L. Bai, Q. Zhang, L. He, Y. Cheng, P. Humphreys, Y. Li, S. Brin, A. Cassirer, Y. Miao, L. Zilka, T. Tobin, K. Xu, L. Proleev, D. Sohn, A. Magni, L. A. Hendricks, I. Gao, S. Ontanon, O. Bunyan, N. Byrd, A. Sharma, B. Zhang, M. Pinto, R. Sinha, H. Mehta, D. Jia, S. Caelles, A. Webson, A. Morris, B. Roelofs, Y. Ding, R. Strudel, X. Xiong, M. Ritter, M. Dehghani, R. Chaabouni, A. Karmarkar, G. Lai, F. Mentzer, B. Xu, Y. Li, Y. Zhang, T. L. Paine, A. Goldin, B. Neyshabur, K. Baumli, A. Levskaya, M. Laskin, W. Jia, J. W. Rae, K. Xiao, A. He, S. Giordano, L. Yagati, J.-B. Lespiau, P. Natsev, S. Ganapathy, F. Liu, D. Martins, N. Chen, Y. Xu, M. Barnes, R. May, A. Vezer, J. Oh, K. Franko, S. Bridgers, R. Zhao, B. Wu, B. Mustafa, S. Sechrist, E. Parisotto, T. S. Pillai, C. Larkin, C. Gu, C. Sorokin, M. Krikun, A. Guseynov, J. Landon, R. Datta, A. Pritzel, P. Thacker, F. Yang, K. Hui, A. Hauth, C.-K. Yeh, D. Barker, J. Mao-Jones, S. Austin, H. Sheahan, P. Schuh, J. Svensson, R. Jain, V. Ramasesh, A. Briukhov, D.-W. Chung, T. von Glehn, C. Butterfield, P. Jhakra, M. Wiethoff, J. Frye, J. Grimstad, B. Changpinyo, C. L. Lan, A. Bortsova, Y. Wu, P. Voigtlaender, T. Sainath, S. Gu, C. Smith, W. Hawkins, K. Cao, J. Besley, S. Srinivasan, M. Omernick, C. Gaffney, G. Surita, R. Burnell, B. Damoc, J. Ahn, A. Brock, M. Pajarskas, A. Petrushkina, S. Noury, L. Blanco, K. Swersky, A. Ahuja, T. Avrahami, V. Misra, R. de Liedekerke, M. Iinuma, A. Polozov, S. York, G. van den Driessche, P. Michel, J. Chiu, R. Blevins, Z. Gleicher, A. Recasens, A. Rrustemi, E. Gribovskaya, A. Roy, W. Gworek, S. M. R. Arnold, L. Lee, J. Lee-Thorp, M. Maggioni, E. Piqueras, K. Badola, S. Vikram, L. Gonzalez, A. Baddepudi, E. Senter, J. Devlin, J. Qin, M. Azzam, M. Trebacz, M. Polacek, K. Krishnakumar, S. yiin Chang, M. Tung, I. Penchev, R. Joshi, K. Olszewska, C. Muir, M. Wirth, A. J. Hartman, J. Newlan,

S. Kashem, V. Bolina, E. Dabir, J. van Amersfoort, Z. Ahmed, J. Cobon-Kerr, A. Kamath, A. M. Hrafnkelsson, L. Hou, I. Mackinnon, A. Frechette, E. Noland, X. Si, E. Taropa, D. Li, P. Crone, A. Gulati, S. Cevey, J. Adler, A. Ma, D. Silver, S. Tokumine, R. Powell, S. Lee, K. Vodrahalli, S. Hassan, D. Mincu, A. Yang, N. Levine, J. Brennan, M. Wang, S. Hodkinson, J. Zhao, J. Lipschultz, A. Pope, M. B. Chang, C. Li, L. E. Shafey, M. Paganini, S. Douglas, B. Bohnet, F. Pardo, S. Odoom, M. Rosca, C. N. dos Santos, K. Soparkar, A. Guez, T. Hudson, S. Hansen, C. Asawaroengchai, R. Addanki, T. Yu, W. Stokowiec, M. Khan, J. Gilmer, J. Lee, C. G. Bostock, K. Rong, J. Caton, P. Pejman, F. Pavetic, G. Brown, V. Sharma, M. Lučić, R. Samuel, J. Djolonga, A. Mandhane, L. L. Sjösund, E. Buchatskaya, E. White, N. Clay, J. Jiang, H. Lim, R. Hemsley, Z. Cankara, J. Labanowski, N. D. Cao, D. Steiner, S. H. Hashemi, J. Austin, A. Gergely, T. Blyth, J. Stanton, K. Shivakumar, A. Siddhant, A. Andreassen, C. Araya, N. Sethi, R. Shivanna, S. Hand, A. Bapna, A. Khodaei, A. Miech, G. Tanzer, A. Swing, S. Thakoor, L. Aroyo, Z. Pan, Z. Nado, J. Sygnowski, S. Winkler, D. Yu, M. Saleh, L. Maggiore, Y. Bansal, X. Garcia, M. Kazemi, P. Patil, I. Dasgupta, I. Barr, M. Giang, T. Kagohara, I. Danihelka, A. Marathe, V. Feinberg, M. Elhawaty, N. Ghelani, D. Horgan, H. Miller, L. Walker, R. Tanburn, M. Tariq, D. Shrivastava, F. Xia, Q. Wang, C.-C. Chiu, Z. Ashwood, K. Baatarsukh, S. Samangooei, R. L. Kaufman, F. Alcober, A. Stjerngren, P. Komarek, K. Tsihlas, A. Boral, R. Comanescu, J. Chen, R. Liu, C. Welty, D. Bloxwich, C. Chen, Y. Sun, F. Feng, M. Mauger, X. Dotiwalla, V. Hellendoorn, M. Sharman, I. Zheng, K. Haridasan, G. Barth-Maron, C. Swanson, D. Rogozińska, A. Andreev, P. K. Rubenstein, R. Sang, D. Hurt, G. Elsayed, R. Wang, D. Lacey, A. Ilić, Y. Zhao, A. Iwanicki, A. Lince, A. Chen, C. Lyu, C. Lebsack, J. Griffith, M. Gaba, P. Sandhu, P. Chen, A. Koop, R. Rajwar, S. H. Yeganeh, S. Chang, R. Zhu, S. Radpour, E. Davoodi, V. I. Lei, Y. Xu, D. Toyama, C. Segal, M. Wicke, H. Lin, A. Bulanova, A. P. Badia, N. Rakićević, P. Sprechmann, A. Filos, S. Hou, V. Campos, N. Kassner, D. Sachan, M. Fortunato, C. Iwuanyanwu, V. Nikolaev, B. Lakshminarayanan, S. Jazayeri, M. Varadarajan, C. Tekur, D. Fritz, M. Khalman, D. Reitter, K. Dasgupta, S. Sarcar, T. Ornduff, J. Snaider, F. Huot, J. Jia, R. Kemp, N. Trdin, A. Vijayakumar, L. Kim, C. Angermueller, L. Lao, T. Liu, H. Zhang, D. Engel, S. Greene, A. White, J. Austin, L. Taylor, S. Ashraf, D. Liu, M. Georgaki, I. Cai, Y. Kulizhskaya, S. Goenka, B. Saeta, Y. Xu, C. Frank, D. de Cesare, B. Robenek, H. Richardson, M. Alnahlawi, C. Yew, P. Ponnapalli, M. Tagliasacchi, A. Korchemniy, Y. Kim, D. Li, B. Rosgen, K. Levin, J. Wiesner, P. Banzal, P. Srinivasan, H. Yu, Çağlar Ünlü, D. Reid, Z. Tung, D. Finchelstein, R. Kumar, A. Elisseeff, J. Huang, M. Zhang, R. Aguilar, M. Giménez, J. Xia, O. Dousse, W. Gierke, D. Yates, K. Jalan, L. Li, E. Latorre-Chimoto, D. D. Nguyen, K. Durden, P. Kallakuri, Y. Liu, M. Johnson, T. Tsai, A. Talbert, J. Liu, A. Neitz, C. Elkind, M. Selvi, M. Jasarevic, L. B. Soares, A. Cui, P. Wang, A. W. Wang, X. Ye, K. Kallarackal, L. Loher, H. Lam, J. Broder, D. Holtmann-Rice, N. Martin, B. Ramadhana, M. Shukla, S. Basu, A. Mohan, N. Fernando, N. Fiedel, K. Paterson, H. Li, A. Garg, J. Park, D. Choi, D. Wu, S. Singh, Z. Zhang, A. Globerson, L. Yu, J. Carpenter, F. de Chaumont Quitry, C. Radebaugh, C.-C. Lin, A. Tudor, P. Shroff, D. Garmon, D. Du, N. Vats, H. Lu, S. Iqbal, A. Yakubovich, N. Tripuraneni, J. Manyika, H. Qureshi, N. Hua, C. Ngani, M. A. Raad, H. Forbes, J. Stanway, M. Sundararajan, V. Ungureanu, C. Bishop, Y. Li, B. Venkatraman, B. Li, C. Thornton, S. Scellato, N. Gupta, Y. Wang, I. Tenney, X. Wu, A. Shenoy, G. Carvajal, D. G. Wright, B. Bariach, Z. Xiao, P. Hawkins, S. Dalmia, C. Farabet, P. Valenzuela, Q. Yuan, A. Agarwal, M. Chen, W. Kim, B. Hulse, N. Dukkipati, A. Paszke, A. Bolt, K. Choo, J. Beattie, J. Prendki, H. Vashisht, R. Santamaria-Fernandez, L. C. Cobo, J. Wilkiewicz, D. Madras, A. Elqursh, G. Uy, K. Ramirez, M. Harvey, T. Liechty, H. Zen, J. Seibert, C. H. Hu, A. Khorlin, M. Le, A. Aharoni, M. Li, L. Wang, S. Kumar, N. Casagrande, J. Hoover, D. E. Badawy, D. Soergel, D. Vnukov, M. Miecnikowski, J. Simsa, P. Kumar, T. Sellam, D. Vlasic, S. Daruki, N. Shabat, J. Zhang, G. Su, J. Zhang, J. Liu, Y. Sun, E. Palmer, A. Ghaffarkhah, X. Xiong, V. Cotruta, M. Fink, L. Dixon, A. Sreevatsa, A. Goedeckemeyer, A. Dimitriev, M. Jafari, R. Crocker, N. FitzGerald, A. Kumar, S. Ghemawat, I. Philips, F. Liu, Y. Liang, R. Sterneck, A. Repina, M. Wu, L. Knight, M. Georgiev, H. Lee, H. Askham, A. Chakladar, A. Louis, C. Crous, H. Cate, D. Petrova, M. Quinn, D. Owusu-Afriyie, A. Singhal, N. Wei, S. Kim, D. Vincent, M. Nasr, C. A. Choquette-Choo, R. Tojo, S. Lu, D. de Las Casas, Y. Cheng, T. Bolukbasi, K. Lee, S. Fatehi, R. Ananthanarayanan, M. Patel, C. Kaed, J. Li, S. R. Belle, Z. Chen, J. Konzelmann, S. Põder, R. Garg, V. Koverkathu, A. Brown, C. Dyer, R. Liu, A. Nova, J. Xu, A. Walton, A. Parrish, M. Epstein, S. McCarthy, S. Petrov, D. Hassabis, K. Kavukcuoglu, J. Dean, and O. Vinyals. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024.

[86] Q. Team. Introducing qwen2, June 2024.

[87] A. V. Tkachenko and S. Maslov. Onset of natural selection in populations of autocatalytic heteropolymers. *The Journal of chemical physics*, 149(13), 2018.

[88] S. Vig, A. Kogar, M. Mitrano, A. Husain, L. Venema, M. Rak, V. Mishra, P. Johnson, G. Gu, E. Fradkin, et al. Measurement of the dynamic charge response of materials using low-energy, momentum-resolved electron energy-loss spectroscopy (m-eels). *SciPost Physics*, 3(4):026, 2017.

[89] V. Volterra. Variations and fluctuations of the number of individuals in animal species living together. *ICES Journal of Marine Science*, 3(1):3–51, 1928.

[90] L. K. Wagner. Correlations and effective interactions from first principles using quantum monte carlo. *Handbook of Materials Modeling: Methods: Theory and Modeling*, pages 417–433, 2020.

[91] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018.

[92] X. Wang, Z. Hu, P. Lu, Y. Zhu, J. Zhang, S. Subramaniam, A. R. Loomba, S. Zhang, Y. Sun, and W. Wang. Scibench: Evaluating college-level scientific problem-solving abilities of large language models, 2024.

[93] X. Wang, Z. Wang, J. Liu, Y. Chen, L. Yuan, H. Peng, and H. Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *ArXiv preprint*, abs/2309.10691, 2023.

[94] Y. Wang, W. Wang, S. Joty, and S. C. H. Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation, 2021.

[95] Z. Wang, Y. Fu, A. Goyal, and S. Maslov. Emergent ecological advantage of sequential metabolic strategies in complex microbial communities. Manuscript unsubmitted, jun 2024.

[96] Z. Wang, A. Goyal, V. Dubinkina, A. B. George, T. Wang, Y. Fridman, and S. Maslov. Complementary resource preferences spontaneously emerge in diauxic microbial communities. *Nature communications*, 12(1):6661, 2021.

[97] Z. Wang, A. Goyal, and S. Maslov. The ecological consequences of microbial metabolic strategies in fluctuating environments. *bioRxiv*, pages 2023–07, 2023.

[98] W. A. Wheeler, S. Pathak, K. G. Kleiner, S. Yuan, J. N. Rodrigues, C. Lorsung, K. Krongchon, Y. Chang, Y. Zhou, B. Busemeyer, et al. Pyqmc: An all-python real-space quantum monte carlo module in pyscf. *The Journal of Chemical Physics*, 158(11), 2023.

[99] B. Widom. Some Topics in the Theory of Fluids. *The Journal of Chemical Physics*, 39(11):2808–2812, 12 1963.

[100] P. Yin, B. Deng, E. Chen, B. Vasilescu, and G. Neubig. Learning to mine aligned code and natural language pairs from stack overflow. In *International Conference on Mining Software Repositories*, MSR, pages 476–486. ACM, 2018.

[101] P. Yin, W.-D. Li, K. Xiao, A. Rao, Y. Wen, K. Shi, J. Howland, P. Bailey, M. Catasta, H. Michalewski, A. Polozov, and C. Sutton. Natural language to code generation in interactive data science notebooks, 2022.

[102] Q. Zhu, D. Guo, Z. Shao, D. Yang, P. Wang, R. Xu, Y. Wu, Y. Li, H. Gao, S. Ma, et al. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*, 2024.

A Appendix

A.1 Prompt

PROBLEM DESCRIPTION:
You will be provided with problem steps along with background knowledge necessary for solving the problem. Your task will be to develop a Python solution focused on the next step of the problem-solving process.

PROBLEM STEPS AND FUNCTION CODE:
Here, you’ll find the Python code for the initial steps of the problem-solving process. This code is integral to building the solution.
{problem_steps_str}

NEXT STEP - PROBLEM STEP AND FUNCTION HEADER:
This part will describe the next step in the problem-solving process. A function header will be provided, and your task is to develop the Python code for this next step based on the provided description and function header.
{next_step_str}

DEPENDENCIES:
Use only the following dependencies in your solution. Do not include these dependencies at the beginning of your code.
{dependencies}

RESPONSE GUIDELINES:
1. Start with the scientific background required for the next step, formatted as a comment.
2. Then write the complete and executable Python program for the next step in a single block.
3. Your response should focus exclusively on implementing the solution for the next step, adhering closely to the specified function header and the context provided by the initial steps.
4. DO NOT include previous function code, example usage or test code in your response.
5. Ensure your response is in the format of “‘python“‘ and includes the necessary background as a comment at the top.
Example:
Background: [Here, insert the necessary scientific knowledge required for the next step.]
[Insert the Python code here based on the provided function header and dependencies.]

Table 4: Prompt w/o Background

PROBLEM DESCRIPTION:
You will be provided with problem steps along with background knowledge necessary for solving the problem. Your task will be to develop a Python solution focused on the next step of the problem-solving process.

PROBLEM STEPS AND FUNCTION CODE:
Here, you'll find the Python code for the initial steps of the problem-solving process. This code is integral to building the solution.
{problem_steps_str}

NEXT STEP - PROBLEM STEP AND FUNCTION HEADER:
This part will describe the next step in the problem-solving process. A function header will be provided, and your task is to develop the Python code for this next step based on the provided description and function header.
{next_step_str}

DEPENDENCIES:
Use only the following dependencies in your solution. Do not include these dependencies at the beginning of your code.
{dependencies}

RESPONSE GUIDELINES:
1. Write the complete and executable Python program for the next step in a single block.
2. Your response should focus exclusively on implementing the solution for the next step, adhering closely to the specified function header and the context provided by the initial steps.
3. DO NOT include previous function code, example usage or test code in your response.
4. Ensure your response is in the format of “‘python“‘ and includes the necessary background as a comment at the top.

Table 5: Prompt w/ Scientists' Background

A.2 Python libraries used in SciCode.

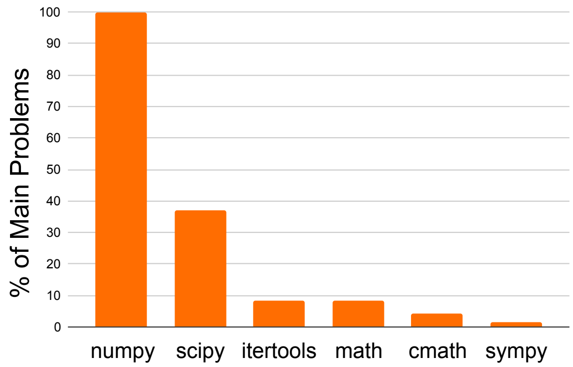
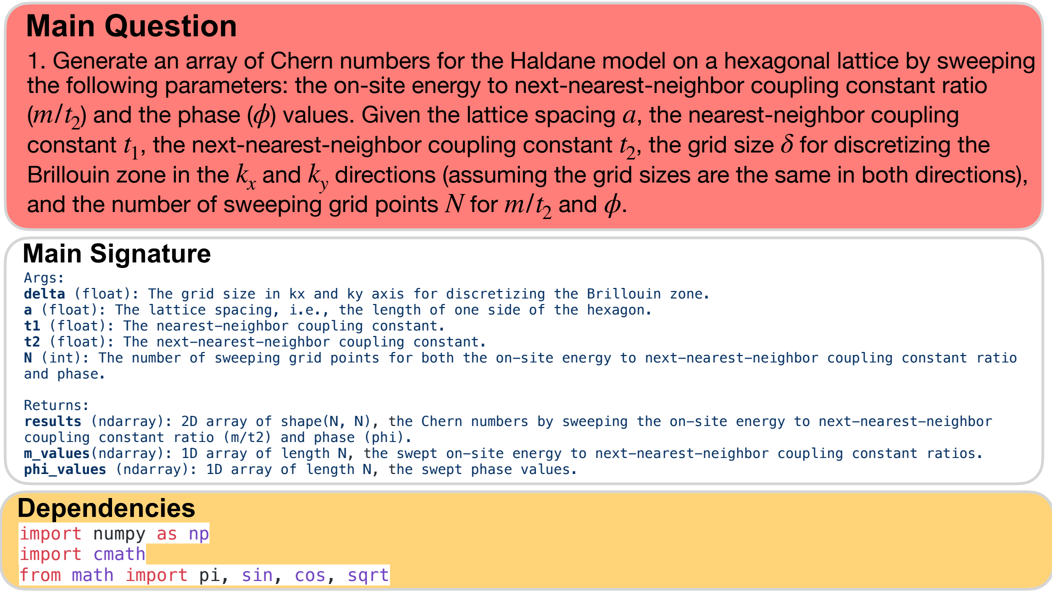


Figure 4: Python libraries used by problems in SciCode.

A.3 SciCode Full Problem Example

A.3.1 Example Main Problem



Main Question

1. Generate an array of Chern numbers for the Haldane model on a hexagonal lattice by sweeping the following parameters: the on-site energy to next-nearest-neighbor coupling constant ratio (m/t_2) and the phase (ϕ) values. Given the lattice spacing a, the nearest-neighbor coupling constant t_1, the next-nearest-neighbor coupling constant t_2, the grid size δ for discretizing the Brillouin zone in the k_x and k_y directions (assuming the grid sizes are the same in both directions), and the number of sweeping grid points N for m/t_2 and ϕ.

Main Signature

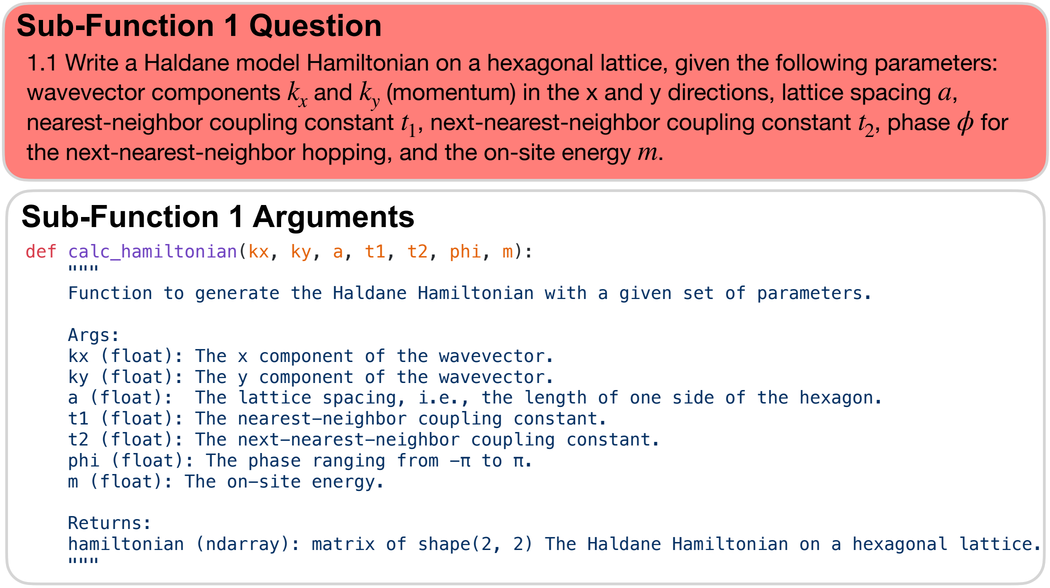
```
Args:
delta (float): The grid size in kx and ky axis for discretizing the Brillouin zone.
a (float): The lattice spacing, i.e., the length of one side of the hexagon.
t1 (float): The nearest-neighbor coupling constant.
t2 (float): The next-nearest-neighbor coupling constant.
N (int): The number of sweeping grid points for both the on-site energy to next-nearest-neighbor coupling constant ratio
and phase.

Returns:
results (ndarray): 2D array of shape(N, N), the Chern numbers by sweeping the on-site energy to next-nearest-neighbor
coupling constant ratio (m/t2) and phase (phi).
m_values(ndarray): 1D array of length N, the swept on-site energy to next-nearest-neighbor coupling constant ratios.
phi_values (ndarray): 1D array of length N, the swept phase values.
```

Dependencies

```
import numpy as np
import cmath
from math import pi, sin, cos, sqrt
```

Figure 5: SciCode Example General Problem and Dependencies

A.3.2 Example Subproblems



Sub-Function 1 Question

1.1 Write a Haldane model Hamiltonian on a hexagonal lattice, given the following parameters: wavevector components k_x and k_y (momentum) in the x and y directions, lattice spacing a, nearest-neighbor coupling constant t_1, next-nearest-neighbor coupling constant t_2, phase ϕ for the next-nearest-neighbor hopping, and the on-site energy m.

Sub-Function 1 Arguments

```
def calc_hamiltonian(kx, ky, a, t1, t2, phi, m):
    """
    Function to generate the Haldane Hamiltonian with a given set of parameters.

    Args:
    kx (float): The x component of the wavevector.
    ky (float): The y component of the wavevector.
    a (float):  The lattice spacing, i.e., the length of one side of the hexagon.
    t1 (float): The nearest-neighbor coupling constant.
    t2 (float): The next-nearest-neighbor coupling constant.
    phi (float): The phase ranging from −π to π.
    m (float): The on-site energy.

    Returns:
    hamiltonian (ndarray): matrix of shape(2, 2) The Haldane Hamiltonian on a hexagonal lattice.
    """
```

Figure 6: SciCode Example Subproblem 1

Subproblem 1 Background
Source: Haldane, F. D. M. (1988). Model for a quantum Hall effect without Landau levels: Condensed-matter realization of the" parity anomaly". Physical review letters, 61(18).
We denote $\{\mathbf{a}_i\}$ are the vectors from a B site to its three nearest-neighbor A sites, and $\{\mathbf{b}_i\}$ are next-nearest-neighbor distance vectors, then we have

$$\begin{aligned}\mathbf{a}_1 &= (0, a),\\ \mathbf{a}_2 &= \left(\frac{\sqrt{3}a}{2}, -\frac{a}{2}\right),\\ \mathbf{a}_3 &= \left(-\frac{\sqrt{3}a}{2}, -\frac{a}{2}\right)\end{aligned}$$

$$\begin{aligned}\mathbf{b}_1 &= \mathbf{a}_2 - \mathbf{a}_3 = (\sqrt{3}a, 0),\\ \mathbf{b}_2 &= \mathbf{a}_3 - \mathbf{a}_1 = \left(-\frac{\sqrt{3}a}{2}, -\frac{3a}{2}\right),\\ \mathbf{b}_3 &= \mathbf{a}_1 - \mathbf{a}_2 = \left(-\frac{\sqrt{3}a}{2}, \frac{3a}{2}\right)\end{aligned}$$

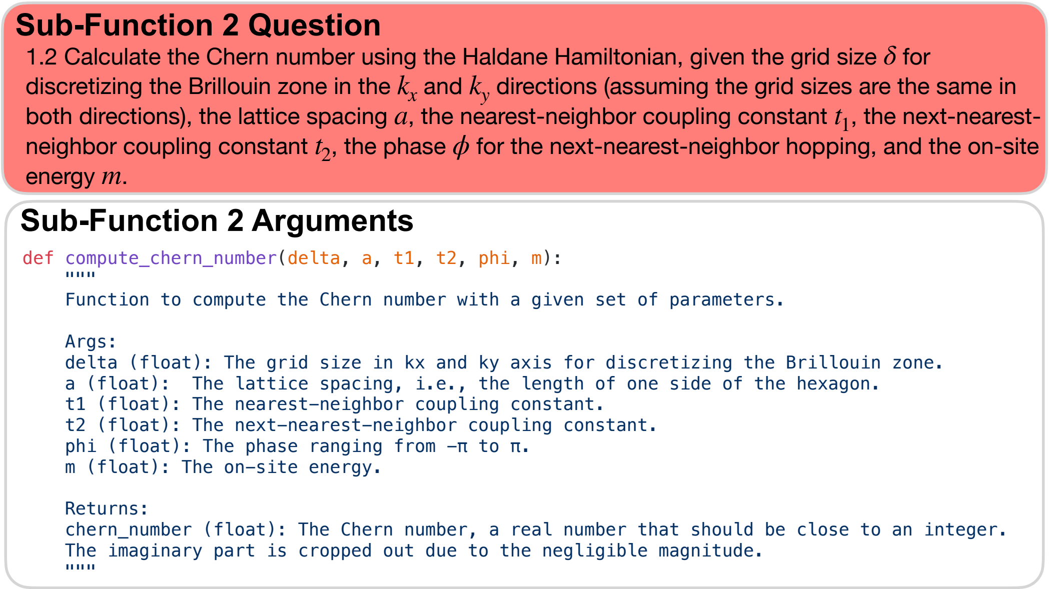
Then the Haldane model on a hexagonal lattice can be written as

$$H(k) = d_0 I + d_1\sigma_1 + d_2\sigma_2 + d_3\sigma_3$$

$$\begin{aligned}d_0 &= 2t_2\cos\phi\sum_i \cos(\mathbf{k}\cdot\mathbf{b}_i)\\ &= 2t_2\cos\phi\left[\cos\left(\sqrt{3}k_x a\right) + \cos\left(-\frac{\sqrt{3}k_x a}{2} + \frac{3k_y a}{2}\right) + \cos\left(-\frac{\sqrt{3}k_x a}{2} - \frac{3k_y a}{2}\right)\right]\\ d_1 &= t_1\sum_i \cos(\mathbf{k}\cdot\mathbf{a}_i)\\ &= t_1\left[\cos\left(k_y a\right) + \cos\left(\frac{\sqrt{3}k_x a}{2} - \frac{k_y a}{2}\right) + \cos\left(-\frac{\sqrt{3}k_x a}{2} - \frac{k_y a}{2}\right)\right]\\ d_2 &= t_1\sum_i \sin(\mathbf{k}\cdot\mathbf{a}_i)\\ &= t_1\left[\sin\left(k_y a\right) + \sin\left(\frac{\sqrt{3}k_x a}{2} - \frac{k_y a}{2}\right) + \sin\left(-\frac{\sqrt{3}k_x a}{2} - \frac{k_y a}{2}\right)\right]\\ d_3 &= m - 2t_2\sin\phi\sum_i \sin(\mathbf{k}\cdot\mathbf{b}_i)\\ &= m - 2t_2\sin\phi\left[\sin\left(\sqrt{3}k_x a\right) + \sin\left(-\frac{\sqrt{3}k_x a}{2} + \frac{3k_y a}{2}\right) + \sin\left(-\frac{\sqrt{3}k_x a}{2} - \frac{3k_y a}{2}\right)\right]\end{aligned}$$

where σ_i are the Pauli matrices and I is the identity matrix.

Table 6: Background Augmented by Scientists for Subproblem 1



Sub-Function 2 Question

1.2 Calculate the Chern number using the Haldane Hamiltonian, given the grid size δ for discretizing the Brillouin zone in the k_x and k_y directions (assuming the grid sizes are the same in both directions), the lattice spacing a, the nearest-neighbor coupling constant t_1, the next-nearest-neighbor coupling constant t_2, the phase ϕ for the next-nearest-neighbor hopping, and the on-site energy m.

Sub-Function 2 Arguments

```
def compute_chern_number(delta, a, t1, t2, phi, m):
    """
    Function to compute the Chern number with a given set of parameters.

    Args:
    delta (float): The grid size in kx and ky axis for discretizing the Brillouin zone.
    a (float):  The lattice spacing, i.e., the length of one side of the hexagon.
    t1 (float): The nearest-neighbor coupling constant.
    t2 (float): The next-nearest-neighbor coupling constant.
    phi (float): The phase ranging from -π to π.
    m (float): The on-site energy.

    Returns:
    chern_number (float): The Chern number, a real number that should be close to an integer.
    The imaginary part is cropped out due to the negligible magnitude.
    """
```

Figure 7: SciCode Example Subproblem 2

Subproblem 2 Background

Source: Fukui, Takahiro, Yasuhiro Hatsugai, and Hiroshi Suzuki. "Chern numbers in discretized Brillouin zone: efficient method of computing (spin) Hall conductances." Journal of the Physical Society of Japan 74.6 (2005): 1674-1677.

Here we can discretize the two-dimensional Brillouin zone into grids with step $\delta k_x = \delta k_y = \delta$. If we define the U(1) gauge field on the links of the lattice as $U_\mu(\mathbf{k}_l) := \frac{\langle n(\mathbf{k}_l)|n(\mathbf{k}_l+\hat{\mu})\rangle}{|\langle n(\mathbf{k}_l)|n(\mathbf{k}_l+\hat{\mu})\rangle|}$, where $|n(\mathbf{k}_l)\rangle$ is the eigenvector of Hamiltonian at $\mathbf{k}_l$, $\hat{\mu}$ is a small displacement vector in the direction μ with magnitude δ, and $\mathbf{k}_l$ is one of the momentum space lattice points l. The corresponding curvature (flux) becomes

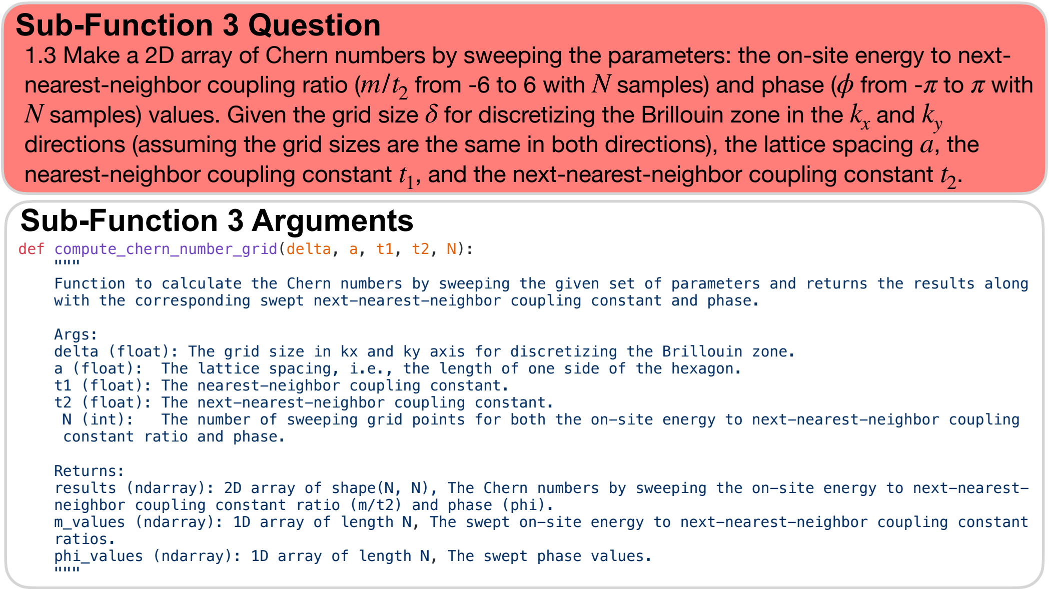
$$F_{xy}(\mathbf{k}_l) := \ln\left[U_x(\mathbf{k}_l)U_y(\mathbf{k}_l+\hat{x})U_x^{-1}(\mathbf{k}_l+\hat{y})U_y^{-1}(\mathbf{k}_l)\right]$$

and the Chern number of a band can be calculated as

$$c = \frac{1}{2\pi i}\sum_l F_{xy}(\mathbf{k}_l),$$

where the summation is over all the lattice points l. Note that the Brillouin zone of a hexagonal lattice with spacing a can be chosen as a rectangle with $0 \le k_x \le k_{x0} = \frac{2\sqrt{3}\pi}{3a}, 0 \le k_y \le k_{y0} = \frac{4\pi}{3a}$.

Table 7: Background Augmented by Scientists for Subproblem 2



Sub-Function 3 Question

1.3 Make a 2D array of Chern numbers by sweeping the parameters: the on-site energy to next-nearest-neighbor coupling ratio (m/t_2 from -6 to 6 with N samples) and phase (ϕ from -π to π with N samples) values. Given the grid size δ for discretizing the Brillouin zone in the k_x and k_y directions (assuming the grid sizes are the same in both directions), the lattice spacing a, the nearest-neighbor coupling constant t_1, and the next-nearest-neighbor coupling constant t_2.

Sub-Function 3 Arguments

```
def compute_chern_number_grid(delta, a, t1, t2, N):
    """
    Function to calculate the Chern numbers by sweeping the given set of parameters and returns the results along
    with the corresponding swept next-nearest-neighbor coupling constant and phase.

    Args:
    delta (float): The grid size in kx and ky axis for discretizing the Brillouin zone.
    a (float):  The lattice spacing, i.e., the length of one side of the hexagon.
    t1 (float): The nearest-neighbor coupling constant.
    t2 (float): The next-nearest-neighbor coupling constant.
     N (int):   The number of sweeping grid points for both the on-site energy to next-nearest-neighbor coupling
     constant ratio and phase.

    Returns:
    results (ndarray): 2D array of shape(N, N), The Chern numbers by sweeping the on-site energy to next-nearest-
    neighbor coupling constant ratio (m/t2) and phase (phi).
    m_values (ndarray): 1D array of length N, The swept on-site energy to next-nearest-neighbor coupling constant
    ratios.
    phi_values (ndarray): 1D array of length N, The swept phase values.
    """
```

Figure 8: SciCode Example Subproblem 3

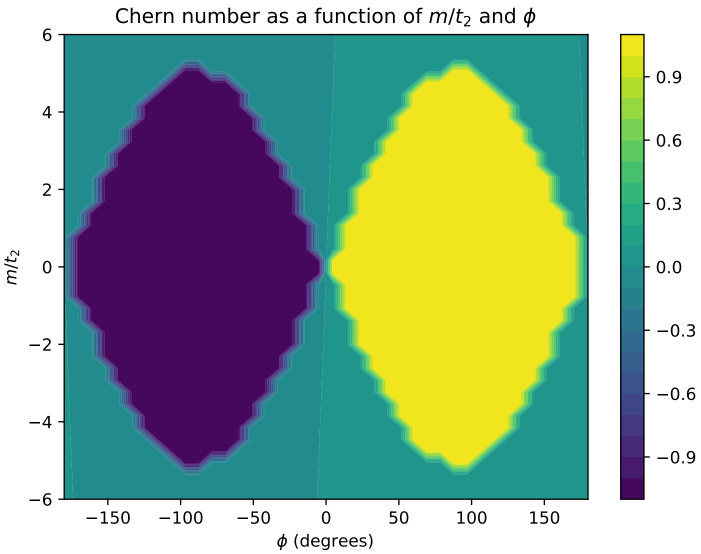
A.3.3 Example Domain Specific Test Cases

Both the k-space and sweeping grid sizes are set to very rough values to make the computation faster, feel free to increase them for higher accuracy.

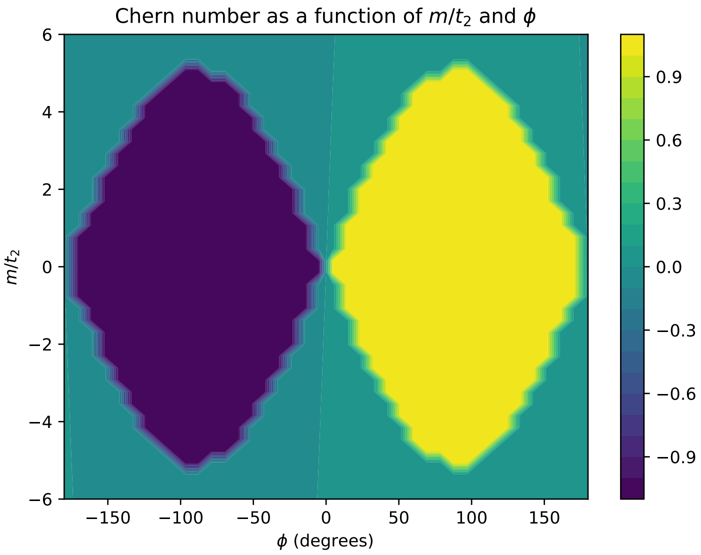
At zero on-site energy, the Chern number is 1 for $\phi > 0$, and the Chern number is -1 for $\phi < 0$.

For complementary plots Figure 9, we can see that these phase diagrams are similar to the one in the original paper: Fig.2 in Haldane, F. D. M. (1988). To achieve a better match, decrease all grid sizes.

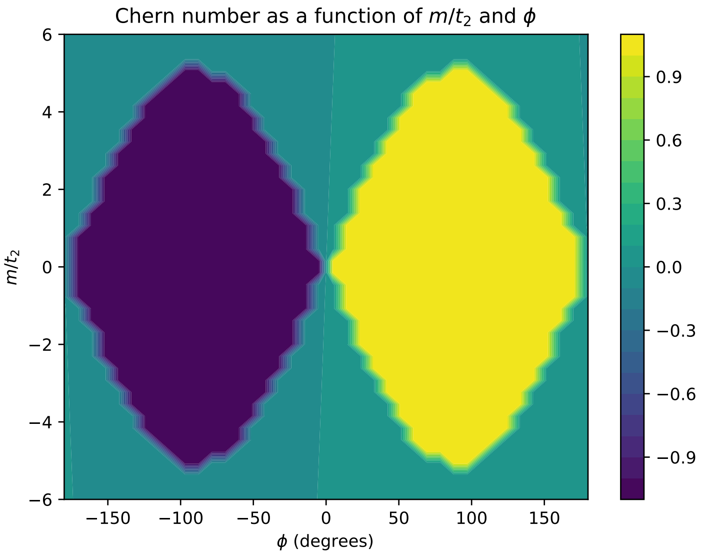
Compare the following three test cases. We can find that the phase diagram is independent of the value of t_1, and the ratio of t_2/t_1, which is consistent with our expectations.



(a) delta = 2π/30, a = 1.0, t1 = 4.0, t2 = 1.0, N = 40



(b) delta = $2\pi/30$, a = 1.0, t1 = 5.0, t2 = 1.0, N = 40



(c) delta = $2\pi/30$, a = 1.0, t1 = 1.0, t2 = 0.2, N = 40

Figure 9: Complementary Figures of Domain Specific Test Cases