# HARDMath: A Benchmark Dataset for Challenging Problems in Applied Mathematics

Jingxuan Fan (jfan@g.harvard.edu)*, Sarah Martinson*, Erik Y. Wang*, Kaylie Hausknecht*, Jonah Brenner, Danxian Liu, Nianli Peng, Corey Wang, Michael Brenner **School of Engineering and Applied Sciences, Harvard University**

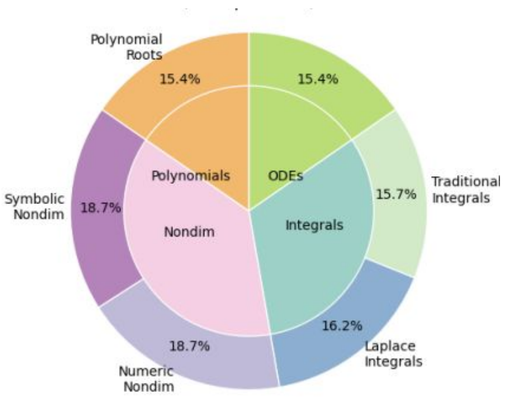
## The HARDMath Dataset

**Problem:** existing benchmarks lack graduate-level applied math!

**We provide:**

- ➔ **1426 graduate-level** applied math problems across **6** domains, targeting **approximation reasoning** & **asymptotic methods**
- ➔ An **algorithmic framework** that generates questions and solutions with quality guarantees

**HARDMath** dataset composition




**Sample Problem and Solution (*Integral*)**

**Problem:**
Consider the integral $I(\epsilon) = \int_0^{56.00} \frac{1}{\epsilon + 2.0x^{6.0} + 2.0x^{9.0} + 5.0x^{11.0} + 5.0x^{13.0}} dx$. Develop analytical formulas that approximate $I(\epsilon)$ for different regimes of $\epsilon$.

**Solution:** The integral is of the form $I(\epsilon) = \int_0^{56} \frac{1}{\epsilon + P(x)} dx$ where $P(x)$ is a polynomial. Thus, its value can be estimated as the product between a height and a width.
Since the integrand is maximized at $x = 0$, the height can be set to $\frac{1}{\epsilon}$.
For small $\epsilon$, we define the width as the point where the integrand becomes half of its maximum height. This corresponds to solving for $x$ given $P(x) = \epsilon$. Applying dominant balance, considering the term in $P(x)$ with the smallest degree, the width is approximated as $\left(\frac{1}{2.0*\epsilon}\right)^{1/6.0}$. Therefore, the analytical approximation of the integral for small $\epsilon$ is $I(\epsilon) = \frac{0.8909}{\epsilon^{0.8333}}$.
For an intermediate regime where $\epsilon$ is large, we also define the width based on the term with the largest degree. The width is approximated as $\left(\frac{1}{5.0*\epsilon}\right)^{1/13.0}$. Therefore, the analytical approximation of the integral for large $\epsilon$ is $I(\epsilon) = \frac{0.7647}{\epsilon^{0.8333}}$.
If the width of the integral exceeds the range of integration, we consider one more regime for very large $\epsilon$. The width is then just the range of integration, so in this regime, the integral can be approximated as $\frac{L}{\epsilon}$. Therefore, the analytical approximation of the integral for very large $\epsilon$ is $I(\epsilon) = \frac{56}{\epsilon}$.
Altogether, the solutions at small, large, and very large $\epsilon$ are $\boxed{\frac{0.89}{\epsilon^{0.83}}, \frac{0.76}{\epsilon^{0.83}}, \frac{56}{\epsilon}}$.

## Evaluation

- ➔ Answer + **procedural grading** with **LLM grader and rubrics**
- ➔ Evaluations on frontier models including **STEM-optimized o1**
- ➔ Detailed f**ew-shot performance** behavior and **error analysis**

Evaluation accuracy (percentage) on **HARDMath-mini**

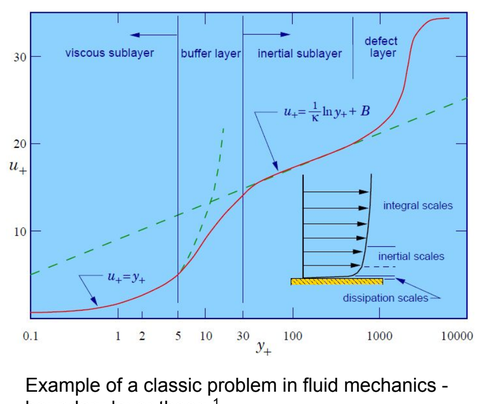
| Model | ALL | Nondim | Roots | ODEs | Integrals |
|---|---|---|---|---|---|
| **Closed-source models** | | | | | |
| GPT-3.5 (0 shot) | 6.04 | 5.05 | 17.2 | 1.39 | 3.33 |
| GPT-3.5 (1 shot CoT) | 14.2 | 6.11 | 29.3 | 6.94 | 18.2 |
| GPT-3.5 (5 shot CoT) | 24.6 | 24.3 | 35.0 | 16.2 | 23.1 |
| GPT-4 (0 shot) | 14.0 | 6.04 | 33.7 | 7.87 | 14.9 |
| GPT-4 (1 shot CoT) | 37.6 | 36.5 | 52.8 | 15.9 | 40.5 |
| GPT-4 (5 shot CoT) | 43.8 | 48.6 | 57.3 | 21.7 | 41.4 |
| o1-mini (0 shot CoT) | 29.8 | 38.1 | 24.3 | 10.2 | 32.5 |
| o1-mini (5 shot CoT) | 62.3 | 84.5 | 62.1 | 30.6 | 46.5 |
| **Open-source models** | | | | | |
| Llama3-8b (0 shot) | 3.67 | 0.50 | 11.5 | 4.63 | 2.52 |
| Llama3-8b (5 shot CoT) | 20.2 | 17.9 | 17.1 | 12.0 | 28.1 |
| CodeLlama-13b (0 shot) | 1.94 | 0.00 | 8.73 | 1.85 | 0.50 |
| CodeLlama-13b (5 shot CoT) | 9.79 | 8.41 | 13.1 | 9.7 | 9.57 |

Performance comparison to widely used benchmarks

| Model | HARDMath | GSM8K | MATH |
|---|---|---|---|
| **Closed-source models** | | | |
| GPT-4 (0 shot) | 14.0 | | 73.4[2] |
| GPT-4 (5 shot CoT) | 43.8 | 92.0[3] | |
| o1-mini (0 shot) | 29.8 | | 90.0[4] |
| o1-mini (5 shot CoT) | 62.3 | | |
| **Open-source models** | | | |
| Llama3-8b (5 shot CoT) | 20.2 | 79.6 (8-shot)[5] | 30.0 (4-shot)[5] |

- ➔ Performance on **HARDMath** is much worse than widely used math benchmarks under comparable prompting conditions, **even with partial credit**
- ➔ Offers new opportunities for evaluation and model development

## Why HARDMath? Scientific Research!

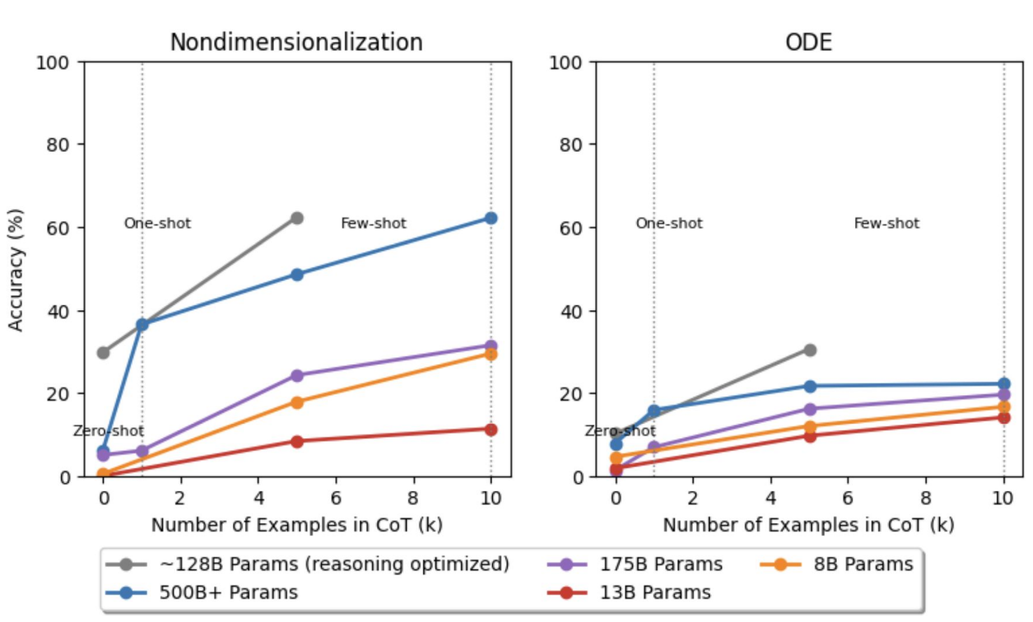



Example of a classic problem in fluid mechanics - boundary layer theory[1]

**Boundary Layer Problem**

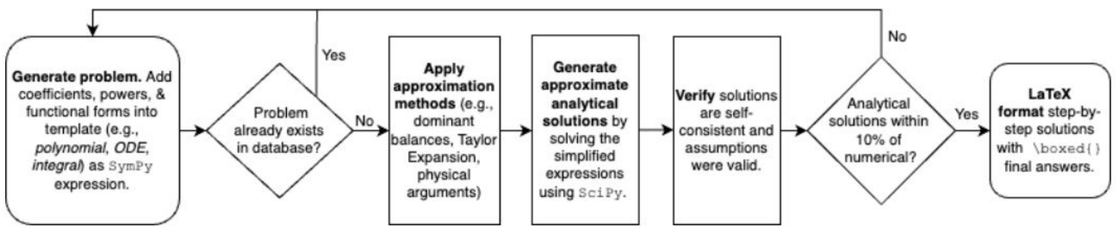
The viscous sublayer ($y^+ \ll 1$) can be analyzed starting from the Navier-Stokes equations: $\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u}\right) = -\nabla p + \mu \nabla^2 \mathbf{u}, \quad \nabla \cdot \mathbf{u} = 0$.
**Non-dimensionalization:** Define non-dimensional variables: $u^+ = \frac{u}{u_\tau}$, $y^+ = \frac{y u_\tau}{\nu}$, $u_\tau = \sqrt{\frac{\tau_w}{\rho}}$, $\nu = \frac{\mu}{\rho}$. The streamwise momentum equation simplifies to: $\frac{\partial \tau}{\partial y^+} = 0$, where $\tau = \mu \frac{\partial u}{\partial y}$.
**Small $y^+$ regime:** In the viscous sublayer, inertial terms $\mathbf{u} \cdot \nabla \mathbf{u}$ are negligible, and the momentum equation becomes: $\frac{\partial u}{\partial y} = \frac{\tau_w}{\mu}$. In non-dimensional form: $\frac{\partial u^+}{\partial y^+} = 1$. Integrating yields: $u^+ = y^+$.

Analytically solving classical fluid mechanics problems such as the one above requires a combination of different problem types and problem-solving methods included in the **HARDMath** dataset, like nondimensionalization, ODEs, and dominant balance.



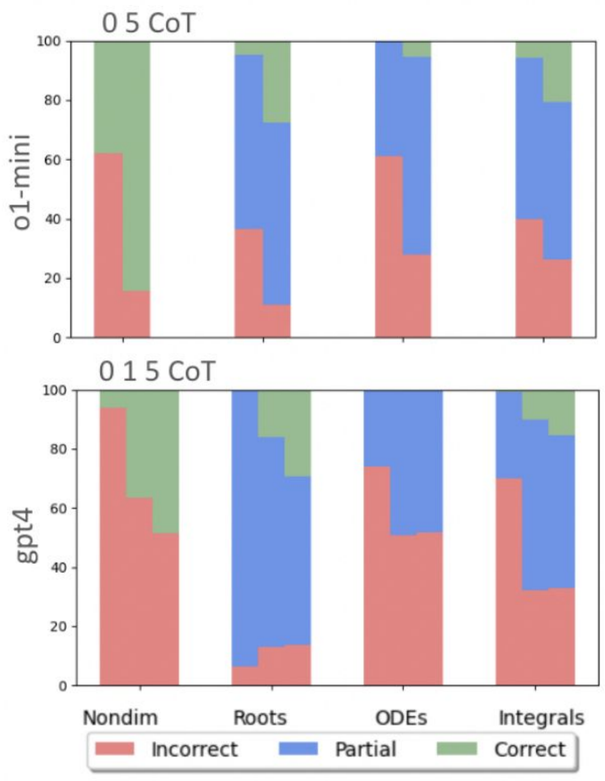

Performance gain from few-shot CoT prompting is model size and problem type dependent: the more challenging type *ODE* shows minimal performance increase after few-shot CoT

## Dataset Generation




- ➔ Parameter combination checks ensure generating unique problems
- ➔ Leveraging SymPy and SciPy to perform the calculations required to derive our approximate, analytical solutions
- ➔ Checking against numerical solver “ground-truth” results ensures correctness of analytical approximation provided in the solution

## Error Analysis




- ➔ Few-shot CoT prompting increases model performance by changing correct-partial compositions
- ➔ Highly dependent on model and problem types

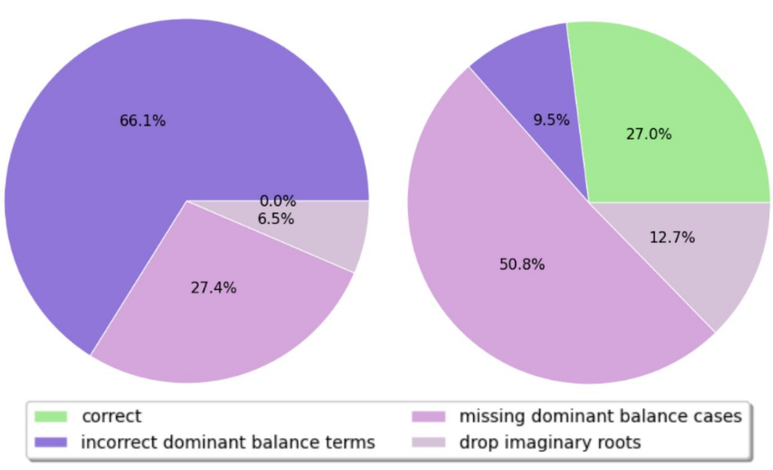
Example error composition analysis for *Polynomial Roots* problems

***Polynomial Roots*** **(error mode: incorrect dominant balance terms)**

**Question:** Consider the polynomial $P(x) = \epsilon x^6 + x^4 - 1$. Find approximate expressions for all roots of the polynomials in the limit of small positive $\epsilon$ and large positive $\epsilon$ (single term approximation).
**Model Response:** For large $\epsilon$ When $\epsilon$ is large, the term $\epsilon x^6$ dominates the polynomial for most $x$. However, focusing on the dominant term gives: $P(x) \approx \epsilon x^6$. Setting this equal to zero for an approximation suggests that we don't get meaningful roots from this term alone since $x^6 = 0$ doesn't provide roots that satisfy the original polynomial's requirement for non-zero roots.
**Grading**: The response wrongly sets up the dominant balance as $A = 0$ and does not provide a concrete list of roots for large $\epsilon$. Therefore, score for large positive $\epsilon$ is $\boxed{0.0}$

***Polynomial Roots*** **(error mode: missing dominant balance cases)**

**Question:** Consider the polynomial $P(x) = \epsilon x^8 + x^4 - 1$. Find approximate expressions for all roots of the polynomials in the limit of small positive $\epsilon$ and large positive $\epsilon$ (single term approximation).
**Model Response:** When $\epsilon$ is small, the term $\epsilon x^8$ becomes negligible, so we balance term B and C and the polynomial simplifies to: $P(x) \approx x^4 - 1$. Solving the equation gives roots $[1, -1, i, -i]$.
**Grading**: The response only includes the roots from the balance $B + C = 0$ and completely misses the roots from the balance $A + B = 0$. Therefore, score for small positive $\epsilon$ is $\boxed{0.5}$




GPT-4 0 shot CoT error composition

GPT-4 5 shot CoT error composition

## References

1. LearnCAx (n.d.). “Basics of y-plus, boundary layer, and wall function in turbulent flows.” In: LearnCAx Knowledge Base. Available at: https://learncax.com

2. OpenAI (2024). “Simple Evals.” In: GitHub Repository. Available at: https://github.com/openai/simple-evals

3. Achiam, Josh, Adler, Steven, Agarwal, Sandhini, Ahmad, Lama, Akkaya, Ilge, Aleman, Florencia Leoni, Almeida, Diogo, Altenschmidt, Janko, Altman, Sam, Anadkat, Shyamal, et al. (2023). “GPT-4 Technical Report.” In: arXiv preprint arXiv:2303.08774.

4. OpenAI (2024). “OpenAI o1-mini.” In: Advancing Cost-Efficient Reasoning. Available at: https://openai.com/index/openai-o1-mini-advancing-cost-efficient-reasoning/

5. Meta AI (2024). “Meta Llama 3.” In: Meta AI Blog. Available at: https://ai.meta.com/blog/meta-llama-3/

**Arxiv**

**Github**