Delphi Model Evaluation Report: General Reasoning in Commercial Fast LLMs

Gensyn AI Team

We track the performance of the fast variants of commercial LLMs by popular frontier labs. Over three weeks, we will run evaluations across 11 general AI reasoning benchmarks curated to capture broad real-world reasoning ability. The score of each benchmark is normalized to a 0–100 scale, and a model's final score is the average across all of them, giving a single, clean metric of practical capability.

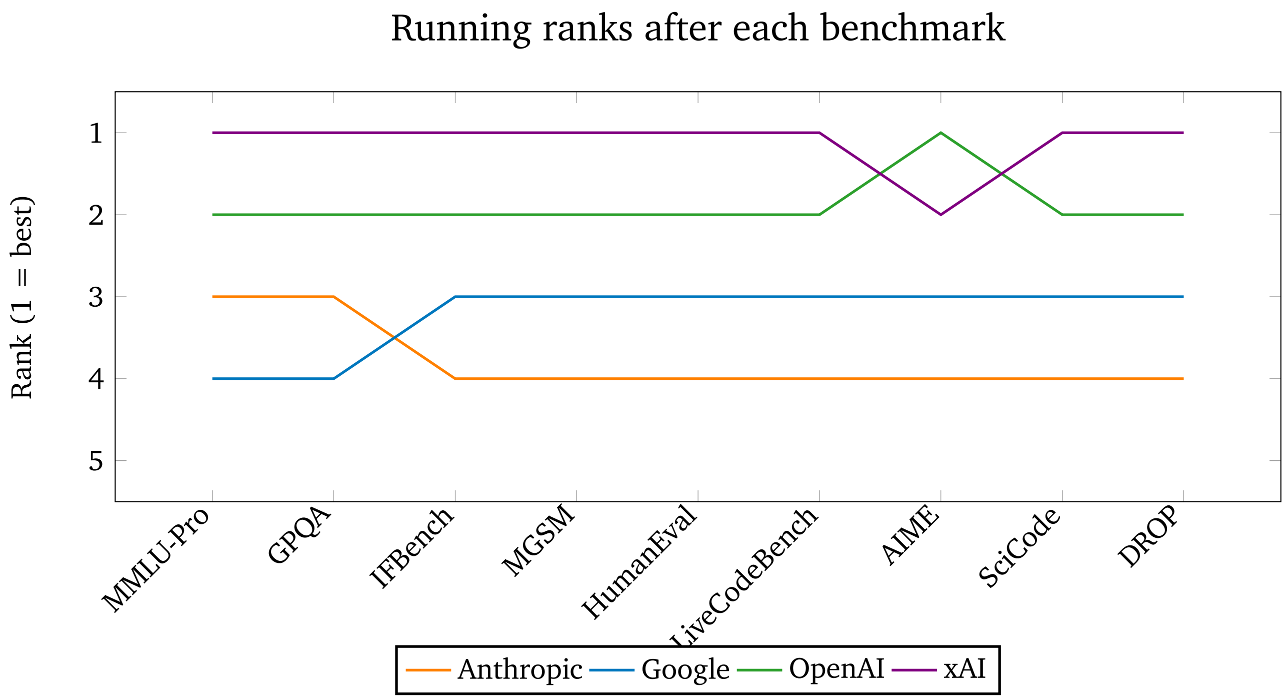
Entrant models

We evaluate the following models, using the default API accesses provided and versions current as of February 3, 2026.

1. Claude Haiku 4.5 from Anthropic.
   - Cost: $1/input MTok, $5/output MTok.
2. Gemini 3 Flash Preview from Google.
   - Cost: $0.50/input MTok, $3/output MTok.
3. GPT 5 Mini from OpenAI.
   - Cost: $0.25/input MTok, $2/output MTok.
4. Grok 4.1 Fast Reasoning from xAI.
   - Cost: $0.20/input MTok, $0.50/output MTok.

Model settings: We use the default temperature setting (which is typically 1.0). Unless otherwise noted for a given benchmark, we also used default settings for thinking level (also referred to as thinking budget or effort, depending on the provider).

Average scores



Rank	Family	Model	Avg. score	Δ rank	Notes
1	xAI	Grok 4.1 Fast Reasoning	79.47	-	-
2	OpenAI	GPT 5 Mini	78.02	-	-
3	Google	Gemini 3 Flash Preview	64.71	-	-
4	Anthropic	Claude Haiku 4.5	63.57	-	-

Avg. score is the mean of normalized benchmark scores. Δ *rank* is the change in rank compared to the last evaluation.

Per-Benchmark Breakdown

Benchmark	Claude Haiku 4.5	Gemini 3 Flash Preview	GPT 5 Mini	Grok 4.1 Fast Reasoning
MMLU-Pro (19)	78.86	52.50	81.65	85.08
GPQA-Diamond (16)	62.63	66.41	77.27	84.34
IFBench (15)	28.03	61.09	59.05	52.18
MGSM (17)	91.09	92.00	90.11	89.45
HumanEval (11)	91.46	94.51	94.51	95.12
LiveCodeBench (14)	56.49	46.54	86.26	84.55
AIME	34.67	44.33	94.00	89.33
SciCode (18)	37.15	31.60	38.19	43.40
DROP (12)	91.75	93.44	81.14	91.83
?	[redacted]	[redacted]	[redacted]	[redacted]
?	[redacted]	[redacted]	[redacted]	[redacted]

Benchmark 9: DROP

The DROP (Discrete Reasoning Over Paragraphs) benchmark (12) evaluates reading comprehension and discrete reasoning capabilities over textual passages. It consists of questions derived from Wikipedia paragraphs across diverse topics, including history, sports, and current events. Unlike traditional reading comprehension datasets, DROP requires models to perform discrete reasoning operations such as addition, subtraction, comparison, and multi-step aggregation over different parts of the text. Performance can be measured using both Exact Match (EM) and F1 score, where F1 accounts for partial credit through token overlap between predicted and reference answers.

Experimental setup. We use the scripts provided in the OpenAI simple-evals repository (7), which evaluates models on the development split of DROP with approximately 9,500 examples. For instruction-tuned models, we use a 3-shot prompting approach with examples drawn randomly from the training set, where the model is instructed to think step-by-step and output answers in the format "Answer: $ANSWER". For base (non-instruct) models, we use completion-style prompts by providing a clearer example/test separation and extracting answers using pattern matching for constructs like "finalAnswer: X". Temperature is set to 0.0 for greedy decoding. Results are reported as average F1 score across all examples, which better captures partial correctness than exact match alone.

Benchmark 8: SciCode

The SciCode benchmark (18) evaluates code generation capabilities for solving realistic scientific research problems. It consists of 80 main problems decomposed into 338 subproblems, covering 16 subdomains from 6 domains: Physics, Math, Material Science, Biology, and Chemistry. Problems are derived from real scientific workflows and focus on numerical methods, system simulations, and scientific calculations. Unlike exam-style benchmarks, SciCode requires models to demonstrate deep scientific knowledge, reasoning, and code synthesis abilities. A main problem is considered solved only if all its subproblems pass the scientist-annotated test cases.

Experimental setup. We evaluate models on the the *test* split available at the benchmark repository (8), which consists of 65 main problems and 288 subproblems. Models are evaluated in zero-shot mode with scientist-annotated background knowledge provided (`with_background=True`), which supplies the necessary scientific context and reasoning steps. This setting shifts the evaluation focus towards coding and instruction-following capabilities. Results are presented as subproblem accuracy. Note that temperature is set to 0 for deterministic outputs. While the models were given a maximum output budget of 16,384 tokens, nearly all responses were under a few thousand tokens.

We make the following observation:

- As in a prior coding benchmark (Benchmark 6: LiveCodeBench), Grok 4.1 Fast Reasoning scored the highest and Gemini 3 Flash Preview scored the lowest.

Benchmark 7: American Invitational Mathematics Examination (AIME)

The "American Invitational Mathematics Examination" is a competition-level dataset of math problems targeted at high school students. There are two parts, each composed of 15 questions; we evaluate our models on all 30 questions. Questions in this dataset are scored using a judge model to parse answers and semantically compare them to a given ground truth answer. The dataset can be accessed here: HuggingFace:math-ai/aime25. Frontier models perform well on the dataset, with GPT-5, for example, achieving 99% (6) without tools, such as a Python interpreter.

Experimental setup. We run each model 10 times on each question, sampling with temperature 0.7, top-p 0.95, and report the average accuracy across the 10 runs. We use `Qwen/Qwen3-30B-A3B-Instruct-250` as the scoring model, adapting the code from Humanity's Last Exam (3). The models were given a budget of 12,288 output tokens (but the models usually finished in a few thousand tokens). We also tested various system prompts and chose the best accuracy for each model among the prompts.

We make the following observation:

• The pattern in scores is very similar to Benchmark 6: LiveCodeBench, with GPT 5 Mini and Grok 4.1 Fast Reasoning performing very well and Claude Haiku 4.5 and Gemini 3 Flash Preview performing significantly worse.

Benchmark 6: LiveCodeBench

LiveCodeBench (14) is a large-scale, execution-based benchmark designed to evaluate code generation models on realistic competitive programming tasks. It consists of several hundred problems sourced from platforms such as Codeforces, LeetCode, and AtCoder, spanning a wide range of algorithmic domains, including data structures, graph algorithms, dynamic programming, and number theory. Solutions are assessed by compiling and executing generated code against test cases under strict correctness constraints. Note that while LiveCodeBench has a time-ordered dataset to help prevent contamination with a model's own training data, we evaluate models on the full set of problems. After preventing contamination, frontiers models like Gemini 3 score just above 90% (2).

Experimental setup. We evaluate all 1055 problems from LiveCodeBench (5). We perform pass@1 evaluation with the repository's default settings: temperature=0.2, and maximum output length of 2048 tokens.

We make the following observations:

• GPT 5 Mini and Grok 4.1 Fast Reasoning performed very well, while Claude Haiku 4.5 and Gemini 3 Flash Preview performed significantly worse.

• This is unlike the previous coding benchmark, Benchmark 5: HumanEval, where all models scored above 90%. This indicates that LiveCodeBench might be a better benchmark to judge frontier model performance.

Benchmark 5: HumanEval

The HumanEval benchmark (11) assesses code generation capabilities by making models complete Python function implementations given a function signature and docstring. It consists of 164 handwritten programming problems covering fundamental algorithms, data structures, string manipulation, and mathematical operations. We use the EvalPlus framework(13), which extends the original benchmark with comprehensive test suites containing approximately 80 times more test cases per problem to rigorously evaluate both functional correctness against edge cases. HumanEval is a valuable benchmark as its compact, specification-based challenges effectively measure how well a model can convert natural language descriptions into working code. It offers clear insights into its reasoning capabilities and ability to generalize when generating programs.

Experimental setup. We evaluate all 164 problems from HumanEval, using the EvalPlus library (4), which extends the original benchmark with comprehensive test suites. For pass@k evaluation, we generate k=10 samples per problem with temperature=0.2, yielding 1,640 total samples. Results are presented as pass@10 metrics. Note that the models are given a maximum output length of 2048 tokens.

We make the following observation:

• All models performed very well, scoring between 91% and 96%. As with Benchmark 4: MGSM, the HumanEval dataset is likely a part of the models' training data.

Benchmark 4: Multilingual Grade School Math Benchmark (MGSM)

The Multilingual Grade School Math (MGSM) benchmark (17) assesses mathematical reasoning abilities across eleven languages. It is built from 250 elementary-level math word problems drawn from the English GSM8K dataset (10), which were professionally translated into Bengali, Chinese, French, German, Japanese, Russian, Spanish, Swahili, Telugu, and Thai. The tasks involve multi-step arithmetic reasoning that a middle-schooler should be able to solve.

Experimental setup. We evaluate all 250 problems in each of the eleven languages, yielding 2,750 total instances, and score performance using exact match on the final extracted numerical answer after normalizing for commas and trailing decimal zeros. We use the evaluation scripts from the `openai/simple-evals` repository (7), adapted for local model inference using vLLM. Prompts are formatted in the target language with language-specific instructions that request reasoning followed by a numeric answer in a specified format. Models are sampled with greedy decoding (temperature=0.0) and a maximum generation length of 2,048 tokens. Reported results correspond to the mean accuracy across the 11 languages.

We make the following observation:

- All models performed very well, scoring between 89% and 92%. The MGSM dataset is likely a part of the models' training data.

Benchmark 3: IF Bench

The Instruction Following benchmark (IFBench) from AllenAI assesses instruction-following across diverse tasks like counting, formatting, and text manipulation (15). The models are evaluated on how well they follow constraints in their responses, such as "Include keyword beneath in the 23rd sentence, as the 34th word of that sentence," "Use at least 3 different coordinating conjunctions in the response," or "The second word in your response and the second to last word in your response should be the word vibrant."

Experimental setup. We use the 294-question single-turn dataset with 5 repetitions per question, evaluated via pass@1 scoring (1). Responses are scored using the repository's official evaluation code in loose mode, which tolerates formatting variations and extraneous text by testing multiple output formats (removing leading/trailing lines, asterisks, etc.). Results reflect average prompt-level accuracy across all questions and runs. In our experiments, models are sampled with temperature 0.7, top-p 0.95, and a 4096 token limit.

We make the following observations:

- Gemini 3 Flash Preview bounced back with the highest score. This is the first benchmark where Grok 4.1 Fast Reasoning is not at the top.
- Claude Haiku 4.5 performed significantly worse than the others.

Benchmark 2: GPQA Diamond

The GPQA (Graduate-Level Google-Proof Q&A) Diamond dataset consists of highly challenging multiple-choice problems in biology, physics, and chemistry, requiring scientific knowledge and reasoning abilities to answer. Each question is authored by subject-matter experts and is intentionally formulated to be difficult for non-experts to solve, even with access to online resources. The most difficult "Diamond" split includes 198 questions for which both expert annotators selected the correct answer, while most non-experts did not.

Experimental setup. We evaluate the models following the guidelines from the benchmark authors (16) and using the OpenAI simple-eval library (7). Every question is presented 10 times, each time with the answer choices in a different randomized order. The models are allowed a maximum output length of 2028 tokens.

We make the following observations:

- Grok 4.1 Fast Reasoning achieves the highest score again.
- Claude Haiku 4.5 has a good score in the first benchmark, but scored the lowest in this one.

Benchmark 1: MMLU-Pro

MMLU-Pro (19) is an enhanced benchmark for evaluating language models, building on the original MMLU dataset with significantly harder, reasoning-intensive questions and 10 answer options instead of 4. It contains over 12,000 curated questions from academic sources spanning 14 fields, including Biology, Computer Science, Mathematics, Physics, and Law. Experiments demonstrate MMLU-Pro substantially increases difficult with model accuracies dropping 16–33% compared to standard MMLU. Notably, Chain-of-Thought prompting yields greater improvements on MMLU-Pro than on the original benchmark, indicating the dataset demands deeper, more structured reasoning.

We evaluate the models following the guidelines from the benchmark authors (on `TIGER-Lab/MMLU-Pro repository` (9)). The models are allowed a maximum output length of 2048 tokens.

We observed the following from the performance of the Delphi entrant models.

- Gemini 3 Flash Preview performed significantly worse than the other three, scoring around 52%, whereas the remaining models scored between 78% and 85%.

References

[1] allenai/IFBench: A new, challenging benchmark for precise instruction following. https://github.com/allenai/IFBench.

[2] Artificial Analysis: LiveCodeBench Leaderboard. https://artificialanalysis.ai/evaluations/livecodebenchl.

[3] cais/hle: a multi-modal benchmark at the frontier of human knowledge, designed to be the final closed-ended academic benchmark of its kind with broad subject coverage. https://huggingface.co/datasets/cais/hle.

[4] evalplus/evalplus: Rigourous evaluation of LLM-synthesized code. https://github.com/evalplus/evalplus/tree/master.

[5] LiveCodeBench/LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. https://github.com/LiveCodeBench/LiveCodeBench.

[6] OpenAI: Introducing GPT-5. https://openai.com/index/introducing-gpt-5/.

[7] OpenAI simple-eval: a lightweight library for evaluating language models. https://github.com/openai/simple-evals/tree/main.

[8] scicode-bench/SciCode: A benchmark that challenges language models to code solutions for scientific problems. https://github.com/scicode-bench/SciCode.

[9] TIGER-AI-Lab/MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark. https://github.com/TIGER-AI-Lab/MMLU-Pro, 2024.

[10] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[11] Xueying Du, Mingwei Liu, Kaixin Wang, Hanlin Wang, Junwei Liu, Yixuan Chen, Jiayi Feng, Chaofeng Sha, Xin Peng, and Yiling Lou. Evaluating large language models in class-level code generation. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, ICSE '24, New York, NY, USA, 2024. Association for Computing Machinery.

[12] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[13] Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[14] Alex Gu Wen-Ding Li Fanjia Yan Tianjun Zhang Sida Wang Armando Solar-Lezama Koushik Sen Ion Stoica Naman Jain, King Han. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint*, 2024.

[15] Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. Generalizing verifiable instruction following, 2025.

[16] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

[17] Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. Language models are multilingual chain-of-thought reasoners, 2022.

[18] Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. Scicode: A research coding benchmark curated by scientists, 2024.

[19] Y. Wang, X. Ma, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. Ren, A. Arulraj, X. He, Z. Jiang, T. Li, M. Ku, K. Wang, A. Zhuang, R. Fan, X. Yue, and W. Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.