STEER-ME: Assessing the Microeconomic Reasoning of LLMs

Narun Raman, Taylor Lundy, Thiago Amin, Kevin Leyton-Brown, Jesse Perla

University of British Columbia

Beyond Single-Number Benchmarks

LLMs are now evaluated on dozens of benchmarks, but it's often not easy to diagnose or navigate results on a bag of questions which are aggregated into a single number.

For applications involving decision-making and trade-offs, users need to:

- measure performance across a structured space of concepts
- see how robust models are to contexts and numeric changes
- diagnose failure modes

Why Microeconomics?

- **Used in practice:** People already ask LLMs to explain price changes, policy effects, and personal finance decisions
- **Rich but structured:** Demand, equilibrium, and welfare can stress models, but there is always a right answer
- **Codifiable:** These right answers can be solved by programs

Comprehensive by Design

We taxonomized the space of non-strategic microeconomics to get broad coverage of reasoning tasks:

Consumption Decisions

No. of elements: 22
No. of types: 14
No. of questions: 3,295,770

Production Decisions

No. elements: 16
No. of types: 20
No. of questions: 1,333,330

Multi-Agent Decisions

No. elements: 10
No. of types: 6
No. of questions: 750,060

Evaluating Equilibria

No. elements: 10
No. of types: 5
No. of questions: 698,370

For each (element, type) tuple, we created questions in different:

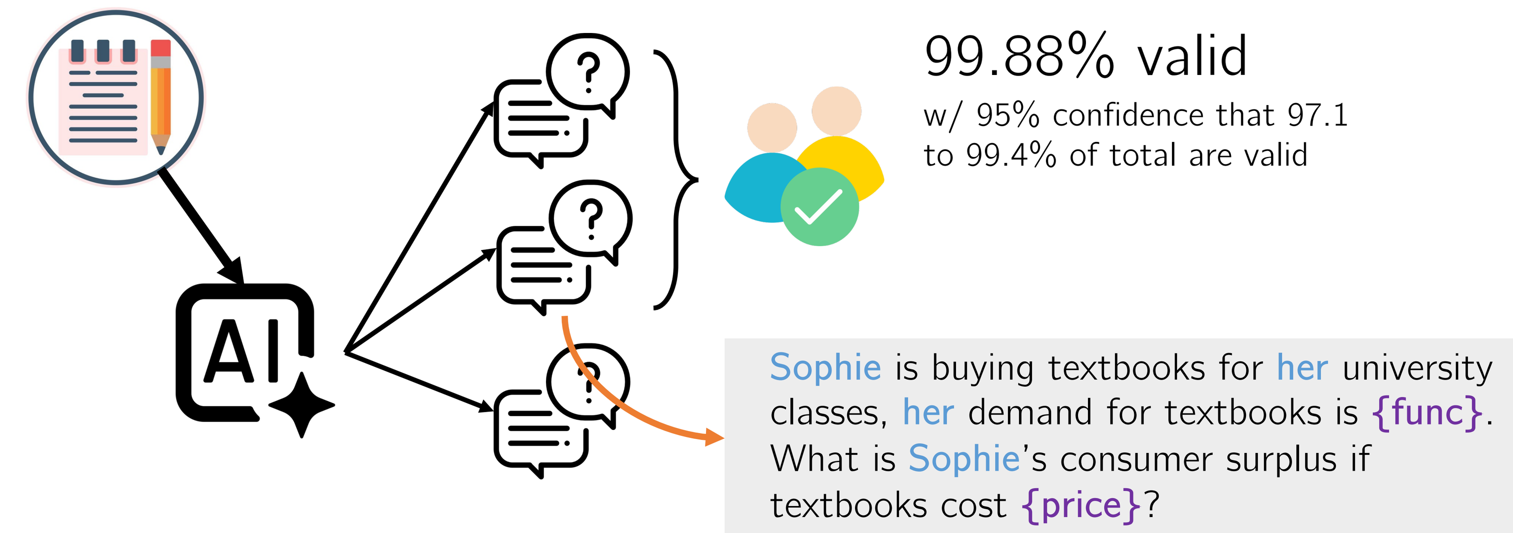
- **Domains:** finance, healthcare, public policy, etc.
- **Perspectives:** 1st/2nd/3rd person

Each question was then instantiated in 50 different numerical parameterizations!

The STEER-ME Benchmark

Each question in STEER-ME is:

1. **Auto-generated.** We developed an LLM-based pipeline, called auto-STEER, that turns a small set of hand-written templates into many domains, perspectives, and numeric instances.



2. **Solver-backed.** Each question has a fully realized program that maps numerical parameters to the correct answer

```python
def pv_correct(cash_flows: list[float], r: float) -> float:
    """
    Present value of a stream of cash_flows at interest rate r.
    cash_flows[t] is the cash flow at period t (t = 0,1,2,...).
    """
    return sum(cf / ((1 + r) ** t) for t, cf in enumerate(cash_flows))
```

Because this logic lives in code, we can implement incorrect solvers whose outputs match incorrect LLM responses for diagnosis:

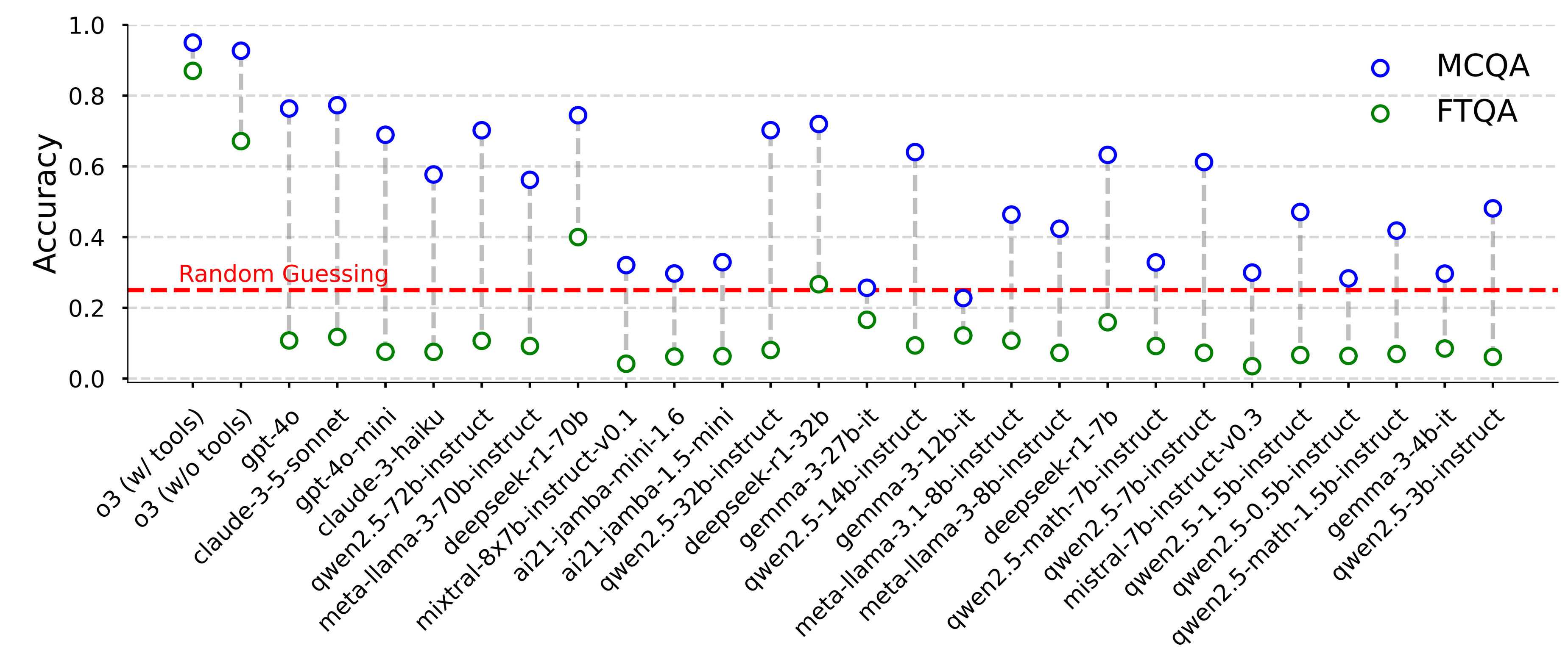
```python
def pv_incorrect(cash_flows: list[float], r: float) -> float:
    """
    Incorrect strategy: collapse multi-period cash flows into a
    single period and ignore discounting entirely.
    """
    return sum(cash_flows)
```

3. **Multi-formatted.** Every question appears in MCQA and FTQA form to evaluate whether ability to pick the right option correlates with reasoning from first-principles

Sophie is buying textbooks for her university classes, her demand for textbooks is $P = 152 - 5Q$. What is Sophie's consumer surplus if textbooks cost \$102?
A. \$125
B. \$500
C. \$250
D. \$50

FTQA does not include the multiple-choice options

Result 1: Good at picking, mixed at reasoning



Partly explained by random guessing, partly by format-specific strategies:

1. **Option gaming:** LLMs plug in the options directly into the functions in the question and pick whichever yielded the best result
2. **Contextual anchoring:** answer choices serve as an implicit signal to LLMs, guiding them toward the correct answer

Result 2: Often made incorrect simplifications

1. In 36.2% of Claude 3.5 Sonnet's incorrect FTQA responses to Intertemporal Consumption Smoothing the model collapsed multi-period cash flows into a single period
   a. *GPT-4o, Claude 3.5 Sonnet, and DeepSeek models ignored crucial aspects of the problem (e.g., risk preferences) between 30-40% of the time*
2. Similarly, most models solved for Marshallian demand rather than Hicksian when asked (51.2% of the time for GPT-4o), potentially because Hicksian is less-taught in texts
3. Surprisingly, not even the biggest closed-source models, except reasoning models, consistently computed Deadweight Loss, a task whose only math requirement is computing the area of a triangle!