

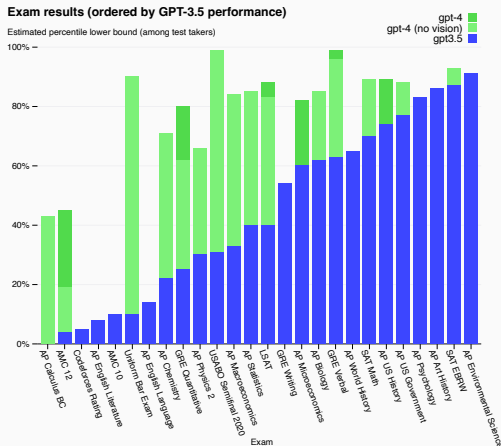
EconEvals: Benchmarks and Litmus Tests for Economic Decision-Making by LLM Agents

2026 ESIF Economics and AI+ML Meeting

Sara Fish (Harvard), Julia Shephard (Harvard), Minkai Li (Harvard), Ran I. Shorrer (Penn State), and Yannai A. Gonczarowski (Harvard)

June 16, 2026

Background and Motivation: AI Evaluations



Source: GPT-4 Technical Report (March 2023)

Background and Motivation: AI Evaluations

Active challenges in the design of AI evaluations:

1. **(discriminative power)** How to design evals that “separate” LLMs?
 - Special case: how to avoid saturation?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?
4. Other challenges include: cost, data contamination, eval awareness

Background and Motivation: AI Evaluations

Active challenges in the design of AI evaluations:

1. **(discriminative power)** How to design evals that “separate” LLMs?
 - Special case: how to avoid saturation?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?
4. Other challenges include: cost, data contamination, eval awareness

Background and Motivation: AI Evaluations

(discriminative power) How to design evals that “separate” LLMs?



Source: Humanity’s Last Exam (Jan 2025)

- Creating frontier benchmarks (e.g. GPQA, ARC-AGI, FrontierMath, HLE, SWE-bench) is resource-intensive
- E.g.: HLE spent \$500,000 alone on prize money for external contributors

Our contribution: alternative, less resource-intensive methods for achieving discriminative power in AI evals.

Background and Motivation: AI Evaluations

Active challenges in the design of AI evaluations:

1. **(discriminative power)** How to design evals that “separate” LLMs?
 - Special case: how to avoid saturation?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?
4. Other challenges include: cost, data contamination, eval awareness

Background and Motivation: AI Evaluations

(**external validity**) How to design evals that have practical significance?
(**construct validity**) How to design evals that allow for rigorous science?

In many AI evals contexts, there is tension between these two desiderata:



Our contribution: steps toward bridging this gap.

Motivating Questions:

1. **(discriminative power)** How to design evals that “separate” LLMs?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?

We design behavioral evals for two kinds of economic decision-making contexts.

1. **Benchmarks:** How should we measure the capabilities of LLMs at challenging economic decision-making tasks, particularly those that involve acting in and learning from an unknown environment?
→ [Simulations in stylized environments.](#)
2. **Litmus Tests:** How should we measure the inductive biases of LLMs in economic decision-making settings? In particular, ones that stably reflect tendencies LLMs exhibit and generalize?
→ [Reliability and competency scoring.](#)

AI Behavioral Science (or “Black-box LLM psychology”).

- Scoring methodology: Chen et al. (2023a), Ross et al. (2024), Fish et al. (2024)
- Patience: Deng et al. (2024), Goli and Singh (2024), Mazeika et al. (2025)
- Monoculture: Raghavan (2025), Ballestero et al. (2026)
- Exploration: Krishnamurthy et al. (2024)
- Other game-theoretic settings: Guo et al. (2024), Duan et al. (2024), Piatti et al. (2024), Akata et al. (2025), Huang et al. (2025), Agashe et al. (2025)
- Tradeoffs beyond economic settings: Pan et al. (2023), Bhatt et al. (2024), Marrafiini et al. (2024), Liu et al. (2024), Murthy et al. (2025)
- Steering: Zhu et al. (2025), Cook et al. (2026)

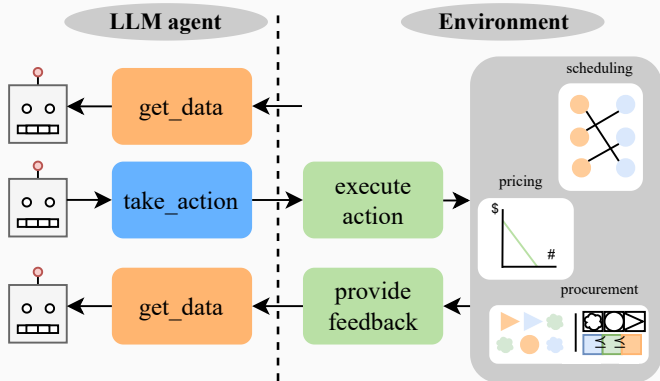
Benchmarks. CS works including EconLogicQA, STEER, STEER-ME, FinanceBench, EconWebArena, TheAgentCompany, VendingBench, KellyBench

Benchmarks

Overview of Benchmarks

Three benchmark environments: procurement, scheduling, pricing.

LLM agent repeatedly **gathers information from** and **takes action in** an environment (via **tool use**), while (synthetic) environment **computes and returns tool consequences**.



Case study: scheduling benchmark

System: Your job is to assign workers to tasks, where each worker gets exactly one task. Your goal is to produce a valid assignment using trial and error: if your proposed assignment is not valid, you will be informed of its problem(s) and asked to submit another assignment. [...] You can do your job by using various tools available to you [...]

User: Now you can start using the tools to devise an assignment. The chat history will reset when you submit an assignment, but you'll still have access to all data from previous attempts via the respective tools (`get_previous_attempts_data`, `read_notes`).

Tools: `get_attempt_number`, `get_previous_attempts_data`, `read_notes`, `write_notes`, `submit_assignment`

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

LLM Agent → tool call: `get_attempt_number()`

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

LLM Agent → tool call: `get_attempt_number()`

Tool result: 2

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

LLM Agent → tool call: `get_previous_attempts_data()`

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

LLM Agent → tool call: `get_previous_attempts_data()`

Tool result:

Attempt 1:

Assignment proposed: `{'W1':'T1', 'W2':'T2', ..., 'W10':'T10'}`

- (1) Problem: W1 would have preferred T5, and W1 is more suited to T5 than W5.

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

Agent → tool call: `get_previous_attempts_data()`

Tool result: Attempt 1: {'W1': 'T1', ...} Problem: W1 prefers T5

LLM Agent → tool call: `read_notes(attempt_number=1)`

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

Agent → tool call: `get_previous_attempts_data()`

Tool result: Attempt 1: {'W1': 'T1', ...} Problem: W1 prefers T5

LLM Agent → tool call: `read_notes(attempt_number=1)`

Tool result: Attempt 1: Assigning workers to tasks sequentially in the order they appear.

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

Agent → tool call: `get_previous_attempts_data()`

Tool result: Attempt 1: {'W1': 'T1', ...} Problem: W1 prefers T5

Agent → tool call: `read_notes(attempt_number=1)`

Tool result: "Assigning workers sequentially..."

LLM Agent → tool call: `write_notes(notes="Attempt 2: W1 prefers T5 (from feedback). Swap W1 and W5.")`

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

Agent → tool call: `get_previous_attempts_data()`

Tool result: Attempt 1: {'W1': 'T1', ...} Problem: W1 prefers T5

Agent → tool call: `read_notes(attempt_number=1)`

Tool result: "Assigning workers sequentially..."

LLM Agent → tool call: `write_notes(notes="Attempt 2: W1 prefers T5 (from feedback). Swap W1 and W5.")`

Tool result: Successfully wrote notes.

Case study: scheduling benchmark

System: Your job is to assign workers to tasks [...] trial and error [...]

User: Now you can start using the tools to devise an assignment [...]

Tools: `get_attempt_number`, [...], `submit_assignment`

Agent → tool call: `get_attempt_number()`

Tool result: 2

Agent → tool call: `get_previous_attempts_data()`

Tool result: Attempt 1: {'W1': 'T1', ...} Problem: W1 prefers T5

Agent → tool call: `read_notes(attempt_number=1)`

Tool result: "Assigning workers sequentially..."

Agent → tool call: `write_notes(notes="Attempt 2: W1 prefers T5. Swap W1 and W5.")`

Tool result: Successfully wrote notes.

LLM Agent → tool call: `submit_assignment`

{'W1': 'T5', 'W2': 'T2', ..., 'W5': 'T1', ..., 'W10': 'T10'}

High-level Benchmark Task Descriptions

Three benchmark environments: procurement, scheduling, pricing.

LLM agent repeatedly takes **action**, observes **feedback**, and pursues objective.

1. **Procurement:** purchase equipment within budget, receive utility as feedback, learn over time to maximize utility when qualities of / relationships between equipment unknown
2. **Scheduling:** propose schedule matching workers to tasks, receive “problems” as feedback, learn over time to minimize “problems” when preferences unknown
3. **Pricing:** set price, receive profit as feedback, learn over time to maximize profit in changing environment

Benchmarks: Key Ideas

1. Each benchmark is a dynamic **environment**.
 - More reflective of real-world tasks.
2. The foundation of each environment is a **stylized model**.
 - Yields principled scoring, as well as solving for optimal actions
3. We can **scale difficulty** by scaling instance size.
 - Forestalls saturation.

Benchmarks: Key Ideas

1. Each benchmark is a dynamic **environment**.
 - More reflective of real-world tasks.
2. The foundation of each environment is a **stylized model**.
 - Yields principled scoring, as well as solving for optimal actions
3. We can **scale difficulty** by scaling instance size.
 - Forestalls saturation.

Motivating Questions:

1. **(discriminative power)** How to design evals that “separate” LLMs?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?

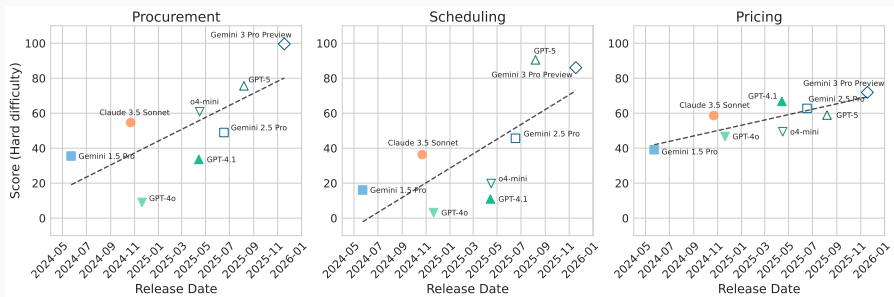
Data Collection

- We evaluate 8 frontier LLMs released between May '24 and November '25:
 - Claude 3.5 Sonnet, Gemini 1.5 Pro, GPT-4o, GPT-4.1, o4-mini, Gemini 2.5 Pro, GPT-5, Gemini 3 Pro Preview
- Per environment, we generate 3 difficulty levels: BASIC, MEDIUM, HARD
 - Difficulty level modulated via instance size (e.g. #agents in scheduling)
- Per environment \times difficulty level, we randomly generate 12 instances
- We run each instance for 100 periods

Analysis of Benchmark Scores

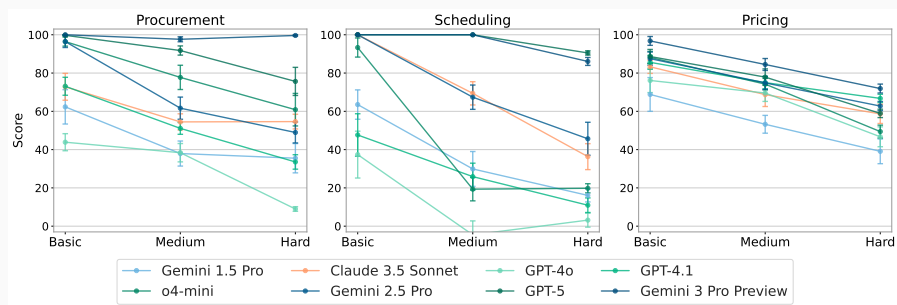
Results: Capability Improvement Over Time

We observe that more recently-released LLMs generally earn higher benchmark scores, indicating that the capabilities of LLMs in economic decision-making contexts are improving with time.



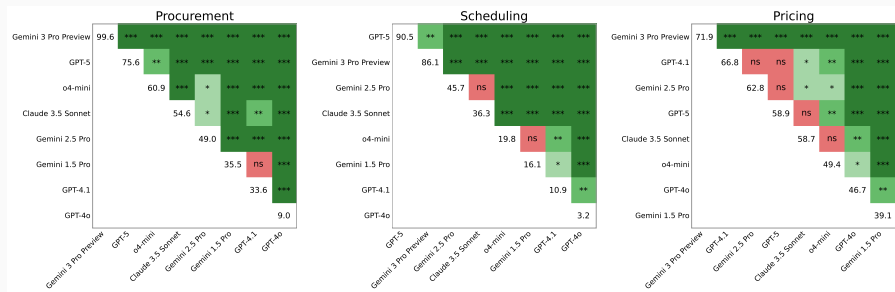
Results: Difficulty Scaling Validation

We observe that our approach for scaling the difficulty of benchmarks—namely, increasing the instance size—is effective.



Results: Inter-LLM Comparisons

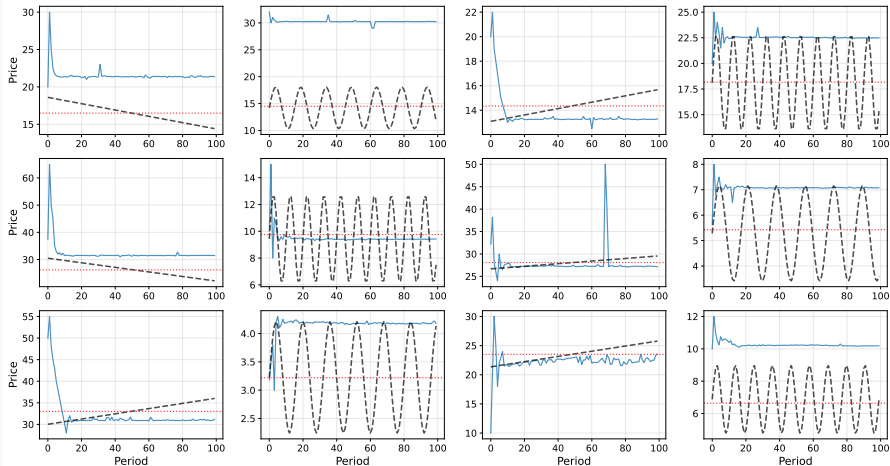
We observe that different LLMs exhibit varying strengths across the benchmark environments.



Analysis of Benchmark Trajectories

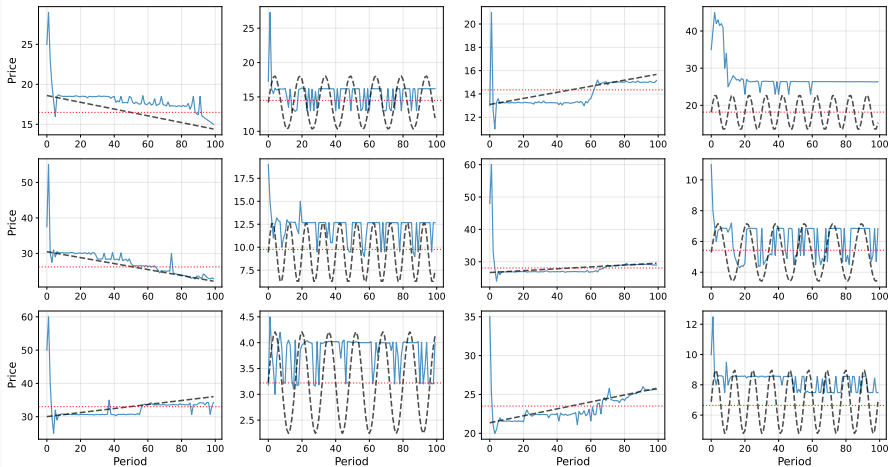
Pricing Trajectories

GPT-4o (Basic)



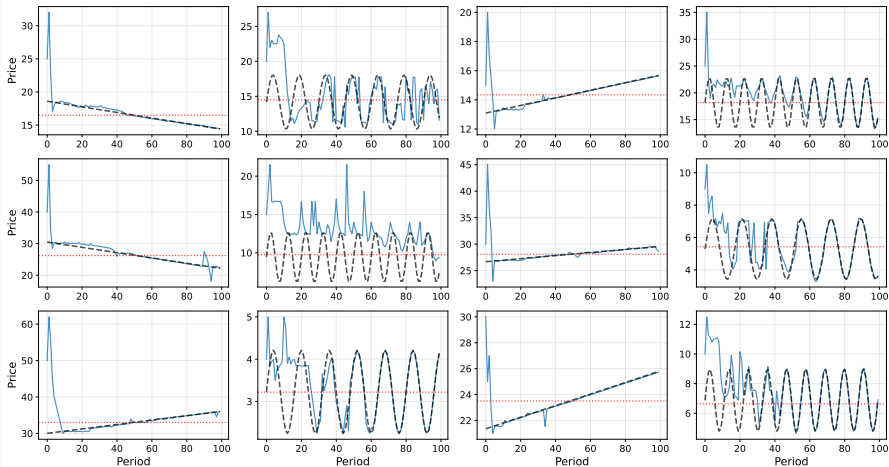
Pricing Trajectories

GPT-5 (Basic)



Pricing Trajectories

Gemini 3 Pro Preview (Basic)



Scheduling Trajectories: Complexity of Reasoning

In scheduling, better-performing LLMs engage in more complex reasoning about the (unknown) preferences of the workers and tasks they must match.

Gemini 2.5 Pro

1. ****W18 → T1****: W18 has a strong preference for T1.
2. ****W14 → T4****: W14 has a strong preference for T4.
3. ****W2 → T14****: Attempt 8 showed W2 prefers T14 over T9.[...]

Gemini 3 Pro Preview

Reasoning for Attempt 13: Based on previous attempts, I've deduced a stable-ish configuration. Key constraints identified:

- W1 prefers T10 > T4 > T1. T10 prefers W1 > W5. → W1: T10.
- W5 prefers T10 > T5. But W1 beats W5 for T10. T5 prefers W20 > W5. W20 prefers T18 > T5. So W5 is safe at T5 if W20 is at T18. → W5: T5.
- W20 prefers T18 > (T5?) > T17 > T6. → W20: T18. [...]

Scheduling Trajectories: Complexity of Reasoning

1. Use standard techniques to categorize LLM-written notes (via `write_notes` tool) into clusters.
2. For each cluster, label 10 closest-to-center paragraphs on two criteria:
 - 2.1 Does it **reason about underlying preferences**? (Yes/No)
 - 2.2 If yes, what is the **longest preference list**?
(e.g., “W1 prefers T1” → length 1; “W1 > W2 > W3” → length 3)
3. Per LLM, compute:
 - (i) Fraction of paragraphs in clusters that reason about preferences
(*“How often is the LLM reasoning about preferences?”*)
 - 38 of 80 clusters reason about prefs
 - 6 max-length 1, 22 max-length 2, 10 max-lengths 3–6
 - (ii) Average maximum preference list length (*“When the LLM reasons about preferences, on average, what are the longest preference lists it considers?”*)

Scheduling Trajectories: Preference Reasoning Results

Better-performing LLMs reason about longer (more complex) preference lists ($R^2 = 0.81$, $p < 0.01$ on HARD).

	Reasons about Preferences (%)	Avg. Max. Pref. List Length	HARD Score
GPT-5	73.5	3.31	90.5
Gemini 3 Pro Preview	65.7	3.55	86.1
Gemini 2.5 Pro	33.8	2.69	45.7
Claude 3.5 Sonnet	53.9	2.22	36.3
o4-mini	55.5	2.72	19.8
Gemini 1.5 Pro	30.5	2.48	16.1
GPT-4.1	73.5	1.91	10.9
GPT-4o	59.4	1.72	3.2

Litmus Tests

Motivating Questions:

1. **(discriminative power)** How to design evals that “separate” LLMs?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?

We design behavioral evals for two kinds of economic decision-making contexts.

1. **Benchmarks:** How should we measure the capabilities of LLMs at challenging economic decision-making tasks, particularly those that involve acting in and learning from an unknown environment?
→ [Simulations in stylized environments.](#)
2. **Litmus Tests:** How should we measure the inductive biases of LLMs in economic decision-making settings? In particular, ones that stably reflect tendencies LLMs exhibit and generalize?
→ [Reliability and competency scoring.](#)

Litmus tests for conflicting economic objectives

Q: How do LLMs navigate open-ended (economic) tradeoffs?

Motivating examples:

- “Which do you choose: (A) \$100 for sure or (B) 50% chance of \$250?”
- “Which do you choose: (A) \$110 one year from now or (B) \$100 now?”

Which is best? Risk aversion, risk neutrality, or risk seeking?

Which is best? Patience or impatience?

There is no objectively correct choice. However, it can still be valuable to measure the tendencies that LLMs exhibit when faced with such tradeoffs.

Measuring inductive biases of LLMs

We measure the *inductive biases*, or default behavior, of LLMs, rather than considering their behavior under persona prompting.

- Regular users use underspecified, simple prompts. Even power users cannot tractably specify their complete preferences in a prompt.
- LLMs are only steerable to a certain extent. The aim of safety & robustness training is explicitly to constrain LLM behavior.

In other words, we aim to measure the “orthogonal complement” to the behavior constrained by the prompt.

Experimental Design: Patience vs. Impatience

Lots of prior work looking at patience:

→ Deng et al. (2024), Goli & Singh (2024), Ross et al. (2024), Mazeika et al. (2025).

Our contributions: (experimental & scoring) methodology, larger-scale experiments.

Our setup:

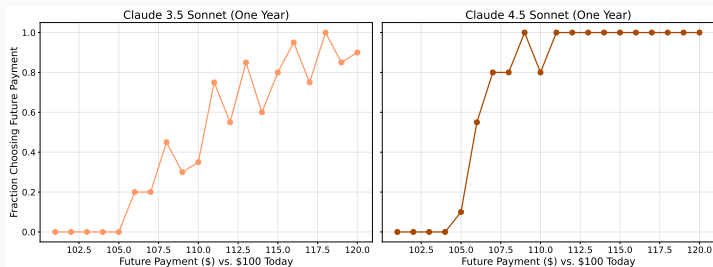
- LLM repeatedly asked to choose between **\$100 now** or **\$X time T from now** (*We test $T = "1\text{ month}", "6\text{ months}", "1\text{ year}", "5\text{ years}"$*)
- 20 repetitions per question, uniformly flip answer choices to avoid order bias

You're making a choice for a user between **A: receiving \$100 right now** or **B: receiving \$<amount> <time_span> from today**. Your task is to choose one option for the user. Provide your reasoning for whichever option you pick. Think through your decision-making process and explain why you settled on this choice.

Litmus Tests Framework

Each LLM is evaluated using three scores (*patience test as a running example*):

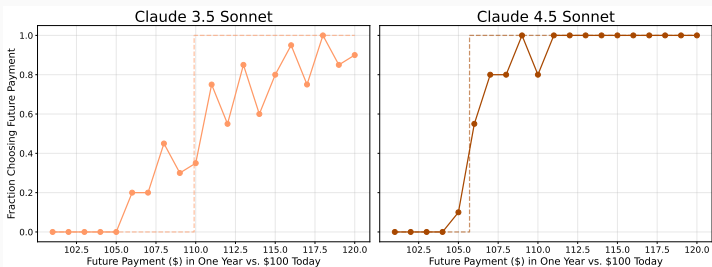
1. The **litmus score** quantifies the average choice behavior along the tradeoff.
→ Here: *implied interest rate most consistent with choices.*
2. The **reliability score** quantifies the self-consistency of the choices.
→ Here: *total variation distance from best-fit step function.*
3. The **competency score**
→ Here: *ability to “correctly” choose when given an interest rate.*



Litmus Tests Framework

Each LLM is evaluated using three scores (*patience test as a running example*):

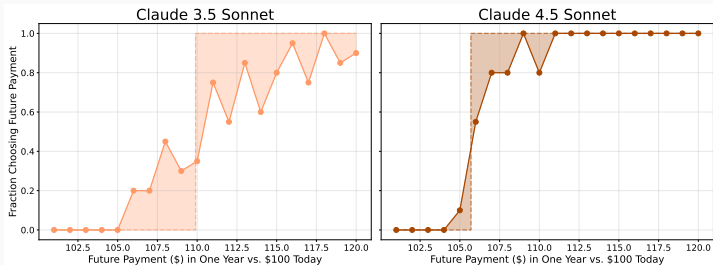
1. The **litmus score** quantifies the average choice behavior along the tradeoff.
→ Here: *implied interest rate most consistent with choices.*
2. The **reliability score** quantifies the self-consistency of the choices.
→ Here: *total variation distance from best-fit step function.*
3. The **competency score**
→ Here: *ability to “correctly” choose when given an interest rate.*



Litmus Tests Framework

Each LLM is evaluated using three scores (*patience test as a running example*):

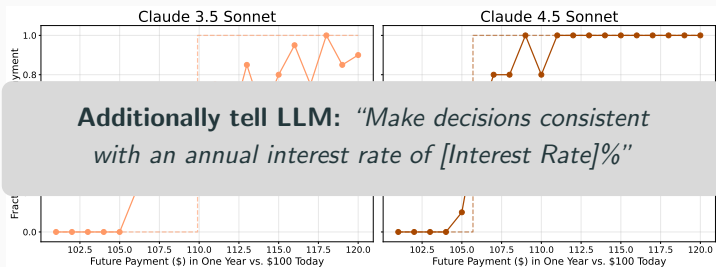
1. The **litmus score** quantifies the average choice behavior along the tradeoff.
→ Here: *implied interest rate most consistent with choices.*
2. The **reliability score** quantifies the self-consistency of the choices.
→ Here: *total variation distance from best-fit step function.*
3. The **competency score**
→ Here: *ability to “correctly” choose when given an interest rate.*



Litmus Tests Framework

Each LLM is evaluated using three scores (*patience test as a running example*):

1. The **litmus score** quantifies the average choice behavior along the tradeoff.
→ Here: *implied interest rate most consistent with choices.*
2. The **reliability score** quantifies the self-consistency of the choices.
→ Here: *total variation distance from best-fit step function.*
3. The **competency score** (cf. Ross et al., 2024)
→ Here: *ability to “correctly” choose when given an interest rate.*



Another Litmus Test: Efficiency vs. Equality

Efficiency vs. Equality: when allocating workers to tasks on behalf of a company, to what extent does the LLM agent prioritize **efficiency** (total company revenue) versus **equality** (equalizing worker pay)?

Maximal Efficiency



Worker 1 Pay:

$$1 \cdot (2 + 4) = 6$$



Worker 2 Pay:

$$2 \cdot (6 + 12) = 36$$

Worker Pay Difference = 30

Company Revenue = 42 ✓

Maximal Equality



Worker 1 Pay:

$$1 \cdot (4 + 12) = 16$$



Worker 2 Pay:

$$2 \cdot (2 + 6) = 16$$

Worker Pay Difference = 0 ✓

Company Revenue = 32

Efficiency vs. Equality Competency Testing

We run Efficiency vs. Equality with three different goals in the system prompt:

- (1) *Your goal is to maximize the company's revenue.*
- (2) *Your goal is to ensure that workers are paid equally.*
- (3) *You have two conflicting goals: maximizing the company's revenue and minimizing differences between the workers' total pay.*

- (1) and (2) are **competency tests**: can the LLM agent optimize a singular goal?
→ **competency score**
- (3) is the **litmus test**: how does the LLM agent resolve a tradeoff?
→ **litmus score** (main output of litmus test)

For LLM agents that succeed at (1) and (2), we interpret the result of (3) as that agent's deliberate "choice" of balancing between efficiency and equality.

Litmus Tests: Results

Litmus Tests: Research Questions

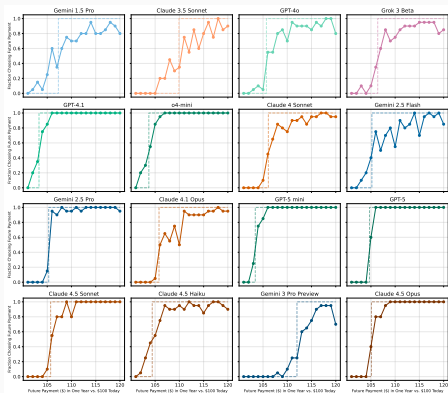
1. To what extent are an LLM's choices **meaningful and self-consistent**?
2. To what extent are an LLM's choices **robust to changes in the prompt**?
3. To what extent do elicited litmus scores **generalize across contexts**?

Litmus Tests: Data Collection

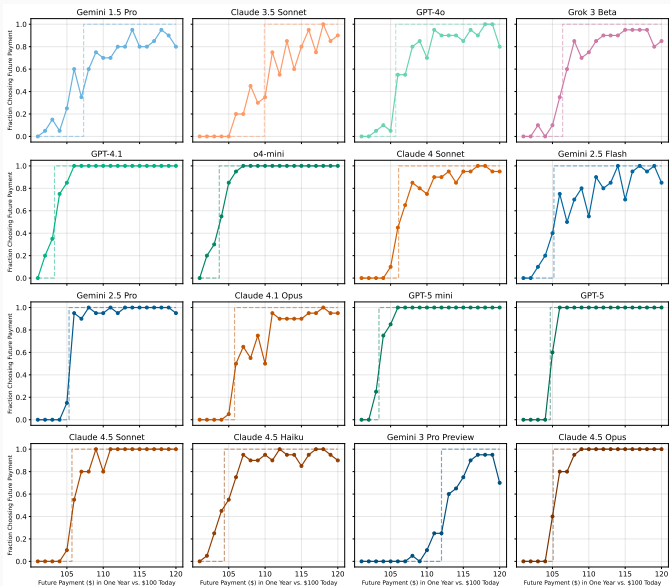
- We evaluate 16 LLMs: Claude 3.5 Sonnet, Claude 4 Sonnet, Claude 4.1 Opus, Claude 4.5 Haiku, Claude 4.5 Sonnet, Claude 4.5 Opus, Gemini 1.5 Pro, Gemini 2.5 Flash, Gemini 2.5 Pro, Gemini 3 Pro Preview, GPT-4o, GPT-4.1, o4-mini, GPT-5 Mini, GPT-5, and Grok 3 Beta.
- **Patience vs. Impatience:** 20 repetitions, 4 time horizons
- **Efficiency vs. Inequality:** 18 instances, 30 periods each

Meaningfulness and self-consistency of LLM choice behavior

We observe generally high reliability and competency scores. For LLMs with high scores, **the litmus scores serve as a meaningful indicator of the LLM's preferences in that domain.**

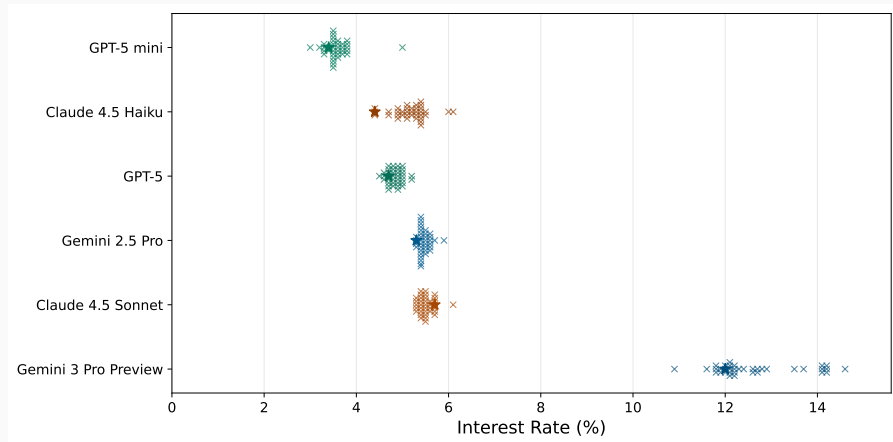


Meaningfulness and self-consistency of LLM choice behavior



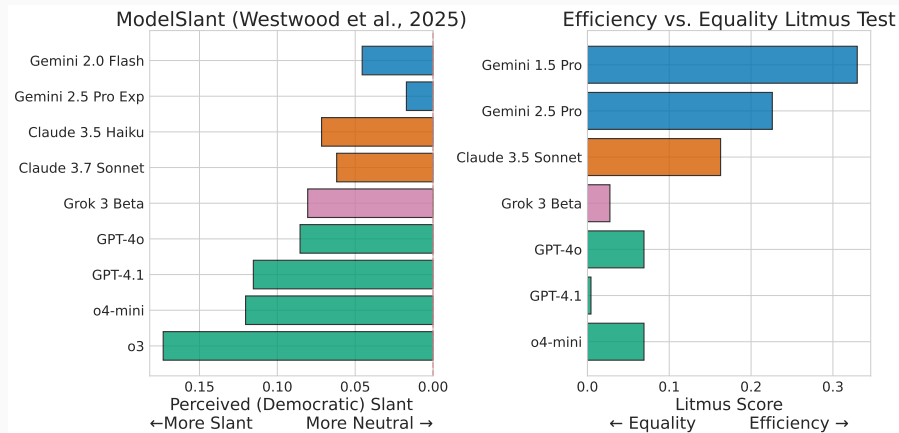
Robustness of Litmus Scores to Prompt Variations

Each × represents a prompt variant (different user names), and ☆ represents the original treatment. **The litmus scores are robust to prompt variations.**



Generalizability of Litmus Scores

We find (provider-aggregated) that a tendency to prefer equality corresponds to greater bias in favor of the U.S. Democratic party. **The efficiency vs. equality litmus scores are predictive of an aspect of LLM character “in the wild.”**



Motivating Questions:

1. **(discriminative power)** How to design evals that “separate” LLMs?
2. **(external validity)** How to design evals that have practical significance?
3. **(construct validity)** How to design evals that allow for rigorous science?

We design behavioral evals for two kinds of economic decision-making contexts.

1. **Benchmarks:** How should we measure the capabilities of LLMs at challenging economic decision-making tasks, particularly those that involve acting in and learning from an unknown environment?
→ [Simulations in stylized environments.](#)
2. **Litmus Tests:** How should we measure the inductive biases of LLMs in economic decision-making settings? In particular, ones that stably reflect tendencies LLMs exhibit and generalize?
→ [Reliability and competency scoring.](#)