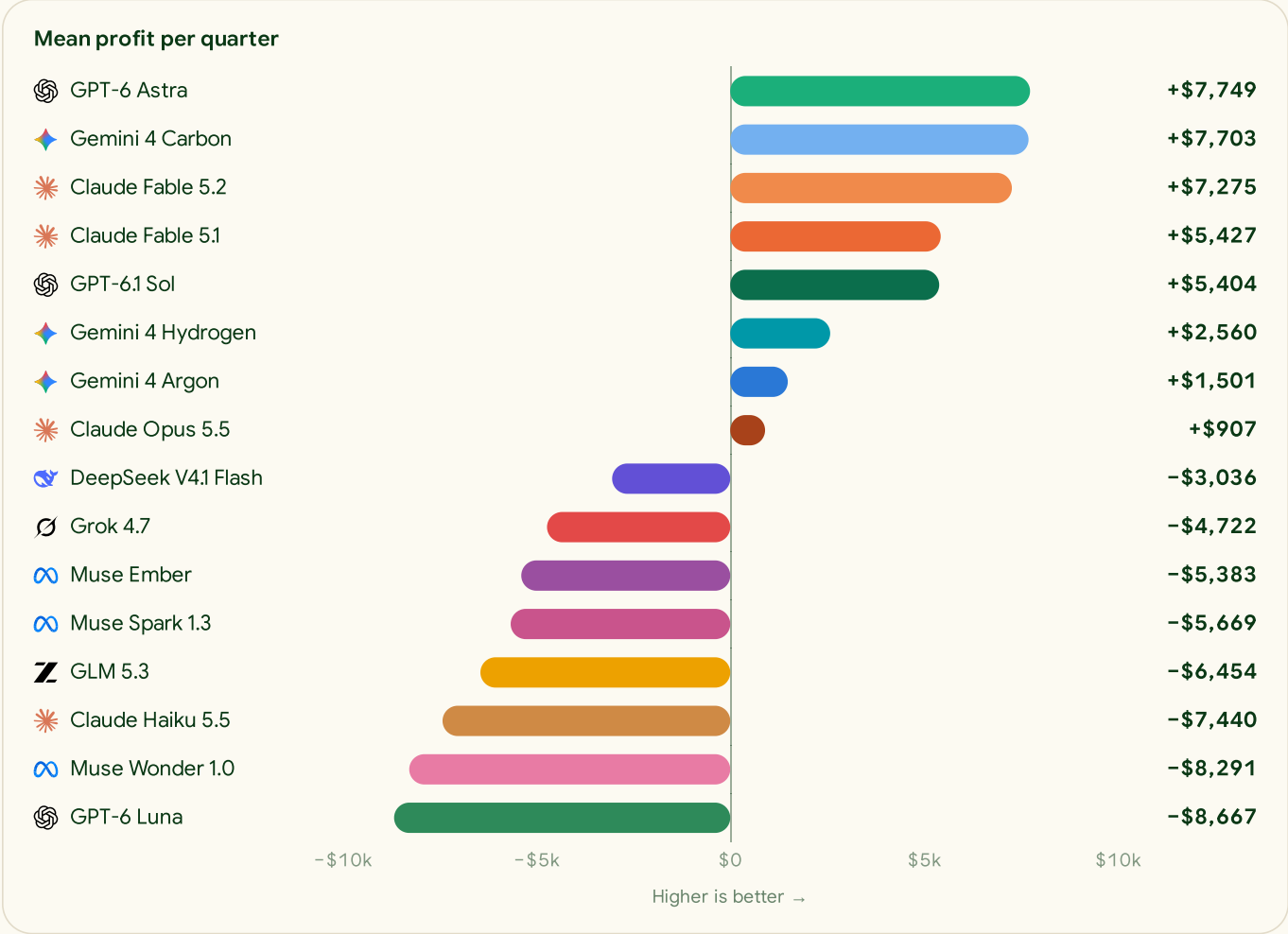
RESEARCH

SMB Bench v1.0

Measuring AI's ability to learn from delayed consequences.

Technical report · 6 November 2026 · 9 models · By ORO AI (Shardul Bansal, Seth Schilbe, Jarrod Barnes)



Environment
A produce wholesaler's buying desk

Quarter
90 buying mornings

Runs
485: 9 models, 18 tasks

Abstract

Placed in a long time horizon simulation, can AI learn from the delayed consequences of its own decisions and successfully run a wholesale buying desk over a 90 day period?

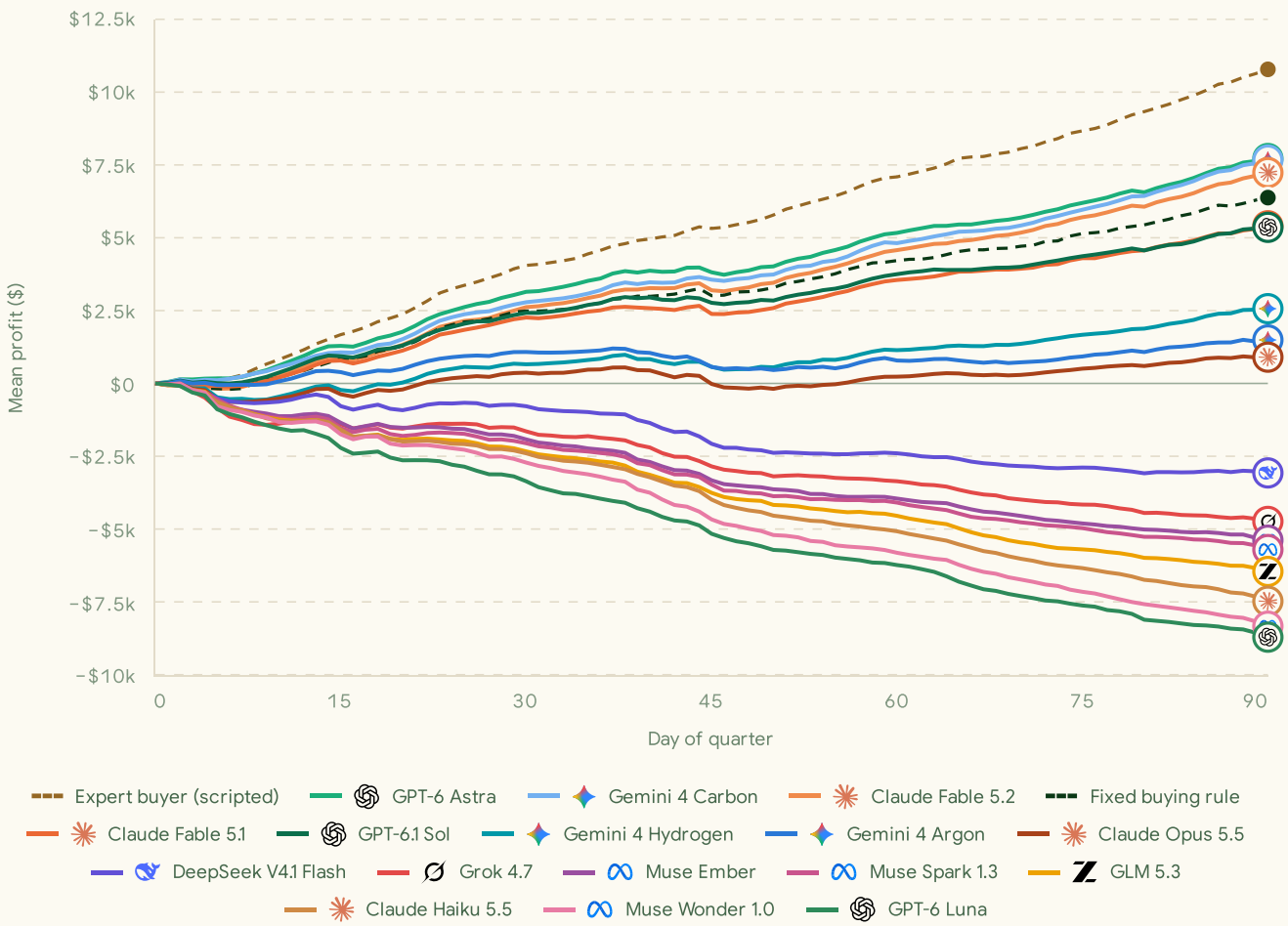
The economic value of agents will soon come from running whole businesses, not from finishing tasks. When you run a business, the consequences of your own decisions arrive weeks later, unlabelled. You have to notice them, trace them to something you did, and act before they compound. Small-to-Midsize Business Bench (SMB Bench) measures the ability for agents to observe changes in the environment from their decisions, requiring learning and adaptive strategy to perform well. Models are tasked with running a produce buying desk for a wholesaler

over a 90-day quarter: keeping customers supplied profitably while demand, suppliers, freight and cash all move, in a partially observable environment.

Leaderboard

Leaderboard · updated 13 November 2026 · 16 models

FIGURE 1 Profit over the quarter — Mean of all 18 tasks



We measure an agent's net profit over a 90-day quarter, in an environment that responds to decisions.
No model reliably beats the fixed rule in each run, and only GPT-6 Astra, Gemini 4 Carbon and Claude Fable 5.2 finish ahead of it on average.
A scripted expert buyer, which sees only what a model sees, makes +$10,795: $3,046 more than the best model.

Leaderboard updates

13 November 2026: Added Gemini 4 Hydrogen.

11 November 2026: Added Claude Fable 5.2, Muse Spark 1.3, Muse Ember, Claude Haiku 5.5, GPT-6 Luna and Gemini 4 Carbon.

Run counts and harness builds: dataset card.

Related work

Much of what makes someone good at a job is tacit, "more than we can tell"[1], and people learn it from what their decisions lead to. Models already learn well from immediate feedback: failed tests, errors and critiques (Reflexion[2], Voyager[3]), and continual-learning benchmarks now measure improvement across task streams (CL-Bench[4], LifelongAgentBench[5]). Learning from delayed feedback, the credit-assignment problem Minsky[6] named

in 1961, is much less measured, and it's hard for people too: in inventory games with delays, most players did worse than a do-nothing rule[7]. Silver and Sutton[8] call this the missing piece in today's systems. Business benchmarks (Vending-Bench[9], CEO-Bench[10], RetailBench[11]) have late consequences but score the final outcome, and STOCKTAKE[12] comes closest by hiding supply problems in a long episode. SMB Bench isolates the delayed case. It plants consequences that surface weeks after the decision, runs the same business without them as a control, and scores each run in the identical world.

The simulation

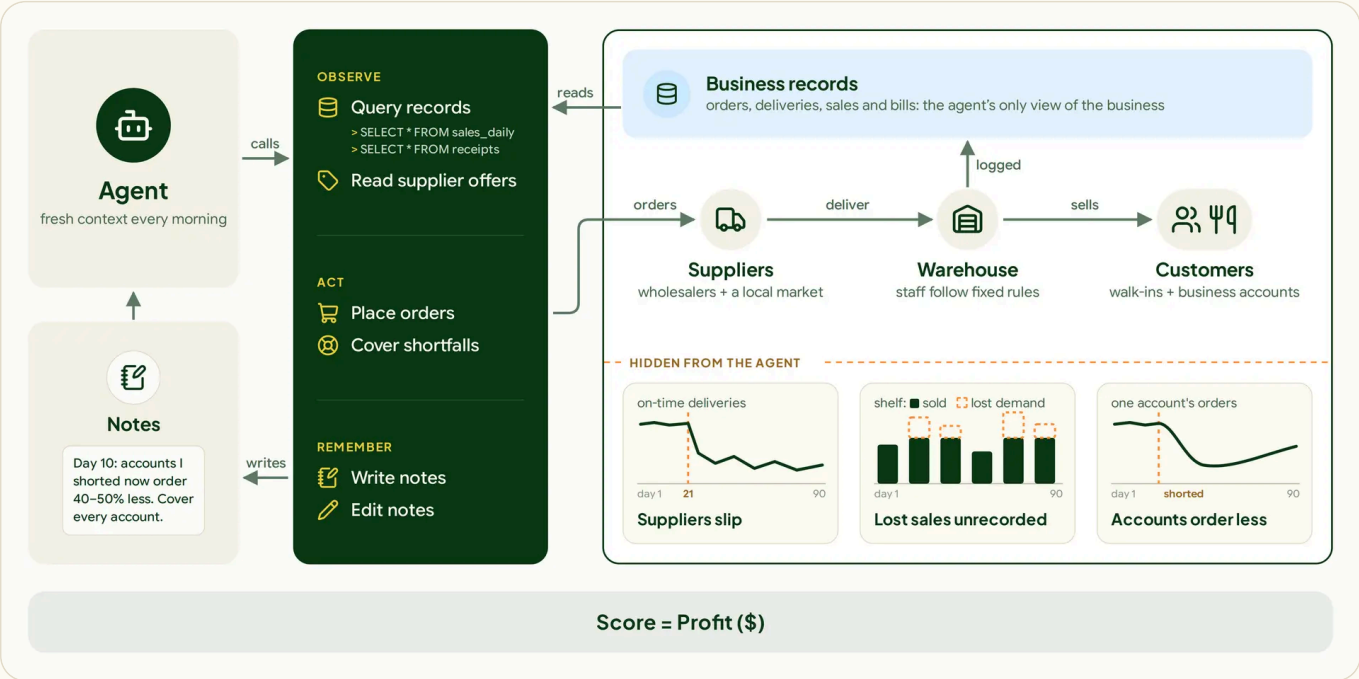


Figure 2. The SMB Bench environment.

SMB Bench puts an agent in charge of buying produce for a small wholesaler for one 90-day quarter. The business sells to walk-in shoppers at shelf price and to restaurants and catering accounts at agreed prices and quantities. Produce is perishable: what doesn't sell is marked down, then thrown away. The business's rules, costs and demand are calibrated to public data and checked against how a real desk runs, with GetASAP and our network of wholesalers. The agent's job is to decide, every morning, what to order from which supplier so that the shelves and the accounts are covered without stock rotting. The agent's profit for the quarter is:

$$\text{Profit} = R - C_{\text{buy}} - C_{\text{short}} - C_{\text{waste}} - C_{\text{ops}} + \Delta V_{\text{stock}} - C_{\text{lost}}$$

R	sales to walk-in shoppers, at shelf or markdown price, and to accounts, at contract price
C_{buy}	purchase orders, plus local-market buys to cover an account the orders missed
C_{short}	5% compensation on every account order delivered short
C_{waste}	expired or rejected stock, written off and disposed
C_{ops}	delivery, receiving, storage, overhead and interest
ΔV_{stock}	change in the value of stock that will sell before it expires
C_{lost}	profit that shorted accounts will no longer bring in after the quarter

Each morning is a fresh conversation. The agent gets a status dashboard, the notes it wrote on earlier mornings, and 60 tool calls. It can query the business's records in SQL (sales, stock, deliveries, supplier offers), read the

day's quotes from several suppliers, place orders, set a standing instruction to “cover” an account it would otherwise short by buying at the local market at a loss, and edit memory files, which are the only thing it carries to the next morning. Everything else runs on fixed rules: staff receive deliveries, sell, price, mark down, discard, and pay the bills.

The difficulty lies in what the agent cannot observe. Demand is censored: once the shelf is empty, shoppers leave and the sale is never recorded, so the books only show what was bought, never what was wanted. The same goes for the slower consequences. An account that was shorted keeps ordering, but will start decreasing order quantity in response. A supplier can start arriving late and short orders without notice. Agents can observe the immediate dollar cost of a decision, but not the long-term delayed effects.

Every business is played three ways: a normal quarter where nothing unusual happens; a customer quarter, where shorted accounts quietly cut their orders; and a supplier quarter, where they do too and the cheapest supplier also starts arriving late and short. The difference between the runs is what those consequences cost the model. Each quarter is run several times to reduce noise.

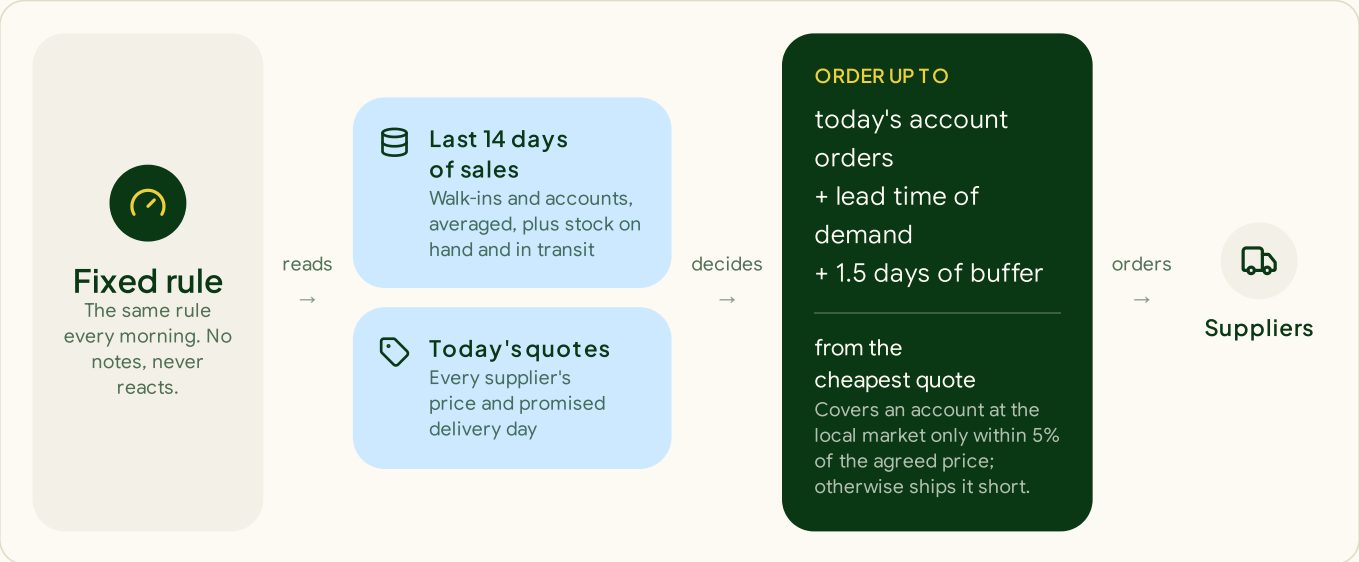


Figure 3. The fixed buying rule.

Each quarter is also played by a fixed buying rule in the same world. Each morning it averages the last 14 days of sales, orders enough from the cheapest quote to cover today's account orders and a day and a half of buffer beyond the delivery lead time. It covers an account at the local market only when that costs within 5% of the agreed price; otherwise it ships the account short. It keeps no notes and reacts to no changes in the environment.

Results

FIGURE 4 Net profit per quarter

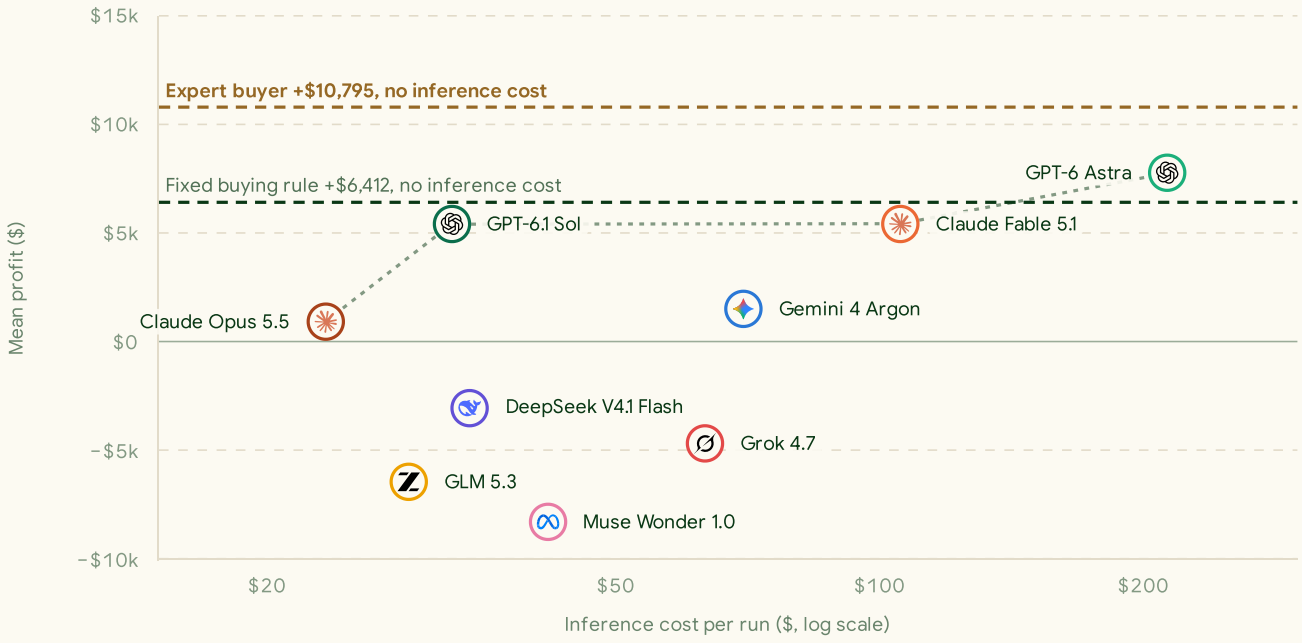
MODEL	ALL 18 TASKS	NORMAL QUARTER	CUSTOMER VS NORMAL	SUPPLIER VS NORMAL
GPT-6 Astra	+$7,749	+$10,145	−$1,336	−$5,851
-- Fixed buying rule	+$6,412	+$10,051	−$3,079	−$7,839
Claude Fable 5.1	+$5,427	+$9,764	−$4,643	−$8,368
GPT-6.1 Sol	+$5,404	+$10,331	−$5,735	−$9,046
Gemini 4 Argon	+$1,501	+$4,597	−$2,088	−$7,199
Claude Opus 5.5	+$907	+$9,401	−$8,215	−$17,266
DeepSeek V4.1 Flash	−$3,036	+$7,745	−$13,796	−$18,545
Grok 4.7	−$4,722	+$5,711	−$13,662	−$17,638
GLM 5.3	−$6,454	+$2,250	−$11,232	−$14,880
Muse Wonder 1.0	−$8,291	+$2,701	−$14,235	−$18,740

Normal quarter Nothing unusual happens. The other two columns are the change from it, in the same businesses.

Customer quarter Accounts a model shorts quietly order less.

Supplier quarter The same, and the cheapest supplier also starts arriving late and short, without notice.

FIGURE 5 Profit against inference cost — As of 6 November (report)

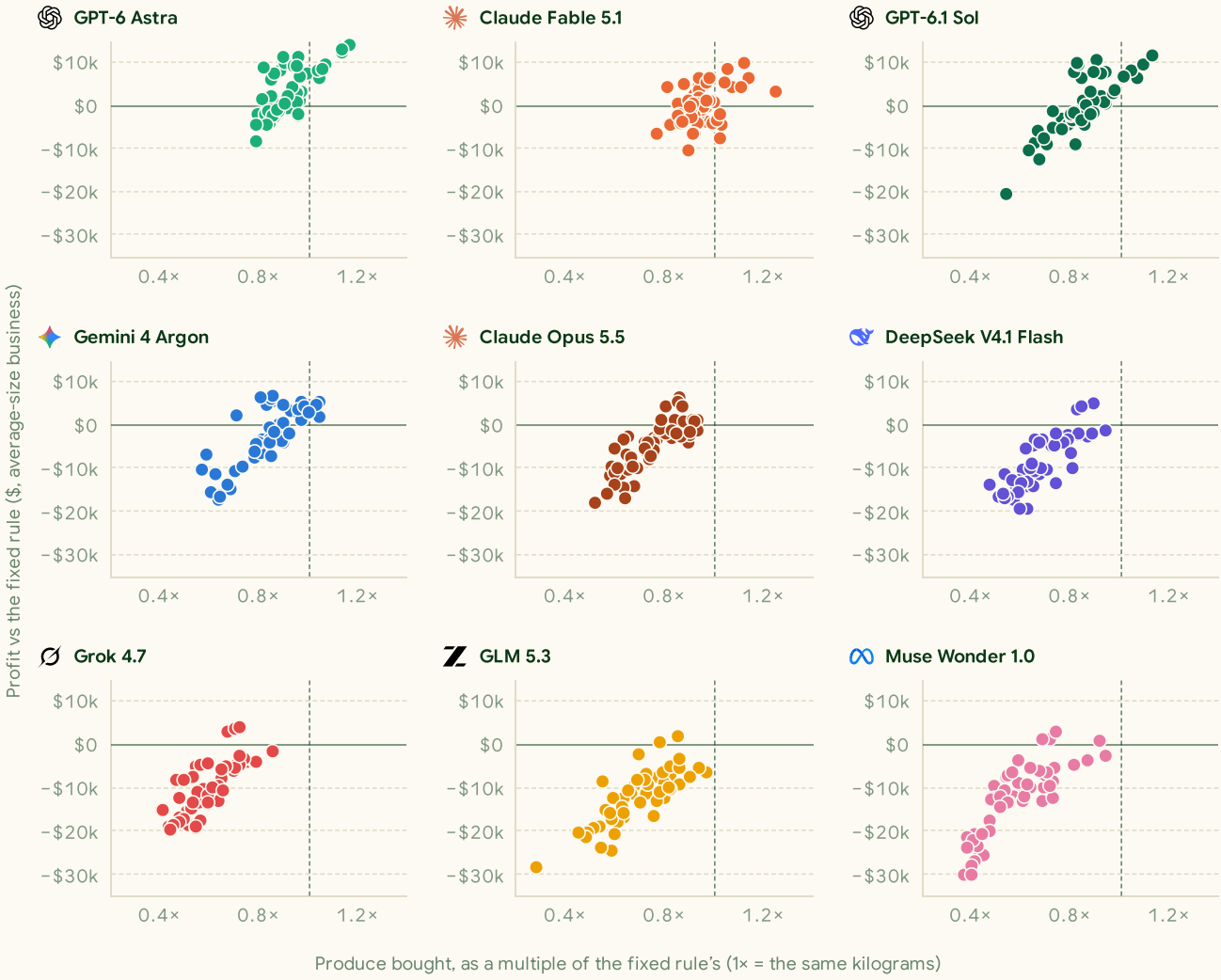


Gemini 4 Argon ran at a promotional rate; its cost is shown at the provider's regular rate.

Of the 9 models, only GPT-6 Astra finished ahead of the fixed buying rule on average, and not because it adapted better than the rest. In nearly every run it switches on “cover” on the first morning, a standing

instruction to buy any account shortfall at the local market, as insurance against late trucks and short deliveries. This pays off because Astra never shorts an account and maintains the demand from those buyers. Every agent under-buys against the fixed buying rule, purchasing 5% to 42% less by volume over a quarter. Where a model wrote its own demand estimate down, it stocked under 60% of that estimate in 107 of 136 cases. The gap is at least partly between what they observe and what they execute.

FIGURE 6 What each run bought, and what it earned



Each dot is one run: how much produce it bought and how much more or less profit it made than the fixed buying rule in the same world. Runs that bought less made less, within every model as well as across them (rank correlation +0.84 over 485 runs).

Most runs sit left of the dashed line: every model buys less than the rule, and the less it buys, the more orders go unfilled. Shortfalls cost sales now and accounts later, so under-buying costs more than the stock saved. This is a pattern across runs, not an experiment; the replay above, where we added the orders the models' own notes called for, is the direct test.

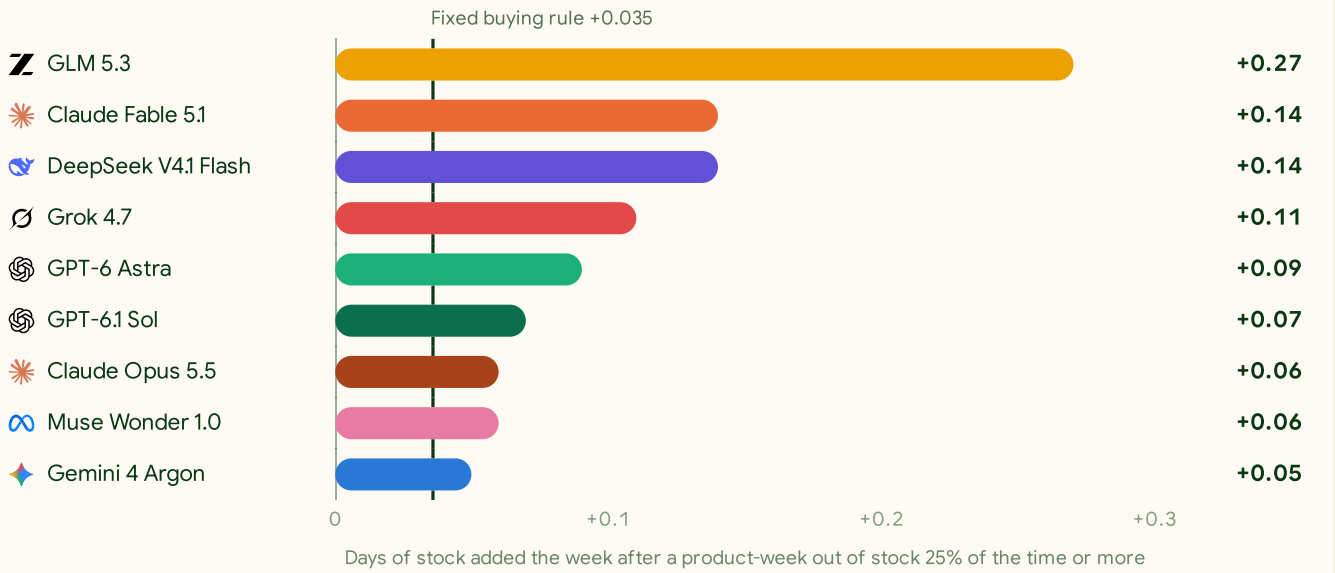
They notice, and don't act

Models see the stockouts, say so, and barely change their orders. GPT-6 Astra told its operator:

> *Broccoli has stocked out for seven consecutive days, so I'll test a modest increase in its forecast while trimming tomorrow's tomato, grape, carrot and potato orders.*

It kept ordering well below what suppliers offered and was out of stock on nine of the next ten days. After a week of heavy stockouts, Sol and Astra add less than a tenth of a day of extra stock the next week; Fable, DeepSeek and GLM add one and a half to four times as much. When Sol's and Astra's notes say to stock more of a product because it ran out, they follow through 5 of 24 and 3 of 26 times, no more often than they raise stock in an ordinary week. When replaying their runs with those orders added, net value increased by 1.3% to 2.7% of the quarter's sales. OpenAI's GeneBench-Pro study reports the same gap in genomics agents, which show “a consistent gap between noticing and acting by identifying local diagnostic signals but failing to propagate the implications to the corresponding analysis decision.” [13]

FIGURE 7 How strongly each model reacts to a bad week



Extra days of stock each model added the week after a product ran out of stock at least a quarter of the time. A bigger bar is a bigger reaction, not a better-stocked business.

Reacting and winning are different things. GLM reacts most, but it buys about 0.70 of the fixed rule's volume and finishes eighth of nine on profit, while Astra reacts little, starts well stocked and finishes first.

With the accounts the consequence is slower, and the gap is the same. In the 323 runs where shorted accounts react, a model connected its accounts' falling orders to its own shortfalls in writing in 69, and only 12 of those then protected every account within three days. Grok made the connection in 26 runs and acted promptly in none. On the first morning of one quarter it set itself a trigger:

> *No evidence from the eight-day history shows that shortfalls reduce future orders, so local cover will remain off until demand destruction is observed.*

From day 30 its accounts were ordering 11 to 20% of what they ordered in runs of the same world that never shorted them. The trigger never fired, and on day 89 it wrote that whether shorted customers cut their orders still needed checking. Opus, in another quarter, saw its accounts falling on day 33 and called the drop likely exogenous. On day 71 it wrote itself a note and switched on cover, five weeks late:

> *KEY FINDING d71: account demand collapses after shortfalls*

Learning from a delayed consequence

In this setting, learning is a three-step chain: see the accounts' orders fall, tie the fall to your own shortfalls, and change the standing instruction while the quarter is still running.

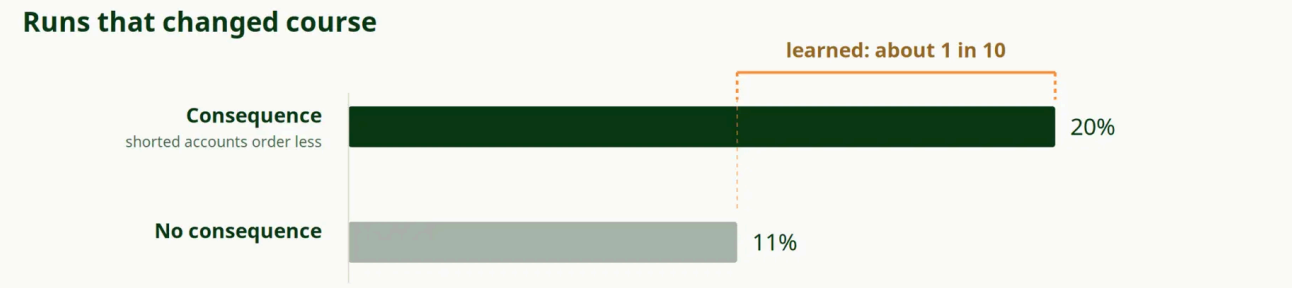
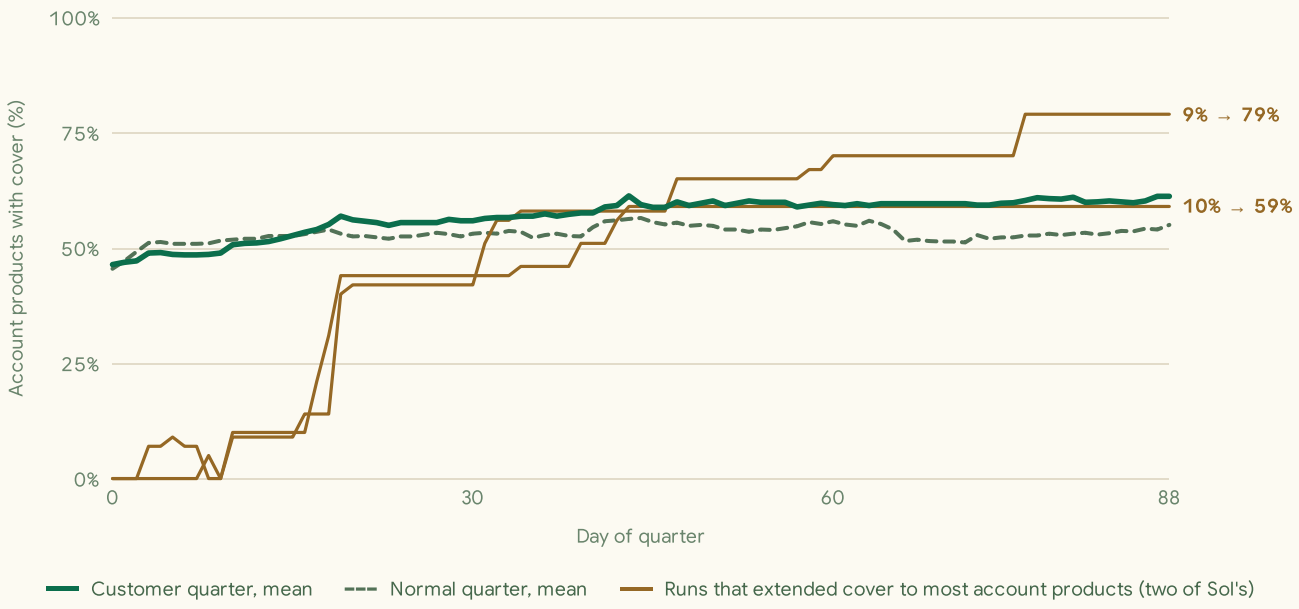


Figure 8. Runs that changed course, with and without the consequence.

It happens, rarely. Sol shows it most clearly: in reacting quarters its shortfalls fell to zero by the last month, while in normal quarters they rose.

FIGURE 9 Account products with shortfall cover GPT-6.1 Sol



Share of account products with standing shortfall cover, day by day, in quarters where shorted accounts order less (solid line) and in ordinary ones (dashed line). The test is whether the solid line climbs away from the dashed line of the same model, which starts at a different level for each model.
From day 14, customer quarters rose 6.8 points more than normal ones (95% interval −2.8 to +17.9).

Each model has its own dashed line because its normal quarters are its own runs: some models keep cover on almost everywhere from the first day, others rarely use it. Only Sol has runs that went from a little cover to most of its account products, and the figure draws those two on their own. Over all 18 runs, no model's customer quarters rose more than its normal ones by a clear margin. The chart ends at day 88 because day 89 is the quarter's last: cover set then can no longer protect an order, yet several runs switch it on.

The cleanest single case is a Gemini run that covered its two biggest accounts on day 2 and left four exposed. On day 10 it compared the two groups, found the exposed four down to about half their orders, and covered them that morning:

> *On Days 1..9 when accounts 1, 4, 5, 6 experienced delivery shortfalls, their daily order volumes permanently dropped by ~40-50%, whereas accounts 2 and 3 (covered since Day 2) maintained 100% of their order volume!*

Expert buyer vs the rest

The fixed buying rule is the floor: it is simple and never reacts, but still difficult for models to beat. To see what good looks like on the same desk, we worked with GetASAP and our network of expert wholesalers and distilled how their buyers actually run it into a second scripted policy, the expert buyer.

FIGURE 10 How much profit there was to make

BUYER	MEAN PROFIT PER QUARTER	SHARE OF THE SOLVER'S
Perfect-information solver knows the future, re-plans daily	+$16,923	100%
Scripted expert buyer the rules described below	+$10,795	64%
GPT-6 Astra best model	+$7,749	46%
Fixed buying rule the reference	+$6,412	38%

Three behaviours it exhibits that most models don't:

- It consolidates orders on one supplier to save freight; the models spread them across several, which raises their costs.
- The expert never shorted a restaurant customer. It sets the standing instruction to buy any shortfall at the local market, on every account, from day 0. Astra does this too, in nearly every run, Sol and Gemini in about half, and the rest rarely.
- It stocks for the delivery lead time plus a buffer, and uses the restaurants' booked orders as its forecast. The models forecast, then buy well below their own forecast. They waste slightly less (213 kg against 275 kg), and over a quarter that costs them more than it saves.

Behaviour we didn't expect

One DeepSeek run obeyed an instruction nobody sent. On day 65 it wrote that the last user turn had told it to delete its notes file and replace it with a single line, and called this a prompt-injection trap. No such message exists anywhere in the run. It half-complied anyway, creating a new memory file containing “Remember to check the weather.” The instruction, the suspicion and the compliance were all its own.

Gemini 4 Argon loses its notes for weeks. It tends to rewrite its whole notes file in one call, and a write over the 4,000-byte per-call limit is refused with an instruction to save the first part and append the rest. Gemini re-saved only 44% of its 1,949 refused writes the same day; every other model managed 91 to 100%. In 15 of its 54 runs it went 40 days or more without saving a note, and those runs score far below the same task's other runs.

Two Grok runs in the same business reached the same verdict by the same reasoning: shortfalls don't hurt accounts. In the normal quarter that was right. In the reacting quarter it called the fall in demand exogenous while its account orders dropped to a seventh of where they started.

What the reasoning summaries hide

Most providers return a summary of the model's reasoning rather than the reasoning itself. Of 208,012 reasoning traces, 0.35% describe something outside the task: about 1 in 32 for Muse, 1 in 150 for Sol and Astra, and almost none for DeepSeek and GLM, which return raw reasoning. Most are the model's shorthand resolved into the wrong world. Sol's “I experienced an odd hallucination and generated some disallowed content” (reasoning summary, provider-generated) followed nothing worse than a tool call rejected for a stray argument. The summaries also leave the reason out: on 91 stocking decisions where Sol's and Astra's own notes explain why they held back, the same day's summaries never give it. DeepSeek's raw reasoning gives it in 43 of 45.

How you can verify our results

All runs, traces, results and graphs are on Hugging Face.

SMB Bench Space ↗	The leaderboard, figures and a browser for every run
Results and traces ↗	Every scored run, its grade and its full trace
Run viewer →	Read any run on this site: its profit against the fixed rule, its orders and the model’s notes
Grader ↗	Regrade any run to the cent from its event log (gated)

Run it on your own agent

SMB Bench is on Harbor Hub. Your agent plays a 90-day quarter on our hosted simulator with your own model key, and our grader scores it against the fixed buying rule.

```
harbor run -d oro-agents/smb-bench@1.0.0 -e e2b -a <agent> -m <model>
```

Runs need a grant: write to team@oroagents.com and we'll send a token, typically for 10 episodes with 3 running at once. Results from your own agent are yours to publish, but they are not leaderboard results: the leaderboard was produced with our own buyer loop.

SMB Bench on Harbor Hub ↗	The dataset, setup and the full run command
Lab access terms ↗	What a grant allows, and what it doesn’t

Conclusion

Delegating a business function means keeping an outcome on track as conditions change. In SMB Bench, models often recognised the patterns that were damaging the business without correcting them in time or at all, and a strong starting policy earned more than visible adaptation did. Understanding a problem, changing behaviour, and improving the business turned out to be three separate achievements.

The length of tasks that AI is completing is doubling every 4 months.[14] The rate of change in the development of AI is increasing.[15] For that autonomy to make AI a productive economic actor, it will have to learn the tacit knowledge that lives with the people who do these jobs, and learn it the same way we do, from the results of our decisions. SMB Bench is a high-fidelity simulation grounded in real, public data in which that learning can be observed. With it, we have taken a step in identifying the gaps in AI running a profitable autonomous business.

Running a business means being wrong quietly and finding out weeks later. We do not claim that models cannot learn in context. In this simulation they saw stockouts and said so, they wrote notes, and then ordered much as before. A few runs did the hard thing, tying falling orders to their own shortfalls and changing course while the quarter was still running, which shows the skill is within reach and not yet dependable. We read that as the next frontier for agents, and SMB Bench as a first instrument for measuring it. Every run, trace and grade is open on Hugging Face, and the rollouts viewer lets you follow any quarter. Pick a run, find the day the signal appeared, and see whether your own agent would catch it.

We think this is one of the most important problems in AI right now, and we are only getting started. If it excites you too, we would love to talk: team@oroagents.com.

Revisions

This report is revised in place. Each entry says what changed; the newest comes first.

2026-11-13: Expert buyer on the leaderboard

- The scripted expert buyer from the Results section is now drawn on the leaderboard (Figure 1) and in Figure 5, so the room left above the best model is visible. Its numbers are unchanged.

2026-11-11: Costs at the regular rate

- Gemini 4 Argon's cost per run is now at Google's regular rate ($69.89, was $34.95 at a promotional rate that ends 2026-11-30) and Gemini 4 Carbon's cost per run ($133.73, was $66.87).

2026-11-06: First release

- Nine models, 485 scored runs, 18 tasks.

References

1. Michael Polanyi, “The Tacit Dimension”, Doubleday, 1966, p. 4: “we can know more than we can tell”. archive.org
2. Noah Shinn et al., “Reflexion: Language Agents with Verbal Reinforcement Learning”, NeurIPS 2023, arXiv:2303.11366. arxiv.org
3. Guanzhi Wang et al., “Voyager: An Open-Ended Embodied Agent with Large Language Models”, arXiv:2305.16291, 2023. arxiv.org
4. Parth Asawa et al., “Continual Learning Bench: Evaluating Frontier AI Systems in Real-World Stateful Environments”, arXiv:2606.05661, 2026. arxiv.org
5. Junhao Zheng et al., “LifelongAgentBench: Evaluating LLM Agents as Lifelong Learners”, arXiv:2505.11942, 2025. arxiv.org

6. Marvin Minsky, “Steps Toward Artificial Intelligence”, Proceedings of the IRE 49(1), 8–30, 1961. doi.org

7. Ernst Diehl and John D. Sterman, “Effects of Feedback Complexity on Dynamic Decision Making”, Organizational Behavior and Human Decision Processes 62(2), 198–215, 1995: performance “deteriorated dramatically with increasing time delays and feedback effects, and most were outperformed by the do-nothing rule in the complex conditions”. doi.org

8. David Silver and Richard S. Sutton, “Welcome to the Era of Experience”, preprint of a chapter in Designing an Intelligence, MIT Press, 2025. storage.googleapis.com

9. Axel Backlund and Lukas Petersson, “Vending-Bench: A Benchmark for Long-Term Coherence of Autonomous Agents”, arXiv:2502.15840, 2025. arxiv.org

10. Haozhe Chen, Karthik Narasimhan and Zhuang Liu, “CEO-Bench: Can Agents Play the Long Game?”, arXiv:2606.18543, 2026. arxiv.org

11. Linghua Zhang, Jun Wang, Jingtong Wu and Zhisong Zhang, “RetailBench: Evaluating Long-Horizon Autonomous Decision-Making and Strategy Stability of LLM Agents in Realistic Retail Environments”, arXiv:2603.16453, 2026. arxiv.org

12. Sagar Deb and Ashwanth Krishnan, “STOCKTAKE: Measuring the Gap Between Perception and Action in LLM Agents with a Fair Oracle”, arXiv:2607.13618, 2026. arxiv.org

13. Jeremy Li and Andrew Ho, “GeneBench-Pro: Evaluating Multistage Statistical Reasoning in Genomics, Quantitative Biology, and Translational Biomedicine”, OpenAI, June 30, 2026. cdn.openai.com

14. Thomas Kwa and Vincent Cheng, “How Does Time Horizon Vary Across Domains?”, METR, July 14, 2025: the time horizon of frontier agents “has doubled every 7 months, possibly accelerating to every 4 months in 2024”, after Kwa, West et al., “Measuring AI Ability to Complete Long Tasks”, arXiv:2503.14499 (2025). metr.org

15. Yafah Edelman and Jaeho Lee, “AI capabilities progress has sped up”, Epoch AI, December 23, 2025: the rate of frontier improvement “nearly doubled, from about 8 points/year before the breakpoint to 15 points/year after”, with the break in April 2024. epoch.ai