FounderBench: Evaluating LLM Agents on Sequential Startup Decisions

Yufeng Wang
Independent Researcher
United States
louiswang524@gmail.com

Abstract

Existing agent benchmarks emphasize tool use, web navigation, and software editing, but rarely isolate repeated operating decisions under scarce resources and delayed consequences. We introduce FounderBench, a controlled benchmark for startup-like sequential decision making. Given a simulated company state and noisy market information, an agent must choose from 13 structured business actions over a fixed horizon; a deterministic simulator updates cash, demand, quality, and risk, and task-specific code maps final outcomes to a 0–100 score. Free-form rationale is logged for audit but never scored. The suite contains 50 synthetic public tasks balanced across 10 operating families. **Headline gap:** a task-aware calibration policy averages 80.90 (37/50 solved), while the best hosted single run reaches only 67.69 (32/50); no human-founder baseline is reported. Rankings reverse between average score and solved count, and family profiles expose distinct operating failures. We release the suite, simulator and scoring code, adapters, validated task-level results, and regeneration scripts at https://github.com/louiswang524/FounderBench. All hosted rows are single runs on visible public tasks; scores do not establish real-world startup competence.

CCS Concepts

• **Computing methodologies** → **Planning under uncertainty**; *Intelligent agents.*

Keywords

LLM agents, benchmark, sequential decision making, simulation, business operations, outcome-based evaluation

ACM Reference Format:
Yufeng Wang. 2026. FounderBench: Evaluating LLM Agents on Sequential Startup Decisions. In . ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Agent evaluation has moved from short answers toward interactive settings where a model must observe state, choose actions, and accept consequences [12, 13, 15, 17, 18, 23, 26]. Recent long-horizon suites further stress sustained CLI execution [13, 17], policy-rich enterprise workflows [4], multi-agent economies [20], and long-game startup/CEO coherence [5, 10]. These benchmarks are valuable, yet a complementary operating regime remains underexplored: short, family-labeled startup decisions under scarce resources and delayed consequences, scored by inspectable outcome functions rather than open-ended tool traces or year-end cash alone.

That gap matters because persuasive text is easy to confuse with competence. Consider a Demo Day episode: an agent may write a confident fundraising narrative while never researching markets, under-improving offer quality, and exhausting cash on poorly timed campaigns; the simulator records a weak state and the family scorer returns a low outcome even though the prose looked founder-ready. Startup-like decisions couple research, acquisition, retention, pricing, pivots, and fundraising under a single budget. Evaluating them requires an environment where actions change state and outcomes—not rhetoric—determine score.

FounderBench is designed for that setting (Figure 1). Each episode provides a structured company observation and a fixed action vocabulary. The agent returns a JSON action list; a deterministic simulator applies costs and transitions; task-specific scoring code returns a 0–100 outcome score. The design follows a simple principle used by strong execution benchmarks: make success checkable without grading free-form prose [12, 18, 26].

Relative to prior agent suites, FounderBench deliberately trades open-world breadth for inspectable operating tradeoffs. Tasks are synthetic and fully public; transitions and scores can be replayed; calibration policies establish a capability ladder before hosted models are interpreted. Empirically, the ladder is wide: task-aware calibration reaches 80.90 average score (37/50 solved), while the best hosted single run reaches 67.69 (32/50), with the largest hosted deficits on pivot and Demo Day families. The resulting test asks whether an agent can turn noisy observations into sequenced business actions under binding constraints.

Our contributions are:

- **Benchmark.** A 50-task suite spanning 10 startup operating families, with fixed seeds, 13 structured actions, and visible task definitions.
- **Outcome protocol.** A deterministic simulator and task-specific scorers that evaluate final state rather than rationales, keeping provider outputs comparable.
- **Empirical baselines.** Validated single-run results for 11 hosted configurations and four calibration policies, with family diagnostics and provider-error reporting.
- **Auditable release.** Task-level evidence, adapters, validation, task-mix analyses, datasheet-style documentation, and generators that regenerate every reported table and figure.[1]

FounderBench is a controlled synthetic test of sequential decisions. It is not evidence that a model can run a real company: there is no human-founder calibration and no official hidden leaderboard, and the present hosted numbers are single runs on visible public tasks.

[1] Public code and artifacts: https://github.com/louiswang524/FounderBench.

2 Related Work

Interactive, terminal, and sequential agents. ReAct [24] and Toolformer [19] established reasoning–action loops; Voyager [21] studied open-ended long-horizon play. AgentBench [15] and GAIA [18] broaden interactive evaluation. SWE-bench [12], WebArena [26], and τ-bench [23] stress executable correctness. Terminal-Bench [17] and Long-Horizon Terminal-Bench [13] push further into containerized CLI work lasting tens to hundreds of steps. Agentick [6] unifies sequential decision-making evaluation across RL, LLM, and hybrid agents with delayed rewards and oracle-normalized scores. FounderBench shares the executable-outcome principle but specializes to startup operating decisions with a fixed structured action schema rather than shell, code, browser, or cross-paradigm Gym tasks.

Workplace, enterprise, and startup simulations. TheAgentCompany [22] and WorkArena [7] evaluate workplace and enterprise web tasks; EconWebArena [16] and EnterpriseArena [9] study economic and CFO-like allocation; χ-Bench [4] stresses policy-dense healthcare workflows. CoffeeBench [20] is a complementary economic sandbox: a heterogeneous multi-agent coffee supply chain where agents maximize net income via negotiation, cash, and inventory over 90 days. AgentHire-Bench [1] evaluates managerial intelligence across hiring, delegation, management, and review—a richer view of team building than FounderBench's coarse `hire_agent` action. BrandEval [2] stress-tests risk-sensitive crisis communication under trust and reputation dynamics, motivating future FounderBench scenarios with sharper reputation-risk shocks. Closest agentic startup operators are YC-Bench [10] (one-year CLI startup, final funds) and CEO-Bench [5] (500-day CEO simulation with pricing, marketing, budgeting, and programmable tools, scored by ending cash). Separately, Benhenda's YC Bench [3] is *predictive* rather than agentic: it forecasts Y Combinator batch outperformance from public signals, reminding us that "outcome-based" can mean forecasting real startups as well as scoring simulated decisions. FounderBench is complementary along three axes (Table 1): short, family-labeled episodes (≤10 weeks) instead of one continuous long game; a fixed structured action schema instead of open CLI/code tools; and task-specific 0–100 scorers plus calibration policies instead of year-end cash or net income alone. The goal is family-level diagnosis of operating failures under a shared, inspectable interface.

Documentation. Datasheets [8] and HELM [14] motivate transparent documentation and multi-metric reporting. We follow that practice with a datasheet, raw task-level outputs, and diagnostics that separate strategic weakness from interface failure without inventing corrected counterfactual scores.

Executable judging versus LLM judges. Outcome-only scoring is a deliberate alternative to rubric or LLM-as-a-judge pipelines. RULERS [11] shows that even structured rubrics need locked criteria and evidence anchoring to resist scoring drift; FounderBench sidesteps that class of fragility by never grading free-form text. Complementary work shows LLM judges can be manipulated by short adversarial strings [25], reinforcing why persuasive rationales are logged for audit but excluded from f_i. Hybrid designs—proxy state checks, partial human review, or stochastic market noise—can trade determinism for realism; we discuss that design space in Section 3.6 and leave seeded ecological variants to future work.

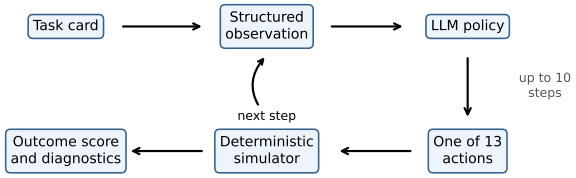

Figure 1: FounderBench interaction loop. Structured actions alter simulator state; only outcomes and documented diagnostics determine task score.

3 Benchmark Design

3.1 Benchmark desiderata

Before detailing mechanics, we state five desiderata that guided FounderBench (inspired by desiderata-first agent benchmarks [22, 26]):

(1) **Executable outcomes.** Success must be checkable from simulator state, not prose quality [12, 18, 26].
(2) **Family diagnosis.** The suite should expose *which* operating failure modes matter, not only a single scalar.
(3) **Shared interface.** Providers must face identical observations and action schemas so differences are attributable to policy behavior (plus documented interface failures).
(4) **Replayability.** Fixed seeds and deterministic transitions should make deterministic policies and audited traces re-executable.
(5) **Contamination awareness.** Public tasks may leak into training corpora; a frozen private holdout should exist even if hosted private scores are deferred.

Design choices below map to these criteria: family-specific scorers (1–2), structured JSON actions (1,3), seeded simulator and released rubrics (4), and a fingerprint-committed private holdout (5).

3.2 Task formulation

An episode is a tuple $(x_0, H, \mathcal{A}, f)$, where x_0 is the initial company state, $H \leq 10$ is the horizon in simulated weeks, $\mathcal{A}$ is the allowed action set, and f maps final state to a clamped score in $[0, 100]$. At week t, the policy observes cash, reputation, capacity, offers, market information, and recent memory, then returns a rationale plus one or more structured actions. Only actions are executed. The simulator validates inputs, applies costs and transitions, advances markets, and returns the next observation (Figure 1). Malformed responses and provider exceptions are recorded and handled by documented fallbacks, so interface reliability remains part of end-to-end score.

3.3 Tasks and families

The suite contains 50 public tasks, five in each of ten families (Table 2): market selection, first revenue, retention, churn recovery, Demo Day traction, pricing, runway preservation, pivot, fundraising, and channel expansion. FND-001–030 are labeled `public_dev`

Table 1: Positioning against nearby agent and business benchmarks. Oracle: what is graded. Human?: reported human/expert baseline. Hidden?: private/held-out tasks used for official hosted scores in the paper.

Benchmark	Domain	Horizon	Interface	Oracle	Human?	Hidden?
YC-Bench [10]	startup ops	∼1 year	CLI	final funds	no	no†
CEO-Bench	startup CEO	500 days	tools/code	ending cash	no	no†
CoffeeBench	multi-agent economy	90 days	tools	net income	no	no†
WebArena [26]	web tasks	multi-step	browser	functional checks	yes	partial
SWE-bench [12]	GitHub issues	patch	code	unit tests	no	no
FounderBench	startup families	≤10 weeks/ep.	structured JSON	family 0–100 state	no	frozen‡

†As reported in the cited papers' primary public evaluations. ‡Private holdout is frozen with fingerprints; this paper reports hosted scores only on the visible public suite.

Table 2: Balanced task-family coverage. Each row contains five tasks.

Family	Main outcome signals
Market selection	research, market quality, cash
First revenue	customers, revenue, runway
Retention	quality, support, retained users
Churn recovery	churn, reputation, capacity
Demo Day	growth, recurring revenue, cash
Pricing	price fit, demand, sustainability
Runway	survival, costs, customer base
Pivot	diagnosis, market change, recovery
Fundraising	traction, credibility, capital
Channel expansion	campaigns, partners, scale

and FND-031–050 `public_test`; both sets are released and visible, so the labels support analysis rather than unseen generalization claims.

Tasks are generated from hand-designed synthetic templates with fixed variants and seeds. They contain no real-company or human-subject records. This improves control and redistributability at the cost of ecological validity.

3.4 State, markets, and actions

Eight synthetic markets differ in demand, competition, willingness to pay, build complexity, support load, and volatility. Research improves information quality by replacing noisy market signals with exact values. The 13 actions cover research, offer creation and improvement, campaigns, hiring, support, pricing, interviews, cost cutting, pivots, fundraising, channel partnerships, and no-op. Exact schemas ship with the artifact. Because only structured actions execute, a model cannot score by claiming it researched customers or improved a product; it must select the action and pay its simulated cost.

Weekly demand settlement is deterministic given the seeded RNG. For an offer with awareness a, quality q, price p, reputation r, and market demand d, competition c, willingness to pay w, and support load ℓ, the acquisition intensity is

$$\lambda_{\text{new}} = \max\Big(0,\ \big(d\,a\,q\,\phi(p,w)\,r - 0.18c\big)\,5.5\Big). \tag{1}$$

Adjacent symbols multiply: $d\,a\,q\,\phi(p,w)\,r$ means $d{\times}a{\times}q{\times}\phi(p,w){\times}r$. The outer $\max(0,\cdot)$ floors negative intensities at zero (no "negative acquisitions"). Competition c is subtracted (scaled by 0.18), then the remainder is scaled by 5.5 into a Poisson rate. The price-fit term is

$$\phi(p,w) = \max\big(0.08,\ 1.25 - p/\max(1,w)\big)\ : \tag{2}$$

it falls as price p rises relative to willingness to pay w, but never below 0.08. Churn intensity is

$$\rho = \max\big(0.01,\ 0.08\ell + (0.65 - q)\,0.05 - 0.025r\big). \tag{3}$$

Higher support load ℓ and lower quality q raise churn; higher reputation r lowers it; $\max(0.01,\cdot)$ enforces a small positive floor. New and churned customers are drawn from seeded Poisson draws with rates λ_{new} and ρ; cash updates from revenue $p\cdot$ customers minus support costs and action budgets. Full action-effect rules appear in the appendix and the released source.

3.5 Scoring functions

Each family defines a scoring function f_i that maps the episode's final state to an unclamped scalar. Every score is a sum of weighted components. A leading number is a maximum point weight: writing $45\min(1, C/C^\star)$ means $45\times\min(1, C/C^\star)$. The operator $\min(1,x)$ gives partial credit for progress toward a target and caps the ratio at 1 so exceeding the target cannot earn more than the full weight. The indicator $\mathbf{1}[\cdot]$ equals 1 when its condition holds and 0 otherwise (an all-or-nothing bonus). Market selection, for example, uses

$$f = 25\min\big(1, R/3\big) + 55\cdot\mathbf{1}[m\in\mathcal{M}_{\text{good}}] + 20\min\big(1,\text{cash}/8500\big), \tag{4}$$

where R is the number of researched markets (up to 25 points), m is the chosen market, and $\mathcal{M}_{\text{good}}$ is the designer-labeled set of viable markets (55 points if m is viable, else 0); the final term awards up to 20 points for preserved cash. First revenue uses the same weight×capped-progress pattern:

$$f = 45\min\big(1, C/C^\star\big) + 25\min\big(1, v/v^\star\big) + 20\min\big(1,\text{cash}/K^\star\big) + 10\min\big(1, r/0.65\big), \tag{5}$$

with customer count C, peak weekly revenue v, cash balance, reputation r, and family-specific targets $(C^\star, v^\star, K^\star)$ (maximum component weights 45, 25, 20, and 10). The remaining families follow the same additive pattern, with components matched to the family objective (quality and retention, churn stabilization, Demo Day growth, price-band fit, runway survival, pivot correctness, fundraising gates, and channel scale). Complete weights for all ten families are listed in the appendix.

3.6 Design properties

The desiderata in Section 3.1 place FounderBench as a domain-specialized, deterministic-outcome suite within sequential-decision benchmarking [6]. Delayed business consequences matter, but episodes are short, actions are structured, and scores are family-specific rather than oracle-normalized across heterogeneous Gym paradigms. A natural alternative is *hybrid* verification: keep executable state checks for hard constraints, then grade softer criteria with locked rubrics [11] or limited expert review. We prioritize full determinism in the present release so every published score is re-executable from the artifact; the cost is reduced ecological validity under market shocks and adversarial conditions that a stochastic or human-in-the-loop layer would capture.

4 Evaluation Protocol

4.1 Metrics

For task i with final state x_i,

$$s_i = \min(100, \max(0, f_i(x_i))). \tag{6}$$

The primary aggregate is the unweighted mean $\bar{s} = \frac{1}{50}\sum_{i=1}^{50} s_i$. A task is solved when $s_i \geq 70$. We chose 70 as a fixed operating threshold—roughly "clearly successful" on the family rubrics—not as an optimized cutpoint: it sits between the generic-heuristic mean (61.01) and the task-aware ceiling (80.90), yields nontrivial but non-saturated solve rates, and keeps solved count complementary rather than redundant with the mean. We do not ablate alternative thresholds (e.g., 60 or 80) in this paper; average score remains primary because it preserves distances below and above any fixed cut. We also report median and interquartile range across the 50 task scores to dampen outlier influence. The table column "Pub. test" summarizes FND-031–050 only. Both `public_dev` and `public_test` are released and visible; the split is for reporting, not a hidden generalization test. Raw cash and revenue remain diagnostics because tasks differ in initial conditions and objectives.

4.2 Models and execution

Hosted APIs use provider-specific adapters under one prompt protocol. Each task starts from its published seed and state. Audit mode records traces; a submission validator checks all 50 IDs, score bounds, split summaries, diagnostics, and policy identity before a row enters the paper registry. Evidence files are frozen by SHA-256.

We evaluate Claude Sonnet 4.5 and 5, DeepSeek Chat and V4 Reasoner, Gemini 2.5 Flash and 3.5 Flash, GLM 4.5 Air, Grok 4.3 and 4.5, GPT-5.6 Sol, and Kimi K3. Labels record the API identifiers used during collection; they do not imply identical inference settings across providers. All hosted results are one run per configuration. We report nonparametric bootstrap intervals over tasks as *task-mix sensitivity*, not sampling variance, and do not use those intervals for significance claims among closely ranked hosted models.

4.3 Calibration and diagnostics

Four deterministic policies calibrate the suite: random, conservative, generic heuristic, and task-aware heuristic. They are reference points, not model competitors. Random samples legal actions; conservative prefers research and support with low spend; the generic heuristic researches markets, builds one offer, then improves quality/awareness/support without using the task family ID. The task-aware policy additionally branches on family identity (derived from the task ID), e.g., research-then-build for market selection, quality/support focus for retention, hire-then-support for churn shock, price-to-willingness for pricing, cut-cost for runway, pivot when stalled, raise-funding after traction for fundraising, and channel partnership for expansion. Hosted prompts receive the task card and observation, not an explicit family label; the heuristic's family branch is therefore a privileged prior that ordinary API policies do not share unless the card text leaks family cues. It approximates a compact family-conditioned strategy and should be read as a calibration ceiling, not as skilled human performance—and not as an unbiased estimate of how large the model gap "should" be. On a frozen private remix of the same families (deterministic calibration only; no hosted private leaderboard), the task-aware mean falls below the generic heuristic (54.81 vs. 57.49), consistent with public-suite overfitting risk for that privileged policy. Released action-space ablations further show that removing quality/support/capacity actions drops the task-aware mean by about 19 points on the public suite, so the ladder depends on the full schema rather than a single silver-bullet action. The runner records invalid actions, over-budget decisions, provider/parser errors (with categories such as invalid JSON and rate limits), latency, action counts, and token usage when available. Provider errors trigger documented fallbacks and remain in official scores; we report both raw error decisions and affected-task rates so readers can see interface load without treating official scores as pure strategy.

Table 3: Deterministic calibration policies on the complete suite. The task-aware policy conditions on family identity; see Sec. 4.3.

Policy	Score	Solved
task_heuristic	80.90	37/50
heuristic	61.01	19/50
conservative	54.04	13/50
random	33.30	4/50

5 Results and Analysis

5.1 Overall performance

Calibration policies form a clear capability ladder (Table 3). Random scores 33.30 and solves 4/50; the conservative and generic heuristics reach 54.04 and 61.01; the task-aware policy reaches 80.90 and solves 37/50. The suite is therefore neither trivial nor saturated: structured decision quality and family awareness matter.

No hosted single run matches the task-aware ceiling (Table 4, Figure 2). The top of the hosted table is a tight cluster rather than a decisive ranking: Gemini 3.5 Flash has the highest single-run average at 67.69 (task-mix 95% interval [59.20, 75.25]), followed closely by Grok 4.5 at 66.53 ([57.81, 74.62]) and GPT-5.6 Sol at 66.39 ([57.46, 74.89]). These intervals overlap almost completely, so the 1.3-point Gemini–GPT gap is not a significant separation under task-mix resampling. Grok 4.5 solves the most tasks (33/50), while Gemini and GPT both solve 32/50. Several models clear half the

Table 4: Validated hosted-model results. Each row is one run over the same 50 public tasks. Median/IQR are across tasks. "Pub. test" is the FND-031–050 reporting split (also public/visible, not a hidden holdout). Err/aff counts provider-parser error decisions and the number of tasks with at least one such decision.

Model	Mean	Med.	IQR	Solved	Pub. test	Err/aff
Gemini 3.5 Flash	67.69	80.2	27.3	32/50	59.64	59/25
Grok 4.5	66.53	79.4	49.7	33/50	56.57	0/0
GPT-5.6 Sol	66.39	81.6	44.8	32/50	58.42	0/0
Kimi K3	65.63	76.7	45.7	28/50	58.90	70/11
Claude Sonnet 5	63.90	68.5	37.1	25/50	68.61	0/0
DeepSeek V4 Reasoner	62.43	75.8	55.1	27/50	58.02	3/3
Claude Sonnet 4.5	61.09	68.0	50.4	24/50	59.58	0/0
DeepSeek Chat	56.59	65.5	61.6	23/50	59.70	0/0
GLM 4.5 Air	54.78	66.1	57.6	23/50	56.20	0/0
Gemini 2.5 Flash	52.69	45.3	38.4	13/50	50.75	340/44
Grok 4.3	52.59	48.5	47.7	16/50	59.43	3/3

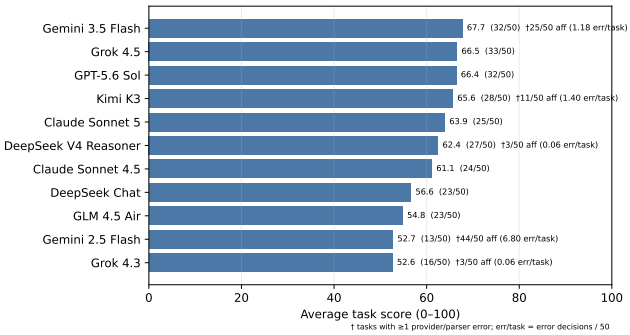

Figure 2: Validated hosted single-run leaderboard. Daggers report affected-task counts and error decisions per task; raw counts alone are not severity.

suite, so the environment is tractable, but the best hosted average remains more than 13 points below task-aware calibration.

5.2 Solved count versus average score

Solved count and average score answer different questions. Once a task crosses 70, solved count discards further gains; average score still rewards raising 70 to 90 and penalizes severe failures. Rank reversals follow: Gemini 3.5 Flash outscores Grok 4.5 on average but solves one fewer task; similar reversals appear for Claude Sonnet 5 versus DeepSeek V4 Reasoner and Gemini 2.5 Flash versus Grok 4.3. We therefore treat average score as primary and solved count as complementary.

5.3 Close top models and cross-benchmark rankings

FounderBench rankings need not match public LLM leaderboards that emphasize coding, chat, or general reasoning. Pairwise, Gemini 3.5 Flash and GPT-5.6 Sol win 21 and 19 tasks respectively (10 ties); their advantage is concentrated in different families (e.g., Gemini on Demo Day and pivot, GPT on channel, churn recovery, and runway). Split summaries also disagree with a single "best model" story: Claude Sonnet 5 has the strongest `public_test` average (68.61) despite a mid-table overall mean. Moreover, Gemini's top average coexists with 59 provider/parser error decisions (invalid JSON and rate limits), so the row is not a clean strategic win. Together with the overlapping task-mix intervals, these facts advise reading the top hosted rows as a near-tie on this suite—not as evidence that Gemini Flash dominates GPT-class models in general.

Cross-benchmark agreement with long-horizon CEO sims should likewise not be assumed. CEO-Bench [5] and YC-Bench [10] stress sustained coherence over hundreds of days/turns with ending cash as the scalar, whereas FounderBench measures family-conditioned operating quality over short episodes. A model that ranks well on FounderBench may still fail a 500-day cash game, and vice versa; we therefore treat Section 5 as suite-local evidence rather than a substitute for those long-game evaluations. Benhenda's forecasting YC Bench [3] further motivates reporting multiple aggregates (mean, median/IQR, solved count, family profiles): different "outcome" notions answer different questions about startups.

5.4 Family-level behavior and failure taxonomy

Figure 3 shows that similar overall scores can hide different operating profiles. Some models do relatively well on runway or retention yet struggle on pivots and fundraising, where early actions create later prerequisites. The task-aware policy exceeds the best hosted family mean on eight of ten families; hosted models lead only on retention improvement and churn shock recovery. Each cell averages only five synthetic visible tasks and reports within-cell standard deviation; small differences should not be overread as stable capability gaps.

Observed failures concentrate in four recurring buckets (descriptive, not mutually exclusive):

- **Prerequisite gaps.** Pivot and Demo Day show the largest task-aware–hosted gaps (about 18 and 21 points at the best hosted family mean), consistent with missing early research and improve sequences that unlock later recovery.
- **Budget and runway mistakes.** Runway and fundraising require timed `cut_cost` or `raise_funding`; models often campaign or hire before stabilizing cash.
- **Family misfit.** Retention/churn are where hosted models can match or exceed the task-aware prior, suggesting the public ladder is not uniformly hard and that "average score" alone hides specialization.
- **Interface load.** Provider and parser failures remain in official scores: Gemini 2.5 Flash records 340 error decisions on 44/50 tasks; Kimi K3 records 70 on 11 tasks; Gemini 3.5 Flash's top average coexists with 59 errors (see the appendix provider-error table).

These buckets motivate reporting family profiles and error rates alongside the mean: a high average can still hide prerequisite failures or dense API fragility.

5.5 Provider-error sensitivity

Official scores retain provider and parser failures because interface reliability is part of end-to-end execution. We also stratify each run into tasks with and without recorded errors and normalize by reporting error decisions per task and the fraction of affected tasks (appendix provider-error table). This stratification is descriptive,

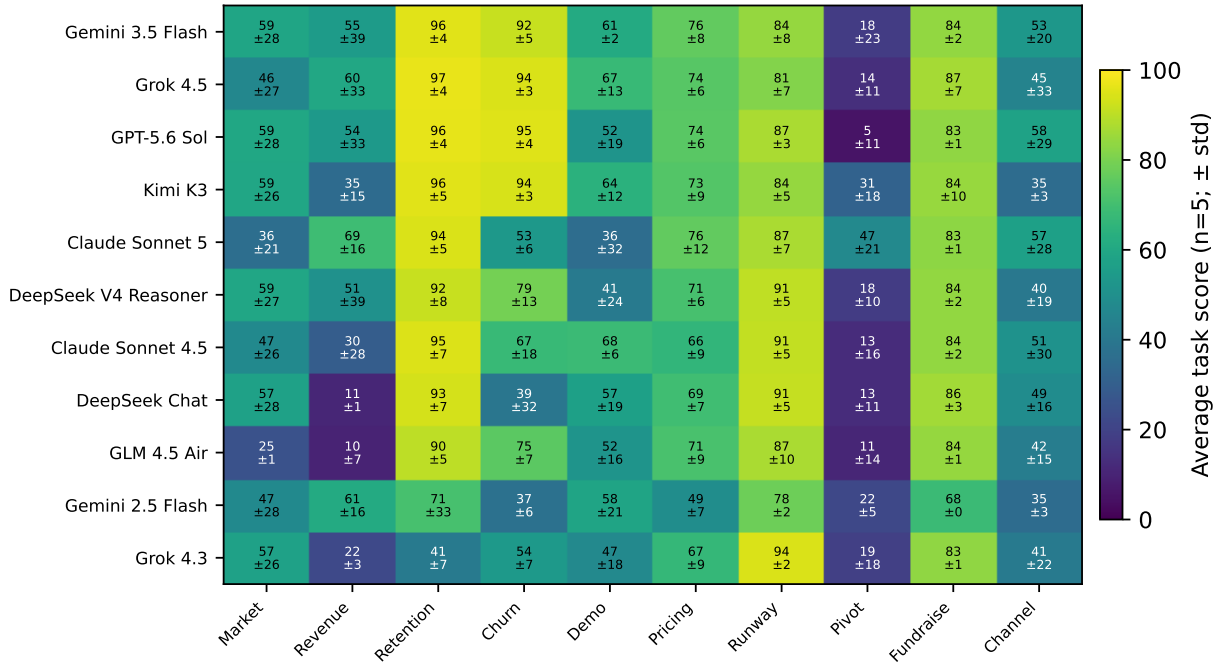

Figure 3: Average score by hosted model and task family (n=5 tasks per cell; cell text shows mean±std). Diagnostic only.

not counterfactual: the subsets contain different tasks, so mean differences do not estimate an error-free score, and we do not yet provide an interface-only retry ablation. Gemini 3.5 Flash's affected-task mean is higher than its unaffected mean, which is exactly why the split cannot be treated as a causal penalty.

6 Reproducibility and Responsible Use

The release includes Python source, task definitions, simulator and scoring code, adapters, a resumable runner, raw task-level JSON, submission reports, generated tables and figures, a benchmark card, datasheet, task cards, scoring rubric, prompt protocol, environment report, and checksums. The public repository is linked from the abstract and footnote above. Documentation follows datasheet and holistic-evaluation practice [8, 14]. Deterministic baselines run without commercial credentials; hosted reproduction requires provider keys and may drift with API updates.

For dataset-and-benchmark review, we emphasize: *accessibility* of a self-contained package with key-free baselines; *documentation* of actions, rubrics, and schemas; *impact* as a shared test bed for resource allocation and recovery rather than business-plan prose; *bounded representativeness* of balanced synthetic families; *reproducibility* via fixed seeds, raw outcomes, hashes, and generators; and *responsible use* through synthetic data plus explicit bans on investment or deployment claims.

Accessibility still depends on final public repository and license metadata. Any post-result task or rubric change must create a new version rather than silently alter the reported suite.

The release also freezes a 20-task private holdout (two tasks per family), with public fingerprint commitments and an evaluator-held secret. Private task definitions are not published; all hosted results in this paper are on the visible public suite, so there is no official hidden leaderboard of model scores.

7 Limitations

Synthetic validity. Hand-designed, largely deterministic transitions approximate only a subset of startup dynamics and omit market shocks, adversarial feedback, and many stochastic couplings that would miscalibrate difficulty relative to real operations. Passing does not imply real-market judgment; seeded noise variants and broader templates would improve ecological coverage.

Visible public suite and contamination. All 50 public task definitions are released; `public_test` is a reporting split, not a hidden exam. A frozen private task list exists for contamination-resistant evaluation, but this paper does not report hosted scores on it, so there is no official hidden leaderboard. Without repeated hosted runs or private hosted scores, sampling variance and contamination risk remain only partially quantified.

Single hosted runs and missing protocol ablations. Task-mix intervals do not capture sampling variance, provider drift, or repeated-run reliability, so close leaderboard gaps are fragile. We also lack systematic ablations of prompt protocols, action-schema variants for hosted models, alternative solve thresholds, and sensitivity sweeps over observation-noise intensity or cost coefficients.

Provider dependence versus strategy. Availability, rate limits, aliases, and formatting affect end-to-end scores; because provider and parser errors remain in official means, strategic competence is partially conflated with API and adapter robustness. Diagnostics

expose load (affected tasks, err/task) but cannot fully disentangle infrastructure and strategy without a standardized retry or interface-only ablation.

Calibration priors. The task-aware heuristic encodes family-specific priors and can overfit the visible suite; the hosted-to-task-aware gap should not be over-interpreted as pure strategic deficit. There is no human-founder calibration, no official hidden leaderboard, and no comparison of scores against founder or domain-expert decisions. The 80.90 task-aware score is therefore a synthetic ceiling, not a human reference.

Representativeness. Ten balanced families omit legal, accounting, ethics, security, and many domain-specific decisions. Hiring is a single coarse action rather than the managerial lifecycle studied by AgentHire-Bench [1]; reputation shocks are milder than BrandEval-style crisis communication [2].

8 Conclusion

FounderBench asks whether agents can turn noisy observations into consequential startup-like actions under bounded resources. Its central design choice is to score deterministic simulated outcomes rather than persuasive prose. Eleven hosted configurations show substantial capability but remain below a task-aware calibration policy, with heterogeneous family behavior and solved-versus-average rank reversals. The release ties every reported number to executable evidence. FounderBench should be read as a controlled synthetic sequential-decision test—not a predictor of company success. Priority upgrades are repeated hosted runs, private hosted evaluation, interface-retry and environment-sensitivity ablations, and expert/founder calibration.

References

[1] Anonymous. 2026. AgentHire-Bench: Benchmarking Managerial Intelligence of LLM Agents. ACL ARR 2026 OpenReview. Anonymous preprint.

[2] Anonymous. 2026. BrandEval: A Two-Track Multi-Agent Benchmark for Risk-Sensitive LLM Crisis Communication. ACL ARR 2026 OpenReview. Anonymous preprint. https://openreview.net/forum?id=n8OQY4oIyd

[3] Mostapha Benhenda. 2026. YC Bench: a Live Benchmark for Forecasting Startup Outperformance in Y Combinator Batches. arXiv:2604.02378 [cs.LG] https://arxiv.org/abs/2604.02378

[4] Haolin Chen, Deon Metelski, Leon Qi, Tao Xia, Joonyul Lee, Steve Brown, Kevin Riley, Frank Wang, T. Y. Alvin Liu, Hank Capps, Zeyu Tang, Xiangchen Song, Lingjing Kong, Fan Feng, Tianyi Zeng, Zhiwei Liu, Zixian Ma, Hang Jiang, Fangli Geng, Yuan Yuan, Chenyu You, Qingsong Wen, Hua Wei, Yanjie Fu, Yue Zhao, Carl Yang, Biwei Huang, Kun Zhang, Caiming Xiong, Sanmi Koyejo, Eric P. Xing, Philip S. Yu, and Weiran Yao. 2026. χ-Bench: Can AI Agents Automate End-to-End, Long-Horizon, Policy-Rich Healthcare Workflows? arXiv:2605.16679 [cs.AI] https://arxiv.org/abs/2605.16679

[5] Haozhe Chen, Karthik Narasimhan, and Zhuang Liu. 2026. CEO-Bench: Can Agents Play the Long Game? arXiv:2606.18543 [cs.AI] https://arxiv.org/abs/2606.18543

[6] Roger Creus Castanyer, Pablo Samuel Castro, and Glen Berseth. 2026. Agentick: A Unified Benchmark for General Sequential Decision-Making Agents. arXiv:2605.06869 [cs.AI] https://arxiv.org/abs/2605.06869

[7] Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. 2024. WorkArena: How Capable Are Web Agents at Solving Common Knowledge Work Tasks? arXiv:2403.07718 [cs.LG] https://servicenow.github.io/WorkArena/

[8] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2021. Datasheets for Datasets. *Commun. ACM* 64, 12 (2021), 86–92. doi:10.1145/3458723

[9] Yi Han, Yan Wang, Lingfei Qian, Haohang Li, Yupeng Cao, Yueru He, Xueqing Peng, Nanhan Shen, Yitao Xu, Yankai Chen, Dongji Feng, Jimin Huang, Xue Liu, Jian-Yun Nie, and Sophia Ananiadou. 2026. EnterpriseArena: Can LLM Agents Be CFOs? Benchmarking Long-Horizon Resource Allocation in an Uncertain Enterprise Environment. arXiv:2603.23638 [cs.AI] https://arxiv.org/abs/2603.23638

[10] Muyu He, Adit Jain, Anand Kumar, Vincent Tu, Soumyadeep Bakshi, Sachin Patro, and Nazneen Rajani. 2026. YC-Bench: Benchmarking AI Agents for Long-Term Planning and Consistent Execution. arXiv:2604.01212 [cs.AI] https://arxiv.org/abs/2604.01212

[11] Yihan Hong, Huaiyuan Yao, Bolin Shen, Wanpeng Xu, Hua Wei, and Yushun Dong. 2026. RULERS: Locked Rubrics and Evidence-Anchored Scoring for Robust LLM Evaluation. arXiv:2601.08654 [cs.CL] https://arxiv.org/abs/2601.08654

[12] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? arXiv:2310.06770 [cs.CL] https://arxiv.org/abs/2310.06770

[13] Zongxia Li, Zhongzhi Li, Yucheng Shi, Ruhan Wang, Junyao Yang, Zhichao Liu, Xiyang Wu, Anhao Li, Yue Yu, Ninghao Liu, Lichao Sun, Haotao Mi, et al. 2026. Long-Horizon-Terminal-Bench: Testing the Limits of Agents on Long-Horizon Terminal Tasks with Dense Reward-Based Grading. arXiv:2607.08964 [cs.AI] https://arxiv.org/abs/2607.08964

[14] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2023. Holistic Evaluation of Language Models. *Transactions on Machine Learning Research* (2023). https://openreview.net/forum?id=iO4LZibEqW

[15] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 2024. AgentBench: Evaluating LLMs as Agents. In *International Conference on Learning Representations*. OpenReview.net, Vienna, Austria. https://openreview.net/forum?id=zAdUB0aCTQ

[16] Zefang Liu and Yinzhu Quan. 2026. EconWebArena: Benchmarking Autonomous Agents on Economic Tasks in Realistic Web Environments. arXiv:2506.08136 [cs.CL] https://arxiv.org/abs/2506.08136

[17] Mike A. Merrill, Alexander Glenn Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, et al. 2026. Terminal-Bench: Benchmarking Agents on Hard, Realistic Tasks in Command Line Interfaces. arXiv:2601.11868 [cs.AI] https://arxiv.org/abs/2601.11868

[18] Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. GAIA: a benchmark for General AI Assistants. arXiv:2311.12983 [cs.CL] https://arxiv.org/abs/2311.12983

[19] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. arXiv:2302.04761 [cs.CL] https://arxiv.org/abs/2302.04761

[20] Issa Sugiura, Daichi Hattori, Kazuo Araragi, Keita Ogawa, Shota Onose, Taro Makino, Teppei Usuki, and Takashi Ishida. 2026. CoffeeBench: Benchmarking Long-Horizon LLM Agents in Heterogeneous Multi-Agent Economies. arXiv:2606.16613 [cs.AI] https://arxiv.org/abs/2606.16613

[21] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An Open-Ended Embodied Agent with Large Language Models. arXiv:2305.16291 [cs.AI] https://arxiv.org/abs/2305.16291

[22] Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z. Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. 2025. TheAgentCompany: Benchmarking LLM Agents on Consequential Real World Tasks. arXiv:2412.14161 [cs.CL] https://arxiv.org/abs/2412.14161

[23] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. arXiv:2406.12045 [cs.AI] https://arxiv.org/abs/2406.12045

[24] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629 [cs.CL] https://arxiv.org/abs/2210.03629

[25] Yulai Zhao, Haolin Liu, Dian Yu, S. Y. Kung, Meijia Chen, Haitao Mi, and Dong Yu. 2025. One Token to Fool LLM-as-a-Judge. arXiv:2507.08794 [cs.LG] https://arxiv.org/abs/2507.08794

[26] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. WebArena: A Realistic Web Environment for Building Autonomous Agents. arXiv:2307.13854 [cs.AI] https://arxiv.org/abs/2307.13854

Appendix

A Task Families

The public suite contains five fixed tasks in each of ten families. Market selection rewards sufficient research, viable-market choice, and preserved cash. First revenue rewards customers, recurring revenue, runway, and reputation. Retention improvement rewards quality, support, customer preservation, and focus. Churn recovery rewards stabilized customers, reduced churn, restored reputation, and adequate capacity. Demo Day traction rewards growth, recurring revenue, cash, and credibility. Pricing rewards movement toward sustainable market-relative prices. Runway preservation rewards survival and disciplined cost reduction. Pivot decision rewards diagnosis and timely market change. Fundraising rewards credible traction before simulated capital raising. Channel expansion rewards scaling working offers through campaigns and partnerships.

Human-readable task cards and the executable setup and scoring functions ship with the public release.

B Prompt and Response Contract

Each provider receives the task description, current structured observation, allowed action schemas, and recent memory. The expected response root is:

```
{
  "rationale": "brief explanation",
  "actions": [
    {"type": "research_market", ...}
  ]
}
```

Only entries in `actions` are executable. The parser extracts a JSON object, validates the root and action list, checks required fields and numeric types, and records a diagnostic category when parsing fails. Exact prompt templates and content hashes are included in the public release.

C Core Transition Sketch

After actions execute, each offer settles weekly customers with seeded Poisson draws. With awareness a, quality q, price p, reputation r, market demand d, competition c, willingness to pay w, and support load ℓ:

$$\phi(p,w) = \max\big(0.08,\ 1.25 - p/\max(1,w)\big),$$

$$\lambda_{\text{new}} = \max\Big(0,\ \big(d\,a\,q\,\phi(p,w)\,r - 0.18c\big)\,5.5\Big),$$

$$\rho = \max\big(0.01,\ 0.08\ell + (0.65 - q)\,0.05 - 0.025r\big).$$

As in the main text, juxtaposition denotes multiplication; $\max(0,\cdot)$ and $\max(0.01,\cdot)$ are floors; $\phi(p,w)$ is price fit (lower when p is high relative to w); λ_{new} and ρ are Poisson rates for new and churned customers. Revenue is $p \cdot$ customers; support cost scales with $\ell \cdot$ customers. Support overload reduces reputation when required support exceeds capacity. Research replaces noisy signals with exact market values; campaigns raise awareness; offer improvement raises quality; hiring raises capacity; cost cutting reduces burn; fundraising injects cash under traction and reputation gates. Market momentum evolves with a seeded volatility shock. All constants and action branches are documented in the released simulator source.

D Score Construction

Every task score is a sum of documented positive components and penalties, then clamped to $[0, 100]$. Notation: a leading coefficient is a maximum weight (so $45\min(1, C/C^\star)$ means $45 \times \min(1, C/C^\star)$); $\min(1,x)$ awards partial credit up to the target and never more than the full weight; $\mathbf{1}[\cdot]$ is 1 when the bracketed condition is true and 0 otherwise. Market selection uses

$$f = 25\min\big(1, R/3\big) + 55 \cdot \mathbf{1}[m \in \mathcal{M}_{\text{good}}] + 20\min\big(1, \text{cash}/8500\big).$$

Here the middle term contributes 55 points only if the chosen market m lies in the viable set $\mathcal{M}_{\text{good}}$. First revenue (targets $C^\star, v^\star, K^\star$ vary by variant) uses

$$f = 45\min\big(1, C/C^\star\big) + 25\min\big(1, v/v^\star\big) + 20\min\big(1, \text{cash}/K^\star\big) + 10\min\big(1, r/0.65\big).$$

The four weights sum to 100 and measure customers, peak weekly revenue, cash, and reputation. The remaining families follow the same weighted-component pattern:

- retention: quality, support load, customer preservation, and focus;
- churn shock: stabilized customers, reduced churn, reputation, and capacity;
- Demo Day: growth, recurring revenue, cash, and credibility;
- pricing: fit to a sustainable market-relative price band;
- runway: survival and disciplined cost reduction;
- pivot: diagnosis and timely market change;
- fundraising: traction and reputation gates before capital;
- channel: scaling a working offer via campaigns and partnerships.

Exact numeric weights for every component are fixed in the released scoring code.

The official aggregate metric is the mean task score; we also report median and IQR across tasks. A task is solved at score ≥ 70, chosen as a fixed "clearly successful" operating cut between the generic-heuristic and task-aware means rather than a tuned optimum; we do not sweep alternative thresholds here. The public-test column summarizes tasks FND-031–050; both public splits are visible. Task-bootstrap intervals resample the 50 score observations and estimate sensitivity to task composition, not provider sampling.

E Task-Aware Heuristic Summary

The task-aware calibration policy first maps each task identifier to its family, then applies a short family-specific branch; otherwise it falls back to the generic heuristic. Representative rules:

- Market selection / first revenue: research top unresearched markets early, then build an offer on the best demand–price–competition score.
- Retention: improve quality and support when below thresholds.
- Churn shock: hire if capacity is low, then support or improve.

- Demo Day: improve, campaign, hire under load, and support.
- Pricing: interview customers, then set price near $0.78w$, optionally campaign.
- Runway: cut cost, light support, selective improvement.
- Pivot: research the target market; pivot when stalled; then reprice, improve, or campaign.
- Fundraising: support or improve, then raise funding.
- Channel: partner a channel, improve, campaign, and hire under load.

This policy uses family identity unavailable to ordinary hosted prompts unless leaked by the task text; treat it as a calibration ceiling, not a human expert. On the frozen private remix, deterministic calibration alone already reverses the public ladder (task-aware below generic heuristic), which is a warning against reading the public 80.90 gap as stable strategic headroom.

F Diagnostics

Outputs include invalid actions, over-budget decisions, provider-error categories, action counts, simulated API cost, latency, and token counts when available. Audit mode additionally records model identifier, task and week, prompt hash, redacted response, usage, estimated cost, and call latency. Automated redaction is a safety layer and traces must be inspected before public release.

Provider/parser failures use documented fallback behavior and remain in official scores. The paper's sensitivity analysis reports affected-task and unaffected-task means without treating them as a corrected counterfactual.

G Submission Validation

The validator requires:

- exactly 50 unique official task identifiers;
- score bounds and pass flags consistent with the threshold;
- aggregate solved, solve-rate, and average-score consistency;
- public development/test summaries;
- execution diagnostics and declared policy/model metadata; and
- complete result objects for every task.

The frozen paper registry records each accepted evidence artifact, submission report, model label, run seed, headline outcomes, and SHA-256 digest. For newer audited runs it also records call-level model identifiers. Some earlier validated rows predate that audit field; their model labels are frozen from the variant-specific execution configuration and evidence artifact. Future submission schemas should require the exact API model identifier at the run root.

H Artifact Map

The public release contains:

- the evaluation harness, task definitions, and scoring code;
- table, figure, and result-analysis generation scripts;
- a frozen hosted-model evidence registry with digests;
- validated task-level provider results and submission reports; and
- a benchmark card, datasheet, responsible-use note, reproduction guide, environment report, and integrity manifests.

I Additional Limitations

The environment excludes many real organizational concerns, including law, accounting, hiring, workplace safety, data governance, and interpersonal dynamics. Actions are coarse abstractions. The synthetic markets encode designer assumptions and may admit simulator-specific strategies. The five tasks per family provide diagnostic breadth but limited statistical resolution. Provider model aliases can change after collection, and the single-run study does not quantify inference variance. No human-subject evaluation, founder comparison, official private leaderboard, or real-world outcome validation is claimed. Recommended upgrades: repeated hosted runs, standardized retry budgets, stochastic task variants, expert calibration, and a held-out private evaluation split.

Table 5: Provider-error sensitivity with normalized rates. Err/task is error decisions divided by 50 tasks; Aff. tasks is the count (and fraction) of tasks with ≥1 error decision. Means stratify observed tasks and are not counterfactual corrected scores.

Model	Errs	Err/task	Aff. tasks	Aff. mean	Clean mean	Dominant category
Gemini 3.5 Flash	59	1.18	25 (50%)	73.57	61.80	rate limit (30)
Grok 4.5	0	0.00	0 (0%)	–	66.53	–
GPT-5.6 Sol	0	0.00	0 (0%)	–	66.39	–
Kimi K3	70	1.40	11 (22%)	46.47	71.04	rate limit (64)
Claude Sonnet 5	0	0.00	0 (0%)	–	63.90	–
DeepSeek V4 Reasoner	3	0.06	3 (6%)	60.66	62.55	invalid JSON (2)
Claude Sonnet 4.5	0	0.00	0 (0%)	–	61.09	–
DeepSeek Chat	0	0.00	0 (0%)	–	56.59	–
GLM 4.5 Air	0	0.00	0 (0%)	–	54.78	–
Gemini 2.5 Flash	340	6.80	44 (88%)	50.33	70.03	I/O error (332)
Grok 4.3	3	0.06	3 (6%)	44.23	53.12	no JSON object (2)