International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Agentic Flakiness: Reframing LLM Agent Reliability Through the Lens of Flaky-Test Research

Omkar Manohar Ghag

Independent Researcher, M.S. in Telecommunication, University of Pittsburgh, USA
ORCID: 0009-0007-1519-7052

Abstract: ***Large language model (LLM) agents are nondeterministic: the same task, run twice under identical conditions, may succeed once and fail once. The agent-evaluation literature has begun to measure this- most directly through the $pass\hat{}k$ metric introduced with τ-bench- but treats it as a novel phenomenon. It is not. Software engineering has studied the same phenomenon under the name flaky tests since at least 2014, and has produced a root-cause taxonomy, detection algorithms, cost models, and CI-level management policy. This paper argues that agent evaluation should import that body of work. We give a structural mapping from established flaky-test root causes to sources of agentic nondeterminism, showing the correspondence is mechanistic rather than metaphorical: batch-noninvariant GPU kernels are the agentic form of floating-point reduction-order flakiness. Building on the latent-success representation of recent Markov-chain reliability work, we prove that $pass\hat{}k$ converges as $k \to \infty$ to the agent's deterministic pass rate D, and is flat in k if and only if no task is flaky. This yields the Flake Gap $\Phi = pass\hat{}1 - D$: the portion of a benchmark score an agent cannot reproduce on demand. Finally we show Φ is not identifiable from currently published numbers—fitting a zero-and-one- inflated Beta latent model to reported τ-bench results constrains D only to $[0.00, 0.42]$ on τ-retail, leaving the deployable pass rate uncertain by 61% of the headline score. Benchmarks should therefore publish per-task trial counts rather than aggregate $pass\hat{}k$***

Keywords: LLM agents, agent evaluation, flaky tests, nondeterminism, reliability, software testing, survival analysis.

1. Introduction

An LLM agent that books a flight correctly on Monday may fail the identical booking on Tuesday. Nothing changed- not the prompt, not the tools, not the database. The agent-evaluation community has documented this behavior care-fully. Yao et al. [1], introducing τ-bench, proposed the pass^k metric- the probability that *all* k independent trials of a task succeed- precisely because averaging over trials concealed how unreliable agents were. Their headline finding was stark: GPT-4o achieved roughly 61% on τ-retail at pass^1, but un-der 25% at pass^8.

This framing treats nondeterministic task outcomes as a new problem introduced by stochastic language models. We argue it is an old problem in a new setting. A software test that passes on some runs and fails on others, with no change to the code under test, is called a *flaky test*. Luo et al. [2] published the first systematic empirical study of flaky tests in 2014, analyzing 201 flakiness-fixing commits across 51 open-source projects and classifying ten root causes. Twelve years of subsequent work has produced detection tools, repair techniques, industrial cost measurements, and management pol-icy. Google reported that around 16% of its tests exhibit flakiness; GitHub reported that in 2020, roughly one commit in eleven had at least one red build attributable to a flaky test [3].

The two literatures have essentially no contact: agent-evaluation papers do not cite flaky-test research, and flaky-test surveys do not discuss LLM agents. The cost is concrete- the agent community is re-deriving results soft-ware engineering settled years ago, while lacking the machinery to answer the question its own metrics raise.

That question is the following. When an agent scores 0.69 on a benchmark, how much of that score can it reproduce on demand? A score assembled from tasks the agent solves reliably is worth something quite different, operationally, from an identical score assembled from coin flips. Current reporting does not distinguish them. Our contributions:

1) A structural mapping (Section III) between the flaky-test root-cause taxonomy and sources of agentic non-determinism, including an exact mechanistic correspondence at the numerical level.
2) Two results extending the latent-success formalism of [10] (Section IV): pass^$k \to D$, the deterministic pass rate, as $k \to \infty$; and $pass\hat{}k$ is constant in k if and only if every task is deterministic. Together these define the *Flake Gap* Φ.
3) An identifiability analysis (Section V) showing Φ can-not be recovered from published τ-bench figures, with a quantified bound on the resulting uncertainty.
4) A set of reporting and management recommendations (Section VI) ported from CI practice.

2. Background and Related Work

a) Reliability metrics for agents

Early agent benchmarks—SWE-bench [4] for code, and agent scaffolds such as ReAct [5]—reported single-trial success. τ-bench [1] introduced $pass\hat{}k$ alongside the familiar $pass\hat{}k$, giving, for a task run n times with c successes, the unbiased

estimators

$$\widehat{\text{pass}\hat{}k} = \frac{\binom{c}{k}}{\binom{n}{k}}, \qquad \widehat{\text{pass@}k} = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}. \qquad (1)$$

The asymmetry matters: *pass^k* rewards *discovery* (at least one success), while pass^*k* rewards *consistency* (no failures). Deployment cares about the latter.

Closest to our formal treatment is TraceToChain [10], which fits agent execution traces to an absorbing discrete-time Markov chain and shows that *pass^k*, pass^*k*, and the reliability decay curve are projections of a single first-passage distribution. Their Theorem 4 derives the latent-heterogeneity representation we adopt in (3), along with the Jensen bound restated as Proposition 3 below, noting that equality requires an almost-surely constant success probability. We take that representation as established and push it in a direction they do not consider: the $k \to \infty$ limit, and what that limit implies for reporting practice. Notably, their machinery comes from classical software reliability theory- Kemeny-Snell, Cheung, Goel–Okumoto- not flaky-test research: the two literatures remain disconnected even in the work closest to ours. Separately, work on replayable financial agents [11] observes that a fully deterministic agent has *pass^k* invariant in *k*; Proposition 2 supplies the converse.

Separately, Cemri et al. [6] built MAST, a taxonomy of multi-agent system failures derived from over 1,600 anno-tated execution traces across seven frameworks, identifying 14 failure modes in three categories with inter-annotator agreement $\kappa = 0.88$. MAST catalogs *what* goes wrong se-mantically; it does not model the statistics of whether a given failure recurs.

A third thread models reliability against task length. Kwa et al. [7] defined the 50%-task-completion time horizon and found it doubling roughly every seven months. Ord [8] ob-served that this data is well fit by a constant hazard rate, im-plying exponentially decaying success and a per-agent "half-life." That model has since been revised: using a Weibull survival function

$$S(t) = \exp[-(\lambda t)^{\kappa}], \qquad (2)$$

Hamilton's analysis found κ significantly below 1 for every model examined—clustered near 0.6, against roughly 0.37 for humans- indicating hazard rates that *decline* as a task proceeds rather than staying constant. Ord has publicly ac-cepted this correction [9]. We adopt the Weibull view where task-length effects matter, but note that our main results in Section IV are distribution-free and do not depend on which survival family is correct.

b) Flaky tests

A test is flaky if it produces different outcomes across runs on unchanged code. Luo et al. [2] classified root causes into ten categories, of which async-wait accounted for nearly half of the fixes examined and test-order dependency for about 12%. Parry et al. [12] surveyed the field; Rasheed et al. [3] ex-tended coverage to 560 academic and 91 practitioner sources, adding platform dependency, external-state dependency, and algorithmic nondeterminism- the last introduced by Dutta et al. [13] specifically for probabilistic and machine-learning code.

Three findings from this literature are directly relevant here. First, detection is expensive: rerun-based methods dominate, and Gruber et al. [15] estimated that roughly 170 re-runs may be needed to establish that a Python test is not flaky for non-order-dependent reasons. DeFlaker [16] avoids re-runs using differential coverage- an approach with no cur-rent agentic analogue. Second, management, not elimination, is the industrial norm: the dominant response is *quarantine*, isolating flaky tests from the critical path pending diagnosis. Third, and most provocatively, Harman and O'Hearn [17] argued that all tests should be assumed flaky by default, with flakiness quantified rather than treated as a binary defect.

Rasheed et al. [3] explicitly flag as open the question of whether "a test that only passes after several reruns provide[s] the same level of assurance as a test that always passes." Section IV answers the agentic form of that question.

3. A Structural Mapping

Fig. 1 maps flaky-test root causes onto agentic ones. Most rows are close structural analogues; one is an identity.

Async wait, the largest single category of test flakiness [2], occurs when a test issues an asynchronous call and pro-ceeds without correctly waiting. Agents fail identically when a tool call returns before the underlying state has settled. **Test-order dependency** arises from state leaking between tests through shared fixtures; the agentic analogue is context-order dependence, since conversation history is a mutable shared fixture and a step's outcome depends on what earlier steps wrote into it- MAST's "inter-agent misalignment" category [6] catalogs instances. **Randomness** maps to sampling temperature: Dutta et al. [13] identified unsynchronized seeds and algorithmic nondeterminism as flakiness causes in ML code, exactly the mechanism by which a nonzero-temperature agent flakes.

The final row is the strongest claim. Luo et al. [2] list floating-point operations as a root cause, because non-associative addition under varying reduction order yields different results. He and colleagues [14] showed that LLM in-ference is nondeterministic even at temperature zero for precisely this reason: serving frameworks batch incoming re-quests by arrival timing, and several kernels are not batch-invariant, so the same prompt in a batch of 1 and a batch of 32 takes different numerical paths. The two phenomena are not analogous- they are the same phenomenon, one layer apart.

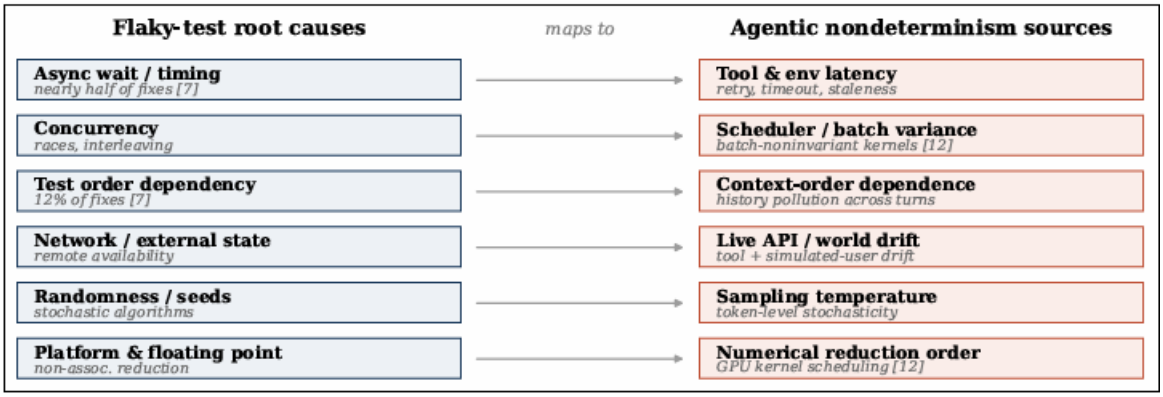

Figure 1: Structural mapping from established flaky-test root causes [2, 3] to sources of nondeterminism in LLM agent execution. The bottom row is an exact mechanistic correspondence, not an analogy: both arise from non-associative floating-point reduction under variable scheduling [14]

This matters practically: it means an agent can flake with no sampling randomness at all, and that a nontrivial share of agentic flakiness is an infrastructure property rather than a model property.

4. A Formal Framework

a) Latent success model

Let a benchmark contain tasks indexed by i, and let $p_i \in [0, 1]$ be the latent probability that the agent succeeds on task i in a single trial, under fixed scaffold and environment. Write P for the distribution of p_i induced by drawing a task at ran-dom. Assuming trials are conditionally independent given the task—the assumption already implicit in -

$$\text{pass}^\wedge k = \mathbb{E}_P\left[p^k\right], \qquad \text{pass@}k = 1 - \mathbb{E}_P\left[(1-p)^k\right]. \quad (3$$

So *pass^k,* is the k-th raw moment of the latent success distribution- the representation established by Tran-Truong and Le [10]. Our contribution begins with what it implies as k grows.

b) Limiting behavior

Define the *deterministic pass rate*

$$D = \mathbb{P}_P(p = 1), \qquad (4)$$

the fraction of tasks the agent solves with certainty.

Theorem 1 (pass^k converges to the deterministic pass rate)**.**

$$\lim_{k\to\infty} \text{pass}^\wedge k = D.$$

Proof. For $p \in [0, 1)$, $p^k \to 0$; for $p = 1$, $p^k = 1$ for all k. Thus $p^k \to \mathbf{1}\{p = 1\}$ pointwise. Since $0 \le p^k \le 1$ and the constant 1 is integrable on a probability space, dominated convergence gives $\mathbb{E}[p^k] \to \mathbb{E}[\mathbf{1}\{p = 1\}] = \mathbb{P}(p = 1) = D$.

Proposition 2 (Flatness characterizes determinism)**.** pass^k = pass^1 *for all* $k \ge 1$ *if and only if* $p \in \{0, 1\}$ *almost surely.*

Proof. ($\Leftarrow$) If $p \in \{0,1\}$ a.s. then $p^k = p$ a.s., so all moments coincide. ($\Rightarrow$) Take $k = 2$: $\mathrm{E}[p^2] = \mathrm{E}[p]$ gives $\mathrm{E}[p(1-p)] = 0$. The integrand is nonnegative on $[0,1]$, so $p(1-p) = 0$ a.s., i.e. $p \in \{0, 1\}$ a.s. *$p \in \{0, 1\}$ a.s.*

Proposition 3 (Jensen bound [10])**.** pass^$k \ge$ (pass^1)^k*, with equality iff p is a.s. constant.*

Stated here for completeness; it follows from convexity of $x \mapsto x^k$ on $[0, 1]$ for $k \ge 1$. It has a practical reading: an evaluator who estimates pass^8 by raising pass^1 to the eighth power will always be pessimistic, and the size of the error measures how heterogeneous task difficulty is- not how flaky the agent is.

Proposition 2 isolates the signal. The *decay of* pass^k *in* k, not its level, is what measures flakiness. Two agents with identical pass^1 can have completely different decay profiles (Fig. 2).

c) The Flake Gap

Theorem 1 licenses a definition.

Corollary 4 (Flake Gap)**.** *Let*

$$\Phi = \text{pass}^\wedge 1 - D = \mathbb{E}_P[p \cdot \mathbf{1}\{p < 1\}] \ge 0. \qquad (5)$$

Then Φ *is the portion of the reported single-trial score contributed by tasks the agent cannot solve reliably, and* $D =$ pass^1 $- \Phi$ *is the portion it can.*

Φ separates two deficits that a single score conflates. A task with $p_i = 0$ is a *capability* failure: the agent cannot do it, and more trials will not help. A task with $p_i = 0.5$ is a *reliability* failure: the capability is present but not dependable. These call for different engineering responses- training or scaffolding versus retries, verification, or quarantine and pass^1 prices them identically.

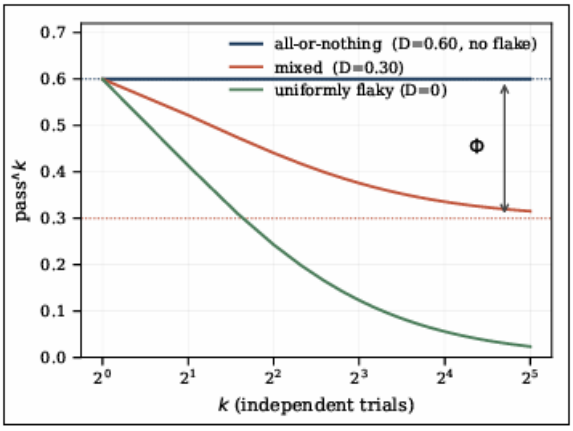

Figure 2: Three latent distributions with *identical* pass^1 = 0.60 but different flakiness. Dotted lines mark each D, the limit from Theorem 1. The level of pass^k says little; its decay is the signal

d) A parametric latent model

To estimate Φ from finite trials we need a family for P that admits atoms at both endpoints- a continuous density cannot represent deterministic tasks. We use a zero-and-one-inflated Beta: with probability π_0 the task is never solved, with probability π_1 always, and otherwise $p \sim \text{Beta}(\alpha, \beta)$. Using the closed form for Beta raw moments,

$$\text{pass}^{\wedge}k = \pi_1 + (1 - \pi_0 - \pi_1) \prod_{j=0}^{k-1} \frac{\alpha + j}{\alpha + \beta + j}, \qquad (6)$$

and $D = \pi_1$ directly, consistent with Theorem 1 since the product term vanishes as $k \to \infty$ for any finite α, β. Both (6) and Theorem 1 were verified numerically to six decimal places.

Is the Flake Gap Identifiable?

Equation (6) has four free parameters. Published agent evaluations typically report pass^1 and one or two deeper values. This suggests an identifiability problem, and there is one.

We take publicly reported τ-bench results for Claude 3.5 Sonnet- pass^1 = 0.692, pass^4 = 0.462 on τ-retail, and 0.460/0.225 on τ-airline [Leaderboard figures for the τ-bench harness [1]. Scores are scaffold-and simulator-dependent; we use them as a realistic worked example, not as authoritative measurements] and ask which values of D are consistent with them. For each candidate $D = \pi_1$ on a grid, we solve for (π_0, α, β) reproducing both reported values to within 10^{-8}, discarding degenerate solutions with α or β above 20.

The feasible set is not a point but a wide interval (Fig. 3): $D \in [0.00, 0.42]$ with $\Phi \in [0.27, 0.69]$ on τ-retail, and $D \in [0.00, 0.21]$ with $\Phi \in [0.25, 0.46]$ on τ-airline. On τ-retail, every value of D from 0.00 to 0.42 is exactly consistent with the published numbers. The agent may solve 42% of tasks with certainty, or none of them, and the reported figures cannot tell us which. The uncertainty in the deployable pass rate is 61% of the headline score.

The constructive reading is visible in Fig. 3: the curves are pinned where data exists and diverge past it. The two τ-retail extremes differ at pass^8 (0.30 versus 0.43)- so the ambiguity is resolvable, and cheaply, by reporting deeper k. Better still, the per-task trial counts (c_i, n_i) that benchmarks already compute internally identify the latent distribution directly, with no parametric assumption. Aggregating them away destroys the information.

This is the reporting discipline flaky-test practice arrived at: Facebook's Probabilistic Flakiness Score assigns each test an individual reliability estimate rather than a suite-level aver-age, so regressions in specific tests can be detected and acted on [3]

5. Implication

a) Retry budgets have a break-even point

Retrying is the standard industrial response to flakiness and transfers directly. For latent p, success within a budget of r attempts has probability $1-(1-p)^r$, with $1/p$ expected at-tempts to first success (Fig. 4). The asymmetry is severe: at $p = 0.7$ three attempts reach 97.3%, while at $p = 0.3$ the same budget reaches 65.7% and costs 3.3 attempts. Suite-level cost $\sum_i 1/p_i$ is dominated by the worst tasks and diverges as any $p_i \to 0$, so retry budgets bound cost only once capability failures ($p_i \approx 0$) are separated from reliability failures- exactly the separation Φ provides

b) Agent quarantine

CI systems rarely fix flaky tests immediately; they quarantine them- removing them from the blocking path, filing a ticket, tracking them on a dashboard- with practitioner guidance that quarantine be bounded in size and duration or it silently erodes coverage [3]. The agentic analogue routes task classes with estimated p_i below a threshold to human re-view or a verification step rather than autonomous execution, while remaining instrumented. The warning transfers too: an unbounded quarantine is indistinguishable from reduced coverage.

c) Reporting recommendations

Three concrete changes: (i) publish per-task trial counts (c_i, n_i) alongside aggregate scores; (ii) report pass^k at several k, since the decay profile carries the signal any single k omits; (iii) report D with an interval, since D rather than pass^1 is what a deployment can rely on.

6. Threats to Validity

Our formalism assumes trials are conditionally independent given the task- the assumption already embedded in the pass^k estimators (1), but an imperfect one: caching, shared infrastructure state, and correlated batch composition can couple trials, and positive correlation would bias D upward.

The latent p_i is also defined relative to a fixed scaffold, model

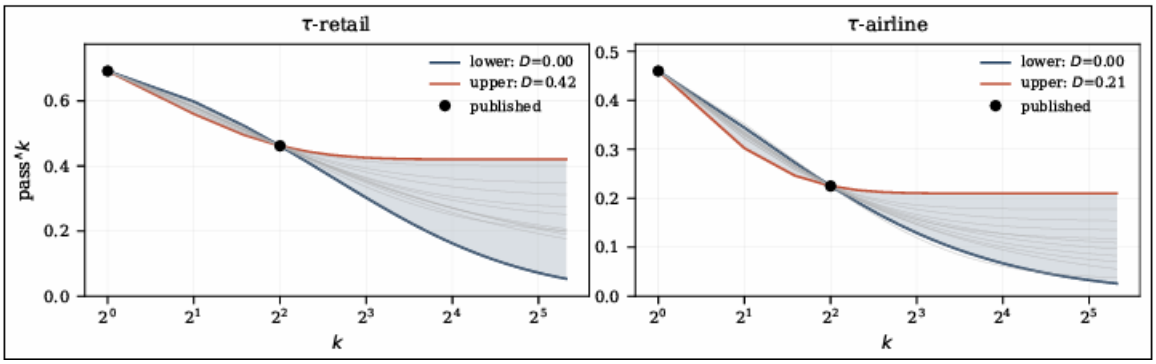

Figure 3: All latent distributions exactly reproducing the reported τ-bench pass^1 and pass^4 (black markers). Curves are pinned at $k \leq 4$ and fan out beyond it: reported numbers constrain D only to a wide interval. Reporting pass^8 would separate these hypotheses

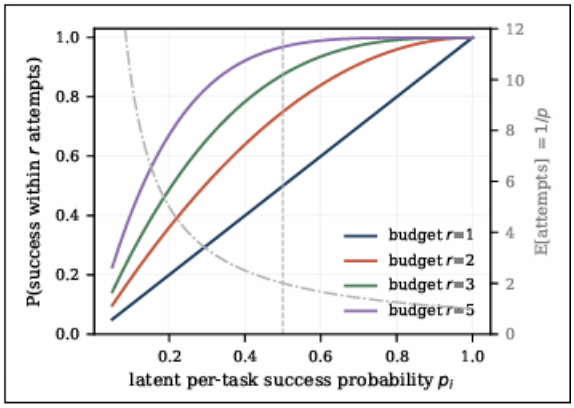

Figure 4: Retry economics. Solid: success probability within budget r. Dash-dotted: expected attempts $1/p$, which diverges for low-p tasks and dominates suite cost.

version, and environment, so D estimated in one harness need not transfer to another.

The analysis in Section V is conditional on the zero-and-one-inflated Beta family; a different family yields a different feasible set. The qualitative conclusion is family-independent- two moments cannot pin down a distribution with atoms at both endpoints plus a continuous component- but our numerical values illustrate the magnitude of the problem rather than estimate any model's true D.

Finally, Section III is argued structurally, not measured. Establishing empirically that agentic failures distribute across these categories as test flakiness does- as Luo et al. [2] did by examining fix commits- requires an annotated corpus of agent traces. We regard this as the most valuable next step; MAST-Data [6] is a plausible substrate.

7. Conclusion

Agent nondeterminism is not a new class of problem. It is flakiness, arriving in a new system layer, and in at least one case- batch-noninvariant numerical reduction- it is the identical mechanism software testing has documented for a decade. Recognizing this makes formal progress available: pass^k is the k-th moment of a latent success distribution, it converges to the deterministic pass rate, and the gap between that limit and the headline score is the part of a result that cannot be relied upon. That gap is currently unmeasurable from what benchmarks publish, and the fix- report per-task trial counts, or deeper k- is inexpensive. The larger opportunity is the decade of detection, repair, and management technique flaky-test research has already built.

References

[1] S. Yao, N. Shinn, P. Razavi, and K. Narasimhan, "τ-bench: A bench-mark for tool-agent-user interaction in real-world domains," in *Proc. ICLR*, 2025. arXiv:2406.12045.

[2] Q. Luo, F. Hariri, L. Eloussi, and D. Marinov, "An empirical analysis of flaky tests," in *Proc. FSE*, 2014, pp. 643–653.

[3] S. Rasheed, A. Tahir, J. Dietrich, N. Hashemi, and L. Zhang, "Test flakiness' causes, detection, impact and responses," *J. Systems and Software*, 2024. arXiv:2212.00908.

[4] E. Jimenez et al., "SWE-bench: Can language models resolve real- world GitHub issues?" in *Proc. ICLR*, 2024. arXiv:2310.06770.

[5] S. Yao et al., "ReAct: Synergizing reasoning and acting in language models," in *Proc. ICLR*, 2023. arXiv:2210.03629.

[6] M. Cemri et al., "Why do multi-agent LLM systems fail?" in *Proc. NeurIPS*, 2025. arXiv:2503.13657.

[7] T. Kwa et al., "Measuring AI ability to complete long software tasks," in *Proc. NeurIPS*, 2025. arXiv:2503.14499.

[8] T. Ord, "Is there a half-life for the success rates of AI agents?" arXiv:2505.05115, May 2025.

[9] T. Ord, "Hazard rates for AI agents decline as a task goes

on," *toby-ord.com*, Feb. 2026.
[10] P. T. Tran-Truong and X.-B. Le, "Measuring the unmeasurable: Markov chain reliability for LLM agents," arXiv:2604.24579, Apr. 2026.
[11] "Replayable financial agents: A determinism-faithfulness harness for tool-using LLM agents," arXiv:2601.15322, 2026.
[12] O. Parry, G. M. Kapfhammer, M. Hilton, and P. McMinn, "A survey of flaky tests," *ACM TOSEM*, vol. 31, no. 1, pp. 1–74, 2021.
[13] S. Dutta, A. Shi, R. Choudhary, Z. Zhang, A. Jain, and S. Misailovic, "Detecting flaky tests in probabilistic and ML applications," in *Proc. ISSTA*, 2020, pp. 211–224.
[14] H. He and Thinking Machines Lab, "Defeating nondeterminism in LLM inference," *Connectionism*, Sep. 2025.
[15] M. Gruber, S. Lukasczyk, F. Kroiß, and G. Fraser, "An empirical study of flaky tests in Python," in *Proc. ICST*, 2021. arXiv:2101.09077.
[16] J. Bell, O. Legunsen, M. Hilton, L. Eloussi, T. Yung, and D. Marinov, "DeFlaker: Automatically detecting flaky tests," in *Proc. ICSE*, 2018, pp. 433–444.
[17] M. Harman and P. O'Hearn, "From start-ups to scale-ups: Opportuni-ties and open problems for program analysis," in *Proc. SCAM*, 2018, pp. 1–23.