FlowX.AI

TECHNICAL PAPER

LIV · VALIDITY ESTIMATES FOR FACTS IN ENTERPRISE AGENT MEMORY

Which Facts Have Gone Stale?

A per-fact validity estimate learned from an organisation's own verification history, tested against simple schedules on public change data and in simulation, with every hypothesis registered before the test.

Bogdan Răduță
Head of Research, FlowX.AI
bogdan.raduta@flowx.ai

FlowX.AI · Applied Research
November 2026 · Version 1.0.0 · doi.org/10.5281/zenodo.24077472

ABSTRACT

Agents that act on stored enterprise facts act on values that may have changed since they were last verified. We study a per-fact validity estimate with calibration as a target, learned from an organisation's own change and re-verification events, that gives the probability that a stored value has not changed since its last check and chooses which facts to re-verify under a fixed budget. The ingredients are established; we evaluate their combination for agent memory, with the per-class Poisson estimator of Cho and Garcia-Molina as the main baseline, on LIV-Bench: `tacit-drift`, a seeded simulator whose parameters encode stated assumptions rather than real-world rates, and a task built from Wikidata statement lifespans. In the simulator, at an equal budget, a discrete-time hazard model finds 18.8 to 71.5 percent more stale facts per unit of check cost than the better of per-class Poisson and a calendar schedule in all 8 pre-registered cells with heterogeneous entities and activity signals, and does not beat per-class Poisson in the negative control. On Wikidata we detected no improvement (−0.3 percent, interval −2.1 to 1.7), also with co-change features added by amendment, and 4 other model-based baselines found more stale facts per check. In a simulated loan-intake process, consequence-weighted re-verification lowered test-window cost per case by 6.8 percent against the better freshness-only policy, short of the pre-registered 10 percent, and raised failed actions. In a second pre-registered experiment with complete case follow-up and frozen models, an avoidability-aware allocation lowered total cost per enrolled case by 1.55 units (interval 1.33 to 1.77) against a freshness-only allocation that uses the same validity scores but not the rule and cost tables, at least the smallest worthwhile saving of 1 unit, without lowering completion; the pilot decisions before that test are disclosed. In a third pre-registered experiment in the same simulated process that changed only the scorer under that allocation, the learned validity model lowered total cost per enrolled case against per-class Poisson by 0.60 units (interval 0.53 to 0.68), less than the smallest worthwhile saving, without lowering completion.

KEYWORDS Agent memory · Fact staleness · Survival analysis · Re-verification budgets · Pre-registration · Simulation

1 Introduction

An agent that drafts a loan offer, routes a compliance case or sends a contract reads facts from memory: who employs the customer, where a company has its headquarters, who owns it. Each fact was true when it was last verified. Since then it may have changed without anyone noticing. Know your customer (KYC) rules already treat this as an operational duty: the FATF asks that customer due diligence data be "kept up-to-date and relevant by undertaking reviews of existing records, particularly for higher-risk categories of customers" [9], and sets no fixed interval, while national rulebooks can fix one; the Reserve Bank of India's FAQ states that periodic updation "shall be carried out at least once in every two years for high-risk, eight years for medium risk and ten years for low-risk customers" [19]. Industry points of view describe a move from fixed periodic reviews to event-driven monitoring [16].

This paper asks two questions about that duty when checks cost money. Can a per-fact model of the probability that a stored fact is still valid, learned from the organisation's own checks and the signals it already receives, choose which facts to re-verify better than the established change-rate estimator from web crawling, at the same check budget? And, given such an estimate, does it pay to weight checks by the cost that a stale fact would cause in the actions that use it?

What is established. Estimating change rates from irregular checks that observe only whether a change occurred, and scheduling refreshes under a budget, is established in web-crawl research [5, 6]. Learning the probability that an attribute value changes within a time gap exists in temporal record linkage [14]. Predicting the stability of knowledge-base facts exists [8, 21]. Budgeted, consequence-weighted re-verification of agent memory also exists: ERRAND funds a priced re-validation only when its value of information, which combines a staleness belief with the loss of serving a stale item, clears a running wage under an action budget [24]. Our greedy rule, P(stale) times a weight over the check cost, is close to that index. We do not claim any of these ingredients; Section 2 sets out what differs from ERRAND, the closest work.

What we found. Every finding below is pre-registered unless labelled otherwise. *Scoring.* In the `ta-citdrift` simulator, under stated assumptions, a covariate-conditioned discrete-time hazard model fitted to interval-censored check histories finds 18.8 to 71.5 percent more stale facts per unit of check cost than the better of per-class Poisson and a calendar schedule in every pre-registered cell with heterogeneous entities and activity signals, and does not beat per-class Poisson where Poisson is correctly specified. On public Wikidata change histories no improvement was detected, also with co-change features added by a pre-registered amendment, and 4 other survival and decay models did better than the hazard policy. The gain therefore depends on informative per-fact signals, which we have not shown in public data. *Weighting.* In a simulated loan-intake process, weighting checks by the cost of acting on a wrong value, taken from the process's rule and cost tables, lowered cost against freshness-only allocation: in an earlier experiment by less than the pre-registered margin and with more failed actions, and in a second experiment with complete case follow-up and frozen models by at least the smallest worthwhile saving, without lowering completion. Every allocation in the second experiment uses the same validity scores, so it is evidence about how to weight checks, not about the validity model. A third experiment changed only the scorer under that allocation: the hazard model lowered cost against per-class Poisson, by less than the smallest worthwhile saving, without lowering completion; with freshness-only allocation its reported saving was larger. *What did*

not hold. Dependency signals missed their pre-registered effect size; calibration met its bar on Wikidata but not at the longest simulator horizon; the conformal lower bounds were mostly trivial and, post hoc (C5a), carry no guarantee; and for values that can change and come back, the survival target is not the probability that the stored value is correct (post hoc, C5a).

Contributions.

1. a task definition for per-fact validity under interval-censored verification, scored by stale facts found per check (or per cost unit) at an equal budget, with calibration and ranking measures (Section 3);

2. LIV-Bench: the `tacitdrift` simulator with a model-blind ledger, its sweep design, and a task built from Wikidata statement lifespans (Sections 3.1 and 5);

3. a reference model and verification policy, compared at an equal budget with established change-rate, survival and decay baselines (Section 4);

4. a pre-registered evaluation that reports negative results beside positive ones, with three disclosed amendments: the second a policy experiment with complete case follow-up, the third a comparison of scorers on downstream cost (Section 6); the audit trail of decisions, corrections and post-hoc analyses is in the appendices.

2 Related work

Change-rate estimation and refresh scheduling. Cho and Garcia-Molina model each element as changing by a Poisson process and give estimators for regular and irregular access when only the existence of a change between accesses is observed [6]. For irregular intervals the estimator is the maximum likelihood solution of $\Sigma_i\, t_{ci} / (e^{\lambda\, t_{ci}} - 1) = \Sigma_j\, t_{uj}$ over the intervals with (t_{ci}) and without (t_{uj}) a detected change [6]. The paper notes the singularity when every access detects a change and assumes a stationary rate. Its companion defines freshness and age, compares uniform, proportional and optimal refresh allocation, and weights freshness by element importance [5]. That importance weighting is a direct precursor of the consequence weights of E3; our weight is the cost of acting on a wrong value in a specific downstream action, not an access rate. We use the per-class Poisson estimator as the main baseline because it is the established method for exactly our observation model. Later crawl-scheduling work allocates a crawl budget across sources by importance and change rate, with the LambdaCrawl family of policies, and learns per-source rates with Cho and Garcia-Molina's estimator inside model-based reinforcement learning, without conditioning rates on features [12, 13]; we reimplement LambdaCrawl allocation as an additional baseline.

Decay curves and validity of stored values. Li et al. define disagreement decay, the probability that an entity changes an attribute value within a time gap, learned from value life spans with censored spans handled explicitly [14]. Per-attribute validity curves are therefore not new; what we add is a per-fact, covariate- and event-conditioned curve with calibration as a target. Data-quality research defines the currency of a value as the conditional probability that its shelf life exceeds its age given other attributes of the record, with distributions taken from external statistics or historical data rather than fitted to censored verification logs [11]. Knowledge-base work pairs a classifier of the likelihood that a fact is outdated (with features such as update frequency and existence time) with human verification of high-likelihood facts [10]. Half-life regression fits a feature-conditioned exponential decay to recall outcomes and uses it to schedule reviews [20]; we reimplement a model of that form as a baseline. Razniewski poses the choice of an update frequency for decaying information as a trade-off between the benefit of the information, the cost of refreshing and the speed of decay [18]; we have read only its abstract, and the comparison with its method is left for the next draft.

Stability of Wikidata facts. Dikeoulias et al. predict whether a property of an entity stays unchanged over one year from two annual Wikidata dumps, with property-presence and object-value features, reporting precision, recall and F-score of the change class, and find that most knowledge-base changes are late completions rather than real-world change [8]. A follow-up adds Wikipedia text and embeddings on a balanced task [21]. Neither reports calibration or models time to change under censoring. Their completion finding is a direct caveat for our Wikidata task (Section 8).

"When Facts Expire". Soulard et al. release a Wikidata-derived benchmark for temporal validity in knowledge graphs [22]. We could not open the CIKM paper itself; from the authors' dataset record, repository and thesis chapter, the task is binary: given a quadruple (subject, property, object, time), decide whether the fact is plausible in that temporal context, with negatives made by corrupting the time into gaps of the subject and property timeline. It measures something adjacent to our task. It validates the asserted time interval of a historical fact with symmetric accuracy; it does not ask

whether a stored fact is still true now, has no re-verification events arriving over time and no check budget, and has no cost of acting on a stale value. LIV-Bench asks the second kind of question.

STALE and agent memory. STALE tests whether LLM agents recognise, from new evidence, that a stored memory is no longer valid, over conflict scenarios that include propagated (cascading) conflicts and an implicit policy adaptation dimension [4]. The best evaluated baseline model reaches 55.2 percent mean accuracy across six settings, judged by an LLM, as reported by its authors [4]. STALE is about detecting staleness once evidence is in the context; LIV-Bench is about predicting it before any evidence arrives and deciding what to check, scored per check at equal budget. The two are complementary: a validity estimate can decide which memories to re-verify, and a STALE-style reader decides what to do once the new evidence is present. Zep keeps a bi-temporal record of when facts held and when the system learned them, and invalidates an edge when a new, contradicting edge arrives [17]; it estimates no probability of validity and schedules no re-verification. Other recent agent-memory systems assign each memory a type-conditioned perishability with an LLM and use it in retrieval scoring, without a fitted survival model or a re-verification budget [2].

ERRAND. The closest work is ERRAND [24]. Agents with a frozen policy consult a store whose items were true at handover; each item flips as a two-state Markov chain, a staleness belief relaxes in closed form between receipts, free en-route receipts update the beliefs, and a priced re-validation is funded only when its value per action clears a running wage under an action budget. Its value of information is $\min(q\,l,\ (1-q)\,g)$, with q the staleness belief and l the misuse loss of serving a stale item, scaled by a usage rate and a durability horizon. It is evaluated in two synthetic drifting tool-use worlds with a preregistered analysis, and it reports that a policy with cadences set from published half-life priors collapses when drift decouples from the calendar. Budgeted, consequence-weighted re-verification of agent memory is therefore not ours. What LIV adds, and what ERRAND does not contain as far as we could verify: a staleness probability learned by hazard regression on fact covariates and signals from interval-censored check histories, rather than per-item Markov intensities; calibration of that probability as an evaluation target; dependency edges between facts as co-change signals, where ERRAND declares items independent and stress-tests correlated flips; and a task from real change histories. Our results qualify two of these: dependency signals missed their pre-registered size (H2), and on the real change histories no improvement over simple baselines was detected while several survival and decay baselines did better (H1, E1 and E1b). Our action-cost evaluations are simulator-only: E3 did not reach its pre-registered size, the redesigned experiment (H4r) did, the scorer comparison (H4s) was supported below the worthwhile saving, and ERRAND's policy is not among our baselines; comparing against it is future work.

3 Task, benchmark and metrics

Unit and observations. A fact is (entity, attribute, value, last verified at, verification method). A check at time c reveals only whether the stored value still equals the true value. For a value that never returns to an earlier state, a check that finds it equal also shows that it has not changed, so a change is known only to lie in the interval $(L, R]$ between the last check that found the value unchanged and the check that found the change (interval censoring). A value that can return (in the simulator, the income band) can change and come back between two checks; the check then reports no change, and L need not lie before the earliest change. A fact is observed only through the intervals its syncs close: after its last check it is unobserved, not known to be unchanged, so no right-censored interval is added at the end of a window. The model outputs $S(t)$, the probability that the stored value is unchanged t days after its last verification, and from it P(valid now) and P(valid at $t{+}h$) for a horizon h. For values that never return, $S(t)$ is the probability that the stored value is still correct. For a value that can return, survival to the earliest change is not that probability: it counts a value that changed and came back as invalid, while the value is again correct; and since the model's training labels come from checks of equality, the fitted model estimates neither quantity exactly for such values.

Policies and budget. Time runs in weekly decision epochs. In each epoch a policy spends a fixed budget of check cost (equal for all policies), carrying any unspent remainder to the next epoch; checks of different attributes may cost different amounts. Every policy spends 5 percent of its budget on uniformly random checks, which keeps calibration estimable.

Metrics. The headline metric is *stale found per check*: checks that found a changed value divided by checks spent over the test period. When check costs differ (the simulator), the denominator is cost units (*stale found per cost unit*) and the unweighted version is also reported (D27). The *relative gain* of policy A over B is $(m_A - m_B)/m_B$. Freshness and age follow Cho and Garcia-Molina's definitions, computed from ground truth and time-averaged over the test period [5]. Concordance is Harrell's C on time to change, with risk $1 - P$(valid at $t_0 + h$). Calibration is the expected calibration error (ECE) of P(valid at $t_0 + h$) with 15 equal-mass bins, and the Brier score at each horizon with inverse probability of censoring weights; the integrated Brier score is compared with Kaplan-Meier per class. In the simulator, calibration is scored against whether the stored value equals the true value (a value that changed and returned counts as valid, as a check would see it). Conformal coverage is the share of test facts whose time to change is at least the predicted lower bound; in the simulator the time to change is the true time of the earliest change.

Intervals. Every headline number is pooled over 5 seeds with a 95 percent percentile interval from a paired cluster bootstrap (2000 resamples): the two policies are compared on the same seeds and worlds, and the resampling units are (seed, entity) clusters in the simulator and subjects (with all seeds of a subject kept together) on Wikidata, where the lifespans are the same for every seed. All splits are temporal; settings are chosen on a validation window (by validation log loss in E1 and E1b; for E2 see the correction in Section 5) and the test window is run once with frozen settings.

3.1 The simulator and the benchmark

The simulator encodes assumptions, not facts. Every parameter of `tacitdrift` is a scenario setting chosen to give a usable dynamic range on a laptop. None of its rates is a claim about real change rates, and none of the simulator results below should be read as an estimate of how often real enterprise facts change. The settings are listed in the repository (simulator specification and D21).

World. 5000 retail-customer entities, each with five attributes (employer, income band, address, phone, e-mail), give 25,000 facts. Covariates are segment, tenure, channel and product count; a latent gamma frailty, never visible to models, carries heterogeneity that the covariates do not explain; it multiplies the rates of all attributes of an entity, so attributes are dependent through it even without triggered co-changes. Employer, address, phone and e-mail values never repeat; the income band moves one band up or down at each change between its lowest and highest band, so it can return to an earlier value. Each (entity, attribute) changes by a process with hazard

$$\lambda_{ia}(d) = b_a \; e^{\beta_a^\top x_i} \; z_i^{\gamma_a} \; \kappa_a \; (d / s_a)^{\kappa_a - 1},$$

with d the time since the value was set and κ the duration dependence. A dependency graph (employer to income band, employer to address, address to phone) adds triggered co-changes: when a parent changes, the child changes with a configured probability after an exponential delay.

Ledger and signals. The organisation sees only a ledger of stored values. Checking a fact is the only way to learn its true value and costs budget. Optional signals are customer-initiated updates (free, after a delay) and activity signals with configured recall and precision that carry no value. Models receive a read-only ledger view; tests check that no reference to the world is reachable from it and that no signal is visible before its emission time.

Phases. A warm-up period up to day 364 uses a shared uniform-random logging policy to produce unbiased interval-censored training labels; validation runs to day 546 and the test period to day 1092, with each policy on its own copy of the ledger and refits every 91 days.

Sweep. The core grid crosses heterogeneity (homogeneous, moderate, strong; the frailty variance is zero in the homogeneous cells), dependencies (off, on; “off” sets the triggered co-change probabilities to zero and keeps the shared frailty) and signals (off, on, strong) at $\kappa = 1$ and a budget of 1 percent of facts per week. One-axis sweeps from the central cell (moderate heterogeneity, dependencies and signals on) vary κ (0.7, 1.5) and the budget (0.5, 2 and 5 percent), and the negative-control corner (homogeneous, no dependencies, no signals) is run at every κ. In total 25 cells times 5 seeds give 125 runs. The design is meant to show both where the hazard model should win and where it should not: in the control corner per-class Poisson is correctly specified.

LIV-Bench. The benchmark consists of the simulator, the harness (policies under an equal budget, metrics, paired bootstrap), the Wikidata task of Section 5, and the reference model. A versioned `tacitdrift` package (version 0.2.0, contract version 0.2), its documentation, the reference model behind a command-line entry point and a model card that lists limits are prepared locally (decision D42). Nothing has been published, so this draft does not claim a release.

4 The model

Discrete-time hazard. Each fact's history is expanded into person-period rows on a grid of time since the value was recorded, with a bin width chosen on validation. LightGBM is fitted to the per-period change indicator h_k, and

$$S(t) = \Pi_{k \le k(t)} (1 - h_k).$$

Training units are the check intervals $(v, c]$ between consecutive synchronisations of a fact (onboarding, check, customer update); chained intervals of one stored value give the same likelihood as the whole episode with left truncation at each confirmation. The duration origin is the onboarding or the last sync that found a new value, so a value that changed and returned between checks keeps its old origin although its true duration restarted. Customer updates close intervals too and arrive after changes, so the observation process is partly informative; we have no observation-model argument that every observed interval behaves like standard interval censoring (Appendix G).

Interval censoring. An interval with a change contributes $S(L) - S(R)$, which spans several person-period rows. We fit it by expectation maximisation over the candidate periods of each interval, with a LightGBM refit as the M-step (a generalised EM, since a boosted fit does not maximise the weighted likelihood exactly); zero EM iterations is the uniform spread of the event over the interval, and validation chooses between them (D3). When every fact is checked on one shared grid, the hazard shape inside a check interval is not identified (D4), so all experiments draw check times with irregular per-fact offsets.

Features. Fact properties (attribute class, verification method, period index, count of earlier detected changes), entity covariates (segment, tenure, channel, product count in the simulator; property and a coarse entity class on Wikidata), co-change signals (whether a parent attribute in the dependency graph was observed to change since this fact's last verification, and how long ago), and activity signals (counts and recency per signal type). Features of a training interval are evaluated at its closing time with information strictly before it, the same way as at prediction time; a forecast freezes features at the forecast time. No sensitive attribute is a feature; the exact list is in the repository's `docs/features.md`.

Calibration. For each horizon, a recalibration map (none or isotonic) is chosen on validation by out-of-fold Brier score and applied on the test period.

Conformal lower bound. We also report a one-sided split-conformal lower bound on the time to change, at $\alpha = 0.1$. Post hoc (C5a), its reduction from the last unchanged check to the time to change fails for values that can return, and its score function was tuned on the data that calibrate it, so we claim no finite-sample guarantee and report its coverage as an empirical observation; Appendix F gives the construction.

Verification policy. In each epoch the greedy policy ranks facts by $P(\text{stale}) \cdot w / c$, with c the check cost and $w = 1$ for the freshness-only policy evaluated here, and checks from the top until the budget is spent. The consequence-weighted policy (w the expected cost of acting on a wrong value) is evaluated in Section 6.3.

Baselines. The main baseline is the per-class Poisson rate estimated with Cho and Garcia-Molina's irregular-interval maximum likelihood estimator [6], used greedily by $P(\text{stale}) = 1 - e^{-\lambda t}$. Its handling when no interval or every interval detected a change is our own choice (the paper states only the second case). Further baselines: a pooled-by-type Poisson rate (per attribute and covariate stratum, falling back to the class rate below 20 changes); a fixed calendar schedule (round robin on time since verification); uniform random checks; an age-threshold rule; Kaplan-Meier per class; an interval-censored Weibull model and an interval-censored piecewise Cox model (both via `lifelines`), with midpoint-imputed Cox as a labelled approximation; our reimplementation of a half-life-regression-style model; and our reimplementation of LambdaCrawl allocation. Each model-based baseline is run through the same greedy policy. The pre-registered H1 anchor is the better of per-class Poisson and calendar in each setting (D27).

5 Experimental setup

Pre-registration. Hypotheses, metrics and thresholds were approved by the human before any experiment ran and are frozen (Table 5 in Appendix A). Evaluation rules that the hypotheses left open were fixed in decision D27 before any pooled number was computed; one raw metrics file of one simulator run had been opened while checking the result schema, and this is disclosed in D27.

E1: Wikidata statement lifespans. Wikidata is public data about mostly notable entities and reflects editing behaviour; it is not enterprise data. We use corporate properties that resemble KYC facts: chief executive officer (P169), chairperson (P488), headquarters location (P159), owned by (P127, restricted to business-type subjects) and director or manager (P1037), statements with a start time, month or day precision for the headline run, deprecated rank and human subjects dropped. The data come from the Wikidata Query Service on the snapshot date 2026-10-06. One fact is one statement, valid from its start to its end; a check finds a change once the end has passed. The lifespans are retrospective: start and end come from qualifiers in the snapshot, which record when a statement held, not when an organisation or Wikidata learned it, and the simulated organisation records each statement at its start. A simulated logging process checks each held statement with probability 1 percent per week at a uniform time inside the week, which gives irregular, interval-censored labels. Models are fitted on what is observable before the landmark 2014-01-01, validated on outcomes up to 2018-01-01, refitted once at 2018-01-01 with frozen settings, and tested from 2018-01-01 to the snapshot (457 weekly epochs, a pool of 19,833 statements, 8153 held at the start). Policy budgets are 0.5, 1 (primary) and 2 percent of alive statements per week (computed from the end dates, which an organisation would not know; Appendix G).

E1b: amendment A1. After the E1 test result (no gain), the human approved one new, separately labelled test, added as a dated amendment: as-of co-change features (whether another tracked property of the same subject started or ended within recent windows, and how long ago), tested once on two held-out properties never used before, board member (P3320) and parent organization (P749). Amendment A1 does not change H1 or its verdict.

E2: the simulator sweep. As in Section 3.1. The H1 metric in E2 is stale found per cost unit because attribute check costs differ (D27).

E2 tuning criterion (correction). An array-shape error made the E2 hazard settings be chosen by a criterion that ignores which prediction belongs to which validation check (D37, D39). The settings were still chosen on validation data only and fixed before the test window; the pre-registered E2 results are kept as run, and Appendix E describes the error and a post-hoc re-run of the central cell.

Not run. The LLM judgement baseline (H5) was deferred by the human: no provider, model or spend cap was chosen, so H5 is withdrawn and reported as not run.

6 Results

Table 1. Every pre-registered hypothesis and its verdict.

Hypothesis	Pre-registered verdict	Ledger
H1 overall (E1 and E2)	neither success nor kill	C1, untested as a whole
H1, E2 part	success (8 of 8 cells)	C1a, supported
H1, E1 part	not supported (CI includes zero)	C1b, refuted as a gain claim
H1-E1b (amendment A1)	neither (CI includes zero)	C1c, not supported
E1 and E1b together	no detected improvement over the better anchor	C1d, supported (null finding)
E1, other baselines (reported)	4 model-based baselines found more stale facts per check	C1e, supported (reported comparison)
H1c (negative control)	supported	C2, supported
H2 (dependency signals)	neither (0 of 9 cells)	C3, refuted at the pre-registered size
H3a (calibration)	neither: E1 met, E2 missed at the longest horizon	C4, partly supported
H3b (conformal bound)	neither: coverage above nominal, gap above the bar; no guarantee for returning values	C5, not supported
H4 (consequence weights)	neither: cost reduction with CI above zero but below the bar; more failed actions	C6 and C6a, refuted; C6b and C6e, supported (smaller reduction; more failed actions)
H4r (amendment A2)	supported: saving at least X, completion not lower by more than m	C9, supported
H4s (amendment A3)	supported: saving interval above zero, upper bound below X; completion not lower by more than m	C10, supported (below the worthwhile saving)
H5 (LLM judgement)	not run (deferred by the human)	C7, untested

6.1 Does a learned validity score find more stale facts per check?

In the simulator (E2). In all 8 scoped cells the hazard policy finds more stale facts per cost unit than the better of per-class Poisson and calendar, with gains from 18.8 to 71.5 percent and every interval above zero (Table 2); the best of the two anchors was per-class Poisson in every scoped cell. The success rule holds in 4 of 4 cells without dependencies and 4 of 4 with them, so H1's E2 part succeeds under the strict reading and under either block alone. Against the additional baselines the smallest gain over LambdaCrawl was 15.0 percent and over pooled-by-type Poisson 17.8 percent.

The unweighted per-check measure, reported alongside the pre-registered one (D27), tells a less uniform story. Against per-class Poisson the hazard policy wins per check in all scoped cells, but calendar finds more stale facts per check than the hazard policy in 2 cells, both with moderate heterogeneity and signals on (right of Table 2). The greedy policy divides P(stale) by check cost, so at an equal cost budget it makes more, cheaper checks: in the central cell 126,569 checks over all seeds

against 97,012 for calendar. Its advantage in these cells is an advantage per unit of cost, which is the pre-registered metric, not per check.

Table 2. H1 in the scoped E2 cells (κ = 1, budget 1 percent). Left: the pre-registered metric, relative gain of the hazard policy over the better of per-class Poisson and calendar in stale found per cost unit (the better anchor was per-class Poisson in every cell). Right: the unweighted per-check gain, reported alongside (D27), against per-class Poisson and against calendar. 95 percent paired cluster bootstrap intervals. Simulated data under stated assumptions.

			Per cost unit, H1 (%)		Per check (%)		
Heterog.	Dep.	Signals	vs better anchor		vs Poisson	vs calendar	
moderate	off	on	18.8	[17.8, 19.9]	24.9	−8.1	[−9.0, −7.2]
moderate	off	strong	44.5	[43.1, 45.8]	66.5	22.6	[21.4, 23.7]
moderate	on	on	23.0	[21.9, 24.0]	25.7	−5.1	[−6.0, −4.2]
moderate	on	strong	48.9	[47.5, 50.2]	66.5	25.8	[24.7, 26.9]
strong	off	on	39.8	[38.1, 41.6]	48.1	11.3	[9.9, 12.6]
strong	off	strong	65.2	[63.1, 67.3]	89.5	42.3	[40.6, 44.1]
strong	on	on	43.2	[41.5, 44.9]	47.1	13.7	[12.3, 15.0]
strong	on	strong	71.5	[69.2, 73.6]	90.9	47.5	[45.6, 49.3]

FIG. 1 · GAIN BY SIMULATOR CELL

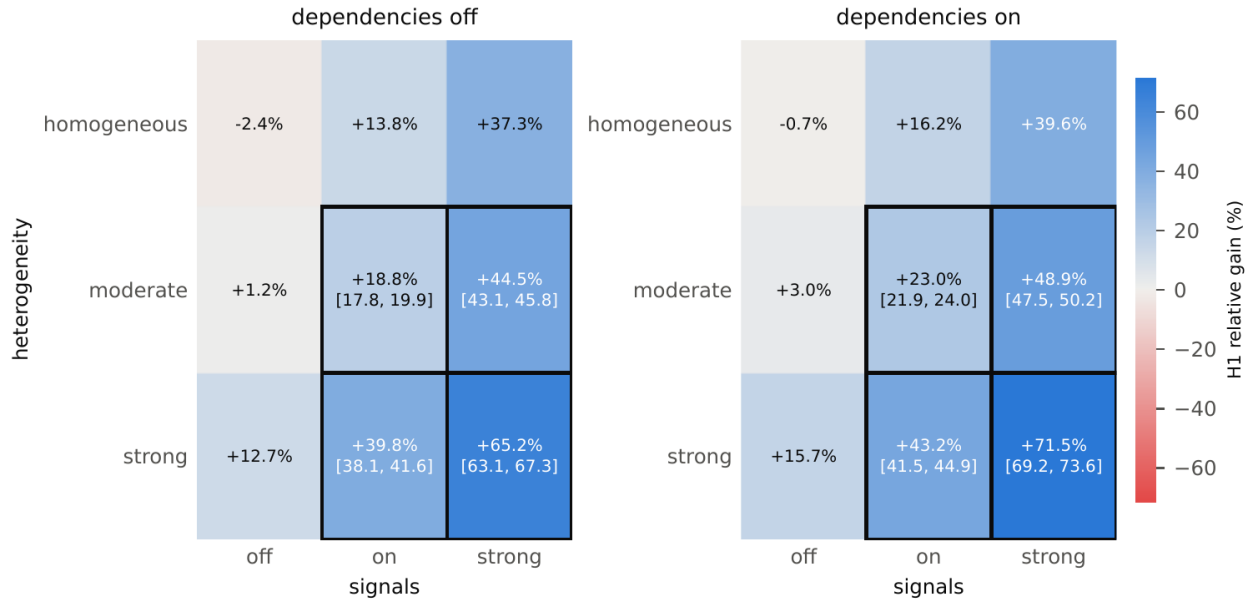


Figure 1. Where the hazard model wins and where it does not (E2 core grid, κ = 1, budget 1 percent). Relative gain in stale found per cost unit over the better of per-class Poisson and calendar, pooled over seeds. Outlined cells are the pre-registered scope and carry their 95 percent intervals; the other cells are point estimates outside the pre-registered scope.

Where it does not win. Figure 1 shows the full core grid. Without signals, the gain is near zero under homogeneous and moderate heterogeneity (point estimates from −2.4 to 3.0 percent), which is where per-class Poisson captures most of the structure. Two patterns outside the pre-registered scope are point estimates only and exploratory: with strong heterogeneity and no signals the hazard policy still gains (12.7 and 15.7 percent), and with homogeneous rates but signals present it gains from 13.8

to 39.6 percent. In this simulator, signals are the larger source of the advantage; heterogeneity alone gives less.

One-axis sweeps over the budget and the duration dependence, freshness and age, and the full policy table of the central cell are in Appendix E.

The negative control (H1c). In the homogeneous, memoryless cell without dependencies or signals, where per-class Poisson is correctly specified, the hazard policy's gain over per-class Poisson is −2.4% [−2.9, −1.9]: below the 5 percent bar, so H1c is supported. The harness shows no sign of leakage in this cell, and the hazard model pays a small price for its flexibility where there is nothing to learn.

On Wikidata (E1). At the primary budget no difference between the hazard policy and the better anchor (calendar) is detected: relative gain −0.3% [−2.1, 1.7]. The interval includes zero and small differences in either direction; no equivalence margin was pre-specified, so this is not evidence that the two perform the same. At 0.5 percent it is worse than calendar (−3.5% [−5.6, −1.3]), and at 2 percent no difference from calendar is detected (−0.3% [−1.9, 1.5]). H1's E1 part is not supported. Table 6 (Appendix B) compares all policies at the primary budget: the interval-censored Weibull and Cox models, the midpoint Cox model and the half-life-regression-style model find more stale facts per check than the hazard model, by small margins with intervals below zero (ledger C1e). The registered anchors are the weaker comparison on this task: 4 of the evaluated model-based baselines did better than the hazard model; two sensitivity analyses do not change this (Appendix B).

Co-change features on held-out properties (E1b, amendment A1). On the held-out properties at the primary budget, the hazard policy with co-change features gains −3.3% [−8.9, 2.8] over per-class Poisson: neither success nor kill, so H1-E1b is not supported. No improvement is detected, and the interval includes a disadvantage as large as −8.9 percent. The survival and decay baselines of Table 6 were not run on the held-out properties. Concordance at the longest horizon is 0.605 with co-change features, 0.595 without and 0.536 for per-class Poisson (no interval computed). Results at the other budgets and against its own ablation are in Appendix B.

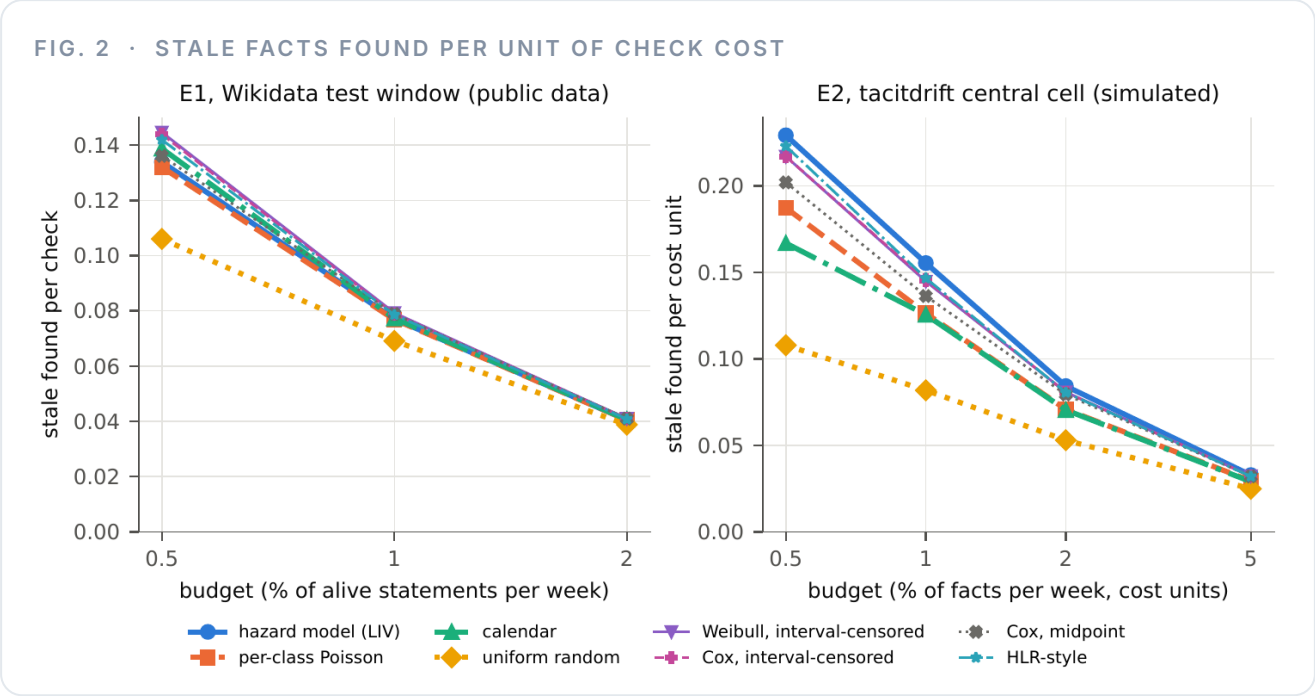


Figure 2. Stale facts found per check (E1, public Wikidata data) and per cost unit (E2 central cell, simulated) against the check budget, for the hazard policy, per-class Poisson, calendar and uniform random, and (thin lines) the 4 model-based baselines that found more stale facts per check than the hazard policy on E1 at the primary budget: interval-censored Weibull and Cox, midpoint Cox and the HLR-style model. Means over 5 seeds (E1) and pooled over seeds (E2).

Figure 2 summarises the contrast between the two settings. On Wikidata the hazard, per-class Poisson and calendar policies are close at every budget, and the survival and decay baselines of Table 6 lie slightly above the hazard policy at the primary budget; in the simulator the hazard policy stays above both anchors at every budget, and the gap narrows as the budget grows.

6.2 Dependency signals, calibration and bounds

Dependency signals (H2). Removing the two co-change features lowers stale found per cost unit on the child attributes (income band, address, phone) by at most 2.9 percent in the scoped cells, short of the 5 percent bar in all of them: success in 0 of 9 cells (Table 13, Appendix E). H2 is not killed, because 4 cells have intervals above zero. Post hoc (C3a), the dependencies-off cells registered as a leakage control keep the shared frailty, so they cannot separate leakage from information (Appendix E). The ablation does not support a claim that dependency signals help at the pre-registered size.

Calibration (H3a). On Wikidata the hazard model's P(valid) is calibrated on the later period at every horizon, with no recalibration chosen on validation (Table 11, Appendix E): the ECE meets the 0.05 bar. In the simulator central cell the recalibrated ECE meets the 0.03 bar at 30, 90 and 180 days but not at 365 days (0.049), without reaching the 0.08 kill bar; one seed reaches 0.081 there. The 365-day horizon has no validation pairs because validation spans a shorter window, so its recalibration map is the pooled one; and forecasts freeze the signal state over the horizon, which matters most at long horizons. Without recalibration the simulator ECE is several times larger. H3a's verdict is neither. The integrated Brier score on Wikidata is 0.075 for the hazard model against 0.088 for Kaplan-Meier per class.

Conformal lower bounds (H3b). Observed coverage is above the nominal level and most bounds are trivial. In the simulator central cell the coverage of the true change time is 0.991 against the nominal 0.9 (gap 0.091, bar 0.02), and 58 percent of the bounds are trivial; the median gap between the true change time and the bound is 484 days. On Wikidata the pre-registered coverage of L is 0.943 (gap 0.043, bar 0.03), coverage of the change time is 0.998, and 91 percent of the bounds are trivial, with a median gap of 2785 days. A conservative bound fails a two-sided gap rule; that is what happened. These coverages are empirical observations, not a verified guarantee: in the simulator the reduction from L to T fails for the income band, and in both settings the bound is calibrated on the validation data its model settings were chosen on (Appendix F; post hoc, ledger C5a). The committed E2 outputs store pooled coverage only, so coverage on the token attributes alone, where the reduction holds, cannot be computed without re-running E2, which we have not done. In their current form the bounds are not useful for most facts.

6.3 Does weighting checks by consequence lower cost?

The process. The three experiments in this subsection join the drifting facts of `tacitdrift` to a minimal loan-intake process, standalone and inspired by SimBank [7] (decision D35). Each entity opens a case about once a year; a scripted agent submits using the stored employer and income band and, after a successful submit, approves and sends the contract using the stored address, phone and email. Outcomes follow deterministic rules over the stored and the true values (rejection, rework, contact rework, deadline breach, and a share replaced by noise), priced by a stated cost table; some rejections are caused by the true income band, and no check can prevent them. Appendix C gives the rules and the cost table; all of them are scenario settings, not claims about real lending. Policies rank facts by $P(\text{stale}) \cdot w / c$ with the hazard model of Section 4 and differ in the weight w: `fresh_global` ($w = 1$), `fresh_case` (the fact's exposure to pending actions of open cases), rule-table weights (the expected cost of acting on a wrong value, from the rule and cost tables, one wrong fact at a time), and, in the second experiment, `conseq_avoid` (below).

6.3.1 The E3 experiment (H4)

Against the better freshness-only arm (`fresh_case`), the rule-table-weighted arm lowered the cost accrued in the test window per case opened in it by 6.8% [6.2, 7.3]: the interval is above zero but below the pre-registered 10 percent, so H4 is neither supported nor killed (C6, C6b). It raised failed actions per case from 0.369 to 0.410 (C6e), trading fewer rejected submits for more contract reworks, and weights from the learned CASS gate did not lower cost in the primary cell (−2.0% [−2.7, −1.3]; C6f), a handicapped test (Appendix G). The measure stopped at the end of the test window, so the cases opened last were not followed to their outcome. Appendix C gives the full results and the post-hoc analyses, which decide nothing.

6.3.2 The redesigned experiment (Phase 2b-r, H4r)

Why a second experiment. E3 left H4 at "neither", and two external reviews then found that its measure stopped following cases at the end of the window, that it mixed immediate and full-path effects, and that a follow-up design we had drafted would have set its bars after a pilot. We wrote a redesign and registered it as amendment A2 before any pilot ran: one new deciding hypothesis, H4r, tested once on fresh seeds in a harness shared with the CASS paper. H4, its verdict and every E3 number are unchanged; H4r is new evidence, not a re-test of H4. The design was written after the Phase 2 results and the reviews were known, which we state as a limit.

Design. The scenario, cost table and primary cell are those of E3 (moderate heterogeneity, dependencies and signals on, a background budget of 1 percent of total fact check cost per week), with 5000 entities per world. Before the test, the organisation logs uniformly random background checks and a randomised pre-action slice that checks an action's used facts right before it executes with probability $q = 20$ percent. After a burn-in of $b = 2184$ days, the scorer is fitted on the next 364 days, its settings are selected on the following 91 days, it is refitted with the chosen settings on both windows, and its calibration is chosen on the 91 days after that; then everything is frozen, so unlike E3 the scorer is not refitted during the test. Every arm runs on its own fork of the shared warm-up with common random numbers. The cohort is every case opened in the 546-day enrolment window (days 2730 to 3276), and every arm keeps running 30 days longer, so every enrolled case reaches its terminal event. The primary outcome is total realised cost per enrolled case: the cost of every realised outcome of every action of the case, every check a decision policy spent on it, and the background check spend of the enrolment window divided by the number of enrolled cases (5.23 units in every arm). The second outcome is completion, the share of enrolled cases that end with a successful contract. As in E3, 5 percent of each arm's background budget is spent uniformly at random.

H4r compares two background allocations, both without a decision gate: `conseq_avoid` against `fresh_case`. `conseq_avoid` is a labelled structural heuristic. It values a check on a fact by the expected immediate outcome cost that its pending uses would save if the fact were made fresh with the other used facts left at their stored values, each use weighted by the probability that the case reaches it, per unit of check cost, computed from the rule and cost tables, the stored values and the frozen scorer's P(stale), with staleness states enumerated under an independence approximation and with the rejections that no check can prevent priced as unavoidable. It reads no true value. It does not estimate the marginal value of checking a fact earlier, so H4r compares two policies, not the causal value of their weights. The comparator `fresh_case` uses the same stored values, the same P(stale) and the same open-case schedule, but not the rule and cost tables: it weights every pending use equally. It was fixed before any pilot; `rule_table`, reported, is the allocation with the same structural knowledge as `conseq_avoid`. Because every allocation in this experiment ranks facts by the same frozen P(stale), no contrast here measures the value of the validity model itself; E1 and E2 address that question on stale facts found per check, and Phase 2c (Section 6.3.4) on cost. Reported cells, without verdicts, are `fresh_global` and `rule_table` (E3's rule-table weights) without a gate, the CASS decision policies under both allocations (always check, a risk rule tuned on the calibration window and an uplift rule), an age rule under `fresh_case` only, 11 primary cells in all, and a transfer condition with 5 arms whose warm-up logging is a calendar review round robin with a slice of 5 percent.

The seed is the unit of inference. The deciding interval is the two-sided 95 percent Student t interval over 10 seeds (200 to 209) of paired per-seed differences. H4r is supported if the lower bound of the saving is above zero and the lower bound of the completion difference is above $-m$; the smallest worthwhile saving $X = 1$ cost unit per enrolled case and the margin $m = 1$ percentage point were fixed before any pilot, and the words "at least X" are permitted only if the saving's lower bound is at least X. An exact sign-flip test over seeds and a (seed, entity) cluster bootstrap are reported beside the t interval, which decides; relative reductions carry seed-bootstrap intervals (2000 resamples).

Pilot decisions. A blinded pilot on 8 seeds that are never confirmatory recorded no arm's cost or effect. Four steps departed from the registered procedure: the burn-in rule was amended after the pilot stopped on it, the slice rate q was set by a human override of the support rule, the seed count fol-

lowed from moving the CASS hypothesis to reported, and a drift check that could not be computed before the test without the test window was replaced after the test by the registered logging-regime check (D70). Appendix A gives each step, who decided it and when. No pilot effect informed any setting, and no truth informed the arms, features, hyperparameters, comparator, X, m or the readings. The test ran once (D69).

Table 3. Phase 2b-r primary condition: total realised cost per enrolled case (cost-table units, mean over 10 seeds with a per-cell 95 percent t interval) and completion. Cells share seeds and cohorts, so differences between cells are judged by the paired contrasts in the text, not by these per-cell intervals. Simulated data.

Allocation	Decision policy	Cost per enrolled case	Completion (%)
`fresh_case`	none (H4r comparator)	35.5 [34.7, 36.2]	69.8
`conseq_avoid`	none (H4r)	33.9 [33.1, 34.7]	71.6
`rule_table`	none	34.3 [33.5, 35.2]	71.2
`fresh_global`	none	44.0 [43.2, 44.7]	62.3
`fresh_case`	age rule	36.5 [35.0, 38.0]	71.4
`fresh_case`	risk rule	37.2 [36.4, 37.9]	71.3
`fresh_case`	uplift rule	37.2 [35.6, 38.9]	71.5
`fresh_case`	always check	41.0 [40.2, 41.8]	72.8
`conseq_avoid`	risk rule	36.4 [35.5, 37.4]	72.2
`conseq_avoid`	uplift rule	36.5 [34.6, 38.4]	72.2
`conseq_avoid`	always check	41.0 [40.2, 41.8]	72.8

H4r. Against `fresh_case`, `conseq_avoid` lowered total realised cost per enrolled case by 1.55 cost units (95 percent interval 1.33 to 1.77) and raised completion by 1.7 percentage points (1.5 to 2.0). The lower bound of the saving is above $X = 1$, so the registered reading applies: `conseq_avoid` lowered total cost per enrolled case by at least 1 cost unit, with completion not lower by more than 1 percentage point (completion was in fact higher). H4r is supported (ledger C9). The robustness checks agree: the one-sided exact sign-flip test gives $p = 0.001$, and the cluster bootstrap gives a saving of 1.55 units (1.41 to 1.69). In relative terms the saving is 4.4% [3.8, 4.9] of total cost per case and 14.4% [12.9, 15.8] of the cost above the unavoidable initial-submit component, a denominator that includes the background spend (C9a): most of the cost of a case is out of reach of any background check. This is a simulator result under stated assumptions, for one scenario and one primary cell.

Reported cells. In the transfer condition the same contrast saves 1.59 units (1.33 to 1.85) with completion 1.8 points higher (1.5 to 2.1; C9b). `rule_table` also saves against `fresh_case`, 1.12 units (0.93 to 1.32), or 3.2% [2.7, 3.6], while `fresh_global`, which ignores open cases, has a saving of −8.51 units (−8.94 to −8.08), that is, a higher cost (C9c). Under always check the two allocations had the same cost on every seed (difference 0.00, interval 0.00 to 0.00; C9d). Every gated cell has a higher mean cost per case than the same allocation without a gate (Table 3, point estimates); the CASS paper reports the paired contrasts. The registered interactions between allocation and decision policy are in Appendix D.

Stationarity and shift. Reported after the test (Appendix D): the logging regime was within the amended drift rule on 10 of 10 seeds in the primary condition (C9e), but the frozen models were de-

ployed where used facts were stale far less often than in the window they were fitted on (C9f). Some registered reported quantities were not computed; Appendix D lists them.

6.3.3 Post hoc: rule-table weights against avoidability-aware weights (D72)

This analysis is post hoc. `conseq_avoid` against `rule_table` was not registered. We chose it after reading the test results, to ask what the avoidability-aware weight adds over the rule-table weight; both rank facts by the same frozen P(stale). It is computed by `scripts/p2r_posthoc.py` from the saved test outputs, through the go/no-go script's own integrity checks, runs no experiment and decides nothing. `conseq_avoid` cost 0.43 units less per enrolled case than `rule_table` (0.34 to 0.51), or 1.2% [1.1, 1.5], with completion 0.4 points higher (0.3 to 0.5); the difference had the same sign on 10 of 10 seeds (two-sided exact sign-flip $p = 0.002$; C9g). As a ratio of point estimates, without an interval, `rule_table` reaches 72 percent of the saving of `conseq_avoid` over `fresh_case`. Most of the H4r saving is therefore available from rule-table weights alone; the other differences in the weight of `conseq_avoid` (avoidability, reach, the exclusion of legitimate rejections and joint staleness) add a smaller part, of the same sign on every seed, whose interval lies below X. This contrast does not separate those differences from one another.

6.3.4 Does the learned scorer lower cost? (Phase 2c, H4s)

Why a third experiment. Every Phase 2b-r allocation ranks facts by the same frozen P(stale), so that experiment says how to weight checks given a validity estimate, not what the estimate is worth in decisions; E1 and E2 compared the hazard model with per-class Poisson on stale facts found per check, not on downstream cost. After the Phase 2b-r results were known we registered amendment A3 with one new deciding hypothesis, H4s, whose contrast changes only the scorer. H4r, its verdict and every Phase 2b-r number are unchanged; H4s does not re-test H4r.

Design. The harness, scenario, primary cell, windows, burn-in, slice rate, outcomes, statistics and readings are those of Phase 2b-r (Section 6.3.2); every arm runs without a decision gate and with the same background check spend. Each arm uses one of three scorers, all fitted on the same refit intervals closed by exogenous checks in the warm-up (LIV's settings selected as in Phase 2b-r), with the same target, and recalibrated on the calibration window by the same rule, so calibration alone does not separate them: LIV's hazard model with its Phase 2b-r settings; per-class Poisson, one constant change rate per attribute (the estimator of Cho and Garcia-Molina, the paper's main baseline, fixed as the comparator before any Phase 2c output existed); and per-type Poisson, one rate per attribute, segment and channel, with the per-class rate where a stratum has fewer than 20 changes, reported as a stronger simple scorer. H4s compares `conseq_avoid` with LIV's scorer against `conseq_avoid` with per-class Poisson. It is supported if the lower bound of the saving is above zero and the lower bound of the completion difference is above $-m$; $X = 1$ cost unit and $m = 1$ percentage point are those of H4r, and "at least X" is again permitted only if the saving's lower bound is at least X. Reported arms, without verdicts, are `conseq_avoid` with per-type Poisson and `fresh_case` with LIV's scorer and with per-class Poisson. A blinded pilot on 4 seeds (300 to 303, pilot seeds of Phase 2b-r, never confirmatory) kept a report that holds only spreads, harness checks and scorer checks, with no arm's cost or effect (the raw per-arm outputs went to an untracked cache that was not read), and no stop rule fired. The registered power rule gave a simulated power of 100 percent at a saving of X with 10 seeds (target 80 percent), the smallest count it allows; the human approved that count and the freeze, and the test ran once on seeds 210 to 219, which no earlier experiment used.

Table 4. Phase 2c, primary condition, no decision gate: total realised cost per enrolled case (cost-table units, mean over 10 seeds with a per-cell 95 percent t interval) and completion. Arms share seeds and cohorts, so differences between them are judged by the paired contrasts in the text, not by these per-cell intervals. Simulated data.

Allocation	Scorer	Cost per enrolled case	Completion (%)
conseq_avoid	LIV hazard model (H4s)	33.65 [33.25, 34.06]	71.9
conseq_avoid	per-class Poisson (H4s comparator)	34.26 [33.85, 34.67]	71.7
conseq_avoid	per-type Poisson	34.25 [33.84, 34.66]	71.7
fresh_case	LIV hazard model	35.33 [34.81, 35.86]	70.0
fresh_case	per-class Poisson	36.93 [36.45, 37.42]	68.7

H4s. Under conseq_avoid, LIV's scorer lowered total realised cost per enrolled case against per-class Poisson by 0.60 cost units (95 percent interval 0.53 to 0.68) and raised completion by 0.23 percentage points (0.16 to 0.31). The lower bound of the saving is above zero and its upper bound is below X, so the registered reading applies: LIV's scorer lowered total cost per enrolled case, by less than the pre-specified worthwhile saving of 1 unit, with completion not lower by more than 1 percentage point. H4s is supported, below the worthwhile saving (ledger C10). The robustness checks agree: the one-sided exact sign-flip test gives $p = 0.001$, the smallest value it can take with 10 seeds, and the cluster bootstrap gives a saving of 0.60 units (0.53 to 0.68). Against a total of about 34.26 units per case (Table 4) the saving is small. This is a simulator result under stated assumptions, for one scenario and one primary cell, in a world whose activity signals and heterogeneous entities give the hazard model information that per-class Poisson cannot use (Section 3.1).

Reported quantities. Under fresh_case the same scorer contrast saves 1.60 units (1.20 to 2.00), or 4.3% [3.5, 5.3], with completion 1.31 points higher (0.89 to 1.73; C10a). The interaction, the conseq_avoid saving over fresh_case with LIV's scorer minus the same saving with per-class Poisson, is 1.00 units (0.61 to 1.38; C10b): the learned scorer is worth less once checks are weighted by consequence, and the consequence weights are worth less on top of the learned scorer. conseq_avoid with per-class Poisson still saves 1.07 units (0.80 to 1.34) against fresh_case with LIV's scorer, with completion 1.63 points higher (1.40 to 1.87), and the H4r contrast on these new seeds is 1.68 units (1.42 to 1.94), reported only (C10c). Because of the interaction, the combined saving cannot be split uniquely between allocation and scorer; what the data do show is that consequence weighting with per-class Poisson cost less than freshness-only allocation with the learned scorer (C10c). No difference was detected between per-type and per-class Poisson (difference −0.01, interval −0.08 to 0.06), and LIV's scorer saved 0.59 units (0.52 to 0.66) against it (C10d). On the uniform test slice of each conseq_avoid arm, LIV's scorer had lower log loss (0.287 against 0.334 for per-class and 0.333 for per-type Poisson) and Brier score (0.084 against 0.098 and 0.098), with similar expected calibration error (0.030, 0.031 and 0.028; C10e); all three were recalibrated by the same rule. As in E3, where it raised failed actions (C6e), consequence weighting lowered cost while raising avoidable failed actions: per enrolled case, 0.087 with LIV's scorer and 0.131 with per-class Poisson under conseq_avoid, against 0.062 and 0.088 under fresh_case (means over seeds, no paired interval; legitimate rejections were 0.225 in every arm; C10g). This is consistent with an allocation that spends checks where a stale fact is costly rather than where failures are most frequent. Computed but not shown here: the sign-flip tests and cluster bootstrap of the reported contrasts, the scorer quality of

the `fresh_case` arms and the other cost components of each arm (all in `results/go-nogo_p2c.json`).

7 Discussion and implications for practice

> *We framed per-fact validity for enterprise agent memory as an interval-censored survival problem scored by stale facts found per unit of check cost at an equal budget, with the per-class Poisson estimator from web crawling as the main baseline.*
>
> LIV · FROM THE CONCLUSION

Why the simulator and Wikidata disagree. In the simulator, the facts carry covariate effects, frailty and activity signals that the hazard model can use and per-class Poisson cannot; in the heatmap's exploratory point estimates, signals contribute more than heterogeneity. On Wikidata the E1 feature set gave the hazard model little beyond property, entity class and age, which the simpler survival models also see, and the interval-censored Weibull and Cox models and the half-life-style model did slightly better. The co-change features of amendment A1 helped against their own ablation at the lowest budget, but did not move the budgeted policy past the anchors on held-out properties. Wikidata also reflects editing behaviour: ends are often recorded late or not at all, which blurs the signal any model can learn. We therefore read the evidence as: in the simulator the hazard model helps when the organisation has informative per-fact signals, and we have not shown such signals in public data.

What Phase 2b-r adds, and what it does not. With complete follow-up, a comparator fixed before any pilot that uses the same validity scores and case schedule but not the rule and cost tables, and bars set before any pilot, an allocation that prices avoidable failures from the rule and cost tables did better than the freshness-only allocation by more than the smallest worthwhile saving (Section 6.3.2). Its size relative to the total cost of a case, a post-hoc comparison with a rule-table-weighted allocation, and the covariate shift between the fitting window and deployment are reported, not tested, in Sections 6.3.2 and 6.3.3. Every allocation in that experiment ranks facts by the same frozen scorer, so it says how to weight checks given a validity estimate; whether the estimate itself beats simpler change-rate models is the question of E1 and E2, where the answer was positive in the simulator and not detected on Wikidata, and of Phase 2c, which asks it on cost.

What Phase 2c adds. In the simulator, with signals the hazard model can use, the learned scorer lowered cost against per-class Poisson under both allocations, under the consequence-weighted allocation by less than the smallest worthwhile saving, and the two savings are not additive (interaction C10b; Section 6.3.4). Consequence weighting with a simple change-rate model did better than freshness-only allocation with the learned scorer (C10c), and the learned scorer added a further saving below the worthwhile size (C10). This is one scenario of one simulator; it suggests, but does not show, that where the rules and costs of the consuming processes are known, encoding them should come before learning a validity model.

What this suggests in practice. The statements below read simulation results under stated assumptions and one public dataset; none has been tested on an organisation's own data.

- *Weight re-verification by the actions that use a fact* (simulation; H4r, C9). Where an organisation holds the rules and costs of the processes that consume its facts, allocating checks by the cost that a

stale fact would cause there, on top of a validity estimate, lowered total cost per case against freshness alone. How much of that comes from each part of the weight is a post-hoc question (Section 6.3.3). With per-class Poisson in place of the learned scorer the allocation still cost less than freshness-only allocation driven by the learned scorer (Phase 2c, C10c).

- *Learn a validity model only where informative per-fact signals exist* (simulation and Wikidata; C1a, C1d, C1e). The hazard model helped in the simulator when entities differed and activity signals carried information (also on cost, by less than the worthwhile saving under consequence weighting; C10, C10a), and gave no detected gain on public change histories, where simpler survival and decay models did better. Without such signals, per-class change rates, a calendar schedule and simple interval-censored survival models are the defaults to beat.
- *Keep review floors and check the target.* A low *P*(stale) can order checks above mandated review floors but must not replace them, and for values that can return to an earlier state the estimate is not the probability that the stored value is correct (Section 8).
- *What would confirm this.* A shadow study on one real workflow that records checks, signals, outcomes and their costs; the repository holds a protocol for it (`protocols/kyc_shadow_study.md`), not executed.

8 Limits

- **Gradual change versus sudden events.** The model learns gradual drift. Sanctions, death and insolvency are sudden, high-consequence events that need evidence-driven checks (registry feeds, screening), not a survival curve. The simulator does not model them.
- **Not a truth oracle.** The model estimates whether a stored value has changed since it was verified. It cannot tell whether the value was correct when it was recorded; in the simulator the value at onboarding is always correct.
- **Rare attributes.** Attributes with few recorded changes give few events, wide intervals and fragile hazard estimates; pooling or priors are needed, as for the Poisson estimator when no or every check detects a change.
- **Drift.** Change rates, signal quality and editing behaviour move over time. A single refit inside a test window of several years (E1) and frozen features over a forecast horizon are both approximations; recalibration and periodic refits are needed in use.
- **Fairness and privacy.** No sensitive attribute is a feature, but covariates can proxy for them and checks can fall unevenly on groups; check burdens and error rates should be audited per segment. On Wikidata, counts of sitelinks and statements partly measure public attention (they were excluded for leakage). Enterprise use needs privacy review of the change and verification logs.
- **Gaming and over-trust.** A low $P(\text{stale})$ must never lower scrutiny for high-risk categories. Mandated review floors, such as fixed re-KYC intervals by risk class, stay in force; the model can only prioritise checks above those floors. Parties who learn which signals trigger checks could avoid them.
- **Simulator assumptions.** `tacitdrift` uses log-linear covariate effects, gamma frailty, Weibull durations and triggered co-changes; a model that matches these forms has an advantage that real data may not give. The sweep includes cells where it has none, but every simulator number is conditional on these assumptions and is not a real-world rate.
- **Public-data bias.** Wikidata describes notable, mostly public organisations and reflects editing behaviour. Most knowledge-base changes are late completions rather than real-world change [8]; statements without an end time may have ended (right-censoring is not clean); only statements with a start time and fine date precision are used. Wikidata is not enterprise data, and there is no enterprise verification process behind it.
- **Values that can return.** The survival target and the conformal reduction assume that a value does not change and come back between checks. The simulator's income band can, so for it survival to the earliest change is not the probability that the stored value is correct, an undetected return keeps the old duration origin, and the premise $L \le T$ fails (post hoc, ledger C5a). The four other simulator attributes and every Wikidata statement never repeat.
- **Phase 2b-r is one scenario, designed late.** H4r was designed after the Phase 2 results and two external reviews were known, and it is tested in one scenario, one primary cell and one transfer condition of the simulator. Simulator truths that the design and its checks use (which failures were avoidable, the unavoidable cost component) cannot be measured in a bank. `conseq_avoid` uses the rule and cost tables and an independence approximation over staleness states; its weights are a heuristic, not causal values.

- **Phase 2c shares that scenario.** H4s was registered after the Phase 2b-r results were known and runs in the same scenario and primary cell, without the transfer condition or a decision gate. Its simulator gives the hazard model signals that per-class Poisson cannot use; with fewer or weaker signals, as on Wikidata, the scorer contrast may vanish. The three scorers share one recalibration rule, and the pilot reused Phase 2b-r pilot seeds.
- **Frozen models under shift.** In Phase 2b-r and Phase 2c every model is frozen before the test and deployed in a regime where used facts were stale much less often than in the window it was fitted on. No refit or recalibration in deployment was tested.
- **Experiment-specific limits.** Limits of the Wikidata task (scope, retrospective lifespans, an oracle budget, optimistic co-change timing), of the observation process (customer updates as inspections), of the conformal calibration and the dependency control, of the E2 tuning correction, of E3 (rule-table weights, incomplete follow-up, a handicapped learned arm), of the Phase 2b-r pilot decisions, and the missing comparison with ERRAND are listed in Appendix G.

9 Reproducibility

All code, configurations and results files are in the repository. Experiments run on a laptop CPU. The simulator is deterministic from a seed, with independent random streams per component. The Wikidata data are fetched from the public Query Service with a descriptive User-Agent and cached locally; the queries are listed verbatim in the data card, and because the service is live, the same queries on a later date may return different rows, so the snapshot used here (the cached query responses and the derived statement and subject files, with the fetch date in its fetch summary) is included in the repository with checksums. Wikidata releases its structured data under CC0. The go/no-go file is produced from the results by a script that reads the frozen hypotheses. Every number in this paper is a macro written by `scripts/paper_numbers.py` from the results files, and every figure is drawn by `scripts/paper_figures.py`; a test checks that regenerating the numbers gives identical output and that the paper source contains no hand-typed result number. Per-check logs are regenerated by the experiment scripts rather than stored. Phase 2b-r runs from one frozen configuration and analysis contract shared with the CASS repository; a manifest records the SHA-256 of every input and output file and both go/no-go scripts refuse any file that does not match it, any unfrozen input and any pilot seed. Its per-case and per-decision files are not stored in the repository because of their size; their hashes are in the manifest. Phase 2c runs from its own frozen configuration and analysis contract under the same manifest checks, and its per-case files and manifest are stored in the repository. The post-hoc contrast of Section 6.3.3 is computed by `scripts/p2r_posthoc.py` from those files, and the post-hoc E3 analyses by the scripts named in Appendix C.1. This version (1.0.0) is archived with its code, configurations, results and the Wikidata snapshot at Zenodo, DOI `10.5281/zenodo.24077472`; the companion CASS release is DOI `10.5281/zenodo.24077491`. One item is open: the `make reproduce` target exists, but its experiment steps are placeholders, so regenerating every number from scratch still means running the experiment scripts one by one.

10 Conclusion

We framed per-fact validity for enterprise agent memory as an interval-censored survival problem scored by stale facts found per unit of check cost at an equal budget, with the per-class Poisson estimator from web crawling as the main baseline. In the `tacitdrift` simulator, under stated assumptions, a discrete-time hazard model with a greedy policy finds between 18.8 and 71.5 percent more stale facts per cost unit than the better of per-class Poisson and a calendar schedule in every pre-registered cell with heterogeneous entities and signals, and does not beat per-class Poisson in the control cell where Poisson is correctly specified (C1a, C2). On public Wikidata data no improvement over the better of per-class Poisson and a calendar schedule was detected (−0.3 percent, interval −2.1 to 1.7), nor with co-change features on held-out properties, and 4 other model-based baselines found more stale facts per check (C1d, C1e). In a simulated loan-intake process, weighting checks by consequence costs from the rule and cost tables lowered test-window cost per case by 6.8 percent in an earlier experiment, below the pre-registered 10 percent and with more failed actions (C6b, C6e). In a second experiment with complete case follow-up and frozen models, an avoidability-aware allocation lowered total cost per enrolled case by 1.55 cost units (interval 1.33 to 1.77), at least the pre-specified 1 unit, without lowering completion, against a freshness-only allocation that uses the same validity scores and case schedule but not the rule and cost tables (H4r supported, C9). That result concerns how checks are weighted given a validity estimate, not the validity model itself. In a third experiment that changed only the scorer under that allocation, the hazard model lowered total cost per enrolled case against per-class Poisson by 0.60 units (interval 0.53 to 0.68), less than the pre-specified 1 unit, without lowering completion (H4s supported below the worthwhile saving, C10). Apart from the Wikidata findings these are simulator results; Section 8 lists what they do not cover.

References

[1] Andreas Bender, David Ruegamer, Fabian Scheipl, and Bernd Bischl. *A general machine learning framework for survival analysis.* In ECML PKDD 2020, 2020. arXiv:2006.15442.

[2] Kartikey Singh Bhandari, Aarya Wadhwani, Dhruv Kumar, and Pratik Narang. *Caching for the future: scrub jay episodic memory principles for agent memory systems, 2026.* arXiv:2608.04746.

[3] Emmanuel J. Candès, Lihua Lei, and Zhimei Ren. *Conformalized survival analysis.* JRSSB, 85(1):24–45, 2023. arXiv:2103.09763.

[4] Hanxiang Chao, Yihan Bai, Rui Sheng, Tianle Li, and Yushi Sun. *STALE: Can LLM agents know when their memories are no longer valid?, 2026.* arXiv:2605.06527.

[5] Junghoo Cho and Hector Garcia-Molina. *Effective page refresh policies for Web crawlers.* ACM Transactions on Database Systems, 28(4):390–426, December 2003.

[6] Junghoo Cho and Hector Garcia-Molina. *Estimating frequency of change.* ACM Transactions on Internet Technology, 3(3):256–290, August 2003.

[7] Jakob De Moor, Hans Weytjens, Johannes De Smedt, and Jochen De Weerdt. *SimBank: from simulation to solution in prescriptive process monitoring, 2025.* arXiv:2506.14772.

[8] Ioannis Dikeoulias, Jannik Strötgen, and Simon Razniewski. *Epitaph or breaking news? analyzing and predicting the stability of knowledge base properties.* In Companion Proceedings of the 2019 World Wide Web Conference (WWW '19 Companion), pages 1155–1158, New York, NY, USA, 2019. ACM.

[9] Financial Action Task Force. *International standards on combating money laundering and the financing of terrorism & proliferation: The FATF recommendations.* `https://www.fatf-gafi.org/content/dam/fatf-gafi/recommendations/fatf-recommendations-2012.pdf`, 2026. Updated June 2026.

[10] Shuang Hao, Chengliang Chai, Guoliang Li, Nan Tang, Ning Wang, and Xiang Yu. *Outdated fact detection in knowledge bases.* In ICDE 2020, 2020.

[11] Bernd Heinrich and Mathias Klier. *Metric-based data quality assessment: developing and evaluating a probability-based currency metric.* Decision Support Systems, 72:82–96, 2015.

[12] Andrey Kolobov, Yuval Peres, Cheng Lu, and Eric Horvitz. *Staying up to date with online content changes using reinforcement learning for scheduling.* In NeurIPS 2019, 2019.

[13] Andrey Kolobov, Yuval Peres, Eyal Lubetzky, and Eric Horvitz. *Optimal freshness crawl under politeness constraints.* In SIGIR 2019, 2019.

[14] Pei Li, Xin Luna Dong, Andrea Maurino, and Divesh Srivastava. *Linking temporal records.* Proceedings of the VLDB Endowment, 4(11):956–967, 2011.

[15] Weiguang Liu, Aureo de Paula, and Elie Tamer. *Prediction sets and conformal inference with interval outcomes, 2026.* arXiv:2501.10117, version 4.

[16] Aakash Pattani. *Revolutionising due diligence for the digital age.* `https://www.deloitte.com/uk/en/services/financial-advisory/blogs/2024/future-of-financial-crime.html`, 2024. Perspective, 6 March 2024.

[17] Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. *Zep: A temporal knowledge graph architecture for agent memory, 2025.* arXiv:2501.13956.

[18] Simon Razniewski. *Optimizing update frequencies for decaying information.* In Proceedings of the 25th ACM International Conference on Information and Knowledge Management, pages 1191–1200, 2016.

[19] Reserve Bank of India. *FAQs on master direction on KYC.* `https://www.rbi.org.in/commonman/english/Scripts/FAQs.aspx?Id=3782`. Page dated 09/06/2025, Q 22.

[20] Burr Settles and Brendan Meeder. *A trainable spaced repetition model for language learning.* In ACL 2016, 2016.

[21] Suhas Shrinivasan and Simon Razniewski. *How stable is knowledge base knowledge?, 2022.* arXiv:2211.00989.

[22] Thibaut Soulard, Fatiha Saïs, and Joe Raad. *When facts expire: Benchmarking temporal validity in knowledge graphs.* In Proceedings of the 34th ACM International Conference on Information and Knowledge Management, pages 6534–6538, 2025.

[23] Krithika Suresh, Cameron Severn, and Debashis Ghosh. *Survival prediction models: an introduction to discrete-time modeling.* BMC Medical Research Methodology, 2022.

[24] Beining Wu, Zihao Ding, and Jun Huang. *ERRAND: budgeted maintenance of agent memory, 2026.* arXiv:2609.29545.

Appendix A Pre-registration details

Table 5. Pre-registered hypotheses (abridged; full text in the repository's `docs/hypotheses.yaml`). Gains are relative gains of the hazard policy over the better of per-class Poisson and calendar in stale found per check (per cost unit in E2).

ID	Experiment	Success rule (pre-registered)
H1	E1, E2	E2: gain ≥ 10% with CI above zero in at least 3 of the 4 scoped cells; E1: CI above zero
H1c	E2	negative control: gain below 5% (CI may include zero)
H2	E2	drop without dependency features ≥ 5% with CI above zero in at least half of the scoped cells
H3a	E1, E2	ECE ≤ 0.03 (E2) and ≤ 0.05 (E1) at every horizon; kill if above 0.08 on E2
H3b	E1, E2	coverage gap ≤ 0.02 (E2) and ≤ 0.03 (E1) at α = 0.1
H4	E3	expected cost reduction ≥ 10% with CI above zero, primary cell
H5	E1	concordance non-inferior to an LLM judgement within 0.01 (only if run)
H1-E1b	E1b	amendment A1: CI above zero at the primary budget, on held-out properties

H1 scope ambiguity and its resolution (D27). The pre-registered E2 scope (heterogeneity moderate or strong, signals on or strong, budget 1 percent) selects 8 cells once both dependency levels are counted, while the success rule speaks of "the 4 scoped cells". Before computing, we chose the strict reading: success needs the rule to hold in at least 3 of 4 cells at each dependency level; the kill rule uses all 8 cells. Results per dependency block are reported too.

Pilot decisions, disclosed. A blinded pilot on 8 seeds that are never confirmatory (300 to 307) wrote only standard deviations, correlations, support tables, harness checks and timings, never an arm's cost or an effect. Four steps departed from the registered procedure. Three of them, and the start of the fourth, were human decisions recorded before the test; the rest of the fourth (D70) was made after the test and affects no verdict.

1. **Burn-in rule amended (D58, D60).** Under the registered drift rule no candidate burn-in from 728 to 2184 days passed on every pilot seed (stop S1): its fixed outcome-share tolerance was about one sampling standard error at the observed window sizes, and the queue-load imbalance does not shrink with burn-in. The rule was amended before any re-run: an outcome-share gap now passes if it is within 3.36 unpooled two-sample standard errors (Bonferroni over 64 comparisons at a 5 percent family-wise rate), and queue load is reported rather than gating. The amendment was made after seeing the pilot's logging-regime checks and before any arm ran; under it the rule chose b = 2184 days.

2. **Slice rate set by override (D61, D62).** The support rule asked for 100 randomly treated decisions with an avoidable failure per action kind in the fitting window of every pilot seed. At a slice of 10 percent the smallest counts were 85 (submit) and 43 (contract); at 20 percent they were 161 and 99 (stop S2). The human accepted q = 20 percent as an override; on the confirmatory seeds the contract count was again below 100 on 2 of 10 primary seeds (reported, not gating); in the transfer condition every count was between 16 and 49. These counts concern the CASS models, but the slice's checks also train the scorer, so the override shaped the scores that the H4r allocations use.

3. **Seed count (D63, D64).** The power rule gave H4r a power of 99.99 percent at 10 seeds, the smallest candidate, at a true saving of X and no completion difference. The CASS hypothesis H6r reached at most 31 percent at 30 seeds against a target of 80 percent (stop S3), so the human moved H6r to reported in the CASS paper and set $n = 10$. X and m were not changed.

4. **A fix, and a drift check moved after the test (D66, D67, D70).** A launcher bug that made the run's own output files count as uncommitted changes, so that the run would have refused to start, was fixed before the freeze. The freeze checklist's drift item compares the fitting window with a test-length window, which on a confirmatory seed is the test window itself, so before the test the human chose to report a substitute comparison after it (D66). After the test that substitute failed on every seed for a reason unrelated to drift, and the human replaced it with the registered logging-regime check (D70; Appendix D).

The burn-in rule reads the true stale rate of used facts and the support rule reads which failures were avoidable; the burn-in rule was amended after the pilot and the slice rate was set by override. No pilot effect informed any setting, and no truth informed the arms, features, hyperparameters, comparator, X, m or the readings. The test ran once (D69).

Appendix B Wikidata task details

Table 6. E1 (Wikidata, public data), test window 2018 to the snapshot, primary budget 1 percent per week, 5 seeds. Last column: relative gain of the hazard policy over each policy in stale found per check, with 95 percent intervals from a paired bootstrap over 14,848 subjects. Negative values mean the policy beats the hazard model.

Policy	Stale per check	Freshness	Age (days)	Hazard vs policy (%)
hazard model (LIV)	0.077	0.962	17.8	
per-class Poisson (main baseline)	0.077	0.956	16.2	0.3 [−1.6, 2.2]
pooled-by-type Poisson	0.077	0.957	16.1	0.0 [−1.8, 1.9]
calendar	0.077	0.960	10.2	−0.3 [−2.1, 1.7]
LambdaCrawl (reimpl.)	0.078	0.961	10.4	−1.2 [−3.0, 0.7]
HLR-style (reimpl.)	0.079	0.964	12.7	−1.9 [−3.1, −0.8]
Weibull, interval-censored	0.079	0.966	9.6	−2.8 [−3.9, −1.6]
Cox, interval-censored	0.079	0.965	10.6	−2.3 [−3.5, −1.2]
Cox, midpoint (approximation)	0.079	0.965	11.1	−2.3 [−3.5, −1.0]
Kaplan-Meier per class	0.074	0.952	25.5	3.7 [1.8, 5.8]
age threshold	0.077	0.958	13.0	−0.5 [−2.2, 1.5]
uniform random	0.069	0.930	46.2	11.5 [9.4, 13.7]

Data. Of 68,659 fetched rows, 27,482 statements on 15,749 subjects are kept (41,045 dropped for year or coarser precision). Employer (P108) and position held (P39) were dropped by the human's decision (D24), which keeps E1 corporate-only and free of personal data.

Two subject covariates (sitelink and statement counts) were measured at the snapshot, after both landmarks, and the statement count includes the outcome statements; both were excluded from every run for leakage, a choice accepted by the human together with the frozen settings (D25, D26). The chosen hazard settings are a bin width of 30 days and 3 EM iterations.

E1b. The verification rule for property IDs was widened to accept an English alias before any P749 statement was fetched (its current label is "parent organization or unit"), and the human accepted P749 at the freeze (D30). Settings were tuned on validation and frozen by the human (D30). The held-out test pool has 3812 statements, 1966 held at the start of the test window and 460 ends inside it. A run of the E1b model on the original five properties re-uses a spent test set and is labelled exploratory.

Sensitivity analyses. Two sensitivity analyses do not change this: including year-precision statements (dates drawn uniformly within the year), the gain over calendar is 0.17% [−1.11, 1.47]; on a proxy for recently edited subjects (subjects with a tracked statement starting or ending in the five years before the landmark), it is −7.38% [−11.07, −3.37] against calendar.

E1b, further results. At 0.5 and 2 percent the gains are −2.3% [−7.8, 3.7] and −1.6% [−6.6, 3.8]. The co-change features do help the hazard model against its own ablation at 0.5 percent (8.5% [2.6, 15.3]) but not at 1 (1.7% [−2.0, 5.6]) or 2 percent (0.3% [−2.8, 3.5]); the hazard model without them is worse than

the better anchor at the primary budget (−4.9% [−9.0, −0.6]). The exploratory re-use of the spent E1 window gives 0.4% [−1.5, 2.5]; it is reported for completeness and carries no confirmatory weight.

Appendix C E3 in full

Setup. E3 joins the drifting facts of `tacitdrift` to a minimal loan-intake process, standalone and inspired by SimBank (decision D35). Each entity opens a case about once a year. The scripted agent submits 14 days after opening, using the stored employer and income band; after a successful submit it approves 3 days later and sends the contract, using the stored address, phone and email; the deadline is 30 days after opening. Outcomes are decided by rules over the stored and the true values: a wrong employer or an income band stored above the truth gives a rejection, and so does a true income band at or below 2 whatever the ledger holds (a failure that no check can prevent); an income band stored below the truth or a wrong address gives a rework (10 days for the document), a wrong phone or email alone a contact rework (2 days); a rework past the deadline becomes a deadline breach, and 5 percent of outcomes are replaced by noise. The cost table charges 100 for a rejection, 30 for a rework, 5 for a contact rework, 50 for a breach and 1 per check cost unit. All of these are scenario settings, not claims about real lending. The world has 5000 entities. Warm-up logging runs until day 364, hazard settings are chosen on the validation window up to day 546, and every arm then runs on its own copy of the ledger and the process over the test window up to day 1092, at an equal check budget with a shared randomised slice of 5 percent. The settings were frozen before the test window was run (decision D40), and the test window was run once.

The arms rank facts by $P(\text{stale}) \cdot w / c$ with the hazard model of Section 4: `calendar`; `fresh_global` ($w = 1$, the E2 policy); `fresh_case` (w = the fact's exposure to pending actions of open cases, all costs equal); `conseq_oracle` (w = the expected cost of acting on a wrong value, from the rule and cost tables, never from true values); and `conseq_learned`, which takes w from the CASS consequence gate trained on warm-up outcomes. The learned arm is reported only and never decides H4. We call the weights of `conseq_oracle` *rule-table weights*: despite the arm's code name they are not cost-optimal, since they price one wrong fact at a time and ignore both the rejections that no check can prevent and interactions between wrong facts (ledger C6k). Expected cost per case, the registered measure, is the cost of the realised outcomes of every case opened in the test window plus the arm's check cost, divided by the number of those cases. Evaluation stops at the end of the window, at day 1092, while a case's initial submit comes 14 days and its deadline 30 days after opening, so the cases opened last are not followed to their outcome. The measure is therefore the cost accrued in the window per case opened in the window, not the complete cost of those cases; Section C.1 counts the incomplete cases and repeats the comparison on the complete ones. Following the strict reading fixed before any E3 run (D35), H4 is decided on expected cost, in the primary cell only (budget 1 percent of fact cost per week, signals and dependencies on), against the better of `fresh_global` and `fresh_case` by point estimate on test, so that a gain cannot come only from knowing which entities have open cases; fewer failed actions is claimed only if their reduction also has an interval above zero. Intervals are paired cluster bootstrap intervals (2000 resamples, clusters of seed and entity, ratio of totals pooled over 5 seeds, 100 to 104).

Table 7. E3 primary cell, test window: costs and failed actions accrued in the window per case opened in it (cost-table units), pooled over 5 seeds and 37,315 cases per arm; checks summed over seeds. Check cost is equal by construction (equal budget); arms differ in how many checks the budget buys because check costs differ by attribute. Simulated data.

Arm	Expected cost	Outcome cost	Check cost	Failed actions	Checks
`calendar`	46.36	41.14	5.23	0.533	96,297
`fresh_global`	48.44	43.22	5.23	0.513	126,179
`fresh_case`	37.30	32.07	5.23	0.369	108,730
`conseq_oracle`	34.78	29.55	5.23	0.410	78,921
`conseq_learned` (reported)	38.03	32.81	5.23	0.403	103,211

H4. The better freshness-only arm on test is `fresh_case` (Table 7); `fresh_global`, which ignores open cases, costs more than calendar. Against `fresh_case`, the rule-table-weighted arm lowers expected cost per case by 6.8% [6.2, 7.3] (a difference of −2.52 units per case, interval −2.74 to −2.31). The interval is above zero but the reduction is below the pre-registered 10 percent, so the pre-registered verdict is *neither*: neither success nor kill (ledger C6, with the smaller reduction recorded as C6b). The statement's second part does not hold: the rule-table-weighted arm has *more* failed actions per case than `fresh_case`, a relative reduction of −11.2% [−12.2, −10.1] (C6e; the pre-registered claim of fewer failed actions, C6a, is refuted). The arm spends checks on the facts whose errors cost most: summed over seeds it lets fewer submits be rejected (9631 against 10,762) and more contracts go to rework (3660 against 877). Fewer failed actions and lower cost are different objectives, and the weights target the second; Section C.1 breaks both down by class.

Table 8. E3 test window: relative reduction per case against the better freshness-only arm of each cell (`fresh_case` in every cell), with 95 percent paired cluster bootstrap intervals. Positive means lower than the comparator; a negative failed-actions value means more failed actions. Only the primary cell's cost column decides H4; the other cells and the learned arm are reported. Simulated data.

Cell	`conseq_oracle`, cost	`conseq_oracle`, failed actions	`conseq_learned`, cost
Primary (decides H4)	6.8% [6.2, 7.3]	−11.2% [−12.2, −10.1]	−2.0% [−2.7, −1.3]
Budget 0.5 percent	17.7% [16.9, 18.5]	−5.2% [−6.1, −4.2]	8.0% [7.2, 8.8]
Budget 2 percent	0.3% [0.1, 0.4]	−0.5% [−0.8, −0.2]	−5.5% [−5.9, −5.0]
Signals off	9.6% [9.0, 10.2]	−16.6% [−17.8, −15.5]	1.7% [1.0, 2.4]

Reported cells. In all 3 reported cells the rule-table-weighted arm again has lower cost and more failed actions than `fresh_case`, with every interval excluding zero (Table 8, C6c). No reported cell goes the other way on cost. The reduction is largest at the smallest budget (17.7 percent), close to zero at the largest (0.3 percent) and 9.6 percent without signals. These cells do not change the H4 verdict.

Learned weights. With weights from the CASS gate, the arm costs more than `fresh_case` in the primary cell (−2.0% [−2.7, −1.3]) and more than the rule-table-weighted arm in every cell (relative reduction against it −9.4 percent in the primary cell, interval −10.1 to −8.7); it beats `fresh_case` only at budget 0.5 percent and without signals (C6f; the claim that learned weights reduce cost, C6d, is refuted in the primary cell). Between 30 and 52 percent of the gate's per-use cost differences per run were negative, that is, the gate expected a higher cost after a check; they are floored at zero (decision

D38). The gate's difference is a predictive contrast learned from logged outcomes, not a causal effect of checking, and this arm does not show that learned consequence weights help here.

FIG. 3 · COST PER CASE BY ALLOCATION

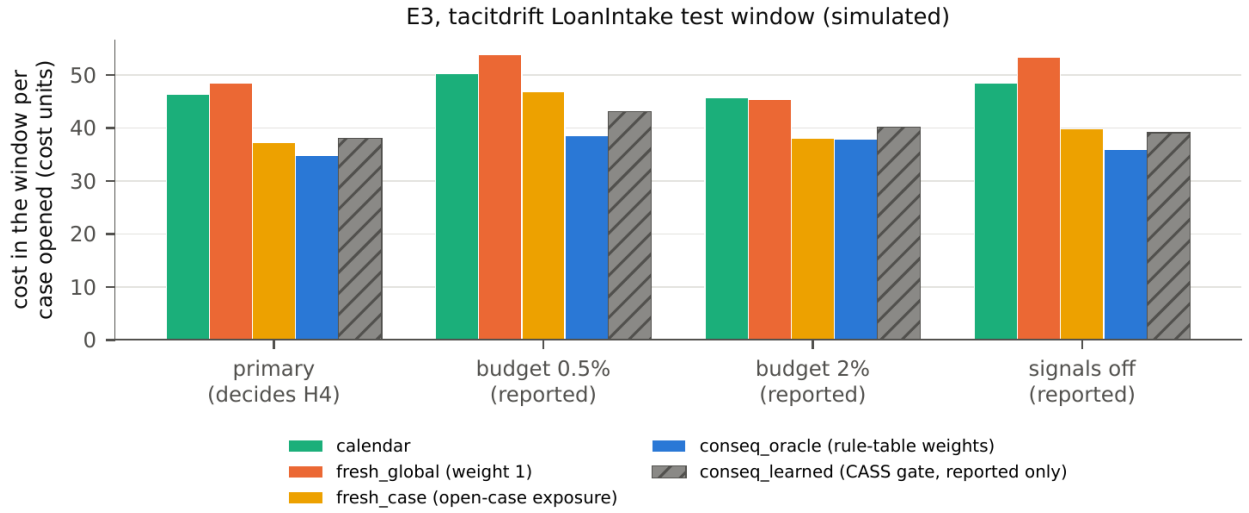


Figure 3. E3 test window: expected cost per case (outcome cost plus check cost, accrued in the window per case opened in it) of every arm in the primary cell and the 3 reported cells, pooled over 5 seeds. Point estimates; intervals for the comparisons are in Table 8. Simulated data.

C.1 Post-hoc analyses of E3 (D46)

Scripts. The post-hoc E3 analyses (Section C.1) are computed by `scripts/e3_posthoc.py` from those per-case and per-check outputs of the single E3 test pass and from the committed per-decision log of CASS E4, and the exploratory check on synthetic seeds by `scripts/e3_posthoc_avoid.py`. The post-hoc analyses after the external reviews (D49) are computed by `scripts/audit_posthoc.py` from the same outputs and the committed go/no-go and E1 summary files.

This analysis is post hoc. An adversarial review after the single test pass found gaps in how E3 was reported, and two later external reviews found more (D49). Every quantity in this subsection was chosen after the test results were seen, is computed by `scripts/e3_posthoc.py` (D46) or `scripts/audit_posthoc.py` (D49) from the existing test outputs, and decides nothing: the H4 verdict above stands. E3's own files do not record the true income band or the noise draw, so the analysis reads them from the per-decision log of CASS E4, which runs the same worlds, cases and keyed noise streams on the same seeds and executes every case's initial submit at the same time as every E3 arm. For every case that this log marks as an unavoidable rejection, every E3 arm records exactly one rejected submit and nothing else.

Where the cost comes from. Table 9 splits each arm's expected cost per case in the primary cell into the check cost; an *unavoidable rejection floor*, the cases whose initial submit is rejected because the true income band is at or below 2 and whose outcome was not replaced by noise, the same in every arm; noise, the exact cost of noisy initial submits plus the expected cost of the noise draw for every later action (an approximation); and the remainder, the rest of the outcome cost. For `fresh_case` the floor is 22.67 of 37.30 cost units per case (61 percent; per seed 22.57 to 22.82 of 37.03 to 37.69), the check cost 5.23 and noise about 3.88, which leaves a remainder of about 5.52. The floor is an exact per-case component: these cases are rejected at their initial submit in every arm, and no refresh

policy can prevent it. The remainder is not a bound on what refresh could avoid. Its noise term is an expected cost along each arm's own action path, and a policy that changes the path (for example by letting a case close earlier) changes its exposure to later noise; nor is the cost of a perfect-information path a lower bound, since an external audit constructed a reachable case in which that path costs more than a feasible path that leaves a fact stale. The remainder is about 14.8 percent of the comparator's cost per case (per seed 14.4 to 15.1; C6h), an approximate decomposition. Against that share, the pre-registered 10 percent bar corresponds to about 68 percent and the observed reduction to about 46 percent; these ratios describe the decomposition, not a share of an achievable maximum. Measured above the floor, which needs no approximation, the rule-table-weighted arm's cost is lower than that of `fresh_case` by 17.2% [16.0, 18.5] (C6g). The bar was fixed before this structure was known, and it is not changed.

Table 9. Post hoc (D46): decomposition of the expected cost per case in the E3 primary cell, test window, pooled over 5 seeds. Floor: initial submits rejected because the true income band is at or below 2, with no noise (exact, equal in every arm). Noise: exact for the initial submit (2.27 per case), expected for every later action along the arm's own path. Remainder: the rest of the outcome cost; it is not a bound on any policy. * Approximate. Simulated data; decides nothing.

Arm	Expected cost	Check cost	Floor	Noise*	Remainder*
`calendar`	46.36	5.23	22.67	3.90	14.57
`fresh_global`	48.44	5.23	22.67	3.74	16.81
`fresh_case`	37.30	5.23	22.67	3.88	5.52
`conseq_oracle`	34.78	5.23	22.67	4.10	2.78
`conseq_learned` (reported)	38.03	5.23	22.67	3.95	6.18

Failed actions by class. Counting every failed action once treats a contact rework (cost 5) like a rejection (cost 100), so under this cost table the failed-action count and the cost need not agree. Table 10 breaks failures down by class. Against `fresh_case`, the rule-table-weighted arm has fewer submit rejections above the floor (0.031 against 0.062 per case) and more contact reworks (0.080 against 0.009). The cost-weighted failure measure, which is the outcome cost per case, falls by 7.9% [7.2, 8.5], while failed actions above the floor rise (relative reduction −29.0% [−32.0, −26.2]; C6i).

Table 10. Post hoc (D46): failed actions per case by class in the E3 primary cell, test window, pooled over 5 seeds, and their cost per case. The last cost row, the cost-weighted failure measure, equals the outcome cost per case. Contract reworks are split into document and contact reworks from each case's outcome cost and class counts. Simulated data; decides nothing.

		Per case		Cost per case	
Failure class	**Unit cost**	`fresh_case`	`conseq_oracle`	`fresh_case`	`conseq_oracle`
Submit rejected, floor	100	0.227	0.227	22.67	22.67
Submit rejected, other	100	0.062	0.031	6.17	3.14
Submit rework	30	0.026	0.020	0.78	0.60
Submit deadline breach	50	0.012	0.012	0.62	0.62
Contract rework, document	30	0.014	0.018	0.43	0.54
Contract rework, contact	5	0.009	0.080	0.05	0.40
Contract deadline breach	50	0.009	0.010	0.45	0.52
Contract rejected (noise)	100	0.009	0.011	0.90	1.05
All failed actions		0.369	0.410	32.07	29.55

Incomplete follow-up and complete cases (D49). Also post hoc, after two external reviews. Evaluation stops at day 1092, but cases opened up to that day need until their deadline. In the primary cell, 969 of the 37,315 population cases (2.6 percent) have no judged outcome by the end of the window, the same number in every arm, and 2102 (5.6 percent) have a deadline at or after it (C6m). Every judged action of a case falls at or before its deadline, so the 35,213 cases whose deadline lies inside the window are complete at its end. Restricted to them, with the simulator's own definition of that population (outcome cost of those cases plus the arm's check cost, per case), the rule-table-weighted arm lowers cost per case against `fresh_case` by 6.7% [6.2, 7.3]; the relative reduction of its failed actions per case is −11.2% [−12.3, −10.2], and that of the learned arm's cost −1.9% [−2.7, −1.2] (C6n). On the complete cases the comparison reads as the registered one. Shared truncation does not by itself make a comparison unbiased, since arms differ in reworks and closures, which is why the paper describes the registered measure as cost accrued in the window per case opened in it.

Seed-level spread. Per seed, the reduction of expected cost against `fresh_case` ranges from 6.5 to 7.0 percent. Over the 5 seeds the mean is 6.8 percent with a 95 percent Student *t* interval of [6.5, 7.1] (4 degrees of freedom), beside the pooled cluster bootstrap interval of Section C, which treats entities rather than seeds as the resampled unit (C6j). For failed actions per case the per-seed mean is −11.2 percent, *t* interval [−12.6, −9.7]. Neither interval changes the reading of H4.

An exploratory weight on synthetic seeds. The rule-table weights ignore that a submit whose true income band is at or below the threshold is rejected whatever is checked. The reviewer tried a variant that multiplies each submit use by an estimate of the probability that the submit is not rejected anyway, from the stored band and the model's *P*(stale) of it. We reproduced that check with a committed script (`scripts/e3_posthoc_avoid.py`) on two synthetic seeds that are not E3 seeds (7 and 8), in the validation window only, with one run per seed and no intervals. Against `fresh_case` the rule-table weights lowered expected cost by 7.1 and 6.7 percent and the adjusted weight by 8.2 and 7.7 percent (C6l). This is exploratory: it is not evidence about H4 and would need a pre-registered test on new seeds.

Why E3 fell short of its bar. The registered results show that the rule-table-weighted arm traded failure classes: it let fewer submits be rejected and more contracts go to rework, which lowered cost and raised the failed-action count (C6b, C6e), so the two parts of H4's statement measure different things under this cost table. The learned arm is not a fair test of learned weights (Section 8). Post-hoc analyses that bear on the shortfall, a cost decomposition with an exact unavoidable rejection component, failures by class and an exploratory reweighting, are in Section C.1; they decide nothing.

Appendix D Phase 2b-r: further reports

Interactions. The registered interaction $I = C[A,P] - C[A,N] - C[F,P] + C[F,N]$ for a decision policy P is 0.79 (0.20 to 1.38) for the uplift rule, 0.80 (0.45 to 1.14) for the risk rule and 1.55 (1.33 to 1.77) for always check; the last equals the H4r saving by construction, because the two allocations cost the same under always check. We report the interaction estimates and their intervals only.

Stationarity and deployment shift, reported after the test. The faithful check continues the logging regime alone through a test-length window on each confirmatory seed and applies the amended drift rule, as the pilot did: the regime was within the rule on 10 of 10 seeds in the primary condition and 10 of 10 in the transfer condition (C9e). Queue load, which the amended rule reports without gating, had a standardised mean difference above 0.10 on 7 of 10 seeds in the primary condition (largest 0.88) and 7 of 10 in the transfer condition (largest 0.97). The substitute comparison adopted before the test (D66) set the fitting window against the enrolment window of the `fresh_case` arm without a gate; it failed on every seed, because it measures the planned change from the logging regime to the deployed allocation, not drift (D70). We report it as the covariate shift that arm's decisions presented to the frozen models: the stale rate of the facts used at decisions was 0.094 to 0.108 per seed in the fitting window and 0.018 to 0.022 in the enrolment window (primary; transfer 0.077 to 0.088 against 0.020 to 0.023; C9f). The scorer and the CASS models were fitted where used facts were stale far more often than at those decisions; other arms and the calibration window were not compared.

Registered quantities not reported. Of the reported quantities registered with A2, the scorer's calibration on the test window's random checks, P(valid) at decisions, and the decision-state comparison for arms other than `fresh_case` (parts of R9) were not computed; the optional coverage run (R12) was built but not run; the failure split by cause and the excess over the perfect-information reference (parts of R1) are in the results file but not tabulated here; the slice's data-collection cost (R8) is reported in the CASS paper.

Appendix E Further simulator results

E2 runs. The 125 runs span 7 commits and 37 started with uncommitted changes in the repository, made while the E1 code was being built. We re-ran one such run at a later commit with the same seed; every non-timing field of its metrics matched exactly (D28), and the only intervening changes on E2 code paths were a hook whose default is the original design and a new bootstrap function.

The H2 dependencies-off control. In the dependencies-off cells, 6 of 9 cells show drops whose intervals exclude zero, in both directions. These cells were registered as a leakage control (a significant drop there was to indicate leakage), but they are not an independence control: switching dependencies off removes only the triggered co-changes and keeps the entity frailty that multiplies every attribute's rate, so with moderate or strong heterogeneity an observed parent change can carry real information about a child's hazard. The control therefore cannot separate leakage from that information, and our earlier reading, that the ablation lies within model-fit noise, is not established either. Post hoc (ledger C3a): 1 of the significant control drops is positive, in a homogeneous cell where no frailty is shared, and 5 are negative, where removing the features helped; neither pattern looks like leakage. Separating shared frailty from triggered co-change would need simulator conditions with each alone.

Table 11. H3a: expected calibration error of the hazard model's P(valid at t_0 + h) on the test period (E2 central cell, mean over seeds; E1 primary run), with concordance for reference. Bars: 0.03 (E2) and 0.05 (E1).

Horizon (days)	E2 ECE, recal.	E2 ECE, raw	E1 ECE	E2 concordance	E1 concordance
30	0.024	0.097	0.008	0.659	0.706
90	0.019	0.110	0.008	0.659	0.717
180	0.024	0.132	0.012	0.648	0.727
365	0.049	0.201	0.015	0.629	0.736

One-axis sweeps (point estimates, exploratory). In the central cell the gain is 22.4, 23.0, 19.0 and 11.4 percent at budgets of 0.5, 1, 2 and 5 percent: it shrinks as the budget grows and more stale facts can be found by any policy (Figure 2, right). With $\kappa = 0.7$ the central-cell gain is 40.2 percent and with $\kappa = 1.5$ it is 16.2 percent; in the negative-control corner at those κ values it is −0.9 and −0.4 percent.

Freshness and age. Table 12 lists all policies in the central cell. The hazard policy has the highest freshness (0.896) but not the lowest age: several policies, among them calendar (36.3 days), LambdaCrawl (34.0 days) and per-class Poisson (38.9 days), have a lower time-averaged age than the hazard policy (41.8 days). The greedy policy maximises stale facts found per check, not age; a facts-found objective and an age objective rank policies differently, as Cho and Garcia-Molina show for freshness and age [5].

Table 12. All policies in the E2 central cell (moderate heterogeneity, dependencies and signals on, κ = 1, budget 1 percent), pooled over 5 seeds; equal budget in cost units. Simulated data.

Policy	Stale per cost unit	Stale per check	Freshness	Age (days)
hazard model (LIV)	0.155	0.239	0.896	41.8
hazard, no co-change features	0.152	0.234	0.895	42.7
per-class Poisson (main baseline)	0.126	0.190	0.882	38.9
pooled-by-type Poisson	0.127	0.190	0.882	38.9
calendar	0.125	0.252	0.877	36.3
LambdaCrawl (reimpl.)	0.130	0.224	0.883	34.0
HLR-style (reimpl.)	0.147	0.224	0.894	37.9
Weibull, interval-censored	0.145	0.212	0.892	41.2
Cox, interval-censored	0.145	0.217	0.892	40.4
Cox, midpoint (approximation)	0.136	0.199	0.885	48.6
Kaplan-Meier per class	0.119	0.163	0.876	45.3
age threshold	0.093	0.109	0.860	61.9
uniform random	0.082	0.162	0.838	77.8

Table 13. H2: relative drop in stale found per cost unit on child attributes when the co-change features are removed, with 95 percent intervals. Left: scoped cells; right: the dependencies-off control. Simulated data.

Heterogeneity	Signals	Drop, dependencies on (%)	Drop, dependencies off, control (%)
homogeneous	off	0.8 [0.2, 1.4]	−3.0 [−3.8, −2.2]
homogeneous	on	0.2 [−0.5, 0.9]	0.9 [0.0, 1.7]
homogeneous	strong	−0.4 [−0.9, 0.2]	−1.1 [−1.7, −0.5]
moderate	off	1.1 [0.4, 1.8]	−2.9 [−3.7, −2.1]
moderate	on	2.0 [1.3, 2.7]	−3.5 [−4.4, −2.6]
moderate	strong	0.2 [−0.3, 0.8]	−0.7 [−1.4, 0.0]
strong	off	2.9 [2.2, 3.7]	0.1 [−0.8, 1.0]
strong	on	0.5 [−0.3, 1.2]	0.0 [−0.9, 0.8]
strong	strong	0.0 [−0.5, 0.5]	0.0 [−0.7, 0.6]

Reliability curves. Not shown. The results files store the ECE per horizon but no binned reliability data, so drawing the curves needs the harness to write the bins and the evaluation to be re-run; that is open.

E2 tuning criterion (correction, D37, D39). In each E2 run the hazard settings (bin width, EM or uniform spread, LightGBM settings; the model without co-change features uses the same ones) and the weight of the HLR-style model were chosen on the validation checks, but not by the per-check log loss we intended. An array-shape error broadcast the predictions against the labels, so the criterion was the mean log loss over all pairs of a validation prediction and a validation label: it compares every prediction with the share of unchanged checks and ignores which prediction belongs to which

check. The settings were still chosen on validation data only and fixed before the test window, so the E2 comparisons are runs of fixed, validation-chosen settings, but they were not tuned by validation log loss. The age threshold (Youden index) and the recalibration maps (Brier score) used other criteria and are not affected, nor are E1 and E1b. The pre-registered E2 results are kept as run; Section E.1 reports a post-hoc re-run of the central cell with the per-check criterion.

E.1 Post-hoc sensitivity: the E2 tuning criterion (D39)

This analysis is post hoc. The E2 test windows of seeds 0 to 4 had already been seen when it was decided, it does not replace the pre-registered E2 results above, and it decides no hypothesis. After the tuning error described in Appendix E was found, the human chose to keep E2 as run and to re-run the central cell (moderate heterogeneity, dependencies and signals on, $\kappa = 1$, budget 1 percent) with the per-check log loss as the tuning criterion, everything else unchanged. The original configuration re-run at the new code reproduced all 5 committed central-cell runs with 0 differing fields, so the two arms differ only in the criterion. The per-check criterion chose a different hazard setting in 1 of 5 seeds and the same one in 4, and an HLR weight of 0.01 instead of 0.00 in every seed. Table 14 compares the two criteria with the go/no-go's bootstrap. The hazard policy under the per-check criterion against itself under the original one gains 0.0% [−0.2, 0.2] per cost unit. Re-applying the pre-registered rules with the central cell's per-check values in place of the original ones, all other cells unchanged, changes 0 verdicts; the other cells were not re-run.

Table 14. Post-hoc sensitivity (D39): the E2 central cell under the original tuning criterion and under the per-check log loss, 5 seeds each, 95 percent paired cluster bootstrap intervals as in the go/no-go. The left column equals the pre-registered results. Simulated data.

Central cell, E2	Original criterion (as run)	Per-check criterion (post hoc)
H1 gain vs better anchor (%)	23.0 [21.9, 24.0]	23.0 [21.9, 24.1]
H2 drop, child attributes (%)	2.0 [1.3, 2.7]	1.7 [1.0, 2.4]
H3a ECE, recal., 30 / 90 / 180 / 365 days	0.024 / 0.019 / 0.024 / 0.049	0.024 / 0.019 / 0.024 / 0.049
H3b coverage of *T* (trivial bounds, %)	0.991 (58)	0.991 (59)
HLR-style, stale per cost unit	0.147	0.147
Hazard vs HLR-style (%)	6.0 [5.3, 6.7]	5.7 [5.0, 6.4]
HLR weight chosen	0.00	0.01

Appendix F The conformal lower bound

Survival models and conformal bounds. Discrete-time hazards fitted on person-period data with any binary classifier, or reduced to Poisson regression so that boosted trees can be used, are standard [1, 23]. Our conformal lower bound follows the interval-outcome conformal sets of Liu, de Paula and Tamer, which are valid in finite samples under exchangeability whatever produced the interval [15], and the argument of Candès, Lei and Ren that a calibrated lower bound on a lower surrogate of the time to event is a valid, conservative bound on it [3]; the specialisation to the last unchanged check is our derivation. That argument needs the surrogate to lie below the event time, which in our simulator fails for values that can return (Section 4).

Conformal lower bound. We give a lower bound $q(X)$ on the time to change T, aiming at marginal coverage $P(T \geq q(X)) \geq 1 - \alpha$ at $\alpha = 0.1$, by one-sided split conformal calibration on the left endpoint L of each interval (the last check at which the fact was still unchanged). The reduction to T is our derivation from the interval-outcome conformal sets cited in Section 2: a bound that covers L also covers T if $L \leq T$. That premise holds for values that never repeat (the simulator's employer, address, phone and e-mail, and every Wikidata statement, whose end is final), but not for the simulator's income band, whose value can change and return before a check, so that L can exceed T. The split-conformal argument also needs a score function fixed independently of the calibration data; in E1 and E2 the hazard settings were chosen on the same validation window whose earliest check intervals calibrate the bound. We therefore claim no finite-sample guarantee for the bound and report its coverage as an empirical observation. Beyond these two points the argument assumes exchangeable calibration and test units (check schedules drawn the same way, which a temporal split does not ensure) and gives a marginal guarantee only.

Appendix G Experiment-specific limits

- **Scope of E1.** E1 is corporate-only by the human's decision (employer and position held were dropped), so person-level facts are evaluated only in the simulator.
- **Optimistic co-change timing in E1b.** Other properties' ends are treated as known on their dates without a check; real systems learn of them later.
- **Customer updates as inspections.** In the simulator, customer-initiated updates arrive after changes, which shifts change times within intervals, and corrections after a failed action depend on outcomes. Training uses intervals closed by checks and by updates, so the observation process is partly informative; we give no observation-model argument or sensitivity analysis showing that every observed interval behaves like standard interval censoring.
- **Conformal calibration reuses selection data.** In E2 the hazard settings are chosen on the validation checks and the bound is calibrated on the earliest validation interval of each fact; in E1 the frozen settings were tuned on the validation window whose intervals calibrate the bound. The score function therefore depends on the calibration data, and the fixed-score split-conformal argument does not strictly hold. One interval per fact also leaves attributes of the same entity dependent, and a temporal split does not make calibration and test units exchangeable.
- **The dependency control.** The H2 dependencies-off cells keep the entity frailty shared by all attributes, so they are not an independence control; a valid control would separate shared frailty from triggered co-change (post hoc, ledger C3a).
- **Retrospective lifespans and an oracle budget in E1.** Lifespans come from start and end qualifiers in a later snapshot, which record when a statement held, not when it was known, and the organisation is assumed to record a statement at its start. The weekly budget is proportional to the number of statements truly alive, computed from their end dates, which an organisation holding the facts would not know. Equal budgets keep the comparison controlled, but this is not an implementable budget rule.
- **The E2 tuning correction covers one cell.** The tuning error of Appendix E was corrected only in a post-hoc re-run of the central cell (Section E.1); the settings of every other cell were chosen by the faulty criterion, and the central-cell result does not validate them.
- **No comparison with ERRAND.** We name ERRAND [24] as the closest prior work but did not run its policy as a baseline.
- **Downstream cost in the simulator only.** The action-cost evaluation (E3) uses a minimal, standalone loan-intake process with a stated cost table and deterministic outcome rules; its numbers depend on those settings and are not real lending costs. The weights that decide H4 are rule-table weights: they use the true rule and cost tables but price one wrong fact at a time, so they are not cost-optimal (post hoc, C6k). Post hoc, most of the cost per case is initial-submit rejections that no refresh can prevent (C6g); the noise and remainder in that decomposition are approximate, and the remainder is not a bound on any policy (C6h). Cost and failed actions pulled in different directions, so a deployment must choose which it optimises.
- **Incomplete follow-up in E3.** Evaluation stops at the end of the test window, so the cases opened last are not followed to their outcome; the measure is the cost accrued in the window per case

opened in it, not complete case cost. Post hoc, on the cases complete at the window's end the comparison reads the same (C6m, C6n).

- **The learned-weights arm was handicapped.** Its result is not a fair test of learned consequence weights. The CASS gate prices every rework at the full rework cost (decision D38), so a wrong phone or email gets 6 times the weight the rule table gives it. The gate is trained on the warm-up, which is not stationary: every stored value starts with its onboarding verification, and the shared logging checks a fact about once every 700 days on average, so the warm-up up to day 364 covers about 52 percent of one such interval, and its fact ages and verification methods differ from those of the test window. And its weight is a predictive contrast that is confounded: in the warm-up a freshly verified fact mostly comes from the correction after a rework, so recent verification co-occurs with late re-executions (a possible reason, not verified, for the many negative per-use differences). The contrast also edits a fact's verification time and method with the stored value fixed, while a real check can reveal a different value, so it does not average over check results; it is a predictive difference, not the effect of checking; and if an expected check benefit already averages over whether the fact is stale, multiplying it by P(stale) in the greedy rule counts that uncertainty twice. A check can also raise later cost by changing a case's path, so negative differences are not necessarily noise.
- **Pilot decisions in Phase 2b-r.** The burn-in rule was amended after the pilot stopped on it, the slice rate was set by a human override of the support rule, and the seed count followed from moving the CASS hypothesis to reported (Section 6.3.2). The burn-in and support rules read simulator truth, and none of these decisions saw an arm's cost. Two choices were made after the test, with its results read: replacing the failed substitute drift comparison with the registered logging-regime check (D70), and the post-hoc contrast of Section 6.3.3 (D72); neither changes the verdict.

FlowX.AI

FlowX.AI builds and orchestrates enterprise-grade AI agents for regulated industries, with a focus on banking, financial services, and insurance. The study behind this paper was run on public Wikidata change histories and on simulators; nothing in it was tested on the FlowX.AI Platform or on client data.

FlowX.AI Technical Paper · LIV version 1.0.0 · doi.org/10.5281/zenodo.24077472 · Bogdan Răduță, Head of Research · bogdan.raduta@flowx.ai

© 2026 FlowX.AI. Text and code licensed under the Apache License, Version 2.0. Preprint, not peer reviewed. The archived release at the DOI above holds the LaTeX edition of this paper and the complete source.