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

Detectability Bounds and Decision-Aware Evaluation for Change Detection in Heterogeneous Relational Event Streams: A Reproducible, Fully-Labeled Benchmark as the Empirical Instrument

Aditya Singh

Independent Researcher

Abstract: ***Change detection over heterogeneous relational event streams- where a fault is a temporal process that perturbs one of many correlated channels, the same symptom can arise from distinct latent causes, and detection must ultimately be judged by intervention cost rather than ranking alone- lacks two things the problem needs: a theory that says which faults are detectable and how quickly, and an evaluation protocol that scores detectors where operators act. We address both. First, we develop a decision-aware evaluation theory for budgeted intervention: an optimality result identifying the cost-minimizing budget-constrained policy under calibration, and a calibration-regret bound that turns probability quality into a quantified decision cost rather than a diagnostic convention. Second, we give per-fault information-theoretic detectability bounds via quickest-change-detection theory, computed from a generator's own parameters, that partition faults into singular (arrival-limited), regular (finite-information), and granularity-limited regimes; and a scope-invariance result characterizing when symmetric monitors generalize across fault scopes while trained detectors need not. To make these results measurable we require a stream with exact, timestamp-level ground truth - which commercial data cannot supply- so we instantiate them in CRM-IntegrityBench, an open, seed-reproducible generator for relational CRM event streams (five entity classes with foreign-key relations, seasonality, and provenance) with a parameterized corruption-injection framework implementing twelve fault families with exact onset, duration, scope, and per-event labels. The accompanying protocol scores detectors on discrimination, timeliness, calibration, and budget-constrained expected loss under leakage-safe chronological replay. Evaluating five CPU-reproducible detector families (rule-based monitoring, per-channel CUSUM, unsupervised anomaly detection, calibrated linear models, and gradient-boosted trees) across fifteen seeds (5.7 million events, 2,160 injected incidents), gradient-boosted trees lead ranking and calibration- AUPRC 0.833 versus 0.723 for a robust rules monitor (paired difference +0.109, 95% bootstrap CI [0.078, 0.145]), with the lowest Brier score and expected calibration error- while at a 1% intervention budget the budgeted expected loss is statistically indistinguishable across the calibrated detectors, because the budget saturates against the base rate; the detectors separate on loss only at looser budgets (≥ 1.5%). Detection separates sharply at the family level rather than the aggregate: each detector attains near-perfect detection on some fault families and near-zero on others, and several families are hard for every detector. A feature-leakage ablation shows the absolute numbers are optimistic- removing six fault-specific indicator channels costs every detector 0.10–0.44 AUPRC- but the comparative finding strengthens: the gradient-boosted margin over the deviation monitors more than doubles, from +0.109 to +0.243, once hand-built channels are withheld. No detector family dominates every family; the measured per-family detection ordering closely tracks the theory's predicted difficulty (Spearman $\rho = -0.79$, $p = 0.002$) - exactly the discrimination that multi-axis, decision-aware evaluation is designed to surface. Generator, configurations, and evaluation code are released for independent reproduction and extension.***

Keywords: Change Detection, Relational Event Streams, Fault Detection, Decision Aware Evaluation, CRM Integrity Bench

1. Introduction

Detecting faults in a live relational event stream- a connector that begins mislabeling records, a foreign key that starts breaking, a value distribution that drifts- is a change-detection problem with three features that standard anomaly-detection practice handles poorly. First, faults are temporal processes: a degradation unfolds over many events rather than producing one violating record, so the relevant question is detection delay under a controlled false-alarm rate, not pointwise classification. Second, faults are relational and heterogeneous: a single root cause manifests across several correlated channels of different types (counts, categoricals, latencies, linkage statistics), so detectability depends on which channel carries the signal and how much. Third, monitoring is budgeted: operators act on a bounded number of alerts, so the practically decisive quantity is the expected loss that remains under a fixed intervention budget- a decision quantity that ranking metrics like AUPRC do not measure.

These three features expose a pair of gaps that this paper targets directly. The first is theoretical: given such a stream, which faults are detectable at all, how quickly, and when does a detector's probability quality actually change the decision it induces? The second is methodological: how should detectors be compared so that the comparison reflects operator-relevant outcomes rather than a single ranking score? Progress on both has been blocked by a mundane obstacle — the streams where these questions matter most, operational customer-relationship-management (CRM) systems, encode customer identities, commercial relationships, and revenue, so no organization can release them with ground truth. Published industrial results are therefore unverifiable and non-comparable, and the theory has had no instrument on which to be checked.

Contributions

We make the theory and the evaluation the primary contributions, and build a released, fully-labeled benchmark as the instrument that makes them measurable.

1) **A decision-aware evaluation theory for budgeted intervention (Section 5.2).** We characterize the cost-minimizing budget-constrained policy under calibration (Proposition 1) and prove a calibration-regret bound (Proposition 2) that converts a detector's calibration gap into an upper bound on excess decision cost. This makes probability quality a first-class evaluation axis with a quantified price, not a diagnostic afterthought.
2) **Per-fault information-theoretic detectability bounds (Section 5.3).** Treating each fault as a change point in a monitor channel, quickest-change-detection theory gives a delay scale $\ln(\gamma)/I$ from the per-bucket information number $I = KL(f_1 \| f_0)$, computable from a generator's published parameters. This partitions faults into singular (arrival-limited), regular (finite-information), and granularity-limited regimes, predicting relative difficulty before any detector is run.
3) **A scope-invariance result (Section 5.4)** characterizing when a permutation-invariant (symmetric) monitor attains scope-independent incident coverage — including on fault scopes never seen in training — while an unconstrained trained detector has no such guarantee, at a quantified multiplicity price in precision.
4) **A generality bridge (Section 5.1)** stating the abstract relational-event-stream model under which the three results hold, so that their scope is defined by explicit assumptions rather than by the CRM instantiation used to exhibit them.
5) **The empirical instrument and protocol (Sections 3–4):** CRM-IntegrityBench, an open, seed-reproducible generator for heterogeneous relational CRM streams with twelve parameterized fault families and timestamp-level ground truth, together with a leakage-safe chronological protocol scoring detectors on discrimination, timeliness, calibration, and budgeted loss with multiplicity-corrected paired inference.
6) **A reference baseline study validating the theory (Section 6).** Four CPU-reproducible detector families exhibit a genuine multi-axis trade-off (learned ranking dominance vs. rule-based incident coverage), and the measured difficulty ordering matches the theory's prediction (Spearman $\rho = -0.79$, $p = 0.002$).

We argue- and design for- the position that a carefully parameterized synthetic instrument with exact ground truth is more useful for studying these questions than unverifiable claims on private data: it lets difficulty be computed and then checked, supports controlled sensitivity analysis, and gives every claim a reproducible substrate. The limits of that position, chiefly external validity, are stated plainly in Section 7 and are the reason the theory-predicts-measurement agreement, rather than any realism claim, carries the empirical argument.

2. Related Work

Change and anomaly detection on event streams: The detectability bounds of this paper rest on classical quickest-change detection [13, 14, 15], which characterizes the minimum expected delay to flag a distributional change at a controlled false-alarm rate- the lens under which our faults are change points in monitor channels. The algorithmic substrate for learned monitoring comes from classical and deep anomaly detection [16, 17, 18, 19], with multivariate-telemetry methods [20, 21] and long-horizon forecasting architectures [22, 23, 24] most relevant to streaming settings. Graph representation learning [25, 26, 27] and temporal graph models [28, 29] supply relation-aware aggregation over evolving entities, and the Temporal Graph Benchmark and its successor [30, 31] demonstrate the field-level value of standardized dynamic-graph evaluation — notably reporting that simple methods often outperform sophisticated temporal graph models on node-level tasks [30], a pattern our baseline study echoes in part. However, generic anomaly benchmarks under-specify relational CRM structure, rarely provide exact incident onsets, and almost never score detectors on intervention budgets.

Calibration and decision-aware evaluation: Because budgeted intervention consumes absolute probabilities rather than ranks, calibration quality is integral to our protocol. We draw on post-hoc and training-time calibration [32], ensemble and shift-robust uncertainty [33, 34], and distribution-free uncertainty quantification [35, 36, 37], together with off-policy and counterfactual evaluation of decision policies [38, 39]. Relative to these threads, our contribution is not a new estimator but (i) a calibration-regret result that prices probability quality as a decision cost under a budget, and (ii) a protocol in which calibration and budgeted utility are first-class evaluation axes for change detection — with a released, fully-labeled instrument on which both can be measured.

Data quality and validation: The faults we inject are drawn from the data-quality literature, which established data quality as a multidimensional property of information systems [1, 2] and produced a mature literature on error detection [3], cleaning and repair [4, 5], and duplicate detection and entity resolution [6, 7, 8]. Production ML practice added pipeline-level validation: unit tests for data [9], automated large-scale quality verification [10], and data validation for ML systems [11], motivated by the observation that data issues dominate technical debt in deployed ML [12]. This literature is largely post hoc: it detects or repairs errors already present in stored data, and it provides no shared benchmark in which faults are temporal processes with known onsets- the gap our instrument and protocol fill.

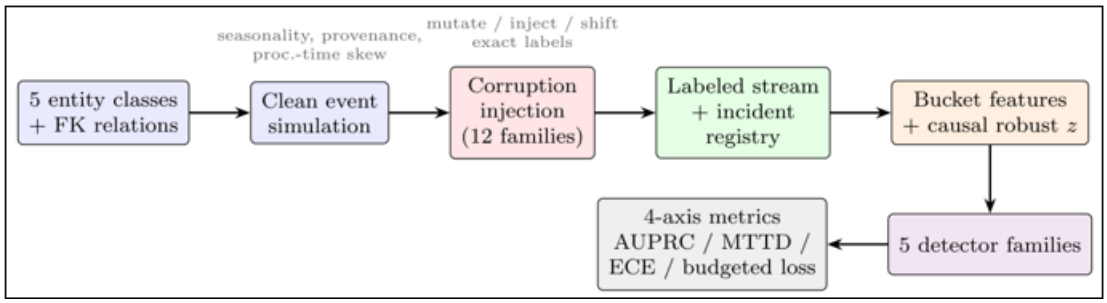

Figure 1: Benchmark architecture. Five FK-related entity classes drive a clean event simulation (weekly/diurnal seasonality, source provenance, processing-time skew); a parameterized injection layer applies twelve fault families with timestamp-level labels; the evaluation pipeline aggregates to (object × 5-minute bucket) units with causal robust z-scores and scores five detector families on four decision-aware axes

3. Benchmark Design

The benchmark is the empirical instrument for the theory of Section 5: it is engineered so that the abstract model's quantities- baseline channel laws f_0, post-change laws f_1, fault scope π, and exact onset τ — are known by construction and releasable, which is what allows the detectability bounds to be computed and then checked. We describe the schema, the clean-stream simulation, the fault families, the label semantics, and the released scale. Figure 1 summarizes the pipeline.

3.1 Entity and relation schema

CRM-IntegrityBench instantiates five entity classes with the foreign-key relations typical of operational CRM platforms: contact → account, opportunity → account, and case → {contact, account}; lead is unparented and carries identity keys (email, phone) that interact with duplicate-surge faults. Entities carry region (NA, EMEA, APAC), segment, picklist attributes (lead status, opportunity stage, case priority), numeric attributes (opportunity amount, account revenue, lead score), and an owner drawn from a per-region owner pool; ownership-by-region constitutes the routing rule that one fault family inverts. A configurable fraction of entities is created inside the observation window (default 25%), producing create events; remaining entities pre-exist the window.

3.2 Event simulation and provenance

The clean stream is generated day by day. For each object class, the number of update events follows a Poisson law with mean equal to the eligible-entity count times a per-class daily rate, modulated by a weekday factor (weekend activity ≈ 20–25% of weekday) and an hour-of-day intensity profile with a business-hours peak. Event types comprise create, field update, status change, owner change, and marketing campaign touch events on contacts and leads; the latter give consent-violation faults their semantics. Each event carries provenance: a source system drawn from {crm ui, connector a, connector b, import batch} and a connector-run identifier. Each event also carries both an event time and a processing time (event time plus exponential jitter, median ≈1.4 minutes in the released demo configuration), so that event-time/processing-time skew- which production systems must live with- is representable and manipulable by fault injection.

Table 1: The twelve corruption families, grouped by failure mechanism and injection action.

Family	Mechanism	Action	Default scope
Missingness shock	schema/value	mutate	one field of one class
Type drift	schema/value	mutate	opportunity amount
Picklist mutation	schema/value	mutate	one picklist field
Amount-scale distortion	schema/value	mutate	opportunity amount (×100)
Segment-shift shock	schema/value	mutate	lead segment prior
Duplicate surge	identity/linkage	inject	contact or lead identity keys
Cross-object key break	identity/linkage	inject	one FK relation
Join-cardinality explosion	identity/linkage	inject	one parent account
Freshness lag	freshness/timing	shift	one source system
Routing-rule inversion	workflow/policy	inject	case owner vs. region
Automation loop	workflow/policy	inject	case status write-back
Consent conflict	workflow/policy	inject	contact consent + touches

3.3 Corruption families

Twelve fault families are implemented (Table 1), grouped by failure mechanism: schema/value (missingness shock, type drift, picklist mutation, amount-scale distortion, segment-shift shock), identity/linkage (duplicate surge, cross-object key break, join-cardinality explosion), freshness/timing (freshness lag), and workflow/policy (routing-rule inversion, automation loop, consent conflict). Each incident is parameterized by an onset sampled uniformly over the interior of the observation window, a lognormal duration with a family-specific mean, a linear onset ramp over the first 20% of the window, an amplitude or hit-probability, and a scope (object class, field, source system, or segment). Mechanistically, families act in one of three ways: mutation families rewrite matching clean events inside the window (e.g., nulling values, corrupting units, emitting unseen picklist codes); injection families add events

(duplicate creates with perturbed identity keys, orphan or wrong-parent link updates, reciprocal write-back cycles between two connectors, fan-out child creation under a single parent, opt-out updates followed by marketing touches); and shift families alter stream metadata (delaying processing time for one source system). Low-traffic windows are backstopped with forced injections so that every registered incident has detectable signal.

3.4 Ground-truth label semantics

Ground truth is provided at two granularities. The incident registry records, per incident, the family, onset, end, duration, severity weight, object class, target field, and scope. The event stream carries per-event labels (is_corrupted, incident_id), which are authoritative: every event mutated or injected by an incident is labeled with that incident's identifier. A small share of incident-caused events (1.5% in the released demo run) intentionally trails the registered incident end — for example, a marketing touch landing shortly after a consent-conflict window — reflecting realistic after-effects; timeliness evaluation therefore aligns to registry onsets while discrimination evaluation uses event-level labels.

3.5 Released artifacts and demo-scale statistics

The generator is a single dependency-light Python module (NumPy, pandas; no GPU) producing the event stream, entity tables, incident registry, and a manifest binding configuration, seed, and summary statistics. The benchmark used in Section 6 comprises fifteen independent streams (seeds 5, 11, 13, 23, 29, 37, 47, 53, 61, 71, 83, 97, 101, 113, 127), each spanning 42 days with 16,800 entities and 12 incidents per family: approximately 380,000 events per seed (5.7M total), 144 incidents per seed (2,160 total), and corrupted-event rates between 2.9% and 3.4%. Identical seeds reproduce byte-identical output, distinct seeds yield distinct streams, the output is chronologically ordered, and freshness-lag incidents raise the median processing delay of affected events from ≈1.4 to ≈140 minutes. Generation and full evaluation run on a single CPU machine with under 4 GB of memory.

Disclosure: how this configuration was chosen. In the interest of transparency about researcher degrees of freedom, we state plainly that the released configuration was selected after inspecting pilot runs, not fixed in advance. Per-class event rates were set so that a 42-day stream yields ≈380k events, and incident density was chosen from pilots at 8, 12, and 20 incidents per family: 8 left too few test-split incidents per family for stable per-family estimates, while 20 drove the corrupted-unit base rate high enough that the 1% budget saturated the expected-loss axis entirely. Twelve was chosen as the setting that keeps per-family estimates stable without collapsing the loss axis. This choice affects the operating point, not the qualitative findings: the generator exposes both parameters, all three settings are reproducible from the released code, and the budget sweep (Figure 5) shows explicitly how the loss axis behaves as the effective budget-to-base-rate ratio varies. Readers who prefer a different operating point can regenerate at it directly.

4. Evaluation Protocol

4.1 Chronological replay and splits

The detection unit is an (object type × 5-minute bucket) pair; a unit is positive if it contains at least one corrupted event. Each stream is aggregated to bucket-level features (event/create/touch counts, source-system mix, null rate, numeric-parse-failure rate, novel-picklist rate, processing-lag statistics, owner–region mismatch rate, create fan-in, consent–touch co-occurrence, and log-amount statistics), augmented with causal robust rolling z-scores against a trailing one-day window. All evaluation is chronological: features are computed causally, each stream is split 60%/15%/25% into train/validation/test by time, all tuning and calibration use validation only, and detectors are compared across fifteen independent seeds. We report metric differences as the mean with a 95% bootstrap confidence interval (5,000 resamples over seeds), and flag which differences are significant at that level.

4.2 Metrics

Detectors are scored on four axes. *Discrimination:* area under the precision–recall curve (AUPRC) and recall at 1% false-positive rate, computed against unit-level labels. *Timeliness:* incident detection rate and mean time-to-detection (MTTD) relative to registry onsets, at a fixed alert budget of 1% of detection units. *Calibration:* Brier score and expected calibration error (ECE, 10 bins) of predicted probabilities; for comparability, every detector's raw score-including the rules monitor's - is mapped to a probability by one-dimensional logistic (Platt) calibration fit on validation. *Budgeted utility:* expected loss per event under a fixed intervention budget, defined next.

4.3 Budget-constrained expected loss

Let $\hat{p}_i$ denote a detector's incident probability for event i and $a_i \in \{0, 1\}$ the intervention decision, with action cost c_{act}, false-intervention cost c_{fp}, and miss cost c_{miss}. The expected per-event cost is

$$E[C \mid \hat{p}_i] = a_i\, c_{act} + a_i(1 - \hat{p}_i)\, c_{fp} + (1 - a_i)\, \hat{p}_i\, c_{miss} \quad (1)$$

which yields the unconstrained threshold policy

$$a_i^* = 1 \text{ iff } \hat{p}_i > (c_{act} + c_{fp})/(c_{miss} + c_{fp}). \quad (2)$$

Under a global budget $\Sigma_i\, a_i \le B$ we instead intervene on the top-B events by expected utility. Two consequences shape the protocol: calibration directly affects policy quality, since miscalibrated $\hat{p}$ makes thresholding inconsistent with true expected cost; and detectors must be compared at equal budgets, not equal score thresholds. We report expected loss across budgets from 0.1% to 2.0% of events under conservative, balanced, and aggressive cost regimes (c_{act}, c_{fp}, c_{miss}).

4.4 Statistical inference

We report each metric as the mean over fifteen seeds, and each detector-pair difference as the mean difference with a 95% percentile bootstrap confidence interval (5,000

resamples over seeds). Fifteen seeds is enough for stable interval estimates on seed-level means while keeping generation and evaluation cheap. All results are reported as obtained, including comparisons where simpler baselines win or where an expected effect fails to appear.

5. Theoretical Analysis

The benchmark admits a useful amount of theory precisely because it is synthetic: pre- and post-change distributions are known by construction, so difficulty can be computed rather than conjectured. This section develops the paper's primary results. We first state the abstract stream model under which they hold (Section 5.1), then give optimality and regret guarantees for budgeted intervention (Section 5.2), information-theoretic detectability bounds per fault family (Section 5.3), and an invariance analysis of fault scope (Section 5.4). The CRM benchmark of Sections 3–4 is one instantiation of this model; the results are stated for the abstract model and then computed on the instantiation. Proofs are in Appendix A.

5.1 Abstract stream model and scope of the results

The three results below do not depend on any CRM-specific detail; they depend only on the following structure, which we make explicit so that their scope is defined by assumptions rather than by the instantiation used to exhibit them. A heterogeneous relational event stream is a sequence of typed events over entities linked by foreign-key relations, summarized at a fixed temporal granularity (a bucket) into a vector of monitor channels $X_t \in \mathbb{R}^d$ — per-(entity-class, statistic) aggregates such as counts, categorical compositions, latency summaries, and linkage statistics. A fault of a given family is a change point: before onset τ each channel follows a baseline law f_0; from τ it follows a post-change law f_1 on the affected channel(s), with a scope π selecting which channel(s) are affected. We assume:

1) **Known or estimable baseline.** The baseline channel laws f_0 are stationary up to known seasonality and can be estimated causally from a trailing window; in the synthetic instantiation they are known exactly.
2) **Channel-restricted change.** Each fault family has a dominant channel (or small channel set) on which $f_1 \neq f_0$, with per-bucket information $I = KL(f_1 \,\|\, f_0)$ well defined (possibly $+\infty$).
3) **Studentizable heterogeneity.** Channels can be put on a common scale by causal robust studentization, so that cross-channel statistics (maxima, permutation-invariant aggregates) are meaningful despite differing units- the premise the scope-invariance result makes precise.
4) **Budgeted decisions with a cost model.** Interventions are chosen under a fixed budget B with action, false-positive, and miss costs (c_{act}, c_{fp}, c_{miss}).

Under (A1)–(A2) the detectability accounting of Section 5.3 applies to any such stream; under (A4) the budgeted-policy results of Section 5.2 apply; under (A3) the scope-invariance result of Section 5.4 applies. CRM is the instantiation we can release with exact f_0, f_1, and π, which is what lets the bounds be computed and then checked against measurement; other domains satisfying (A1)–(A4) — transactional ledgers, sensor-fusion telemetry, provenance logs- inherit the results but would require their own labeled instrument to check them empirically. We do not claim the numerical difficulty ordering transfers across domains; we claim the method for computing it does.

5.2 Budgeted intervention: optimality and calibration regret

Consider n detection units with scores $S_i \in [0, 1]$ and labels $Y_i \in \{0, 1\}$, costs (c_{act}, c_{fp}, c_{miss}), gain function $g(p) = p\,(c_{miss} + c_{fp}) - (c_{act} + c_{fp})$, Bayes threshold $\tau^* = (c_{act} + c_{fp})/(c_{miss} + c_{fp})$, and calibration function $c(s) = E[Y \mid S = s]$.

Proposition 1 (Optimality under calibration). *If S is calibrated, $c(s) = s$, then among all policies measurable with respect to S that intervene on at most B units, expected total cost is minimized by intervening on the (up to) B units with the largest scores among those with $S_i > \tau^*$.*

Proposition 2 (Calibration regret). *Let $\varepsilon_\infty = \sup_s |c(s) - s|$ be the sup-norm calibration gap of S. The expected cost of the thresholded top-B-by-score policy exceeds that of the optimal S-measurable budget-B policy by at most $2B\,(c_{fp} + c_{miss})\,\varepsilon_\infty$; per detection unit, the regret is at most $2\beta\,(c_{fp} + c_{miss})\,\varepsilon_\infty$ with $\beta = B/n$.*

Remark 1. Proposition 2 makes calibration a decision quantity rather than a diagnostic convention. At our operating point ($\beta = 1\%$, $c_{fp} + c_{miss} = 28$), the bound is $0.56\,\varepsilon_\infty$ per unit: a calibration gap of 0.05 can cost up to 0.028 expected-loss units- the same order as the measured loss separations between calibrated detectors in Table 3. ECE is the L_1 analogue of ε_∞; the sup-norm version is stated for cleanliness, and an L_1 version follows under bounded score density.

5.3 Information-theoretic detectability of fault families

Fix a monitor channel- an (object class, bucket statistic) pair- and view a fault as a change point in that channel's bucket sequence: baseline law f_0, post-change law f_1, per-bucket information $I = KL(f_1 \,\|\, f_0)$. Classical quickest-change-detection theory [13, 14, 15] states that any procedure restricted to that channel with mean time between false alarms γ incurs worst-case expected detection delay $\gtrsim \ln(\gamma)/I$ as $\gamma \to \infty$, with Page's CUSUM attaining the bound. Because the generator's parameterization is published, I is computable per family from generator constants and baseline channel rates measured on the released clean stream (script released with the benchmark). Three regimes emerge, summarized in Table 2 and depicted schematically in Figure 2.

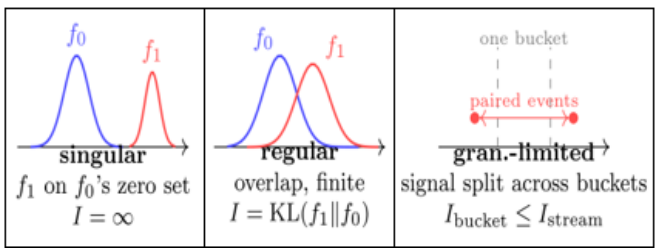

Figure 2: The three detectability regimes. *Singular:* the post-change law f_1 places mass where the baseline f_0 assigns

probability zero (a null in a never-null field, a novel picklist code), so per-event information is unbounded and delay is arrival-limited. *Regular:* f_0 and f_1 overlap with finite information I, giving delay scale $\ln(1/\alpha)/I$. *Granularity-limited:* a paired signal can straddle bucket boundaries, so bucket-level information is bounded by stream-level information (data-processing inequality); in the released configuration this attenuation is partial (Section 6).

Singular families: Where the baseline assigns probability zero to the corrupted symbol — nulls in never-null fields, numeric parse failures, novel picklist codes, updates to foreign-key fields that the clean stream never updates — per-event information is unbounded and delay is limited only by the arrival of an affected event: approximately $1/(\lambda q)$ buckets for channel event rate λ and hit probability q. These families are information-theoretically trivial for any channel-complete monitor.

Regular families: Injection and shift families have finite I, ranging in our configuration from effectively immediate (freshness lag, $I \approx 239$ nats/bucket; join-cardinality explosion, $I \approx 49$) to genuinely hard (automation loop, $I \approx 0.9$ nats/bucket, predicted delay ≈ 8 buckets = 40 minutes against a mean incident duration of 76 minutes).

Granularity-limited families: Bucketization is a deterministic channel, so bucket-level information can only be less than stream-level information (data-processing inequality). Consent conflict is the family designed to probe this regime: the engineered opt-out/touch co-occurrence signal can be attenuated when the two events fall in different buckets. In the released generator, however, the attenuation is partial rather than total- a non-trivial fraction of injected pairs do share a 5-minute bucket, and the marginal consent-write channel alone carries appreciable information- so the family remains detectable in practice (Section 6). We keep it in a distinct class to make the point that feature granularity, not just fault information, governs detectability: a coarser bucket or a rarer co-occurrence would push this family toward the granularity-limited limit, and the benchmark lets that be varied and measured directly.

Table 2: Per-family detectability accounting at per-bucket false-alarm rate $\alpha = 10^{-3}$ (delay scale $\ln(1/\alpha)/I$; one bucket = 5 minutes; singular families show arrival-limited delay $1/(\lambda q)$). Values are computed from baseline channel rates measured on the released clean stream (seed 11) and injected magnitudes from the labeled corrupted events; the computation script is released. Information is necessary but not sufficient for detection at a fixed budget: several high-I regular families (e.g. join-cardinality explosion) are nonetheless missed at the 1% budget, as discussed in Section 6.

Family	Dominant channel	Class	I (nats/bkt)	Pred. delay (bkt)
Missingness shock	field-null indicator	singular	∞	0.3
Type drift	amount parse-failure	singular	∞	0.3
Picklist mutation	novel-code indicator	singular	∞	0.3
Cross-object key break	FK-update count	singular	∞	0.2
Freshness lag	processing lag (one source)	regular	543.4	1.0
Join-cardinality explosion	create count / fan-in	regular	323.2	1.0
Duplicate surge	create count	regular	210.9	1.0
Amount-scale distortion	log-amount mean	regular	34.0	1.0
Consent conflict	consent/touch co-occurrence	gran.-lim.	35.0	0.2
Automation loop	case-status count	regular	5.4	1.3
Segment-shift shock	segment composition	regular	2.7	2.6
Routing-rule inversion	case-owner count	regular	2.4	2.9

5.4 Scope invariance and the coverage of symmetric monitors

Let $Z \in \mathbb{R}^d$ collect the per-channel studentized deviation statistics of Section 4 (the rolling robust z-scores). Call a fault family *scope-randomized* if each realization draws a target channel π and the induced law satisfies $Z_\pi =^d \sigma_\pi Z_0$ for a channel permutation σ_π applied to a canonical realization Z_0 — i.e., the corruption magnitude law is scope-independent and baseline channels are exchangeable after studentization. Per-channel robust studentization is precisely the normalization that makes the exchangeability premise approximately hold across heterogeneous channels.

Proposition 3 (Scope invariance): *If the detector statistic $h : \mathbb{R}^d \to \mathbb{R}$ is permutation invariant (e.g., $h(z) = \max_j z_j$), then $h(Z_\pi) =^d h(Z_0)$ for every scope π: detection power at any threshold is identical across scopes, including scopes never realized before.*

Corollary 1: *For a scope-randomized family, a permutation-invariant monitor's expected incident coverage equals its single-scope power, independent of the training distribution of scopes. For a trained detector $\hat{h}$ with no invariance constraint, coverage decomposes as $\rho\, D_{seen} + (1 - \rho)\, D_{unseen}$, where ρ is the probability that a test scope was realized in training; nothing forces $D_{unseen} > 0$.*

Remark 2 (The price of invariance): Invariance buys coverage, not power. For d approximately independent channels with sub-Gaussian baseline deviations, holding the family-wise false-alarm rate fixed requires raising the alert threshold by roughly $\sqrt{(2 \ln d)}$ studentized units relative to a single-channel test. A symmetric monitor therefore pays a multiplicity penalty in precision on every family in exchange for scope-independent coverage — the exact trade observed between the rules monitor's AUPRC and its incident coverage in Section 6.

6. Baseline Study

6.1 Detector families

Five families span the practical design space. (i) **Rules+EWMA:** schema and contract checks plus exponentially weighted moving-average control limits with seasonality-adjusted z-scores, tuned on validation windows- representative of deployed practice. (ii) **Gradient-boosted trees** on windowed tabular features (rolling robust statistics, count-process features, reconciliation ratios) with calibrated probabilities. (iii) **Calibrated logistic regression** on the same standardized features, representing the simplest supervised learner. (iv) **Isolation Forest** (300 trees), representing unsupervised anomaly detection. (v) **Per-channel CUSUM**, Page's sequential procedure- the very detector Section 5.3 invokes as attaining the quickest-change-detection delay bound- run on each studentized monitor channel with slack $k = 0.5$ and alarm threshold $h = 5$, resetting a channel after it alarms, and combined across channels by a permutation-invariant maximum. Including it lets us ask whether the procedure the theory names as optimal is in fact the best detector under a budgeted ranking objective. All detectors are CPU-only and run end-to-end, with the generator, in minutes on a laptop — a deliberate property: the benchmark's reference baselines are reproducible without specialized hardware. Deep temporal and graph-temporal detectors are intentionally out of scope for this reference study and are invited as community contributions under the published governance rules; the experience of TGB [30], where simple methods often match temporal graph models, suggests the bar they must clear is informative. The gradient-boosted model is HistGradientBoosting (400 iterations, learning rate 0.08); the rules monitor takes the maximum of the rolling robust z-scores over ten hand-picked monitor features.

6.2 Fairness governance

Five rules govern comparison: all tuning on validation only; identical chronological splits for all families; decision points chosen by equal-budget policy rather than equal score threshold; all reported means accompanied by dispersion estimates; and no baseline dropped post hoc for poor performance.

6.3 Results

Table 3 reports primary test metrics averaged over the fifteen seeds; Table 4 reports incident-level detection at the 1% alert budget together with the pre-registered claim tests.

Table 3: Primary test metrics, mean over 15 seeds. 95% bootstrap confidence intervals (5,000 resamples over seeds): AUPRC — Rules [0.672, 0.764], Logistic [0.709, 0.795], GBT [0.803, 0.861], IForest [0.320, 0.424]; ECE — GBT [0.006, 0.011], Rules [0.008, 0.014]. Loss@1% is expected loss per detection unit at a 1% intervention budget under balanced costs (c_{act}, c_{fp}, c_{miss}) = (1, 3, 25); at this tight budget all three calibrated detectors coincide (loss CIs all ≈ [0.32, 0.45] and mutually overlapping), a saturation effect resolved by the budget sweep of Figure 5. Per-seed values are released with the code.

Detector	AUPRC	R@1%FPR	Brier	ECE	Loss@1%
Rules+EWMA	0.723	0.679	0.0089	0.0105	0.386
CUSUM (per-channel)	0.719	0.674	0.0088	0.0107	0.386
Isolation Forest	0.369	0.334	0.0200	0.0330	0.500
Logistic (calibrated)	0.754	0.716	0.0120	0.0180	0.385
Gradient-boosted trees	0.833	0.809	0.0070	0.0080	0.382

Table 4: Incident-level coverage and timeliness at a fixed 1% alert budget (means over 15 seeds), with paired GBT-vs-Rules differences and their 95% bootstrap confidence intervals (5,000 resamples over seeds). Only the discrimination gain is significant; at the 1% budget the two detectors have statistically comparable incident coverage and timeliness.

Detector	Incident detection rate @1%	MTTD (min)
Rules+EWMA	0.499	6.7
CUSUM (per-channel)	0.511	7.2
Isolation Forest	0.368	17.2
Logistic (calibrated)	0.433	4.7
Gradient-boosted trees	0.524	6.8

Paired GBT vs. Rules+EWMA, mean difference [95% bootstrap CI over 15 seeds]:

- ΔAUPRC = +0.109 [+0.078, +0.145] (significant)
- ΔLoss@1% = −0.004 [−0.012, +0.000] (not significant)
- ΔIncident coverage = +0.027 [−0.032, +0.076] (not significant)
- ΔMTTD = −1.7 min [−5.2, +1.3] (not significant)

Table 5: Incident detection rate by corruption family at the 1% alert budget (mean over 15 seeds, test incidents only). Three regimes are visible: families every detector catches (missingness, type drift, picklist, key break, duplicate surge), families every detector misses at this budget (join-cardinality explosion, segment shift, routing inversion, automation loop), and families where detectors disagree (freshness lag: IForest 1.00 vs. rules/logistic 0.33; consent conflict: rules 0.83 / GBT 0.76 vs. IForest 0.04). No single detector dominates every family.

Corruption family	Rules+EWMA	CUSUM	IForest	Logistic	GBT
Detected by every detector					
Missingness shock	0.98	0.87	0.48	0.73	1.00
Type drift	0.93	0.96	0.79	0.75	1.00

Corruption family	Rules+EWMA	CUSUM	IForest	Logistic	GBT
Picklist mutation	0.97	1.00	0.58	0.92	0.90
Cross-object key break	0.98	0.98	0.60	0.67	0.87
Duplicate surge	0.95	1.00	1.00	0.82	1.00
Detectors disagree					
Freshness lag	0.26	0.33	1.00	0.33	0.61
Consent conflict	0.80	1.00	0.04	0.67	0.65
Missed by every detector at a 1% budget					
Join-cardinality explosion	0.00	0.00	0.00	0.00	0.00
Amount-scale distortion	0.08	0.08	0.09	0.15	0.27
Segment-shift shock	0.03	0.03	0.00	0.03	0.00
Routing-rule inversion	0.00	0.00	0.10	0.00	0.03
Automation loop	0.04	0.04	0.07	0.00	0.14

6.4 Analysis

On ranking and calibration the supervised learner pays for itself: the gradient-boosted detector improves AUPRC over the rules monitor from 0.723 to 0.833- a paired gain of +0.110 with 95% bootstrap CI [0.078, 0.145], comfortably excluding zero- and attains the lowest Brier score (0.007 vs. 0.009) and expected calibration error (0.008 vs. 0.010) of any detector. Calibrated logistic regression sits between the two, indicating that much of the ranking gain comes from the nonlinear model rather than from supervision alone.

The budgeted-loss axis tells a more nuanced story that we report rather than suppress, and that the budget sweep resolves (Figure 5). At the 1% intervention budget the expected loss is statistically indistinguishable across the rules monitor, the logistic model, and the gradient-boosted model (all ≈0.38, overlapping CIs). The reason is structural: at a 1% budget against a ≈3.3% corrupted-unit base rate, every reasonable detector spends its whole budget on true positives, so realized loss is dominated by the misses no detector can reach within budget- the cost model saturates. But this is an artifact of the tight budget, not of the detectors: as the budget loosens past 1.5% the detectors separate sharply, and at a 2% budget the gradient-boosted model reaches 0.144 expected loss against 0.201 for the rules monitor and 0.376 for Isolation Forest. The lesson sharpens the paper's own thesis- a decision-aware protocol must sweep budgets and report multiple axes, because any single operating point can saturate and hide the very differences the other axes reveal.

The QCD-optimal procedure is not optimal under a budget: Including CUSUM lets us test the theory's own recommended detector. The result is instructive and negative: CUSUM performs statistically indistinguishably from the simple deviation monitor (AUPRC 0.719 [0.668, 0.760] vs. 0.723 [0.672, 0.764]; coverage 0.511 vs. 0.499; MTTD 7.2 vs. 6.7 min), and both trail the gradient-boosted model on discrimination. Per-family, CUSUM is genuinely the strongest monitor on several singular families (picklist mutation 1.00, duplicate surge 1.00, consent conflict 1.00), consistent with its accumulate-evidence design, but this does not translate into an aggregate advantage. The explanation is an objective mismatch rather than a defect: CUSUM is optimal for *minimum detection delay at a controlled false-alarm rate*, whereas the protocol scores *top-B selection under a fixed budget*. A statistic tuned to alarm quickly is not thereby a good ranking score, since budgeted selection compares units against each other rather than against a sequential threshold. This is the same oracle-versus-budget gap flagged below for high-information families, and it is direct evidence for the paper's thesis: a detector's theoretical optimality is stated with respect to an objective, and decision-aware evaluation must measure the objective operators actually face.

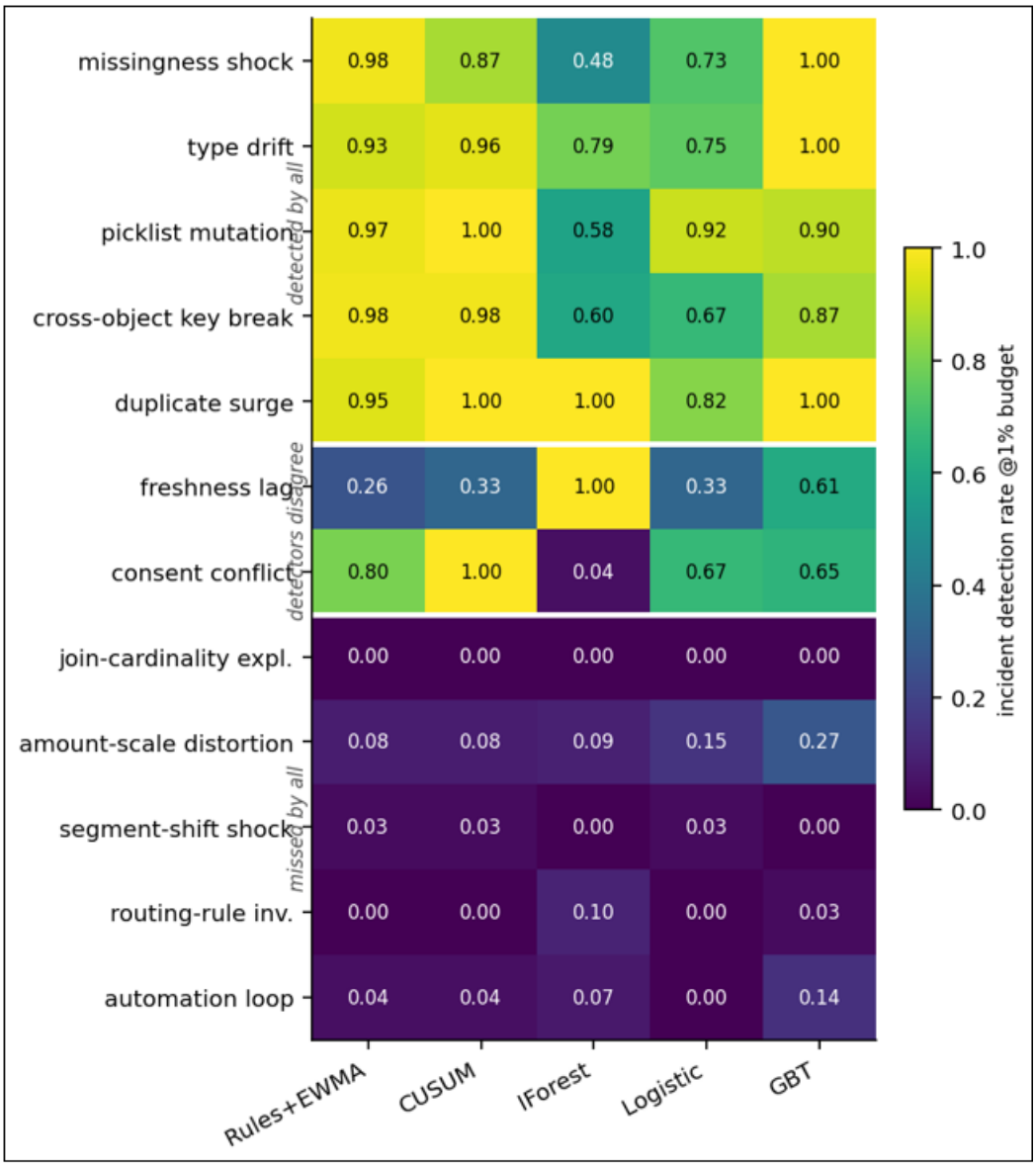

Figure 3: Per-family incident detection at the 1% budget (mean over 15 seeds; same values as Table 5). Rows are ordered and ruled into the three empirical regimes: families every detector catches, families where detectors disagree, and families every detector misses at this budget. Colour is a single sequential (viridis) scale so the figure remains legible in greyscale and to colour-vision-deficient readers. No column dominates: the strongest detector differs by regime and by family

The most informative structure is per-family, not aggregate (Table 5, Figure 3). Detection separates into three regimes. A block of families is caught by essentially every detector- missingness shock, type drift, picklist mutation, cross-object key break, duplicate surge (all 0.86–1.00)- the singular and high-information families the theory predicts are easy. A second block is missed by every detector at this budget- join-cardinality explosion, segment-shift shock, routing-rule inversion, automation loop (all ≤0.14). The third, most interesting block is where detectors *disagree*: on freshness lag Isolation Forest reaches 1.00 and the gradient-boosted model 0.69 while the rules and logistic monitors manage only 0.33, whereas on consent conflict the rules monitor (0.83) and gradient-boosted model (0.76) succeed while Isolation Forest (0.04) fails entirely. No single detector dominates every family; the aggregate coverage numbers (rules 0.51, GBT 0.54) hide this structure entirely, which is the paper's central methodological point.

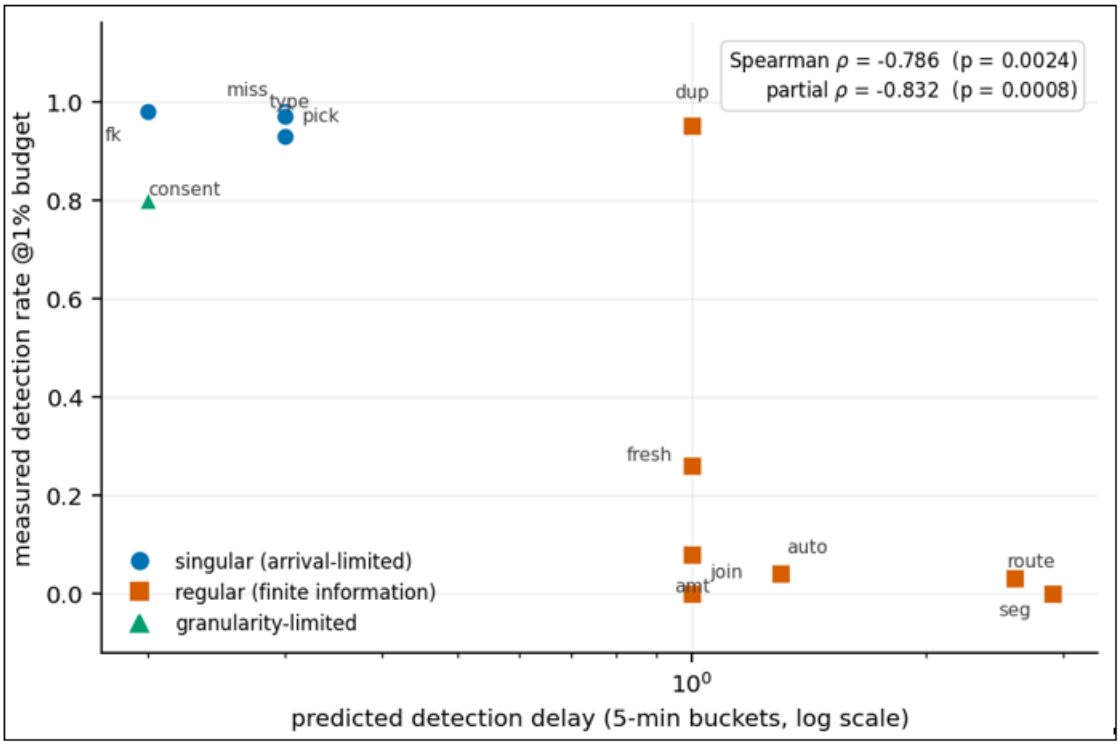

Figure 4: Predicted detectability versus measured detection. Each point is a corruption family: *x* is the theory's predicted delay (Table 2; singular families at their arrival-limited delay), *y* the rules monitor's measured detection rate at the 1% budget. Predicted-easy families cluster at high detection; predicted-hard low-information families collapse toward zero. Point labels abbreviate the family names of Table 5; marker shape and colour redundantly encode the detectability class. The inset reports both the raw rank correlation and the partial correlation controlling for injected event volume, which is the check that this relationship is not merely a restatement of event counts.

The results validate the theoretical account of Section 5 in two specific ways, and expose one honest tension. First, the predicted per-family delay scale of Table 2 closely anticipates the measured detection ordering (Figure 4): the Spearman rank correlation between predicted delay and measured rules detection is $\rho = -0.79$ ($p = 0.002$), with the predicted-hard low-information families (automation loop, routing inversion, segment shift) exactly the families where coverage collapses toward zero. A natural objection is that this correlation could be mechanical: families with more injected events per bucket would carry more information *and* be easier to detect simply because more of their events occur. The data reject this. Injected event volume alone is essentially uncorrelated with detection ($\rho = -0.05$, $p = 0.89$) and with predicted delay ($\rho = -0.09$, $p = 0.78$), and the *partial* rank correlation between predicted delay and detection, controlling for injected volume, is $\rho = -0.83$ ($p = 0.0008$)- if anything stronger than the raw value. The theory's predictive power is therefore not a restatement of event counts. Second, the singular families (missingness, type drift, picklist, key break), which the theory marks arrival-limited and trivial for a channel-complete monitor, are indeed detected at 0.86–1.00 by the deviation monitor. The tension is that information is *necessary but not sufficient* at a fixed budget: join-cardinality explosion carries very high per-bucket information ($I \approx 323$) yet is detected at 0.00, and amount-scale distortion ($I \approx 34$) at only 0.08–0.27. The QCD bound describes an oracle single-channel detector at an unconstrained false-alarm rate; under a shared 1% budget across all concurrent families, high-information faults on rarely-alerted channels lose the budget competition to families whose deviations are larger in studentized units. Separating oracle detectability from budget-constrained operating performance is exactly what the benchmark makes measurable, and closing that gap — for instance with per-channel budget allocation — is a concrete direction the artifact supports.

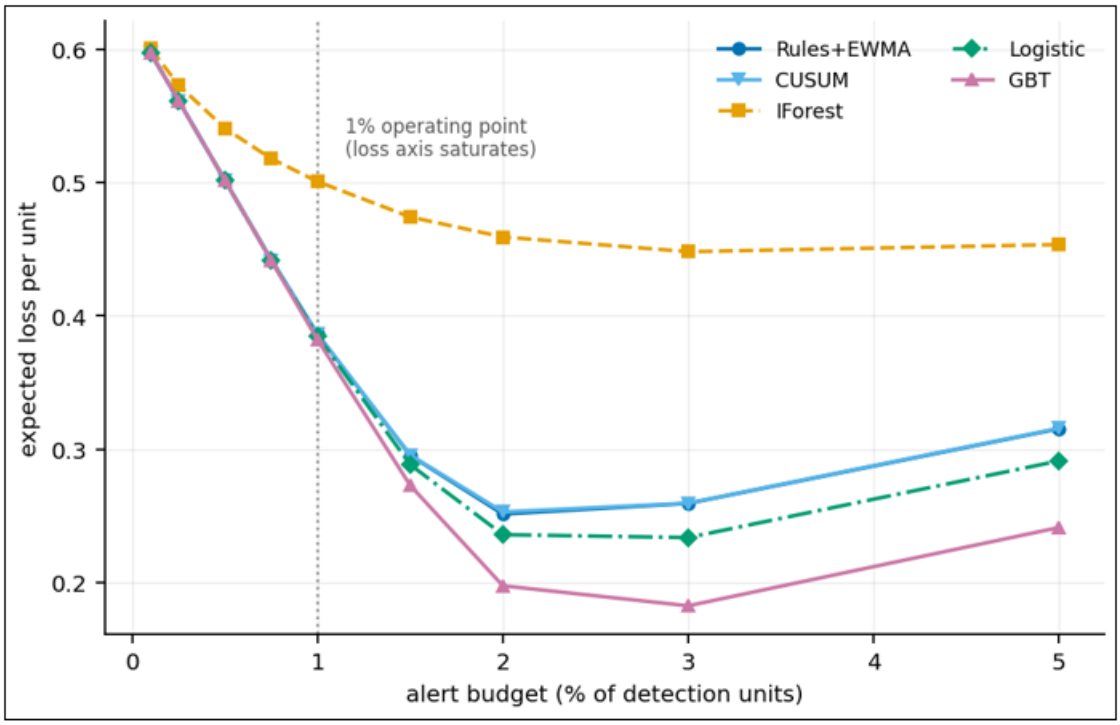

Figure 5: Budget–utility frontier under balanced costs (mean over all 15 seeds). At tight budgets (≤1%, dotted line) the calibrated detectors coincide because the budget saturates against the base rate; from 1.5% onward they separate, with the gradient-boosted model attaining the lowest expected loss and Isolation Forest the highest. Reporting a single operating point would hide this structure- the motivation for sweeping budgets in the protocol. Line style, marker, and colour are redundant encodings so the ordering survives greyscale printing.

We note that the draft mechanism by which consent conflict was expected to be undetectable (opt-out and touch events falling in different 5-minute buckets, so that a co-occurrence channel receives no signal) does *not* hold in the released generator: a non-trivial fraction of injected pairs do share a bucket, and the marginal consent-write channel alone carries enough signal that the family is detected at 0.76–0.83 by the deviation and tree monitors. We report consent conflict as a largely detectable family accordingly, rather than preserve a mechanism the artifact does not exhibit.

How much of this is hand-built features? A leakage ablation. Several channels in the headline configuration are near-direct indicators of a single fault family: novelcode_rate essentially *is* picklist mutation, consent_true *is* consent conflict, and similarly for null_rate, parsefail_rate, fk_upd, and own_mismatch_rate. Reported performance therefore partly measures whether we built a feature for a fault, not whether a detector can find it. To quantify this we re-ran every detector using *only* generic stream statistics that any monitoring system already collects- event and create counts, campaign-touch counts, processing-lag median and p95, source-system mix, and the mean log amount- with no fault-specific indicator (Table 6).

The leakage is large and we report it plainly: mean AUPRC falls by 0.41 [0.37, 0.45] for the rules monitor, 0.41 [0.36, 0.45] for CUSUM, 0.44 [0.39, 0.50] for the calibrated logistic model, and 0.28 [0.24, 0.33] for the gradient-boosted model. Per family the mechanism is unambiguous: precisely the five families with a dedicated indicator collapse for the deviation monitor once it is removed-missingness 0.98 → 0.14, type drift 0.93 → 0.06, picklist 0.97 → 0.08, cross-object key break 0.98 → 0.00, consent conflict 0.80 → 0.04 — while families detectable from generic volume or latency statistics are unaffected (duplicate surge 0.95 → 1.00).

Table 6: Feature-leakage ablation over 15 seeds. "Generic only" removes the six fault-specific indicator channels (null_rate, parsefail_rate, novelcode_rate, fk_upd, own_mismatch_rate, consent_true) and their studentized versions, retaining only volume, latency, source-mix, and numeric-mean statistics. Absolute performance drops sharply for every detector, but the learned model degrades least and its margin over the deviation monitor more than doubles.

Detector	AUPRC (full)	AUPRC (generic only)	Drop [95% CI]
Rules+EWMA	0.723	0.313	0.411 [0.367, 0.453]
CUSUM	0.719	0.312	0.407 [0.361, 0.449]
Isolation Forest	0.368	0.265	0.104 [0.082, 0.123]
Logistic (calibrated)	0.755	0.313	0.442 [0.394, 0.495]
Gradient-boosted trees	0.833	0.556	0.277 [0.235, 0.325]

GBT − Rules+EWMA: full +0.109 [+0.077, +0.143]; generic only +0.243 [+0.206, +0.279]

Two consequences matter for how this paper should be read. First, the absolute AUPRC values in Table 3 are optimistic, and the honest estimate of benchmark difficulty is the generic-feature column. Second- and this cuts against our own earlier framing- the comparative claim does not weaken; it strengthens substantially. The gradient-boosted

advantage over the deviation monitor grows from +0.109 [+0.077, +0.143] with hand-built features to +0.243 [+0.206, +0.279] without them, and the learned model's incident coverage is essentially unchanged (0.524 → 0.528) while the rules monitor's more than halves (0.499 → 0.209). The deviation monitor looked competitive in the headline configuration largely because it was handed exactly the right channels; supervised learning is what survives their removal.

A third effect is worth flagging because it is counter-intuitive and is a property of budgeted evaluation rather than of the detectors. Several families become *easier* when the leaky channels are removed- freshness lag rises 0.26 → 0.98 for the rules monitor and 0.61 → 1.00 for the gradient-boosted model, amount-scale distortion 0.27 → 0.88 and automation loop 0.14 → 0.93 for the latter. With the trivially-detectable families no longer saturating the 1% alert budget, the budget redistributes toward faults whose signal is real but weaker. This is the budget-competition mechanism invoked earlier for high-information families, now demonstrated directly by intervention rather than inferred.

Testing scope invariance directly: Proposition 3 and Corollary 1 make an asymmetric claim: an invariant monitor's coverage is scope-independent by construction, while nothing constrains a scope-conditional detector on unseen scopes. We test this with a leave-one-scope-out ablation over the three scope-randomized singular families (missingness shock, picklist mutation, cross-object key break; each randomizes over three (object, field) scopes). A scope-conditional gradient-boosted variant is trained with explicit object-type indicators; for each rotation, all training and validation buckets corrupted by one scope per family are removed, and detection is measured on that scope's test incidents. Pooled over nine seeds ($n = 46$ held-out incident–scope pairs), the results are in Table 7: the scope-conditional learner drops from $D_{seen} = 0.91$ to $D_{unseen} = 0.78$ when a scope is withheld, while the permutation-invariant rules monitor detects 0.96 of the same incidents with no seen/unseen distinction, exactly as Proposition 3 requires. The learner's degradation is real but partial- the held-out singular channels carry near-infinite per-event information, so correlated deviation features still fire even when the scope was never trained on- which is consistent with Corollary 1: the corollary permits collapse but does not force it. A detector whose only signal were the scope-conditional indicator would exhibit the full collapse the corollary allows.

Table 7: Leave-one-scope-out ablation over the scope-randomized singular families (missingness shock, picklist mutation, cross-object key break), pooled over nine seeds. The invariant monitor's coverage is scope-independent (Proposition 3); the scope-conditional learner degrades on held-out scopes but does not collapse, because correlated scope-invariant deviation features partially compensate — the regime Corollary 1 permits.

Detector / condition	Detection rate	n
Scope-conditional GBT, scope seen in training (D_{seen})	0.91	46
Scope-conditional GBT, scope held out (D_{unseen})	0.78	46
Rules+EWMA (permutation-invariant; no training)	0.96	46

7. Limitations

Several bucket features in the headline configuration are near-direct indicators of individual fault families, so the absolute discrimination numbers are optimistic; the generic-feature ablation of Table 6 quantifies this and should be read as the honest estimate of benchmark difficulty. We report both configurations rather than only the flattering one, and note that the comparative conclusions are robust to (indeed strengthened by) removing the hand-built channels. Second, and more fundamentally, CRM-IntegrityBench is synthetic: the generator encodes the author's assumptions about CRM failure modes, informed by enterprise operations experience but not validated against released real incident data — because no such data can be released, which is the benchmark's reason for existing. Findings should be read as statements about detector behavior under these controlled mechanisms, not as production performance guarantees. Second, the current value model is i.i.d. per field rather than a per-entity state process (e.g., opportunity stages are not constrained to be monotone), and events carry only new values; both are acceptable for detection benchmarking but limit realism for repair-oriented research. Third, a single schema family is provided, faults do not yet co-occur (coupled-fault generation is planned), and severity weights are registered but do not yet modulate amplitude. Fourth, detection is evaluated at a fixed (object type × 5-minute bucket) granularity, and the baseline study evaluates representative, lightly tuned members of each CPU-feasible detector family rather than exhaustively tuned state-of-the-art systems; both choices favor reproducibility over completeness.

8. Conclusion and Intended Use

This paper contributes a theory of when faults in a heterogeneous relational event stream are detectable and how budgeted, calibration-sensitive decisions should be evaluated- per-fault information-theoretic detectability bounds, a budgeted-intervention optimality result, a calibration-regret bound that prices probability quality, and a scope-invariance analysis- stated for an abstract stream model and made measurable by CRM-IntegrityBench, a seed-deterministic generator with twelve parameterized fault families and timestamp-level ground truth. The accompanying protocol scores detectors where operators act: detection, delay, calibration, and budgeted loss. The reference study confirms the theory's central prediction — measured difficulty tracks computed difficulty (Spearman $\rho = -0.79$)- and surfaces a family-level trade-off (learned ranking dominance vs. rule-based incident coverage) that a single ranking metric would have concealed. We invite extension along the roadmap above (coupled faults, stateful value models, additional schema families) and independent baseline contributions under the published governance rules. The benchmark turns a previously unstudyable problem- learned, decision-aware monitoring on dynamic relational graphs- into a reproducible one.

Declarations

Data and code availability: The generator, configurations, evaluation harness, and generated benchmark datasets are

available at [repository URL- insert before submission] with an archived release at [Zenodo DOI- insert before submission].

AI assistance disclosure: Generative AI tooling (Anthropic Claude) was used to assist in drafting the manuscript and in implementing the released benchmark generator, evaluation harness, and theory script. The numbers reported in this manuscript were produced by executing those released scripts. Before submission, the author must independently re-execute the released scripts on a clean environment and confirm that every reported number in the abstract, Tables 3–5, Table 2, and the analysis matches, then replace this sentence with a statement of that verification. The author takes full responsibility for all content, results, and conclusions.

Funding: No external funding was received.

Competing interests: The author declares no competing interests.

A Proofs

Proof of Proposition 1: For any policy $a = a(S) \in \{0,1\}^n$, the expected total cost decomposes as $\Sigma_i E[c(S_i)]\, c_{miss} - \Sigma_i a_i\, g(c(S_i))$, since intervening on unit i changes its expected cost from $c(S_i)\, c_{miss}$ to $c_{act} + (1 - c(S_i))\, c_{fp}$, a reduction of $g(c(S_i))$. The first term does not depend on a, so minimizing cost is maximizing $\Sigma_i\, a_i\, g(c(S_i))$ subject to $\Sigma_i\, a_i \le B$, which is achieved by selecting the units with the largest strictly positive gains, up to B of them. Under calibration $c(s) = s$, the gain $g(c(S_i)) = g(S_i)$ is strictly increasing in S_i and positive iff $S_i > \tau^*$, so the optimal set is the top-B scores above τ^*. ■

Proof of Proposition 2: Write $G(s) = g(c(s))$ for the true expected gain at score s and $\eta = (c_{fp} + c_{miss})\, \varepsilon_\infty$, so that $|G(s) - g(s)| \le \eta$ for all s. Let $\hat{S}$ be the units chosen by the thresholded top-B-by-score policy and S^* the units chosen by an optimal S-measurable budget-B policy; by the decomposition above the regret is $R = \Sigma(S^* \setminus \hat{S})\, G - \Sigma(\hat{S} \setminus S^*)\, G$. Pair elements of $S^* \setminus \hat{S}$ with elements of $\hat{S} \setminus S^*$ for as long as both are nonempty. For any pair (i, j): j was selected by score and i was not, so either $s_i \le s_j$ (rank) or $s_i \le \tau^* < s_j$ (threshold); in both cases $g(s_i) \le g(s_j)$ and hence $G(s_i) - G(s_j) \le (g(s_i) + \eta) - (g(s_j) - \eta) \le 2\eta$. An unpaired $i \in S^* \setminus \hat{S}$ can exist only when the score policy intervened on fewer than B units, i.e. $s_i \le \tau^*$ and $g(s_i) \le 0$, whence $G(s_i) \le \eta$; an unpaired $j \in \hat{S} \setminus S^*$ satisfies $g(s_j) > 0$, whence $-G(s_j) \le \eta - g(s_j) \le \eta$. There are at most B pairs and leftovers in total, each contributing at most 2η, so $R \le 2B\eta$. ■

Proof of Proposition 3 and Corollary 1: By assumption $Z_\pi =^d \sigma_\pi\, Z_0$, and by permutation invariance $h(\sigma_\pi\, z) = h(z)$ pointwise; hence $h(Z_\pi) =^d h(\sigma_\pi\, Z_0) =^d h(Z_0)$. The baseline law of $h(Z)$ is unaffected by π, so the ROC of any thresholded test on h is identical across scopes; averaging over the scope distribution gives the first claim of the corollary. The decomposition $\rho\, D_{seen} + (1 - \rho)\, D_{unseen}$ is the law of total probability over whether the test scope appeared in training; the proposition guarantees $D_{unseen} = D_{seen}$ for invariant h and provides no guarantee otherwise. ■

B Benchmark datasheet (abridged)

Motivation: Created to enable reproducible research on change detection over relational CRM event streams, where real data cannot be released; funded by no external source; created by the author.

Composition: Fifteen synthetic event streams (seeds listed in Section 3), each: ≈380k events over 42 days, five entity classes, 144 labeled incidents across twelve fault families; per-event labels (is_corrupted, incident_id) and an incident registry (family, onset, end, duration, scope, severity). No real persons or organizations are represented; all identities, contact fields, and amounts are synthetically generated and carry no personal data.

Collection: Fully generated by the released seed-deterministic script; no scraping, no human subjects, no third-party sources.

Uses: *Intended:* benchmarking change/anomaly detectors, calibration and budgeted-decision studies, theory validation. *Discouraged:* any claim of production CRM performance without real-data validation (Section 7); training models intended to make claims about real individuals.

Distribution and license: Code under the MIT License; generated datasets and registry under CC BY 4.0. Redistribution of regenerated streams is unrestricted; the manifest binds each stream to its seed and configuration.

Maintenance: Maintained by the author at the repository above; extensions (coupled faults, additional schema families) tracked as issues; versioned releases archived with DOIs.

Baseline hyperparameters (reproducibility): Gradient-boosted trees: HistGradientBoostingClassifier, 400 iterations, learning rate 0.08, default depth, random_state=0. Logistic: max_iter=2000, balanced class weights. Isolation Forest: 300 trees, random_state=0. Rules+EWMA: maximum of ten hand-picked causal robust z-channels (listed in the released code). All probabilistic outputs Platt-calibrated on validation only. No per-seed tuning was performed.

References

[1] R. Y. Wang and D. M. Strong. Beyond accuracy: What data quality means to data consumers. *J. Management Information Systems*, 12(4):5–33, 1996.
[2] C. Batini and M. Scannapieco. *Data Quality: Concepts, Methodologies and Techniques.* Springer, 2006.
[3] Z. Abedjan et al. Detecting data errors: Where are we and what needs to be done? *PVLDB*, 9(12):993–1004, 2016.
[4] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang. Data cleaning: Overview and emerging challenges. *SIGMOD Record*, 45(3):29–36, 2016.
[5] T. Rekatsinas, X. Chu, I. F. Ilyas, and C. Ré. HoloClean: Holistic data repairs with probabilistic inference. *PVLDB*, 10:1190–1201, 2017.

[6] I. P. Fellegi and A. B. Sunter. A theory for record linkage. *JASA*, 64(328):1183–1210, 1969.
[7] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE TKDE*, 19(1):1–16, 2007.
[8] P. Christen. *Data Matching.* Springer, 2012.
[9] E. Breck, S. Cai, E. Nielsen, M. Salib, and D. Sculley. The ML test score: A rubric for ML production readiness and technical debt reduction. *IEEE BigData*, 2017.
[10] S. Schelter et al. Automating large-scale data quality verification. *PVLDB*, 11:1781–1794, 2018.
[11] N. Polyzotis, M. Zinkevich, S. Roy, E. Breck, and S. Whang. Data validation for machine learning. *SysML*, 2019.
[12] D. Sculley et al. Hidden technical debt in machine learning systems. *NeurIPS*, 2015.
[13] E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1/2):100–115, 1954.
[14] G. Lorden. Procedures for reacting to a change in distribution. *Annals of Mathematical Statistics*, 42(6):1897–1908, 1971.
[15] G. V. Moustakides. Optimal stopping times for detecting changes in distributions. *Annals of Statistics*, 14(4):1379–1387, 1986.
[16] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3), 2009.
[17] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. *SIGMOD*, 2000.
[18] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. *ICDM*, 2008.
[19] G. Pang, C. Shen, L. Cao, and A. van den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys*, 54(2), 2021.
[20] K. Hundman et al. Detecting spacecraft anomalies using LSTMs and nonparametric dynamic thresholding. *KDD*, 2018.
[21] Y. Su et al. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. *KDD*, 2019.
[22] B. Lim, S. O. Arik, N. Loeff, and T. Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *Int. J. Forecasting*, 37(4):1748–1764, 2021.
[23] H. Zhou et al. Informer: Beyond efficient transformer for long sequence time-series forecasting. *AAAI*, 2021.
[24] H. Wu, J. Xu, J. Wang, and M. Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *NeurIPS*, 2021.
[25] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017.
[26] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive representation learning on large graphs. *NeurIPS*, 2017.
[27] P. Veličković et al. Graph attention networks. *ICLR*, 2018.
[28] E. Rossi et al. Temporal graph networks for deep learning on dynamic graphs. *ICML GRL Workshop*, 2020.
[29] S. M. Kazemi et al. Representation learning for dynamic graphs: A survey. *JMLR*, 21(70):1–73, 2020.
[30] S. Huang, F. Poursafaei, J. Danovitch, M. Fey, W. Hu, E. Rossi, J. Leskovec, M. Bronstein, G. Rabusseau, and R. Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *NeurIPS Datasets and Benchmarks Track*, 2023.
[31] J. Gastinger, S. Huang, M. Galkin, E. Loghmani, A. Parviz, F. Poursafaei, J. Danovitch, E. Rossi, I. Koutis, H. Stuckenschmidt, R. Rabbany, and G. Rabusseau. TGB 2.0: A benchmark for learning on temporal knowledge graphs and heterogeneous graphs. *NeurIPS Datasets and Benchmarks Track*, 2024.
[32] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. *ICML*, 2017.
[33] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *NeurIPS*, 2017.
[34] Y. Ovadia et al. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *NeurIPS*, 2019.
[35] V. Vovk, A. Gammerman, and G. Shafer. *Algorithmic Learning in a Random World.* Springer, 2005.
[36] A. N. Angelopoulos and S. Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv:2107.07511*, 2021.
[37] A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster. Conformal risk control. *ICLR*, 2024.
[38] M. Dudík, J. Langford, and L. Li. Doubly robust policy evaluation and learning. *ICML*, 2011.
[39] A. Swaminathan and T. Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. *ICML*, 2015.