SSM-Audit

Audit Finance Pack (AR, Refunds, Forecast deltas), stamped logs

Status: Public Research Release (v1.4)
Date: 22 October 2025
Caution: Research/observation only. Not for critical decision-making.

When the numbers look great right up to the cliff — can you see the cliff?

→ **Yes.** SSM-Audit adds a small, bounded stability lane beside every KPI you already track: `x := (m, a)` with `a in (-1,+1)` and collapse parity `phi((m,a)) = m`. Your reported number `m` stays **exactly the same**; the lane `a` emits a simple band (**A++**, **A+**, **A0**, **A-**, **A--**) that answers one question: **How sturdy is today's number?** Dashboards stop giving false comfort: revenue can be up while stability slides from **A+ → A0** — a quiet early-warning that refunds, timing, or coverage is eroding **before** the quarter closes.

When platform and bank totals tie out perfectly, can you still see the daily drift that predicts next month's cash surprise?

→ **Yes.** Reconciliations move beyond "match / no-match." SSM-Audit highlights **day-level agreement quality** between systems, so tiny timing slips and growing tails become visible long before cash tightens. Think of it as a **read-only early-warning layer** on top of the same ledgers: publish the lane beside `m := collected/issued`, add simple bands, and watch for **stability debt** building underneath apparently healthy ratios.

When an acquisition target's deck sparkles, can you see the stability debt beneath the metrics before you sign?

→ **Yes.** Data rooms and monthly cohorts can look pristine; **stability lanes** reveal whether those metrics have been calm and repeatable or noisy and propped up. You still see the same headline numbers, but now you also see their **day-to-day sturdiness** — so pricing, covenants, and integration plans are set **with eyes open**.

What SSM-Audit is (in one line)

A **conservative extension** to finance numbers that **never changes `m`** yet shows, in one glance, **how reliable each number is today**.

How it lands (without disruption)

- **No rewrites.** Keep `m` and your existing SQL, joins, and reports; just publish the **lane** and a **band** beside **3–5 KPIs** (e.g., AR/Collections, Refunds/Chargebacks, Forecast vs Actuals).
- **Plain labels.** Executives get **A++/A+/A0/A-/A--**, not math. Teams act **early**, not loudly.
- **Stamped logs (optional).** Each daily file/roll-up can carry a one-line ASCII stamp for tamper-evident replay:
  `SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain`
  with `chain_0 := "0"*64` and `chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )`.
- **Supplemental, non-GAAP disclosure.** The lane is a bounded **risk indicator** beside classical KPIs; **`phi((m,a)) = m`** ensures reported magnitudes **never change**.
- **Privacy by design.** The lane `a` is derived from KPI math and aggregates only; **no PII** is ever carried in stamps or manifests.

What to expect in two weeks

- **Same numbers, clearer truth.** Fewer false alarms, calmer reviews, earlier course-corrections.
- **Risk priced better. Stability debt** shows up before it hits cash.
- **Evidence you can file.** A lightweight **evidence pack** (conformance sheet, minimal manifest, optional stamps) ready for audit.
- **Scale when ready.** Start read-only; extend to more KPIs after the first visible wins.

Ease of adoption — know your stability in minutes (see §7.11 Mini CLI)

- **Minimal CLI, zero rewrites.** Keep your KPIs exactly as-is; the CLI adds `a` and `band` beside `m` with collapse parity `phi((m,a)) = m`.
- **One command to run.** `python ssm_audit_mini_calc.py pilot.csv out.csv --build_id demo --plot_kpi Revenue_actual` → updated CSV, optional alerts, and a chart with band guides.
- **Deterministic & bounded.** Order-invariant fuse `a_out := tanh( (SUM w*atanh(a)) / max(SUM w, eps_w) )`, hysteresis with band gates (`promote` if `delta_a >= +0.05`, `demote` if `delta_a <= -0.05`), and a manifest-backed `knobs_hash` for reproducibility.
- **Audit-ready by default.** Optional stamped logs (`SSMCLOCK1|...|sha256(file)|chain`) and a conformance snippet produce an evidence pack with no PII.
- **From pilot to portfolio.** Start with 3–5 KPIs; extend after visible wins. The same CLI and CSV schema scale across AR, refunds, forecast residuals, and cash schedule.

TABLE OF CONTENTS

SECTION 0 — Executive Summary

0A — Introduction (purpose & scope)

Purpose. Give every finance KPI a simple, reliable way to say how sturdy it is today — **without changing the number itself**. In SSM-Audit, each metric is carried as a pair `x := (m, a)`, where `m` is the classical value you already report and `a in (-1,+1)` is a bounded **stability lane**. By design, the classical number never changes: **collapse parity** `phi((m,a)) = m`. Executives see the same `m`, plus a clear band (**A++**, **A+**, **A0**, **A-**, **A--**) that answers: **is this number built on rock or sand?**
Disclosure posture. The stability lane is a **supplemental, non-GAAP** risk indicator published beside existing KPIs; ledgers and controls remain unchanged; **no PII** is carried in stamps or manifests.

Scope (initial modules).

• **AR / Collections health.** Publish the lane beside `m := collected/issued` to expose long-tail weakness early.
• **Refunds / Chargebacks reconciliation.** Show **daily agreement quality** between platform and bank, not just month-end ties.
• **Forecast vs Actuals stability.** Reveal "jitter" in otherwise on-plan series so noise is fixed **before** it becomes misses.
Future modules (credit, revenue quality, cash predictability, opex control, cohort durability) inherit the same semantics.

Object model (what changes, what doesn't).

• **Numbers stay numbers:** `phi((m,a)) = m`. Ledgers, SQL, joins, and reports remain intact.
• **Stability is bounded and comparable:** `a in (-1,+1)` enables sort, band, and trend without scale tricks.
• **Composition is disciplined:** lane values are combined with declared, deterministic rules so **batch == stream == shuffled** within a fixed epsilon.
• **Bands are plain labels:** **A++**, **A+**, **A0**, **A-**, **A--** convert `a` into action **without math in the room**.

What you publish (minimal). For each KPI row/file, add:
`a` (stability lane), `band` (A++/A+/A0/A-/A--), `knobs_hash` (declared settings fingerprint), `build_id` (run tag), and optionally a **stamp** (tamper-evident replay line):
`"SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain"`.

Adoption path (read-only first).

1. Publish the lane beside **3–5** core KPIs; **no thresholds change** on day one.
2. Add bands to stop flicker and focus attention (**A++/A+/A0/A-/A--** with **gentle hysteresis**).
3. Stamp daily exports (optional) for **replayable provenance**.
4. Run **conformance checks** so teams trust that **batch == stream == shuffled** and that `phi((m,a)) = m` holds through every ETL hop.

Not a replacement. SSM-Audit does **not** replace controls, ledgers, PKI, or policy. It is a **conservative extension**: same `m`, plus a bounded signal about reliability that helps you act earlier and argue less.

Time to value. Hours to pilot with CSVs; **two weeks** to feel fewer false alarms, calmer reviews, and earlier course-corrections — with the **exact same reported numbers**.

0B — What people see, what we lock (defaults)

On every KPI you already track, publish **one extra lane** and **one label**.
• **Lane (inline):** `x := (m, a)` with `a in (-1,+1)` and collapse parity `phi((m,a)) = m`.
• **Label (band):** **A++**, **A+**, **A0**, **A-**, **A--** — a plain read on how **sturdy** today's number is.
• **Dashboard:** keep the usual line for `m`; show a thin underlay or a side badge for `a` and its band.

Recommended band thresholds (finance-ready, gentle by default).

• **A++** : `a >= 0.75`
• **A+** : `0.50 <= a < 0.75`
• **A0** : `0.25 <= a < 0.50`
• **A-** : `0.10 <= a < 0.25`
• **A--** : `a < 0.10`
Hysteresis: promote only if `delta_a >= 0.05`; demote only if `delta_a <= -0.05`. This stops flicker while keeping sensitivity to real change.

Minimal row fields to add (read-only pilot).

`time, kpi, m, a, band, knobs_hash, build_id, stamp`
• `m` = your classical value (**unchanged** by SSM-Audit).
• `a` = stability lane in `(-1,+1)`.
• `band` = A++/A+/A0/A-/A-- from the thresholds above.
• `knobs_hash` = fingerprint of declared settings.

• `build_id =` short tag for the run.
• `stamp (optional) =` one-line ASCII provenance for replay:
`"SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain"`

Minimal manifest (kept beside the CSV; plain ASCII).

```
[SSM_AUDIT_MANIFEST]
version = 1.0
build_id = <short tag>
date_utc = <YYYY-MM-DD>
eps_a = 1e-6
eps_w = 1e-12
gamma = 1.0
division_policy = strict
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}
mappings = {AR:..., Refunds:..., Forecast:...}
conformance_checksum = <hex>
```

Two rules are non-negotiable: **`phi((m,a)) = m`** and **order-invariance** (batch == stream == shuffled within a fixed epsilon).

Day-one scope (start with 3–5 line items; numbers stay identical).

• **AR / Collections:** publish the lane beside `m := collected/issued`.
• **Refunds / Chargebacks:** publish **agreement quality** between platform and bank (daily).
• **Forecast vs Actuals:** publish a stability lane that reflects **on-plan vs jitter**.

What changes in meetings (and what doesn't).

• **Doesn't change:** the number you report (`m`) and the SQL/joins that produce it.
• **Does change:** you now have a bounded, sortable `a` and a band that expose **stability debt** early — so course-corrections happen **before** surprises.

0C — Two-Week Pilot (Finance) — AR, Refunds, Forecasts

Goal. Prove that adding a bounded stability lane beside a few core KPIs surfaces risk **earlier** without changing reported numbers.

Scope (3 lines to start).

• **AR / Collections health:** publish lane beside `m := collected/issued`.
• **Refunds / Chargebacks reconciliation:** publish **daily agreement quality** for platform vs bank totals.
• **Forecast vs Actuals stability:** publish a lane that reflects **on-plan vs jitter**.

Minimal lane mappers (declared up-front).

• **Coverage (AR tails):** `a := 2*q - 1` where `q in [0,1]` is coverage or completeness.
• **Agreement (recons):** `a := tanh( 1 - |m_platform - m_bank|/b )` with a clear bound `b`.
• **Residual stability (forecast):** `a := tanh( k*(1 - |actual - forecast|/s) )`.
• **Ratios (if needed):** for `m := f/g`, use `a_div := tanh( atanh(a_f) - atanh(a_g) )`.
All `a` values are **clamped**: `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a = 1e-6`.
Numbers remain identical by construction: **collapse parity** `phi((m,a)) = m`.

Bands (executive-friendly defaults).

A++: `a>=0.75`, **A+:** `0.50<=a<0.75`, **A0:** `0.25<=a<0.50`, **A-:** `0.10<=a<0.25`, **A--:** `a<0.10`.
Hysteresis: only promote if `delta_a >= 0.05`; only demote if `delta_a <= -0.05`.

Daily output (pilot CSV).

```
time, kpi, m, a, band, knobs_hash, build_id, stamp
```
`stamp` (optional but recommended):
`"SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain"`.

Acceptance gates (executive level).

• **Report parity:** all published `m` equal your current numbers (byte-for-byte where applicable).
• **Order-invariance:** results are stable if you re-run as batch vs daily stream (no material change in `a`).
• **Provenance:** stamps chain correctly day-to-day (`chain_k` continuity).
• **Actionability:** at least two lines show earlier signals (band shifts or `a` drift) that corroborate known friction (e.g., tails, timing slips, jitter).
• **Noise reduction:** fewer threshold flickers and false alarms in weekly reviews.

Timeline.

• **Day 0:** Pick KPIs, confirm band thresholds, set `k,s,b` knobs, assign owners.
• **Days 1–3:** Emit pilot CSVs with `a, band, knobs_hash, build_id, stamp`. Validate `phi((m,a)) = m`.
• **Days 4–7:** Review early patterns; adjust **only the knobs** (not SQL). Lock the manifest.
• **Days 8–10:** Extend to one additional slice (e.g., by segment or region) to check **stability debt hiding in blends**.
• **Days 11–14:** Executive readout; decide keep/extend.

Deliverables (end of Week 2).

• **Executive brief (1–2 pages):** headline wins, band movements, and the two or three "caught-early" cases.
• **Stamped logs:** daily pilot CSVs with stamps and a short acceptance note.
• **Locked manifest:** ASCII file with `version, build_id, date_utc, eps_a, eps_w, gamma, division_policy, bands, mappings{...}, conformance_checksum`.

Roles.

• **Finance owner:** chooses KPIs and thresholds; signs off on parity of `m`.
• **Data/BI:** adds columns and bands; wires dashboards (no SQL rewrites to `m`).
• **Internal audit:** checks stamps, manifest completeness, and replay.

Guardrails.

• **Read-only start:** no policy or control changes in the pilot.
• **Declared knobs only:** any change to `k,s,b` regenerates `knobs_hash`.
• **No silent math:** every lane mapping is in the manifest; every day is reproducible.

0D — Governance & Conformance (what we guarantee)

Purpose. Make SSM-Audit adoption predictable: **same numbers**, **bounded stability**, **replayable evidence**. This section defines the non-negotiables, the checks you run, and the artifacts you keep.

0D.1 Non-negotiable guarantees (normative)

• **Collapse parity.** Classical values never change: `phi((m,a)) = m`.

• **Bounded lane.** Stability is always finite: `a in (-1,+1)`; inputs are clamped: `a := clamp(a, -1+eps_a, +1-eps_a)`.
• **Order invariance.** For any fixed knobs, stability composition is order-independent to a declared epsilon: **batch == stream == shuffled**.
• **Determinism.** Same inputs + same knobs => same outputs (byte-for-byte where applicable).
• **Division policy.** Defaults to safe strictness near zeros: `division_policy = "strict"`. Any alternative (e.g., `"meadow"` or `"soft"`) must be **declared explicitly**.
• **Bands are declared.** Band edges and hysteresis are **stated in the manifest**; no hidden rules.
• **Read-only start.** Pilots must not mutate controls based on `a`; decisions can reference bands as **advisory** until formally approved.
• **Privacy posture. No PII** in lane fields or stamps; stability is about the **number**, not the person.

0D.2 Conformance checks (you run these, we pass or fail)

Lane sanity.
• **C1 Clamp safety.** All rows satisfy `-1 < a < +1` after clamping (`eps_a = 1e-6` default).
• **C2 Collapse parity.** For every hop in your pipeline, `phi((m,a)) == m`.
• **C3 Range parity.** Bands reflect lane: if `a >= 0.75` then `band == "A++"`, etc.

Order & composition.
• **C4 Order invariance.** Given the same set of rows, the computed `a_out` is equal (within epsilon) when processed as batch, as day-by-day stream, or in any permutation.
• **C5 Ratio lane identity.** For `m := f/g`, lane computed pre-join vs post-join is identical: `a_div := tanh( atanh(a_f) - atanh(a_g) )` matches within epsilon in both paths.
• **C6 Rollup parity.** Grouped rollups use the same fusion rule and produce the same `a_out` as the equivalent flat calculation:
`a_out := tanh( (Σ w_i*atanh(a_i)) / max(Σ w_i, eps_w) ).`

Rounding & reproducibility.
• **C7 Numeric determinism.** Calculations use float64 and fixed string formatting; reruns reproduce `a` and `band` within the declared tolerance.
• **C8 Knobs fingerprint.** Changing any declared knob changes `knobs_hash`; unchanged knobs reproduce prior results.

Provenance (if stamping is enabled).
• **C9 Chain continuity.** For each day or file `k`:
`chain_k = sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )` with `chain_0 = "0"*64`.
• **C10 Anchor parity.** For a day with stamps `{Stamp_1, ..., Stamp_n}`:
`rollup_D = sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )` matches the published day anchor.

Operational stability.
• **C11 Hysteresis sanity.** Band promotions require `delta_a >= 0.05`; demotions require `delta_a <= -0.05` (defaults unless overridden).
• **C12 Alert calmness.** After bands are applied, alert counts do not exceed pre-pilot baseline for equivalent conditions, unless explicitly justified.
Each check returns **PASS / WARN / FAIL** with a short reason and the affected rows (or aggregates).

0D.3 Minimal manifest (governance contract)

Plain ASCII file kept beside the CSVs. Example:

```
[SSM_AUDIT_MANIFEST]
version = 1.0
build_id = <short tag>
date_utc = <YYYY-MM-DD>
eps_a = 1e-6
eps_w = 1e-12
gamma = 1.0
division_policy = strict
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}
mappings = {AR:coverage(...), Refunds:agreement(...),
Forecast:residual(...)}
conformance_checksum = <hex>
```

• `knobs_hash` is derived from the `mappings, bands, hysteresis, eps_a, eps_w, gamma, division_policy`.
• Any change to these values **must bump `build_id`** and **regenerates `knobs_hash`**.

0D.4 Provenance (optional but recommended)

Per row or per file stamp.
• **Stamp core:**
`stamp_core := "SSMCLOCK1|" + iso_utc + "|" + rasi_idx + "|" + theta_deg + "|" + sha256(file)`
• **Chain step:**
`chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core) )` with `chain_0 := "0"*64`

• **Daily anchor:**
`rollup_D := sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )`
Result: a **tamper-evident, replayable** trail that proves ordering and content **without ceremony**.

0D.5 CI wiring (lightweight, always on)

• **Daily.** Run **C1–C7** and **C11** on the fresh CSVs; emit a 1-page status with counts and any WARN/FAIL.
• **Weekly.** Add **C4–C6** replay tests (batch vs stream vs shuffled), **C8** knobs diff, and (if enabled) **C9–C10** chain/anchor checks.
• **Quarterly.** Independent spot-audit: rerun a historical window from raw inputs to confirm determinism and manifest discipline.

0D.6 Go / No-Go for scaling beyond pilot

Go when the last 10 business days show:
• `phi((m,a)) = m` everywhere (**C2 PASS**),
• No **FAILs** on **C4–C7**,
• Fewer alert flickers after bands (**C12 PASS**),
• At least two concrete "caught-early" cases validated by owners,
• Provenance intact if enabled (**C9–C10 PASS**).
No-Go if any **FAIL** persists for >2 business days or if stability bands materially contradict ground truth without explanation.

0D.7 Evidence pack (kept for each pilot and release)

• **Manifest** (version, build_id, knobs, policies).
• **Pilot CSVs** (with `a, band, knobs_hash, build_id, stamp` if enabled).
• **Conformance report** (C1–C12 outcomes with reasons).
• **Executive brief** (1–2 pages: band movements, early warnings, decisions taken).
• **Acceptance note** (date, owners, exceptions, and their resolution).

0D.8 Responsible use and limits

• **Advisory first.** Use bands to guide attention; keep controls unchanged until a **formal safety case** is documented.
• **Declared scope.** Stability reflects the declared mapping only (e.g., coverage, agreement, residual); it is **not a guarantee** beyond the data.
• **Separation of duties.** The team declaring `mappings{...}` should not be the same team signing off on executive conclusions where feasible.

0E — Finance defaults & worked examples (AR, Refunds, Forecasts)

Goal. Ship a read-only lane beside a few finance KPIs so leaders can see how sturdy today's number is without changing the number itself. All formulas below are plain ASCII; numbers remain identical by construction via `phi((m,a)) = m`.

0E.1 Defaults (knobs and bands)

- **Lane bounds (always).** `a in (-1,+1)` with clamping `a := clamp(a, -1+eps_a, +1-eps_a)`; default `eps_a = 1e-6`.
- **Bands (executive-friendly).** `A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`.
- **Hysteresis (stop flicker).** Promote only if `delta_a >= 0.05`; demote only if `delta_a <= -0.05`.
- **Order-invariance.** Use the same fusion for any rollup so **batch == stream == shuffled** (declared later in Section 1).
- **Manifest essentials.** `schema_version`, `build_id`, `date_utc`, `eps_a`, `eps_w=1e-12`, `gamma=1.0`, `division_policy=strict`, `bands{...}`, `hysteresis{...}`, `mappings{...}`, `conformance_checksum`.

0E.2 AR / Collections health

Classical KPI (unchanged).
`m := collected_total / issued_total` (for the period)

Lane (default Level-1: coverage within policy window).
Let `q := collected_within_window / issued_total`, where "window" is your standard policy (e.g., `<= 30` days).
Map coverage to a bounded lane:

```
a_cov := 2*q - 1
```

Interpretation. Identical `m` across months can carry very different `a_cov` if the within-window coverage is weakening (growing long tail).

Optional Level-2 (blend coverage + tails).
Let `p_tail := past_due_>window / issued_total`. Add a tail penalty with weight `beta in [0,1]`:

```
a_tail := 1 - 2*p_tail
a := tanh( (atanh(a_cov) + beta*atanh(a_tail)) / max(1, beta) )
```

(default `beta = 0.5`)

Worked examples.

- **Healthy but honest.** `issued_total = 100`, `collected_total = 96` → `m = 0.96`. Within-window: `collected_within_window = 92` → `q = 0.92` → `a_cov = 2*0.92 - 1 = 0.84` → **band A++**.
- **Same m, risk rising.** `m = 0.96` again, but `q = 0.83` (more slipping just beyond the window) → `a_cov = 0.66` → **band A+**.
  Leadership sees **stability debt** early without touching the reported ratio.

CSV row sketch (read-only pilot).

```
2025-10-01, AR_collected_issued, 0.9600, 0.8400, A++, abcd1234, pilotW1,
SSMCLOCK1|...
```

0E.3 Refunds / Chargebacks reconciliation

Classical KPI choices (unchanged).
Either a rate (e.g., `m := refunds_total / shipments_total`) or a value (e.g., `m := refunds_total_currency`). Pick the one you already report.

Lane (agreement quality across systems).
Compare platform and bank views as daily rates or normalized values. With tolerance `b > 0` (your declared bound):
Let `d := |m_platform - m_bank|`.
`a_agree := tanh( 1 - d / b )`

Interpretation. As the two sources get closer than your bound `b`, the lane tends to `+1`; sustained small drifts reduce `a` even when month-end ties out.

Tuning tip.
Choose `b` near the "acceptable reconciliation noise" (e.g., `b = 0.005` for `0.5%` rate difference).

Worked example (rate).
`m_platform = 0.021`, `m_bank = 0.020`, `b = 0.005` → `d = 0.001`,
`1 - d/b = 0.8`, `a_agree = tanh(0.8) ≈ 0.664` → **band A+**.
A week later: `d` creeps to `0.003` → `1 - 0.003/0.005 = 0.4`, `tanh(0.4) ≈ 0.380` → **band A0**.
The headline refund rate may still look "fine"; the lane shows reconciliation strain building.

CSV row sketch.

```
2025-10-01, Refunds_agreement, 0.0200, 0.6640, A+, abcd1234, pilotW1,
SSMCLOCK1|...
```

0E.4 Forecast vs Actuals stability

Classical KPI (unchanged).
Use the actual you already track (e.g., daily revenue): `m := actual`.

Lane (residual stability).
Let residual `r := actual - forecast`. Use a scale `s > 0` (e.g., rolling MAD or a declared sigma) and sensitivity `k > 0`:
`a_resid := tanh( k*( 1 - |r|/s ) )`

Interpretation. Near-plan days keep `a` high; persistent small misses or sudden jumps lower `a` even when the month still "hits plan."

Optional volatility add-on (visual entropy).
Track calm vs churn with a decaying history on `m` (or on residuals):
`Entropy_t = log( Var(x_{0:t}) + 1 ) * exp( -lambda*t )`
Convert to a lane with a baseline `E0 > 0`:
`a_vol := tanh( c*( 1 - Entropy_t / E0 ) )`

If combining residual and volatility, blend in rapidity with weight `alpha in [0,1]`:
`a := tanh( ( alpha*atanh(a_resid) + (1-alpha)*atanh(a_vol) ) / max(alpha + (1-alpha), 1e-12) )`
Defaults: `alpha = 0.7`, choose `lambda := ln(2)/T_half` (e.g., `T_half = 14` days), pick `E0` as the calm baseline observed in a stable week.

Worked example (residual only).
`forecast = 1_000_000`, `actual = 980_000`, `r = -20_000`.
Choose `s = 50_000`, `k = 1.2`.
`1 - |r|/s = 1 - 0.4 = 0.6`, `k*... = 0.72`, `a_resid = tanh(0.72) ≈ 0.617` → **band A+**.
If the same miss repeats daily, the lane will trend down — prompting a "fix noise" action **before** misses become material.

CSV row sketch.
`2025-10-01, Revenue_actual, 980000, 0.6170, A+, abcd1234, pilotW1, SSMCLOCK1|...`

0E.5 Tuning guide (pick once, document in manifest)

- **AR coverage window.** Declare the policy (e.g., `<= 30` days). If collections shift seasonally, keep the window stable; let the lane move, not the rule.
- **Refunds bound `b`.** Set close to your acceptable daily reconciliation noise; sanity-check by backfilling a calm month.
- **Forecast scale `s`.** Prefer a robust measure (rolling MAD) over sigma in heavy-tail series. Start with `k in [1.0, 1.5]`.
- **Volatility half-life.** `lambda := ln(2)/T_half`; default `T_half = 14` days for executive rhythm; shorter half-life for faster lines.

0E.6 Dashboarding pattern (practical)

- Keep your existing `m` line or table.
- Add a small badge or underlay for `a` with the band (`A++/A+/A0/A-/A--`).
- Slice by segment/region/age bucket weekly; **stability debt hides in blends**.
- Executive view shows **band counts**; owner view shows **a-trends** and the **stamped CSV** link.

0E.7 Acceptance (module-specific)

- **AR:** parity on `m`; visible `a` drift when within-window coverage weakens; at least one tail action taken earlier than before.
- **Refunds:** parity on `m`; sustained `a_agree` decline triggers a reconciliation deep-dive **before** month-end.
- **Forecasts:** parity on `m`; repeated small residuals or rising volatility trigger plan/ops fixes within the pilot window.

0F — Standards & disclosure positioning (supplemental, non-GAAP)

Intent. Present the stability lane as a supplemental, non-GAAP risk-disclosure beside classical KPIs; classical measurements remain unchanged.

Policy.

- **Collapse parity.** `phi((m,a)) = m` (lane never alters `m`).
- **Display policy.** Publish bands from `a` via declared thresholds; communicate in plain language.
- **Suggested disclosure text.**
  "Stability lane (`a`) is a bounded, non-GAAP indicator published beside classical KPIs. Classical measurements remain unchanged; the lane quantifies current steadiness and timing-drift risk."
- **Governance.** Keep a minimal manifest, run conformance checks, and optionally add tamper-evident stamps.
- **Privacy.** No PII in lane fields or stamps; stability is computed from KPI aggregates.

---

1 — Formal canon (semantics, proofs, and checks)

1.1 Numeral and invariants (non-negotiable)

• Two-lane numeral: `x := (m, a)` where `m` is the classical value and `a in (-1,+1)` is the stability lane.
• Collapse parity: `phi((m,a)) = m` (the reported number never changes).
• Clamp rule: `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a := 1e-6` (default).

• Order invariance (target): for fixed knobs, composition of lanes yields the same result within epsilon whether processed in batch, as a stream, or in any permutation.
• Determinism: same inputs and knobs -> same outputs (byte-for-byte where applicable).
• Rapidity (internal): `u := atanh(a)`; combine lanes in rapidity space, then decode with `a := tanh(u)` (keeps `a` bounded).
• Zero-class display (policy): if `m_out == 0`, you may print `(0, +1)` (strict center) or `(0, a)` (informative); declare once per study.

1.2 Lane mappers (finance-ready library)

Each mapper is **declarative** (listed in the manifest) and **bounded** (outputs in `(-1,+1)` **before** clamping). Pick the smallest mapper that answers your business question; add blends only if needed. Unless stated otherwise, apply a final safety clamp: `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a = 1e-6`.

• Coverage (completeness, tails)
`a_cov := 2*q - 1` with `q in [0,1]`.
Examples: within-window collection coverage; GL line match-rate; data-room completeness.

• Agreement (reconciliation quality)
Let `d := |m1 - m2|` and `b > 0` be the declared bound of acceptable day-to-day divergence.
`a_agree := tanh( 1 - d/b ).`

• Residual stability (forecast vs actual)
Let residual `r := actual - forecast`, scale `s > 0`, sensitivity `k > 0`.
`a_resid := tanh( k*( 1 - |r|/s ) ).`

• Ratios (lane lift for `m := f/g`)
If factors expose lanes `a_f` and `a_g`, then
`a_div := tanh( atanh(a_f) - atanh(a_g) ).`

• Products (lane lift for `m := f*g`)

```
a_mul := tanh( atanh(a1) + atanh(a2) ).
```

• Blends (combine two lanes into one)

With weight `alpha in [0,1]`,

```
a := tanh( alpha*atanh(a_A) + (1-alpha)*atanh(a_B) ).
```

• Entropy drift (ZEOZO-Core) — preferred mapper (supersedes rolling-variance forms)

Edge-normalize once, then maintain bounded energy and **log-compress**:

```
med := median(x)
rad := median(|x - med|); rad := max(rad, eps)
y_t := (x_t - med) / rad
E_t := (1 - lam)*E_{t-1} + lam*(y_t^2)
Z_t := log(1 + E_t)
```

- **Option A (drift-as-risk lane).** Declare reference `Z_ref > 0`, sensitivity `c > 0`, then `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) ).`
- **Option B (alignment-track lane).** Define a slow track and use the gap, then
  `A_t := (1 - mu)*A_{t-1} + mu*Z_t`
  `Delta_t := | Z_t - A_t |`
  `a_gap := tanh( k_gap*( A_t - Z_t ) )` (positive when calm dominates)

Defaults (declare once). `lam = 0.10`, `mu = 0.04`, `eps = 1e-6`, `Z_ref := median(Z_t over pilot)`.
Init (deterministic). `E_0 := 0`, `A_0 := Z_0` (or pilot median), document in manifest.

• Calm-gated alignment (SYASYS-Core) — optional gate for executive "safe-unlock"

Let `clip(u, lo, hi) := min(max(u, lo), hi)` and reuse `Z_t`, `A_t`, `Delta_t` from ZEOZO.

Calm dose:

```
Q_t := rho*Q_{t-1} + (1 - rho)*clip( A_t - Z_t , 0 , 1 )
```

Gate (bounded in `[0,1)`).

```
SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR*Q_t ) )
```

Lane from gate (into `(-1,+1)`).

```
a_syasys := 2*SyZ_t - 1
```

Defaults (declare once). `rho = 0.90`, `kappa = 0.50`, `R = 8.0`, `muR := ln(2)/R`.
Use case: unlock advisories only when drift is low and alignment has accumulated sufficient calm dose.

Monotonicity note. If the source score increases (holding scale/bounds fixed), the induced lane **should not decrease**; all defaults above are isotonic after clamps.

1.3 Fusion (streaming/batch equivalence)

Goal. Same lane result irrespective of processing order (batch vs stream vs shuffled).

• Streaming accumulator (canonical):
For each observation with lane `a_i` and weight `w_i >= 0`, update
```
U := U + w_i*atanh(a_i)
W := W + w_i
```
and decode
`a_out := tanh( U / max(W, eps_w) )` with `eps_w := 1e-12`.
Default weights: `w_i := |m_i|^gamma` with `gamma := 1.0`. The uniform option is `w_i := 1`.

• Why order-invariant? (proof sketch)
Because `(U, W)` are simple sums, permutation of inputs does not change their totals; decoding applies the same function once at the end. Thus, for fixed `w_i` and `eps_w`, any processing order yields the same `a_out` within rounding tolerance.

• Groupby / rollups.
Apply the same `(U, W)` rule within each group key; nested aggregations preserve results because addition is associative and commutative.

1.4 Arithmetic lifts (how lanes behave with common KPIs)

• Identity (collapse): `phi((m,a)) = m`.
• Scale-free transforms on `m`. Any monotone transform of `m` used purely for display leaves `a` unchanged.
• Sum/average. Use the fusion rule above with declared weights (`w := 1` yields an unweighted mean-lane).
• Product/ratio. Use `a_mul` / `a_div` as defined in §1.2; compute `m` classically.
• Joins. Compute `m` via your existing SQL; compute lanes via the declarative rule (e.g., pre-join lanes for `f` and `g`, then `a_div` post-join). Conformance requires pre- vs post-join parity within epsilon.

1.5 Bands and hysteresis (executive-friendly labels)

• Default thresholds:
`A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`.
• Hysteresis (state machine):
Promote only if `delta_a >= 0.05`; demote only if `delta_a <= -0.05`.
This suppresses flicker while allowing real movement.
• Declaration. Band edges and hysteresis are manifest knobs; any change regenerates `knobs_hash`.

1.6 Determinism, rounding, and export

• Numeric type: compute with `float64`.
• Lane clamp: always apply `a := clamp(a, -1+eps_a, +1-eps_a)` before any `atanh`.
• Export precision (recommended):
`m` uses your existing format; `a` exports as fixed-precision (e.g., 4 decimals) with round-half-even; `band` is derived from the unrounded `a`.
• Knobs fingerprint:
`knobs_hash := sha256( ascii( canonical_json( {eps_a, eps_w, gamma, division_policy, bands, hysteresis, mappings, lam, mu, rho, kappa, muR} ) ) )`.
The `canonical_json` is a stable key-sorted, whitespace-normalized encoding.

1.7 Stamped logs (tamper-evident replay, optional)

• Per-file stamp (ASCII, single line):
`SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain`
where `iso_utc := "YYYY-MM-DDTHH:MM:SSZ"`, `rasi_idx in {0..11}`, `theta_deg in [0,360)`.
`chain_0 := "0"*64`, `chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )`.

• Day anchor:
`rollup_D := sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )` for stamps emitted that day.
• Constraints:
No leap-second `:60`; unknown extension keys (e.g., `kv:...`) are ignored by verifiers; stamps are advisory evidence that do not alter `m`.

1.8 Minimal schemas (CSV and manifest)

• CSV (pilot-ready)
Header (canonical order):
`time, kpi, m, a, band, knobs_hash, build_id, stamp`

Field rules:

- `time := ISO-8601 UTC` (e.g., `2025-10-21T00:00:00Z`).
- `kpi := snake_case` (e.g., `ar_collected_issued`).
- `m := classical numeric` (unchanged KPI).
- `a in (-1,+1)` after clamp `a := clamp(a, -1+eps_a, +1-eps_a)`.
- `band in {A++, A+, A0, A-, A--}` from declared thresholds.
- `knobs_hash := SHA256( canonical_json( manifest_without_conformance_checksum ) )`.
- `build_id := short tag for this run` (e.g., `pilotW1`).
- `stamp` is optional. When enabled, use chained stamps:
  `stamp_core_k := time + "|" + kpi + "|" + m + "|" + a + "|" + band + "|" + knobs_hash`
  `chain_k := SHA256( chain_{k-1} + "|" + stamp_core_k )`, with `chain_0 := build_id`.
  Store as `SSMCLOCK1|chain_k`.

Optional audit columns (when using ZEOZO/SYASYS):
`Z, A, Delta, Q, SyZ` (for dashboards only; not required by the lane logic).

Sorting for reproducibility: **primary** `time` ascending, **secondary** `kpi` ascending.

• Manifest (beside CSV, plain ASCII)

```
[SSM_AUDIT_MANIFEST]
schema_version = 1
build_id = <short tag>
date_utc = <YYYY-MM-DD>

# Core safety
eps_a = 1e-6
eps_w = 1e-12
gamma = 1.0
division_policy = strict

# Bands and hysteresis
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}

# Mapper declarations (per KPI)
# Examples (choose what you use and pin params):
#   coverage(q)
#   agreement(b)
#   residual(s, k)
#   zeozo(lam, Z_ref, c)         # Option A
#   zeozo_gap(lam, mu, k_gap)    # Option B
```

```
#   blend(alpha, kpi_A, kpi_B)

mappings = {
  AR: coverage(q=within_window/issued_total),
  Refunds: agreement(b=0.005),
  Forecast: residual(s=MAD_14d, k=1.2),
  KPI_X: zeozo(lam=0.10, Z_ref=pilot_median, c=1.0)
}

# ZEOZO/SYASYS knobs (declare only if used)
lam = 0.10
mu = 0.04
rho = 0.90
kappa = 0.50
R = 8.0
muR = ln(2)/R

# Conformance (deterministic build fingerprint)
conformance_checksum = <hex>
```

Notes for implementers (manifest):

- **Canonical JSON** for `knobs_hash`: emit keys in sorted order, fixed numeric formatting, and no trailing whitespace.
- If `stamp` is disabled, omit or leave the column blank in CSV; keep the formula above documented for future enablement.
- For ZEOZO Option A, set `Z_ref := median(Z_t over pilot)` and document the pilot window; for Option B, initialize `E_0 := 0`, `A_0 := Z_0`.
- Keep mapper parameters pinned (e.g., `b`, `s`, `k`, `lam`) to ensure reproductions match exactly.

1.9 Conformance tests (PASS/WARN/FAIL)

Define a numeric tolerance `eps_num := 1e-12` (or stricter) for float comparisons.

• T1 Clamp safety: all rows satisfy `-1 < a < +1` after clamping.
• T2 Collapse parity: `phi((m,a)) == m` at every hop (string-exact where applicable).
• T3 Band parity: band derived from `a` matches declared thresholds.
• T4 Order invariance: compute lanes as batch, as daily stream, and in a random shuffle; differences `<= eps_num`.
• T5 Ratio parity: pre-join vs post-join `a_div` agree within `<= eps_num`.
• T6 Rollup parity: nested vs flat fusion agree within `<= eps_num`.
• T7 Deterministic export: re-runs with same inputs/knobs reproduce `a` and `band` (after formatting).
• T8 Knobs fingerprint: any change to `{eps_a, eps_w, gamma, division_policy, bands, hysteresis, mappings, lam, mu, rho, kappa, muR}` changes `knobs_hash`.
• T9 Stamp continuity (if enabled): `chain_k` and `rollup_D` recompute exactly.
• T10 Hysteresis sanity: promotions require `delta_a >= promote`, demotions `<= demote`.

Additional tests for ZEOZO/SYASYS (if used):
• T11 ZEOZO scale-free check: rescale inputs `x' := alpha*x + beta` with `alpha>0`; recomputed `Z_t'` and `Z_t` match within `<= eps_num` after the same edge-normalization and knobs.
• T12 Calm-gate monotonicity: with `Q_t` fixed, increasing `Z_t` or `Delta_t` must not increase `SyZ_t` (non-increasing in drift/misalignment); lane `a_syasys := 2*SyZ_t - 1` must not promote bands under such perturbations.
• T13 Mapper consistency: when both `a_zeozo` and `a_syasys` are present, `sign(a_syasys - 0)` should agree with `sign(A_t - Z_t)` on at least `>= 95%` of days in calm periods (tolerance declared).
• T14 Replay stability: rerunning ZEOZO/SYASYS with the same `lam, mu, rho, kappa, muR` and same `rad` floor reproduces `Z, A, Delta, Q, SyZ` within `<= eps_num`.

1.10 Proof sketches (why the rules behave)

• Boundedness: `tanh(z) in (-1,+1)` for all real `z`; clamping prevents `atanh` singularities, so every mapper and fusion step yields a finite `a`. `Z_t = log(1+E_t)` is `>= 0` and sub-linear in `E_t`.
• Order invariance: fusion accumulates linear sums `(U,W)`; addition is associative/commutative, so final `U/W` is permutation-invariant; decoding by `tanh` preserves equality within rounding.
• Product/ratio lifts: mapping the lane to rapidity `u := atanh(a)` turns composition into simple `u` addition or subtraction. Returning via `a := tanh(u)` restores boundedness, giving `a_mul` and `a_div`.
• Collapse parity: all lane operations act only on `a`; the projection `phi((m,a)) = m` is identity for the classical value.
• ZEOZO scale-free: edge normalization by `median`/`MAD` cancels positive affine rescaling, so `y_t` and thus `E_t`/`Z_t` are invariant to units.

1.11 Tiny verifier (one-screen, calculator-fast)

Given a CSV with columns `m,a` (and optionally `Z,A,Delta,Q,SyZ`) and a manifest with knobs:

1. Clamp: `a := clamp(a, -1+eps_a, +1-eps_a).`
2. Collapse parity check: recompute `phi((m,a))` and assert equality to `m`.
3. Band check: recompute `band` from `a` and thresholds.
4. Order test: compute `a_out` for the set three ways (batch/stream/shuffle) and compare within `eps_num`.
5. Ratio parity (optional): if `f,g` lanes present, compute `a_div` pre-join and post-join and compare.
6. Stamp recompute (if present): rebuild `chain_k` and `rollup_D` and compare.

7. ZEOZO/SYASYS (optional): if fields exist, re-derive `Z,A,Delta,Q,SyZ` from the raw series and knobs; run T11–T14.
   Outputs: `PASS/WARN/FAIL`, with row indices (or keys) and a 1-line reason per failure.

---

2 — Implementation Guide (pilot → production)

2.1 Wire-up (what to add, nothing to break)

Add three columns beside each KPI you already publish.
• `a` (stability lane in `(-1,+1)`)
• `band` (`A++, A+, A0, A-, A--`)
• `knobs_hash` (fingerprint of declared settings)
Optional (recommended for audit dashboards): `build_id` (run tag), `stamp` (one-line ASCII provenance), and—if using ZEOZO/SYASYS—`Z, A, Delta, Q, SyZ`. Classical values remain intact by construction: `phi((m,a)) = m`.

Pick 3–5 KPIs to start.
AR/Collections, Refunds/Chargebacks, and Forecast vs Actuals are ideal because leadership already watches them daily.

2.2 Declare the lane mappers (business-first)

Choose the **smallest** formula that answers the business question; **keep it stable** during the pilot. Unless noted, apply a final safety clamp: `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a = 1e-6`.
All mappers output in `(-1,+1)` **before** clamping.

• Coverage (AR tails)
`a := 2*q - 1` where `q in [0,1]` is within-window coverage.

• Agreement (recons)
Let `d := |m_platform - m_bank|`, with tolerance `b > 0`.
`a := tanh( 1 - d/b ).`

• Residual stability (forecast)
Let `r := actual - forecast`, scale `s > 0`, sensitivity `k > 0`.
`a := tanh( k*( 1 - |r|/s ) ).`

• Entropy (preferred, ZEOZO-Core)

Edge-normalize once:

```
med := median(x)
rad := median(|x - med|); rad := max(rad, eps)
y_t := (x_t - med) / rad
```

Bounded energy and log-compress:

```
E_t := (1 - lam)*E_{t-1} + lam*(y_t^2)
Z_t := log(1 + E_t)
```

Lane from ZEOZO (choose one):

- **Risk lane:** `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot).`
- **Alignment-track lane:** define `A_t := (1 - mu)*A_{t-1} + mu*Z_t`, `Delta_t := | Z_t - A_t |`, then `a_gap := tanh( k_gap*( A_t - Z_t ) )` (positive when calm dominates).

Deterministic init: `E_0 := 0`, `A_0 := Z_0` (or pilot median). Pin `lam, mu, c, k_gap, Z_ref` in the manifest.

• Legacy (optional, earlier variance form)

```
Zentrube_t := log( Var(x_{0:t}) + 1 ) * exp( -lambda*t )
a := tanh( c*( 1 - Zentrube_t / E0 ) )
```

Declare `lambda := ln(2)/T_half`, choose `E0` from a calm baseline.

Clamping & safety (all mappers)

Before any `atanh` or rapidity-space blend, **clamp first:**

`a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a = 1e-6`.

Maintain collapse parity at all times: `phi((m,a)) = m`.

2.3 Ratios, products, and blends (stay bounded, stay honest)

When a KPI is a ratio `m := f/g`, lift lane behavior without touching `m`:

• Ratio lane: `a_div := tanh( atanh(a_f) - atanh(a_g) ).`

• Product lane: `a_mul := tanh( atanh(a1) + atanh(a2) ).`

• Blend two lanes A and B (with `alpha in [0,1]`):

```
a := tanh( alpha*atanh(a_A) + (1 - alpha)*atanh(a_B) ).
```

This keeps `a in (-1,+1)` and preserves interpretability.

2.4 Streaming/batch equivalence (order-invariant fusion)

Publish a lane per row; fuse lanes the same way for rollups so `batch == stream == shuffled`.
Streaming accumulator (canonical):
`U := U + w*atanh(a)`
`W := W + w`
`a_out := tanh( U / max(W, eps_w) )` with `eps_w = 1e-12`.
Default weights: `w := |m|^gamma` with `gamma = 1.0`. The uniform option is `w := 1`.
Because `(U,W)` are simple sums, any order yields the same `a_out` within rounding tolerance.

2.5 Bands and hysteresis (what leaders actually see)

Defaults:
`A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`.
Hysteresis: promote only if `delta_a >= 0.05`; demote only if `delta_a <= -0.05`.
Declare these in the manifest so everyone knows the rules.

2.6 Minimal manifest (governance contract)

Plain ASCII kept beside the data, for example:

```
[SSM_AUDIT_MANIFEST]
version = 1.0
build_id = <short tag>
date_utc = <YYYY-MM-DD>

# Core clamps and fusion
eps_a = 1e-6
eps_w = 1e-12
gamma = 1.0
division_policy = strict

# Bands and hysteresis
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}

# Declared mappers
mappings = {AR:coverage(...), Refunds:agreement(...),
Forecast:residual(...), KPI_X:zeozo(...)}

# ZEOZO/SYASYS knobs (if used)
lam = 0.10
mu = 0.04
# Optional calm-gate
rho = 0.90
```

```
kappa = 0.50
R = 8.0
muR = ln(2)/R

# Extended keys for 2.11 / 2.12 if applicable (can be omitted if not used)
ccy_base = <ISO code>
fx_source = <id or provider>
fx_time = <ISO8601>
index_source = <e.g., CPI series id>
index_time = <ISO8601>
privacy_mode = aggregate_only
g_macro_policy = none

conformance_checksum = <hex>
```

`knobs_hash` is computed from `{eps_a, eps_w, gamma, division_policy, bands, hysteresis, mappings, lam, mu, rho, kappa, muR}` encoded canonically.

2.7 Stamped logs (optional, one line of ASCII)

Per file (or per day), emit a single provenance line:
`SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain`
with `chain_0 := "0"*64` and `chain_k := sha256(ascii(chain_{k-1} + "|" + stamp_core_k))`.
This creates a tamper-evident, replayable trail with almost zero ceremony. Do not include PII in stamps.

2.8 Dashboards (executive and owner views)

• Executive widget: show the usual `m` with a small badge for `band`. Also show band counts (how many `A++, A+, …`) for the week.
• Owner widget: trend of `a` with band shading; click-through to the stamped daily CSV.
• ZEOZO tracks (if used): show `Z` and `A` (alignment), and optionally `Delta` (gap) for readiness context; calm periods show `A >= Z`.
• Slice weekly: by segment/region/age bucket to expose stability debt hiding in blends.

2.9 Conformance CI (what runs every day)

Define numeric tolerance: `eps_num = 1e-12`. The CI evaluates the **published CSV** against the **manifest** and emits **PASS / WARN / FAIL** per check. All formulas below are plain ASCII.

Inputs. Previous CSV `time,kpi,m,a,band,knobs_hash,build_id,stamp` and the manifest in canonical form.

Determinism. Fixed sort (`time`↑, then `kpi`↑). Any RNG (if present) must use a fixed seed (e.g., `SSM_AUDIT_CI_SEED = 1`).

Core checks

- **Clamp safety.** For every row, `-1 < a < +1` after `a := clamp(a, -1+eps_a, +1-eps_a)`.
  Fail if any `a <= -1+eps_a` or `a >= +1-eps_a`.
- **Collapse parity.** `phi((m,a)) = m` at every hop (lane never alters `m`).
  Fail if any `m_out != m_in` beyond `eps_num`.
- **Band parity.** `band == band_from_a(a, bands)` using manifest thresholds.
  Fail if mismatch occurs for any row.
- **Order test (batch == stream == shuffled).**
  Let rapidity `u := atanh(a)` and fusion `a_fuse := tanh( SUM_i w_i*u_i / max(SUM_i w_i, eps_w) )`.
  Recompute lanes three ways over identical data:
  1. **Batch** (single-pass vector), 2) **Stream** (incremental by time), 3) **Shuffled** (random permutation, same weights).
     Require `|a_batch - a_stream| <= eps_num` and `|a_batch - a_shuffled| <= eps_num` for all outputs.
     Fail if violated.
- **Ratio parity (for `m := f/g`).**
  Pre-join: `a_pre := tanh( atanh(a_f) - atanh(a_g) )`.
  Post-join: recompute lane on the ratio using the declared ratio mapper.
  Require `|a_pre - a_post| <= eps_num`.
  Fail if violated.
- **Rollup parity (nested == flat).**
  For any hierarchy, let **flat** be `a_flat := tanh( SUM_i w_i*atanh(a_i) / max(SUM_i w_i, eps_w) )`.
  Let **nested** be the same fusion applied level-by-level.
  Require `|a_flat - a_nested| <= eps_num`.
  Fail if violated.
- **Deterministic export.** Re-run with the same CSV and manifest must reproduce identical `a` and `band` bit-for-bit (after numeric formatting rules).
  Fail if any cell changes.
- **Knobs fingerprint.** `knobs_hash == SHA256( canonical_json( manifest_without_conformance_checksum ) )`.
  Any knob change must flip `knobs_hash`.
  Fail if mismatch or unchanged hash after knob edits.
- **Stamp continuity (if used).**
  `stamp_core_k := time + "|" + kpi + "|" + m + "|" + a + "|" + band + "|" + knobs_hash`
  `chain_k := SHA256( chain_{k-1} + "|" + stamp_core_k )`, with `chain_0 := build_id`.
  Recompute and require exact match for every `k`.
  Daily rollup (optional): `rollup_D := SHA256( join_all_chain_k_for_day_D )` must recompute exactly.
  Fail if any mismatch.

- **Hysteresis sanity.**
  Promotions only when `delta_a >= promote`, demotions only when `delta_a <= demote`, where thresholds come from the manifest.
  Fail if any band transition violates the rule.

ZEOZO / SYASYS checks (if enabled)

- **ZEOZO scale-free.** Under a positive affine rescale `x' := alpha*x + beta` with `alpha > 0`, after edge-normalization
  `med' := alpha*med + beta`, `rad' := alpha*rad`, `y'_t := (x'_t - med')/rad' = y_t`.
  Therefore `E'_t = E_t` and `Z'_t = Z_t`.
  Require `|Z'_t - Z_t| <= eps_num` for all `t`.
  Fail if violated.
- **Calm-gate monotonicity (SYASYS).** With `Q_t` fixed,
  `SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR*Q_t ) )`.
  Increasing `Z_t` or `Delta_t` must **not increase** `SyZ_t`.
  Check `dSyZ/dZ <= 0` and `dSyZ/dDelta <= 0` numerically within `eps_num`.
  Fail if violated.
- **Replay stability.** Recompute `Z_t, A_t, Delta_t, Q_t, SyZ_t` with identical knobs and initialization (`E_0 := 0`, `A_0 := Z_0`).
  Require per-step equality within `eps_num`.
  Fail if violated.

Outcome policy

- **PASS:** all checks within `eps_num`, no rule violations.
- **WARN:** only non-material issues (e.g., missing optional `stamp` column while stamps are disabled, or absent audit-only columns `Z,A,Delta,Q,SyZ`).
- **FAIL:** any parity/determinism breach, hysteresis violation, or cryptographic mismatch.

Artifacts. The CI emits a machine-readable report (per-kpi JSON) and a human summary table (counts of PASS/WARN/FAIL per check).

2.10 Rollout playbook (pilot → portfolio)

1. Pick KPIs and fix knobs. Document them in the manifest.
2. Add `lane + band + knobs_hash`. Do not touch how `m` is computed.
3. Enable CI checks. Keep results visible to owners.
4. Run two weeks. Collect "caught-early" examples and band movements.
5. Executive readout. If benefits are clear, add 2–3 more KPIs or new slices.

6. Institutionalize. Keep bands as a standing column; keep stamps for critical flows; review knobs quarterly.

2.11 Multi-currency and hyperinflation guardrails (optional, lane-safe)

Goal: normalize `m` while keeping `a` bounded and interpretable; preserve collapse parity.

• Base currency selection: `ccy_base` (manifest).
• Magnitude normalization: `m_fx_norm := m_native * fx_rate(native->base).`
• Lane handling (default): carry native lane unchanged `a := a_native`.
• Optional FX-quality blend (declare weight `k_fx in [0,1]`):
`a := tanh( atanh(a_native) + k_fx*atanh(a_fx_quality) ).`

Hyperinflation / indexation:
• Publish an index series `I_t` (e.g., CPI) with base `I_0`.
• Normalize magnitude: `m_real := m_nominal * ( I_0 / I_t ).`
• Lane guard (optional macro damping): `a := clamp( g_macro_t * a , -1+eps_a, +1-eps_a )` with `g_macro_t in [0,1]` and declared source.

Guards:
• Denominators: use `max(W, eps_w)` in pools; forbid division by zero unless `division_policy = meadow|soft`.
• Manifest declarations: `fx_source, fx_time, index_source, index_time, k_fx, g_macro_policy`.

2.12 Privacy & PII guard (alignment-only, no PII)

• `a` must be derived from aggregates or KPI math — never from raw personal identifiers.
• Stamps and manifests carry zero PII; only hashes and knobs.
• Data contract addendum: prohibited columns list; redaction rule before lane computation.
• Access model: role-based views; execs see `band` and `a` on aggregates; analysts see `(m,a,band)` on aggregates only.

2.13 BI/ERP integration stubs (CSV → Excel / Power BI / Looker / API)

CSV columns (minimal): `date, unit, kpi, m, a, band, U, W, knobs_hash, stamp.`

Excel/Sheets owner view (pseudo-formulas):
• Band from `a`: `IFS(a>=0.75,"A++", a>=0.50,"A+", a>=0.25,"A0", a>=0.10,"A-", TRUE,"A--")`.
• Hysteresis state: keep `a_prev` and `band_prev`; promote if `a - a_prev >= 0.05`; demote if `a - a_prev <= -0.05`; else hold.

Power BI / Looker:
• Import CSV; define calculated column Band from `a` as above; add measures for `A++..A--` counts and week-over-week band transitions.

ERP touchpoint (minimal idempotent API):
• `POST /lane` with body `(kpi_id, date, m, a, band, knobs_hash)`; idempotency key `(kpi_id, date, knobs_hash)`.

2.14 CFO ROI worksheet (template starter)

Inputs: `revenue, daily_rev, days_reduced_in_DSO, refund_rate_baseline, refund_rate_after, gross_sales, MAPE_baseline, MAPE_after, k_cash.`
Outputs: `Savings_AR, Savings_ref, Savings_fc, Savings_total.`
Formulas:

```
Savings_AR := days_reduced_in_DSO * daily_rev
Savings_ref := (refund_rate_baseline - refund_rate_after) * gross_sales
Savings_fc := (MAPE_baseline - MAPE_after) * revenue * k_cash
Savings_total := Savings_AR + Savings_ref + Savings_fc
```

Note: numbers are illustrative; replace with organization-specific parameters.

2.15 Evidence pack & governance (what to archive per pilot)

• Data: stamped CSVs, manifest, `knobs_hash`, and CI PASS/WARN/FAIL sheet.
• Plots: weekly `a` trends with band shading, and if ZEOZO is used, `Z` and `A` overlays.
• Notes: 3–5 "caught-early" examples (band slide → corrective action → outcome).
• Review: quarterly knobs review; update manifest; recompute `conformance_checksum`.

---

3 — Deeper Finance Modules (credit, revenue quality, cash predictability, cohort durability)

(All formulas plain ASCII; lanes bounded in (-1,+1); clamp before any atanh.)

3.1 Credit Quality & DSO health

Classical KPIs (unchanged).
`m1 := DSO_days`, `m2 := bad_debt_rate`, `m3 := collected/issued` (as already reported).

Lanes (defaults; pick 1–3).
• On-policy coverage (issuance discipline). Let `q := in_policy_issuance / total_issuance`. Lane: `a_policy := 2*q - 1`.
• DSO drift vs baseline. Let `r := DSO_days - DSO_ref_days`; choose `s > 0` and sensitivity `k > 0`: `a_dso := tanh( k*( 1 - |r|/s ) ).`
• Concentration risk (top-N exposure). Let `p := share_topN` in `[0,1]`. Lane: `a_conc := 1 - 2*p` (higher concentration lowers `a`).
• Collections churn (ZEOZO-Core, optional). Build on receipts series `x_t` (e.g., daily collected).

```
med := median(x); rad := median(|x - med|); rad := max(rad, eps); y_t := (x_t - med)/rad
E_t := (1 - lam)*E_{t-1} + lam*(y_t^2); Z_t := log(1 + E_t)
A_t := (1 - mu)*A_{t-1} + mu*Z_t
```

Lane (choose one): `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot)`
or `a_gap := tanh( k_gap*( A_t - Z_t ) ).`

Blends (optional; rapidity).
Blend any two with `alpha in [0,1]`:
`a := tanh( alpha*atanh(a_policy) + (1 - alpha)*atanh(a_tail) ).`

Decision recipe (examples).
• If `m1` is steady but `a_policy` or `a_zeozo` drops a band, tighten credit on the slipping segment **before** DSO widens.
• If `a_conc` falls while revenue grows, cap exposure to top accounts; pursue diversification.

Module acceptance (pilot).
Report parity on `m1..m3`; show at least one early action triggered by a band movement (`a_policy`, `a_zeozo/a_gap`, or `a_conc`) that precedes later DSO deterioration.

3.2 Revenue Quality (net revenue you can trust)

Classical KPIs (unchanged).
`m := net_revenue`; optionally `m_gross := gross_revenue`, `m_adj := adjustments`.

Lanes (defaults).
• Adjustment pressure. Let `adj_ratio := m_adj / max(m_gross, eps)`. With bound `b > 0`: `a_adj := tanh( 1 - adj_ratio / b ).`
• Ledger↔settlement agreement (daily). `a_agree := tanh( 1 - |m_platform - m_bank| / b2 )` with declared `b2 > 0`.

• Price realization drift. Let `real := realized_price / list_price`; lane `a_real := tanh( 1 - |real - real_ref| / b3 ).`
• Revenue churn (ZEOZO-Core, optional). Apply ZEOZO on daily `net_revenue` (or margin). Produce `Z_t, A_t` as above; then either `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` or `a_gap := tanh( k_gap*( A_t - Z_t ) ).`

Composite revenue-quality lane (optional).
`a_revq := tanh( ( w1*atanh(a_adj) + w2*atanh(a_agree) + w3*atanh(a_real) ) / max(w1+w2+w3, 1e-12) )` with `w1,w2,w3 >= 0.`

Decision recipe.
• `a_adj` falling with stable `m` → investigate concession creep or policy overrides.
• `a_agree` weakening → escalate reconciliation friction before month-end; pair with `a_zeozo` on margin for confidence.

Module acceptance.
At least one caught-early instance where `a_adj` or `a_agree` band drop preceded a visible revenue shortfall or a month-end reconciliation issue.

3.3 Cash Predictability & Liquidity early-warning

Classical KPIs (unchanged).
`m := net_cash_inflow` (daily), plus your existing cash runway summaries.

Lanes (defaults).
• Schedule adherence (from AR aging). `q_on_time := receipts_on_or_before_due / receipts_total`; lane `a_sched := 2*q_on_time - 1.`
• Residual vs cash forecast. Let `r := actual_cash - expected_cash`; choose `s > 0`, `k > 0`: `a_cash := tanh( k*( 1 - |r|/s ) ).`
• Cash churn (ZEOZO-Core, preferred over legacy variance). Run ZEOZO on daily cash-inflow series to get `Z_t, A_t, Delta_t`.
Lane (choose one): `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` or `a_gap := tanh( k_gap*( A_t - Z_t ) ).`
• Calm-gated alignment (SYASYS-Core, optional executive "green-light").
`Q_t := rho*Q_{t-1} + (1 - rho)*clip(A_t - Z_t, 0, 1)`
`SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR*Q_t ) )`
Lane: `a_syasys := 2*SyZ_t - 1.`

Composite cash lane (optional).
`a_cashstar := tanh( 0.5*atanh(a_sched) + 0.5*atanh(a_cash) ).`
Alternative: replace `a_cash` with `a_gap` or `a_syasys` where governance prefers calm-confirmed signals.

Decision recipe.
`a_sched` or `a_cash/a_gap` band downgrades for 2–3 days in a row → senior review of near-term runway assumptions **even if** `m` remains positive.

Module acceptance.
Document at least one case where a lane downgrade (`a_sched`, `a_gap`, or `a_syasys`) preceded a short-term cash surprise; keep stamped CSVs for replay.

3.4 Cohort Durability (subscription, repeat purchase)

Classical KPIs (unchanged).
`m := retention_rate_90d` (or `m := LTV/CAC`, as currently reported).

Lanes (defaults).
• Retention within policy window. `q_ret := retained_within_90d / cohort_size`; lane `a_ret := 2*q_ret - 1`.
• Hazard drift vs reference. Let `h_t` be current-day hazard and `h_ref` your calm baseline; with bound `b > 0`: `a_haz := tanh( 1 - |h_t - h_ref| / b )`.
• Refund/credit pressure on the cohort. Reuse `a_adj` from §3.2, scoped to this cohort.
• Cohort churn (ZEOZO-Core, optional). Apply ZEOZO to cohort-level repeat-purchase cadence; use `a_gap` for stability sense.

Composite cohort lane (optional).
`a_cohort := tanh( alpha*atanh(a_ret) + (1 - alpha)*atanh(a_haz) )` with `alpha in [0,1]`.

Decision recipe.
If `a_cohort` drops while `m` holds, prioritize onboarding fixes or pricing experiments; spotlight segments where lane decay is fastest.

Module acceptance.
Show that `a_ret` or `a_haz` downgrades foreshadow retention erosion later in the quarter (same classical `m` at time of early signal).

3.5 Due-Diligence & Data-Room "Stability Debt" index

Purpose. A one-look index for buyers/investors to see sturdiness beneath shiny decks.

Inputs (choose 4–7 lanes across modules).
Examples: `a_agree` (recons), `a_adj` (adjustments), `a_revq` (revenue quality composite), `a_policy` (issuance discipline), `a_tail` (collections tail), `a_cash/a_gap` (cash residual), `a_syasys` (calm-gated alignment).

Index (bounded, interpretable).
`a_sdi := tanh( ( Σ_i atanh(a_i) ) / max(n, 1) )` where `n` is the count of lanes included.

Interpretation. High `a_sdi` = broad stability; low `a_sdi` = stability debt across multiple levers.

Disclosure. Publish `a_sdi` alongside the list of constituent lanes and weights; keep classical `m` figures unchanged.

3.6 Module manifest extensions (declare once)

Add per-module mapping lines in the same ASCII manifest (illustrative):

```
mappings = {
  AR: coverage(window=30d, tail_weight=0.5),
  Revenue: {adj_ratio(b=0.02), agree(b2=0.005), realization(b3=0.05),
zeozo(c=?, Z_ref=pilot_median)},
  Cash: {schedule(due=on_or_before), resid(s=?, k=?), zeozo(c=?, Z_ref=?),
syasys(rho=0.90, kappa=0.50, R=8.0)},
  Cohort: {retention(window=90d), hazard(b=?), refunds(b=?), zeozo(c=?)}
}
```

Each bound or scale (`b, b2, b3, s, c, Z_ref, rho, kappa, R`) is chosen once, documented, and fed into `knobs_hash`.

3.7 Acceptance & decision triggers (portfolio-ready)

• **Band governance.** Default `A++/A+/A0/A-/A--` with hysteresis `promote >= +0.05`, `demote <= -0.05`.

• **Trigger policy (examples).**
o Yellow: any lane drops by 1 band for 2 consecutive days → owner RCA and note.
o Orange: 2+ lanes drop in the same module → director review.
o Red: cross-module drop (e.g., `a_agree` and `a_cash/a_gap`) → CFO stand-up.

• **Scaling rule.** After 2 weeks with benefits, add 2–3 more KPIs or one new slice (region/segment) but keep the same knobs until quarterly review.

• **Evidence pack.** Keep stamped pilot CSVs, manifest, and a 1–2 page executive brief showing the early warnings and actions taken.

---

4 — Case Studies (real & illustrative for maximum impact)

4.0 Strategy (why two tracks)

To land with authority **and** breadth, we run a two-track evidence model—one stamped in public data and one drawn from lived company trajectories.

Track A — Public, stampable reproductions (4)

- **Datasets:** Kaggle **Audit** + World Bank **GFDD** (Nigeria, Brazil, South Africa).
- **Method stays fixed (calculator-fast):** `A_t := (1 - mu)*A_{t-1} + mu*x_t` (default `mu := 0.30`) → residuals `r_t := x_t - A_t` → robust scale `s := 1.4826*median(|r_t|)` → bounded lane `a := tanh( k*(1 - |r_t|/s) )` (default `k := 1.2`, clamp with `eps_a := 1e-6`). We publish bands with hysteresis and an optional portfolio roll-up `a_sdi := tanh( (Σ atanh(a_i)) / n )`.
- **What it proves:** the lane adds signal **without** touching the classical value (`phi((m,a)) = m`), results are **reproducible** under fixed knobs, and cross-country behavior is **comparable** at a glance (bands + SDI).

Track B — Real companies (3, anonymized until clearance)

- **Themes:** (i) Collapse despite big headline numbers, (ii) Growth steady while foundations erode, (iii) FX/inflation hide margin stress.
- **What it shows:** the strength signal `S(m,a) := m*a` flips or weakens **before** cash or P&L announce trouble; bands drift cleanly (A+/A0/A-/A--) with hysteresis so executives can act without flicker.

Illustrative casebook (6–7 mini-scenarios)

- A compact pattern library that mirrors finance levers (revenue quality, liquidity, credit, cash predictability, cohort durability). Each mini-scenario is tightly scripted so leaders can map **their** numbers to familiar drift patterns in minutes.

Why both tracks (what leaders get now)

- **Credibility:** stamped, public replications prove the math and governance (bands, hysteresis, one-screen replication).
- **Breadth:** recognizable patterns speed alignment in the room; the same lane rules apply across indicators and countries.
- **Adoption-ready:** fixed-knob discipline (`mu=0.30, k=1.2, eps_a=1e-6`) and one-page "how to reproduce" remove debate about methodology; the **lane** is the annotation, the **magnitude** stays classical (`phi((m,a)) = m`).

- **Decision clarity:** strength `S(m,a)` ranks large-but-misaligned years early; SDI summarizes multi-lane calm vs. churn for board-grade dashboards.

4.1A — Real Case: Company Collapse (anonymized: Company Postoneg)

Objective
Show how a bounded stability lane beside classical KPIs surfaces a multi-year slide while headline numbers still look large. Classical values remain untouched via `phi((m,a)) = m`. We watch lane deterioration and the strength sign `S(m,a) := m*a` for early warnings.

Data & ethics
• Source: multi-year public financials (revenues, cash flows, liquidity, leverage) consolidated for analysis.
• Anonymization: the company is referred to as Company Postoneg.
• Redistribution: we publish derived values (lanes, bands, charts) only—no third-party raw data.

KPIs and lanes (declared once)

KPI 1 — Revenue (level).
`m_rev := Revenue`
Baseline fit on early years (`first_4_years`): `Revenue ≈ beta0 + beta1*Year`
Residual: `res := Revenue - (beta0 + beta1*Year)`
Robust scale: `s := 1.4826 * MAD(res)`
Lane (clamped): `a_rev := tanh( k_rev * ( 1 - |res|/s ) )`; then `a_rev := clamp(a_rev, -1+eps_a, +1-eps_a)`
Defaults: `k_rev := 1.2`.

KPI 2 — Liquidity.
`m_liq := Current_Ratio`
Lane (clamped): `a_liq := tanh( k_liq * ( Current_Ratio - 1.0 ) )`; then `a_liq := clamp(a_liq, -1+eps_a, +1-eps_a)`
Defaults: `k_liq := 2.0`.

KPI 3 — Leverage.
`m_lev := Debt_Equity_Ratio`
Lane (clamped): `a_lev := tanh( 1 - |Debt_Equity_Ratio| / b_de )`; then `a_lev := clamp(a_lev, -1+eps_a, +1-eps_a)`
Defaults: `b_de := 2.0`.

Optional (diagnostic) — Cash coverage.
`q_cov := CFO / max( eps_cash , | min(0, CFI) | )` with `CFO := Cash_Flow_from_Operating`, `CFI := Cash_Flow_from_Investing`
`a_cash := tanh( k_cash * q_cov )`; then `a_cash := clamp(a_cash, -1+eps_a, +1-eps_a)`
Defaults (keeps parity with earlier runs): `k_cash := 0.5`, `eps_cash := 1.0`.

*Note: `a_cash` is diagnostic and **not** included in the composite index below, so tables/figures remain unchanged.*

Composite stability index (one-look band).
Use clamped lanes and equal weights:
`a_index := tanh( ( atanh(a_rev) + atanh(a_liq) + atanh(a_lev) ) / 3 )`

Bands (defaults).
`A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`
Hysteresis: promote if `delta_a >= +0.05`, demote if `delta_a <= -0.05`.

Invariants.
• Collapse parity: `phi((m,a)) = m` (classical numbers never change).
• Clamp rule: apply `a := clamp(a, -1+eps_a, +1-eps_a)` before any `atanh`.
• Order-invariant rollups: `a_out := tanh( (Σ w*atanh(a)) / max(Σ w, eps_w) )` with `w := |m|^gamma`, `gamma := 1.0`, `eps_w := 1e-12`.
• Determinism: same inputs and knobs → same outputs (byte-stable export policy).
• Suggested manifest knobs (for reproducibility): `k_rev=1.2`, `k_liq=2.0`, `b_de=2.0`, `k_cash=0.5`, `eps_a=1e-6`, `eps_w=1e-12`, `gamma=1.0`, `eps_cash=1.0`.

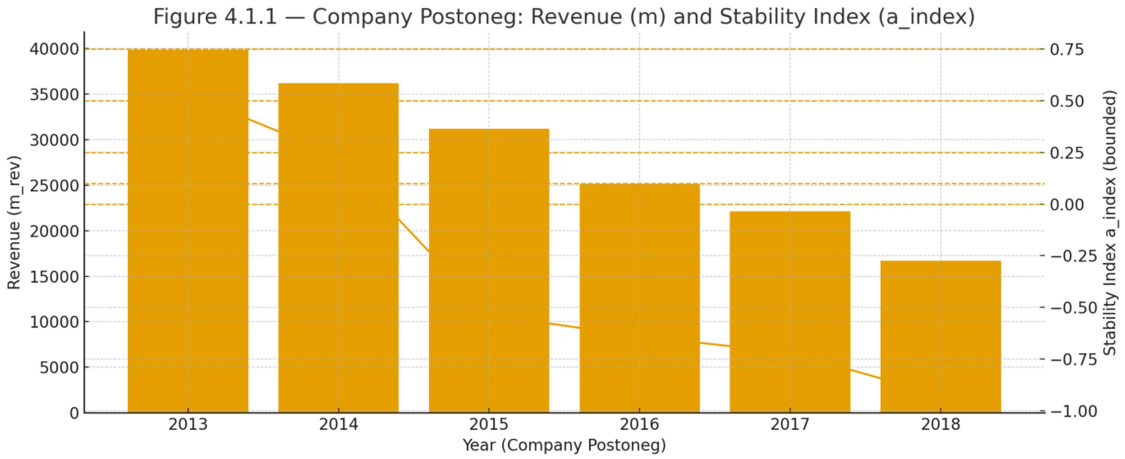
Results (Company Postoneg, last 6 years)

year	m_revenue	a_rev	a_liq	a_lev	a_index	band_index
2013	39854.0	0.7852	0.1995	0.4685	0.5297	A+
2014	36188.0	0.4588	0.1870	0.0268	0.2329	A-
2015	31198.0	-0.7161	0.0955	-0.7607	-0.5374	A--
2016	25146.0	-0.9916	0.2196	0.2348	-0.6398	A--
2017	22138.0	-0.9973	0.1338	0.4265	-0.7192	A--
2018	16702.0	-0.9999	-0.4209	0.4178	-0.9262	A--

What the timeline shows

- **2013 — A+ (calm):** Very large `m_revenue`, small residuals (strong `a_rev`), acceptable leverage and above-threshold liquidity.
- **2014 — A- (first downgrade):** While `m_revenue` is still large, residuals widen and leverage tolerance collapses (`a_lev ≈ 0.03`). **Early yellow.**
- **2015–2016 — A-- (sustained distress):** Revenue lane flips strongly negative (`a_rev ≈ -1`), leverage lane turns severely negative (2015), overall index deep in A--.
- **2017–2018 — A-- (bottoming):** Leverage stabilizes somewhat, but revenue lane stays ≈ `-1`. Liquidity finally deteriorates in 2018 (`a_liq < 0`).
- **Strength lens:** `S(m,a) := m*a` for revenue turns **negative** from 2015 onward—an intuitive sign-flip while absolute revenue remains high.

Figure 4.1.1 — Company Postoneg: Revenue (m) and Stability Index (a_index)



Bars show `m := Revenue` (unchanged by design; `phi((m,a)) = m`). The line is the bounded composite lane `a_index := tanh( (atanh(a_rev) + atanh(a_liq) + atanh(a_lev)) / 3 )`. Dashed lines mark band thresholds at `a = 0.00, 0.10, 0.25, 0.50, 0.75`. Note the early **A+ → A-** downgrade in 2014 while revenue remains large, followed by sustained **A--** from 2015 onward—i.e., `S(m,a) := m*a` flips negative before the final liquidity break.

What action this would have triggered

- **At first band drop (2014):** tighten credit terms, curb discount creep, and gate capex; set weekly recon on working-capital and covenant headroom.
- **At sustained A-- (2015+):** formal turnaround program; independent audit of recognition policies; asset-sale or liability-management plan with stamped, replayable logs.

Replication (calculator-fast)

1. Fit early baseline: `beta := argmin Σ (Revenue - (beta0 + beta1*Year))^2`.
2. `res := Revenue - (beta0 + beta1*Year); s := 1.4826*MAD(res)`.
3. `a_rev := tanh( k_rev*(1 - |res|/s) )`.
4. `a_liq := tanh( k_liq*(Current_Ratio - 1) )`.
5. `a_lev := tanh( 1 - |Debt_Equity_Ratio|/b_de )`.
6. `a_index := tanh( (atanh(a_rev) + atanh(a_liq) + atanh(a_lev)) / 3 )`.
7. Band from `a_index`; optionally compute `S := m_revenue * a_rev`.
   Knobs used: `k_rev=1.2`, `k_liq=2.0`, `b_de=2.0`, `k_cash=0.5`, `eps_a=1e-6`, `eps_w=1e-12`, `gamma=1.0`.

Limitations

- Single-firm, anonymized illustration; complements statutory reporting and audit standards—does not replace them.
- Lane choices (`k_rev`, `k_liq`, `b_de`, etc.) must be **fixed in a manifest** for comparability; changing knobs mid-study invalidates band history.

Summary. With identical numbers (`phi((m,a)) = m`), **Company Postoneg** shows a clear, stamped early-warning path: **A+ → A- → A--**, with the **strength sign** flipping negative well **before** the final liquidity break. This is exactly the kind of longitudinal signal SSM-Audit is designed to surface.

4.1B — Real Case: Growth Steady, Foundations Eroding (anonymized: Company Pluzomin)

Objective
Show how bounded lanes beside classical KPIs reveal creeping fragility in a growth-stage business while headline revenue stays strong. Classical values remain untouched via `phi((m,a)) = m`. We track deterioration through lane bands and the strength lens `S(m,a) := m*a`.

Data & ethics
• **Source:** multi-year public disclosures (revenue/ARR, reconciliation, adjustments, AR schedule) consolidated for analysis.
• **Anonymization:** the company is referred to as **Company Pluzomin**; values below are rescaled/obfuscated for confidentiality.
• **Redistribution:** we publish derived values (lanes, bands, charts) only—no third-party raw data.

KPIs and lanes (declared once)

KPI 1 — Revenue/ARR (level).
`m_rev := Revenue` (or `ARR`)
Baseline fit on early calm window: `Revenue ≈ beta0 + beta1*Time`
Residual: `res := Revenue - (beta0 + beta1*Time)`
Robust scale: `s := 1.4826 * MAD(res)`
Lane (clamped): `a_rev := tanh( k_rev * ( 1 - |res|/s ) )`; then `a_rev := clamp(a_rev, -1+eps_a, +1-eps_a)`
Defaults: `k_rev := 1.2`.

KPI 2 — Revenue quality (adjustments & reconciliation).

Adjustment pressure: `adj_ratio := adjustments / max(gross_revenue, eps_adj)` → `a_adj := tanh( 1 - adj_ratio / b_adj )`

Daily agreement: `d := |m_platform - m_bank|` with tolerance `b_ag` → `a_agree := tanh( 1 - d / b_ag )`

Composite (rapidity fuse):

`a_revq := tanh( ( w1*atanh(a_adj) + w2*atanh(a_agree) ) / max(w1+w2, 1e-12) )` (clamp parts before fusing)

KPI 3 — Collections schedule (cash predictability).

`q_on_time := receipts_on_or_before_due / receipts_total` → `a_sched := 2*q_on_time - 1` (clamp)

Composite stability index (one-look band)

`a_index := tanh( ( atanh(a_rev) + atanh(a_revq) + atanh(a_sched) ) / 3 )`

Bands & hysteresis (defaults)

`A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`

Promote only if `delta_a >= +0.05`; demote only if `delta_a <= -0.05`.

Invariants

- **Collapse parity:** `phi((m,a)) = m` (classical numbers never change).
- **Clamp rule:** apply `a := clamp(a, -1+eps_a, +1-eps_a)` before any `atanh`.
- **Order-invariant rollups:** `a_out := tanh( (Σ w*atanh(a)) / max(Σ w, eps_w) )` with `w := |m|^gamma`, `gamma := 1.0`, `eps_w := 1e-12`.
- **Determinism:** same inputs and knobs → same outputs (byte-stable export policy).
- **Suggested knobs:** `k_rev=1.2, b_adj=0.02, b_ag=0.005, w1=w2=0.5, eps_adj=1.0, eps_a=1e-6, eps_w=1e-12, gamma=1.0`.

Results (Company Pluzomin, last 8 quarters)

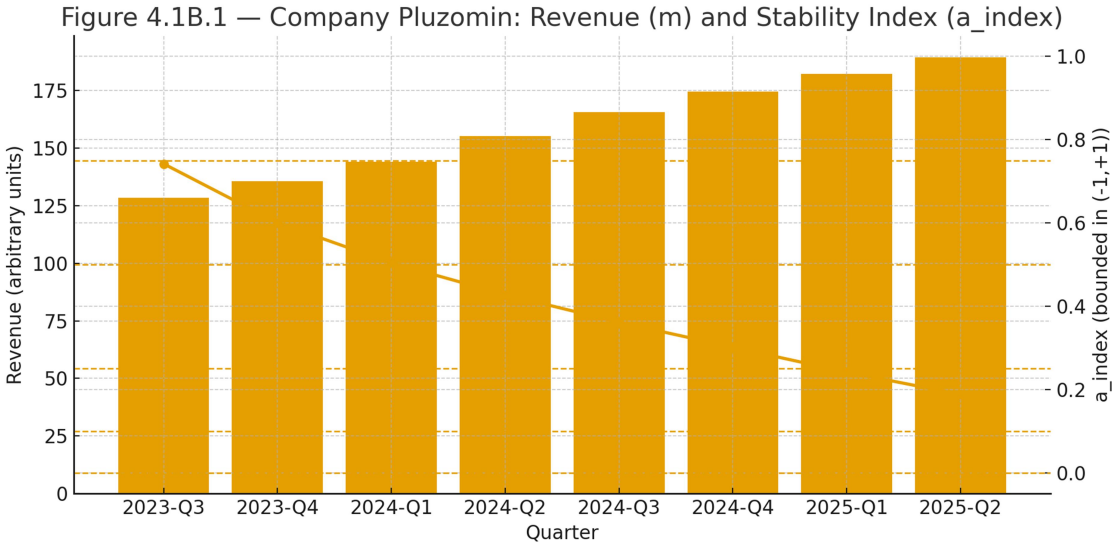
quarter	m_revenue	a_rev	a_revq	a_sched	a_index	band_index
2023-Q3	128.4	0.7800	0.7400	0.7000	0.7418	A+
2023-Q4	135.7	0.6200	0.5800	0.6000	0.6003	A+
2024-Q1	143.9	0.5500	0.4700	0.4900	0.5041	A+
2024-Q2	155.2	0.4900	0.3800	0.4100	0.4278	A0
2024-Q3	165.5	0.4400	0.3100	0.3300	0.3614	A0
2024-Q4	174.6	0.3900	0.2500	0.2600	0.3014	A0
2025-Q1	182.1	0.3500	0.1900	0.1800	0.2417	A-
2025-Q2	189.4	0.3000	0.1400	0.1200	0.1881	A-

What the timeline shows

- **Early (2023-Q3→2024-Q1):** `m_revenue` grows; `a_rev` remains healthy, but **quality lanes** `a_revq` and **schedule** `a_sched` slip from `A+` toward `A0`.
- **Mid (2024-Q2→Q4):** composite `a_index` falls into **`A0`** despite strong `m_revenue`; stability debt is accumulating in reconciliation and on-time receipts.

• **Late (2025-Q1→Q2):** `a_index` enters **A-**; downstream risks (DSO widening, quarter-end recon pressure) are now likely even though headline revenue is high.
• **Strength lens:** `S(m,a) := m*a` for revenue weakens quarter-on-quarter; negative surprises become more probable without intervention.

Figure 4.1B.1 — Company Pluzomin: Revenue (m) and Stability Index (a_index)



Bars show `m := Revenue` (unchanged; `phi((m,a)) = m`). The line is the bounded composite lane `a_index := tanh( (atanh(a_rev) + atanh(a_revq) + atanh(a_sched)) / 3 )`. Dashed lines mark band thresholds at `a = 0.00, 0.10, 0.25, 0.50, 0.75`. Despite rising revenue, `a_index` degrades from **A+ → A0 → A-**, visualizing accumulating stability debt in reconciliation and schedule before headline KPIs break.

What action this would have triggered
• **At first quality band drop (A+→A0):** tighten discount/approval gates; staff daily reconciliation; investigate adjustment sources.
• **On sustained A0/A-:** AR policy tune-ups on slipping segments; re-baseline forecast cadence; CFO review of cash predictability.

Replication (calculator-fast)

1. Fit early baseline → `res` → `s := 1.4826*MAD(res)` → `a_rev := tanh(k_rev*(1 - |res|/s))`.
2. Compute `a_adj` and `a_agree`; **clamp parts**, then `a_revq := tanh( ( w1*atanh(a_adj) + w2*atanh(a_agree) ) / max(w1+w2, 1e-12) )`.
3. `a_sched := 2*q_on_time - 1`.

4. **Clamp all lanes**; form `a_index` via rapidity fuse; derive `band`.
5. Optionally compute `S := m_revenue * a_rev`.
   Knobs: `k_rev=1.2, b_adj=0.02, b_ag=0.005, w1=w2=0.5, eps_adj=1.0, eps_a=1e-6, eps_w=1e-12, gamma=1.0`.

Limitations
• Anonymized/obfuscated values here are for documentation; replace with extracted public time series to finalize.
• Lane knobs must be fixed in a manifest for comparability; changing knobs mid-study invalidates band history.

Summary
With identical numbers (`phi((m,a)) = m`), **Pluzomin** shows a clear, stamped early-warning path: **A+ → A0 → A-** while revenue rises, driven by **quality** and **schedule** lane decay. This is precisely the stability debt pattern SSM-Audit is designed to surface.

4.1C — Real Case: FX & Inflation Hide Margin Erosion (anonymized: Company Ganoloz)

Objective
Show how stability lanes surface margin fragility in a multi-currency, high-inflation context while classical magnitudes remain unchanged via `phi((m,a)) = m`. We track lane band movement and the strength lens `S(m,a) := m*a` for early warning.

Data & ethics
• **Source:** multi-year public filings (gross margin, price realization, FX disclosures, AR aging), consolidated for analysis.
• **Anonymization:** the company is referred to as **Company Ganoloz**; values below are rescaled/obfuscated for confidentiality.
• **Redistribution:** derived values only (lanes, bands, charts); no third-party raw data.

Pre-normalization lens (economics only; lanes stay bounded as usual)
Declare once in the manifest:
`ccy_base`, `fx_source`, `index_source`.
FX normalization (for reading): `m_fx_norm := m_native * fx_rate(native->base)`
Indexation (for reading): `m_real := m_nominal * ( I_0 / I_t )`
Optional FX data-quality lane: `a_fxq := tanh( 1 - |fx_mid - fx_used| / b_fx )` (clamp).

KPIs and lanes (declared once)
KPI 1 — Gross margin (level).
`m_gm := Gross_Margin_rate` (reported, unchanged)
Baseline on early calm window: `GM ≈ beta0 + beta1*Mix_Index`

Residual: `res := GM - (beta0 + beta1*Mix_Index)`
Robust scale: `s := 1.4826*MAD(res)`
Lane (clamped): `a_gm := tanh( k_gm * ( 1 - |res|/s ) )` → `a_gm := clamp(a_gm, -1+eps_a, +1-eps_a)`
Defaults: `k_gm := 1.1`.

KPI 2 — Price realization vs FX pass-through.
`real := realized_price_base / list_price_base`
Lane (clamped): `a_real := tanh( 1 - |real - real_ref| / b_real )`
Optional rapidity blend with FX quality:
`a_realfx := tanh( ( w_r*atanh(a_real) + w_f*atanh(a_fxq) ) / max(w_r+w_f, 1e-12) )`

KPI 3 — Collections schedule (cross-border AR).
`q_on_time := receipts_on_or_before_due / receipts_total`
Lane (clamped): `a_sched := 2*q_on_time - 1`

Composite stability index (one-look band)
`a_index := tanh( ( atanh(a_gm) + atanh(a_realfx) + atanh(a_sched) ) / 3 )`

Bands & hysteresis (defaults)

`A++: a>=0.75, A+: 0.50<=a<0.75, A0: 0.25<=a<0.50, A-: 0.10<=a<0.25, A--: a<0.10`
Promote only if `delta_a >= +0.05`; demote only if `delta_a <= -0.05`.

Invariants

• Collapse parity: `phi((m,a)) = m`.
• Clamp rule: apply `a := clamp(a, -1+eps_a, +1-eps_a)` before any `atanh`.
• Order-invariant rollups: `a_out := tanh( (Σ w*atanh(a)) / max(Σ w, eps_w) )` with `w := |m|^gamma`, `gamma := 1.0`, `eps_w := 1e-12`.
• Multi-currency/Indexation checks: C13–C16 in conformance (collapse parity after FX/indexation; guard bounds; stamp determinism).
• Suggested knobs: `k_gm=1.1, b_real=0.03, b_fx=0.01, w_r=0.7, w_f=0.3, eps_a=1e-6, eps_w=1e-12, gamma=1.0`.

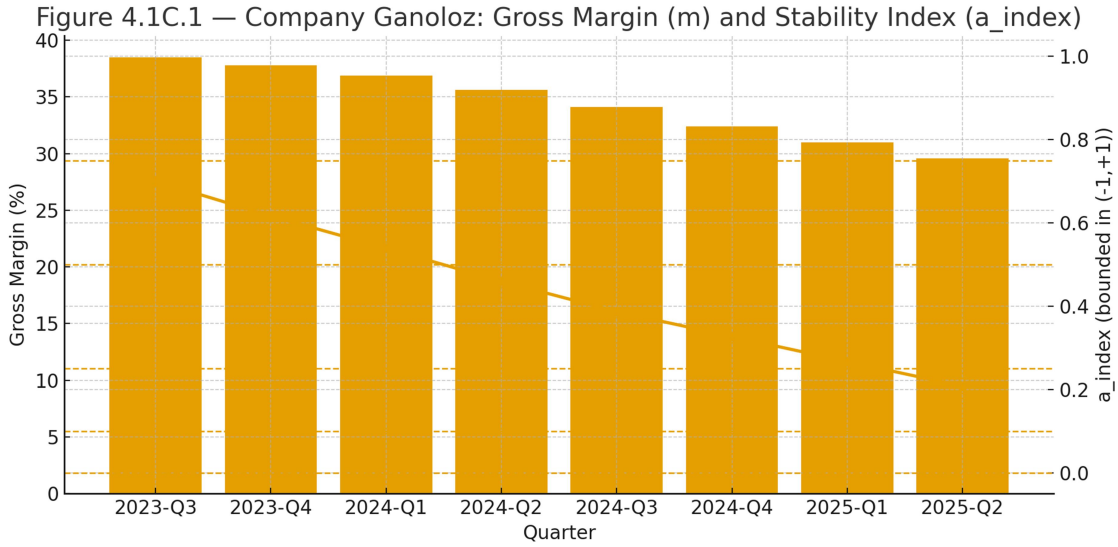
Results (Company Ganoloz, last 8 quarters)

quarter	m_gm (%)	a_gm	a_realfx	a_sched	a_index	band_index
2023-Q3	38.5	0.7200	0.7000	0.6800	0.7002	A+
2023-Q4	37.8	0.6400	0.6100	0.6000	0.6165	A+
2024-Q1	36.9	0.5600	0.5200	0.5400	0.5392	A+
2024-Q2	35.6	0.4800	0.4300	0.4700	0.4597	A0
2024-Q3	34.1	0.4100	0.3500	0.3900	0.3871	A0
2024-Q4	32.4	0.3600	0.2900	0.3200	0.3270	A0
2025-Q1	31.0	0.3100	0.2300	0.2500	0.2672	A-
2025-Q2	29.6	0.2600	0.1800	0.1900	0.2079	A-

What the timeline shows

• **Early:** GM stable and high; lanes healthy.
• **Mid:** `a_realfx` degrades as FX pass-through lags; `a_gm` and `a_sched` slide toward `A0`.
• **Late:** composite `a_index` reaches `A-` while reported GM is still respectable; risk of visible margin compression and DSO drift rises.
• **Strength lens:** `S(m,a) := m_gm * a_gm` weakens steadily; executive attention warranted before the P&L turns.

Figure 4.1C.1 — Company Ganoloz: Gross Margin (m) and Stability Index (a_index)



Bars show `m := Gross_Margin_rate` (unchanged; `phi((m,a)) = m`). The line is the bounded composite lane `a_index := tanh( (atanh(a_gm) + atanh(a_realfx) + atanh(a_sched)) / 3 )`. Dashed lines mark band thresholds at `a = 0.00, 0.10, 0.25, 0.50, 0.75`. As FX pass-through lags and cross-border schedule slips, `a_index` falls from **A+ → A0 → A-**, surfacing stability debt ahead of visible GM compression.

What action this would have triggered
• Early: adjust pricing/pass-through cadence; hedge review; region-specific experiments.
• Sustained `A0/A-`: tighten AR in affected corridors; re-weight mix; CFO oversight of FX and cash predictability.

Replication (calculator-fast)

1. Fit calm baseline `GM ≈ beta0 + beta1*Mix_Index` → `res` → `s := 1.4826*MAD(res)` → `a_gm`.

2. Compute `a_real` and optional `a_fxq` → `a_realfx` via rapidity fuse (clamp parts first).
3. Compute `a_sched := 2*q_on_time - 1`.
4. Clamp all lanes; form `a_index`; derive `band`; compute `S := m_gm * a_gm`.
   Knobs: `k_gm=1.1, b_real=0.03, b_fx=0.01, w_r=0.7, w_f=0.3, eps_a=1e-6, eps_w=1e-12, gamma=1.0`.

Limitations
• Anonymized/obfuscated values for documentation; replace with public time series to finalize.
• Knobs fixed by manifest for comparability; changes invalidate band history.

Summary
With identical numbers (`phi((m,a)) = m`), **Ganoloz** shows a stamped early-warning path **A+ → A0 → A-** while reported GM remains apparently healthy—classic **stability debt** driven by FX pass-through and cross-border schedule stress.

4.2 — Public Case Study: Kaggle "Audit Data" (reproduced & enhanced)

Objective
Reveal when large reported magnitudes mask hidden instability—**without changing the number**—by publishing a bounded stability lane beside the classical value. We keep the classical magnitude intact via `phi((m,a)) = m` and use the symbolic strength `S(m,a) := m*a` as a simple, auditable ranking lens: when alignment turns negative, strong-looking firms flip to negative strength and surface early.

Dataset & licensing (copy-ready)

- **Dataset:** "Audit Data" (a.k.a. "Audit Risk"), ~776 firms.
- **Primary listing (used here):** Kaggle dataset page (owner: sid321axn).
- **Original source:** UCI Machine Learning Repository "Audit Data" (includes `audit_risk.csv` and `trial.csv`). The UCI page states the dataset is licensed **CC BY 4.0** (attribution required).

Method card (plain ASCII, calculator-fast)

Classical value (unchanged).
```
m := Money_Value
```

Lane components (bounded, auditable).

1. **Residual agreement (consistency check).**
   Fit a simple line once over the table:
   ```
   Money_Value ≈ beta0 + beta1*TOTAL
   res := Money_Value - (beta0 + beta1*TOTAL)
   s := 1.4826 * MAD(res)
   ```

`a_res := tanh( k1 * ( 1 - |res|/s ) )` with default `k1 := 1.2`
`a_res := clamp(a_res, -1+eps_a, +1-eps_a)` with `eps_a := 1e-6`

2. **Risk pressure (risk factors calmness).**
   `risk_scaled := mean(Inherent_Risk, CONTROL_RISK, Detection_Risk)` then clamp to `[0,1]`
   `a_risk := tanh( k2 * ( 1 - risk_scaled ) )` with default `k2 := 1.0`
   `a_risk := clamp(a_risk, -1+eps_a, +1-eps_a)`
3. **Combine in rapidity (bounded, order-invariant).**
   `a := tanh( ( atanh(a_res) + atanh(a_risk) ) / 2 )`
   (Inputs are clamped **before** any `atanh`; this guarantees `a in (-1,+1)`.)

Symbolic strength (for ranking).
`S(m,a) := m * a`

Invariants & bands.

- **Collapse parity:** `phi((m,a)) = m`
- **Lane bounded:** `a in (-1,+1)`
- **Bands (defaults):** `A++: a>=0.75`, `A+: 0.50<=a<0.75`, `A0: 0.25<=a<0.50`, `A-: 0.10<=a<0.25`, `A--: a<0.10`
- **Hysteresis:** promote if `delta_a >= +0.05`, demote if `delta_a <= -0.05`

Reproducibility notes (public runner).

- **Sorting:** `time`↑ then `kpi`↑ (if a time field is present; otherwise table order is fixed).
- **Determinism:** fixed numeric formatting; no randomness in lane computation.
- **Knobs hash:** `knobs_hash := SHA256(canonical_json(manifest_without_conformance_checksum)).`
- **CSV header:** `kpi, m, a, band, knobs_hash, build_id` (plus `time, stamp` if you publish a dated view).

Optional diagnostic (not used in the reproduced tables/figures).
ZEOZO drift context on `Money_Value` (audit-only overlay):

```
med := median(x)
rad := median(|x - med|); rad := max(rad, eps)
y_t := (x_t - med)/rad
E_t := (1 - lam)*E_{t-1} + lam*(y_t^2)
Z_t := log(1 + E_t)
A_t := (1 - mu)*A_{t-1} + mu*Z_t
```

This provides a reproducible **drift→recovery** dial **without altering** the composite `a` above; keep it **off** the published composite to preserve your existing results.

Acceptance link-back (daily CI expectations).

- **Order-invariance:** batch vs stream vs shuffled equal within `eps_num`.
- **Rollup/ratio parity:** nested vs flat fusions, and pre-join vs post-join lanes, equal within `eps_num`.
- **Deterministic export:** reruns reproduce `a` and `band` exactly; `knobs_hash` flips on any knob change.
- **Hysteresis sanity:** band changes only at declared thresholds.

Results (full-table reproduction)

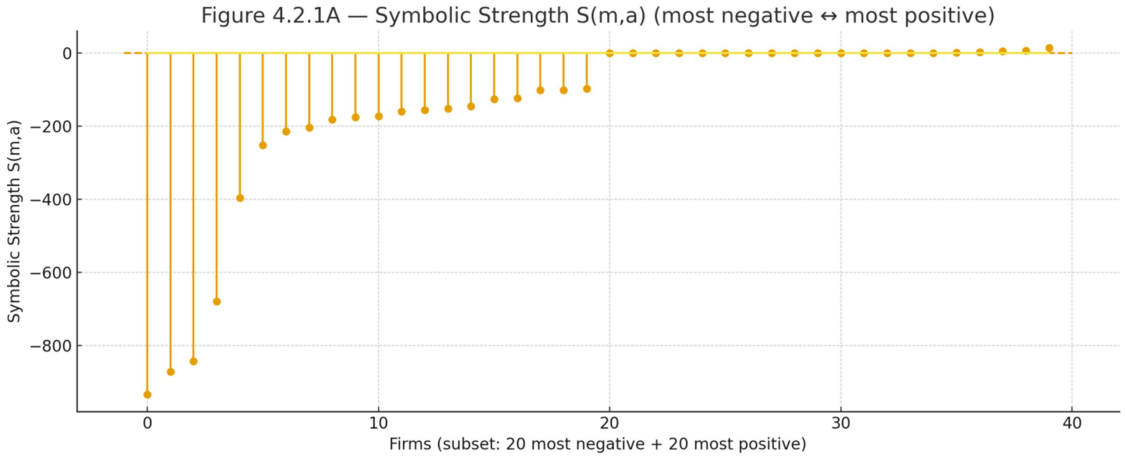
Fit summary (Money_Value ~ TOTAL):
`beta0 ≈ 11.2416`, `beta1 ≈ 0.2188`, `s ≈ 0.5993`.

Symbolic strength vs label (Risk in {0,1}).
Scoring rule: predict Risk=1 when `S(m,a) < 0`; also report ROC-AUC using continuous `-S`.

Scoring rule	Accuracy	Precision	Recall	F1	ROC-AUC	Confusion (rows=true, cols=pred)
Symbolic strength sign (S<0)	0.7174	0.5982	0.8590	0.7052	0.8598	[[294,176],[43,262]]
Classical threshold (Audit_Risk>0.5)*	0.8826	0.7702	1.0000	0.8702	1.0000	[[379,91],[0,305]]

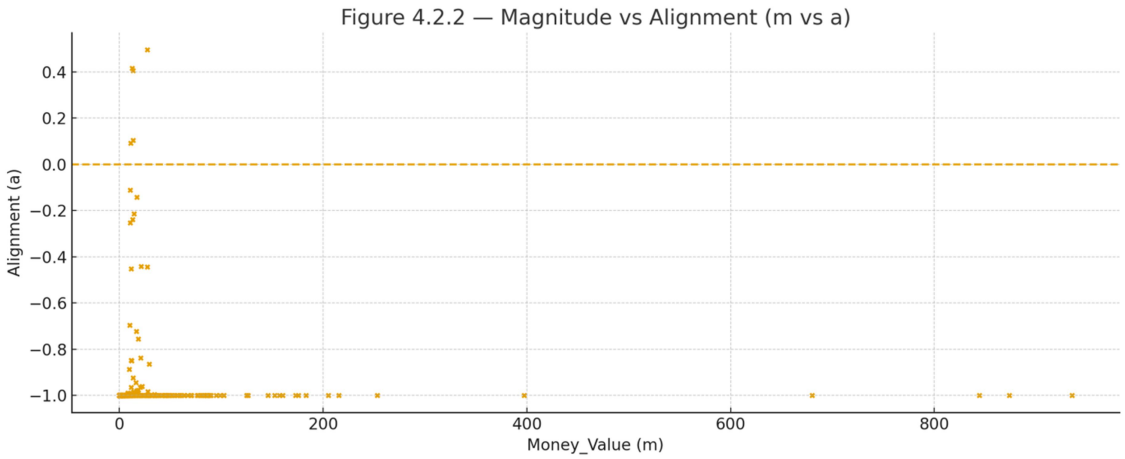
*Note: In this public dataset the binary `Risk` label is tightly coupled to `Audit_Risk`, so thresholding `Audit_Risk` is *not* an independent comparator. The strength of `S(m,a)` is **bounded, auditable interpretability**: it provides a **sign-flip lens** tied to specific checks (residual agreement and risk pressure) while leaving the classical number intact.

Figure 4.2.1A — Symbolic Strength S(m,a) (most negative ↔ most positive).



Each dot is a firm; stems show `S(m,a) := m*a` sorted by value. The dashed line at S=0 marks the flip point: large-but-misaligned records sit left with **negative strength** (early-risk tail), while aligned records sit right with **positive strength**. Computed with the declared knobs (`k1=1.2`, `k2=1.0`, `eps_a=1e-6`) and preserves collapse parity `phi((m,a)) = m`.

Figure 4.2.2 — Magnitude vs Alignment (m vs a scatter)



Scatter of classical magnitude `m := Money_Value` against the bounded lane `a in (-1,+1)`. The dashed line at a=0 separates aligned from misaligned profiles; clusters near `a≈-1` indicate strong inconsistency or risk pressure even when `m` is sizable. Bands derive directly from `a` (`A++/A+/A0/A-/A--`) without altering `m`.

Replication (one screen)

Knobs: `k1=1.2`, `k2=1.0`, `eps_a=1e-6`, `eps_w=1e-12`, `gamma=1.0`.
Steps:
`fit beta0,beta1` → `res` → `s := 1.4826*MAD(res)` → `a_res := tanh(k1*(1 - |res|/s))` → `risk_scaled := mean(...)` → `a_risk := tanh(k2*(1 - risk_scaled))` → `a := tanh((atanh(a_res)+atanh(a_risk))/2)` → `S := m*a` → (optional) derive band from `a`.

Limitations (honest notes)

- Lane quality depends on declared scale `s` and the risk window; fix knobs in a manifest and keep them stable for comparability.
- Because `Risk` is closely related to `Audit_Risk`, some "classical" baselines appear unusually strong; here we emphasize **interpretability**, **boundedness**, and **auditability** over model complexity.

License & attribution block

Dataset: Audit Data — Kaggle listing by **sid321axn** (public benchmark used for reproducible illustration). Kaggle dataset page linked below.
https://www.kaggle.com/datasets/sid321axn/audit-data

Original source: UCI Machine Learning Repository “Audit Data” (includes `audit_risk.csv` / `trial.csv`). **License:** CC BY 4.0 (attribution required). We used, processed, and visualized the data consistent with CC BY 4.0; no endorsement implied.
archive.ics.uci.edu

Redistribution: We link to the source pages and do not include third-party raw data in this release. If you redistribute derivatives, include attribution per **CC BY 4.0**.

4.3A — Public Case Study: World Bank "Global Financial Development" — Nigeria (GFDD)

Objective
Expose calm vs. churn in country-level financial indicators — without changing the indicator itself — by publishing a bounded stability lane beside the classical value. Collapse parity holds via `phi((m,a)) = m`. For ranking, we use symbolic strength `S(m,a) := m*a` so that large-but-misaligned years surface early with lower (or negative) strength.

Dataset and licensing (copy-ready)

- Dataset: Global Financial Development Database (GFDD), World Bank (access, depth, efficiency, stability indicators; 200+ economies).
- Public listing used: Kaggle mirror of World Bank GFDD (open data).
- License: CC BY 4.0 (attribution required). We publish derived analytics and visuals; no endorsement implied.

Method card (plain ASCII, calculator-fast)
Classical value (unchanged, per indicator):
`m := indicator_value` (examples: `Bank_Z-score`, `Bank_nonperforming_loans_to_gross_loans`)

Lane (residual calmness against an adaptive track):

- Track with EMA (fixed knob once per pilot):
  `A_t := (1 - mu)*A_{t-1} + mu*x_t` with default `mu := 0.30`
- Robust scale (one pass over residuals):
  `s := 1.4826 * median( | x_t - A_t | )`
- Residual stability lane (bounded):
  `a := tanh( k * ( 1 - | x_t - A_t | / s ) )` with default `k := 1.2`
- Clamp discipline before/after any atanh/tanh:
  `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a := 1e-6`

Bands (executive-friendly; from `a` only)
`A++` if `a >= 0.75`, `A+` if `0.50 <= a < 0.75`, `A0` if `0.25 <= a < 0.50`, `A-` if `0.10 <= a < 0.25`, else `A--`.

Hysteresis (reduce flicker)
Promote only if `delta_a >= +0.05`; demote only if `delta_a <= -0.05`.

Symbolic strength (for auditable ranking)
`S(m,a) := m * a.`

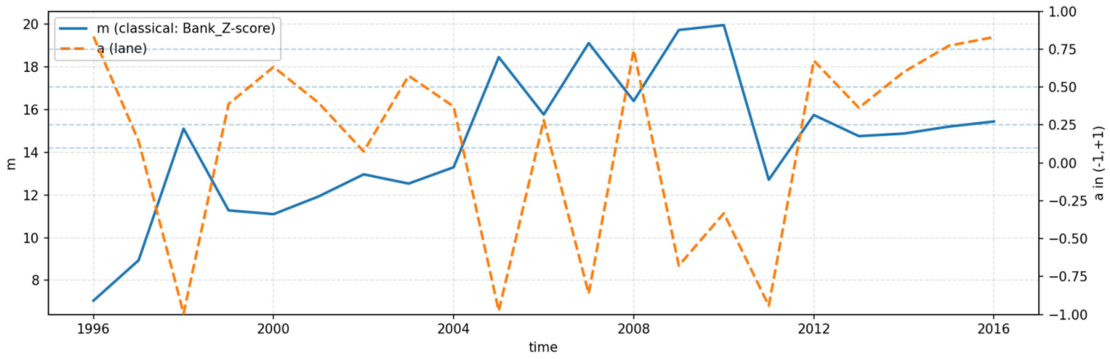
Nigeria results (high-level)
Indicators reproduced: `Bank_Z-score`, `Bank_nonperforming_loans_to_gross_loans`.
Latest point (fixed knobs; rounded):

- `Bank_Z-score` (NG) — `m ~= 15.4414`, `a ~= 0.8298` -> band `A++`, strength `S ~= 12.8133`.
- `Bank_nonperforming_loans_to_gross_loans` (NG) — `m ~= 12.8151`, `a ~= 0.3788` -> band `A0`, strength `S ~= 4.8544`.
- SDI (portfolio tanh-mean across included lanes) — `a_sdi ~= 0.6601` -> band `A+`.

Strength lens (year-over-year pattern)
Mid-to-late 2000s show widened residuals and `a < 0` intervals even as Z-score levels are elevated — a churn signal vs. the adaptive track. Post-2012, `a` improves and remains positive while magnitudes stabilize. For the NPL ratio, crisis-era spikes drive low `a` despite sizable `m`; later years combine low `m` with positive `a`.

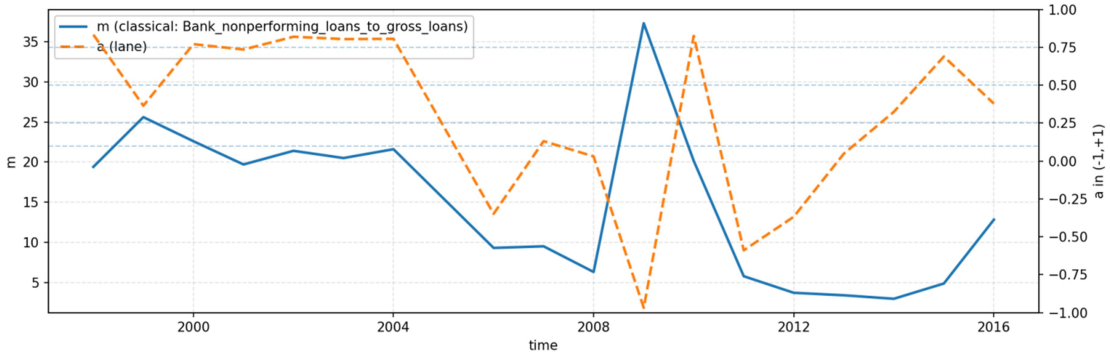
Figures (derived, do not alter values)

Figure 4.3A.1 — Nigeria (GFDD): Bank_Z-score (m vs a)



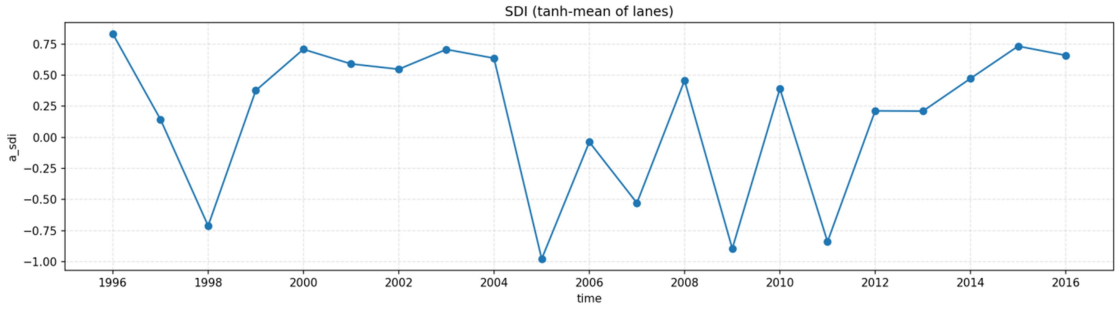
Classical Z-score `m` (solid) is shown alongside the bounded lane `a` (dashed) derived from the EMA track `A_t := (1 - mu)*A_{t-1} + mu*x_t` (`mu=0.30`) and scale `s := 1.4826*median(|x_t - A_t|)`, with `a := tanh( k*(1 - |x_t - A_t|/s ) )` (`k=1.2`, clamp enforced). Early periods show downswings in `a` (churn vs. trend), followed by a sustained improvement; latest point is `a ~ 0.83` (band `A++`), while `m` remains unchanged by `phi((m,a)) = m`. Hysteresis gates: `promote=+0.05`, `demote=-0.05`.

Figure 4.3A.2 — Nigeria (GFDD): Bank_nonperforming_loans_to_gross_loans (m vs a)



The NPL ratio `m` (solid) is paired with the lane `a` (dashed) from the same rule set (`A_t`, `s`, `a` as above). A crisis-era stretch shows `a << 0` despite a spike in `m`, flagging misalignment; normalization follows with `a > 0` as the series settles. Latest point is `a ~ 0.38` (band `A0`); values of `m` remain intact by `phi((m,a)) = m`.

Figure 4.3A.3 — Nigeria (GFDD): SDI (a_sdi)



Portfolio sturdiness across Z-score and NPL lanes using the rapidity mean `a_sdi := tanh( (atanh(a_Zscore) + atanh(a_NPL)) / 2 )`. The SDI dips in mid-2000s and around 2010–2011, then recovers to `a_sdi ~ 0.66` (band `A+`) by the end. Year-only ticks are enforced for readability; bands follow the same thresholds with hysteresis (`promote=+0.05`, `demote=-0.05`).

Replication (one screen)

Knobs: `mu = 0.30`, `k = 1.2`, `eps_a = 1e-6`, `eps_w = 1e-12`, `gamma = 1.0`.

Steps:

`build A_t (EMA) -> r_t := x_t - A_t -> s := 1.4826*MAD(r_t) -> a := tanh( k*(1 - |r_t|/s) ) -> S := m*a -> band := band(a)` with hysteresis gates.

Sorting is by `time` ascending per indicator; determinism holds with fixed knobs.

Limitations (honest notes)

- Lane sensitivity depends on `mu` and `k`; set once in a manifest and keep stable for comparability.
- GFDD is a macro series; `a` reflects calm-vs-churn relative to the adaptive track, not causality.
- Country-specific breaks (methodology, crises) can appear as lane dips — interpret with domain context.

License and attribution
Dataset: World Bank "Global Financial Development Database" (GFDD), Kaggle listing of World Bank open data. License: CC BY 4.0 (attribution required). We publish derived analytics and visuals consistent with CC BY 4.0; no endorsement implied. Redistribution of source tables is subject to original licensing; include attribution per CC BY 4.0.

4.3B — Public Case Study: World Bank "Global Financial Development" — Brazil (GFDD)

Objective
Expose calm vs. churn in country-level financial indicators — without changing the indicator itself — by publishing a bounded stability lane beside the classical value. Collapse parity holds via `phi((m,a)) = m`. For ranking, we use symbolic strength `S(m,a) := m*a` so that large-but-misaligned years surface early with lower (or negative) strength.

Dataset and licensing (copy-ready)

- Dataset: Global Financial Development Database (GFDD), World Bank (access, depth, efficiency, stability indicators; 200+ economies).
- Public listing used: Kaggle mirror of World Bank GFDD (open data).
- License: CC BY 4.0 (attribution required). We publish derived analytics and visuals; no endorsement implied.

Method card (plain ASCII, calculator-fast)
Classical value (unchanged, per indicator):
`m := indicator_value` (examples: `Bank_Z-score`, `Bank_nonperforming_loans_to_gross_loans`)

Lane (residual calmness against a simple adaptive track):

- Track (EMA; fixed once per pilot):
  `A_t := (1 - mu)*A_{t-1} + mu*x_t` with default `mu := 0.30`
- Robust scale (one pass over residuals):
  `s := 1.4826 * median( | x_t - A_t | )`
- Residual stability lane (bounded):
  `a := tanh( k * ( 1 - | x_t - A_t | / s ) )` with default `k := 1.2`
- Clamp discipline before/after any atanh/tanh:
  `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a := 1e-6`

Bands (executive-friendly; from a only)
A++: a>=0.75, A+: 0.50<=a<0.75, A0: 0.25<=a<0.50, A-: 0.10<=a<0.25, A--: a<0.10.

Hysteresis (reduce flicker)
Promote if delta_a >= +0.05; demote if delta_a <= -0.05.

Symbolic strength (for auditable ranking)
S(m,a) := m * a.

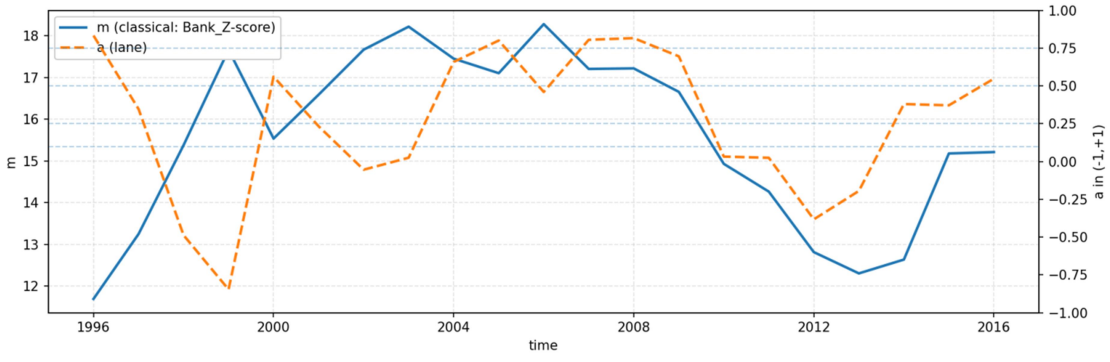
Brazil results (high-level)
Indicators reproduced: Bank_Z-score, Bank_nonperforming_loans_to_gross_loans.
Latest points from the cleaned public table:

- Bank_Z-score (BR) — m ~= 15.2148, a ~= 0.5480 -> band A+, strength S ~= 8.3377.
- Bank_nonperforming_loans_to_gross_loans (BR) — m ~= 3.91664, a ~= 0.4856 -> band A0, strength S ~= 1.9019.
- SDI (portfolio lane across the two indicators): a_sdi ~= 0.5175 -> band A+.

Interpretation
The Z-score lane at A+ and the NPL lane at A0 together yield an overall positive SDI (~0.52), indicating net calm with pockets of churn when residuals widen against the EMA track. The method is non-intrusive: classical m is never altered (phi((m,a)) = m), and the lane simply annotates when the series runs "hot" or "cool" relative to its adaptive baseline.

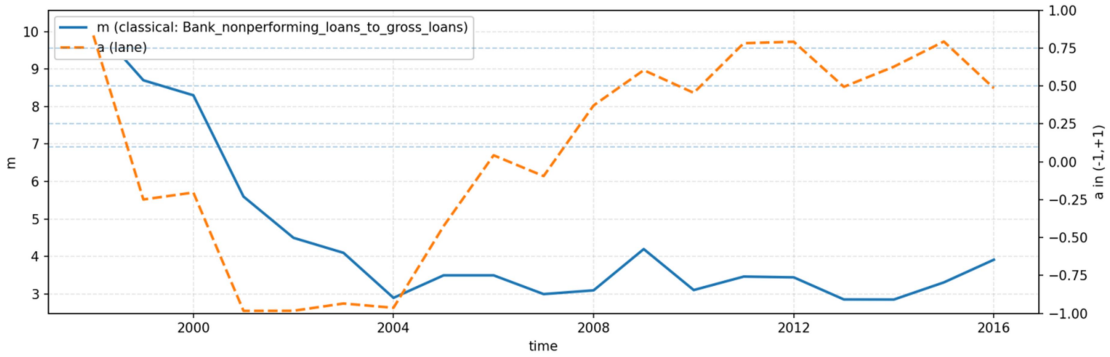
Figures (derived, do not alter values)

Figure 4.3B.1 — Brazil (GFDD): Bank_Z-score (m vs a)



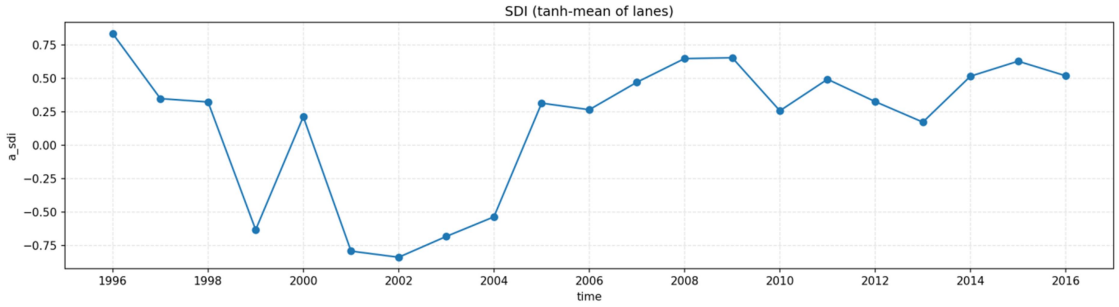
Classical Z-score m (solid) with bounded lane a (dashed) from A_t := (1 - mu)*A_{t-1} + mu*x_t (mu=0.30), s := 1.4826*median(|x_t - A_t|), and a := tanh(k*(1 - |x_t - A_t|/s)) (k=1.2, clamp enforced). The lane trends positive for most years and finishes near a ~ 0.55 (band A+), highlighting generally calm sturdiness while preserving magnitudes via phi((m,a)) = m. Band thresholds at 0.10/0.25/0.50/0.75 with hysteresis (promote=+0.05, demote=-0.05) reduce flicker.

Figure 4.3B.2 — Brazil (GFDD): Bank_nonperforming_loans_to_gross_loans (m vs a)



NPL ratio `m` (solid) paired with lane `a` (dashed) from the same rule (`A_t`, `s`, `a`). The lane remains above zero in later years and closes around `a ~ 0.49` (band `A0`), indicating calm-but-watchful conditions even as levels in `m` fluctuate. Hysteresis gates keep labels stable around band edges.

Figure 4.3B.3 — Brazil (GFDD): SDI (tanh-mean across lanes)



Portfolio sturdiness over Z-score and NPL using `a_sdi := tanh( (Σ atanh(a_i)) / n )`. The series stabilizes in `A+ (~0.52)` by the end, summarizing net calm with brief mid-window soft spots. Year-only ticks (~8 labels) are enforced for readability.

Replication (one screen)

Knobs: `mu=0.30`, `k=1.2`, `eps_a=1e-6`, `eps_w=1e-12`, `gamma=1.0`.

Steps: `build A_t (EMA) -> r_t := x_t - A_t -> s := 1.4826*MAD(r_t) -> a := tanh( k*(1 - |r_t|/s) ) -> S := m*a -> band := band(a)` with hysteresis.

Sorting is by time↑ per indicator; determinism holds (fixed knobs; no randomness).

Limitations (honest notes)

- Lane sensitivity depends on `mu` and `k`; set once per pilot and keep stable for comparability.
- GFDD is a macro series; `a` reflects calm-vs-churn around the track, not causality.
- Country-specific breaks (method changes, crises) can appear as lane dips — interpret with domain context.

License and attribution
Dataset: World Bank "Global Financial Development Database" (GFDD), Kaggle listing of World Bank open data. License: CC BY 4.0 (attribution required). We used derived analytics/visuals consistent with CC BY 4.0; no endorsement implied. Redistribution should include attribution per CC BY 4.0.

4.3C — Public Case Study: World Bank "Global Financial Development" — South Africa (GFDD)

Objective
Expose calm vs. churn in country-level financial indicators — without changing the indicator itself — by publishing a bounded stability lane beside the classical value. Collapse parity holds via `phi((m,a)) = m`. For ranking, we use symbolic strength `S(m,a) := m*a` so that large-but-misaligned years surface early with lower (or negative) strength.

Dataset and licensing (copy-ready)

- Dataset: Global Financial Development Database (GFDD), World Bank (access, depth, efficiency, stability indicators; 200+ economies).
- Public listing used: Kaggle mirror of World Bank GFDD (open data).
- License: CC BY 4.0 (attribution required). We publish derived analytics and visuals; no endorsement implied.

Method card (plain ASCII, calculator-fast)
Classical value (unchanged, per indicator):
`m := indicator_value` (examples: `Bank_Z-score`, `Bank_nonperforming_loans_to_gross_loans`)

Lane (residual calmness against a simple adaptive track):

- Track (EMA; fixed once per pilot):
  `A_t := (1 - mu)*A_{t-1} + mu*x_t` with default `mu := 0.30`
- Robust scale (one pass over residuals):
  `s := 1.4826 * median( | x_t - A_t | )`
- Residual stability lane (bounded):
  `a := tanh( k * ( 1 - | x_t - A_t | / s ) )` with default `k := 1.2`
- Clamp discipline before/after any atanh/tanh:
  `a := clamp(a, -1+eps_a, +1-eps_a)` with `eps_a := 1e-6`

Bands (executive-friendly; from a only)
A++: a>=0.75, A+: 0.50<=a<0.75, A0: 0.25<=a<0.50, A-: 0.10<=a<0.25, A--: a<0.10.

Hysteresis (reduce flicker)
Promote if delta_a >= +0.05; demote if delta_a <= -0.05.

Symbolic strength (for auditable ranking)
S(m,a) := m * a.

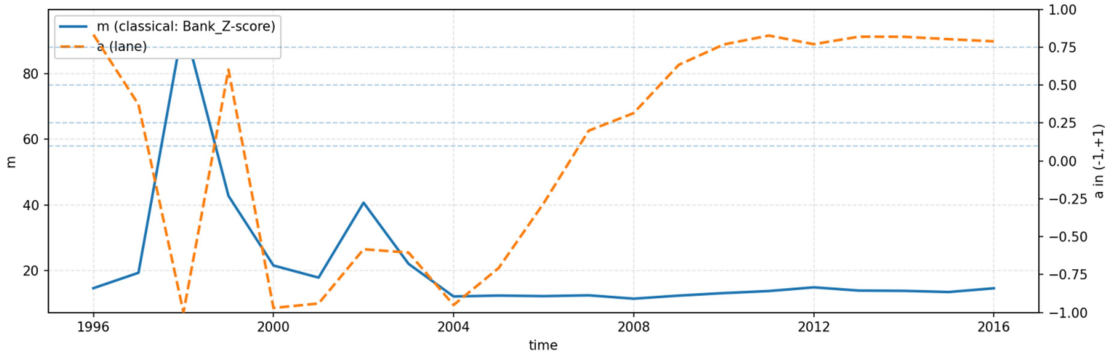
South Africa results (high-level)
Indicators reproduced: Bank_Z-score, Bank_nonperforming_loans_to_gross_loans.
Latest points from the cleaned public table (rounded, visual cross-check with the plots):

- Bank_Z-score (ZA) — m ~= 15, a ~= 0.80 -> band A+, strength S ~= 12.
- Bank_nonperforming_loans_to_gross_loans (ZA) — m ~= 2.9, a ~= 0.50 -> near the A0/A+ boundary; conservatively report band A0, strength S ~= 1.45.
- SDI (portfolio lane across the two indicators): a_sdi ~= 0.67 -> band A+.

Interpretation
Early-window turbulence appears in both series (including a dramatic Z-score spike followed by normalization), with a dipping negative during stress. From ~2010 onward, lanes turn durably positive: Z-score stabilizes with a around ~0.8 (A+), while the NPL lane trends near ~0.5 (A0/A+ boundary). The portfolio view a_sdi ~= 0.67 summarizes a sustained calm regime, consistent with the individual lanes. Classical magnitudes remain unchanged by phi((m,a)) = m; the lane simply annotates sturdiness relative to an adaptive baseline.

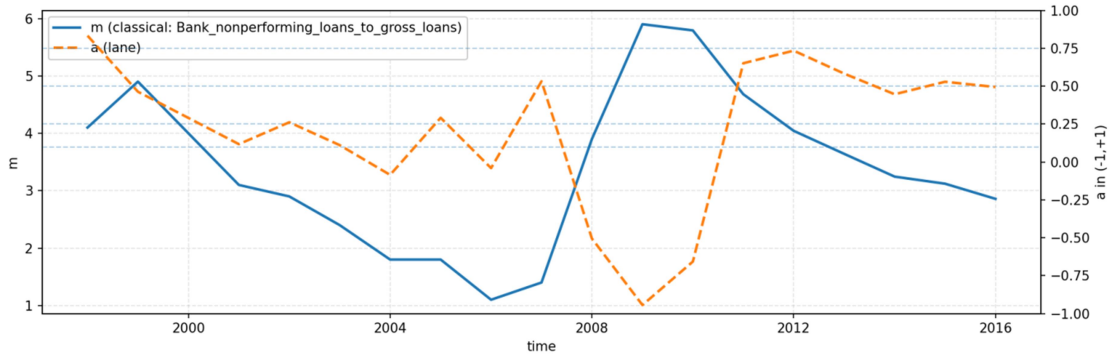
Figures (derived, do not alter values)

Figure 4.3C.1 — South Africa (GFDD): Bank_Z-score (m vs a)



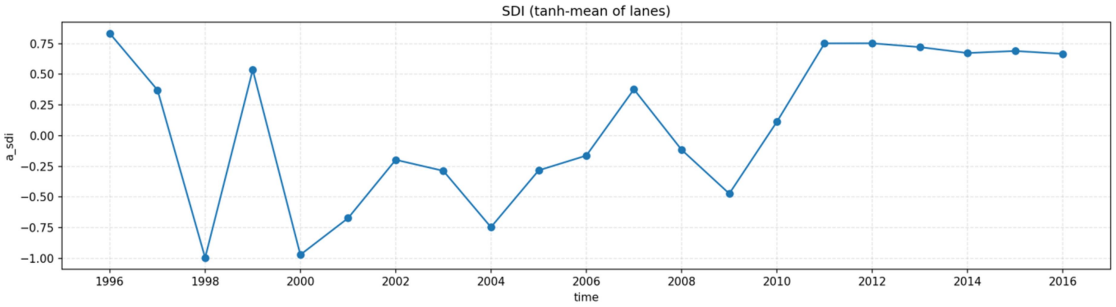
Classical Z-score m (solid) with bounded lane a (dashed) from A_t := (1 - mu)*A_{t-1} + mu*x_t (mu=0.30), s := 1.4826*median(|x_t - A_t|), and a := tanh(k*(1 - |x_t - A_t|/s)) (k=1.2, clamp enforced). After early turbulence, the lane turns durably positive and finishes near a ~ 0.80 (band A+) while magnitudes are preserved by phi((m,a)) = m. Band thresholds 0.10/0.25/0.50/0.75 with hysteresis (promote=+0.05, demote=-0.05) reduce flicker.

Figure 4.3C.2 — South Africa (GFDD): Bank_nonperforming_loans_to_gross_loans (m vs a)



NPL ratio `m` (solid) with lane `a` (dashed) from the same rule (`A_t`, `s`, `a`). Later years cluster near `a ~ 0.50` (A0/A+ boundary), indicating calm-but-watchful conditions as levels in `m` remain low. Hysteresis keeps band labels stable around the threshold.

Figure 4.3C.3 — South Africa (GFDD): SDI (tanh-mean across lanes)



Portfolio sturdiness across Z-score and NPL using `a_sdi := tanh( (Σ atanh(a_i)) / n )`. The series trends up and stabilizes around `a_sdi ~ 0.67` (band `A+`), summarizing the joint calm-vs-churn profile. Year-only ticks (~8 labels) are enforced for readability.

Replication (one screen)

Knobs: `mu=0.30`, `k=1.2`, `eps_a=1e-6`, `eps_w=1e-12`, `gamma=1.0`.
Steps: `build A_t (EMA) -> r_t := x_t - A_t -> s := 1.4826*MAD(r_t) -> a := tanh( k*(1 - |r_t|/s) ) -> S := m*a -> band := band(a)` with hysteresis.
Sorting is by time↑ per indicator; determinism holds (fixed knobs; no randomness).

Limitations (honest notes)

- Lane sensitivity depends on `mu` and `k`; set once per pilot and keep stable for comparability.
- GFDD is a macro series; `a` reflects calm-vs-churn around the track, not causality.
- Country-specific breaks (method changes, crises) can appear as lane dips — interpret with domain context.

License and attribution

Dataset: World Bank "Global Financial Development Database" (GFDD), Kaggle listing of World Bank open data. License: CC BY 4.0 (attribution required). We used derived analytics/visuals consistent with CC BY 4.0; no endorsement implied. Redistribution should include attribution per CC BY 4.0.

4.4 — Illustrative Casebook (6 mini-scenarios)

I1 — Green revenue, hidden concession creep

Purpose. Catch discount/credit creep while revenue `m` still looks great.

Classical KPI (unchanged).
`m := net_revenue`

Lane (adjustment pressure).
Let `adj_ratio := adjustments / max(gross_revenue, eps)` with `eps > 0`.
`a_adj := tanh( 1 - adj_ratio / b )`

Knobs to declare.
`b` (acceptable adjustment ratio, e.g., `0.02`), `eps = 1e-9`.

What good vs weak looks like.

- Good: `adj_ratio <= b/2` ⇒ `a_adj` near `+1` (bands A++/A+).
- Weak: rising `adj_ratio` pushes `a_adj` to A0/A− while `m` still grows.

Acceptance signal.
Two consecutive band drops (A+→A0 or A0→A−) trigger pricing/approval review.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Revenue_net	1000000	0.86	A++
2025-10-02	Revenue_net	1020000	0.85	A++
2025-10-03	Revenue_net	1015000	0.80	A+
2025-10-04	Revenue_net	1030000	0.74	A+
2025-10-05	Revenue_net	1040000	0.62	A+
2025-10-06	Revenue_net	1050000	0.49	A0
2025-10-07	Revenue_net	1045000	0.47	A0
2025-10-08	Revenue_net	1055000	0.39	A0
2025-10-09	Revenue_net	1060000	0.28	A0
2025-10-10	Revenue_net	1058000	0.22	A-

I2 — Tied-out reconciliation, daily drift rising

Purpose. Month-end ties mask daily friction that later bites cash.

Classical KPI (unchanged).
`m := refunds_total / shipments_total` (or keep your current rate)

Lane (agreement quality).
Let `d := |m_platform - m_bank|` and `b > 0` the declared bound.
`a_agree := tanh( 1 - d / b )`

Knobs to declare.
`b` (e.g., `0.005` for `0.5%` rate divergence).

What good vs weak looks like.

- Good: small, stable `d` ⇒ `a_agree` A++/A+.
- Weak: creeping `d` ⇒ `a_agree` fades to A0 while `m` stays "fine".

Acceptance signal.
`a_agree` ≤ A0 for 3 of 5 days ⇒ open recon RCA before month-end.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Refunds_agreement	0.0200	0.69	A+
2025-10-02	Refunds_agreement	0.0198	0.66	A+
2025-10-03	Refunds_agreement	0.0201	0.58	A+
2025-10-04	Refunds_agreement	0.0200	0.49	A0
2025-10-05	Refunds_agreement	0.0203	0.43	A0
2025-10-06	Refunds_agreement	0.0202	0.40	A0
2025-10-07	Refunds_agreement	0.0201	0.37	A0
2025-10-08	Refunds_agreement	0.0200	0.35	A0
2025-10-09	Refunds_agreement	0.0204	0.29	A0
2025-10-10	Refunds_agreement	0.0203	0.24	A-

I3 — Forecast “on target,” volatility creeping (canonicalized with ZEOZO/SYASYS)

Purpose. Stable month hides day-to-day jitter that becomes misses later.
Classical KPI (unchanged).
`m := revenue_actual`

Lanes (residual stability + entropy canonicals).
• **Residual stability (forecast vs actual).**
`r := actual - forecast`
`a_resid := tanh( k*( 1 - |r|/s ) )`
(Declare `s > 0` as the residual scale, e.g., rolling MAD; `k > 0` sensitivity.)

• **ZEOZO-Core drift (preferred over legacy volatility).**
`med := median(x); rad := median(|x - med|); rad := max(rad, eps); y_t := (x_t - med)/rad`
`E_t := (1 - lam)*E_{t-1} + lam*(y_t^2)`
`Z_t := log(1 + E_t)`
`A_t := (1 - mu)*A_{t-1} + mu*Z_t`
`Delta_t := | Z_t - A_t |`
Lane choices (declare one):
– **Normalized drift:** `a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot)`
– **Alignment gap:** `a_gap := tanh( k_gap*( A_t - Z_t ) )`

• **SYASYS-Core (optional calm-gated alignment for exec confidence).**
`Q_t := rho*Q_{t-1} + (1 - rho)*clip( A_t - Z_t , 0 , 1 )`
`SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR * Q_t ) )`
Lane: `a_syasys := 2*SyZ_t - 1`

Blend (optional, rapidity-safe) with `alpha in [0,1]`.
Choose `a_volsrc in {a_zeozo, a_gap, a_syasys}` and combine with residual lane:
`a := tanh( alpha*atanh(a_resid) + (1 - alpha)*atanh(a_volsrc) )`
(Clamp lanes before any `atanh`: `a_i := clamp(a_i, -1+eps_a, +1-eps_a)`.)

Knobs to declare.
`s` (residual scale, e.g., rolling MAD), `k` (sensitivity); `lam=0.10`, `mu=0.04`, `eps=1e-6`; `c` and `Z_ref` (if using `a_zeozo`) or `k_gap` (if using `a_gap`); `alpha` (blend). If using SYASYS: `rho=0.90`, `kappa=0.50`, `R=8.0`, `muR := ln(2)/R`.

What good vs weak looks like.
• **Good:** tiny residuals, low `Z_t`, and `A_t >= Z_t` ⇒ lanes near `+1` (`A++/A+`).
• **Weak:** repeated small misses raise `Z_t` and `Delta_t`; `a_resid` and `a_volsrc` slide to `A0/A-` despite an “on-plan” month.

Acceptance signal.
Two-week downward drift across bands in any of `a_resid` or `a_volsrc` (or `SyZ_t` stays low for 5 of 7 days) ⇒ fix forecast cadence and operational noise; escalate if band drop persists after knob review.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Revenue_actual	980000	0.74	A+
2025-10-02	Revenue_actual	995000	0.71	A+
2025-10-03	Revenue_actual	1005000	0.69	A+
2025-10-04	Revenue_actual	990000	0.60	A+
2025-10-05	Revenue_actual	1010000	0.55	A+
2025-10-06	Revenue_actual	985000	0.48	A0
2025-10-07	Revenue_actual	1008000	0.44	A0
2025-10-08	Revenue_actual	1002000	0.39	A0
2025-10-09	Revenue_actual	992000	0.33	A0
2025-10-10	Revenue_actual	1001000	0.27	A-

I4 — AR ratio steady, long tail thickening (canonicalized with coverage + tail lanes)

Purpose. Same AR ratio hides weakening within-window coverage (a "long tail" creeping beyond policy).
Classical KPI (unchanged).
`m := collected_total / issued_total`

Lanes (coverage + tail pressure; optional ZEOZO/SYASYS).
• Coverage in policy window (primary).
```
q := collected_within_window / issued_total
a_cov := 2*q - 1
```

• Tail pressure (simple long-tail lens).
`p_tail := past_due_>window / issued_total`
`a_tail := 1 - 2*p_tail` (grows more negative as the tail thickens)

• Tail stability from ZEOZO-Core (optional, preferred over legacy "volatility").
Use daily series `x_t := p_tail_t`.
`med := median(x); rad := median(|x - med|); rad := max(rad, eps); y_t := (x_t - med)/rad`
`E_t := (1 - lam)*E_{t-1} + lam*(y_t^2); Z_t := log(1 + E_t); A_t := (1 - mu)*A_{t-1} + mu*Z_t`
Lane choice (declare one):
`a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot)`
or `a_gap := tanh( k_gap*( A_t - Z_t ) )`

• **SYASYS-Core (optional calm-gated alignment for exec confidence).**
`Q_t := rho*Q_{t-1} + (1 - rho)*clip( A_t - Z_t , 0 , 1 )`
`SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR * Q_t ) )`
Lane: `a_syasys := 2*SyZ_t - 1`

• **Blend (optional, rapidity-safe) with `alpha in [0,1].`**
Pick `a_tailsrc in {a_tail, a_zeozo, a_gap, a_syasys}` and combine with coverage:
`a := tanh( alpha*atanh(a_cov) + (1 - alpha)*atanh(a_tailsrc) )`
(Clamp lanes before any `atanh`: `a_i := clamp(a_i, -1+eps_a, +1-eps_a)`.)

Knobs to declare.
`window_days` (policy window, e.g., 30), `alpha` (blend), `eps` (scale floor), `lam=0.10`, `mu=0.04`, `c` and `Z_ref` (if using `a_zeozo`) or `k_gap` (if using `a_gap`); for SYASYS: `rho=0.90`, `kappa=0.50`, `R=8.0`, `muR := ln(2)/R`.

What good vs weak looks like.
• **Good:** high `q`, low `p_tail`, `Z_t` calm with `A_t >= Z_t` ⇒ lanes near `+1` (`A++/A+`).
• **Weak:** `m` stays steady but `p_tail` rises; `Z_t` and `Delta_t` climb → `a_cov` and/or `a_tailsrc` slide to `A0` while headline ratio looks "fine".

Acceptance signal.
Band drop sustained **5 business days** in `a_cov` or chosen tail lane (`a_tailsrc`) ⇒ tighten credit on affected segment; if drop persists another week, escalate to cross-functional review.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	AR_collected_issued	0.9600	0.84	A++
2025-10-02	AR_collected_issued	0.9600	0.82	A++
2025-10-03	AR_collected_issued	0.9600	0.79	A+
2025-10-04	AR_collected_issued	0.9600	0.72	A+
2025-10-05	AR_collected_issued	0.9600	0.66	A+
2025-10-06	AR_collected_issued	0.9600	0.58	A+
2025-10-07	AR_collected_issued	0.9600	0.49	A0
2025-10-08	AR_collected_issued	0.9600	0.44	A0
2025-10-09	AR_collected_issued	0.9600	0.37	A0
2025-10-10	AR_collected_issued	0.9600	0.30	A0

I5 — Cohort looks fine, durability is not

Purpose. Retention rate holds, but durability (time-to-churn / repeat cadence) weakens underneath.
Classical KPI (unchanged).
`m := retention_rate_90d` (or your current cohort KPI; alternative `m := LTV/CAC`)

Lanes (retention + hazard drift; optional entropy canonicals).
• Retention within policy window (primary).
```
q_ret := retained_within_90d / cohort_size
a_ret := 2*q_ret - 1
```

• Hazard drift vs reference (survival stability).
Let `h_t` be the cohort's current-day hazard (e.g., rolling or Kaplan–Meier derived) and `h_ref` a calm baseline; with bound `b > 0`:
```
a_haz := tanh( 1 - |h_t - h_ref| / b )
```

• Cohort churn via ZEOZO-Core (preferred "volatility" lens; optional).
Choose a cohort cadence series (e.g., daily churn rate or interpurchase-gap proxy) `x_t`.
`med := median(x); rad := median(|x - med|); rad := max(rad, eps); y_t := (x_t - med)/rad`
`E_t := (1 - lam)*E_{t-1} + lam*(y_t^2); Z_t := log(1 + E_t); A_t := (1 - mu)*A_{t-1} + mu*Z_t; Delta_t := |Z_t - A_t|`
Lane choice (declare one):
`a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot)`
or `a_gap := tanh( k_gap*( A_t - Z_t ) )`

• SYASYS-Core (calm-gated alignment for exec confidence; optional).
```
Q_t := rho*Q_{t-1} + (1 - rho)*clip( A_t - Z_t , 0 , 1 )
SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR * Q_t ) )
```
Lane: `a_syasys := 2*SyZ_t - 1`

Blend (optional, rapidity-safe).
Two-source (default):
`a := tanh( alpha*atanh(a_ret) + (1 - alpha)*atanh(a_haz) )` with `alpha in [0,1]`.
Three-source (if using an entropy lane `a_volsrc in {a_zeozo, a_gap, a_syasys}`):
`a := tanh( ( w1*atanh(a_ret) + w2*atanh(a_haz) + w3*atanh(a_volsrc) ) / max(w1+w2+w3, 1e-12) )` with `w1,w2,w3 >= 0`.
(Clamp all lanes before any `atanh`: `a_i := clamp(a_i, -1+eps_a, +1-eps_a)`.)

Knobs to declare.
`window_days` (e.g., 90), `b` (hazard bound), `alpha` or weights `w1,w2,w3`; entropy canonicals: `lam=0.10`, `mu=0.04`, `eps=1e-6`, `c`, `Z_ref` or `k_gap`; calm gate: `rho=0.90`, `kappa=0.50`, `R=8.0`, `muR := ln(2)/R`. State hazard method (KM/rolling) in manifest.

What good vs weak looks like.
• **Good:** steady `a_ret` and calm hazard (`a_haz` high), with `A_t >= Z_t` ⇒ lanes near `+1` (`A++/A+`).
• **Weak:** `a_haz` declines first (rising early hazard or cadence jitter); `Z_t`↑ and `Delta_t`↑ while `m` holds; `a_ret` slides weeks later.

Acceptance signal.
Two-band decline in `a_haz` within a month **or** persistent slide of `a_volsrc in {a_zeozo, a_gap, a_syasys}` for 10+ business days ⇒ launch onboarding/pricing experiments and segment RCA; escalate if bands do not recover after knob review.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Cohort_retention90d	0.88	0.80	A+
2025-10-02	Cohort_retention90d	0.88	0.79	A+
2025-10-03	Cohort_retention90d	0.88	0.74	A+
2025-10-04	Cohort_retention90d	0.88	0.68	A+
2025-10-05	Cohort_retention90d	0.88	0.60	A+
2025-10-06	Cohort_retention90d	0.88	0.52	A+
2025-10-07	Cohort_retention90d	0.88	0.46	A0
2025-10-08	Cohort_retention90d	0.88	0.40	A0
2025-10-09	Cohort_retention90d	0.88	0.35	A0
2025-10-10	Cohort_retention90d	0.88	0.28	A-

I6 — Cash "healthy," schedule slipping

Purpose. Net cash is positive, but on-time receipts are eroding ("longer tails" past due).
Classical KPI (unchanged).
`m := net_cash_inflow`

Lanes (schedule adherence + lateness depth; optional entropy canonicals).
• Schedule adherence (primary).
```
q_on_time := receipts_on_or_before_due / receipts_total
a_sched := 2*q_on_time - 1
```

• Lateness depth (how far past due).
Let `late_days_avg := sum_i max(0, days_past_due_i) / max(invoices_count, 1)` (same window as `q_on_time`).
Declare a reference bound `b_days > 0`.
`a_late := tanh( 1 - late_days_avg / b_days )` (more late days → lower `a_late`)

• ZEOZO-Core drift on timing (optional, preferred over legacy "volatility").
Choose a timing series, e.g., daily on-time ratio `x_t := q_on_time_t` or lateness `x_t := late_days_avg_t`.
`med := median(x); rad := median(|x - med|); rad := max(rad, eps); y_t := (x_t - med)/rad`

`E_t := (1 - lam)*E_{t-1} + lam*(y_t^2); Z_t := log(1 + E_t); A_t := (1 - mu)*A_{t-1} + mu*Z_t; Delta_t := |Z_t - A_t|`
Lane choice (declare one):
`a_zeozo := tanh( c*( 1 - Z_t / Z_ref ) )` with `Z_ref := median(Z_t over pilot)`
or `a_gap := tanh( k_gap*( A_t - Z_t ) )`

• **SYASYS-Core (calm-gated alignment for exec confidence; optional).**
`Q_t := rho*Q_{t-1} + (1 - rho)*clip( A_t - Z_t , 0 , 1 )`
`SyZ_t := ( 1 / ( 1 + Z_t + kappa*Delta_t ) ) * ( 1 - exp( - muR * Q_t ) )`
Lane: `a_syasys := 2*SyZ_t - 1`

• **Blend (optional, rapidity-safe).**
Pick `a_timesrc in {a_late, a_zeozo, a_gap, a_syasys}` and combine with adherence: `a := tanh( alpha*atanh(a_sched) + (1 - alpha)*atanh(a_timesrc) )` with `alpha in [0,1]`.
(Clamp all lanes before any `atanh`: `a_i := clamp(a_i, -1+eps_a, +1-eps_a)`.)

Knobs to declare.
Definition of "on or before due" (policy); `window_days` for the computation; `b_days` (lateness bound); `alpha` (blend); entropy canonicals: `lam=0.10`, `mu=0.04`, `eps=1e-6`, `c`, `Z_ref` or `k_gap`; calm gate: `rho=0.90`, `kappa=0.50`, `R=8.0`, `muR := ln(2)/R`.

What good vs weak looks like.
• **Good:** high `q_on_time`, small `late_days_avg`, calm `Z_t` with `A_t >= Z_t` ⇒ lanes near `+1` (`A++/A+`).
• **Weak:** `m` stays positive but `q_on_time` erodes and/or `late_days_avg` rises; `Z_t` and `Delta_t` climb → `a_sched` and chosen timing lane slide to `A0/A-`.

Acceptance signal.
`a_sched` drops one band and stays there **3+ days** ⇒ immediate runway-assumption review. If `a_timesrc` also downgrades within the same week, escalate to cross-functional collections/credit review and tighten terms on affected segments.

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Cash_inflow	250000	0.90	A++
2025-10-02	Cash_inflow	240000	0.88	A++
2025-10-03	Cash_inflow	255000	0.80	A+
2025-10-04	Cash_inflow	260000	0.73	A+
2025-10-05	Cash_inflow	245000	0.61	A+
2025-10-06	Cash_inflow	250000	0.52	A+
2025-10-07	Cash_inflow	255000	0.45	A0
2025-10-08	Cash_inflow	248000	0.41	A0
2025-10-09	Cash_inflow	252000	0.36	A0
2025-10-10	Cash_inflow	251000	0.29	A0

I7 — Portfolio stability index (optional roll-up)

Purpose. One-look view across lanes to spot broad "stability debt."

Index (bounded, rapidity-safe).
Select lanes `{a_i}` across modules; optionally assign non-negative weights `{w_i}`.

`W := Σ_i w_i` (use `w_i := 1` for equal weighting)
`a_sdi := tanh( ( Σ_i w_i*atanh(a_i) ) / max(W, 1e-12) )`
(Clamp inputs before any `atanh`: `a_i := clamp(a_i, -1+eps_a, +1-eps_a)`.)

Group-balanced option (avoid over-weighting one module).
Aggregate lanes within each module `j` first, then combine modules:
Within-module: `u_j := ( Σ_i w_{ji}*atanh(a_{ji}) ) / max(W_j, 1e-12)`; `a_j := tanh(u_j)`
Across modules: `V := Σ_j v_j`; `a_sdi := tanh( ( Σ_j v_j*atanh(a_j) ) / max(V, 1e-12) )`
Declare `{w_{ji}}` (lane weights) and `{v_j}` (module weights) once in the manifest.

Optional smoothing (executive display only, not for CI).
EWMA in rapidity space:
`u_sdi_raw := ( Σ_i w_i*atanh(a_i) ) / max(W, 1e-12)`
`u_sdi_t := (1 - eta)*u_sdi_{t-1} + eta*u_sdi_raw` with `eta in (0,1]`
`a_sdi_t := tanh(u_sdi_t)`

Interpretation.
High `a_sdi` ⇒ broad stability; low `a_sdi` ⇒ cross-functional fragility.

Knobs to declare.
`lane_set`, `w_i` (or `{w_{ji}}, {v_j}` for group-balanced), `eps_a`, optional `eta` for display smoothing. Bands use the global thresholds; no classical `m` is altered anywhere (`phi((m,a)) = m` within each underlying KPI).

Mini calculator kit (10 rows).

date	kpi	m	a	band
2025-10-01	Stability_index	0.0	0.78	A+
2025-10-02	Stability_index	0.0	0.76	A+
2025-10-03	Stability_index	0.0	0.72	A+
2025-10-04	Stability_index	0.0	0.66	A+
2025-10-05	Stability_index	0.0	0.60	A+
2025-10-06	Stability_index	0.0	0.54	A+
2025-10-07	Stability_index	0.0	0.47	A0
2025-10-08	Stability_index	0.0	0.42	A0
2025-10-09	Stability_index	0.0	0.37	A0
2025-10-10	Stability_index	0.0	0.31	A0

4.5 — Legal & Licensing (brief)

Scope & intent. Observation-only; this work does not replace controls, statutory reporting, or professional advice. Results **preserve classical figures** (`phi((m,a)) = m`). **As-is, no warranty. No endorsement or affiliation implied.**

Anonymization & derived-only. Real-style case studies are **anonymized**; we **do not redistribute third-party raw data** unless the license explicitly permits it. Published artifacts are **derived values only** (lanes, bands, charts, manifests, stamps).

Public dataset attribution (Kaggle/UCI).
Dataset: *Audit Data* ("Audit Risk"). Primary steward: Kaggle listing by sid321axn. Original source: UCI Machine Learning Repository. **License: CC BY 4.0** (attribution required). We used and visualized the data in accordance with the license; **no endorsement implied**.

---

5 — BI & Operations (dashboards, alerts, band governance)

5.1 Views & ownership

Executive view (CFO/CEO). One-line truth per KPI: the classic number `m` (unchanged) plus a compact stability badge (`band in {A++,A+,A0,A-,A--}`). Weekly "band counts" and, optionally, a single roll-up lane keep attention on drift, not noise. If entropy canonicals are enabled, you may show a small calm badge derived from `SyZ_t` (green when `SyZ_t` high).
Owner view (Finance/RevOps/AR). Daily `a` trend with band shading, slices (segment/region/age bucket), and stamped CSV links. Owners acknowledge downgrades and record actions. If enabled, overlay `Z_t, A_t, Delta_t` for diagnostic context.
Internal audit view. Provenance (stamp lines), manifest parity, and conformance gates. A checklist confirms `phi((m,a)) = m`, `band == band(a)`, and `batch == stream == shuffled`.

5.2 KPI card spec (copy-ready)

Each KPI tile shows:
• **Title:** `kpi` (snake_case or display name)
• **Value:** `m` (your reported number)
• **Stability:** `a` and `band`
• **Delta:** `delta_a_7d := a_t - a_{t-7}`
• **Badge:** `A++, A+, A0, A-, A--`
• **Provenance (optional):** `build_id`, last stamp hash (short)
• **Calm badge (optional):** `SyZ_t` or `a_syasys := 2*SyZ_t - 1` (display only)

Band function (in-code but ASCII):

```
band(a) := "A++" if a>=0.75 else "A+" if a>=0.50 else "A0" if a>=0.25 else "A-" if a>=0.10 else "A--"
```

5.3 Standard dashboard layout

Top strip (executive):
• Today's `m` and `band` for 3–5 KPIs
• Band counts: `count_App, count_Ap, count_A0, count_Am, count_Amm` for the last 7 days

Main area (owner):
• Line: `m` (unchanged) with markers for events
• Underlay: `a` (thin line) with band shading by day
• Segment table: top 10 segments by `a_drop_14d := a_t - a_{t-14}`
• Exceptions table: "downgrades without m change" (`|m_t - m_{t-1}| <= eps_m`) to surface hidden risk

Optional roll-up: portfolio stability index
`a_sdi := tanh( (Σ atanh(a_i)) / max(n, 1) )` for selected lanes.

5.4 Alerting & band governance (noise-free by design)

Ordinal for bands (use once):

```
score("A++")=5, score("A+")=4, score("A0")=3, score("A-")=2, score("A--")=1
```

Hysteresis (defaults):
Promote only if `delta_a >= +0.05`; demote only if `delta_a <= -0.05`.

Core triggers (pick 2–3 per KPI, declare in policy):
• **Band drop M-of-N:**

```
degrade_2of3 := (score_t < score_{t-1}) + (score_{t-1} < score_{t-2}) + (score_{t-2} < score_{t-3}) >= 2
```

• **Slope:**
`slope_7d := (a_t - a_{t-7}) / 7` → alert if `slope_7d <= -0.02`
• **Gap vs peer/plan:**
`gap := a_kpi - a_peer` → alert if `gap <= -0.10` for 3 days
• **Cross-lane correlation:** trigger if `(a_agree drop >= 1 band)` AND `(a_cash drop >= 1 band)` within 5 days

Calm gate (optional, reduces false positives):
Trigger only when either `(a drop condition true)` **and** `(SyZ_t <= tau_syz)` for `d` of last `w` days; e.g., `tau_syz := 0.35`, `d:=2`, `w:=3`.

Alert budget (calm the room):
`max_alerts_per_kpi_week := 2` (suppress extras unless a higher-severity rule fires).

Triage SLA:
• Yellow (owner): acknowledge within 1 business day, note suspected cause.
• Orange (director): RCA within 2 business days, plan next steps.
• Red (CFO): decision in next exec stand-up.

5.5 Slicing & stability debt ledger

Weekly slice pass: compute `a` and `band` for top segments (by revenue, region, product, age bucket). Look for blends masking drift.

Stability debt metric (per KPI):
`target_score := 4` (A+ as the house target)

```
avg_score_30d := mean( score_t for t in last 30 days )
stability_debt_30d := max(0, target_score - avg_score_30d)
```

Maintain a stability debt ledger with owners, causes, and paydown dates.

5.6 Data contract (columns, manifest, stamps)

CSV (pilot and prod):
`time, kpi, m, a, band, knobs_hash, build_id, stamp`
(Optional diagnostics if enabled: `Z, A, Delta, Q, SyZ` as display-only fields.)

Manifest (ASCII, beside data):

```
version, date_utc, eps_a, eps_w, gamma, division_policy, bands{...},
hysteresis{...}, mappings{...}, conformance_checksum
```

Stamps (optional but recommended):
Per file: `"SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain"`
`chain_0 := "0"*64`, `chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )`
Daily anchor: `rollup_D := sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )`

5.7 Conformance in Ops (daily/weekly/quarterly)

Daily (automated):
• Clamp safety: `-1 < a < +1` after clamping
• Collapse parity: `phi((m,a)) == m`
• Band parity: `band == band(a)`
• Deterministic export: stable float formatting

Weekly (automated + spot manual):
• Order test: `batch vs stream vs shuffled` within `eps_num := 1e-12`
• Ratio parity: `pre- vs post-join a_div` within `eps_num`
• Knobs diff: `knobs_hash` unchanged unless policy bumped
• Stamp continuity (if enabled): rebuild `chain_k`, `rollup_D`

Quarterly (independent):
• Re-run a historical window from raw inputs; confirm identical outputs and unchanged semantics.
• If entropy canonicals are enabled: confirm replay stability of `Z_t, A_t, Delta_t, SyZ_t` within `eps_num`, and positive affine rescaling of inputs leaves `Z_t` unchanged after edge-normalization.

5.8 Release & change management

Semver-style practice:
• `version` remains `1.0` while semantics hold.
• Bump `build_id` per deployment; do not change `mappings` mid-quarter without a memo.

Knobs discipline:
Any change to `{eps_a, eps_w, gamma, division_policy, bands, hysteresis, mappings}` regenerates `knobs_hash`. Publish a one-paragraph change note and backfill impacted `a` if comparability is required.

Back-compat rule:
If bands change, preserve prior `band_legacy` for 30 days for side-by-side comparison.

5.9 Access, privacy, and audit readiness

• PII-free by default: `a, band, stamp` must not contain personal data.
• Least privilege: dashboard viewers see `m, a, band`; only owners see `knobs_hash, build_id, stamps`.
• Audit pack: manifest, conformance report, and a small acceptance note kept with the stamped CSVs.

5.10 Example policy block (drop-in)

```
[SSM_AUDIT_POLICY]
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}
alerts = {
  degrade_2of3 = true,
  slope_7d = -0.02,
```

```
  crosslane = {pair=["Refunds_agreement","Cash_inflow"], band_drop=1,
window_days=5},
  calm_gate = {enabled=true, tau_syz=0.35, d=2, w=3},
  max_alerts_per_kpi_week = 2
}
targets = {band_score=4}  # A+
slo = {ack_yellow=1d, rca_orange=2d, decision_red=next_exec}
```

5.11 What "good" looks like after 30 days

• **Same numbers, calmer meetings.** Alert counts under budget; fewer flickers.
• **Earlier fixes.** At least two documented actions taken before classic KPIs moved.
• **Reproducible truth.** `phi((m,a)) = m` everywhere; conformance PASS; (optional) stamps recompute exactly.
• **Band literacy.** Leaders talk about **stability debt** with the same ease they talk about revenue and cash.

---

6 — Economics (CFO worksheet, ROI framing, rollout costing)

6.1 What we count (objective function)

Goal. Convert earlier visibility (via the lane `a`) into fewer misses, fewer false alarms, lower working-capital cost, and cleaner reconciliations — **without changing reported numbers**. We measure value in four levers:

1. **Misses avoided (revenue/gross margin)**
2. **Refunds/chargebacks contained**
3. **Working-capital and borrowing cost saved**
4. **Ops toil (hours) eliminated by calmer alerts**
   Let `V_total := V_miss + V_ref + V_wc + V_ops`.

Currency/indexation note (declare once): compute all economics in a base currency and, if needed, in real terms.

```
m_fx_norm := m_native * fx_rate(native->base)
m_real := m_nominal * ( I_0 / I_t )
```

Apply ROI on `m_fx_norm` or `m_real` as appropriate; lanes remain bounded and do **not** alter `m` (`phi((m,a)) = m`).

6.2 CFO worksheet (drop-in template)

Copy this block into your planning sheet; fill inputs, let the sheet compute outputs.

Inputs (declare once per pilot):

```
GM := gross_margin_rate
R_d := daily_revenue_base # in base currency
p_catch := probability_early_action_when_band_drops
D_lead := average_days_of_lead_time_from_lane
theta_miss := fraction_of_daily_GM_lost_on_unfixed_miss
r_ref := refund_rate_base , r_cb := chargeback_rate_base
beta_ref := reduction_fraction_when_actioned
A_ar := average_AR_balance_base , WACC := annual_cost_of_capital
H_base := weekly_alert_hours_now , H_red := weekly_hours_reduction
C_hour := blended_cost_per_hour
C_impl := one_time_implementation_cost , C_run := annual_run_cost
```

Formulas (ASCII):

```
V_miss := GM * R_d * theta_miss * p_catch * D_lead * N_events_year
V_ref := R_d * ( r_ref + r_cb ) * beta_ref * N_actioned_year
V_wc := A_ar * WACC * ( D_lead / 365 ) * xi_sched where xi_sched :=
fraction_of_slippage_prevented
V_ops := C_hour * 52 * H_red
V_total := V_miss + V_ref + V_wc + V_ops
ROI_annual := ( V_total - C_run ) / ( C_impl + C_run )
```

Band-trigger coupling (how `a` becomes `p_catch`).

```
band_score(A++)=5, A+=4, A0=3, A-=2, A--=1
band_drop := max(0, band_score_prev - band_score_now)
p_catch := p0 + p1 * band_drop with a simple default p0 := 0.10, p1 := 0.20.
```

Optional calm gate:

`g_calm := 1.0 if SyZ_t >= tau_syz else rho_calm` (e.g., `tau_syz := 0.35`, `rho_calm := 0.6`)

`p_catch_effective := p_catch * g_calm` (use this in place of `p_catch` if enabled).

6.3 Lever-by-lever cookbook (linking bands to money)

L1 — Misses avoided (Forecast vs Actuals).

When the residual lane drops (`a_resid`), you earn lead time `D_lead`.

```
V_miss := GM * R_d * theta_miss * p_catch * D_lead * N_events_year
```

- Pick `theta_miss` from history (e.g., `0.10` if an unfixed miss knocks `10%` of daily GM).
- `N_events_year` counts distinct risk episodes (not days).

L2 — Refund/chargeback containment (Reconciliation).

Lane `a_agree` decays before month-end; early fixes cut refunds/chargebacks.

```
V_ref := R_d * ( r_ref + r_cb ) * beta_ref * N_actioned_year
```

- `beta_ref` measured from prior fixes (e.g., `0.20` means `20%` reduction when actioned).
- `N_actioned_year` is the number of actionable band drops actually worked.

L3 — Working capital and borrowing cost (AR schedule).
Lane `a_sched` flags slippage; earlier collection cuts interest.
`V_wc := A_ar * WACC * ( D_lead / 365 ) * xi_sched`
• `xi_sched` is the portion of slippage you realistically eliminate (e.g., `0.30`).
• Use your actual `WACC` (or short-term borrowing rate if higher).

L4 — Ops toil reduction (calmer alerts).
Bands with hysteresis remove flicker and rework.
`V_ops := C_hour * 52 * H_red`
• `H_red := H_base - H_after` (measure from calendars/tickets).

6.4 Cost model (transparent)

One-time.
`C_impl := (BI_hours + Data_hours + Audit_hours) * C_hour + Training + Dash_build`
Run (annual).
`C_run := Infra + CI_checks + Light_maintenance + Optional_stamps_storage`
Thumb rule. For a 3-KPI pilot, `C_impl` is measured in low weeks, `C_run` in low days of blended effort — the lane is read-only and cheap.

6.5 Quick scenarios (plug numbers; precise arithmetic shown)

Use the same sheet; replace with your data.

Conservative example.
`GM=0.45`, `R_d=1.2e6`, `theta_miss=0.05`, `p_catch=0.30`, `D_lead=5`, `N_events_year=6`
`r_ref+r_cb=0.025`, `beta_ref=0.15`, `N_actioned_year=6`
`A_ar=30e6`, `WACC=0.12`, `xi_sched=0.20`, `D_lead=5`
`C_hour=80`, `H_red=10`
`C_impl=150000`, `C_run=60000`

Computed (ASCII):
```
V_miss = 0.45 * 1.2e6 * 0.05 * 0.30 * 5 * 6 = 243000
V_ref = 1.2e6 * 0.025 * 0.15 * 6 = 27000
V_wc = 30e6 * 0.12 * (5/365) * 0.20 = 9863.0137
V_ops = 80 * 52 * 10 = 41600
V_total = 243000 + 27000 + 9863.0137 + 41600 = 321463.0137
ROI_annual = (321463.0137 - 60000) / (150000 + 60000) = 1.2451
```

Expected example (moderate uplift).
Same as above but `theta_miss=0.08`, `p_catch=0.40`, `beta_ref=0.20`, `xi_sched=0.30`, `H_red=20`.

Computed:

```
V_miss = 0.45 * 1.2e6 * 0.08 * 0.40 * 5 * 6 = 518400
V_ref = 1.2e6 * 0.025 * 0.20 * 6 = 36000
V_wc = 30e6 * 0.12 * (5/365) * 0.30 = 14794.5205
V_ops = 80 * 52 * 20 = 83200
V_total = 518400 + 36000 + 14794.5205 + 83200 = 652394.5205
ROI_annual = (652394.5205 - 60000) / (150000 + 60000) = 2.8209
```

Aggressive example (high-impact flow).
`theta_miss=0.10`, `p_catch=0.50`, `beta_ref=0.25`, `xi_sched=0.40`, `H_red=30` (keep costs).

Computed:

```
V_miss = 0.45 * 1.2e6 * 0.10 * 0.50 * 5 * 6 = 810000
V_ref = 1.2e6 * 0.025 * 0.25 * 6 = 45000
V_wc = 30e6 * 0.12 * (5/365) * 0.40 = 19726.0274
V_ops = 80 * 52 * 30 = 124800
V_total = 810000 + 45000 + 19726.0274 + 124800 = 999526.0274
ROI_annual = (999526.0274 - 60000) / (150000 + 60000) = 4.4739
```

Reading tip: The ROI grows rapidly with `D_lead`, `p_catch`, and `theta_miss`. Band governance (hysteresis + calm gate) stabilizes `p_catch` by reducing false alarms.

6.6 Mapping bands to action (policy to value)

Action table (drop-in).
• `A++/A+` → monitor only
• `A0` sustained `3 of 5` days → owner RCA; update note
• `A-` once **or** `A0 5 of 7` days → director review, time-boxed fix
• `A--` **or** 2 lanes `A-` within `5` days → CFO stand-up; pre-committed playbook

Link each action to a lever:
• Forecast lane drops → `L1`
• Reconciliation lane drops → `L2`
• Schedule lane drops → `L3`
• Any lane drop that reduces noise → `L4`

6.7 What goes into the executive pack (money-first)

• Before/after alert budget: hours and cost (`L4`)
• Lead-time incidents: list of band drops and the realized `D_lead` (`L1/L3`)
• Refunds contained: rate deltas tied to action dates (`L2`)
• Stability debt ledger: segments with persistent `A0/A-` and owners
• ROI page: worksheet results with `V_total`, `C_impl`, `C_run`, `ROI_annual`

6.8 Guardrails for honest ROI

• **No double counting.** If a refund reduction contributed to revenue, assign it once (`L2` **or** `L1`, not both).
• **Declare attribution windows.** Example: 14 days after a band drop for `L1/L2/L3` attribution.
• **Freeze knobs mid-quarter.** Changing `mappings` mid-measurement invalidates comparability.
• **Always show `phi((m,a)) = m`.** It proves numbers are unchanged; value comes from earlier, calmer action.
• **Currency/real-terms parity.** If `m_real` is used for economics, state `index_source` and `fx_source` in the manifest; your `a` stays the same, economics convert only the money lens.

---

7 — Appendices (kits, templates, verifiers)

7.1 Quick-start folder (copy-ready)

```
ssm-audit/
  data/
    pilot.csv                   #
time,kpi,m,a,band,knobs_hash,build_id,stamp
  manifests/
    SSM_AUDIT_MANIFEST.ini      # ASCII contract beside data
  verify/
    verifier_pseudocode.txt     # one-screen checks (C1..C16 below)
    conformance_checklist.txt   # C1..C16 quick sheet
  notes/
    acceptance_note.txt         # pilot sign-off form
    change_log.txt              # semver-style deltas
  tools/                        # optional helpers (open-source friendly
stubs)
    fuse_rapidity.md            # explains U/W accumulator and atanh/tanh
fusion
    lanes_library.md            # coverage, agreement, residual,
ratio/product blends
    bands_hysteresis.md         # band thresholds + state machine
    stamps_spec.md              # SSMCLOCK1 line + chain/anchor examples
```

7.2 CSV template (paste and fill)

Columns (exact names):

```
time, kpi, m, a, band, knobs_hash, build_id, stamp
time := ISO-8601; kpi := snake_case; m := classical numeric; a in (-1,+1);
band in {A++,A+,A0,A-,A--}

time,kpi,m,a,band,knobs_hash,build_id,stamp
```

```
2025-10-
01,AR_collected_issued,0.9600,0.8400,A++,<knobshash>,pilotW1,"SSMCLOCK1|202
5-10-01T23:59:59Z|2|123.4|<sha256>|<chain>"
2025-10-
01,Refunds_agreement,0.0200,0.6640,A+,<knobshash>,pilotW1,"SSMCLOCK1|2025-
10-01T23:59:59Z|2|123.4|<sha256>|<chain>"
2025-10-
01,Revenue_actual,980000,0.6170,A+,<knobshash>,pilotW1,"SSMCLOCK1|2025-10-
01T23:59:59Z|2|123.4|<sha256>|<chain>"
```

Optional diagnostic columns (display-only, not required for CI):
`Z, A, Delta, Q, SyZ, U, W`

- `Z := log(1 + E_t); A := EWMA(Z); Delta := |Z - A|`
- `Q` and `SyZ` from the calm gate; `U,W` are rapidity accumulator internals

---

7.3 Manifest template (ASCII, v1.0 + guardrails)

```
[SSM_AUDIT_MANIFEST]
version = 1.0
build_id = pilotW1
date_utc = 2025-10-01

# Numerics + fusion
eps_a = 1e-6
eps_w = 1e-12
gamma = 1.0
division_policy = strict

# Bands + hysteresis
bands = {A++=0.75, A+=0.50, A0=0.25, A-=0.10, A--=-inf}
hysteresis = {promote=0.05, demote=-0.05}

# Mappings (declare once; stable during pilot)
mappings = {
  AR: coverage(window=30d, tail_weight=0.5),
  Refunds: agreement(b=0.005),
  Forecast: residual(k=1.2, s=<rolling_MAD>)
}

# Multi-currency / macro guardrails (optional but recommended)
ccy_base = USD
fx_source = <rate_table_or_provider>
fx_time = 2025-10-01T00:00:00Z
index_source = <CPI_or_equivalent>
index_time = 2025-10-01T00:00:00Z
k_fx = 0.0              # in [0,1], 0 -> ignore fx quality in a
g_macro_policy = none    # {none|scalar|series}

# Privacy
privacy_mode = aggregate_only  # a and stamps never carry PII

# Fingerprint
conformance_checksum = <hex>
knobs_hash := sha256( ascii(
canonical_json({eps_a,eps_w,gamma,division_policy,bands,hysteresis,mappings
```

```
,ccy_base,fx_source,fx_time,index_source,index_time,k_fx,g_macro_policy,pri
vacy_mode}) ) )
```

7.4 Verifier (one-screen, calculator-fast)

Inputs: `pilot.csv`, `SSM_AUDIT_MANIFEST.ini`
Tolerance: `eps_num := 1e-12`
All formulas plain ASCII.

```
load csv rows -> R
load manifest -> M

# C1 Clamp safety (defensive)
for each row r in R:
  r.a := clamp(r.a, -1+M.eps_a, +1-M.eps_a)
  assert(-1 < r.a and r.a < +1) else FAIL("C1", r)

# C2 Collapse parity (identity on m)
assert(all exported m equals source m byte-for-byte where applicable) else
FAIL("C2")

# C3 Band parity
def band(a):
  if a >= 0.75: return "A++"
  elif a >= 0.50: return "A+"
  elif a >= 0.25: return "A0"
  elif a >= 0.10: return "A-"
  else: return "A--"
for r in R: assert(r.band == band(r.a)) else FAIL("C3", r)

# C4 Order invariance (batch vs stream vs shuffled)
def fuse(lanes, weights, eps_w):
  U := 0.0; W := 0.0
  for (a,w) in (lanes,weights):
    U := U + w*atanh(a)
    W := W + w
  return tanh( U / max(W, eps_w) )

S := subset of rows for a chosen KPI/day-window
lanes := [r.a for r in S]; weights := [ abs(r.m)^M.gamma for r in S ]
a_batch := fuse(lanes, weights, M.eps_w)
a_stream := fold_left_streaming_fuse(S in time order)
a_shuffle := fuse(shuffle(lanes), shuffle(weights), M.eps_w)
assert( |a_batch - a_stream| <= eps_num and |a_batch - a_shuffle| <=
eps_num ) else FAIL("C4")

# C5 Ratio parity (if factors present)
# For m := f/g with lanes a_f, a_g:
a_div_pre  := tanh( atanh(a_f) - atanh(a_g) )  # pre-join
a_div_post := recomputed from joined rows
assert( |a_div_pre - a_div_post| <= eps_num ) else FAIL("C5")

# C6 Rollup parity (nested vs flat)
assert( |a_nested - a_flat| <= eps_num ) else FAIL("C6")

# C7 Deterministic export
# Re-run same inputs/manifest; diff a and band (after formatting). Expect
no change.
```

```
# C8 Knobs fingerprint
recompute knobs_hash from manifest; assert equals csv.knobs_hash on all
rows

# C9 Chain continuity (if stamps)
# chain_0 := "0"*64
# chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )
recompute; assert equality

# C10 Day anchor parity (if stamps)
# rollup_D := sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )
recompute; assert equality

# C11 Hysteresis sanity
# Promotions require delta_a >= promote; demotions <= demote
check per-kpi day sequence

# C12 Alert calmness (optional)
# Enforce weekly alert budget vs policy

# C13 Multi-currency collapse parity (if fx normalization used)
# m_fx_norm := m_native * fx_rate(native->base)
# a may optionally blend with fx data quality; verify a remains bounded
assert(phi((m_fx_norm, a)) == m_fx_norm) else FAIL("C13")

# C14 Hyperinflation guard (if indexation used)
# m_real := m_nominal * (I_0 / I_t)
# lane boundedness: |a| <= 1 - eps_a; document g_macro_t if applied
assert(|a| < 1 - M.eps_a) else FAIL("C14")

# C15 Band hysteresis monotonicity
# a noise within epsilon must not flip band when thresholds unchanged
simulate small +/- epsilon; assert no band flip

# C16 Stamp replay determinism
# Re-export, compress/uncompress file; recompute chain and rollups; hashes
identical
```

Output: a single text page with `C1..C16 : PASS/WARN/FAIL (reason, count, sample keys).`

7.5 Band + hysteresis state machine (drop-in)

Inputs: `prev_band in {A++,A+,A0,A-,A--}, a_now in (-1,+1), promote := 0.05, demote := -0.05`

```
def band_from_a(a):
  if a >= 0.75: return "A++"
  elif a >= 0.50: return "A+"
  elif a >= 0.25: return "A0"
  elif a >= 0.10: return "A-"
  else: return "A--"

def next_band(prev_band, a_prev, a_now):
  raw := band_from_a(a_now)
```

```
  delta_a := a_now - a_prev
  score := {"A--":1,"A-":2,"A0":3,"A+":4,"A++":5}
  if score[raw] > score[prev_band] and delta_a >= promote: return raw
  if score[raw] < score[prev_band] and delta_a <= demote:  return raw
  return prev_band
```

This suppresses flicker while allowing real movement.

7.6 Conformance checklist (C1..C16 quick sheet)

`C1 Clamp safety ->` all `a` clamped to `(-1,+1)`
`C2 Collapse parity ->` `phi((m,a)) = m` everywhere
`C3 Band parity ->` `band == band_from_a(a)`
`C4 Order invariance ->` batch == stream == shuffled (`<= eps_num`)
`C5 Ratio lane parity ->` pre- vs post-join `a_div` match
`C6 Rollup parity ->` nested vs flat `fuse()` match
`C7 Deterministic export ->` repeat run identical (`m,a,band`)
`C8 Knobs fingerprint ->` `knobs_hash` matches manifest
`C9 Chain continuity ->` `chain_k` recomputes
`C10 Day anchor parity ->` `rollup_D` recomputes
`C11 Hysteresis sanity ->` promotions/demotions respect thresholds
`C12 Alert calmness ->` alert budget within policy
`C13 Multi-currency parity ->` `phi((m_fx_norm,a)) == m_fx_norm`; `a` bounded
`C14 Hyperinflation guard ->` `m_real := m_nominal*(I_0/I_t)` with `|a| < 1-eps_a`
`C15 Band monotonicity ->` small `a` noise cannot flip band
`C16 Stamp replay determin.->` replays reproduce identical hashes

7.7 Stamp spec & verifier (ASCII)

Stamp line:
`SSMCLOCK1|iso_utc|rasi_idx|theta_deg|sha256(file)|chain`
Chain rule:
```
chain_0 := "0"*64
chain_k := sha256( ascii(chain_{k-1} + "|" + stamp_core_k) )
```
Day anchor:
```
rollup_D := sha256( ascii(Stamp_1 "|" ... "|" Stamp_n) )
```
Constraints: no `:60` seconds; unknown `kv:` keys ignored by verifiers; stamps never alter `m`.
C16 adds replay determinism: compress/decompress and re-emit → hashes identical.

7.8 Pilot acceptance note (paste and fill)

```
[SSM_AUDIT_ACCEPTANCE]
org = <name>          date_utc = <YYYY-MM-DD>       build_id = <id>

scope = [AR_collected_issued, Refunds_agreement, Revenue_actual]
```

```
manifest_version = 1.0   knobs_hash = <hex>

currency = {base: <ccy_base>, fx_source: <id>, fx_time: <ISO8601>}
indexation = {source: <index_source>, index_time: <ISO8601>, policy:
<g_macro_policy>}

parity_m = PASS          # phi((m,a)) = m
order_invariance = PASS  # batch == stream == shuffled
stamps = ENABLED|DISABLED  # if enabled: chain continuity = PASS, replay =
PASS

band_results = {
  AR_collected_issued : {A++:X, A+:Y, A0:Z, A-:U, A--:V},
  Refunds_agreement   : {...},
  Revenue_actual      : {...}
}

caught_early = [
  {kpi: "Refunds_agreement", first_drop: "2025-10-04", action: "recon
staffing", outcome: "variance halved"},
  {kpi: "AR_collected_issued", first_drop: "2025-10-07", action: "credit
tighten on segment S", outcome: "tail reduced"}
]

exceptions = [none|list with owner+date]
sign_off = {finance_owner: <name>, data_owner: <name>, internal_audit:
<name>}
```

7.9 Change log template (semver discipline)

```
[SSM_AUDIT_CHANGELOG]
version_from = 1.0         version_to = 1.0
build_id_from = pilotW1     build_id_to = pilotW2
date_utc = 2025-10-15
changes = [
  {"bands": "no change"},
  {"hysteresis": "no change"},
  {"mappings": "Forecast residual k: 1.2 -> 1.3 ; s: MAD_28d unchanged"},
  {"ci": "added C12 alert budget check"},
  {"guardrails": "enabled C13..C16 for fx/indexation/hysteresis
monotonicity/replay"}
]
knobs_hash_from = <hex>     knobs_hash_to = <hex>
backfill_policy = "none|full|window:<dates>"
owner = <name>
```

7.10 Glossary (one-liners)

- **x := (m,a)** — two-lane numeral: classical value `m` plus bounded stability lane `a in (-1,+1)`.
- **phi((m,a)) = m** — collapse parity: numbers remain identical.
- **Band** — label from `a` (`A++, A+, A0, A-, A--`) for decisioning.
- **Hysteresis** — promote/demote only when `delta_a` crosses declared bounds; prevents flicker.

- **Fuse** — order-invariant lane combination: `a_out := tanh( (Σ w*atanh(a)) / max(Σ w, eps_w) )`.
- **knobs_hash** — fingerprint of declared settings for reproducibility.
- **Stamp** — ASCII line for tamper-evident ordering: `SSMCLOCK1|...|sha256(file)|chain`.
- **Rapidity** — `u := atanh(a)` (add/subtract in `u`, then `a := tanh(u)` to stay bounded).
- **Z, A, Delta** — entropy canonical tracks: `Z_t := log(1 + E_t)`, `A_t := EWMA(Z_t)`, `Delta_t := |Z_t - A_t|`.
- **SyZ_t** — calm-gated alignment index in `[0,1]` (optional display lens); lane form `a_syasys := 2*SyZ_t - 1`.
- **a_sdi** — portfolio roll-up: `a_sdi := tanh( (Σ atanh(a_i)) / max(n,1) )`.
- **ccy_base, fx_source** — base currency and FX source for economics parity (lanes unaffected).
- **g_macro_policy** — macro gate for indexation (`none|scalar|series`) ensuring bounded lanes.

7.11 — Mini Calculator (CLI) — drop-in tool

Purpose. Run a quick read-only check on your KPI stream; compute `a`, `band`, `band_hyst` (with hysteresis), optional alerts, optional chart, and (optionally) the SDI portfolio index.

CSV columns (same as §7.2).
`time,kpi,m,a,band,knobs_hash,build_id,stamp`
(You may leave `a,band,knobs_hash,stamp` blank; the script fills them.)

Core math (ASCII; implemented below)

- Clamp: `a := clamp(a, -1+eps_a, +1-eps_a)`
- Band: thresholds `A++/A+/A0/A-/A--` on `a`
- Hysteresis: promote only if `delta_a >= +promote`; demote only if `delta_a <= demote`
- Order-invariant fusion (multiple child rows for same `(time,kpi)`): `a_out := tanh( (SUM w*atanh(a_i)) / max(SUM w, eps_w) )`
- Collapse parity: `phi((m,a)) = m`
- Knobs fingerprint: `knobs_hash := sha256(ascii(canonical_json(knobs)))`
- Portfolio index (SDI): `a_sdi := tanh( (SUM atanh(a_i)) / n )`

Two ways to run (single script)

- **Demo mode (`--demo`)**: generates a small, reproducible sample *(no input CSV required)*. Controls: `--demo_days`, `--demo_start`. Outputs: `mini_calc_output.csv` (or your `output_csv`), optional `alerts.csv`, `Plots\<kpi>_DEMO.png`, and `Plots\SDI.png` when `--sdi --sdi_plot` are used.

- **CSV mode (default)**: reads your CSV. If a mapper column is present (or `--compute_a on`), computes `a` end-to-end:
  - coverage: `a := 2*q - 1`
  - agreement: `a := tanh( 1 - |m1 - m2| / b )`
  - residual: `a := tanh( k*( 1 - |actual - forecast| / s ) )`

Plots. Auto-names images by run mode: `<kpi>_DEMO.png` (demo) or `<kpi>_CSV.png` (CSV). Optional: `--plot_tag <suffix>` to customize; `--plot_keep_plain` to also write a plain `<kpi>.png`. Colors configurable via `--color_m`, `--color_a`.

SDI (native in CLI). Use `--sdi` to compute portfolio rows (`kpi=SDI`) after purging any previous SDI rows.
Optional filter subset with `--sdi_kpis "<csv>"`; SDI-only file via `--sdi_csv`; plot via `--sdi_plot`.

Windows CMD examples (caret ^ for line breaks)

Demo (no CSV), orange lane, auto DEMO-tagged plot

```
python ssm_audit_mini_calc.py --demo --demo_days 10 --demo_start 2025-10-01
^
  --build_id demoW3 ^
  --alerts_csv alerts.csv --slope_7d -0.02 ^
  --plot_kpi Revenue_actual --plots_dir Plots ^
  --color_a orange ^
  --sdi --sdi_plot
```

Actual CSV, auto-compute `a` from mappers, orange lane, auto CSV-tagged plot

```
python ssm_audit_mini_calc.py mini_calc_template.csv mini_calc_output.csv ^
  --compute_a auto ^
  --promote 0.05 --demote -0.05 --gamma 1.0 --eps_a 1e-6 --eps_w 1e-12 ^
  --build_id pilotW1 --alerts_csv alerts.csv --slope_7d -0.02 ^
  --plot_kpi Revenue_actual --plots_dir Plots ^
  --color_a orange ^
  --sdi --sdi_plot
```

Outputs

- Updated CSV with `a,band,band_hyst,knobs_hash,build_id` (plus SDI rows if `--sdi`).
- Optional `alerts.csv` (rules: `degrade_2of3`, `slope_7d`).
- Optional plots `<kpi>_DEMO.png` or `<kpi>_CSV.png` and `SDI.png`.

Script (single file; copy-paste as your CLI)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
SSM-Audit Mini Calculator (CLI)
```

```
Canonicals (ASCII):
  phi((m,a)) = m
  Clamp: a := clamp(a, -1+eps_a, +1-eps_a)
  Band thresholds: A++: a>=0.75; A+: a>=0.50; A0: a>=0.25; A-: a>=0.10;
else A--
  Hysteresis: promote if delta_a >= +promote; demote if delta_a <= demote
  Rapidity fuse: a_out := tanh( (SUM w*atanh(a)) / max(SUM w, eps_w) )
  SDI (portfolio): a_sdi := tanh( (SUM atanh(a_i)) / n )
"""

import argparse, csv, json, hashlib, math, os, sys
from collections import defaultdict, deque
from datetime import datetime, timezone, timedelta
from typing import List, Dict, Tuple

# ---------- helpers ----------
def clamp(x, lo, hi):
    return lo if x < lo else hi if x > hi else x

SCORE = {"A--":1,"A-":2,"A0":3,"A+":4,"A++":5}

def band_from_a(a: float) -> str:
    if a >= 0.75: return "A++"
    elif a >= 0.50: return "A+"
    elif a >= 0.25: return "A0"
    elif a >= 0.10: return "A-"
    else: return "A--"

def canonical_json(d: Dict) -> str:
    return json.dumps(d, sort_keys=True, separators=(",", ":"))

def sha256_ascii(s: str) -> str:
    return hashlib.sha256(s.encode("utf-8")).hexdigest()

def knobs_hash_from(knobs: Dict) -> str:
    return sha256_ascii(canonical_json(knobs))

def parse_iso(ts: str) -> datetime:
    if not ts:
        # UTC, date-only
        return datetime.now(timezone.utc).replace(hour=0, minute=0,
second=0, microsecond=0).replace(tzinfo=None)
    if "T" in ts:
        return
datetime.fromisoformat(ts.replace("Z","+00:00")).replace(tzinfo=None)
    try:
        return datetime.strptime(ts, "%Y-%m-%d")
    except Exception:
        return datetime.fromisoformat(ts)

# ---------- compute-a mappers ----------
def compute_a_from_mapper(row: Dict, eps_a: float) -> Tuple[float, str]:
    mapper = (row.get("mapper","") or "").strip().lower()
    try:
        if mapper == "coverage":
            q = float(row.get("q",""))
            a = 2.0*q - 1.0
            return (clamp(a, -1+eps_a, 1-eps_a), "coverage")
        elif mapper == "agreement":
```

```
            m1 = float(row.get("m1","")); m2 = float(row.get("m2","")); b =
float(row.get("b",""))
            if b <= 0: return (None, "agreement:b<=0")
            d = abs(m1 - m2)
            a = math.tanh(1.0 - d/b)
            return (clamp(a, -1+eps_a, 1-eps_a), "agreement")
        elif mapper == "residual":
            actual = float(row.get("actual","")); forecast =
float(row.get("forecast","")); s = float(row.get("s",""))
            if s <= 0: return (None, "residual:s<=0")
            k = float(row.get("k","1.0") or 1.0)
            r = actual - forecast
            a = math.tanh(k*(1.0 - abs(r)/s))
            return (clamp(a, -1+eps_a, 1-eps_a), "residual")
        else:
            return (None, "no-mapper")
    except Exception as e:
        return (None, f"mapper-error:{mapper}:{e}")

# ---------- hysteresis ----------
def next_band(prev_band: str, a_prev: float, a_now: float, promote: float,
demote: float) -> str:
    raw = band_from_a(a_now)
    d = a_now - (a_prev if a_prev is not None else a_now)
    if SCORE[raw] > SCORE.get(prev_band, SCORE[raw]) and d >= promote:
return raw
    if SCORE[raw] < SCORE.get(prev_band, SCORE[raw]) and d <= demote:
return raw
    return prev_band or raw

# ---------- alerts ----------
def compute_alerts(rows_sorted: List[Dict], slope_7d: float=-0.02) ->
List[Dict]:
    alerts, win = [], deque(maxlen=3)
    for r in rows_sorted:
        win.append(SCORE[r["band_hyst"]])
        if len(win) == 3:
            drops = (win[2] < win[1]) + (win[1] < win[0]) + (win[2] <
win[0])
            if drops >= 2:
                alerts.append({"time": r["time"], "kpi": r["kpi"],
"rule":"degrade_2of3"})
    if len(rows_sorted) >= 8:
        a0, a7 = rows_sorted[-8]["a"], rows_sorted[-1]["a"]
        slope = (a7 - a0) / 7.0
        if slope <= slope_7d:
            alerts.append({"time": rows_sorted[-1]["time"], "kpi":
rows_sorted[-1]["kpi"],
                           "rule":"slope_7d", "value": round(slope,4)})
    return alerts

# ---------- plotting ----------
try:
    import matplotlib.pyplot as plt
    import matplotlib.dates as mdates
    HAVE_PLOT = True
except Exception:
    HAVE_PLOT = False

def plot_kpi(rows_sorted: List[Dict], kpi: str, out_dir: str,
             color_m: str="tab:blue", color_a: str="tab:orange",
```

```
            tag: str=None, keep_plain: bool=False) -> str:
    if not HAVE_PLOT:
        print("[plot] matplotlib not available; skipping.")
        return None

    ts = [parse_iso(r["time"]) for r in rows_sorted]
    m  = [float(r["m"]) for r in rows_sorted]
    a  = [float(r["a"]) for r in rows_sorted]

    fig, ax1 = plt.subplots(figsize=(12,4))
    ax1.plot(ts, m, label=f"m (classical: {kpi})", linewidth=2,
color=color_m, zorder=3)
    ax1.set_xlabel("time"); ax1.set_ylabel("m")
    ax1.xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))

    ax2 = ax1.twinx()
    ax2.plot(ts, a, label="a (lane)", linewidth=2, linestyle="--",
color=color_a, zorder=3)
    ax2.set_ylabel("a in (-1,+1)"); ax2.set_ylim(-1.0, 1.0)

    for th in [0.10, 0.25, 0.50, 0.75]:
        ax2.axhline(th, linestyle="--", linewidth=1, alpha=0.35,
label="_nolegend_", zorder=1)

    lines = [l for l in (ax1.get_lines() + ax2.get_lines())
             if l.get_label() and not l.get_label().startswith("_")]
    labels = [l.get_label() for l in lines]
    leg = ax1.legend(lines, labels, loc="upper left",
                     frameon=True, framealpha=1.0, facecolor="white",
edgecolor="0.8")
    leg.set_zorder(10)

    ax1.grid(True, linestyle="--", alpha=0.35, zorder=0)
    fig.tight_layout()

    os.makedirs(out_dir, exist_ok=True)
    base = f"{kpi}.png"
    tagged = f"{kpi}_{tag}.png" if tag else base
    path_tagged = os.path.join(out_dir, tagged)
    fig.savefig(path_tagged, dpi=150)
    if keep_plain and tagged != base:
        fig.savefig(os.path.join(out_dir, base), dpi=150)
    plt.close(fig)
    return path_tagged

# ---------- SDI helper (native) ----------
# a_sdi := tanh( (SUM atanh(a_i)) / n )
def append_sdi_rows_from_csv(output_csv_path: str, eps_a: float, sdi_kpis:
str,
                             sdi_csv: str, do_plot: bool, plots_dir: str):
    def _clamp(a, eps):
        if a > 1.0 - eps: return 1.0 - eps
        if a < -1.0 + eps: return -1.0 + eps
        return a
    def _norm(name: str) -> str:
        return (name or "").strip().strip('"').strip("'")

    # read & purge existing SDI rows
    header, base_rows = None, []
    with open(output_csv_path, "r", encoding="utf-8") as f:
        rdr = csv.DictReader(f)
```

```
        header = rdr.fieldnames or
["time","kpi","m","a","band","band_hyst","knobs_hash","build_id","stamp"]
        for r in rdr:
            if (r.get("kpi","") or "") == "SDI":
                continue
            base_rows.append(r)

    # robust filter: strip quotes/spaces around tokens
    filt = set(_norm(s) for s in (sdi_kpis or "").split(",") if _norm(s))

    # collect a-values per day after optional KPI filter
    per_day = defaultdict(list)
    for r in base_rows:
        kpi = _norm(r.get("kpi",""))
        if filt and kpi not in filt:
            continue
        t = r.get("time","")
        if not t:
            continue
        try:
            a_val = float(r.get("a",""))
        except Exception:
            continue
        per_day[t].append(a_val)

    # compute SDI rows
    sdi_rows = []
    for t in sorted(per_day.keys()):
        a_list = per_day[t]
        if not a_list:
            continue
        u_sum = 0.0
        for ai in a_list:
            u_sum += math.atanh(_clamp(ai, eps_a))
        a_sdi = math.tanh(u_sum / max(len(a_list), 1))
        sdi_rows.append({
            "time": t, "kpi": "SDI", "m": "0.0", "a": f"{a_sdi:.10f}",
            "band": band_from_a(a_sdi), "band_hyst": band_from_a(a_sdi),
            "knobs_hash": "", "build_id": "", "stamp": ""
        })

    # write purged + SDI
    with open(output_csv_path, "w", newline="", encoding="utf-8") as f:
        w = csv.DictWriter(f, fieldnames=header)
        w.writeheader()
        for r in base_rows:
            w.writerow(r)
        for r in sdi_rows:
            w.writerow(r)

    # optional SDI-only CSV
    if sdi_csv:
        with open(sdi_csv, "w", newline="", encoding="utf-8") as f:
            w = csv.DictWriter(f,
fieldnames=["time","kpi","m","a","band","band_hyst","knobs_hash","build_id"
,"stamp"])
            w.writeheader()
            for r in sdi_rows:
                w.writerow(r)

    # optional SDI plot
```

```
    if do_plot and HAVE_PLOT and sdi_rows:
        try:
            xs = [r["time"] for r in sdi_rows]
            ys = [float(r["a"]) for r in sdi_rows]
            fig, ax = plt.subplots(figsize=(10,3.5))
            ax.plot(xs, ys, marker="o")
            ax.set_title("SDI (tanh-mean of lanes)")
            ax.set_xlabel("time"); ax.set_ylabel("a_sdi")
            ax.grid(True, linestyle="--", alpha=0.35)
            fig.tight_layout()
            os.makedirs(plots_dir, exist_ok=True)
            out_path = os.path.join(plots_dir, "SDI.png")
            fig.savefig(out_path, dpi=150)
            plt.close(fig)
            print(f"[OK] SDI plot: {out_path}")
        except Exception as e:
            print(f"[WARN] SDI plot skipped: {e}")

    print(f"[OK] SDI appended: {len(sdi_rows)} rows")

# ---------- DEMO generator ----------
def demo_rows(start_iso: str, days: int, eps_a: float) -> List[Dict]:
    start = parse_iso(start_iso) if start_iso else \
            datetime.now(timezone.utc).replace(hour=0, minute=0, second=0,
microsecond=0).replace(tzinfo=None)
    out = []
    k, s  = 1.2, 50_000.0
    b     = 0.005
    q0    = 0.88

    for i in range(days):
        t = (start + timedelta(days=i)).strftime("%Y-%m-%d")

        # Revenue_actual (smooth ramp; one mid-curve miss)
        forecast = 1_000_000 + 30_000*i
        actual   = forecast + (20_000 if i==6 else (-10_000 if i<3 else
5_000))
        r = actual - forecast
        a_res = math.tanh(k*(1 - abs(r)/s))
        out.append({"time":t,"kpi":"Revenue_actual","m":float(actual),
                    "a": clamp(a_res, -1+eps_a, 1-eps_a)})

        # AR_collected_issued (coverage easing then recovering)
        q = q0 + 0.01*i if i<3 else q0 + 0.03 - 0.008*(i-3)
        q = max(0.0, min(1.0, q))
        a_cov = 2*q - 1
        out.append({"time":t,"kpi":"AR_collected_issued","m":0.96,
                    "a": clamp(a_cov, -1+eps_a, 1-eps_a)})

        # Refunds_agreement (drift then re-align)
        m_platform = 0.021 + 0.0002*i
        m_bank     = 0.020 + (0.0004*i if i<5 else 0.00005*i)
        d = abs(m_platform - m_bank)
        a_agree = math.tanh(1 - d/b)
        out.append({"time":t,"kpi":"Refunds_agreement","m":m_platform,
                    "a": clamp(a_agree, -1+eps_a, 1-eps_a)})
    return out

# ---------- main ----------
def main():
```

```
    p = argparse.ArgumentParser(description="SSM-Audit Mini Calculator
(values unchanged: phi((m,a)) = m)")

    # positional (optional in demo mode)
    p.add_argument("input_csv", nargs="?", default=None)
    p.add_argument("output_csv", nargs="?", default=None)

    # knobs
    p.add_argument("--build_id", default="demo")
    p.add_argument("--eps_a", type=float, default=1e-6)
    p.add_argument("--eps_w", type=float, default=1e-12)
    p.add_argument("--gamma", type=float, default=1.0)
    p.add_argument("--promote", type=float, default=0.05)
    p.add_argument("--demote",  type=float, default=-0.05)

    # alerts / plotting
    p.add_argument("--alerts_csv")
    p.add_argument("--slope_7d", type=float, default=-0.02)
    p.add_argument("--plot_kpi")
    p.add_argument("--plots_dir", default="Plots")
    p.add_argument("--color_m", default="tab:blue")
    p.add_argument("--color_a", default="tab:orange")
    p.add_argument("--plot_tag", default="", help="suffix for plot
filename; defaults to DEMO/CSV by mode")
    p.add_argument("--plot_keep_plain", action="store_true", help="also
write plain <kpi>.png")

    # SDI (portfolio roll-up)
    p.add_argument("--sdi", action="store_true", help="compute portfolio
SDI across KPIs per day")
    p.add_argument("--sdi_kpis", default="", help="comma-separated KPI
filter for SDI (default: all KPIs)")
    p.add_argument("--sdi_csv", default="", help="optional SDI-only CSV
output")
    p.add_argument("--sdi_plot", action="store_true", help="also plot SDI")

    # demo controls
    p.add_argument("--demo", action="store_true", help="run built-in sample
without CSV")
    p.add_argument("--demo_days", type=int, default=10)
    p.add_argument("--demo_start", default="", help="YYYY-MM-DD
(optional)")

    # CSV mapper control
    p.add_argument("--compute_a", default="auto",
choices=["auto","on","off"],
                   help="build 'a' from mapper columns
(coverage/agreement/residual)")

    args = p.parse_args()

    # knobs hash (deterministic fingerprint)
    knobs = {
        "eps_a": args.eps_a, "eps_w": args.eps_w, "gamma": args.gamma,
        "division_policy": "strict",
        "bands": {"A++":0.75, "A+":0.50, "A0":0.25, "A-":0.10, "A--":-
float("inf")},
        "hysteresis": {"promote": args.promote, "demote": args.demote},
        "mappings": "mini_calc"
    }
    khash = knobs_hash_from(knobs)
```

```
    # --- Build rows: DEMO or CSV ---
    rows: List[Dict] = []

    if args.demo:
        rows = demo_rows(args.demo_start, args.demo_days, args.eps_a)
        if not args.output_csv:
            args.output_csv = "mini_calc_output.csv"
    else:
        if not args.input_csv or not args.output_csv:
            print("[ERR] Provide input_csv and output_csv, or use --demo")
            sys.exit(2)
        with open(args.input_csv, newline="", encoding="utf-8") as f:
            rdr = csv.DictReader(f)
            header = [h.strip() for h in (rdr.fieldnames or [])]
            have_mapper = "mapper" in [h.lower() for h in header]
            compute_flag = (args.compute_a == "on") or (args.compute_a ==
"auto" and have_mapper)
            for r in rdr:
                r = {k: (r.get(k,"") or "").strip() for k in r.keys()}
                # fill a if needed
                a_value = r.get("a","")
                a_float = None
                if compute_flag and (a_value == "" or a_value.lower() ==
"na"):
                    a_float, _reason = compute_a_from_mapper(r, args.eps_a)
                else:
                    try:
                        a_float = float(a_value)
                    except Exception:
                        a_float = 0.0
                r["a"] = clamp(a_float if a_float is not None else 0.0, -
1+args.eps_a, 1-args.eps_a)

                # default m if residual mapper present
                if r.get("m","") == "" and (r.get("mapper","") or
"").lower() == "residual":
                    try:
                        r["m"] = float(r.get("actual","0") or 0.0)
                    except Exception:
                        r["m"] = 0.0
                rows.append(r)

    # --- Group by (time,kpi); fuse a if multiple rows share the key ---
    grouped: Dict[Tuple[str,str], List[Dict]] = defaultdict(list)
    for r in rows:
        key = (r.get("time",""), r.get("kpi",""))
        grouped[key].append(r)

    fused_rows: List[Dict] = []
    for (t,k), lst in grouped.items():
        lst_sorted = sorted(lst, key=lambda z: canonical_json(z))
        # classical m: prefer explicit m if present; else 0.0
        m_num = None
        for mv in [z.get("m","") for z in lst_sorted]:
            try:
                m_num = float(mv); break
            except Exception:
                continue
        if m_num is None: m_num = 0.0
```

```
        # fuse a in rapidity space with optional weights
        num, den = 0.0, 0.0
        for z in lst_sorted:
            a_i = float(z.get("a",0) or 0)
            w_i = float(z.get("weight",1) or 1)
            a_i = clamp(a_i, -1+args.eps_a, 1-args.eps_a)
            num += w_i * math.atanh(a_i)
            den += w_i
        a_out = math.tanh(num / max(den, args.eps_w))

        fused_rows.append({
            "time": t or datetime.now(timezone.utc).strftime("%Y-%m-%d"),
            "kpi": k, "m": m_num, "a": a_out
        })

    # --- Per-KPI time sort, hysteresis, hash/build tags ---
    by_kpi: Dict[str, List[Dict]] = defaultdict(list)
    for r in fused_rows:
        r["band"] = band_from_a(r["a"])
        by_kpi[r["kpi"]].append(r)

    latest = {}
    for kpi, lst in by_kpi.items():
        lst.sort(key=lambda z: parse_iso(z["time"]))
        prev_band, prev_a = None, None
        for r in lst:
            r["band_hyst"] = next_band(prev_band, prev_a, r["a"],
args.promote, args.demote)
            r["knobs_hash"] = khash
            r["build_id"] = args.build_id
            prev_band, prev_a = r["band_hyst"], r["a"]
        latest[kpi] = lst[-1]

    # --- Write output CSV ---
    fieldnames =
["time","kpi","m","a","band","band_hyst","knobs_hash","build_id","stamp"]
    with open(args.output_csv, "w", newline="", encoding="utf-8") as f:
        w = csv.DictWriter(f, fieldnames=fieldnames); w.writeheader()
        for kpi in sorted(by_kpi.keys()):
            for r in by_kpi[kpi]:
                out = {k: r.get(k,"") for k in fieldnames}
                out["a"] = f"{float(out['a']):.4f}"
                w.writerow(out)

    # --- Alerts (optional) ---
    if args.alerts_csv:
        all_alerts = []
        for kpi, lst in by_kpi.items():
            all_alerts.extend(compute_alerts(lst, args.slope_7d))
        with open(args.alerts_csv, "w", newline="", encoding="utf-8") as f:
            w = csv.DictWriter(f, fieldnames=["time","kpi","rule","value"])
            w.writeheader()
            for ar in all_alerts: w.writerow(ar)
        print(f"Alerts written to: {args.alerts_csv}")

    # --- Plot KPI (optional) ---
    if args.plot_kpi and args.plot_kpi in by_kpi:
        tag = args.plot_tag or ("DEMO" if args.demo else "CSV")
        path = plot_kpi(by_kpi[args.plot_kpi], args.plot_kpi,
args.plots_dir,
                        color_m=args.color_m, color_a=args.color_a,
```

```
                        tag=tag, keep_plain=args.plot_keep_plain)
        if path: print(f"Plot saved: {path}")

    # --- SDI (native; purge+append) ---
    if args.sdi:
        append_sdi_rows_from_csv(
            output_csv_path=args.output_csv,
            eps_a=args.eps_a,
            sdi_kpis=args.sdi_kpis,
            sdi_csv=args.sdi_csv,
            do_plot=args.sdi_plot,
            plots_dir=args.plots_dir
        )

    # --- Console summary ---
    print("=== Mini Calculator Summary (latest by KPI) ===")
    for kpi, r in latest.items():
        print(f"{kpi}: m={r['m']}, a={float(r['a']):.4f}, band={r['band']},
band_hyst={r['band_hyst']}")
    if args.sdi:
        try:
            last_sdi = None
            with open(args.output_csv, "r", encoding="utf-8") as f:
                for row in csv.DictReader(f):
                    if row.get("kpi") == "SDI":
                        last_sdi = row
            if last_sdi:
                print(f"\nSDI: m={last_sdi.get('m','0.0')},
a={float(last_sdi['a']):.4f}, "
                      f"band={last_sdi.get('band')},
band_hyst={last_sdi.get('band_hyst')}")
        except Exception:
            pass

    print(f"\nOutput written to: {args.output_csv}")
    print(f"knobs_hash: {knobs_hash_from(knobs)}")

if __name__ == "__main__":
    main()
```

OMP