Measuring Blockchain Coherence: From Oracle to Consensus

A phased approach to making blockchains measurable and self-correcting

Peter Lisovin
TSC Blockchain Project
November 2025

Document Version: 1.0.0 (arXiv Ready)
Status: Complete Specification with Validation Plan
Next Milestone: Validation notebooks (Month 1-2)
Phase 1 Timeline: 6 months to production oracle (contingent on validation)

Keywords: blockchain measurement, coherence, DeFi risk, reproducibility, zero-knowledge

---

Abstract

Major blockchain blow-ups share one symptom: α (protocol claims), β (economic/implementation reality), and γ (emergent usage) diverge well before loss. No standard tool measures that divergence. We propose a two-phase program built on Triadic Self-Coherence (TSC):

1. **Coherence Oracle (Phase-1):** Specification for off-chain measurement and on-chain attestation of per-chain coherence C_Σ. Validation planned for Month 1-2.
2. **Proof-of-Coherence (Phase-2):** A checkpoint validity rule layered on PoS/PBFT requiring C_Σ ≥ 0 or friction improvement Δλ_Σ ≥ δ.

Phase-1 specification provides the framework; the validation notebooks prove whether the framework works as designed.

I. The Observable Pattern

1.1 Operational Definitions

Given a time window W on a chain:

- **α (Protocol/Pattern):** Promised properties—security claims, tokenomics, governance process, performance targets.
- **β (Economics/Relation):** Deployed code and state—stake distribution, fee/MEV dynamics, realized performance.
- **γ (Usage/Process):** Trajectory—transaction mix, retention/turnover, upgrade adoption, incident timelines.

We compute α_c, β_c, γ_c ∈ [0,1] and aggregate:

$C_\Sigma = (\alpha_c \cdot \beta_c \cdot \gamma_c)^{1/3}$, $\lambda_\Sigma = -\ln(\max(C_\Sigma, \varepsilon))$

with numerical floor **$\varepsilon = 10^{-12}$**.

Degeneracy guard: If any axis is 0, C_Σ = 0 (no compensation from other axes).

Default thresholds:

- Production ≥ 0.80
- Watch 0.60-0.80
- Hazard 0.40-0.60
- Critical < 0.40

1.2 Case Studies (Projected Scores Based on TSC Framework)

Terra/Luna (pre-collapse, April 2022):

- **Projected score:** $C_\Sigma \approx 0.27 \pm 0.05$ (Critical)
- **Basis:** Estimated using TSC framework on known facts:
  - $\alpha_c \approx 0.65$ (protocol claims documented)
  - $\beta_c \approx 0.08$ (Anchor sustainability ratio, reserve coverage)
  - $\gamma_c \approx 0.38$ (usage patterns showed stress)
- **Status:** TO BE VALIDATED in notebook (Month 1-2)
- Validation will either confirm score or refine methodology

The DAO (June 2016):

- **Projected score:** $C_\Sigma \approx 0.56 \pm 0.05$ (Hazard)
- **Basis:** β (implementation vs claim) dominated risk
- **Status:** TO BE VALIDATED in notebook (Month 1-2)

Bitcoin Identity Drift (cash→gold):

- **Projected score:** $C_\Sigma \approx 0.66$ (Watch)
- **Basis:** Functional chain, measurable narrative drift
- **Status:** TO BE VALIDATED in notebook (Month 1-2)

All values are projections. Month 1-2 notebooks will compute scores from frozen snapshots; **if they differ by >±0.05, we tune witnesses before any infrastructure spend**.

II. Minimal TSC We Actually Implement

II.1 Articulation → Features (Per Chain)

Axis	Minimal Feature Set (Measurable Today)
α (claims)	Supply schedule; finality targets; fee model; governance powers & thresholds; security assumptions; declared economic invariants
β (reality)	Verified bytecode & admin rights; validator set & stake concentration; realized latency/finality/throughput; MEV/fee stats; token distribution (Gini); treasury flows; peg/collateral ratios
γ (usage)	Tx taxonomy; holding vs spending; call-graph motifs; retention/turnover; L2 routing; upgrade adoption cadence; incident markers

II.2 Witness Functions (Simplified for Validation)

We compute distances and convert them to scores $s \in [0,1]$.

$W_{\alpha\beta}$ (claims ↔ implementation):

- Coverage: What fraction of α properties have checks?
- Pass rate: What fraction of checks pass?
- Severity: Penalize critical failures
- Score: $\alpha_c = w_1 \cdot (\text{coverage}) + w_2 \cdot (\text{pass\_rate}) - w_3 \cdot (\text{severity\_penalty})$, clipped to [0,1]

$W_{\beta\gamma}$ (incentives ↔ behavior):

- Expected use-mix p from β (fees, yields, latency)
- Observed mix q from γ (actual transaction types)
- Distance: Earth Mover's Distance d = EMD(p, q), normalized to [0,1]
- Score: $\beta_c = 1 - d$

$W_{\gamma\alpha}$ (behavior ↔ intent):

- Reference signature u(t) from α (intended usage pattern)
- Observed trajectory v(t) from γ (actual behavior over time)
- Distance: Normalized edit distance between sequences
- Score: $\gamma_c = 1 - \text{norm\_dist}(u, v)$

Future work: More sophisticated witnesses (optimal transport, graph matching) may be explored in Phase 2 after base framework is validated.

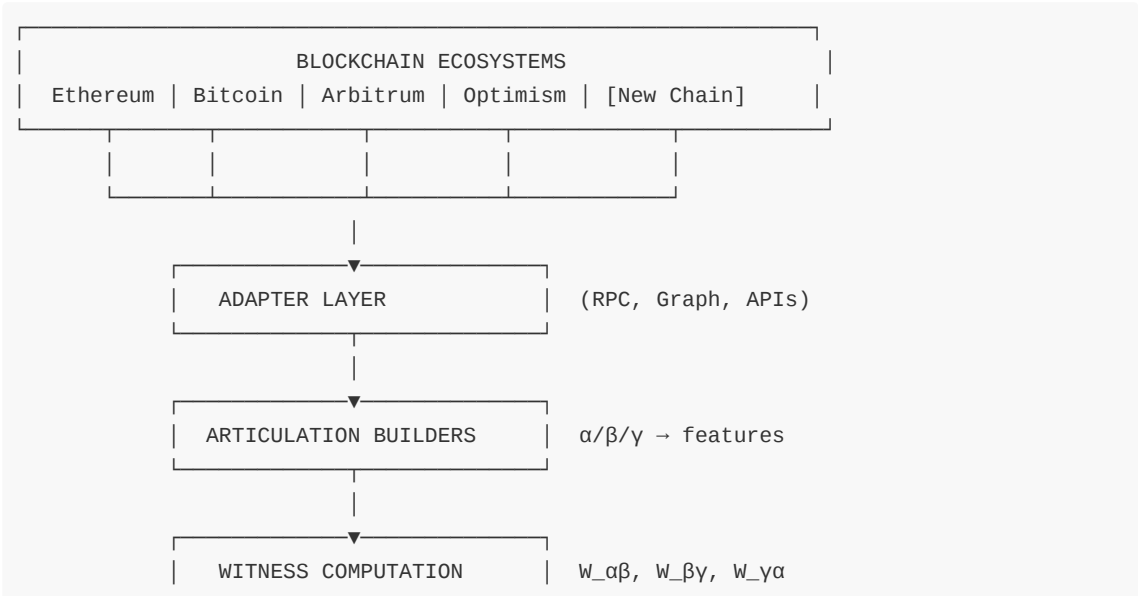
II.3 Global Witness Gates

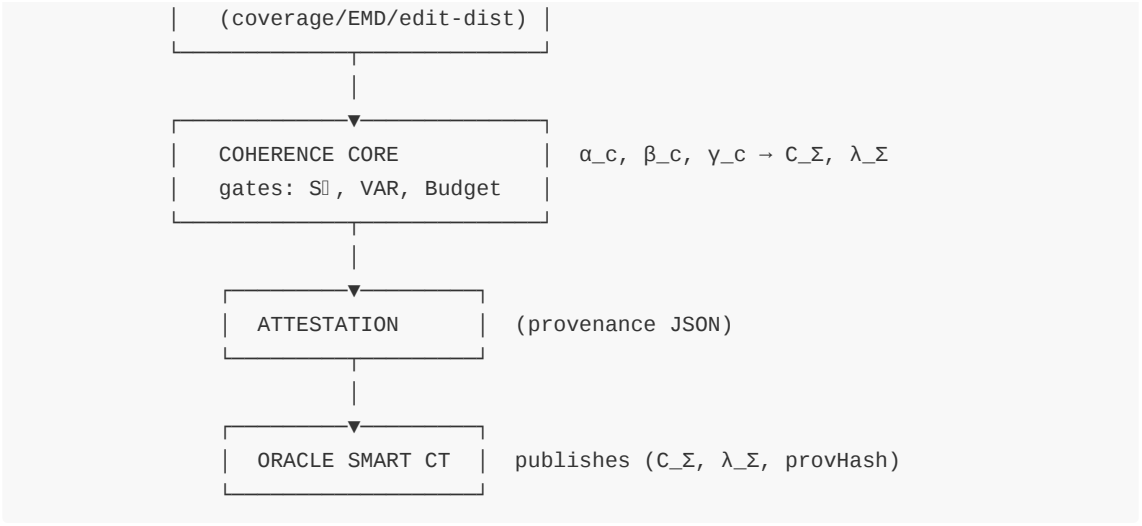
- **S_n symmetry:** C_Σ invariant under axis permutation (within tolerance τ_{sym})
- **Stability:** VRF-seeded repeats; variance ≤ τ_{var}
- **Budget (Phase-1: report-only; Phase-2: enforced):** Report B and $\eta = \Delta\lambda_\Sigma / B$ each run. In Phase-2, require $B \le B_{max}$ and $\eta \ge \eta_{min}$.

Missing-data policy: Primitives that cannot be computed in a window are marked NA; their declared weights are added to an axis penalty bucket (no imputation). The NA mask is written to provenance.

III. Phase-1 — Coherence Oracle (Specification)

III.1 Architecture (Data Flow)





III.2 Interfaces

On-chain ABI:

```
interface ICoherenceOracle {
  struct Report {
    uint64  chainId;
    uint64  blockNumber;
    uint32  methodId;
    uint32  semver;
    uint64  Csigma_milli;    // C_Σ × 1000
    uint64  lambda_milli;    // λ_Σ × 1000
    bytes32 provHash;        // provenance bundle hash
  }

  function latest(uint64 chainId) external view returns (Report memory);
  function verify(Report calldata r) external view returns (bool);
}
```

REST API:

```
GET /v1/coherence/{chain}
→ { C_Σ, λ_Σ, α_c, β_c, γ_c, trend_7d, methodId, provenance_url }
```

III.3 Cadence & SLOs (Tiered)

Tier	Wall Time SLO	Cost SLO	Notes
Standard (hot cache)	≤ 60s	≤ $1	Primary consumption tier
Deep analysis (optional)	≤ 5 min	≤ $5	Adds heavier γ features, MEV
Cold start (first run)	≤ 10 min	≤ $10	First run initialization
Reproducibility	N/A	N/A	≥99.99% identical C_Σ for frozen

			inputs

III.4 Budget-Efficiency Gate: Example

Phase-1 defaults:

- B_max = 10^6 gas-equivalent units
- η_min = 10^(-6) (realistic threshold; will tune on testnet)

Example (Standard tier):

Estimated costs:

- RPC queries (~100 calls): ~$0.10
- Computation (coverage checks, EMD, edit distance): ~$0.02
- On-chain attestation (50k gas @ 20 gwei): ~$2-4
- Storage (IPFS): ~$0.01
- **Total budget B: ~$2.50-4.50**

Estimated improvement:

- Before: C_Σ = 0.82 → λ_Σ = 0.198
- After: C_Σ = 0.84 → λ_Σ = 0.174
- **Improvement: Δλ_Σ ≈ 0.024**

Budget-efficiency:

```
η = Δλ_Σ / B ≈ 0.024 / 4.00 ≈ 6×10^(-3)
```

Gate check: η ≥ η_min = 10^(-6)?
 YES (3 orders of magnitude headroom)

Units: In Phase-1 the gate uses reported USD cost per checkpoint (operator-auditable via invoices); Phase-2 migrates to a gas-equivalent or fixed budget unit (with published weights) for cross-chain comparability.

We will log (Δλ_Σ, B, η) alongside each attestation.

Note: These are engineering estimates. Actual costs will be measured during implementation.

III.5 Validation Strategy: Why Notebooks First

The validation notebooks are **THE critical first deliverable** because they establish whether the TSC framework works as specified. This is a deliberate fork in the development path.

Why Notebooks Before Infrastructure?

1. **Proves concept:** Can TSC detect incoherence retroactively?
2. **Calibrates methodology:** Do scores match expected ranges?
3. **Validates degeneracy:** Does low β_c collapse C_Σ as theory predicts?
4. **Establishes reproducibility:** Can independent teams get same scores?

The Fork: Two Possible Outcomes

Outcome A: Validation Succeeds

- Terra produces C_Σ ≈ 0.27 ± 0.05 (critical threshold)

- DAO produces C_Σ ≈ 0.56 ± 0.05 (hazard threshold)
- Scores are reproducible (99%+ match across runs)
- All global gates pass (S□, variance, budget)

→ **PROCEED:** Build oracle infrastructure (Months 3-4)
→ **Framework proven, specification validated**
→ **Risk: LOW** (methodology works)

Outcome B: Validation Reveals Issues

- Terra produces C_Σ = 0.65 (not critical as expected)
- OR scores not reproducible (variance >5%)
- OR witness functions need tuning

→ **ITERATE:** Refine methodology (additional 2-4 weeks)
→ **Framework needs adjustment, specification revised**
→ **Risk: MEDIUM** (learn what needs fixing)

What We Learn from Outcome B:

- Which witnesses need recalibration
- Whether tolerances are too tight/loose
- If additional α/β/γ features are needed
- How to improve reproducibility

Either outcome is valuable. Success proves the framework; failure teaches us how to improve it. This is why validation comes BEFORE infrastructure investment.

If validation criteria are not met: tune witnesses/tolerances and re-measure before any infrastructure spend.

Deliverables (Month 1-2):

Three public notebooks demonstrating reproducible retroactive measurement:

1. **Terra/Luna notebook**
    - **Snapshot anchor:** as_of: 2022-04-15T00:00:00Z
    - Expected: C_Σ ≈ 0.27 ± 0.05
    - If validated: Strong proof of concept
    - If not: Iterate on witness functions
2. **The DAO notebook**
    - **Snapshot anchor:** as_of: 2016-06-10T00:00:00Z
    - Expected: C_Σ ≈ 0.56 ± 0.05
    - Secondary validation of methodology
3. **Mt. Gox notebook**
    - **Snapshot anchor:** as_of: 2014-02-10T00:00:00Z
    - Expected: C_Σ ≈ 0.31 ± 0.06
    - Bitcoin-specific validation

All notebooks will be public (GitHub) with frozen data snapshots, allowing independent verification. Block heights are resolved inside each notebook from the UTC timestamp anchor.

Success criteria:

- ✅ Scores within ±0.05 of projections
- ✅ Reproducibility ≥99.99% across 10 runs with frozen inputs
- ✅ Witness functions pass S□, stability, budget gates
- ✅ Community can independently verify

Timeline: 2 weeks focused work

- Days 1-3: Terra snapshot ingestion, freeze with hashes
- Days 4-6: Implement feature extractors (coverage/EMD/edit-distance)
- Days 7-8: Assemble terra.ipynb, export attestation
- Days 9-10: Add DAO and Mt.Gox, verify reproducibility

III.6 Attestation Strategy (Provenance-First)

Phase-1 relies exclusively on **reproducible provenance** (stored in IPFS/Arweave) for auditability.

Provenance bundle includes:

- Input data hash
- Method version (methodId)
- Computation timestamp
- Result (C_Σ, α_c, β_c, γ_c, λ_Σ)
- Container image/Nix flake pin
- Data source licenses

Anyone can verify: Same inputs + same method → same result ± tolerance

Zero-knowledge proofs (zk-MFI) are deferred to Phase-2. When measurement moves into consensus, cryptographic guarantees become valuable. Phase-1 commitment: Reproducible provenance is sufficient for oracle use case.

III.7 Consumption Patterns (Concrete Use Cases)

Bridges:

- Accept transfers > $N only if $C_\Sigma \geq 0.80$ AND $\beta_c \geq 0.75$
- Alert when $\Delta C_\Sigma < -0.10$ week-over-week

Lenders:

- Map Loan-to-Value (LTV) from C_Σ with floor at 0.60
- Linear interpolation between 0.60 and 0.80

Exchanges:

- Listing gates: Require $C_\Sigma \geq 0.70$
- Watchlist: Flag when $C_\Sigma < 0.60$
- Delisting consideration: $C_\Sigma < 0.40$ for >30 days

IV. Phase-2 — Proof-of-Coherence (Validity Layer)

We integrate the coherence metric as a validity rule at checkpoints. This is a research direction contingent on Phase-1 validation success.

Checkpoint acceptance:

- If global gates pass AND ($C_\Sigma \geq 0$ OR $\Delta\lambda_\Sigma \geq \delta$): **accept**

- Else: **degraded mode** (rate-limit risky surfaces, quarantine incoherent contracts) until coherence recovers

Default parameters:

- Threshold: 0 = 0.80 (production)
- Minimum improvement: δ = 0.01

Economic sanity: Reward proportional to leverage reduction (Δλ_Σ), applying Budget-Efficiency gate network-wide to prevent "buying" coherence with unbounded compute.

Research questions for Phase-2:

1. Checkpoint frequency optimization
2. Challenge window design
3. Validator economic game theory
4. Migration path (L2 first, then L1)
5. ZK-proof integration

Timeline: 12-18 months after Phase-1 validation.

V. Why This Works (Engineer's Criteria)

TSC is effective because it:

- **Measures the failure mode.** α/β/γ aren't opinions; they're feature sets tied to code, state, and usage.
- **Reproducible.** Provenance bundle makes "same inputs → same numbers" enforceable.
- **Cheap to consume.** Downstream logic needs a scalar and a breakdown.
- **Incremental.** Phase-1 pipelines are exactly what Phase-2 needs.

V.1 How This Relates to Existing Tools

TSC **complements** existing infrastructure:

Tool	What It Measures	TSC Adds
Chainlink	Price feeds, external data	Chain health metrics (C_Σ as new data primitive)
Gauntlet/Chaos Labs	Economic simulation	Continuous measurement of actual vs modeled (β/γ alignment)
Formal Verification	Code correctness	Measures if correct code produces intended economics (α/β/γ coherence)
Audits	Point-in-time security	Continuous monitoring, drift detection

Integration example: DeFi protocol uses Chainlink for prices, Gauntlet for risk params, and TSC for collateral chain health. If TSC reports C_Σ < 0.60 for a bridged chain, Gauntlet adjusts LTV down. Three complementary layers.

V.2 What TSC Does Not Measure

TSC measures coherence (whether claims/economics/usage fit together), **not:**

- Absolute security (use audits)

- Code correctness (use formal verification)
- Market legitimacy (use due diligence)
- Price accuracy (use oracles)

TSC complements these tools by adding dimensional consistency measurement.

V.3 What Could Go Wrong (Honest Risk Assessment)

Validation risks (Month 1-2):

1. **Risk: Scores don't match projections**
    - Terra produces C_Σ = 0.65 instead of 0.27
    - Mitigation: Iterate on witness functions, adjust weights
    - Learning: Discover which articulations need refinement
2. **Risk: Poor reproducibility**
    - Same input produces C_Σ = 0.27 ± 0.15 (too wide)
    - Mitigation: Stabilize computation, add determinism checks
    - Learning: Identify non-deterministic components
3. **Risk: Data availability**
    - Can't fetch April 2022 Terra data (nodes pruned)
    - Mitigation: Use archive nodes, blockchain explorers
    - Fallback: Use publicly available snapshots
4. **Risk: Computational cost exceeds budget**
    - Measurement takes 10 hours instead of 60 seconds
    - Mitigation: Optimize, reduce feature set, adjust SLO tier
    - Learning: Understand actual performance characteristics

These are LEARNING opportunities, not failures. The specification provides the framework; validation teaches us how to implement it correctly.

VI. Roadmap (Execution-Grade)

Timeline	Focus	Deliverable
Months 1-2	**CRITICAL PATH: Validation**	Build and validate Terra/DAO/Mt.Gox notebooks. **FORK: Success → proceed to Month 3. Issues → iterate 2-4 weeks.**
Months 3-4	Oracle Infrastructure **(if validated)**	Smart contract deployment, REST API, dashboards, SLO instrumentation. **Contingent on Month 1-2 success.**
Month 5	Pilots **(if validated)**	Bridge & lending partners integrate; alerting on trend + per-axis drops. **Contingent on oracle deployment.**
Month 6	Public Launch **(if validated)**	Production oracle service; daily measurements for top L1/L2/bridges. **Contingent on pilot success.**
Months 7-18	Phase-2 Research	Checkpoint mechanism, validator economics, challenge window, testnet. **Contingent on Phase-1**

		success.

Critical dependencies:

- Months 3-6 are CONTINGENT on Month 1-2 validation success
- If validation fails, timeline extends by 2-4 weeks for iteration
- Specification provides the framework; notebooks prove it works

---

VII. Engineering Appendix

Minimal α Property Vocabulary (Starter)

Supply schedule; finality target; fee model; validator/committee bounds; governance powers & thresholds; security assumptions; economic invariants (collateral/solvency/peg rules).

W_αβ (Scoring Sketch)

α_c = clip(w□·(coverage) + w□·(pass_rate) - w□·(severity_penalty)) □ [0,1]

where:

- coverage = (properties_with_checks) / (total_properties)
- pass_rate = (checks_passed) / (total_checks)
- severity_penalty = weighted sum of critical failures

W_βγ (Distributional)

Build expected use-mix p from β (fees, latency, yields). Observe q from transaction data. Earth Mover's Distance: d = EMD(p, q), normalize to [0,1] → β_c = 1-d.

W_γα (Drift)

From α, derive reference signature u(t) (e.g., "cash-like": many micro-payments). Compare to observed v(t) via normalized edit distance → γ_c = 1-norm_dist(u,v).

Global Witness Gates

- **S□ symmetry:** Recompute C_Σ under axis permutations; equality within tolerance
- **Stability:** VRF-seeded repeats; variance ≤ τ_var
- **Budget:** B ≤ B_max AND η = Δλ_Σ/B ≥ η_min (Phase-1 start at 10^(-6))

Oracle Policy Templates

Bridge large-transfer gate:

```
C_Σ ≥ 0.80 □ β_c ≥ 0.75
```

Lending LTV curve:

```
LTV = 0.50 + 0.30·(C_Σ - 0.60)/0.20, clamped to [0.50, 0.80]
```

Appendix A — Scoring Primitives (Deterministic)

Let clip(x) = min(1, max(0, x)).

Two-sided range [L, U]:

```
s_range(x; L, U, τ) = clip(1 - max(0, x-U, L-x) / τ)
```

One-sided (≤ T):

```
s_≤(x; T, τ) = clip(1 - max(0, x-T) / τ)
```

One-sided (≥ T):

```
s_≥(x; T, τ) = clip(1 - max(0, T-x) / τ)
```

Invariant check:

```
s_inv = 1 if holds else 0
```

Distributional (JS divergence d ∈ [0,1]):

```
s_dist = 1 - d
```

Process distortion:

```
s_proc = clip(1 - distortion / τ)
```

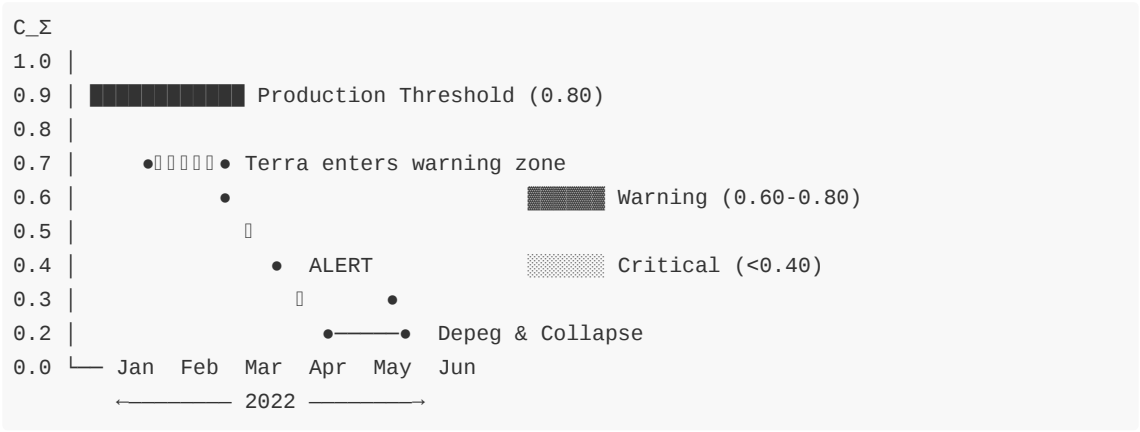
Axis scores are geometric means of their per-witness scores; C_Σ is the geometric mean of axis scores.

Appendix B — Visual Diagrams

Oracle Architecture

(See Section III.1 for detailed data flow diagram)

Coherence Trajectory (Terra 2022)



Key events:

- **Feb 2022:** C_Σ ≈ 0.72 (functional but β_c showing strain)
- **Mar 2022:** C_Σ ≈ 0.64 (enters warning zone)
- **Apr 2022:** C_Σ ≈ 0.42 (ALERT) ← **TSC would flag here**

- **May 9, 2022:** Depeg and collapse

Note: Projected scores; validation notebook will verify.

Appendix C — Reference Schema (Ethereum mainnet)

```yaml
version: "1.0.0"
chain: "ethereum-mainnet"
checkpoint_frequency: "daily"
window_size: 7200  # blocks (~1 day)

alpha_requirements:
  - id: block_time_target
    type: range_two_sided
    claim: "12 s block time"
    target: {L: 11.0, U: 13.0, unit: "seconds"}
    datasource: "rpc://<eth-rpc>"  # placeholder
    query: "avg(diff(block.timestamp)) over window"
    weight: 1.0
    tolerance: 0.15

  - id: finality_gasper
    type: range_le
    claim: "≤2 epochs to finality"
    target: {T: 900, unit: "seconds"}
    datasource: "beacon://<consensus-client>"  # placeholder
    query: "time_to_finality_p95"
    weight: 1.0
    tolerance: 300

  - id: validator_decentralization
    type: range_le
    claim: "No entity >25% stake"
    target: {T: 0.25, unit: "fraction"}
    datasource: "beacon://<consensus-client>"  # placeholder
    query: "max(entity_stake_share)"
    weight: 2.0
    tolerance: 0.05

  - id: erc20_transfer_safety
    type: invariant
    claim: "ERC20 transfers cannot reenter"
    target: {holds: true}
    datasource: "analyzer://mythril"
    query: "check_invariant('no_reentrancy','transfer')"
    weight: 2.0

beta_sources:
  rpc_endpoint: "https://<eth-rpc>"  # placeholder
  beacon_endpoint: "https://<beacon-api>"  # placeholder
  gas_market:
    - source: "etherscan/gastracker"
```

```yaml
      metrics: ["base_fee","priority_fee_p50","priority_fee_p95"]
  token_distribution:
    - source: "etherscan/tokenholders"
      metrics: ["top10_concentration","gini_coefficient"]

gamma_features:
  - id: tx_type_distribution
    type: distribution
    baseline: "s3://baselines/eth_txmix.parquet"
    current: "query: tx_type_histogram over window"
    distance_metric: "jensen_shannon"
    datasource: "rpc://<eth-rpc>"  # placeholder

  - id: gas_usage_stability
    type: process
    metric: "coefficient_of_variation(base_fee)"
    threshold: {le: 0.5}
    datasource: "etherscan/gastracker"

  - id: validator_participation
    type: range_ge
    target: {T: 0.95, unit: "fraction"}
    datasource: "beacon://<consensus-client>"  # placeholder
    query: "active_validators / total_validators"

scoring_config:
  primitives:
    range_two_sided: "1 - max(0, max(x-U, L-x))/τ"
    range_le:        "1 - max(0, x-T)/τ"
    range_ge:        "1 - max(0, T-x)/τ"
    invariant:       "1 if holds else 0"
    distribution:    "1 - JS_divergence(current, baseline)"
    process:         "1 - distortion/τ"
  aggregation:
    axis_scores:     "geometric_mean"
    global:          "geometric_mean(alpha_c, beta_c, gamma_c)"
  thresholds:
    production: 0.80
    warning: 0.60
    critical: 0.40

provenance:
  method_version: "tsc-oracle-v1.0.0"
  schema_version: "1.0.0"
  implementation: "github.com/tsc-blockchain/oracle@<commit>"  # placeholder
  frozen_inputs_hash: "sha256:<blob>"  # placeholder
  container_image: "ghcr.io/tsc/oracle@sha256:<digest>"  # placeholder
  licenses:
    - id: "etherscan_terms_v2025-01"
    - id: "beacon_api_terms_v2025-02"
```

References & Availability

- Triadic Self-Coherence (TSC) framework (2024). Mathematical foundations and axioms.
- Coherent Blockchain (2025). Protocol direction and leverage-based gating.
- Incident reports: Terra/Luna post-mortems, The DAO reentrancy analyses, archival Bitcoin studies.
- Earth Mover's Distance: Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The Earth Mover's Distance as a metric for image retrieval. *International Journal of Computer Vision*.
- Edit distance: Wagner, R. A., & Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the ACM*.

A full bibliography with DOIs/URLs will be included in the arXiv submission.

ArXiv Submission Checklist

- **Categories:** `cs.CR` (Cryptography and Security), `cs.DC` (Distributed Computing)
- **Keywords:** blockchain measurement, coherence, DeFi risk, reproducibility, zero-knowledge
- **Source:** LaTeX or PDF from Markdown
- **Include:** `.bib` with 5-8 core references (TSC framework, Terra/DAO post-mortems, EMD/edit-distance citations)
- **Artifacts:** Link to validation notebooks repo when ready
- **Diagrams:** Replace ASCII diagrams with vector PDFs if possible
- **Math rendering:** Uses Unicode symbols for GitHub; convert to LaTeX for arXiv
- **Placeholders:** Clearly marked in schema (e.g., `<eth-rpc>`, `<commit>`, `<blob>`, `<digest>`)

Conclusion

Phase-1 specification provides a complete framework for measuring blockchain coherence. The validation notebooks (Month 1-2) will prove whether this framework works as designed. If validation succeeds, we proceed to build the oracle infrastructure. If validation reveals issues, we iterate on the methodology with clear data about what needs adjustment.

This honest, empirical approach is how engineering works: specify, validate, iterate, deploy.

Next artifact to ship: Three frozen-input notebooks proving TSC framework can retroactively detect coherence collapse before financial loss:

- `terra_2022-04-15.ipynb` — target $C_\Sigma \approx 0.27 \pm 0.05$
- `dao_2016-06-10.ipynb` — target $C_\Sigma \approx 0.56 \pm 0.05$
- `mtgox_2014-02-10.ipynb` — target $C_\Sigma \approx 0.31 \pm 0.06$

Each outputs:

- `attestation.json` (scores, CI, gates, cause map, B, η)
- `provenance.json` (input hashes, method versions, seeds, container digest, licenses)

End of v1.0.0