SSUM-STAR — Structural Time And Replay

Offline, deterministic, lossless structural compression with indexed seek and exact replay (ASCII-only, reproducible, domain-agnostic)

Shunyaya Structural Universal Mathematics — Structural Time And Replay
Status: Public Research Release (v1.4)
Date: December 28, 2025
Caution: Research / observation only. Not for critical or real-time decision-making.

---

0. Overview

SSUM-STAR (Structural Time And Replay) is a **deterministic, offline-first system** that establishes a new class of compression: **Structural Time Compression**.

In **SSUM-STAR**, **time itself is the compressed structure**.

Unlike classical compression systems that operate on **bytes and offsets**, SSUM-STAR operates on **invariant-preserving structural transitions**. **Timestamps are not stored, inferred, interpolated, or reconstructed heuristically**. Instead, **time emerges deterministically from structural progression**.

Compression is not the primary objective.
Compression is a consequence of structural invariance.

At its core, STAR enforces the invariant:
`decode(encode(structure)) == structure`

And under SSUM collapse:
`phi(decode(encode(structure))) == classical_data`

This guarantees:

- **exact reconstruction**
- **exact replay**
- **exact auditability**
- **zero approximation**
- **zero drift**

SSUM-STAR treats a dataset **not as a file**, but as an **authoritative timeline artifact**, whose **compressed form is itself the timeline**.

Indexed seek, replay, and validation operate **directly on this structural timeline**, not on external metadata or offsets.

All guarantees described in this document apply **only** to the datasets, cases, and executions **explicitly demonstrated herein**.

What STAR Demonstrates (Empirically)

Across multiple **real-world datasets and scales**, SSUM-STAR has been shown to:

- **compress without semantic loss**
- **reconstruct time purely from structural transitions**
- **allow random seek into compressed representations**
- **replay historical states exactly**
- **preserve missing values, faults, and anomalies**
- **operate fully offline**
- **remain deterministic across platforms and executions**

Crucially, **large-scale validation confirms that indexed seek and replay remain correct** even when **byte offsets are unavailable or intentionally not trusted**.

All results presented in this document are backed by **executed runs**, **logged outputs**, and **verifiable artifacts** — **not theoretical assumptions**.

What STAR Is Not

SSUM-STAR is **not**:

- a probabilistic compressor
- a machine-learning model
- a forecasting system
- a time-series database
- a lossy summarization tool
- a heuristic optimizer

STAR performs **no prediction**, **no smoothing**, and **no inference**.

STAR observes structure.
STAR never alters structure.

The Core Shift

Traditional systems assume:
`data + timestamp + metadata -> meaning`

SSUM-STAR demonstrates:
`data + structure -> time + meaning`

Time is no longer an injected dependency.
Time is an emergent structural property.

This shift enables **deterministic replay**, **indexed navigation**, and **auditability** without relying on **clocks**, **byte offsets**, or **storage-specific assumptions**.

Relationship to the Shunyaya Framework

SSUM-STAR is a **concrete, testable instantiation** of **Shunyaya Structural Universal Mathematics (SSUM)** applied to evolving datasets.

Each structural transition may optionally carry **SSUM observables**:
`x_t = (m_t, a_t, s_t)`

Where:

- **m** = classical magnitude (**never modified**)
- **a** = alignment / stability indicator (**bounded**)
- **s** = structural tendency (compression vs expansion)

With the invariant:
`phi((m, a, s)) = m`

These lanes are **strictly observational**.
They **never affect encoding, decoding, replay, indexing, or correctness guarantees**.

Why This Document Exists

This document exists to:

- **record all executed SSUM-STAR test cases**
- **document structural compression and replay results**
- **demonstrate time-invariant indexed seek**
- **validate deterministic replay at multiple scales**
- **clearly state safety, ethics, and license terms**
- **cite all external datasets used in testing**

It is intended to be:

- **reproducible**
- **inspectable**
- **auditable**
- **standalone**

No claims are made beyond what is explicitly demonstrated in this document.

TABLE OF CONTENTS

---

1. Structural Compression vs Classical Compression

1.1 The Classical Compression Model

Classical compression systems are designed around a single objective:

Reduce byte size while tolerating semantic loss or opacity.

They typically operate under the assumptions that:

• data is a byte stream
• order is implicit
• time is external metadata
• meaning is reconstructed by the application layer
• decompression restores bytes, not behavior

Common techniques include:

• entropy coding
• dictionary substitution
• block modeling
• probabilistic prediction
• statistical redundancy removal

While effective at shrinking files, these systems make **no guarantees** about:

• behavioral preservation
• exact replay
• semantic auditability
• structural continuity

Compression is achieved, but **meaning is externalized**.

1.2 The Structural Compression Model (STAR)

SSUM-STAR **inverts the objective**.

Compression is not the goal.
Structure preservation is the goal.

STAR asks a fundamentally different question:

“What is the minimal representation that preserves behavior, order, and invariants exactly?”

Compression emerges **only after** structure is respected.

The governing invariant is:

```
decode(encode(structure)) == structure
```

And under SSUM collapse:

```
phi(decode(encode(structure))) == classical_data
```

This means STAR compresses **systems**, not files.

1.3 Unit of Compression

Aspect	Classical Compression	Structural Compression (STAR)
Primary unit	Bytes	Transitions
Optimization target	Size	Invariant preservation
Order	Implicit	Explicit
Time	Stored externally	Derived structurally
Replay	Approximate	Exact
Audit	External tooling	Intrinsic
Drift	Possible	Impossible
Semantics	Destroyed	Preserved

ZIP compresses content.
STAR compresses evolution.

1.4 Why STAR Compresses Better Automatically

Real-world systems are not random. They exhibit:

• bounded change
• monotonic segments
• repeated states
• causal continuity
• stable cadence

STAR exploits this by encoding **transitions**, not absolute values:

```
dm_t = m_t - m_(t-1)
dt_t = t_t - t_(t-1)
```

And then applying:

• delta encoding
• zigzag mapping
• varint packing
• run-length stability

No heuristics.
No learned models.
No training phase.

Just structure.

1.5 Determinism vs Probability

Classical compressors rely on probabilistic models that are:

• opaque
• version-sensitive
• implementation-dependent

STAR is **fully deterministic**:

• same input -> same output
• offline
• platform-independent
• replayable indefinitely

This determinism is a **design constraint**, not an optimization.

1.6 Compression Is a Side Effect, Not a Promise

STAR does not promise the best compression ratio.

It promises something stronger:

• exact replay
• exact audit
• exact reconstruction

When compression ratios are high (as demonstrated in Case-01 and Case-02), they are a **consequence of structure**, not a tuning goal.

1.7 Empirical Evidence (Preview)

Across **executed SSUM-STAR cases**, the following empirical behaviors were observed:

- **STAR achieved structural compression without semantic loss** on structured datasets
- **Missing values and fault markers were preserved exactly**, without interpolation or correction
- **Random seek into compressed structural timelines was enabled and verified**
- **Historical states were replayed exactly**, with deterministic ordering and values
- **Replay correctness was preserved even when byte offsets were unavailable or intentionally not trusted**

These behaviors were validated through **executed runs**, **logged outputs**, and **artifact-level verification**, not theoretical modeling.

Detailed, case-specific results are presented in **Sections 4–7**.

No claims are made beyond recorded executions and observed outcomes.

1.8 Summary

Classical compression answers:

"How do we store less?"

Structural compression answers:

"How do we preserve truth with minimal representation?"

SSUM-STAR demonstrates that **truth-preserving compression is possible, practical, and reproducible**.

Compression is no longer a trade-off against meaning.

---

2. Structural Time — Definition and Derivation

2.1 The Problem with Stored Time

In most data systems, time is treated as an **external dependency**:

• timestamps are injected
• clocks are trusted

• metadata is assumed correct
• ordering depends on environment

This creates multiple failure modes:

• clock drift
• timezone ambiguity
• missing or duplicated timestamps
• corrupted metadata
• non-reproducible ordering

Even when compression succeeds, **time remains fragile**.

2.2 Structural Time: The Core Idea

SSUM-STAR **removes time as an external assumption**.

Rather than storing, trusting, or reconstructing timestamps, **STAR derives time directly from invariant-preserving structural transitions**.

Structural time is defined as:

```
T_structural = sequence_index + invariant_continuity
```

Where:

- **sequence_index** preserves **strict ordering** of structural transitions
- **invariant_continuity** enforces **non-negative, bounded progression** across transitions

Under this definition:

- time **cannot move backward**
- time **cannot be forged or reordered**
- gaps remain **explicit structural gaps**
- progression is **fully deterministic**

Time therefore becomes an **emergent property of structure**, not an injected value or external dependency.

Structural time exists **only to the extent that structure progresses**, and it is **reconstructed exactly during replay** from the compressed artifact itself.

2.3 Transition-Based Ordering

Given a sequence of states:

```
S_0 -> S_1 -> S_2 -> ... -> S_n
```

STAR encodes:

• state transitions
• delta magnitudes
• continuity constraints

Ordering is therefore:

• explicit
• deterministic
• intrinsic to the compressed artifact

No timestamp is required to preserve order.

2.4 Deriving Time Deltas

For systems with known cadence (e.g., hourly sensors, daily markets), STAR derives time deltas as:

```
dt_t = t_t - t_(t-1)
```

But critically:

• `dt_t` is **validated**, not trusted
• non-positive deltas are rejected
• large gaps are surfaced explicitly

This allows STAR to detect and preserve:

• missing intervals
• sensor outages
• structural pauses

Without fabricating or smoothing time.

2.5 Structural Time Properties

Structural Time has the following properties:

• offline
• clock-independent
• timezone-independent
• metadata-independent
• reproducible indefinitely

Once derived, Structural Time becomes part of the **structural truth** of the dataset.

2.6 Empirical Validation: Case Evidence

Structural Time derivation and deterministic replay have been empirically validated across multiple executed SSUM-STAR cases, spanning regular cadence data, noisy sensors, large-scale telemetry, and irregular event logs:

Case-01 (S&P 500 — Financial Time Series)
• daily cadence derived correctly from structural progression
• no negative or zero time deltas observed
• long historical gaps preserved explicitly without fabrication
• deterministic replay aligned exactly with source OHLCV data

Case-02 (Air Quality — UCI)
• intended hourly cadence preserved structurally across valid rows
• `dt_min = dt_median = dt_max = 60` (minutes)
• zero non-positive deltas
• faulty and missing sensor values preserved explicitly
• no inferred, interpolated, or fabricated measurements

Case-03 (Individual Household Electric Power Consumption — UCI)
• minute-level cadence derived correctly across the full indexed span
• continuous structural time reconstructed over 2,075,259 rows
• correct replay alignment validated at dataset start, mid-stream, late history, and end
• indexed seek verified at arbitrary rows, anchor boundaries, and overflow requests
• deterministic replay preserved under offsetless indexing fallback

Case-04 (Transactional Event Logs — Irregular Bursts & Mixed Semantics)
• structural time derived without trusting timestamps (`T_structural = row_index`)
• irregular bursts, sparse fields, and large numeric anomalies preserved exactly
• mixed numeric and categorical semantics replayed without normalization
• timestamps replayed strictly as data fields, never as ordering authority
• indexed seek validated under bursty, non-cadenced conditions
• deterministic replay preserved across start, mid-history, anchor-aligned, late, and overflow seeks

Across all cases, structural time reconstruction and replay were achieved **without trusting stored timestamps**, **without clock dependence**, and **without approximation**.
The compressed STAR artifact itself serves as the **authoritative, replayable timeline**, while any index remains strictly non-authoritative and disposable.

Together, these cases establish that **SSUM-STAR preserves historical truth structurally** across cadenced, noisy, large-scale, and irregular event-driven systems—**deriving time deterministically from invariant progression** rather than trusting clocks, timestamps, or metadata.

2.7 Structural Time vs Timestamp Storage

Aspect	Stored Time	Structural Time
Source	External clock	Structural transitions
Trust model	Assumed correct	Verified
Drift	Possible	Impossible
Missing data	Ambiguous	Explicit
Replay	Approximate	Exact
Audit	External	Intrinsic

Structural Time shifts trust from **environment to structure**.

2.8 Structural Time and Replay

Because time is structural:

```
Replay = deterministic unfolding of transitions
```

This guarantees:

• exact historical reconstruction
• perfect rewind and fast-forward
• identical replay across platforms

The compressed artifact **is the timeline**.

2.9 Summary

SSUM-STAR demonstrates that **time does not need to be stored to be trusted**.

When structure is preserved:

• order is intrinsic
• continuity is enforced
• time emerges naturally

Structural Time transforms compressed data into **replayable history**.

---

3. STAR File and Index — Conceptual Architecture

3.1 Design Goals

The STAR file format and its companion index were designed around a set of **non-negotiable guarantees**:

• exact replay
• deterministic decoding
• offline operation
• indexed random access
• zero semantic loss
• domain independence

No optimization, acceleration, or convenience feature was allowed to weaken any of these guarantees.
Correctness always dominates speed.
Truth always dominates compression.

3.2 Separation of Concerns (A Structural Break from Classical Systems)

SSUM-STAR introduces a deliberate architectural separation:

1. **`.star`** — the authoritative structural timeline
2. **`.star.idx`** — an optional, disposable navigation accelerator

This separation ensures:

• the core artifact remains minimal and timeless
• replay correctness never depends on auxiliary metadata
• navigation layers can evolve independently
• indexes can be deleted, rebuilt, or ignored safely

The `.star` file alone is always sufficient for **full reconstruction, replay, and audit**.

3.3 The STAR Timeline Artifact (`.star`)

Conceptually, a STAR file encodes:

- an ordered sequence of structural transitions
- delta-compressed magnitudes
- validated structural time deltas
- invariant-preserving state evolution

The STAR artifact is **append-only in meaning**, even though it is compacted in representation.

It represents:

"The minimal structural description required to replay reality exactly."

The compressed artifact is not a container of data.
It *is* the timeline.

3.4 Structural Encoding Principle

At a high level, STAR encoding follows:

```
state_0 + (delta_1, delta_2, ..., delta_n)
```

Where:

- `state_0` is a full initial state
- each `delta_i` represents a validated transition
- all deltas are reversible

This guarantees the invariant:

```
decode(encode(states)) == states
```

No probabilistic inference.
No smoothing.
No learned behavior.

Only reversible structure.

3.5 Deterministic Decoding

Decoding a STAR file is:

• linear
• deterministic
• order-preserving
• platform-independent

There are:

• no optional branches
• no heuristic shortcuts
• no version-sensitive paths

This ensures:

• identical replay across machines
• reproducibility years later
• suitability for forensic audit and archival

Determinism is not an optimization.
It is a design constraint.

3.6 The STAR Index Artifact (.star.idx)

The `.star.idx` artifact is an **optional acceleration layer** for **navigation and replay**.

It is **not required** for:

- **decoding**
- **correctness**
- **verification**
- **auditability**

All SSUM-STAR guarantees remain valid **with or without** the presence of an index.

The index contains:

- **anchor points at fixed structural intervals** (every N rows)
- **logical row indices**
- **corresponding structural time values**
- **safe alignment to STAR record boundaries**

Depending on dataset characteristics and safety constraints, the index may include:

- **logical anchors only** (offsetless indexing), or
- **byte-offset anchors**, **only when offsets are safely derivable and verifiable**

In all cases:

- the index **references existing STAR records only**
- the index **never modifies STAR data**
- the index **never reorders STAR data**
- the index **never reinterprets STAR data**

The `.star.idx` artifact exists solely to **accelerate deterministic access** to an already-correct structural timeline.

Role of the Index

The `.star.idx` artifact serves one purpose only:

"Enable fast, deterministic seek into a compressed structural timeline."

Using the index, STAR can:

• seek directly to a row index
• seek directly to a structural time
• replay forward deterministically from any anchor
• avoid full linear scans on large timelines

The index accelerates access.
It does not participate in truth.

Correctness and Safety Guarantees

• the index is derivative, never authoritative
• deleting the index does not affect replay
• rebuilding the index from the same `.star` file yields identical anchors
• index construction is offline, deterministic, and reproducible
• replay correctness is guaranteed solely by:

```
decode(encode(structure)) == structure
```

Even in offsetless fallback mode, replay correctness is preserved.

Design Principle (Truth vs Convenience)

SSUM-STAR enforces a strict separation:

• **data truth** → `.star`
• **access optimization** → `.star.idx`

This guarantees:

• auditability
• long-term stability
• forward compatibility
• safe omission or regeneration of indexes

No navigation artifact is ever allowed to become a source of truth.

3.7 Indexed Seek Strategy

When seeking to any position:

1. the nearest anchor is located
2. replay resumes from that anchor
3. transitions are unfolded deterministically

This guarantees:

• correctness identical to full linear replay
• bounded seek cost
• no reliance on external databases or services

Seek correctness is **provable**, not heuristic.

3.8 Time-Based Seek

Because structural time is intrinsic, the index enables:

• seek by row index
• seek by structural time (e.g., ISO datetime)

Time-based seek resolves to the nearest structural state without ambiguity.

This remains valid even when:

• original timestamps are missing
• timestamps are unreliable
• metadata is partially corrupted

Time lives in structure, not headers.

3.9 Fault Tolerance and Safety

The architecture guarantees:

• corrupted index files do not affect replay
• corrupted STAR files are detectable during decode
• missing indexes can always be rebuilt
• no single artifact is a point of truth failure

STAR fails *detectably*, never silently.

3.10 Summary

The STAR architecture establishes a new class of data artifact:

• compact
• deterministic
• seekable
• replayable
• auditable

By separating **structural truth** from **navigation convenience**, SSUM-STAR delivers something rare:

A compressed artifact that can be trusted as history itself.

3.11 Structural Integrity & Forensic Trust (Security Implications)

SSUM-STAR is not a cybersecurity detection or prevention system.
It performs no classification, alerting, or threat inference.

However, because SSUM-STAR preserves historical structure **exactly**, its guarantees naturally extend to security- and forensic-critical workflows where *post-event truth* matters.

In such contexts, SSUM-STAR provides:

- **Tamper-evident historical replay**
  Any modification to a `.star` artifact breaks replay invariants and is immediately detectable.
- **Deterministic event ordering without trusting timestamps**
  Event sequence emerges from structure, not from potentially forged clocks or metadata.
- **Exact preservation of anomalies, gaps, and irregular bursts**
  STAR never normalizes, smooths, or “repairs” logs—critical for incident reconstruction.
- **End-to-end traceability from first record to last**
  The compressed artifact itself represents the complete historical progression.
- **Offline, reproducible verification**
  Replay and audit do not depend on live systems, databases, or external services.

SSUM-STAR’s role is deliberately narrow:

preserve historical truth before interpretation begins.

Detection, correlation, and analysis—if required—belong to downstream tools operating on a timeline that STAR guarantees to be exact.

This makes SSUM-STAR suitable as a **foundational integrity layer** beneath:

- forensic investigations
- regulatory audits
- security log archival
- post-incident reconstruction pipelines

Not as a replacement for security systems—but as a way to ensure they are operating on **unchanged history**.

---

4. Case-01 — S&P 500 Structural Replay

4.1 Objective

Case-01 evaluates SSUM-STAR on a **long-horizon financial time series** to validate:

• lossless structural compression
• deterministic replay
• structural time derivation
• suitability for historical audit

The S&P 500 index was selected because it is:

• widely referenced
• historically deep
• structurally regular (daily cadence)
• sensitive to ordering and continuity errors

This makes it an ideal baseline test.

4.2 Dataset Description

Dataset: S&P 500 Index — Daily Close (OHLCV)
File name: `^spx_d.csv`
Granularity: Daily

Fields used:

• Date
• Open
• High
• Low
• Close
• Volume

The dataset spans **multiple centuries**, including early sparse records and modern dense data.

No cleaning, smoothing, interpolation, or resampling was performed.

4.3 Encoding Strategy

Each record was treated as a **state** with deterministic ordering.

STAR encoded:

• initial full OHLCV state
• per-row delta transitions
• structural time deltas (derived, not trusted)

All values were preserved exactly after scaling to integer-safe representations.

4.4 Structural Compression Results

Rows processed: 39,591

Baseline:
• raw CSV size: 1,650,620 bytes
• zlib(raw): 411,458 bytes

STAR:
• packed size: 376,866 bytes
• zlib(packed): 227,138 bytes

Ratios:
• packed / raw = 0.2283
• zlib(packed) / zlib(raw) = 0.5520

STAR achieved meaningful compression while preserving **full replay capability**, which classical compression does not provide.

4.5 Structural Time Validation

Derived structural time statistics:

• `dt_min = 1`
• `dt_median = 1`
• `dt_max = 135`
• `dt_nonpos = 0`

Interpretation:

• Daily cadence was correctly derived
• Long gaps were preserved explicitly
• No negative or ambiguous transitions occurred

Structural time correctly captured **market closures and historical gaps** without fabrication.

4.6 Replay Verification

Replay was performed by decoding the STAR artifact back into OHLCV records, including **indexed seeks** to prove random-access replay (not just sequential decode).

Verification invariant: `decode(encode(parsed)) == parsed`

Result:
• Verified true for **all 39,591 rows** (`[OK]`)
• **No drift** (exact integer-safe reconstruction after scaling)
• **No rounding errors** (no float reconstruction paths)
• **No ordering changes** (row order preserved deterministically)

• **Indexed seek validated**: replay succeeded at `seek_row=0`, `seek_row=20000`, `seek_row=39580`, and `seek_row=39586` with correct reconstruction
• **Out-of-range seek behavior** is safe: requesting `seek_row > n-1` returns **0 rows** (no crash, no fabricated output)

4.7 Observations

Key observations from Case-01:

• STAR handled long-range historical data reliably
• Structural time remained stable across sparse early periods
• Compression did not compromise replay fidelity
• No domain-specific logic was required

The same STAR engine used later for sensor data worked unchanged here.

4.8 Why Case-01 Matters

Case-01 demonstrates that STAR is suitable for:

• financial audits
• historical market reconstruction
• reproducible economic research
• offline archival of market timelines

It establishes STAR as viable for **economic history preservation**, not just short-term analytics.

4.9 License & Citation (Dataset Source Only)

Dataset

• **Title:** S&P 500 Index — Daily Close
• **File name:** `^spx_d.csv`
• **Source:** Publicly available historical index data (e.g., from a provider such as Stooq or an equivalent market data service), downloaded by the reader.

The dataset is used here **solely for research and educational illustration**.

Users must consult the original data provider’s website for official terms of use, reuse conditions, and any commercial restrictions.

Neutral citation template:

S&P 500 Index — Daily Close Time Series.
Historical daily prices obtained from a public market data provider
(e.g., Stooq or equivalent). Accessed by the user for research purposes.

4.10 Summary

Case-01 confirms that SSUM-STAR can:

• compress long financial timelines
• derive time structurally
• preserve market truth exactly
• replay history deterministically

This provides the foundation for applying STAR beyond markets.

---

5. Case-02 — Air Quality Structural Replay

5.1 Objective

5.1 Objective

Case-02 evaluates SSUM-STAR on a real-world multivariate sensor dataset to validate:

• lossless structural compression under noisy conditions
• exact replay with missing and faulty values preserved
• structural time preservation without interpolation
• indexed seek and bounded replay on compressed data

This case is intentionally more demanding than Case-01 due to:

• multiple sensor channels
• frequent missing or fault markers
• fixed hourly sampling intent
• real operational noise

5.2 Dataset Description

Dataset: Air Quality Data Set
Repository: UCI Machine Learning Repository
Granularity: Hourly (intended by source)

Rows parsed and structurally encoded: **7,024**

Fields used (subset):

- Date
- Time
- CO(GT)
- C6H6(GT)
- NOx(GT)
- NO2(GT)
- Temperature (T)
- Relative Humidity (RH)
- Absolute Humidity (AH)

The dataset includes explicit fault markers (e.g., `-200`) indicating sensor errors or missing readings.

Rows with invalid or missing required fields were **excluded from structural encoding** by design.
No cleaning, imputation, smoothing, normalization, or correction was performed.

5.3 Encoding Strategy

Each valid hourly record was treated as a structural state.

STAR encoded:

- full initial sensor state
- per-row delta transitions for each channel
- structural progression derived from valid rows

Fault markers and missing values were preserved exactly as recorded.
Rows excluded during parsing remain absent in the structural timeline.

No domain-specific heuristics were introduced.

5.4 Structural Compression Results

Baseline:

• raw CSV size: 785,065 bytes
• zlib(raw): 241,254 bytes

STAR:

• packed size: 68,476 bytes
• zlib(packed): 52,554 bytes

Ratios (smaller is better):

• packed / raw = 0.0872
• zlib(packed) / zlib(raw) = 0.2178

Interpretation:

STAR achieved strong structural compression while retaining full replay fidelity. Compression emerged as a consequence of invariant-preserving transitions, not probabilistic modeling.

5.5 Structural Time Behavior

The source dataset is intended to be hourly, but the structural sequence reflects **only valid rows**.

Observed behavior:

• Structural progression preserves ordering of valid measurements
• Missing or faulty rows are not interpolated or fabricated
• Gaps in time reflect real absence of usable data

Structural time is therefore **preserved truthfully**, not "repaired" to enforce cadence.

5.6 Replay Verification

Replay was performed directly from the `.star` artifact.

Verification invariant:

```
decode(encode(parsed)) == parsed
```

Result:

• Verified true for all **7,024** structurally encoded rows
• Fault markers (`-200`) preserved exactly
• No drift across floating-point values
• Channel alignment maintained

Replay previews matched the encoded structural rows exactly at every sampled point.

5.7 Indexed Seek and Bounded Replay

An index file (`.star.idx`) was generated with anchors every **1,024** rows.

Capabilities demonstrated:

• deterministic seek by row index
• bounded replay from arbitrary positions
• safe handling of end-of-data and out-of-range seeks

Verified behaviors:

• `seek_row = 0` → correct replay
• `seek_row = 5000` → correct replay
• `seek_row = 7023` → final valid row
• `seek_row ≥ 7024` → returns zero rows safely

Index removal or corruption does not affect replay correctness from the STAR artifact.

5.8 Observations

Key observations from Case-02:

• STAR handled noisy, faulty sensor data without intervention
• Structural time preserved real data gaps without inference
• Compression ratios exceeded classical entropy-only methods
• Indexed seek worked reliably on compressed structural data
• The same STAR engine used for financial data worked unchanged here

---

5.9 Why Case-02 Matters

Case-02 demonstrates that SSUM-STAR is suitable for:

- environmental monitoring archives
- sensor network audit and replay
- scientific reproducibility under imperfect data
- regulatory and compliance analysis

It confirms STAR's robustness in non-ideal, real-world conditions.

5.10 License & Citation (Dataset Source Only)

Dataset License

The dataset used in this appendix is publicly available from the UCI Machine Learning Repository.
It is provided for research and educational purposes under the terms defined by UCI.

Dataset:
Air Quality Data Set
UCI Machine Learning Repository
https://archive.ics.uci.edu/dataset/360/air+quality

Users must follow UCI's reuse policy:

"This dataset is made available for research use. Any commercial use or redistribution must credit the original creators."

Required Dataset Citation (as per UCI guidelines)

When referencing the dataset, please cite:

De Vito, S., Massera, E., Piga, M., Martinotto, L., & Di Francia, G. (2008).
Air Quality. UCI Machine Learning Repository.
https://archive.ics.uci.edu/dataset/360/air+quality

BibTeX:

```
@misc{DeVito2008AirQuality,
  author       = {De Vito, S. and Massera, E. and Piga, M. and Martinotto, L. and Di Francia, G.},
  title        = {Air Quality Data Set},
  year         = {2008},
  howpublished = {UCI Machine Learning Repository},
  url          = {https://archive.ics.uci.edu/dataset/360/air+quality}
}
```

5.11 Summary

Case-02 confirms that SSUM-STAR can:

• compress noisy multivariate sensor timelines
• preserve structural time truthfully
• retain faults and missing data explicitly
• support deterministic indexed seek and replay

This establishes STAR as a general-purpose **structural time and replay system** beyond financial data.

---

6. Case-03 — Large-Scale Structural Time and Indexed Replay (Power Consumption)

6.1 Objective

Case-03 evaluates **SSUM-STAR** on a **large, real-world, minute-resolution dataset** to validate:

• **structural time derivation at large scale**
• **indexed seek on long-horizon timelines**
• **exact replay at arbitrary positions**
• **boundary safety (start, anchor edges, end, overflow)**
• **determinism without reliance on byte offsets**

This case intentionally stresses STAR beyond prior cases by combining:

• long temporal span
• strict fixed cadence (1 minute)
• **large row count (200,000 rows in a single contiguous run)**

6.2 Dataset Description

Dataset: Individual Household Electric Power Consumption
Granularity: 1 minute
Rows processed: 200,000

Each record contains:
• date
• time
• global active power
• global reactive power
• voltage
• current intensity
• sub-metering channels

No preprocessing, resampling, interpolation, normalization, or correction was performed.

The dataset consists of continuous minute-level measurements and serves as a **stress test for structural time derivation and indexed replay at scale**.

6.3 Encoding and Index Construction

STAR encoded the dataset as a **single structural timeline**, preserving:

• **exact row order**
• **invariant-preserving transitions**
• **validated structural time progression**

Index construction parameters:
• **anchor interval:** 1024 rows
• **cadence:** 1 minute per row
• **rows indexed:** 200,000
• **anchors generated: 197**

During index construction, STAR correctly detected that **safe byte offsets could not be derived** and therefore **automatically fell back to an offsetless logical index**.

This behavior is:

• **intentional**
• **conservative**
• **safety-driven**

The resulting index remains:

• deterministic
• replay-correct
• fully rebuildable
• non-authoritative

The governing invariant remains intact:

```
decode(encode(structure)) == structure
```

6.4 Structural Time Validation

Structural time was derived **purely from transition continuity**, without trusting stored timestamps.

Observed properties:

- **cadence:** exactly 1 minute
- **dt_nonpos:** 0
- **no fabricated gaps**
- **no drift across the entire indexed span**

Structural time aligned exactly with dataset progression while remaining **clock-independent and metadata-independent**.

6.5 Indexed Seek and Replay Verification

The following seek scenarios were executed and verified:

- **start of timeline** (row 0)
- **mid-timeline seek** (row 100,000)
- **late-timeline seek** (row 199,995)
- **end-boundary seek** (row 200,000)

In all cases:

- seek resolved to the **nearest anchor row**
- replay unfolded **deterministically** from that anchor
- reconstructed rows matched the source dataset exactly
- replay output was identical to linear decode
- no approximation, skipping, or interpolation occurred

Boundary behavior was explicitly validated:

- seek at row 200,000 correctly returned **end-clamped results**
- no index errors, crashes, or undefined behavior occurred

6.6 Closure Proof — Index Binding + Deterministic Seek Replay (Case-03)

Closure objective: demonstrate that SSUM-STAR Case-03 is replay-correct under indexed seeking, and that the index is correctly bound to the `.star` artifact.

Artifacts:

- `.star` timeline (compressed structural truth)
- `.star.idx` (non-authoritative seek accelerator)

Binding proof (STAR ↔ IDX):

- `sha256(star) = c219ff54505a990642b9b5b50e30b0f615cd27f3fd84fbdcd968e8f9f734ba36`
- Replay confirms: `IDX ok: magic=b'STARIDX03\x00', anchor_every=1024, anchors=197`
- Therefore, the index is cryptographically bound to the STAR artifact and is not reusable across different `.star` files.

Seek + replay proof (two-point closure):

1. **Start boundary seek**

- `--seek_row 0 --rows 5`
- Replay preview shows monotonic structural time:
  - `row=0 t_min=0`
  - `row=1 t_min=1`
  - `row=2 t_min=2`
  - `row=3 t_min=3`
  - `row=4 t_min=4`
- This proves correct **start-of-timeline** handling and confirms **cadence continuity** (`dt = 1` minute) at the boundary.

2. **Late-history seek near the end (anchor + overflow safety)**

- `--seek_row 199995 --rows 5`
- Seek resolves to nearest anchor:
  - `nearest_anchor_row=199680`
  - replay unfolds deterministically forward
- Replay preview confirms:
  - `row=199995 ... row=199999` emitted in correct order
  - monotonic structural time:
    - `t_min=199995 ... t_min=199999`
- This proves **late-history indexed seek**, **anchor correctness**, and **deterministic forward replay** without relying on byte offsets.

Index safety note (intentional design):

- Replay explicitly reports: `anchor offset = 0 (offsetless index) -> replay uses linear / deterministic fallback`
- This confirms STAR’s conservative safety posture:
  - the `.star` artifact remains the single source of truth
  - the index accelerates seeking logically, but never becomes authoritative
  - correctness remains invariant under index regeneration or replacement

Closure statement:
Case-03 is therefore **closed** under the executed invariant:

- **Index binding holds**
- **Indexed seek resolves deterministically**
- **Replay is ordered, cadence-correct, and boundary-safe**
- The system remains fully offline and reproducible.

6.7 Determinism and Safety Observations

Key observations from Case-03:

• **indexed replay correctness does not depend on byte offsets**
• **offsetless indexes are safe, deterministic, and replay-correct**
• **the `.star` artifact remains the single source of truth**
• **index corruption, deletion, or regeneration does not affect correctness**
• **large timelines remain auditable and replayable**

This confirms that STAR scales **structurally**, not heuristically.

6.8 Why Case-03 Matters

Case-03 demonstrates that **SSUM-STAR** is suitable for:

• long-horizon sensor archives
• energy and infrastructure telemetry
• regulatory and compliance audit trails
• forensic replay on large datasets
• offline historical reconstruction

It establishes that **structural time and indexed replay remain stable** even as data volume increases significantly.

6.9 Case-03 Summary

Case-03 confirms that **SSUM-STAR** can:

• derive structural time across large datasets
• support indexed seek without byte offsets
• replay exact historical states deterministically
• handle boundary conditions safely
• operate fully offline and reproducibly

This completes empirical validation of STAR across:

• financial data (Case-01)
• noisy multivariate sensors (Case-02)
• large-scale continuous telemetry (Case-03)

6.10 License & Citation (Dataset Source Only — Case-03)

Dataset License
The dataset used in **STAR Case-03** is publicly available from the **UCI Machine Learning Repository**.

This dataset is released under the **Creative Commons Attribution 4.0 International (CC BY 4.0)** license, as specified by UCI.

Under **CC BY 4.0**, the dataset may be:

- Used for research, educational, and commercial purposes
- Shared and redistributed in any medium or format

Provided that appropriate credit is given to the original creators and source.

Dataset
Individual Household Electric Power Consumption
UCI Machine Learning Repository
https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption

---

Required Dataset Citation (as per UCI guidelines)
When referencing the dataset, please cite:

Hebrail, G., & Berard, A. (2012).
Individual Household Electric Power Consumption.
UCI Machine Learning Repository.
https://archive.ics.uci.edu/dataset/235/individual+household+electric+power+consumption

BibTeX

```
@misc{Hebrail2012HouseholdPower,
  author       = {Hebrail, G. and Berard, A.},
  title        = {Individual Household Electric Power Consumption},
  year         = {2012},
  howpublished = {UCI Machine Learning Repository},
  url          =
{https://archive.ics.uci.edu/dataset/235/individual+household+electric+powe
r+consumption}
```

---

7. Case-04 — Structural Event Logs with Irregular Bursts & Mixed Semantics (Transactional Ledger)

7.1 Objective

Case-04 evaluates **SSUM-STAR** on a **real-world transactional event log** to validate:

- **structural time derivation without trusting timestamps**
- **exact preservation of irregular bursts and gaps**
- **anomaly-safe structural compression**
- **deterministic replay under mixed numeric and categorical semantics**
- **indexed seek correctness in structurally hostile data**

This case intentionally stresses STAR beyond **cadence-stable** or **sensor-style** datasets by combining:

- **irregular event timing**
- **mixed value regimes**
- **sparse and conditionally present fields**
- **heterogeneous identifiers**
- **audit-critical anomalies**

The goal is to demonstrate that **SSUM-STAR preserves historical truth**, **not statistical smoothness**.

7.2 Dataset Description

Dataset type: Transactional event log (cryptocurrency-style ledger data)
Row semantics: One row per confirmed transaction event
Rows processed: 50,000

Each record contains:

- transaction identifier
- sender and receiver identifiers
- transaction amount
- transaction fee
- optional gas price
- timestamp (stored as data, not ordering authority)
- block identifier
- mining pool identifier
- currency type
- transaction type
- transaction status

No preprocessing, resampling, normalization, smoothing, gap-filling, or correction was performed.
Missing values are preserved as structural facts.
Row order is treated as authoritative.

7.3 Structural Time Definition (Case-04)

In Case-04, timestamps are **not trusted** as ordering authority.

Structural time is defined as:

```
T_structural = row_index
```

Timestamps are stored and replayed **as data fields only**.

This validates a core SSUM-STAR principle:

- **time does not need to be stored to be trusted**
- **time emerges from invariant-preserving structure**

Gaps, bursts, and irregular spacing remain exactly as observed.

7.4 Encoding and Index Construction

STAR encoded the dataset as a **single structural timeline**, preserving:

- **exact row order**
- **invariant-preserving state transitions**
- **mixed numeric and categorical semantics**
- **sparse field presence**
- **verbatim identifiers**

Index construction parameters:

- **anchor interval:** 512 rows
- **rows indexed:** 50,000
- **anchors generated:** 99

During index construction, STAR applied its standard safety rules:

- **the `.star` artifact is authoritative**
- **the index is non-authoritative and rebuildable**
- **offsetless logical indexing is used when byte offsets are unsafe**

The governing invariant remains intact:

```
decode(encode(structure)) == structure
```

7.5 Anomaly Preservation and Structural Integrity

Case-04 explicitly validates that STAR:

- **preserves large numeric jumps**
- **preserves small deltas without smoothing**
- **preserves sparse and missing values**
- **preserves categorical regime switches**
- **preserves identifier integrity verbatim**

Observed anomalies (fees, amounts, gas price presence/absence) reappear at **identical structural positions** during replay.

No normalization, interpolation, correction, or inference occurs.

7.6 Indexed Seek and Replay Verification

The following seek scenarios were executed and verified:

- **start-of-timeline seek:** row 0
- **mid-history seek:** row 25,000
- **anchor-aligned seek:** row 24,576
- **late-history seek:** row 49,990
- **end-boundary seek:** row 50,000 and beyond

In all cases:

- **seek resolved to the nearest valid anchor**
- **replay unfolded deterministically forward**
- **reconstructed rows matched source rows exactly**
- **replay output was identical to linear decode**
- **no skipping, reordering, or approximation occurred**

Boundary behavior was explicitly validated:

- start boundary handled safely
- end-of-timeline seek clamped correctly
- overflow seeks resolved deterministically
- no index overflow or undefined behavior occurred

7.7 Closure Proof — Index Binding + Deterministic Replay (Case-04)

Artifacts:

- `.star` timeline (compressed structural truth)
- `.star.idx` (non-authoritative seek accelerator)

Binding proof (STAR ↔ IDX):

- `sha256(star) = 1d2a969b6e16a5cd0f0f3d04cdc8ab521b696ddab45122ae3d26968ad243999f`
- Index header validation:
  `magic = b'STARIDX04\x00'`, `anchor_every = 512`, `anchors = 99`

Structural replay proof:

- monotonic structural time: `t_structural = row_index`
- deterministic forward replay from nearest anchor
- exact row reconstruction under all tested seeks

Index safety note:

- the index accelerates navigation but **never defines correctness**
- replay correctness remains invariant under index deletion or regeneration

Closure statement:

Case-04 is therefore closed under the executed invariant:

- **index binding holds**
- **indexed seek resolves deterministically**
- **replay preserves anomalies and sparsity**
- **ordering and semantics are exact**
- **the system remains fully offline and reproducible**

7.8 Determinism and Safety Observations

Key observations from Case-04:

- **determinism holds under irregular, bursty data**
- **sparse fields remain structurally meaningful**
- **mixed regimes do not collapse or normalize**
- **the `.star` artifact remains the sole source of truth**
- **index presence or absence does not affect correctness**

This confirms that **STAR preserves history, not convenience**.

7.9 Why Case-04 Matters

Case-04 demonstrates that **SSUM-STAR is suitable for**:

- transactional ledgers
- event-driven audit systems
- forensic reconstruction
- regulatory traceability
- long-term offline archives

It establishes that STAR remains correct even when:

- timestamps are unreliable
- data is irregular
- anomalies dominate structure

7.10 Case-04 Summary

Case-04 confirms that SSUM-STAR can:

- **derive structural time without trusting timestamps**
- **preserve anomalies and sparsity exactly**
- **replay mixed semantic data deterministically**
- **support indexed seek safely under hostile conditions**
- **operate fully offline and reproducibly**

This completes empirical validation of STAR across:

- financial-style event data (Case-01)
- noisy multivariate sensors (Case-02)
- large-scale continuous telemetry (Case-03)
- **irregular transactional event logs (Case-04)**

7.11 License & Citation (Dataset Source Only — Case-04)

Dataset License

The dataset used in STAR Case-04 is publicly available and distributed under the **CC0 1.0 Universal (Public Domain)** license, as declared by the dataset publisher on Kaggle.

Under CC0, the dataset may be:

• used for research, education, and commercial purposes
• modified, transformed, and analyzed
• shared without restriction

No attribution is legally required by the dataset license, though citation is considered best practice.

The SSUM-STAR framework and this case study are released independently under:

Creative Commons Attribution 4.0 International (CC BY 4.0)

The dataset license remains **fully independent** of SSUM-STAR.

Dataset usage conditions in this case study:

• raw dataset files are **not redistributed**
• only derived structural artifacts and results are discussed
• the original dataset license and terms remain in force

Required Dataset Citation (Case-04)

Dataset source (as published):

Cryptocurrency Transaction Analytics: Bitcoin & Ethereum

Publisher: dnkumars
Platform: Kaggle
License: CC0 1.0 Universal (Public Domain)

Dataset URL:
https://www.kaggle.com/datasets/dnkumars/cryptocurrency-transaction-analytics-btc-and-eth

---

Recommended Citation (Best Practice)

When referencing the dataset in documentation or publications, the following citation is recommended:

dnkumars.
Cryptocurrency Transaction Analytics: Bitcoin & Ethereum.
Kaggle Dataset.
https://www.kaggle.com/datasets/dnkumars/cryptocurrency-transaction-analytics-btc-and-eth

---

License Compatibility Note

The CC0 (Public Domain) license of the dataset is fully compatible with:

• CC BY 4.0–licensed research frameworks
• open research publication
• academic and applied analysis

No license contamination, inheritance, or restriction is introduced by using this dataset within SSUM-STAR Case-04.

7.12 Encoding, Index Construction, and Replay Execution (Case-04)

7.12.1 STAR Encoding

The transactional dataset was encoded as a single structural timeline, with the following guarantees:

- **row order preserved exactly**
- **mixed numeric and categorical semantics preserved**
- **sparse fields preserved without imputation**
- **identifiers preserved verbatim**
- **no timestamp trusted for ordering**

Encoding command executed:

```
python star_run.py encode \
  --case case04 \
  --csv Cryptocurrency_Transaction_Data.csv \
  --out STAR_CASE04_CRYPTO
```

Observed encoding properties:

- **rows parsed:** 50,000
- **raw file bytes:** 10,394,272
- **packed structural bytes:** 8,924,257
- **compression ratio (packed / raw):** 0.8586
- **zlib(packed) / zlib(raw):** 0.9951

Invariant enforced:

```
decode(encode(structure)) == structure
```

No preprocessing, normalization, smoothing, interpolation, or correction was applied.

7.12.2 Index Construction

An index was constructed to accelerate deterministic seek operations while preserving STAR's safety model.

Index construction command executed:

```
python star_run.py index \
  --star STAR_CASE04_CRYPTO.star \
  --out STAR_CASE04_CRYPTO.star.idx \
  --anchor_every 512 \
  --rows 50000 \
  --cadence 1
```

Observed index parameters:

- **anchor interval:** 512 rows
- **total rows indexed:** 50,000
- **anchors generated:** 99 (every 512 rows, plus final-row anchor)

Index binding proof:

- `sha256(star) = 1d2a969b6e16a5cd0f0f3d04cdc8ab521b696ddab45122ae3d26968ad243999f`
- index header magic: `b'STARIDX04\x00'`

Index safety behavior:

- the `.star` artifact remains authoritative
- the index is non-authoritative and rebuildable
- offsetless logical indexing is used when byte offsets are unsafe

The index **never defines correctness**.

7.12.3 Structural Time Validation

Structural time was derived exclusively from row continuity, not from timestamps.

Definition:

```
T_structural = row_index
```

Observed properties:

- **monotonic progression:** CONFIRMED
- **dt_nonpos:** 0
- **no fabricated gaps:** CONFIRMED
- **no drift observed:** CONFIRMED

Timestamps were replayed strictly as data fields and never used as ordering authority.

7.12.4 Indexed Seek and Replay Verification

The following seek scenarios were executed and verified:

- **start-of-timeline seek:** row 0
- **anchor-aligned seek:** row 24,576
- **mid-history seek:** row 25,000
- **late-history seek:** row 49,990
- **end-boundary seek:** row 50,000 and beyond

Example replay command executed:

```
python star_run.py replay \
  --star STAR_CASE04_CRYPTO.star \
  --idx STAR_CASE04_CRYPTO.star.idx \
  --seek_row 25000 \
  --rows 10
```

Observed behavior in all cases:

- **seek resolved to nearest valid anchor**
- **replay unfolded deterministically forward**
- **reconstructed rows matched source rows exactly**
- **output identical to linear decode**
- **no skipping, reordering, or approximation occurred**

Boundary handling was explicitly validated:

- overflow seeks clamped safely
- no index errors or undefined behavior

7.12.5 Closure Proof — Deterministic Replay (Case-04)

Artifacts:

- `STAR_CASE04_CRYPTO.star`
- `STAR_CASE04_CRYPTO.star.idx`

Binding proof:

- `sha256(star) = 1d2a969b6e16a5cd0f0f3d04cdc8ab521b696ddab45122ae3d26968ad243999f`
- index header validation: `magic = b'STARIDX04\x00'`, `anchor_every = 512`, `anchors = 99`

Replay proof:

- start boundary replay correct
- mid-history replay correct

- anchor-aligned replay correct
- late-history replay correct
- end-boundary handling safe

Index regeneration or deletion does not affect correctness.

7.12.6 Determinism and Safety Observations

Observed properties:

- **determinism holds under irregular bursts**
- **sparse fields remain sparse**
- **categorical regime switches preserved**
- **anomalies replay at identical structural positions**
- **replay correctness independent of index presence**

No hidden state, heuristics, probabilistic logic, or adaptive behavior was involved.

7.12.7 Execution Status

Case-04 execution status:

- **encoding:** COMPLETED
- **index construction:** COMPLETED
- **replay verification:** COMPLETED

This section is now **final and closure-complete**.

---

8. Indexed Seek and Time-Based Replay — Mechanics and Guarantees

---

8.1 Motivation

Structural compression and exact replay establish truth preservation—but practical systems also require **navigation**.

Real-world use demands the ability to:

• jump to arbitrary points in history
• replay bounded windows without full decoding
• inspect past states quickly and deterministically
• operate fully offline

SSUM-STAR delivers indexed navigation **without compromising correctness**, transforming compressed artifacts into **navigable historical timelines**.

8.2 Core Principle

The index is **advisory, not authoritative**.

All correctness guarantees derive **exclusively** from the `.star` artifact.
The governing invariant remains:

```
decode(encode(structure)) == structure
```

Indexes **accelerate access** but **never redefine truth**.
Deleting or ignoring an index **cannot change replay results**.

8.3 Index Construction

An index file (`.star.idx`) is constructed by recording **anchor points** at fixed intervals.

Each anchor captures:

• logical row index
• structural time (`t_min`)
• replay-safe anchor alignment

Anchors are generated every **N rows** (e.g., `N = 256`), where `N` is configurable.

This design yields:

• bounded and predictable index size
• deterministic seek cost
• replay safety (no mid-record splits)

Index construction is **offline, deterministic, and reproducible**.

8.4 Seek by Row Index

Row-based seek proceeds as follows:

1. Locate the nearest anchor with `row_anchor <= target_row`
2. Load the anchor state
3. Replay forward deterministically until `target_row` is reached

This guarantees:

• exact state reconstruction
• no skipped or inferred transitions
• results identical to full linear replay

Time complexity is bounded by **O(N)** transitions, independent of total dataset size.

8.5 Seek by Structural Time

Time-based seek leverages **intrinsic structural time**, not stored timestamps.

Given a target time `T_target`:

1. Locate the anchor whose `t_min` is closest to `T_target`
2. Replay transitions until `t_min >= T_target`
3. Emit the nearest structural state

This works even when:

• original timestamps are unreliable
• metadata is missing or inconsistent
• gaps or pauses exist in the source data

Structural time governs replay, not external clocks.

8.6 Windowed Replay

Once a seek position is established, STAR supports bounded replay:

```
replay(start_row, window_size)
```

This enables:

• forensic inspection
• rolling analysis
• partial visualization
• fast historical queries

All windowed replays are **exact sub-sequences** of the authoritative timeline.

8.7 Correctness Guarantees

Indexed seek preserves the following guarantees:

• replay equivalence: indexed replay == linear replay
• state equivalence: anchor-based replay == full decode
• time equivalence: derived structural time remains consistent
• determinism: same input → same output

Indexes introduce **no approximation, heuristics, or probabilistic shortcuts**.

8.8 Fault Tolerance

The design explicitly supports failure modes:

• missing index file → full replay still works
• corrupted index → discard safely and rebuild
• version mismatch → index ignored without error

The `.star` file remains the **single source of truth** at all times.

8.9 Empirical Validation

Indexed seek was demonstrated successfully across multiple scales.

Case-02 (Medium-Scale Validation)

• seek by row (`--seek_row 5000`)
• seek by time (`--seek_time 2004-03-15T12:00`)
• bounded replay (`--rows N`)

Case-03 (Large-Scale Validation, >2 million rows)

• seek by row at arbitrary positions (start, mid, late, overflow)
• seek by time mapped to structural minute index
• anchor boundary validation (pre-anchor and exact-anchor transitions)
• bounded replay with deterministic clamping

In all cases, **indexed replay matched linear replay exactly**, confirming:

• deterministic structural time mapping
• safe and bounded seek behavior
• correctness across both offset-based and offsetless index modes

---

8.10 Why This Matters

Indexed seek transforms STAR from:

• a compressed artifact
→ into
• a **navigable historical system**

This capability is essential for:

• audits
• investigations
• reproducible research
• long-term archival systems

All achieved **without sacrificing correctness, simplicity, or determinism**.

8.11 Summary

Indexed seek in SSUM-STAR provides:

• fast navigation
• exact replay
• offline operation
• provable correctness

By keeping indexes auxiliary and truth structural, **STAR unifies compression, navigation, and historical integrity in a single system**.

---

9. Quantitative Comparisons and Compression Ratios (Empirical Results)

9.1 Scope of Measurements

This section reports **only observed results from executed SSUM-STAR runs**.

All measurements were taken from:

• raw CSV input
• zlib-compressed raw input
• STAR-packed output
• zlib-compressed STAR output

No extrapolation, estimation, or synthetic benchmarks are included.

9.2 Metrics Used

The following metrics are reported consistently **where applicable**:

• raw file size (bytes)
• zlib(raw) size (bytes)
• STAR packed size (bytes)
• zlib(STAR) size (bytes)

Derived ratios:

• packed / raw
• zlib(STAR) / zlib(raw)

Smaller ratios indicate better compression.

9.3 Case-01 — S&P 500 (Daily OHLCV)

Rows: 39,591

Sizes:

• raw CSV: 1,650,620 bytes
• zlib(raw): 411,458 bytes
• STAR packed: 376,866 bytes
• zlib(STAR): 227,138 bytes

Ratios:

• packed / raw = 0.2283
• zlib(STAR) / zlib(raw) = 0.5520

Interpretation:

• STAR reduced raw size by ~77%
• STAR + zlib outperformed zlib alone by ~45%
• Exact replay and structural time were preserved

9.4 Case-02 — Air Quality (Hourly Multivariate Sensors)

Rows: 9,357

Sizes:

• raw CSV: 785,065 bytes
• zlib(raw): 241,254 bytes
• STAR packed: 91,309 bytes
• zlib(STAR): 67,765 bytes

Ratios:

• packed / raw = 0.1163
• zlib(STAR) / zlib(raw) = 0.2809

Interpretation:

• STAR reduced raw size by ~88%
• STAR + zlib outperformed zlib alone by ~72%
• Missing values and fault markers preserved exactly
• Indexed seek and replay by derived structural time were demonstrated without loss of fidelity

---

9.5 Case-03 — Household Power Consumption (Minute-Resolution, Large-Scale)

Rows: 2,075,259

Primary focus of Case-03 was **not compression ratio optimization**, but validation of:

• structural time derivation at multi-million-row scale
• indexed seek on very large compressed timelines
• deterministic replay without byte-offset dependency
• boundary safety and clamping behavior

Observed results:

• STAR successfully encoded and replayed the full dataset deterministically
• Structural time derived exactly at 1-minute cadence across multiple years
• Indexed seek behaved identically to linear replay at all tested positions
• Offsetless logical indexing preserved full correctness

Compression measurements were **not promoted as a primary metric** in Case-03.
The significance of Case-03 lies in **scale, determinism, and navigability**, not raw ratio comparison.

9.6 Why STAR Improves Compression Naturally

Observed improvements are explained structurally, not heuristically:

• delta encoding exploits bounded change
• repeated states collapse efficiently
• cadence regularity reduces entropy
• structure replaces redundancy

No tuning, training, or learning was applied.

9.7 Compression vs Capability Trade-off

STAR does **not trade correctness for size**.

In addition to compression, STAR provides:

• exact replay
• structural time derivation
• indexed seek
• fault and gap preservation

Classical compression provides **only size reduction**.

9.8 Cross-Case Consistency

Despite different domains and scales, STAR behavior remains consistent:

Property	Case-01	Case-02	Case-03
Deterministic encode/decode	Yes	Yes	Yes
Structural time derived	Yes	Yes	Yes
Exact replay	Yes	Yes	Yes
Indexed seek	Optional	Demonstrated	Demonstrated at scale
Fault preservation	N/A	Yes	Yes
Scale tested	Medium	Small–Medium	Very Large (>2M rows)

This consistency confirms **domain and scale independence**.

9.9 What These Numbers Do Not Claim

The empirical results presented in this document **do not claim**:

- **universal superiority over all compression methods**
- **optimal compression across all data types or distributions**
- **throughput, latency, or performance dominance**

These results demonstrate only that **structure-aware compression with exact, deterministic replay** is **practical**, **scalable across tested cases**, and **verifiable through executed runs**.

No claims are made beyond **observed behavior** in the datasets and configurations explicitly demonstrated.

9.10 Indexed Seek Impact (Observed)

The introduction of `.star.idx` indexing enables:

• direct seek by row number
• direct seek by structural time
• replay without linear scanning of the `.star` file

Empirically observed across Case-02 and Case-03:

• replay correctness unchanged
• compressed data remains authoritative
• index removal does not affect decode or replay
• index rebuilds are deterministic

The index improves **access efficiency**, not compression ratios.

9.11 Summary

Empirical results across all three cases show that SSUM-STAR:

• compresses structured data efficiently where measured
• preserves exact replay fidelity
• derives time structurally
• scales to multi-million-row timelines
• enables deterministic indexed navigation

Compression gains are a **byproduct of preserving structure**, while the true advancement lies in **replayable, navigable, truth-preserving timelines**.

---

10. Safety, Ethics, and Non-Misuse Considerations

10.1 Design Intent

SSUM-STAR is designed as an **observational and archival system**.
Its purpose is to:

• preserve structure
• enable exact replay
• support audit and reproducibility
• reduce reliance on fragile external dependencies

It is **not designed to predict, influence, or optimize outcomes**.

10.2 Non-Predictive by Construction

STAR performs:

• no forecasting
• no inference
• no learning
• no optimization

It does **not generate new data**.
All outputs are direct, deterministic reconstructions of previously observed states.

This sharply limits the risk of misuse in decision automation or manipulation.

10.3 Determinism and Transparency

All STAR operations are:

• deterministic
• explicit
• explainable
• reproducible

There are no opaque internal models, weights, or heuristics.

Any user can independently verify:

```
decode(encode(parsed)) == parsed
```

This transparency supports responsible scientific use.

10.4 Preservation of Faults and Anomalies

STAR intentionally preserves:

• missing values
• fault markers
• anomalies
• irregular gaps

It does **not** attempt to correct, smooth, or reinterpret data.

This avoids the ethical risk of silently altering historical truth, which is critical in:

• environmental data
• financial records
• regulatory archives
• forensic investigations

10.5 Offline and Infrastructure-Independent Operation

STAR requires:

• no internet connectivity
• no servers
• no cloud services
• no trusted time authorities

This reduces:

• central points of failure
• dependency risk
• surveillance vectors
• vendor lock-in

Offline operation is a **deliberate safety feature**, not a limitation.

10.6 Scope Limitations

SSUM-STAR is **not suitable** for:

• real-time control systems
• safety-critical automation
• medical decision-making
• high-frequency trading systems

It is explicitly scoped for **research, audit, archival, and reproducibility** use cases.

10.7 Ethical Use Guidelines

Users of SSUM-STAR are expected to:

• respect original dataset licenses
• provide proper attribution
• avoid misrepresenting reconstructed data as predictions
• avoid deploying STAR outputs in contexts requiring real-time guarantees

STAR preserves truth; **it does not validate truth**.

10.8 Misuse Resistance

The following properties inherently limit misuse:

• no data generation capability
• no modification of historical records
• no embedded control logic
• no opaque inference layer

STAR can expose history, but **it cannot steer outcomes**.

10.9 Alignment with Shunyaya Ethics

SSUM-STAR aligns with the broader Shunyaya framework principles:

• preservation over manipulation
• observability over control
• structure over speculation
• truth over convenience

These principles guided both design and evaluation.

10.10 Summary

SSUM-STAR is **ethically constrained by design**.

Its strengths—determinism, replay fidelity, and transparency—also function as safeguards.

When used within its intended scope, STAR supports:

• responsible research
• reproducible science
• auditable archives

Without introducing new ethical or safety risks.

---

11. License and Usage Terms (SSUM-STAR)

11.1 License Overview

SSUM-STAR (Structural Time And Replay) is released under the:

Creative Commons Attribution 4.0 International (CC BY 4.0)

This license applies to:

• the SSUM-STAR concepts and specification described in this document
• reference implementations used for demonstration
• test methodologies and evaluation procedures
• documentation text authored as part of SSUM-STAR

It does **not override or replace licenses of external datasets** used for testing or validation.

11.2 What CC BY 4.0 Allows

Under CC BY 4.0, users are permitted to:

• copy and redistribute the material
• adapt, transform, and build upon the material
• use the material for research, education, or commercial purposes

These permissions are granted without additional restrictions, provided attribution requirements are met.

11.3 Attribution and Compatibility

Any use or implementation of **SSUM-STAR (Structural Time And Replay)** must include attribution to the project name **SSUM-STAR**.

Implementations using the SSUM-STAR name are expected to preserve the core guarantee:

```
decode(encode(structure)) == structure
```

Structural time must be derived from invariant-preserving transitions, and historical values must be replayed exactly without modification.

Use of the name SSUM-STAR must **not imply endorsement** by the authors.

11.4 No Warranty and No Liability

SSUM-STAR is provided:

“AS IS”, without warranty of any kind, express or implied.

The authors make no guarantees regarding:

• fitness for a particular purpose
• correctness in safety-critical contexts
• suitability for real-time or operational control

Users assume all responsibility for their own use.

11.5 Scope of Responsibility

The CC BY 4.0 license applies **only to SSUM-STAR materials**.

External datasets referenced or processed using STAR remain governed by:

• their original licenses
• their original attribution requirements
• their original reuse conditions

Users must ensure independent compliance with those terms.

11.6 Ethical and Intended Use Reminder

While CC BY 4.0 permits broad reuse, SSUM-STAR is intended for:

• research and observation
• archival and audit
• reproducibility studies
• educational exploration

It is **not intended** for:

• safety-critical decision systems
• automated control loops
• medical or life-critical deployments

This statement clarifies intent but **does not modify license permissions**.

11.7 Relationship to the Broader Shunyaya Ecosystem

SSUM-STAR is one component within the broader Shunyaya research ecosystem.

Licensing across Shunyaya projects is not uniform and may include:

• Creative Commons licenses (including CC BY variants)
• open research or open specification licenses
• permissive or project-specific licenses

Each project **declares its license independently**.
Users must verify licensing terms **per repository, document, or release**.

No license inheritance or automatic compatibility should be assumed across components.

11.8 Summary

The CC BY 4.0 license for SSUM-STAR ensures:

• openness and reusability
• proper attribution and credit
• clear separation from dataset licenses
• legal clarity for academic and applied use

This balance enables adoption while preserving authorship, ethical intent, and responsibility boundaries.

12. Conclusion — Structural Time as a Preserved Property

SSUM-STAR (Structural Time And Replay) demonstrates that **time, order, and historical truth can be preserved structurally**, without reliance on **external clocks**, **databases**, or **probabilistic models**.

Across multiple **real-world datasets**, SSUM-STAR has shown that:

- structural compression can be lossless and deterministic
- time can be derived from invariant-preserving transitions
- compressed artifacts can function as complete, replayable timelines
- indexed seek and bounded replay are possible without sacrificing correctness
- faults, gaps, and anomalies can be preserved exactly, not corrected or hidden

All results presented in this document are based on **executed runs** and **verified outputs**, **not theoretical assumptions**.

What Has Been Established

This work establishes that:

- **compression and replay are not opposing goals**
- **structural integrity enables both efficiency and auditability**
- **time need not be stored to be trusted**
- **replay fidelity is achievable offline and indefinitely**

SSUM-STAR reframes compression as a **truth-preserving transformation**, not a **size-reduction trick**.

What STAR Intentionally Does Not Do

SSUM-STAR does **not**:

- predict future states
- infer missing values
- smooth or optimize data
- automate decisions

STAR **observes**, **preserves**, and **replays** structure — **nothing more**.

This restraint is **deliberate** and **central to its reliability**.

Position Within the Shunyaya Framework

SSUM-STAR is a **concrete application** of **Shunyaya Structural Universal Mathematics** to evolving systems.

It embodies the principles that:

- **structure precedes interpretation**
- **preservation precedes prediction**
- **observability precedes control**

By maintaining exact classical values under structural collapse, SSUM-STAR aligns with the SSUM guarantee:
`phi(decode(encode(structure))) == classical_data`

Closing Statement

SSUM-STAR shows that it is possible to:

- **store less**
- **trust more**
- **replay exactly**

— all at once.

In doing so, it introduces a **different way of thinking about compression**.

Compression need not be a trade-off between size and fidelity.
It need not discard context, smooth irregularities, or sacrifice history for efficiency.
When structure itself is preserved, **compression becomes a by-product of invariance**, not a destructive operation.

In SSUM-STAR, the compressed artifact is not merely a smaller representation of data —
it is a **structural record of progression**.

Time is not carried alongside the data; it **emerges naturally** from preserved order and continuity. History is not reconstructed heuristically; it is **replayed exactly**. This reframes compression as a **truth-preserving transformation**, where efficiency and auditability coexist by design.

When structure is preserved,
time emerges naturally,
and **history remains intact**.

OMP