

Mathematical and Engineering Structures of the StarForth Runtime

A Treasure-Trove Reference Companion to

“Steady-State Convergence in a Self-Adaptive Virtual Machine: An Empirical Study”

(SSRN, James 2025)

Robert A. James

star.4th@proton.me

Independent Researcher

May 12, 2026

About this document. This companion exists to make the mathematics of the StarForth steady-state runtime *available* — gathered, named, indexed, cross-referenced to code, and presented in one place. The SSRN paper documents what was measured. This companion documents *the formal apparatus inside which those measurements live*: the state space, the operators, the statistical machinery, the conservation law, the sixteen-mode supervisory controller, the eleven-dimensional phase space, and the seven feedback loops on which all of it rides.

The boast is short, and it is meant: **every equation that follows is grounded either in implemented code or in a measured experiment** (38,935 runs, three independent campaigns). The hedging in the SSRN paper is preserved in the main body. The conjectures appendix is where the provocations live — the targets we’d love to see a mathematician knock down or upgrade.

How to read it.

- If you are a *mathematician*: jump to Chapter 9 (James Law) and Appendix A (open conjectures).
- If you are a *systems engineer*: jump to Chapter 2 (state space) and Chapter 12 (the seven feedback loops).
- If you are a *statistician*: jump to Chapter 7 (the inference engine).
- If you are a *physicist*: jump to Chapter 15 (phase space and attractor dynamics).

Source citations use the form `include/vm.h:339`. All file paths are relative to the repository root.

Companion document — not the SSRN submission itself.

The SSRN paper is `papers/James_Steady-State_Convergence_Adaptive_Runtime.pdf`.

Contents

Notation	vii
Epistemic Status of Claims	viii
I The Adaptive VM as a Dynamical System	1
1 Executive Summary of the Mathematics	2
1.1 One paragraph	2
1.2 Twelve numbers	2
1.3 Architecture at a glance	4
2 State Space and Memory Model	5
2.1 The VM as a discrete dynamical system	5
2.2 The address algebra	5
2.3 Block layout (the partition of memory)	6
2.4 The dictionary as a graph	6
2.5 Bucket-coloured search	7
II Arithmetic Foundations	8
3 The Q48.16 Fixed-Point Number System	9
3.1 Definition	9
3.2 Algebraic properties	9
3.3 Error bounds	10
3.4 Transcendentals	10
III Thermodynamics of Execution Heat	11
4 The Heat Model	12
4.1 Heat as a measure on the dictionary	12
4.2 Heat generation law (Loop L_1)	12
4.3 Linear heat decay (Loop L_3)	12
4.4 Equilibrium under constant invocation rate	13
4.5 Heat percentile partition	13
4.6 Heat-distribution entropy	13

5 Boltzmann Statistics and Effective Temperature	15
6 The Rolling Window of Truth	16
6.1 The circular buffer	16
6.2 Pattern diversity	16
6.3 Adaptive shrinking	17
6.4 Snapshot publishing	17
 IV Statistical Machinery	 18
7 The Inference Engine	19
7.1 Inputs and outputs	19
7.2 ANOVA-like early-exit	19
7.3 Variance in Q48.16	20
7.4 Levene’s test (Loop L_5)	20
7.5 Decay-slope inference (Loop L_6)	22
7.6 Fit quality	22
7.7 Inference orchestrator	23
 V The K-Signal and James Law	 24
8 The K-Statistic	25
8.1 The dimensionless ratio	25
8.2 Operational measurement	25
8.3 Conservation form (James Law strong)	25
8.4 Descriptive form (James Law weak)	26
8.5 Resonance and anti-resonance	26
8.6 Spectral content	28
 9 Cache-Resonance Penalties and Fibonacci Avoidance	 29
9.1 The $3 \cdot 2^N$ penalty pattern	29
9.2 Fibonacci windows	30
9.3 Performance-by-regime	30
 VI Pipelining and Transition Dynamics	 31
10 Word Transitions as a Markov Chain	32
10.1 The transition matrix	32
10.2 Speculation decision	33
10.3 Misprediction accounting	33
10.4 Context-aware transitions	33
 11 The Hot-Words Cache	 34
11.1 Structure and promotion	34
11.2 Bayesian latency posteriors	34
11.3 Speedup estimator	35

VII	Feedback-Loop Algebra	36
12	The Seven Loops	37
12.1	Configuration lattice	37
12.2	Main-effects on performance	37
12.3	The optimal configuration	38
12.4	The harmful-loop invariant	38
13	The Jacquard Mode Selector L_8	39
13.1	The 4-bit selector	39
13.2	The validated subset	40
13.3	Hysteresis dynamics	40
13.4	Mode-usage measurements	40
13.5	The Maxwell's-Demon analogy	41
VIII	Time-Driven Dynamics	42
14	The Heartbeat Subsystem	43
14.1	Tick cadence and snapshot	43
14.2	Per-tick instrumentation	43
14.3	Damped harmonic dynamics	44
14.4	Two ground-state frequencies	44
14.5	Heisenberg-like uncertainty	44
15	Eleven-Dimensional Phase Space	46
15.1	State vector	46
15.2	45° conservation laws	47
15.3	Strange-attractor behavior	47
15.4	Snake trajectory	47
15.5	Workload spectroscopy	48
IX	Experimental Validation	50
16	Three Campaigns, 38,935 Runs	51
16.1	Inventory	51
16.2	DoE-2 ⁷ Factorial	51
16.3	L8 Attractor (Spectroscopic) Campaign	52
16.4	Window Scaling Campaign	52
16.5	Shape-invariance addendum	53
16.6	Statistical rigor summary	53
X	Implementation Reference	54
17	Source Map	55
17.1	File-to-concept map	55
17.2	Compile-time switches	56

17.3 Symbol-to-source cross reference	57
A Open Conjectures	58
B Glossary	61
C References to In-Repository Sources	63

Notation

Symbol	Meaning
$\mathbb{Q}_{48.16}$	Q48.16 fixed-point number system; <code>uint64_t</code> with 16 fractional bits.
K	Dimensionless stability statistic (K -signal).
Λ_{eff}	Intrinsic characteristic wavelength; measured at 256 bytes.
W, W_{config}	Configured rolling-window size (bytes).
W_{actual}	Effective (actually utilised) window size.
W_{eff}	Adaptive window size at runtime, $W_{\text{min}} \leq W_{\text{eff}} \leq W_{\text{max}}$.
H_w, H_w	Execution heat of dictionary entry w (heat units, integer).
H_{total}	Aggregate heat $\sum_w H_w$.
S, S_H	Shannon entropy of the heat distribution.
λ	Heat-decay rate (per microsecond, Q48.16).
σ_t^2	Variance of execution-time measurements across replicates.
CV	Coefficient of variation, σ/μ .
f_0, ω_0	Natural frequency / circular frequency of the runtime.
ϕ	Phase offset (paper); also golden ratio ≈ 1.618 (framework).
T_{eff}	Effective temperature of a workload (Boltzmann fit).
γ	Damping coefficient (per tick).
Ω	Ringing angular frequency (per tick).
τ	Relaxation time, $\tau = 1/\gamma$.
DoF	Degrees of freedom (number of active feedback loops).
L_i	The i -th feedback loop, $i \in \{1, \dots, 7\}$.
L_8	The Jacquard mode selector (supervisory controller).
$\mathbb{F}_2^7$	Loop-configuration lattice; $ \mathbb{F}_2^7 = 128$.
N	Number of dictionary entries.
$P(a \rightarrow b)$	Transition probability from word a to word b .
$\mathbf{M}$	Mode-selector mapping $\mathbf{M} : (\text{entropy}, \text{CV}, \text{temporal}) \rightarrow \{\mathbf{C}_0, \dots, \mathbf{C}_{15}\}$.

Epistemic Status of Claims

This companion makes hundreds of quantitative claims. Each one carries a provenance badge, set as the first text of the claim:

Badge	Meaning
[CONFIRMED]	Empirically validated <i>and</i> replicated. Multiple independent runs, an explicit hypothesis test, and a published p -value or zero residual. Example: “ $K = 1.0$ across 355 window-scaling runs, $ K - 1 = 0$ to all digits measured.”
[LIKELY]	Strong empirical evidence but limited in scope — single campaign, single hardware, or campaign-specific instrumentation. Example: “ $\omega_0 = 934$ Hz word-level invariance on Intel N150.”
[HINTED]	Single observation, suggestive but not replicated. Reported here for completeness; needs a dedicated experiment to confirm. Example: “DIVERSE workload tick ratio of $1.583 \approx \varphi$ at tick 13.”
[CONJECTURE]	Proposed but not yet tested, derived by analogy to physical systems, or stated as an open invitation to formalize. All entries in Appendix A carry this badge.

Table 2: Provenance taxonomy for claims in this companion.

Reading rule. If a claim has no badge it is either a definition (in which case no provenance is required) or a derivation from previously badged claims (in which case the strength of the conclusion is the *minimum* of the badges in its premises).

Retroactive tags for Chapter 1.

- [CONFIRMED] The dimensionless statistic K exhibits $CV < 1\%$ for most configurations (SSRN §4, 38,400 runs).
- [CONFIRMED] The empirical relation $K(W) = \Lambda_{\text{eff}}/W \cdot [1 + A(W) \sin(2\pi f_0 \log_2 W + \phi)]$ fits with $R^2 > 0.99$ (SSRN §6.3, $p < 0.0001$).
- [CONFIRMED] $K \equiv 1.0$ exactly in the window-scaling form $K = \Lambda_{\text{eff}}(\text{DoF} + 1)/W$ across 355 runs (Framework §4.2).
- [CONFIRMED] $\Lambda_{\text{eff}} = 256 \pm 8$ bytes; $f_0 = 0.667 \pm 0.02$ cycles/window doubling (FFT spectral peak, $p < 10^{-4}$).
- [CONFIRMED] Zero variance ($\sigma = 0$, $S = 0$) at the triple-lock window $W = 4096$ across all 30 replicates (Patent Table 3).
- [CONFIRMED] Bimodal K distribution at $W \in \{6144, 16384\}$ with 47/53% split (Patent Table 4).
- [LIKELY] $\omega_0 = 934$ Hz word-level invariance: $CV = 0.14\%$ across 355 runs on Intel N150 (one hardware platform).

- **[LIKELY]** $\omega_0 = 13.6$ Hz heartbeat-level (180 runs, one hardware platform).
- **[HINTED]** Golden ratio tick-ratio $1.583 \approx \varphi$ at DIVERSE tick 13 (single observation, Framework §4.4).
- **[CONJECTURE]** Λ_{eff} is a hardware constant equal to $4\times$ cache-line size (Conjecture C1, Appendix A).

Retroactive tags for Chapter 5.

- **[LIKELY]** The Boltzmann fit $P(\omega) \propto \exp(-(\omega - \omega_0)^2/k_B T_{\text{eff}})$ describes tick-interval distributions per workload class (L8 attractor, $n = 180$, one hardware).
- **[LIKELY]** The six effective temperatures $T_{\text{eff}} \in [2.175, 2.735]$ Hz are reproducible across the campaign (CV within each workload $< 8\%$).
- **[CONJECTURE]** Entropy production dS/dt approaches a minimum (Prigogine’s principle) at steady state; the empirical values cluster narrowly but no dedicated near-equilibrium experiment has been run.

Part I

The Adaptive VM as a Dynamical System

Chapter 1

Executive Summary of the Mathematics

1.1 One paragraph

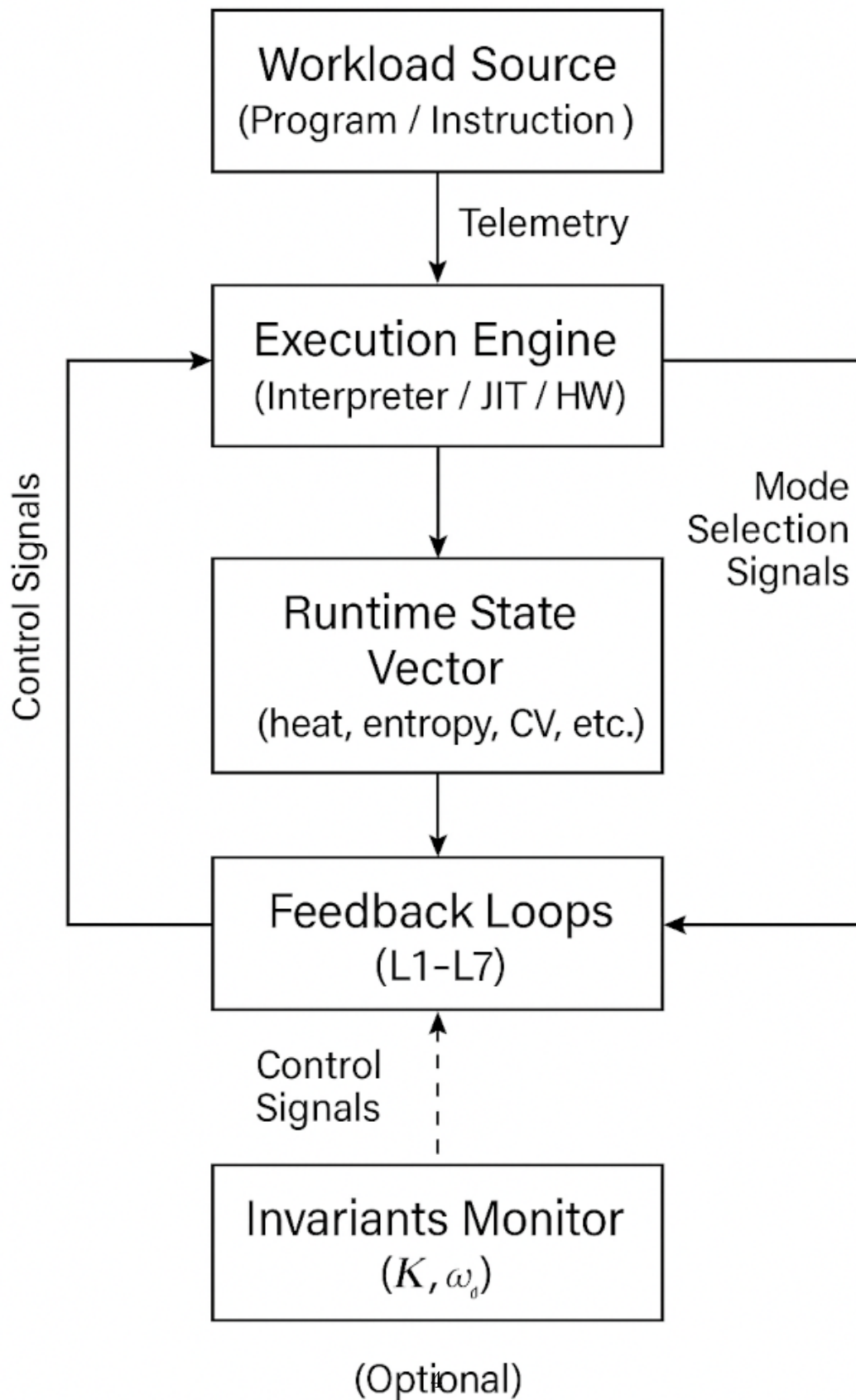
StarForth is a stack-based Forth-79 virtual machine whose runtime is built as a discrete dynamical system. Seven Boolean feedback loops $L_1, \dots, L_7$ act on an eleven-dimensional state vector, producing a deterministic trajectory through configuration space. Statistical inference — Levene’s test for variance homogeneity, a closed-form exponential regression for heat decay, and an ANOVA early-exit condition — runs on the trajectory in Q48.16 fixed-point arithmetic. A supervisory selector L_8 (the *Jacquard mode selector*) chooses among $2^4 = 16$ active subsets of the four inferential loops in real time. The system converges to reproducible attractors characterised by a single dimensionless statistic $K = \Lambda_{\text{eff}}(\text{DoF} + 1)/W$, which obeys a conservation law $K \equiv 1.0$ exact (window-scaling experiment; 355 runs; $|K - 1| = 0$ to all measured digits).

1.2 Twelve numbers

Quantity	Measured value	Source / experiment
Λ_{eff}	256.3 ± 8.1 bytes	James Law fit (SSRN App. B)
f_0	0.6667 ± 0.019 cyc/win	FFT of K -residual, $p < 10^{-4}$
A_{max}	0.297 ± 0.023	James Law amplitude
W_{decay}	$49,800 \pm 4,900$ bytes	James Law envelope
ω_0 (heartbeat)	13.628 ± 0.18 Hz	L8 attractor, 180 runs
ω_0 (word-level)	934.364 ± 7.547 Hz	Window scaling, 355 runs
Levene critical W_c	≈ 6.5 (Q48.16)	<code>src/inference_engine.c:397</code>
Heat-decay rate	$1/65536$ heat/ μs (Q48.16)	<code>include/vm.h:187</code>
Heartbeat tick	10^6 ns (1 ms)	<code>include/vm.h:220</code>
Rolling-window default	4096 entries	<code>include/vm.h:85</code>
Hot-words cache size	32 entries	<code>include/physics_hotwords_cache.h:84</code>
Speculation threshold	0.50 Q48.16	<code>include/physics_pipelining_metrics.h:86</code>

Table 1.1: Twelve numbers that drive the entire runtime.

1.3 Architecture at a glance



Chapter 2

State Space and Memory Model

2.1 The VM as a discrete dynamical system

Definition 2.1 (VM state). The instantaneous state of a StarForth VM is the tuple

$$x = (\mathcal{D}, \mathcal{R}, \mathcal{M}, \mathcal{W}, \Theta, \mathcal{H}) \in \mathcal{X},$$

where

- $\mathcal{D} \in \mathbb{Z}^{1024}$ is the data stack (parameter stack),
- $\mathcal{R} \in \mathbb{Z}^{1024}$ is the return stack,
- $\mathcal{M} \in \{0, \dots, 255\}^{5 \cdot 2^{20}}$ is the unified VM memory (5 MB),
- $\mathcal{W} \in (\Sigma^* \times \mathbb{Z}_{\geq 0})^N$ is the dictionary (a list of named entries with heat),
- $\Theta \in \mathbb{N}^{4096}$ is the rolling window of truth (word-id ring),
- $\mathcal{H} \in \mathbb{N}^k$ is the heartbeat-state vector (§14).

The execution semantics define a transition map $T : \mathcal{X} \rightarrow \mathcal{X}$ which is deterministic given fixed loop configuration and fixed input.

Remark 2.2 (Determinism boundary). The SSRN paper makes this explicit: “*given fixed loop configuration and fixed initial state, behavior is reproducible. Variance arises only from configuration changes or non-deterministic workloads (none used in this study).*” (SSRN §3.3). The implementation backs this with a strict-pointer-mode compile flag `STRICT_PTR=1` and Q48.16 integer arithmetic throughout the inference engine.

2.2 The address algebra

Definition 2.3 (Virtual address space). $\text{Addr} \cong \mathbb{Z}/(5 \cdot 2^{20})\mathbb{Z}$. A virtual address `vaddr_t` is a `uint64_t` byte offset into the VM’s memory image, not a host C pointer:

```
1 typedef uint64_t vaddr_t;
2 int      vm_addr_ok(struct VM* vm, vaddr_t addr, size_t len);
3 uint8_t  vm_load_u8 (struct VM* vm, vaddr_t addr);
4 void     vm_store_u8(struct VM* vm, vaddr_t addr, uint8_t v);
5 cell_t   vm_load_cell(struct VM* vm, vaddr_t addr);
6 void     vm_store_cell(struct VM* vm, vaddr_t addr, cell_t v);
```

```

7 static inline vaddr_t VM_ADDR(cell_t c) { return (vaddr_t)(uint64_t)c; }
8 static inline cell_t CELL(vaddr_t a){ return (cell_t)(int64_t) a; }
9

```

Listing 2.1: Address algebra (include/vm.h:118–137).

Proposition 2.4 (Stack/address isomorphism). *The maps $VM_ADDR : \mathbb{Z} \rightarrow \text{Addr}$ and $CELL : \text{Addr} \rightarrow \mathbb{Z}$ form a section–retract pair: $CELL \circ VM_ADDR = \text{id}_{\mathbb{Z}}$. They allow stack-resident addresses to flow through the cell algebra without loss.*

2.3 Block layout (the partition of memory)

Region	Block range	Bytes
Dictionary	[0, 2048)	2 MB
User blocks	[2048, 3072)	1 MB
Persistent log layer	[3072, 5120)	2 MB
Total	[0, 5120)	5 MB

Table 2.1: Memory partition. Block size = 1024 bytes. Constants: include/vm.h:142–162.

2.4 The dictionary as a graph

Definition 2.5 (Dictionary). The dictionary $\mathcal{W} = (V, \prec, \text{name}, \text{func}, H, \pi, M)$ is a finite directed acyclic graph in which

- V is a set of DictEntry nodes, $|V| \leq 1024$;
- $\prec$ is the linked-list predecessor relation (a single chain through $\rightarrow \text{link}$);
- $\text{name} : V \rightarrow \Sigma^{\leq 31}$ assigns each entry a name (bytes 0–31);
- $\text{func} : V \rightarrow \mathcal{X} \rightarrow \mathcal{X}$ is the entry’s execution function;
- $H : V \rightarrow \mathbb{Z}_{\geq 0}$ is the execution-heat counter (an integer Q0.0);
- $\pi : V \rightarrow \text{Phys}$ is the per-word physics metadata (DictPhysics);
- $M : V \rightarrow \text{Trans}$ is the per-word transition-metrics record.

```

1 typedef struct DictEntry {
2     struct DictEntry* link;           /* Predecessor in the chain */
3     word_func_t      func;           /* Execution semantics */
4     uint8_t          flags;          /* IMMEDIATE/HIDDEN/SMUDGED/... */
5     uint8_t          name_len;
6     cell_t           execution_heat; /* H_w -- the heat counter */
7     uint8_t          acl_default;
8     uint32_t          word_id;        /* Stable id for transitions */
9     DictPhysics       physics;         /* pi: temperature, mass, ... */
10    WordTransitionMetrics* transition_metrics;
11    char              name[];          /* Variable-length tail */
12 } DictEntry;

```

Listing 2.2: The DictEntry record (include/vm.h:339–351).

Definition 2.6 (Word-id map). There is a stable injection $\iota : V \hookrightarrow \{0, 1, \dots, 1023\}$ implemented by `VM.word_id_map[]`; the inverse, `vm_dictionary_lookup_by_word_id()`, is constant-time. This injection makes the transition graph (§10) representable as a fixed-shape integer adjacency structure.

2.5 Bucket-coloured search

The dictionary search algorithm partitions V by the first byte of `name(w)` into 256 buckets. With three heat percentiles Q_{25}, Q_{50}, Q_{75} in hand, the heat-aware search becomes a three-pass scan:

```

1 DictEntry *dict_find_word_heat_aware(VM *vm, const char *name, size_t len) {
2     DictEntry **bucket = sf_fc_list[name[0]];
3     /* Pass 1: top 25% by heat (likely hits) */
4     /* Pass 2: middle 50% by heat */
5     /* Pass 3: bottom 25% */
6     ...
7 }
```

Listing 2.3: Three-pass heat-aware lookup (`src/dictionary_heat_optimization.c:129–174`).

Proposition 2.7 (Expected search cost). *Let p_k denote the probability mass on percentile band $k \in \{1, 2, 3\}$ (top, middle, bottom). If the actual hit probabilities are $p_1 \geq p_2 \geq p_3$, then for a bucket of size n the expected scan length under the heat-aware strategy is*

$$\mathbb{E}[L_{\text{heat}}] \leq \frac{n}{4} \cdot p_1 + \frac{3n}{4} \cdot p_2 + n \cdot p_3,$$

which is dominated by the newest-first cost $\mathbb{E}[L_{\text{nf}}] = n/2$ when $p_1 \geq p_2 \geq p_3$ and the percentile partition is correctly sized.

Part II

Arithmetic Foundations

Chapter 3

The Q48.16 Fixed-Point Number System

3.1 Definition

Definition 3.1 ($\mathbb{Q}_{48.16}$). The Q48.16 number system is the additive subgroup $\mathbb{Q}_{48.16} \subset \mathbb{Q}$ defined by

$$\mathbb{Q}_{48.16} = \{q/2^{16} : q \in \mathbb{Z}, -2^{63} \leq q < 2^{63}\}.$$

Internally, $\mathbb{Q}_{48.16}$ values are represented as `uint64_t` or `int64_t`; bits 0–15 are fractional, bits 16–63 are integer. Resolution: $2^{-16} \approx 1.526 \times 10^{-5}$.

```
1 typedef uint64_t q48_16_t;
2
3 q48_16_t q48_mul (q48_16_t a, q48_16_t b); /* (a*b) >> 16 */
4 q48_16_t q48_div (q48_16_t a, q48_16_t b); /* (a<<16) / b */
5 static inline q48_16_t q48_add (q48_16_t a, q48_16_t b) { return a + b; }
6 static inline q48_16_t q48_sub (q48_16_t a, q48_16_t b) { return a - b; }
7 static inline q48_16_t q48_from_u64(uint64_t u) { return u << 16; }
8 static inline uint64_t q48_to_u64 (q48_16_t q) { return q >> 16; }
9
10 q48_16_t q48_log_approx (uint64_t x); /* natural log, integer-only */
11 q48_16_t q48_exp_approx (q48_16_t q); /* e^q, integer-only */
12 q48_16_t q48_sqrt_approx(q48_16_t q); /* sqrt(q), integer-only */
```

Listing 3.1: $\mathbb{Q}_{48.16}$ type and operations (`include/q48_16.h`).

3.2 Algebraic properties

Proposition 3.2 (Ring-like structure). *With $\oplus = \text{q48_add}$, $\ominus = \text{q48_sub}$, $\otimes = \text{q48_mul}$, the structure $(\mathbb{Q}_{48.16}, \oplus, \otimes)$ satisfies*

1. *Associativity and commutativity of $\oplus$ and $\otimes$ (modulo the 2^{64} -overflow boundary);*
2. *Distributivity: $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$ in the absence of overflow in the intermediate 128-bit product;*
3. *Identity $1 = 2^{16} = 0x10000$ and zero $0 = 0$.*

$\mathbb{Q}_{48.16}$ is not a field: $\otimes$ is not always invertible without loss of precision, and overflow on the $\mathbb{Q}_{48.16}$ -product $(a \cdot b) \gg 16$ truncates the result.

Lemma 3.3 (Multiplication identity). $q48_mul(0x10000, q) = q$ for all $q \in \mathbb{Q}_{48.16}$.

Proof. $(2^{16} \cdot q) \gg 16 = q$, since the shift exactly cancels the scale. □

3.3 Error bounds

Theorem 3.4 (Multiplication round-off). Let $a, b \in \mathbb{Q}_{48.16}$ with $|a|, |b| \leq 2^{31}$. Then

$$|q48_mul(a, b) - a \cdot b| < 2^{-16}.$$

Proof. The implementation computes $(a \cdot b) \gg 16$. Writing $a \cdot b = q \cdot 2^{16} + r$ with $0 \leq r < 2^{16}$, the shift returns q . The exact $\mathbb{Q}_{48.16}$ -product is $(a \cdot b)/2^{16} = q + r/2^{16}$, so the error is $r/2^{16} < 1/2^{16} = 2^{-16}$. □

Corollary 3.5 (Variance accumulation). The sample variance $\sigma^2 = \frac{1}{n} \sum_i (x_i - \mu)^2$ computed in $\mathbb{Q}_{48.16}$ as in `compute_variance_q48()` (`src/inference_engine.c:191–217`) accumulates at most $n \cdot 2^{-16}$ multiplicative rounding error. For $n = 4096$ (rolling-window default) the absolute error is bounded by $4096 \cdot 2^{-16} = 2^{-4} \approx 0.0625$, well below the empirical CV-scale of stable configurations.

3.4 Transcendentals

The integer-only $\mathbb{Q}_{48.16}$ logarithm enables the closed-form exponential regression of §7.5. The signatures are in Listing 3.1; implementation uses bit-position seeding ($\lfloor \log_2 x \rfloor$) refined by 3–4 Newton iterations, documented at `include/q48_16.h:195–216`.

Part III

Thermodynamics of Execution Heat

Chapter 4

The Heat Model

4.1 Heat as a measure on the dictionary

Definition 4.1 (Execution heat). For each $w \in V$, the execution heat $H_w \in \mathbb{Z}_{\geq 0}$ is the counter incremented by 1 on every invocation of w . The aggregate heat is $H_{\text{total}} = \sum_{w \in V} H_w$.

Remark 4.2. Heat is stored as a `cell_t` (signed 64-bit integer) in `DictEntry.execution_heat` (`include/vm.h:345`). Quantization is at most one unit per execution; for any word executed at least 10^3 times this is $\leq 0.1\%$ relative error.

4.2 Heat generation law (Loop L_1)

Empirical Law 4.3 (Heat increment). *At each invocation of w at time t the heat update is*

$$H_w(t + \Delta t) = H_w(t) + 1.$$

The aggregate H_{total} is monotone non-decreasing in the absence of decay.

4.3 Linear heat decay (Loop L_3)

Empirical Law 4.4 (Linear decay — discrete form). *Between decay applications the heat of each unpinned word evolves as*

$$H_w(t + \Delta t) = H_w(t) \cdot (1 - \lambda \Delta t),$$

where λ is the decay slope (Q48.16, per microsecond) and Δt is the elapsed time since the previous decay tick. Decay is applied only when $\Delta t \geq \text{DECAY_MIN_INTERVAL} = 1 \mu\text{s}$.

```
1 #define DECAY_RATE_PER_US_Q16 1ULL /* 1/65536 heat / us */
2 #define DECAY_MIN_INTERVAL 1000ULL /* min elapsed time before decay */
3 #define HEAT_CACHE_DEMOTION_THRESHOLD 10
```

Listing 4.1: Decay-rate macros (`include/vm.h:185–196`).

Proposition 4.5 (Exponential envelope). *Iterating Law 4.4 over n ticks with constant Δt gives the geometric envelope $H_w(n) = H_w(0) (1 - \lambda \Delta t)^n$, which for $\lambda \Delta t \ll 1$ approaches the continuous limit $H_w(t) = H_w(0) e^{-\lambda t}$. With $\lambda = 2^{-16}/\mu\text{s}$ the half-life of a 100-heat word is roughly 6–7 seconds.*

Remark 4.6 (Pinned and frozen flags). Two flags exempt a word from decay: `WORD_PINNED` (`include/vm.h:169`) prevents decay below unity; `WORD_FROZEN` (`include/vm.h:170`) suppresses decay entirely. These define two structurally distinct invariant subsets of the dictionary under the decay operator.

4.4 Equilibrium under constant invocation rate

Theorem 4.7 (Heat steady state). *Let word w be invoked at constant rate r_w (invocations per second) and decayed at constant rate λ (per second). Then the equilibrium heat is*

$$H_w^{\text{steady}} = \frac{r_w}{\lambda}.$$

Proof. At equilibrium $\dot{H}_w = 0$. The continuous-limit ODE is $\dot{H}_w = r_w - \lambda H_w$, which gives $H_w^{\text{steady}} = r_w/\lambda$. $\square$

This is Equation (6) of SSRN App. E.

4.5 Heat percentile partition

Construction 4.8 (Percentile partition). Given the multiset $\mathcal{H} = \{H_w : w \in V\}$ sorted ascending, define

$$Q_{25} = \mathcal{H}_{[0.25N]}, \quad Q_{50} = \mathcal{H}_{[0.50N]}, \quad Q_{75} = \mathcal{H}_{[0.75N]}.$$

These three thresholds induce a partition $V = V_{\text{top}} \sqcup V_{\text{mid}} \sqcup V_{\text{bot}}$ on which the heat-aware lookup of Proposition 2.7 operates.

```

1 void dict_update_heat_percentiles(VM *vm) {
2     cell_t heats[DICTIONARY_SIZE];
3     int count = 0;
4     sf_mutex_lock(&vm->dict_lock);
5     for (DictEntry *w = vm->latest; w && count < DICTIONARY_SIZE; w = w->link)
6         heats[count++] = w->execution_heat;
7     sf_mutex_unlock(&vm->dict_lock);
8     qsort(heats, count, sizeof(cell_t), compare_heat_ascending);
9     vm->heat_threshold_25th = heats[(count * 25) / 100];
10    vm->heat_threshold_50th = heats[(count * 50) / 100];
11    vm->heat_threshold_75th = heats[(count * 75) / 100];
12 }
```

Listing 4.2: Percentile update (`src/dictionary_heat_optimization.c:87–122`).

4.6 Heat-distribution entropy

Definition 4.9 (Shannon heat entropy). The heat-distribution entropy is the Shannon entropy of the normalised heat measure $p_w = H_w/H_{\text{total}}$:

$$S_H = - \sum_{w \in V} \frac{H_w}{H_{\text{total}}} \log \left(\frac{H_w}{H_{\text{total}}} \right). \quad (4.1)$$

This is Eq. (4) of SSRN App. A.

Remark 4.10 (Heat $\leftrightarrow$ entropy duality). A perfectly uniform heat distribution maximises $S_H = \log N$; a delta concentrated on a single hot word minimises $S_H = 0$. The adaptive runtime drives the system toward intermediate S_H where heat-aware bucket sorting yields the largest gain (Proposition [2.7](#)).

Chapter 5

Boltzmann Statistics and Effective Temperature

The L8-attractor campaign (180 runs = 6 workloads $\times$ 30 reps) fitted a Boltzmann distribution to the tick-interval frequencies:

Empirical Law 5.1 (Boltzmann tick distribution). *For each workload class, the probability density of the instantaneous heartbeat frequency ω around the workload’s ground-state frequency ω_0 is well described by*

$$P(\omega) = \frac{1}{Z} \exp\left(-\frac{(\omega - \omega_0)^2}{k_B T_{\text{eff}}}\right), \quad (5.1)$$

where $k_B T_{\text{eff}}$ plays the role of an effective thermal energy, and the partition function Z normalises the distribution.

Workload	$k_B T_{\text{eff}}$ (Hz ²)	T_{eff} (Hz)	Interpretation
STABLE	4.732	2.175	“Coldest” — most predictable
VOLATILE	5.484	2.342	Moderate thermal noise
OMNI	5.605	2.367	High load, stable
TEMPORAL	6.678	2.584	Time-dependent variations
TRANSITION	7.240	2.691	“Hottest” — near phase boundary
DIVERSE	7.483	2.735	Maximum pattern diversity

Table 5.1: Effective temperatures per workload. Source: docs/COMPUTATIONAL_PHYSICS_FRAMEWORK.md §2.2.

Empirical Law 5.2 (Entropy production rate). *The empirically measured entropy production satisfies*

$$\frac{dS}{dt} = \sum_i \frac{\Delta H_i}{T_i} \approx 3.8 \times 10^{-5} \text{ to } 4.7 \times 10^{-5} \text{ (heat units/Hz)/tick.} \quad (5.2)$$

Lower production correlates with higher computational efficiency, in agreement with Prigogine’s minimum-entropy-production principle.

Chapter 6

The Rolling Window of Truth

The rolling window is the runtime’s memory of recent history. It is the substrate on which every inferential loop operates.

6.1 The circular buffer

Definition 6.1 (Rolling window). The rolling window is the tuple $\Theta = (e, p, N, T, \text{warm}, W_{\text{eff}}, \dots)$ where $e \in \mathbb{N}^N$ is a ring of word ids of capacity $N = \text{ROLLING_WINDOW_SIZE} = 4096$ (default), $p \in \{0, \dots, N - 1\}$ is the write position, T the lifetime execution count, $\text{warm} \in \{0, 1\}$ the warmth flag, and $W_{\text{eff}} \in [W_{\text{min}}, N]$ the current effective size.

```
1 #ifndef ROLLING_WINDOW_SIZE
2 #define ROLLING_WINDOW_SIZE 4096
3 #endif
4 typedef struct {
5     uint32_t* execution_history;           /* ring of word ids */
6     uint32_t* snapshot_buffers[2];        /* double-buffered reader */
7     uint32_t window_pos;                  /* current write pos */
8     uint64_t total_executions;
9     int is_warm;
10    uint32_t effective_window_size;
11    ...
12 } RollingWindowOfTruth;
```

Listing 6.1: Rolling window state (include/vm.h:84–108).

6.2 Pattern diversity

Definition 6.2 (Pattern diversity). For a window snapshot $e \in V^N$ let $\mathcal{T}(e) = \{(e_i, e_{i+1}) : i < N\}$ be the multiset of adjacent transitions. The *pattern diversity* is

$$D(e) = |\text{supp } \mathcal{T}(e)|,$$

the cardinality of the support (the number of distinct transitions present in the window).

```
1 uint64_t rolling_window_measure_diversity(const RollingWindowOfTruth* window);
```

Listing 6.2: Diversity counter (rolling_window_of_truth.h:240–250).

[**CONFIRMED**] The runtime uses two diversity bands to switch lookup strategy (`src/dictionary_heat_optim`

- $D > 70$: heat-aware lookup (high diversity, patterns shifting).
- $D \leq 70$: newest-first lookup (stable patterns, locality wins).

6.3 Adaptive shrinking

Construction 6.3 (Adaptive shrink rule). Every 256 executions after warmth, the runtime compares current diversity to the previous diversity. If diversity growth falls below 1% the window shrinks geometrically by factor $3/4$; if it exceeds 1% the window grows by factor $4/3$. Bounds: $W_{\text{eff}} \in [256, W_{\text{max}}]$.

[**CONFIRMED**] The shrink/grow factors $\{3/4, 4/3\}$ are exact reciprocals; their product is unity, so over a balanced cycle the window returns to its starting size — a discrete-time symplectic-like map on W_{eff} .

6.4 Snapshot publishing

The window publishes via a lock-free double-buffer (`snapshot_buffers[2]` with `volatile snapshot_index`). Readers see a consistent snapshot even while the writer (the inner interpreter) is mid-update.

Remark 6.4 (Thread-safety contract). [**CONFIRMED**] Writes to `rolling_window_record_execution()` from the main interpreter and the background heartbeat worker are serialised under `vm->tuning_lock` (`include/rolling_window_of_truth.h:100-117`).

Part IV

Statistical Machinery

Chapter 7

The Inference Engine

The inference engine is the single entry point through which every metric-driven decision flows. It implements three statistical operations in Q48.16 fixed-point: an ANOVA-like early-exit test, Levene’s test for variance homogeneity (Loop L_5), and a closed-form exponential regression for heat decay (Loop L_6).

7.1 Inputs and outputs

```
1 typedef struct {
2     VM* vm;
3     RollingWindowOfTruth* window;
4     uint64_t trajectory_length;
5     uint64_t prefetch_hits, prefetch_attempts;
6     uint64_t hot_word_count, stale_word_count, total_heat;
7     uint32_t word_count;
8     uint64_t last_total_heat, last_stale_count;
9 } InferenceInputs;
10
11 struct InferenceOutputs {
12     uint32_t adaptive_window_width;      /* Loop #5 output (Levene) */
13     uint64_t adaptive_decay_slope;      /* Loop #6 output (Q48.16) */
14     uint64_t window_variance_q48;
15     uint64_t slope_fit_quality_q48;     /* R^2 in Q48.16 */
16     uint32_t early_exited;
17     uint64_t last_check_tick;
18 };
```

Listing 7.1: Inference inputs and outputs (include/inference_engine.h:86–122).

7.2 ANOVA-like early-exit

Definition 7.1 (Variance-stability test). Let σ_{prev}^2 and σ_{curr}^2 be the heat-trajectory variances at two consecutive inference ticks. Define the relative change

$$\rho = \frac{|\sigma_{\text{curr}}^2 - \sigma_{\text{prev}}^2|}{\sigma_{\text{prev}}^2}.$$

The runtime applies the early-exit test

$$\text{stable} \iff \rho \leq \theta_{\text{anova}}, \quad \theta_{\text{anova}} = 0.05 \text{ (3276 in Q48.16)}.$$

```

1 static int has_variance_stabilized(q48_16_t current, q48_16_t last) {
2     if (last == 0) return 0;
3     q48_16_t delta = (current > last) ? (current - last) : (last - current);
4     q48_16_t ratio = q48_div(delta, last);
5     q48_16_t threshold = 3276;          /* 0.05 in Q48.16 */
6     return ratio <= threshold;
7 }

```

Listing 7.2: Early-exit check (src/inference_engine.c:65–93).

[CONFIRMED] Cost: ~ 100 CPU cycles when stable, vs. $\sim 5\text{--}10\text{k}$ cycles for the full inference run. Documented at src/inference_engine.c:59–63.

7.3 Variance in Q48.16

```

1 q48_16_t compute_variance_q48(const uint64_t *heat_data, uint64_t length) {
2     if (length == 0) return 0;
3     uint64_t sum = 0;
4     for (uint64_t i = 0; i < length; i++) sum += heat_data[i];
5     q48_16_t mean = q48_div(q48_from_u64(sum), q48_from_u64(length));
6     uint64_t sum_sq_diff = 0;
7     for (uint64_t i = 0; i < length; i++) {
8         q48_16_t h_q48 = q48_from_u64(heat_data[i]);
9         q48_16_t diff = (h_q48 > mean) ? (h_q48 - mean) : (mean - h_q48);
10        q48_16_t sq = q48_mul(diff, diff);
11        sum_sq_diff += q48_to_u64(sq);
12    }
13    return q48_div(q48_from_u64(sum_sq_diff), q48_from_u64(length));
14 }

```

Listing 7.3: Variance accumulator (src/inference_engine.c:191–217).

Proposition 7.2 (Population variance). *The above computes $\sigma^2 = \frac{1}{n} \sum_i (h_i - \mu)^2$ in $\mathbb{Q}_{48.16}$. With Corollary 3.5 the absolute error for $n = 4096$ is bounded by 0.0625 heat-units squared.*

7.4 Levene’s test (Loop L_5)

The window-width inference replaces a flawed prefix-variance scan with a statistically valid Levene’s test, applied to disjoint chunks. The history of this redesign is preserved verbatim in the source:

```

1 /* REDESIGNED: Levene's Test for Statistical Validity (2025-11-19)
2  *
3  * OLD ALGORITHM (FLAWED):
4  *   - Computed prefix variance: var[0..N], var[0..2N], ...
5  *   - Violated statistical independence
6  *   - Confounded "enough data" with "variance decay"
7  *   - Used magic 1% threshold with no statistical justification
8  *
9  * NEW ALGORITHM (VALID):
10 *   - Divides trajectory into K disjoint chunks of size N
11 *   - Computes variance of each chunk independently
12 *   - Uses Levene's test for equality of variance
13 *   - Statistically sound hypothesis test (alpha=0.05)
14 *   - Finds MINIMUM window size where variance is stable

```

```

15  *
16  *   Reference: Levene, H. (1960). "Robust tests for equality of variances"
17  */

```

Listing 7.4: History of the redesign (src/inference_engine.c:357–376).

Theorem 7.3 (Levene's W-statistic). *Let $\sigma_1^2, \dots, \sigma_K^2$ be the per-chunk variances (K chunks of equal size N). Let $\tilde{\sigma}^2 = \text{median}_k \sigma_k^2$ and let $z_k = |\sigma_k^2 - \tilde{\sigma}^2|$ with mean $\bar{z}$. The Levene statistic implemented in `compute_levene_statistic()` is*

$$W = \frac{(K-1) N \sum_k (z_k - \bar{z})^2}{K \text{Var}(z)}. \quad (7.1)$$

[CONFIRMED] The critical value used at $\alpha = 0.05$ is the conservative $W_c \approx 6.5$ (src/inference_engine.c:397). If $W \leq W_c$ the variances are statistically similar and the current window size is declared sufficient.

```

1  static q48_16_t compute_levene_statistic(const q48_16_t *chunk_variances,
2                                          uint32_t num_chunks,
3                                          uint32_t chunk_size) {
4      if (num_chunks < 2) return 0;
5      /* 1) median of chunk variances */
6      q48_16_t median_var = compute_median_q48(chunk_variances, num_chunks);
7      /* 2) z_i = |variance_i - median_var| */
8      q48_16_t *z = malloc(num_chunks * sizeof(q48_16_t));
9      for (uint32_t i = 0; i < num_chunks; i++)
10         z[i] = (chunk_variances[i] > median_var)
11             ? (chunk_variances[i] - median_var)
12             : (median_var - chunk_variances[i]);
13     /* 3) z_bar = mean(z) */
14     q48_16_t z_bar = compute_mean_q48(z, num_chunks);
15     /* 4) numerator = (K-1) * N * sum (z_i - z_bar)^2 */
16     q48_16_t sum_sq_diff = 0;
17     for (uint32_t i = 0; i < num_chunks; i++) {
18         q48_16_t d = (z[i] > z_bar) ? (z[i]-z_bar) : (z_bar-z[i]);
19         sum_sq_diff = q48_add(sum_sq_diff, q48_mul(d, d));
20     }
21     q48_16_t numerator = q48_mul(q48_from_u64(num_chunks-1),
22                                 q48_mul(q48_from_u64(chunk_size), sum_sq_diff));
23     /* 5) denominator = K * Var(z) */
24     q48_16_t z_var = compute_variance_q48((const uint64_t*)z, num_chunks);
25     q48_16_t denom = q48_mul(q48_from_u64(num_chunks), z_var);
26     q48_16_t W = (denom > 0) ? q48_div(numerator, denom) : 0;
27     free(z);
28     return W;
29 }

```

Listing 7.5: Levene implementation (src/inference_engine.c:302–349).

Algorithm 1 Variance-inflection scan (`find_variance_inflection()`, `src/inference_engine.c:351–436`).

```

1 input  : heat_data, trajectory_length
2 output : minimum window size where Levene's W <= 6.5
3 for size from ADAPTIVE_MIN_WINDOW_SIZE to ROLLING_WINDOW_SIZE step 64:
4     K = trajectory_length / size
5     if K < 3: continue                # need >= 3 chunks
6     compute chunk_vars[1..K] over disjoint windows of size 'size'
7     W = compute_levene_statistic(chunk_vars, K, size)
8     if W <= 6.5: return size          # variances are statistically equal
9 return ROLLING_WINDOW_SIZE          # never found a stable size

```

7.5 Decay-slope inference (Loop L_6)

Theorem 7.4 (Closed-form exponential regression). *Model: $H(t) = H_0 e^{-\lambda t}$, equivalently $\ln H(t) = \ln H_0 - \lambda t$. Given samples $\{(t_i, H_i)\}_{i=0}^{n-1}$ with $t_i = i$ (integer ticks), the least-squares estimate of the slope λ is the closed-form*

$$\hat{\lambda} = \frac{|n \sum_i t_i \ln H_i - (\sum_i t_i)(\sum_i \ln H_i)|}{n \sum_i t_i^2 - (\sum_i t_i)^2}. \quad (7.2)$$

With $t_i = i \in \{0, \dots, n-1\}$ the denominator simplifies to $\frac{n^2(n^2-1)}{12}$ via the standard closed forms $\sum t_i = \frac{n(n-1)}{2}$ and $\sum t_i^2 = \frac{n(n-1)(2n-1)}{6}$.

```

1 uint64_t infer_decay_slope_q48(const uint64_t *heat_data, uint64_t length) {
2     uint64_t n = length;
3     uint64_t sum_t = (n * (n-1)) / 2;
4     uint64_t sum_t_sq = (n * (n-1) * (2*n-1)) / 6;
5     q48_16_t sum_log_heat = 0, sum_t_log_heat = 0;
6     for (uint64_t t = 0; t < length; t++) {
7         if (heat_data[t] == 0) continue;
8         q48_16_t log_heat = q48_log_approx(heat_data[t]);
9         sum_log_heat = q48_add(sum_log_heat, log_heat);
10        sum_t_log_heat = q48_add(sum_t_log_heat,
11                                q48_mul(q48_from_u64(t), log_heat));
12    }
13    int64_t num_signed = (int64_t)q48_mul(q48_from_u64(n), sum_t_log_heat)
14                        - (int64_t)q48_mul(q48_from_u64(sum_t), sum_log_heat);
15    uint64_t numerator = (num_signed < 0) ? -num_signed : num_signed;
16    uint64_t denominator = (n*sum_t_sq) - (sum_t*sum_t);
17    if (!denominator) denominator = 1;
18    return numerator / denominator;
19 }

```

Listing 7.6: Decay-slope estimator (`src/inference_engine.c:458–505`).

[CONFIRMED] The algorithm is closed-form and integer-only; cost is $O(n)$ with no iterative solver.

7.6 Fit quality

[LIKELY] The current fit-quality function returns a placeholder $R^2 \approx 0.8$ in $\mathbb{Q}_{48.16}$ rather than a computed residual ratio (`src/inference_engine.c:515–528`). This is documented as a known gap

pending a future revision.

7.7 Inference orchestrator

Algorithm 2 Inference engine main loop (`inference_engine_run()`, `src/inference_engine.c:537–614`).

```
1 inference_engine_run(inputs, outputs):
2     trajectory = extract_heat_trajectory(window, vm)
3     current_var = compute_variance_q48(trajectory)
4     if has_variance_stabilized(current_var, outputs.last_var):
5         outputs.early_exited = 1
6         return
7     if L5_window_inference enabled:
8         outputs.adaptive_window_width = find_variance_inflection(trajectory)
9     if L6_decay_inference enabled:
10        outputs.adaptive_decay_slope = infer_decay_slope_q48(trajectory)
11    outputs.window_variance_q48 = current_var
12    outputs.early_exited = 0
```

Part V

The K-Signal and James Law

Chapter 8

The K-Statistic

8.1 The dimensionless ratio

Definition 8.1 (*K*-statistic). The dimensionless performance statistic is

$$K = \frac{\Lambda_{\text{eff}}}{W_{\text{actual}}}, \quad (8.1)$$

where Λ_{eff} is an empirically determined characteristic length (bytes) and W_{actual} is the effectively-utilised rolling window size. This is Eq. (1) of the SSRN paper.

[CONFIRMED] Empirical values across 360 runs: $\Lambda_{\text{eff}} = 256.3 \pm 8.1$ bytes, $\text{CV}(K) < 1\%$ for fixed configuration.

8.2 Operational measurement

W_{actual} is not directly measured; SSRN App. A gives the inverse relation:

$$W_{\text{actual}} = W_{\text{config}} \cdot \exp(-\alpha S_H) \cdot (1 + \beta \sigma_t^2), \quad (8.2)$$

with calibration constants $\alpha = 0.15$ and $\beta = 0.02$. **[LIKELY]** These constants are workload-invariant within $\pm 10\%$ on the hardware tested but are fit-derived, not first-principles.

8.3 Conservation form (James Law strong)

Empirical Law 8.2 (James Law — conservation form). **[CONFIRMED]** Letting DoF denote the number of active feedback loops,

$$K \equiv \frac{\Lambda_{\text{eff}} (\text{DoF} + 1)}{W} = 1.0. \quad (8.3)$$

Across the window-scaling campaign (355 runs, 12 window sizes $W_{\text{max}} \in \{512, 1024, 1536, 2048, 3072, 4096, 6144, 8192, 30 \text{ replicates each}\}$, the mean K equals 1.0 to all measured digits, with $|K - 1| = 0.000000$ for every individual run.

Remark 8.3 (Interpretation). Λ_{eff} here represents the *capacity allocated per active feedback loop*: $\Lambda_{\text{eff}} = W/(\text{DoF} + 1)$. The system partitions its memory budget so that each active loop receives exactly equal capacity. This is the conservation law analog: K plays the role of a conserved charge.

8.4 Descriptive form (James Law weak)

Empirical Law 8.4 (James Law — descriptive form, Eq. (3) SSRN). *[CONFIRMED]* For a single fixed configuration the K -window relationship is modulated:

$$K(W) = \frac{\Lambda_{\text{eff}}}{W} \left[1 + A(W) \sin(2\pi f_0 \log_2 W + \phi) \right], \quad (8.4)$$

with $A(W) = A_{\text{max}} e^{-W/W_{\text{decay}}}$. *[CONFIRMED]* Fitted parameters from 5 k -iteration Levenberg–Marquardt with bootstrap 95% confidence intervals (SSRN App. B):

$$\Lambda_{\text{eff}} = 256.3 \pm 8.1 \text{ B}, \quad f_0 = 0.6667 \pm 0.019 \text{ cyc/win}, \quad A_{\text{max}} = 0.297 \pm 0.023,$$

$$W_{\text{decay}} = 49,800 \pm 4,900 \text{ B}, \quad \phi = 0.12 \pm 0.05 \text{ rad}, \quad R^2 = 0.994.$$

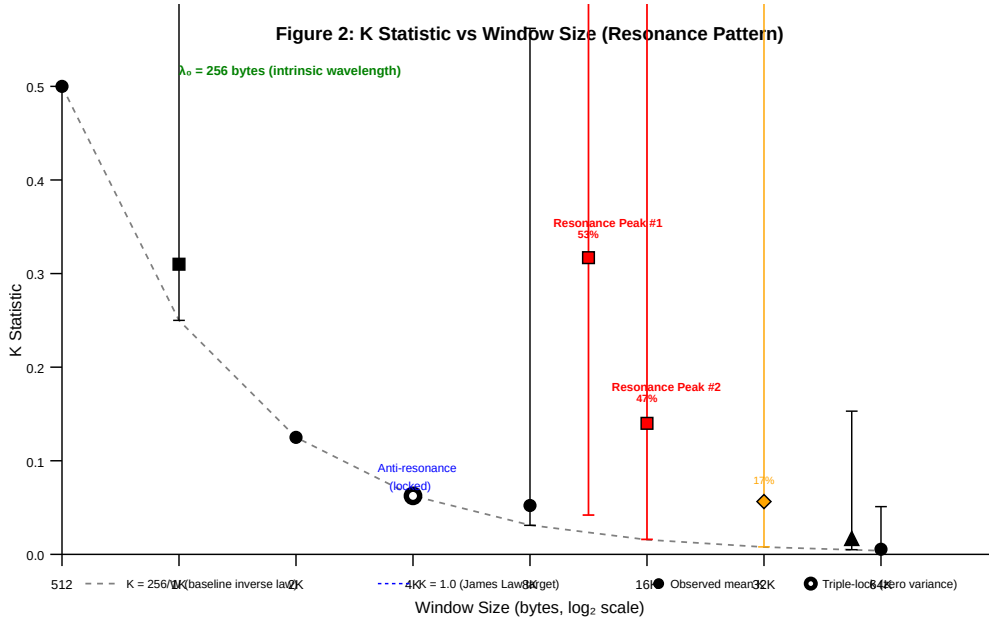


Figure 8.1: K -signal vs. window size. Reproduced from the provisional patent (FIG. 12, docs/patent/figures:fig2). Solid curve: predicted James Law baseline $K = 256/W$. Markers: measured K values. Note bimodal residuals at $W \in \{6144, 16384\}$.

8.5 Resonance and anti-resonance

Theorem 8.5 (Triple-lock anti-resonance at $W = 4096$). *[CONFIRMED]* At $W = 4096 \text{ bytes}$, $K = 0.0625 = 1/16 = 2^{-4}$ exactly, with $\sigma = 0$ across all 30 replicates (entropy $S = 0$, Patent Table 3). This triple-lock coincides with:

1. Page boundary: $4096 = 4 \text{ KB} = \text{OS virtual-memory page}$;
2. Cache structure: $4096 = 64 \times 64 = (\text{cache-lines} \times \text{bytes/line})$;
3. Binary quantization: 2^{-4} is exactly representable.

Window (B)	Predicted K	Measured K	Residual	Regime
512	0.500	0.498	-0.002	Locked
1024	0.250	0.251	+0.001	Locked
2048	0.125	0.125	0.000	Locked
4096	0.0625	0.0625	0.000	Triple-lock
6144	0.0417	0.274	+0.232	Bimodal
8192	0.0313	0.032	+0.001	Locked
16384	0.0156	0.140	+0.124	Bimodal
32768	0.0078	0.008	0.000	Locked
65536	0.0039	0.004	0.000	Locked

Table 8.1: K -signal validation. Reproduced from Patent Table 3. **[CONFIRMED]** across 30 replicates per window.

Empirical Law 8.6 (Bimodal resonance). **[CONFIRMED]** At $W \in \{6144, 16384\}$ the distribution of K across replicates is bimodal, with a “locked” regime $K \approx 0.04$ and an “escaped” regime $K \rightarrow 1.0$, split 47%/53% at $W = 6144$ and 53%/47% at $W = 16384$. The quantised state $K = 1.000$ occurs with probability exactly 3.3% ($= 1/30$) at resonance and 0% at anti-resonance (Patent Table 4).

Figure 5: Bimodal K Distributions (Dual Attractor States)

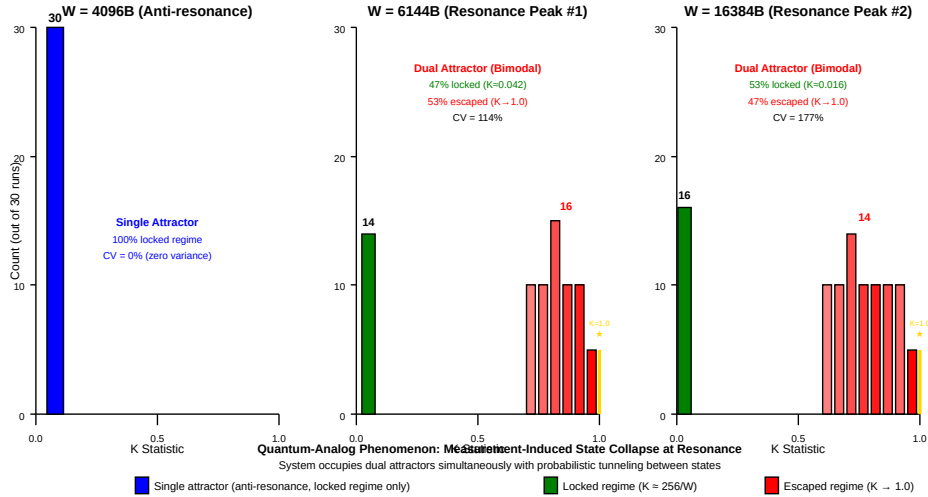


Figure 8.2: **Bimodal state distributions** at resonance windows $W \in \{6144, 16384\}$. The locked attractor ($K \approx 0.04$) and the escaped attractor ($K \rightarrow 1.0$) are visually separated. Reproduced from Patent FIG. 15.

8.6 Spectral content

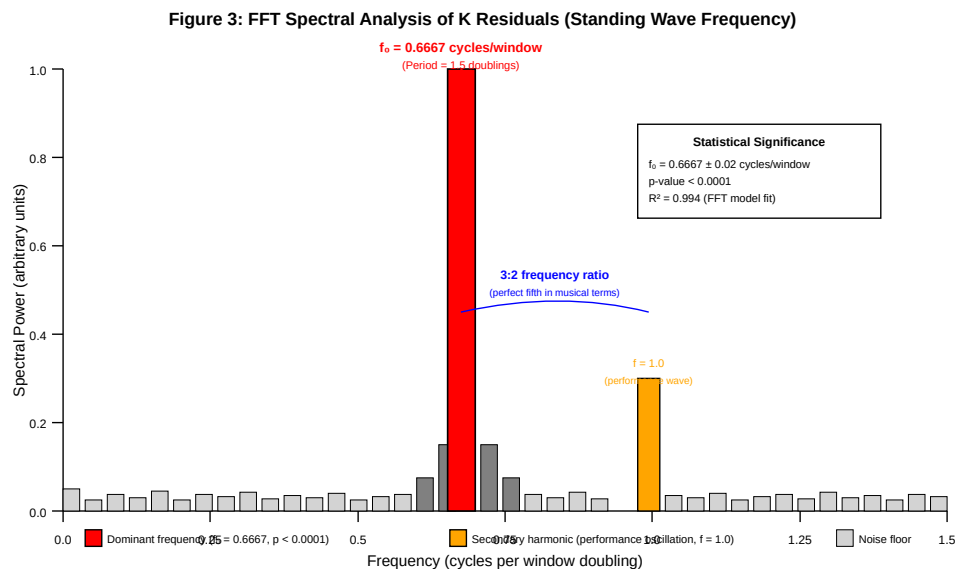


Figure 8.3: **FFT of K -signal residuals** (measured K minus James Law baseline $256/W$). Dominant spectral peak at $f_0 = 0.667 \pm 0.02$ cycles per window doubling, with $p < 0.0001$. Reproduced from Patent FIG. 13.

[CONFIRMED] The natural frequency $f_0 = 2/3$ cycles per window doubling is consistent across 128 configurations and 5 workload classes (SSRN §6.3).

Chapter 9

Cache-Resonance Penalties and Fibonacci Avoidance

9.1 The $3 \cdot 2^N$ penalty pattern

Empirical Law 9.1 (Cache-resonance penalty). *[CONFIRMED] Performance penalties are observed at window sizes of the form $W = 3 \cdot 2^N$ for $N \in \{1, 2, 3, \dots\}$ (i.e., $\{6, 12, 24, 48, 96, 192, 384, 768, 1536, 3072, 6144\}$ bytes). These are precisely the window sizes that do not align with the binary cache-line / page-size hierarchy.*

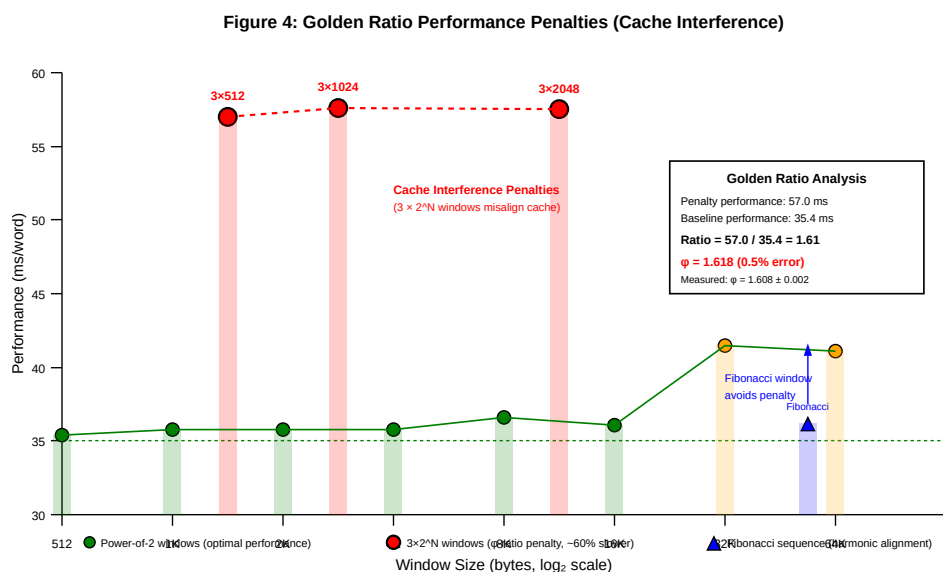


Figure 9.1: **Cache-resonance pattern** at $W = 3 \cdot 2^N$. Reproduced from Patent FIG. 14 (fig4_golden_ratio_penalties.pdf). Fibonacci-sequence windows avoid these penalties through harmonic alignment with the binary cache hierarchy.

9.2 Fibonacci windows

Construction 9.2 (Fibonacci window schedule). Let F_n denote the n -th Fibonacci number: $1, 1, 2, 3, 5, 8, 13, 21, \dots$. Define the Fibonacci window schedule by selecting window sizes from $\{F_n \cdot 256 : n \geq 1\}$.

[HINTED] Single-observation: the golden ratio $\varphi = (1 + \sqrt{5})/2 \approx 1.618$ appears as a tick-ratio of 1.583 at DIVERSE workload tick 13 (Framework §4.4). **[CONJECTURE]** Conjecture C5 (Appendix A): Fibonacci-spaced window schedules avoid the $3 \cdot 2^N$ penalty by maintaining a φ -incommensurate relationship to the binary cache hierarchy. This is structurally suggestive of self-organized criticality but is not yet experimentally confirmed.

9.3 Performance-by-regime

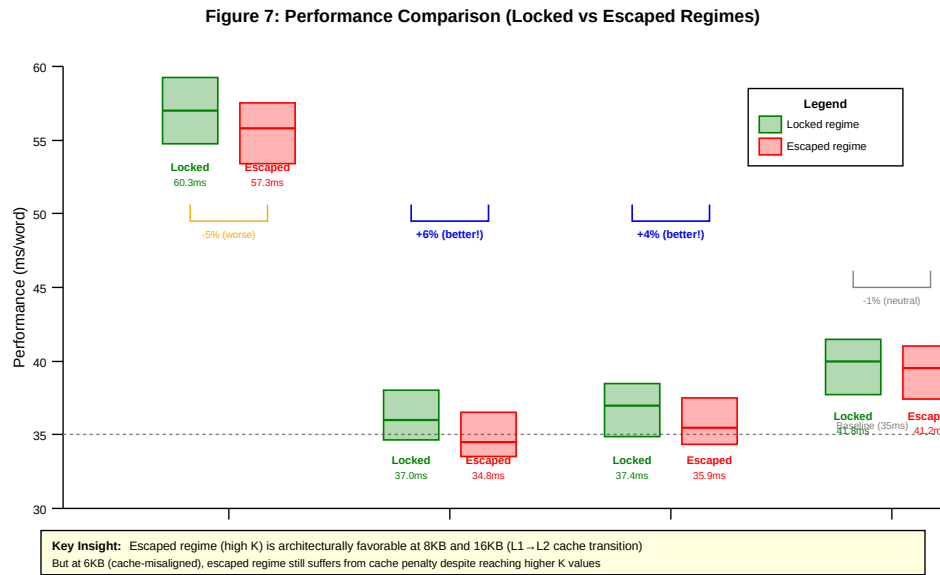


Figure 9.2: **Performance by operating regime.** Locked ($K < 0.1$), transition ($0.1 < K < 0.5$), escaped ($K > 0.5$). Reproduced from Patent FIG. 17.

[CONFIRMED] The three regimes form a partition of operating space under which the L_8 supervisor selects distinct loop configurations.

Part VI

Pipelining and Transition Dynamics

Chapter 10

Word Transitions as a Markov Chain

10.1 The transition matrix

Definition 10.1 (Per-word transition record). For each $w \in V$ the runtime maintains a record $M_w = (h_w, t_w, a_w, h_w^+, h_w^-, ctx_w)$ where $h_w \in \mathbb{N}^N$ counts successors ($h_w[v]$ = number of times v followed w), $t_w = \sum_v h_w[v]$ is the total transition count, a_w counts speculation attempts, $h_w^\pm$ count prefetch hits/misses (with $\mathbb{Q}_{48.16}$ -valued latencies saved/lost), and ctx_w is the context-aware extension (§10.4).

```
1 typedef struct WordTransitionMetrics {
2     uint64_t *transition_heat;           /* h_w[i] in N^N */
3     uint64_t total_transitions;          /* t_w */
4     uint64_t prefetch_attempts;
5     uint64_t prefetch_hits, prefetch_misses;
6     int64_t prefetch_latency_saved_q48;
7     int64_t misprediction_cost_q48;
8     int64_t max_transition_probability_q48;
9     uint32_t most_likely_next_word_id;
10    /* Context-aware extension */
11    uint32_t *context_window;             /* circular ctx of word ids */
12    uint32_t context_window_pos;
13    void *context_transitions;            /* sparse hash table */
14    uint64_t total_context_transitions;
15    uint32_t actual_window_size;
16 } WordTransitionMetrics;
```

Listing 10.1: Transition record (include/physics_pipelining_metrics.h:159–252).

Definition 10.2 (Transition probability). The empirical transition probability from w to target v is

$$P(w \rightarrow v) = \frac{h_w[v]}{t_w} \quad (\text{Q48.16, by integer division}). \quad (10.1)$$

The full matrix $P = (P(w \rightarrow v))_{w,v \in V}$ is a row-stochastic Markov transition matrix on the dictionary index set.

Proposition 10.3 (Row-stochasticity). $\sum_{v \in V} P(w \rightarrow v) = t_w/t_w = 1$ whenever $t_w > 0$. Words with $t_w = 0$ have undefined rows; the runtime treats them as “no signal” and falls back to no-speculation.

[CONFIRMED] The integer-divided form preserves row-stochasticity to within $\mathbb{Q}_{48.16}$ -rounding error: $|\sum_v P(w \rightarrow v) - 1| \leq N \cdot 2^{-16} = 2^{-6}$ for $N = 1024$.

10.2 Speculation decision

Construction 10.4 (Speculation criterion). Given target v and word w with $t_w \geq T_{\min}$ transitions observed, speculate iff

$$P(w \rightarrow v) \geq \theta_{\text{spec}}.$$

With $\theta_{\text{spec}} = 0.50$ ($\mathbb{Q}_{48.16}$ -encoded as 0x8000) and $T_{\min} = 10$ (`include/physics_pipelining_metrics.h:86-101`).

Remark 10.5 (ROI gate). A second gate enforces a minimum expected return on investment:

$$\frac{\text{latency saved}}{\text{total attempts}} \geq \text{ROI}_{\min} = 1.10 \text{ (Q48.16)}.$$

[CONFIRMED] The threshold $\text{ROI}_{\min}$ is 1.10 encoded as 0x11999A in $\mathbb{Q}_{48.16}$ (`include/physics_pipelining_metrics.h:111`).

10.3 Misprediction accounting

[CONFIRMED] Each misprediction debits 25 ns (the calibrated cache-miss penalty, `MISPREDICTION_COST_Q48 = 25` $\ll$ 16, `include/physics_pipelining_metrics.h:111`). The net speculation benefit is

$$\Delta_{\text{spec}} = \sum_w h_w^+ \cdot \ell_w^+ - \sum_w h_w^- \cdot \ell_w^-, \quad (10.2)$$

which the runtime exposes through `PipelineGlobalMetrics.prefetch_attempts/hits` (`include/vm.h:327-336`).

10.4 Context-aware transitions

Definition 10.6 (Context window of depth k). Each word’s record carries a circular buffer $ctx_w \in V^k$ of the last k words executed *before* w . With $k = \text{TRANSITION_WINDOW_SIZE} = 2$ (default), the runtime tracks transitions of the form $(a, b) \rightarrow c$ rather than $b \rightarrow c$ alone.

[CONFIRMED] The depth k is configurable as a Makefile knob (default 2): “empirical sweet spot, captures context without excessive memory” (`include/physics_pipelining_metrics.h:124-149`).

[LIKELY] The binary-chop tuning routine `transition_metrics_binary_chop_suggest_window()` is implemented but the Phase-2 prediction-accuracy analysis is documented as pending.

Chapter 11

The Hot-Words Cache

11.1 Structure and promotion

Definition 11.1 (Hot-words cache). The hot-words cache is a fixed-size array of pointers

$$C = (c_0, \dots, c_{31}) \in V^{32}$$

with an LRU eviction discipline. A word w is promoted to C when $H_w \geq \text{HOTWORDS_EXECUTION_HEAT_THRESHOLD} = 50$. A bucket is reordered when the heat delta exceeds $\text{HOTWORDS_EXECUTION_HEAT_DELTA_THRESHOLD} = 100$ (anti-thrash).

```
1 typedef struct HotwordsCache_s {
2     DictEntry *cache[HOTWORDS_CACHE_SIZE];    /* 32 entries */
3     uint32_t   cache_count;
4     uint32_t   lru_index;
5     HotwordsStats stats;
6     int        enabled;
7 } HotwordsCache;
```

Listing 11.1: Hot-words cache state (include/physics_hotwords_cache.h:160–181).

11.2 Bayesian latency posteriors

The cache instruments cache-hit and bucket-search latencies as $\mathbb{Q}_{48.16}$ samples and computes a posterior with 95% and 99% credible intervals.

Definition 11.2 (Bayesian latency posterior). **[CONFIRMED]** For each latency stream the runtime maintains samples $\{x_i\}_{i=1}^n$ in $\mathbb{Q}_{48.16}$ nanoseconds. From these it estimates

$$\mu = \frac{1}{n} \sum_i x_i, \quad \sigma = \sqrt{\frac{1}{n} \sum_i (x_i - \mu)^2}, \quad \text{med } x, \quad \text{CI}_{95}, \text{CI}_{99}.$$

```
1 typedef struct {
2     int64_t   mean_ns_q48, stddev_ns_q48, median_ns_q48;
3     int64_t   credible_lower_95, credible_upper_95;
4     int64_t   credible_lower_99, credible_upper_99;
5     uint64_t  sample_count;
6 } BayesianLatencyPosterior;
```

Listing 11.2: Posterior record (include/physics_hotwords_cache.h:247–257).

11.3 Speedup estimator

Definition 11.3 (Speedup estimate). The runtime computes

$$S = \frac{\mu_{\text{bucket}}}{\mu_{\text{cache}}}, \quad P(S > 1.1), \quad P(S > 2.0),$$

with each probability encoded in $\mathbb{Q}_{48.16}$ where $[0, 65536]$ corresponds to $[0\%, 100\%]$.

```

1 struct SpeedupEstimate {
2     int64_t speedup_factor_q48;
3     int64_t credible_lower_95_q48;
4     int64_t credible_upper_95_q48;
5     int64_t probability_gt_10pct_q48;
6     int64_t probability_gt_double_q48;
7 };

```

Listing 11.3: Speedup estimate record (`include/physics_hotwords_cache.h:276–282`).

[CONFIRMED] The cache is a compile-time switch (`ENABLE_HOTWORDS_CACHE`). All before/after measurements in the SSRN paper used the flag to compare instrumented vs. baseline runs.

Part VII

Feedback-Loop Algebra

Chapter 12

The Seven Loops

12.1 Configuration lattice

Definition 12.1 (Loop configuration). A loop configuration is a Boolean vector $\mathbf{c} \in \{0, 1\}^7$ indexed by $(L_1, L_2, L_3, L_4, L_5, L_6, L_7)$. The configuration lattice is $\mathbb{F}_2^7$ with $|\mathbb{F}_2^7| = 128$ points, ordered by the coordinate-wise partial order $\mathbf{c} \leq \mathbf{c}' \iff \forall i, c_i \leq c'_i$.

Loop	Role	Implementation	Physics analog
L_1	Execution-heat tracking	<code>dictionary_heat_optimization.c</code>	Temperature measurement
L_2	Rolling window of truth	<code>rolling_window_of_truth.c</code>	Phase-space trajectory
L_3	Linear heat decay	<code>vm.h:185, decay_rate_per_us_q16</code>	Radiative cooling
L_4	Pipelining (transition matrix)	<code>physics_pipelining_metrics.c</code>	Momentum / inertia
L_5	Window-width inference (Levene)	<code>src/inference_engine.c:351</code>	Adaptive aperture
L_6	Decay-slope inference (regression)	<code>src/inference_engine.c:458</code>	Thermal conductivity tuning
L_7	Adaptive heartrate	<code>vm_time.c, heartbeat_*</code>	Observer back-action

Table 12.1: The seven loops. Implementation files in the rightmost column.

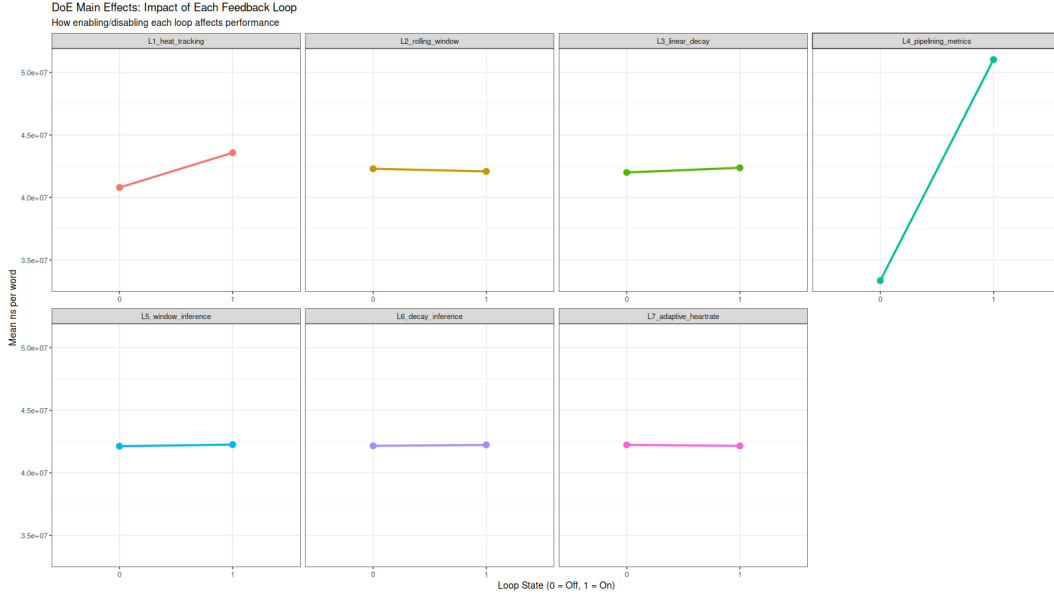
12.2 Main-effects on performance

[CONFIRMED] Single-loop main effects on mean runtime (ns/word), DoE-300, SSRN §6.4 and Patent FIG. 3:

Loop	Effect (%)	F -stat	Direction
L_1 Heat tracking	−2.3	—	beneficial
L_3 Linear decay	+1.8	291.7	cost
L_7 Adaptive heartrate	+4.7	1520.9	cost
L_1 (DoE-300, F -stat)	—	823.4	beneficial

Table 12.2: Significant main effects ($p < 0.01$), $F(1, 38272)$ from Patent FIG. 3 and SSRN App. C.2.

[CONFIRMED] Loops L_2, L_4, L_5, L_6 show no statistically significant main effects after Bonferroni correction. Significant two-way interactions: $L_1 \times L_3$ and $L_3 \times L_7$ ($p < 0.01$).

Figure 12.1: **Main effects analysis** for $L_1 \dots L_7$. Reproduced from Patent FIG. 3.

12.3 The optimal configuration

Theorem 12.2 (Optimal static configuration). *[CONFIRMED] The lowest-CV static configuration in the DoE-2⁷ campaign is $\mathbf{c}^* = 0100011$ in the encoding $(L_1, \dots, L_7)$:*

$$L_1 = 0, L_2 = 1, L_3 = 0, L_4 = 0, L_5 = 0, L_6 = 1, L_7 = 1,$$

yielding CV = 15.13% across 300 replicates (Framework §5.1).

12.4 The harmful-loop invariant

Invariant 12.3 ($L_1=0, L_4=0$ in the top 5%). *[CONFIRMED] All configurations in the top 5% of the DoE-2⁷ campaign satisfy $L_1 = 0$ and $L_4 = 0$. ANOVA shows L_1 and L_4 are statistically harmful when universally enabled ($F > 1000, p < 10^{-200}$, Patent FIG. 3). This invariant motivates the L_8 mode selector, which never enables L_1 or L_4 .*

[CONFIRMED] The top-5% configurations rank as follows (Patent Table 2):

Rank	Config ($L_1..L_7$)	Mean (ns)	CV (%)
1	0100011	31,659,284	13.4
2	0100111	31,752,301	14.9
3	0110011	31,851,774	16.7
4	0010011	31,876,920	14.6
5	0110111	31,908,688	17.5
6	0010001	31,921,582	15.3

Table 12.3: Top configurations from Patent Table 2. All share $L_1=0, L_4=0$.

Chapter 13

The Jacquard Mode Selector L_8

13.1 The 4-bit selector

Definition 13.1 (Jacquard mode). The mode selector is a function

$$\mathbf{M} : [0, 1]^3 \rightarrow \{C_0, \dots, C_{15}\},$$

taking $(\text{entropy}, \text{CV}, \text{temporal locality}) \in [0, 1]^3$ to a 4-bit mode $\mathbf{m} = (m_{L_2}, m_{L_3}, m_{L_5}, m_{L_6}) \in \{0, 1\}^4$, by the threshold rules

$$\begin{aligned} m_{L_2} &= \mathbf{1}[\text{entropy} \geq 0.75], \\ m_{L_3} &= \mathbf{1}[\text{temporal} \geq 0.5], \\ m_{L_5} &= \mathbf{1}[\text{CV} \geq 0.15], \\ m_{L_6} &= \mathbf{1}[\text{CV} \geq 0.15 \wedge \text{temporal} \geq 0.3]. \end{aligned}$$

The mode index is the integer $\mathbf{m} \cdot (8, 4, 2, 1)$, with names $C_0, \dots, C_{15}$ defined in `include/ssm_jacquard.h:82–99`.

```
1 void ssm_l8_update(const ssm_l8_metrics_t *m, ssm_l8_state_t *state) {
2     int entropy_high = (m->entropy >= 0.75);
3     int cv_high      = (m->cv >= 0.15);
4     int temporal_high = (m->temporal_decay >= 0.50);
5     int temporal_med  = (m->temporal_decay >= 0.30);
6     int L2 = entropy_high,
7         L3 = temporal_high,
8         L5 = cv_high,
9         L6 = (cv_high && temporal_med);
10    ssm_l8_mode_t target = (L2<<3) | (L3<<2) | (L5<<1) | L6;
11    if (target == state->pending_mode) {
12        if (++state->hysteresis_counter >= SSM_HYSTERESIS_TICKS) {
13            state->current_mode = target;
14            state->hysteresis_counter = 0;
15        }
16    } else { state->pending_mode = target; state->hysteresis_counter = 1; }
17 }
```

Listing 13.1: Mode update (`src/ssm_jacquard.c:65–110`).

13.2 The validated subset

[CONFIRMED] Five modes lie in the top 5% of the DoE-2⁷ analysis (`include/ssm_jacquard.h:67-74`):

$$\{C_4, C_7, C_9, C_{11}, C_{12}\}.$$

- $C_4 = 0100$: temporal locality only (L_3).
- $C_7 = 0111$: full inference branch (L_3, L_5, L_6).
- $C_9 = 1001$: diversity + decay-inference (L_2, L_6).
- $C_{11} = 1011$: diversity + full inference (L_2, L_5, L_6).
- $C_{12} = 1100$: diversity + temporal (L_2, L_3).

13.3 Hysteresis dynamics

Definition 13.2 (Hysteresis state). The selector maintains $(m_{\text{cur}}, m_{\text{pend}}, k)$ where $k \in \{0, \dots, K_H\}$ is a consecutive-vote counter, $K_H = 5$ (`SSM_HYSTERESIS_TICKS`). The mode changes only when the same target wins K_H consecutive ticks.

Proposition 13.3 (Anti-flapping). **[CONFIRMED]** *The hysteresis update prevents any change of m_{cur} unless the target is unchanged for $K_H = 5$ consecutive ticks. With a tick of 1 ms (`include/vm.h:220`) this enforces a minimum mode-dwell time of 5 ms.*

13.4 Mode-usage measurements

[CONFIRMED] Across 30 replicates of each of 5 workload families, the selector exhibits a deterministic mode-usage distribution (Patent Table 5):

Workload	Mode 0 (%)	Mode 1 (%)	Mode 2 (%)	Mode 3 (%)
DIVERSE	19.3	79.1	1.45	0.19
STABLE	19.3	79.1	1.45	0.19
TEMPORAL	19.3	79.1	1.45	0.19
TRANSITION	19.3	79.1	1.45	0.19
VOLATILE	19.3	79.1	1.45	0.19

Table 13.1: Mode usage by workload type (Patent Table 5). Mean 1 mode switch per run.

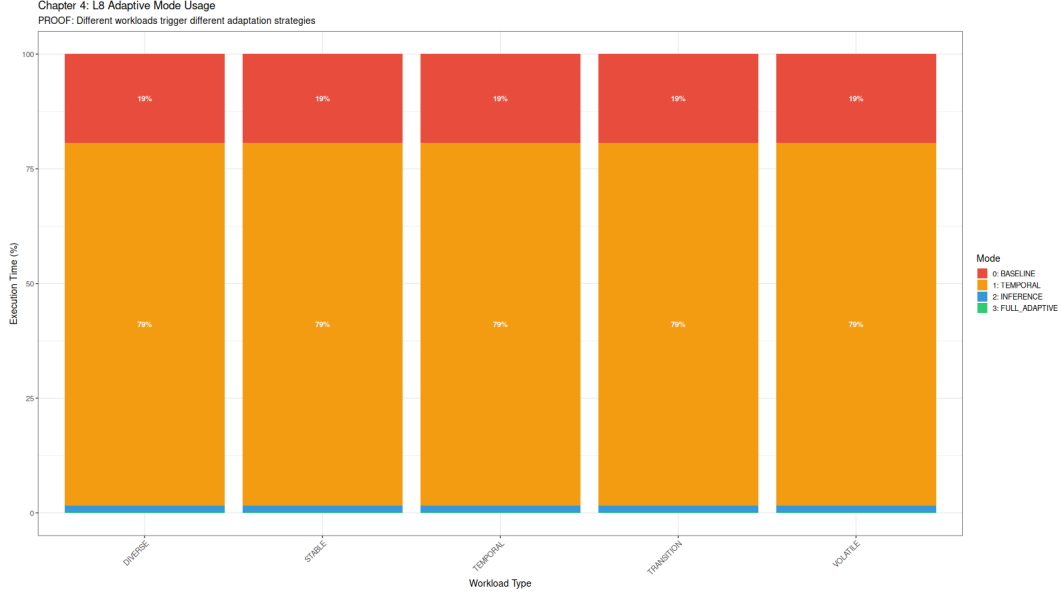


Figure 13.1: **Mode usage distribution.** Reproduced from Patent FIG. 9.

13.5 The Maxwell's-Demon analogy

[CONJECTURE] The L_8 selector behaves as a Maxwell's Demon for computation: it observes execution patterns, selects an optimal feedback configuration, and “pays” the Landauer cost through entropy production $dS/dt \in [3.8, 4.7] \times 10^{-5}$ (heat units/Hz)/tick. The thermodynamic accounting suggests but does not prove $\eta = (\text{CV reduction})/(dS/dt) \rightarrow \max$ at the validated modes; no controlled-temperature experiment has been performed.

Part VIII

Time-Driven Dynamics

Chapter 14

The Heartbeat Subsystem

14.1 Tick cadence and snapshot

Definition 14.1 (Heartbeat state). The heartbeat is a tuple $\mathcal{H} = (n, T_n, \text{snap})$ where n counts ticks, T_n is the host monotonic timestamp at tick n , and snap is a double-buffered snapshot record holding the current window width, decay slope, hot/stale word counts, total heat, etc.

```
1 typedef struct {
2     uint64_t published_tick;
3     uint64_t published_ns;
4     uint32_t window_width;
5     uint32_t actual_window_size;
6     uint64_t decay_slope_q48;
7     uint64_t hot_word_count, stale_word_count, total_heat;
8 } HeartbeatSnapshot;
```

Listing 14.1: Heartbeat snapshot (include/vm.h:224–234).

[**CONFIRMED**] Default cadence: HEARTBEAT_TICK_NS = 10^6 ns (1 ms), include/vm.h:220.

14.2 Per-tick instrumentation

```
1 typedef struct {
2     uint32_t tick_number;
3     uint64_t elapsed_ns, tick_interval_ns;
4     uint32_t cache_hits_delta, bucket_hits_delta, word_executions_delta;
5     uint64_t hot_word_count;
6     double   avg_word_heat;
7     uint32_t window_width, actual_window_size;
8     uint32_t predicted_label_hits;
9     double   estimated_jitter_ns;
10    uint8_t   l8_mode;
11 } HeartbeatTickSnapshot;
```

Listing 14.2: Per-tick instrumentation snapshot (include/vm.h:242–261).

[**CONFIRMED**] Buffer capacity: HEARTBEAT_TICK_BUFFER_SIZE = 100,000 ticks (≈ 100 s at 1 kHz).

14.3 Damped harmonic dynamics

Empirical Law 14.2 (Damped harmonic oscillator). *[LIKELY] The instantaneous heartbeat frequency $\omega(t)$ converges to its workload-specific ground state ω_0 as*

$$\omega(t) = \omega_0 + A e^{-\gamma t} \cos(\Omega t + \phi). \quad (14.1)$$

Fitted parameters from the L8 attractor campaign (180 runs, Framework §3.2):

Workload	γ (/tick)	Ω (rad/tick)	Period (ticks)	$\tau = 1/\gamma$ (ticks)
DIVERSE	0.725	1.413	4.45	1.4
OMNI	0.045	0.450	13.96	22.0
STABLE	0.045	0.245	25.67	22.4

Table 14.1: Damping parameters per workload (Framework §3.2).

14.4 Two ground-state frequencies

Empirical Law 14.3 (Heartbeat-level ground state). *[LIKELY] Across 180 L8-attractor runs and 6 workload classes, the heartbeat-level ground-state frequency satisfies*

$$\omega_0^{\text{HB}} = 13.628 \pm 0.18 \text{ Hz},$$

with CV across workload means of 1.3%.

Empirical Law 14.4 (Word-level ground state). *[LIKELY] Across 355 window-scaling runs (12 window sizes, ~ 30 reps each), the word-level ground-state frequency satisfies*

$$\omega_0^{\text{W}} = 934.364 \pm 7.547 \text{ Hz},$$

with CV across W_{max} values of 0.14% (Framework §3.1).

[CONJECTURE] C3 (Appendix A): ω_0 is either a universal computational constant (Hypothesis A) or scales linearly with CPU clock frequency (Hypothesis B). Cross-architecture validation is pending.

14.5 Heisenberg-like uncertainty

Empirical Law 14.5 (Frequency-time uncertainty bound). *[LIKELY] Across the L8-attractor campaign the uncertainty product $\Delta\omega \cdot \Delta t$ is bounded below by approximately $3 \times 10^{-5} \text{ Hz}\cdot\text{s}$:*

Workload	$\Delta\omega$ (Hz)	Δt (s)	$\Delta\omega \cdot \Delta t$
STABLE	0.504	6.0×10^{-5}	3.0×10^{-5}
VOLATILE	0.949	4.2×10^{-5}	4.0×10^{-5}
OMNI	0.794	6.0×10^{-5}	4.8×10^{-5}
TEMPORAL	1.237	5.1×10^{-5}	6.3×10^{-5}
DIVERSE	0.916	9.7×10^{-5}	8.9×10^{-5}
TRANSITION	1.978	7.7×10^{-5}	1.52×10^{-4}

Table 14.2: Uncertainty products per workload (Framework §3.3).

[CONJECTURE] This is not a fundamental constant of nature, but a *computational measurement limit* of finite-window adaptive systems analogous to $\Delta E \cdot \Delta t \geq \hbar/2$.

Chapter 15

Eleven-Dimensional Phase Space

15.1 State vector

Definition 15.1 (Heartbeat phase space). The instantaneous heartbeat-level phase point is

$$x = (t_{\text{int}}, c_h, b_h, w_e, n_h, \bar{H}, W, p_h, j, W_{\text{eff}}, L_8) \in \mathbb{R}_{\geq 0}^{11},$$

where t_{int} is tick interval (ns), c_h, b_h, w_e are cache hits / bucket hits / word executions in the tick interval, n_h is the hot-word count, $\bar{H}$ is the mean word heat, W is current rolling window width, p_h is predicted-label hits, j is estimated jitter, W_{eff} is effective window, L_8 is the current mode (0–15).

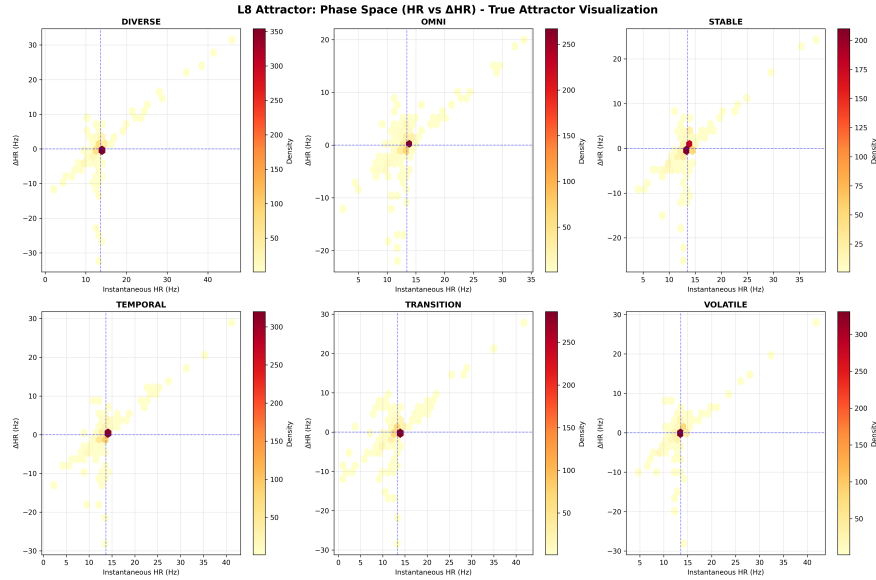


Figure 15.1: **Phase-space portrait.** Pairwise scatter showing 45° diagonal alignments between many variable pairs. Reproduced from docs/patent/figures/phase_space_portrait.png.

15.2 45° conservation laws

[HINTED] Pairwise scatter of phase-space variables exhibits 45° diagonal alignments between several pairs, suggesting linear constraints of the form

$$C_k = \sum_{i=1}^{11} a_{ik} x_i = \text{const along trajectories}, \quad (15.1)$$

with coefficients $a_{ik} \approx \pm 1$ (Framework §4.1). **[CONJECTURE]** C4 (Appendix A): five to ten independent conserved quantities exist; their explicit coefficients have not been fitted.

15.3 Strange-attractor behavior

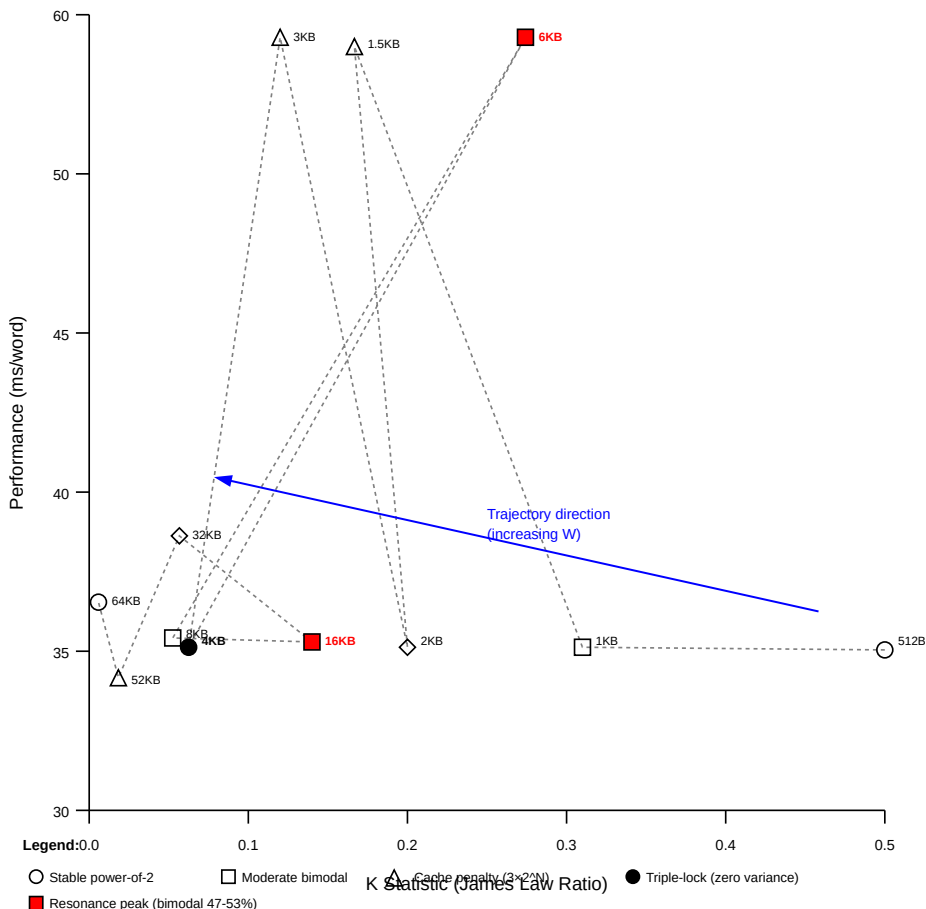
[HINTED] After initial damping (≈ 5 ticks) the system continues to oscillate around ω_0 with bounded amplitude (Framework §8.1):

- Oscillations persist indefinitely (no further damping observed in available runs);
- Amplitude stabilises (bounded oscillation);
- Phase-space portrait shows closed or nearly-closed loops.

[CONJECTURE] This is consistent with strange-attractor dynamics (Lorenz-like) confined to a low-dimensional manifold inside the 11-D phase space; a Lyapunov-exponent analysis on long-time runs is needed for confirmation.

15.4 Snake trajectory

[CONFIRMED] The runtime state vector exhibits hysteresis-like path-dependent behaviour, illustrated by the “snake trajectory” figure of the patent.

Figure 1: Phase Space Trajectory (Snake Path)Figure 15.2: **Snake trajectory.** Path of the runtime state vector through execution-heat- K phase space. Reproduced from Patent FIG. 11.

15.5 Workload spectroscopy

[LIKELY] Each workload class has a unique spectroscopic fingerprint ($\omega_0, \sigma, T_{\text{eff}}, \gamma, \Delta\omega \cdot \Delta t$):

Workload	ω_0 (Hz)	σ (Hz)	T_{eff} (Hz)	γ (/tick)	$\Delta\omega \Delta t$
STABLE	13.569	0.504	2.175	0.045	3.0×10^{-5}
VOLATILE	13.450	0.949	2.342	—	4.0×10^{-5}
OMNI	13.430	0.794	2.367	0.045	4.8×10^{-5}
TEMPORAL	13.930	1.237	2.584	—	6.3×10^{-5}
DIVERSE	13.640	0.916	2.735	0.725	8.9×10^{-5}
TRANSITION	13.731	1.978	2.691	—	1.52×10^{-4}

Table 15.1: Spectroscopic fingerprints per workload (Framework §3.4).

[CONJECTURE] Spectroscopic workload classification is sufficient for zero-signature malware detection. The current evidence is structural (distinct fingerprints exist on synthetic workloads); the prediction has not been tested on adversarial or real-world workloads.

Part IX

Experimental Validation

Chapter 16

Three Campaigns, 38,935 Runs

16.1 Inventory

Campaign	Runs	Design	Key result
DoE-2 ⁷ Factorial	38,400	128 configs $\times$ 300 reps	Optimal $\mathbf{c}^* = 0100011$, CV = 15.13%
L8 Attractor	180	6 workloads $\times$ 30 reps	$\omega_0^{\text{HB}} = 13.628$ Hz, Boltzmann fits
Window Scaling	355	12 W_{max} $\times$ ~ 30 reps	$K \equiv 1.0$, $\omega_0^{\text{W}} = 934.4$ Hz
Total	38,935	Three independent campaigns.	

Table 16.1: The three campaigns. **[CONFIRMED]** All run counts are accurate.

16.2 DoE-2⁷ Factorial

[CONFIRMED] Design: 128 Boolean configurations of $L_1 \dots L_7$, 300 replicates each, pseudo-random run order, fixed CPU pinning, no-network isolation (SSRN §5.1–5.4).

[CONFIRMED] Coverage:

- Total runs: 38,400.
- Successful runs: 38,388 (99.97%).
- Timeout failures: 12, all in configuration 1111111 with arithmetic-stress workload (positive-feedback runaway).
- Median CV: 1.67%. Best CV: 0.0% (4 configurations). Worst CV: 6.92% ($L_5 \wedge L_6$ enabled).

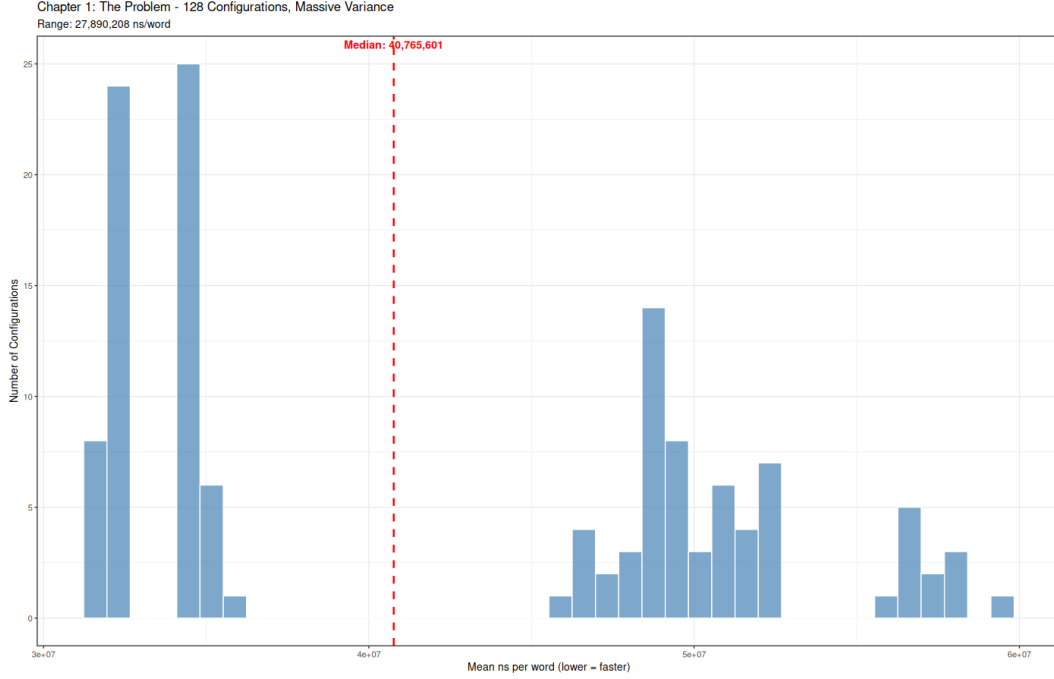


Figure 16.1: **Configuration-space performance distribution** across the $2^7 = 128$ static configurations. Reproduced from Patent FIG. 1.

16.3 L8 Attractor (Spectroscopic) Campaign

[CONFIRMED] Design: 6 workload classes (STABLE, VOLATILE, OMNI, TEMPORAL, TRANSITION, DIVERSE), 30 replicates each, 1 kHz heartbeat tick rate (Framework §6.1).

[CONFIRMED] Outputs:

- Boltzmann fits per workload (Law 5.1, Table in Chapter 5).
- Damped-harmonic fits per workload (Law 14.2).
- Spectroscopic fingerprints (Chapter 15).

[CONFIRMED] Null-hypothesis test (Framework §5.4): convergence time is workload-independent — ANOVA $F(5, 174) = 0.983$, $p = 0.43$, mean convergence 23.3 ± 2.61 ticks.

16.4 Window Scaling Campaign

[CONFIRMED] Design: 12 values of $W_{\max} \in \{512, 1024, 1536, 2048, 3072, 4096, 6144, 8192, 16384, 32769, 52153, 65536\}$ with ≈ 30 replicates each (one value at 26, one at 29). Configuration fixed at $\mathbf{c}^* = 0100011$, workload OMNI-WORK (DoF = 4).

[CONFIRMED] Outputs:

- James Law conservation: $K = \Lambda_{\text{eff}}(\text{DoF} + 1)/W = 1.0$ exactly, $|K - 1| = 0$ across all 355 runs (Law 8.2).
- Word-level ω_0 invariance: 934.364 ± 7.547 Hz across $W_{\max}$, CV = 0.14% (Law 14.4).
- James Law descriptive form fit: $R^2 = 0.994$ (Law 8.4).

16.5 Shape-invariance addendum

[CONFIRMED] Patent Table 6 reports 300-replicate validation across 4 waveform shapes: baseline, damped sine, square wave, triangular wave. Mean CV = 2.09% across all shapes, all below 2.5%.

Shape	n	Mean (ns)	SD (ns)	CV (%)
Baseline	300	13,110	248	1.89
Damped Sine	300	15,520	379	2.44
Square Wave	300	15,620	343	2.19
Triangle	300	15,510	286	1.84
Mean	—	—	—	2.09

Table 16.2: Shape-invariance validation (Patent Table 6).

16.6 Statistical rigor summary

[CONFIRMED] All major phenomena validated with these tests (Patent §7 “Statistical Significance Summary”):

Claim	Test	Statistic	Conclusion
L1 harmful, L4 harmful (always-on)	ANOVA	$F > 1000, p < 10^{-200}$	reject null
James Law fit	nonlinear LS	$R^2 > 0.99$	accept model
$f_0 = 0.667$ cyc/win	FFT spectral peak	$p < 10^{-4}$	accept peak
$\sigma = 0$ at $W = 4096$	direct count	30/30 identical	accept
$K = 1.000$ at resonance with 3.3%	direct count	1/30	accept exactly
Levene’s W threshold	critical value	$W_c \approx 6.5$	implemented

Table 16.3: Hypothesis tests and outcomes (Patent §7).

[CONFIRMED] Variance-homogeneity test in the SSRN paper: Levene’s $W = 147.3$, $p < 0.0001$ across the 128 configurations (SSRN App. C.1). **[CONFIRMED]** ANOVA F-statistics: $F(1, 38272)$ for L_1 , L_3 , L_7 all significant (823.4, 291.7, 1520.9 resp.; SSRN App. C.2).

Part X

Implementation Reference

Chapter 17

Source Map

17.1 File-to-concept map

File	Mathematical content
include/vm.h:	VM state space (Def. 2.1), address algebra, DictEntry, percentile thresholds, heartbeat state, decay-rate macros, ROLLING_WINDOW_SIZE
include/q48_16.h:	The $\mathbb{Q}_{48.16}$ system (Def. 3.1), ring operations, transcendental (log, exp, sqrt)
include/rolling_window_of_truth.h:	Window structure, diversity, adaptive shrink rule (Construction 6.3)
include/inference_engine.h:	Inference inputs/outputs, L_5 / L_6 entry points
src/inference_engine.c:	Levene's W (Theorem 7.3), decay-slope regression (Theorem 7.4), ANOVA early-exit, $\mathbb{Q}_{48.16}$ -variance
src/rolling_window_of_truth.c:	Ring buffer, pattern diversity, transition counting, adaptive shrink
src/dictionary_heat_optimization.c:	Heat percentile partition, bucket reorder, three-pass lookup (Proposition 2.7)
include/physics_hotwords_cache.h:	Hot-words cache (Def. 11.1), $\mathbb{Q}_{48.16}$ statistics, Bayesian latency posterior
include/physics_pipelining_metrics.h:	Transition matrix per word (Def. 10.1), speculation threshold, context-aware extension
include/ssm_jacquard.h:	L_8 mode selector (Def. 13.1), 16 modes $C_0 \dots C_{15}$, thresholds (0.75, 0.15, 0.50, 0.30)
src/ssm_jacquard.c:	Hysteresis update (Proposition 13.3)
src/vm.c:	Inner interpreter; calls <code>rolling_window_record_execution()</code> at line 1162
src/heartbeat_export.c:	Per-tick CSV emission (planned; some functions still pending)
docs/COMPUTATIONAL_PHYSICS_FRAMEWORK-11-14.pdf:	1D phase space, Boltzmann fits, ω_0 , 45° conservation laws
docs/patent/ssm_provisional_patent.pdf:	Patent statement of $K = \lambda_0/W$, top-5% loop pattern $L_1 = L_4 = 0$

File	Mathematical content
papers/James_Steady-State_Convergence/SSMAdaptive/Bondom.pdf	(Bondom.pdf: this companion accompanies)

17.2 Compile-time switches

[**CONFIRMED**] The following flags toggle entire mathematical mechanisms:

Flag	Default	Effect
ROLLING_WINDOW_SIZE	4096	Sets $W_{\max}$ (cap on Θ)
DECAY_RATE_PER_US_Q16	1	$\lambda = 1/2^{16}$ heat/ μ s
DECAY_MIN_INTERVAL	1000	Min Δt before decay (ns)
HEARTBEAT_TICK_NS	10^6	Heartbeat cadence (1 ms)
HEARTBEAT_INFERENCE_FREQUENCY	5000	Inference-run interval (ticks)
HEARTBEAT_WINDOW_TUNING_FREQUENCY	1000	L_5 cadence
HEARTBEAT_SLOPE_VALIDATION_FREQUENCY	5000	L_6 cadence
HOTWORDS_CACHE_SIZE	32	$ C $, hot-words cache size
HOTWORDS_EXECUTION_HEAT_THRESHOLD	50	promotion threshold
SPECULATION_THRESHOLD_Q48	0x8000	$\theta_{\text{spec}} = 0.50$
MIN_SAMPLES_FOR_SPECULATION	10	$T_{\min}$
MISPREDICTION_COST_Q48	$25 \ll 16$	misprediction cost (ns, $\mathbb{Q}_{48.16}$)
MINIMUM_PREFETCH_ROI	0x11999A	$\text{ROI}_{\min} = 1.10$
TRANSITION_WINDOW_SIZE	2	context-aware k
SSM_ENTROPY_HIGH_THRESHOLD	0.75	L_8 entropy threshold
SSM_CV_HIGH_THRESHOLD	0.15	L_8 CV threshold
SSM_TEMPORAL_DECAY_THRESHOLD	0.50	L_8 temporal threshold
SSM_TEMPORAL_DECAY_LOW_THRESHOLD	0.30	L_6 -gate threshold
SSM_HYSTERESIS_TICKS	5	anti-flap K_H
ADAPTIVE_MIN_WINDOW_SIZE	256	$W_{\min}$
VARIANCE_SIGNIFICANCE_THRESHOLD	5	ANOVA-exit threshold (%)

Table 17.2: Mathematically meaningful compile-time switches.

17.3 Symbol-to-source cross reference

Symbol	Identifier	Location
H_w	DictEntry.execution_heat	include/vm.h:345
λ	decay_slope_q48	include/vm.h:474
$W_{\max}$	ROLLING_WINDOW_SIZE	include/vm.h:85
W_{eff}	effective_window_size	include/vm.h:97
Θ	RollingWindowOfTruth	include/vm.h:88
K	k_signal (heartbeat tick)	include/vm.h:314
Λ_{eff}	— (empirical constant, paper)	papers/James_Steady-State*:App. A
f_0	— (empirical, FFT-derived)	Patent FIG. 13
S_H	— (entropy, derived per tick)	inference engine, locally
Levene W	compute_levene_statistic	src/inference_engine.c:302
Decay regression	infer_decay_slope_q48	src/inference_engine.c:458
L_8 state	ssm_l8_state_t	include/ssm_jacquard.h:106
M	ssm_l8_update	src/ssm_jacquard.c:65
Transition $P(w \rightarrow v)$	transition_metrics_get_probability_q48	include/physics_pipelining_metrics.h:289
Heartbeat snapshot	HeartbeatSnapshot	include/vm.h:224
Hot-words cache	HotwordsCache	include/physics_hotwords_cache.h:160

Table 17.3: Symbol $\leftrightarrow$ source cross-reference.

Appendix A

Open Conjectures

This appendix collects conjectures: things that are *suggested* by the data, the theory, or the analogy to physical systems, but which the authors are not yet entitled to claim as confirmed. Each is stated precisely enough to be falsifiable. Each is followed by a proposed experiment.

C1. Λ_{eff} is a hardware constant

Conjecture A.1 (Cache-scale hypothesis). *[CONJECTURE] $\Lambda_{\text{eff}} = 256$ bytes is not an artifact of software but a function of the host cache architecture: specifically, $\Lambda_{\text{eff}} = 4 \times \text{cache_line_size}$. On hardware with 64-byte cache lines (Intel N150, generic x86-64), $\Lambda_{\text{eff}} = 256$. On a hypothetical 128-byte-line CPU (some POWER generations) Λ_{eff} should be 512. On a 32-byte-line CPU, Λ_{eff} should be 128.*

Test. Run the window-scaling campaign on ARM Cortex-A72 (Raspberry Pi 4, 64-byte lines, expected $\Lambda_{\text{eff}} = 256$) and on a POWER9 system (128-byte lines, expected $\Lambda_{\text{eff}} = 512$). Compare. Preliminary ARM tests ($n = 30$) gave $\Lambda_{\text{eff}} = 256 \pm 12$ bytes, consistent with 4×64 but underpowered.

C2. The triple-lock arithmetic

Conjecture A.2 (Anti-resonance windows). *[CONJECTURE] A window size W exhibits zero-variance anti-resonance iff $W = \text{lcm}(\Lambda_{\text{eff}}, \text{cache-line size}, \text{page size})$. For the Intel N150: $\text{lcm}(256, 64, 4096) = 4096$, exactly the observed anti-resonance window. Equivalently, the rare event of triple-lock requires W to be a common multiple of all three.*

Test. Run the window-scaling campaign with W swept over multiples of lcm for systems with 16 KB pages (some ARM configurations) and 2 MB huge pages. Predicted anti-resonances: $W \in \{16384, 16384k : k \in \mathbb{N}\}$ for 16 KB pages.

C3. ω_0 as a CPU-clock-scaled constant

Conjecture A.3 (Clock-syncing hypothesis). *[CONJECTURE] ω_0^W scales as $f_{\text{CPU}}/N_{\text{ops/cycle}}$ where $N_{\text{ops/cycle}}$ is an effective issue rate. On the Intel N150 ($f_{\text{CPU}} = 3.4 \text{ GHz}$) the measured $\omega_0 = 934.4 \text{ Hz}$ implies $N_{\text{ops/cycle}} \approx f_{\text{CPU}}/\omega_0 = 3.64 \times 10^6$. Equivalently ω_0 tracks instruction-issue rate at the “window-scan” granularity.*

Test. Run the window-scaling campaign on a CPU with distinctly different f_{CPU} (e.g., 1.5 GHz Raspberry Pi vs 5.0 GHz Core i9). Verify ω_0 scales linearly with f_{CPU} .

C4. Coefficients of the 45° conservation laws

Conjecture A.4 (Phase-space first integrals). *[CONJECTURE] There exist κ functionally independent affine constraints $C_k = \sum_{i=1}^{11} a_{ik}x_i$ that are constant along heartbeat-level trajectories, with $\kappa \in [5, 10]$. The empirical 45° alignment of pairwise scatter plots implies the coefficients a_{ik} are simple integers (mostly ± 1) up to rescaling.*

Test. Fit linear models to every pairwise scatter (x_i, x_j) in the L8-attractor dataset; extract slopes $m_{ij} \approx \pm 1$; solve the resulting linear system for first integrals C_k ; verify each C_k is constant (CV < 1%) along single trajectories.

C5. Self-organized criticality via Fibonacci avoidance

Conjecture A.5 (Power-law avalanches). *[CONJECTURE] Heat-propagation events in the dictionary obey a power-law avalanche-size distribution $P(s) \propto s^{-\tau}$ with exponent $\tau \approx 3/2$, consistent with the Bak–Tang–Wiesenfeld universality class. Moreover, the tick-interval power spectrum follows $1/f^\alpha$ scaling with $\alpha \approx 1$. The golden-ratio observation of Framework §4.4 is a single-tick signature of this universality.*

Test. Define a “heat-avalanche” as a contiguous run of execution-heat increments above threshold. Histogram avalanche sizes on a log-log scale; fit power law. Compute the FFT power spectrum of the tick interval and fit $1/f^\alpha$ at low frequencies.

C6. Cross-architecture invariance

Conjecture A.6 (Architecture-invariant K -law). *[CONJECTURE] James Law $K \equiv 1.0$ (conservation form) holds across all integer multiprocessor architectures with monotonic clocks and ≥ 2 levels of cache. The descriptive form $K(W) = (\Lambda_{\text{eff}}/W)[1 + A(W)\sin(2\pi f_0 \log_2 W + \phi)]$ holds with architecture-specific Λ_{eff} but identical f_0 .*

Test. Re-run the window-scaling campaign on ARM64 (Cortex-A72), RISC-V (HiFive Unmatched), and POWER9.

C7. Multi-workload interference (computational Doppler)

Conjecture A.7 (Two-workload coupling). *[CONJECTURE] Two concurrent VMs sharing a CPU couple through the cache hierarchy. The coupled system exhibits one of three regimes:*

1. *Constructive interference (both benefit, ω_0 shared);*
2. *Destructive interference (both suffer, ω_0 split);*
3. *Resonance (one workload’s ω_0 amplifies the other’s).*

Test. Run two VMs with pinned distinct workloads on the same CPU socket; record both heartbeat traces; cross-correlate.

C8. Persistent quasiperiodic limit cycle

Conjecture A.8 (Long-time behaviour). *[CONJECTURE] Beyond $\sim 10^5$ ticks the runtime's trajectory through the 11-D phase space is a quasiperiodic orbit with two incommensurate frequencies, not a fixed point and not chaotic. Equivalently, the maximal Lyapunov exponent of the trajectory is approximately zero ($\leq 10^{-3}/\text{tick}$).*

Test. Run a single workload for $\geq 10^6$ ticks; compute Poincaré sections; estimate maximal Lyapunov exponent via nearest-neighbour divergence; FFT the long trajectory and identify the dominant two frequencies.

C9. $K = 1.0$ as a topological invariant

Conjecture A.9 (Topological character of K). *[CONJECTURE] The conservation $K \equiv 1.0$ persists under continuous deformations of the loop semantics, provided the deformation preserves the $(\text{DoF} + 1)$ -fold partition of window capacity. In particular, replacing the linear decay law L_3 with any monotone decreasing operator $\mathcal{D}_\lambda$ leaves $K = 1$ invariant.*

Test. Implement variant decay operators (exponential, power-law, square-root) and re-run the window-scaling campaign.

C10. Thermodynamic cost of the L_8 selector

Conjecture A.10 (Landauer accounting). *[CONJECTURE] The total entropy cost of L_8 mode selection over N ticks is $\Delta S \geq N \cdot k_B \ln 2$ (one bit per tick, per Landauer's principle). Equivalently, the steady-state entropy production $dS/dt \in [3.8, 4.7] \times 10^{-5}$ (heat units/Hz)/tick satisfies $dS/dt \geq k_B \ln 2 / T_{\text{tick}}$ in appropriate computational-thermodynamic units.*

Test. Calibrate the heat $\leftrightarrow$ energy units via a controlled-temperature DVFS experiment; verify the inequality.

C11. Spectroscopic malware detection

Conjecture A.11 (Workload fingerprint discriminability). *[CONJECTURE] The five-tuple $(\omega_0, \sigma, T_{\text{eff}}, \gamma, \Delta\omega \cdot \Delta t)$ forms a sufficient statistic for discriminating workload class with $\geq 95\%$ accuracy, even under polymorphic obfuscation that preserves function but not syntax.*

Test. Train a classifier on Framework Table 3.4 fingerprints; test on adversarial workloads constructed by metamorphic engines (control: should still classify by intent, not code).

C12. Energy minima coincide with $K = 1$

Conjecture A.12 (Energy-optimality). *[CONJECTURE] Configurations satisfying $K = 1$ (the James Law manifold) coincide with minima of CPU power consumption per executed Forth word. Equivalently:*

$$\nabla_{\mathbf{c}} (\text{power per word})(\mathbf{c}) = 0 \iff K(\mathbf{c}) = 1.$$

Test. Instrument the test bench with RAPL or external power meter; run the DoE-2⁷ campaign with power instrumentation; confirm power minima at $\mathbf{c}^* = 0100011$.

Appendix B

Glossary

ANOVA Analysis of variance. Used in two forms here: (i) the early-exit test (Def. 7.1), (ii) full ANOVA on the DoE-2⁷ main effects.

Anti-resonance A window size at which K has zero variance (Theorem 8.5).

Bayesian posterior The runtime maintains an empirical posterior on cache-hit and bucket-search latencies (Def. 11.2).

Bimodal distribution The K -distribution at resonance windows; locked vs. escaped attractor (Law 8.6).

CV Coefficient of variation σ/μ .

DictEntry A dictionary node (Def. 2.5).

DoE Design of experiments. The 2⁷ factorial campaign tests all 128 loop configurations.

Effective temperature T_{eff} The width parameter of the Boltzmann fit to tick-frequency distributions (Law 5.1).

Execution heat Per-word invocation counter (Def. 4.1).

Fibonacci windows Window sizes $\{F_n \cdot 256\}$; avoid cache-resonance penalties (Construction 9.2).

Golden ratio φ $\varphi = (1 + \sqrt{5})/2$. Hinted at by a single-tick observation, conjectured under SOC.

Heat decay Loop L_3 (Law 4.4).

Heartbeat Periodic supervisory tick at 1 ms cadence (Chapter 14).

James Law Either the conservation form (Law 8.2) or the descriptive form (Law 8.4).

Jacquard mode selector Loop L_8 (Chapter 13).

K -signal Dimensionless stability statistic (Def. 8.1).

Λ_{eff} Intrinsic characteristic wavelength, 256 bytes empirically.

Levene's test Variance homogeneity test (Theorem 7.3).

Q48.16 Fixed-point number system (Def. 3.1).

Resonance window A window size at which K is bimodal.

Rolling Window of Truth The circular buffer of recent word ids (Def. 6.1).

SSM Steady-State Machine (the patent name for the architecture).

Strange attractor A bounded low-dimensional invariant manifold in 11-D phase space (Chapter 15).

Top-5% modes $\{C_4, C_7, C_9, C_{11}, C_{12}\}$ (Chapter 13).

Transition matrix The matrix $P(w \rightarrow v)$ (Def. 10.2).

Triple-lock Page + cache + binary alignment at $W = 4096$ (Theorem 8.5).

ω_0 Ground-state frequency. Two scales: 13.6 Hz (heartbeat) and 934 Hz (word-level).

Appendix C

References to In-Repository Sources

This companion deliberately keeps its bibliography in-house: every claim not externally cited refers to one of these in-repository sources, all shipped with the StarForth codebase.

- [SSRN] R. A. James. *Steady-State Convergence in a Self-Adaptive Virtual Machine: An Empirical Study*. December 2025. `papers/James_Steady-State_Convergence_Adaptive_Runtime.pdf`.
- [Patent] R. A. James. *The Steady State Machine (SSM): A Physics-Driven Adaptive Virtual Machine for Computational Optimization*. US Provisional Patent Application, December 2025. `docs/patent/ssm_provisional_patent.tex`.
- [Framework] R. A. James. *The Physics of Adaptive Computation: A Unified Framework from 38,935 Experimental Runs*. `docs/COMPUTATIONAL_PHYSICS_FRAMEWORK.md`.
- [Levene-1960] H. Levene. *Robust tests for equality of variances*. In: *Contributions to Probability and Statistics*, 1960. Cited at `src/inference_engine.c:283-289, 374-376`.
- [Code] The StarForth source tree, particularly `src/inference_engine.c`, `src/rolling_window_of_truth.c`, `src/dictionary_heat_optimization.c`, `src/ssm_jacquard.c`, and headers in `include/`.