The Quotient Chain: $E \to \mathbb{N} \to Q$ at Every Operation

Claude and MJC

11 February 2026

Abstract

We derive the complete quotient decomposition of the Universal Model's forward pass. Every operation in the UM—input lookup, pattern accumulation, binary-ES softmax, recurrent propagation, output prediction—is a quotient: a ratio of microstate counts that measures how much an event narrows the space of consistent dataset positions. The chain $E \to \mathbb{N} \to Q$ maps events to counts to quotients at each step. We show that the model's bpc decomposes exactly as bpc $= \frac{1}{N}\sum_t \log_2 Q_{\text{out}}(t)$, that each $Q_{\text{out}}(t)$ factors through the quotient chain as a product of per-layer contributions, and that the recurrent quotient through W_h compounds multiplicatively—explaining why W_h is the bottleneck for both quantization (export gap) and generalization (BPTT-50 horizon). The quotient framework unifies the microstate/macrostate picture with the operational semantics of prediction.

1 The Three-Step Map

Every operation in the UM forward pass factors into three steps:

Definition 1 ($E \to \mathbb{N} \to Q$)**.** *1. E: **Identify the event.** Which event space, which specific event within it.*

*2. $\mathbb{N}$: **Count it.** How many dataset positions are consistent with this event. The count $c(e) \in \mathbb{N}$.*

*3. Q: **Compute the quotient.** $Q(e) = N/c(e) = \lambda(e)$, the luck of the event. How much the event narrows the microstate space.*

The quotient Q is the fundamental quantity: it measures the *information content* of an event. $Q = 1$ means the event tells us nothing (all N positions are consistent). $Q = N$ means the event specifies a unique position. $\log_2 Q$ is the information in bits.

2 Layer-by-Layer Quotient Derivation

We trace $E \to \mathbb{N} \to Q$ through each layer of the sat-rnn's forward pass, using the isomorphic UM representation.

2.1 Layer 0: The input quotient

At position t, the input byte x_t is observed.

$$E: \quad e = x_t \in E_{\text{in}} = \{0, \ldots, 255\} \tag{1}$$

$$\mathbb{N}: \quad c(x_t) = |\{s : D[s] = x_t\}| \tag{2}$$

$$Q: \quad Q_{\text{in}}(t) = \frac{N}{c(x_t)} = \lambda(x_t) \tag{3}$$

The input quotient partitions $N = 1024$ positions by byte identity. For the sat-rnn's dataset (first 1024 bytes of enwik9, 52 distinct bytes):

Byte	c	$Q = N/c$	$\log_2 Q$ (bits)
space	127	8.06	3.01
e	107	9.57	3.26
<	47	21.79	4.44
3	3	341.3	8.41
z	1	1024	10.00

The marginal entropy is:

$$H_0 = \frac{1}{N} \sum_t \log_2 Q_{\text{in}}(t) = \sum_b f(b) \log_2 \frac{1}{f(b)} = 4.74 \text{ bpc} \tag{4}$$

This is the bpc achievable with no model at all—just the byte frequency distribution.

2.2 Layer 1: The W_x pattern quotient

Each W_x pattern connects an input event to a hidden event. Pattern p_{jb}: "input byte b activates neuron j."

$$E: \quad p_{jb} = (x_t = b) \to h_j^{\pm} \tag{5}$$

$$\mathbb{N}: \quad s_{jb} = |W_x[j, b]| \cdot \frac{2}{\ln 2} \quad \text{(the SN strength in the doubled-E mapping)} \tag{6}$$

$$Q: \quad Q_{jb} = 2^{s_{jb}} \quad \text{(the evidence ratio)} \tag{7}$$

The pattern strength s_{jb} is the log-evidence: how many bits of support the input event b provides for hidden event h_j^+ (if $W_x[j, b] > 0$) or h_j^- (if $W_x[j, b] < 0$).

The accumulator difference for neuron j after the W_x contribution:

$$D_j^{(x)} = \sum_b s_{jb} \cdot \mathbf{1}[x_t = b] = s_{j,x_t} \tag{8}$$

Only the pattern for the actual input byte fires. The quotient at this stage:

$$Q_j^{(x)} = 2^{|D_j^{(x)}|} = 2^{|s_{j,x_t}|} \tag{9}$$

This is the evidence ratio for neuron j from the input alone. When $|W_x[j, x_t]|$ is large, the input strongly determines the hidden event; when small, the input barely influences neuron j.

2.3 Layer 2: The W_h recurrent quotient

The recurrent patterns connect hidden events at time $t-1$ to hidden events at time t. Pattern $p^{(h)}_{kj}$: "$h^{\pm}_j$ at $t-1$ implies $h^{\pm}_k$ at t."

$$E: \quad p^{(h)}_{kj} = (h^{\pm}_j, t-1) \to (h^{\pm}_k, t) \tag{10}$$

$$\mathbb{N}: \quad s^{(h)}_{kj} = |W_h[k,j]| \cdot \frac{2}{\ln 2} \tag{11}$$

$$Q: \quad Q^{(h)}_{kj} = 2^{s^{(h)}_{kj} \cdot |h_j(t-1)|} \tag{12}$$

The recurrent contribution to neuron k:

$$D^{(h)}_k = \sum_{j=0}^{127} \text{sign}(W_h[k,j]) \cdot s^{(h)}_{kj} \cdot h_j(t-1) \tag{13}$$

In the Boolean regime ($h_j \approx \pm 1$), each recurrent pattern either fires at full strength (if h_j matches the pattern's sign) or reverse fires (if h_j has the opposite sign). The cumulative recurrent quotient for neuron k:

$$Q^{(h)}_k = 2^{|D^{(h)}_k|} \tag{14}$$

Observation 1 (Recurrent quotients compound)**.** *Over d timesteps, the recurrent quotient for information entering through neuron j at time $t-d$ and reaching neuron k at time t is:*

$$Q^{(d)}_{k \leftarrow j} = \prod_{s=0}^{d-1} Q^{(h)}_{j_s, j_{s+1}} \tag{15}$$

where $j_0 = j$, $j_d = k$, and the product runs over the chain of neurons connecting them. This product is multiplicative in Q, additive in $\log Q$:

$$\log_2 Q^{(d)}_{k \leftarrow j} = \sum_{s=0}^{d-1} s^{(h)}_{j_{s+1}, j_s} \cdot |h_{j_s}(t-d+s)| \tag{16}$$

This is why W_h quantization is the bottleneck: each rounding error δs in a recurrent pattern strength contributes additively to $\log Q$, so over d steps the total error is $\sim d \cdot \delta s$. But Q is multiplicative, so the error in Q grows *exponentially* with d.

2.4 Layer 3: The bias quotient

The bias $b_h[j]$ is an always-on pattern: it fires regardless of input or previous hidden state.

$$E: \quad p^{(b)}_j = \text{"always"} \to h^{\pm}_j \tag{17}$$

$$\mathbb{N}: \quad s^{(b)}_j = |b_h[j]| \cdot \frac{2}{\ln 2} \tag{18}$$

$$Q: \quad Q^{(b)}_j = 2^{s^{(b)}_j} \tag{19}$$

The bias sets the *prior* quotient: before seeing any input or recurrent evidence, how much does neuron j favor h^+_j over h^-_j? The bias quotient is the default odds ratio.

2.5 Layer 4: The binary-ES softmax (hidden quotient)

The total accumulator difference for neuron j at time t:

$$D_j(t) = D_j^{(x)} + D_j^{(h)} + s_j^{(b)} \tag{20}$$

The binary-ES softmax computes:

$$P(h_j^+) = \sigma(D_j \ln 2) = \frac{1}{1 + 2^{-D_j}} \tag{21}$$

Definition 2 (Hidden quotient)**.** *The hidden quotient for neuron j at position t:*

$$Q_j^{(hid)}(t) = \frac{1}{\max(P(h_j^+), P(h_j^-))} = \frac{1 + 2^{-|D_j|}}{1} \in [1, 2] \tag{22}$$

When $|D_j| \gg 0$ (saturated), $Q_j^{(\text{hid})} \to 1$: the hidden event is determined with certainty, no information is lost. When $D_j = 0$ (maximally uncertain), $Q_j^{(\text{hid})} = 2$: a full bit of information is unresolved.

Proposition 1 (Cumulative hidden quotient)**.** *The cumulative hidden quotient across all 128 neurons:*

$$Q_H(t) = \prod_{j=0}^{127} Q_j^{(hid)}(t) \tag{23}$$

measures the total information loss at the hidden bottleneck. For the sat-rnn (mean $|D_j| = 60.5$):

$$\log_2 Q_H \approx 128 \cdot \log_2(1 + 2^{-60.5}) \approx 128 \cdot 2^{-60.5} \approx 10^{-16} \text{ bits} \tag{24}$$

The hidden quotient is negligible: the saturated neurons carry essentially zero information loss.

This confirms that the hidden layer in the sat-rnn is Boolean: the quotient through the hidden softmax is effectively 1 (no uncertainty). The information loss happens elsewhere—in the W_y projection and in the pattern inventory's incompleteness.

2.6 Layer 5: The W_y output accumulation

Each W_y pattern connects a hidden event to an output event. Pattern $p_{oj}^{(y)}$: "$h_j^{\pm}$ predicts output byte o."

$$E: \quad p_{oj}^{(y)} = h_j^{\pm} \to (y = o) \tag{25}$$

$$\mathbb{N}: \quad s_{oj}^{(y)} = |W_y[o, j]| \tag{26}$$

$$Q: \quad Q_{oj}^{(y)} = 2^{s_{oj}^{(y)} \cdot |h_j(t)|} \tag{27}$$

The output accumulator:

$$A(o, t) = \sum_{j=0}^{127} W_y[o, j] \cdot h_j(t) + b_y[o] \tag{28}$$

2.7 Layer 6: The output softmax (prediction quotient)

The output probability:

$$P(o \mid h_t) = \frac{e^{A(o,t)}}{\sum_{o'} e^{A(o',t)}} = \frac{2^{A(o,t)/\ln 2}}{\sum_{o'} 2^{A(o',t)/\ln 2}} \tag{29}$$

Definition 3 (Output quotient)**.**

$$Q_{out}(o,t) = \frac{1}{P(o \mid h_t)} = \frac{\sum_{o'} e^{A(o',t)}}{e^{A(o,t)}} = \frac{Z(t)}{e^{A(o,t)}} \tag{30}$$

where $Z(t) = \sum_{o'} e^{A(o',t)}$ *is the output partition function.*

The output quotient for the *actual* next byte $y_{t+1} = D[t+1]$:

$$Q_{\text{out}}(t) = Q_{\text{out}}(y_{t+1}, t) = \frac{Z(t)}{e^{A(y_{t+1},t)}} \tag{31}$$

The bpc at position t is $\log_2 Q_{\text{out}}(t)$, and the model's average bpc is:

$$\boxed{\text{bpc} = \frac{1}{N} \sum_{t=0}^{N-1} \log_2 Q_{\text{out}}(t)} \tag{32}$$

3 The Quotient Decomposition Theorem

We now derive the exact decomposition of Q_{out} into per-layer quotient contributions.

Theorem 1 (Quotient decomposition)**.** *The output quotient at position* t *decomposes as:*

$$\log_2 Q_{out}(t) = \underbrace{\log_2 Z(t) - A(y_{t+1}, t)/\ln 2}_{output\ softmax} \tag{33}$$

where the output accumulator $A(y_{t+1}, t)$ *decomposes additively:*

$$A(y,t) = \underbrace{b_y[y]}_{prior} + \sum_{j=0}^{127} W_y[y,j] \cdot \underbrace{\tanh\Big(\overbrace{W_x[j,x_t]}^{input} + \overbrace{\sum_k W_h[j,k] h_k(t-1)}^{recurrent} + \overbrace{b_h[j]}^{bias}\Big)}_{h_j(t)} \tag{34}$$

In the Boolean regime ($h_j \approx \pm 1$)*, this simplifies to:*

$$A(y,t) \approx b_y[y] + \sum_{j=0}^{127} W_y[y,j] \cdot sign(z_j(t)) \tag{35}$$

where $z_j(t) = W_x[j,x_t] + \sum_k W_h[j,k] h_k(t-1) + b_h[j]$ *is the pre-activation.*

Proof. The first equation follows from $P(y) = e^{A(y)}/Z$, hence $1/P(y) = Z/e^{A(y)}$. The additive decomposition of A follows from the definitions. The Boolean simplification follows from $\tanh(z) \approx \text{sign}(z)$ when $|z| \gg 1$ (margin > 1.0 for 98.9% of neuron-steps). □

3.1 Factoring the quotient by source

Each component of $A(y,t)$ has a quotient interpretation:

1. **Prior quotient** ($b_y[y]$):

$$Q_{\text{prior}}(y) = e^{-b_y[y]} \cdot Z_b \quad \text{where} \quad Z_b = \sum_o e^{b_y[o]} \tag{36}$$

   The bias implements the marginal distribution: $P_0(y) \propto e^{b_y[y]}$. For the pattern-chain UM at order 0: $b_y[y] \approx \ln c(y)$, giving $Q_{\text{prior}}(y) = N/c(y) = \lambda(y)$.

2. W_y **quotient** (per neuron j):

$$q_j(y,t) = e^{W_y[y,j] \cdot h_j(t)} \tag{37}$$

   Each neuron contributes a multiplicative factor to the output evidence. In Boolean regime: $q_j = e^{\pm W_y[y,j]}$. If neuron j is positive and $W_y[y,j] > 0$, the factor $q_j > 1$ (evidence for y); if $W_y[y,j] < 0$, the factor $q_j < 1$ (evidence against y).

3. **Recurrent quotient** (implicit in $h_j(t)$): The hidden state $h_j(t)$ carries the product of all recurrent quotients from the past. In the Boolean regime:

$$\text{sign}(h_j(t)) = \text{sign}\left(W_x[j, x_t] + \sum_k W_h[j,k] \cdot \text{sign}(h_k(t-1)) + b_h[j] \right) \tag{38}$$

   The recurrent quotient is discrete: a product of sign contributions that determines the Boolean state.

4 Quotient Amplification and the Export Gap

4.1 The error amplification theorem

Theorem 2 (Recurrent quotient error amplification)**.** *Let $\hat{s}_{kj} = s_{kj} + \delta_{kj}$ be a quantized recurrent pattern strength with error δ_{kj}. After d recurrent steps, the error in $\log_2 Q$ for a single-neuron chain $j_0 \to j_1 \to \cdots \to j_d$ is:*

$$\Delta \log_2 Q = \sum_{s=0}^{d-1} \delta_{j_{s+1}, j_s} \cdot |h_{j_s}(t-d+s)| \tag{39}$$

In the Boolean regime ($|h| = 1$):

$$|\Delta \log_2 Q| \le d \cdot \max_{k,j} |\delta_{kj}| \tag{40}$$

The error grows linearly in $\log Q$, hence exponentially *in Q:*

$$\frac{Q + \Delta Q}{Q} = 2^{\Delta \log_2 Q} \le 2^{d \cdot \delta_{\max}} \tag{41}$$

Proof. The log-quotient for a chain is additive over steps. Each step contributes $s_{j_{s+1}, j_s} \cdot |h_{j_s}|$ to $\log_2 Q$. Replacing s with $s + \delta$ adds $\delta \cdot |h|$ per step. Summing over d steps gives the result. □

4.2 Numerical consequences

For the sat-rnn with 8-bit SN quantization (log-scale, $c = 500$):

- Per-weight resolution: $\delta_{\max} \approx 0.002$ in SN strength.
- Per-step error: $|\Delta \log_2 Q| \leq 0.002$ per neuron per step.
- Over $d = 50$ steps (BPTT horizon): $|\Delta \log_2 Q| \leq 0.1$ bits.
- Over $d = 1024$ steps (full dataset): $|\Delta \log_2 Q| \leq 2.0$ bits.

But the error is not a single chain: each step involves 128 neurons, and the errors can constructively interfere. The RMS error over all 128 neurons:

$$\mathrm{RMS}(\Delta \log_2 Q_k) = \sqrt{128} \cdot \delta_{\max} \approx 0.023 \text{ per step} \tag{42}$$

After d steps (with partial cancellation, factor $\sqrt{d}$):

$$\mathrm{RMS}_d \approx 0.023 \cdot \sqrt{d} \tag{43}$$

At $d = 50$: RMS ≈ 0.16 bits. At $d = 1024$: RMS ≈ 0.73 bits. But this assumes independent errors; the actual dynamics can be chaotic (constructive interference at some c values, destructive at others), which explains Table 5 of the export gap paper.

5 The Quotient Chain as Information Flow

5.1 Information enters at the input

The input quotient $Q_{\mathrm{in}}(x_t) = N/c(x_t)$ sets the initial information content. For common bytes ($Q \approx 8$, ~ 3 bits), the input narrows the microstate space by a factor of 8. For rare bytes ($Q \approx 341$, ~ 8.4 bits), the narrowing is much sharper.

5.2 Information propagates through W_h

The recurrent quotient carries information from past inputs into the present hidden state. The "highway" metaphor: W_h preserves the quotient from d steps ago, subject to the amplification/decay of Theorem 2.

The effective depth of the quotient chain is bounded by the BPTT-50 training horizon. Beyond 50 steps, the recurrent quotients are uncontrolled: the weights were never optimized to preserve information at this depth. This is the origin of the "chaotic" export gap.

5.3 Information exits at the output

The output quotient $Q_{\mathrm{out}}(y, t)$ measures how much information the model has accumulated about the next byte. The ideal model has $Q_{\mathrm{out}}(y, t) = N/c(y \mid \text{context})$—the output quotient equals the conditional luck of the output event.

The *residual quotient* $Q_{\mathrm{out}}/Q_{\mathrm{ideal}}$ measures the model's inefficiency: positions where it assigns too much probability to wrong outputs or too little to the right one.

5.4 The quotient budget

Proposition 2 (Quotient budget)**.** *At each position t, the total log-quotient is:*

$$\log_2 Q_{out}(t) = \underbrace{\log_2 Q_{prior}(y_{t+1})}_{\text{marginal cost}} - \underbrace{\sum_j W_y[y_{t+1}, j] \cdot h_j(t)/\ln 2}_{\text{evidence from hidden state}} + \underbrace{\log_2 Z(t) - \log_2 Z_b}_{\text{partition function ratio}} \tag{44}$$

The model "pays" the marginal cost (how rare is the output byte?) and "earns back" evidence from the hidden state (patterns matching the output). The net cost is the bpc.

This decomposition shows that better prediction means lower Q_{out}, which means the hidden-state evidence outweighs the marginal cost. Each hidden neuron j contributes $W_y[y, j] \cdot h_j/\ln 2$ bits of evidence—the W_y quotient from Section 3.1.

6 The Quotient at Each Level of the Model Hierarchy

6.1 Marginal model: $Q = \lambda(y)$

The marginal model uses only the output prior. The quotient is the output luck: $Q(y) = N/c(y)$. Average bpc: $H_0 = 4.74$.

6.2 Bigram model: $Q = \lambda(y \mid x)$

The bigram model uses the input quotient to refine the output:

$$Q_{\text{bigram}}(y, x) = \frac{c(x)}{c(x, y)} = \lambda(y \mid x) \tag{45}$$

The quotient is the conditional luck of the output given the input. Average bpc: 2.05. The quotient reduction from marginal: a factor of $2^{4.74-2.05} = 6.4\times$ on average.

6.3 Skip-k-gram model: $Q = \lambda(y \mid x_{d_1}, \ldots, x_{d_k})$

The skip-k-gram model uses k input quotients at offsets $d_1, \ldots, d_k$:

$$Q_{\text{skip}}(y, \mathbf{x}_d) = \frac{c(\mathbf{x}_d)}{c(\mathbf{x}_d, y)} \tag{46}$$

where $\mathbf{x}_d = (x_{t-d_1}, \ldots, x_{t-d_k})$.

Model	bpc	$\mathbb{E}[\log_2 Q]$	$E \to \mathbb{N} \to Q$ path
Marginal	4.74	4.74	$y \to c(y) \to N/c(y)$
Bigram	2.05	2.05	$(x, y) \to c(x, y) \to c(x)/c(x, y)$
Skip-4 $[1, 8, 20, 3]$	0.069	0.069	$(\mathbf{x}_d, y) \to c(\mathbf{x}_d, y) \to \cdots$
Contig. order-12	0.067	0.067	as above, $k = 12$
Skip-8 (greedy)	0.043	0.043	as above, $k = 8$

At each level, the $E \to \mathbb{N} \to Q$ chain lengthens: more events → more specific count → sharper quotient → lower bpc.

6.4 RNN model: $Q = Q_{\text{out}}(y, h)$

The RNN compresses the skip-k-gram quotient through the hidden bottleneck. The quotient chain:

$$(x_t, h_{t-1}) \xrightarrow{W_x, W_h} z_t \xrightarrow{\tanh} h_t \xrightarrow{W_y} A(y,t) \xrightarrow{\text{softmax}} Q_{\text{out}}(y,t) \tag{47}$$

The RNN's quotient is an approximation of the ideal conditional quotient $\lambda(y \mid \text{full context})$, compressed through 128 binary hidden events. The approximation quality depends on how well the W_h highway preserves the input quotients from past positions.

7 The Quotient Interpretation of Training

7.1 What BPTT optimizes

BPTT minimizes the average $\log_2 Q_{\text{out}}$ (the bpc). In quotient terms:

$$\frac{\partial \text{bpc}}{\partial w_{ij}} = \frac{1}{N} \sum_t \frac{\partial \log_2 Q_{\text{out}}(t)}{\partial w_{ij}} \tag{48}$$

Each weight update adjusts the quotient chain: changing $W_h[k, j]$ modifies the recurrent quotient through neuron j, which propagates through all downstream hidden and output quotients.

7.2 Why the Hebbian rule approximates BPTT

The Hebbian update $\Delta W_h[k, j] \propto \text{cov}(h_j(t), h_k(t{+}1))$ is the first-order quotient approximation. The covariance measures: "when event h_j^+ occurs at t, how much more likely is event h_k^+ at $t{+}1$?" This is:

$$\text{cov}(h_j, h_k) = \mathbb{E}[h_j h_k] - \mathbb{E}[h_j]\mathbb{E}[h_k] \approx P(h_j^+, h_k^+) - P(h_j^+)P(h_k^+) \tag{49}$$

In quotient terms:

$$\text{cov} \propto \frac{P(h_j^+, h_k^+)}{P(h_j^+)P(h_k^+)} - 1 = \frac{Q_j \cdot Q_k}{Q_{jk}} - 1 \tag{50}$$

where $Q_{jk} = N/c(h_j^+, h_k^+)$ is the joint quotient. The Hebbian rule detects when the joint quotient Q_{jk} is less than the product of marginals $Q_j \cdot Q_k$—i.e., when the events co-occur more often than chance. This is the pointwise mutual information.

The Hebbian prediction achieves $r = 0.56$ on important W_h entries because the pairwise quotient captures the dominant recurrent relationships. The remaining variance ($1 - 0.56^2 = 69\%$) comes from higher-order quotients: triples, quadruples, and the non-linear interactions that BPTT's Jacobian chain captures but pairwise covariance misses.

7.3 Why analytic construction beats training

The analytic construction computes $W_y[o, j] = s \cdot \log(P(o|h_j^+)/P(o|h_j^-))$ directly from conditional quotients. This is exact $E \to \mathbb{N} \to Q$ at the output layer: the log-quotient of output event o conditioned on hidden event h_j^+ vs h_j^-.

The trained model finds W_y by gradient descent, which navigates a non-convex loss landscape. The analytic construction navigates the *quotient space* directly, which is convex (log-quotients add linearly). This is why 1.89 bpc (analytic) beats 4.97 bpc (trained): the quotient space has no local optima.

8 Q as the Universal Currency

Theorem 3 (Q converts between representations)**.** *Given any two representations of the same predictive model—UM patterns, RNN weights, skip-k-gram counts, or Boolean automaton states—the quotient Q provides a common currency for comparison:*

$$Q_{UM}(y \mid C) = Q_{RNN}(y \mid h) = Q_{skip}(y \mid \mathbf{x}_d) \tag{51}$$

whenever the three models make the same prediction for output y given context C (equivalently, hidden state h, or skip context $\mathbf{x}_d$).

This is the content of the isomorphism: the doubled-E construction ensures $Q_{\text{UM}} = Q_{\text{RNN}}$ exactly. The pattern-chain UM ensures $Q_{\text{skip}} \approx Q_{\text{RNN}}$ approximately (within 0.01 bpc at order 10).

Q is the natural quantity for comparing models because it measures the same thing in all representations: how much the model narrows the microstate space. A quotient of 100 means "this prediction eliminates 99% of positions." Whether the elimination is achieved by counting patterns, propagating hidden states, or looking up skip-k-gram tables, the quotient is the same.

9 Conclusion

The forward pass of the UM is a quotient chain: $E \to \mathbb{N} \to Q$ at every operation, with quotients compounding multiplicatively through the recurrent layer. The bpc is the average log-quotient at the output. The export gap is the error amplification of recurrent quotients. The Hebbian rule is the first-order quotient approximation of BPTT. The analytic construction beats training because quotient space is convex.

References

[1] Michaeljohn Clement. CMP. 2026. `https://cmpr.ai/cmp.pdf`

[2] Claude and MJC. The SN Export Gap. 7 Feb 2026.

[3] Claude and MJC. The Pattern-Chain UM. 8 Feb 2026.

[4] Claude and MJC. The Hidden Quotient. 8 Feb 2026.

[5] Claude and MJC. From Counting to Construction. 11 Feb 2026.

[6] Claude and MJC. E onto $\mathbb{N}$. 11 Feb 2026.