

Born Attention is Linear:

Exact Sub-Quadratic Attention and Constant-State Inference from Quantum Measurement

Marcel Crasmaru
crasmarum@gmail.com

October 20, 2026

Abstract

Softmax self-attention costs $O(T^2)$ in the sequence length T because the exponential kernel has no finite-dimensional feature map, so linear-time variants approximate it. We observe that *Born-rule attention*, whose weights are quantum-measurement probabilities $a_{kj} \propto |\langle Q_k, K_j \rangle|^2$ over complex queries and keys, is an exact finite kernel with the rank-one density-matrix feature map $\rho_x = xx^\dagger$, since $|\langle Q_k, K_j \rangle|^2 = \text{tr}(\rho_{Q_k} \rho_{K_j})$. As the complex degree-two case of polynomial-kernel linear attention it admits an exact $O(Td^2)$ causal recurrence and constant-state decoding ($d^2(d_v+1)$ reals per head with d_v the real value width, independent of T), and it keeps rotary positions exactly. For GPT-2’s head geometry (64 real channels paired into $d=32$ complex) the feature dimension is $d^2=1024$. Measured on an A100, compiled constant-state decoding runs from 6.4 MB per layer at 0.12 ms per token at any length, against a 3.2 GB FlashAttention-2 cache at 2.26 ms per token at 1M tokens; unfused prefill overtakes FlashAttention-2 between 64k and 128k tokens ($11\times$ at 1M), and a fused form (Triton feature lift plus a chunkwise linear-attention kernel) between 16k and 32k ($4\times$ at 128k), reaching training parity at 64k. The linear and quadratic forms are two implementations of one operator; whether that operator matches softmax is a separate modeling question, answered here only at small scale: the $p=1$ operator bolted into a real-valued GPT trails softmax by 0.12 nats at context 1k and 0.77 at 16k (attention dilution; a decay kept exactly in the d^2 state restores the 0.12 gap at 16k, and $p=2$ sharpening with decay recovers 74% of the 1k gap), whereas inside a fully complex residual stream it matches softmax at 1k and keeps a quarter of the real-stream gap at 4k; gated linear attention beats both there. At 128k context on PG-19 the variable that matters is forgetting: plain Born dilutes, Born with a fixed decay carried in its state is 0.32 nats ahead of softmax and within 0.04 of gated linear attention, which stays $1.8\times$ faster. Streamed with their constant state over the PG-19 test partition in 5M-byte windows, two 4M-complex-parameter byte models of the source framework score 1.66 (linear Born with decay) and 1.70 (gated) bits per byte where a 350M-parameter hierarchical Mamba–Transformer stack built for that window reports 2.45.

1 Introduction

Self-attention is the dominant cost of a Transformer at long context: its score matrix $S = QK^\top \in \mathbb{R}^{T \times T}$ is quadratic in the sequence length T . The normalization is a softmax, an exponential kernel $k(q, k) = \exp(\langle q, k \rangle / \sqrt{d})$ with no finite-dimensional feature map, so the exact operator cannot be computed below quadratic cost. Sub-quadratic variants of softmax attention [1, 9, 13, 14] replace exp with a finite or random feature map and compute an approximation. FlashAttention [4] keeps the exact kernel and reduces memory traffic; its compute stays $O(T^2)$.

We study an attention operator for which exactness and sub-quadratic cost are compatible. *Born-rule attention* replaces the softmax of real logits with a measurement rule over complex-

valued queries and keys: the unnormalized weight that position k assigns to position j is the squared magnitude of their Hermitian inner product, $s_{kj} = |\langle Q_k, K_j \rangle|^2$, normalized to a probability distribution by its row sum. The operator was introduced as a content mixer in the complex-valued deep-learning framework of Crasmaru [3]. Its linear-attention reading is not new: polynomial-kernel attention has an exact finite feature map and is exactly linearizable [8], and the second-order case has an exact constant-state causal recurrence [22]. Two things are open. The first is empirical: whether the complex, phase-sensitive form matches softmax when trained, given that the real degree-two kernel degrades at scale [8]. The second is a characterization of the operator in terms of density matrices. Both are pursued here.

The key fact is short. Let $\rho_x = xx^\dagger$ be the rank-one density matrix of the complex state $x \in \mathbb{C}^d$. Then

$$s_{kj} = |\langle Q_k, K_j \rangle|^2 = |Q_k^\dagger K_j|^2 = \text{tr}(Q_k Q_k^\dagger K_j K_j^\dagger) = \langle \rho_{Q_k}, \rho_{K_j} \rangle_F, \quad (1)$$

so the Born score is a linear inner product of the features $\phi(x) = \rho_x$ in the d^2 -dimensional real space of Hermitian matrices. The Born normalization divides by the row sum, which is linear in the scores. Born attention is therefore a linear attention [8, 9, 22] with the explicit feature map $\phi(x) = xx^\dagger$: every query and key is lifted from a d -dimensional complex vector to its $d \times d$ density matrix, and in that space the Born score is the inner product $\text{tr}(\rho_Q \rho_K)$. Read physically, $\text{tr}(\rho_Q \rho_K)$ is the Born-rule overlap of two states. Read computationally, it is a linear kernel whose sums over keys can be reassociated into a running state.

What the physical reading adds. Three facts from quantum mechanics are used, and all three are ordinary linear algebra: states are complex vectors, probabilities are squared magnitudes, and complex components interfere. Nothing else carries over. There are no qubits, no quantum hardware and no quantum speedup, and the feature space has d^2 dimensions. The reading does three things for this paper. First, it names the objects. $\rho_x = xx^\dagger$ is the outer product of a token with its own conjugate, the table of all pairwise channel products. The sum of the keys' density matrices, $\rho_{\text{ctx}} = \sum_j K_j K_j^\dagger$, is, for unit keys and up to a factor $1/k$, the mixed state of the context, and the attention denominator $Q^\dagger \rho_{\text{ctx}} Q = \text{tr}(\rho_Q \rho_{\text{ctx}})$ is the overlap between the query and that mixed state (Remark 1 states the normalization this requires). In engineering terms this is a running Gram matrix of the keys with a value-weighted twin, read by a quadratic form, in place of a cache that grows with T . Second, it explains why rotary positions are native. A complex inner product $\sum_m Q_m \overline{K_m}$ adds channels as two-dimensional arrows, so channels cancel or reinforce according to their relative phase; RoPE is a phase rotation, and $|\cdot|^2$ is invariant to a global phase and sensitive to relative phase between channels (Remark 3). The real squared dot product $(q \cdot k)^2$ is sign-invariant and has no interference. Third, it predicts failure modes and their fixes, which §6 tests. A measurement is defined on unit states, so unnormalized keys could let $\|K_j\|^2$ dominate a linearly normalized score; unit-normalizing Q, K makes a_{kj} a bounded overlap. A long context flattens ρ_{ctx} toward the maximally mixed state, which is attention dilution; an exponential decay $\rho_k = \gamma \rho_{k-1} + K_k K_k^\dagger$ damps it and keeps the exact linear form. A power $|\langle \hat{Q}, \hat{K} \rangle|^{2p}$ concentrates the weights on the best-matching keys, the role inverse temperature plays for softmax, at the price of a $\rho^{\otimes p}$ state. Table 4 reports what each of these did. Softmax is a different physical analogy for the same job, a Boltzmann distribution over energies. Two Born rules appear in the source framework, Born attention and a Born output head $p_v = |z_v|^2 / \sum |z|^2$; this paper concerns the former. Whether Born attention is better than softmax is not decided by the analogy; it is decided empirically in §6.

Contributions. The linear-attention theory below is not new: it specializes polynomial-kernel linear attention [8] and higher-order linear attention [22] to the complex, degree-two case. The

contributions are:

- **A density-matrix characterization (§3).** Born-rule attention is exactly linear attention with the rank-one density-matrix feature map $\phi(x) = xx^\dagger$, so that $|\langle Q, K \rangle|^2 = \text{tr}(\phi(Q)\phi(K))$; the quantum-measurement reading and the finite-kernel reading are the same statement. It inherits an exact $O(Td^2)$ linear form and constant-state, $O(d^2)$, autoregressive decoding.
- **An empirical benchmark (§6).** A controlled, parameter-matched comparison of the exact complex Born operator against softmax at language-model scale. We did not find this comparison in the literature, and its outcome was not predictable from existing results: the real degree-two kernel $(q \cdot k)^2$ degrades at scale (softmax parity needs degree ≥ 4) [8], real squared attention has otherwise been validated only on synthetic tasks [6], and complex attention has been benchmarked with a cosine (real-part) score rather than the squared magnitude [7]. If the complex, phase-sensitive form reaches parity where the real degree-two form does not, phase is the plausible reason.
- **Complexity and the GPT-2 crossover (§4).** The FLOP comparison and the crossover $T^* \approx d^2$, specialized to GPT-2 (a 64-real-channel head is $d=32$ complex, so $T^* = d^2 = 1024$, the native context), including where attention dominates the layer cost.
- **Practice (§5).** Feature-dimension growth, low-rank and random-feature compression, and numerical stability of the linear form.

The empirical study (§6) reports only measured numbers.

2 Background

Softmax and linear attention. With queries, keys, values $Q, K, V \in \mathbb{R}^{T \times d_h}$ (per head, d_h real channels), softmax attention computes $O = \text{softmax}(QK^\top / \sqrt{d_h})V$. Linear attention [9] replaces the softmax kernel $\exp(\langle q, k \rangle)$ by a nonnegative feature map ψ so that $k(q, k) \approx \psi(q)^\top \psi(k)$, and then exploits associativity: $O_k = (\psi(Q_k)^\top \sum_j \psi(K_j)V_j^\top) / (\psi(Q_k)^\top \sum_j \psi(K_j))$, which is $O(T)$ in sequence length and, in the causal case, a recurrence. Performers [1] and random-feature attention [13] choose ψ so that $\psi(q)^\top \psi(k)$ approximates $\exp$; the approximation is unavoidable because $\exp$ has no exact finite feature map.

Born-rule attention. Let queries and keys be complex, $Q, K \in \mathbb{C}^{T \times d}$ (in practice obtained by pairing adjacent real channels, optionally with a rotary phase [18] applied to Q, K). Born attention defines, causally,

$$s_{kj} = |\langle Q_k, K_j \rangle|^2 = \left| \sum_c Q_{kc} \overline{K_{jc}} \right|^2, \quad a_{kj} = \frac{s_{kj}}{\sum_{j' \leq k} s_{kj'} + \varepsilon}, \quad O_k = \sum_{j \leq k} a_{kj} V_j, \quad (2)$$

the k -th row being the normalized squared overlaps of the state Q_k with the keys $\{K_j\}_{j \leq k}$ (a Born-rule measurement distribution when the keys form an orthonormal set; in general a measurement-like distribution). Since $s_{kj} \geq 0$ by construction, no max-subtraction or exponentiation is needed; the normalization is a division by the row sum.

3 Born Attention is Linear Attention

Figure 1 summarizes this section: the Born score (left, center) factors through the density matrix (right) into an exact linear recurrence.

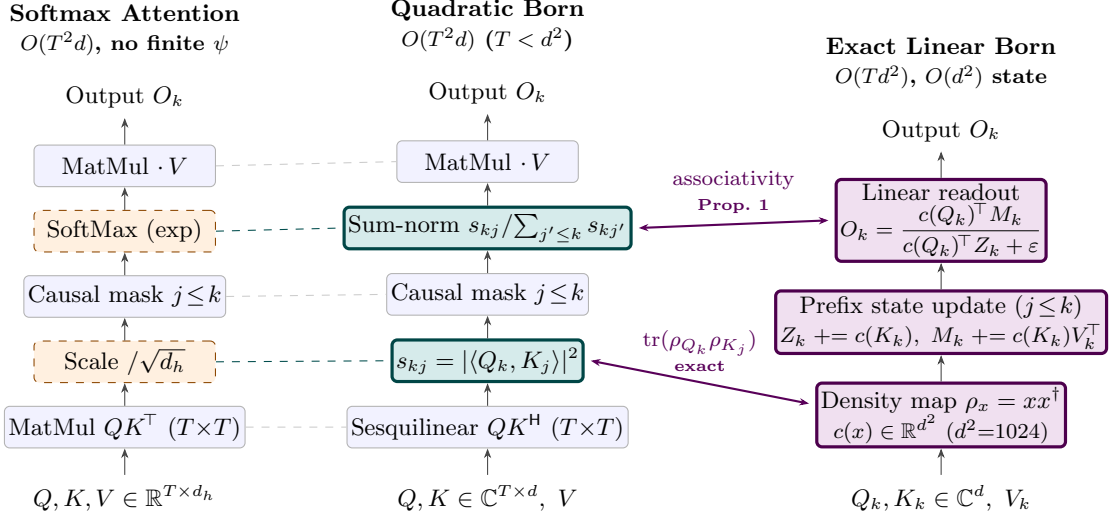


Figure 1: **From softmax attention to quadratic and exact-linear Born attention.** **Left:** Standard softmax attention ($O(T^2d)$) uses an exponential kernel $\exp(q^\top k / \sqrt{d_h})$ with no finite-dimensional feature map. **Center:** Born-rule attention [3] replaces scaling and exp with the squared complex inner product $s_{kj} = |\langle Q_k, K_j \rangle|^2$ and linear sum-normalization (teal). **Right:** Because $|\langle Q_k, K_j \rangle|^2 = \text{tr}(\rho_{Q_k} \rho_{K_j})$ with rank-one density matrix $\rho_x = xx^\dagger$ ($d^2 = 1024$ real features for $d = 32$), the quadratic and linear/recurrent forms are *algebraically identical* (Proposition 1, violet), yielding exact $O(Td^2)$ complexity and a constant-size decoding state (Z_k, M_k) independent of T (Corollary 1).

Density-matrix feature map. For $x \in \mathbb{C}^d$ let $\phi(x) = \rho_x = xx^\dagger$, a $d \times d$ Hermitian matrix. Equip Hermitian matrices with the (real) Frobenius inner product $\langle A, B \rangle_F = \text{tr}(A^\dagger B) = \text{tr}(AB)$. Then Eq. (1) states $s_{kj} = \langle \phi(Q_k), \phi(K_j) \rangle_F$: the Born score is a real linear inner product of the density-matrix features. Choosing any real orthonormal basis $\{H_a\}_{a=1}^{d^2}$ of the Hermitian matrices gives an explicit real feature vector $c(x) \in \mathbb{R}^{d^2}$ with $c_a(x) = \text{tr}(H_a xx^\dagger) = x^\dagger H_a x$, and $s_{kj} = \langle c(Q_k), c(K_j) \rangle$. By Hermitian symmetry the feature has exactly d^2 real degrees of freedom.

Exact linear (causal) form. Define the prefix state, for $k = 1, \dots, T$,

$$Z_k = \sum_{j \leq k} c(K_j) \in \mathbb{R}^{d^2}, \quad M_k = \sum_{j \leq k} c(K_j) V_j^\top \in \mathbb{R}^{d^2 \times d_v}, \quad (3)$$

(the vectorized real feature $c(\cdot)$). Then, directly from Eq. (2),

$$O_k = \frac{c(Q_k)^\top M_k}{c(Q_k)^\top Z_k + \varepsilon} \quad (4)$$

because $\sum_{j \leq k} s_{kj} V_j = \sum_{j \leq k} \langle c(Q_k), c(K_j) \rangle V_j = c(Q_k)^\top M_k$ and $\sum_{j \leq k} s_{kj} = c(Q_k)^\top Z_k$. The states satisfy the recurrence $Z_k = Z_{k-1} + c(K_k)$, $M_k = M_{k-1} + c(K_k) V_k^\top$.

Proposition 1 (Exactness). *For every k , the output O_k of Eq. (4) equals the output of the quadratic Born attention Eq. (2) exactly (identical up to floating-point summation order), with no approximation of the kernel.*

Proof. Immediate from the linearity of the sums over j and the identity $s_{kj} = \langle c(Q_k), c(K_j) \rangle$ of Eq. (1); no feature-map approximation is introduced. $\square$

Corollary 1 (Constant-state streaming inference). *Autoregressive decoding maintains the fixed-size state (Z_k, M_k) of $d^2 + d^2 d_v$ reals, updated in $O(d^2 d_v)$ per step, and emits O_k via Eq. (4). The state size is independent of the context length T : Born attention decodes with no key/value cache that grows with context.*

Remark 1 (The state is a mixed-state density operator; no basis needed). *One need not materialize a basis $\{H_a\}$. Keeping the complex matrices, let $\rho_k = \sum_{j \leq k} K_j K_j^\dagger \in \mathbb{C}^{d \times d}$ and $\mathcal{M}_k = \sum_{j \leq k} (K_j K_j^\dagger) \otimes V_j$. Then $\sum_{j \leq k} |\langle Q_k, K_j \rangle|^2 = Q_k^\dagger \rho_k Q_k$ and $\sum_{j \leq k} |\langle Q_k, K_j \rangle|^2 V_j = \mathcal{M}_k (Q_k^\dagger, Q_k, \cdot)$, so a matrix-vector product and a contraction suffice. When keys are unit-normalized, $\frac{1}{k} \rho_k$ is the mixed-state density operator of the context $\{K_1, \dots, K_k\}$, and the denominator $Q_k^\dagger \rho_k Q_k = \text{tr}(\rho_{Q_k} \rho_k)$ is the quantum fidelity between the pure query state and that mixed context state, provided both are normalized: the query to unit norm and the context sum to unit trace. Linear Born attention thus replaces the key/value cache with a $d \times d$ density operator (plus a $d \times d \times d_v$ value tensor) that summarizes the history.*

Remark 2 (Why softmax cannot do this). *The softmax kernel $\exp(\langle q, k \rangle)$ has no finite-dimensional feature map, so no exact Z_k, M_k of finite size exist; linear-time softmax variants approximate $\exp$ by a chosen ψ . Born’s kernel is a degree-two polynomial in $(Q, \bar{Q})$ and $(K, \bar{K})$ and therefore has the exact finite feature map $\phi(x) = xx^\dagger$.*

Remark 3 (Exact rotary positions in the linear form). *Rotary embeddings act as a unitary diagonal $R_k = \text{diag}(e^{ik\theta_1}, \dots, e^{ik\theta_d})$ on Q_k, K_j . Because ϕ is quadratic, $\phi(R_k x) = R_k \phi(x) R_k^\dagger$, so $\langle \phi(R_k Q_k), \phi(R_j K_j) \rangle_F = |\langle R_k Q_k, R_j K_j \rangle|^2 = |\langle Q_k, R_{j-k} K_j \rangle|^2$ depends only on the relative position $j - k$, exactly, and this holds inside the linear recurrence, since the prefix states accumulate the rotated features $\phi(R_j K_j)$ and the query contracts with $\phi(R_k Q_k)$. A nonlinear feature map ψ has no such identity ($\psi(R_k q)^\top \psi(R_j k)$ is not a function of $j - k$), so random-feature approximations of softmax do not carry RoPE exactly. The identity is a property of polynomial feature maps, so the real degree-two kernel shares it.*

4 Complexity and the GPT-2 Crossover

Per attention layer and token the quadratic form costs $O(Td)$ per query for the scores and the value mixing, i.e. $O(T^2 d)$ over the sequence; the linear form costs $O(d^2 d_v)$ per step, i.e. $O(Td^2)$ over the sequence (the value width d_v enters both and cancels in the ratio). The linear form is cheaper once

$$T \gtrsim T^* = d^2. \quad (5)$$

Attention is only part of a layer: the token-wise projections and feed-forward cost $\approx 12 d_{\text{model}}^2$ per token, so the quadratic attention share of a layer’s matmul FLOPs is $f = \frac{T}{6 d_{\text{model}} + T}$. Replacing the quadratic attention block by the exact linear one multiplies the *layer* cost by $(1 - f) + f d^2/T$, i.e. an overall speedup of $[(1 - f) + f d^2/T]^{-1}$.

Specialization to GPT-2. A GPT-2 head has $d_h = 64$ real channels at every model size (the head count scales with width). Pairing adjacent channels into (re, im) gives a *complex* head dimension $d = d_h/2 = 32$, so the density-matrix feature lives in $\mathbb{C}^{32 \times 32}$ with $d^2 = 1024$ real degrees of freedom

and $T^* = d^2 = 1024$, the native context length, for the whole family. This is smaller than the real degree-two kernel at the same real width: $(q^\top k)^2$ on $\mathbb{R}^{64}$ has $\binom{65}{2} = 2080$ symmetric features against 1024 here, because $U(1)$ phase invariance couples each (re,im) pair. The Born kernel is correspondingly a subspace of the real degree-two kernels on the same channels: a smaller state and, as a kernel, strictly less expressive. Whether phase structure compensates is the empirical question of §6. Table 1 instantiates GPT-2 small ($d_{\text{model}} = 768$): the linear form is at break-even at the native $T = 1024$ and is cheaper for longer context, approaching the asymptotic d^2/T ratio.

Table 1: Estimated FLOP comparison for GPT-2 small ($d_{\text{model}} = 768$, complex head dim $d = 32$, $d^2 = 1024$, $T^* = 1024$). “Attn. share” is $f = T/(6d_{\text{model}} + T)$; the last column is the overall per-layer speedup of exact Born-linear over exact (quadratic) attention, $[(1 - f) + fd^2/T]^{-1}$. These are FLOP-based estimates; measured wall-clock (which favors the quadratic form via FlashAttention and is affected by the d^2 constant) is reported in §6.

T	attn. share f	d^2/T	overall (linear/quadratic)
1024 (native)	0.18	1.0	1.0× (break-even)
2048	0.31	0.5	1.2×
4096	0.47	0.25	1.6×
8192	0.64	0.125	2.3×
16384	0.78	0.0625	3.7×
32768	0.88	0.031	6.7×

5 Making the Linear Form Practical

Feature dimension. The exact feature has d^2 real dimensions (1024 at GPT-2’s complex head dim $d=32$), and the streaming state is $d^2 d_v$ per head. This is the cost of exactness and sets the $T^* \approx d^2$ crossover; it is favorable for small head dimensions (and, through $U(1)$ phase invariance, about half the real degree-two feature count at the same real width) and grows quickly with d . Concretely the decode state is $BH d^2(d_v+1)$ reals per layer and equals a bf16 key/value cache at a context of about d^2 tokens, independent of batch size; Appendix B tabulates it for three head geometries.

Low-rank / random features. When the exact d^2 features are too large, $\phi(x) = xx^\dagger$ can be compressed: a learned or random projection $P \in \mathbb{R}^{r \times d^2}$ yields r -dimensional features with $r \ll d^2$, trading exactness for a tunable $O(Tr)$ cost. The exact map is available as the $r \rightarrow d^2$ limit, which softmax random features do not have, so the error is a truncation error. Quantifying this trade-off is left to future work.

Numerical stability. The scores are magnitudes, so there is no overflow and no max-subtraction; accumulating Z_k, M_k in fp32 is adequate. Scale invariance of the normalization means ε only needs to be small relative to a typical row sum; scaling the features by $1/d$ keeps $\mathbb{E}[s_{kj}] = O(1)$ across head dimensions. A pre-attention norm on Q, K (as in the complex-valued networks where Born attention originates) keeps the state well-conditioned.

Quadratic vs. linear at short context. Below T^* the quadratic form is preferred; it also admits a FlashAttention-style fused kernel. The Born score removes the online-softmax machinery (no running max, no rescaling, only additive running sums), so a fused quadratic Born kernel is simpler

than FlashAttention’s forward, at the cost of a second $QK^\top$ matmul for the imaginary part. The fused quadratic kernel is the short-context complement to the linear form.

6 Experiments

All results are measured. They are small in scale: short probes at sizes S and M, three seeds per row for the real-stream table. A GPT-2-small run to convergence is deferred, since the systems claims do not depend on model size and the real-stream gap at context 1k is flat across sizes and steps. Two settings are used. In setting (R), a real-stream GPT, Born is a drop-in score function: queries and keys are complexified locally per layer, while the residual stream, values and read-out are real (WikiText-103 [11], GPT-2 BPE, parameter-matched against softmax). In setting (C), a complex-stream network [3], the representation is complex throughout, so phase propagates across layers (character-level `tiny-shakespeare`, 8 blocks). Setting (R) asks how far a phase-sensitive score gets inside a stream that carries no phase; it is also the setting in which the softmax comparison can be made parameter-matched today. Setting (C) is the representation the operator was designed for, and it has a matched softmax baseline (Table 7) alongside the comparison against the parameter-free Fourier mixer.

6.1 Results

Exact-equivalence check (verified). On random inputs ($B=2$, $T=128$, $d=32$, $d_v=64$), the linear form (Eq. (4)) matches quadratic Born attention to 5.6×10^{-6} maximum absolute error (relative 2.1×10^{-6}), float32 tolerance, a numerical confirmation of Proposition 1. A chunkwise-parallel implementation (chunk 64; intra-chunk quadratic, inter-chunk contraction with the exclusive chunk-prefix state, as in Zhang et al. [22]) matches to 3.6×10^{-7} and never materializes a $T \times T$ matrix; it is the form used for long-context training and prefill.

Streaming decode: constant state (measured). Table 2 reports single-layer autoregressive decode at the GPT-2-small head geometry (one A100-40GB) for three operators: quadratic Born attention (eager PyTorch), softmax through `scaled_dot_product_attention` (the FlashAttention-2 and memory-efficient kernels, bf16 key/value cache), and the linear form in two implementations, the complex eager one and a real-block form ($\rho = R + iI$ with $R = cc^\top + ee^\top$, $I = ec^\top - ce^\top$ for $k = c + ie$; the same d^2 state) compiled with `torch.compile(mode="reduce-overhead")`, i.e. CUDA graphs. The real form is checked against the quadratic reference to 1.5×10^{-7} before timing. The linear state is 6.4 MB per layer independent of context, against a cache of 403 MB (bf16 softmax) or 805 MB (complex64 Born) at $T=128k$: $63\times$ and $126\times$ smaller. On latency, the eager linear step is launch-bound at ≈ 0.67 ms (six small kernels); compiled, it is ≈ 0.12 ms, flat in T . This moves the crossover against FlashAttention-2 softmax from $\approx 10^5$ tokens down to between 16k and 64k (0.06 and 0.17 ms for softmax) and makes the linear step faster than eager quadratic Born at every length measured. At 1M tokens FlashAttention-2 decodes at 2.26 ms per token from a 3.2 GB per-layer cache (≈ 38 GB for the twelve layers of GPT-2-small at batch size one); the compiled linear step takes 0.12 ms from 6.4 MB, $19\times$ faster and $500\times$ smaller. The structural result is the memory, an $O(d^2)$ state against an $O(T)$ cache (Corollary 1). The latency result is that the constant-state step is cheap once launch overhead is removed, and that a bandwidth-bound cache read grows with T while a state read does not.

Prefill: two crossovers (measured). Table 3 times a full causal prefill of one attention layer (GPT-2-small head geometry, $d=32$, 12 heads, one A100-40GB) for quadratic Born attention

Table 2: Single-layer streaming decode (GPT-2-small head geometry; one A100-40GB): per-token latency and state/cache memory versus context length T . Quadratic Born is eager PyTorch (complex64 cache); softmax is `scaled_dot_product_attention` (FlashAttention-2 class, bf16 cache); linear Born is given eager (complex) and compiled (real-block form, CUDA graphs). The linear density-matrix state is constant; both caches grow as $O(T)$.

T	Quadratic Born		Softmax (FA-2)		Linear Born		
	lat. (ms)	mem (MB)	lat. (ms)	mem (MB)	eager (ms)	compiled (ms)	mem (MB)
1k	0.15	6	0.03	3	0.69	0.12	6.4
4k	0.15	25	0.03	13	0.68	0.12	6.4
16k	0.15	101	0.06	50	0.67	0.12	6.4
64k	0.42	403	0.17	201	0.67	0.12	6.4
128k	0.82	805	0.31	403	0.67	0.12	6.4
256k	1.63	1611	0.60	805	0.68	0.12	6.4
512k	3.26	3221	1.15	1611	0.68	0.12	6.4
1M	6.08	6443	2.26	3221	0.68	0.12	6.4

in eager PyTorch (materialized $T \times T$ scores), softmax through `scaled_dot_product_attention` (FlashAttention-2, bf16), and the chunkwise exact-linear Born form of §3 (complex eager, chunk 512). Against the unfused quadratic Born, the linear form wins from between 1k and 2k on, where the FLOP analysis of §4 puts it ($T^*=d^2=1024$): $1.5\times$ at 2k, $3.0\times$ at 4k, $6.3\times$ at 8k, and at 16k the quadratic form no longer fits in 40 GB. Against FlashAttention-2 softmax the crossover is two orders of magnitude higher, between 64k and 128k. Below it the fused kernel wins: it runs near tensor-core peak, ≈ 180 TFLOP/s at 32k, while the chunkwise einsum path reaches ≈ 8 , so the linear form’s roughly eightfold FLOP advantage at 32k is not realized. Above it, the unfused eager-PyTorch loop overtakes the fused kernel despite a twenty-fold utilization handicap (104 against 147 ms at 128k), because from 64k to 128k FlashAttention-2 quadruples ($36 \rightarrow 147$ ms) while the linear form doubles ($52 \rightarrow 104$ ms), and the margin widens linearly with context from there. With the streaming form (superblocks of 16k tokens, chunk 512, carried (ρ, M) state; verified against the quadratic reference to 7.5×10^{-7}) the lead is $1.4\times$ at 128k, $2.7\times$ at 256k, $5.5\times$ at 512k and $11\times$ at 1M tokens (0.90 s against 10.0 s per layer), unfused, in eager PyTorch, within 17 GB.

A kernel cannot move the FlashAttention-2 crossover to T^* . The FLOP count against softmax is per token: FlashAttention-2 does $128T$ (causal, head dim 64); the exact linear form must build and read the $d^2 d_v$ state for every token, ≈ 1.15 MFLOP at this geometry regardless of T . Equal work is therefore at $T \approx 9k$, a floor shared by every exact d^2 -feature linear attention [8, 22], and the measured 64k to 128k is that floor times a ten- to twenty-fold utilization deficit. Three attempts to remove the deficit without a custom kernel failed, and they locate the problem: a bf16 real-block rewrite of the chunkwise form (1.6 to $2.1\times$ slower than complex eager), `torch.compile` on either (slower still), and a Khatri–Rao formulation in which the state build and read-out are single real GEMMs on tensor cores ($M = [C|E]^T [C \odot V | E \odot V]$, read-out $[a|b] W$ with $W = [[M_R, -M_I], [M_I, M_R]]$; verified to 9×10^{-7}), which runs at a flat 8.5 TFLOP/s in TF32 and 6.5 in bf16, 2.2 to $2.9\times$ slower than the complex path. The flat utilization is the diagnosis: every eager formulation materializes an $O(Cd^2)$ or $O(Cd d_v)$ intermediate per chunk and is bound by memory traffic. A fused chunkwise kernel avoids those intermediates (the $K \odot V$ tile built in shared memory, the state tile accumulated in registers), which is how gated-linear-attention kernels reach a large fraction of FlashAttention throughput; its ceiling here is the $\approx 9k$ floor, and a realistic landing is 15 to 30k. For a systems reader: the FLOP crossover at $T^* \approx 1k$ is realized only against an unfused quadratic baseline; against FlashAttention-2

the exact linear form is slower below $\sim 100k$ when unfused and an order of magnitude faster at 1M; the fused measurement below closes most of the distance between 9k and 100k. These are per-operator ratios; whole-model ratios are smaller (Table 1) because the MLP and projections are unchanged.

Fused (measured). The diagnosis above was then tested. The feature lift was written as a Triton kernel (forward, and a backward that applies the unpacked Hermitian gradient as a $d \times d$ matvec per token; both verified against autograd to 10^{-5}), and the lifted $d^2=1024$ -wide features were handed to the chunkwise linear-attention kernel of the flash-linear-attention library [20], which tiles the feature dimension across thread blocks and so resolves the register-tile problem noted above; the Born denominator comes from a ones column appended to V . The fused form agrees with the chunkwise reference to 4×10^{-3} with fp32 features and 4×10^{-2} with bf16. Its time is $3.5\times$ below the eager path at 32k (7.5 against 26 ms), of which the lift is 1.4 ms, and the crossover against FlashAttention-2 moves from 64k to 128k down to between 16k and 32k ($0.70\times$ at 16k, $1.3\times$ at 32k, $4.3\times$ at 128k, $8.3\times$ at 256k), the 15 to 30k landing predicted from the 9k FLOP floor. For training (forward plus backward through both kernels) the crossover is at about 64k ($1.07\times$ slower there, $1.8\times$ faster at 128k). The fused form keeps the lifted features and the per-chunk states for the backward, so it uses about $7\times$ FlashAttention-2’s memory at these lengths (7.2 against 1.1 GB at 32k), still linear in T . What remains between 16k and the 9k floor is the lift’s own traffic (2 KB per token and head) and the use of a generic kernel where one specialised to the lifted structure would do; the order-of-magnitude claim no longer rests on an unfused baseline. The same sweep on an H100 PCIe (80 GB, same software) against FlashAttention-3 (v3.0.1, built from source) moves the picture by one octave: fused Born 2.8/5.8/11.3/22.8/45.8 ms at 16k/32k/64k/128k/256k against FlashAttention-3 at 1.05/4.3/17.6/73.9/296 (PyTorch’s cuDNN attention backend is within 5% of it), so the forward crossover is between 32k and 64k and the lead $3.2\times$ at 128k, $6.5\times$ at 256k. FlashAttention-3’s backward is $1.8\times$ faster than FlashAttention-2’s, so training parity moves from 64k to between 64k and 128k (98 against 58 ms at 64k, 195 against 246 at 128k). Against FlashAttention-2 on the same card the crossover stays between 16k and 32k. GLA remains $3.5\times$ faster than the fused form and is faster than FlashAttention-3 from 16k on ($11\times$ at 128k).

Precision and code path. All quality numbers in this section come from the training script’s quadratic branch (bf16 score GEMMs, fp32 normalization) or, where stated (the training twins and the 16k decay run), its eager chunkwise linear path in fp32; none use the fused bf16 path, which is a speed measurement of the same operator within the tolerance above. The 4×10^{-2} figure is the bf16 rounding of the lifted features themselves: bf16 keeps about three significant digits, so 4×10^{-3} relative on outputs of order one is the expected rounding, and it is the precision at which every training run in this paper already executes its score GEMMs under autocast. The 4×10^{-3} fp32-feature figure is accumulation order in the chunk kernel. The lift’s forward and backward match autograd to 10^{-5} . Measured end to end on one model and batch (same weights, same tokens; `fused.check.py`): hidden states differ by 0.5 to 0.6% relative, the loss by 10^{-4} (the trained 128k Born model at 32k) to 7×10^{-4} (random init, 4k), and the gradients by 19% in norm with cosine 0.98 on the trained model, 46 to 58% at random init where gradients are small and noisy. For comparison, the bf16 quadratic branch that trained every quality model in this paper differs from the fp32 chunkwise path by 28% in gradient norm at the same point. The fused gradient is therefore a bf16-noisy estimate of the same order as the one every training run here already accepts. No model was trained on the fused path; that remains untested.

Against a sub-quadratic competitor (measured). The GLA column of Table 3 makes the comparison the FA-2 crossover alone does not: gated linear attention [21], run through the same

library’s chunk kernel with 64 real features per head and a per-dimension forget gate (the kernel size a GPT-2 head would use), has a flat 0.7 ms floor, overtakes FlashAttention-2 at 8k and is 19× faster at 128k. The fused Born form is 4.4× slower than GLA at every length from 16k on (33.7 against 7.7 ms at 128k) and uses 4× its memory (28.6 against 7.1 GB for a training step at 128k). That is the cost of the exact $d^2=1024$ feature width against 64 approximate features: 16× the state for 4.4× the time, because the small- K kernel is bandwidth-bound. Whether exactness buys quality for that cost is answered by the GLA column of Table 7.

Table 3: Single-layer causal prefill (GPT-2-small head geometry; one A100-40GB; mean of 5 after warm-up): wall-clock in ms for quadratic Born (eager PyTorch, complex64), softmax (`scaled_dot_product_attention`, FlashAttention-2, bf16), and the chunkwise exact-linear Born form in two implementations: complex eager (chunk 512) and *fused* (a Triton kernel for the d^2 feature lift plus the chunkwise linear-attention kernel of flash-linear-attention, bf16 features), each with its speedup against FlashAttention-2 (> 1 = linear faster); GLA is gated linear attention [21] through the same library’s chunk kernel (64 real features per head, per-dimension forget gate, bf16), the sub-quadratic competitor. OOM: the $T \times T$ score matrix does not fit in 40 GB. The 32k softmax entry spans two runs. Eager rows from 256k use the streaming form (superblocks of 16k, carried state), which also gives 112 ms at 128k; from 512k the fused form’s lifted features (2 KB per token and head) do not fit in 40 GB.

T	quad. Born	softmax (FA-2)	linear (eager)	vs. quad.	vs. FA-2	linear (fused)	vs. FA-2	GLA	vs. FA-2
1k	0.80	0.05	1.47	0.54×	0.03×	0.66	0.06×	0.70	0.06×
2k	2.68	0.08	1.83	1.5×	0.04×	0.68	0.12×	0.70	0.12×
4k	10.4	0.24	3.43	3.0×	0.07×	0.95	0.25×	0.70	0.33×
8k	42.0	0.73	6.67	6.3×	0.11×	1.83	0.40×	0.73	1.0×
16k	OOM	2.55	13.5	—	0.19×	3.61	0.70×	0.93	2.7×
32k	OOM	9.3–12.0	26.1	—	0.36–0.46×	7.48	1.3 ×	1.81	5.2×
64k	OOM	36.3	52.1	—	0.70×	15.7	2.4×	3.61	10×
128k	OOM	147	104	—	1.4×	33.7	4.3×	7.73	19×
256k	OOM	602	225	—	2.7×	70.5	8.3×	16.0	36×
512k	OOM	2459	450	—	5.5×	OOM	—	32.0	77×
1M	OOM	10006	904	—	11.1 ×	—	—	64.0	156×

Born needs a complex representation (R vs. C). The two settings behave differently (note the different tasks and units: R is BPE WikiText-103, nats/token; C is character `tiny-shakespeare`, nats/char). In the real-stream setting, Born trails softmax by ≈ 0.12 to 0.15 nats/token at equal token budgets (3.47 against 3.35, i.e. perplexity ≈ 32 against 28.5, at a matched step; not converged): phase-sensitive scoring alone, inside an otherwise real network, does not reach parity at this budget. In the complex-stream setting the same operator is strong at the probe budget (1.78 ± 0.05 nats/char for the full stack, Table 9) and, in the source framework’s converged runs, reaches the softmax-parity point (1.52 nats/char [3]). The reading we adopt is that the operator’s value is tied to a complex representation that carries phase across layers; inside a real network it is handicapped, so a real-stream shortfall is weak evidence against the operator. A second suspect in setting (R) was the absence of a per-head query/key norm: since $s_{kj} = \|Q_k\|^2 \|K_j\|^2 \cos^2 \theta_{kj}$ and the normalization is linear, high-norm keys could dominate regardless of alignment, and the Born rule is defined on unit states. This was tested and is not the cause: unit-normalizing Q, K per head (making $s_{kj} \in [0, 1]$ a bounded overlap) leaves the gap unchanged, 3.463 ± 0.001 nats/token over three seeds at matched step 5000 against 3.471 ± 0.003 without the norm (softmax 3.350 ± 0.000), and the curves coincide at every checkpoint from step 1000 on. Key-norm hijacking is therefore ruled out; the remaining

candidates are attention dilution over T and distribution sharpness, whose ablations follow.

Mechanism ablations (setting R; three seeds). Table 4 compares, at matched step 5000 (the same checkpoint as the QK-norm result above), plain Born attention with the mechanisms the density-matrix picture suggests (§1): unit-state normalization (QK-norm), a RetNet-style per-head decay γ_h^{k-j} on the scores (the amplitude-damping reading of dilution), and the $p=2$ power score $|\langle Q, K \rangle|^4$ (the two-copy, sharpening reading). Sharpness is the dominant lever: $p=2$ alone recovers $\approx 64\%$ of the plain-Born gap to softmax, decay alone $\approx 30\%$, and the two together $\approx 74\%$ (3.381 against softmax 3.350; perplexity 29.4 against 28.5), with the gaps still closing at the last common checkpoint (the recovered fractions are within a few points of their step-4000 values). QK-norm contributes $\approx 6\%$, consistent with the negative result above. The trade-off is explicit: $|\langle Q, K \rangle|^4$ is a degree-four kernel and is not linearizable with the d^2 density-matrix state (it needs $\rho^{\otimes 2}$, i.e. d^4 features), so this quality fix is paid for in the exact-linear form, whereas decay alone keeps it. Every row is three seeds; the seed standard deviation is at most 0.004 nats, an order of magnitude below the differences between rows. All rows are size S at 5000 steps, so the fractions are specific to that budget.

Table 4: Mechanism ablations in the real-stream setting (R): size S, WikiText-103 BPE, val nats/token at matched step 5000 (chance 10.82), mean $\pm$ std over three seeds for every row. “Recovered” is the fraction of the plain-Born gap to softmax (0.121) that is closed, from the means.

variant	val @5000 (3 seeds)	gap to softmax	recovered
softmax (target)	3.350 $\pm$ 0.000	—	—
Born, plain	3.471 $\pm$ 0.003	0.121	0
+ QK-norm	3.463 $\pm$ 0.001	0.114	$\approx 6\%$
+ decay	3.435 $\pm$ 0.001	0.085	$\approx 30\%$
+ decay + QK-norm	3.425 $\pm$ 0.003	0.076	$\approx 37\%$
+ $p=2$	3.393 $\pm$ 0.001	0.043	$\approx 64\%$
+ decay + $p=2$	3.381$\pm$0.000	0.031	$\approx 74\%$

Training with the linear form. To connect the linear form as a decode microbenchmark with the linear form as a training operator, we trained the chunkwise linear Born model (§3, chunk 512) and its quadratic twin, with identical operator (the QK-norm variant), data and seed, at context 4096 for 3000 steps (size S, micro-batch 2). Memory: the linear form trains at 15.6 GB against 27.1 GB for the quadratic twin, and at micro-batch 4 the quadratic form does not fit a 40 GB A100 (out of memory) while the linear form uses 30.5 GB. Speed: 4.0 against 9.3 s per step, $2.3\times$ in favor of the linear form, in the same eager PyTorch. Loss: the two validation curves, which Proposition 1 says should coincide up to rounding, do so at every checkpoint: 5.946/5.947, 5.072/5.064, 4.573/4.566, 4.270/4.276, 4.004/4.002 and finally 3.884/3.885 nats/token at steps 500 to 3000 (linear/quadratic), never more than 0.008 apart and with no drift. The per-step training losses agree to 0.01 for the first 1000 steps and then differ by up to 0.04 at individual steps: the two runs sum the same quantities in a different order under bf16, so their parameter trajectories separate slowly, as two runs of one model from bit-different seeds would, while the validation loss stays identical. This is the training-time counterpart of the decode equivalence above: the same model, trained to the same loss, at 58% of the memory and 43% of the wall-clock.

The linear form inside the source framework (setting C). The chunkwise linear operator is implemented natively in the complex-valued framework of Crasmaru [3], as an execution mode of its Born-attention layer, with a hand-written backward: the layer’s existing quadratic CUDA kernel (verified against finite differences) is the oracle, and the linear mode reproduces its forward output and all input gradients to 10^{-3} absolute on gradients of order one, across eight configurations from one chunk per sequence to one token per chunk and one to four heads. Because the framework’s single head is $d = E/2$ complex dims wide ($d = 128$ at $E = 256$, so $d^2 = 16,384$ features and a compute crossover near 16k tokens), the layer was first made multi-head; the numbers below use four heads of 32 dims, the geometry of the rest of this paper. Table 5 times the attention layer alone (forward plus backward, batch 4, chunk 256, FP32, cuBLAS batched GEMMs for the state readout and update) against the framework’s quadratic kernel on an A100. The quadratic kernel’s backward is one thread per query with atomic accumulation, so the linear form wins at every length: $5.5\times$ at 1k, $9\times$ at 4k, $37\times$ at 16k, where the quadratic form needs 14.9 GB of scratch against 6.1 GB total, and it continues to 64k tokens at 1.24 s per step while the quadratic form does not fit. Both time and memory of the linear form grow linearly with N (memory from a 4 GB floor that is the cuBLAS context, paid once by any net that uses it). On an 11 GB GTX 1080 Ti the same sweep gives 16/32/66/133/261 ms at 1k to 16k for the linear form against 121 ms/0.8 s/5.0 s/25 s and out-of-memory for the quadratic one. End to end, a four-block character model trained at context 1024 with the two forms reaches 2.567 (linear) against 2.638 and 2.541 nats/char for two same-seed runs of the quadratic kernel, whose atomic backward makes it nondeterministic by up to 0.17 nats at intermediate checkpoints: the linear run lies between the two quadratic runs at every checkpoint, within the quadratic kernel’s own run-to-run spread. Run alone on the A100, the same four-block model at batch 8 takes 1.35 s per training step with the quadratic kernel and 0.21 s with the linear mode at context 1024 ($6.4\times$ for the whole step, attention included), and 0.78 s per step at context 4096, where the quadratic kernel does not fit in 40 GB.

Provenance. The companion paper’s results all come from the framework’s quadratic kernel, its only Born-attention path until 2026-10-16 (revisions up to 735c269b); the linear mode (commits 6ac32023, e7b405e2, dd16867a, 2e752136 of 2026-10-16: multi-head, linear forward, backward, gradient fix) is a second, GPU-only execution mode of the same layer, with the quadratic kernel kept as default and oracle. Table 5 (revision 2e752136) and Appendix A (revision b26f632b) are the only results in either paper that use it.

A second mixer was then added to the framework for a comparison that this paper could not otherwise make natively: gated linear attention over complex amplitudes, $o_t = \sum_{j \leq t} w_{tj} s \langle q_t, k_j \rangle v_j$ ($s = d_h^{-1/2}$) with a scalar per-head sigmoid gate carried exactly in a $d_h \times d_h$ complex state (the degree-one, un-normalized counterpart of Born attention), with the same chunkwise structure, tests and constant-state decoder. On the PG-19 character model of Appendix A at a single-GPU budget (batch 4, 6000 steps, same seed and tokens) it reaches 1.418 nats/char against 1.938 for linear Born, ahead at every checkpoint from step 500, and trains without any normalizer in the mix; this Born run has no decay, so how much of the margin is forgetting is settled by the mirror’s three-arm control (plain Born, decayed Born, complex GLA at this architecture). Trained without the learned positional table (RoPE only), both mixers stream past their training context: Appendix A scores the PG-19 test partition in 5M-byte windows at 1.66 bits per byte for linear Born with the fixed decay and 1.70 for the gated mixer, where the hierarchical MBLM stack [5] built for that window reports 2.45. Constant-state decoding is wired into the framework’s generator as well. Its graphs are static (every layer carries its token count), so the decoder is a second graph with one token per step, built from the restored model with the same layers and wiring, in which the Born layers keep the running density state of Corollary 1, RoPE takes an absolute position, and the learned positional

table is sliced one row per token. Fed the tokens one at a time it reproduces the fixed-window model’s next-token distribution to 10^{-5} at every position of a 512-token window, and generates 300 characters in 2.6 s on CPU against 101 s for the generator that recomputes the window per token (40×; both single-model, eight threads).

Table 5: Born attention inside the source framework, quadratic kernel vs. the native chunkwise linear mode: forward+backward time per step and peak GPU memory, attention layer only, 4 heads of 32 complex dims, batch 4, chunk 256, FP32, A100-40GB. The linear form’s memory includes a ≈ 4 GB cuBLAS context.

N	quad. ms	quad. GB	linear ms	linear GB	speedup
1k	108	0.25	19.6	4.1	5.5×
2k	222	0.52	39	4.2	5.7×
4k	692	1.36	78	4.5	8.9×
8k	2147	4.2	155	5.0	14×
16k	11325	14.9	310	6.1	37×
32k	OOM	—	620	8.3	—
64k	OOM	—	1242	12.6	—

Long-context usefulness (setting R, context 16k). The 1M-token advantage is a statement about cost. The question here is whether a linear Born model trained long uses the context, and at this budget it uses much less of it than softmax does. We trained the chunkwise linear Born model (QK-norm, chunk 512) and a softmax model, both size S, at context 16,384 on WikiText-103 for 3000 steps at the same tokens per step, and scored both on 32 held-out 16k windows (Table 6). Three measurements. (i) Validation loss by the context available to a token falls with context for both models, and both show the same mild rise in the last bucket, so that shape comes from the data. (ii) The same tokens scored in independent 512-token windows: softmax is 0.19 nats better with the full window than in 512-token windows, Born 0.07. (iii) A real passage presented twice: softmax’s loss on the second copy drops by 0.66 nats at 512 tokens back and 0.27 at 2k, and by nothing at 8k; Born’s drops by 0.11 at 512 and by nothing beyond. So the linear Born model does use context, but it extracts a third of the long-range gain that softmax extracts, and its retrieval reaches a quarter as far. Behind this is a widening training gap: at context 1024 Born trails softmax by 0.12 nats at matched steps (flat across sizes and steps); at 16k the gap is 0.18 at step 500, 0.59 at 1500 and 0.77 at 3000 (4.286 against 3.513), while the same operator at context 4096 reaches 3.884 at step 3000. Softmax is nearly indifferent to context length; $p=1$ Born degrades as T grows. That is the dilution mechanism of §1: a polynomial score under sum-normalization spreads attention over more keys as the window grows, where the exponential keeps it peaked. Of the two sharpening fixes of Table 4, decay is the one that keeps the exact d^2 state, and it is kept exactly in the chunkwise form (the per-chunk states weight keys by γ^{C-1-j} , the prefix combines chunks with $\gamma^{(c-c'-1)C}$, and query i reads it with γ^{i+1} ; verified against the quadratic decay path to 2×10^{-4}). With decay (per-head $\gamma_h = 1 - 2^{-(5+h)}$, the RetNet schedule, nothing tuned) the same 16k run reaches 3.636 at step 3000: the gap to softmax falls from 0.77 to 0.12 nats, the 1k value, so decay removes the context dependence of the real-stream gap (84% of the 16k gap recovered with the d^2 state intact, against 30% of the 1k gap in Table 4), and it was ahead of softmax through step 1500. A plausible reason for that early lead, stated as a hypothesis: at initialization any near-uniform attention over 16k tokens averages 16k random value vectors into noise (signal of order $T^{-1/2}$), for softmax too; the decay schedule $\gamma_h = 1 - 2^{-(5+h)}$ gives head h an effective window of $\approx 32 \cdot 2^h$ tokens from the first step, a high-SNR local signal that softmax has to learn by sharpening, which it does between

steps 1500 and 2000 here. On usefulness it is a trade: the decayed model extracts 60% of softmax’s long-range gain (0.115 against 0.192 nats; plain Born 0.068), while its repeat-copy retrieval at 512 tokens back is weaker than plain Born’s (0.06 against 0.11 nats, softmax 0.66) and absent beyond, which is what forgetting by construction must do: decay cures dilution by damping the far keys that retrieval needs. Sharpness without forgetting is what softmax has and what the $p=2$ score buys at the price of the d^4 state.

Table 6: Long-context usefulness at context 16,384 (setting R, size S, 3000 steps, 32 held-out windows, nats/token; the decay column keeps the exact d^2 state): loss by context available to the token; the full window against the same tokens scored in independent 512-token windows; and the loss on the second copy of a passage presented twice, by passage length (retrieval from that far back).

	Born (linear, QK-norm)	+ decay (in the state)	softmax
val @ step 3000 (training)	4.286	3.636	3.513
tokens with < 512 context	4.432	3.810	3.745
512–2k	4.287	3.666	3.544
2k–8k	4.226	3.599	3.474
8k–16k	4.317	3.636	3.517
full 16k window	4.284	3.630	3.511
same tokens, 512-token windows	4.352	3.745	3.703
gain from context beyond 512	0.068	0.115	0.192
repeat-copy 512: first → second	3.968 → 3.860	3.300 → 3.236	3.298 → 2.643
repeat-copy 2k: first → second	4.241 → 4.232	3.633 → 3.623	3.554 → 3.282
repeat-copy 8k: first → second	4.275 → 4.323	3.665 → 3.669	3.555 → 3.560

Matched softmax baseline inside the complex stream (setting C, WikiText-103). This is the experiment that can change the quality story: the same complex-valued network, with only the mixer swapped. We mirrored the source framework’s block in PyTorch (complex residual stream, complex GELU, token norm, one head of $d_h=96$ complex dims, RoPE, Born read-out head; 43M parameters, six layers) and trained it on WikiText-103 BPE for 5000 steps at context 1024 with three mixers and nothing else changed: Born attention, softmax of the Born score $|\langle q, k \rangle|^2$, and softmax of $\text{Re}\langle q, k \rangle / \sqrt{d_h}$ (Table 7). Inside the complex stream the gap is within noise: 4.564 (Born) against 4.553 and 4.529 (softmax of the Born score, softmax of the real part) at step 5000, i.e. 0.01 to 0.035 nats, and within 0.04 at every checkpoint. A fourth mixer answers the cost question raised by the prefill table: the reference gated-linear-attention layer on the same packed stream (its forget and output gates added, 44.6M parameters, no rotary positions) reaches 4.482 at step 5000, 0.05 to 0.08 nats below all three exact-score mixers, and leads from step 2000 on. At context 1k the exact kernel therefore buys no quality over the gated approximation that runs $4.4\times$ faster, and context widens the margin: at 4k GLA reaches 4.652 at step 3000 against 4.829 / 4.739 / 4.697, a lead of 0.05 to 0.18 nats, having started behind (6.941 at step 500). In the real stream at the same budget the gap is 0.12 nats and, as shown above, grows with context. The same three mixers at context 4096 (bottom of Table 7) start at parity (6.824 / 6.823 / 6.836 at step 500) and end at 4.829 / 4.739 / 4.697 at step 3000: a gap of 0.09 to 0.13 nats, against 0.44 in the real stream at the same context and step (3.884 against 3.446). The complex stream thus removes about three quarters of the long-context gap; the residual grows with context (0.01 to 0.035 at 1k, 0.09 to 0.13 at 4k), which is dilution again, at a quarter of the real-stream rate. Two readings are consistent

with this. Either the complex stream supplies the sharpness that the real stream lacks (interference across layers), or Born and softmax are both limited by this mirror, which trains about a nat worse than the real-stream GPT at the same step; the absolute level of the mirror is not tuned. What the comparison establishes is narrower: the real-stream gap is a property of the real-stream setting, not of the operator.

Table 7: Matched mixers inside the complex stream (setting C): the PyTorch mirror of the source framework’s block on WikiText-103 BPE, size S (43M), val nats/token by step at context 1024 (top) and 4096 (bottom); GLA is the reference gated-linear-attention layer on the same packed stream (44.6M parameters, its gates added). The real-stream GPT (setting R) at the same budget is shown for the gap, not the level.

step	Born	softmax(Born score)	softmax(Re)	GLA	R: softmax	R: Born
1000	5.788	5.754	5.749	5.798	4.598	4.789
2000	4.970	4.944	4.931	4.899	3.739	3.963
3000	4.708	4.721	4.666	4.625	3.492	3.657
4000	4.608	4.596	4.567	4.518	3.403	3.535
5000	4.564	4.553	4.529	4.482	3.350	3.466
<i>context 4096 (micro-batch 4, same tokens per step; R columns: softmax / linear Born at 4k)</i>						
500	6.824	6.823	6.836	6.941	5.742	5.946
1500	5.430	5.347	5.303	5.289	4.120	4.573
3000	4.829	4.739	4.697	4.652	3.446	3.884

Training at 128k context: Born, decayed Born, GLA and softmax (setting C, PG-19).

The long-context question is sharpest where the exact linear form is already cheaper than quadratic attention, so we trained the same shallow complex-stream net (two blocks, $E=256$, four heads of $d=32$, 11M parameters) at context 131,072 on PG-19 with five mixers and nothing else changed (Table 8): plain linear Born, linear Born with the fixed per-head decay of Table 4 carried exactly in the d^2 state, the reference GLA layer, the complex-amplitude gated mixer of the source framework (§6, setting C, mirrored in PyTorch), and FlashAttention-2 softmax on the real part of the complex score. For the GLA and softmax reference arms the attention mixer operates on the packed real representation of the complex stream ($2E$ real channels: the GLA layer with its own real projections, the softmax as scaled-dot-product attention on the packed q, k, v , whose score is $\text{Re}\langle q, k \rangle$), while the embedding, residual stream, normalization and feed-forward layers remain complex-valued in every arm. Three facts, in decreasing order of robustness. (i) Plain Born dilutes: 4.565 nats, 0.35 behind softmax and 0.70 behind GLA, the 16k pattern of Table 6 continued. (ii) Decay removes almost all of it: 3.903, within 0.04 of GLA at step 4000 and within 0.02 to 0.08 at every earlier checkpoint, with no learned parameter, and 0.32 ahead of softmax, which also has no way to forget at this length. The complex-amplitude gated mixer lands at 4.013, between decayed Born and the real-valued GLA layer, so at this length the complex amplitudes buy nothing over real gates. What plain Born lacks at long context is therefore forgetting, not exactness or gating; whether the d^2 state adds anything over a first-order fixed-decay state (RetNet) is not tested here. (iii) GLA remains the cheaper operator: 0.96 s per whole-model step against 1.73 for the eager Born path and 1.97 for the softmax model. The caveats are those of this whole section: one seed, two blocks, 4000 steps, GLA without rotary positions. One measurement puts the loss differences in context: for decayed Born, GLA and softmax the full-window loss is within 0.013 nats of the loss on independent 4k windows of the same tokens, and for plain Born the full window is 0.039 nats worse than the 4k

windows, dilution measured directly. The repeat-copy probe shows no retrieval for any mixer at any distance from 2k to 256k (second copy within 0.01 of the first), so none of these small models uses context beyond a few thousand tokens. The 128k numbers are differences in local modelling under long-context training, and the decay’s gain is the gain of a local inductive bias. Fed 512k windows, four times the training length, the three constant-state models show no degradation: their per-position loss in the 128k to 256k and 256k to 512k buckets stays within the in-window spread (two windows; GLA 3.96/3.87 against 3.95 to 4.02 inside, decayed Born 4.00/3.90 against 4.00 to 4.06, plain Born 4.69/4.62 against 4.63 to 4.72), so neither the gated nor the exact state drifts when run past the length it was trained at.

Table 8: Training at context 131,072 on PG-19 (8k byte-level BPE; shallow complex-stream net: 2 blocks, $E=256$, 4 heads of $d=32$; 4000 steps of 262k tokens; one A100; seed 1000): validation nats/token by step, whole-model step time, and the context actually used, measured as the gap between the full 128k window and independent 4k windows on the same tokens. Born runs the eager fp32 chunkwise path (not the fused one); “+decay” is the fixed per-head γ_h of Table 4 carried exactly in the state; GLA is the reference layer (no rotary positions, +0.8M gate parameters); softmax is FlashAttention-2 on $\text{Re}\langle q, k \rangle$.

mixer	1000	2000	3000	4000	s/step	full – 4k-window
Born, plain	4.804	4.649	4.630	4.565	1.74	+0.039
Born + decay	4.341	4.067	3.977	3.903	1.73	−0.008
GLA	4.323	4.037	3.901	3.865	0.96	−0.011
GLA, complex amplitudes	4.461	4.201	4.044	4.013	1.08	−0.002
softmax (FA-2)	4.759	4.389	4.297	4.218	1.97	−0.013

How many Born layers, and where (setting C). Replacing the Fourier mixer by Born attention in k of 8 complex blocks (others left as parameter-free Bluestein mixers), validation loss falls monotonically in the mean with k (Table 9; three seeds per row). A single Born layer recovers roughly half of the 0→8 gain ($2.16 \rightarrow 1.95$ to 2.00 of $2.16 \rightarrow 1.78$); a second adds little beyond seed noise (1.90 ± 0.03); the full stack is clearly best (1.78 ± 0.05 , 0.12 below the hybrid). The benefit therefore does not saturate after one layer, and it does not accrue uniformly per layer either. Placement of a lone Born layer is not significant: last against first is 1.95 ± 0.03 against 2.00 ± 0.07 , inside the seed variance. The robust effects are the first Born layer and the full stack; an early single Born layer is not privileged, which argues against a design that embeds in the complex domain, measures once and continues in the real domain, although one Born layer is a reasonable cheap operating point.

7 Related Work

Linear attention via feature maps originates with Katharopoulos et al. [9]; Performers [1], random-feature attention [13], and cosFormer [14] design feature maps to approximate the softmax kernel. FlashAttention [4] keeps the exact quadratic kernel but makes it IO-optimal. Fourier and spectral token mixers [10] remove content-based attention entirely.

Polynomial-kernel and higher-order linear attention. The exactness our characterization relies on is established prior work. Kacham et al. [8] use the exact finite feature map of the degree- p polynomial kernel ($\langle x, y \rangle^p = \langle x^{\otimes p}, y^{\otimes p} \rangle$) to linearize attention, sketching it for large p ; Zhang et al.

Table 9: Born-layer count/placement in the complex-stream network (char `tiny-shakespeare`, 8 blocks; val nats/char at a 3000-step probe, mean $\pm$ std over 3 seeds for every row; lower is better). The 2/8 and 8/8 rows agree with the independent three-seed probe of Crasmaru [3] (1.88 ± 0.05 , 1.83 ± 0.06); the 0 row is the 8-block Bluestein baseline (that work’s 2.43 was a 6-block model).

Born blocks (of 8)	placement	val (nats/char)
0 (Bluestein only)	—	2.16 ± 0.03
1	first (L_0)	2.00 ± 0.07
1	last (L_7)	1.95 ± 0.03
2	L_3, L_7	1.90 ± 0.03
8	all	1.78 ± 0.05

[22] give the exact second-order case as a constant-state causal recurrence with associative-scan training, structurally identical to our §3 over the reals. Our role is to identify Born attention as the complex, degree-two instance (density-matrix features); the linear form itself is not claimed as new. Table 10 places the operator among these relatives. What is newly enabled is narrow: (a) the same 64 real channels give 1024 exact features instead of the 2080 of the real degree-two kernel, halving the state at equal width, because the $U(1)$ phase is quotiented out; (b) lift, state and rotary positions are one object, the density matrix, so the constant-state decoder of §6 carries RoPE exactly with no extra machinery; (c) the quantum reading predicted the two mechanisms (decay, multi-copy sharpening) that the ablations confirmed. The recurrence, the chunkwise form and the associative scan are inherited.

Table 10: Born attention among its closest relatives (per head; “features” is the feature count at GPT-2’s 64 real channels per head; $d_v=64$; T the context). “Exact” means the causal sums are computed without approximation of the stated score. Rotary positions are exact for every *polynomial* feature map (Remark 3); only nonlinear random-feature maps lose the relative-position identity.

method	score	exact	features	state per head	rotary positions
softmax / FlashAttention-2 [4]	$\exp(\langle q, k \rangle / \sqrt{d_h})$	yes ($O(T^2)$)	none finite	$2d_v T$ cache	exact
Performer / RFA [1, 13]	exp via m random features	no	m	$m(d_v+1)$	approximate
PolySketchFormer [8]	$(q \cdot k)^p$, sketched	$p=2$ unsketched	2080 ($p=2$) or sketch r	$r(d_v+1)$	exact unsketched
Higher-order linear attn. [22]	second-order prefix statistics	yes	second order in d_h	constant in T	not discussed
Gated linear attention [21]	$q \cdot k$, per-dim. forget gate	different operator	64	$64 d_v$	gate carries position
Born (this paper)	$ \langle q, k \rangle ^2$, complex	yes	1024 (d^2 , packed)	$d^2(d_v+1)$	exact

Gated and retentive linear attention. First-order recurrent models such as RetNet [19] and gated linear attention [21] keep a $d_h \times d_v$ state by combining a degree-one inner product with fixed or data-dependent decay gates. Born attention pays a $d^2 \times d_v$ state for an exact degree-two Hermitian score, raising the score’s feature rank from d_h to d^2 with no gating parameters and keeping rotary positions exact, and it remains compatible with scalar per-head decay in the same chunkwise kernels (§6). The measured price is in Tables 3 and 7: $4.4\times$ the time of GLA from 16k on, and no quality advantage over it at context 1k or 4k.

Squared and complex attention, empirically. The empirical picture for this specific operator is not closed. Real degree-two attention $(q \cdot k)^2$ degrades at scale (Kacham et al. [8] report softmax parity only for degree ≥ 4), and the real squared, sign-invariant “expressive attention” of Gros

[6] is validated on synthetic autoregressive tasks rather than language modeling. Complex-valued attention has been benchmarked by Hioki [7], with a cosine (real-part) score and gating rather than the squared magnitude; that work reports that unnormalized complex attention stays near chance, consistent with the role of Born’s sum normalization. Quantum-inspired works compute normalized squared inner products on small, often quantum-circuit, benchmarks. We did not find a controlled, parameter-matched benchmark of the exact complex Born operator against softmax at language-model scale; that gap is the empirical contribution of §6.

Born-rule attention itself was introduced in the complex-valued framework of Crasmaru [3], which extends to content-based token mixing the Born-rule (squared-magnitude) read-out of Crasmaru [2], where networks compute in the DFT (Hadamard) frequency domain. Concurrent work studies closely related operators without the linear form: Reinhardt and Hauser [17] give a quantum-circuit roadmap relating Born measurement to softmax on the probability simplex, and Nebli et al. [12] build complex phase-state sequence models with quadratic Born-form scores to 253M parameters (relaxing a Born vocabulary head to a tied softmax). Both evaluate the scores in $O(T^2)$ quadratic form; neither derives the exact $O(Td^2)$ density-matrix recurrence, the $T^*=d^2$ crossover, or the exact-RoPE property (§3). Learnable low-rank polynomial sketches for linear attention remain an active direction [15], consistent with §5.

8 Limitations and Conclusion

What the evidence supports, and what it does not. It supports four statements. (i) The quadratic and linear Born operators are algebraically equivalent and agree numerically to 10^{-6} in fp32 and 10^{-3} through the fused bf16 path. (ii) The recurrent state is independent of T for fixed d , d_v : $d^2(d_v+1)$ reals per head (Appendix B). (iii) On one A100 (and, within one octave, on an H100 PCIe), as a computational result only, the fused form overtakes FlashAttention-2 between 16k and 32k tokens forward and near 64k for training, the streaming form by $11\times$ at 1M, and gated linear attention, with $16\times$ fewer features, stays $4.4\times$ faster than the fused form; at $d=64$ the crossover moves to 128k and at $d=128$ it does not occur (Table 14). (iv) Born attention’s quality depends on its surrounding representation and on forgetting: at 128k a fixed decay in the state is worth 0.66 nats, more than any difference between mixers. The evidence does not establish that Born attention is a generally competitive replacement for softmax (it is size S, 3 to 5k steps, one to three seeds, contexts 1k to 128k, the 128k runs two blocks deep), that the complex representation is the causal source of the complex-stream parity (two readings remain, §6), or that the speedups transfer to larger heads, where d^2 grows and both crossovers move out (Table 13). The sub-quadratic advantage is asymptotic: below $T^* \approx d^2$ the quadratic form is faster, FlashAttention pushes the wall-clock crossover beyond the FLOP crossover, and the exact feature dimension grows with head dimension, which motivates low-rank compression and reintroduces approximation.

The sharpness and state-size trade-off. The experiments isolate one trade-off. Inside a real-valued Transformer (setting R), $p=1$ Born attention trails softmax, and the gap grows with context through attention dilution (Table 6). Amplitude-damping decay keeps the exact $d^2=1024$ state, closes 30% of the 1k gap and 84% of the 16k gap (restoring the 1k-level gap at 16k, at the price of retrieval beyond a few hundred tokens); closing 74% at 1k needs the $p=2$ score $|\langle Q, K \rangle|^4$ (Table 4), whose exact state is $\rho^{\otimes 2}$, $d^4 \approx 10^6$ features. A learned temperature is not a shortcut: under sum-normalization the only temperature-like knob is the exponent itself. Two routes reconcile softmax-level sharpness with a d^2 -scale state: (a) sketch $\rho^{\otimes 2}$ to rank $r \approx d^2$ with the polynomial sketches of Kacham et al. [8], an experiment rather than a derivation; or (b) run the operator in a

complex residual stream (setting C), where multi-layer interference supplies the sharpness at $p=1$: the matched comparison shows no gap at context 1k and a quarter of the real-stream gap at 4k (Table 7). The linear form itself is not new (§7); what this work establishes is the density-matrix characterization ($O(Td^2)$ compute, $d^2(d_v+1)$ state, exact RoPE) and the measured boundaries of both regimes.

A A character model on PG-19 with the native linear operator

To exercise the source-framework implementation of §6 on a real corpus at long context, we trained a character-level model on PG-19 [16] (pre-1919 Project Gutenberg books) and sampled from it with the constant-state decoder. The numbers are those of a small model trained for two hours; they are reported for completeness and reproducibility, not as a benchmark result (published PG-19 byte-level models at $\sim 1\text{B}$ parameters reach ≈ 1 bit per byte).

Data. The first 1,200 books of the training split in the official file order, concatenated with a blank line between books: 472M characters, 174 distinct byte values; the last 10% of the stream is held out as the training-time validation set. The 50-book official validation split is used only for the final score.

Architecture. The language model of the source framework’s `dp_shakespeare` example, all blocks Born: byte embedding (174×256 complex) plus a learned positional table (4096×256), then 6 pre-norm blocks of [TokenNorm $\rightarrow$ complex GELU $\rightarrow Q, K, V$ projections ($256 \rightarrow 128$ complex each) $\rightarrow$ RoPE on Q, K (per-head ladder) $\rightarrow$ Born attention with 4 heads of 32 complex dims in the chunkwise linear mode (chunk 512) $\rightarrow$ output projection ($128 \rightarrow 256$) $\rightarrow$ residual $\rightarrow$ TokenNorm $\rightarrow$ FFN $256 \rightarrow 1024 \rightarrow 256$ with complex GELU $\rightarrow$ residual], a final TokenNorm, a $256 \rightarrow 174$ read-out and the per-position Born-rule cross-entropy. 5.07M complex parameters (3.93M in the blocks, 1.05M positional, 0.09M embedding and read-out), i.e. 10.1M real numbers, a 41 MB checkpoint.

Training. Context 4096 characters, global batch 32 (4 per GPU, 131k characters per step) on $8 \times \text{A100-40GB}$ in the framework’s data-parallel mode (one process per GPU, NCCL all-reduce), 10,000 steps (1.3B characters, about three passes over the subset), Adam with peak learning rate 10^{-3} (warm-up then cosine), gradient clipping 0.5, Born-loss floor $\varepsilon = 10^{-3}$, FP32 complex throughout, no dropout. 0.82 s per step, 29 GB per GPU, 2.3 h wall-clock. The quadratic kernel does not fit this configuration (Table 5).

Table 11: PG-19 character model: validation loss (nats/char, held-out 10% of the training stream) by step. Chance is $\ln 174 = 5.16$.

step	500	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
val	2.486	2.345	2.016	1.814	1.650	1.523	1.407	1.362	1.304	1.312	1.300

Speed. Training: 0.82 s per step of 131k characters, i.e. 160k characters per second across the eight GPUs (20k per GPU) at 99 to 100% GPU utilisation with the CPU idle, FP32 complex with exact (non-TF32) GEMMs. From Table 5 the six linear attention layers account for roughly 470 of the 820 ms; the quadratic kernel does not fit this configuration, and at context 1024 it is $6.4\times$ slower per whole step. The same operator in the PyTorch trainer of §6 runs about $3\times$

Table 12: PG-19 test partition in 5,242,880-byte windows (eight windows, state reset at each window boundary), bits per byte on the same partition, window length and metric as the MBLM 5M-context configuration [5]. The two models of this appendix share recipe, data, seed and budget and differ only in the mixer. MBLM’s short-context configurations (1.24 at 98k on 200 GB) are not what is compared.

	MBLM (5M context)	linear Born + decay	gated complex mixer
test bits per byte (range per window)	2.448	1.656 (1.60–1.79)	1.700 (1.64–1.84)
validation, 5M windows / 4k resets	—	1.690 / 1.696	1.732 / 1.736
best held-out loss, training (nats/char)	—	1.121	1.148
parameters	350M	4.03M complex (8.06M real)	same
training bytes	100 GB, full corpus	0.49 GB, 1,200 books	same
hardware, time	1×A100, 15 h	1×GH200, 2.4 h	1×GH200, 2.1 h
state per head	5M-byte context	$d^2(d_v+1)$, constant	d_h^2 , constant
scoring	5M-byte context	1,280 carried 4k forwards per window	

faster per token (bf16 projections) and FlashAttention softmax about $10\times$; those are precision and fusion gaps, not operator gaps (§6). Generation: with the constant-state decoder on 8 CPU threads the final checkpoint produces 30 characters per second at a per-character cost that does not depend on the position, against 0.03 (34 s per character) for the generator that recomputes the full 4096-character window per token. On this checkpoint the attention state could run indefinitely, but the learned positional table has 4096 rows, so past that point the generator re-prefills from the last 2048 characters (a sliding window, paid once per 2048 characters rather than per character); a RoPE-only model does not need it.

Result. Training-time validation 1.300 nats/char at step 10,000 (1.286 on the final, larger validation pass at the end of training; training loss 1.278, so no overfitting over three passes). On the official validation split, scored with the constant-state decoder in consecutive 4096-character windows (state and positions reset at each window boundary, the first character of a window unscored) over 1.2M characters sampled across the 50 books: 1.349 nats/char = 1.95 bits per byte. The decoder reproduces the fixed-window model’s distribution to 7×10^{-6} on this checkpoint.

Streaming the test partition in 5M-byte windows. The MBLM hierarchy [5] (a Mamba-2 stage over two Transformer stages, dimension 256, 350M parameters) reports 2.448 bits per byte on the PG-19 test partition when trained and evaluated with a 5,242,880-byte context (100 GB of PG-19 bytes, 15 h on one A100). A constant-state model needs no hierarchy for that window. We trained two byte models with the recipe above but without the learned positional table (RoPE only, so the streaming twin is not bound to N): $E=256$, six blocks, four heads, 4.03M complex parameters, batch 8 at $N=4096$ bytes, 15,000 steps (0.49 GB of the same 1,200 books, on one GH200), one with linear Born attention and the fixed per-head decay of Table 4 carried in its state (2.4 h, best held-out loss 1.121 nats/char) and one with the gated complex mixer of §6 (2.1 h, 1.148). Without the positional table the decayed Born model is ahead of the gated one at every checkpoint, the reverse of the order the two mixers showed with the table at the single-GPU budget above. Each model was scored on the test partition (100 books, 41.3 MB) in consecutive 5,242,880-byte windows, each one unbroken constant-state pass (1,280 GPU forwards of 4,096 bytes with the state carried between them and RoPE at the absolute byte position; state reset only at a window boundary): 1.656 bits per byte for decayed Born (1.148 nats; 1.60 to 1.79 per window) and 1.700 for the gated mixer (1.178 nats; 1.64 to 1.84 per window; Table 12). The GPU pass reproduces

the one-byte-at-a-time CPU decoder exactly for the gated model on the prefixes where both were run, and to within 0.1% of the summed loss for the Born model, where the fp32 chunkwise kernel accumulates the d^2 features and the normalizer in a different order. The 16 test bytes outside the models' 174-byte vocabulary are charged 8 bits each and the 1,520 trailing bytes of the partial last window are dropped. On the validation partition the two models score 1.690 and 1.732 bits per byte in 5M-byte windows against 1.696 and 1.736 with the state reset every 4,096 bytes: the carried state neither drifts over five million bytes nor adds much beyond the training context, consistent with the 128k probes of §6. Against the 2.448 of the hierarchical stack the decayed Born model is 0.79 bits per byte lower with 1/40 of the parameters and 1/200 of the training bytes. It is a comparison at the window that stack was built for, not at its short-context strength (1.24 at a 98k context on 200 GB).

Sample. Temperature 0.8, prompt in bold, 800 characters, unedited:

CHAPTER I.

It was a cold, bright morning in the spring of the year, and the old house stood silent among the elms. The doctor had not yet come down, and Margaret, who had risen early, was walking in the garden with a book she did not read.

And then, about the great and should glance, have looked pounds! When he and he drifting the deer did the first distressed that Griem gradually say, and usually come to her fine body upon the hedge-friends with some very knees after a very saving myself with indulgent, to a hole room soldiers. Her pasture, was not yet beautiful there at any fire time to have sent her on her merry little soaked and upon a pretty five and the ground of business, however; then he replied. The window was a personal granted and pipe and the figures and still the good, she said, "I should not just along them better to think me a long along they understand the share incident more than the ballant was sent a long persecute trial of the snow after they got a forest time to his fervene was surrounded to ac

B Decode state versus head geometry

The constant state is $d^2(d_v+1)$ reals per head with packed Hermitian features ($2d^2(d_v+1)$ in the unpacked real-block form used for the compiled decoder of Table 2, which is why that table reads 6.4 MB rather than 3.2), hence $BHd^2(d_v+1)$ per layer at batch B and H heads, times the bytes per element. A bf16 key/value cache costs $4Hd_v$ bytes per token and layer (real width $2d=d_v$ for keys), so the state equals the cache at a context of $T_{\text{mem}} = d^2(d_v+1)b_s/(4d_v) \approx d^2b_s/4$ tokens for b_s bytes per state element: about d^2 tokens for an fp32 state against a bf16 cache, $d^2/2$ at equal precision, and independent of batch size and head count, since both sides scale with BH . Table 13 evaluates this for three head geometries: the state is economical where it is measured (GPT-2 small) and at the 7B-class geometry, and the break-even context grows as d^2 , the same growth that moves the compute crossover of §4.

The time side moves with the state. Table 14 repeats the prefill measurement of Table 3 at three head geometries with the fused Born path, FlashAttention-2 and GLA. At $d=32$ ($d^2=1024$) the forward crossover against FlashAttention-2 lies between 16k and 32k; at $d=64$ (4096 features) it moves to 128k and a training step no longer fits in 40 GB beyond 32k; at $d=128$ (16,384 features) there is no crossover up to 32k and the forward does not fit beyond it. GLA's lead over the fused form grows from 4 to $5\times$ to $19\times$ to $50\times$ across the same geometries. Exact Born attention is therefore a small-head-dimension operator: attractive at $d\leq 32$, marginal at $d=64$, and not competitive at

Table 13: Per-layer decode state (batch 1) against the bf16 key/value cache for three head geometries; T_{mem} is the context at which the cache reaches the state size.

geometry	H	d	d_v	state fp32 / bf16 (MB)	cache (KB/token)	T_{mem} fp32	T_{mem} bf16
GPT-2 small (64-real heads)	12	32	64	3.2 / 1.6	3.1	1.0k	0.5k
7B-class (128-real heads)	32	64	128	67.6 / 33.8	16.4	4.1k	2.1k
256-real heads	8	128	256	134.7 / 67.4	8.2	16.4k	8.2k

$d=128$. On an H100 PCIe (80 GB) the same sweep gives the same ordering, with the $d=64$ forward crossover between 64k and 128k and a training step fitting up to 64k.

Table 14: Head-dimension sensitivity of the prefill crossover (one layer, batch 1, A100-40GB, bf16; forward ms per layer; fused Born = Triton lift + chunk kernel; $d_v=2d$ real). OOM: does not fit in 40 GB.

geometry	T	Born fwd	FA-2 fwd	GLA fwd	Born/FA-2	Born/GLA	Born f+b	FA-2 f+b
$12 \times d=32$ ($d^2=1024$)	16k	3.6	2.5	0.9	$1.43\times$	$3.8\times$	34	9.5
	32k	7.5	9.3	1.8	$0.80\times$	$4.2\times$	72	35
	128k	33.8	144	7.4	$0.23\times$	$4.6\times$	299	537
$8 \times d=64$ (4096)	16k	19.6	3.0	1.1	$6.5\times$	$18\times$	227	11
	32k	40.6	10.9	2.2	$3.7\times$	$19\times$	457	41
	128k	168	167	9.0	$1.01\times$	$19\times$	OOM	623
$4 \times d=128$ (16,384)	4k	17.4	0.34	0.7	$51\times$	$25\times$	2114	1.3
	32k	143	11.6	2.8	$12\times$	$52\times$	OOM	49
	64k	OOM	44	5.7	—	—	OOM	191

C Reproducibility

Hardware. PyTorch measurements and training: single A100-SXM4-40GB GPUs of an eight-GPU node (prefill, decode and the real- and complex-stream training runs use one GPU each; the 16k runs micro-batch 1; the PG-19 model all eight). The fused and GLA benchmarks: a separate single A100-SXM4-40GB. The 11 GB sweep: a GTX 1080 Ti (CUDA 11.3). The Hopper repeat of the prefill, GLA and head-dimension sweeps (§6, Appendix B): one H100 PCIe 80 GB. The 5M-byte-window byte models and their scoring (Appendix A): one GH200 (96 GB, sm_90 build).

Software. PyTorch 2.7.0 (CUDA 12.8 wheels), Triton 3.3, flash-linear-attention 0.5.2; FlashAttention-3 v3.0.1 built from source on the H100. The source framework is C++/CUDA compiled with `nvcc` for sm_80 (sm_90 on the GH200), fp32 throughout with non-TF32 (pedantic) cuBLAS GEMMs and strided-batched GEMMs for the linear state.

Timing. Prefill and decode: mean of 5 iterations after 2 warm-ups, device-synchronized, batch 1, one layer at the GPT-2-small head geometry (12 heads, $d=32$ complex, $d_v=64$); forward-plus-backward: 3 after 2; peak memory is the allocator high-water mark, reset per configuration; the compiled decode step runs under CUDA graphs (`torch.compile`, reduce-overhead mode); FlashAttention-2 is `scaled_dot_product_attention` in bf16 with `is_causal`; streaming prefill uses 16k superblocks, chunk 512 and a carried state.

Data and training. WikiText-103 tokenized with the GPT-2 BPE (`tiktoken`) into train/validation bins; size S of the training script (43M parameters, six layers); 262,144 tokens per optimizer step, AdamW ($\beta_2=0.95$), peak learning rate 6×10^{-4} with 2000 warm-up steps and cosine decay to a tenth, weight decay 0.1, gradient clip 1.0, dropout 0.1, bf16 autocast; the 16k runs use chunk 512. Seeds: 1000 for single-seed rows, 1000/1001/1002 for the three-seed rows (Table 4, the key-norm comparison and the placement sweep); Tables 6 and 7, the training twins and the source-framework sweep are single-seed. PG-19: Appendix A.

Scripts. `bench_linear.py` (equivalence check, eager prefill and decode), `bench_prod.py` (FlashAttention-2 baselines, compiled decode), `bench_1m.py` (streaming prefill to 1M), `bench_kr.py` (the Khatri–Rao attempt), `fused/born_triton.py`, `born fla.py`, `bench_fused.py`, `bench_gla.py` (Triton lift, FLA wrapper, the fused and GLA sweeps, with their A100 logs); `prepare_wt103.py`, `train_wt103.py` (settings R and C via `--stream`; mixers via `--attn/--cattn`; `--born_linear`, `--decay`, `--born_pow`, `--qk_norm`), `cnet_torch.py`, `eval_ctx.py` (full-window versus independent-window and repeated-passage evaluation); in the source framework `test_bornattn_mh.cpp` (oracle tests), `bench_bornattn.cpp` (layer sweep), `dp_shakespeare.cpp` (training) and `generate.cpp` (streaming decode, `-check`).

Artifact. The scripts and the A100 logs are in the authors’ repository, released with this preprint.

References

- [1] Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *International Conference on Learning Representations (ICLR)*, 2021.
- [2] Marcel Crasmaru. On the equivalence of convolutional and Hadamard networks using DFT. *arXiv preprint arXiv:1810.11650*, 2018. URL <https://arxiv.org/abs/1810.11650>.
- [3] Marcel Crasmaru. CNet: A complex-valued deep learning framework with wirtinger autodifferentiation and FFT–hadamard convolution. *arXiv preprint arXiv:2610.08592*, 2026. URL <https://arxiv.org/abs/2610.08592>.
- [4] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- [5] Eric Egli, Matteo Manica, and Jannis Born. Multiscale byte language models: A hierarchical architecture for causal million-length sequence modeling. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2502.14553.
- [6] Claudius Gros. Reorganizing attention-space geometry with expressive attention. *arXiv preprint arXiv:2407.18601*, 2024.
- [7] Leona Hioki. Complex-valued phase-coherent transformers. *arXiv preprint arXiv:2609.22415*, 2026.
- [8] Praneeth Kacham, Vahab Mirrokni, and Peilin Zhong. Polysketchformer: Fast transformers via sketching polynomial kernels. In *International Conference on Machine Learning (ICML)*, 2024.

- [9] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning (ICML)*, 2020.
- [10] James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. FNet: Mixing tokens with fourier transforms. In *Proceedings of NAACL-HLT*, 2022.
- [11] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.
- [12] Ahmed Nebli, Hadi Saadatdoorabi, Christopher Keibel, and Kevin Yam. The quantum shortcut: Complex phase-state dynamics reduce the optimization steps of sequence models. *arXiv preprint arXiv:2608.14691*, 2026.
- [13] Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations (ICLR)*, 2021.
- [14] Zhen Qin, Weixuan Sun, Hui Deng, Dongxu Li, Yunshen Wei, Baohong Lv, Junjie Yan, Lingpeng Kong, and Yiran Zhong. cosformer: Rethinking softmax in attention. In *International Conference on Learning Representations (ICLR)*, 2022.
- [15] Zongyue Qin, Haoran Deng, Jason Cong, and Yizhou Sun. Learnable low-rank polynomial sketch for effective linear attention. *arXiv preprint arXiv:2610.14219*, 2026.
- [16] Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations (ICLR)*, 2020. Introduces the PG-19 benchmark.
- [17] E. A. F. Reinhardt and A. J. Hauser. A quantum roadmap for softmax attention: Exact born-rule analogs for softmax attention on the probability simplex. *arXiv preprint arXiv:2608.11173*, 2026.
- [18] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. 2024.
- [19] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.
- [20] Songlin Yang and Yu Zhang. Flash linear attention: efficient triton implementations of linear attention models. <https://github.com/fla-org/flash-linear-attention>, 2024.
- [21] Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2312.06635.
- [22] Yifan Zhang, Zhen Qin, Mengdi Wang, and Quanquan Gu. Higher-order linear attention. *arXiv preprint arXiv:2510.27258*, 2025.