CyFA: Linear Sequence Modeling with Relative-Time-Partitioned Memory

Anonymous authors
Paper under double-blind review

Abstract

Linear recurrent models compress history into a fixed-size state, forcing the same finite memory to accommodate an ever-growing sequence of writes. Existing updates handle this pressure through superposition, forgetting, or replacement, which can make earlier associations difficult to recover. We introduce CyFA (Cyclic Flow Attention), which instead reuses memory by cyclically transporting the surviving history as new information arrives. Earlier writes move away from the age-zero end of the memory while new key–value pairs enter there, producing a relative-time-partitioned state in which multiple historical associations can coexist across the cycle. An exact change of coordinates reduces this moving memory to two scalar-decay linear attention recurrences, enabling efficient chunk-wise training. Across 400M–1.4B pretraining experiments, CyFA consistently outperforms linear attention models on recall-intensive tasks under matched recurrent state sizes. At 400M, it reaches 42.60 on FDA versus 26.07 for KDA, while maintaining strong language modeling and higher training throughput than models with fine-grained gating or Delta Rule updates.

1 Introduction

For several years, Transformers have been the dominant architecture for sequence modeling, combining scalable parallel training with expressive content-dependent interactions (Vaswani et al., 2017). At long sequence lengths, however, full attention incurs quadratic computation, while autoregressive decoding maintains a key–value cache that grows linearly with the context length. Linear recurrent neural networks (Linear RNNs), exemplified by linear attention and modern state space models, offer an efficient alternative by compressing the prefix into a fixed-size recurrent state. This enables linear-time sequence processing and constant-memory decoding (Katharopoulos et al., 2020; Gu et al., 2022; Dao & Gu, 2024). However, this efficiency requires successive writes to share a finite representation. In early linear attention, additive updates therefore superpose increasingly many associations in the same state (Schlag et al., 2021).

Modern models control this pressure through forgetting and replacement. Forget gates attenuate the existing state, Delta Rule updates replace the association aligned with the incoming key, and slot memories forget old contents as their slots are reused (Gu & Dao, 2023; Yang et al., 2024a; Schlag et al., 2021; Yang et al., 2024b; Zhang et al., 2024). These mechanisms have made recurrent models much stronger, but historical recall remains difficult when several earlier associations must coexist.

A natural response is to make forgetting and replacement more selective. This has led to increasingly fine-grained gates and erase-and-write rules, which reduce unnecessary information loss but place more computation inside the recurrence and require more involved chunk-wise algorithms (Kimi Team, 2025; Hatamizadeh et al., 2026; Peng et al., 2025). This leaves open a more basic design choice: **how should the finite coordinates of recurrent memory be reused as new information arrives?**

We introduce CyFA (Cyclic Flow Attention), which reuses memory through cyclic transport.[1] As each new key–value pair arrives, the surviving history advances coherently through the relative-time slots, leaving the age-zero end for the new write. Repeating this process organizes the state by model age: recent information remains near the front of the flow, while earlier information is carried

[1] A shared shift of key and value states around a ring of memory slots.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

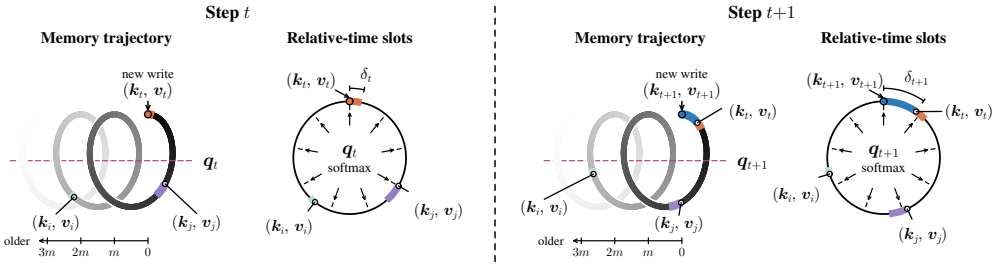

Figure 1: **Cyclic transport in CyFA.** CyFA advances the surviving history along a periodic trajectory while each new write enters at age zero. The learned clock controls the transport distance, and the resulting trajectory is projected onto relative-time slots for softmax readout. Lighter color denotes decay. We omit $\mathbf{R}$ and β_t.

progressively farther around the cycle. A learned model clock makes the amount of transport data dependent. Figure 1 shows the resulting memory evolution.

An exact change of coordinates expresses CyFA as two scalar-decay recurrences with a token-wise slot readout, enabling efficient chunk-wise training. Experiments show that CyFA improves recall and efficiency together under matched main recurrent state sizes: it gains 5.74–6.94 percentage points in average recall accuracy over KDA across 400M–1.4B models (Table 2), while its 400M model achieves 1.37× KDA's training throughput at an 8K sequence length (Figure 5).

2 Background and Preliminaries

2.1 Linear Attention

Linear attention summarizes the key–value prefix in a fixed-size associative state. Let $\boldsymbol{q}_t, \boldsymbol{k}_t \in \mathbb{R}^{d_k}$, $\boldsymbol{v}_t \in \mathbb{R}^{d_v}$, and $\mathbf{S}_t \in \mathbb{R}^{d_k \times d_v}$. The basic causal recurrence is $\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{k}_t \boldsymbol{v}_t^\top$, with readout $\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t = \sum_{i \le t} \boldsymbol{v}_i (\boldsymbol{k}_i^\top \boldsymbol{q}_t)$ (Katharopoulos et al., 2020).

This additive update superposes increasingly many writes in the same matrix (Schlag et al., 2021). Mamba-2 introduces a data-dependent scalar forget gate to reduce interference from stale information (Dao & Gu, 2024). Let α_t denote the forget gate and β_t the strength of the new write. This gives $\mathbf{S}_t = \alpha_t \mathbf{S}_{t-1} + \beta_t \boldsymbol{k}_t \boldsymbol{v}_t^\top$. DeltaNet instead replaces the value currently associated with the incoming key by writing only a residual (Schlag et al., 2021; Yang et al., 2024b): $\mathbf{S}_t = \mathbf{S}_{t-1} + \beta_t \boldsymbol{k}_t \big(\boldsymbol{v}_t - \mathbf{S}_{t-1}^\top \boldsymbol{k}_t\big)^\top$. Gated DeltaNet (GDN) and Kimi Delta Attention (KDA) combine these two ideas with scalar and vector-valued decay, respectively (Schlag et al., 2021; Yang et al., 2025; Kimi Team, 2025).

Chunk-wise parallelism. The recurrent form is efficient for autoregressive decoding, but its token-wise dependency limits parallelism during training. Chunk-wise algorithms reduce this sequential dependency by propagating states only between chunks while computing tokens within each chunk in parallel (Yang & Zhang, 2024). Let each chunk contain C tokens, $\mathbf{S}_{[i]} = \mathbf{S}_{iC}$ be the state after i chunks, and $\mathbf{Q}_{[i+1]}, \mathbf{K}_{[i+1]}, \mathbf{V}_{[i+1]}$ denote the next chunk. Let $\mathbf{M}$ be the within-chunk causal mask. For plain linear attention, the boundary state is $\mathbf{S}_{[i+1]} = \mathbf{S}_{[i]} + \mathbf{K}_{[i+1]}^\top \mathbf{V}_{[i+1]}$, and the chunk output is $\mathbf{O}_{[i+1]} = \mathbf{Q}_{[i+1]} \mathbf{S}_{[i]} + \Big(\mathbf{Q}_{[i+1]} \mathbf{K}_{[i+1]}^\top \odot \mathbf{M}\Big) \mathbf{V}_{[i+1]}$. Scalar forgetting retains this simple structure with cumulative rescaling, whereas fine-grained gates and Delta Rule updates require more involved chunk-wise computation (Yang et al., 2024a; 2025). Table 1 summarizes these update rules as linear attention interfaces together with the relative computational cost of their chunk-wise algorithms.

2.2 Slot Attention

Another way to replace an unbounded key–value cache with fixed-size memory is to maintain m key rows and m value rows while retaining softmax retrieval. Let $\mathbf{K}_t \in \mathbb{R}^{m \times d_k}$ and $\mathbf{V}_t \in \mathbb{R}^{m \times d_v}$. The

Table 1: Representative linear attention interfaces and their relative training costs. For RetNet and Mamba-2, write gates are omitted because they can be absorbed into $\boldsymbol{k}_t$ or $\boldsymbol{v}_t$. The original GLA left-multiplies the state by $\mathrm{Diag}(\boldsymbol{\alpha}_t)$, while the hypothetical GLA (col) right-multiplies it.

Method	State Update	Readout	Linear Attention Interface	Cost
LA	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{LA}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t\}_{t=1}^T)$	Low
RetNet	$\mathbf{S}_t = \gamma\mathbf{S}_{t-1} + \boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{FixedScalarGatedLA}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \gamma\}_{t=1}^T)$	Low
Mamba-2	$\mathbf{S}_t = \alpha_t\mathbf{S}_{t-1} + \boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{ScalarGatedLA}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \alpha_t\}_{t=1}^T)$	Low
GLA	$\mathbf{S}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t)\mathbf{S}_{t-1} + \boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{RowGatedLA}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \boldsymbol{\alpha}_t\}_{t=1}^T)$	Mid
GLA (col)	$\mathbf{S}_t = \mathbf{S}_{t-1}\,\mathrm{Diag}(\boldsymbol{\alpha}_t) + \boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{ColumnGatedLA}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \boldsymbol{\alpha}_t\}_{t=1}^T)$	Mid
DeltaNet	$\mathbf{S}_t = (\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top)\mathbf{S}_{t-1} + \beta_t\boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{DeltaRule}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \beta_t\}_{t=1}^T)$	Mid
GDN	$\mathbf{S}_t = \alpha_t(\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top)\mathbf{S}_{t-1} + \beta_t\boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{ScalarGatedDeltaRule}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \beta_t, \alpha_t\}_{t=1}^T)$	Mid
KDA	$\mathbf{S}_t = (\mathbf{I} - \beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top)\,\mathrm{Diag}(\boldsymbol{\alpha}_t)\mathbf{S}_{t-1} + \beta_t\boldsymbol{k}_t\boldsymbol{v}_t^\top$	$\boldsymbol{o}_t = \mathbf{S}_t^\top \boldsymbol{q}_t$	$\mathrm{RowGatedDeltaRule}(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t, \beta_t, \boldsymbol{\alpha}_t\}_{t=1}^T)$	High

readout is

$$\boldsymbol{o}_t = \mathbf{V}_t^\top \,\mathrm{softmax}(\mathbf{K}_t\boldsymbol{q}_t). \tag{1}$$

Sliding-window attention (SWA) keeps the most recent m pairs exactly(Beltagy et al., 2020). Let $\boldsymbol{e}_i$ be the standard basis vector in $\mathbb{R}^m$ indexed by $i = 0, \ldots, m-1$, and let $\mathbf{Z} = \sum_{i=0}^{m-2} \boldsymbol{e}_{i+1}\boldsymbol{e}_i^\top$. The resulting recurrences are $\mathbf{K}_t = \mathbf{Z}\mathbf{K}_{t-1} + \boldsymbol{e}_0\boldsymbol{k}_t^\top$ and $\mathbf{V}_t = \mathbf{Z}\mathbf{V}_{t-1} + \boldsymbol{e}_0\boldsymbol{v}_t^\top$.

Tokens are discarded once they leave the SWA window. To retain information beyond this horizon, ABC learns a write vector $\boldsymbol{\phi}_t \in \mathbb{R}^m$ that distributes each key–value pair across the slots and allows multiple tokens to accumulate in the same slot (Peng et al., 2022): $\mathbf{K}_t = \mathbf{K}_{t-1} + \boldsymbol{\phi}_t\boldsymbol{k}_t^\top$ and $\mathbf{V}_t = \mathbf{V}_{t-1} + \boldsymbol{\phi}_t\boldsymbol{v}_t^\top$. GSA further introduces a slot-wise forget gate $\boldsymbol{\alpha}_t \in (0,1)^m$, whose complement controls the new write (Zhang et al., 2024): $\mathbf{K}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t)\mathbf{K}_{t-1} + (\mathbf{1} - \boldsymbol{\alpha}_t)\boldsymbol{k}_t^\top$ and $\mathbf{V}_t = \mathrm{Diag}(\boldsymbol{\alpha}_t)\mathbf{V}_{t-1} + (\mathbf{1} - \boldsymbol{\alpha}_t)\boldsymbol{v}_t^\top$.

Two-pass computation. Separating the key- and value-state computations reveals that ABC and GSA can be decomposed into two calls to the linear attention interfaces summarized in Table 1, with a token-wise softmax between them:

$$\begin{array}{cc} \text{ABC} & \text{GSA} \\ \{\boldsymbol{o}'_t\}_{t=1}^T = \mathrm{LA}\big(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{\phi}_t\}_{t=1}^T\big), & \{\boldsymbol{o}'_t\}_{t=1}^T = \mathrm{ColumnGatedLA}\big(\{\boldsymbol{q}_t, \boldsymbol{k}_t, \mathbf{1} - \boldsymbol{\alpha}_t, \boldsymbol{\alpha}_t\}_{t=1}^T\big), \\ \boldsymbol{o}''_t = \mathrm{softmax}(\boldsymbol{o}'_t), & \boldsymbol{o}''_t = \mathrm{softmax}(\boldsymbol{o}'_t), \\ \{\boldsymbol{o}_t\}_{t=1}^T = \mathrm{LA}\big(\{\boldsymbol{o}''_t, \boldsymbol{\phi}_t, \boldsymbol{v}_t\}_{t=1}^T\big) & \{\boldsymbol{o}_t\}_{t=1}^T = \mathrm{RowGatedLA}\big(\{\boldsymbol{o}''_t, \mathbf{1} - \boldsymbol{\alpha}_t, \boldsymbol{v}_t, \boldsymbol{\alpha}_t\}_{t=1}^T\big) \end{array} \tag{2}$$

This decomposition allows them to use hardware-efficient chunk-wise algorithms for linear attention.

3 Cyclic Flow Attention

3.1 Relative-Time-Partitioned Memory

A fixed recurrent memory must repeatedly reuse the same finite set of coordinates. CyFA does so by advancing the existing memory before admitting the next write. The current key–value pair always enters at age zero, while earlier information is carried toward larger model ages by subsequent state transitions. Repeated transport therefore gives the state a relative-time organization.

SWA provides the simplest discrete example of this update pattern. Each step shifts the existing rows by one position and writes the current pair at row zero. However, SWA spends one full slot per token and discards the oldest pair once the window is full. We retain its shift-and-insert structure while removing both restrictions.

Continuous cyclic flow. We first remove the hard boundary of SWA by closing its one-way shift into a cyclic permutation matrix. For m active relative-time slots, let $\boldsymbol{e}_r \in \mathbb{R}^m$ denote the standard basis vector indexed by $r = 0, \ldots, m-1$ and define $\mathbf{P} = \sum_{r=0}^{m-2} \boldsymbol{e}_{r+1}\boldsymbol{e}_r^\top + \boldsymbol{e}_0\boldsymbol{e}_{m-1}^\top$. Existing memory therefore moves toward larger token ages, while the last slot wraps around instead of being deleted. This removes the fixed horizon, but each token still advances memory by one full slot. To allow

several tokens to share the temporal resolution of one slot, we extend $\mathbf{P}$ to fractional shifts. Since $\mathbf{P}$ is circulant, the discrete Fourier basis diagonalizes it (Gray, 2006). We take m to be odd[2] and use the equivalent orthogonal real basis $\mathbf{\Phi} \in \mathbb{R}^{m\times m}$, whose columns are the normalized DC component followed by cosine–sine pairs at frequencies $j = 1, \ldots, (m-1)/2$. With $\mathrm{Rot}(\theta) = \left[\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right]$, define

$$\mathcal{U}(\tau) = 1 \oplus \bigoplus_{j=1}^{(m-1)/2} \mathrm{Rot}\left(\frac{2\pi j\tau}{m}\right), \qquad \mathbf{P}(\tau) = \mathbf{\Phi}\mathcal{U}(\tau)\mathbf{\Phi}^\top, \qquad \tau \in \mathbb{R}. \tag{3}$$

This orthogonal construction satisfies $\mathbf{P}(0) = \mathbf{P}(m) = \mathbf{I}$, $\mathbf{P}(1) = \mathbf{P}$, and $\mathbf{P}(\tau_1)\mathbf{P}(\tau_2) = \mathbf{P}(\tau_1+\tau_2)$. The group law makes successive transport distances additive, while orthogonality makes every shift reversible. Fractional steps therefore move the existing state without the boundary loss of SWA's one-way shift. A fixed fractional step would still assign the same temporal spacing to every token. At step t, a clock increment $\delta_t \in (0, 1)$ sets the distance by which both existing states are shifted. The cumulative clock satisfies $\lambda_t = \lambda_{t-1} + \delta_t$. A write inserted at step i has therefore traveled a total distance $\lambda_t - \lambda_i$ by step t, which we call its model age. Since cyclic flow retains old content rather than truncating it, we also use a scalar forget gate $\alpha_t \in (0, 1)$ (Dao & Gu, 2024; Yang et al., 2025) and a write strength $\beta_t \in (0, 1)$. Let $\mathbf{K}_t \in \mathbb{R}^{m\times d_k}$ and $\mathbf{V}_t \in \mathbb{R}^{m\times d_v}$ denote the key and value states. Their updates are

$$\mathbf{K}_t = \alpha_t\mathbf{P}(\delta_t)\mathbf{K}_{t-1} + \beta_t\boldsymbol{e}_0\boldsymbol{k}_t^\top, \qquad \mathbf{V}_t = \alpha_t\mathbf{P}(\delta_t)\mathbf{V}_{t-1} + \beta_t\boldsymbol{e}_0\boldsymbol{v}_t^\top. \tag{4}$$

The same shift acts on the rows of both states, carrying earlier key–value information together. The current pair is then added at row zero. Following an earlier write through subsequent updates gives

$$\mathbf{K}_t = \sum_{i\le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j\right) \mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0\boldsymbol{k}_i^\top. \tag{5}$$

Each term follows a write from its insertion at row zero through all subsequent shifts. Their composition gives $\mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0$, the write's current relative-time profile. Every later step applies the same orthogonal shift to all existing contributions, preserving their relative geometry during transport. Integer model ages recover exact slot shifts, while fractional ages follow their continuous periodic interpolation. The same expansion holds for the value state with $\boldsymbol{v}_i^\top$ in place of $\boldsymbol{k}_i^\top$; Appendix A details the transport and interpolation properties.

The key and value states now contain aligned contributions transported from earlier writes. We retrieve from learned combinations of their relative-time slots, applying the same readout matrix $\mathbf{R} \in \mathbb{R}^{m\times m}$ to both states. We initialize $\mathbf{R}$ to the identity and compute

$$\boldsymbol{o}_t = (\mathbf{R}\mathbf{V}_t)^\top \mathrm{softmax}(\mathbf{R}\mathbf{K}_t\boldsymbol{q}_t). \tag{6}$$

The matrix $\mathbf{R}$ is applied only at readout; subsequent state updates continue to use Eq. (4).[3]

3.2 Scalar-Decay Recurrence in Absolute-Clock Coordinates

Directly evaluating Eq. (4) would mix all slot rows at every token through a dense $m \times m$ matrix multiplication. This sequential transition precludes the compact matrix-multiply form required for efficient chunk-wise computation (Yang et al., 2024a). We instead represent the same evolving memory in coordinates that account for the accumulated transport without repeatedly shifting the stored matrices. With $\widehat{\mathbf{K}}_t = \mathbf{\Phi}^\top\mathbf{K}_t$, $\widehat{\mathbf{V}}_t = \mathbf{\Phi}^\top\mathbf{V}_t$, and $\boldsymbol{b} = \mathbf{\Phi}^\top\boldsymbol{e}_0$, Eq. (4) becomes

$$\widehat{\mathbf{K}}_t = \alpha_t\mathcal{U}(\delta_t)\widehat{\mathbf{K}}_{t-1} + \beta_t\boldsymbol{b}\boldsymbol{k}_t^\top, \qquad \widehat{\mathbf{V}}_t = \alpha_t\mathcal{U}(\delta_t)\widehat{\mathbf{V}}_{t-1} + \beta_t\boldsymbol{b}\boldsymbol{v}_t^\top. \tag{7}$$

The complex DFT basis diagonalizes the circulant transition, and $\mathcal{U}(\delta_t)$ is the equivalent real block-diagonal representation with the 2×2 rotation blocks in Eq. (3). Equation (7) removes the dense matrix multiplication, but frequency-dependent rotations remain in the recurrence. We next express

[2] For even m, the real Fourier basis contains an additional unpaired Nyquist component. Odd m leaves one DC component and $(m-1)/2$ cosine–sine pairs.

[3] For simplicity, we omit the conventional dot-product scaling factor from the notation throughout.

the state in absolute-clock coordinates. Let $\overline{\mathbf{K}}_t = \mathcal{U}(-\lambda_t)\widehat{\mathbf{K}}_t$ and $\overline{\mathbf{V}}_t = \mathcal{U}(-\lambda_t)\widehat{\mathbf{V}}_t$. Because $\mathcal{U}(-\lambda_t)\mathcal{U}(\delta_t) = \mathcal{U}(-\lambda_{t-1})$, the recurrences reduce to

$$\overline{\mathbf{K}}_t = \alpha_t \overline{\mathbf{K}}_{t-1} + \beta_t \mathcal{U}(-\lambda_t)\boldsymbol{b}\boldsymbol{k}_t^\top, \qquad \overline{\mathbf{V}}_t = \alpha_t \overline{\mathbf{V}}_{t-1} + \beta_t \mathcal{U}(-\lambda_t)\boldsymbol{b}\boldsymbol{v}_t^\top. \tag{8}$$

The relation to the physical shift-and-insert update is explicit:

$$\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_{t-1} = \mathbf{P}(\delta_t)\mathbf{K}_{t-1}, \quad \boldsymbol{\Phi}\mathcal{U}(\lambda_t)\mathcal{U}(-\lambda_t)\boldsymbol{b} = \boldsymbol{e}_0. \tag{9}$$

The temporal write vector is the fixed age-zero insertion expressed in absolute-clock coordinates. Reconstructing at the current clock maps the previous stored state to its shifted version and the new write vector to $\boldsymbol{e}_0$. Thus Eq. (8) implements the same shift-and-insert update as Eq. (4) without explicitly shifting either stored matrix. The stored states themselves undergo **only the scalar decay** α_t.

At readout, the current clock value maps the absolute-clock state back to relative-time slots: $\mathbf{K}_t = \boldsymbol{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_t$ and likewise for $\mathbf{V}_t$. Substituting these relations into Eq. (6) and reassociating the key contraction gives

$$\boldsymbol{o}_t = \overline{\mathbf{V}}_t^\top \mathcal{U}(-\lambda_t)\boldsymbol{\Phi}^\top \mathbf{R}^\top \operatorname{softmax}\big(\mathbf{R}\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_t\boldsymbol{q}_t\big). \tag{10}$$

3.3 Two-Pass Chunk-Wise Computation

Equation (8) now has exactly the scalar-decay, rank-one update used by the ScalarGatedLA interface in Table 1. This lets us realize the key and value states as two scalar-gated linear attention calls, connected only by the token-wise slot readout:

$$\begin{aligned}
\{\boldsymbol{o}'_t\}_{t=1}^T &= \text{ScalarGatedLA}\left(\{\boldsymbol{q}_t,\, \boldsymbol{k}_t,\, \beta_t\mathcal{U}(-\lambda_t)\boldsymbol{b},\, \alpha_t\}_{t=1}^T\right), \\
\boldsymbol{o}''_t &= \mathcal{U}(-\lambda_t)\boldsymbol{\Phi}^\top \mathbf{R}^\top \operatorname{softmax}(\mathbf{R}\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\boldsymbol{o}'_t), \\
\{\boldsymbol{o}_t\}_{t=1}^T &= \text{ScalarGatedLA}\left(\{\boldsymbol{o}''_t,\, \beta_t\mathcal{U}(-\lambda_t)\boldsymbol{b},\, \boldsymbol{v}_t,\, \alpha_t\}_{t=1}^T\right).
\end{aligned} \tag{11}$$

The first pass contracts the stored key state with $\boldsymbol{q}_t$. Reconstructing relative-time coordinates after this contraction gives exactly $\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\boldsymbol{o}'_t = \mathbf{K}_t\boldsymbol{q}_t$, so the slot transform acts on an m-dimensional vector rather than the full key state. After learned slot mixing and softmax, the transpose readout map converts the weights for the second pass over the stored value state. Neither transported state needs to be materialized at each token.

The shared clock sequence $\{\lambda_t\}_{t=1}^T$ is obtained from $\{\delta_t\}_{t=1}^T$ by a parallel prefix sum. Since the middle map is token-wise, the two recurrent passes can independently use existing hardware-efficient chunk-wise algorithms for ScalarGatedLA (Dao & Gu, 2024; Yang & Zhang, 2024), which are more efficient than the algorithms required by fine-grained gating and Delta Rule updates (Yang et al., 2024a; 2025; Kimi Team, 2025). A detailed formulation is given in Section B.

Relation to Blurry Window Attention. Blurry Window Attention (BLA) (Laborieux et al., 2026) uses the same Fourier–Dirichlet profile under the fixed-rate clock $\delta_t \equiv 1/\rho$. This specialization matches the temporal profile, while the full state update and readout remain different. Appendix A.6 gives the detailed comparison.

3.4 Network Design

Neural architecture. Following the LLaMA-style Transformer architecture (Touvron et al., 2023), we form the backbone by replacing standard self-attention with CyFA and stacking CyFA layers with SwiGLU feed-forward layers (Shazeer, 2020), as illustrated in Figure 2. For each CyFA head, linear projections produce $\boldsymbol{q}_t^h$, $\boldsymbol{k}_t^h$, $\boldsymbol{v}_t^h$, the clock increment δ_t^h, the forget gate α_t^h, and the write strength β_t^h. The forget gate follows the parameterization in prior works (Dao & Gu, 2024; Yang et al., 2025). Each head uses a learned readout matrix $\mathbf{R}^h$, initialized to $\mathbf{I}$. A short convolution is applied to $\boldsymbol{q}$, $\boldsymbol{k}$, and $\boldsymbol{v}$ (Gu & Dao, 2023; Dao & Gu, 2024; Yang et al., 2025; Kimi Team, 2025). We apply RMSNorm to both queries and keys for QK normalization (Qwen Team, 2026). The output transformation consists of a head-wise RMSNorm (Qin et al., 2022; Lan et al., 2025; Yang et al., 2024b; 2025; Kimi Team, 2025) and an elementwise output gate from a low-rank sigmoid projection (Qiu et al., 2025; Kimi Team, 2025).

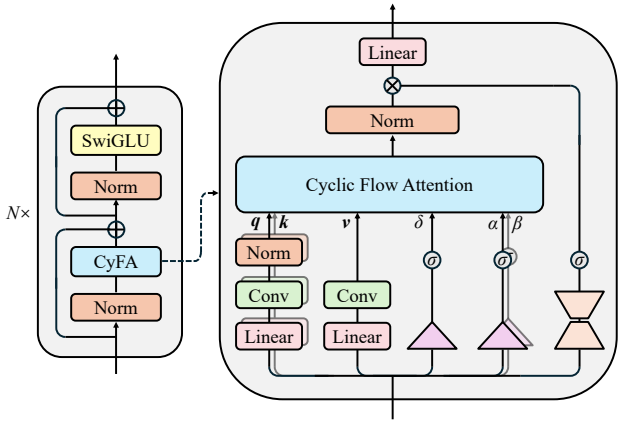

Figure 2: Overall architecture of CyFA. We use $\alpha_t = \exp(-A\ \mathrm{softplus}(\mathbf{W}_\alpha \boldsymbol{x}_t + \boldsymbol{b}_\alpha))$ but omit it for brevity.

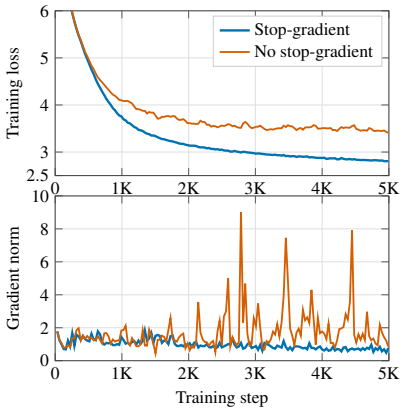

Figure 3: Training loss and gradient norm with and without stop-gradient.

Clock gradient stabilization. Each clock increment changes the subsequent transport of the existing history and can therefore affect later outputs. In the absolute-clock implementation, these dependencies pass through the cumulative clock. In practice, backpropagation through this long path produced exploding gradients. We therefore apply stop-gradient to the layer input of the clock-increment projection:

$$\delta_t^h = \sigma((\mathbf{W}_\delta\, \mathrm{sg}(\boldsymbol{x}_t) + \boldsymbol{b}_\delta)_h)\,, \tag{12}$$

Gradients through the temporal write vectors and token-wise readout continue to train $\mathbf{W}_\delta$ and $\boldsymbol{b}_\delta$, while no gradient from the clock-increment branch flows into $\boldsymbol{x}_t$. Figure 3 shows that stop-gradient yields a smooth decrease in training loss and stable gradient norms, whereas removing it produces persistent loss oscillations and repeated gradient spikes after the initial descent. Section D.1 analyzes the source of this instability.

4 Experiments

Experimental setup. We compare CyFA with a LLaMA-style Transformer baseline (Vaswani et al., 2017; Touvron et al., 2023) and representative linear architectures, including SWA (Beltagy et al., 2020), GSA (Zhang et al., 2024), Raven (Afzal et al., 2026), BLA (Laborieux et al., 2026), Mamba-2 (Dao & Gu, 2024), Mamba-3 (Lahoti et al., 2026), GDN (Yang et al., 2025), and KDA (Kimi Team, 2025). We pretrain all models at approximately 400M parameters and extend selected comparisons to 800M and 1.4B. For CyFA, we use $m = 127$ active slots.[4] At each scale, we adjust the number of heads and their state dimensions so that all recurrent models have the same total state size.[5] At 1.4B, we additionally evaluate publicly released checkpoints from the FLA model collection on Hugging Face[6] (Yang & Zhang, 2024).

We implement all models with the `flash-linear-attention` library (Yang & Zhang, 2024). Pretraining uses the SlimPajama-627B dataset (Soboleva et al., 2023). All models share the Mistral tokenizer (Jiang et al., 2023). Initial pretraining uses a context length of 2,048 tokens. The 400M, 800M, and 1.4B models receive 15B, 30B, and 100B training tokens, respectively. Corresponding global batches contain 0.5M tokens at the 400M and 800M scales and 1M tokens at the 1.4B scale. For the 800M and 1.4B models evaluated in Section 4.2, we continue pretraining for 1,024 steps with a context length of 8,192 tokens to improve their long-context ability. Across scales, optimization uses AdamW with a peak learning rate of 3×10^{-4} and a weight decay of 0.01 (Loshchilov & Hutter, 2019). The learning-rate schedule uses 1,000 warmup steps followed by cosine decay to 3×10^{-5} (Loshchilov & Hutter, 2017). We clip the gradient norm at 1.0. Training runs on eight NVIDIA RTX Pro 6000 GPUs. Additional experimental details are provided in Section C.

[4] To retain an odd number of active slots while using a hardware-friendly power-of-two state dimension, we allocate $m^* = m + 1 = 128$ physical rows and mask the padding row during writing and readout.

[5] The public checkpoints do not follow the matched state size, so their results are provided for reference.

[6] `https://huggingface.co/fla-hub`

Table 2: Main language modeling results. Commonsense Reasoning Tasks are evaluated with `lm-evaluation-harness` (Gao et al., 2023). Recall-Intensive Tasks are evaluated with `prefix-linear-attention` (Arora et al., 2024), with inputs truncated to 2K tokens. Public checkpoints are marked with †. Transformer is excluded from ranking among recurrent models.

Model	Perplexity		Commonsense Reasoning Tasks							Recall-Intensive Tasks						
	Wiki. ppl↓	**Lamb.** ppl↓	**ARC-e** acc↑	**ARC-c** acc_n ↑	**Hella.** acc_n ↑	**Lamb.** acc↑	**PIQA** acc↑	**Wino.** acc↑	**Avg.** acc↑	**FDA** acc↑	**SWDE** acc↑	**SQD** acc↑	**NQ** acc↑	**TQA** acc↑	**DROP** acc↑	**Avg.** acc↑
400M parameters with 15B training tokens, $L = 24$, and $d = 1024$																
Transformer	27.30	74.28	44.74	24.23	35.25	26.02	64.69	51.46	41.07	53.04	38.89	30.70	16.41	45.68	19.17	33.98
SWA	33.29	89.22	46.13	22.61	34.52	25.05	63.71	49.33	40.23	15.62	8.72	21.46	5.29	31.52	16.20	16.47
GSA	28.56	87.55	46.25	**25.00**	34.40	23.85	64.80	51.78	41.01	5.54	14.15	23.65	10.90	41.05	17.54	18.81
Raven	32.14	91.49	43.81	23.21	32.58	24.22	64.42	50.91	39.86	19.80	25.77	25.73	9.76	40.17	15.86	22.85
BLA ($\rho = 2$)	32.88	79.86	44.49	22.78	33.90	25.21	64.31	50.75	40.24	12.62	11.72	28.08	5.67	38.86	16.63	18.93
BLA ($\rho = 5$)	33.87	173.33	43.52	23.98	32.18	19.02	63.71	51.30	38.95	5.99	16.40	23.55	6.72	39.63	14.37	17.78
Mamba-2	28.25	89.36	45.75	22.61	34.95	22.34	63.98	52.33	40.33	9.08	21.18	25.13	11.97	40.23	16.10	20.62
Mamba-3	26.75	49.99	46.46	22.70	36.16	27.96	64.31	53.12	41.79	24.89	27.84	29.53	14.92	43.84	17.73	26.46
GDN	27.10	62.92	43.94	23.21	35.25	25.91	64.42	51.78	40.75	17.35	23.81	27.31	15.11	42.36	18.78	24.12
KDA	**25.76**	51.17	**47.10**	23.21	36.50	27.36	**65.51**	51.70	41.90	26.07	25.02	29.49	14.82	45.38	17.06	26.31
CyFA	25.85	**48.17**	45.92	24.06	**36.84**	**29.26**	65.29	**53.43**	**42.47**	**42.60**	**35.61**	**34.63**	**17.33**	**48.70**	**20.65**	**33.25**
800M parameters with 30B training tokens, $L = 24$, and $d = 1536$																
Transformer	20.41	23.32	50.29	24.83	42.97	36.99	67.90	50.67	45.61	49.32	42.92	39.17	22.24	54.44	21.27	38.23
GDN	20.44	25.42	52.02	**27.05**	42.87	36.06	68.50	53.75	46.71	34.15	28.58	32.65	18.37	52.61	19.26	30.94
KDA	19.83	22.20	52.36	25.51	44.03	36.76	68.66	53.51	46.81	28.25	30.18	33.93	21.95	53.55	19.21	31.18
CyFA	**19.51**	**18.93**	**53.62**	26.11	**44.65**	**39.74**	**70.13**	**54.70**	**48.16**	**43.69**	**40.58**	**39.03**	**24.17**	**55.21**	**20.12**	**37.13**
1.4B parameters with 100B training tokens, $L = 24$, and $d = 2048$																
Transformer†	17.65	18.61	56.02	28.33	49.11	40.95	69.86	54.85	49.85	55.31	44.70	43.10	24.58	59.12	21.66	41.41
RetNet†	18.23	23.19	57.11	26.88	48.08	37.36	68.88	54.22	48.76	20.98	27.18	33.79	15.55	53.44	19.65	28.43
GLA†	17.68	19.72	55.18	27.30	48.80	40.33	70.02	52.96	49.10	27.25	31.12	34.73	22.27	55.45	19.26	31.68
GSA†	16.81	15.69	58.75	28.33	50.96	41.98	71.93	52.72	50.78	23.89	29.99	35.98	23.15	57.46	20.89	31.89
KDA	15.97	**11.88**	58.75	**28.41**	53.85	**47.76**	71.87	56.67	52.89	42.60	40.77	37.79	24.83	58.89	21.47	37.73
CyFA	**15.47**	12.77	**60.98**	28.24	**54.32**	46.38	**72.09**	**57.14**	**53.19**	**57.77**	**46.58**	**42.83**	**29.87**	**61.49**	**22.28**	**43.47**

4.1 Language Modeling

Commonsense Reasoning. Following prior work (Zhang et al., 2024), we evaluate zero-shot performance on ARC-Easy (ARC-e) and ARC-Challenge (ARC-c) (Clark et al., 2018), HellaSwag (Hella.) (Zellers et al., 2019), LAMBADA (Lamb.) (Paperno et al., 2016), PIQA (Bisk et al., 2020), and WinoGrande (Wino.) (Sakaguchi et al., 2021). These tasks typically involve short inputs and place limited demands on in-context retrieval, allowing recurrent models to perform well and sometimes surpass the Transformer baseline. As shown in Table 2, CyFA achieves the highest average accuracy at every model scale, highlighting the potential of the architecture.

Recall-Intensive Tasks. Following the protocol of Arora et al. (2024), we evaluate real-world recall-intensive tasks, including FDA (Arora et al., 2023), SWDE (Arora et al., 2023; Lockard et al., 2019), SQD (Rajpurkar et al., 2018), NQ (Kwiatkowski et al., 2019), TQA (Joshi et al., 2017), and DROP (Dua et al., 2019). These benchmarks expose the recall gap between compressed recurrent states and Transformers (Arora et al., 2024). As shown in Table 2, CyFA is the strongest recurrent model **on every task at every scale** and sharply narrows this gap within the 2K context. At 1.4B, CyFA achieves a higher average recall score than the public Transformer checkpoint (43.47 vs. 41.41), although their training configurations differ. The largest gains occur on FDA and SWDE, where the model must retrieve one value among many coexisting field–value associations.

4.2 Long-Context Ability

For the evaluations in this section, we continue pretraining each model for 1,024 steps at an 8K context length to activate its long-context capabilities. Section E reports results before continuation.

Long-Context Understanding. We evaluate long-context understanding with LongBench (Bai et al., 2024). As shown in Table 3, CyFA achieves the best average score at both scales and leads most tasks, with particularly strong gains on Code. KDA leads the Few-Shot TREC (TRE) task, while CyFA's broader advantage across categories demonstrates robust long-context understanding.

Table 3: LongBench results after 1,024-step continuation at an 8K training context, with inputs truncated to 8K tokens.

Scale	Model	Code		Summarization			SingleQA			MultiQA			Few-Shot			Avg.
		LCC	RBP	GvR	QMS	MNs	NQA	QQA	MQA	HQA	2WM	MSQ	TRE	TQA	SAM	
800M	Transformer	42.46	37.20	0.68	11.41	0.56	1.54	3.70	6.32	3.58	7.97	2.40	16.50	26.21	5.50	11.86
	GDN	43.65	35.64	**5.93**	10.95	2.14	3.32	1.67	11.58	**5.02**	8.79	2.62	31.00	26.04	**15.61**	14.57
	KDA	41.47	39.06	5.39	11.54	2.22	2.72	5.29	10.88	3.90	8.12	2.42	**47.50**	25.55	10.47	15.47
	CyFA	**47.84**	**43.59**	3.54	**15.43**	**5.57**	**3.50**	**6.17**	**14.03**	4.63	**10.27**	**2.87**	32.50	**37.78**	12.90	**17.19**
1.4B	KDA	46.34	42.56	4.26	13.75	7.68	1.97	1.55	7.93	**6.40**	8.66	3.54	**57.50**	49.09	**29.11**	20.02
	CyFA	**53.60**	**46.40**	**5.98**	**14.50**	**11.58**	**2.82**	**5.71**	**14.85**	5.84	**10.08**	**4.88**	43.50	**56.37**	26.91	**21.64**

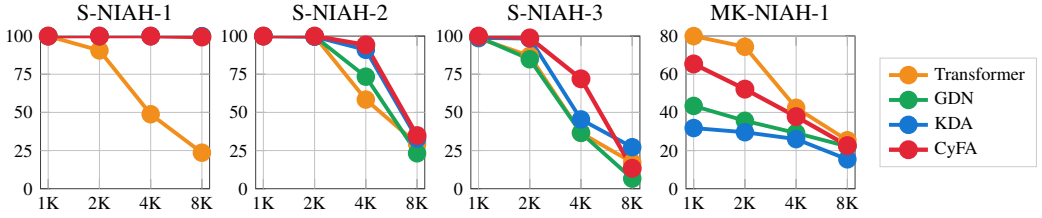

Figure 4: RULER accuracy of 800M models after 1,024-step continuation at an 8K training context.

Needle-In-A-Haystack (NIAH). We evaluate long-context retrieval with RULER (Hsieh et al., 2024). Figure 4 shows that the single-needle S-NIAH-1/2/3 tasks are comparatively favorable to recurrent models. Moving to MK-NIAH-1 causes a large task-level drop for GDN and KDA even at 1K–2K. CyFA incurs a substantially smaller drop at these lengths and remains clearly ahead through 4K, demonstrating greater tolerance to multiple coexisting key–value associations.

4.3 Efficiency

Figure 5 shows that CyFA's efficient two-pass scalar-decay formulation delivers higher end-to-end training throughput than models with more complex recurrent updates. Among the recurrent baselines, it trails only Mamba-2, which uses a single-pass scalar recurrence.

4.4 Ablations

Table 4 presents the ablation results. For method design, we ablate forgetting ($\alpha_t = 1$), data-dependent clocking by learning one token-independent $\delta_t^h \equiv \sigma(b_\delta^h) \in (0, 1)$ per head, and the learned readout matrix ($\mathbf{R} = \mathbf{I}$). For architecture, we assess ShortConv and QK normalization and test the SiLU feature map on $\boldsymbol{q}$, $\boldsymbol{k}$, and $\boldsymbol{v}$ commonly used in linear attention (Yang et al., 2025; Kimi Team, 2025). With the existing softmax nonlinearity, SiLU provides no clear benefit, so CyFA does not include this feature map. Varying m further shows that recall depends strongly on the number of relative-time slots.

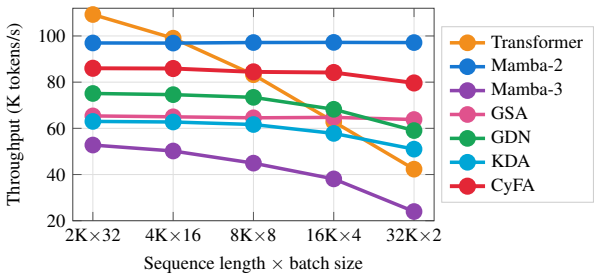

Figure 5: End-to-end training throughput of 400M models. Raven and BLA are omitted because both reuse the GSA computation.

Table 4: CyFA ablations at 400M scale.

Variant	Wiki. ppl↓	Lamb. ppl↓	Common Avg.↑	Recall Avg.↑
CyFA w. $m = 127$	25.85	48.17	42.47	33.25
Method				
w. $\alpha_t = 1$	30.93	115.89	39.61	28.05
w. $\delta_t^h \equiv \sigma(b_\delta^h)$	26.40	51.98	42.15	30.32
w. $\mathbf{R} = \mathbf{I}$	26.26	58.75	41.89	31.79
Architecture				
w/o. ShortConv	26.34	52.05	42.04	31.70
w/o. QK Norm	26.29	51.37	42.61	30.12
w. SiLU	25.77	45.11	42.50	32.26
Number of Slots				
w. $m = 63$	26.38	51.34	42.18	26.68
w. $m = 255$	25.31	46.79	42.82	34.83

4.5 Clock Increment Analysis

Across the 96 heads of the 400M CyFA model, 56 (58.3%) have δ_t concentrated near 1, while 40 (41.7%) exhibit substantial input-dependent variation in δ_t. Within the input-dependent heads, we further observe that the same token can receive markedly different clock increments across semantic contexts (Table 5).

Table 5: Case study on δ_t.

Head / Word	Context	δ_t
L15H3 `about`	"about 45 km"	0.355
	"thought about what causes"	0.057
L13H2 `single`	Musical noun	0.815
	Numerical modifier	0.297

5 Related Work

Linear recurrent models improve fixed-size memory through increasingly expressive state updates. RetNet and Mamba-2 use fixed or data-dependent scalar decay, while GLA and HGRN2 use fine-grained vector-valued decay (Sun et al., 2023; Dao & Gu, 2024; Yang et al., 2024a; Qin et al., 2024). DeltaNet adds key-dependent correction, extended by Gated DeltaNet and KDA with scalar and vector-valued decay (Schlag et al., 2021; Yang et al., 2025; Kimi Team, 2025). Recent work introduces low-rank closed-loop feedback (Hu et al., 2025), vector-valued replacement (Peng et al., 2025), multiple Householder updates (Siems et al., 2025), preconditioned or momentum delta updates (Tumma et al., 2026; Huang et al., 2026), decoupled erase/write gates or addresses (Hatamizadeh et al., 2026; Li et al., 2026), and uncertainty-aware residual writes (Bui et al., 2026). These recurrences remain chunk-wise trainable, but require richer block algebra. Delta-rule kernels use compact WY representations and UT transforms (Yang et al., 2024b), while combining fine-grained decay with low-rank correction yields DPLR transitions (Kimi Team, 2025). Instead, CyFA reuses finite state coordinates through cyclic transport, with scalar decay plus rank-one writes in both recurrent passes.

Another line reduces interference by partitioning or expanding recurrent memory. Slot memories learn input-dependent write allocation, from dense controls in ABC and GSA to sparse Top-K routing in Raven (Peng et al., 2022; Zhang et al., 2024; Afzal et al., 2026). Other approaches increase representational capacity through multiple independent recurrent states (Du et al., 2026), sparse state expansion (Pan et al., 2025; Cabannes et al., 2026), or higher-order tensor states (Herranz-Celotti & Guigue, 2026). HOLA instead augments the recurrent state with a bounded exact KV memory (Cui, 2026). Temporal multi-state models maintain several summaries of the past: Log-Linear Attention uses a logarithmically growing Fenwick hierarchy (Guo et al., 2025), DLA maintains a bounded chronological state cache through adaptive merging (Wang et al., 2026), and DART decodes attention keys and values from retained Mamba-2 chunk states (Qian et al., 2026). These methods change the memory budget or maintain multiple recurrent states, whereas our matched-state setting asks how the same finite coordinates should be reused as new writes arrive. In Log-Linear Attention and DLA, cross-partition weights are predicted from the current representation rather than obtained by matching the query against keys stored in each partition. BLA is the closest fixed-state temporal construction, using a fixed-resolution Fourier–Dirichlet window (Laborieux et al., 2026); Section A.6 gives a detailed comparison. CyFA cyclically transports the existing history before each age-zero insertion. A learned clock controls the transport distance, while slot activation remains content-based.

Rotational and complex dynamics also appear in recurrent models, from fixed phases in RetNet to data-dependent rotations in Selective RoPE and Mamba-3 (Sun et al., 2023; Movahedi et al., 2026; Lahoti et al., 2026). These rotations act on content-carrying feature or query/key channels. CyFA instead applies them along the relative-time slot axis shared by the key and value states. In absolute-clock coordinates, the recurrence contains only scalar decay and rank-one writes.

6 Conclusion

CyFA introduces a different principle for reusing finite recurrent memory. Before each age-zero insertion, cyclic transport advances the surviving history through the finite state. Repeated transport forms a moving memory organized by relative model time, distributing associations around the cycle. An exact coordinate reformulation preserves efficient scalar-decay computation. Across model scales, CyFA strengthens recall-intensive performance under matched state sizes while maintaining language-modeling quality and high throughput, establishing state organization as a central design dimension for recurrent sequence models.

AI USE STATEMENT

Generative AI tools were used to aid or polish the writing, organize visual materials, and assist with manuscript formatting. The authors reviewed all AI-assisted content and take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

Section 3 specifies the CyFA recurrence, temporal readout, two-pass computation, and network design. Section A provides the complete temporal-coordinate derivations, and Section B details the chunk-wise computation. The datasets and evaluation protocols are described in Sections 4.1 and 4.2, while Section C reports the model configurations, state-size matching, optimization settings, training schedules, and long-context continuation setup. Additional diagnostic and intervention results are provided in Section D.

REFERENCES

Arshia Afzal, Aviv Bick, Eric P. Xing, Volkan Cevher, and Albert Gu. Raven: High-recall sequence modeling with sparse memory routing. *arXiv preprint arXiv:2607.25357*, 2026.

Simran Arora, Brandon Yang, Sabri Eyuboglu, Avanika Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. Language models enable simple systems for generating structured views of heterogeneous data lakes. *Proceedings of the VLDB Endowment*, 17(2):92–105, 2023. doi: 10.14778/3626292.3626294. URL https://www.vldb.org/pvldb/vol17/p92-arora.pdf.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: Closing the recall gap for recurrent language models. *arXiv preprint arXiv:2407.05483*, 2024.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 3119–3137, 2024. doi: 10.18653/v1/2024.acl-long.172.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 7432–7439, 2020. doi: 10.1609/aaai.v34i05.6239.

Ngoc Bui, Tinglin Huang, and Rex Ying. Kalman delta networks: Uncertainty-aware associative memory. *arXiv preprint arXiv:2609.07816*, 2026. URL https://arxiv.org/abs/2609.07816.

Loïc Cabannes, Pierre-Emmanuel Mazaré, Gergely Szilvasy, Matthijs Douze, Maria Lomeli, Ilze Amanda Auzina, Justin Carpentier, Gabriel Synnaeve, and Hervé Jégou. Sparse delta memory: Scaling the state of linear RNNs through sparsity. *arXiv preprint arXiv:2607.07386*, 2026.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Wanyun Cui. A hippocampus for linear attention: An exact memory for what the recurrent state forgets. *arXiv preprint arXiv:2607.02303*, 2026. URL https://arxiv.org/abs/2607.02303.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning*, 2024.

Jusen Du, Weigao Sun, Disen Lan, Jiaxi Hu, Tao Zhang, and Yu Cheng. MoM: Linear sequence modeling with mixture-of-memories. In *International Conference on Learning Representations*, 2026.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2368–2378, 2019. doi: 10.18653/v1/N19-1246.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 2023. URL https://zenodo.org/records/10256836.

Robert M. Gray. Toeplitz and circulant matrices: A review. *Foundations and Trends in Communications and Information Theory*, 2(3):155–239, 2006. doi: 10.1561/0100000006.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Han Guo, Songlin Yang, Tarushii Goel, Eric P. Xing, Tri Dao, and Yoon Kim. Log-linear attention. *arXiv preprint arXiv:2506.04761*, 2025.

Ali Hatamizadeh, Yejin Choi, and Jan Kautz. Gated DeltaNet-2: Decoupling erase and write in linear attention. *arXiv preprint arXiv:2605.22791*, 2026.

Luca Herranz-Celotti and Vincent Guigue. RunningTensor: Generalizing linear attention to higher-order recurrent states. *arXiv preprint arXiv:2609.12814*, 2026. URL https://arxiv.org/abs/2609.12814.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Jiaxi Hu, Yongqi Pan, Jusen Du, Disen Lan, Xiaqiang Tang, Qingsong Wen, Yuxuan Liang, and Weigao Sun. Improving bilinear RNN with closed-loop control. In *Advances in Neural Information Processing Systems 38*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/9a439efaa34fe37177eba00737624824-Abstract-Conference.html.

Yulong Huang, Xiang Liu, Hongxiang Huang, Xiaopeng Lin, Zunchang Liu, Xiaowen Chu, Zeke Xie, and Bojun Cheng. MDN: Parallelizing stepwise momentum for delta linear attention. *arXiv preprint arXiv:2605.05838*, 2026.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1601–1611, 2017. doi: 10.18653/v1/P17-1147.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, 2020.

Kimi Team. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019. doi: 10.1162/tacl_a_00276.

Axel Laborieux, Christos Sourmpis, Juan Gabriel Kostelec, and Qinghai Guo. Blurry window attention. *arXiv preprint arXiv:2606.09862*, 2026.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. In *International Conference on Learning Representations*, 2026.

Disen Lan, Weigao Sun, Jiaxi Hu, Jusen Du, and Yu Cheng. Liger: Linearizing large language models to gated recurrent structures. In *Forty-second International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/lan25b.html.

Xiao Li, Chengruidong Zhang, Hao Luo, Xi Lin, Zekun Wang, Zihan Qiu, Yunfei Mao, Langshi Chen, Man Yuan, Minmin Sun, Huiqiang Jiang, Siqi Zhang, Rui Men, Wei Hu, Gong Cheng, Bo Zheng, Dayiheng Liu, and Jingren Zhou. Erase-then-delta attention: Decoupling erase and write addresses in delta-rule linear attention. *arXiv preprint arXiv:2606.26560*, 2026.

Colin Lockard, Prashant Shiralkar, and Xin Luna Dong. OpenCeres: When open information extraction meets the semi-structured web. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3047–3056, 2019. doi: 10.18653/v1/N19-1309.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Sajad Movahedi, Timur Carstensen, Arshia Afzal, Frank Hutter, Antonio Orvieto, and Volkan Cevher. Selective rotary position embedding. In *International Conference on Learning Representations*, 2026.

Yuqi Pan, Yongqi An, Zheng Li, Yuhong Chou, Ruijie Zhu, Xiaohui Wang, Mingxuan Wang, Jinqiao Wang, and Guoqi Li. Scaling linear attention with sparse state expansion. *arXiv preprint arXiv:2507.16577*, 2025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 1525–1534, 2016. doi: 10.18653/v1/P16-1144.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, Jiaju Lin, Jiaxing Liu, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, Nathan Wilce, Johan S. Wind, Tianyi Wu, Daniel Wuttke, and Christian Zhou-Zheng. RWKV-7 "goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*, 2025. URL https://arxiv.org/abs/2503.14456v2.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. ABC: Attention with bounded-memory control. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

Yixiao Qian, Song Chen, Pengkai Wang, Jiaxu Liu, Shengze Cai, and Chao Xu. DART: Decoded attention over recurrent states for efficient long-context sequence modeling. *arXiv preprint arXiv:2608.02032*, 2026. URL https://arxiv.org/abs/2608.02032.

Zhen Qin, Xiaodong Han, Weixuan Sun, Dongxu Li, Lingpeng Kong, Nick Barnes, and Yiran Zhong. The devil in linear transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 7025–7041. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.EMNLP-MAIN.473. URL https://doi.org/10.18653/v1/2022.emnlp-main.473.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated linear RNNs with state expansion. *arXiv preprint arXiv:2404.07904*, 2024.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Advances in Neural Information Processing Systems 38*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/904e89bb4e632e75fb47f093b620b257-Abstract-Conference.html.

Qwen Team. Qwen3.5: Towards native multimodal agents. Qwen Technical Blog, 2026. URL https://qwen.ai/blog?id=qwen3.5.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 784–789, 2018. doi: 10.18653/v1/P18-2124.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021. doi: 10.1145/3474381. URL https://doi.org/10.1145/3474381.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, 2021.

Noam Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. DeltaProduct: Improving state-tracking in linear RNNs via householder products. In *Advances in Neural Information Processing Systems*, 2025.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R. Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token, cleaned and deduplicated version of RedPajama. Cerebras blog, June 2023. URL https://www.cerebras.ai/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama.

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.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Neehal Tumma, Noel Loo, and Daniela Rus. Preconditioned DeltaNet: Curvature-aware sequence modeling for linear recurrences. *arXiv preprint arXiv:2604.21100*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Xin Wang, Hui Shen, Boyuan Zheng, Xueshen Liu, Minkyoung Cho, Zhongwei Wan, Zesen Zhao, Zhuoqing Mao, Shen Yan, and Mi Zhang. Dynamic linear attention. *arXiv preprint arXiv:2606.10650*, 2026.

Songlin Yang and Yu Zhang. FLA: A triton-based library for hardware-efficient implementations of linear attention mechanism. https://github.com/fla-org/flash-linear-attention, January 2024.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56501–56523. PMLR / OpenReview.net, 2024a. URL https://proceedings.mlr.press/v235/yang24ab.html.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*, 2024b.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=r8H7xhYPwz.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

Yu Zhang, Songlin Yang, Ruijie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling. In *Advances in Neural Information Processing Systems*, 2024.

A CYCLIC TRANSPORT AND RELATIVE-TIME COORDINATES

The main text defines CyFA through cyclic transport in relative-time slots and then derives the absolute-clock coordinates used for computation. This section develops their equivalence and geometry in detail. Throughout, m is the odd number of active relative-time slots, $r, s \in \{0, \dots, m-1\}$ index those slots, and λ_t denotes the clock value defined in Section 3.1. The padded physical row used by the implementation is discussed at the end of this section.

A.1 CONTINUOUS CYCLIC SHIFTS IN A REAL FOURIER BASIS

We begin by making the real Fourier basis in Eq. (3) explicit. Its columns are indexed by one DC component followed by cosine–sine pairs:

$$\mathbf{\Phi}_{r,0} = \frac{1}{\sqrt{m}}, \qquad \mathbf{\Phi}_{r,2j-1} = \sqrt{\frac{2}{m}} \cos\left(\frac{2\pi jr}{m}\right), \qquad \mathbf{\Phi}_{r,2j} = \sqrt{\frac{2}{m}} \sin\left(\frac{2\pi jr}{m}\right), \tag{13}$$

for $j = 1, \dots, (m-1)/2$. Consequently,

$$\boldsymbol{b} = \mathbf{\Phi}^\top \boldsymbol{e}_0 = \left[\frac{1}{\sqrt{m}}, \sqrt{\frac{2}{m}}, 0, \dots, \sqrt{\frac{2}{m}}, 0\right]^\top. \tag{14}$$

Proposition 1 (continuous cyclic-shift group). The matrix $\mathbf{\Phi}$ is orthogonal. For every $\tau \in \mathbb{R}$, $\mathcal{U}(\tau)$ and $\mathbf{P}(\tau) = \mathbf{\Phi}\mathcal{U}(\tau)\mathbf{\Phi}^\top$ are orthogonal, and

$$\mathbf{P}(\tau_1)\mathbf{P}(\tau_2) = \mathbf{P}(\tau_1 + \tau_2), \qquad \mathbf{P}(\tau)^{-1} = \mathbf{P}(-\tau), \qquad \mathbf{P}(\tau + m) = \mathbf{P}(\tau). \tag{15}$$

Moreover, $\mathbf{P}(1)$ is exactly the cyclic permutation $\mathbf{P}$ defined in Section 3.1.

Proof. The standard roots-of-unity identity

$$\sum_{r=0}^{m-1} \exp\left(\frac{2\pi \mathrm{i}\ell r}{m}\right) = \begin{cases} m, & \ell \equiv 0 \pmod m, \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

implies orthogonality of the DC, cosine, and sine columns in Eq. (13); the chosen normalizations therefore give $\mathbf{\Phi}^\top\mathbf{\Phi} = \mathbf{I}$. Each block of $\mathcal{U}(\tau)$ is a planar rotation, so $\mathcal{U}(\tau)^\top = \mathcal{U}(-\tau)$ and $\mathcal{U}(\tau_1)\mathcal{U}(\tau_2) = \mathcal{U}(\tau_1 + \tau_2)$. Conjugating these identities by $\mathbf{\Phi}$ yields Eq. (15). Since every frequency is an integer multiple of $2\pi/m$, $\mathcal{U}(m) = \mathbf{I}$.

It remains to identify the unit shift. Let the j-th cosine and sine columns of $\mathbf{\Phi}$ be denoted by their sampled functions. The cyclic permutation defined in Section 3.1 acts as $(\mathbf{P}\boldsymbol{x})_r = \boldsymbol{x}_{r-1 \bmod m}$, and hence

$$\mathbf{P}\cos\left(\frac{2\pi jr}{m}\right) = \cos\left(\frac{2\pi j}{m}\right)\cos\left(\frac{2\pi jr}{m}\right) + \sin\left(\frac{2\pi j}{m}\right)\sin\left(\frac{2\pi jr}{m}\right), \tag{17}$$

$$\mathbf{P}\sin\left(\frac{2\pi jr}{m}\right) = -\sin\left(\frac{2\pi j}{m}\right)\cos\left(\frac{2\pi jr}{m}\right) + \cos\left(\frac{2\pi j}{m}\right)\sin\left(\frac{2\pi jr}{m}\right). \tag{18}$$

Thus the representation of $\mathbf{P}$ in each cosine–sine pair is precisely $\mathrm{Rot}(2\pi j/m)$, while the DC component is fixed. Therefore $\mathbf{\Phi}^\top\mathbf{P}\mathbf{\Phi} = \mathcal{U}(1)$ and $\mathbf{P} = \mathbf{P}(1)$. □

Successive shifts compose exactly, and every shift has an inverse. Fractional transport therefore preserves the existing state while extending the integer cyclic shift to arbitrary clock increments.

A.2 TRANSPORT BETWEEN RELATIVE-TIME SLOTS

To see how a shift moves stored content, consider a state with a single nonzero row s. After multiplication by $\mathbf{P}(\tau)$, its contribution to row r is scaled by $\boldsymbol{e}_r^\top\mathbf{P}(\tau)\boldsymbol{e}_s$. To evaluate these transport coefficients, first note from Eqs. (13)–(14) that

$$\mathbf{\Phi}^\top \boldsymbol{e}_r = \mathcal{U}(r)\boldsymbol{b}, \qquad r = 0, \dots, m-1. \tag{19}$$

This identity expresses each integer slot in the real Fourier basis.

Proposition 2 (fractional cyclic-shift kernel). For any $r, s \in \{0, \dots, m-1\}$ and $\tau \in \mathbb{R}$,

$$\boldsymbol{e}_r^\top \mathbf{P}(\tau) \boldsymbol{e}_s = \boldsymbol{b}^\top \mathcal{U}(\tau + s - r) \boldsymbol{b} \tag{20}$$

$$= \frac{1}{m} \left[1 + 2 \sum_{j=1}^{(m-1)/2} \cos\left(\frac{2\pi j(\tau + s - r)}{m} \right) \right] \tag{21}$$

$$= \frac{\sin\big(\pi(\tau + s - r)\big)}{m \sin\big(\pi(\tau + s - r)/m\big)}, \tag{22}$$

where the final ratio is interpreted by continuity when its denominator vanishes.

Proof. Using Eq. (19), orthogonality of $\mathcal{U}$, and its composition law,

$$\boldsymbol{e}_r^\top \mathbf{P}(\tau) \boldsymbol{e}_s = (\boldsymbol{\Phi}^\top \boldsymbol{e}_r)^\top \mathcal{U}(\tau) (\boldsymbol{\Phi}^\top \boldsymbol{e}_s) \tag{23}$$

$$= (\mathcal{U}(r)\boldsymbol{b})^\top \mathcal{U}(\tau)\mathcal{U}(s)\boldsymbol{b} = \boldsymbol{b}^\top \mathcal{U}(\tau + s - r)\boldsymbol{b}. \tag{24}$$

Substituting Eq. (14) gives Eq. (21). Writing the cosine sum as a symmetric geometric series,

$$\frac{1}{m} \sum_{j=-(m-1)/2}^{(m-1)/2} \exp\left(\frac{2\pi \mathrm{i} j x}{m} \right) = \frac{\sin(\pi x)}{m \sin(\pi x / m)}, \tag{25}$$

with $x = \tau + s - r$, gives Eq. (22). □

The kernel therefore gives the coefficient with which content from source row s is transported to destination row r.

Cardinality. For integer a, Eq. (22) is one when $r \equiv s + a \pmod m$ and zero at every other integer slot. Therefore

$$\mathbf{P}(a)\boldsymbol{e}_s = \boldsymbol{e}_{s+a \bmod m}. \tag{26}$$

Integer transport therefore moves the complete row exactly to another slot.

Partition of unity and energy preservation. The DC component is fixed by every $\mathcal{U}(\tau)$, while all non-DC components sum to zero over the integer grid. Hence

$$\mathbf{1}^\top \mathbf{P}(\tau) \boldsymbol{e}_s = 1. \tag{27}$$

Because $\mathbf{P}(\tau)$ is orthogonal,

$$\|\mathbf{P}(\tau)\boldsymbol{e}_s\|_2 = 1, \qquad \big(\mathbf{P}(\tau)\boldsymbol{e}_s\big)^\top \big(\mathbf{P}(\tau)\boldsymbol{e}_{s'}\big) = \delta_{s,s'}. \tag{28}$$

Thus, at every fractional shift, the m translated profiles form a complete orthonormal coordinate system. Fractional transport moves a row continuously between integer slots using cardinal interpolation coefficients. These coefficients are not probabilities and can have signed sidelobes; probabilistic normalization is introduced only by the softmax readout in Eq. (6).

A.3 Evolution of Stored Writes

We now follow each write from its insertion at row zero through subsequent state transitions.

Proposition 3 (unrolled relative state). Starting from zero state, the recurrence in Eq. (4) gives

$$\mathbf{K}_t = \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathbf{P}(\lambda_t - \lambda_i) \boldsymbol{e}_0 \mathbf{k}_i^\top, \tag{29}$$

$$\mathbf{V}_t = \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathbf{P}(\lambda_t - \lambda_i) \boldsymbol{e}_0 \mathbf{v}_i^\top. \tag{30}$$

Consequently, physical row r of the key state is

$$\boldsymbol{e}_r^\top \mathbf{K}_t = \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \big[\mathbf{P}(\lambda_t - \lambda_i) \boldsymbol{e}_0\big]_r \mathbf{k}_i^\top. \tag{31}$$

Proof. The statement is immediate at $t = 0$. Assuming Eq. (29) at $t-1$, Eq. (4) and the group law give

$$\mathbf{K}_t = \alpha_t \mathbf{P}(\delta_t) \sum_{i \le t-1} \beta_i \left(\prod_{j=i+1}^{t-1} \alpha_j \right) \mathbf{P}(\lambda_{t-1} - \lambda_i) \boldsymbol{e}_0 \mathbf{k}_i^\top + \beta_t \boldsymbol{e}_0 \mathbf{k}_t^\top \tag{32}$$

$$= \sum_{i \le t-1} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathbf{P}(\delta_t + \lambda_{t-1} - \lambda_i) \boldsymbol{e}_0 \mathbf{k}_i^\top + \beta_t \boldsymbol{e}_0 \mathbf{k}_t^\top \tag{33}$$

$$= \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathbf{P}(\lambda_t - \lambda_i) \boldsymbol{e}_0 \mathbf{k}_i^\top. \tag{34}$$

The value-state identity is identical. Left-multiplying by $\boldsymbol{e}_r^\top$ yields Eq. (31). □

The factor $\mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0$ is obtained by composing every shift applied after write i entered at row zero. Relative-time profiles therefore arise from the evolution of stored information.

Proposition 4 (coherent cyclic transport). For every previously written token $i \le t$,

$$\mathbf{P}(\lambda_{t+1} - \lambda_i)\boldsymbol{e}_0 = \mathbf{P}(\delta_{t+1})\mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0. \tag{35}$$

Moreover, for any two writes $i, j \le t$, their relative-time-profile inner product is independent of the current readout position:

$$\big(\mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0\big)^\top \big(\mathbf{P}(\lambda_t - \lambda_j)\boldsymbol{e}_0\big) \tag{36}$$

$$= \boldsymbol{e}_0^\top \mathbf{P}(\lambda_i - \lambda_j)\boldsymbol{e}_0 = \frac{\sin\big(\pi(\lambda_i - \lambda_j)\big)}{m \sin\big(\pi(\lambda_i - \lambda_j)/m\big)}. \tag{37}$$

Proof. Equation (35) follows from $\lambda_{t+1} = \lambda_t + \delta_{t+1}$ and the group law in Proposition 1. For the inner product, orthogonality and the same group law give

$$\big(\mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0\big)^\top \big(\mathbf{P}(\lambda_t - \lambda_j)\boldsymbol{e}_0\big) \tag{38}$$

$$= \boldsymbol{e}_0^\top \mathbf{P}(\lambda_i - \lambda_t)\mathbf{P}(\lambda_t - \lambda_j)\boldsymbol{e}_0 = \boldsymbol{e}_0^\top \mathbf{P}(\lambda_i - \lambda_j)\boldsymbol{e}_0. \tag{39}$$

The closed form is Proposition 2 with $r = s = 0$. □

Every clock step therefore transports all existing writes by the same orthogonal cyclic shift. Their relative geometry is preserved, while the complete history moves relative to the fixed age-zero entrance at which the next write is added.

Proposition 5 (order-preserving adaptive spacing). For any $i < j \le t$,

$$\big(\lambda_t - \lambda_i\big) - \big(\lambda_t - \lambda_j\big) = \lambda_j - \lambda_i = \sum_{u=i+1}^{j} \delta_u \in (0, j-i). \tag{40}$$

Hence the learned clock preserves chronological order: at every later readout, write i has a strictly larger model age than write j. At the same time, the separation assigned to the interval $i+1, \dots, j$ is data dependent.

Proof. The first equality cancels the common current clock λ_t. The second follows by telescoping $\lambda_u = \lambda_{u-1} + \delta_u$. Since every $\delta_u \in (0, 1)$, the sum is strictly between 0 and $j - i$. □

In particular,

$$\lambda_t - \lambda_i = \sum_{u=i+1}^{t} \delta_u, \qquad \lambda_t - \lambda_i < m \iff \sum_{u=i+1}^{t} \delta_u < m. \tag{41}$$

The number of token positions traversed before a write completes one cycle is therefore determined by the accumulated clock increments rather than by a fixed token window. Small increments compress

more consecutive tokens into one unit of model age, while larger increments separate them more strongly.

Because the coordinate system is cyclic, model ages that differ by an integer number of cycles have the same relative position and are superposed within the fixed state. The scalar amplitudes in Proposition 3 determine how much evidence survives across repeated cycles.

Corollary 1 (overlap of transported state contributions). Ignoring only the positive scalar amplitudes in Proposition 3, the Frobenius inner product between the key-state contributions of writes i and j is

$$\left\langle \mathbf{P}(\lambda_t - \lambda_i)\boldsymbol{e}_0\boldsymbol{k}_i^\top, \mathbf{P}(\lambda_t - \lambda_j)\boldsymbol{e}_0\boldsymbol{k}_j^\top \right\rangle_{\mathrm{F}} \tag{42}$$

$$= (\boldsymbol{k}_i^\top \boldsymbol{k}_j)\frac{\sin\big(\pi(\lambda_i - \lambda_j)\big)}{m\sin\big(\pi(\lambda_i - \lambda_j)/m\big)}. \tag{43}$$

The value-state contributions satisfy the same identity with $\boldsymbol{k}_i^\top \boldsymbol{k}_j$ replaced by $\boldsymbol{v}_i^\top \boldsymbol{v}_j$.

Proof. For vectors $\boldsymbol{a}, \boldsymbol{c}$ and $\boldsymbol{x}, \boldsymbol{y}$, $\langle \boldsymbol{a}\boldsymbol{x}^\top, \boldsymbol{c}\boldsymbol{y}^\top \rangle_{\mathrm{F}} = (\boldsymbol{a}^\top \boldsymbol{c})(\boldsymbol{x}^\top \boldsymbol{y})$. Applying this identity and Proposition 4 gives Eq. (43). The omitted amplitudes simply multiply the two sides. □

This factorization quantifies the geometry established between writes when they enter the moving state. Subsequent common shifts preserve that geometry during transport.

A.4 Absolute-Clock Coordinates as an Exact Change of Coordinates

The three key-state representations used in the main text are $\mathbf{K}_t$ in relative-time slots, $\widehat{\mathbf{K}}_t$ in Fourier coordinates, and $\overline{\mathbf{K}}_t$ in absolute-clock coordinates, with

$$\widehat{\mathbf{K}}_t = \boldsymbol{\Phi}^\top \mathbf{K}_t, \qquad \overline{\mathbf{K}}_t = \mathcal{U}(-\lambda_t)\widehat{\mathbf{K}}_t, \tag{44}$$

and identical definitions for $\mathbf{V}_t$. These coordinates reproduce the two parts of the physical update separately:

$$\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_{t-1} = \boldsymbol{\Phi}\mathcal{U}(\lambda_t - \lambda_{t-1})\boldsymbol{\Phi}^\top \mathbf{K}_{t-1} = \mathbf{P}(\delta_t)\mathbf{K}_{t-1}, \tag{45}$$

$$\boldsymbol{\Phi}\mathcal{U}(\lambda_t)\mathcal{U}(-\lambda_t)\boldsymbol{b} = \boldsymbol{\Phi}\boldsymbol{b} = \boldsymbol{e}_0. \tag{46}$$

The temporal write vector is the fixed age-zero insertion expressed in absolute-clock coordinates. Reconstructing at the current clock maps the previous stored state to its shifted version and the new write vector to $\boldsymbol{e}_0$. The scalar recurrence in Eq. (8) thus implements exactly the transport-and-insert update in Eq. (4).

The unrolled states show how this equivalence persists over the complete history. Applying $\boldsymbol{\Phi}^\top$ to Eq. (29) gives

$$\widehat{\mathbf{K}}_t = \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathcal{U}(\lambda_t - \lambda_i)\boldsymbol{b}\mathbf{k}_i^\top. \tag{47}$$

Multiplying by $\mathcal{U}(-\lambda_t)$ then yields

$$\overline{\mathbf{K}}_t = \sum_{i \le t} \beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) \mathcal{U}(-\lambda_i)\boldsymbol{b}\mathbf{k}_i^\top, \tag{48}$$

The value state follows identically with $\mathbf{v}_i^\top$ in place of $\mathbf{k}_i^\top$. In absolute-clock coordinates, the direction representing each earlier write remains fixed; its motion in relative-time slots is recovered by the current reconstruction map. Equation (48) also directly implies the scalar-decay recurrence in Eq. (8).

The complete relative-time state is recovered exactly:

$$\mathbf{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_t = \sum_{i\le t}\beta_i\left(\prod_{j=i+1}^{t}\alpha_j\right)\mathbf{\Phi}\mathcal{U}(\lambda_t-\lambda_i)\boldsymbol{b}\mathbf{k}_i^\top \tag{49}$$

$$= \sum_{i\le t}\beta_i\left(\prod_{j=i+1}^{t}\alpha_j\right)\mathbf{P}(\lambda_t-\lambda_i)\boldsymbol{e}_0\mathbf{k}_i^\top = \mathbf{K}_t. \tag{50}$$

The same reconstruction holds for $\mathbf{V}_t$. Substitution into Eq. (6) gives Eq. (10), so both the update and the layer output are unchanged by the coordinate transformation.

Proposition 6 (clock-origin invariance). Let every clock value be shifted by an arbitrary constant $c \in \mathbb{R}$, so that $\lambda'_t = \lambda_t + c$. If the absolute-clock states are formed using the shifted temporal write vectors, then

$$\overline{\mathbf{K}}'_t = \mathcal{U}(-c)\overline{\mathbf{K}}_t, \qquad \overline{\mathbf{V}}'_t = \mathcal{U}(-c)\overline{\mathbf{V}}_t, \tag{51}$$

and the output in Eq. (10) is unchanged.

Proof. Equation (51) follows term by term from $\mathcal{U}(-\lambda_i - c) = \mathcal{U}(-c)\mathcal{U}(-\lambda_i)$. At readout,

$$\mathbf{\Phi}\mathcal{U}(\lambda_t + c)\overline{\mathbf{K}}'_t = \mathbf{\Phi}\mathcal{U}(\lambda_t + c)\mathcal{U}(-c)\overline{\mathbf{K}}_t = \mathbf{\Phi}\mathcal{U}(\lambda_t)\overline{\mathbf{K}}_t, \tag{52}$$

and the same identity holds for the value state. Substitution into Eq. (6), or equivalently Eq. (10), leaves $\mathbf{o}_t$ unchanged. □

The clock origin is therefore arbitrary. Although Eq. (8) uses clock values computationally, the represented memory and the layer output depend only on model ages $\lambda_t - \lambda_i$.

A.5 Temporal Addresses and the Two-Pass Readout

The temporal write vectors arise from expressing fixed age-zero insertion in absolute-clock coordinates. They also admit an address-matching interpretation of the same readout: $\mathcal{U}(-\lambda)\boldsymbol{b}$ can be viewed as a continuous family of temporal addresses, while $\mathbf{k}_i$ and $\mathbf{v}_i$ remain the key and value content.

The similarity between two such addresses is

$$\big(\mathcal{U}(-\lambda_i)\boldsymbol{b}\big)^\top\big(\mathcal{U}(-\lambda_j)\boldsymbol{b}\big) = \boldsymbol{b}^\top\mathcal{U}(\lambda_i-\lambda_j)\boldsymbol{b} \tag{53}$$

$$= \frac{\sin\big(\pi(\lambda_i-\lambda_j)\big)}{m\sin\big(\pi(\lambda_i-\lambda_j)/m\big)}. \tag{54}$$

Addresses whose clock values differ by a nonzero integer within a cycle are orthogonal, while fractional separations produce the corresponding cardinal interpolation similarity.

The first pass in Eq. (11) gives

$$\mathbf{o}'_t = \overline{\mathbf{K}}_t\mathbf{q}_t = \sum_{i\le t}\beta_i\left(\prod_{j=i+1}^{t}\alpha_j\right)(\mathbf{k}_i^\top\mathbf{q}_t)\mathcal{U}(-\lambda_i)\boldsymbol{b}. \tag{55}$$

The key–query match $\mathbf{k}_i^\top\mathbf{q}_t$ therefore scales the temporal address of write i. The first pass does not yet choose a value; it accumulates key evidence in the temporal-address space.

The r-th canonical relative-time slot at token t corresponds, in absolute-clock coordinates, to the query address

$$\boldsymbol{e}_r^\top\mathbf{\Phi}\mathcal{U}(\lambda_t) = \big(\mathcal{U}(r-\lambda_t)\boldsymbol{b}\big)^\top. \tag{56}$$

Its match to the address of write i is

$$\big(\mathcal{U}(r-\lambda_t)\boldsymbol{b}\big)^\top\mathcal{U}(-\lambda_i)\boldsymbol{b} = \boldsymbol{b}^\top\mathcal{U}(\lambda_t-\lambda_i-r)\boldsymbol{b} \tag{57}$$

$$= \big[\mathbf{P}(\lambda_t-\lambda_i)\boldsymbol{e}_0\big]_r. \tag{58}$$

Combining Eqs. (55) and (58), the contribution of write i to relative-time slot r before the learned readout is

$$\beta_i \left(\prod_{j=i+1}^{t} \alpha_j \right) (\mathbf{k}_i^\top \mathbf{q}_t) \big[\mathbf{P}(\lambda_t - \lambda_i) \boldsymbol{e}_0 \big]_r. \tag{59}$$

The address inner product in Eq. (58) is exactly the transport coefficient derived in Proposition 2. The address view therefore recovers the same relative-time readout from the absolute-clock representation rather than introducing a separate state evolution.

The learned readout matrix $\mathbf{R}$ replaces the canonical address queries by task-adaptive linear combinations. Row s of the learned readout satisfies

$$\boldsymbol{e}_s^\top \mathbf{R} \boldsymbol{\Phi} \mathcal{U}(\lambda_t) = \sum_{r=0}^{m-1} \mathbf{R}_{s,r} \big(\mathcal{U}(r - \lambda_t) \boldsymbol{b} \big)^\top. \tag{60}$$

Thus $\mathbf{R}$ learns a bank of temporal-address queries anchored to model age. Because the current clock value is applied before $\mathbf{R}$, the same row of $\mathbf{R}$ has the same relative-time meaning for every clock value.

The transpose readout map in the middle line of Eq. (11) has the matching address interpretation. For any slot-weight vector $\mathbf{w} \in \mathbb{R}^m$,

$$\mathcal{U}(-\lambda_t) \boldsymbol{\Phi}^\top \mathbf{R}^\top \mathbf{w} = \sum_{r=0}^{m-1} [\mathbf{R}^\top \mathbf{w}]_r \mathcal{U}(r - \lambda_t) \boldsymbol{b}. \tag{61}$$

After setting $\mathbf{w}$ to the softmax weights in Eq. (11), the second pass queries the value state with this mixture of current relative-time addresses. Since $\overline{\mathbf{V}}_t$ stores every $\mathbf{v}_i$ under the same temporal address $\mathcal{U}(-\lambda_i)\boldsymbol{b}$, the address inner products in Eq. (58) recover exactly the desired relative-time value aggregation.

A.6 Fixed-Rate Specialization and Relation to BLA

If the learned clock is constrained to a constant increment $\delta_u \equiv 1/\rho$, then $\lambda_t - \lambda_i = (t-i)/\rho$. Writing $T = \rho m$, Proposition 2 gives

$$\big[\mathbf{P}((t-i)/\rho)\, \boldsymbol{e}_0 \big]_r = \frac{1}{m} \left[1 + 2 \sum_{h=1}^{(m-1)/2} \cos\left(\frac{2\pi h (t - i - \rho r)}{T} \right) \right]. \tag{62}$$

This is the fixed-resolution Fourier–Dirichlet profile used by Blurry Window Attention (BLA; Laborieux et al., 2026). Under this constraint, every ρ token steps advance one relative-time slot and one cycle spans T token steps. CyFA retains the same cardinal frame without imposing an affine relation between token distance and model age: the cumulative increments in Eq. (41) determine the transport distance.

This specialization matches BLA's temporal profile, but the complete state updates and readouts differ. With state decay, BLA uses the profile-coupled GSA recurrence

$$\mathbf{K}_t = \mathrm{Diag}(\mathbf{1} - \boldsymbol{\phi}_t) \mathbf{K}_{t-1} + \boldsymbol{\phi}_t \mathbf{k}_t^\top, \tag{63}$$

with an analogous update for $\mathbf{V}_t$. Its position within the cycle is affine in the token index with period T. In its efficient form, BLA keeps the cumulative state in absolute coordinates and uses joint row-permutation invariance of key–value softmax. CyFA uses a data-dependent monotone clock and a separate scalar forget gate. Before learned slot mixing and softmax, it reconstructs the frame relative to the current token and applies $\mathbf{R}$. Thus Eq. (62) establishes the profile correspondence, not equality of the complete layers.

A.7 Active Coordinates and Padded Storage

All derivations above operate on the $m = 127$ active relative-time slots. The implementation allocates $m^* = 128$ physical rows for hardware convenience. The additional row is outside the Fourier frame, is masked during writing and readout, and remains zero. Equivalently, the implemented operators act as the derived m-dimensional maps on the active invariant subspace and as zero on the padded row. The padding therefore changes neither the recurrence nor any of the coordinate identities above.

B HARDWARE-EFFICIENT CHUNK-WISE PARALLELISM

Interface states. Across chunks, CyFA stores the absolute-clock states $\overline{\mathbf{K}}$ and $\overline{\mathbf{V}}$. The state of the first ScalarGatedLA call is

$$\mathbf{S}_t^{(K)} = \alpha_t \mathbf{S}_{t-1}^{(K)} + \mathbf{k}_t \big(\beta_t \mathcal{U}(-\lambda_t)\boldsymbol{b}\big)^\top = \overline{\mathbf{K}}_t^\top, \tag{64}$$

so its output is $(\mathbf{S}_t^{(K)})^\top \mathbf{q}_t = \overline{\mathbf{K}}_t \mathbf{q}_t = \mathbf{o}'_t$. The state of the second call is

$$\mathbf{S}_t^{(V)} = \alpha_t \mathbf{S}_{t-1}^{(V)} + \beta_t \mathcal{U}(-\lambda_t)\boldsymbol{b}\mathbf{v}_t^\top = \overline{\mathbf{V}}_t, \tag{65}$$

and its output is $\overline{\mathbf{V}}_t^\top \mathbf{o}''_t$, exactly as in Eq. (10).

Scalar-decay chunk-wise form. We follow the chunk notation of Section 2.1: $\mathbf{S}_{[i]} = \mathbf{S}_{iC}$ is the state after i chunks, and $\mathbf{Q}_{[i+1]}, \mathbf{K}_{[i+1]}, \mathbf{V}_{[i+1]}$ contain the next C tokens. Consider the ScalarGatedLA recurrence in Table 1, $\mathbf{S}_t = \alpha_t \mathbf{S}_{t-1} + \boldsymbol{k}_t \boldsymbol{v}_t^\top$. For the $(i+1)$-th chunk, define

$$\gamma_{[i+1]}^r = \prod_{j=1}^{r} \alpha_{iC+j}, \qquad \big(\mathbf{\Gamma}_{[i+1]}\big)_{rs} = \begin{cases} \gamma_{[i+1]}^r / \gamma_{[i+1]}^s, & r \geq s, \\ 0, & r < s, \end{cases} \tag{66}$$

for $r, s = 1, \ldots, C$. Thus $\mathbf{\Gamma}_{[i+1]}$ is the decay-aware counterpart of the causal mask $\mathbf{M}$ in Section 2.1. For any block $\mathbf{X}_{[i+1]} \in \mathbb{R}^{C \times d}$, let

$$\begin{aligned} \big(\overleftarrow{\mathbf{X}}_{[i+1]}\big)_{r:} &= \gamma_{[i+1]}^r \big(\mathbf{X}_{[i+1]}\big)_{r:}, \\ \big(\overrightarrow{\mathbf{X}}_{[i+1]}\big)_{r:} &= \frac{\gamma_{[i+1]}^C}{\gamma_{[i+1]}^r} \big(\mathbf{X}_{[i+1]}\big)_{r:}. \end{aligned} \tag{67}$$

The scalar recurrence then admits the exact chunk-wise form

$$\begin{aligned} \mathbf{S}_{[i+1]} &= \gamma_{[i+1]}^C \mathbf{S}_{[i]} + \mathbf{K}_{[i+1]}^\top \overrightarrow{\mathbf{V}}_{[i+1]}, \\ \mathbf{O}_{[i+1]} &= \overleftarrow{\mathbf{Q}}_{[i+1]} \mathbf{S}_{[i]} + \Big(\mathbf{Q}_{[i+1]} \mathbf{K}_{[i+1]}^\top \odot \mathbf{\Gamma}_{[i+1]}\Big) \mathbf{V}_{[i+1]}. \end{aligned} \tag{68}$$

The first term in the output accounts for preceding chunks, while the second computes all causal interactions inside the current chunk in parallel. In practice, the decay factors are obtained from chunk-local cumulative sums of $\log \alpha_t$.

CyFA chunk-wise form. For the absolute-clock recurrence in Eq. (8), stack the temporal write vectors of chunk $i+1$ into the temporal-write matrix $\mathbf{A}_{[i+1]} \in \mathbb{R}^{C \times m}$, whose r-th row is

$$\big(\mathbf{A}_{[i+1]}\big)_{r:} = \big(\beta_{iC+r} \mathcal{U}(-\lambda_{iC+r})\boldsymbol{b}\big)^\top. \tag{69}$$

We extend the bracket notation to the absolute-clock states, $\overline{\mathbf{K}}_{[i]} = \overline{\mathbf{K}}_{iC}$ and $\overline{\mathbf{V}}_{[i]} = \overline{\mathbf{V}}_{iC}$. Applying Eq. (68) to the first ScalarGatedLA call in Eq. (11) gives

$$\begin{aligned} \overline{\mathbf{K}}_{[i+1]} &= \gamma_{[i+1]}^C \overline{\mathbf{K}}_{[i]} + \overrightarrow{\mathbf{A}}_{[i+1]}^\top \mathbf{K}_{[i+1]}, \\ \mathbf{O}'_{[i+1]} &= \overleftarrow{\mathbf{Q}}_{[i+1]} \overline{\mathbf{K}}_{[i]}^\top + \Big(\mathbf{Q}_{[i+1]} \mathbf{K}_{[i+1]}^\top \odot \mathbf{\Gamma}_{[i+1]}\Big) \mathbf{A}_{[i+1]}. \end{aligned} \tag{70}$$

The middle line of Eq. (11) is then applied independently to each row of $\mathbf{O}'_{[i+1]}$, producing $\mathbf{O}''_{[i+1]}$. The second ScalarGatedLA call similarly becomes

$$\begin{aligned} \overline{\mathbf{V}}_{[i+1]} &= \gamma_{[i+1]}^C \overline{\mathbf{V}}_{[i]} + \mathbf{A}_{[i+1]}^\top \overrightarrow{\mathbf{V}}_{[i+1]}, \\ \mathbf{O}_{[i+1]} &= \overleftarrow{\mathbf{O}''}_{[i+1]} \overline{\mathbf{V}}_{[i]} + \Big(\mathbf{O}''_{[i+1]} \mathbf{A}_{[i+1]}^\top \odot \mathbf{\Gamma}_{[i+1]}\Big) \mathbf{V}_{[i+1]}. \end{aligned} \tag{71}$$

Both passes use the same scalar forget gate α_t, and therefore share the factors $\{\gamma_{[i+1]}^r\}_{r=1}^C$ and the mask $\mathbf{\Gamma}_{[i+1]}$. The scalar carry across a chunk represents the complete physical transport over that chunk. At the current chunk boundary,

$$\mathbf{\Phi}\mathcal{U}(\lambda_{(i+1)C}) \Big(\gamma_{[i+1]}^C \overline{\mathbf{K}}_{[i]}\Big) = \gamma_{[i+1]}^C \mathbf{P}(\lambda_{(i+1)C} - \lambda_{iC}) \mathbf{K}_{iC}. \tag{72}$$

The new-write terms recover their corresponding cumulative shifts through the same coordinate identity. In relative-time slots, the carried state has therefore advanced by the clock distance accumulated across the chunk. The scalar boundary recurrence accounts for the same transport without explicitly applying the corresponding shift matrix.

Implementation. We use $C = 64$ and reuse the scalar-decay chunk primitives from `flash-linear-attention` for both passes (Yang & Zhang, 2024). The first pass produces tiled partial outputs that are reduced inside the token-wise slot-readout kernel, so the full $\mathbf{O}'$ need not be materialized. The clock $\{\lambda_t\}_{t=1}^T$ is computed by a prefix sum of $\{\delta_t\}_{t=1}^T$; its backward pass is the corresponding reverse prefix sum. Within each pass, the only recurrent dependency over the sequence is the propagation of the chunk-boundary state.

C Experimental Details

C.1 Architecture and State Matching

Backbone and mixer configurations. All models pretrained for our comparison use an $L = 24$ pre-norm LLaMA-style backbone, with each layer comprising a sequence mixer followed by a SwiGLU feed-forward block.

To isolate the recurrent update, we instantiate BLA with the same surrounding mixer design as CyFA, retaining ShortConv, RMSNorm-based QK normalization, head-wise RMSNorm, and the low-rank sigmoid output gate. We replace only the core recurrence with the decayed BLA variant described by Laborieux et al. (2026). At the time of our experiments, no official BLA code was publicly available, so we implement this recurrence in its GSA-equivalent form: the Dirichlet interpolation profile $\boldsymbol{\phi}_t$ provides the slot-wise write weights, while $\mathbf{1} - \boldsymbol{\phi}_t$ decays the previous state. This parameterization allows us to use the Triton GSA chunk operator from `flash-linear-attention` (Zhang et al., 2024; Yang & Zhang, 2024).

Mamba-2 and Mamba-3 each use one Mamba mixer per backbone layer (Dao & Gu, 2024; Lahoti et al., 2026), with an expansion factor of 2, $d_{\text{state}} = 128$, and $d_{\text{head}} = 64$. We use the rank-4 MIMO variant of Mamba-3.

State-size accounting. In our state-size calculation, we count the scalar entries in the main recurrent state of each layer and omit the ShortConv cache.

SWA, GSA, and Raven store key and value tensors with m slots per head, giving state shapes $H \times m \times d_k$ and $H \times m \times d_v$. At 400M, we use $H = 4$, $m = 128$, and $d_k = d_v = 256$, for a total of

$$Hm(d_k + d_v) = 4 \times 128 \times (256 + 256) = 262{,}144$$

scalar entries per layer.

For hardware-efficient computation, we implement CyFA and BLA with $m = 127$ active slots stored in $m^* = m + 1 = 128$ physical rows. At $H = 4$ and $d_k = d_v = 256$, each has state size

$$Hm^*(d_k + d_v) = 4 \times 128 \times (256 + 256) = 262{,}144.$$

GDN and KDA maintain one $d_v \times d_k$ state matrix per head. With $H = 4$ and $d_k = d_v = 256$, their state size is

$$Hd_kd_v = 4 \times 256 \times 256 = 262{,}144.$$

Mamba-2 and Mamba-3 maintain a recurrent state of shape $H \times d_{\text{head}} \times d_{\text{state}}$. At $d = 1024$, the expansion factor of 2 and $d_{\text{head}} = 64$ give $H = 2d/d_{\text{head}} = 32$ heads. Together with $d_{\text{state}} = 128$, this yields

$$Hd_{\text{head}}d_{\text{state}} = 32 \times 64 \times 128 = 262{,}144.$$

All recurrent architectures therefore have the same per-layer main recurrent state size at 400M. At larger scales, we preserve each method's per-head state shape and increase only its number of heads, maintaining the match across methods.

Table 6 summarizes these model configurations. Public checkpoints use their released architectures, tokenizers, and context configurations and are reported as separate groups in the result tables.

Table 6: Configurations of the language models pretrained for our comparison. Public checkpoints are marked with †.

Method	Params (M)	State Size	Configuration
400M parameters with $L = 24$ and $d = 1024$			
Transformer	374	–	$H = 16,\ d_k = d_v = 64$
SWA	374	262,144	$H = 4,\ d_k = d_v = 256,\ m = 128$
GSA	386	262,144	$H = 4,\ d_k = d_v = 256,\ m = 128$
Raven	412	262,144	$H = 4,\ d_k = d_v = 256,\ m = 128$
BLA	387	262,144	$H = 4,\ d_k = d_v = 256,\ m^* = 128$
Mamba-2	375	262,144	$H = 32,\ d_{\text{head}} = 64,\ d_{\text{state}} = 128$
Mamba-3	378	262,144	$H = 32,\ d_{\text{head}} = 64,\ d_{\text{state}} = 128$
GDN	400	262,144	$H = 4,\ d_k = d_v = 256$
KDA	400	262,144	$H = 4,\ d_k = d_v = 256$
CyFA	389	262,144	$H = 4,\ d_k = d_v = 256,\ m^* = 128$
800M parameters with $L = 24$ and $d = 1536$			
Transformer	778	–	$H = 24, d_k = d_v = 64$
GDN	835	393,216	$H = 6,\ d_k = d_v = 256$
KDA	816	393,216	$H = 6,\ d_k = d_v = 256$
CyFA	800	393,216	$H = 6,\ d_k = d_v = 256,\ m^* = 128$
1.4B parameters with $L = 24$ and $d = 2048$			
Transformer†	1,364	–	$H = 32,\ d_k = d_v = 64$
RetNet†	1,352	1,048,576	$H = 8,\ d_k = 256,\ d_v = 512$
GLA†	1,366	524,288	$H = 4,\ d_k = 256,\ d_v = 512$
GSA†	1,377	262,144	$H = 4,\ d_k = d_v = 512,\ m = 64$
KDA	1,416	524,288	$H = 8,\ d_k = d_v = 256$
CyFA	1,394	524,288	$H = 8,\ d_k = d_v = 256,\ m^* = 128$

Table 7: Pretraining configuration by model scale. Per-device batch size denotes the number of sequences processed before gradient accumulation.

Scale	Training Tokens	Context	Global Batch (Tokens)	Per-Device Batch	Gradient Accumulation	Wall Time
400M	15B	2,048	0.5M	32	1	∼10 hours
800M	30B	2,048	0.5M	16	2	∼1 day
1.4B	100B	2,048	1M	16	4	∼1 week

C.2 Pretraining Configuration

All model implementations use the `flash-linear-attention` library (Yang & Zhang, 2024). The pretraining dataset is SlimPajama-627B. All models use the Mistral tokenizer. Every run uses a context length of 2,048 tokens.

We optimize with AdamW using a peak learning rate of 3×10^{-4} and a weight decay of 0.01. The schedule consists of 1,000 warmup steps followed by cosine decay to 3×10^{-5}. We clip the gradient norm at 1.0.

All runs use eight NVIDIA RTX Pro 6000 GPUs. Table 7 reports the training-token budget, effective batch size, gradient accumulation, and wall-clock time for each model scale.

Long-context continuation. For the 800M and 1.4B evaluations in Section 4.2, we resume the pretrained checkpoints and optimizer states for 1,024 steps at a context length of 8,192. To preserve the initial number of tokens per optimizer step, we use a per-device batch size of 4 with two gradient-accumulation steps at 800M (0.5M tokens) and four at 1.4B (1M tokens). The learning rate remains fixed at 3×10^{-5}, the terminal value of the preceding cosine schedule. All other settings match initial pretraining.

D Additional Analysis

D.1 Gradient Propagation Through the Learned Clock

We analyze the instability associated with removing sg from Eq. (12) using paired backward probes on four 2,048-token training sequences from the 400M CyFA checkpoint. A forward-equivalent intervention restores only the derivative from δ_t to $\boldsymbol{x}_t$, leaving the loss and all forward activations unchanged. KDA serves as a reference with data-dependent recurrent control but no shared cumulative clock.

Averaged over the four sequences, CyFA with sg has a global gradient norm of 5.653, compared with 2.309 for KDA. This difference is highly localized: the clock projection has gradient norm 5.052 and accounts for 78.3% of the squared gradient norm. Excluding this projection leaves a norm of 2.447, only $1.06\times$ that of KDA. The elevated global norm therefore does not arise from uniformly larger gradients throughout CyFA.

The concentration follows from the cumulative clock. Because $\lambda_t = \sum_{s\le t} \delta_s$, its backward recurrence is $\partial\mathcal{L}/\partial\delta_s = \sum_{t\ge s} \partial\mathcal{L}/\partial\lambda_t$. Each clock increment therefore receives credit from all subsequent positions through the shared address used by the key write, value write, and slot readout. In the layers with the largest clock gradients, this reverse cumulative sum amplifies the local λ-gradient norm by 3.4–$9.6\times$, while coherent accumulation of the resulting per-token parameter gradients contributes another 7.4–$11.8\times$. The three address paths are individually large and frequently oppose one another, leaving a cancellation-sensitive residual in the shared low-dimensional clock projection.

Removing sg opens a second amplification path by injecting this future-accumulated gradient into the residual stream. The global norm rises to 12.006 and remains 8.682 after excluding the clock projection, or $3.76\times$ the KDA norm, with the additional gradient spreading into the value write, output, MLP, and earlier layers. The effect also grows with sequence length: without sg, the non-clock gradient relative to KDA increases from $1.14\times$ at 256 tokens to $4.58\times$ at 2,048 tokens, whereas it remains within 1.05–$1.08\times$ with sg. Thus, stop-gradient does not remove the accumulated clock gradient. It confines that gradient to the clock projection, which remains trainable, and prevents its length-dependent propagation through the backbone.

D.2 Input Dependence of the Learned Clock

We probe the final 400M CyFA checkpoint using the configuration in Table 6: 24 CyFA layers with four heads per layer, $d = 1024$, $d_k = d_v = 256$, $m = 127$ active slots, and $m^* = 128$ physical rows. We collect δ_t from 64 consecutive 2,048-token WikiText-2 sequences, totaling 131,072 tokens and 12,582,912 head–token observations. The following 16 sequences, containing 32,768 tokens, are reserved for held-out interventions.

We classify the post-sigmoid clock increments using their empirical quantiles. A head is near one when $p_{05}(\delta_t) > 0.95$ and input dependent when $p_{95}(\delta_t) - p_{05}(\delta_t) \ge 0.1$. All 96 heads fall into one of these two regimes: 56 are concentrated near one, with mean 0.9961 and standard deviation 0.0065, while 40 are input dependent, with mean 0.5118 and standard deviation 0.1627. Excluding the first layer, where the clock sees only the token embedding, current-token identity explains 49.4% of the variation in the input-dependent heads and context among occurrences of the same token explains the remaining 50.6%. This split is consistent with the same-token cases in Table 5.

We then intervene on the 40 input-dependent heads while leaving the remaining model unchanged. The baseline held-out NLL is 2.6105. Replacing each head's increments by its mean raises NLL by 0.0450, showing that their variation is used by the model. Cyclically shifting the original increments by 257 tokens preserves their values and marginal distribution but breaks their alignment with the corresponding inputs, increasing NLL by 0.1238. Setting $\delta_t = 1$ throughout produces the largest increase, 0.1381. Each intervention degrades all 16 held-out sequences. The stronger effect of misalignment shows that CyFA relies on the correspondence between semantic context and clock increment, rather than merely on a nonconstant clock.

Table 8: Held-out clock interventions for the 400M CyFA model.

Clock Configuration	NLL	ΔNLL
Original δ_t	2.6105	–
Per-head mean	2.6555	+0.0450
Shifted by 257 tokens	2.7344	+0.1238
$\delta_t = 1$	2.7486	+0.1381

E Additional Experimental Results

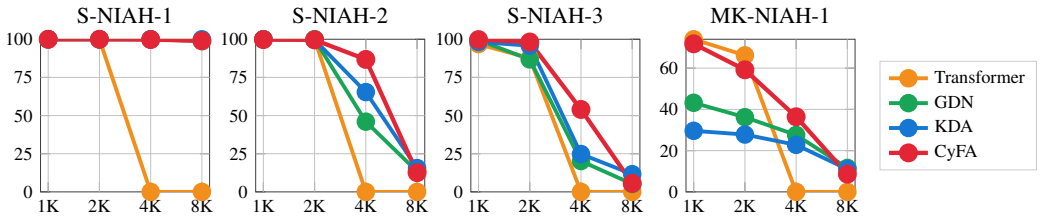

Figure 6: RULER accuracy before continuation.

Table 9: LongBench results before continuation with inputs truncated to 8K tokens.

Scale	Model	Code		Summarization			SingleQA			MultiQA			Few-Shot			Avg.
		LCC	RBP	GvR	QMS	MNs	NQA	QQA	MQA	HQA	2WM	MSQ	TRE	TQA	SAM	
800M	Transformer	22.08	6.40	0.74	0.74	0.67	0.00	0.73	1.67	0.32	0.21	0.00	1.50	0.78	0.15	2.57
	GDN	43.96	37.53	**5.82**	10.64	2.26	**4.34**	1.81	**13.07**	**4.88**	**10.03**	2.12	27.00	25.29	**9.11**	14.13
	KDA	42.16	39.23	3.87	11.78	2.09	3.43	5.79	12.51	4.08	7.69	2.62	**40.50**	25.65	5.82	14.80
	CyFA	**47.31**	**43.92**	3.15	**15.99**	**5.64**	2.71	**5.81**	13.06	4.76	9.25	**2.94**	23.00	**38.40**	4.59	**15.75**
1.4B	KDA	46.12	42.70	4.44	**14.05**	8.68	2.13	0.17	7.72	6.27	7.88	2.90	**53.50**	41.11	**28.97**	19.05
	CyFA	**52.65**	**45.07**	**12.58**	13.91	**13.52**	**2.79**	**5.26**	**14.11**	**7.54**	**12.46**	**3.44**	37.50	**48.67**	17.13	**20.47**