Design Principles for Sequence Models via Coefficient Dynamics

Anonymous authors
Paper under double-blind review

Abstract

Deep sequence models, ranging from Transformers and State Space Models (SSMs) to more recent approaches such as gated linear RNNs, fundamentally compute outputs as linear combinations of past value vectors. To draw insights and systematically compare such architectures, we develop a unified framework that makes this output operation explicit, by casting the linear combination coefficients as the outputs of autonomous linear dynamical systems driven by *impulse inputs*. This viewpoint, in spirit substantially different from approaches focusing on connecting linear RNNs with linear attention, reveals a common mathematical theme across diverse architectures and crucially captures softmax attention, on top of RNNs, SSMs, and related models. In contrast to new model proposals that are commonly evaluated on benchmarks, we derive design principles linking architectural choices to model properties. Thereby identifying tradeoffs between expressivity and efficient implementation, geometric constraints on input selectivity, and stability conditions for numerically stable training and information retention. By connecting several insights and observations from recent literature, the framework both explains empirical successes of recent designs and provides guiding principles for systematically designing new sequence model architectures.

1 Introduction

Sequence modeling lies at the core of modern machine learning, powering advances in natural language processing, computer vision, speech recognition, and robotics. At the heart of this field is a deceptively simple question (Elman, 1990): how should a model combine past information to produce meaningful representations and make predictions about the future? Classical approaches such as recurrent neural networks (RNNs; Hochreiter & Schmidhuber (1997)) framed this as a recursive state-update problem, whereas the Transformer (Vaswani et al., 2017) introduced attention mechanisms that compute adaptive linear combinations over past tokens. More recently, state space models (SSMs) (Gu et al., 2022a; Gu & Dao, 2023) or equivalently linear attention models (Katharopoulos et al., 2020; Dao & Gu, 2024; Schlag et al., 2021) have further broadened the design space by importing ideas from dynamical systems, delivering foundation models with linear sequence length complexity. Despite their differences in formulation, computational complexity, and performance on crucial benchmark tasks (Arora et al., 2023; Jelassi et al., 2024), both linear and softmax attention models share a common thread: their core sequence mixing mechanism computes a linear combination of value vectors with coefficients determined by the interaction of queries, keys, and additional learned parameters. For attention, these coefficients are derived through similarity functions and normalizations; for SSMs and (gated) linear attention, they emerge from the evolution of hidden states under linear dynamics. Yet, while this unifying perspective is informally recognized, most architectural innovations are still introduced and validated primarily through benchmarks. This benchmark-driven approach has fueled rapid empirical progress, but complicates the disentanglement of which design elements – e.g. normalization, gating, or state-update dynamics – truly account for observed improvements. What is lacking is a set of general design principles that explain why specific models succeed and guide the systematic development of new ones, beyond approaches focusing on *ad-hoc* analyses of single-task setups (Jelassi et al., 2024; Merrill et al., 2024; Arora et al., 2024).

In this work, we develop a principled framework for sequence model design grounded in dynamical systems theory. Specifically, we show that the coefficients governing linear combinations of value vectors, can be expressed as the output of autonomous linear dynamical systems subject to *impulse* inputs. Our approach reveals the common mathematical structure underlying diverse architectures and provides a principled foundation for designing new sequence models. Since the present study is restricted to single-layer models, multi-layer composition can produce richer behaviors – such as formation of induction heads (Olsson et al., 2022) – beyond what our base framework captures. Nevertheless, by laying out the core design principles at the layer level, we aim to establish the foundations for understanding and designing more complex systems. Our *contributions* are as follows:

- **Design principles** for sequence models that stem from a unifying theory on the dynamic computations in sequence models, clarifying the role of sequence model components.
- **Analysis of tradeoffs and constraints** concerning expressivity, efficient implementation, input selectivity, and stability requirements in sequence models.
- **A unifying formalization of sequence models** that captures most recent architectures as special cases (Table 1), enabling systematic comparison without reliance on experimental benchmarks.
- **Empirical validation** demonstrates that our theory translates into practice.

2 Related Work

We have recently witnessed a revived interest in efficient RNNs and linear attention mechanisms in language modeling as well as image/video processing (Liu et al., 2024) and DNA modeling (Nguyen et al., 2023; Schiff et al., 2024). Stemming from the seminal S4 work by Gu et al. (2020; 2022a), early SSMs (Gu et al., 2022b; Smith et al., 2023) and linear recurrent networks (Orvieto et al., 2023), outperformed attention on pattern-recognition and long-range reasoning benchmarks (Tay et al., 2021) and were rapidly adapted to the language domain (Wang et al., 2022; Fu et al., 2023). RetNet (Sun et al., 2023) and GateLoop (Katsch, 2023) paved the way for Mamba (Gu & Dao, 2023), which combined input selectivity with efficient training (linear in sequence length) to achieve improved performance in language modeling. Drawing inspiration from the known connection between RNNs and linear attention (Katharopoulos et al., 2020; Schlag et al., 2021), advancements such as GLA, Deltanet, xLSTM (Yang et al., 2023; 2024; Beck et al., 2024), and Mamba2 (Dao & Gu, 2024) have aided the unification of modern efficient sequence modeling approaches in terms of latent efficient matrix multiplications (Yang & Zhang, 2024). On top of this unified framework – *which crucially does not contain softmax attention* – several improvements were recently proposed, such as Gated Deltanet (Yang et al., 2025a), DeltaProduct (Siems et al., 2025), and Log-Linear Attention (Guo et al., 2025). We have also witnessed cross-fertilization between new linear RNN/attention ideas (forget gate, positional encodings) and the softmax world, with examples such as FoX (Lin et al., 2025) and Path (Yang et al., 2025b).
As flagship models are starting to adopt hybrid solutions based on mixing linear and softmax attention to enhance training and inference efficiency (AI21 Labs, 2024; Nano, 2025; Qwen Team, 2025), additional work is needed to further understand which architectural properties impact performance depending, on the specific nature (e.g. recall, associativity, memorization, compression) of the task at hand (Poli et al., 2024). While early studies show promise across benchmarks, especially for hybrid models at high pretraining budgets (Waleffe et al., 2024), others point to fundamental limitations of fixed-memory processing in recall, copy (Arora et al., 2023; Jelassi et al., 2024) and retrieval tasks (Guo et al., 2025). Expressivity of new linear RNN layers is well-studied (Orvieto et al., 2024; Cirone et al., 2024; Merrill et al., 2024), yet less is known about how architectural details impact model capabilities – especially as linear RNNs can be substantially more challenging to optimize (Zucchet & Orvieto, 2024; Okpekpe & Orvieto, 2025), – and we lack a unified framework able to capture both mechanisms under the same formalism, without resorting to infinite-dimensional expansions (Sieber et al., 2024). Our *coefficient dynamics* are inspired by this issue.

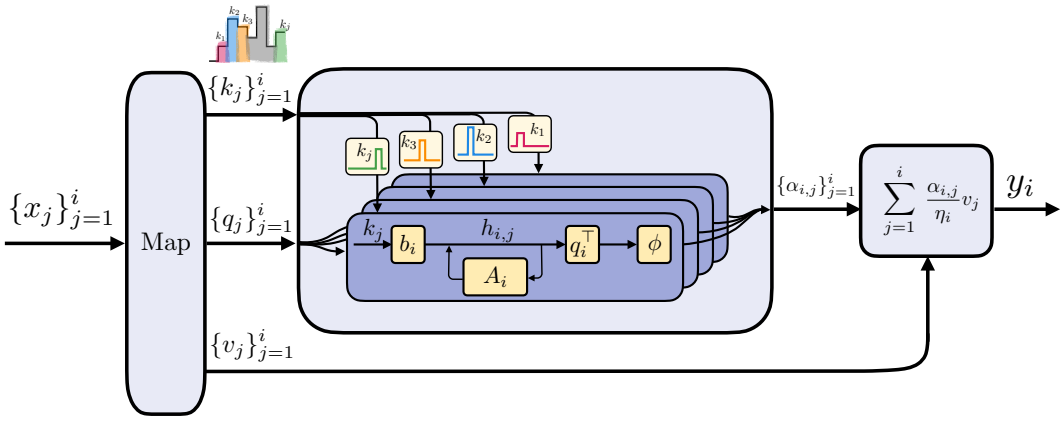


Figure 1: Graphical representation of dynamical system (3).

3 Linear Combinations with Temporal Dynamics

We consider a one-layer causal sequence model (e.g. attention, S6), i.e. a parametric mechanism that defines $\{x_j\}_{j=1}^i \mapsto y_i$, where i denotes the (time) position index $i = 1, \ldots, L$ and $x_i \in \mathbb{R}^d, y_i \in \mathbb{R}^{d_v}$ are the input and output respectively. In the following, we provide a formulation that encompasses most existing sequence model architectures using what we name *coefficient dynamics*. Following the standard transformer notation (Vaswani et al., 2017), and given the duality between attention, SSMs, and other recurrent architectures (Dao & Gu, 2024; Sieber et al., 2024), we define:

$$q_i = W_Q x_i, \quad k_i = W_K x_i, \quad v_i = W_V x_i, \tag{1}$$

where $W_Q \in \mathbb{R}^{n \times d}$, $W_K \in \mathbb{R}^{n \times d}$, and $W_V \in \mathbb{R}^{d_v \times d}$ are the learned parameters of the model.

Let $\{v_j\}_{j=1}^i$ denote the value vectors at (time) position i, then most existing sequence model can be fundamentally viewed as computing linear combinations of these value vectors:[1]

$$y_i = \sum_{j=1}^{i} \frac{\alpha_{i,j}}{\eta_i} v_j, \tag{2}$$

where each coefficient $\alpha_{i,j}$ represents the contribution of the value vector at position j to the output at position i and $\eta_i : \mathbb{R} \to \mathbb{R}$ denotes the normalization factor. In standard architectural proposals, such as softmax attention or linear attention, coefficients $\alpha_{i,j}$ are typically computed using queries, keys, and additional learned parameters. For consistency with existing literature, we refer to $\alpha_{i,j}$ as the *attention coefficients*.

In our framework, we interpret the coefficients $\alpha_{i,j}$ as the output measurement of a latent linear dynamical system with impulse input, which we call the *coefficient dynamics*. In contrast to previous works, which study how tokens are mixed as the iteration counts progress (Ali et al., 2024), we consider the tokens individually. This orthogonal perspective allows to precisely model both softmax and linear attention, with no need for infinite dimensional approximations. Specifically, for each key position j, we define an autonomous dynamical system with initial state $h_{i-1,j} = 0, \forall i \leq j$, i.e.,

$$h_{i,j} = A_i h_{i-1,j} + b_i u_i, \tag{3a}$$

$$u_i = \begin{cases} k_j, & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases} \tag{3b}$$

$$\alpha_{i,j} = \phi\left(q_i^\top h_{i,j}\right), \tag{3c}$$

[1]For simplicity, we ignore potential output projections from the linear combination to the output.

where in standard dynamical systems' terminology, $h_{i,j} \in \mathbb{R}^n$, $u_i \in \mathbb{R}^n$, and $\alpha_{i,j} \in \mathbb{R}$ are the state, input, and output of the system at time i with an impulse input at time j, respectively. The readout map is given by $g_i(x) \coloneqq \phi\left(q_i^\top x\right)$, where $q_i \in \mathbb{R}^n$ is the query, $k_j \in \mathbb{R}^n$ is the key, and $\phi(\cdot) : \mathbb{R} \to \mathbb{R}$ is a general nonlinear map. $A_i \in \mathbb{R}^{n\times n}$ and $b_i \in \mathbb{R}$ are the evolution matrix and the input scaling, respectively. For A_i we focus on scalar matrices, $A_i = \lambda_i \mathbb{I}_n$, diagonal matrices, $A_i = \mathrm{diag}(\lambda_i)$, and Householder matrices.[2] However, generally A_i can have no particular structure. We consider the *autonomous* evolution of dynamical system (3), i.e., given a zero initial condition, we study the system's evolution when it receives an impulse input at time j and then evolves without further input.

Remark 1. *This formulation differs from existing state space models (e.g., Mamba) in two key aspects: (i) the system receives input only at a single time step $i = j$, and (ii) the hidden state is zero immediately before the impulse, not at global time $i = -1$.*

It is possible to recover an analytic expression for the coefficients $\alpha_{i,j}$ in (2) by writing the convolution expression associated with (3). In particular, the state (or "transformed key") evolves, for $j \leq i$, as $h_{i,j} = \left(\prod_{t=j+1}^{i} A_t\right) b_j k_j$. Hence, we obtain the form:

$$\begin{aligned} &\alpha_{i,j} = \phi(x_i^\top T_q^\top T_{h,i} x_j), \\ &\text{with} \quad T_{h,i} := \left(\textstyle\prod_{t=j+1}^{i} A_t\right) b_j W_K, \quad T_q := W_Q. \end{aligned} \tag{4}$$

This interpretation allows for the identification of four classes of parameters with distinct roles:

- **Readout map ($\phi(\cdot)$):** Applies pointwise (often nonlinear) transformations to dot product $q_i^\top h_{i,j}$.
- **Evolution matrices (A_t):** Control the temporal evolution of keys through scaling (diagonal matrices) or rotation (orthogonal matrices). Importantly, the evolution matrices define a sequence of transformation on the keys between input time j and readout time i.
- **Scaling parameters (b_j):** Scale individual keys at their input time j.
- **Normalization factors (η_i):** Normalize the $\alpha_{i,j}$ coefficients, before the linear combination.

This framework provides a unified lens for understanding existing architectural proposals. We show in Table 1 that popular architectures can be recovered as special cases of this framework.

4 Design Principles for Sequence Models

In the following, we provide principles that can inform the architecture design for sequence models. These principles are grounded in theoretical results for the coefficient dynamics (as described in (2) & (3)) and followed by a discussion of their practical applicability. Formal proofs for the lemmas and corollaries are provided in Appendices C.1 and C.2, respectively.

4.1 On choosing the Readout Map $\phi(\cdot)$

A fundamental issue with softmax attention is its quadratic complexity in sequence length, directly linked to its nonlinear readout map $\phi(\cdot) = \exp(\cdot)$. This fact is captured by the following principle.

Principle 1. *On modern hardware, a sequence model can only be efficiently computed using a recurrent formulation (e.g. parallel scan) if the readout function $\phi(\cdot)$ is polynomial.*

Lemma 1. *A recurrent formulation of* (3) *with finite memory (state) exists if the readout map $\phi(\cdot) : \mathbb{R} \to \mathbb{R}$ is a polynomial of finite degree P. The requisite memory (state) then expands to $\mathbb{R}^{n^P \times d_v}$.*

[2] A *Householder matrix* is an orthogonal matrix of the form $A_i = I - 2z_i z_i^\top / \|z_i\|^2$, with z_i a nonzero vector; and represents a reflection through the hyperplane orthogonal to v_i.

Table 1: Coefficient dynamics of popular architectures; the derivations are provided in Appendix B.

Architecture	Parameters				Coefficients
	A_t	b_j	$\phi(\cdot)$	η_i	$\alpha_{i,j}$
Softmax Attention (Vaswani et al., 2017)	$\mathbb{I}_n$	$\frac{1}{\sqrt{n}}$	$\exp(\cdot)$	$\sum_{j=0}^{i} \alpha_{i,j}$	$\frac{\exp(q_i^\top k_j/\sqrt{n})}{\sum_{j=1}^{i} \exp(q_i^\top k_j/\sqrt{n})}$
Linear Attention (Katharopoulos et al., 2020)	$\mathbb{I}_n$	$\frac{1}{\sqrt{n}}$	$\psi(\cdot)\psi(\cdot)$	$\sum_{j=0}^{i} \alpha_{i,j}$	$\frac{\psi(q_i)^\top \psi(k_j/\sqrt{n})}{\psi(q_i)^\top \sum_{j=1}^{i} \psi(k_j/\sqrt{n})}$
Normalized Attention (Sieber et al., 2024)	$\mathbb{I}_n$	$\frac{1}{\sqrt{n}}$	$\mathrm{Id}(\cdot)$	η_i	$\frac{q_i^\top k_j}{\eta_i \sqrt{n}}$
GLA (Yang et al., 2023)	$\mathrm{diag}(\alpha_i)$	$\frac{1}{\sqrt{n}}$	$\mathrm{Id}(\cdot)$	1	$q_i^\top \left(\prod_{t=j+1}^{i} \mathrm{diag}(\alpha_t)\right) \frac{k_j}{\sqrt{n}}$
Mamba-2 (Dao & Gu, 2024)	$e^{-\Delta_t A}$	Δ_j	$\mathrm{Id}(\cdot)$	1	$q_i^\top \left(\prod_{t=j+1}^{i} e^{(-\Delta_t A)}\right) \Delta_j k_j$
DeltaNet (Schlag et al., 2021)	$\mathbb{I} - \beta_t k_t k_t^\top$	$\frac{\beta_j}{\sqrt{n}}$	$\mathrm{Id}(\cdot)$	1	$q_i^\top \left(\prod_{t=j+1}^{i} \mathbb{I} - \beta_t k_t k_t^\top\right) \frac{\beta_j k_j}{\sqrt{n}}$
Gated DeltaNet (Yang et al., 2025a)	$\alpha_t(\mathbb{I} - \beta_t k_t k_t^\top)$	$\frac{\beta_j}{\sqrt{n}}$	$\mathrm{Id}(\cdot)$	1	$q_i^\top \left(\prod_{t=j+1}^{i} \alpha_t(\mathbb{I} - \beta_t k_t k_t^\top)\right) \frac{\beta_j k_j}{\sqrt{n}}$
mLSTM (Beck et al., 2024)	e^{f_t}	$\frac{e^{i_j}}{\sqrt{n}}$	$\mathrm{Id}(\cdot)$	$\frac{\eta_i}{o_i}$	$\frac{o_i}{\eta_i} q_i^\top \left(\prod_{t=j+1}^{i} e^{f_t}\right) \frac{e^{i_j}}{\sqrt{n}} k_j$

Various linear attention (e.g., Katharopoulos et al. (2020); Choromanski et al. (2021)) and SSM/RNN (e.g., Dao & Gu (2024); Beck et al. (2024); De et al. (2024)) proposals implicitly utilize this principle to design sequence models with linear complexity in sequence length. In practice, ϕ is often chosen to be linear, to avoid exponential computational costs, which has several geometric implications that directly affect performance on crucial recall and retrieval tasks (Arora et al., 2024; Guo et al., 2025).

Beyond implementation considerations, in domains such as language processing, an important feature of sequence models is their ability to selectively attend to different tokens in the input sequence. This is often referred to as *input selectivity*. Mathematically, this translates to the model's capacity to set coefficients $\alpha_{i,j}$ equal to zero (or near-zero in practice) for uninformative tokens and to high values for informative tokens. Crucially, how informative a token is depends on the query, as is clear for tasks such as associative recall (Arora et al., 2023).

Principle 2. *The capability of a model to robustly set coefficients to zero (or near-zero) depends on the geometry of the zero-level set of the readout map $\phi(\cdot)$. Intuitively, if the zero (or near-zero) set has large measure in $\mathbb{R}$, values can be suppressed more robustly. Conversely, if the zero set is thin, suppression is fragile to learn.*

Lemma 2. *Let $\phi : \mathbb{R} \to \mathbb{R}$ be the readout map with connected zero-level set $\mathcal{Z} = \{z \in \mathbb{R} \mid \phi(z) = 0\}$, and let $z = q_i^\top h_{i,j}$, with T_q, $T_{h,i}$ defined in (4). Let $\mathcal{T}$ be the set of linear transformations that map to the zero-level set, i.e., $\mathcal{T} = \{(T_q, T_{h,i}) \mid q_i^\top h_{i,j} \in \mathcal{Z}$ s.t. $q_i = T_q x_i$, $h_{i,j} = T_{h,i} x_j\}$ for any input pairs $x_i, x_j \in \mathbb{R}^d$. The measure $|\mathcal{T}|$ is then proportional to the zero-level set measure $|\mathcal{Z}|$, i.e., $|\mathcal{T}| = c_\phi |\mathcal{Z}|$, where $c_\phi > 0$ depends on $\phi(\cdot)$ and represents the mean geometric density of these transformations.*

Although Lemma 2 only considers zero-level sets, large near-zero level sets $\{x \mid -\epsilon \leq \phi(x) \leq \epsilon\}$ with small $\epsilon > 0$, provide effective input selectivity in practice. Intuitively, the proof follows from the fact that for an interval $\mathcal{Z}$, we can construct a set of linear combinations $\mathcal{T}_z = \{(T_q, T_{h,i}) \mid q_i^\top h_{i,j} = z\}$ that achieves z for each $z \in \mathcal{Z}$. A larger set $\mathcal{Z}$ then implies that $\mathcal{T}$ is constructed from more subsets $\mathcal{T}_z$. Hence, when the zero-level set $\mathcal{Z}$ has large measure in $\mathbb{R}$ ($|\mathcal{Z}| \gg 0$), many linear transformations $(T_q, T_{h,i})$ can achieve $\alpha_{i,j} = 0$, enabling robust input selectivity. While our analysis establishes the existence of these beneficial configurations, determining whether gradient descent actually converges to any transformation in $\mathcal{T}$, requires a deeper analysis of specific optimization dynamics, loss functions, and architectures. However, as argued

in Goodfellow et al. (2016), the abundance of high-dimensional parameter configurations, yielding a desired behavior, increases the likelihood that gradient descent will converge to such parameters during training.

Example 1 (Standard nonlinear readout maps)**.** *The dominant choice for readout map $\phi(\cdot)$ is the exponential function* $\exp(\cdot)$ *(as in softmax attention), which is a positive monotonic non-decreasing function that is close to zero on the negative domain, i.e., $\phi(x) \to 0$ for $x < 0$. In this case, $\alpha_{i,j} \approx 0$ can be achieved by $q_i^\top h_{i,j} < 0$, which happens if the angle between the two vectors $q_i, h_{i,j}$ lies in $[\frac{\pi}{2} + n\pi, \frac{3\pi}{2} + n\pi]$ for any $n \in \mathbb{Z}^+$. This results in a large set $\mathcal{T}$, since many combinations of $q_i, h_{i,j}$ will lead to coefficients being approximately zero.*

Together, Lemmas 1 and 2 show the tradeoff in the choice of $\phi(\cdot)$ and offer a new perspective on the tradeoff between SSMs and Transformers (Gu, 2025): on the one hand, a nonlinear choice facilitates input selectivity; on the other, a linear choice facilitates efficient computation. To address this trade-off, various kernel approximation methods, i.e. $\phi \approx \psi_q(\cdot)\psi_k(\cdot)$, have been proposed in the literature, e.g. (Katharopoulos et al., 2020; Tsai et al., 2019; Arora et al., 2024), which try to approximate the behavior of $\phi(\cdot) = \exp(\cdot)$ as closely as possible, while striving to fulfill Principle 2. However, regardless of how exact the approximation is, there is a fundamental geometric issue with these approaches, as outlined in the next principle.

Principle 2.1. *If the readout map is of the form $\phi(\cdot) = \psi_q(\cdot)\psi_k(\cdot)$ (e.g.,kernel approximation), its zero-level set is of measure zero. Hence, by Principle 2, input selectivity is fragile to learn.*

Corollary 2.1. *Consider the kernel approximation of $\phi(\cdot) : \mathbb{R}^n \to \mathbb{R}^q$ to be $\tilde{\phi}(q_i^\top h_{i,j}) = \psi_q(q_i)^\top \psi_h(h_{i,j})$, with $\psi_q : \mathbb{R}^n \to \mathbb{R}^q$ and $\psi_h : \mathbb{R}^n \to \mathbb{R}^q$, such that it approximates the readout map $\phi(\cdot) \approx \tilde{\phi}(\cdot)$ in* (3)*. Then, the zero-level set of the approximation $\tilde{\phi}(\cdot)$ is the singleton $\mathcal{Z} = \{0\}$ and by Lemma 2 the set of linear transformations $\mathcal{T}$ is small.*

By Principle 2 and the results from Lemma 2, choosing kernel approximations as readout maps hinders the capacity for input selectivity. To overcome this limitation, the kernel approximations $\psi_q(\cdot), \psi_k(\cdot)$ need to be designed carefully. While, setting ψ_q, ψ_h to positive functions, e.g. $\psi_q(x) = \psi_h(x) = \mathrm{elu}(x) + 1$ (Katharopoulos et al., 2020), preserves the positivity of the coefficients, it eliminates much of the crucial geometric information stored in q_i and $h_{i,j}$, since the angle between these vectors is not preserved. To alleviate this issue, some approximation proposals induce a bias towards near-zero coefficients by explicitly using orthogonal feature maps, e.g. FAVOR+ (Choromanski et al., 2021) or random feature maps (Peng et al., 2021).

We note that Principle 2 presents general considerations for driving coefficients $\alpha_{i,j}$ to zero. Yet, by (2), i coefficients are simultaneously computed using a single query q_i and i states (or transformed keys) $h_{i,j}$ with $j = 1, \ldots, i$. In what follows, we illustrate that achieving near-zero coefficients $\alpha_{i,j}$ *simultaneously* is a more restrictive setup than the general setting of Principle 2.

Principle 2.2. *If the readout map is the identity $\phi(\cdot) = \mathrm{Id}(\cdot)$, the number of coefficients that can be simultaneously suppressed by a nonzero query vector, is limited to $n - 1$. Beyond this limit, suppression necessarily collapses to the trivial (zero) query vector.*

Corollary 2.2. *Let $y_L \in \mathbb{R}^{d_v}$ be the solution to* (2)*, where $\alpha_{L,j} : \mathbb{R}^n \to \mathbb{R};\ q_L \mapsto \alpha_{L,j}(q_L)$ is defined in* (3) *with identity readout map and normalization factor, i.e., $\phi(\cdot) = \mathrm{Id}(\cdot)$, $\eta_L = 1$. Consider the set of linearly independent states $\mathcal{H} = \{h_{L,t} \mid c_1 h_{L,1} + \cdots + c_t h_{L,t} = 0 \implies c_1 = \cdots = c_t = 0\}$. Then, a nonzero q_L that achieves $\alpha_{L,j} = 0$, $\forall h_{L,j} \in \mathcal{H}$ exists if and only if $\dim\big(\mathrm{span}\{h_{L,j} \in \mathcal{H}\}\big) < n$. In particular, given a nonzero q_L, the measure $|\mathcal{H}| \leq n - 1$.*

Principle 2.2 is primarily a limiting factor for identity readout maps, since zero coefficients $\alpha_{i,j}$ are only achieved for $q_i^\top h_{i,j} = 0$, i.e., q_i and $h_{i,j}$ are orthogonal. For nonlinear readout maps (see Example 1), many configurations of q_i and $h_{i,j}$ achieve $\alpha_{i,j} = 0$.

Can learnable parameters save us? We established that identity readout maps $\phi(\cdot) = \mathrm{Id}(\cdot)$ can suppress coefficients – if q_i, $h_{i,j}$ are orthogonal – but results in a much smaller set $\mathcal{T}$ than for nonlinear $\phi(\cdot)$, which in principle makes the learning problem harder. However, since $T_{h,i}$ in (4) is a function of $A_{i,j} \coloneqq \prod_{t=j+1}^{i} A_t$

and b_j, these can be carefully parametrized and leveraged to achieve $h_{i,j} = 0$, which also results in zero coefficients. This is the central design paradigm of many SSM proposals, e.g. Dao & Gu (2024); Yang et al. (2025a). We elaborate more on how these choices can compensate for a linear readout map in the next section. However, it is important to note that although the choices of $A_{i,j}$ and b_j can be used to suppress certain tokens, these are typically designed for input selectivity on the keys. For example, choosing $A_t = 1 - \lambda_t$, $b_j = \lambda_j$ (as in GRU Cho et al. (2014)),[3] enables selection of the state or key ((3)). While this is straightforward to understand in an SSM setting, where the state aggregates the keys over time, our framework directly links this gating behavior to the coefficients $\alpha_{i,j}$. Choosing $\lambda_j = 0$ for a key k_j results in suppression of this key in all future time steps, i.e., $\alpha_{i,j} = 0, \forall i \geq j$. Additionally, choosing $\lambda_t = 1$ or equivalently $A_t = 0$ for any $t \in [j, i]$, results in a zero state $h_{i,j}$ for all $i \geq t$, thus suppressing the coefficients. This shows that input selectivity design in SSMs, i.e., design of A_t, b_j, is a special case of the more general input selectivity design for coefficients considered in this work.

4.2 On choosing the Evolution Matrices A_t

It is well-known that softmax attention cannot distinguish between two identical inputs that appear at different (time) positions in the sequence (Yun et al., 2020). The reason for this is the evolution matrix ($A_t = \mathbb{I}$) used in softmax attention and is commonly mitigated via positional embeddings (e.g. Su et al. (2024)). However, careful design of the evolution matrix enables processing of positional information in the attention mechanism, as evidenced by SSMs (Dao & Gu, 2024).

Principle 3. *If $A_t = \mathbb{I}$, no positional information is contained in the coefficients $\alpha_{i,j}$. This means for a sequence model choosing A_t to be the identity, positional embeddings are needed.*

Definition 1 (Positional Information)**.** *$\alpha_{i,j}$ in (4) are said to have positional information, if for two identical inputs $x_j = x_{\bar{j}} = \bar{x}$, the resulting coefficients are not identical, i.e., $\alpha_{i,j} \neq \alpha_{i,\bar{j}}$.*

Lemma 3. *The coefficients $\alpha_{i,j}$ in (4) have positional information if and only if $A_t \neq \mathbb{I}_n, \forall t$.*

By Lemma 3, positional information can be embedded in the coefficients by designing A_t appropriately, further strengthening our intuition why linear RNN/SSM proposals (such as Mamba-2, DeltaNet, mLSTM) do not require positional embeddings to learn in-context recall tasks (Arora et al., 2023; Poli et al., 2024; Okpekpe & Orvieto, 2025) or language (Gu & Dao, 2023). However, despite positional embeddings not being strictly necessary, using positional embeddings for RNNs and SSMs ($A_t \neq \mathbb{I}$) can be beneficial in practice (Morita, 2024).

Beyond enabling processing of positional information, the choices of A_t can selectively suppress tokens under some circumstances. Specifically, as discussed in Principle 2, when the readout map $\phi(\cdot)$ is a linear function, $\alpha_{i,j} = 0$ is only achieved when q_i and $h_{i,j}$ are orthogonal, or if one of the vectors $q_i^\top, h_{i,j}$ (or both) is zero. Since relying on the orthogonality of $q_i^\top$ and $h_{i,j}$ leads to harder learning problem (per Principle 2), many architectures that use a linear readout map for computational reasons (as per Principle 1), rely on dynamics (3) to achieve selectivity, e.g. Mamba-2. In these models, the right choice of the dynamics components A_t and b_j is crucial to ensure that $h_{i,j}$ can be effectively driven to zero. Additionally, the evolution matrices A_t can be used to rotate non-orthogonal keys over time, to achieve $q_i^\top h_{i,j} = 0$, despite $q_i^\top k_j \neq 0$.

Principle 4. *Assuming a linear readout map $\phi(\cdot)$, the structure imposed on the evolution matrices A_t limits the operations (e.g. scaling, rotation) that can be performed on the keys.*

Lemma 4. *Imposing any of the following structures on A_t in (3), results in the corresponding allowed transformations for the keys $\{k_j\}_{j=1}^i$:*

a) $A_t = \lambda_t \mathbb{I}_n, \lambda_t \in \mathbb{R}, |\lambda_t| \leq 1$; allows scaling of keys (including flipping, if $\lambda_t < 0$) uniformly along all dimensions n,

[3] A similar inverse relationship between A_t and b_j is found in many SSM proposals, see e.g. Dao & Gu (2024); De et al. (2024); Schlag et al. (2021).

b) $A_t = diag(\lambda_t), \lambda_t \in \mathbb{R}^n, |\lambda_t^{(r)}| \leq 1$ *where* $\lambda^{(r)}$ *denotes the r-th entry of* λ_t*; allows scaling of keys (including flipping, if* $\lambda_t^{(r)} < 0$*) separately along dimensions* n*,*

c) $A_t = \mathbb{I}_n - \beta_t \lambda_t \lambda_t^\top, \beta_t \in [0, 2], \lambda_t \in \mathbb{R}^n$*, (Householder matrix); allows scaling and specific rotations of keys.*

Lemma 4 provides guidance on how to design the evolution matrices A_t, depending on the transformations that should be enabled. Note that it is possible to combine these structures, e.g., combine a) and c) in Lemma 4 to obtain a scaled Householder matrix (Yang et al., 2025a). This list is not exhaustive and more general structures can be imposed on A_t to achieve other transformations. Yet, these structures might be prohibited by current CUDA kernel implementations (Siems et al., 2025).

4.3 On choosing the Scaling Parameters b_j

As discussed in Section 4.1, the scaling parameters b_j are often linked to the evolution matrices A_t (see Table 1). While the need for scaling/normalization of keys has been established in previous architectural proposals, there is a lack of consensus on how to design b_j. The following result is based on classical signal propagation analyses (Glorot & Bengio, 2010; He et al., 2015).

Principle 5. *Choosing the scaling parameter* $b_j = \mathcal{O}(1/\sqrt{n})$ *ensures that the dot product* $q_i^\top h_{i,j}$ *has* $\mathcal{O}(1)$ *variance; anything larger than this scale lets the variance grow with* n*.*

Lemma 5. *Consider two i.i.d. normally distributed random vectors* $x_i, x_j \sim \mathcal{N}(0, \Sigma)$*. Then, the dot product* $q_i^\top h_{i,j}$ *with* $q_i, h_{i,j}$ *defined in (4), has zero-mean and variance* $Var(q_i^\top h_{i,j}) = tr(\Sigma_q \Sigma_h)$ *with* $\Sigma_q = T_q \Sigma T_q^\top$*,* $\Sigma_h = T_{h,i} \Sigma T_{h,i}^\top$*. Assuming* $\Sigma = \sigma \mathbb{I}_d$*, both* Σ_q *and* Σ_h *are positive semi-definite and the variance of the dot product scales as* $Var(q_i^\top h_{i,j}) = \mathcal{O}(n)$*.*

Note that in Vaswani et al. (2017, Section 3.2.1), the statement of Lemma 5 is discussed informally. While the scaling factor choice $b_j = 1/\sqrt{n}$ is widely used for attention-based models, in SSMs or RNNs, b_j is more often designed in combination with evolution matrices A_t. Since in these models, A_t and b_j have an inverse relationship and A_t is typically designed to have eigenvalues close to the unit circle, this forces b_j to be small. If the entangling of A_t and b_j is abandoned, Principle 5 provides a minimum scaling level that is essential for stable training of a sequence model.

Example 2. *Mamba-2 parameterizes both* A_t *and* b_j *with* Δ_j *(see Table 1), which is biased to lie in the range* $\Delta_j \in [0.001, 0.1]$*. This range can be thought of as* $1/\sqrt{n}$ *for* $n \in [1e2, 1e6]$*, thus fulfilling Principle 5 for a wide range of dimensions* n*.*

4.4 On choosing the Normalization Factors η_i

While the scaling factors b_j scale the keys, the normalization factors η_i directly scale the $\alpha_{i,j}$s. Therefore, the normalization factors offer a way to re-scale all coefficients at time index i (amplifying large, reducing small) or normalize the linear combination at every time index individually.

Principle 6. *If the readout map* $\phi(\cdot)$ *is unbounded and/or* A_t *are not stable, the normalization factors* η_i *need to be designed to counteract the growth in the coefficients.*

Lemma 6. *Consider a fixed index* j *and let* $s_i := q_i^\top h_{i,j}$*,* $\alpha_i = \phi(s_i)$ *in (3). Let* $\phi : \mathbb{R} \to [0, \infty)$ *be nondecreasing and unbounded, and let* $g : [0, \infty) \to (0, \infty)$ *be an increasing and unbounded comparison function, such that* $L := \limsup_{i\to\infty} \frac{\phi(s_i)}{g(s_i)} < \infty$*. If* η_i *is chosen such that* $\liminf_{i\to\infty} \frac{\eta_i}{g(s_i)} := m > 0$*, then the normalized coefficients are bounded:* $\limsup_{i\to\infty} \frac{\alpha_i}{\eta_i} \leq \frac{L}{m}$*.*

As an example, consider the mLSTM architecture (Beck et al., 2024), which allows unstable evolution matrices A_t with eigenvalues outside the unit circle and an identity readout map. To fulfill Principle 6, the normalization factors η_i need to be designed such that they grow linearly (or superlinearly) with the state $h_{i,j}$. The model does this by relying on a separate normalization state that uses the same dynamics as the

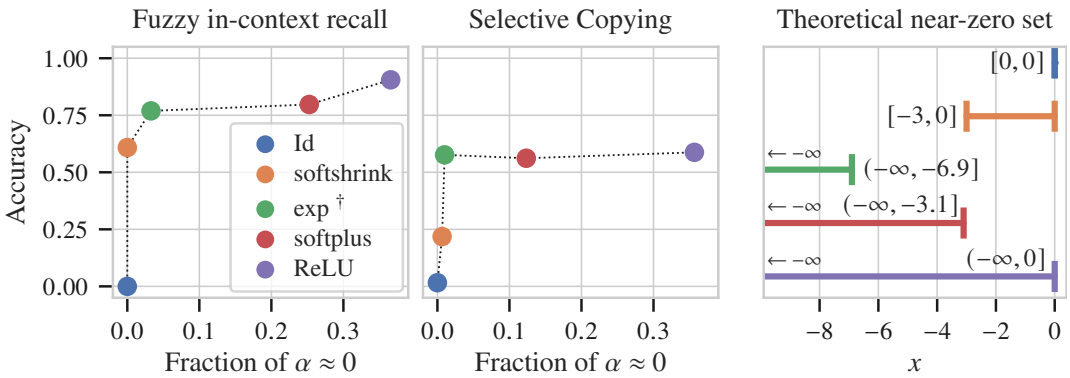


Figure 2: **(Principle 2)** Performance of different readout maps $\phi(\cdot)$ on two MAD tasks against the fraction of coefficients with near zero values ($|\alpha| \leq 0.001$; left & middle) and the theoretical near-zero sets of each readout map ($|\phi(x)| \leq 0.001$; right). The other parameters A_t, b_j, η_i are fixed, thus the setting for † is equivalent to softmax attention.

state (see Table 1). An alternative option to design the normalization factors, is to choose η_i as a function of the coefficients $\alpha_{i,j}$, i.e., $\eta_i = f(\alpha_{i,j})$. In this case, only linear (or superlinear) growth of f is required to fulfill Principle 6 (see Remark 6). A simple choice of a superlinear f is used in softmax attention, and is discussed in the following corollary.

Corollary 6.1. *Choosing* $\eta_i = \sum_{j=1}^{i} \alpha_{i,j}$ *in* (2) *constrains the normalized coefficients to* $\tilde{\alpha}_{i,j} = {}^{\alpha_{i,j}}/_{\eta_i} \in [0, 1]$ *and imposes* $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$.

Corollary 6.1 additionally imposes a constraint on the linear combination in (2), which restricts the outputs y_i to lie in a well-defined geometric space spanned by the value vectors. This fact and its implications are further discussed in Appendix A.

5 Experimental Validation

In this section, we empirically validate the principles discussed in Section 4 on selected tasks of the MAD benchmark (Poli et al., 2024). Additional experiments are reported in Appendix D and all details of the experimental setup are discussed in Appendix E.

Principle 1 Given that the principle concerns the implementation of sequence models, we show its implication on the model's throughput in Appendix D.

Principle 2 Figure 2 shows the accuracy of different readout maps $\phi(\cdot)$ on the fuzzy in-context recall and selective copying tasks of MAD. The other parameters are fixed to $A_t = \mathbb{I}$, $b_j = {}^1/\sqrt{n}$, $\eta_i = \sum_j \alpha_{i,j}$, across all readout maps. Since input selectivity is crucial for both tasks, we plot accuracy against the fraction of coefficients close to 0 (near-zero set with $\epsilon = 0.001$) as well as the theoretical near-zero set of each $\phi(\cdot)$. *The results show, that more coefficients are set close to zero as the near-zero set of the readout map grows. For both tasks this increases performance.* Additional results and the results for Principles 2.1 & 2.2 are provided in Appendix D.

Principle 3 We show that $A_t \neq \mathbb{I}_n$ can replace positional embeddings (PE) for the simplest choice of A_t. Figure 3 shows the accuracy of four models on the noisy in-context recall of MAD, where we set $b_j = {}^1/\sqrt{n}$, $\eta_i = \sum_j \alpha_{i,j}$, and vary $A_t \in \{\mathbb{I}_n, 0.95\,\mathbb{I}_n\}$, $\phi(\cdot) = \{\exp(\cdot), \mathrm{softplus}(\cdot), \mathrm{ReLU}(\cdot)\}$, and wether the models have PE or not. We choose a constant $A_t = 0.95\,\mathbb{I}_n$, since it directly connects to ALiBi (Press et al., 2022). While ALiBi enables positional information via additive biases to the attention matrix, $A_t = 0.95\,\mathbb{I}_n$ does the same but via multiplication. *The experiment shows that noisy in-context recall is solved by both PE and* $A_t \neq \mathbb{I}_n$

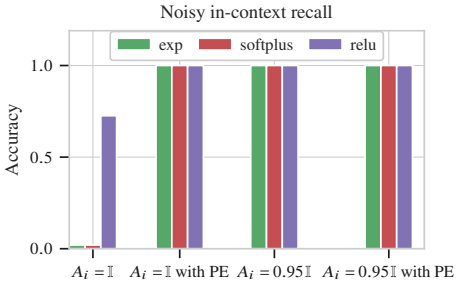


Figure 3: **(Principle 3)** Performance of two A_t choices with and without positional embeddings (PE) on the noisy in-context recall task.

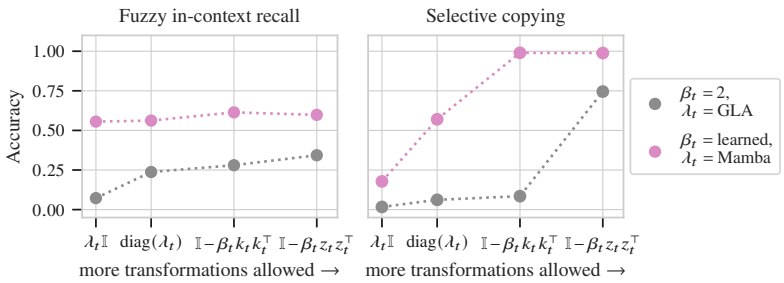


Figure 4: **(Principle 4)** Performance of four A_t choices (scalar, diagonal, Householder with k_t, Householder with learned z_t) on two MAD tasks, with all other parameters fixed. The scalar/diagonal parameter(s) λ_t are using either the GLA (gray) or Mamba-2 (magenta) parameterization and the Householder scaling β_t is either fixed (gray) or learned (magenta).

for all readout maps. For $A_t = \mathbb{I}_n$ without PE, only $\phi(\cdot) = ReLU(\cdot)$ achieves non-random performance but does not perform perfectly.

Principle 4 To show the effect of allowed transformations in A_t on the performance, we ablate four structures imposed on A_t – scalar, diagonal, Householder with keys k_t, and Householder with a learned vector z_t – on the fuzzy in-context recall and selective copying tasks of MAD; the other parameters are fixed to $\phi(\cdot) = \mathrm{Id}(\cdot)$, $\eta_i = 1$, $b_j = 1/\sqrt{n}$ across all choices of A_t. For each A_t, we additionally vary how the scalar and diagonal are parameterized (either using GLA (Yang et al., 2023) or Mamba-2 (Gu & Dao, 2023) parameterizations), and the scaling factor of the rotation vector in the Householder matrix (either $\beta_t = 2$ or learned from the input). *On both tasks, the performance generally increases as more transformations are allowed in A_t (blue line). However, the parameterization of the involved parameters is important and good parameterizations can considerably improve performance (orange line).*

Principles 5 & 6 These principles are mainly concerned with training stability, which we show by ablating various choices of b_j and η_i in Appendix D.

6 Conclusions

This paper studies sequence models with a focus on the *coefficient dynamics* that multiply value vectors for a single layer. We view the sequence model outputs as linear combinations of past values with coefficients produced by autonomous linear dynamics with impulse inputs given by the keys. This view captures a broad class of existing sequence models, which can be recovered as a special case of this formalization. Furthermore, we show how studying the *coefficient dynamics* sheds light onto how normalization, geometric operations, and state updates influence input selectivity, efficient implementations, and training stability. These insights, formalized as mathematical results, unlock six design principles for sequence models that enable the tailored design of models. Experimental results validate these principles.

Limitations: The present study is limited to single-layer setups and more work is needed to translate the proposed principles to multi-layer models. It also does not cover optimization considerations and the role of other components in the architecture, such as convolutions, gates, and specific positional embeddings, which could inform new principles or strengthen existing ones. Finally, the principles are validated on synthetic datasets and require additional experiments on real-world applications, such as language modeling.

References

AI21 Labs. Ai21-jamba-large-1.7. https://huggingface.co/ai21labs/AI21-Jamba-Large-1.7, 2024. Accessed: 2025-09-22.

Ameen Ali, Itamar Zimerman, and Lior Wolf. The hidden attention of mamba models. *arXiv preprint arXiv:2403.01590*, 2024.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and Improving Recall in Efficient Language Models. *arXiv:2312.04927*, 2023.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. Simple linear attention language models balance the recall-throughput tradeoff. In *Forty-first International Conference on Machine Learning*, 2024.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended Long Short-Term Memory. *arXiv preprint arXiv:2405.04517*, 2024.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. Massive exploration of neural machine translation architectures. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1442–1451, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1151. URL https://aclanthology.org/D17-1151/.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi (eds.), *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL https://aclanthology.org/W14-4012/.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. Rethinking Attention with Performers. In *International Conference on Learning Representations*, 2021.

Nicola Muca Cirone, Antonio Orvieto, Benjamin Walker, Cristopher Salvi, and Terry Lyons. Theoretical foundations of deep selective state-space models. *arXiv preprint arXiv:2402.19047*, 2024.

Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. 2023.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms with Structured State Space Duality. In *ICML 2024*, 2024.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models, 2024. URL https://arxiv.org/abs/2402.19427.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Herbert Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93(3):418–491, 1959.

Stephen H Friedberg, Arnold J Insel, and Lawrence E Spence. *Linear algebra*. Prentice Hall, 1997.

Daniel Y. Fu, Tri Dao, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry Hungry Hippos: Towards Language Modeling with State Space Models, 2023. URL https://arxiv.org/abs/2212.14052.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Albert Gu. On the tradeoffs of state space models and transformers, 2025. URL https://goombalab.github.io/blog/2025/tradeoffs/.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, 2023. URL https://arxiv.org/abs/2312.00752.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. HiPPO: Recurrent Memory with Optimal Polynomial Projections. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1474–1487. Curran Associates, Inc., 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently Modeling Long Sequences with Structured State Spaces. In *The International Conference on Learning Representations (ICLR)*, 2022a.

Albert Gu, Ankit Gupta, Karan Goel, and Christopher Ré. On the Parameterization and Initialization of Diagonal State Space Models, 2022b. URL https://arxiv.org/abs/2206.11893.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

DongNyeong Heo and Heeyoul Choi. Generalized probabilistic attention mechanism in transformers, 2024. URL https://arxiv.org/abs/2410.15578.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Samy Jelassi, David Brandfonbrener, Sham M Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*, 2024.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Tobias Katsch. Gateloop: Fully data-controlled linear recurrence for sequence modeling, 2023.

Zhixuan Lin, Evgenii Nikishin, Xu Owen He, and Aaron Courville. Forgetting transformer: Softmax attention with a forget gate. *arXiv preprint arXiv:2503.02130*, 2025.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. *arXiv preprint arXiv:2404.08819*, 2024.

Takashi Morita. Positional encoding helps recurrent neural networks handle a large vocabulary. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=PtnwXd13SF.

NVIDIA Nemotron Nano. Efficient hybrid mamba-transformer reasoning model. *arXiv preprint arXiv:2508.14444*, 2025.

Eric Nguyen, Michael Poli, Marjan Faizi, Armin Thomas, Michael Wornow, Callum Birch-Sykes, Stefano Massaroli, Aman Patel, Clayton Rabideau, Yoshua Bengio, et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *Advances in neural information processing systems*, 36:43177–43201, 2023.

Destiny Okpekpe and Antonio Orvieto. Revisiting associative recall in modern recurrent models. In *First Workshop on Scalable Optimization for Efficient and Adaptive Foundation Models*, 2025.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting Recurrent Neural Networks for Long Sequences. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 26670–26698. PMLR, 23–29 Jul 2023.

Antonio Orvieto, Soham De, Caglar Gulcehre, Razvan Pascanu, and Samuel L Smith. Universality of linear recurrences followed by non-linear projections: Finite-width guarantees and benefits of complex eigenvalues. *International Conference on Machine Learning*, 2024.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=QtTKTdVrFBB.

Michael Poli, Armin W. Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, Ce Zhang, and Stefano Massaroli. Mechanistic design and scaling of hybrid architectures. In *ICML*, 2024. URL https://openreview.net/forum?id=GDp7Gyd9nf.

Ofir Press, Noah Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=R8sQPpGCv0.

Qwen Team. Qwen3-next: Towards ultimate training and inference efficiency. https://qwen.ai/blog?id=e34c4305036ce60d55a0791b170337c2b70ae51d&from=home.latest-research-list, 2025. Accessed: 2025-09-22.

Oliver Richter and Roger Wattenhofer. Normalized attention without probability cage, 2020. URL https://arxiv.org/abs/2005.09561.

Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, pp. 9355–9366. PMLR, 2021.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Jerome Sieber, Carmen A Alonso, Alexandre Didier, Melanie N Zeilinger, and Antonio Orvieto. Understanding the differences in foundation models: Attention, state space models, and recurrent neural networks. *Advances in Neural Information Processing Systems*, 37:134534–134566, 2024.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. Deltaproduct: Improving state-tracking in linear rnns via householder products. *arXiv preprint arXiv:2502.10297*, 2025.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified State Space Layers for Sequence Modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL https://www.sciencedirect.com/science/article/pii/S0925231223011864.

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, 2023.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long Range Arena : A Benchmark for Efficient Transformers. In *International Conference on Learning Representations (ICLR)*, 2021.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: a unified understanding of transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Johannes von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. pp. 35151–35174, 2023. URL https://arxiv.org/abs/2212.07677.

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

Junxiong Wang, Jing Nathan Yan, Albert Gu, and Alexander M Rush. Pretraining without attention. *arXiv preprint arXiv:2212.10544*, 2022.

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

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=y8Rm4VNRPH.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=r8H7xhYPwz.

Songlin Yang, Yikang Shen, Kaiyue Wen, Shawn Tan, Mayank Mishra, Liliang Ren, Rameswar Panda, and Yoon Kim. Path attention: Position encoding via accumulating householder transformations. *arXiv preprint arXiv:2505.16381*, 2025b.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxRM0Ntvr.

Nicolas Zucchet and Antonio Orvieto. Recurrent neural networks: vanishing and exploding gradients are not the end of the story. *Advances in Neural Information Processing Systems*, 37:139402–139443, 2024.

Contents of Appendix

A Interpretation of linear combination classes **17**

B Linear Combination Coefficients in Existing Architectures **18**

B.1 Softmax Attention 18

B.2 Linear Attention 18

B.3 Normalized Attention 19

B.4 Gated Linear Attention (GLA) 19

B.5 Mamba-2 20

B.6 DeltaNet 20

B.7 Gated DeltaNet 21

B.8 mLSTM 22

C Proofs **23**

C.1 Lemmas 23

C.2 Corollaries 29

D Extended Experimental Validation **30**

D.1 Principle 1 30

D.2 Principle 2 and Sub-Principles 30

D.3 Principle 5 32

D.4 Principle 6 34

E Details on Experimental Setup **36**

E.1 Global settings 36

E.2 Model Parameters per Experiment 36

Table 2: Summary of linear combination classes depending on the choice of coefficients.

Type	Restrictions	Output Space	Example
Convex comb.	$\tilde{\alpha}_{i,j} \geq 0$ & $\sum_j \tilde{\alpha}_{i,j} = 1$	Simplex	Softmax Attention
Conical comb.	$\tilde{\alpha}_{i,j} \geq 0$	Positive Orthant	Normalized Attention
Affine comb.	$\sum_j \tilde{\alpha}_{i,j} = 1$	Hyperplane	GPAM (Heo & Choi, 2024)
Linear comb.	None	$\mathbb{R}^{d_v}$	Mamba-2, DeltaNet

A Interpretation of linear combination classes

Our perspective of viewing a sequence model's output as a linear combination of value vectors ((2)) has important considerations. In particular, restrictions imposed on the normalized coefficients $\tilde{\alpha}_{i,j} = \frac{\alpha_{i,j}}{\eta_i}$ fully determine the output space of the model, as spanned by the value vectors. Specifically, any linear combination of the form (2) can be divided into four different classes depending on the restrictions imposed. These classes together with their restrictions, output space, and example architectures are provided in Table 2. The output space of these classes are also visualized in Figure 5.

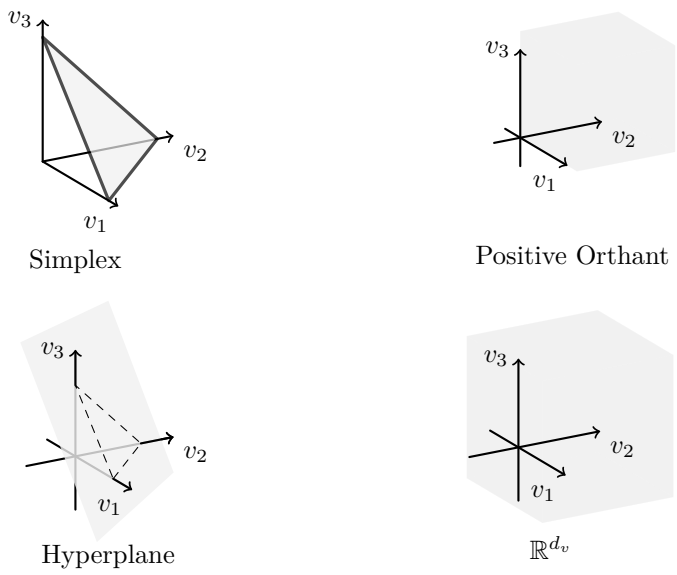


Figure 5: Visual representation of output spaces referenced in Table 2.

The structure of the output space defined by the normalized coefficients $\tilde{\alpha}_{i,j}$ and the value vectors $\{v_1, \ldots, v_i\}$ can be linked to a model's performance, however this area remains underexplored. For instance, the output of softmax attention lies in the convex hull (simplex) of previous value vectors, which has been linked to various performance traits, such as in-context learning (von Oswald et al., 2023),[4] and failure to represent basic logical operators (`XOR`) (Richter & Wattenhofer, 2020). More research into the role of linear combination classes and their effect on a model's performance or properties is needed to better understand the implications, but we believe this paper has shown the advantages of viewing sequence models through the lens of linear combinations.

[4]Their derivation of the in-context gradient step hinges on the probability simplex imposed by the softmax operation.

B Linear Combination Coefficients in Existing Architectures

In the following, we derive the coefficient dynamics of all architectures listed in Table 1. To do this, we make use of a model's representation for a single output token (e.g. (5)) and (4) to transfer each model into a linear combination ((2)).

B.1 Softmax Attention

Lemma 7 (Coefficient Dynamics of Softmax Attention)**.** *Softmax attention (Vaswani et al., 2017, eq. 1)*

$$y_i = \sum_{j=0}^{i} \frac{\exp(\frac{q_i^\top k_j}{\sqrt{n}})v_j}{\sum_{j=1}^{i} \exp(\frac{q_i^\top k_j}{\sqrt{n}})} \tag{5}$$

can be written in the form of (2)*, by setting* $A_i = \mathbb{I}_n$*,* $b_j = \frac{1}{\sqrt{n}}$*,* $\phi(\cdot) = \exp(\cdot)$*, and* $\eta_i = \sum_{j=1}^{i} \exp(\frac{q_i^\top k_j}{\sqrt{n}})$*. The resulting normalized coefficients satisfy* $\tilde{\alpha}_{i,j} \geq 0$ *and* $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$*, placing the output in the convex hull of the value vectors.*

Proof. First, identify the following parameters, by comparing (5) to (2):

$$\alpha_{i,j} = \exp(\frac{q_i^\top k_j}{\sqrt{n}}), \quad \eta_i = \sum_{j=1}^{i} \exp(\frac{q_i^\top k_j}{\sqrt{n}}).$$

Then, the parameter choices $A_i = \mathbb{I}_n$, $b_j = \frac{1}{\sqrt{n}}$, and $\phi(\cdot) = \exp(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

The properties $\tilde{\alpha}_{i,j} \geq 0$ follow from $\exp(\cdot) \geq 0$, and $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$ follows by construction of the normalization factor η_i. Therefore, the output $y_i = \sum_{j=1}^{i} \tilde{\alpha}_{i,j} v_j$ lies in the convex hull of $\{v_1, \ldots, v_i\}$. □

B.2 Linear Attention

Lemma 8 (Coefficient Dynamics of Linear Attention)**.** *Linear attention (Katharopoulos et al., 2020, eq. 4)*

$$y_i = \sum_{j=1}^{i} \frac{\psi(q_i)^\top \psi(k_j/\sqrt{n})v_j}{\sum_{j=0}^{i} \psi(q_i)^\top \psi(k_j/\sqrt{n})} \tag{6}$$

with $\psi(\cdot) \geq 0$*, can be written in the form of* (2) *by setting* $A_i = \mathbb{I}_n$*,* $b_j = \frac{1}{\sqrt{n}}$*,* $\phi(\cdot) = \psi(\cdot)\psi(\cdot)$*, and* $\eta_i = \sum_{j=1}^{i} \psi(q_i)^\top \psi(k_j/\sqrt{n})$*. The resulting normalized coefficients satisfy* $\tilde{\alpha}_{i,j} \geq 0$ *and* $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$*, placing the output in the convex hull of the value vectors.*

Proof. First, identify the following parameters, by comparing (6) to (2):

$$\alpha_{i,j} = \psi(q_i)^\top \psi(\frac{k_j}{\sqrt{n}}), \quad \eta_i = \sum_{j=1}^{i} \psi(q_i)^\top \psi(k_j/\sqrt{n}).$$

Then, the parameter choices $A_i = \mathbb{I}_n$, $b_j = \frac{1}{\sqrt{n}}$, and $\phi(\cdot) = \psi(\cdot)\psi(\cdot)$[5] ensure the coefficients $\alpha_{i,j}$ align with (4). Note that $\psi(\cdot)$ can be seen as preprocessing steps of the queries and keys, therefore allowing to write the coefficients as $\alpha_{i,j} = \tilde{q}_i^\top \tilde{k}_j/\sqrt{n}$ with $\tilde{q}_i = \psi(q_i)$ and $\tilde{k}_j = \psi(k_j)$, which reveals that the readout map is effectively the identity as discussed in Section 4.1.

The properties $\tilde{\alpha}_{i,j} \geq 0$ follow from $\psi(\cdot) \geq 0$, and $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$ follows by construction of the normalization factor η_i. Therefore, the output $y_i = \sum_{j=1}^{i} \tilde{\alpha}_{i,j} v_j$ lies in the convex hull of $\{v_1, \ldots, v_i\}$. □

[5] Katharopoulos et al. (2020) uses $\psi(x) = \mathrm{elu}(x) + 1$, but there exist many more proposals for $\psi(\cdot)$ in the literature.

B.3 Normalized Attention

Lemma 9 (Coefficient Dynamics of Normalized Attention)**.** *Normalized attention (Sieber et al., 2024, eq. 22)*

$$y_i = \sum_{j=1}^{i} \frac{\frac{q_i^\top k_j}{\sqrt{n}} v_j}{\exp(W_\eta x_i)} \tag{7}$$

can be written in the form of (2) *by setting* $A_i = \mathbb{I}_n$*,* $b_j = \frac{1}{\sqrt{n}}$*,* $\phi(\cdot) = Id(\cdot)$*, and* $\eta_i = \exp(W_\eta x_i)$*. The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, identify the following parameters, by comparing (7) to (2):

$$\alpha_{i,j} = \frac{q_i^\top k_j}{\sqrt{n}}, \quad \eta_i = \exp(W_\eta x_i).$$

Then, the parameter choices $A_i = \mathbb{I}_n$, $b_j = \frac{1}{\sqrt{n}}$, and $\phi(\cdot) = \mathrm{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

Note that positive preprocessing functions as in linear attention ($\psi(\cdot) \geq 0$) are allowed, which would constrain the normalized coefficients to $\tilde{\alpha}_{i,j} \geq 0$. Otherwise, there are no restrictions on the resulting normalized coefficients $\tilde{\alpha}_{i,j}$, placing the output in the full space $\mathbb{R}^{d_v}$. □

B.4 Gated Linear Attention (GLA)

Lemma 10 (Coefficient Dynamics of Gated Linear Attention)**.** *Gated linear attention (Yang et al., 2023, eq. 3)*[6]

$$s_t = diag(\alpha_t) s_{t-1} + k_t v_t \tag{8}$$

can be written in the form of (2) *by setting* $A_i = diag(\alpha_i)$*,* $b_j = \frac{1}{\sqrt{n}}$*,* $\phi(\cdot) = Id(\cdot)$*, and* $\eta_i = 1$*. The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, we need to unroll (8) to get an elementwise convolutional representation (see Sieber et al. (2024); Dao & Gu (2024)):

$$s_i = \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \mathrm{diag}(\alpha_t) \right) k_j v_j,$$

before computing the output by multiplying the queries:

$$o_i = q_i^\top \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \mathrm{diag}(\alpha_t) \right) k_j v_j.$$

Note that Yang et al. (2023) implicitly assumes that the keys are scaled by $\frac{1}{\sqrt{n}}$,[7] and rearranging above equation yields

$$y_i = \sum_{j=1}^{i} q_i^\top \left(\prod_{t=j+1}^{i} \mathrm{diag}(\alpha_t) \right) \frac{k_j}{\sqrt{n}} v_j.$$

We identify the following parameters by comparing to (2)

$$\alpha_{i,j} = q_i^\top \left(\prod_{t=j+1}^{i} \mathrm{diag}(\alpha_t) \right) \frac{k_j}{\sqrt{n}}, \quad \eta_i = 1.$$

[6]Here, we assume the vector product $k_t v_t$ is well-defined. This is explicit in Yang et al. (2023), but only needed if recurrences are computed, which is not the case in this paper. Therefore, assume here that $k_t v_t$ computes $k_t v_t^\top$ and that the product $q_i^\top s_i$ is transposed after summation. Then, this is the equivalent setting to (Yang et al., 2023, eq. 3) but with all quantities transposed.

[7]See codebase: `https://github.com/fla-org/flash-linear-attention`.

Then, the parameter choices $A_i = \text{diag}(\alpha_i)$,[8] $b_j = \frac{1}{\sqrt{n}}$, and $\phi(\cdot) = \text{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4). The resulting normalized coefficients $\tilde{\alpha}_{i,j}$ are unrestricted in sign and magnitude, placing the output in the full space $\mathbb{R}^{d_v}$. □

B.5 Mamba-2

Lemma 11 (Coefficient Dynamics of Mamba-2)**.** *Mamba-2 (Dao & Gu, 2024)*

$$y_i = \sum_{j=1}^{i} q_i^\top \left(\prod_{t=j+1}^{i} \exp(-\Delta_t A) \right) \Delta_j k_j v_j \tag{9}$$

can be written in the form of (2) *by setting* $A_i = \exp(-\Delta_i A)$*,* $b_j = \Delta_j$*,* $\phi(\cdot) = Id(\cdot)$*, and* $\eta_i = 1$*. The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, note that (9) is directly obtained from state-space duality (SSD; see Sieber et al. (2024); Dao & Gu (2024)). We identify the following parameters by comparing to (2)

$$\alpha_{i,j} = q_i^\top \left(\prod_{t=j+1}^{i} \exp(-\Delta_t A) \right) \Delta_j k_j, \quad \eta_i = 1.$$

Then, the parameter choices $A_i = \exp(-\Delta_i A)$, $b_j = \Delta_j$, and $\phi(\cdot) = \text{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

Note that the *discretization step* Δ_j is computed from the input as $\Delta_j = \text{softplus}(W_\Delta x_i + \beta)$. The resulting normalized coefficients $\tilde{\alpha}_{i,j}$ are unrestricted in sign and magnitude, placing the output in the full space $\mathbb{R}^{d_v}$. □

B.6 DeltaNet

Lemma 12 (Coefficient Dynamics of DeltaNet)**.** *DeltaNet (Schlag et al., 2021, eq. 24)*[9]

$$s_t = s_{t-1} + \beta_t k_t (v_t - k_t^\top s_{t-1}) = (\mathbb{I} - \beta_t k_t k_t^\top) s_{t-1} + \beta_t k_t v_t, \tag{10}$$

can be written in the form of (2) *by setting* $A_i = \mathbb{I} - \beta_i k_i k_i^\top$*,* $b_j = \frac{\beta_j}{\sqrt{n}}$*,* $\phi(\cdot) = Id(\cdot)$*, and* $\eta_i = 1$*. The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, we need to unroll (10) to get an elementwise convolutional representation (see Sieber et al. (2024); Dao & Gu (2024)):

$$s_i = \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \mathbb{I} - \beta_t k_t k_t^\top \right) \beta_j k_j v_j,$$

before computing the output by multiplying the queries:

$$o_i = q_i^\top \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \mathbb{I} - \beta_t k_t k_t^\top \right) \beta_j k_j v_j.$$

[8] In Yang et al. (2023) the evolution matrix is computed as the low-rank parameterization $\alpha_i = \sigma(W_\alpha^2 W_\alpha^1 x_i + b_\alpha)^{\frac{1}{\tau}}$ with $\tau = 16$.

[9] Here, we assume the vector product $k_t v_t$ is well-defined. This is explicit in Schlag et al. (2021), but only needed if recurrences are computed, which is not the case in this paper. Therefore, assume here that $k_t v_t$ computes $k_t v_t^\top$ and that the product $q_i^\top s_i$ is transposed after summation. Then, this is the equivalent setting to (Schlag et al., 2021, eq. 24) but with all quantities transposed.

Note that Schlag et al. (2021, Section 3.1) assumes that the keys are scaled by $\frac{1}{\sqrt{n}}$, and rearranging above equation yields

$$y_i = \sum_{j=1}^{i} q_i^\top \left(\prod_{t=j+1}^{i} \mathbb{I} - \beta_t k_t k_t^\top \right) \frac{\beta_j}{\sqrt{n}} k_j v_j.$$

We identify the following parameters by comparing to (2)

$$\alpha_{i,j} = q_i^\top \left(\prod_{t=j+1}^{i} \mathbb{I} - \beta_t k_t k_t^\top \right) \frac{\beta_j}{\sqrt{n}} k_j, \quad \eta_i = 1.$$

Then, the parameter choices $A_i = \mathbb{I} - \beta_i k_i k_i^\top$, $b_j = \frac{\beta_j}{\sqrt{n}}$,[10] and $\phi(\cdot) = \mathrm{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

As discussed in Schlag et al. (2021, Section 4.2), the keys k_i need to be normalized (i.e. $\|k_i\| = 1$), otherwise the evolution matrix A_i is not a proper Householder matrix and its eigenvalues can lie outside the unit circle leading to instability. This normalization can be seen as a preprocessing step of the keys. Additionally, DeltaNet allows preprocessing of keys and queries with positive functions (similar to linear attention, i.e., $\psi(\cdot) \geq 0$), which would constrain the normalized coefficients to $\tilde{\alpha}_{i,j} \geq 0$. Otherwise, there are no restrictions on the resulting normalized coefficients $\tilde{\alpha}_{i,j}$, placing the output in the full space $\mathbb{R}^{d_v}$. □

B.7 Gated DeltaNet

Lemma 13 (Coefficient Dynamics of Gated DeltaNet)**.** *Gated DeltaNet (Yang et al., 2025a, eq. 10)*[11]

$$s_t = \alpha_t (\mathbb{I} - \beta_t k_t k_t^\top) s_{t-1} + \beta_t k_t v_t, \tag{11}$$

can be written in the form of (2) *by setting* $A_i = \alpha_t \big(\mathbb{I} - \beta_i k_i k_i^\top\big)$, $b_j = \frac{\beta_j}{\sqrt{n}}$, $\phi(\cdot) = Id(\cdot)$, *and* $\eta_i = 1$. *The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, we need to unroll (11) to get an elementwise convolutional representation (see Sieber et al. (2024); Dao & Gu (2024)):

$$s_i = \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \alpha_t (\mathbb{I} - \beta_t k_t k_t^\top) \right) \beta_j k_j v_j,$$

before computing the output by multiplying the queries:

$$o_i = q_i^\top \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \alpha_t (\mathbb{I} - \beta_t k_t k_t^\top) \right) \beta_j k_j v_j.$$

Note that Yang et al. (2025a) implicitly assumes that the keys are scaled by $\frac{1}{\sqrt{n}}$,[12] and rearranging above equation yields

$$y_i = \sum_{j=1}^{i} q_i^\top \left(\prod_{t=j+1}^{i} \alpha_t (\mathbb{I} - \beta_t k_t k_t^\top) \right) \frac{\beta_j}{\sqrt{n}} k_j v_j.$$

[10] The *writing strength* is computed as $\beta_i = \sigma(W_\beta)$, with $\sigma(\cdot)$ the sigmoid function. It can be multiplied by 2 to allow negative eigenvalues of A_i.

[11] Here, we assume the vector product $k_t v_t$ is well-defined. This is explicit in Yang et al. (2025a), but only needed if recurrences are computed, which is not the case in this paper. Therefore, assume here that $k_t v_t$ computes $k_t v_t^\top$ and that the product $q_i^\top s_i$ is transposed after summation. Then, this is the equivalent setting to (Yang et al., 2025a, eq. 10) but with all quantities transposed.

[12] See codebase: `https://github.com/fla-org/flash-linear-attention`.

We identify the following parameters by comparing to (2)

$$\alpha_{i,j} = q_i^\top \left(\prod_{t=j+1}^{i} \alpha_t (\mathbb{I} - \beta_t k_t k_t^\top) \right) \frac{\beta_j}{\sqrt{n}} k_j, \quad \eta_i = 1.$$

Then, the parameter choices $A_i = \alpha_i(\mathbb{I} - \beta_i k_i k_i^\top)$,[13] $b_j = \frac{\beta_j}{\sqrt{n}}$,[14] and $\phi(\cdot) = \mathrm{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

As discussed in DeltaNet, the keys k_i need to be normalized (i.e. $\|k_i\| = 1$), otherwise the evolution matrix A_i is not a proper Householder matrix and its eigenvalues can lie outside the unit circle leading to instability. This normalization can be seen as a preprocessing step of the keys. The resulting normalized coefficients $\tilde{\alpha}_{i,j}$ are unrestricted in sign and magnitude, placing the output in the full space $\mathbb{R}^{d_v}$. □

B.8 mLSTM

Lemma 14 (Coefficient Dynamics of mLSTM)**.** *mLSTM (Beck et al., 2024, eq. 19 - 27)*[15]

$$s_t = \exp(f_t) s_{t-1} + \exp(i_t) k_t v_t, \tag{12a}$$

$$y_t = \frac{o_t}{\eta_t} q_t^\top s_t \tag{12b}$$

can be written in the form pf (2) *by setting* $A_i = \exp(f_t)$, $b_j = \frac{\exp(i_t)}{\sqrt{n}}$, $\phi(\cdot) = Id(\cdot)$, *and* $\eta_i = \frac{\eta_i}{o_i}$. *The resulting normalized coefficients impose no additional constraints on the output.*

Proof. First, we need to unroll (12a) to get an elementwise convolutional representation (see Sieber et al. (2024); Dao & Gu (2024)):

$$s_i = \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \exp(f_t) \right) \exp(i_j) k_j v_j,$$

before computing the output as

$$y_i = \frac{o_i}{\eta_i} q_i^\top \sum_{j=1}^{i} \left(\prod_{t=j+1}^{i} \exp(f_t) \right) \exp(i_j) k_j v_j.$$

Note that Beck et al. (2024, eq. 23) assumes that the keys are scaled by $\frac{1}{\sqrt{n}}$, and rearranging above equation yields

$$y_i = \sum_{j=1}^{i} \frac{o_i}{\eta_i} q_i^\top \left(\prod_{t=j+1}^{i} \exp(f_t) \right) \frac{\exp(i_j)}{\sqrt{n}} k_j v_j.$$

We identify the following parameters by comparing to (2)

$$\alpha_{i,j} = q_i^\top \left(\prod_{t=j+1}^{i} \exp(f_t) \right) \frac{\exp(i_j)}{\sqrt{n}} k_j, \quad \eta_i = \frac{\eta_i}{o_i}.$$

Then, the parameter choices $A_i = \exp(f_i)$, $b_j = \frac{\exp(i_j)}{\sqrt{n}}$, and $\phi(\cdot) = \mathrm{Id}(\cdot)$ ensure the coefficients $\alpha_{i,j}$ align with (4).

[13] α_i is computed equivalently to $\exp(-\Delta_i A)$ in Mamba-2.

[14] The *writing strength* is computed as $\beta_i = \sigma(W_\beta)$, with $\sigma(\cdot)$ the sigmoid function. It can be multiplied by 2 to allow negative eigenvalues of A_i.

[15] Here, we assume the vector product $k_t v_t$ is well-defined. This is explicit in Beck et al. (2024), but only needed if recurrences are computed, which is not the case in this paper. Therefore, assume here that $k_t v_t$ computes $k_t v_t^\top$ and that the product $q_i^\top s_i$ is transposed after summation. Then, this is the equivalent setting to (Beck et al., 2024, eq. 19) but with all quantities transposed.

Note that the forget and input gates f_i, i_i are computed by linear input projections and that $\exp(f_i)$, $\exp(i_i)$ need to be regularized for numerical stability, since both can grow exponentially fast. In mLSTM this is achieved by multiplying $\exp(m_{i-1} - m_i)$ and $\exp(m_i)$ to the forget and input gates, respectively, where m_i is computed via another recurrence (for all details see Beck et al. (2024)). The resulting normalized coefficients $\tilde{\alpha}_{i,j}$ are unrestricted in sign and magnitude, placing the output in the full space $\mathbb{R}^{d_v}$. □

C Proofs

This appendix is split into two parts: Section C.1 provides the proofs of all lemmas in the main text and Section C.2 provides the proofs of all corollaries in the main text. For convenience, the lemmas and corollaries are copied before the proofs.

C.1 Lemmas

C.1.1 Lemma 1

Lemma 1. *A recurrent formulation of* (3) *with finite memory (state) exists if the readout map* $\phi(\cdot) : \mathbb{R} \to \mathbb{R}$ *is a polynomial of finite degree* P*. The requisite memory (state) then expands to* $\mathbb{R}^{n^P \times d_v}$*.*

Proof. Let the readout map be a general polynomial of degree P

$$\phi(x) = \sum_{p=0}^{P} c_p x^p.$$

Given the dot product $x = q_i^\top h_{i,j}$, we can rewrite the polynomial evaluation using the property that the p-th power of an inner product is equivalent to the inner product of the p-th tensor powers

$$(q_i^\top h_{i,j})^p = (q_i^{\otimes p})^\top h_{i,j}^{\otimes p}.$$

Substituting this expansion into the linear combination (2) yields

$$y_i = \sum_{j=1}^{i} \frac{\phi(q_i^\top h_{i,j})}{\eta_i} v_j = \frac{1}{\eta_i} \sum_{j=1}^{i} \left(\sum_{p=0}^{P} c_p (q_i^{\otimes p})^\top h_{i,j}^{\otimes p} \right) v_j.$$

Because the tensor power of the query $(q_i^{\otimes p})^\top$ does not depend on the summation index j, we can pull it outside the inner sum. Rearranging the terms to isolate the memory state gives

$$y_i = \frac{1}{\eta_i} \sum_{p=0}^{P} c_p (q_i^{\otimes p})^\top \left(\sum_{j=1}^{i} h_{i,j}^{\otimes p} v_j^\top \right).$$

We can now define a separate recurrent memory state for each polynomial degree p, i.e.,

$$S_i^{(p)} = \sum_{j=1}^{i} h_{i,j}^{\otimes p} v_j^\top \in \mathbb{R}^{n^p \times d_v}.$$

To prove this is a valid recurrent formulation, we must show that $S_i^{(p)}$ can be updated using only $S_{i-1}^{(p)}$ and the inputs at time i. Using the original coefficient dynamics where $h_{i,j} = A_i h_{i-1,j}$ for $i > j$, we express the tensor power as

$$h_{i,j}^{\otimes p} = (A_i h_{i-1,j})^{\otimes p} = A_i^{\otimes p} h_{i-1,j}^{\otimes p}.$$

Applying this to our state definition yields the recurrent update rule

$$S_i^{(p)} = A_i^{\otimes p} S_{i-1}^{(p)} + b_i^p k_i^{\otimes p} v_i^\top,$$

and the final output at step i is computed by projecting the augmented queries against these states

$$y_i = \frac{1}{\eta_i} \sum_{p=0}^{P} c_p (q_i^{\otimes p})^\top S_i^{(p)}.$$

Thus, a finite memory recurrent formulation exists for any polynomial map, though the state dimension grows exponentially with the polynomial degree P. □

Remark 2. *The polynomial definition of the readout map ϕ also covers the case where $\phi(\cdot) = \exp(\cdot)$ – the softmax attention readout map – is approximated by a Taylor series.*

C.1.2 Lemma 2

Lemma 2. *Let $\phi : \mathbb{R} \to \mathbb{R}$ be the readout map with connected zero-level set $\mathcal{Z} = \{z \in \mathbb{R} \mid \phi(z) = 0\}$, and let $z = q_i^\top h_{i,j}$, with T_q, $T_{h,i}$ defined in (4). Let $\mathcal{T}$ be the set of linear transformations that map to the zero-level set, i.e., $\mathcal{T} = \{(T_q, T_{h,i}) \mid q_i^\top h_{i,j} \in \mathcal{Z}$ s.t. $q_i = T_q x_i$, $h_{i,j} = T_{h,i} x_j\}$ for any input pairs $x_i, x_j \in \mathbb{R}^d$. The measure $|\mathcal{T}|$ is then proportional to the zero-level set measure $|\mathcal{Z}|$, i.e., $|\mathcal{T}| = c_\phi |\mathcal{Z}|$, where $c_\phi > 0$ depends on $\phi(\cdot)$ and represents the mean geometric density of these transformations.*

Proof. Given that the set $\mathcal{Z}$ is connected on $\mathbb{R}$, the set is defined by the interval $\mathcal{Z} = [\underline{z}, \overline{z}] \subset \mathbb{R}$. Then, since $T_q, T_{h,i} \in \mathbb{R}^{n \times d}$, we consider the Euclidean space E^{2nd} of pairs $(T_q, T_{h,i})$, equipped with the Lebesgue measure $\lambda(E)$. We define the map

$$F : E \to \mathbb{R}, \qquad F(T_q, T_{h,i}) = (T_q x_i)^\top (T_{h,i} x_j),$$

which is smooth and its derivative $\nabla F(T_q, T_{h,i})$ is non-zero for $x_i \neq 0$ and $x_j \neq 0$; we treat the trivial case of either $x_i = 0$ or $x_j = 0$ at the end of the proof.

Then, for each $z \in [\underline{z}, \overline{z}]$, we construct the subset $\mathcal{T}_z$ as a fiber[16] of F, i.e.,

$$\mathcal{T}_z := F^{-1}(\{z\}) = \{(T_q, T_{h,i}) \in E \mid (T_q x_i)^\top (T_{h,i} x_j) = z\}.$$

Therefore, the set of all linear transformations that achieve a value in $\mathcal{Z}$ is

$$\mathcal{T} = F^{-1}(\mathcal{Z}) = \bigcup_{z \in \mathcal{Z}} \mathcal{T}_z.$$

To prove the lemma, we need to compute the Lebesgue measure of $\mathcal{T}$[17], i.e., $|\mathcal{T}| = \int_E \mathbf{1}_{F^{-1}(\mathcal{Z})} \lambda(t)$, where $t = (T_q, T_{h,i})$ and $\mathbf{1}_{F^{-1}(\mathcal{Z})}$ is the indicator function selecting all $t \in \mathcal{T}$. Since $x_i, x_j \neq 0$, F is smooth and $\nabla F \neq 0$ on E, we can apply the coarea formula (Federer, 1959), i.e.,

$$\int_E \mathbf{1}_{F^{-1}(\mathcal{Z})}(t) \lambda(t) = \int_{\mathcal{Z}} \left(\int_{F^{-1}(z)} \frac{1}{\|\nabla F(t)\|} dH^{2nd-1}(t) \right) dz$$

where H^{2nd-1} is the $(2nd-1)$-dimensional Hausdorff measure and $\|\nabla F(t)\|$ is the Euclidean norm of the gradient of F at t. We then define

$$g(z) := \int_{F^{-1}(z)} \frac{1}{|\nabla F(t)|} \, d\mathcal{H}^{2nd-1}(t),$$

which denotes the size of each subset (fiber) $\mathcal{T}_z$. Therefore, the measure of $\mathcal{T}$ is

$$|\mathcal{T}| = \int_{\mathcal{Z}} g(z) dz,$$

[16] The fiber $f^{-1}(y)$ of a function f is the preimage of the singleton y.
[17] Intuitively, the volume of the set $\mathcal{T}$.

and by choosing the constant c_ϕ as

$$c_\phi = \frac{1}{|\mathcal{Z}|}\int_{\mathcal{Z}} g(z)dz,$$

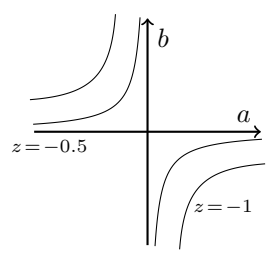
gives the desired relation $|\mathcal{T}| = c_\phi|\mathcal{Z}|$.

In case $x_i, x_j = 0$, the dot product $q_i^\top h_{i,j}$ is trivially zero, thus $F(t) = 0, \forall t = (T_q, T_{h,i}) \in E$. In this case, $|\mathcal{T}| = 0$, if $0 \notin \mathcal{Z}$, i.e., empty because $z = 0$ is not contained in the zero-level set, or $|\mathcal{T}| = |E|$, i.e., the measure of the Euclidean space E. □

Remark 3. *To intuitively understand Lemma 2 and its proof, consider the simplest case: $q_i = T_q x_i = a x_i \in \mathbb{R}$, $h_{i,j} = T_{h,i} x_j = b x_j \in \mathbb{R}$, thus all variables are scalars. Additionally, consider a readout function $\phi(\cdot)$ with zero-level set $\mathcal{Z} = [-1, 0]$. In this setting, we want to find all pairs $t = (a, b)$ such that $ab(x_i x_j) = z$ and we define $X = x_i x_j \neq 0$. Then, all t that achieve z in the zero-level set, need to lie on the hyperbola*

$$a \cdot b = \frac{z}{X}.$$

This hyperbola exactly describes the set $\mathcal{T}_z$ in the proof and there are infinitely many combinations (a, b) that lie in this set. To get to $\mathcal{T}$, we need to integrate over the zero-level set $\mathcal{Z}$ to account for all possible z. These sets (hyperbolas) are also visualized in the figure below.



C.1.3 Lemma 3

Lemma 3. *The coefficients $\alpha_{i,j}$ in (4) have positional information if and only if $A_i \neq \mathbb{I}_n, \forall i$.*

Proof. Consider the definition of the coefficients ((4)):

$$\alpha_{i,j} = \phi\left(q_i^\top \left(\prod_{t=j+1}^{i} A_t\right) b_j k_j\right).$$

We first consider the case $A_t = \mathbb{I}_n, \forall t$, then

$$\alpha_{i,j} = \phi\left(q_i^\top b_j k_j\right).$$

Assuming the scaling factors are computed from the input $b_j = f(x_j)$, the factors b_j contain no positional information. Specifically, because b_j is computed from the same input x_j as k_j, we can rewrite the key using a change of variable $\tilde{k}_j = b_j k_j = g(x_j)$, i.e., the new key is computed from a single input x_j, albeit in a nonlinear fashion. Therefore, we consider the simplified coefficients

$$\bar{\alpha}_{i,j} = \phi\left(q_i^\top \tilde{k}_j\right).$$

Given two keys $\tilde{k}_j, \tilde{k}_{\bar{j}}$ computed from the same input x_j (i.e., $\tilde{k}_j = \tilde{k}_{\bar{j}}$), the coefficients for these two keys are equivalent $\bar{\alpha}_{i,j} = \bar{\alpha}_{i,\bar{j}}$ and thus the coefficients have no positional information, i.e., the coefficients cannot be discriminated depending on the index j.

Now, consider the case $A_t \neq \mathbb{I}_n, \forall t$, then

$$\alpha_{i,j} = \phi\left(q_i^\top \left(\prod_{t=j+1}^{i} A_t\right) b_j k_j\right).$$

Using the same simplification for b_j as above and let $A_{i,j} = \prod_{t=j+1}^{i} A_t$, we get

$$\bar{\alpha}_{i,j} = \phi\left(q_i^\top A_{i,j}\tilde{k}_j\right).$$

The evolution $A_{i,j}$ contains information about the (time) index evolution from j to i, since it defines a map $f : \mathbb{R}^n \to \mathbb{R}^n$ that maps any vector $a_j \in \mathbb{R}^n$ to $a_i = A_i \cdot \dots \cdot A_{j+1}a_j$, i.e., describes the evolution of a_j from (time) index j to i. Consider again two keys $\tilde{k}_j$, $\tilde{k}_{\bar{j}}$ computed from the same input x_j (i.e., $\tilde{k}_j = \tilde{k}_{\bar{j}}$), then their respective coefficients are not equivalent due to $A_{i,j}$:

$$\bar{\alpha}_{i,j} = \phi\left(q_i^\top A_{i,j}\tilde{k}_j\right) \neq \phi\left(q_i^\top A_{i,\bar{j}}\tilde{k}_{\bar{j}}\right) = \bar{\alpha}_{i,\bar{j}}.$$

Specifically, if $\bar{j} > j$, then the following relation holds

$$A_{i,j} = \prod_{t=j+1}^{i} A_t = \prod_{t=\bar{j}+1}^{i} A_t \prod_{t=j+1}^{\bar{j}} A_t = A_{i,\bar{j}}A_{\bar{j},j},$$

and thus there exists a linear relation between $A_{i,j}$ and $A_{i,\bar{j}}$. Therefore, the coefficients have positional information, quantified by $A_{\bar{j},j}$. □

Remark 4. *The proof of Lemma 3 relies strictly on the assumption that the scaling factor b_j is a function exclusively of the input vector x_j (i.e., $b_j = f(x_j)$) and contains no absolute positional information itself. However, if an architecture parameterizes b_j to explicitly depend on the time index j – such as $b_j = f(j, x_j)$ – this introduces an alternative pathway for positional awareness. In such a scenario, even if the evolution matrix is the identity ($A_t = \mathbb{I}_n$), the coefficients $\alpha_{i,j}$ would retain positional information. This is because two identical inputs appearing at different sequence positions ($x_j = x_{\bar{j}}$ with $j \neq \bar{j}$) would yield distinct scaling factors ($b_j \neq b_{\bar{j}}$), allowing the model to discriminate between them. As an example consider preprocessing the inputs by a short convolution (see e.g. Gu & Dao (2023)), i.e., $\{\bar{x}_j\}_{j=1}^{i} = \Phi \star \{x_j\}_{j=1}^{i}$, where Φ denotes the convolution weights. In this case, $\bar{x}_j$ contains scaled copies of previous inputs, depending on the convolution dimension, e.g., $\bar{x}_j = \Phi_0 x_j + \Phi_1 x_{j-1} + \Phi_2 x_{j-2} + \Phi_3 x_{j-3}$. Therefore, preprocessing the inputs with a convolution can offer another way to embed positional information in the coefficients $\alpha_{i,j}$. However, note that this "moving averaging" of inputs might not work for certain repeating input signals.*

C.1.4 Proof of Lemma 4

Lemma 4. *Imposing any of the following structures on A_i in (3), results in the corresponding allowed transformations:*

a) $A_i = \lambda_i \mathbb{I}_n, \lambda_i \in \mathbb{R}, |\lambda_i| \leq 1$; allows scaling of keys (including flipping, if $\lambda_i < 0$) uniformly along all dimensions n,

b) $A_i = diag(\lambda_i), \lambda_i \in \mathbb{R}^n, |\lambda_i^{(r)}| \leq 1$ where $\lambda^{(r)}$ denotes the r-th entry of λ_i; allows scaling of keys (including flipping, if $\lambda_i^{(r)} < 0$) separately along dimensions n,

c) $A_i = \mathbb{I}_n - \beta_i \lambda_i \lambda_i^\top, \beta_i \in [0, 2], \lambda_i \in \mathbb{R}^n$, (Householder matrix); allows scaling and specific rotations of keys.

Proof. All three items follow by a direct computation of A_i's action on an arbitrary key vector $k_j \in \mathbb{R}^n$ and by inspecting eigenvalues/eigenvectors.

(a) Uniform scaling. If $A_i = \lambda_i \mathbb{I}_n$ then for any key k_j we have

$$A_i k_j = \lambda_i \mathbb{I}_n k_j = \lambda_i k_j,$$

so A_i scales k_j uniformly in every coordinate by the scalar λ_i. If $|\lambda_i| \leq 1$ the operator is a contraction; if $\lambda_i < 0$ it also flips the sign of every component (a global reflection through the origin).

(b) Coordinate-wise scaling. If $A_i = \mathrm{diag}(\lambda_i^{(1)}, \ldots, \lambda_i^{(n)})$, then

$$(A_i k_j)^{(r)} = \lambda_i^{(r)} k_j^{(r)}, \qquad r = 1, \ldots, n.$$

Thus each coordinate is scaled independently by $\lambda_i^{(r)}$. The constraints $|\lambda_i^{(r)}| \leq 1$ render the operation a contraction in each coordinate; negative entries produce sign flips on the corresponding coordinate only.

(c) Rank-one update / Householder-type transform. Given $A_i = \mathbb{I}_n - \beta_i \lambda_i \lambda_i^\top$, we decompose any $k_j \in \mathbb{R}^n$ into components parallel and orthogonal to λ_i:

$$k_j = k_j^{\parallel} + k_j^{\perp}, \qquad k_j^{\parallel} = \frac{\lambda_i^\top k_j}{\|\lambda_i\|^2} \lambda_i, \quad \lambda_i^\top k_j^{\perp} = 0,$$

where $\|\cdot\|$ as the Euclidean in $\mathbb{R}^n$. Then

$$A_i k_j = (\mathbb{I}_n - \beta_i \lambda_i \lambda_i^\top)(k_j^{\parallel} + k_j^{\perp}) = (1 - \beta_i \|\lambda_i\|^2) k_j^{\parallel} + k_j^{\perp}.$$

Hence $k_j^{\perp}$ (every vector orthogonal to λ_i) is an eigenvector with eigenvalue 1, and the direction λ_i itself is an eigenvector with eigenvalue $1 - \beta_i \|\lambda_i\|^2$. Therefore, A_i acts by scaling the component along λ_i by the factor $1 - \beta_i \|\lambda_i\|^2$, while leaving the orthogonal complement unchanged.

Special cases and remarks:

- If one chooses $\beta_i = 2/\|\lambda_i\|^2$ (and $\lambda_i \neq 0$), then $1 - \beta_i \|\lambda_i\|^2 = -1$ and

  $$A_i = \mathbb{I}_n - \frac{2}{\|\lambda_i\|^2} \lambda_i \lambda_i^\top$$

  is a classical Householder reflection (an orthogonal matrix with determinant -1) that reflects across the hyperplane orthogonal to λ_i.

- For general $\beta_i \in [0, 2]$ and $\|\lambda_i\|$ normalized (or when $\beta_i \|\lambda_i\|^2 \in [0, 2]$), the eigenvalue $1 - \beta_i \|\lambda_i\|^2$ lies in $[-1, 1]$, so the operator is a contraction on the λ_i-direction and may flip the sign (reflection), if the eigenvalue is negative.

- Although a single rank-one transform of this form does not produce an arbitrary rotation, compositions of Householder reflections (each of which is of the form $\mathbb{I}_n - 2\lambda\lambda^\top$ with unit $\lambda \in \mathbb{R}^n$ can generate any orthogonal matrix (rotations and reflections). Hence, by composing appropriate special cases of the matrices in (c), orthogonal rotations can be realized.

Combining these elementary computations proves that each structural constraint on A_i yields the stated class of allowed transformations. □

Remark 5. *In Lemma 4 and the proof above, we make no assumption on the magnitude of λ_i in a Householder-type evolution matrix. However, as shown in the proof, assuming unit λ_i is necessary to define all transformations. This is the reason why the keys k_j and queries q_i are L2 normalized in Yang et al. (2024; 2025a) for the computation of A_i.*

C.1.5 Proof of Lemma 5

Lemma 5. *Consider two i.i.d. normally distributed random vectors $x_i, x_j \sim \mathcal{N}(0, \Sigma)$. Then, the dot product $q_i^\top h_{i,j}$ with $q_i, h_{i,j}$ defined in (4), has zero-mean and variance $Var(q_i^\top h_{i,j}) = tr(\Sigma_q \Sigma_h)$ with $\Sigma_q = T_q \Sigma T_q^\top$, $\Sigma_h = T_{h,i} \Sigma T_{h,i}^\top$. Assuming $\Sigma = \sigma \mathbb{I}_d$, both Σ_q and Σ_h are positive semi-definite and the variance of the dot product scales as $Var(q_i^\top h_{i,j}) = \mathcal{O}(n)$.*

Proof. We proof the lemma in three parts, first zero mean, then the variance relation, and finally the scaling under the isotropy assumption.

Mean. Since x_i and x_j are independent and zero-mean,

$$\mathbb{E}[q_i^\top h_{i,j}] = \mathbb{E}[(T_q x_i)^\top (T_{h,i} x_j)] = \mathbb{E}_{x_i}\mathbb{E}_{x_j}[x_i^\top T_q^\top T_{h,i} x_j] = 0,$$

hence the mean is zero.

Variance. Since the mean is zero,

$$\begin{aligned}\mathrm{Var}(q_i^\top h_{i,j}) &= \mathbb{E}\big[(q_i^\top h_{i,j})^2\big] \\ &= \mathbb{E}\big[(T_q x_i)^\top (T_{h,i} x_j)(T_{h,i} x_j)^\top (T_q x_i)\big]\,.\end{aligned}$$

Conditioning on x_i and taking expectation over x_j first,

$$\begin{aligned}\mathbb{E}_{x_j}\big[(T_{h,i} x_j)(T_{h,i} x_j)^\top\big] &= T_{h,i}\,\mathbb{E}[x_j x_j^\top]\,T_{h,i}^\top \\ &= T_{h,i}\Sigma T_{h,i}^\top := \Sigma_h,\end{aligned}$$

and due to symmetry $\mathbb{E}_{x_i}\big[(T_q x_i)(T_q x_i)^\top\big] = T_q \Sigma T_q^\top = \Sigma_q$. Therefore,

$$\begin{aligned}\mathbb{E}\big[(q_i^\top h_{i,j})^2\big] &= \mathbb{E}_{x_i}\big[(T_q x_i)^\top \Sigma_h (T_q x_i)\big] \\ &= \mathbb{E}_{x_i}\big[\mathrm{tr}\big(\Sigma_h (T_q x_i)(T_q x_i)^\top\big)\big] \\ &= \mathrm{tr}\big(\Sigma_q \Sigma_h\big),\end{aligned}$$

due to the previous relations and the cyclic property of the trace operator.

Isotropy. If $\Sigma = \sigma \mathbb{I}_d$, then

$$\Sigma_q = \sigma T_q T_q^\top, \qquad \Sigma_h = \sigma T_h T_h^\top,$$

which are symmetric positive semidefinite by definition of matrix outer products (Friedberg et al., 1997). Therefore,

$$\begin{aligned}\mathrm{Var}(q_i^\top h_{i,j}) &= \mathrm{tr}\big(\Sigma_q \Sigma_h\big) \\ &= \sigma^2 \mathrm{tr}\big(T_q T_q^\top T_{h,i} T_{h,i}^\top\big) \\ &= \sigma^2 \|T_q^\top T_{h,i}\|_F^2,\end{aligned}$$

where we used the cyclic property of the trace and the identity $\|M\|_F = \mathrm{tr}(MM^\top)$ (Friedberg et al., 1997), with $\|\cdot\|_F$ denoting the Frobenius norm. To prove the scaling statement, we assume T_q, $T_{h,i}$ are properly normalized weight matrices (i.e. the matrices have entries $\mathcal{O}(1/\sqrt{n})$, such that the output variance remains $\mathcal{O}(1)$). This is assumption is common in practice. Therefore, we have $\|T_q\|_F = \mathcal{O}(\sqrt{n})$, $\|T_{h,i}\|_F = \mathcal{O}(\sqrt{n})$ and thus $\|T_q^\top T_{h,i}\|_F^2 = \mathcal{O}(n)$ and $\mathrm{Var}(q_i^\top h_{i,j}) = \mathcal{O}(n)$. Importantly, given common normalization choices for T_q, $T_{h,i}$, which keep per-coordinate variances to $\mathcal{O}(1)$, the dot product variance grows linearly in n. □

C.1.6 Proof of Lemma 6

Lemma 6. *Consider a fixed index j and let $s_i := q_i^\top h_{i,j}$, $\alpha_i = \phi(s_i)$ in (3). Let $\phi : \mathbb{R} \to [0,\infty)$ be nondecreasing and unbounded, and let $g : [0,\infty) \to (0,\infty)$ be an increasing and unbounded comparison function, such that $L := \limsup_{i\to\infty} \frac{\phi(s_i)}{g(s_i)} < \infty$. If η_i is chosen such that $\liminf_{i\to\infty} \frac{\eta_i}{g(s_i)} := m > 0$, then the normalized coefficients are bounded: $\limsup_{i\to\infty} \frac{\alpha_i}{\eta_i} \le \frac{L}{m}$.*

Proof. Fix $\varepsilon \in \big(0, \min\{1, m\}\big)$. By the definition of the limit superior, there exists $N_1 \in \mathbb{N}$ such that for all $i \ge N_1$,

$$\frac{\phi(s_i)}{g(s_i)} \;\le\; L + \varepsilon.$$

By the definition of the limit inferior, there exists $N_2 \in \mathbb{N}$ such that for all $i \geq N_2$,

$$\frac{\eta_i}{g(s_i)} \;\geq\; m - \varepsilon.$$

Hence for all $i \geq N := \max\{N_1, N_2\}$ we have

$$\frac{\alpha_i}{\eta_i} = \frac{\phi(s_i)}{\eta_i} = \frac{\phi(s_i)}{g(s_i)} \cdot \frac{g(s_i)}{\eta_i} \leq \frac{L+\varepsilon}{m-\varepsilon}.$$

Since finitely many initial indices do not affect the limit superior, letting $\varepsilon \to 0$ yields

$$\limsup_{i\to\infty} \frac{\alpha_i}{\eta_i} \leq \frac{L}{m},$$

which proves the lemma. □

Remark 6. *Assuming the normalization factor η_i is a function of the coefficients themselves, i.e., $\eta_i = f(\alpha_{i,j})$, simplifies Lemma 6 significantly, since we only need to analyze $\frac{\alpha_i}{f(\alpha_i)}$ (without comparison function g). Hence, if $f(\cdot)$ grows linearly, $\alpha_i/f(\alpha_i)$ converges to a constant $m > 0$; if $f(\cdot)$ grows superlinearly, $\alpha_i/f(\alpha_i)$ converges to 0; and $f(\cdot)$ grows sublinearly, $\alpha_i/f(\alpha_i)$ blows up.*

C.2 Corollaries

C.2.1 Proof of Corollary 2.1

Corollary 2.1. *Consider the kernel approximation of $\phi(\cdot) : \mathbb{R}^n \to \mathbb{R}^q$ to be $\tilde{\phi}(q_i^\top h_{i,j}) = \psi_q(q_i)^\top \psi_h(h_{i,j})$, with $\psi_q : \mathbb{R}^n \to \mathbb{R}^q$ and $\psi_h : \mathbb{R}^n \to \mathbb{R}^q$, such that it approximates the readout map $\phi(\cdot) \approx \tilde{\phi}(\cdot)$ in (3). Then, the zero-level set of the approximation $\tilde{\phi}(\cdot)$ is the singleton $\mathcal{Z} = \{0\}$ and by Lemma 2 the set of linear transformations $\mathcal{T}$ is small.*

Proof. Given that the kernel approximation is defined as $\tilde{\phi}(q_i^\top h_{i,j}) = \psi_q(q_i)^\top \psi_h(h_{i,j})$, we can rewrite this approximation using a change of variables as

$$\tilde{\phi}(q_i^\top h_{i,j}) = \tilde{q}_i^\top \tilde{h}_i, j, \text{ with } \tilde{q}_i \coloneqq \psi_q(q_i),\ \tilde{h}_{i,j} \coloneqq \psi_h(h_i, j). \tag{13}$$

Note that (13) is equivalent to a dot product with an identity readout map $\mathrm{Id}(\cdot)$, albeit in different dimensions than the untransformed dot product $q_i^\top h_{i,j}$ ($\tilde{q}_i \in \mathbb{R}^q$ compared to $q_i \in \mathbb{R}^n$). Given that the zero-level set of a dot product is the singleton $\{0\}$, the zero-level set of $\tilde{\phi}(\cdot)$ is $\mathcal{Z} = \{0\}$. Hence, $|\mathcal{Z}| = 0$ and the measure of the set of linear transformations reduces to $|\mathcal{T}| = g(\{0\})$ (see proof of Lemma 2 for the exact definitions). Therefore, $\mathcal{T}$ is reduced to a single fiber, i.e., $\mathcal{T} = \{(T_q, T_{h,i}) \mid (T_q x_i)^\top (T_{h,i} x_j) = 0\}$. □

C.2.2 Proof of Corollary 2.2

Corollary 2.2. *Let $y_L \in \mathbb{R}^{d_v}$ be the solution to (2), where $\alpha_{L,j} : \mathbb{R}^n \to \mathbb{R};\ q_L \mapsto \alpha_{L,j}(q_L)$ is defined in (3) with identity readout map and normalization factor, i.e., $\phi(\cdot) = \mathrm{Id}(\cdot)$, $\eta_L = 1$. Consider the set of linearly independent states $\mathcal{H} = \{h_{L,t} \mid c_1 h_{L,1} + \cdots + c_t h_{L,t} = 0 \implies c_1 = \cdots = c_t = 0\}$. Then, a nonzero q_L that achieves $\alpha_{L,j} = 0$, $\forall h_{L,j} \in \mathcal{H}$ exists if and only if $\dim\big(\mathrm{span}\{h_{L,j} \in \mathcal{H}\}\big) < n$. In particular, given a nonzero q_L, the measure $|\mathcal{H}| \leq n-1$.*

Proof. The conditions $\alpha_{L,j} = q_L^\top h_{L,j} = 0,\ \forall h_{L,j} \in \mathcal{H}$ are equivalent to the linear homogeneous system

$$H^\top q_L = 0, \tag{14}$$

where $H \in \mathbb{R}^{n \times |\mathcal{H}|}$ with $|\mathcal{H}|$ the measure of set $\mathcal{H}$,[18] is a matrix whose columns consist of the vectors $h_{L,j} \in \mathcal{H}$. The solution space for q_L is the nullspace of $H^\top$. By the rank–nullity theorem (Friedberg et al., 1997, Theorem 2.3), we get

$$\dim\big(\ker(H^\top)\big) = n - \mathrm{rank}(H^\top) = n - \mathrm{rank}(H).$$

[18] This means the number of vectors $h_{L,j}$ contained in the set.

Hence a nonzero q_L solving (14) exists if and only if $\text{rank}(H) < n$, i.e., if and only if the vectors $h_{L,j} \in \mathcal{H}$ span a proper subspace of $\mathbb{R}^n$.

Consequently, the largest possible number of linearly independent constraints (equivalently, independent $h_{L,j}$) that admit a nonzero solution is $n-1$. Equivalently, at most $n-1$ independent coefficients $\alpha_{L,j}$ can be forced to zero simultaneously, with a nonzero choice of q_L. □

C.2.3 Proof of Corollary 6.1

Corollary 6.1. *Choosing $\eta_i = \sum_{j=1}^{i} \alpha_{i,j}$ in (2) constrains the normalized coefficients to $\tilde{\alpha}_{i,j} = {}^{\alpha_{i,j}}/_{\eta_i} \in [0,1]$ and imposes $\sum_{j=1}^{i} \tilde{\alpha}_{i,j} = 1$.*

Proof. Consider the linear combination ((2))

$$y_i = \sum_{j=1}^{i} \frac{\alpha_{i,j}}{\eta_i} v_j.$$

Then, setting $\eta_i = \sum_{j=1}^{i} \alpha_{i,j}$, results in

$$y_i = \frac{\sum_{j=1}^{i} \alpha_{i,j} v_j}{\sum_{j=1}^{i} \alpha_{i,j}} = \sum_{j=1}^{i} \tilde{\alpha}_{i,j} v_j,$$

with $\tilde{\alpha}_{i,j} = {}^{\alpha_{i,j}}/_{\eta_i}$ as used in the statement. Therefore, the coefficients $\tilde{\alpha}_{i,j}$ trivially sum to 1. Additionally, this restricts the coefficients and specializes the linear combination to a convex combination (if $\tilde{\alpha}_{i,j} \geq 0$) or an affine combination, as discussed in Appendix A. Therefore, the output y_i lies in either the simplex or a hyperplane spanned by the value vectors v_j (Friedberg et al., 1997). □

D Extended Experimental Validation

In this appendix, we provide additional experiments to validate the principles stated in the main text.

D.1 Principle 1

Since Principle 1 and Lemma 1 are existence results, the principle cannot be validated experimentally. However, the principle has profound implications on the efficiency of sequence models as shown in Figure 6. With respect to sequence length, a recurrent implementation has linear complexity, while a non-recurrent implementation has quadratic complexity. Figure 6 shows the computation time of FlashAttention-2 (Dao, 2023) – a non-recurrent implementation – and SSD (Dao & Gu, 2024) – a reccurent implementation – against sequence length.[19]

D.2 Principle 2 and Sub-Principles

Figure 7 adds to Figure 2 in the main text, by adding the memorization task and additionally showing the behavior of the investigated readout maps $\phi(\cdot)$ close to zero, which highlights how the theoretical near-zero sets are computed.

Figure 10 additionally validates Principle 2 by showing performance against the ratio of coefficients $\alpha_{i,j}$ set to large values ($\alpha \in [0.9, 1]$) and coefficients contained in the near-zero set ($|\alpha| \leq 0.001$); since the ratio is small, we plot it on a log-scale. This ratio captures not only a readout map's ability to suppress coefficients, but also its ability to select coefficients close to 1. *The results in Figure 10 suggest that as the ability of a readout map to set coefficients to zero increases, the model becomes more selective (also setting more coefficients to 1), increasing the performance of all three tasks.*

[19]The data for this plot was obtained using `https://github.com/state-spaces/mamba`.

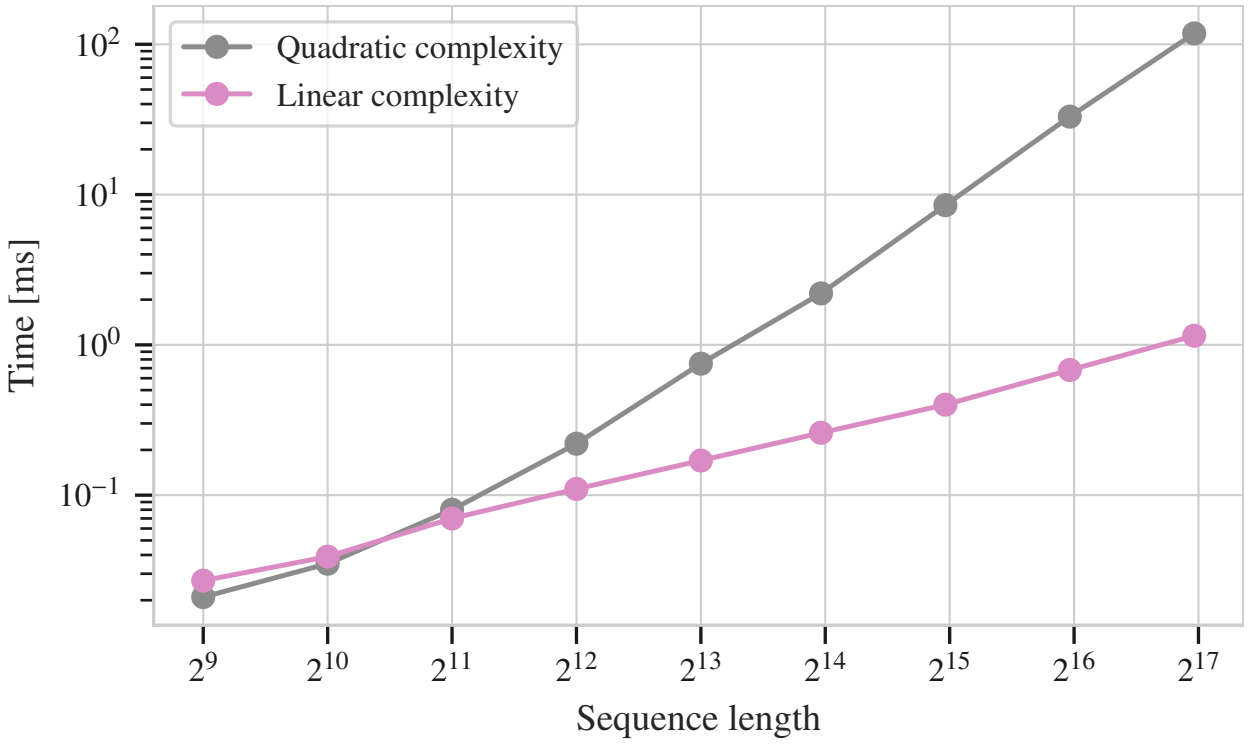


Figure 6: **(Principle 1)** Computation time of a recurrent (magenta) and non-recurrent (gray) implementation against sequence length.

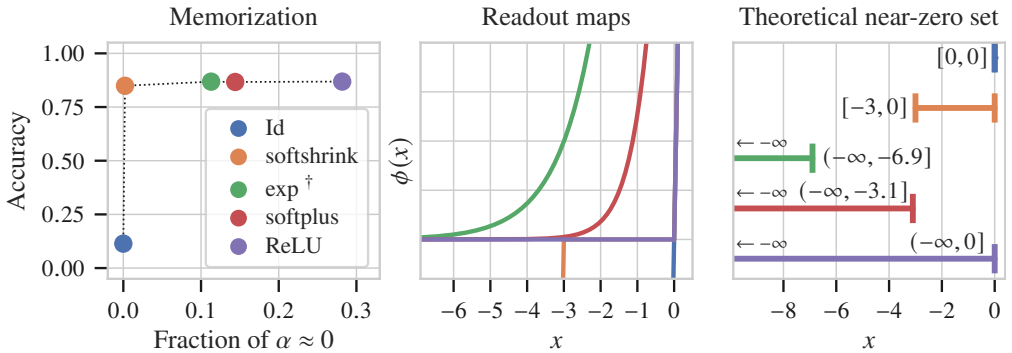


Figure 7: **(Principle 2)** Performance of different readout maps $\phi(\cdot)$ on the memorization task of MAD against the fraction of coefficients with near zero values ($|\alpha| \leq 0.001$; left). The other parameters A_i, b_j, η_i are fixed, thus the setting for † is equivalent to softmax attention. Behavior of the investigated readout maps $\phi(\cdot)$ close to zero (middle) and the theoretical near-zero sets of each readout map ($|\phi(x)| \leq 0.001$; right).

Principle 2.1 To validate Principle 2.1, we ablate four kernel approximation maps $\psi(\cdot) \in \{\mathrm{ReLU}(\cdot), \mathrm{ELU}(\cdot) + 1, \mathrm{softplus}(\cdot), \exp(\cdot)\}$ that approximate the readout map as $\phi(\cdot) \approx \psi(\cdot)\psi(\cdot)$. The other coefficient dynamics parameters are fixed to $A_i = \mathbb{I}_n$, $b_j = {}^1/\sqrt{n}$, and $\eta_i = \sum_j \alpha_{i,j}$. In Figure 8, we report the performance of all approximation maps on the fuzzy in-context recall and selective copying tasks of MAD, against the fraction of coefficients in the near-zero set (equivalent to Figure 2). Additionally, we provide the performance of softmax attention ($\phi(\cdot) = \exp(\cdot)$) and pure linear attention ($\phi(\cdot) = \mathrm{Id}(\cdot)$) as baselines. *As dictated by the principle, the kernel approximations set a smaller fraction of coefficients in the near-zero set – compared to the readout maps in Figure 2 – thus deteriorating performance on both tasks.* However,

note that most kernel approximations ($\psi(\cdot) \in \{\mathrm{ELU}(\cdot)+1, \mathrm{softplus}(\cdot), \exp(\cdot)\}$) are more efficient at setting coefficients close to zero than the identity readout map.

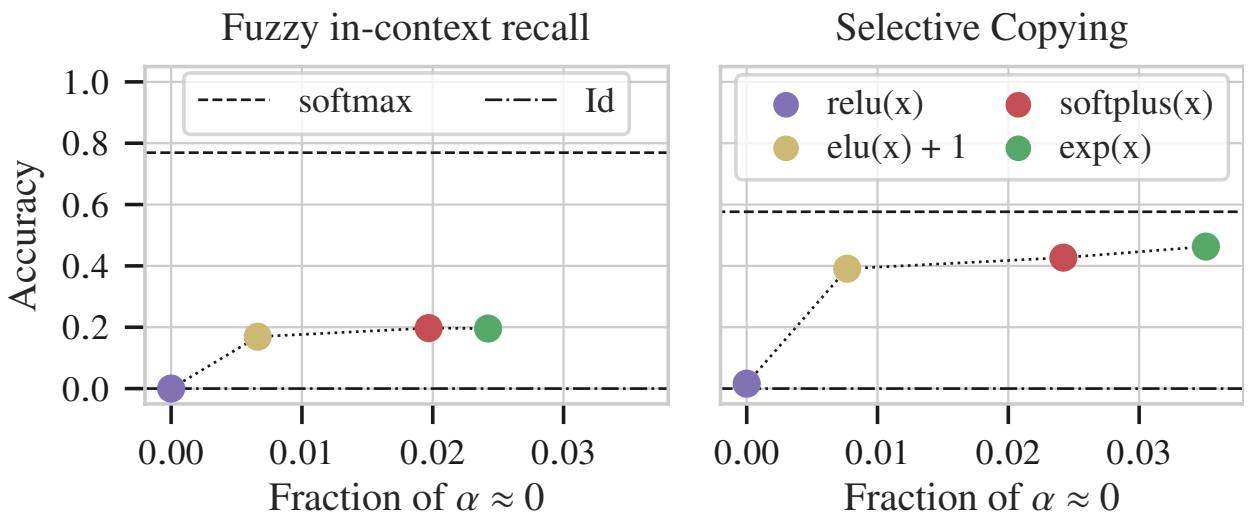


Figure 8: **(Principle 2.1)** Performance of different kernel approximations $\psi(\cdot)$ of the readout map $\phi(\cdot) \approx \psi(\cdot)\psi(\cdot)$ on two MAD tasks against the fraction of coefficients with near zero values ($|\alpha| \leq 0.001$). The other parameters A_i, b_j, η_i are fixed. Dotted lines indicate the performance level of softmax attention ($\phi(\cdot) = \exp(\cdot)$) and pure linear attention ($\phi(\cdot) = \mathrm{Id}(\cdot)$) from Figure 2, as a baseline.

Principle 2.2 To validate Principle 2.2, we investigate the number of coefficients that are set to zero per time index i. Since no coefficient $\alpha_{i,j}$ is exactly set to zero due to numerical precision, we treat $|\alpha_{i,j}| \leq 1e{-}6$ as zero. Using two models trained for Figure 8 (Principle 2.1), i.e., readout approximations $\psi(\cdot) \in \{\mathrm{softplus}(\cdot), \exp(\cdot)\}$, we compute for each time index i and each head, the average number of zero coefficients over a sample input batch of size 64. Since the two models have state dimension $n = 128$ and 16 heads, the effective state dimension[20] is $\tilde{n} = \frac{n}{16} = 8$, therefore we expect no more than $\tilde{n} - 1 = 7$ zero coefficients per time index and head (Principle 2.2). The results are shown in Figure 9 for one example head of each model on both tasks. *The figure shows that the allowed number of zero coefficients is never exceeded and in general longer sequence lengths allow to set more coefficients to zero (compare the two tasks with different sequence lenghts).* This can be explained by the *linear independence* of states $h_{i,j}$ required by Corollary 6.1: for longer sequences, there exist more linearly independent states, which facilitates setting more coefficients to zero.

D.3 Principle 5

To validate Principle 5, we ablate four choices of scaling factors b_j, on four combinations of readout maps $\phi(\cdot)$ and evolution matrices A_i. We choose two scaling factors that adhere to the principle ($b_j = 1/\sqrt{n}$, $b_j = z_j/\sqrt{n}$) and two that do not ($b_j = 1$, $b_j = z_j$), where z_j is input-dependent, i.e., $z_j = \sigma(W_z x_j)$ with $\sigma(\cdot)$ the sigmoid function. For readout maps and evolution matrices, we choose the combinations $(\phi(\cdot), A_i) = \{(\exp(\cdot), \mathbb{I}_n), (\mathrm{ReLU}(\cdot), \mathbb{I}_n), (\mathrm{Id}(\cdot), \alpha_i \mathbb{I}_n), (\mathrm{Id}(\cdot), \mathbb{I}_n - \beta_i k_i k_i^\top)\}$, where α_i is parameterized equivalently to $\exp(-\Delta_i A)$ in Mamba-2 (Dao & Gu, 2024) and β_i is input-dependent, i.e., $\beta = 2\sigma(W_\beta x_i)$ with $\sigma(\cdot)$ the sigmoid function. These models are trained on the fuzzy in-context recall task of MAD and the results are shown in Figure 11. We plot the accuracy on the task, across three different seeds, against the effective head dimension, i.e., $n/\text{nr. of heads}$.[21] *The results in Figure 11 suggest that for scaling factors not adhering to the principle ($b_j = 1$, $b_j = z_j$), the performance depends on the random seed, as we notice large performance fluctuations. Conversely, for scaling factors adhering to the principle ($b_j = 1/\sqrt{n}$, $b_j = z_j/\sqrt{n}$),*

[20] In a multi-head setting, the state dimension n in Principle 2.2 needs to be replaced by the head dimension $\tilde{n} = n/\text{nr. of heads}$.
[21] In a multi-head setting, the state dimension n in $1/\sqrt{n}$ is replaced by the head dimension $\tilde{n} = n/\text{nr. of heads}$.

Figure 9: **(Principle 2.2)** Number of zero coefficients per time index i, for one head of two models $\psi(\cdot) \in \{\text{softplus}(\cdot), \exp(\cdot)\}$ trained on fuzzy in-context recall and selective copying (same models as in Fig. 8).

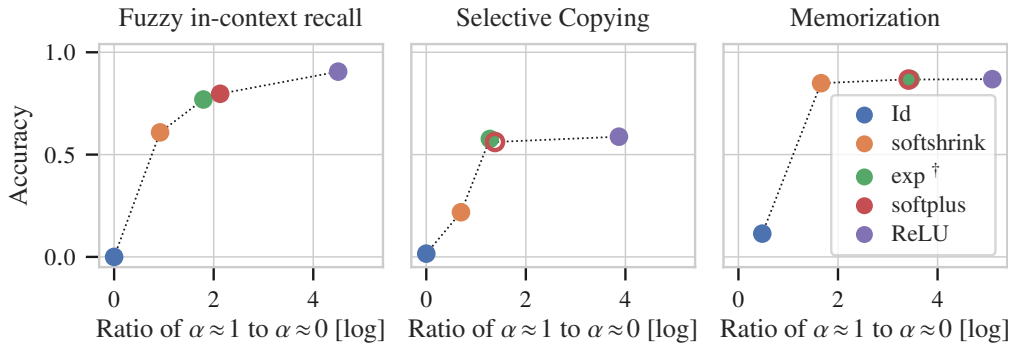


Figure 10: **(Principle 2)** Performance of different readout maps $\phi(\cdot)$ on three MAD tasks with respect to the ratio of coefficients $\alpha_{i,j}$ set to large (≈ 1) and near zero values (≈ 0). The other parameters A_i, b_j, η_i are fixed, thus the setting for † is equivalent to softmax attention.

the performance remains largely constant across seeds. This behavior is expected from Principle 5, since the dot product $q_i^\top h_{i,j}$ has larger variance for b_j not adhering to the principle. Note that the performance fluctuations are largest for readout map $\phi(\cdot) = \exp(\cdot)$. We hypothesize that this is due to the exponential growth of the readout map, as dot products $q_i^\top h_{i,j}$ with large variance are either pushed to zero or large values. This is necessarily less pronounced for the other three combinations, although performance fluctuations are still more pronounced for scaling factors not adhering to the principle. While there are indications that performance fluctuates more as the head dimension increases (see e.g. "scalar" for $b_j = 1$) – which is expected per Principle 5 – this pattern is not consistent. We hypothesize that the head dimension is too small for this pattern to emerge consistently.

We also validate Principle 5 on a more realistic dataset than the previous synthetic one. For this, we train a standard softmax attention model and a standard Mamba-2 model on Wikitext-103 (Merity et al., 2016). For both models, we modify the scaling factor b_j and fix all other parameters to the standard choices of each model. Both models, have the same size but softmax attention is equipped with additional positional embeddings, which explains the higher parameter count in Table 3. The results in Table 3 suggest that there exists a performance gap, yet small, between scaling factors adhering to the principle ($b_j = 1/\sqrt{n}$, $b_j = \Delta_j$)[22] and scaling factors that do not ($b_j = 1$). The finding that softmax attention without scaling of the keys

[22]In Mamba-2, Δ_j adheres to the principle as discussed in Example 2.

Table 3: Perplexity of sequence models with different choices of scaling factor b_j pretrained on Wikitext-103.

	Mamba-2			Softmax Attention	
scaling parameter	$b_j = 1$	$b_j = 1/\sqrt{n}$	$b_j = \Delta_j$	$b_j = 1$	$b_j = 1/\sqrt{n}$
parameter count	110 M	110 M	110 M	120 M	120 M
perplexity	29.23	28.84	28.87	32.69	31.73

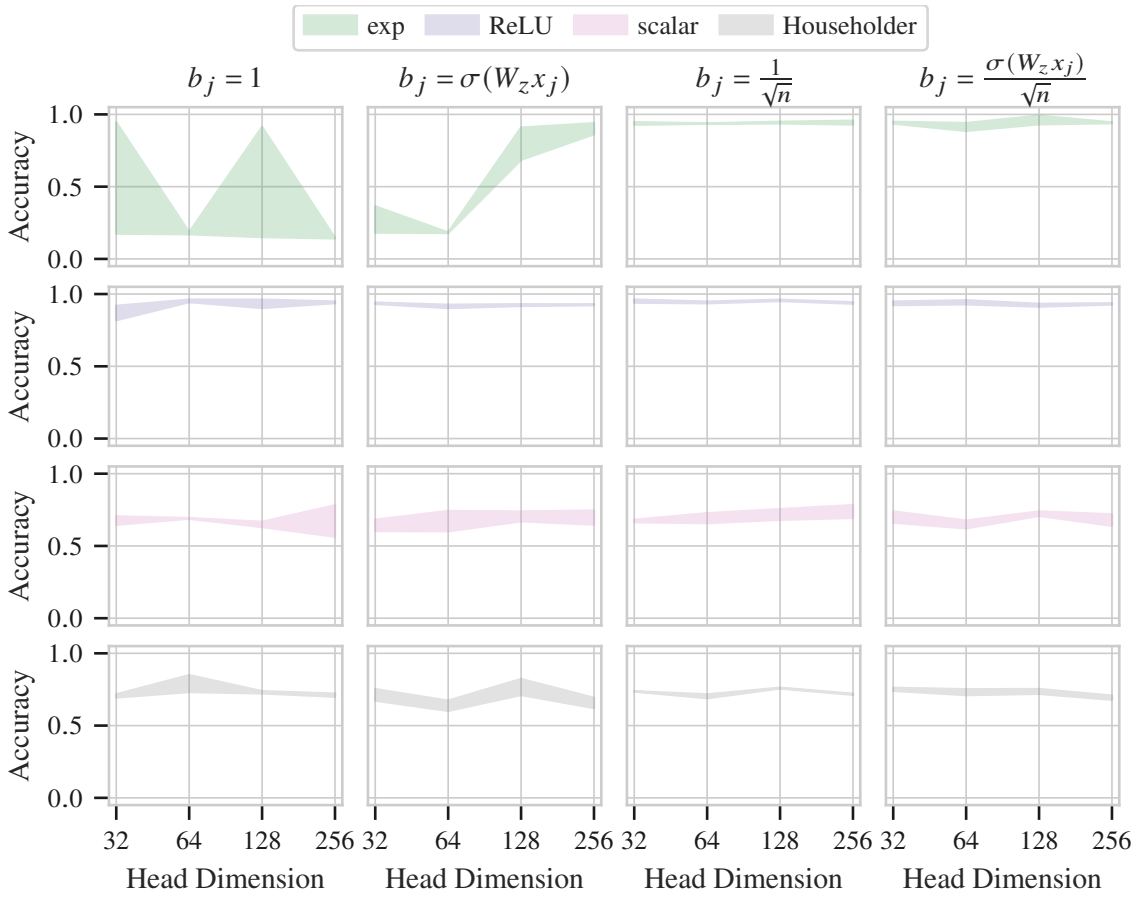


Figure 11: **(Principle 5)** Minimal and maximal performance across three seeds on the fuzzy in-context recall task, for different choices of scaling factor b_j and combinations $(\phi(\cdot), A_i)$. The legend refers to the combinations $(\phi(\cdot), A_i) = \{(\exp(\cdot), \mathbb{I}_n), (\mathrm{ReLU}(\cdot), \mathbb{I}_n), (\mathrm{Id}(\cdot), \alpha_i \mathbb{I}_n), (\mathrm{Id}(\cdot), \mathbb{I}_n - \beta_i k_i k_i^\top)\}$ and each column shows the results for a single b_j.

can perform well in practice, has also been observed in Britz et al. (2017). Finally, note that decoupling the parameters A_i and b_j in Mamba-2 does not hurt performance, with the scaling factor $b_j = 1/\sqrt{n}$ (with Δ_i still used to compute A_i) performing on par with $b_j = \Delta_j$.

D.4 Principle 6

Principle 6 is mainly concerned with training stability related to the evolution matrices A_i and normalization factors η_i. Due to Corollary 6.1, choosing $\eta_i = \sum_j \alpha_{i,j}$ guarantees stable training and has additional implications on the output space of the sequence model (see Appendix A). We hence focus on evolution

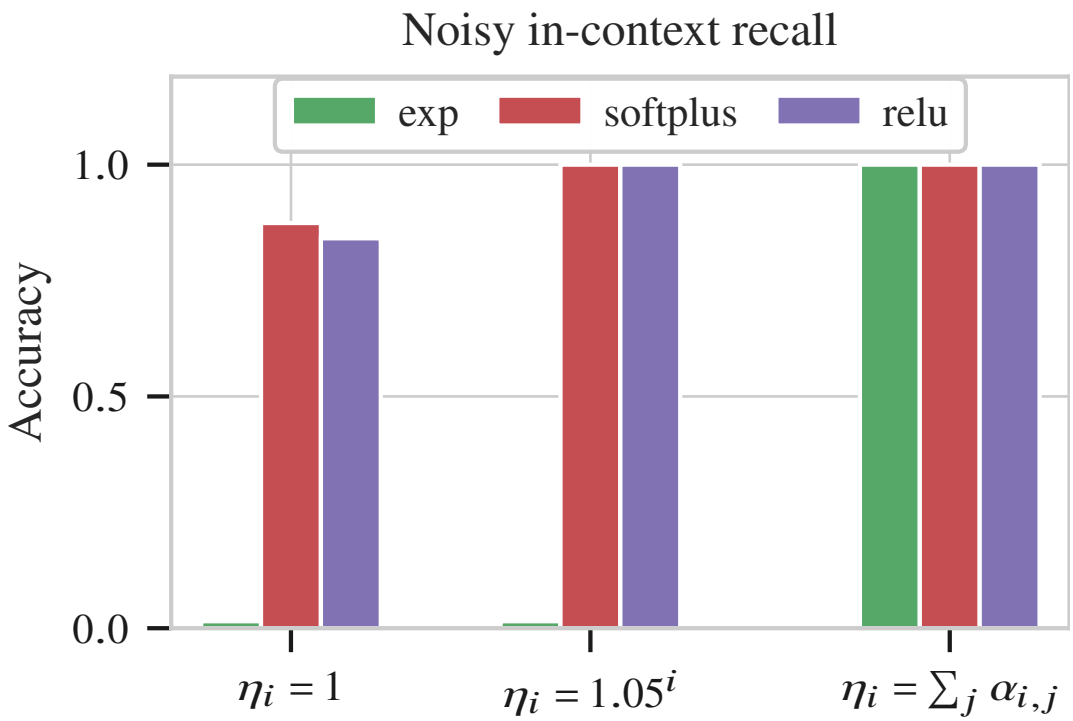


Figure 12: **(Principle 6)** Performance of three normalization factor choices η_i on the noisy in-context recall task, for unstable evolution matrix $A_i = 1.05\,\mathbb{I}_n$, $b_j = 1$, and readout maps $\phi(\cdot) \in \{\exp(\cdot), \text{softplus}(\cdot), \text{ReLU}(\cdot)\}$.

matrices to experimentally validate the principle. It is widely known, that stable A_i (all eigenvalues lie in the unit circle of the complex plane) guarantee numerically stable training, as evidenced by the design of A_i in e.g. S4 (Gu et al., 2022a), LRU (Orvieto et al., 2023), Mamba-2 (Dao & Gu, 2024), GLA (Yang et al., 2023), and DeltaNet (Schlag et al., 2021). To validate the principle, we therefore consider the simple unstable evolution matrix $A_i = 1.05\,\mathbb{I}_n$ and show that with appropriate design of η_i, we are able to solve noisy in-context recall. We fix $b_j = 1$, $A_i = 1.05\,\mathbb{I}_n$ and ablate the readout maps $\phi(\cdot) \in \{\exp(\cdot), \text{softplus}(\cdot), \text{ReLU}(\cdot)\}$ and normalization factors $\eta_i \in \{1, 1.05^i, \sum_j \alpha_{i,j}\}$; results are shown in Figure 12. *As expected by Corollary 6.1, the normalization factor $\eta_i = \sum_j \alpha_{i,j}$ ensures stable training and achieves perfect accuracy.* The results of the other two choices need additional discussion.

First, consider $\eta_i = 1$: for readout map $\exp(\cdot)$, training is unstable and the model fails to learn anything, yet for readout maps $\text{softplus}(\cdot), \text{ReLU}(\cdot)$ training is stable and the model achieves good performance, albeit suboptimal. This is explained by the growth of the readout maps, while $\exp(\cdot)$ grows superlinearly, $\text{softplus}(\cdot), \text{ReLU}(\cdot)$ grow linearly, thus the coefficients $\alpha_{i,j}$ grow faster for $\phi(\cdot) = \exp(\cdot)$ than the other two maps, as the state $h_{i,j}$ grows due to $\prod_{t=j+1}^{i} 1.05\,\mathbb{I}_n$ (see (4)). Since the task has short sequence length (128) and the chosen evolution matrix is mildly unstable (1.05), readout maps with linear growth can still achieve good performance.

Now consider $\eta_i = 1.05^i$: for readout map $\exp(\cdot)$, training is unstable and the model fails to learn anything, yet for readout maps $\text{softplus}(\cdot), \text{ReLU}(\cdot)$ training is stable and the model achieves perfect performance. This is again explained by the growth of the readout maps, when considering the normalized coefficients $\frac{\alpha_{i,j}}{\eta_i}$. For $\phi(\cdot) = \exp(\cdot)$, the coefficients $\alpha_{i,j}$ grow faster than the normalization factors η_i, leading to instability. For the other two readout maps, the coefficients $\alpha_{i,j}$ grow as fast as the normalization factors η_i, thus stabilizing training. Further experimental validation of the principle, is given by the experimental results in Beck et al. (2024), since the model could not have been trained without proper design of the normalization factors η_i.

E Details on Experimental Setup

The experimental results provided in Section 5 and Appendix D are performed on the mechanistic architecture design (MAD) benchmark (Poli et al., 2024). To perform the experiments, we modified the MAD code base[23] and integrated it in our existing code base, which will be released upon publication. For pretraining in Appendix D.3, we used the Wikitext-103 (Merity et al., 2016) dataset. All experiments were run on a cluster equipped with 8 NVIDIA RTX 4090 GPUs.

E.1 Global settings

Wikitext The Wikitext-103 dataset is tokenized using the standard GPT-2 tokenizer (`vocab_size`: 50257) and chunked into blocks of sequence length 1024, before being processed by the model.

MAD Tasks We perform experiments on the following four tasks of MAD: selective copying, memorization, noisy in-context recall, and fuzzy in-context recall. For a detailed exposition of these tasks, we refer to Poli et al. (2024, Section 3.1). Below, we report the dataset parameters used for our experiments; we refer to Poli et al. (2024, Appendix B) for a detailed explanation of each parameter and its effect on the task difficulty.

- **Selective Copying**: `vocab_size`: 64, `seq_len`: 512, `num_tokens_to_copy`: 32, `selective`: True, `num_train_examples`: 3200.
- **Memorization**: `vocab_size`: 4096, `seq_len`: 32, `frac_noise`: 0.0 (no noise), `num_train_examples`: 256.
- **Noisy In-context Recall**: `vocab_size`: 32, `seq_len`: 128, `multi_query`: True, `frac_noise`: 0.2 (noisy), `noise_vocab_size`: 16, `num_train_examples`: 3200.
- **Fuzzy In-context Recall**: `vocab_size`: 32, `seq_len`: 128, `multi_query`: True, `frac_noise`: 0.0 (no noise), `num_train_examples`: 12800.

Training For MAD tasks, we use the parameters shown in Table 4. These are the same parameters as proposed in Poli et al. (2024), but we sweep a larger range of learning rates. All performances/accuracies reported in this paper, correspond to the best performance/accuracy across the learning rate and weight decay sweep reported in Table 4.

For pretraining on Wikitext, we use the training parameters stated in Table 5.

Models The model parameters for each experiment are reported individually in the following section. We use the two block designs shown in Figure 13. Generally, we use the Transformer style block for attention-based models (i.e. readout map $\phi(\cdot) \neq \mathrm{Id}(\cdot)$) and the Gated DeltaNet style block for SSMs/RNNs (i.e. readout map $\phi(\cdot) = \mathrm{Id}(\cdot)$). Additionally, we typically use interleaved MLP blocks after each block (either Type 1 or 2), which do not count towards the number of layers. For example, a two layer model with interleaved MLP blocks, would effectively consist of four layers, two sequence mixing layers and two MLP layers. The MLP block consists of two dense layers with SwiGLU activation (Shazeer, 2020) and inner dimension reported per experiment.

E.2 Model Parameters per Experiment

In the following, we detail the model parameters used to experimentally validate each principle individually.

- The model parameters used to obtain Figures 2, 7 & 10 and validate Principle 2 are provided in Table 6.
- The model parameters used to obtain Figures 8 & 9 and validate Principles 2.1 and 2.2 are provided in Table 7.

[23] `https://github.com/athms/mad-lab`

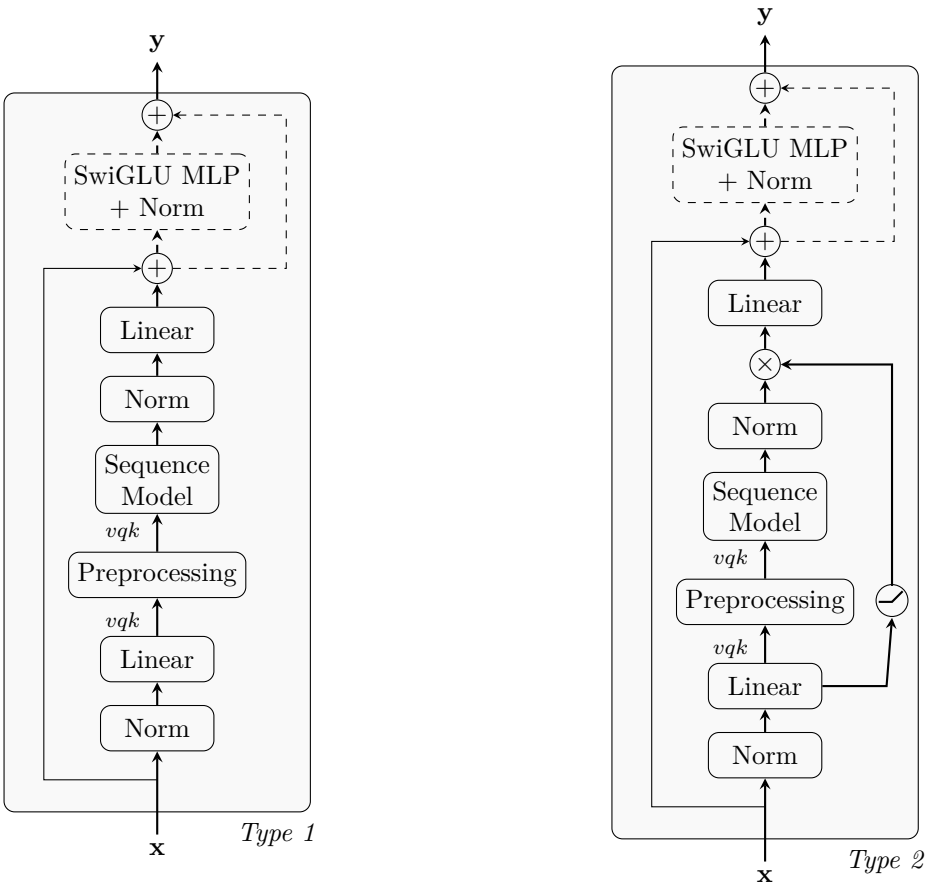


Figure 13: Block designs used in experiments: Transformer style (left; Type 1) and Gated DeltaNet style (right, Type2) from (Yang et al., 2025a, Fig. 1). Dashed parts show the optional SwiGLU MLP.

Table 4: Training parameters used on the MAD benchmark.

Optimizer	`AdamW`
Optimizer Momentum	$(\beta_1, \beta_2) = (0.9, 0.98)$
Dropout	`None`
Batch Size	128
Training Epochs	200
Learning Rate Schedule	`Cosine Decay`
Learning Rate Warmup	`None`
Number of Test Samples	1280
Learning Rate	$[1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}, 1e^{-2}]$
Weight Decay	$[0.0, 0.1]$

Table 5: Training parameters used for pretraining on Wikitext.

Optimizer	`AdamW`
Optimizer Momentum	$(\beta_1, \beta_2) = (0.9, 0.95)$
Dropout	`None`
Batch Size	8
Training Steps	130k
Learning Rate Schedule	`Cosine Decay`
Warmup Steps	3k
Learning Rate	0.0006
Weight Decay	0.1

- The model parameters used to obtain Figure 3 and validate Principle 3 are provided in Table 8.
- The model parameters used to obtain Figure 4 and validate Principle 4 are provided in Table 9.
- The model parameters used to obtain Figure 11 & Table 3 and validate Principle 5 are provided in Table 10 and Table 11, respectively.
- The model parameters used to obtain Figure 12 and validate Principle 6 are provided in Table 12.

Table 6: Model parameters used for Figures 2, 7 & 10. These are the same parameters used in Poli et al. (2024).

Coefficient Dynamics Params.	Detailed in Section 5
Key (k_j) & Querry (q_i)	None
Block Design	Transformer style (Type 1)
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	128
Value Dimension d_v	128
Number of Heads	16
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	Yes

Table 7: Model parameters used for Figures 8 & 9. These are the same parameters used in Poli et al. (2024).

Coefficient Dynamics Params.	Detailed in Appendix D.2
Key (k_j) & Querry (q_i)	Detailed in Appendix D.2
Block Design	Transformer style (Type 1)
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	128
Value Dimension d_v	128
Number of Heads	16
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	Yes

Table 8: Model parameters used for Figure 3. These are the same parameters used in Poli et al. (2024).

Coefficient Dynamics Params.	Detailed in Section 5
Key (k_j) & Querry (q_i)	None
Block Design	Transformer style (Type 1)
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	128
Value Dimension d_v	128
Number of Heads	16
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	Detailed in Section 5

Table 9: Model parameters used for Figure 4.

Coefficient Dynamics Params.	Detailed in Section 5
Key (k_j) & Querry (q_i)	`ShortConv(dim=` 4)
Block Design	Gated DeltaNet style (Type 2)
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	128
Value Dimension d_v	128
Number of Heads	8
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	No

Table 10: Model parameters used for Figure 11. Below, if a parameter is declared with "or", the first option is for attention style models and the second for SSM/RNN style models.

Coefficient Dynamics Params.	Detailed in Appendix D.3
Key (k_j) & Querry (q_i)	None or `ShortConv(dim=` 4)
Block Design	Type 1 or Type 2
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	$[128, 256, 512, 1024]$
Value Dimension d_v	128
Number of Heads	4
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	Yes or No

Table 11: Model parameters used for Table 3. Below, if a parameter is declared with "or", the first option is for softmax attention and the second for Mamba-2.

Coefficient Dynamics Params.	Softmax attention and Mamba-2 standard; varying b_j
Key (k_j) & Querry (q_i)	None or `ShortConv(dim=` 4)
Block Design	Type 1 or Type 2
Number of Layers	8
Embedding Dimension d	768
Inner (State) Dimension n	768
Value Dimension d_v	768
Number of Heads	12
Interleaved MLP Layers	Yes (dim = 512)
Positional Embedding	Yes or No

Table 12: Model parameters used for Figure 12.

Coefficient Dynamics Params.	Detailed in Appendix D.4
Key (k_j) & Querry (q_i)	None
Block Design	Transformer style (Type 1)
Number of Layers	2
Embedding Dimension d	128
Inner (State) Dimension n	128
Value Dimension d_v	128
Number of Heads	16
Interleaved MLP Layers	Yes (dim = 256)
Positional Embedding	Yes