LinQuant: Functional Invariances for Low-Precision Linear Recurrences

Anonymous authors
Paper under double-blind review

Abstract

Linear recurrences are emerging as practical alternatives to softmax attention in large language models, appearing in both standalone and hybrid architectures. However, despite their rapid emergence, low-precision inference remains much less developed for these models than for softmax attention. During prefill, chunkwise-parallel algorithms cast much of the recurrence as dense matrix multiplications, suggesting a natural path to low-precision execution. Simply quantizing these operations in their native form, however, can be fragile. In this work, we identify functional invariances that preserve the full-precision computation while making the operands more amenable to quantization. Building on this observation, we introduce `LinQuant`, a framework for low-precision recurrent prefill based on seven such invariances, including rotations, diagonal rescalings, and alternative placements of recurrent decay and write factors. We derive when these transformations remain valid across common recurrence families spanning linear, gated, and delta-rule updates, and show that they provide complementary degrees of freedom for improving low-precision accuracy. Finally, we translate this framework to production by augmenting SSDi8, an existing low-precision prefill pipeline for Mamba-2, with the applicable invariances. `LinQuant` improves perplexity at matched latency on Mamba-2 1.3B and 8B and Nemotron-H 8B, demonstrating the practical utility of invariance-based quantization for recurrent models.

1 Introduction

Linear recurrences offer an efficient, sub-quadratic alternative to softmax attention for sequence modeling (Katharopoulos et al., 2020; Dao & Gu, 2024; Yang et al., 2025; Team et al., 2025). They summarize past tokens in a fixed-size state, avoiding the growing key-value cache of softmax attention, and are increasingly used in both standalone and hybrid language models (NVIDIA, 2025; Amini et al., 2025; Peng et al., 2025; Beck et al., 2024). For prefill, these models are typically evaluated using chunkwise-parallel algorithms, which process blocks of tokens in parallel through dense matrix multiplications while propagating a recurrent state between them (Dao & Gu, 2024; Yang & Zhang, 2024). This mirrors the blockwise organization of efficient attention kernels such as FlashAttention and makes recurrent prefill naturally amenable to low-precision execution.

Low-precision attention has been studied extensively, with optimized integer and floating-point kernels now forming a mature part of the Transformer inference stack (Zhang et al., 2025; Shah et al., 2024). In contrast, low-precision kernels for linear recurrences remain largely unexplored. Prior work has studied quantization of recurrent models (Xu et al., 2025; Chiang et al., 2025), largely targeting the projections and activations surrounding the Mamba-2 recurrence. SSDi8, on the other hand, executes the recurrent computation in low precision, demonstrating an efficient INT8 implementation specific to Mamba-2 (Kim et al., 2026). We instead study the space more generally, investigating low-precision computation across the design space of modern linear recurrent architectures.

Executing sequence mixers in low-precision can come at the cost of performance. One powerful lever to mitigate this are *functional invariances*. Such transformations preserve the full-precision operator while changing the numerics presented to a quantizer. We distinguish two broad classes. *Mixer-agnostic* invariances act on the residual stream or individual linear layers and do not depend on the algebra of the sequence operator (Xiao et al., 2023; Ashkboos et al., 2024; Liu et al., 2025). Because these transformations are largely independent of the mixer, they transfer naturally

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

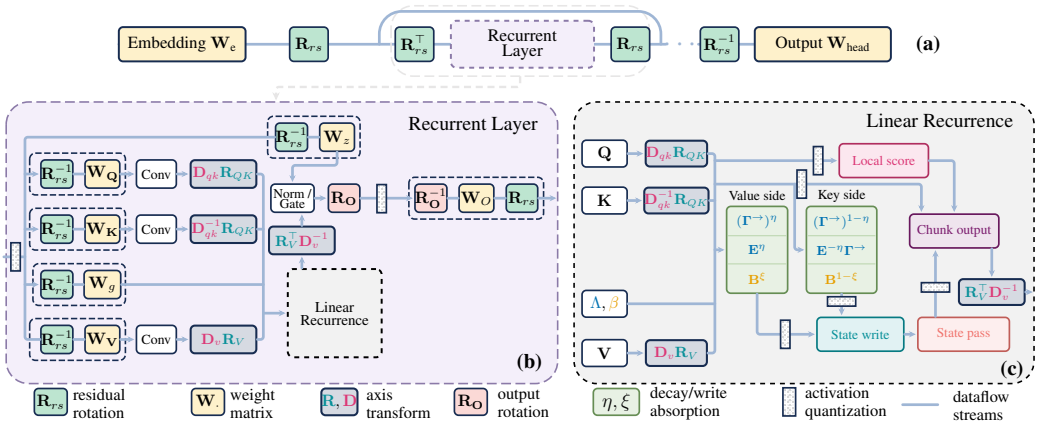


Figure 1: Schematic of a typical recurrent model, at different resolutions. The considered full precision invariances can be classified as mixer-external changes of basis (gray, axis transforms) and mixer-internal decay allocation (green). Those invariances change the distribution of operands exposed directly to quantization. Style inspired by (Liu et al., 2025, Figure 1).

to recurrent architectures and have already been adopted by methods such as MambaQuant and Quamba2 (Xu et al., 2025; Chiang et al., 2025). *Mixer-specific* invariances instead rely on the sequence operator itself, and can be classified as either *mixer-external* or *mixer-internal*. The former materialize as changes of basis and can be applied outside of the sequence-mixer, while the latter are functions of a recurrences' algorithmic realization (Figure 1). Transformer PTQ has exploited such invariances within softmax attention, for example through transformations of its query, key, and value representations (Ashkboos et al., 2024; Liu et al., 2025). In contrast, the mixer-specific invariances of linear recurrences have yet to be characterized.

Linear recurrences present a qualitatively different invariance structure from softmax attention. Their transformations must remain consistent with a state that is updated and carried across chunks, while decay, gating, and delta-rule updates impose additional algebraic constraints. At the same time, these same mechanisms create new degrees of freedom that have no direct attention analogue. We therefore ask which attention-inspired invariances survive in recurrences, what new restrictions arise from the recurrent computation, and what additional invariances are unique to linear recurrences. Our main contributions are summarized as follows:

- We introduce `LinQuant`, a framework for computing low-precision linear recurrences based on both mixer-agnostic and mixer-specific functional invariances, and characterize their applicability across common recurrent architectures.
- Across a controlled set of linear recurrences, we show how recurrence structure, query-key normalization, quantization scale granularity, and state-storage basis determine which functionally equivalent realization is most robust to low precision.
- We implement `LinQuant` in INT8 chunkwise kernels and fuse query–key transformations into SSDi8 without increasing measured prefill latency.
- We apply `LinQuant` to production W4A8 models with INT8 recurrent states, improving over SSDi8 on Mamba-2 1.3B and 8B and Nemotron-H 8B.

2 Background and Preliminaries

2.1 Linear Recurrences

Causal softmax attention (Vaswani et al., 2017) computes

$$\mathbf{o}_t = \mathbf{V}_{1:t}^\top \operatorname{softmax}\left(\mathbf{K}_{1:t}\mathbf{q}_t/\sqrt{D_k}\right),$$

from queries and keys $\mathbf{q}_t, \mathbf{k}_t \in \mathbb{R}^{D_k}$ and values $\mathbf{v}_t \in \mathbb{R}^{D_v}$, where D_k and D_v denote the *state* (key/query) and *value* dimensions, respectively. Linear attention replaces the nonlinear softmax

Table 1: Applicability of the seven invariances introduced in Section 3 to common linear recurrences. ×: not applicable. ✓: applicable. (✓): applicable under constraints.

Nr.	Transformation	LA	Mamba-2	GLA	DN	GDN	KDA
(1/7)	$\mathbf{R}_{QK}$	✓	✓	(✓)	✓	✓	(✓)
(2/7)	$\mathbf{D}_{QK}$	✓	✓	✓	×	×	×
(3/7)	$\mathbf{R}_{V}$	✓	✓	✓	✓	✓	✓
(4/7)	$\mathbf{D}_{V}$	✓	✓	✓	✓	✓	✓
(5/7)	$\left((\mathbf{\Gamma}^{\rightarrow}_{[i]})^{\eta},\ (\mathbf{\Gamma}^{\rightarrow}_{[i]})^{1-\eta}\right)$	×	✓	(✓)	×	✓	(✓)
(6/7)	$\left(\mathbf{E}^{\eta}_{[i]},\ \mathbf{E}^{1-\eta}_{[i]}\right)$	×	✓	✓	×	✓	✓
(7/7)	$\left(\mathbf{B}^{\xi}_{[i]},\ \mathbf{B}^{1-\xi}_{[i]}\right)$	(✓)	(✓)	(✓)	✓	✓	✓

operator with an unnormalized linear kernel (Katharopoulos et al., 2020). In its simplest form, associativity allows past keys and values to be summarized in a fixed-size matrix-valued state $\mathbf{S}_t \in \mathbb{R}^{D_v \times D_k}$,

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top, \qquad \mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t.$$

Modern linear recurrences extend this basic update with decay, write gating, and delta-rule corrections (Dao & Gu, 2024; Yang et al., 2024b;a; Team et al., 2025). We capture these axes through the common recurrence

$$\mathbf{S}_t = \mathbf{S}_{t-1}\, \mathbf{\Lambda}_t \left(\mathbf{I} - \delta\, \beta_t\, \mathbf{k}_t \mathbf{k}_t^\top\right) + \beta_t\, \mathbf{v}_t \mathbf{k}_t^\top, \tag{1}$$

where $\beta_t \in (0, 2)$, $\|\Lambda_t\| \leq 1$, and $\delta \in \{0, 1\}$ is fixed for a given recurrence. Equation 1 separates three structural choices.

Decay controls how the previous state is forgotten before new information is written. A scalar gate $\mathbf{\Lambda}_t = \alpha_t \mathbf{I}$ uses one decay rate per recurrent head (Dao & Gu, 2024; Yang et al., 2025), whereas a diagonal gate $\mathbf{\Lambda}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$ allows the decay to vary across state channels (Yang et al., 2024b; Team et al., 2025). **Write gating** scales the new association $\mathbf{v}_t \mathbf{k}_t^\top$. **Delta correction** (Schlag et al., 2021; Yang et al., 2024c) removes the component of the existing state aligned with the current key before writing the new association. Together, these choices organize the space of modern recurrences as shown in Table 3 in Appendix B.1. We later instantiate this taxonomy as a controlled *mixer zoo*, varying the recurrent update while holding the surrounding model architecture and training setup fixed.

2.2 Chunkwise-parallel computation

Chunkwise implementations parallelize Equation (1) within chunks of C tokens (Dao & Gu, 2024; Yang & Zhang, 2024; Tumma et al., 2026), while processing chunks sequentially. Chunk $[i]$ contains positions $iC + 1, \ldots, (i+1)C$, with incoming state $\mathbf{S}_{[i]} := \mathbf{S}_{iC}$, queries/keys $\mathbf{Q}_{[i]}, \mathbf{K}_{[i]} \in \mathbb{R}^{C \times D_k}$, and values/outputs $\mathbf{V}_{[i]}, \mathbf{O}_{[i]} \in \mathbb{R}^{C \times D_v}$. Indices r, s denote positions within chunk $[i]$, and n denotes a channel. For example, $\alpha^r_{[i]} := \alpha_{iC+r}$ and $\beta^r_{[i]} := \beta_{iC+r}$. We collect the write gates in $\mathbf{B}_{[i]} := \mathrm{diag}(\beta^1_{[i]}, \ldots, \beta^C_{[i]})^\top) \in \mathbb{R}^{C \times C}$. Throughout this work, we also refer to matrix multiplications as "contractions". In the following, we lay out chunkwise parallel formulations for the recurrences covered by Equation (1).

Scalar decay. For no delta correction ($\delta = 0$) and scalar decay $\mathbf{\Lambda}_t = \alpha_t \mathbf{I}$, define the cumulative decay $\gamma^{s \to r}_{[i]} := \prod_{u=s+1}^{r} \alpha^u_{[i]}$ and $\gamma^r_{[i]} := \gamma^{0 \to r}_{[i]}$. The prefix and tail vectors $\mathbf{\Gamma}^{\leftarrow}_{[i]}, \mathbf{\Gamma}^{\rightarrow}_{[i]} \in \mathbb{R}^C$ have entries $(\mathbf{\Gamma}^{\leftarrow}_{[i]})_r = \gamma^r_{[i]}$ and $(\mathbf{\Gamma}^{\rightarrow}_{[i]})_r = \gamma^{r \to C}_{[i]}$. The chunkwise parallel realization then reads (Yang et al., 2024c)

$$\mathbf{S}_{[i+1]} = \gamma^C_{[i]} \mathbf{S}_{[i]} + \mathbf{V}^\top_{[i]} \mathbf{B}_{[i]} (\mathbf{\Gamma}^{\rightarrow}_{[i]} \odot \mathbf{K}_{[i]}),$$
$$\mathbf{O}_{[i]} = (\mathbf{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]}) \mathbf{S}^\top_{[i]} + \mathrm{tril}((\mathbf{Q}_{[i]} \mathbf{K}^\top_{[i]}) \odot \mathbf{\Gamma}^{\rightarrow}_{[i]}) \mathbf{B}_{[i]} \mathbf{V}_{[i]}.$$

Diagonal decay. In the absence of delta correction ($\delta = 0$) but with diagonal decay $\mathbf{\Lambda}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$, define a channel-wise diagonal decay $\gamma^{s \to r}_{[i],n} := \prod_{u=s+1}^{r} \alpha^u_{[i],n}$ and $\gamma^r_{[i],n} := \gamma^{0 \to r}_{[i],n}$. The stacked prefix-

and tail-decay now is $\mathbf{\Gamma}^{\leftarrow}_{[i]}, \mathbf{\Gamma}^{\rightarrow}_{[i]} \in \mathbb{R}^{C \times D_k}$, with entries $(\mathbf{\Gamma}^{\leftarrow}_{[i]})_{rn} = \gamma^{r}_{[i],n}$ and $(\mathbf{\Gamma}^{\rightarrow}_{[i]})_{rn} = \gamma^{r \to C}_{[i],n}$. The chunkwise parallel form then reads (Yang et al., 2024b; Team et al., 2025)

$$\mathbf{S}_{[i+1]} = \mathbf{S}_{[i]}\mathrm{diag}(\boldsymbol{\gamma}^{C}_{[i]}) + (\mathbf{B}_{[i]}\mathbf{V}_{[i]})^{\top}(\mathbf{\Gamma}^{\rightarrow}_{[i]} \odot \mathbf{K}_{[i]}),$$
$$\mathbf{O}_{[i]} = (\mathbf{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]})\mathbf{S}^{\top}_{[i]} + \mathrm{tril}((\mathbf{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]})(\mathbf{K}_{[i]} \oslash \mathbf{\Gamma}^{\leftarrow}_{[i]})^{\top})\mathbf{B}_{[i]}\mathbf{V}_{[i]}.$$

Delta correction. For delta correction ($\delta = 1$) without decay, the chunkwise parallel form reads (Yang et al., 2024c)

$$\mathbf{S}_{[i+1]} = \mathbf{S}_{[i]} + \mathbf{F}_{[i]}{}^{\top}\mathbf{K}_{[i]},$$
$$\mathbf{O}_{[i]} = \mathbf{Q}_{[i]}\mathbf{S}^{\top}_{[i]} + \mathrm{tril}(\mathbf{Q}_{[i]}\mathbf{K}^{\top}_{[i]})\mathbf{F}_{[i]},$$

where $\mathbf{F}_{[i]}$ are delta-corrected *pseudo-values* generated by

$$\mathbf{F}_{[i]} = (\mathbf{I} + \mathrm{tril}(\mathbf{B}_{[i]}\mathbf{K}_{[i]}\mathbf{K}^{\top}_{[i]}, -1))^{-1}\mathbf{B}_{[i]}(\mathbf{V}_{[i]} - \mathbf{K}_{[i]}\mathbf{S}^{\top}_{[i]}).$$

Appendix D.1 gives the decaying delta forms (Yang et al., 2025; Team et al., 2025).

3 LinQuant: Functional Invariances for Low-Precision Recurrences

The chunkwise parallel forms in Section 2.2 expose a set of matrix multiplications and a state carried between chunks. Importantly, these chunkwise forms are not unique: the same full-precision recurrence can be expressed through functionally equivalent realizations that expose different quantities to low precision. `LinQuant` exploits this freedom. We first define the low-precision realization used throughout the paper, then derive seven functional invariances: four axis transformations and three absorptions. Table 1 summarizes their applicability; proofs are in Appendix C.

3.1 Low-Precision Recurrence Realization

Across the recurrences in Table 3, we find that the chunkwise computation can be organized into four matrix contractions: a state write, state read, local score, and intra-chunk output. Together with a recurrent state pass between chunks, this decomposition defines the low-precision boundaries we study. Consider for example the recurrence underlying Mamba-2. Using the notation of Section 2.2 and writing $\mathcal{Q}$ for a quantization operator, we consider the following low-precision chunkwise realization:

$$\widehat{\mathbf{H}}_{[i]} = \mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]})^{\top}\mathcal{Q}(\mathbf{\Gamma}^{\rightarrow}_{[i]} \odot \mathbf{K}_{[i]}), \quad \text{(state write)},$$
$$\widehat{\mathbf{S}}_{[i+1]} = \mathcal{Q}(\gamma^{C}_{[i]}\mathbf{S}_{[i]} + \widehat{\mathbf{H}}_{[i]}), \quad \text{(state pass)},$$
$$\widehat{\mathbf{O}}^{\text{inter}}_{[i]} = \mathcal{Q}(\mathbf{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]})\widehat{\mathbf{S}}^{\top}_{[i]}, \quad \text{(state read)},$$
$$\widehat{\mathbf{A}}_{[i]} = (\mathcal{Q}(\mathbf{Q}_{[i]})\mathcal{Q}(\mathbf{K}_{[i]})^{\top}) \odot \mathbf{\Gamma}_{[i]}, \quad \text{(local score)},$$
$$\widehat{\mathbf{O}}^{\text{intra}}_{[i]} = \widehat{\mathbf{A}}_{[i]}\mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]}), \quad \text{(intra-chunk output)}.$$

For models that employ the delta-rule, the Gram and WY solve are executed in full precision. Delta-rule writes and intra-chunk outputs replace the gated values $\mathbf{B}_{[i]}\mathbf{V}_{[i]}$ by the pseudo-values $\mathbf{F}_{[i]}$ computed for the corresponding recurrence. Appendix D.1 provides more details.

`LinQuant` exploits the fact that functional invariances of these operators generally do not commute with quantization $\mathcal{Q}$. These functional invariances fall into two classes: *changes of basis*, which change the representation along the key or value axes, and *absorptions*, which redistribute decay or write factors within an equivalent chunkwise computation. Appendix E.2 studies sensitivity to the stored-state and write-side quantization boundaries.

3.2 Mixer External Invariances: Changes of Basis

Both the key- and the value axis of the recurrent state $\mathbf{S}_t \in \mathbb{R}^{D_v \times D_k}$ expose a choice of basis which leaves the full precision recurrence unchanged. We consider two such transformations along each

axis: orthogonal rotations and positive diagonal rescalings. Notably, changes of basis along the key axis must remain compatible with the recurrent dynamics, whereas the value axis admits the same equivariance across all recurrences 1.

(1/7) Key-side rotation. A joint orthogonal rotation of queries and keys,

$$(\boldsymbol{q}_t, \boldsymbol{k}_t) \to (\mathbf{R}_{QK}\boldsymbol{q}_t, \mathbf{R}_{QK}\boldsymbol{k}_t), \qquad \mathbf{R}_{QK} \in \mathcal{O}(D_k),$$

preserves query/key inner products and is a standard invariance of full precision softmax attention exploited by rotation-based PTQ methods such as QuaRot and SpinQuant (Ashkboos et al., 2024; Liu et al., 2025). In low precision, random rotations spread energy across channels, reducing outlier concentration and improving quantization robustness.

For a linear recurrence, the invariance holds if $\mathbf{R}_{QK}\mathbf{\Lambda}_t = \mathbf{\Lambda}_t\mathbf{R}_{QK}$ (Nazari & Rusch, 2026), and implicitly applies a change of basis to the states key axis, which transforms like $\mathbf{S}_t \to \mathbf{S}_t\mathbf{R}_{QK}^\top$. Invariance thus holds with no decay or scalar decay. With diagonal decay, a general rotation does not commute with $\mathbf{\Lambda}_t$, so the recurrent state cannot be carried in a globally rotated basis. However, query-key and key-key contractions like $\mathcal{Q}(\mathbf{Q})\mathcal{Q}(\mathbf{K}^\top)$ can still be rotated locally before quantizing.

In this work, we use random signed Hadamard rotations. Recurrent models may apply per-channel causal convolutions and nonlinearities between their input projections and the recurrence, which generally do not commute with a dense rotation. We thus apply $\mathbf{R}_{QK}$ online after these operations.

(2/7) Key-side rescaling. The key axis also admits a reciprocal diagonal transformation,

$$(\boldsymbol{q}_t, \boldsymbol{k}_t) \to (\mathbf{D}_{QK}\boldsymbol{q}_t, \mathbf{D}_{QK}^{-1}\boldsymbol{k}_t), \qquad \mathbf{D}_{QK} \succ 0 \text{ diagonal},$$

which likewise preserves query-key inner products. Related rescalings underlie SmoothAttention in QServe and, more broadly, methods such as SmoothQuant (Lin et al., 2025; Xiao et al., 2023).

For a linear recurrence, the state implicitly transforms like $\mathbf{S}_t \to \mathbf{S}_t\mathbf{D}_{QK}^{-1}$. Unlike a dense rotation, $\mathbf{D}_{QK}$ commutes with diagonal decay, making the transformation valid for linear writes under no, scalar, or diagonal decay. It does not, however, preserve delta corrected writes.

$\mathbf{D}_{QK}$ transports magnitude between queries and keys. In this work, we parameterize $\mathbf{D}_{QK} = \mathrm{diag}(\boldsymbol{d})$ by $d_n = (\max_t |k_{t,n}| / \max_t |q_{t,n}|)^\kappa$, where the maxima are computed over calibration tokens within each head. The exponent $\kappa \in [0, 1]$ controls the rescaling strength. Specifically, $\kappa = \frac{1}{2}$ balances the magnitudes of queries and keys. We leave other parametrizations to future work.

(3/7) Value-side rotation. The value axis admits an analogous orthogonal transformation,

$$\boldsymbol{v}_t \to \mathbf{R}_V\boldsymbol{v}_t, \qquad \mathbf{R}_V \in \mathcal{O}(D_v).$$

In softmax attention, value rotations are used for KV-cache quantization (Liu et al., 2025). The operator, however, is not invariant to value rotations, which have to be undone after the softmax. Linear recurrences exhibit the same equivariance. Intuitively, this is because the recurrence acts independently along the value axis, which is implicitly rotated like $\mathbf{S}_t \to \mathbf{R}_V\mathbf{S}_t$.

We note that, compared to softmax attention, the recurrent setting reveals another degree of freedom. The state can, in between chunks, either be stored in its canonical basis, $\mathcal{Q}(\mathbf{S}_t)$, or in a rotated basis, $\mathcal{Q}(\mathbf{R}_V\mathbf{S}_t)$. These realizations are equivalent in full precision but expose different representations to $\mathcal{Q}$. Rotated storage is what the rotation produces by itself, whereas canonical storage returns each chunk's write to the original basis before it joins the carried state. This comes at the cost of an inverse transform per chunk, and keeps the state in the basis whose per-channel scales fit it.

(4/7) Value-side rescaling. Finally, the value axis admits the positive diagonal analogue

$$\boldsymbol{v}_t \to \mathbf{D}_V\boldsymbol{v}_t, \qquad \mathbf{D}_V \succ 0 \text{ diagonal}.$$

The recurrent state transforms accordingly as $\mathbf{S}_t \to \mathbf{D}_V\mathbf{S}_t$, with the inverse rescaling absorbed into the output projection. As with $\mathbf{R}_V$, this transformation is valid across all recurrence families considered here.

In this work, we parametrize $\mathbf{D}_V = \mathrm{diag}(\boldsymbol{d})$ by $d_p = \left((\prod_{j=1}^{D_v} \max_t |v_{t,j}|)^{1/D_v} / \max_t |v_{t,p}|\right)^\kappa$, where the maxima are computed over calibration tokens and the product runs over value channels within each head. This parametrization moves range between value channels. At $\kappa = 1$, the ranges of the values channels are equalized.

3.3 MIXER INTERNAL INVARIANCES: ABSORPTIONS

The first four invariances discussed in the previous section change the basis along the two axes of the recurrent state. The remaining three redistribute multiplicative factors within a chunkwise parallel implementation of the recurrence. In particular, decay and write factors can often be divided between the two sides of a matrix product without changing the full-precision operator.

(5/7) Scalar-decay absorption. For scalar decay, the tail-decay $\mathbf{\Gamma}^{\rightarrow}_{[i]}$ can be split between the value and key sides of the state write:

$$\mathbf{H}_{[i]} = ((\mathbf{\Gamma}^{\rightarrow}_{[i]})^{\eta} \odot (\mathbf{B}_{[i]}\mathbf{V}_{[i]}))^{\top}((\mathbf{\Gamma}^{\rightarrow}_{[i]})^{1-\eta} \odot \mathbf{K}_{[i]}), \qquad \eta \in [0,1].$$

Thus, η continuously interpolates between equivalent realizations of the same write. At $\eta = 0$, all tail decay remains on the keys, which we call *key-side decay absorption*. At $\eta = 1$, it moves entirely to the values, giving *value-side decay absorption*, the endpoint used by SSDi8 (Kim et al., 2026). Intermediate values split the decay between the two sides. The same split applies to delta-rule writes, replacing $\mathbf{B}_{[i]}\mathbf{V}_{[i]}$ by the corresponding pseudo-values $\mathbf{F}_{[i]}$.

(6/7) Diagonal-decay absorption. Diagonal decay varies across state channels and therefore cannot, in general, be transferred entirely to the value side. Nevertheless, we can factor each token's decay into a scalar magnitude and a relative channelwise pattern. This decomposition is not unique: any positive scalar factor yields an exact redistribution. Here, we use the channelwise maximum, and leave alternative choices to future work.

For diagonal tail decay $\mathbf{\Gamma}^{\rightarrow}_{[i]}$, define the per-token scalar magnitude and normalized decay

$$e^{r}_{[i]} := \max_{n} (\mathbf{\Gamma}^{\rightarrow}_{[i]})_{rn}, \qquad \mathbf{E}_{[i]} := \mathrm{diag}(e^{1}_{[i]}, \ldots, e^{C}_{[i]}), \qquad \overline{\mathbf{D}}^{\mathrm{tail}}_{[i]} := \mathbf{E}^{-1}_{[i]}\mathbf{\Gamma}^{\rightarrow}_{[i]}.$$

Then the state write can be expressed as

$$\mathbf{H}_{[i]} = (\mathbf{E}^{\eta}_{[i]}\mathbf{B}_{[i]}\mathbf{V}_{[i]})^{\top}(\mathbf{E}^{1-\eta}_{[i]}\overline{\mathbf{D}}^{\mathrm{tail}}_{[i]} \odot \mathbf{K}_{[i]}), \qquad \eta \in [0,1].$$

Here, η moves the magnitude of the diagonal decay. At $\eta = 0$, the standard key-side form is recovered. At $\eta = 1$, each token's scalar decay magnitude moves to the value side, while the relative channel-dependent decay necessarily remains on the key.

(7/7) Write-factor absorption. For a linear write, the write factor admits an analogous split:

$$\mathbf{H}_{[i]} = (\mathbf{B}^{\xi}_{[i]}\mathbf{V}_{[i]})^{\top}(\mathbf{\Gamma}^{\rightarrow}_{[i]} \odot \mathbf{B}^{1-\xi}_{[i]}\mathbf{K}_{[i]}), \qquad \xi \in [0,1].$$

The endpoints $\xi = 1$ and $\xi = 0$ place the write factor entirely on the value and key sides, respectively.

For delta-corrected writes, the write gate also enters the triangular system used to construct the pseudo-values. Without decay, split it between the two key factors of the Gram matrix:

$$\mathbf{L}^{(\sigma)}_{[i]} := \mathrm{tril}((\mathbf{B}^{\sigma}_{[i]}\mathbf{K}_{[i]})(\mathbf{B}^{1-\sigma}_{[i]}\mathbf{K}_{[i]})^{\top}, -1), \qquad \sigma \in [0,1].$$

Using $\mathbf{B}^{\sigma}_{[i]}$ on the corresponding right-hand side yields transformed pseudo-values $\mathbf{F}^{(\sigma)}_{[i]}$ satisfying

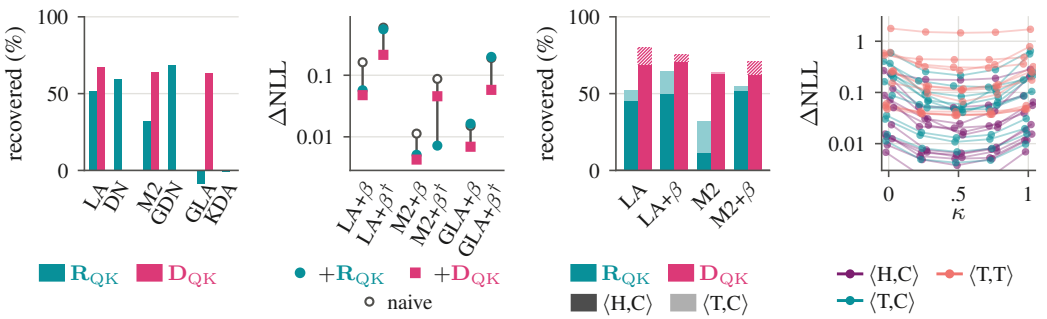
$$\mathbf{F}^{(\sigma)}_{[i]} = \mathbf{B}^{\sigma-1}_{[i]}\mathbf{F}_{[i]},$$

where $\mathbf{F}_{[i]}$ denotes the pseudo-values produced by the standard formulation in Section 2.2. A second parameter τ splits the complementary factor $\mathbf{B}^{1-\sigma}_{[i]}$ between the transformed write operands:

$$\mathbf{V}^{(\sigma,\tau)}_{[i]} = \mathbf{B}^{(1-\sigma)(1-\tau)}_{[i]}\mathbf{F}^{(\sigma)}_{[i]}, \qquad \mathbf{K}^{(\sigma,\tau)}_{[i]} = \mathbf{B}^{(1-\sigma)\tau}_{[i]}\mathbf{K}_{[i]},$$
$$\mathbf{H}_{[i]} = \left(\mathbf{V}^{(\sigma,\tau)}_{[i]}\right)^{\top}\mathbf{K}^{(\sigma,\tau)}_{[i]} = \mathbf{F}^{\top}_{[i]}\mathbf{K}_{[i]}, \qquad \tau \in [0,1].$$

Thus, σ determines how the write factor is distributed inside the Gram construction, while τ determines how the remaining compensation is distributed in the subsequent write. At $\tau = 0$, the compensation remains on the pseudo-values. At $\tau = 1$, it moves to the keys. Setting $\sigma = 1$ recovers the standard triangular formulation, while $\sigma = 1/2$ places square-root write factors on the two Gram operands. Appendix C proves these equivalences and extends the construction to decayed delta-rule recurrences.

Together, the four changes of basis and three absorptions define seven functional degrees of freedom for realizing the same recurrent computation. Their applicability depends on the recurrence structure, while their numerical behavior depends on the resulting low-precision representation, which we explore in the following section.



(a) Recurrence structure. (b) QK norm. (c) Scale granularity. (d) Rescaling exponent.

Figure 2: Key-axis transformations at INT6. (a) Recovered NLL by decay and write rule. (b) Excess NLL with and without QK normalization (†). Panels (a,b) use $\langle \mathrm{T}, \mathrm{C} \rangle$ and $\kappa = 1/2$ for $\mathbf{D}_{QK}$. (c) Recovery under $\langle \mathrm{H}, \mathrm{C} \rangle$ and its change under $\langle \mathrm{T}, \mathrm{C} \rangle$, with selected κ. (d) κ sweep, one curve per model.

4 Experiments

In this section, we first evaluate how the invariances discussed in Section 3 affect quantization error in a controlled model suite (Section 4.1). We then translate these insights to a production setting and test their quality and latency in pretrained Mamba-2 and Nemotron-H models (Section 4.2).

4.1 Analyzing LinQuant

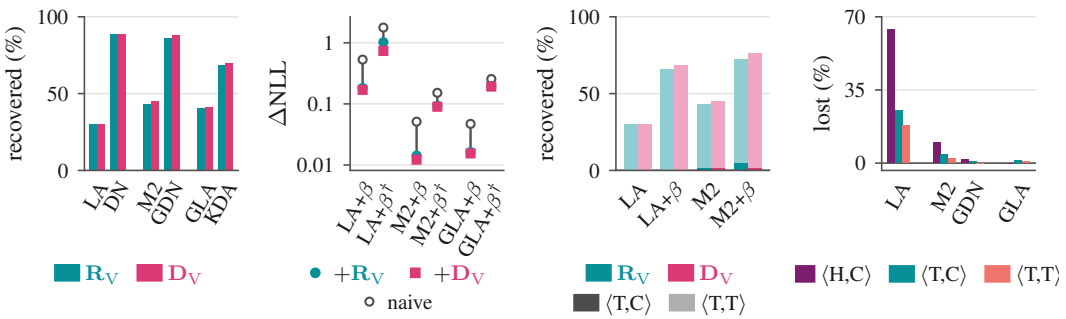
Model suite. To isolate the impact of each component that comprises the linear recurrence, we set up a controlled *mixer zoo*. All models share the same backbone and training recipe. It consists of Linear Attention, Mamba-2, GLA, DeltaNet, Gated DeltaNet, and KDA at 340M and 1.3B parameters, trained with 15B and 100B tokens of SlimPajama, respectively. The 340M suite furthermore contains models that vary write gating β_t and QK normalization. Appendix B.1 gives the full setup.

Quantization settings. We evaluate how the identified invariances affect quantization error using WikiText-2 test NLL, comparing against full precision and a naive quantizer without invariance transforms. We use the signed Hadamard rotations and diagonal rescalings discussed in Section 3.2, calibrated using WikiText-2. We use INT6 as the main diagnostic precision. The end-of-chunk states are passed in FP32, while quantizing a copy to an INT8 readout. Appendix B.1 gives calibration details, and Appendix E reports INT8, INT4, 1.3B, and zero-shot results.

Scale geometries. The queries/read-keys and values can be quantized with different granularities, denoted by $\langle b_{QK}, b_V \rangle$. Here, b_{QK} gives the scale-granularity for queries and read-keys, while b_V gives that of the values. Both can be C (per-channel), H (per head) or T (per-tensor). The scale groupings considered are directly influenced by the chunkwise kernels. In a matrix contraction, output-channel scales can be applied after the product, whereas scales along the summed dimension must enter the contraction. The key axis is contracted in query/key products and state reads. In contrast, the value axis remains an output dimension and admits per-channel scales that can be incorporated in a kernel's output scaling without modifying the integer contraction or requiring an additional pass. We compare the three scale regimes $\langle \mathrm{H}, \mathrm{C} \rangle$, $\langle \mathrm{T}, \mathrm{C} \rangle$, and $\langle \mathrm{T}, \mathrm{T} \rangle$. In the state write contraction, the key dimension stays an outer dimension. We exploit this by quantizing write keys per channel in $\langle H, C \rangle$, where each contraction quantizes its own copy of the keys, so the write can take the finest scale it can absorb. In the other two regimes, write keys are quantized per tensor and reused by each contraction.

We next examine key-axis transformations (1-2), value-axis transformations (3-4), and absorptions (5-7) on quantization error under different scale-granularities, as well as the choice of storage basis.

(1-2) Key-axis transformations. The benefit of $\mathbf{R}_{QK}$ depends on the recurrence structure. With no or scalar decay, it rotates the carried state along the key axis and reduces quantization error. With diagonal decay, it is restricted to local contractions and provides little benefit (Figure 2a). In contrast, $\mathbf{D}_{QK}$ commutes with diagonal decay and improves quantization error where applicable. QK normalization also reduces quantization error (Figure 2b). Coarsening key scales increases the

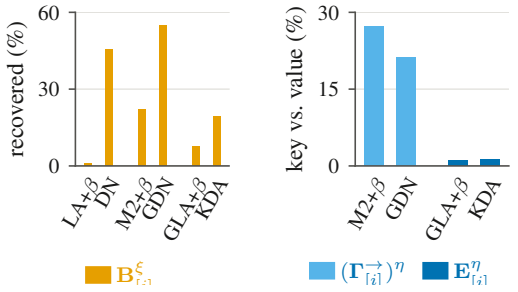


(a) Recurrence structure. (b) QK norm. (c) Scale granularity. (d) Storage basis.

Figure 3: Value-axis transformations at INT6. (a) Recovered excess NLL under $\langle \mathrm{T}, \mathrm{T} \rangle$. (b) Excess NLL under $\langle \mathrm{T}, \mathrm{T} \rangle$ with and without QK normalization (†). (c) Recovery under $\langle \mathrm{T}, \mathrm{C} \rangle$ and $\langle \mathrm{T}, \mathrm{T} \rangle$. (d) Recovery lost by storing the state rotated instead of canonically under $\mathbf{R}_V$, relative to naive.

benefit of $\mathbf{R}_{QK}$, while $\mathbf{D}_{QK}$ remains useful with per-head scales (Figure 2c). The diagonal rescaling is generally best shared between queries and keys (Figure 2d).

(3-4) Value-axis transformations. Both $\mathbf{R}_V$ and $\mathbf{D}_V$ apply to every recurrence. They are most useful with per-tensor value scales, where they recover a large portion of the excess NLL (Figure 3a). The two transformations yield similar gains as value scales are coarsened (Figure 3c). Only $\mathbf{R}_V$ introduces a choice of state basis: quantizing the state in the rotated basis incurs more error than quantizing it in the canonical basis (Figure 3d). While the two transformations give similar quality, the cost of canonical storage favors $\mathbf{D}_V$, which folds into the quantization scales.



(a) Write gate. (b) Decay absorption.

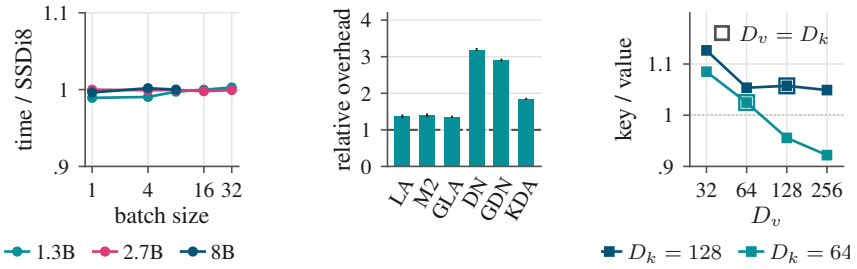
Figure 4: Absorptions at INT6 under $\langle \mathrm{T}, \mathrm{T} \rangle$. (a) Recovered excess NLL from the selected write-factor allocation (ξ or (σ, τ)). (b) NLL increase from placing decay on keys ($\eta = 0$) rather than values ($\eta = 1$), relative to $\eta = 1$.

(5-7) Absorptions. We study absorptions in the coarse-scale regime. Under $\langle \mathrm{T}, \mathrm{T} \rangle$, write-gate absorption recovers a significant portion of the excess NLL (Figure 4a). Decay placement matters most for scalar decay: with per-tensor value scales, moving scalar decay from values to keys increases excess NLL. By contrast, diagonal-decay absorption provides little gain (Figure 4b). We conjecture that variation across the diagonal decay, which remains on the keys, matters more than its overall magnitude and leave other ways of extracting a factor to future work.

Kernels. Each cell runs in one INT8 Triton chunk kernel that realizes all seven invariances and reproduces the harness (Appendix D.2). Only $\mathbf{R}_V$ adds overhead: rescalings fold into the quantization scales, absorptions move factors between operands the kernel already forms, and $\mathbf{R}_{QK}$ fuses into the pass that quantizes queries and keys, whereas canonical storage returns the write and local output to the original basis inside the call (Figure 5b).

4.2 LinQuant in a Production Setting

Production setup. We next apply `LinQuant` to production-scale pretrained Mamba-2 models and Nemotron-H (Blakeman et al., 2025). Our starting point is SSDi8 (Kim et al., 2026), a low-precision recurrent prefill pipeline for Mamba-2, which we augment with the applicable `LinQuant` invariances. The production Mamba-2 recurrence is most analogous to the no-β, no-QK-normalization Mamba-2 model in our mixer zoo. The controlled study therefore points primarily to the key-side changes of basis, $\mathbf{R}_{QK}$ and $\mathbf{D}_{QK}$, while its fine-grained value scales suggest little need for a value-side change of basis. We add these key-side transformations to SSDi8's W4A8 projection and INT8-recurrence pipeline and evaluate both equivalent placements of scalar decay. Table 2 reports quality. Setup and further results are in Appendices B.2 and F.



(a) Prefill time. (b) Value rotation. (c) Decay side.

Figure 5: (a) End-to-end prefill time of `LinQuant` ($\mathbf{R}_{QK} + \mathbf{D}_{QK}$, value-side decay) relative to the matched SSDi8 setting, production Mamba-2 (A5000, $T = 2048$). (b) INT8 zoo kernel with $\mathbf{R}_V$ and canonical storage relative to naive INT8 (A5000, $B = H = 8, T = 2048, D_k = D_v = 128$). (c) Production recurrence latency with decay on keys relative to values (static scales, 64 heads, batch 8).

Key-side transformations transfer. Across every reported Mamba-2 and Nemotron-H model, $\mathbf{R}_{QK}$ improves perplexity over the matched SSDi8 setting, while $\mathbf{D}_{QK}$ provides an additional gain in most configurations (Table 2). $\mathbf{D}_{QK}$ folds into the existing quantization scales, while $\mathbf{R}_{QK}$ is fused into operand preparation, yielding no measurable increase in end-to-end prefill latency (Figure 5a). Full results are reported in Appendix F. The SSDi8 pipeline already uses per-channel value scales, matching the controlled setting in which $\mathbf{R}_V$ and $\mathbf{D}_V$ provide little benefit. Rotated state storage performs worse, while canonical storage does not improve over the unrotated baseline (Appendix F.1). Since canonical $\mathbf{R}_V$ adds overhead (Figure 5b), we apply neither value-side transformation in the production configuration.

Table 2: Production WikiText-2 perplexity. SSDi8 uses value-side decay absorption. The combined transformation is evaluated with decay on either operand. More results in Table 23.

			$\mathbf{R}_{QK}$	$\mathbf{R}_{QK} + \mathbf{D}_{QK}$	
Model	FP16	SSDi8	value	value	key
Mamba-2					
1.3B	10.420	10.928	10.860	10.849	10.830
2.7B	9.060	9.441	9.379	9.383	9.372
8B	7.250	7.620	7.456	7.435	7.437
Nemotron-H					
Base	6.030	6.338	6.186	6.178	6.175
Reasoning	8.423	8.921	8.666	8.650	8.670

Kernels. The optimized SSDi8 kernels quantize operands once and reuse their INT8 representations across multiple contractions. Kim et al. (2026) place scalar decay on the value side, motivated by concerns that key-side decay would require per-position scales and materializing per-head keys. Under matched scales, we find neither is necessary: key-side absorption matches value-side quality (Appendix F), and the per-head decay can be applied in GPU registers without materializing the expanded tensor (Appendix D.2). The remaining difference is operand reuse. Moving decay onto an operand after its INT8 representation has been formed requires rereading and requantizing the floating-point tensor. The cost therefore depends on which operand is affected, its dimension, and how often its quantized representation is reused. Figure 5c shows that value-side absorption is faster for the evaluated Mamba-2 shapes, although the gap narrows and can reverse as the value dimension D_v increases. We therefore retain value-side absorption for its lower runtime on the target Mamba-2 geometries.

5 Conclusion

In this work, we introduce `LinQuant`, a framework that exploits seven functional invariances for low-precision linear recurrences, and characterize their applicability across a wide range of recurrent mixers. These consist of extra-kernel changes of basis and intra-kernel absorptions, namely alternative allocations of decay and write factors between operands. Our controlled study shows their benefits depend on recurrence structure, QK normalization, scale granularity, and the basis used to quantize the state. Notably, changes of basis and write-gate allocation are particularly effective under coarse scales. These findings transfer to production Mamba-2 and Nemotron-H models, where `LinQuant` improves quantization quality without increasing measured prefill latency. Although our implementation focuses on integer prefill, the introduced invariances provide a broader design space (e.g. floating-point) for future low-precision implementations.

AI USE STATEMENT

In this work, we use generative AI tools to formulate mathematical claims, provide critical ingredients for proving them, to design or give feedback on our methodology and experiments, to create or edit code, implement methods, to propose or refine hypotheses, to support qualitative and thematic data analysis, interpret results, and to do a literature review. Additionally, we used generative AI tools for LaTeX formatting, to draft parts of the appendix, and for brainstorming. We have not used generative AI tools for generating synthetic datasets, to help develop theoretical models or conceptual frameworks, to assist with translation, or to clean or reformat datasets. We have reviewed all AI-assisted work, checked the code as well as the reformatted LaTeX tables. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Alexander Amini, Anna Banaszak, Harold Benoit, Arthur Böök, Tarek Dakhran, Song Duong, Alfred Eng, Fernando Fernandes, Marc Härkönen, Anne Harrington, Ramin Hasani, Saniya Karwa, Yuri Khrustalev, Maxime Labonne, Mathias Lechner, Valentine Lechner, Simon Lee, Zetian Li, Noel Loo, Jacob Marks, Edoardo Mosca, Samuel J. Paech, Paul Pak, Rom N. Parnichkun, Alex Quach, Ryan Rogers, Daniela Rus, Nayan Saxena, Bettina Schlager, Tim Seyde, Jimmy T. H. Smith, Aditya Tadimeti, and Neehal Tumma. Lfm2 technical report, 2025. URL https://arxiv.org/abs/2511.23404.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated llms. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/b5b939436789f76f08b9d0da5e81af7c-Abstract-Conference.html.

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, 2024. URL https://arxiv.org/abs/2405.04517.

Aaron Blakeman, Aarti Basant, Abhinav Khattar, Adithya Renduchintala, Akhiad Bercovich, Aleksander Ficek, Alexis Bjorlin, Ali Taghibakhshi, Amala Sanjay Deshmukh, Ameya Sunil Mahabaleshwarkar, et al. Nemotron-h: A family of accurate and efficient hybrid mamba-transformer models. *arXiv preprint arXiv:2504.03624*, 2025.

Makram Chahine, Philipp Nazari, Daniela Rus, and T Konstantin Rusch. The curious case of in-training compression of state space models. *arXiv preprint arXiv:2510.02823*, 2025.

Hung-Yueh Chiang, Chi-Chih Chang, Natalia Frumkin, Kai-Chiang Wu, and Diana Marculescu. Quamba: A post-training quantization recipe for selective state space models. *arXiv preprint arXiv:2410.13229*, 2024.

Hung-Yueh Chiang, Chi-Chih Chang, Natalia Frumkin, Kai-Chiang Wu, Mohamed S. Abdelfattah, and Diana Marculescu. Quamba2: A robust and scalable post-training quantization framework for selective state space models, 2025. URL https://arxiv.org/abs/2503.22879.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tcbBPnfwxS.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of

Proceedings of Machine Learning Research, pp. 5156–5165. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Hyunwoo Kim, Byoungchan Ko, Minseok Kang, Minwoo Kim, Dongjin Lee, Jaehoon Lee, Sungroh Yoon, and Dahuin Jung. Ssdi8: Accurate and efficient 8-bit quantization for state space duality. In *The Fourteenth International Conference on Learning Representations*, 2026.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, volume 6, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html.

Yujun Lin, Haotian Tang, Shang Yang, Zhekai Zhang, Guangxuan Xiao, Chuang Gan, and Song Han. QServe: W4A8KV4 quantization and system co-design for efficient llm serving. In *Proceedings of Machine Learning and Systems*, volume 7, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/hash/fbe2b2f74a2ece8070d8fb073717bda6-Abstract-Conference.html.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: Llm quantization with learned rotations. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ogO6DGE6FZ.

Philipp Nazari and T Konstantin Rusch. The key to state reduction in linear attention: A rank-based perspective. *arXiv preprint arXiv:2602.04852*, 2026.

NVIDIA. Nvidia nemotron 3: Efficient and open intelligence, 2025. URL https://arxiv.org/abs/2512.20856.

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

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers, 2021. URL https://arxiv.org/abs/2102.11174.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. Flashattention-3: Fast and accurate attention with asynchrony and low-precision, 2024. URL https://arxiv.org/abs/2407.08608.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. Deltaproduct: Improving state-tracking in linear rnns via householder products, 2025. URL https://arxiv.org/abs/2502.10297.

Lintang Sutawika, Hailey Schoelkopf, Leo Gao, Stella Biderman, Baber Abbasi, Jonathan Tow, et al. Eleutherai/lm-evaluation-harness: v0.4.1, January 2024. URL https://doi.org/10.5281/zenodo.10600400. Software.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, T. Y. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei, Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi linear: An expressive, efficient attention architecture, 2025. URL https://arxiv.org/abs/2510.26692.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks, 2024. URL https://arxiv.org/abs/2402.04396.

Neehal Tumma, Mathias Lechner, Noel Loo, Ramin Hasani, and Daniela Rus. Leveraging low-rank and sparse recurrent connectivity for robust closed-loop control. In *The Twelfth International Conference on Learning Representations*, 2024.

Neehal Tumma, Noel Loo, and Daniela Rus. Preconditioned deltanet: Curvature-aware sequence modeling for linear recurrences, 2026. URL https://arxiv.org/abs/2604.21100.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 2023. URL https://proceedings.mlr.press/v202/xiao23c.html.

Zukang Xu, Yuxuan Yue, Xing Hu, Zhihang Yuan, Zixu Jiang, Zhixuan Chen, Jiangyong Yu, Chen Xu, Sifan Zhou, and Dawei Yang. Mambaquant: Quantizing the mamba family with variance aligned rotation methods, 2025. URL https://arxiv.org/abs/2501.13484.

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

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024a.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56501–56523. PMLR, 21–27 Jul 2024b. URL https://proceedings.mlr.press/v235/yang24ab.html.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024c.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule, 2025. URL https://arxiv.org/abs/2412.06464.

Jintao Zhang, Jia Wei, Haofeng Huang, Pengle Zhang, Jun Zhu, and Jianfei Chen. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration, 2025. URL https://arxiv.org/abs/2410.02367.

A Extended related work

Linear recurrent and hybrid language models. A growing line of work replaces softmax attention with recurrent sequence mixers whose state size and per-token decode cost do not grow with context length. This is achieved by implicitly summarizing past tokens in a fixed-size hidden state, instead of explicitly in a KV-cache (Katharopoulos et al., 2020; Schlag et al., 2021; Yang et al., 2024b). Selective state-space models such as Mamba and Mamba-2 introduce input-dependent recurrent dynamics and hardware-efficient implementations (Gu & Dao, 2024; Dao & Gu, 2024).

More expressive recurrences modify how information is retained and written into the hidden state. DeltaNet corrects stored associations by removing existing associations from its state (Yang et al., 2024c), while Gated DeltaNet combines this correction mechanism with scalar forgetting (Yang et al., 2025), and KDA uses channelwise decay (Team et al., 2025). Preconditioned DeltaNet further incorporates curvature-aware updates (Tumma et al., 2026), while DeltaProduct further enhances the state transition operator (Siems et al., 2025). In a recent development, large-scale hybrid models combine recurrent and softmax attention layers (Team et al., 2025; Blakeman et al., 2025).

Our work studies how the structure of these different recurrences enables and constrains functional invariances. We then leverage these invariances for post-training quantization (PTQ).

PTQ of softmax Transformers. Weight-only PTQ methods such as GPTQ and AWQ reduce quantization error through reconstruction-based compensation or channel protection (Frantar et al., 2023; Lin et al., 2024). For joint weight and activation quantization, SmoothQuant transfers activation range into weights (Xiao et al., 2023), while QServe combines quantization with efficient serving (Lin et al., 2025). Rotation-based methods redistribute concentrated energy across channels: QuIP# uses Hadamard incoherence processing for weight quantization (Tseng et al., 2024), QuaRot extends rotations to weights, activations, and the KV cache (Ashkboos et al., 2024), and SpinQuant learns rotations from calibration data (Liu et al., 2025). `LinQuant` extends the study of functional invariances to linear recurrences.

Quantization of recurrent language models. Quamba develops an 8-bit PTQ recipe around sensitive selective-scan features and activation outliers (Chiang et al., 2024). MambaQuant uses variance-aligned rotations (Xu et al., 2025), while Quamba2 extends PTQ across Mamba and Mamba-2 at multiple precisions (Chiang et al., 2025).

SSDi8 targets low-precision execution inside the chunkwise Mamba-2 computation (Kim et al., 2026). It reuses quantized keys and queries, absorbs decay into values, and carries an INT8 state, all to maintain a persistent INT8 computational path. This decay placement is the value-side endpoint of our scalar-decay absorption family. Our production pipeline builds on SSDi8 by adding state-axis rotations and rescalings (Section 4.2).

Recurrent-state compression beyond quantization. Chahine et al. (2025) prune the hidden state during training by leveraging the free choice of key-axis basis, which we identify as invariance **(1)**. Similarly, Nazari & Rusch (2026) leverage the free choice of basis to prune the state of pretrained linear recurrences. Low-rank and sparse recurrent connectivity has also been studied outside language modeling (Tumma et al., 2024). In `LinQuant`, we instead retain state dimensions and use equivalent representations to improve quantization, addressing a complementary source of recurrent efficiency.

B Experimental setup

We give the setup and timing methodology for the controlled study and production setting. Results are reported in Appendices E and F.

B.1 Controlled study

Models and training. The 340M zoo contains fifteen recurrence variants, or cells: LA, Mamba-2, and GLA with and without write gating and QK normalization, plus DN, GDN, and KDA with QK normalization. All models use the FLA implementation (Yang & Zhang, 2024) and share a 24-layer architecture with a SwiGLU MLP, an output gate on the mixer, and tied embeddings. At 340M parameters, the model width is 1024 with 8 heads, $D_k = 64$, and $D_v = 128$. At 1.3B, the width is 2048 with 16 heads and the same head dimensions.

Table 3: State-update rules for recurrent models in the mixer-zoo design space. Colors identify decay and write gating.

model	update
Linear attention (Katharopoulos et al., 2020)	$\mathbf{S}_t = \mathbf{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^\top$
Mamba-2 (Dao & Gu, 2024)	$\mathbf{S}_t = \alpha_t \, \mathbf{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^\top$
Gated Linear Attention (GLA) (Yang et al., 2024b)	$\mathbf{S}_t = \mathbf{S}_{t-1} \, \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t \boldsymbol{k}_t^\top$
DeltaNet (Schlag et al., 2021; Yang et al., 2024c)	$\mathbf{S}_t = \mathbf{S}_{t-1} \left(\mathbf{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\top\right) + \beta_t \, \boldsymbol{v}_t \boldsymbol{k}_t^\top$
Gated DeltaNet (Yang et al., 2025)	$\mathbf{S}_t = \alpha_t \, \mathbf{S}_{t-1} \left(\mathbf{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\top\right) + \beta_t \, \boldsymbol{v}_t \boldsymbol{k}_t^\top$
Kimi Delta Attention (KDA) (Team et al., 2025)	$\mathbf{S}_t = \mathbf{S}_{t-1} \, \mathrm{diag}(\boldsymbol{\alpha}_t) \left(\mathbf{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\top\right) + \beta_t \, \boldsymbol{v}_t \boldsymbol{k}_t^\top$

Each cell is an instance of Equation (1) with the decay and write rule of Table 3. Appendix D.1 gives the quantized chunkwise forms. QK normalization applies ℓ_2 normalization to $\boldsymbol{q}_t$ and $\boldsymbol{k}_t$ independently within each head.

Every model is trained from scratch on SlimPajama using FLA and a Llama tokenizer, with 15B tokens for the 340M models and 100B for the 1.3B models. All runs use sequence length 2048 and AdamW with peak learning rate 4×10^{-4}, 1024 warm-up steps, and cosine decay to one tenth of the peak. We clip gradients at 1.

Quantization. Quantization is symmetric, with static scales calibrated by the maximum absolute activation over eight WikiText-2 training windows of 2048 tokens. Contractions take INT8 operands with INT32 accumulation, an FP32 state is carried between chunks, and reads consume an INT8 copy with one scale per head and value channel, calibrated on the carry. We call this copy the *stored state*, and vary operand precision and stored-state precision separately. INT6 and INT4 configurations narrow the operand code range within the same boundaries and kernels. The stored state remains INT8 unless stated otherwise. Chunk size is 64, with 16-token sub-blocks for diagonal decay, as in FLA.

Evaluation. We evaluate perplexity on WikiText-2 test at sequence length 2048. Transformation parameters are selected using cross-entropy on separate WikiText-2 samples. Excess NLL is $\log(\mathrm{PPL}_{\mathrm{quant}}/\mathrm{PPL}_{\mathrm{FP}})$, and NLL recovered is the difference between the naive quantizer's NLL and that of the transformed configuration. We evaluate zero-shot accuracy with the Language Model Evaluation Harness (Sutawika et al., 2024) on LAMBADA, HellaSwag, PIQA, ARC-Easy, ARC-Challenge, and WinoGrande.

Transformation selection. We use the diagonal rescaling parameterizations of Section 3.2 and sweep $\kappa \in \{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$, including both $\mathbf{D}_{QK}$ alone and its composition with $\mathbf{R}_{QK}$ on linear-write cells. The full $\mathbf{D}_V$ sweep is evaluated under shared value scales.

For decaying cells, we sweep $\eta \in \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$. Linear write-factor absorption uses $\xi \in \{1, \frac{1}{2}, 0\}$, while delta cells use $(\sigma, \tau) \in \{1, \frac{1}{2}, 0\}^2$.

We form the *stack* by combining transformations whose selected settings improve held-out cross-entropy by at least 0.003 nats over the naive quantizer. We use canonical state storage unless stated otherwise, and retain the naive quantizer if no transformation qualifies.

Timing. We compare the transformation stack with canonical $\mathbf{R}_V$ against naive INT8 on the A5000, A100, and H100 across batch sizes, with $T = 2048$, $H = 8$, and $D_k = D_v = 128$. We report ratios of median call times after autotuning and warm-up. Both calls include gate processing, operand quantization, and recurrent computation, including preparation for delta cells. Forward rotations performed before the call are excluded.

B.2 PRODUCTION SETTING

We evaluate pretrained Mamba-2 and Nemotron-H models using SSDi8's W4A8 projection and INT8-recurrence pipeline. Unless otherwise stated, the matched baseline uses value-side decay absorption, static per-(H, D_v) value scales, per-group INT8 keys and queries, and an INT8 fixed-point state pass.

Calibration. The controlled study calibrates activations on WikiText-2 (Appendix B.1). In production, static activation scales and $\mathbf{D}_{QK}$ are calibrated on 512 Pile sequences of 512 tokens. Production weight quantization uses 128 WikiText-2 training samples with GPTQ, after fusing the projection-level Hadamard transformations and before finalizing activation scales. Each layer uses maximum absolute activation measurements with the same scale axes as the quantized kernel (Algorithm 3). In the recurrent core, we observe the raw operands $\mathbf{V}, \mathbf{K}, \mathbf{Q}$, decay-weighted operands $\overrightarrow{\mathbf{\Gamma}}_{[i]} \odot \mathbf{V}_{[i]}, \overrightarrow{\mathbf{\Gamma}}_{[i]} \odot \mathbf{K}_{[i]}$, local state contribution $\mathbf{H}$, and chunk product $\mathbf{Z} = \mathbf{Q}\mathbf{K}^\top$.

For $\mathbf{D}_{QK}$, we measure channel maxima of the Mamba-2 key and query streams, conventionally denoted $\mathbf{B}$ and $\mathbf{C}$, after convolution and SiLU. Rescaling is applied at this same point during inference. Moving it before the nonlinear preprocessing would not preserve the function.

Evaluation. We evaluate WikiText-2 test perplexity at sequence length 2048, using the standard tokenizer for each model: EleutherAI/gpt-neox-20b for Mamba-2 1.3B and 2.7B, and NVIDIA SentencePiece for Mamba-2 8B. Downstream evaluation uses the Language Model Evaluation Harness (Sutawika et al., 2024) on SSDi8's six zero-shot tasks: LAMBADA, HellaSwag, PIQA, ARC-Easy, ARC-Challenge, and WinoGrande. For Mamba-2, rows with static-(H, D_v) value-side scales run without SSDi8's output-projection mean correction, as do the tables marked "no compensation"; the remaining Mamba-2 rows apply it with step factor 0.15, and the Nemotron-H rows with 0.1.

Timing. Unless otherwise indicated, latency measurements use an NVIDIA RTX A5000 with batch size $B = 8$ and sequence length $L = 2048$. For component measurements, we instantiate a single Mamba-2 mixer layer with random weights and time each kernel or sub-operation using CUDA events. Each run uses 30 warm-up forward passes for Triton autotuning and allocation, followed by 200 measured forward passes. We synchronize the stream after each measured forward pass, sum calls belonging to the same stage, and report the median stage latency in microseconds.

C THE SEVEN INVARIANCES: STATEMENTS AND PROOFS

We prove the invariances of Section 3 for Equation (1) in exact arithmetic, with positive decay and write gates and consistently transformed initial states. Table 1 summarizes their applicability.

(1/7) Key-axis rotation $\mathbf{R}_{QK}$. Let $\mathbf{R} = \mathbf{R}_{QK}$ be orthogonal and commute with every $\mathbf{\Lambda}_t$. With $\boldsymbol{q}'_t = \mathbf{R}\boldsymbol{q}_t$, $\boldsymbol{k}'_t = \mathbf{R}\boldsymbol{k}_t$, and $\mathbf{S}'_{t-1} = \mathbf{S}_{t-1}\mathbf{R}^\top$, the update and readout satisfy

$$\begin{aligned} \mathbf{S}'_t &= \big(\mathbf{S}_{t-1}\mathbf{\Lambda}_t(\mathbf{I} - \delta\beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\top) + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top\big)\mathbf{R}^\top = \mathbf{S}_t\mathbf{R}^\top, \\ \mathbf{S}'_t\boldsymbol{q}'_t &= \mathbf{S}_t\boldsymbol{q}_t. \end{aligned}$$

Commutation is automatic for absent or scalar decay. For fixed diagonal decay, $\mathbf{R}$ must preserve the common decay eigenspaces. Transforming the decay itself would generally make it dense. Local rotations of both decay-weighted score operands remain valid and leave the carried state unchanged. A global key rotation preserves the state quantizer's scale axes but can change its calibrated maxima.

(2/7) Key-axis rescaling $\mathbf{D}_{QK}$. Let $\mathbf{D} = \mathbf{D}_{QK} \succ 0$ be diagonal, with $\boldsymbol{q}'_t = \mathbf{D}\boldsymbol{q}_t$, $\boldsymbol{k}'_t = \mathbf{D}^{-1}\boldsymbol{k}_t$, and $\mathbf{S}'_{t-1} = \mathbf{S}_{t-1}\mathbf{D}^{-1}$. Since $\mathbf{D}$ commutes with diagonal decay, a linear write gives

$$\mathbf{S}'_t = (\mathbf{S}_{t-1}\mathbf{\Lambda}_t + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\top)\mathbf{D}^{-1} = \mathbf{S}_t\mathbf{D}^{-1}, \qquad \mathbf{S}'_t\boldsymbol{q}'_t = \mathbf{S}_t\boldsymbol{q}_t.$$

For a delta write, matching the erasure terms requires $\mathbf{D}^{-2}\boldsymbol{k}_t\boldsymbol{k}_t^\top\mathbf{D}^{-1} = \boldsymbol{k}_t\boldsymbol{k}_t^\top\mathbf{D}^{-1}$, or $\mathbf{D}^2\boldsymbol{k}_t = \boldsymbol{k}_t$, excluding a general rescaling. The transformation acts after any QK normalization.

(3/7–4/7) Value-axis rotation and rescaling. For any invertible $\mathbf{T}$, set $\boldsymbol{v}'_t = \mathbf{T}\boldsymbol{v}_t$ and $\mathbf{S}'_{t-1} = \mathbf{T}\mathbf{S}_{t-1}$. For either write rule and any diagonal decay,

$$\mathbf{S}'_t = \mathbf{T}\big(\mathbf{S}_{t-1}\mathbf{\Lambda}_t(\mathbf{I} - \delta\beta_t\boldsymbol{k}_t\boldsymbol{k}_t^\mathsf{T}) + \beta_t\boldsymbol{v}_t\boldsymbol{k}_t^\mathsf{T}\big) = \mathbf{T}\mathbf{S}_t, \qquad \mathbf{T}^{-1}\boldsymbol{o}'_t = \boldsymbol{o}_t.$$

Choose $\mathbf{T} = \mathbf{R}_V$ or $\mathbf{T} = \mathbf{D}_V$. Canonical storage undoes $\mathbf{T}$ on each chunk's write and local output. Delta pseudo-values additionally require the incoming-state correction in the transformed value basis. With per-channel scales, diagonal rescaling cancels because $(d_p v_p)/(d_p s_p) = v_p/s_p$, where s_p is the channel scale. Consistently transformed scales therefore leave quantization unchanged, apart from finite-precision rounding.

(5/7–7/7) Decay and linear-write absorptions. All three follow from the token-wise identity

$$c^r_{[i]}\boldsymbol{v}^r_{[i]}(\boldsymbol{k}^r_{[i]})^\mathsf{T} = \big((c^r_{[i]})^\eta\boldsymbol{v}^r_{[i]}\big)\big((c^r_{[i]})^{1-\eta}\boldsymbol{k}^r_{[i]}\big)^\mathsf{T}, \qquad c^r_{[i]} > 0.$$

Apply it to each chunk-write term with $c^r_{[i]}$ equal to the scalar tail decay, the diagonal-decay magnitude $e^r_{[i]}$, or the write gate $\beta^r_{[i]}$. Other factors retain their placement. In particular, the relative channel-dependent decay remains on the key. For delta-rule decay absorption, use the corresponding pseudo-values in place of gated values.

(7/7) Delta write-factor absorption. Using the interval weights of Section 2.2, define the strictly lower-triangular Gram matrix

$$C_{[i],rs} = \mathbf{1}_{s<r}\sum_n K_{[i],rn}K_{[i],sn}\gamma^{s\to r}_{[i],n}.$$

Scalar decay is broadcast across channels. Without decay, $\gamma \equiv 1$. The gate allocation is a diagonal similarity transformation:

$$\begin{aligned}\mathbf{L}^{(\sigma)}_{[i]} &= \mathbf{B}^\sigma_{[i]}\mathbf{C}_{[i]}\mathbf{B}^{1-\sigma}_{[i]} = \mathbf{B}^{\sigma-1}_{[i]}\mathbf{L}^{(1)}_{[i]}\mathbf{B}^{1-\sigma}_{[i]},\\ (\mathbf{I}+\mathbf{L}^{(\sigma)}_{[i]})^{-1}\mathbf{B}^\sigma_{[i]} &= \mathbf{B}^{\sigma-1}_{[i]}(\mathbf{I}+\mathbf{L}^{(1)}_{[i]})^{-1}\mathbf{B}_{[i]}.\end{aligned}$$

Writing $\mathbf{F}_{[i]} := \mathbf{F}^{(1)}_{[i]}$, the complete pseudo-values satisfy

$$\mathbf{F}^{(\sigma)}_{[i]} = (\mathbf{I}+\mathbf{L}^{(\sigma)}_{[i]})^{-1}\mathbf{B}^\sigma_{[i]}\big(\mathbf{V}_{[i]} - (\mathbf{\Gamma}^{\leftarrow}_{[i]}\odot\mathbf{K}_{[i]})\mathbf{S}^\mathsf{T}_{[i]}\big) = \mathbf{B}^{\sigma-1}_{[i]}\mathbf{F}_{[i]}.$$

For the transformed operands of (7/7), the remaining gate powers sum to $1-\sigma$, giving

$$\begin{aligned}\big(\mathbf{V}^{(\sigma,\tau)}_{[i]}\big)^\mathsf{T}(\mathbf{\Gamma}^{\rightarrow}_{[i]}\odot\mathbf{K}^{(\sigma,\tau)}_{[i]}) &= \big(\mathbf{F}^{(\sigma)}_{[i]}\big)^\mathsf{T}\mathbf{B}^{1-\sigma}_{[i]}(\mathbf{\Gamma}^{\rightarrow}_{[i]}\odot\mathbf{K}_{[i]})\\ &= \mathbf{F}^\mathsf{T}_{[i]}(\mathbf{\Gamma}^{\rightarrow}_{[i]}\odot\mathbf{K}_{[i]}).\end{aligned}$$

The same token-wise cancellation preserves the local output. The incoming-state read is unchanged. At $(\sigma,\tau) = (1,1)$, this recovers the standard WY form (Yang et al., 2024c; 2025; Yang & Zhang, 2024).

D Low-precision realizations and kernel implementation

This appendix describes the quantized chunkwise forms, the controlled and production kernels, and the placement and fusion of transformations.

D.1 Quantized chunkwise forms

Section 3.1 gives the scalar-decay linear recurrence underlying Mamba-2. The other five families retain its state write, state pass, state read, local score, and intra-chunk output, but form their operands differently. Throughout this subsection, $\mathcal{Q}$ denotes quantization followed by dequantization at the indicated boundary. The write, read, and local-score contractions use INT8 codes with INT32 accumulation. The intra-chunk output uses FP16 arithmetic on INT8-coded values.

Decay factors. We use the interval weights $\gamma^{s\to r}_{[i]}$ and stacked prefix and tail weights $\mathbf{\Gamma}^{\leftarrow}_{[i]}, \mathbf{\Gamma}^{\rightarrow}_{[i]}$ from Section 2.2.

No decay, linear write (LA). Setting $\gamma \equiv 1$ removes all decay factors. With an optional write gate, the chunk computation becomes

$$\begin{aligned}
\widehat{\mathbf{H}}_{[i]} &= \mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]})^\top \mathcal{Q}(\mathbf{K}_{[i]}),\\
\mathbf{S}_{[i+1]} &= \mathbf{S}_{[i]} + \widehat{\mathbf{H}}_{[i]}, \qquad \widehat{\mathbf{S}}_{[i+1]} = \mathcal{Q}(\mathbf{S}_{[i+1]}),\\
\widehat{\mathbf{O}}^{\text{inter}}_{[i]} &= \mathcal{Q}(\mathbf{Q}_{[i]})\widehat{\mathbf{S}}^\top_{[i]},\\
\widehat{\mathbf{A}}_{[i]} &= \big(\mathcal{Q}(\mathbf{Q}_{[i]})\mathcal{Q}(\mathbf{K}_{[i]})^\top\big) \odot \mathbf{M},\\
\widehat{\mathbf{O}}^{\text{intra}}_{[i]} &= \widehat{\mathbf{A}}_{[i]}\mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]}).
\end{aligned}$$

Diagonal decay, linear write (GLA). For positive decay gates, the masked product in Section 2.2 expands as

$$\left[\operatorname{tril}\left((\boldsymbol{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]})\left(\frac{\mathbf{K}_{[i]}}{\boldsymbol{\Gamma}^{\leftarrow}_{[i]}}\right)^\top\right)\right]_{rs} = M_{rs}\sum_n Q_{[i],rn}K_{[i],sn}\gamma^{s\to r}_{[i],n}. \tag{2}$$

The right-hand side also applies when decay gates are zero. Quantizing the contraction operands for positive gates gives

$$\begin{aligned}
\widehat{\mathbf{H}}_{[i]} &= \mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]})^\top \mathcal{Q}(\boldsymbol{\Gamma}^{\rightarrow}_{[i]} \odot \mathbf{K}_{[i]}),\\
\mathbf{S}_{[i+1]} &= \mathbf{S}_{[i]}\operatorname{diag}(\gamma^C_{[i],1},\dots,\gamma^C_{[i],D_k}) + \widehat{\mathbf{H}}_{[i]}, \qquad \widehat{\mathbf{S}}_{[i+1]} = \mathcal{Q}(\mathbf{S}_{[i+1]}),\\
\widehat{\mathbf{O}}^{\text{inter}}_{[i]} &= \mathcal{Q}(\boldsymbol{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{Q}_{[i]})\widehat{\mathbf{S}}^\top_{[i]},\\
\widehat{A}_{[i],rs} &= \sum_n \mathcal{Q}\big(Q_{[i],rn}\gamma^r_{[i],n}/\gamma^a_{[i],n}\big)\,\mathcal{Q}\big(K_{[i],sn}\gamma^a_{[i],n}/\gamma^s_{[i],n}\big),\\
\widehat{\mathbf{O}}^{\text{intra}}_{[i]} &= \widehat{\mathbf{A}}_{[i]}\mathcal{Q}(\mathbf{B}_{[i]}\mathbf{V}_{[i]}).
\end{aligned} \tag{3}$$

where the displayed local score applies to causal pairs $s \le r$. Entries above the causal diagonal are zero. The decay lies inside the reduction over channels, as in Equation (2). For cross-block scores, the kernel anchors the two decay factors at a position a in the block containing s, then quantizes and multiplies the anchored operands. Within a block, it retains the raw codes and applies $\gamma^{s\to r}_{[i],n}$ inside the FP32 sum. A query–key rotation can act on the anchored operands in registers, but cannot generally be applied once to the unweighted $\mathbf{Q}$ and $\mathbf{K}$ (Section 3).

Delta write (DeltaNet, GDN, KDA). Delta cells first construct pseudo-values that account for interactions among writes and with the incoming state. The Gram matrix, the triangular solve, and the WY intermediates are computed in FP32; only the incoming-state correction uses INT8 codes of $\mathbf{W}_{[i]}$ against the INT8 copy of the incoming state. For absent or scalar decay, let $\boldsymbol{\Gamma}_{[i]}$ denote the causal decay mask. The compensated write-factor placement (σ,τ) gives

$$\begin{aligned}
\mathbf{G}_{[i]} &= \operatorname{tril}\Big(\big((\mathbf{B}^{\sigma}_{[i]}\mathbf{K}_{[i]})\,(\mathbf{B}^{1-\sigma}_{[i]}\mathbf{K}_{[i]})^\top\big) \odot \boldsymbol{\Gamma}_{[i]}, -1\Big), \qquad \mathbf{T}_{[i]} = (\mathbf{I}+\mathbf{G}_{[i]})^{-1},\\
\mathbf{W}_{[i]} &= \mathbf{T}_{[i]}\mathbf{B}^{\sigma}_{[i]}(\boldsymbol{\Gamma}^{\leftarrow}_{[i]} \odot \mathbf{K}_{[i]}), \qquad \mathbf{U}_{[i]} = \mathbf{T}_{[i]}\,\mathbf{B}^{\sigma}_{[i]}\mathbf{V}_{[i]},\\
\widehat{\mathbf{F}}^{(\sigma)}_{[i]} &= \mathbf{U}_{[i]} - \mathcal{Q}(\mathbf{W}_{[i]})\widehat{\mathbf{S}}^\top_{[i]},\\
\mathbf{V}^{(\sigma,\tau)}_{[i]} &= \mathbf{B}^{(1-\sigma)(1-\tau)}_{[i]}\widehat{\mathbf{F}}^{(\sigma)}_{[i]}, \qquad \mathbf{K}^{(\sigma,\tau)}_{[i]} = \mathbf{B}^{(1-\sigma)\tau}_{[i]}\mathbf{K}_{[i]},
\end{aligned}$$

Here $\mathbf{U}_{[i]}, \mathbf{W}_{[i]}$ are WY intermediates, and $\widehat{\mathbf{F}}^{(\sigma)}_{[i]}$ approximates the complete pseudo-values $\mathbf{F}^{(\sigma)}_{[i]}$ using the quantized incoming-state correction. The transformed write operands $\mathbf{V}^{(\sigma,\tau)}_{[i]}, \mathbf{K}^{(\sigma,\tau)}_{[i]}$ enter

the chunk step:

$$\begin{aligned}
\widehat{\mathbf{H}}_{[i]} &= \mathcal{Q}(\mathbf{V}_{[i]}^{(\sigma,\tau)})^\top \mathcal{Q}(\boldsymbol{\Gamma}_{[i]}^{\rightarrow} \odot \mathbf{K}_{[i]}^{(\sigma,\tau)}),\\
\mathbf{S}_{[i+1]} &= \mathbf{S}_{[i]}\mathbf{P}_{[i]} + \widehat{\mathbf{H}}_{[i]}, \qquad \widehat{\mathbf{S}}_{[i+1]} = \mathcal{Q}(\mathbf{S}_{[i+1]}),\\
\widehat{\mathbf{O}}_{[i]}^{\text{inter}} &= \mathcal{Q}(\boldsymbol{\Gamma}_{[i]}^{\leftarrow} \odot \mathbf{Q}_{[i]})\widehat{\mathbf{S}}_{[i]}^\top,\\
\widehat{\mathbf{A}}_{[i]} &= \left(\mathcal{Q}(\mathbf{Q}_{[i]})\mathcal{Q}(\mathbf{K}_{[i]}^{(\sigma,\tau)})^\top\right) \odot \boldsymbol{\Gamma}_{[i]},\\
\widehat{\mathbf{O}}_{[i]}^{\text{intra}} &= \widehat{\mathbf{A}}_{[i]}\mathcal{Q}(\mathbf{V}_{[i]}^{(\sigma,\tau)}),
\end{aligned}$$

where $\mathbf{P}_{[i]}$ is $\mathbf{I}$ for DeltaNet, $\gamma_{[i]}^{C}\mathbf{I}$ for GDN, and $\operatorname{diag}(\gamma_{[i],1}^{C}, \ldots, \gamma_{[i],D_k}^{C})$ for KDA. For KDA, both the Gram and local-score contractions place the channel-dependent decay inside the reduction, the local score using the anchored construction of Equation (3) in place of a scalar mask applied after the product. At $(\sigma, \tau) = (1, 1)$, with no decay and quantization removed, $\mathbf{V}_{[i]}^{(\sigma,\tau)} = \mathbf{F}_{[i]}$ recovers the pseudo-values in Section 2.2. Setting $\sigma = \frac{1}{2}$ instead assigns $\mathbf{B}_{[i]}^{1/2}$ to each Gram operand.

Absorptions. For a linear write with scalar decay, the tail weights $\boldsymbol{\Gamma}_{[i]}^{\rightarrow}$ and write gate $\mathbf{B}_{[i]}$ can each be split between the two operands before quantization. With elementwise powers of the tail vector and broadcasting across channels, the write is

$$\mathcal{Q}\Big((\boldsymbol{\Gamma}_{[i]}^{\rightarrow})^{\eta} \odot (\mathbf{B}_{[i]}^{\xi}\mathbf{V}_{[i]})\Big)^\top \mathcal{Q}\Big((\boldsymbol{\Gamma}_{[i]}^{\rightarrow})^{1-\eta} \odot (\mathbf{B}_{[i]}^{1-\xi}\mathbf{K}_{[i]})\Big).$$

Here $\eta = 1$ and $\xi = 1$ put the respective factors on the value, while zero puts them on the key. Under diagonal decay, only the scalar maximum $e_{[i]}^{r} = \max_n (\boldsymbol{\Gamma}_{[i]}^{\rightarrow})_{rn}$ moves. The relative channel-dependent decay remains on the key. Delta cells use pseudo-values for decay absorption and the compensated (σ, τ) construction for write strength, as proved in Appendix C.

D.2 Kernel implementation

Controlled study. Diagonal rescalings can be incorporated into operand scaling, and absorptions redistribute token-wise factors already used by the write. Algorithm 1 lists one chunk step and marks where each invariance enters.

Algorithm 1 One INT8 chunk kernel and where each invariance enters (per head and chunk. $\mathcal{Q}_b$ quantizes to b-bit codes with the folded scales; the local path and the write quantize their operands separately. $\hat{\mathbf{S}}_{[i]}$ is the INT8 copy of the incoming state, $\boldsymbol{g}$ the tail weights, $\boldsymbol{p}$ the prefix weights, and $\boldsymbol{\Gamma}$ the causal decay mask).

1: once: $\boldsymbol{s}_k \leftarrow \boldsymbol{s}_k \oslash \boldsymbol{d}_{qk}$, $\boldsymbol{s}_q \leftarrow \boldsymbol{s}_q \odot \boldsymbol{d}_{qk}$, $\boldsymbol{s}_v \leftarrow \boldsymbol{s}_v \odot \boldsymbol{d}_v$ ▷ $\mathbf{D}_{qk}, \mathbf{D}_v$: folded into the scale vectors
2: once: $\mathbf{Q} \leftarrow \mathbf{Q}\mathbf{R}_{QK}$, $\mathbf{K} \leftarrow \mathbf{K}\mathbf{R}_{QK}$ if the decay commutes. $\mathbf{V} \leftarrow \mathbf{V}\mathbf{R}_V$ ▷ fused with the quantization pass
3: **if** $\delta = 0$ **then**
4: $\quad \boldsymbol{w}_k \leftarrow \mathbf{B}_{[i]}^{1-\xi}\boldsymbol{g}^{1-\eta}$, $\boldsymbol{w}_v \leftarrow \mathbf{B}_{[i]}^{\xi}\boldsymbol{g}^{\eta}$ ▷ linear write
5: **else**
6: $\quad (\mathbf{K}, \mathbf{V}) \leftarrow (\mathbf{K}_{[i]}^{(\sigma,\tau)}, \mathbf{V}_{[i]}^{(\sigma,\tau)})$ from Algorithm 2 ▷ gate already included
7: $\quad \boldsymbol{w}_k \leftarrow \boldsymbol{g}^{1-\eta}$, $\boldsymbol{w}_v \leftarrow \boldsymbol{g}^{\eta}$ ▷ decay allocation
8: **end if**
9: $\mathbf{O} \leftarrow \boldsymbol{p} \odot \operatorname{deq}\left(\mathcal{Q}_b(\mathbf{Q})\hat{\mathbf{S}}_{[i]}^{\mathsf{T}}\right)$ ▷ read of the incoming state on INT8. Prefix decay after the product (inside the operand under diagonal decay)
10: $\mathbf{A} \leftarrow \operatorname{deq}\left(\mathcal{Q}_b(\mathbf{Q})\mathcal{Q}_b(\mathbf{K})^{\mathsf{T}}\right) \odot \boldsymbol{\Gamma}$ ▷ local score on INT8, raw codes. Decay in the fp32 mask
11: $\mathbf{O} \leftarrow \mathbf{O} + \mathbf{A}\,\mathcal{Q}_b(\mathbf{V})$. For canonical $\mathbf{R}_V$, un-rotate the local product. ▷ intra-chunk output, FP16 on codes
12: $\hat{\mathbf{K}}, \hat{\mathbf{V}} \leftarrow \mathcal{Q}_b(\boldsymbol{w}_k \odot \mathbf{K}),\ \mathcal{Q}_b(\boldsymbol{w}_v \odot \mathbf{V})$ ▷ write operands quantized in the call
13: $\mathbf{H} \leftarrow \operatorname{deq}(\hat{\mathbf{V}}^{\mathsf{T}}\hat{\mathbf{K}})$. Canonical $\mathbf{R}_V$: $\mathbf{H} \leftarrow \mathbf{R}_V^{\mathsf{T}}\mathbf{H}$ ▷ write on INT8. The un-rotation is the one added cost
14: $\mathbf{S} \leftarrow \mathbf{S}\boldsymbol{\Gamma}^{C} + \mathbf{H}$. $\hat{\mathbf{S}}_{[i+1]} \leftarrow \mathcal{Q}_8(\mathbf{S})$ ▷ fp32 carry. INT8 stored copy read by the next chunk

The recurrence determines how those operands are formed. Decay supplies the write's tail factors and the read's prefix factors: unity for LA and DeltaNet, scalars for Mamba-2 and GDN, and channel-dependent vectors for GLA and KDA. Under diagonal decay, query–key rotations are restricted to the

decay-weighted score operands. The write rule determines whether the kernel uses direct keys and values or the transformed write operands constructed by Algorithm 2. An additional linear write gate supplies the factor redistributed by write-strength absorption.

Canonical storage under $\mathbf{R}_V$ adds matrix products inside the call. Each chunk's rotated write is multiplied by $\mathbf{R}_V^\top$ before it joins the FP32 carry, and the local output is returned to the original basis.

The state read stays canonical. Delta cells also rotate the incoming-state correction $\mathcal{Q}(\mathbf{W}_{[i]})\widehat{\mathbf{S}}_{[i]}^\top$ by $\mathbf{R}_V$ before subtracting it from $\mathbf{U}_{[i]}$. Rotated storage needs only one inverse transform of the chunk output, but incurs the accuracy penalty discussed in Appendix E.1. The canonical delta kernels hold the full state and the $D_v \times D_v$ rotation operand per program.

Algorithm 2 WY preparation with compensated write-factor placement (σ, τ), per head and chunk. $\mathbf{M}_<$ is the strictly lower causal mask, including scalar decay when present. Diagonal decay is applied inside the Gram reduction as described above. The Gram matrix, solve, and WY intermediates are FP32; $\widehat{\mathbf{S}}_{\text{in}}$ is the INT8-quantized incoming state and $\mathbf{W}_{[i]}$ is the only quantized operand of the preparation. The resulting transformed write operands are used in the chunk step.

$\mathbf{L}_{[i]} \leftarrow \mathbf{B}_{[i]}^{\sigma}\mathbf{K}_{[i]}$, $\mathbf{G}_{[i]} \leftarrow \mathbf{B}_{[i]}^{1-\sigma}\mathbf{K}_{[i]}$ ▷ Gram factors
$\mathbf{A}_{[i]} \leftarrow (\mathbf{L}_{[i]}\,\mathbf{G}_{[i]}^{\mathsf{T}}) \odot \mathbf{M}_<$ ▷ Gram matrix, fp32
$\mathbf{T}_{[i]} \leftarrow (\mathbf{I} + \mathbf{A}_{[i]})^{-1}$ ▷ unit lower-triangular solve, fp32
$\mathbf{W}_{[i]} \leftarrow \mathbf{T}_{[i]}(\mathbf{L}_{[i]} \odot \mathbf{\Gamma}_{[i]}^{\leftarrow})$, $\mathbf{U}_{[i]} \leftarrow \mathbf{T}_{[i]}\,\mathbf{B}_{[i]}^{\sigma}\mathbf{V}_{[i]}$ ▷ WY intermediates, fp32
$\widehat{\mathbf{F}}_{[i]}^{(\sigma)} \leftarrow \mathbf{U}_{[i]} - \mathcal{Q}_b(\mathbf{W}_{[i]})\,\widehat{\mathbf{S}}_{\text{in}}^{\mathsf{T}}$ ▷ complete pseudo-values
$\mathbf{K}_{[i]}^{(\sigma,\tau)} \leftarrow \mathbf{B}_{[i]}^{(1-\sigma)\tau}\mathbf{K}_{[i]}$, $\mathbf{V}_{[i]}^{(\sigma,\tau)} \leftarrow \mathbf{B}_{[i]}^{(1-\sigma)(1-\tau)}\widehat{\mathbf{F}}_{[i]}^{(\sigma)}$ ▷ transformed write operands
return $\mathbf{K}_{[i]}^{(\sigma,\tau)}$, $\mathbf{V}_{[i]}^{(\sigma,\tau)}$ for Algorithm 1

Production Mamba-2. The production implementation adds state-axis transformations to SSDi8's dataflow, as summarized in Algorithm 3. The added $\mathbf{D}_{qk}$ rescaling is incorporated into operand scaling, and $\mathbf{R}_{QK}$ is applied while splitting and quantizing the convolution output. Neither transformation requires a separate kernel launch or memory pass in this implementation.

Algorithm 3 The Mamba-2 kernel of `LinQuant`: SSDi8's dataflow with $\mathbf{R}_{QK}$ and $\mathbf{D}_{qk}$. Keys and queries are shared by the G groups and read by the H heads. $g(h)$ denotes the group of head h. $\boldsymbol{w}$ denotes the within-chunk decay weights (tail factors for the write, prefix factors for the read), and $\gamma_{[i]}^{C}$ denotes the chunk decay.

Once per layer (the fused pass over the convolution output)
$\mathbf{K} \leftarrow \mathbf{K}\mathbf{D}_{qk}^{-1}\mathbf{R}_{QK}$, $\mathbf{Q} \leftarrow \mathbf{Q}\mathbf{D}_{qk}\mathbf{R}_{QK}$. $\hat{\mathbf{K}} \leftarrow \mathcal{Q}_8(\mathbf{K}, \boldsymbol{s}_K)$, $\hat{\mathbf{Q}} \leftarrow \mathcal{Q}_8(\mathbf{Q}, \boldsymbol{s}_Q)$ ▷ per group. The codes are reused by three stages
Per chunk $[i]$ **and head** h
$\hat{\mathbf{V}} \leftarrow \mathcal{Q}_8(\boldsymbol{w}_{\text{tail}} \odot \mathbf{V}, \boldsymbol{s}_V)$, tile by tile in registers ▷ decay on the value. Static per-(H, D_v) scale or dynamic per tile. Key side: $\hat{\mathbf{K}} \leftarrow \mathcal{Q}_8(\boldsymbol{w}_{\text{tail}} \odot \mathbf{K})$ from an fp re-read instead
$\mathbf{H} \leftarrow \hat{\mathbf{V}}^{\mathsf{T}}\hat{\mathbf{K}}_{g(h)}$ ▷ write: INT8×INT8, INT32 accumulation, one rescale
$\hat{\mathbf{S}}_{[i+1]} \leftarrow \text{fx}(\gamma_{[i]}^{C}\hat{\mathbf{S}}_{[i]}) + \mathcal{Q}_8(\mathbf{H}, \boldsymbol{s}_S)$ ▷ state pass in INT8 fixed point, per-(H, D_v) scale
$\hat{\mathbf{Z}} \leftarrow \mathcal{Q}_8(\hat{\mathbf{Q}}_{g(h)}\hat{\mathbf{K}}_{g(h)}^{\mathsf{T}})$ ▷ local score: INT32 accumulation, requantized per group
$\mathbf{O} \leftarrow \boldsymbol{w}_{\text{pref}} \odot \text{deq}(\hat{\mathbf{Q}}_{g(h)}\hat{\mathbf{S}}_{[i]}^{\mathsf{T}}) + (\text{deq}(\hat{\mathbf{Z}}) \odot \mathbf{\Gamma})\mathbf{V}$ ▷ read on INT8. Intra-chunk output fp16, the scale of $\hat{\mathbf{Z}}$ folded into $\mathbf{\Gamma}$

Transformation placement and rotation fusion. We implement the online rotations using structured signed Hadamard transforms. Forward rotations are fused into kernels that already materialize the activations. The query–key rotation joins the split-and-quantize kernel after convolution. The value rotation joins value quantization. The inverse value rotation must restore the original basis before an incompatible downstream operation. In our Mamba-2 implementation, the gated RMSNorm after the recurrence normalizes across the full hidden dimension. We therefore apply the per-head inverse rotation at the end of the recurrence-output kernel, before this normalization.

Applying decay in registers. Keys are shared by groups, while decay weights can differ across heads. A direct implementation of key-side absorption could therefore materialize a larger, head-indexed tensor of weighted keys. Our write kernel avoids this intermediate tensor. Each program loads its group's key tile and its head's decay weights, then forms the weighted entries in registers:

$$k'_{r,n} = w_{h,r}\, k_{g(h),r,n},$$

The weighted tile is quantized and clipped before the INT8 tensor-core contraction. Programs for heads in the same group read the same underlying key tile and apply their own weights, so head-specific values never need to be stored as an expanded floating-point tensor.

E CONTROLLED-STUDY RESULTS

This appendix reports the controlled-study results for Section 4.1, using the setup in Appendix B.1. We organize them by key-axis transformations (1/7–2/7), value-axis transformations (3/7–4/7), and absorptions (5/7–7/7), followed by combined transformations and full quality tables.

E.1 INDIVIDUAL TRANSFORMATIONS

Key-axis transformations (1/7)-(2/7). Figure 6 shows the response to query–key rescaling, with separate panels for cells and curves for scale regimes. The query-key rescaling sweep tests both range balancing and its composition with rotation.

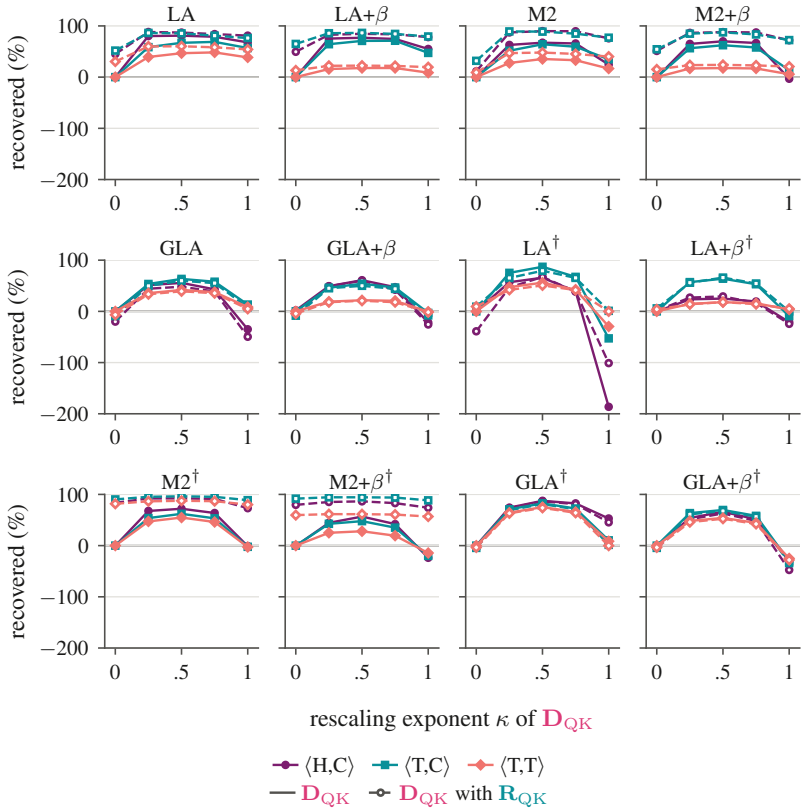


Figure 6: Response to the exponent κ of $\mathbf{D}_{QK}$, alone (solid) and composed with $\mathbf{R}_{QK}$ (dashed, hollow), per linear cell and regime, INT6 codes, INT8 stored state: percent of the naive excess NLL recovered. At $\kappa = 0$ the dashed curve is $\mathbf{R}_{QK}$ alone.

Value-axis transformations (3/7)-(4/7). Figure 7 shows the value-rescaling sweep, with separate panels for cells and curves for scale regimes. The value-rescaling sweep isolates the role of scale granularity: consistent per-channel scales absorb the rescaling, while shared scales allow it to change quantization error.

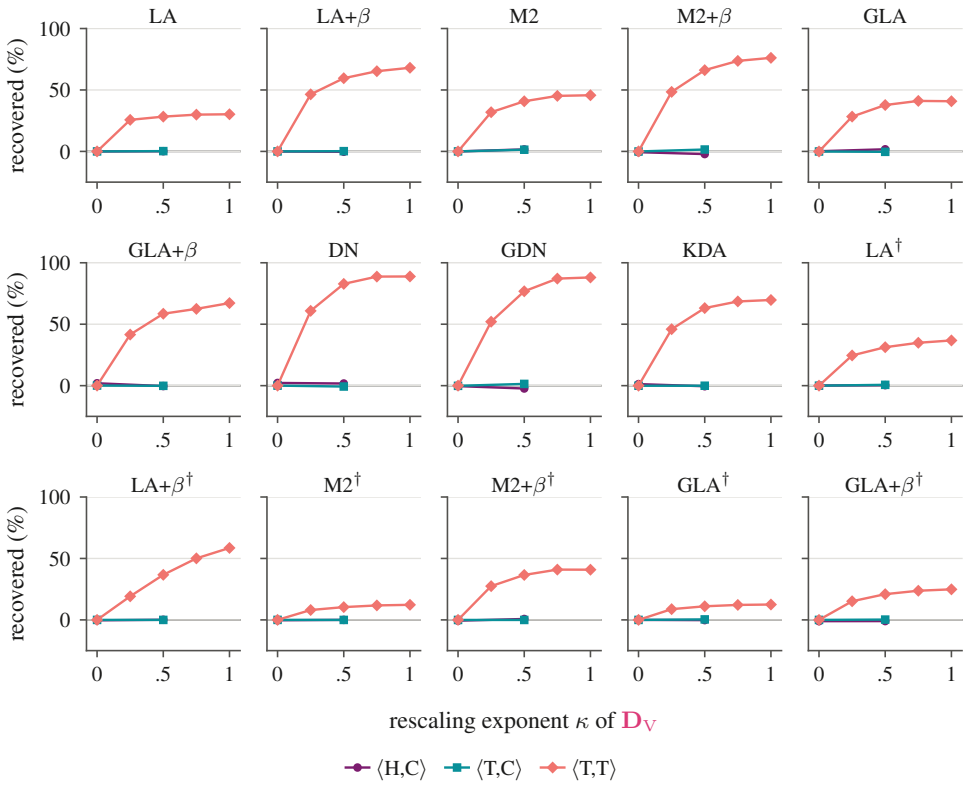


Figure 7: Response to the exponent κ of D_V, percent of the naive excess NLL recovered: per-channel scales absorb a consistent diagonal rescaling, whereas shared scales allow the exponent to change the error. The full grid $\kappa \in \{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$ was run only under coarse scales $\langle \mathrm{T}, \mathrm{T} \rangle$.

State-storage basis. Figure 8 compares canonical and rotated storage for every cell at INT6. The stored state uses one scale per head and value channel. A value rotation mixes these channels before quantization, changing the range covered by each scale. The largest benefit from canonical storage occurs in LA. Decaying and delta-rule cells are much less sensitive, although the sign and size of the difference depend on the configuration.

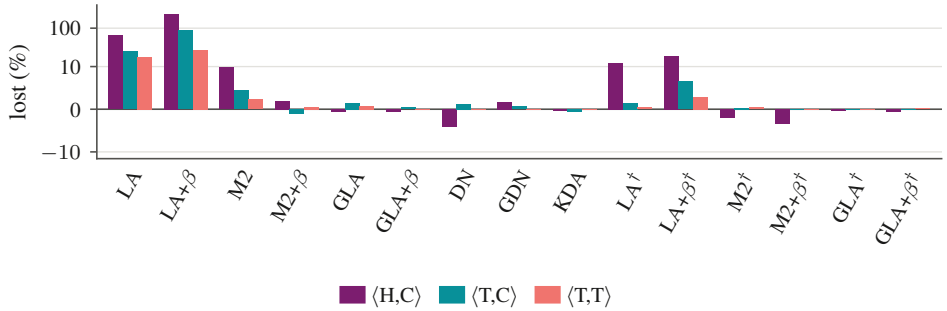


Figure 8: Full storage-basis control for the 340M zoo. Excess NLL of rotated minus canonical storage under R_V, as a percentage of each model's naive excess NLL, per scale regime, INT6 operand codes. Positive values favor canonical storage. Values derive from the rounded perplexities in Appendix E.3.

Canonical value-rotation overhead. Table 4 compares the transformation stack with canonical $\mathbf{R}_V$ against naive INT8 across batch sizes and GPUs, complementing Figure 5b. The timing methodology and boundaries are given in Appendix B.1.

Table 4: Relative INT8 kernel time for the transformation stack with canonical $\mathbf{R}_V$, divided by naive INT8 time (1×: equal time). Ratios use median latencies after autotuning and warm-up at $T = 2048$, $H = 8$, $D_k = D_v = 128$. Timing boundaries are described in Appendix B.1.

	RTX A5000					A100-SXM4-40GB			H100		
Recurrence	$B = 1$	4	8	16	32	$B = 1$	4	8	$B = 1$	4	8
LA	2.36	1.40	1.61	1.54	1.54	2.33	1.38	1.24	2.03	1.96	1.60
Mamba-2	2.78	1.27	1.42	1.35	1.31	2.40	1.58	1.42	1.94	1.91	1.60
GLA	1.72	1.35	1.36	1.37	1.37	2.04	1.64	1.72	1.62	1.78	1.59
DN	2.82	4.13	3.37	3.11	3.16	9.57	14.30	10.56	1.93	1.96	2.43
GDN	2.36	3.62	2.84	2.60	2.69	7.93	12.33	7.90	2.10	2.14	2.27
KDA	2.03	2.24	1.88	1.79	1.79	6.36	8.19	4.99	1.71	1.90	1.77

Absorptions (5/7)-(7/7). The absorption sweeps compare where token-wise factors enter the quantized computation. Scalar-decay placement has relatively small effects in the deployment-oriented regimes. For gated linear writes, the value-side endpoint $\xi = 1$ performs best under per-channel value scales, while coarse scales can favor an interior split. For the delta cells under coarse scales, the selection procedure moves the compensation to the keys ($\tau = 1$) and chooses $\sigma \in \{0, \frac{1}{2}\}$, with small differences between placements. The individual curves make clear how these conclusions depend on the cell and scale regime.

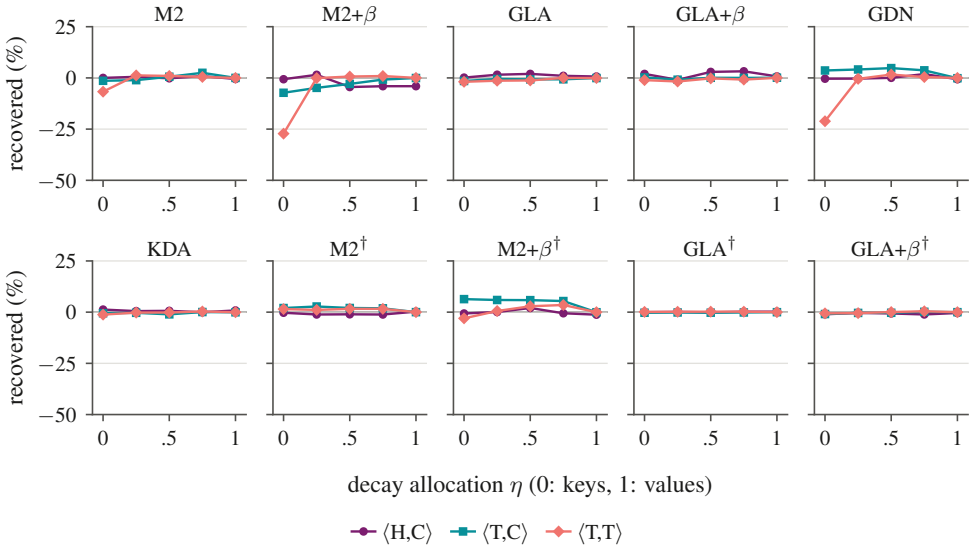


Figure 9: Response to the decay allocation η (decay on the key at 0, on the value at 1) for the decaying cells, INT6 operands and INT8 stored state: percent of the naive excess NLL recovered, one curve per scale regime. Placing scalar decay on the keys raises the excess by 21–27 % on M2+β and GDN under $\langle \mathrm{T}, \mathrm{T} \rangle$. Diagonal decay and the per-channel-value regimes stay within a few percent.

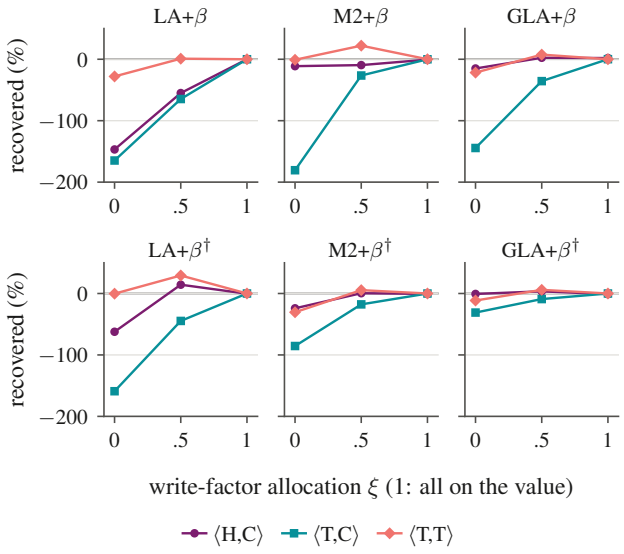


Figure 10: Write-factor allocation on the six gated linear models, with INT6 operands and INT8 stored state: $\mathbf{B}_{[i]}^{\xi}$ multiplies values and $\mathbf{B}_{[i]}^{1-\xi}$ multiplies keys. Percent of the naive excess NLL recovered. Each curve shows one scale regime. † marks models without QK normalization.

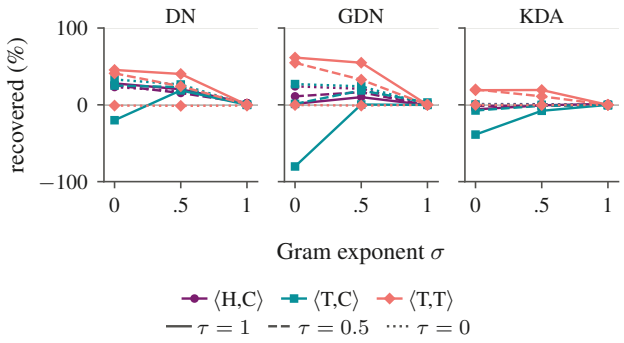


Figure 11: Delta write-factor allocation with INT6 operands and INT8 stored state. Curves vary the Gram exponent σ at fixed τ, which allocates the remaining factor between the transformed write operands: solid $\tau = 1$, dashed $\tau = 1/2$, dotted $\tau = 0$. Percent of the naive excess NLL recovered. Each colour denotes one scale regime.

E.2 Combined transformations and precision controls

Figure 12 compares the two shared-key regimes at INT6. Key scales are held fixed and only value scales change, from per channel on the left to per tensor on the right. State-axis transformations can help in either panel, whereas the larger value-axis gains emerge when values share a scale.

Key and value scale granularity. Value-scale granularity matters more at INT6 than at INT8 (Tables 15 and 16). Coarsening value scales increases INT6 error under both key-scale policies, with DN particularly sensitive. Coarsening key scales instead affects LA most strongly. The decaying cells are less sensitive. With per-head value scales, canonical $\mathbf{R}_V$ largely recovers the quality of per-channel scaling in the decaying cells. Rotated storage retains a penalty in LA, so this recovery depends on the state basis as well as the operand scales. The corresponding per-channel-value baselines are in Tables 13, 14, 9, and 11.

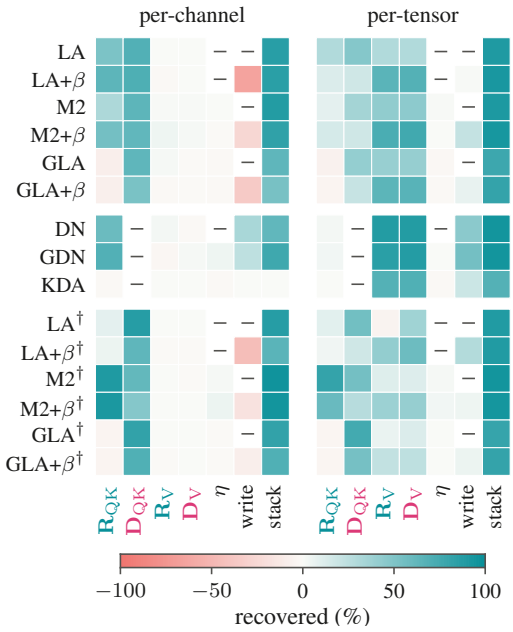


Figure 12: Percentage of the naive quantizer's excess NLL recovered per cell and invariance at its selected setting (negative: the invariance increases the loss), shared keys with per-channel values (left) and with per-tensor values (right).

Table 5 summarizes the results: the stack reduces error under coarse scales, while several INT8 configurations are already close enough to full precision that no transformation passes the selection threshold.

Table 5: WikiText-2 perplexity for full precision, the naive quantizer, and the held-out-selected stack. The INT8 column uses $\langle \mathrm{T}, \mathrm{C} \rangle$, and the INT6 column uses coarse scales $\langle \mathrm{T}, \mathrm{T} \rangle$. A dash means no transformation cleared the promotion threshold.

		INT8, per-tensor keys		INT6, coarse scales	
Cell	FP	Naive	Stack	Naive	Stack
LA	17.12	19.48	17.27	30.69	18.14
LA+β	16.69	17.54	16.80	28.38	17.39
LA, no norm	17.23	17.80	17.39	31.58	18.34
LA+β, no norm	16.98	18.74	17.90	100.25	20.43
Mamba-2, no norm	16.72	16.91	16.73	21.48	16.92
Mamba-2+β, no norm	16.19	16.27	16.19	18.84	16.33
GLA, no norm	16.07	16.28	16.10	20.98	16.91
GLA+β, no norm	15.89	16.01	15.94	20.52	16.65
Mamba-2	15.55	15.61	15.56	16.45	15.63
Mamba-2+β	15.42	15.44	–	16.23	15.47
GLA	15.10	15.12	–	16.05	15.31
GLA+β	15.31	15.32	–	16.05	15.44
DeltaNet	15.68	15.70	–	20.32	15.87
GDN	15.00	15.02	–	17.06	15.10
KDA	14.49	14.51	–	15.84	14.91

Limits at INT4. At INT4, the invariances reduce quantization error substantially but do not consistently recover full-precision quality. With per-channel value scales, key-axis transformations provide the main gains where applicable (Tables 9 and 11). With per-tensor value scales, value-axis transformations and their combinations often recover much of the loss (Tables 10 and 12). The write-scale controls in Table 6 show no improvement from dynamic scaling in the tested configurations.

These experiments use INT4 code ranges within INT8 kernels. Deployment below INT8 therefore requires both a more accurate quantization scheme and an efficient sub-byte implementation.

Table 6: WikiText-2 perplexity for the static and dynamic per-chunk write-scale controls at INT4, with per-head read-key scales and INT8 stored states (340M models).

Model	Static write scales	Dynamic write scales
LA	22.7	26.2
Mamba-2	16.1	16.4
GDN	18.4	18.8

State precision and write-scale controls. State quantization affects accumulating recurrences most strongly (Table 7). Keeping the read state in floating point reduces error in LA and gated LA, while changing little in the decaying cells. Carrying the requantized state into subsequent updates adds error beyond that of an INT8 read copy, supporting the distinction between read precision and carry precision.

Refining write-key scales alone does not resolve the error from coarse keys (Table 8). It recovers part of the loss in linear-write cells, particularly at INT6, but has little effect in delta cells. The remaining error therefore cannot be addressed solely at the write boundary. INT32 local-score accumulation and the FP16 intra-chunk output product remain fixed in these controls.

Table 7: State-quantization controls for the 340M zoo, with all invariances disabled. Entries report WikiText-2 perplexity. For per-tensor keys, we compare an unquantized state with the kernels' FP32 carry and INT8 read copy. For per-head keys, we additionally test an INT8 carry that feeds requantization error into later updates. Quantizing the read copy has the largest effect in non-decaying cells. Quantizing the carry adds further loss in the tested settings.

		per-tensor keys				per-head keys					
		INT8		INT6		INT8			INT6		
cell	fp model	fp state	INT8 stored	fp state	INT8 stored	fp state	INT8 stored	INT8 carry	fp state	INT8 stored	INT8 carry
LA	17.117	17.976	19.482	24.403	25.706	17.298	19.140	19.850	19.954	22.349	23.963
LA+β	16.688	16.825	17.541	19.185	19.631	16.719	17.466	18.196	17.373	18.209	19.118
Mamba-2	15.553	15.579	15.612	15.988	16.029	15.568	15.601	15.700	15.763	15.795	15.914
Mamba-2+β	15.420	15.435	15.443	15.585	15.595	15.432	15.439	15.444	15.518	15.524	15.545
GLA	15.104	15.124	15.124	15.651	15.651	15.120	15.119	15.169	15.307	15.312	15.360
GLA+β	15.308	15.322	15.322	15.544	15.540	15.320	15.318	15.422	15.457	15.456	15.566
DeltaNet	15.678	15.696	15.698	15.921	15.925	15.695	15.697	15.839	15.870	15.867	16.012
GDN	15.004	15.018	15.019	15.171	15.168	15.018	15.017	15.048	15.120	15.122	15.153
KDA	14.489	14.508	14.511	14.847	14.846	14.504	14.504	14.602	14.760	14.764	14.861
LA, no norm	17.230	17.637	17.797	24.756	24.916	17.319	17.525	18.210	18.294	18.502	19.443
LA+β, no norm	16.980	17.895	18.744	29.797	30.719	17.555	18.390	19.031	20.621	21.446	22.277
Mamba-2, no norm	16.721	16.903	16.911	20.827	20.842	16.752	16.761	16.930	17.106	17.115	17.328
Mamba-2+β, no norm	16.189	16.261	16.266	17.641	17.665	16.204	16.209	16.377	16.384	16.391	16.577
GLA, no norm	16.070	16.278	16.282	20.283	20.282	16.195	16.197	16.368	18.087	18.091	18.326
GLA+β, no norm	15.891	16.009	16.009	19.246	19.222	15.937	15.940	16.031	16.709	16.703	16.818

E.3 Full quality tables and zero-shot accuracy

Tables 9-12 report WikiText-2 perplexity at both model sizes with per-tensor key scales and either per-channel or per-tensor value scales. They compare full precision, the naive quantizer, each invariance at its held-out-selected setting (marked $*$), and the stack with canonical or rotated storage. Operand codes use INT8, INT6, or INT4 ranges, while the stored state remains INT8.

Tables 13 and 14 add the per-head-key regime at both sizes. Tables 15 and 16 give the remaining scale regimes. Table 8 reports the attribution controls. Tables 17 and 18 give downstream accuracy. The full floating-point-state controls, parameter sweeps, and the unnormalized cells are provided with the code where they are omitted from these tables.

Table 8: Attribution controls for the 340M zoo with INT8 stored state. Read keys use per-tensor scales, while write keys use per-channel scales. Value scales are per channel or per tensor, as indicated. Relative to the corresponding shared-key regime, refining the write alone recovers at most a quarter of the loss at INT8. At INT6 it recovers a third to a half on the linear-write cells and almost nothing on the delta cells. The coarse read therefore remains the larger term at INT8 and is comparable to the write at INT6 on the linear-write cells.

value scales, codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.
per channel, INT8	LA	17.117	19.153	17.672	22.411	19.141
	LA+β	16.688	17.477	16.925	21.379	17.475
	Mamba-2	15.553	15.601	15.588	15.622	15.598
	Mamba-2+β	15.420	15.442	15.430	15.442	15.439
	GLA	15.104	15.120	15.124	15.117	15.119
	GLA+β	15.308	15.320	15.322	15.320	15.319
	DeltaNet	15.678	15.698	15.691	15.685	15.688
	GDN	15.004	15.018	15.011	15.014	15.015
	KDA	14.489	14.508	14.511	14.509	14.507
	LA, no norm	17.230	17.645	17.682	17.796	17.636
	LA+β, no norm	16.980	18.669	19.359	19.841	18.916
	Mamba-2, no norm	16.721	16.883	16.740	16.881	16.877
	Mamba-2+β, no norm	16.189	16.251	16.195	16.241	16.249
	GLA, no norm	16.070	16.264	16.272	16.267	16.265
	GLA+β, no norm	15.891	15.991	15.998	15.987	15.991
per channel, INT6	LA	17.117	22.422	20.243	26.526	22.415
	LA+β	16.688	18.312	17.526	22.556	18.340
	Mamba-2	15.553	15.796	15.817	15.818	15.798
	Mamba-2+β	15.420	15.536	15.490	15.530	15.525
	GLA	15.104	15.325	15.380	15.315	15.312
	GLA+β	15.308	15.463	15.477	15.465	15.462
	DeltaNet	15.678	15.916	15.771	15.903	15.903
	GDN	15.004	15.145	15.046	15.158	15.154
	KDA	14.489	14.787	14.804	14.791	14.790
	LA, no norm	17.230	20.597	20.238	20.817	20.551
	LA+β, no norm	16.980	29.583	28.484	31.063	29.464
	Mamba-2, no norm	16.721	20.215	17.023	20.205	20.207
	Mamba-2+β, no norm	16.189	17.380	16.282	17.367	17.390
	GLA, no norm	16.070	19.788	20.015	19.809	19.789
	GLA+β, no norm	15.891	18.182	18.325	18.179	18.177
per tensor, INT8	LA	17.117	19.204	17.750	22.413	19.146
	LA+β	16.688	18.027	17.392	21.393	17.486
	Mamba-2	15.553	15.623	15.615	15.622	15.604
	Mamba-2+β	15.420	15.467	15.455	15.441	15.439
	GLA	15.104	15.135	15.139	15.120	15.120
	GLA+β	15.308	15.335	15.336	15.320	15.320
	DeltaNet	15.678	15.935	15.927	15.699	15.697
	GDN	15.004	15.124	15.117	15.022	15.022
	KDA	14.489	14.553	14.556	14.510	14.508
	LA, no norm	17.230	17.780	17.828	18.537	18.348
	LA+β, no norm	16.980	27.492	27.727	20.876	19.988
	Mamba-2, no norm	16.721	16.902	16.758	16.873	16.882
	Mamba-2+β, no norm	16.189	16.288	16.235	16.248	16.254
	GLA, no norm	16.070	16.285	16.297	16.273	16.271
	GLA+β, no norm	15.891	16.020	16.026	15.990	15.990
per tensor, INT6	LA	17.117	26.380	24.831	26.604	22.505
	LA+β	16.688	26.888	26.153	23.013	18.643
	Mamba-2	15.553	16.242	16.277	15.851	15.830
	Mamba-2+β	15.420	16.161	16.092	15.585	15.579
	GLA	15.104	15.678	15.737	15.337	15.338
	GLA+β	15.308	15.952	15.955	15.485	15.486
	DeltaNet	15.678	20.261	20.163	16.137	16.123
	GDN	15.004	17.059	16.967	15.261	15.256
	KDA	14.489	15.766	15.759	14.848	14.846
	LA, no norm	17.230	25.976	25.130	28.496	28.201
	LA+β, no norm	16.980	92.325	88.652	46.653	44.630
	Mamba-2, no norm	16.721	20.817	17.426	20.259	20.260
	Mamba-2+β, no norm	16.189	18.460	17.174	17.478	17.486
	GLA, no norm	16.070	20.411	20.619	20.110	20.085
	GLA+β, no norm	15.891	19.410	19.596	18.291	18.286

Table 9: WikiText-2 perplexity for the 340M zoo under per-tensor key scales and per-channel value scales. Operand codes use INT8, INT6, or INT4 ranges. The stored state remains INT8. An asterisk marks parameters selected by held-out cross-entropy. The stack combines promoted transformations and uses canonical storage. "stack, rot." uses the same transformations with rotated storage. A dash denotes an inapplicable or untested setting, or a stack with no promoted transformation.

codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	η^*	β^*	$\mathbf{D}^*_{qk}$	$\mathbf{D}_{qk}+\mathbf{R}^*_{QK}$	$\mathbf{D}^*_v$	stack	stack, rot.
INT8	LA	17.117	19.482	17.736	22.915	19.502	–	–	17.424	17.265	19.485	17.265	–
	LA+β	16.688	17.541	16.928	21.510	17.519	–	17.693	16.888	16.797	17.540	16.798	–
	Mamba-2	15.553	15.612	15.592	15.631	15.608	15.613	–	15.561	15.556	15.610	15.561	–
	Mamba-2+β	15.420	15.443	15.433	15.442	15.439	15.445	15.446	15.431	15.427	15.444	–	–
	GLA	15.104	15.124	15.126	15.123	15.121	15.125	–	15.116	15.121	15.125	–	–
	GLA+β	15.308	15.322	15.323	15.322	15.321	15.321	15.325	15.311	15.313	15.320	–	–
	DeltaNet	15.678	15.698	15.693	15.688	15.690	–	15.695	–	–	15.696	–	–
	GDN	15.004	15.019	15.012	15.015	15.015	15.019	15.017	–	–	15.018	–	–
	KDA	14.489	14.511	14.514	14.508	14.507	14.507	14.508	–	–	14.509	–	–
	LA, no norm	17.230	17.797	18.183	17.939	17.791	–	–	17.372	17.385	17.789	17.388	–
	LA+β, no norm	16.980	18.744	19.440	19.926	18.988	–	20.263	18.135	17.903	18.751	17.902	–
	Mamba-2, no norm	16.721	16.911	16.744	16.906	16.907	16.901	–	16.800	16.735	16.914	16.735	–
	Mamba-2+β, no norm	16.189	16.266	16.198	16.259	16.263	16.263	16.281	16.244	16.194	16.265	16.194	–
	GLA, no norm	16.070	16.282	16.289	16.278	16.278	16.280	–	16.100	16.104	16.279	16.099	–
	GLA+β, no norm	15.891	16.009	16.015	16.009	16.007	16.009	16.024	15.935	15.936	16.012	15.935	–
INT6	LA	17.117	25.706	20.818	28.442	25.691	–	–	19.428	18.097	25.681	18.097	–
	LA+β	16.688	19.631	17.669	22.619	19.668	–	21.808	17.499	17.079	19.624	17.089	19.236
	Mamba-2	15.553	16.029	15.876	16.044	16.023	16.026	–	15.722	15.603	16.022	15.605	–
	Mamba-2+β	15.420	15.595	15.499	15.584	15.586	15.596	15.641	15.485	15.446	15.592	15.449	–
	GLA	15.104	15.651	15.700	15.650	15.642	15.659	–	15.302	15.321	15.653	15.302	–
	GLA+β	15.308	15.540	15.560	15.543	15.542	15.542	15.624	15.418	15.424	15.541	15.415	–
	DeltaNet	15.678	15.925	15.778	15.921	15.918	–	15.843	–	–	15.927	15.769	–
	GDN	15.004	15.168	15.055	15.173	15.172	15.160	15.128	–	–	15.165	15.042	–
	KDA	14.489	14.846	14.850	14.843	14.845	14.850	14.846	–	–	14.846	–	–
	LA, no norm	17.230	24.916	24.089	25.032	24.897	–	–	18.079	18.596	24.854	18.071	–
	LA+β, no norm	16.980	30.719	29.671	32.049	30.834	–	40.032	21.042	20.775	30.721	20.738	21.141
	Mamba-2, no norm	16.721	20.842	17.086	20.824	20.813	20.757	–	18.178	16.858	20.844	16.848	–
	Mamba-2+β, no norm	16.189	17.665	16.306	17.661	17.660	17.573	17.940	16.943	16.268	17.665	16.264	–
	GLA, no norm	16.070	20.282	20.492	20.260	20.264	20.289	–	16.740	16.791	20.263	16.739	–
	GLA+β, no norm	15.891	19.222	19.383	19.226	19.223	19.233	19.557	16.834	16.902	19.213	16.838	–
INT4	LA	17.117	73.226	44.333	76.041	73.632	–	–	35.896	24.910	72.893	24.736	24.803
	LA+β	16.688	83.120	32.218	86.787	83.585	–	89.672	29.426	22.503	83.450	22.495	–
	Mamba-2	15.553	25.812	20.877	25.803	25.728	25.995	–	18.168	16.556	25.810	16.427	16.425
	Mamba-2+β	15.420	21.059	17.582	20.980	21.018	21.131	22.198	17.172	16.068	21.061	16.070	–
	GLA	15.104	32.431	35.264	32.406	32.350	32.538	–	20.600	21.490	32.432	20.587	–
	GLA+β	15.308	24.896	26.203	25.004	24.962	25.040	26.192	18.760	19.161	24.890	18.766	–
	DeltaNet	15.678	24.523	18.757	24.098	24.010	–	21.416	–	–	24.476	17.685	17.680
	GDN	15.004	20.540	16.282	20.518	20.570	20.485	19.677	–	–	20.525	16.098	16.084
	KDA	14.489	28.732	29.717	28.750	28.599	28.689	28.257	–	–	28.717	–	–
	LA, no norm	17.230	314.039	153.517	317.889	311.775	–	–	37.950	47.213	312.878	37.950	–
	LA+β, no norm	16.980	515.217	151.013	595.755	467.922	–	735.863	68.784	79.739	514.379	69.456	71.150
	Mamba-2, no norm	16.721	316.762	35.612	315.117	315.802	315.758	–	107.195	23.947	316.221	23.555	23.540
	Mamba-2+β, no norm	16.189	202.732	19.859	202.726	200.767	188.882	208.878	64.850	18.533	202.459	18.344	18.351
	GLA, no norm	16.070	173.810	181.020	173.014	173.332	173.702	–	46.247	50.952	173.540	46.248	–
	GLA+β, no norm	15.891	172.781	178.699	171.474	171.329	172.893	164.104	57.650	63.135	173.143	42.062	42.049

Table 10: WikiText-2 perplexity for the 340M zoo with coarse scales (per-tensor key and value scales). Operand codes use INT8, INT6, or INT4 ranges. The stored state remains INT8. An asterisk marks parameters selected by held-out cross-entropy. The stack combines promoted transformations and uses canonical storage. "stack, rot." uses the same transformations with rotated storage. A dash denotes an inapplicable or untested setting, or a stack with no promoted transformation.

codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	η^*	β^*	$\mathbf{D}^*_{qk}$	$\mathbf{D}_{qk}+\mathbf{R}^*_{QK}$	$\mathbf{D}^*_v$	stack	stack, rot.
INT8	LA	17.117	19.576	17.816	22.901	19.496	–	–	17.492	17.351	19.505	17.285	17.388
	LA+β	16.688	18.104	17.406	21.535	17.533	–	18.071	17.352	17.229	17.545	16.809	16.934
	Mamba-2	15.553	15.633	15.617	15.634	15.614	15.635	–	15.582	15.578	15.609	–	–
	Mamba-2+β	15.420	15.469	15.459	15.441	15.438	15.471	15.461	15.458	15.453	15.440	–	–
	GLA	15.104	15.143	15.144	15.123	15.123	15.143	–	15.131	15.136	15.124	–	–
	GLA+β	15.308	15.337	15.339	15.321	15.322	15.337	15.336	15.330	15.332	15.323	–	–
	DeltaNet	15.678	15.936	15.932	15.699	15.700	–	15.809	–	–	15.714	15.693	15.692
	GDN	15.004	15.124	15.114	15.022	15.023	15.131	15.045	–	–	15.013	15.020	15.019
	KDA	14.489	14.559	14.561	14.509	14.510	14.559	14.537	–	–	14.511	–	–
	LA, no norm	17.230	17.936	18.337	18.688	18.519	–	–	17.505	17.518	17.798	17.384	–
	LA+β, no norm	16.980	27.778	27.958	20.969	20.049	–	21.327	26.508	26.370	18.976	17.453	17.719
	Mamba-2, no norm	16.721	16.938	16.763	16.907	16.914	16.927	–	16.820	16.750	16.911	16.751	–
	Mamba-2+β, no norm	16.189	16.299	16.234	16.262	16.267	16.300	16.298	16.278	16.233	16.264	16.197	16.199
	GLA, no norm	16.070	16.298	16.306	16.292	16.288	16.295	–	16.115	16.119	16.277	16.117	–
	GLA+β, no norm	15.891	16.036	16.046	16.012	16.011	16.039	16.037	15.971	15.969	16.010	15.936	15.936
INT6	LA	17.117	30.694	25.699	28.572	25.728	–	–	23.151	21.629	25.772	18.138	18.263
	LA+β	16.688	28.381	26.460	23.104	20.028	–	28.242	25.822	25.191	19.773	17.388	17.471
	Mamba-2	15.553	16.446	16.362	16.073	16.052	16.435	–	16.124	16.035	16.036	15.633	15.635
	Mamba-2+β	15.420	16.230	16.105	15.645	15.641	16.224	16.047	16.083	16.032	15.609	15.473	15.474
	GLA	15.104	16.048	16.113	15.668	15.661	16.066	–	15.646	15.671	15.653	15.309	15.301
	GLA+β	15.308	16.046	16.080	15.559	15.560	16.052	15.989	15.883	15.890	15.547	15.440	15.450
	DeltaNet	15.678	20.323	20.188	16.149	16.145	–	18.064	–	–	16.138	15.871	15.867
	GDN	15.004	17.064	16.980	15.277	15.278	17.054	15.902	–	–	15.236	15.103	15.098
	KDA	14.489	15.837	15.835	14.904	14.903	15.854	15.567	–	–	14.885	14.906	14.909
	LA, no norm	17.230	31.582	29.824	32.679	32.595	–	–	22.486	23.224	25.271	18.336	–
	LA+β, no norm	16.980	100.247	93.348	49.765	47.271	–	59.557	72.936	72.718	35.441	20.429	20.828
	Mamba-2, no norm	16.721	21.479	17.506	20.899	20.877	21.393	–	18.725	17.282	20.830	16.924	16.926
	Mamba-2+β, no norm	16.189	18.844	17.213	17.774	17.776	18.744	18.689	18.059	17.166	17.710	16.326	16.327
	GLA, no norm	16.070	20.977	21.179	20.598	20.596	20.963	–	17.168	17.234	20.289	16.907	16.902
	GLA+β, no norm	15.891	20.523	20.713	19.349	19.342	20.520	20.214	17.873	17.946	19.258	16.647	16.648
INT4	LA	17.117	498.768	453.664	90.852	88.967	–	–	353.550	347.153	81.285	32.750	32.641
	LA+β	16.688	726.570	753.363	104.067	99.728	–	601.108	587.280	584.686	93.606	73.118	73.798
	Mamba-2	15.553	85.441	67.432	28.769	28.613	84.403	–	61.375	53.492	26.368	17.141	17.148
	Mamba-2+β	15.420	55.010	45.621	23.174	23.238	54.784	51.388	45.327	41.282	21.995	16.748	16.760
	GLA	15.104	88.884	95.686	33.795	33.769	88.957	–	52.846	55.400	32.703	21.214	21.174
	GLA+β	15.308	70.122	72.157	26.649	26.525	69.988	66.228	54.487	55.242	25.408	19.410	19.423
	DeltaNet	15.678	168.077	143.663	33.231	33.020	–	135.729	–	–	33.655	20.072	20.090
	GDN	15.004	82.814	67.829	24.308	24.112	81.699	71.295	–	–	26.439	17.421	17.417
	KDA	14.489	111.449	113.296	32.006	31.915	112.008	107.620	–	–	30.747	30.205	30.307
	LA, no norm	17.230	436.488	359.780	323.798	319.046	–	–	175.638	185.580	320.495	66.355	66.399
	LA+β, no norm	16.980	663.576	461.182	708.109	595.618	–	615.277	317.193	351.160	539.578	94.058	94.193
	Mamba-2, no norm	16.721	362.199	93.173	324.449	325.066	362.187	–	215.100	62.621	321.767	25.470	25.444
	Mamba-2+β, no norm	16.189	285.469	47.398	225.767	224.089	274.292	290.144	136.382	44.959	214.049	22.110	22.119
	GLA, no norm	16.070	263.196	267.158	184.275	184.648	263.128	–	117.335	124.023	175.799	54.814	54.808
	GLA+β, no norm	15.891	244.264	248.610	183.650	182.796	244.589	236.569	125.569	132.297	176.261	45.162	45.222

Table 11: WikiText-2 perplexity for the 1.3B zoo under per-tensor key scales and per-channel value scales. Operand codes use INT8, INT6, or INT4 ranges. The stored state remains INT8. An asterisk marks parameters selected by held-out cross-entropy. The stack combines promoted transformations and uses canonical storage. "stack, rot." uses the same transformations with rotated storage. A dash denotes an inapplicable or untested setting, or a stack with no promoted transformation.

codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	η^*	β^*	$\mathbf{D}^*_{qk}$	$\mathbf{D}_{qk}$+$\mathbf{R}^*_{QK}$	$\mathbf{D}^*_v$	stack	stack, rot.
INT8	LA+β	10.276	10.763	10.450	17.762	10.751	–	10.821	10.411	10.346	10.756	10.349	–
	Mamba-2+β	9.755	9.774	9.768	9.765	9.766	9.776	9.778	9.772	9.768	9.774	–	–
	GLA+β	9.189	9.205	9.205	9.194	9.199	9.205	9.207	9.203	9.203	9.205	–	–
	DeltaNet	9.917	9.934	9.928	9.928	9.929	–	9.933	–	–	9.935	–	–
	GDN	9.727	9.731	9.728	9.728	9.728	9.732	9.733	–	–	9.730	–	–
	KDA	9.460	9.467	9.468	9.467	9.465	9.468	9.468	–	–	9.468	–	–
INT6	LA+β	10.276	12.101	11.043	16.779	12.144	–	13.008	10.768	10.550	12.095	10.625	14.356
	Mamba-2+β	9.755	9.839	9.805	9.825	9.826	9.842	9.854	9.811	9.789	9.839	9.789	–
	GLA+β	9.189	9.307	9.304	9.288	9.298	9.305	9.316	9.267	9.266	9.309	9.260	–
	DeltaNet	9.917	10.050	9.979	10.043	10.042	–	10.021	–	–	10.049	9.973	–
	GDN	9.727	9.784	9.751	9.780	9.781	9.785	9.777	–	–	9.784	9.752	–
	KDA	9.460	9.603	9.605	9.595	9.593	9.600	9.605	–	–	9.602	–	–
INT4	LA+β	10.276	60.518	25.381	76.649	67.746	–	81.812	18.310	14.005	61.004	13.987	–
	Mamba-2+β	9.755	11.767	10.524	11.733	11.745	11.795	12.313	10.626	10.179	11.773	10.185	–
	GLA+β	9.189	12.509	13.167	12.446	12.443	12.558	12.837	10.795	11.037	12.512	10.786	–
	DeltaNet	9.917	13.753	11.436	14.174	14.203	–	12.245	–	–	13.776	11.322	–
	GDN	9.727	11.916	10.404	11.788	11.785	11.941	11.743	–	–	11.919	10.329	10.325
	KDA	9.460	20.150	21.012	19.876	19.760	20.190	20.011	–	–	20.018	19.877	19.950

Table 12: WikiText-2 perplexity for the 1.3B zoo with coarse scales (per-tensor key and value scales). Operand codes use INT8, INT6, or INT4 ranges. The stored state remains INT8. An asterisk marks parameters selected by held-out cross-entropy. The stack combines promoted transformations and uses canonical storage. "stack, rot." uses the same transformations with rotated storage. A dash denotes an inapplicable or untested setting, or a stack with no promoted transformation.

codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	η^*	β^*	$\mathbf{D}^*_{qk}$	$\mathbf{D}_{qk}$+$\mathbf{R}^*_{QK}$	$\mathbf{D}^*_v$	stack	stack, rot.
INT8	LA+β	10.276	14.663	13.959	17.864	10.816	–	13.435	13.772	13.706	10.789	10.357	10.427
	Mamba-2+β	9.755	9.863	9.856	9.769	9.770	9.861	9.812	9.861	9.855	9.772	9.769	9.767
	GLA+β	9.189	9.231	9.230	9.197	9.202	9.230	9.228	9.230	9.227	9.201	9.198	9.194
	DeltaNet	9.917	11.161	11.150	9.941	9.948	–	10.476	–	–	9.940	9.936	9.934
	GDN	9.727	9.800	9.799	9.736	9.735	9.799	9.769	–	–	9.732	9.735	9.735
	KDA	9.460	9.520	9.522	9.472	9.470	9.517	9.509	–	–	9.471	9.467	9.470
INT6	LA+β	10.276	148.173	139.550	20.472	14.349	–	136.649	117.682	119.662	12.588	10.842	10.885
	Mamba-2+β	9.755	12.687	12.577	9.898	9.897	12.410	11.475	12.581	12.539	9.895	9.802	9.797
	GLA+β	9.189	10.115	10.132	9.316	9.320	10.121	9.995	10.026	10.050	9.317	9.263	9.258
	DeltaNet	9.917	46.639	46.760	10.406	10.426	–	24.673	–	–	10.462	10.111	10.111
	GDN	9.727	11.401	11.345	9.874	9.877	11.382	10.511	–	–	9.867	9.788	9.792
	KDA	9.460	10.741	10.745	9.648	9.642	10.738	10.528	–	–	9.715	9.653	9.666
INT4	LA+β	10.276	4185.731	4042.798	351.649	325.086	–	4559.968	3703.302	3088.378	198.469	44.904	44.504
	Mamba-2+β	9.755	295.661	265.588	14.212	14.197	287.980	290.828	265.629	254.153	13.105	10.971	10.965
	GLA+β	9.189	160.470	172.806	14.004	14.056	162.990	135.290	126.567	132.256	13.324	11.423	11.421
	DeltaNet	9.917	957.275	899.386	34.650	34.374	–	790.463	–	–	24.762	15.235	15.253
	GDN	9.727	424.986	368.872	16.710	16.756	431.052	435.192	–	–	16.478	16.016	16.100
	KDA	9.460	352.929	362.105	28.135	27.811	354.459	351.016	–	–	27.237	25.376	25.350

Table 13: WikiText-2 perplexity for the 340M zoo with per-head key scales and per-channel value scales, at INT8 and INT6 operand codes. The stored state remains INT8. Columns compare the naive quantizer, individual rotations, and the selected stack, including the canonical-storage variant of value rotation.

codes	cell	fp	none	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_{QK}$+$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	$\mathbf{R}_{QK}$+$\mathbf{R}_V$ canon.	stack rot.	stack canon.
INT8	LA	17.117	19.140	17.628	22.415	19.108	19.128	17.637	19.144	17.637
	LA+β	16.688	17.466	16.913	21.343	19.808	17.461	16.908	19.771	16.903
	Mamba-2	15.553	15.601	15.581	15.616	15.591	15.597	15.580	15.592	15.586
	Mamba-2+β	15.420	15.439	15.429	15.440	15.427	15.438	15.426	15.425	15.428
	GLA	15.104	15.119	15.122	15.118	15.123	15.117	15.123	15.124	15.123
	GLA+β	15.308	15.318	15.321	15.322	15.324	15.318	15.320	15.323	15.321
	DeltaNet	15.678	15.697	15.694	15.686	15.683	15.690	15.685	15.685	15.684
	GDN	15.004	15.017	15.013	15.013	15.009	15.014	15.008	15.010	15.008
	KDA	14.489	14.504	14.507	14.505	14.507	14.503	14.507	14.507	14.506
	LA, no norm	17.229	17.525	17.606	17.675	17.873	17.526	17.601	17.866	17.601
	LA+β, no norm	16.980	18.390	19.098	19.539	21.149	18.646	19.519	21.139	19.530
	Mamba-2, no norm	16.721	16.761	16.730	16.756	16.727	16.761	16.729	16.728	16.729
	Mamba-2+β, no norm	16.189	16.209	16.193	16.203	16.191	16.207	16.191	16.190	16.192
	GLA, no norm	16.070	16.197	16.198	16.197	16.199	16.198	16.199	16.200	16.199
	GLA+β, no norm	15.891	15.940	15.940	15.936	15.938	15.937	15.937	15.939	15.938
INT6	LA	17.117	22.349	19.815	26.479	21.060	22.326	19.783	21.049	19.783
	LA+β	16.688	18.209	17.440	22.413	20.321	18.233	17.467	20.321	17.460
	Mamba-2	15.553	15.795	15.767	15.816	15.760	15.792	15.755	15.768	15.761
	Mamba-2+β	15.420	15.524	15.470	15.525	15.468	15.523	15.466	15.466	15.465
	GLA	15.104	15.312	15.354	15.310	15.352	15.311	15.354	15.354	15.353
	GLA+β	15.308	15.456	15.463	15.455	15.463	15.456	15.465	15.464	15.466
	DeltaNet	15.678	15.867	15.768	15.855	15.756	15.863	15.759	15.745	15.742
	GDN	15.004	15.122	15.039	15.129	15.037	15.127	15.041	15.052	15.051
	KDA	14.489	14.764	14.777	14.760	14.779	14.761	14.771	14.802	14.802
	LA, no norm	17.229	18.502	19.023	18.650	19.195	18.484	19.008	19.208	19.004
	LA+β, no norm	16.980	21.446	21.378	22.551	22.546	21.575	21.359	22.544	21.366
	Mamba-2, no norm	16.721	17.115	16.786	17.104	16.781	17.112	16.780	16.780	16.782
	Mamba-2+β, no norm	16.189	16.391	16.230	16.392	16.231	16.399	16.230	16.229	16.233
	GLA, no norm	16.070	18.091	18.147	18.081	18.125	18.087	18.130	18.131	18.131
	GLA+β, no norm	15.891	16.703	16.738	16.696	16.723	16.701	16.730	16.727	16.723

Table 14: WikiText-2 perplexity for the 1.3B zoo under per-head key scales and per-channel value scales. Operand codes use INT8, INT6, or INT4 ranges. The stored state remains INT8. An asterisk marks parameters selected by held-out cross-entropy. The stack combines promoted transformations and uses canonical storage. "stack, rot." uses the same transformations with rotated storage. A dash denotes an inapplicable or untested setting, or a stack with no promoted transformation.

codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	η^*	β^*	$\mathbf{D}^*_{qk}$	$\mathbf{D}_{qk}$+$\mathbf{R}^*_{QK}$	$\mathbf{D}^*_v$	stack	stack, rot.
INT8	LA	10.153	11.380	10.546	15.980	11.375	–	–	10.310	–	11.383	–	–
	LA+β	10.276	10.725	10.433	17.439	10.718	–	10.753	10.399	10.339	10.724	–	–
	Mamba-2	9.775	9.801	9.788	9.799	9.796	9.800	–	9.784	–	9.800	–	–
	Mamba-2+β	9.755	9.772	9.768	9.764	9.762	9.771	9.768	9.771	9.768	9.771	–	–
	GLA	9.061	9.069	9.071	9.065	9.066	9.068	–	9.068	–	9.069	–	–
	GLA+β	9.189	9.204	9.204	9.193	9.199	9.204	9.205	9.202	9.201	9.204	–	–
	DeltaNet	9.917	9.934	9.927	9.923	9.925	–	9.932	–	–	9.933	–	–
	GDN	9.727	9.730	9.728	9.726	9.727	9.731	9.732	–	–	9.729	–	–
	KDA	9.460	9.463	9.462	9.462	9.461	9.462	9.463	–	–	9.462	–	–
INT6	LA	10.153	13.384	12.276	18.518	13.384	–	–	10.903	–	–	–	–
	LA+β	10.276	11.497	10.812	19.061	11.525	–	11.952	10.533	10.463	11.502	–	–
	Mamba-2	9.775	9.859	9.843	9.858	9.857	–	–	–	–	–	–	–
	Mamba-2+β	9.755	9.817	9.796	9.805	9.807	9.815	9.820	9.799	9.788	9.818	–	–
	GLA	9.061	9.144	9.148	9.139	9.136	–	–	–	–	–	–	–
	GLA+β	9.189	9.271	9.271	9.255	9.262	9.272	9.271	9.235	9.237	9.274	–	–
	DeltaNet	9.917	10.013	9.969	10.009	10.014	–	9.988	–	–	10.016	–	–
	GDN	9.727	9.766	9.748	9.761	9.760	9.765	9.764	–	–	9.764	–	–
	KDA	9.460	9.571	9.574	9.567	9.564	9.569	9.570	–	–	9.570	–	–

Table 15: WikiText-2 perplexity for the 340M zoo in the remaining scale regimes: per-head keys with per-tensor values $\langle \mathrm{H}, \mathrm{T} \rangle$, per-tensor keys with per-head values $\langle \mathrm{T}, \mathrm{H} \rangle$, and per-head keys with per-head values $\langle \mathrm{H}, \mathrm{H} \rangle$. The stored state remains INT8. An asterisk marks the value rescaling exponent selected by held-out cross-entropy. A dash denotes an untested setting.

regime, codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	$\mathbf{D}_v^*$
$\langle \mathrm{H}, \mathrm{T} \rangle$, INT8	LA	17.117	19.561	17.767	22.914	19.489	–
	LA+β	16.688	18.093	17.398	21.562	17.527	–
	Mamba-2	15.553	15.632	15.607	15.633	15.610	–
	Mamba-2+β	15.420	15.468	15.460	15.440	15.436	–
	GLA	15.104	15.141	15.142	15.124	15.124	–
	GLA+β	15.308	15.335	15.340	15.321	15.319	–
	DeltaNet	15.678	15.942	15.932	15.699	15.701	–
	GDN	15.004	15.122	15.119	15.022	15.022	–
	KDA	14.489	14.553	14.558	14.508	14.505	–
	LA, no norm	17.230	17.828	18.253	18.577	18.390	–
	LA+β, no norm	16.980	27.193	27.547	20.572	19.695	–
	Mamba-2, no norm	16.721	16.813	16.752	16.788	16.791	–
	Mamba-2+β, no norm	16.189	16.261	16.235	16.224	16.226	–
	GLA, no norm	16.070	16.229	16.232	16.216	16.219	–
	GLA+β, no norm	15.891	15.987	15.988	15.960	15.956	–
$\langle \mathrm{H}, \mathrm{T} \rangle$, INT6	LA	17.117	30.544	25.064	28.426	25.647	–
	LA+β	16.688	28.294	26.326	22.955	19.928	–
	Mamba-2	15.553	16.451	16.293	16.077	16.051	–
	Mamba-2+β	15.420	16.220	16.087	15.636	15.633	–
	GLA	15.104	16.033	16.093	15.648	15.646	–
	GLA+β	15.308	16.038	16.066	15.546	15.554	–
	DeltaNet	15.678	20.387	20.205	16.117	16.119	–
	GDN	15.004	17.057	16.981	15.260	15.267	–
	KDA	14.489	15.798	15.820	14.873	14.879	–
	LA, no norm	17.230	27.401	27.430	29.168	29.021	–
	LA+β, no norm	16.980	81.912	78.605	36.239	34.547	–
	Mamba-2, no norm	16.721	18.269	17.242	17.855	17.859	–
	Mamba-2+β, no norm	16.189	17.671	17.143	16.753	16.760	–
	GLA, no norm	16.070	19.168	19.235	18.851	18.847	–
	GLA+β, no norm	15.891	19.003	19.074	17.982	17.986	–

Table 15: WikiText-2 perplexity for the 340M zoo in the remaining scale regimes (continued). The stored state remains INT8. All conventions are as above.

regime, codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	$\mathbf{D}_v^*$
⟨T, H⟩, INT8	LA	17.117	19.496	17.756	22.917	19.490	19.506
	LA+β	16.688	17.661	17.039	21.556	17.531	17.568
	Mamba-2	15.553	15.615	15.601	15.632	15.608	15.610
	Mamba-2+β	15.420	15.441	15.431	15.440	15.440	15.437
	GLA	15.104	15.129	15.133	15.122	15.124	15.124
	GLA+β	15.308	15.329	15.330	15.322	15.322	15.320
	DeltaNet	15.678	15.759	15.755	15.691	15.691	15.697
	GDN	15.004	15.027	15.020	15.013	15.014	15.016
	KDA	14.489	14.527	14.531	14.510	14.511	14.507
	LA, no norm	17.230	17.806	18.206	17.950	17.808	17.792
	LA+β, no norm	16.980	21.291	21.434	20.303	19.405	18.764
	Mamba-2, no norm	16.721	16.918	16.751	16.906	16.909	16.909
	Mamba-2+β, no norm	16.189	16.280	16.212	16.263	16.264	16.264
	GLA, no norm	16.070	16.284	16.298	16.279	16.282	16.279
	GLA+β, no norm	15.891	16.021	16.022	16.008	16.009	16.011
⟨T, H⟩, INT6	LA	17.117	26.492	21.651	28.505	25.721	25.746
	LA+β	16.688	21.690	19.888	22.699	19.775	19.610
	Mamba-2	15.553	16.154	16.023	16.058	16.033	16.028
	Mamba-2+β	15.420	15.773	15.665	15.595	15.590	15.618
	GLA	15.104	15.786	15.853	15.657	15.654	15.650
	GLA+β	15.308	15.748	15.759	15.557	15.543	15.543
	DeltaNet	15.678	17.186	17.026	15.999	16.003	16.016
	GDN	15.004	15.499	15.396	15.173	15.177	15.184
	KDA	14.489	15.203	15.200	14.864	14.866	14.860
	LA, no norm	17.230	25.598	24.701	25.257	25.109	24.920
	LA+β, no norm	16.980	47.725	43.866	36.624	34.870	30.792
	Mamba-2, no norm	16.721	21.010	17.230	20.833	20.805	20.818
	Mamba-2+β, no norm	16.189	18.012	16.598	17.683	17.683	17.659
	GLA, no norm	16.070	20.483	20.730	20.346	20.342	20.283
	GLA+β, no norm	15.891	19.651	19.852	19.259	19.237	19.220
⟨T, H⟩, INT4	LA	17.117	226.971	204.342	78.062	75.794	72.881
	LA+β	16.688	287.447	188.151	88.415	85.777	83.318
	Mamba-2	15.553	38.980	30.795	26.602	26.503	25.865
	Mamba-2+β	15.420	29.339	23.869	21.669	21.625	21.077
	GLA	15.104	46.863	50.965	33.162	33.143	32.424
	GLA+β	15.308	36.115	37.912	25.617	25.548	24.919
	DeltaNet	15.678	58.594	45.615	27.762	27.804	29.546
	GDN	15.004	32.946	25.486	21.874	21.771	22.341
	KDA	14.489	46.017	47.867	29.887	29.795	29.537
	LA, no norm	17.230	421.007	257.324	327.655	321.389	318.453
	LA+β, no norm	16.980	502.206	268.156	631.669	504.195	409.701
	Mamba-2, no norm	16.721	336.452	43.592	316.971	317.710	318.339
	Mamba-2+β, no norm	16.189	228.502	26.439	207.316	204.262	203.536
	GLA, no norm	16.070	196.429	203.454	174.852	174.907	173.972
	GLA+β, no norm	15.891	199.794	204.606	175.411	175.033	173.603

Table 15: WikiText-2 perplexity for the 340M zoo in the remaining scale regimes (continued). The stored state remains INT8. All conventions are as above.

regime, codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	$\mathbf{D}_v^*$
⟨H, H⟩, INT8	LA	17.117	19.150	17.657	22.412	19.120	19.135
	LA+β	16.688	17.599	17.036	21.357	17.471	17.467
	Mamba-2	15.553	15.608	15.593	15.621	15.602	15.598
	Mamba-2+β	15.420	15.439	15.427	15.439	15.437	15.437
	GLA	15.104	15.124	15.131	15.119	15.120	15.119
	GLA+β	15.308	15.323	15.327	15.321	15.319	15.318
	DeltaNet	15.678	15.756	15.750	15.689	15.693	15.695
	GDN	15.004	15.026	15.021	15.013	15.013	15.013
	KDA	14.489	14.523	14.525	14.506	14.504	14.506
	LA, no norm	17.230	17.537	17.615	17.696	17.539	17.525
	LA+β, no norm	16.980	20.778	20.978	19.890	19.014	18.400
	Mamba-2, no norm	16.721	16.766	16.734	16.756	16.761	16.760
	Mamba-2+β, no norm	16.189	16.221	16.206	16.205	16.208	16.206
	GLA, no norm	16.070	16.203	16.204	16.201	16.200	16.200
	GLA+β, no norm	15.891	15.946	15.946	15.937	15.938	15.940
⟨H, H⟩, INT6	LA	17.117	22.823	20.511	26.528	22.360	22.319
	LA+β	16.688	20.456	19.584	22.471	18.297	18.215
	Mamba-2	15.553	15.927	15.892	15.829	15.806	15.801
	Mamba-2+β	15.420	15.693	15.630	15.529	15.527	15.532
	GLA	15.104	15.448	15.496	15.328	15.318	15.311
	GLA+β	15.308	15.653	15.665	15.458	15.463	15.455
	DeltaNet	15.678	17.106	16.975	15.944	15.941	15.944
	GDN	15.004	15.466	15.401	15.127	15.131	15.116
	KDA	14.489	15.109	15.118	14.781	14.782	14.777
	LA, no norm	17.230	18.840	19.403	18.956	18.782	18.496
	LA+β, no norm	16.980	32.887	31.967	24.914	23.906	21.485
	Mamba-2, no norm	16.721	17.266	16.918	17.121	17.126	17.128
	Mamba-2+β, no norm	16.189	16.680	16.506	16.416	16.418	16.391
	GLA, no norm	16.070	18.241	18.289	18.145	18.143	18.111
	GLA+β, no norm	15.891	17.015	17.040	16.716	16.717	16.716
⟨H, H⟩, INT4	LA	17.117	137.205	190.179	39.455	35.670	34.993
	LA+β	16.688	138.792	166.964	37.690	32.889	30.687
	Mamba-2	15.553	32.848	27.661	22.250	22.167	21.691
	Mamba-2+β	15.420	25.979	22.744	19.141	19.115	18.691
	GLA	15.104	36.732	39.509	25.832	25.811	25.325
	GLA+β	15.308	29.347	30.620	21.060	21.041	20.562
	DeltaNet	15.678	54.647	43.254	25.684	25.582	26.842
	GDN	15.004	29.861	25.049	19.344	19.310	19.787
	KDA	14.489	36.973	38.158	24.380	24.343	24.150
	LA, no norm	17.230	75.365	60.629	44.516	44.414	38.281
	LA+β, no norm	16.980	148.003	116.088	101.714	99.569	64.984
	Mamba-2, no norm	16.721	55.473	22.790	43.505	43.457	42.790
	Mamba-2+β, no norm	16.189	42.927	23.205	32.441	32.348	31.055
	GLA, no norm	16.070	61.103	65.742	46.684	46.704	46.114
	GLA+β, no norm	15.891	66.865	70.822	46.341	46.408	45.547

Table 16: WikiText-2 perplexity for the 1.3B zoo in the remaining scale regimes: per-head keys with per-tensor values $\langle \mathrm{H}, \mathrm{T} \rangle$, per-tensor keys with per-head values $\langle \mathrm{T}, \mathrm{H} \rangle$, and per-head keys with per-head values $\langle \mathrm{H}, \mathrm{H} \rangle$. The stored state remains INT8. An asterisk marks the value rescaling exponent selected by held-out cross-entropy. A dash denotes an untested setting.

regime, codes	cell	fp	naive	$\mathbf{R}_{QK}$	$\mathbf{R}_V$ rot.	$\mathbf{R}_V$ canon.	$\mathbf{D}_v^*$
$\langle \mathrm{H}, \mathrm{T} \rangle$, INT8	LA+β	10.276	14.652	13.939	17.846	10.814	–
	Mamba-2+β	9.755	9.862	9.856	9.769	9.768	–
	GLA+β	9.189	9.231	9.229	9.197	9.201	–
	DeltaNet	9.917	11.155	11.143	9.942	9.947	–
	GDN	9.727	9.800	9.796	9.735	9.736	–
	KDA	9.460	9.520	9.516	9.469	9.466	–
$\langle \mathrm{H}, \mathrm{T} \rangle$, INT6	LA+β	10.276	150.012	135.042	20.437	14.320	–
	Mamba-2+β	9.755	12.676	12.558	9.890	9.890	–
	GLA+β	9.189	10.106	10.107	9.305	9.310	–
	DeltaNet	9.917	46.516	47.351	10.395	10.392	–
	GDN	9.727	11.409	11.354	9.868	9.866	–
	KDA	9.460	10.736	10.742	9.643	9.641	–
$\langle \mathrm{T}, \mathrm{H} \rangle$, INT8	LA+β	10.276	11.001	10.646	17.706	10.757	10.767
	Mamba-2+β	9.755	9.774	9.768	9.766	9.766	9.769
	GLA+β	9.189	9.208	9.209	9.194	9.199	9.203
	DeltaNet	9.917	10.114	10.097	9.930	9.932	9.933
	GDN	9.727	9.743	9.743	9.730	9.729	9.733
	KDA	9.460	9.477	9.478	9.470	9.467	9.468
$\langle \mathrm{T}, \mathrm{H} \rangle$, INT6	LA+β	10.276	16.458	15.579	16.870	12.267	12.090
	Mamba-2+β	9.755	10.030	9.985	9.840	9.843	9.840
	GLA+β	9.189	9.388	9.388	9.303	9.307	9.307
	DeltaNet	9.917	12.795	12.691	10.127	10.134	10.126
	GDN	9.727	9.994	9.960	9.802	9.802	9.802
	KDA	9.460	9.803	9.812	9.607	9.612	9.610
$\langle \mathrm{H}, \mathrm{H} \rangle$, INT8	LA+β	10.276	10.966	10.624	17.411	10.725	10.730
	Mamba-2+β	9.755	9.772	9.768	9.764	9.764	9.767
	GLA+β	9.189	9.205	9.204	9.192	9.199	9.203
	DeltaNet	9.917	10.111	10.098	9.929	9.930	9.932
	GDN	9.727	9.743	9.742	9.729	9.729	9.731
	KDA	9.460	9.473	9.473	9.463	9.462	9.462
$\langle \mathrm{H}, \mathrm{H} \rangle$, INT6	LA+β	10.276	16.132	15.044	19.211	11.632	11.494
	Mamba-2+β	9.755	10.001	9.982	9.819	9.817	9.818
	GLA+β	9.189	9.347	9.347	9.271	9.275	9.273
	DeltaNet	9.917	12.753	12.679	10.085	10.097	10.065
	GDN	9.727	9.975	9.957	9.782	9.786	9.785
	KDA	9.460	9.758	9.768	9.583	9.582	9.581

Zero-shot accuracy. Tables 17 and 18 compare the full-precision model, naive quantizer, and selected stack using per-tensor key scales and INT8 operand codes at 340M and 1.3B.

Table 17: Zero-shot accuracy for the 340M zoo with per-tensor key scales, per-channel value scales, and INT8 operands. We compare full precision, the naive quantizer, and the selected stack using the Language Model Evaluation Harness. HellaSwag and ARC-Challenge use normalized accuracy. The remaining tasks use accuracy.

cell	method	avg	LAMBADA	HellaSwag	ARC-e	ARC-c	PIQA	WinoGrande
LA	fp	0.398	0.267	0.329	0.397	0.242	0.633	0.520
LA	naive	0.356	0.042	0.324	0.394	0.235	0.621	0.518
LA	stack	0.395	0.258	0.330	0.397	0.242	0.630	0.512
LA+β	fp	0.400	0.283	0.327	0.406	0.226	0.635	0.525
LA+β	naive	0.367	0.095	0.322	0.401	0.220	0.635	0.531
LA+β	stack	0.398	0.268	0.326	0.402	0.226	0.635	0.533
Mamba-2	fp	0.410	0.324	0.352	0.414	0.235	0.639	0.495
Mamba-2	naive	0.409	0.310	0.351	0.417	0.239	0.638	0.499
Mamba-2	stack	0.409	0.316	0.351	0.413	0.239	0.639	0.497
Mamba-2+β	fp	0.413	0.316	0.350	0.408	0.240	0.657	0.510
Mamba-2+β	naive	0.412	0.307	0.350	0.408	0.237	0.655	0.513
GLA	fp	0.419	0.335	0.353	0.413	0.241	0.642	0.532
GLA	naive	0.417	0.331	0.353	0.410	0.242	0.638	0.529
GLA+β	fp	0.416	0.332	0.357	0.417	0.234	0.652	0.506
GLA+β	naive	0.415	0.328	0.356	0.416	0.237	0.646	0.504
DeltaNet	fp	0.402	0.273	0.343	0.403	0.234	0.644	0.518
DeltaNet	naive	0.406	0.283	0.345	0.402	0.241	0.645	0.519
GDN	fp	0.418	0.361	0.351	0.406	0.236	0.641	0.511
GDN	naive	0.413	0.349	0.350	0.407	0.235	0.638	0.499
KDA	fp	0.413	0.341	0.357	0.398	0.225	0.660	0.498
KDA	naive	0.413	0.339	0.357	0.397	0.226	0.664	0.495
LA, no norm	fp	0.397	0.266	0.326	0.406	0.238	0.643	0.501
LA, no norm	naive	0.391	0.242	0.326	0.408	0.233	0.639	0.501
LA, no norm	stack	0.394	0.250	0.326	0.403	0.241	0.640	0.502
LA+β, no norm	fp	0.400	0.273	0.324	0.412	0.237	0.643	0.512
LA+β, no norm	naive	0.372	0.150	0.321	0.391	0.228	0.619	0.526
LA+β, no norm	stack	0.380	0.222	0.324	0.383	0.235	0.616	0.500
Mamba-2, no norm	fp	0.402	0.290	0.334	0.404	0.235	0.642	0.507
Mamba-2, no norm	naive	0.401	0.283	0.333	0.404	0.235	0.638	0.513
Mamba-2, no norm	stack	0.401	0.286	0.334	0.405	0.234	0.640	0.510
Mamba-2+β, no norm	fp	0.405	0.304	0.340	0.407	0.230	0.639	0.507
Mamba-2+β, no norm	naive	0.405	0.299	0.338	0.404	0.235	0.638	0.515
Mamba-2+β, no norm	stack	0.404	0.298	0.339	0.407	0.235	0.641	0.503
GLA, no norm	fp	0.406	0.302	0.339	0.412	0.240	0.650	0.492
GLA, no norm	naive	0.406	0.301	0.338	0.412	0.242	0.643	0.503
GLA, no norm	stack	0.405	0.302	0.338	0.412	0.244	0.645	0.493
GLA+β, no norm	fp	0.406	0.314	0.340	0.390	0.229	0.645	0.518
GLA+β, no norm	naive	0.406	0.311	0.340	0.391	0.226	0.649	0.521
GLA+β, no norm	stack	0.405	0.311	0.340	0.391	0.230	0.645	0.515

Table 18: Zero-shot accuracy for the 1.3B zoo with per-tensor key scales, per-channel value scales, and INT8 operands. We compare full precision, the naive quantizer, and the selected stack using the Language Model Evaluation Harness. HellaSwag and ARC-Challenge use normalized accuracy. The remaining tasks use accuracy.

cell	method	avg	LAMBADA	HellaSwag	ARC-e	ARC-c	PIQA	WinoGrande
LA+β	fp	0.507	0.465	0.511	0.521	0.286	0.713	0.545
LA+β	naive	0.447	0.169	0.491	0.500	0.280	0.702	0.538
LA+β	stack	0.493	0.408	0.507	0.516	0.284	0.707	0.537
Mamba-2+β	fp	0.522	0.480	0.539	0.535	0.296	0.717	0.564
Mamba-2+β	naive	0.521	0.469	0.538	0.536	0.295	0.719	0.570
GLA+β	fp	0.544	0.531	0.564	0.553	0.309	0.725	0.581
GLA+β	naive	0.545	0.518	0.565	0.559	0.310	0.726	0.590
DeltaNet	fp	0.505	0.454	0.524	0.528	0.284	0.709	0.531
DeltaNet	naive	0.508	0.458	0.523	0.532	0.288	0.709	0.537
GDN	fp	0.535	0.511	0.538	0.542	0.294	0.720	0.603
GDN	naive	0.532	0.505	0.537	0.545	0.285	0.720	0.597
KDA	fp	0.540	0.522	0.556	0.560	0.292	0.723	0.584
KDA	naive	0.538	0.514	0.556	0.554	0.296	0.721	0.587

F PRODUCTION RESULTS

This appendix reports the production results for Section 4.2. Setup and timing methodology are given in Appendix B.2. We compare with the official SSDi8 release, then examine query–key transformations, value rotation, and decay allocation. Full quality tables follow.

F.1 ABLATIONS

Comparison with the official SSDi8 release. Our implementation closely matches the official SSDi8 release in perplexity and prefill latency across the three Mamba-2 sizes (Table 19). Both use value-side absorption, static per-(H, D_v) scales, and no recurrent rotation.

Small numerical differences remain in the fixed-point decay gate and FP32 rounding of the fused quantizers. For Nemotron-H, where no official SSDi8 implementation is available, the matched baseline uses our implementation of this setting.

Table 19: Mamba-2 baseline validation on WikiText-2. Quantized rows use W4A8 projections, value-side static-(H, D_v) absorption, no rotation, and no compensation. Full-model prefill is measured on the A5000 at $B = 8, T = 2048$.

Method	1.3B	2.7B	8B
WikiText-2 perplexity			
FP16	10.42	9.06	7.25
SSDi8 W4A8 (official implementation)	10.923	9.430	7.629
Value-side baseline (our re-implementation)	10.928	9.441	7.620
Prefill latency (ms)			
SSDi8 W4A8 (official implementation)	616	1184	2563
Value-side baseline (our re-implementation)	609	1163	2542

Query–key rotation and diagonal rescaling. Query–key rotation lowers perplexity at both decay endpoints and under both scale policies, with the largest gains at 8B (Table 22). The endpoints remain close in quality after rotation.

The exponent sweep favors $\kappa = \frac{1}{2}$, with $\frac{1}{4}$ close behind (Table 20). Incorporating rescaling into the existing kernel leaves measured layer latency unchanged, whereas a separate elementwise operation adds overhead.

Table 20: Production query–key rescaling controls. Top: WikiText-2 perplexity with static value-side decay absorption. Bottom: Mamba-2 1.3B layer latency at $B = 1$, relative to the baseline without rescaling. A dash denotes an unreported absolute time.

Rescaling exponent κ	1.3B	8B
$1/4$	10.858	7.435
$1/2$	10.849	7.435
$3/4$	10.866	7.441
Implementation	**Latency (ms)**	**Relative latency**
Without rescaling	1.498	1.00
Fused rescaling	1.498	1.00
Separate elementwise rescaling	–	1.04

Value rotation. Value rotation provides little benefit under the production pipeline's per-channel value scales (Table 21). Storing the state in the rotated basis increases perplexity. Canonical storage restores baseline quality but does not improve it. The same pattern holds with full-precision projections, where adding canonical $\mathbf{R}_V$ to $\mathbf{R}_{QK}$ gives no further gain.

Figure 13 shows that value-rotation error grows in later layers with small recurrence outputs. This suggests a role for the RMSNorm between the recurrence and output projection: normalizing a small

Table 21: Value-rotation diagnostics on Mamba-2 1.3B: WikiText-2 perplexity with an INT8 fixed-point state. The W8A8 control uses chunk size 256 and 32 calibration sequences of 2048 tokens. Canonical storage is evaluated by fake quantization.

Transformation and state basis	FP projections
No rotation	10.496
$\mathbf{R}_V$, rotated storage	10.533
$\mathbf{R}_V$, canonical storage	10.497

output can amplify its quantization error before it enters the residual stream. Canonical storage avoids the measured perplexity penalty but still requires inverse transforms. Since it provides no quality gain under the production scale policy, `LinQuant` leaves the value axis unchanged.

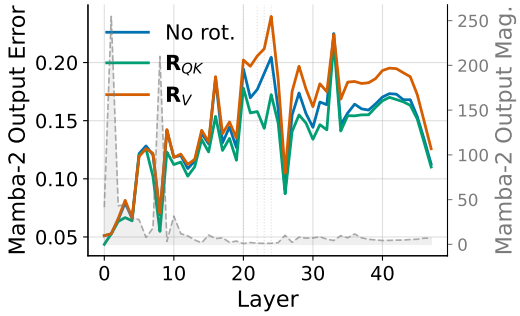


Figure 13: Per-layer relative error against the FP16 recurrence output, alongside its magnitude, for three rotation settings with rotated state storage. Value-rotation error grows in later layers with small recurrence outputs. Section F.1 discusses the interaction with RMSNorm.

Decay allocation. We compare decay placement along two axes: quantization quality and execution cost. Table 22 holds the model and scale policy fixed while moving decay between the key and value, with and without query–key rotation. Figure 14 isolates the affected kernel stage, and Figure 5c varies the operand dimensions.

Table 22: Mamba-2 W4A8 absorption endpoint and rotation ablation, WikiText-2 perplexity, mean ± std over four calibration sets, random Hadamard $\mathbf{R}_{QK}$, without $\mathbf{D}_{qk}$. Single-run results with per-tile dynamic granularities appear in Table 25.

Decay on	Granularity	1.3B		2.7B		8B	
		None	$\mathbf{R}_{QK}$	None	$\mathbf{R}_{QK}$	None	$\mathbf{R}_{QK}$
Value	static-(H, D_v)	10.928 ±0.004	10.860 ±0.007	9.441 ±0.017	9.379 ±0.011	7.620 ±0.009	7.456 ±0.009
Value	dynamic-$(H, 1)$	10.894 ±0.005	10.829 ±0.012	9.433 ±0.006	9.371 ±0.012	7.642 ±0.013	7.447 ±0.008
Key	static-(H, D_k)	10.925 ±0.003	10.849 ±0.008	9.424 ±0.016	9.367 ±0.011	7.601 ±0.009	7.455 ±0.006
Key	dynamic-$(H, 1)$	10.884 ±0.008	10.828 ±0.010	9.424 ±0.011	9.360 ±0.004	7.634 ±0.009	7.444 ±0.009

The endpoints have similar perplexity under the tested scale policies (Table 22). Without rotation, key-side absorption is slightly better in each comparison. Static scales are per channel on the absorbed operand, while dynamic scales are per $(\text{chunk}, H, 1)$ and computed inside the kernel. These results support either placement on quality grounds, consistent with the limited sensitivity to scalar-decay placement at INT8 in the zoo.

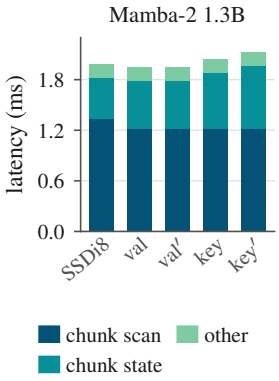


Figure 14: Per-stage Mamba-2 1.3B recurrence latency at $B = 8$, $T = 2048$: the matched value-side baseline and key-/value-side configurations with $\mathbf{R}_{QK}$ and static or dynamic ($'$) scales.

Key-side absorption need not materialize a head-expanded tensor. Each thread block loads its group's keys and applies head-specific decay in registers (Appendix D.2). The remaining difference is in the state write, which reloads the absorbed operand in floating point before weighting and quantizing it. Figure 14 shows that the other stages are unchanged.

Value-side absorption is faster for the tested Mamba-2 geometry, where $D_k = 128$ and $D_v = 64$ (Figure 14). Dynamic scales add cost on the key side because finding the maximum requires another pass over the weighted operand. Figure 15 extends the comparison across batch size and sequence length.

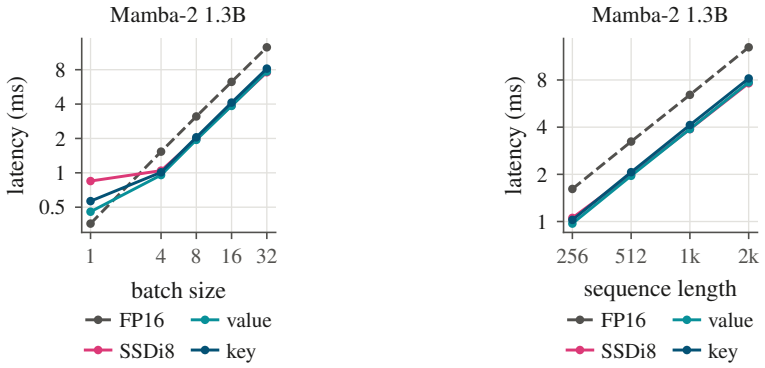


Figure 15: Mamba-2 1.3B recurrence latency against batch size at $T = 2048$ (left) and against sequence length at $B = 32$ (right) for FP16, the official SSDi8 implementation, and the value- and key-side configurations with static scales and $\mathbf{R}_{QK}$. SSDi8 and the value-side configuration have similar latency once launch overhead no longer dominates.

The geometry sweep in Figure 5c shows that wider values can reverse the preferred endpoint at $D_k = 64$, while the key-side penalty persists across the tested widths at $D_k = 128$. Operand dimensions and the kernel's accumulation and rescaling layout therefore both influence the cost of decay placement.

F.2 Full quality tables

Table 23 reports the W4A8 grid with and without diagonal rescaling. Table 24 gives the individual task accuracies for the rescaled configurations. The rotation ablations in Table 25 compare deterministic and random Hadamard signs across decay endpoints and scale policies on WikiText-2 and Pile. Random signs generally perform better and are used in the main experiments.

Table 23: Complete W4A8 production grid. Perplexity is mean ± standard deviation over four calibration sets for Mamba-2 and a single run for Nemotron-H. Avg. is the six-task evaluation average from the default calibration run. "Value" and "Key" denote the decay absorption endpoint. The scale applies to that absorbed operand: static (H, D_v) or (H, D_k), or dynamic per-chunk $(H, 1)$. Dashes indicate unreported configurations.

Configuration	Scale	Mamba-2 1.3B		Mamba-2 2.7B		Mamba-2 8B		Nemo-H Base		Nemo-H Reason.	
		PPL ↓	Avg. ↑	PPL ↓	Avg. ↑	PPL ↓	Avg. ↑	PPL ↓	Avg. ↑	PPL ↓	Avg. ↑
FP16	–	10.420	0.595	9.060	0.638	7.250	0.707	6.030	0.758	8.423	0.731
SSDi8 (official)	static-(H, D_v)	10.923 ±0.007	0.588	9.430	0.626	7.629	0.696	–	–	–	–
Value (R_{QK})	static-(H, D_v)	10.860 ±0.007	0.590	9.379 ±0.011	0.629	7.456 ±0.009	0.703	6.186	0.751	8.666	0.727
+ D_{qk}	static-(H, D_v)	10.849 ±0.011	0.588	9.383 ±0.007	0.627	7.435 ±0.008	0.699	6.178	0.753	8.650	0.728
Value (R_{QK})	dynamic-$(H, 1)$	10.829 ±0.012	0.588	9.371 ±0.012	0.631	7.447 ±0.008	0.704	6.184	0.751	8.657	0.727
+ D_{qk}	dynamic-$(H, 1)$	10.812 ±0.007	0.587	9.363 ±0.004	0.634	7.423 ±0.007	0.697	–	–	–	–
Key (R_{QK})	static-(H, D_k)	10.849 ±0.008	0.589	9.367 ±0.011	0.630	7.455 ±0.006	0.703	6.170	0.750	8.733	0.727
+ D_{qk}	static-(H, D_k)	10.830 ±0.007	0.589	9.372 ±0.003	0.630	7.437 ±0.004	0.697	6.175	0.753	8.670	0.728
Key (R_{QK})	dynamic-$(H, 1)$	10.828 ±0.010	–	9.360 ±0.004	–	7.444 ±0.009	–	–	–	–	–
+ D_{qk}	dynamic-$(H, 1)$	10.807 ±0.005	0.585	9.368 ±0.005	0.630	7.422 ±0.007	0.702	–	–	–	–

Table 24: Per-task zero-shot accuracy of the production configurations with $\mathbf{D}_{qk}$, W4A8 projections, and INT8 recurrence. Evaluation uses the Language Model Evaluation Harness, with normalized accuracy for HellaSwag and ARC-c and accuracy otherwise. Mamba-2 rows use the default calibration set. "Value" and "Key" denote the decay absorption endpoint with the scale of the absorbed operand. The averages are those of Table 23.

Endpoint	Scale	LAMBADA	HellaSwag	ARC-e	ARC-c	PIQA	WinoGrande	Avg.
Mamba-2 1.3B								
Value	static-(H, D_v)	0.644	0.591	0.633	0.323	0.725	0.613	0.588
Value	dynamic-$(H, 1)$	0.640	0.591	0.643	0.332	0.725	0.594	0.587
Key	static-(H, D_k)	0.645	0.593	0.634	0.328	0.726	0.609	0.589
Key	dynamic-$(H, 1)$	0.632	0.591	0.642	0.329	0.723	0.594	0.585
Mamba-2 2.7B								
Value	static-(H, D_v)	0.675	0.652	0.681	0.365	0.755	0.635	0.627
Value	dynamic-$(H, 1)$	0.680	0.655	0.695	0.363	0.757	0.651	0.634
Key	static-(H, D_k)	0.684	0.654	0.689	0.361	0.750	0.642	0.630
Key	dynamic-$(H, 1)$	0.673	0.652	0.689	0.365	0.752	0.649	0.630
Mamba-2 8B								
Value	static-(H, D_v)	0.706	0.767	0.750	0.467	0.795	0.706	0.699
Value	dynamic-$(H, 1)$	0.706	0.769	0.751	0.463	0.791	0.702	0.697
Key	static-(H, D_k)	0.704	0.768	0.748	0.463	0.789	0.707	0.697
Key	dynamic-$(H, 1)$	0.712	0.768	0.751	0.471	0.790	0.719	0.702
Nemotron-H Base								
Value	static-(H, D_v)	0.729	0.804	0.838	0.588	0.814	0.744	0.753
Key	static-(H, D_k)	0.727	0.804	0.838	0.589	0.816	0.747	0.753
Nemotron-H Reasoning								
Value	static-(H, D_v)	0.660	0.799	0.822	0.565	0.791	0.734	0.728
Key	static-(H, D_k)	0.660	0.798	0.822	0.562	0.791	0.735	0.728

Table 25: Mamba-2 W4A8 absorption and rotation ablation on WikiText-2 (WT2) and Pile perplexity, with deterministic (Det.) and random (Rand.) Hadamard rotations, no compensation. Best result for each model size, dataset, and rotation type is bolded.

Absorption	Granularity	Data	Had.	1.3B				2.7B				8B			
				None	$\mathbf{R}_{QK}$	$\mathbf{R}_V$	$\mathbf{R}_{QK}+\mathbf{R}_V$	None	$\mathbf{R}_{QK}$	$\mathbf{R}_V$	$\mathbf{R}_{QK}+\mathbf{R}_V$	None	$\mathbf{R}_{QK}$	$\mathbf{R}_V$	$\mathbf{R}_{QK}+\mathbf{R}_V$
Value	static-(H, D_v)	WT2	Det.	10.930	10.882	11.022	10.914	9.420	9.400	9.447	9.407	7.619	7.463	8.889	7.585
			Rand.	10.930	10.857	11.031	10.886	9.420	9.374	9.447	9.395	7.619	7.467	9.297	7.549
		Pile	Det.	7.415	7.397	7.497	7.428	6.572	6.557	6.596	6.564	6.293	6.213	7.063	6.344
			Rand.	7.415	7.382	7.484	7.407	6.572	6.555	6.604	6.563	6.293	6.206	7.258	6.304
Value	dynamic-(H, D_v)	WT2	Det.	10.885	10.848	10.970	10.849	9.423	9.386	9.434	9.393	7.647	7.452	9.544	7.579
			Rand.	10.885	10.826	10.972	10.819	9.423	9.377	9.427	9.375	7.647	7.456	9.826	7.545
		Pile	Det.	7.408	**7.386**	7.480	7.403	6.570	6.556	6.583	6.559	6.317	6.206	7.371	6.375
			Rand.	7.408	7.370	7.476	7.380	6.570	**6.544**	6.591	6.551	6.317	**6.197**	7.636	6.303
Value	dynamic-$(H, 1)$	WT2	Det.	10.889	10.851	10.973	10.852	9.412	9.384	9.432	9.388	7.639	7.460	9.417	7.579
			Rand.	10.889	10.822	10.984	10.824	9.412	9.373	9.428	9.373	7.639	7.461	9.851	7.537
		Pile	Det.	7.408	7.388	7.473	7.403	6.565	6.554	6.587	6.559	6.315	6.206	7.362	6.362
			Rand.	7.408	7.371	7.464	7.377	6.565	6.547	6.596	6.550	6.315	6.199	7.602	6.302
Key	static-(H, D_k)	WT2	Det.	10.912	10.859	10.980	10.859	9.408	9.392	9.426	9.391	7.599	7.471	8.428	7.555
			Rand.	10.912	10.841	10.990	10.835	9.408	9.370	9.426	9.372	7.599	7.468	8.712	7.520
		Pile	Det.	7.410	7.396	7.473	7.416	6.569	6.560	6.587	6.558	6.283	6.214	6.963	6.335
			Rand.	7.410	7.376	7.459	7.389	6.569	6.548	6.594	6.550	6.283	6.206	7.159	6.279
Key	dynamic-(H, D_k)	WT2	Det.	10.879	**10.845**	10.958	10.846	9.423	**9.383**	9.427	9.389	7.631	**7.450**	9.703	7.577
			Rand.	10.879	10.821	10.952	**10.817**	9.423	9.373	9.417	**9.366**	7.631	7.457	9.842	7.553
		Pile	Det.	7.400	**7.386**	7.473	7.403	6.567	**6.552**	6.582	6.555	6.313	6.207	7.374	6.360
			Rand.	7.400	**7.369**	7.469	7.378	6.567	6.549	6.592	6.552	6.313	**6.197**	7.609	6.311
Key	dynamic-$(H, 1)$	WT2	Det.	10.889	10.848	10.961	10.855	9.404	9.385	9.424	9.387	7.637	7.453	9.547	7.568
			Rand.	10.889	10.818	10.974	**10.817**	9.404	9.377	9.428	**9.366**	7.637	**7.455**	9.801	7.540
		Pile	Det.	7.401	7.389	7.475	7.406	6.565	6.557	6.583	6.554	6.313	**6.205**	7.267	6.344
			Rand.	7.401	7.370	7.459	7.380	6.565	6.548	6.589	6.552	6.313	6.198	7.597	6.300