KALMAN DELTA NETWORKS: UNCERTAINTY-AWARE ASSOCIATIVE MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Linear attention enables efficient long-context inference by compressing token history into a fixed-size recurrent memory. This compression makes each update a trade-off between incorporating new information and preserving useful associations. Models such as DeltaNet, Gated DeltaNet, and KDA predict write strength from the current token representation, without explicitly tracking uncertainty in the stored memory. Yet this uncertainty matters: a new observation should have greater influence when the existing association is uncertain and less when it is already well supported. We introduce *Kalman Delta Networks* (KDNs), a family of linear-attention models that explicitly track memory uncertainty to guide each update. By formulating associative memory as a linear–Gaussian state-space model, KDNs propagate both the memory estimate and its uncertainty, using the Kalman gain to balance accumulated evidence against the reliability of new observations. This formulation also recovers standard delta-rule updates by replacing tracked covariance with a token-predicted isotropic surrogate. To support hardware-efficient training and inference, we derive *Diagonal KDN* and *Isotropic KDN*, which retain one uncertainty value per key channel and per head, respectively. Their uncertainty updates admit associative scans with logarithmic parallel depth, requiring only $O(d_k)$ and $O(1)$ auxiliary state per head. Across controlled pretraining at 750M and 1.3B parameters, both variants consistently improve perplexity and mean downstream accuracy over the evaluated state-of-the-art linear-attention baselines.

1 INTRODUCTION

Self-attention enables flexible retrieval from the preceding context by allowing each query to aggregate the values associated with earlier keys [Vaswani et al., 2017]. This flexibility, however, requires retaining the full token history, so the autoregressive key–value cache grows with context length while query–key interactions remain quadratic in sequence length. Linear attention instead compresses the prefix into a fixed-size recurrent state that can be updated online and evaluated in parallel by a scan [Katharopoulos et al., 2020]. This efficiency turns attention into an online memory-management problem: each token must edit a compressed summary of the past before the model knows which associations future queries will need. An update that is too timid preserves stale information; one that is too aggressive destroys useful memory.

Delta-rule mixers address this problem with a more selective update: they first read the memory's prediction for the current key, then write only the residual [Schlag et al., 2021; Yang et al., 2024b]. This update is commonly interpreted as a single online gradient step on the fast-weight memory's instantaneous squared prediction loss. Gated DeltaNet [Yang et al., 2024a] and KDA [Kimi Team, 2025] retain this residual write while adding Mamba-style state-space transitions [Gu & Dao, 2023; Dao & Gu, 2024] that decay stored content at scalar or channel-wise rates. Their transition and write are therefore typically understood through different lenses: state-space dynamics and online optimization, respectively. More importantly, their write strength is predicted from the current token representation alone, without accounting for the memory's confidence in the stored association. Without tracking this uncertainty, they lack an explicit basis for deciding whether a large residual should revise a tentative association or be discounted to protect a well-supported association. This risks overwriting reliable associations or leaving uncertain ones insufficiently corrected.

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.

In this paper, we provide a unified view of the recurrent updates in delta-rule mixers through a linear–Gaussian state-space formulation, under which uncertainty emerges as their missing state variable. Specifically, we model the recurrent memory as an estimate of a latent, non-stationary key–value map. Each token supplies a noisy observation of that map at one key, while the learned transition describes how the map persists or drifts between observations. Under linear–Gaussian assumptions, the Kalman filter [Kalman, 1960] is the optimal recursive estimator that propagates both the memory estimate and its covariance, balancing memory uncertainty against observation noise to determine the correction gain. The resulting update retains the residual delta-write form while assigning greater weight to new evidence when the memory is uncertain and protecting associations supported by repeated observations. This formulation leads to *Kalman Associative Memory*, which generalizes delta-rule mixers through explicit covariance tracking. DeltaNet, Gated DeltaNet, and KDA thus emerge as covariance-free approximations that retain different memory transitions while omitting this uncertainty recursion.

The exact Kalman update, however, is not a practical linear-attention layer. It carries a dense $d_k \times d_k$ covariance per head, and its gain depends on a state-dependent Riccati recurrence that is poorly suited to efficient parallel scans. To make uncertainty-aware filtering practical, we introduce *Kalman Delta Networks (KDNs)*, a family of scan-compatible approximations to Kalman Associative Memory. We develop two special cases. *Diagonal KDN* imposes diagonal structure on the transition, process noise, and predictive covariance. Because each exact update still yields a dense posterior covariance, we derive an online mean-field variational update that projects the posterior back onto the diagonal family by minimizing reverse KL. Conditional on the retained diagonal predictive prior, this projection preserves the exact one-step posterior mean. To mitigate excessive overwrite under this approximation, we introduce an information-scaling factor that calibrates the retained uncertainty. Diagonal KDN maintains one uncertainty value per key channel and requires $O(d_k)$ auxiliary state per head. In addition, we introduce *Isotropic KDN*, which uses one uncertainty scalar per head, trading expressiveness for $O(1)$ auxiliary state and minimal overhead compared to KDA.

Empirically, under parameter-matched recurrent-only pretraining on FineWeb-Edu [Penedo et al., 2024], both KDN variants achieve lower WikiText and LAMBADA perplexity and higher mean six-task zero-shot accuracy than every evaluated state-of-the-art linear-time recurrent mixer—including Mamba-3 [Lahoti et al., 2026], KDA [Kimi Team, 2025], and GDN-2 [Hatamizadeh et al., 2026]—at both 750M/50B and 1.3B/100B. Diagonal KDN also achieves the highest observed 14-cell RULER aggregate [Hsieh et al., 2024] at both scales. Our contributions are:

- **A principled state-space view of delta-rule models.** We cast the delta rule as an innovation update in a linear–Gaussian state-space model, connecting delta-based recurrent mixers to the Mamba lineage [Gu & Dao, 2023]. DeltaNet, Gated DeltaNet, and KDA use identity, scalar, and diagonal transitions, respectively; unlike Mamba's control-driven additive input write, they correct a predicted key–value map with a key-conditioned residual, while using a token-predicted rather than covariance-derived gain.
- **A hardware-efficient algorithm for uncertainty-aware associative memory.** We derive Diagonal KDN through online variational inference and introduce Isotropic and Diagonal variants with $O(1)$ and $O(d_k)$ uncertainty state per head, respectively. Their uncertainty updates admit an associative gain scan followed by the usual affine memory scan.
- **Overwrite analysis and empirical evidence.** We identify how the diagonal uncertainty approximation can underprotect stored key directions, introduce information scaling to control future overwrite, and show improvements over the evaluated state-of-the-art delta-rule models and Mamba-3 variants in both reported perplexities and mean six-task zero-shot accuracy.

2 PRELIMINARIES

From self-attention to recurrent memory. Causal self-attention [Vaswani et al., 2017] retains all preceding key–value pairs for flexible retrieval, whereas linear attention [Katharopoulos et al., 2020] compresses the prefix into a fixed-size matrix $\boldsymbol{S}_t \in \mathbb{R}^{d_k \times d_v}$. Its write and read operations are:

$$\text{(write)} \quad \boldsymbol{S}_t = \boldsymbol{S}_{t-1} + \boldsymbol{k}_t \boldsymbol{v}_t^\intercal, \qquad \text{(read)} \quad \boldsymbol{o}_t = \boldsymbol{S}_t^\intercal \boldsymbol{q}_t. \tag{1}$$

Queries read from this fixed-size state, avoiding a growing cache. Its additive write, however, cannot revise or forget stored associations.

Delta-rule mixers. Delta-rule mixers address this limitation by first predicting the memory through a transition $\boldsymbol{D}_t$ and then correcting its prediction at the current key with write strength β_t [Schlag et al., 2021; Yang et al., 2024b]:

$$\begin{aligned} \widehat{\boldsymbol{S}}_t = \boldsymbol{D}_t \boldsymbol{S}_{t-1}, \qquad \boldsymbol{S}_t &= \widehat{\boldsymbol{S}}_t + \beta_t \boldsymbol{k}_t \big(\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^\intercal \boldsymbol{k}_t\big)^\intercal \\ &= (\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\intercal) \boldsymbol{D}_t \boldsymbol{S}_{t-1} + \beta_t \boldsymbol{k}_t \boldsymbol{v}_t^\intercal. \end{aligned} \tag{2}$$

DeltaNet uses $\boldsymbol{D}_t = \boldsymbol{I}$, making writes selective but leaving untouched associations unchanged. Gated DeltaNet adds scalar decay $\boldsymbol{D}_t = \alpha_t \boldsymbol{I}$ to forget stale content globally [Yang et al., 2024a]; KDA generalizes this to $\boldsymbol{D}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$, allowing different key channels to retain information at different rates [Kimi Team, 2025]. The progression is therefore from local residual correction, to global forgetting, to channel-wise forgetting. In all three models, however, the scalar β_t is predicted from the current token rather than derived from confidence in the stored association. Appendix B reviews the standard attention and state-space formulations.

Notation. For matrices $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{d_v}]$ and $\boldsymbol{M} = [\boldsymbol{m}_1, \ldots, \boldsymbol{m}_{d_v}]$, we write

$$\boldsymbol{X} \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{M}, \boldsymbol{P}) \quad \Longleftrightarrow \quad \boldsymbol{x}_j \sim \mathcal{N}(\boldsymbol{m}_j, \boldsymbol{P}) \text{ independently for } j = 1, \ldots, d_v. \tag{3}$$

3 Kalman Associative Memory

We revisit DeltaNet-style models through the lens of a linear–Gaussian state-space model, for which the optimal recursive estimator is the Kalman filter [Kalman, 1960]. Under this view, DeltaNet, Gated DeltaNet, and KDA emerge as fixed-gain special cases: they retain the Kalman residual correction but omit covariance tracking and predict the write strength directly from the current input. Forgetting is interpreted as the transition model's prediction of how the latent memory state persists or changes between observations. Tracking the covariance then quantifies confidence in the predicted memory and yields a principled gain for incorporating new evidence.

3.1 Associative Memory as a Linear-Gaussian State-Space Model

To make this filtering view precise, let the memory at time t be a matrix $\boldsymbol{S}_t \in \mathbb{R}^{d_k \times d_v}$ defining the associative map

$$\mathcal{M}_t \colon \mathbb{R}^{d_k} \to \mathbb{R}^{d_v}, \qquad \mathcal{M}_t(\boldsymbol{k}) = \boldsymbol{S}_t^\intercal \boldsymbol{k}. \tag{4}$$

The query read is $\boldsymbol{o}_t = \mathcal{M}_t(\boldsymbol{q}_t) = \boldsymbol{S}_t^\intercal \boldsymbol{q}_t$. The streaming token provides one supervised measurement of this map: under key $\boldsymbol{k}_t$, the memory should return value $\boldsymbol{v}_t$, i.e. $\mathcal{M}_t(\boldsymbol{k}_t) \approx \boldsymbol{v}_t$. This is the shape of a state-space model [Gu & Dao, 2023], in which an unobserved state evolves over time, and each token reveals a noisy linear projection of it.

Latent state. Let $\widetilde{\boldsymbol{S}}_t$ be the latent associative map the recurrent state is trying to track. The transition describes how stored associations persist, decay, or drift before the next measurement arrives:

$$\widetilde{\boldsymbol{S}}_t = \boldsymbol{D}_t \widetilde{\boldsymbol{S}}_{t-1} + \boldsymbol{W}_t, \qquad \boldsymbol{W}_t \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{0}, \boldsymbol{\Omega}_t). \tag{5}$$

The operator $\boldsymbol{D}_t$ is the process model: it is the formal object corresponding to forgetting under information drift, topic drift, and other non-stationarity in the stream. For DeltaNet $\boldsymbol{D}_t = \boldsymbol{I}$; for Gated DeltaNet $\boldsymbol{D}_t = \alpha_t \boldsymbol{I}$; for KDA $\boldsymbol{D}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$. The process noise $\boldsymbol{W}_t$ represents memory drift not captured by deterministic decay, while $\boldsymbol{\Omega}_t$ quantifies uncertainty about that drift. In a language model, this accounts for changes in the latent discourse state that cannot be inferred from retention alone. If an incoming sentence reveals that an entity has moved from Paris to Rome, $\boldsymbol{D}_t$ can attenuate the stale Paris binding, but only the observation can establish the new Rome binding.

Observation. The current key reads one direction of the latent map. The observed value is

$$\boldsymbol{v}_t = \widetilde{\boldsymbol{S}}_t^\intercal \boldsymbol{k}_t + \boldsymbol{e}_t, \qquad \boldsymbol{e}_t \sim \mathcal{N}(0, r_t \boldsymbol{I}_{d_v}). \tag{6}$$

Thus $\boldsymbol{k}_t^\intercal$ is the observation map and $\boldsymbol{v}_t$ is the measurement. The noise term $\boldsymbol{e}_t$ captures the part of the token value that should *not* be treated as a clean memory target. In language modeling, $\boldsymbol{v}_t$ is a contextual feature produced from a noisy token embedding, local syntax, position, layer mixing, and finite model capacity; not all of it is a stable fact that can be stored under key $\boldsymbol{k}_t$. The covariance $r_t \boldsymbol{I}_{d_v}$ sets how reliable this value observation is: small r_t means trust the token and write strongly, while large r_t means treat much of $\boldsymbol{v}_t$ as context-specific noise.

Online filtering problem. We assume $\widetilde{\boldsymbol{S}}_0 \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{0}, \boldsymbol{I})$. Conditioned on the input-dependent quantities $\{(\boldsymbol{D}_t, \boldsymbol{\Omega}_t, \boldsymbol{k}_t, r_t)\}_{t\geq 1}$, the pairs $\{(\boldsymbol{W}_t, \boldsymbol{e}_t)\}_{t\geq 1}$ are independent across time and of $\widetilde{\boldsymbol{S}}_0$, with $\boldsymbol{W}_t \perp \boldsymbol{e}_t$ at each step. Given the observation history through step t, $\mathcal{F}_t = \sigma\{(\boldsymbol{k}_\tau, \boldsymbol{v}_\tau) : \tau \leq t\}$, we estimate the current latent memory by its posterior mean, $\boldsymbol{S}_t = \mathbb{E}[\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_t]$.

3.2 Kalman Filtering

Under the linear–Gaussian assumptions in (5) and (6), the Kalman filter [Kalman, 1960] computes this posterior mean exactly by first predicting the latent memory and then conditioning that prediction on the current token. It carries the filtering law $\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_t \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{S}_t, \boldsymbol{P}_t)$, where $\boldsymbol{S}_t = \mathbb{E}[\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_t]$ and $\boldsymbol{P}_t \in \mathbb{R}^{d_k \times d_k}$ is the common key-space covariance.

Tracking $\boldsymbol{P}_t$ is necessary because each token observes the associative map only at one key direction $\boldsymbol{k}_t$, and the value $\boldsymbol{v}_t$ is noisy. After many tokens, some key directions are well supported by past measurements, while others remain uncertain or have become stale because of drift. The filter uses this uncertainty to calibrate each write, allowing larger corrections in uncertain directions while protecting well-supported associations from noisy measurements.

To derive the filtering recursion, suppose the previous posterior is $\widetilde{\boldsymbol{S}}_{t-1} \mid \mathcal{F}_{t-1} \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{S}_{t-1}, \boldsymbol{P}_{t-1})$. The linear transition with Gaussian process noise preserves this family, giving the predictive distribution $\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_{t-1} \sim \mathcal{N}_{\mathrm{col}}(\widehat{\boldsymbol{S}}_t, \widehat{\boldsymbol{P}}_t)$. The observation in (6) measures each predicted column along the same key $\boldsymbol{k}_t$ with independent Gaussian noise, so the predicted state and $\boldsymbol{v}_t$ are jointly Gaussian. Conditioning this joint Gaussian on $\boldsymbol{v}_t$ yields the residual correction, shared Kalman gain, and covariance update summarized below.

Proposition 3.1: Kalman Associative Memory

Under the linear–Gaussian model in (5)–(6), the Kalman filter gives the exact posterior mean. It predicts memory and uncertainty from $(\boldsymbol{S}_{t-1}, \boldsymbol{P}_{t-1})$ using $(\boldsymbol{D}_t, \boldsymbol{\Omega}_t)$, then updates the memory with the residual $\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^{\intercal}\boldsymbol{k}_t$ weighted by the uncertainty-dependent gain $\boldsymbol{\kappa}_t$.

$$\begin{aligned} &\text{predict:} && \widehat{\boldsymbol{S}}_t = \boldsymbol{D}_t \boldsymbol{S}_{t-1}, && \widehat{\boldsymbol{P}}_t = \boldsymbol{D}_t \boldsymbol{P}_{t-1} \boldsymbol{D}_t^{\intercal} + \boldsymbol{\Omega}_t, \\ &\text{update:} && \boldsymbol{S}_t = \widehat{\boldsymbol{S}}_t + \boldsymbol{\kappa}_t\big(\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^{\intercal}\boldsymbol{k}_t\big)^{\intercal}, && \boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^{\intercal})\widehat{\boldsymbol{P}}_t. \\ &\text{where} && \boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}{r_t + \boldsymbol{k}_t^{\intercal}\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}. \end{aligned} \tag{7}$$

Proof. See Appendix C.1.

Thus the posterior mean is a residual write: the filter adds only the part of the observed value that the predicted memory failed to explain. The corresponding covariance update records the remaining uncertainty after this observation.

3.3 Delta-rule models as fixed-gain Kalman filters

The update above separates two modeling choices: the transition $\boldsymbol{D}_t$ predicts how the memory changes before the token is written, and the gain $\boldsymbol{\kappa}_t$ decides the key-space direction and strength of the residual write. Delta-rule linear attention is the fixed-gain special case of this filter. If we discard the covariance recursion and use an isotropic surrogate $\widehat{\boldsymbol{P}}_t \approx \widehat{b}_t \boldsymbol{I}$ for the predicted uncertainty, then

$$\boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}{r_t + \boldsymbol{k}_t^{\intercal}\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t} \approx \frac{\widehat{b}_t}{r_t + \widehat{b}_t \|\boldsymbol{k}_t\|_2^2}\, \boldsymbol{k}_t \equiv \beta_t \boldsymbol{k}_t. \tag{8}$$

For normalized keys, the exact Kalman gain therefore collapses to the scalar write strength β_t used by the delta-rule family. Substituting $\boldsymbol{\kappa}_t = \beta_t \boldsymbol{k}_t$ into the Kalman update gives the shared fixed-gain residual form. DeltaNet, Gated DeltaNet, and KDA then differ through their process models. Relative to the Kalman optimal update, delta-rule models freeze the uncertainty dynamics and replace the adaptive gain with the fixed first-order direction $\beta_t \boldsymbol{k}_t$.

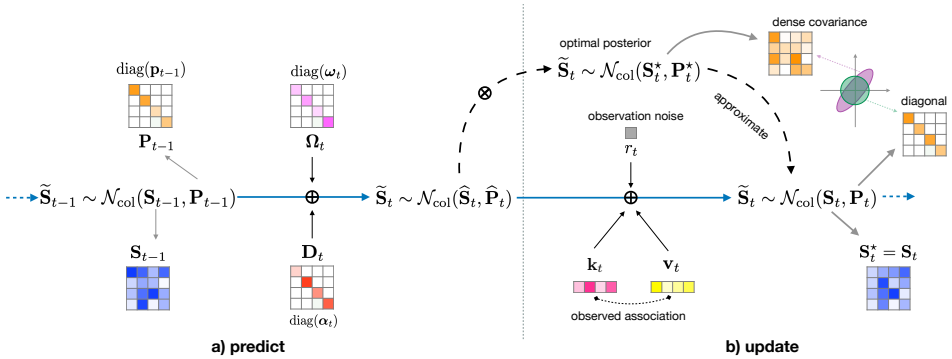

Figure 1: One Kalman Associative Memory update and its mean-field Diagonal KDN approximation. Prediction preserves diagonality; the exact update produces a dense posterior, which reverse-KL projection re-diagonalizes while preserving its mean. Here, $(\boldsymbol{S}_t^\star, \boldsymbol{P}_t^\star)$ is the optimal posterior from Proposition 3.1, and $(\boldsymbol{S}_t, \boldsymbol{P}_t)$ is the variational approximation from Proposition 4.1.

4 Kalman Delta Networks

The Kalman optimal update in (7) is the target, but the exact recursion is not directly compatible with fully parallel linear attention due to its Riccati covariance recursion. This makes the gain depend on accumulated posterior uncertainty and requires a dense $d_k \times d_k$ covariance state per head. We therefore seek a relaxation that preserves both uncertainty-aware updates and parallelizability.

4.1 Diagonal Kalman Delta Network

Let $\mathcal{D}_{++}^{d_k} = \{\text{diag}(\boldsymbol{p}) : \boldsymbol{p} \in \mathbb{R}_{++}^{d_k}\}$. We first restrict the covariance tracked by the filter:

$$\boldsymbol{P}_{t-1} = \text{diag}(\boldsymbol{p}_{t-1}) \in \mathcal{D}_{++}^{d_k}, \qquad \boldsymbol{D}_t = \text{diag}(\boldsymbol{\alpha}_t), \qquad \boldsymbol{\Omega}_t = \text{diag}(\boldsymbol{\omega}_t) \in \mathcal{D}_{++}^{d_k}. \tag{9}$$

The covariance prediction then stays in the diagonal family:

$$\widehat{\boldsymbol{P}}_t = \text{diag}(\widehat{\boldsymbol{p}}_t), \qquad \widehat{\boldsymbol{p}}_t = \boldsymbol{\alpha}_t^2 \odot \boldsymbol{p}_{t-1} + \boldsymbol{\omega}_t. \tag{10}$$

Conditioned on this diagonal predictive covariance, the Kalman gain is still exact:

$$\boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{p}}_t \odot \boldsymbol{k}_t}{r_t + \sum_i \widehat{\boldsymbol{p}}_{t,i} \boldsymbol{k}_{t,i}^2}. \tag{11}$$

Although $\widehat{\boldsymbol{P}}_t$ is diagonal, the exact posterior covariance $\boldsymbol{P}_t^\star = (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^\intercal)\widehat{\boldsymbol{P}}_t$ is generally dense; thus it immediately leaves $\mathcal{D}_{++}^{d_k}$.

Online variational inference. To keep the tracked covariance diagonal, we project the exact posterior after each token onto the mean-field family

$$\mathcal{Q}_{\text{col}} = \left\{ \mathcal{N}_{\text{col}}(\boldsymbol{S}, \text{diag}(\boldsymbol{p})) : \boldsymbol{S} \in \mathbb{R}^{d_k \times d_v},\ \boldsymbol{p} \in \mathbb{R}_{++}^{d_k} \right\}. \tag{12}$$

Let $p(\widetilde{\boldsymbol{S}}_t \mid \boldsymbol{v}_t, \mathcal{F}_{t-1}, \boldsymbol{k}_t, r_t) = \mathcal{N}_{\text{col}}(\boldsymbol{S}_t^\star, \boldsymbol{P}_t^\star)$ denote the exact one-step posterior given by Proposition 3.1. The following proposition characterizes its diagonal variational projection.

Proposition 4.1 (Online diagonal variational update)**.** *Under the linear–Gaussian model with diagonal predictive covariance, treating $\boldsymbol{k}_t$ and r_t as known at step t, the variational approximation*

$$q_t = \arg\min_{q \in \mathcal{Q}_{\text{col}}} \text{KL}(q \,\|\, \mathcal{N}_{\text{col}}(\boldsymbol{S}_t^\star, \boldsymbol{P}_t^\star)) \tag{13}$$

has the unique solution

$$q_t = \mathcal{N}_{\text{col}}(\boldsymbol{S}_t, \text{diag}(\boldsymbol{p}_t)), \tag{14}$$

where

$$\boldsymbol{S}_t = \boldsymbol{S}_t^\star = \widehat{\boldsymbol{S}}_t + \boldsymbol{\kappa}_t \left(\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^\intercal \boldsymbol{k}_t \right)^\intercal, \quad p_{t,i} = \left(\widehat{p}_{t,i}^{-1} + \frac{k_{t,i}^2}{r_t} \right)^{-1}. \tag{15}$$

Thus the projection preserves the exact one-step Kalman posterior mean conditional on the diagonal predictive prior and replaces its shared dense key-space covariance with a diagonal state.

Proof. See Appendix C.2.

Using this variational solution as the next token's prior gives an online mean-field filter. Crucially, the posterior covariance remains in the diagonal family after every token: $\boldsymbol{P}_t = \mathrm{diag}(\boldsymbol{p}_t)$. Defining the token's diagonal precision increment $\boldsymbol{u}_t = (\boldsymbol{k}_t \odot \boldsymbol{k}_t)/r_t$, the covariance update is

$$\widehat{\boldsymbol{p}}_t = \boldsymbol{\alpha}_t^2 \odot \boldsymbol{p}_{t-1} + \boldsymbol{\omega}_t, \qquad \boldsymbol{p}_t = \left(\widehat{\boldsymbol{p}}_t^{-1} + \boldsymbol{u}_t\right)^{-1} = \frac{\widehat{\boldsymbol{p}}_t}{\mathbf{1} + \boldsymbol{u}_t \odot \widehat{\boldsymbol{p}}_t}, \qquad \boldsymbol{p}_0 = \mathbf{1}, \tag{16}$$

where products, ratios, and inverses are coordinatewise. This approximation enables a Möbius associative scan of the covariance recurrence with logarithmic parallel depth, similar to that of Shaj et al. [2026] (see Appendix F.1). Figure 1 summarizes this online predict–update–project cycle.

Information scaling. Diagonal projection discards correlations that encode confidence in combinations of key channels. It can therefore overestimate uncertainty along an already observed key direction, causing repeated or similar keys to trigger overly strong writes. We mitigate this by scaling the post-write precision increment by $\mu > 0$:

$$\boldsymbol{u}_t = \frac{\mu}{r_t}\, \boldsymbol{k}_t \odot \boldsymbol{k}_t. \tag{17}$$

Since $\boldsymbol{\kappa}_t$ is computed beforehand, increasing μ preserves the current write but reduces later effective writes $\beta^{\mathrm{eff}} = \boldsymbol{k}^\intercal \boldsymbol{\kappa}$. Here $\mu = 1$ recovers the variational update, while $\mu = d_k$ offsets the $1/d_k$ information dilution for normalized dense keys (see Appendix D). In Figure 2, increasing μ from 1 to d_k lowers the pooled mean β^{eff}.

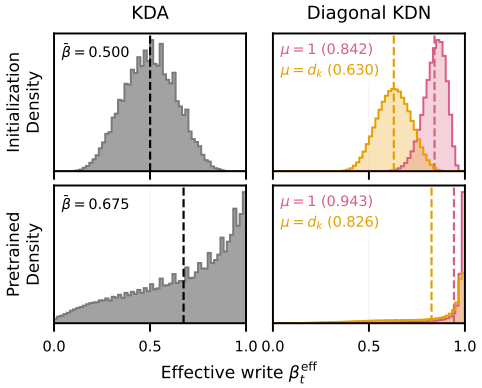

Figure 2: Effective-write distributions before (top) and after 50B-token pretraining (bottom).

The following proposition summarizes the resulting Diagonal Kalman Delta Network.

Proposition 4.2: Diagonal Kalman Delta Network

Let the predictive covariance be restricted to $\mathcal{D}_{++}^{d_k}$ and let $\boldsymbol{\alpha}_t$, $\boldsymbol{\omega}_t$, and r_t be parameterized by

$$\boldsymbol{\alpha}_t = \sigma(\boldsymbol{W}_\alpha \boldsymbol{x}_t + \boldsymbol{b}_\alpha), \quad \boldsymbol{\omega}_t = \mathrm{softplus}(\boldsymbol{W}_\omega \boldsymbol{x}_t + \boldsymbol{b}_\omega), \quad r_t = r_{\min} + \mathrm{softplus}(\boldsymbol{w}_r^\intercal \boldsymbol{x}_t + b_r). \tag{18}$$

Here $\boldsymbol{x}_t$ is the token hidden state, and all weights $\boldsymbol{W}_*$ and $\boldsymbol{w}_*$, together with their biases, are learnable parameters. The covariance states are computed by the per-channel Möbius scan

$$\boldsymbol{M}_{t,i} = \begin{bmatrix} \alpha_{t,i}^2 & \omega_{t,i} \\ u_{t,i}\alpha_{t,i}^2 & 1 + u_{t,i}\omega_{t,i} \end{bmatrix}, \begin{bmatrix} n_{t,i} \\ d_{t,i} \end{bmatrix} = \boldsymbol{M}_{t,i} \begin{bmatrix} n_{t-1,i} \\ d_{t-1,i} \end{bmatrix}, p_{t,i} = \frac{n_{t,i}}{d_{t,i}}, \begin{bmatrix} n_{0,i} \\ d_{0,i} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{19}$$

The adaptive gain at token t uses the pre-write predictive covariance $\widehat{\boldsymbol{p}}_t$:

$$\boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{p}}_t \odot \boldsymbol{k}_t}{r_t + \sum_i \widehat{\boldsymbol{p}}_{t,i} \boldsymbol{k}_{t,i}^2}. \tag{20}$$

With $\boldsymbol{\kappa}_t$ available before the memory scan, the memory update is an affine scan:

$$\boldsymbol{S}_t = (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^\intercal)\, \mathrm{diag}(\boldsymbol{\alpha}_t) \boldsymbol{S}_{t-1} + \boldsymbol{\kappa}_t \boldsymbol{v}_t^\intercal, \quad \boldsymbol{o}_t = \boldsymbol{S}_t^\intercal \boldsymbol{q}_t. \tag{21}$$

Thus the diagonal approximation preserves the two ingredients missing from fixed-gain delta-rule models while keeping the training primitive unchanged: the memory is predicted through $\boldsymbol{D}_t$, and the residual write is weighted by a transition-aware uncertainty estimate.

Hardware-efficient chunkwise implementation. The diagonal approximation directly yields a two-stage parallel implementation. Its per-channel covariance recurrence is a Möbius map, enabling an associative chunkwise kernel that computes all Kalman gains; once those gains are available, the memory update becomes an input-only asymmetric delta rule handled by a compact-WY kernel. Appendix F provides the derivation and implementation details.

Model	Perplexity ↓		Zero-shot accuracy (%) ↑						
	Wiki.	LMB.	LMB.	PIQA	Hella.	Wino.	ARC-e	ARC-c	Avg.
Recurrent-only, 750M parameters, 50B tokens									
DeltaNet [2024b]	19.78	20.17	38.77	69.10	48.04	51.93	65.49	33.28	51.10
Gated DeltaNet [2024a]	19.50	18.08	40.23	69.91	50.22	55.64	**68.10**	32.17	52.71
KDA [2025]	18.85	15.06	44.34	**70.95**	51.23	54.93	67.72	34.04	53.87
Mamba-3 (SISO) [2026]	19.68	17.61	40.33	70.57	50.87	53.83	67.51	34.47	52.93
Mamba-3 (MIMO) [2026]	18.99	15.67	43.24	69.86	**52.24**	56.75	67.72	**36.52**	54.39
GDN-2 [2026]	21.20	17.88	41.18	70.08	46.90	54.54	64.27	31.74	51.45
Isotropic KDN	**18.42**	14.68	44.05	**70.95**	52.12	56.12	68.01	35.24	54.41
Diagonal KDN	18.64	**14.15**	**45.66**	**70.95**	51.91	**57.70**	**68.10**	35.49	**54.97**
Recurrent-only, 1.3B parameters, 100B tokens									
KDA [2025]	15.40	10.09	51.15	74.16	60.61	60.54	73.48	**41.72**	60.28
Mamba-3 (SISO) [2026]	15.94	11.98	47.45	73.61	59.39	57.77	72.64	38.31	58.20
Mamba-3 (MIMO) [2026]	15.63	10.49	50.20	73.94	**60.79**	59.19	73.91	41.04	59.85
GDN-2 [2026]	16.15	11.29	49.45	72.63	57.80	59.19	72.73	39.25	58.51
Isotropic KDN	15.30	9.98	50.96	73.23	60.23	**61.33**	**74.71**	41.64	60.35
Diagonal KDN	**15.04**	**9.75**	**51.87**	**74.21**	60.68	60.62	73.70	41.64	**60.45**
Hybrid and attention-only, 1.3B parameters, 100B tokens									
Transformer [2017] (2K SWA)	16.67	13.10	48.38	71.22	56.62	56.75	68.56	35.84	56.23
KDA [2025] + SWA	15.08	10.81	51.23	72.14	60.33	**61.64**	72.77	41.55	59.94
Mamba-3 (SISO) [2026] + SWA	15.87	11.38	50.46	72.80	59.59	59.27	72.73	40.10	59.16
Mamba-3 (MIMO) [2026] + SWA	15.33	10.96	50.44	72.69	60.00	58.56	72.81	41.21	59.28
GDN-2 [2026] + SWA	16.06	11.12	49.54	71.82	58.44	57.77	71.42	37.63	57.77
Isotropic KDN + SWA	**14.98**	10.90	50.49	72.85	60.31	59.12	72.64	40.87	59.38
Diagonal KDN + SWA	15.10	**10.41**	**51.99**	**72.96**	**60.57**	60.38	**72.98**	**41.81**	**60.11**

Table 1: Language modeling and zero-shot commonsense reasoning.

4.2 Isotropic Kalman Delta Network

Instead of tracking a separate uncertainty for every key channel, we can further restrict the covariance to the isotropic family $\boldsymbol{P}_t \approx b_t \boldsymbol{I}$. Unlike the diagonal family, the isotropic family is not closed under a channel-wise transition, so Isotropic KDN projects both the predicted covariance and the measurement posterior back to this family. For $\boldsymbol{D}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$ and $\boldsymbol{\Omega}_t = \omega_t \boldsymbol{I}$, the trace-projected prediction, uncertainty-derived gain, and memory update are

$$a_t = \frac{1}{d_k}\sum_{i=1}^{d_k} \alpha_{t,i}^2, \quad \widehat{b}_t = a_t b_{t-1} + \omega_t, \quad \beta_t = \frac{\widehat{b}_t}{r_t + \widehat{b}_t \|\boldsymbol{k}_t\|_2^2}, \tag{22}$$
$$\boldsymbol{S}_t = (\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\intercal)\boldsymbol{D}_t \boldsymbol{S}_{t-1} + \beta_t \boldsymbol{k}_t \boldsymbol{v}_t^\intercal.$$

This retains the scalar-gated form used by DeltaNet, Gated DeltaNet, and KDA, but β_t now depends on the transition, process uncertainty, and evidence accumulated from previous tokens through $\widehat{b}_t$. It therefore replaces the independent token gate $\sigma(\boldsymbol{w}_\beta^\intercal \boldsymbol{x}_t + b_\beta)$ with an uncertainty-derived write strength. Appendix E derives the two isotropic projections and the posterior update.

5 Experiments

5.1 Experimental Setup

Pretraining and evaluation datasets. We pretrain all models on FineWeb-Edu [Penedo et al., 2024] at two scales: 750M parameters on 50B tokens and 1.3B parameters on 100B tokens. Evaluations cover language modeling, zero-shot commonsense reasoning, and synthetic and real-world retrieval. Appendix A.1 details the evaluation datasets.

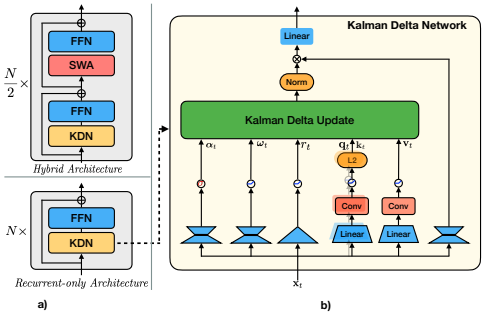

Figure 3: Kalman Delta Network architectures.

Model architectures. KDN builds on KDA [Kimi Team, 2025], replacing its β_t gate with process noise $\boldsymbol{\omega}_t$ and observation noise r_t. We use the recurrent-only and hybrid backbones of GDN-2 [Hatamizadeh et al., 2026], shown in Figure 3, with matched model capacity. At 1.3B, hybrids

Model	S-NIAH-1				S-NIAH-2				S-NIAH-3			MK-NIAH-1		
	1K	2K	4K	8K	1K	2K	4K	8K	1K	2K	4K	1K	2K	4K
Recurrent-only, 1.3B parameters, 100B tokens														
KDA [2025]	99.8	99.2	66.6	30.2	98.0	95.0	62.2	23.6	92.6	77.2	43.2	47.4	38.8	26.6
Mamba-3 (SISO) [2026]	**100.0**	99.8	98.2	62.6	98.6	93.6	62.6	20.6	69.0	50.2	16.6	46.6	38.4	27.2
Mamba-3 (MIMO) [2026]	99.8	99.4	75.2	15.4	**99.6**	**96.0**	64.8	15.4	85.8	63.6	29.8	51.6	42.4	30.4
GDN-2 [2026]	**100.0**	**100.0**	**100.0**	**99.8**	98.6	92.4	67.4	**28.0**	88.0	64.4	29.4	50.6	44.0	31.4
Isotropic KDN	**100.0**	**100.0**	**100.0**	94.6	98.4	91.8	68.4	24.2	86.8	67.6	34.6	41.6	29.8	23.4
Diagonal KDN	**100.0**	**100.0**	99.8	**99.8**	98.6	95.6	**74.6**	25.8	**96.4**	**88.6**	**48.6**	**62.2**	**47.4**	**33.8**

Table 2: In-context retrieval accuracy (%) on RULER single- and multi-key needle-in-a-haystack tasks [Hsieh et al., 2024]. Best per column within each block in **bold**, second best underlined.

Model	SWDE	SQuAD	FDA	TriviaQA	NQ	DROP	Avg.
Recurrent 1.3B / 100B							
KDA [2025]	28.77	**38.70**	27.97	61.97	**24.11**	21.03	33.76
Mamba-3 (SISO) [2026]	26.34	36.35	22.52	60.90	21.98	20.89	31.50
Mamba-3 (MIMO) [2026]	24.93	36.71	25.25	61.02	23.50	21.99	32.23
GDN-2 [2026]	29.90	35.64	21.16	61.49	23.34	20.99	32.09
Isotropic KDN	**33.08**	36.71	22.98	62.09	22.14	23.38	33.40
Diagonal KDN	30.18	38.09	**30.61**	**62.56**	23.63	**24.10**	**34.86**

Table 3: Zero-shot accuracy (%) on real-world retrieval tasks with inputs limited to 2K tokens. *Avg.* is the unweighted six-task mean. Best per column within each block in **bold**, second best underlined.

alternate recurrent layers with 2K sliding-window attention (SWA). Appendix A.1 gives the dimensions and parameter- and state-matching details.

Baselines. We compare with DeltaNet [Yang et al., 2024b], Gated DeltaNet [Yang et al., 2024a], KDA [Kimi Team, 2025], the SISO and MIMO variants of Mamba-3 [Lahoti et al., 2026], and GDN-2 [Hatamizadeh et al., 2026]. Our models are the Isotropic and Diagonal KDN variants developed above. The hybrid block also includes an attention-only Transformer with a 2K sliding window.

5.2 Experimental Results

Language modeling and commonsense reasoning. Table 1 shows consistent gains from KDN across both parameter scales. Among recurrent-only models, both Diagonal and Isotropic KDN outperform KDA and Mamba-3 on the two perplexity benchmarks and in average accuracy at both scales, with Diagonal KDN leading in average accuracy. In the hybrid setting, Diagonal KDN + SWA achieves the highest average accuracy, followed by KDA + SWA, while every recurrent–attention hybrid outperforms the attention-only Transformer.

In-context retrieval. Tables 2 and 3 evaluate synthetic needle-in-a-haystack and real-world retrieval, respectively. Additional results are provided in Appendix A.2. On NIAH, Diagonal KDN achieves the highest aggregate among recurrent-only models at both scales, with particularly strong S-NIAH-3 and multi-key retrieval, thanks to channel-wise uncertainty that protects stored associations under interference. On real-world tasks, Diagonal KDN also achieves the best recurrent-only average and leads FDA, TriviaQA, and DROP. Among hybrids in Tables 5 and 6, Isotropic KDN + SWA has the best average, while Diagonal KDN + SWA ranks second and leads FDA. Additionally, every hybrid model substantially outperforms its recurrent-only counterpart on these tasks.

5.3 Ablation and Runtime

Information scale. Table 4 ablates the information scale using fixed values $\mu \in \{1, \sqrt{d_k}, d_k, 4d_k\}$ and a learned scale initialized at d_k. Increasing the fixed scale steadily improves LAMBADA perplexity, with $4d_k$ performing best, whereas WikiText perplexity and average accuracy peak at d_k. The learned scale performs best on WikiText but trails the fixed d_k setting on LAMBADA and average accuracy, indicating that the overall effect is small and metric-dependent.

μ	Wiki. PPL ↓	LMB. PPL ↓	Avg. ↑
1	18.69	15.51	54.12
$\sqrt{d_k}$	18.93	15.10	54.20
d_k	18.64	14.15	**54.97**
$4d_k$	18.97	**13.91**	54.92
Learned	**18.51**	14.68	54.62

Table 4: Information-scale ablation.

Throughput. Figure 4 benchmarks the mixer layers' throughput using the state-matched 750M configuration on one H200 ($B = 4$, $d_{\text{model}} = 2048$; median over 30 runs). Transformer uses full causal attention, and lengths beyond the 4K training context are scaling probes. Isotropic KDN closely tracks KDA across sequence lengths. Diagonal KDN incurs additional cost from the channel-wise uncertainty scan but remains close to GDN-2 and retains linear scaling. Full attention degrades sharply as sequence length grows.

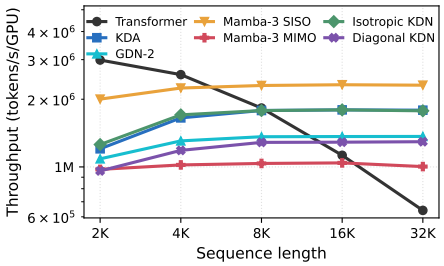

Figure 4: Mixer-layer throughput scaling.

6 Related Work

Fixed-memory attention. Standard self-attention [Vaswani et al., 2017], including its RoPE-based variants [Su et al., 2024], retains token-level history and computes quadratic pairwise interactions. Practical systems reduce these costs through sparse attention [Liu et al., 2025] or bounded KV-cache retention at inference time [Xiao et al., 2024; Bui et al., 2026b;a]. Efficient native alternatives include low-rank projection in Linformer [Wang et al., 2020] and recurrent linear attention, which compresses history into a fixed-size fast-weight state for constant-memory decoding and scan-parallel training [Katharopoulos et al., 2020]. Gated and corrective recurrent designs include GLA, DeltaNet, Gated DeltaNet, Comba, RWKV-7, and KDA [Yang et al., 2023; Schlag et al., 2021; Yang et al., 2024a; Hu et al., 2025; Peng et al., 2025; Kimi Team, 2025]; FoX, DeltaFormer, and PaTH-FoX carry related forgetting, associative correction, and multiplicative positional encoding to exponential-kernel attention [Lin et al., 2025; Zhong et al., 2025; Yang et al., 2025]. GDN-2 further decouples erase and write [Hatamizadeh et al., 2026]. Practical Preconditioned DeltaNet variants [Tumma et al., 2026] combine diagonal curvature with token-predicted gains, whereas KDN derives gains from propagated uncertainty and observation noise.

State-space sequence mixers and Bayesian memory. Structured SSMs provide a complementary approach to fixed-memory sequence mixing: S4 makes long-range continuous-time dynamics tractable via structured convolutions, while S5 recasts them as a scan-parallel MIMO recurrence [Gu et al., 2022; Smith et al., 2023], Mamba adds input selectivity [Gu & Dao, 2023], Mamba-2 derives attention–SSM duality and SSD [Dao & Gu, 2024], and Mamba-3 introduces exponential–trapezoidal discretization, complex dynamics, and SISO/MIMO variants [Lahoti et al., 2026]. KLA [Shaj et al., 2026] augments Mamba-style sequence mixing with a scan-parallel Kalman filter via belief factorization and time-invariant OU dynamics. Unlike Mamba's deterministic, control-driven additive updates, KDN treats token values as noisy observations of a stochastic key–value map and weights each innovation by propagated covariance. Unlike KLA's factorized belief, KDN's exact posterior has dense key-space covariance shared across value channels. Concurrently, Dowling et al. [2026] derive Bayesian Layers with dense covariance propagation. However, their implementation requires quadratic covariance storage and sequential chunk processing, whereas KDN uses scan-compatible isotropic and diagonal approximations.

7 Conclusion and Future Work

We introduced two new recurrent mixers, Isotropic and Diagonal KDNs, that generalize existing delta-rule mixers by capturing uncertainty in stored associations to calibrate residual writes while preserving parallel training. In parameter-matched recurrent-only pretraining at 750M and 1.3B parameters, both variants achieve lower WikiText and LAMBADA perplexity and higher mean six-task zero-shot accuracy than all evaluated recurrent baselines. Diagonal KDN also achieves the highest aggregate RULER score at both scales.

These models are steps toward, rather than full realizations of, Kalman Associative Memory. Exact filtering maintains a dense key-space covariance and a state-dependent Riccati update, whereas the isotropic and diagonal approximations compress this covariance to preserve parallel training. A future direction is to extend diagonal decay with the damped rotations used by Mamba-3 [Lahoti et al., 2026], allowing stored associations to rotate as well as decay. Making this richer transition and covariance update scan-efficient remains open.

AI USE STATEMENT

We used AI tools to polish the writing and assist with implementing and optimizing Triton kernels for Isotropic KDN and Diagonal KDN. We implemented and verified recurrent Python reference versions of both models, which served as correctness references and guardrails for AI agents during kernel implementation and performance optimization. We take responsibility for the final content of this work, including all AI-assisted text and code.

REPRODUCIBILITY STATEMENT

We specify the modeling assumptions and update equations in Sections 3–4, with proofs and further derivations in Appendices C, D, and E. Appendix A.1 documents the datasets, model configurations, parameter- and state-matching procedures, training hyperparameters, and evaluation protocols. Appendix F details the chunkwise gain and memory computations. We also implemented and verified recurrent Python reference versions of both KDN variants to serve as correctness references for Triton kernel development.

REFERENCES

Simran Arora, Brandon Yang, Sabri Eyuboglu, Avanika Narayan, Andrew Hojel, Immanuel Trummer, and Christopher Ré. Language models enable simple systems for generating structured views of heterogeneous data lakes. *Proceedings of the VLDB Endowment*, 2023.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: Closing the recall gap for recurrent language models, 2024. URL https://arxiv.org/abs/2407.05483.

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

Ngoc Bui, Hieu Trung Nguyen, Arman Cohan, and Rex Ying. Make each token count: Towards improving long-context performance with kv cache eviction. *arXiv preprint arXiv:2605.09649*, 2026a.

Ngoc Bui, Shubham Sharma, Simran Lamba, Saumitra Mishra, and Rex Ying. Cache what lasts: Token retention for memory-bounded kv cache in llms. In *International Conference on Learning Representations*, volume 2026, pp. 120316–120342, 2026b.

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

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

Matthew Dowling, Hyungju Jeon, Cristina Savin, and Il Memming Park. Memory by design: Probabilistic sequence layers. *arXiv preprint arXiv:2605.31163*, 2026. URL https://arxiv.org/abs/2605.31163.

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

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

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

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

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024.

Jiaxi Hu, Yongqi Pan, Jusen Du, Disen Lan, Xiaqiang Tang, Qingsong Wen, Yuxuan Liang, and Weigao Sun. Comba: Improving bilinear RNNs with closed-loop control. *arXiv preprint arXiv:2506.02475*, 2025.

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

Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. doi: 10.1115/1.3662552.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

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

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

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

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

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

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

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

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

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8732–8740, 2020.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning (ICML)*, 2021. arXiv:2102.11174.

Vaisakh Shaj, Cameron Barker, Aidan Scannell, Andras Szecsenyi, Elliot J. Crowley, and Amos Storkey. Kalman linear attention: Parallel bayesian filtering for efficient language modelling and state tracking. *arXiv preprint arXiv:2602.10743*, 2026.

Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations*, 2023.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

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

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

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

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

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

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *arXiv preprint arXiv:2406.06484*, 2024b.

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

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

Shu Zhong, Mingyu Xu, Tenglong Ao, and Guang Shi. Understanding transformer from the perspective of associative memory. *arXiv preprint arXiv:2505.19488*, 2025.

A Additional Experiments

A.1 Detailed Experimental Setup

Pretraining and evaluation datasets. We pretrain all models from scratch on FineWeb-Edu [Penedo et al., 2024] at two scales: 750M parameters on 50B tokens and 1.3B parameters on 100B tokens.

- **Language Modeling.** We evaluate language modeling with perplexity on WikiText [Merity et al., 2017] and LAMBADA [Paperno et al., 2016].
- **Zero-shot reasoning.** We evaluate zero-shot reasoning with LAMBADA [Paperno et al., 2016], PIQA [Bisk et al., 2020], HellaSwag [Zellers et al., 2019], WinoGrande [Sakaguchi et al., 2020], and ARC-Easy and ARC-Challenge [Clark et al., 2018].
- **In-context retrieval.** We evaluate in-context retrieval on the single- and multi-key needle-in-a-haystack tasks from RULER [Hsieh et al., 2024]. We use 25-word context increments and deterministic random essay windows rather than 500-word increments and fixed prefixes; every model receives the same 500 samples per cell, with matched keys, values, and needle depths.
- **Real-world retrieval.** Following Just Read Twice (JRT) [Arora et al., 2024], we additionally evaluate real-world retrieval on six cloze-formatted tasks: SWDE [Lockard et al., 2019], SQuAD [Rajpurkar et al., 2018], FDA [Arora et al., 2023], TriviaQA [Joshi et al., 2017], Natural Questions [Kwiatkowski et al., 2019], and DROP [Dua et al., 2019].

Model architectures. The KDN mixer builds on KDA [Kimi Team, 2025], replacing its β_t gate with process noise $\boldsymbol{\omega}_t$ and observation noise r_t. To isolate the mixer, we adopt the recurrent-only and hybrid layer and model architectures of GDN-2 [Hatamizadeh et al., 2026]. This controlled choice might not be optimal; determining the best recurrent-to-attention layer ratio requires a dedicated ablation, as performed for KDA in Kimi Linear [Kimi Team, 2025]. Figure 3 summarizes the recurrent-only and hybrid designs. All recurrent-only models use the same backbone, with feed-forward widths adjusted to match non-embedding parameter counts within 0.03%. The 750M and 1.3B backbones use 16 and 18 layers with $d_{\text{model}} = 2048$ and 2304, respectively, corresponding to KDN mixers with 16 and 18 recurrent heads of dimension 128.

The Mamba-3 baselines are also configured to match the KDNs' total recurrent-state size at both scales. At 1.3B/100B, the hybrid architecture alternates nine recurrent mixer layers with nine sliding-window-attention (SWA) layers using a 2K window.

Implementation details. We use the same training recipe as GDN-2 [Hatamizadeh et al., 2026] for all models: AdamW with a peak learning rate of 4×10^{-4}, $\beta = (0.9, 0.95)$, weight decay 0.1, gradient clipping at 1.0, and a cosine schedule with a 1% token-budget warm-up (0.5B/1B tokens at the two scales). All runs use a global batch of 0.5M tokens, a sequence length of 4K, and the same data seed within each comparison. We set $r_{\min} = 0.01$ and initialize the process noise isotropically by setting $\boldsymbol{W}_\omega = \mathbf{0}$ when training Diagonal KDN. We also learn a scalar initial covariance $c_{0,h} > 0$ for each head h, so that $\widetilde{\boldsymbol{S}}_0^{(h)} \sim \mathcal{N}_{\text{col}}(\mathbf{0}, c_{0,h}\boldsymbol{I}_{d_k})$. Throughout the main experiments, both Isotropic and Diagonal KDN use the fixed information scale $\mu = d_k$.

A.2 Additional Experiment Results

Additional retrieval results. Table 5 reports NIAH results for recurrent-only 750M/50B models and hybrid and attention-only 1.3B/100B models, using the protocol of Table 2. Table 6 reports the hybrid and attention-only real-world retrieval results under the protocol of Table 3.

Model	S-NIAH-1				S-NIAH-2				S-NIAH-3			MK-NIAH-1		
	1K	2K	4K	8K	1K	2K	4K	8K	1K	2K	4K	1K	2K	4K
Recurrent-only, 750M parameters, 50B tokens														
DeltaNet [2024b]	**100.0**	**100.0**	99.6	99.6	97.0	87.0	45.4	15.4	72.0	49.2	17.6	27.2	27.4	22.6
Gated DeltaNet [2024a]	**100.0**	99.8	99.2	87.8	93.2	56.2	27.2	13.8	71.8	23.8	6.6	29.8	25.4	20.2
KDA [2025]	**100.0**	99.8	99.2	82.6	96.4	89.8	63.2	26.0	68.8	43.6	13.6	32.6	29.8	24.2
Mamba-3 (SISO) [2026]	**100.0**	98.8	73.8	33.2	95.4	82.2	43.0	17.6	65.6	26.4	9.0	42.4	27.6	19.6
Mamba-3 (MIMO) [2026]	**100.0**	99.0	79.2	39.6	98.4	**92.6**	57.6	21.6	77.4	49.0	14.2	**50.2**	36.0	25.0
GDN-2 [2026]	**100.0**	99.8	90.2	47.2	95.0	74.6	36.0	12.6	64.6	31.0	9.2	27.8	25.6	22.0
Isotropic KDN	**100.0**	**100.0**	98.2	89.4	96.0	86.0	47.8	16.2	80.4	58.4	22.6	30.4	25.4	22.2
Diagonal KDN	**100.0**	**100.0**	**100.0**	**100.0**	**98.6**	91.4	**71.2**	**29.0**	**84.2**	**71.0**	**32.2**	41.0	**36.4**	**29.6**
Hybrid and attention-only, 1.3B parameters, 100B tokens														
Transformer [2017] (2K SWA)	**100.0**	**100.0**	**52.4**	22.6	**100.0**	**100.0**	**53.4**	19.8	99.2	94.6	35.4	76.6	79.4	45.4
KDA [2025] + SWA	**100.0**	87.8	47.4	21.2	99.8	**100.0**	**53.4**	25.4	99.2	97.0	48.6	86.0	83.0	44.2
Mamba-3 (SISO) [2026] + SWA	99.4	73.8	29.4	15.0	99.6	99.2	52.2	21.8	96.4	86.2	43.4	74.8	76.0	39.0
Mamba-3 (MIMO) [2026] + SWA	**100.0**	99.8	52.0	27.2	99.8	99.8	53.2	25.6	97.8	95.8	50.8	92.8	89.6	45.8
GDN-2 [2026] + SWA	**100.0**	**100.0**	**52.4**	**28.4**	**100.0**	99.8	**53.4**	**27.6**	83.0	67.8	25.2	84.0	76.8	45.6
Isotropic KDN + SWA	**100.0**	**100.0**	**52.4**	**28.4**	**100.0**	**100.0**	**53.4**	**27.6**	**99.8**	98.4	44.6	85.4	89.0	**49.8**
Diagonal KDN + SWA	**100.0**	98.6	51.6	28.0	**100.0**	98.4	**53.4**	26.0	**99.8**	**99.2**	**53.0**	**95.6**	**90.0**	47.8

Table 5: In-context retrieval accuracy (%) for recurrent-only 750M models trained on 50B tokens and hybrid and attention-only 1.3B models trained on 100B tokens. The evaluation protocol matches Table 2. Best per column within each block in **bold**, second best underlined.

Model	SWDE	SQuAD	FDA	TriviaQA	NQ	DROP	Avg.
Hybrid 1.3B / 100B							
Transformer [2017] (2K SWA)	36.08	42.43	52.68	62.09	25.40	21.47	40.02
KDA [2025] + SWA	51.08	43.10	57.86	65.17	28.22	**25.11**	45.09
Mamba-3 (SISO) [2026] + SWA	36.27	43.37	58.95	64.57	26.70	22.47	42.06
Mamba-3 (MIMO) [2026] + SWA	42.55	43.33	64.85	**65.88**	**28.44**	23.53	44.76
GDN-2 [2026] + SWA	47.80	42.73	63.22	62.38	26.10	23.19	44.24
Isotropic KDN + SWA	**53.98**	**43.60**	63.12	65.52	26.89	22.52	**45.94**
Diagonal KDN + SWA	48.08	43.50	**66.58**	64.93	27.72	22.52	45.55

Table 6: Zero-shot real-world retrieval accuracy (%) for hybrid and attention-only 1.3B models trained on 100B tokens, with inputs limited to 2K tokens. *Avg.* is the unweighted six-task mean. Best per column in **bold**, second best underlined.

Length extrapolation. Following Mamba-3 [Lahoti et al., 2026], we evaluate the recurrent-only 1.3B models from Table 1 on WikiText while increasing the context limit from 1K to 32K, eight times the 4K training sequence length. As shown in Figure 5, every model improves on its 4K result at each longer context limit, with most gains saturating by 16K and no abrupt extrapolation failure. Diagonal KDN achieves the lowest word perplexity at every evaluated length, improving from 15.04 at 4K to 14.67 at 32K; Isotropic KDN is consistently second among the recurrent models.

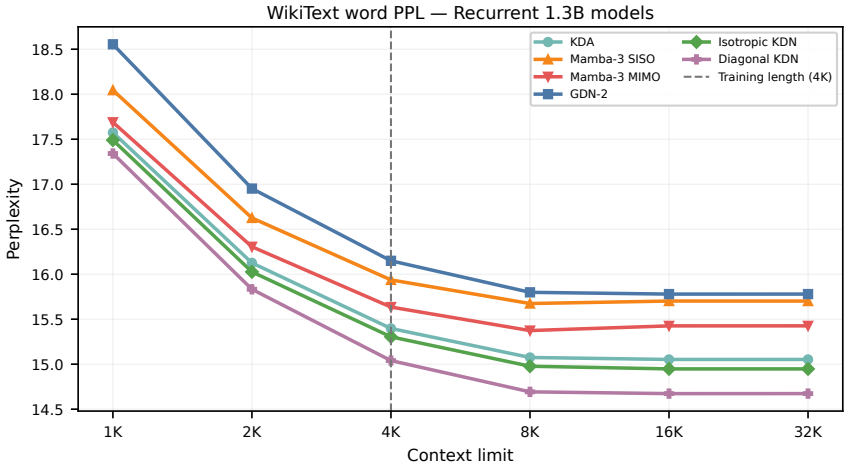

Figure 5: Length extrapolation of recurrent-only 1.3B models trained with 4K sequences. We report WikiText word perplexity (lower is better) as the context limit increases from 1K to 32K; the dashed line marks the training sequence length.

Observation and process noise. Table 7 ablates the observation and process noise at $\mu = 1$ by comparing learned r_t and $\boldsymbol{\omega}_t$ with variants that first fix $r_t = 1$ and then fix both terms to one. Fixing r_t improves WikiText perplexity but slightly degrades LAMBADA perplexity and average accuracy; fixing both terms yields similarly mixed results. Overall, neither constraint consistently improves short-context quality.

Setting ($\mu = 1$)	Wiki. PPL ↓	LMB. PPL ↓	Avg. ↑
Learnable $r_t, \boldsymbol{\omega}_t$	18.69	**15.51**	**54.12**
Fix $r_t = 1$	**18.46**	15.72	53.77
Fix both at 1	18.64	15.84	53.87

Table 7: Observation- and process-noise ablation.

B Attention and State-Space Background

Self-attention. Causal softmax attention stores every preceding key–value pair and retrieves a query-dependent weighted combination [Vaswani et al., 2017],

$$\boldsymbol{o}_t = \sum_{i=1}^{t} \frac{\exp(\boldsymbol{q}_t^\intercal \boldsymbol{k}_i / \sqrt{d_k})}{\sum_{j=1}^{t} \exp(\boldsymbol{q}_t^\intercal \boldsymbol{k}_j / \sqrt{d_k})} \boldsymbol{v}_i. \tag{23}$$

Retaining the complete token history provides direct access to past content, but the autoregressive cache grows linearly with t, and processing a full sequence requires quadratic query–key interactions.

Mamba state-space models. The Mamba family uses the selective linear state-space recurrence

$$\boldsymbol{h}_t = \boldsymbol{A}_t \boldsymbol{h}_{t-1} + \boldsymbol{B}_t \boldsymbol{x}_t, \qquad \boldsymbol{o}_t = \boldsymbol{C}_t^\intercal \boldsymbol{h}_t, \tag{24}$$

where $\boldsymbol{A}_t$, $\boldsymbol{B}_t$, and $\boldsymbol{C}_t$ control retention, writing, and reading. Mamba-1 makes these operations input-dependent [Gu & Dao, 2023], while Mamba-2 connects the resulting selective recurrence to structured masked attention through state-space duality [Dao & Gu, 2024]. Mamba-3 adds exponential–trapezoidal discretization, complex-valued dynamics, and SISO and MIMO variants [Lahoti et al., 2026]. Unlike the delta-rule family in (2), Mamba adds the control $\boldsymbol{B}_t \boldsymbol{x}_t$ directly rather than applying a key-conditioned residual correction.

C Theoretical Proofs

C.1 Proof of the Kalman optimal update

Proof of Proposition 3.1 (Kalman Associative Memory). Let the latent associative-memory state be

$$\widetilde{\boldsymbol{S}}_t \in \mathbb{R}^{d_k \times d_v}, \qquad \mathcal{M}_t(\boldsymbol{k}) = \widetilde{\boldsymbol{S}}_t^{\intercal} \boldsymbol{k}.$$

The linear–Gaussian state-space model is

$$\widetilde{\boldsymbol{S}}_t = \boldsymbol{D}_t \widetilde{\boldsymbol{S}}_{t-1} + \boldsymbol{W}_t, \qquad \boldsymbol{W}_t \sim \mathcal{N}_{\mathrm{col}}(\boldsymbol{0}, \boldsymbol{\Omega}_t), \tag{25}$$

$$\boldsymbol{v}_t = \widetilde{\boldsymbol{S}}_t^{\intercal} \boldsymbol{k}_t + \boldsymbol{e}_t, \qquad \boldsymbol{e}_t \sim \mathcal{N}(0, r_t \boldsymbol{I}_{d_v}). \tag{26}$$

The current model quantities $\boldsymbol{D}_t$, $\boldsymbol{\Omega}_t$, r_t, and $\boldsymbol{k}_t$ are treated as known when the gain is selected. Under these Gaussian assumptions, the filtering posterior at step $t-1$ is

$$p(\widetilde{\boldsymbol{S}}_{t-1} \mid \mathcal{F}_{t-1}) = \mathcal{N}_{\mathrm{col}}(\boldsymbol{S}_{t-1}, \boldsymbol{P}_{t-1}), \tag{27}$$

where $\boldsymbol{S}_{t-1}$ is the posterior mean, i.e. the point estimate of the memory after processing the first $t-1$ key–value pairs, and $\boldsymbol{P}_{t-1}$ is its key-space uncertainty.

Before observing token t, the filter first predicts the next latent memory by marginalizing over the previous posterior:

$$p(\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_{t-1}) = \int p(\widetilde{\boldsymbol{S}}_t \mid \widetilde{\boldsymbol{S}}_{t-1}) p(\widetilde{\boldsymbol{S}}_{t-1} \mid \mathcal{F}_{t-1})\, d\widetilde{\boldsymbol{S}}_{t-1}. \tag{28}$$

Because both factors are linear–Gaussian, the predicted distribution is Gaussian,

$$p(\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_{t-1}) = \mathcal{N}_{\mathrm{col}}(\widehat{\boldsymbol{S}}_t, \widehat{\boldsymbol{P}}_t). \tag{29}$$

Evaluating this Gaussian integral gives the exact Kalman prediction step:

$$\widehat{\boldsymbol{S}}_t = \boldsymbol{D}_t \boldsymbol{S}_{t-1}, \qquad \widehat{\boldsymbol{P}}_t = \boldsymbol{D}_t \boldsymbol{P}_{t-1} \boldsymbol{D}_t^{\intercal} + \boldsymbol{\Omega}_t. \tag{30}$$

At step t, we observe the key–value pair $(\boldsymbol{k}_t, \boldsymbol{v}_t)$, where $\boldsymbol{k}_t^{\intercal}$ is the known observation map. Reading the current predicted memory at $\boldsymbol{k}_t$ gives the expected value $\widehat{\boldsymbol{v}}_t = \widehat{\boldsymbol{S}}_t^{\intercal} \boldsymbol{k}_t$. Since the actual observation is $\boldsymbol{v}_t$, the innovation is its difference from this memory prediction:

$$\boldsymbol{\delta}_t = \boldsymbol{v}_t - \widehat{\boldsymbol{v}}_t. \tag{31}$$

The columnwise predictive state and isotropic observation noise make the pairs $(\widetilde{\boldsymbol{s}}_{t,j}, v_{t,j})$ conditionally independent across j. Conditioned on $(\mathcal{F}_{t-1}, \boldsymbol{k}_t)$, each pair has joint distribution

$$\begin{bmatrix} \widetilde{\boldsymbol{s}}_{t,j} \\ v_{t,j} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \widehat{\boldsymbol{s}}_{t,j} \\ \boldsymbol{k}_t^{\intercal} \widehat{\boldsymbol{s}}_{t,j} \end{bmatrix}, \begin{bmatrix} \widehat{\boldsymbol{P}}_t & \widehat{\boldsymbol{P}}_t \boldsymbol{k}_t \\ \boldsymbol{k}_t^{\intercal} \widehat{\boldsymbol{P}}_t & \gamma_t \end{bmatrix} \right), \qquad \gamma_t := r_t + \boldsymbol{k}_t^{\intercal} \widehat{\boldsymbol{P}}_t \boldsymbol{k}_t. \tag{32}$$

Applying the Gaussian conditioning formula gives

$$\begin{aligned} \widetilde{\boldsymbol{s}}_{t,j} \mid \mathcal{F}_t &\sim \mathcal{N}(\widehat{\boldsymbol{s}}_{t,j} + \boldsymbol{\kappa}_t \delta_{t,j}, \boldsymbol{P}_t)\,, \\ \boldsymbol{\kappa}_t &= \frac{\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}{\gamma_t}, \qquad \boldsymbol{P}_t = \widehat{\boldsymbol{P}}_t - \frac{\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t \boldsymbol{k}_t^{\intercal} \widehat{\boldsymbol{P}}_t}{\gamma_t}. \end{aligned} \tag{33}$$

The posterior remains factorized, with a common gain and covariance across value coordinates. Stacking the columnwise posteriors therefore yields

$$\widetilde{\boldsymbol{S}}_t \mid \mathcal{F}_t \sim \mathcal{N}_{\mathrm{col}}\Big(\widehat{\boldsymbol{S}}_t + \boldsymbol{\kappa}_t \boldsymbol{\delta}_t^{\intercal}, \boldsymbol{P}_t \Big)\,, \qquad \boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^{\intercal}) \widehat{\boldsymbol{P}}_t. \tag{34}$$

This directly establishes the posterior mean and covariance.

It remains to verify the equivalent minimum-MSE characterization. For any candidate gain $\boldsymbol{g} \in \mathbb{R}^{d_k}$, define $\boldsymbol{S}_t(\boldsymbol{g}) = \widehat{\boldsymbol{S}}_t + \boldsymbol{g} \boldsymbol{\delta}_t^{\intercal}$ and

$$J_t(\boldsymbol{g}) := \mathbb{E}\Big[\|\widetilde{\boldsymbol{S}}_t - \boldsymbol{S}_t(\boldsymbol{g})\|_F^2 \;\Big|\; \mathcal{F}_{t-1}, \boldsymbol{k}_t \Big]\,. \tag{35}$$

Using the predictive covariance and independence of the observation noise, direct expansion and completion of the square give

$$\begin{aligned} J_t(\boldsymbol{g}) &= d_v\Big[\operatorname{Tr}(\widehat{\boldsymbol{P}}_t) - 2\boldsymbol{g}^\intercal \widehat{\boldsymbol{P}}_t \boldsymbol{k}_t + \gamma_t \boldsymbol{g}^\intercal \boldsymbol{g}\Big] \\ &= d_v\Big[\operatorname{Tr}(\boldsymbol{P}_t) + \gamma_t \|\boldsymbol{g} - \boldsymbol{\kappa}_t\|_2^2\Big]. \end{aligned} \tag{36}$$

Since $\gamma_t > 0$, this objective has the unique minimizer $\boldsymbol{g} = \boldsymbol{\kappa}_t$, establishing the minimum-MSE characterization. Therefore the exact Kalman Associative Memory update is

$$\begin{aligned} &\widehat{\boldsymbol{S}}_t = \boldsymbol{D}_t \boldsymbol{S}_{t-1}, \qquad && \widehat{\boldsymbol{P}}_t = \boldsymbol{D}_t \boldsymbol{P}_{t-1} \boldsymbol{D}_t^\intercal + \boldsymbol{\Omega}_t, \\ &\boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}{r_t + \boldsymbol{k}_t^\intercal \widehat{\boldsymbol{P}}_t \boldsymbol{k}_t}, && \boldsymbol{S}_t = \widehat{\boldsymbol{S}}_t + \boldsymbol{\kappa}_t \big(\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^\intercal \boldsymbol{k}_t\big)^\intercal, \\ &\boldsymbol{P}_t = (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^\intercal) \widehat{\boldsymbol{P}}_t. \end{aligned} \tag{37}$$

□

C.2 Proof of the online diagonal variational update

Proof of Proposition 4.1 (Online Diagonal Variational Update). Let $\boldsymbol{\Lambda}_t^\star = (\boldsymbol{P}_t^\star)^{-1}$. For $q = \mathcal{N}_{\mathrm{col}}(\boldsymbol{S}, \operatorname{diag}(\boldsymbol{p}))$, the columnwise factorization and the Gaussian KL formula give

$$\begin{aligned} &\mathrm{KL}(q \,\|\, \mathcal{N}_{\mathrm{col}}(\boldsymbol{S}_t^\star, \boldsymbol{P}_t^\star)) \\ &\quad = \frac{1}{2} \operatorname{Tr}[(\boldsymbol{S} - \boldsymbol{S}_t^\star)^\intercal \boldsymbol{\Lambda}_t^\star (\boldsymbol{S} - \boldsymbol{S}_t^\star)] + \frac{d_v}{2} \sum_i [(\boldsymbol{\Lambda}_t^\star)_{ii} p_i - \log p_i] + C, \end{aligned} \tag{38}$$

where $C = \frac{d_v}{2}(\log\det \boldsymbol{P}_t^\star - d_k)$ is independent of $\boldsymbol{S}$ and $\boldsymbol{p}$. Since $\boldsymbol{\Lambda}_t^\star \succ \boldsymbol{0}$, the mean term is uniquely minimized at $\boldsymbol{S} = \boldsymbol{S}_t^\star$. For each $p_i > 0$,

$$\frac{\partial\, \mathrm{KL}}{\partial p_i} = \frac{d_v}{2}\left((\boldsymbol{\Lambda}_t^\star)_{ii} - \frac{1}{p_i}\right), \qquad \frac{\partial^2\, \mathrm{KL}}{\partial p_i^2} = \frac{d_v}{2p_i^2} > 0. \tag{39}$$

Thus the unique minimizing variance is $p_i = 1/(\boldsymbol{\Lambda}_t^\star)_{ii}$. Finally, the exact posterior precision satisfies

$$\boldsymbol{\Lambda}_t^\star = \widehat{\boldsymbol{P}}_t^{-1} + \frac{1}{r_t} \boldsymbol{k}_t \boldsymbol{k}_t^\intercal. \tag{40}$$

Because $\widehat{\boldsymbol{P}}_t = \operatorname{diag}(\widehat{\boldsymbol{p}}_t)$,

$$(\boldsymbol{\Lambda}_t^\star)_{ii} = \widehat{p}_{t,i}^{-1} + \frac{k_{t,i}^2}{r_t}, \qquad p_{t,i} = \left(\widehat{p}_{t,i}^{-1} + \frac{k_{t,i}^2}{r_t}\right)^{-1}. \tag{41}$$

Together with the exact Kalman posterior mean $\boldsymbol{S}_t^\star$ from Proposition 3.1, this proves (15). □

D Overwrite and Protection–Write Decoupling

Empirical symptom. For any delta-style residual update, define the effective write strength

$$\beta_t^{\mathrm{eff}} = \boldsymbol{k}_t^\intercal \boldsymbol{\kappa}_t. \tag{42}$$

Reading the updated memory at the same key shows its operational meaning:

$$\boldsymbol{v}_t - \boldsymbol{S}_t^\intercal \boldsymbol{k}_t = \big(1 - \beta_t^{\mathrm{eff}}\big)\left(\boldsymbol{v}_t - \widehat{\boldsymbol{S}}_t^\intercal \boldsymbol{k}_t\right). \tag{43}$$

Thus β_t^{eff} is the fraction of the current residual removed at its own key; values near one correspond to an almost complete same-key rewrite. We call an update an *overwrite* when β_t^{eff} remains large despite strong prior evidence along $\boldsymbol{k}_t$, causing a new observation to replace rather than cautiously revise the stored association.

To analyze the write strength induced by KDN, we record β_t^{eff} across layers, heads, and tokens on a fixed FineWeb-Edu stream for parameter-matched 750M KDA and Diagonal KDN models, both at initialization and after 50B training tokens. The resulting distributions are shown in Figure 2. At $\mu = 1$, Diagonal KDN concentrates its effective writes near one, indicating frequent overwrite. Increasing μ from 1 to d_k shifts the distribution toward lower effective write strength.

Information scale as future protection. To understand overwrite, consider a normalized key with equal energy on m active channels and isotropic predictive variance on that support: $k_i^2 = 1/m$ and $\widehat{p}_i = p$. Along this key, the exact posterior and its unscaled diagonal projection retain variances

$$V_{\text{exact}} := \boldsymbol{k}^\intercal \boldsymbol{P}^\star \boldsymbol{k} = \frac{rp}{r+p}, \qquad V_{\text{proj}} := \boldsymbol{k}^\intercal \operatorname{diag}(\boldsymbol{p})\boldsymbol{k} = \frac{mrp}{mr+p}. \tag{44}$$

For $m > 1$, $V_{\text{proj}} > V_{\text{exact}}$. Consider an immediate repeat of the same key with no intervening covariance evolution, so that the retained variance V becomes the predictive variance for the next write. Its effective strength is

$$\beta_{t+1}^{\text{eff}}(V) = \frac{V}{r_{t+1} + V}. \tag{45}$$

Because this fraction increases with V, the diagonal projection assigns a stronger correction to the repeated key than the exact posterior.

We introduce an information scale μ to reduce this retained uncertainty. For the causal analysis, let it vary by token as μ_t; the model uses a single constant μ:

$$p_{t,i}(\mu_t) = \left(\widehat{p}_{t,i}^{-1} + \mu_t \frac{k_{t,i}^2}{r_t} \right)^{-1}. \tag{46}$$

For the balanced key above, the scaled projection retains

$$V_{\text{proj}}(\mu) = \frac{mrp}{mr + \mu p}. \tag{47}$$

Thus $\mu = m$ matches the exact one-step directional variance; for an equal-energy key with full support, this gives $\mu = d_k$.

The gain $\boldsymbol{\kappa}_t$ is computed from the predictive state $\widehat{\boldsymbol{p}}_t$ before the scaled uncertainty update, so μ_t does not change the current gain or memory estimate:

$$\frac{\partial \boldsymbol{\kappa}_t}{\partial \mu_t} = \boldsymbol{0}, \qquad \frac{\partial \boldsymbol{S}_t}{\partial \mu_t} = \boldsymbol{0}, \qquad \frac{\partial p_{t,i}}{\partial \mu_t} = -\frac{k_{t,i}^2}{r_t} p_{t,i}^2 \le 0. \tag{48}$$

Its first effect is on future gains. Holding the next token's parameters fixed and defining $z_{t+1} = \sum_i \widehat{p}_{t+1,i} k_{t+1,i}^2$, we obtain

$$\begin{aligned} \frac{\partial z_{t+1}}{\partial \mu_t} &= -\frac{1}{r_t} \sum_i \alpha_{t+1,i}^2 p_{t,i}^2 k_{t,i}^2 k_{t+1,i}^2 \le 0, \\ \frac{\partial \beta_{t+1}^{\text{eff}}}{\partial \mu_t} &= \frac{r_{t+1}}{(r_{t+1} + z_{t+1})^2} \frac{\partial z_{t+1}}{\partial \mu_t} \le 0. \end{aligned} \tag{49}$$

Thus μ is a protection scale: it leaves the current write unchanged while weakening future writes along channels shared with the current key.

The matched 750M sweep exhibits the predicted reduction in write strength. At both initialization and after training, the pooled mean β^{eff} decreases monotonically as μ increases:

Information scale μ	Random init.	Trained at μ	Fixed-weight intervention
1	0.842	0.943	0.886
$\sqrt{d_k}$	0.743	0.891	0.873
d_k	0.630	0.826	0.862
$4d_k$	0.580	0.805	0.857

Table 8: Mean effective write for the 750M Diagonal KDN ($d_k = 128$). The first two columns use the matched random-initialization controls and checkpoints trained at each scale, evaluated on eight 2048-token sequences. The final column uses a separate 4096-token trajectory, changes only the runtime information scale of each trained checkpoint, and averages the resulting layer–head–token means across the four checkpoints; model weights are fixed, while downstream hidden states respond to the changed recurrence. Absolute levels should therefore be compared within, not across, these probe protocols. The intervention isolates the monotone protection effect from retraining compensation.

The intervention reduces the mean from 0.886 at $\mu = 1$ to 0.857 at $4d_k$, while the matched trained models decrease from 0.943 to 0.805. Together with Equation (49), this supports the interpretation of μ as a future-write control rather than an increase in current observation strength.

E Isotropic Kalman Delta Network

Retained isotropic state. Following the notation of Section 2, $\widetilde{\boldsymbol{S}}_t$ is the latent random memory, while $\boldsymbol{S}_t$ and $\boldsymbol{P}_t$ are the retained filtering mean and covariance. Isotropic KDN replaces the d_k diagonal uncertainty values of Diagonal KDN by one scalar per head:

$$\boldsymbol{P}_t = b_t \boldsymbol{I}, \qquad b_t > 0. \tag{50}$$

Two isotropic projections. The isotropic family is not closed under either stage of the Kalman recursion. Starting from $\boldsymbol{P}_{t-1} = b_{t-1}\boldsymbol{I}$, a channel-wise transition $\boldsymbol{D}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$ generally produces an anisotropic predictive covariance. Even after projecting that prediction back to an isotropic covariance, conditioning on a key $\boldsymbol{k}_t$ produces a dense posterior. Isotropic KDN therefore projects once after prediction and once after conditioning.

For any $\boldsymbol{X} \in \mathbb{S}^{d_k}_{++}$, the Frobenius-nearest isotropic matrix is

$$\Pi_{\mathrm{iso}}(\boldsymbol{X}) = \operatorname*{arg\,min}_{b\boldsymbol{I}:\, b>0} \frac{1}{2}\|\boldsymbol{X} - b\boldsymbol{I}\|_F^2 = \frac{1}{d_k}\operatorname{Tr}(\boldsymbol{X})\boldsymbol{I}. \tag{51}$$

Indeed, $\|\boldsymbol{X} - b\boldsymbol{I}\|_F^2 = \|\boldsymbol{X}\|_F^2 - 2b\operatorname{Tr}(\boldsymbol{X}) + d_k b^2$, so the projection preserves the average eigenvalue. We apply this covariance projection after prediction, where the process update is affine. After conditioning, we instead minimize reverse KL over isotropic Gaussians, which preserves the exact posterior mean and retains the harmonic mean of the posterior covariance eigenvalues.

Predictive covariance. The memory mean follows the full channel-wise prediction $\widehat{\boldsymbol{S}}_t = \boldsymbol{D}_t \boldsymbol{S}_{t-1}$. With isotropic process noise $\boldsymbol{\Omega}_t = \omega_t \boldsymbol{I}$, the retained predictive covariance is

$$\begin{aligned} a_t &= \frac{1}{d_k}\operatorname{Tr}(\boldsymbol{D}_t^2) = \frac{1}{d_k}\sum_{i=1}^{d_k} \alpha_{t,i}^2, \qquad \widehat{b}_t = a_t b_{t-1} + \omega_t, \\ \widehat{\boldsymbol{P}}_t &= \Pi_{\mathrm{iso}}(\boldsymbol{D}_t \boldsymbol{P}_{t-1} \boldsymbol{D}_t^{\intercal} + \boldsymbol{\Omega}_t) = \widehat{b}_t \boldsymbol{I}. \end{aligned} \tag{52}$$

Thus $\widehat{b}_t$ is the average predictive marginal variance. The projection is inactive when $\boldsymbol{\alpha}_t$ is constant across channels, as in the scalar decay of Gated DeltaNet [Yang et al., 2024a]. For channel-wise decay as in KDA, the memory retains the full transition while the uncertainty tracker keeps only its isotropic projection.

Posterior covariance. By the columnwise factorization, it suffices to consider value coordinate j. Let $\widehat{\boldsymbol{s}}_{t,j}$ and $\boldsymbol{s}_{t,j}$ denote column j of the predicted memory $\widehat{\boldsymbol{S}}_t$ and filtered memory $\boldsymbol{S}_t$, respectively. Under the retained predictive state, the one-step model is

$$\widetilde{\boldsymbol{s}}_{t,j} \mid \mathcal{F}_{t-1} \sim \mathcal{N}(\widehat{\boldsymbol{s}}_{t,j}, \widehat{\boldsymbol{P}}_t), \qquad v_{t,j} = \boldsymbol{k}_t^{\intercal}\widetilde{\boldsymbol{s}}_{t,j} + e_{t,j}, \qquad e_{t,j} \sim \mathcal{N}(0, r_t). \tag{53}$$

The exact one-step filtering posterior under (53), treating $\boldsymbol{k}_t$ and r_t as known at step t, has the Kalman residual-update mean and a generally dense covariance:

$$\begin{aligned} \boldsymbol{\kappa}_t &= \beta_t \boldsymbol{k}_t, & \beta_t &= \frac{\widehat{b}_t}{r_t + \widehat{b}_t\|\boldsymbol{k}_t\|_2^2}, \\ \boldsymbol{s}_{t,j} &= \widehat{\boldsymbol{s}}_{t,j} + \boldsymbol{\kappa}_t\left(v_{t,j} - \boldsymbol{k}_t^{\intercal}\widehat{\boldsymbol{s}}_{t,j}\right), \\ \boldsymbol{P}_t^{\star} &= (\boldsymbol{I} - \boldsymbol{\kappa}_t \boldsymbol{k}_t^{\intercal})\widehat{\boldsymbol{P}}_t \\ &= \widehat{b}_t \boldsymbol{I} - \frac{\widehat{b}_t^2 \boldsymbol{k}_t \boldsymbol{k}_t^{\intercal}}{r_t + \widehat{b}_t\|\boldsymbol{k}_t\|_2^2}. \end{aligned} \tag{54}$$

The star distinguishes this exact posterior covariance from the retained isotropic covariance $\boldsymbol{P}_t$. Because $\boldsymbol{P}_t^{\star}$ is generally dense, we approximate the exact one-step posterior within the isotropic family

$$\mathcal{Q}_{\mathrm{iso}} = \{\mathcal{N}(\boldsymbol{s}, b\boldsymbol{I}) : \boldsymbol{s} \in \mathbb{R}^{d_k},\ b > 0\}.$$

Proposition E.1 (Online isotropic variational update)**.** *Let $\pi_{t,j} = p(\widetilde{\boldsymbol{s}}_{t,j} \mid v_{t,j}, \mathcal{F}_{t-1}, \boldsymbol{k}_t, r_t)$ denote the exact one-step filtering posterior under* (53)*. The variational approximation is*

$$q_{t,j} = \arg\min_{q \in \mathcal{Q}_{\text{iso}}} \mathrm{KL}(q \,\|\, \pi_{t,j})\,. \tag{55}$$

Its unique solution is

$$\begin{aligned} q_{t,j} &= \mathcal{N}(\boldsymbol{s}_{t,j}, b_t \boldsymbol{I}), \\ \boldsymbol{s}_{t,j} &= \widehat{\boldsymbol{s}}_{t,j} + \boldsymbol{\kappa}_t \left(v_{t,j} - \boldsymbol{k}_t^\intercal \widehat{\boldsymbol{s}}_{t,j}\right), \\ b_t &= \left(\frac{1}{\widehat{b}_t} + \frac{\|\boldsymbol{k}_t\|_2^2}{d_k r_t}\right)^{-1} = \frac{\widehat{b}_t}{1 + \widehat{b}_t \|\boldsymbol{k}_t\|_2^2/(d_k r_t)}, \qquad \boldsymbol{\kappa}_t = \beta_t \boldsymbol{k}_t. \end{aligned} \tag{56}$$

Thus the retained mean $\boldsymbol{s}_{t,j}$ is the exact one-step posterior mean, while $b_t \boldsymbol{I}$ is the isotropic approximation to its covariance.

Proof. The exact posterior is Gaussian with mean $\boldsymbol{s}_{t,j}$ and covariance $\boldsymbol{P}_t^\star$ from (54). Its covariance eigenvalue along $\boldsymbol{k}_t$ and its eigenvalue on the orthogonal complement are, respectively,

$$\lambda_\parallel = \frac{\widehat{b}_t r_t}{r_t + \widehat{b}_t \|\boldsymbol{k}_t\|_2^2}, \qquad \lambda_\perp = \widehat{b}_t. \tag{57}$$

For $q = \mathcal{N}(\boldsymbol{s}, b\boldsymbol{I})$, minimizing the Gaussian KL first sets $\boldsymbol{s} = \boldsymbol{s}_{t,j}$. Up to constants independent of b, the remaining objective is

$$\frac{1}{2}\left[b\left(\lambda_\parallel^{-1} + (d_k - 1)\lambda_\perp^{-1}\right) - d_k \log b\right]. \tag{58}$$

Its unique minimizer is the harmonic mean of these covariance eigenvalues,

$$b_t = \frac{d_k}{\lambda_\parallel^{-1} + (d_k - 1)\lambda_\perp^{-1}} = \left(\frac{1}{\widehat{b}_t} + \frac{\|\boldsymbol{k}_t\|_2^2}{d_k r_t}\right)^{-1}, \tag{59}$$

which is (56). □

Starting from the isotropic variational update above, we introduce an information scale $\mu > 0$ by setting $b_t^{-1} - \widehat{b}_t^{-1} = \mu \|\boldsymbol{k}_t\|_2^2/(d_k r_t)$; $\mu = 1$ recovers the variational update. Like the per-channel diagonal recurrence, the resulting scalar recurrence is a Möbius map and therefore admits an associative prefix scan. Isotropic KDN is summarized below.

Proposition E.2: Isotropic Kalman Delta Network

Let the predictive covariance be restricted to the isotropic family $\{b\boldsymbol{I} : b > 0\}$ and let $\boldsymbol{D}_t$, $\boldsymbol{\Omega}_t$, and r_t be parameterized by

$$\begin{aligned} &\boldsymbol{\alpha}_t = \sigma(\boldsymbol{W}_\alpha \boldsymbol{x}_t + \boldsymbol{b}_\alpha), \quad \omega_t = \mathrm{softplus}(\boldsymbol{w}_\omega^\intercal \boldsymbol{x}_t + b_\omega), \quad r_t = r_{\min} + \mathrm{softplus}(\boldsymbol{w}_r^\intercal \boldsymbol{x}_t + b_r), \\ &\boldsymbol{D}_t = \mathrm{diag}(\boldsymbol{\alpha}_t), \qquad\qquad \boldsymbol{\Omega}_t = \omega_t \boldsymbol{I}. \end{aligned} \tag{60}$$

Here $\boldsymbol{x}_t$ is the token hidden state, and all weights $\boldsymbol{W}_*$ and $\boldsymbol{w}_*$, together with their biases, are learnable parameters. Define the trace-averaged transition factor $a_t = d_k^{-1} \sum_i \alpha_{t,i}^2$. For an information scale $\mu > 0$, define

$$u_t = \mu \frac{\|\boldsymbol{k}_t\|_2^2}{d_k r_t}, \qquad \widehat{b}_t = a_t b_{t-1} + \omega_t, \qquad b_t = \frac{\widehat{b}_t}{1 + u_t \widehat{b}_t}, \qquad b_0 = 1. \tag{61}$$

The covariance states are computed by the scalar Möbius scan

$$\boldsymbol{M}_t = \begin{bmatrix} a_t & \omega_t \\ u_t a_t & 1 + u_t \omega_t \end{bmatrix}, \qquad \begin{bmatrix} n_t \\ d_t \end{bmatrix} = \boldsymbol{M}_t \begin{bmatrix} n_{t-1} \\ d_{t-1} \end{bmatrix}, \qquad b_t = \frac{n_t}{d_t}, \qquad \begin{bmatrix} n_0 \\ d_0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{62}$$

The adaptive gain at token t uses the pre-write predictive covariance $\widehat{b}_t$:

$$\boldsymbol{\kappa}_t = \frac{\widehat{b}_t \boldsymbol{k}_t}{r_t + \widehat{b}_t \|\boldsymbol{k}_t\|_2^2} \equiv \beta_t \boldsymbol{k}_t. \tag{63}$$

With β_t available before the memory scan, the memory update has the same affine form as KDA:

$$\boldsymbol{S}_t = (\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\intercal)\boldsymbol{D}_t \boldsymbol{S}_{t-1} + \beta_t \boldsymbol{k}_t \boldsymbol{v}_t^\intercal, \quad \boldsymbol{o}_t = \boldsymbol{S}_t^\intercal \boldsymbol{q}_t. \tag{64}$$

At each step, the gain is formed from $\widehat{b}_t$ before b_t is updated, so the token-t information increment does not affect the token-t write; it affects only subsequent gains. In all reported Isotropic KDN runs, we fix the information scale to $\mu = d_k$.

Isotropic KDN therefore preserves the predict–update decomposition and Kalman residual write, but compresses the uncertainty state to one scalar per head. Its gain is constrained to $\boldsymbol{\kappa}_t = \beta_t \boldsymbol{k}_t$; Diagonal KDN replaces b_t by the vector $\boldsymbol{p}_t$ and permits an anisotropic gain direction.

F Hardware-Efficient Chunkwise Algorithm

Diagonal KDN is evaluated in two stages. A chunked Möbius scan first computes every adaptive gain $\boldsymbol{\kappa}_t$ from the diagonal covariance recurrence. Once these gains are available, the memory recurrence is an input-only asymmetric delta rule and can be evaluated with a compact WY transform, following the chunkwise algorithms developed for DeltaNet, KDA, and GDN-2 [Yang et al., 2024b; Kimi Team, 2025; Hatamizadeh et al., 2026]. The two stages therefore use a Möbius scan and a WY memory scan, respectively; both expose parallel chunk-local work and retain only a short recurrence across chunks.

F.1 Chunkwise Gain Computation

Associative covariance scan. Because the retained covariance is diagonal, each channel evolves independently. Substituting (17), with $u_{t,i} = \mu k_{t,i}^2 / r_t$, yields the Möbius recurrence

$$p_{t,i} = \frac{\alpha_{t,i}^2 p_{t-1,i} + \omega_{t,i}}{u_{t,i}\alpha_{t,i}^2 p_{t-1,i} + 1 + u_{t,i}\omega_{t,i}}. \tag{65}$$

Representing each Möbius update by a 2×2 matrix turns composition into matrix multiplication and yields the associative scan

$$\boldsymbol{M}_{t,i} = \begin{bmatrix} \alpha_{t,i}^2 & \omega_{t,i} \\ u_{t,i}\alpha_{t,i}^2 & 1 + u_{t,i}\omega_{t,i} \end{bmatrix}, \quad \begin{bmatrix} n_{t,i} \\ d_{t,i} \end{bmatrix} = \boldsymbol{M}_{t,i} \begin{bmatrix} n_{t-1,i} \\ d_{t-1,i} \end{bmatrix}, \quad p_{t,i} = \frac{n_{t,i}}{d_{t,i}}, \quad \begin{bmatrix} n_{0,i} \\ d_{0,i} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{66}$$

Thus the approximation reduces the uncertainty state from a dense $d_k \times d_k$ matrix to d_k scalars and replaces the sequential Riccati recursion with an associative scan of logarithmic parallel depth.

Chunkwise gain scan. Let $\boldsymbol{z}_{t,i} = [n_{t,i}, d_{t,i}]^\intercal$ be the homogeneous covariance state in (66). Partition the sequence into gain chunks $\mathcal{I}_m = \{\ell_m, \dots, r_m\}$ of length at most C_g. Within each chunk, the token-local Möbius matrices are reduced in temporal order to

$$\overline{\boldsymbol{M}}_{m,i} = \boldsymbol{M}_{r_m,i} \boldsymbol{M}_{r_m-1,i} \cdots \boldsymbol{M}_{\ell_m,i}. \tag{67}$$

An exclusive prefix scan over these summaries gives the state entering every chunk,

$$\boldsymbol{z}_{m,i}^{\text{in}} = \overline{\boldsymbol{M}}_{m-1,i} \cdots \overline{\boldsymbol{M}}_{0,i} \boldsymbol{z}_{0,i}, \qquad \boldsymbol{z}_{0,i} = \begin{bmatrix} p_{0,i} \\ 1 \end{bmatrix}, \tag{68}$$

where the product is the identity for the first chunk. Starting from this exclusive carry, all chunks are replayed in parallel. For $t \in \mathcal{I}_m$, the state immediately before token t is

$$\boldsymbol{z}_{t-1,i} = \boldsymbol{M}_{t-1,i} \cdots \boldsymbol{M}_{\ell_m,i} \boldsymbol{z}_{m,i}^{\text{in}}, \tag{69}$$

with an empty product at $t = \ell_m$. The gain is emitted from this exclusive state before $\boldsymbol{M}_{t,i}$ is applied:

$$\widehat{p}_{t,i} = \alpha_{t,i}^2 \frac{n_{t-1,i}}{d_{t-1,i}} + \omega_{t,i}, \quad \boldsymbol{\kappa}_t = \frac{\widehat{\boldsymbol{p}}_t \odot \boldsymbol{k}_t}{r_t + \sum_i \widehat{p}_{t,i} k_{t,i}^2}, \quad \boldsymbol{z}_{t,i} = \boldsymbol{M}_{t,i} \boldsymbol{z}_{t-1,i}. \tag{70}$$

This gives a three-pass implementation: compute one 2×2 summary per chunk and channel, scan the summaries to obtain exclusive chunk carries, and replay the chunks in parallel to emit $\boldsymbol{\kappa}_t$. The ratio n/d is invariant to a common rescaling, so after each multiplication we divide every homogeneous vector or matrix by its largest-magnitude entry. This fp32 renormalization prevents overflow when $\omega_{t,i}$ is very small. Our gain kernel uses $C_g = 256$; the short carry pass is serial over the number of chunks, although the summaries also admit a fully associative prefix scan.

F.2 Chunkwise Memory Scan

With $\boldsymbol{\kappa}_t$ fixed, partition the sequence independently into memory chunks of length C_m. Consider one chunk, suppress its global index, and let $\boldsymbol{S}^{\mathrm{in}}$ be its incoming memory. For local position r, define the coordinatewise cumulative decay

$$\boldsymbol{\gamma}_r = \prod_{j=1}^{r} \boldsymbol{\alpha}_j, \qquad \boldsymbol{\gamma}_0 = \mathbf{1}, \qquad \boldsymbol{S}_r = \mathrm{diag}(\boldsymbol{\gamma}_r) \boldsymbol{R}_r. \tag{71}$$

The normalized state obeys an ordinary two-key delta rule,

$$\boldsymbol{R}_r = \left(\boldsymbol{I} - \overline{\boldsymbol{\kappa}}_r \overline{\boldsymbol{k}}_r^{\intercal}\right) \boldsymbol{R}_{r-1} + \overline{\boldsymbol{\kappa}}_r \boldsymbol{v}_r^{\intercal}, \qquad \overline{\boldsymbol{k}}_r = \boldsymbol{\gamma}_r \odot \boldsymbol{k}_r, \qquad \overline{\boldsymbol{\kappa}}_r = \boldsymbol{\kappa}_r \oslash \boldsymbol{\gamma}_r. \tag{72}$$

Let $\boldsymbol{K}$, $\boldsymbol{K}^{(\kappa)}$, $\boldsymbol{Q}$, $\boldsymbol{V}$, and $\boldsymbol{\Gamma}$ stack the chunk's rows $\boldsymbol{k}_r^{\intercal}$, $\boldsymbol{\kappa}_r^{\intercal}$, $\boldsymbol{q}_r^{\intercal}$, $\boldsymbol{v}_r^{\intercal}$, and $\boldsymbol{\gamma}_r^{\intercal}$, respectively. Let τ denote the query scale, with $\tau = d_k^{-1/2}$ in our implementation and $\tau = 1$ when it is absorbed into $\boldsymbol{q}_r$. Define

$$\overline{\boldsymbol{K}} = \boldsymbol{\Gamma} \odot \boldsymbol{K}, \qquad \overline{\boldsymbol{K}}^{(\kappa)} = \boldsymbol{K}^{(\kappa)} \oslash \boldsymbol{\Gamma}, \qquad \overline{\boldsymbol{Q}} = \tau \boldsymbol{\Gamma} \odot \boldsymbol{Q}. \tag{73}$$

The compact WY factors are

$$\begin{aligned} &\boldsymbol{T} = \mathrm{tril}\left(\overline{\boldsymbol{K}} (\overline{\boldsymbol{K}}^{(\kappa)})^{\intercal}, -1\right), \quad \boldsymbol{A} = (\boldsymbol{I}_{C_m} + \boldsymbol{T})^{-1}, \\ &\boldsymbol{W} = \boldsymbol{A}\overline{\boldsymbol{K}}, \qquad\qquad \boldsymbol{U} = \boldsymbol{A}\boldsymbol{V}, \qquad\qquad \boldsymbol{R} = \boldsymbol{U} - \boldsymbol{W}\boldsymbol{S}^{\mathrm{in}}. \end{aligned} \tag{74}$$

Here $\boldsymbol{T}$ is strictly lower triangular. The implementation computes and stores $\boldsymbol{A}$ with a tiled unit-lower-triangular solve, then forms $\boldsymbol{W}$ and $\boldsymbol{U}$. Define the final-decay-adjusted write matrix and the inclusive causal query–write matrix by

$$\left[\boldsymbol{K}_{\mathrm{tail}}^{(\kappa)}\right]_{r,:} = \left[(\boldsymbol{\gamma}_{C_m} \oslash \boldsymbol{\gamma}_r) \odot \boldsymbol{\kappa}_r\right]^{\intercal}, \quad \boldsymbol{A}_{q\kappa} = \mathrm{tril}\left(\overline{\boldsymbol{Q}} (\overline{\boldsymbol{K}}^{(\kappa)})^{\intercal}\right). \tag{75}$$

The outputs for the whole chunk and its outgoing memory are then

$$\boldsymbol{O} = \overline{\boldsymbol{Q}}\boldsymbol{S}^{\mathrm{in}} + \boldsymbol{A}_{q\kappa}\boldsymbol{R}, \quad \boldsymbol{S}^{\mathrm{out}} = \mathrm{diag}(\boldsymbol{\gamma}_{C_m})\boldsymbol{S}^{\mathrm{in}} + \left(\boldsymbol{K}_{\mathrm{tail}}^{(\kappa)}\right)^{\intercal}\boldsymbol{R}. \tag{76}$$

The diagonal of $\boldsymbol{A}_{q\kappa}$ is included because $\boldsymbol{o}_t$ reads the state after the current write. These equations are exact for the Diagonal KDN recurrence. They reduce to KDA when $\boldsymbol{\kappa}_t = \beta_t \boldsymbol{k}_t$ and otherwise use the same two-key WY structure as GDN-2, with $\boldsymbol{\kappa}_t$ as the write direction and $\boldsymbol{k}_t$ as the erase direction. All chunk-local factors are computed in parallel; only $\boldsymbol{S}^{\mathrm{in}}$ is carried across memory chunks. The implementation forms decay ratios from cumulative log-decay differences for numerical stability and uses $C_m \in \{32, 64\}$, independently of the gain-chunk length.