Exact Flow Linear Attention: Exact Solution from Continuous-Time Dynamics

Jingdi Lei[1*], **Di Zhang**[2*], **Soujanya Poria**[1*]
[1]Nanyang Technological University, [2]Fudan University
jingdi001@e.ntu.edu.sg, di.zhang@ustc.edu,
soujanya.poria@ntu.edu.sg

Abstract

In this paper, we introduce **Exact Flow Linear Attention (EFLA)**, an exact-flow formulation of delta-rule linear attention. We show that the delta-rule update can be interpreted as an explicit Euler discretization of an underlying continuous-time system. EFLA replaces this first-order update with the exact closed-form flow. By exploiting the *rank-1* structure of the dynamics matrix, both the matrix exponential and the input integral collapse to a simple update that preserves delta-rule linear attention's algebraic structure, parameter count, linear-time complexity, and chunk-wise parallelism. This attention mechanism removes the Euler discretization error of the delta-rule dynamics without introducing additional parameters. Experiments on robustness tests, language modeling benchmarks, and the MAD synthetic benchmark show that EFLA improves stability under corrupted and high-energy inputs, reduces perplexity, and achieves stronger downstream performance compared to SSM and Euler-style baselines. These results establish exact-flow integration as a principled and scalable update mechanism for delta-rule linear attention. Code is available at: https://github.com/declare-lab/EFLA.

1 Introduction

As large language models (LLMs) evolve into increasingly capable agents [Yao et al., 2022, Team et al., 2025b, Google, 2025, OpenAI, 2025], inference efficiency has become a critical bottleneck [Dao et al., 2022, Kwon et al., 2023, Kim et al., 2024]. This challenge is especially pronounced in long-context processing and reinforcement learning (RL) environments [Guo et al., 2025, Lai et al., 2025], where models are required to process extended reasoning trajectories, maintain long-range dependencies, and engage in complex tool-use interactions [Lightman et al., 2023, Yao et al., 2023]. In these scenarios, the quadratic time complexity of standard softmax attention [Vaswani et al., 2017] introduces substantial computational overhead, constraining model throughput, scalability to long contexts, and real-time interactivity. [Liu et al., 2023, Jiang et al., 2024, Katharopoulos et al., 2020].

This has motivated a broad line of work on linear-time sequence models, including linear attention and state space models (SSMs). Among them, delta-rule linear attention [Schlag et al., 2021, Yang et al., 2024b] provides an appealing recurrent formulation by maintaining a matrix-valued associative memory. Compared with purely additive linear attention [Katharopoulos et al., 2020], the delta rule introduces an input-dependent correction mechanism: the memory is updated by partially erasing the component associated with the current key and injecting a new key–value association. This update admits efficient recurrent and chunkwise-parallel implementations [Yang et al., 2024b]. Despite its empirical effectiveness, this update is usually motivated as a discrete online learning rule, leaving its underlying continuous-time structure less explored.

In this work, we revisit delta-rule linear attention from the perspective of numerical integration. Under a zero-order-hold (ZOH) assumption, where the current key and value are treated as fixed

*Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

within each token interval, the delta-rule update can be interpreted as an explicit Euler discretization of a continuous-time system. This interpretation exposes a principled source of approximation error: the Euler update is only first-order accurate. Although simple and efficient, Euler discretization can introduce accumulated discretization error and may become inaccurate when the effective dynamics are stiff, for example under large key norms or large update scales. Existing methods often mitigate such issues by introducing gating mechanisms including decay factors, gating functions [Dao and Gu, 2024, Ma et al., 2022, Sun et al., 2023], or adaptive forgetting coefficients [Yang et al., 2023, 2024a, Team et al., 2025a]. This motivates a natural question: instead of heuristically mitigating the instability of Euler-style updates, can we derive an exact update directly from the underlying continuous-time dynamics of delta-rule linear attention?

We answer this question by proposing **Exact Flow Linear Attention (EFLA)**, a drop-in replacement for the delta-rule update that solves the corresponding ZOH continuous-time dynamics in closed form. From the first principle, we eliminate the discretization errors by solving the underlying ODE exactly. EFLA can be mathematically interpreted as the general solution of the first-order ODE, yielding a continuous-time, exact flow update of linear attention. The key observation is that the transition matrix is rank-1 in delta-rule update. Therefore, although a generic matrix exponential would be computationally expensive, both the exponential transition and the input integral collapse into a simple closed form. EFLA preserves the algebraic structure and computational order of delta-rule linear attention while removing the Euler discretization error of the underlying ZOH dynamics.

Since EFLA retains the same rank-1 update form as vanilla delta-rule linear attention, it can be integrated with existing hardware-efficient WY/UT-based chunkwise parallelization schemes. This allows EFLA to preserve linear-time recurrent inference and efficient parallel training. Empirically, we evaluate EFLA on robustness tests, language modeling benchmarks, and the MAD synthetic benchmark. Across these evaluations, EFLA consistently improves over previous Euler discretization baselines in both robustness, convergence, and downstream performance while maintaining comparable training throughput and introducing no additional parameters.

Our contributions are summarized as follows:

- We reinterpret delta-rule linear attention as an explicit Euler discretization of an underlying ZOH continuous-time ODE, and propose **Exact Flow Linear Attention (EFLA)**, which solves this ODE in closed form.
- We show that the rank-1 structure of delta-rule learning makes the exact matrix exponential and input integral analytically tractable, yielding an exact-flow update with the same computational order and algebraic form as delta-rule linear attention.
- We validate EFLA across robustness tests, language modeling benchmarks, and the MAD synthetic benchmark, showing consistent improvements over previous Euler-style baselines while maintaining comparable throughput.

2 Preliminary

Scaled Dot-Product Attention. Given queries $\mathbf{Q} \in \mathbb{R}^{n\times d}$, keys $\mathbf{K} \in \mathbb{R}^{n\times d}$, and values $\mathbf{V} \in \mathbb{R}^{n\times d}$, scaled dot-product attention [Vaswani et al., 2017] is defined as:

$$\mathrm{Attn}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}} + \mathbf{M}\right)\mathbf{V}, \tag{1}$$

where $\mathbf{M} \in \mathbb{R}^{n\times n}$ is the additive causal mask, with zeros on and below the diagonal and $-\infty$ above.

Linear Attention as Associative Memory. Linear attention [Katharopoulos et al., 2020] maintains a matrix-valued recurrent state that accumulates key–value associations:

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{k}_t\mathbf{v}_t^\top, \qquad \mathbf{o}_t = \mathbf{S}_t^\top\mathbf{q}_t. \tag{2}$$

From the fast-weight perspective [Schlag et al., 2021], $\mathbf{S}_t$ serves as an associative memory that stores transient mappings from keys to values. This additive update can be viewed as gradient descent on the unbounded correlation objective:

$$\mathcal{L}_t(\mathbf{S}) = -\langle \mathbf{S}^\top\mathbf{k}_t, \mathbf{v}_t\rangle. \tag{3}$$

Since this objective only reinforces new key–value pairs and does not provide an explicit mechanism for erasing old associations, the accumulated memory may grow without bound and suffer from interference over long contexts.

Delta-rule Learning. Delta-rule linear attention addresses this issue by formulating the memory state as the parameter of an online reconstruction problem:

$$\mathcal{L}_t(\mathbf{S}) = \frac{1}{2}\left\|\mathbf{S}^\top \mathbf{k}_t - \mathbf{v}_t\right\|^2. \tag{4}$$

Taking one gradient step with learning rate β_t gives:

$$\mathbf{S}_t = \mathbf{S}_{t-1} - \beta_t \nabla_{\mathbf{S}} \mathcal{L}_t(\mathbf{S}_{t-1}) = \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top. \tag{5}$$

This update corrects the associative memory toward the mapping $\mathbf{k}_t \mapsto \mathbf{v}_t$. Its rank-1 correction structure enables WY-style products of rank-1 transformations, facilitating hardware-efficient chunkwise parallelization [Bischof and Van Loan, 1987, Yang et al., 2024b].

Euler Methods in ODEs. Given a first-order ordinary differential equation (ODE):

$$\frac{d\mathbf{S}(\tau)}{d\tau} = f(\tau, \mathbf{S}(\tau)), \tag{6}$$

numerical integration methods approximate the solution at discrete time points. The explicit Euler method [Euler, 1792] updates the state by using the derivative at the current position:

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \beta_t f(t-1, \mathbf{S}_{t-1}), \tag{7}$$

where β_t is the step size. Euler discretization is computationally simple but only first-order accurate, with local truncation error $\mathcal{O}(\beta_t^2)$. As shown below, Euler discretization delta-rule updates can be interpreted as such an explicit Euler discretization of an underlying continuous-time memory dynamics.

3 Method

3.1 Exact Flow of Delta-rule Linear Attention

We begin by revisiting DeltaNet [Schlag et al., 2021], which formulates linear attention as online gradient descent on a reconstruction objective:

$$\mathcal{L}_t(\mathbf{S}) = \frac{1}{2}\|\mathbf{S}^\top \mathbf{k}_t - \mathbf{v}_t\|^2. \tag{8}$$

Applying a single gradient descent step with learning rate β_t yields:

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \beta_t \left(-\mathbf{k}_t \mathbf{k}_t^\top \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top\right). \tag{9}$$

To formalize the underlying dynamics, we define the dynamics matrix $\mathbf{A}_t = \mathbf{k}_t \mathbf{k}_t^\top$ and the input forcing term $\mathbf{b}_t = \mathbf{k}_t \mathbf{v}_t^\top$. Since the input sequence is observed only at discrete time steps, we adopt the Zero-Order Hold (ZOH) [Iserles, 2009] assumption to construct a continuous-time interpolation. Under this assumption, $\mathbf{A}_t$ and $\mathbf{b}_t$ are treated as piecewise constant within each update interval. The system evolves according to a first-order ODE:

$$\frac{d\mathbf{S}(t)}{dt} = -\mathbf{A}_t \mathbf{S}(t) + \mathbf{b}_t. \tag{10}$$

It reveals that the standard Delta rule update is only a first-order numerical approximation to the underlying continuous dynamics. This observation motivates a natural question: can we avoid the Euler discretization error by solving the ODE exactly? Eq. (10) admits a closed-form matrix-exponential solution, yielding[2]:

$$\mathbf{S}_t = e^{-\beta_t \mathbf{A}_t} \mathbf{S}_{t-1} + \int_0^{\beta_t} e^{-(\beta_t - \tau)\mathbf{A}_t} \mathbf{b}_t \, d\tau. \tag{11}$$

[2]The detailed derivation of this closed-form expression is provided in Appendix E.

3.2 Efficient Exact Flow via the *Rank-1* Structure

While the infinite-order solution eliminates discretization error, naively evaluating the matrix exponential $e^{-\beta_t \mathbf{A}_t}$ for a general matrix typically requires $\mathcal{O}(d^3)$ complexity [Gu et al., 2020]. We bypass this computational bottleneck by leveraging the *rank-1* structure of the dynamics matrix $\mathbf{A}_t = \mathbf{k}_t \mathbf{k}_t^\top$, which allows the exponential to be computed in linear time.

We observe that $\mathbf{A}_t$ satisfies the idempotence-like property $\mathbf{A}_t^n = \lambda_t^{n-1} \mathbf{A}_t$ for $n \geq 1$, where $\lambda_t = \mathbf{k}_t^\top \mathbf{k}_t$ (see Appendix D for the proof). This property allows us to collapse the Taylor series of the matrix exponential into a computable closed form:

$$e^{-\beta_t \mathbf{A}_t} = \mathbf{I} + \sum_{n=1}^{\infty} \frac{(-\beta_t)^n}{n!} \mathbf{A}_t^n = \mathbf{I} - \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{A}_t. \tag{12}$$

Substituting this transition operator into the integral term $\int_0^{\beta_t} e^{-(\beta_t - \tau)\mathbf{A}_t} \mathbf{b}_t \, d\tau$ yields the exact input injection:

$$\begin{aligned} \mathbf{I}_t &= \int_0^{\beta_t} \left(\mathbf{I} - \frac{1 - e^{-\lambda_t(\beta_t - \tau)}}{\lambda_t} \mathbf{A}_t \right) \mathbf{b}_t \, d\tau \\ &= \beta_t \mathbf{b}_t - \frac{\mathbf{A}_t \mathbf{b}_t}{\lambda_t} \left(\beta_t - \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \right). \end{aligned} \tag{13}$$

Crucially, since $\mathbf{b}_t = \mathbf{k}_t \mathbf{v}_t^\top$ and $\mathbf{A}_t = \mathbf{k}_t \mathbf{k}_t^\top$, we have $\mathbf{A}_t \mathbf{b}_t = \lambda_t \mathbf{b}_t$. This algebraic relationship allows for significant simplification of the integral term:

$$\mathbf{I}_t = \beta_t \mathbf{b}_t - \beta_t \mathbf{b}_t + \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{b}_t = \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{b}_t. \tag{14}$$

Combining these results, the final Exact Flow Linear Attention update rule is given by:

$$\mathbf{S}_t = \left(\mathbf{I} - \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{k}_t^\top \right) \mathbf{S}_{t-1} + \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{v}_t^\top. \tag{15}$$

Thus, EFLA has the same rank-1 algebraic form as the delta-rule update, with α_t replacing β_t. It removes the Euler discretization error of the ZOH dynamics while preserving linear complexity in the sequence length, namely $\mathcal{O}(Ld^2)$ for a length-L sequence with a $d \times d$ memory state.

3.3 Chunkwise Parallelism Form

We observe that the EFLA update rule shares an identical algebraic structure with DeltaNet. Given this structural equivalence, we can seamlessly adapt the hardware-efficient parallelization strategies originally developed for DeltaNet [Yang et al., 2024b]. In this section, we derive the chunkwise parallel formulation specifically for EFLA.

To derive the chunkwise parallel form, we first unroll the recurrence relation. Denoting $\frac{1-e^{-\beta_t \lambda_t}}{\lambda_t} = \alpha_t$, the state update becomes:

$$\mathbf{S}_t = (\mathbf{I} - \alpha_t \mathbf{k}_t \mathbf{k}_t^\top) \mathbf{S}_{t-1} + \alpha_t \mathbf{k}_t \mathbf{v}_t^\top = \sum_{i=1}^{t} \left(\prod_{j=i+1}^{t} (\mathbf{I} - \alpha_j \mathbf{k}_j \mathbf{k}_j^\top) \right) \alpha_i (\mathbf{k}_i \mathbf{v}_i^\top). \tag{16}$$

Then we can define the following variables:

$$\mathbf{P}_i^j = \prod_{t=i}^{j} (\mathbf{I} - \alpha_t \mathbf{k}_t \mathbf{k}_t^\top), \qquad \mathbf{H}_i^j = \sum_{t=i}^{j} \mathbf{P}_{t+1}^j \alpha_t \mathbf{k}_t \mathbf{v}_t^\top \tag{17}$$

where $\mathbf{P}_i^j = \mathbf{I}$ when $i > j$. $\mathbf{P}_i^j$ can be considered as decay factor applied to $\mathbf{S}_i$ to obtain $\mathbf{S}_j$ and $\mathbf{H}_i^j$ is an accumulation term to $\mathbf{S}_j$ from token i. The Chunkwise can be written as follows:

$$\mathbf{S}_{[t]}^r = \mathbf{P}_{[t]}^r \mathbf{S}_{[t]}^0 + \mathbf{H}_{[t]}^r \tag{18}$$

where we define the chunkwise variables $\mathbf{S}_{[t]}^r = \mathbf{S}_{tC+r}$, $\mathbf{P}_{[t]}^r = \mathbf{P}_{tC+1}^{tC+r}$, $\mathbf{H}_{[t]}^r = \mathbf{H}_{tC+1}^{tC+r}$. Here we have $\frac{L}{C}$ chunks of size C. We can use induction to derive the WY representations of $\mathbf{P}_{[t]}^r$ and $\mathbf{H}_{[t]}^r$:

$$\mathbf{P}_{[t]}^r = \mathbf{I} - \sum_{i=1}^{r} \mathbf{k}_{[t]}^i \mathbf{w}_{[t]}^{i\top}, \qquad \mathbf{H}_{[t]}^r = \sum_{i=1}^{r} \mathbf{k}_{[t]}^i \mathbf{u}_{[t]}^{i\top} \tag{19}$$

$$\mathbf{w}_{[t]}^{r\top} = \alpha_{[t]}^r \left(\mathbf{k}_{[t]}^{r\top} - \sum_{i=1}^{r-1} (\mathbf{k}_{[t]}^{r\top} \mathbf{k}_{[t]}^i) \mathbf{w}_{[t]}^{i\top} \right), \quad \mathbf{u}_{[t]}^{r\top} = \alpha_{[t]}^r \left(\mathbf{v}_{[t]}^{r\top} - \sum_{i=1}^{r-1} (\mathbf{k}_{[t]}^{r\top} \mathbf{k}_{[t]}^i) \mathbf{u}_{[t]}^{i\top} \right). \tag{20}$$

subsequently, we can obtain the chunk-level recurrence for states and outputs:

$$\mathbf{S}_{[t]}^r = \mathbf{S}_{[t]}^0 - \left(\sum_{i=1}^{r} \mathbf{k}_{[t]}^i \mathbf{w}_{[t]}^{i\top} \right) \mathbf{S}_{[t]}^0 + \sum_{i=1}^{r} \mathbf{k}_{[t]}^i \mathbf{u}_{[t]}^{i\top} = \mathbf{S}_{[t]}^0 + \sum_{i=1}^{r} \mathbf{k}_{[t]}^i \big(\mathbf{u}_{[t]}^{i\top} - \mathbf{w}_{[t]}^{i\top} \mathbf{S}_{[t]}^0 \big). \tag{21}$$

$$\mathbf{o}_{[t]}^r = \mathbf{S}_{[t]}^{r\top} \mathbf{q}_{[t]}^r = \mathbf{S}_{[t]}^{0\top} \mathbf{q}_{[t]}^r + \sum_{i=1}^{r} (\mathbf{u}_{[t]}^i - \mathbf{S}_{[t]}^{0\top} \mathbf{w}_{[t]}^i)(\mathbf{k}_{[t]}^{i\top} \mathbf{q}_{[t]}^i) \tag{22}$$

letting $\mathbf{S}_{[t]} = \mathbf{S}_{[t]}^0$, the above can be simplified to matrix notations:

$$\mathbf{S}_{[t+1]} = \mathbf{S}_{[t]} + \mathbf{K}_{[t]}^\top \left(\mathbf{U}_{[t]} - \mathbf{W}_{[t]} \mathbf{S}_{[t]} \right) \tag{23}$$

$$\mathbf{O}_{[t]} = \mathbf{Q}_{[t]} \mathbf{S}_{[t]} + \left(\mathbf{Q}_{[t]} \mathbf{K}_{[t]}^\top \odot \mathbf{M} \right) \left(\mathbf{U}_{[t]} - \mathbf{W}_{[t]} \mathbf{S}_{[t]} \right), \tag{24}$$

where $\square_{[t]} = \square_{[t]}^{1:C} \in \mathbb{R}^{C \times d}$ for $\square \in \{\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{O}, \mathbf{U}, \mathbf{W}\}$ defines the chunkwise matrices that are formed from stacking the $\mathbf{q}_t, \mathbf{k}_t, \mathbf{v}_t, \mathbf{o}_t, \mathbf{u}_t, \mathbf{w}_t$ vectors and $\mathbf{M}$ is the lower triangular causal mask.

Finally, we can apply the UT transform [Joffrain et al., 2006] to simplify the recurrence calculations of $\mathbf{u}_{[t]}^r$ and $\mathbf{w}_{[t]}^r$.

$$\mathbf{T}_{[i]} = \left(\mathbf{I} + \mathrm{StrictTril}(\mathrm{diag}(\alpha_t) \mathbf{K}_{[i]} \mathbf{K}_{[i]}^\top) \right)^{-1} \mathrm{diag}(\alpha_t) \tag{25}$$

$$\mathbf{W}_{[t]} = \mathbf{T}_{[t]} \mathbf{K}_{[t]}, \qquad \mathbf{U}_{[t]} = \mathbf{T}_{[t]} \mathbf{V}_{[t]} \tag{26}$$

3.4 Discussion

Directional decay of memory. The EFLA formulation reveals the role of the key norm in controlling memory retention. Since $\mathbf{A}_t = \mathbf{k}_t \mathbf{k}_t^\top$ is symmetric positive semi-definite and rank-1, it has one non-zero eigenvalue $\lambda_t = \|\mathbf{k}_t\|^2$ when $\mathbf{k}_t \neq 0$, while all remaining eigenvalues are zero. Consequently,

$$e^{-\beta_t \mathbf{A}_t} = \mathbf{I} - \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{k}_t^\top. \tag{27}$$

This transition contracts the component of the memory aligned with $\mathbf{k}_t$ by the factor $e^{-\beta_t \lambda_t}$, while leaving the orthogonal subspace unchanged. For $\beta_t \geq 0$, the eigenvalues of the homogeneous transition are $e^{-\beta_t \lambda_t}$ and 1, so the homogeneous flow is non-expansive.

Connection to the delta rule. The effective update coefficient satisfies:

$$\alpha_t = \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t} = \beta_t - \frac{1}{2} \beta_t^2 \lambda_t + \mathcal{O}(\beta_t^3 \lambda_t^2). \tag{28}$$

Therefore, when the effective update scale $\beta_t\lambda_t$ is small, EFLA recovers the delta-rule update up to higher-order terms. For larger $\beta_t\lambda_t$, EFLA uses the bounded exponential flow, whereas the Euler-style delta rule keeps only the first-order approximation.

Role of key norm and stable dynamics. EFLA naturally exposes the role of the key norm through the exact-flow coefficient:

$$\alpha_t = \frac{1 - e^{-\beta_t \lambda_t}}{\lambda_t}, \qquad \lambda_t = \|\mathbf{k}_t\|^2. \tag{29}$$

The denominator λ_t is not an additional normalization trick, but arises directly from solving the rank-1 ODE in closed form. This shows that the strength of the exact-flow update is intrinsically tied to the magnitude of the current key.

This dependence also affects the write term $\alpha_t \mathbf{k}_t \mathbf{v}_t^\top$. When keys are L2-normalized, the model mainly uses the direction of $\mathbf{k}_t$. In contrast, EFLA preserves the magnitude of $\mathbf{k}_t$ in the key–value injection term, allowing the current token to modulate the memory write not only through the learned gate β_t, but also through the signal strength encoded in the key norm.

Finally, the continuous-time dynamics provide a stability interpretation. The homogeneous part of the ODE is governed by:

$$\frac{d\mathbf{S}(\tau)}{d\tau} = -\mathbf{k}_t \mathbf{k}_t^\top \mathbf{S}(\tau), \tag{30}$$

where $-\mathbf{k}_t \mathbf{k}_t^\top$ is negative semi-definite. Therefore, the exact transition contracts the memory component aligned with $\mathbf{k}_t$ by the factor $e^{-\beta_t \lambda_t}$, while leaving the orthogonal subspace unchanged. This contractive flow suppress perturbations along the current key direction and provides a natural explanation for the improved stability and noise robustness observed in our experiments in Section 4.3.

Gated Variant of EFLA. EFLA is also compatible with gated recurrences by applying the exact-flow construction to a gated update mechanism. The derivation of this variant, which we term Gated EFLA, is provided in Appendix F.

4 Experiments

4.1 Experimental Setup

Baselines. We compare EFLA and its gated variants with SSM methods including Mamba-2 [Dao and Gu, 2024], and Euler-style methods including DeltaNet [Yang et al., 2024b], and Gated DeltaNet [Yang et al., 2024a]. EFLA is implemented as a drop-in replacement for the delta-rule state update in Euler-style methods, using the exact-flow coefficient derived in Section 3.

Table 1: Main language modeling results. The 340M models are trained for 8B tokens, whereas the 1.3B models are trained for 50B tokens on the same subset of the SlimPajama dataset [Soboleva et al., 2023] with the Mistral [Jiang et al., 2023] tokenizer. **Perplexity:** Lower (↓) is better. **Accuracy:** Higher (↑) is better. Best results are bolded.

	Perplexity (↓)		**Accuracy** (↑)									
Model	**Wiki.** ppl	**LMB.** ppl	**LMB.** acc	**PIQA** acc	**Hella.** acc_n	**Wino.** acc	**ARC-e** acc	**ARC-c** acc_n	**BoolQ** acc	**OBQA** acc_n	**SciQ** acc	**Avg.**
340M Parameters												
Vanilla												
Mamba-2	36.90	105.69	20.7	**61.9**	30.7	50.2	40.4	22.1	53.0	27.0	71.9	42.0
DeltaNet	38.09	96.26	22.5	60.7	30.1	**51.9**	41.7	21.6	60.1	29.0	70.4	42.2
EFLA	35.26	**79.97**	**23.9**	61.0	30.9	51.1	**42.5**	**23.8**	59.7	**30.8**	72.9	**44.1**
Gated Variants												
Gated DeltaNet	34.93	72.46	25.5	**62.7**	30.8	49.3	41.8	21.7	61.7	27.4	74.2	43.9
Gated EFLA	**34.28**	**69.37**	**26.4**	61.8	**31.1**	**52.1**	**42.0**	**22.8**	**62.1**	**29.4**	**75.3**	**44.8**
1.3B Parameters												
DeltaNet	18.38	17.29	41.8	**69.2**	**44.5**	49.3	52.5	**26.4**	58.1	29.8	82.6	50.5
EFLA	**18.30**	**16.54**	**43.2**	68.9	**44.5**	**52.1**	**54.4**	**26.4**	**60.4**	**31.6**	**84.2**	**51.8**

Training setup. The 340M models are trained for 8B tokens, and the 1.3B models are trained for 50B tokens on the same subset of the SlimPajama dataset [Soboleva et al., 2023] with the Mistral tokenizer [Jiang et al., 2023]. Detailed hyperparameter settings are provided in Appendix C.

Evaluation. We evaluate language modeling perplexity on Wikitext [Merity et al., 2016] and LAMBADA [Paperno et al., 2016], together with a suite of zero-shot commonsense reasoning tasks, including PiQA [Bisk et al., 2020], HellaSwag [Zellers et al., 2019], WinoGrande [Sakaguchi et al., 2021], ARC-easy (ARC-e), ARC-challenge (ARC-c) [Clark et al., 2018], BoolQ [Clark et al., 2019], OpenBookQA (OBQA) [Mihaylov et al., 2018], and SciQ [Johannes Welbl, 2017]. We report perplexity for Wikitext and LAMBADA, and accuracy for the downstream reasoning tasks.

4.2 Main Language Modeling Results

The main language modeling results are shown in Table 1. For 340M-parameter models trained with the same 8B-token budget, EFLA consistently improves over the Euler-style baseline on most metrics. On the perplexity metrics, EFLA reduces Wikitext perplexity from 38.09 to 35.26 and LAMBADA perplexity from 96.26 to 79.97, indicating better language modeling quality under the same training budget. On downstream accuracy, EFLA improves the average score from 42.2% to 44.1%, suggesting that the exact-flow update improves the general effectiveness of the Euler-style recurrence. The comparison with Mamba-2 further shows that EFLA is competitive among efficient recurrent architectures at this scale.

The same trend holds for the gated variants. Replacing the Gated DeltaNet update with the Gated EFLA update reduces Wikitext perplexity from 34.93 to 34.28 and LAMBADA perplexity from 72.46 to 69.37. It also improves the average downstream accuracy from 43.9% to 44.8%. This result is important because it shows that the exact-flow update is not limited to the vanilla formulation, but can also be incorporated into the gated variant and still produce consistent improvements.

At the larger 1.3B scale, the improvements persist under a longer 50B-token training budget. Compared with DeltaNet, EFLA reduces Wikitext perplexity from 18.38 to 18.30 and LAMBADA perplexity from 17.29 to 16.54. It also improves the average downstream accuracy from 50.5% to 51.8%. Therefore, the improvement is not a small-scale artifact, the exact-flow update continues to provide benefits when both model size and training tokens are increased.

4.3 Robustness under Corrupted and High-energy Inputs

Robustness evaluation. As discussed in Section 3, Euler discretization delta-rule updates can be interpreted as explicit Euler discretizations of the corresponding ZOH continuous-time dynamics. While computationally efficient, this first-order discretization may become inaccurate under large effective update scales or corrupted inputs. To evaluate this behavior, we conduct three robustness tests on pixel-level Sequential MNIST [LeCun et al., 2010].

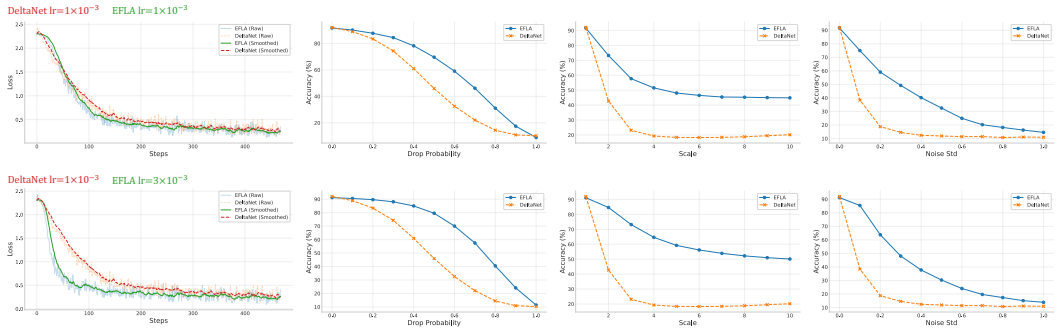
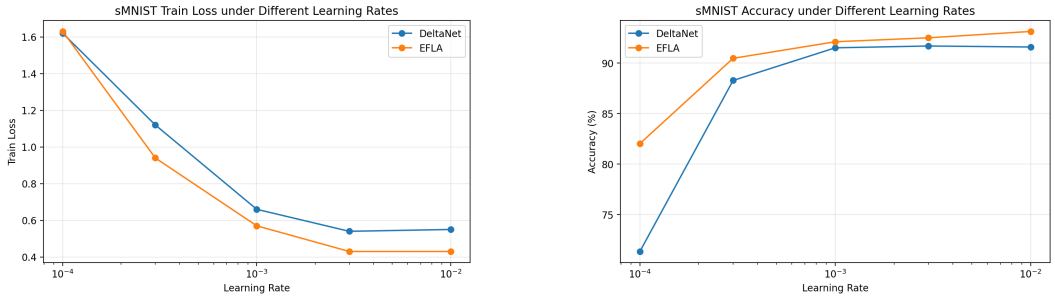


Figure 1: EFLA outperforms DeltaNet in both convergence speed and robustness on sMNIST. The plots illustrate training dynamics and robustness against dropout, intensity scaling, and additive Gaussian noise. EFLA maintains higher accuracy as perturbation intensity increases, especially under the larger learning-rate setting 3×10^{-3}.

Perturbation settings. We consider three types of input perturbations. First, **image pixel dropout** applies Bernoulli dropout to input tokens with probability p, simulating information loss in corrupted



(a) Training loss.

(b) Accuracy.

Figure 2: sMNIST results under different learning rates. EFLA maintains lower training loss and higher accuracy across the tested learning-rate range, indicating stronger robustness to learning-rate selection.

inputs. Second, **OOD intensity scaling** amplifies input signals by a fixed factor, testing stability under high-energy inputs. Third, **additive Gaussian noise** injects random noise with varying standard deviations, evaluating robustness to signal corruption.

Experimental setup. We flatten each 28×28 image into a sequence of length $L = 784$ and use a linear attention classifier with hidden dimension $d = 64$. We compare EFLA with the DeltaNet baseline. Both models are trained with AdamW using a batch size of 128.

Results. Figure 1 shows that DeltaNet is more sensitive to input scaling, with accuracy dropping rapidly as the scale factor increases. In contrast, EFLA maintains higher accuracy under large input scales, consistent with its exact-flow transition, whose homogeneous component remains non-expansive for non-negative decay. EFLA also outperforms DeltaNet under additive noise and pixel dropout, with a slower degradation rate under stronger perturbations. These results support the view that removing the Euler discretization error of the ZOH delta-rule state transition improves robustness under corrupted or high-energy inputs.

4.4 Robustness to Learning Rate Selection

To further examine the optimization behavior of EFLA, we evaluate its sensitivity to the learning rate on the Sequential MNIST. We compare EFLA with DeltaNet under the same training setup while varying the learning rate over $\{1 \times 10^{-4}, 3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}, 1 \times 10^{-2}\}$.

As the results shown in Figure 2, EFLA remains competitive across the entire learning-rate range. At very small learning rates, such as 1×10^{-4}, both methods have relatively high training loss, but EFLA already achieves substantially higher accuracy than DeltaNet. As the learning rate increases, EFLA consistently obtains lower training loss and higher accuracy. Notably, when the learning rate is increased to 1×10^{-2}, DeltaNet shows no further accuracy improvement, whereas EFLA continues to achieve the best accuracy among all tested settings. These results suggest that EFLA is less sensitive to learning-rate selection than the Euler-style DeltaNet update. This behavior is consistent with our theoretical interpretation: by reducing the discretization error introduced at each recurrent state transition, EFLA provides a more faithful state update, which can improve optimization behavior across different learning-rate regimes.

4.5 Results on the MAD Synthetic Benchmark

We further evaluate EFLA on the Mechanistic Architecture Design (MAD) benchmark [Poli et al., 2024], a suite of synthetic token-manipulation tasks designed to isolate architecture-level capabilities. As shown in Figure 3, EFLA improves upon DeltaNet across all six tasks, increasing the average score from 65.7% to 66.4%. The gains are most visible on Memorize and Compress, suggesting that the exact-flow state transition improves the model's ability to maintain and update token-level memory.

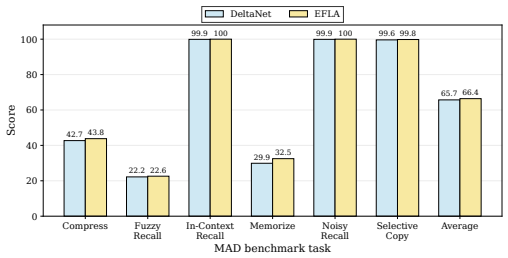


Figure 3: Results on the synthetic MAD benchmark.

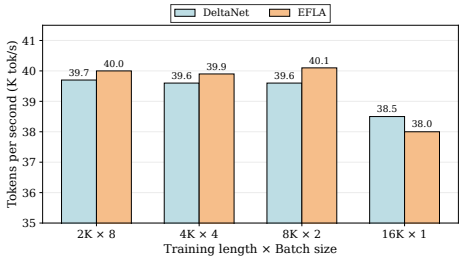


Figure 4: Training throughput of DeltaNet and EFLA.

4.6 Training Efficiency

We compare the training efficiency of DeltaNet and EFLA under matched sequence-length and batch-size configurations. As shown in Figure 4, EFLA maintains nearly identical throughput to DeltaNet across all settings. These results confirm that the exact-flow update preserves the hardware-efficient structure of the delta-rule recurrence and does not introduce meaningful training overhead.

5 Related Work

Kernel Linear Attention and Delta-Rule Updates. Linear-time attention methods reduce the quadratic cost of softmax attention by replacing or reformulating the softmax operation. Kernel-based methods such as Linear Transformers [Katharopoulos et al., 2020] and Performer [Choromanski et al., 2020] express causal attention through recurrently accumulated key–value statistics. From the fast-weight perspective, Schlag et al. [2021] interpret this matrix-valued state as an associative memory.

Delta-rule linear attention [Schlag et al., 2021, Yang et al., 2024b] improves this memory update by solving an online reconstruction problem. A gradient step with step size β_t gives

$$\mathbf{S}_t = (\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)\mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top . \tag{31}$$

This update introduces input-dependent correction and forgetting. Later variants such as Gated DeltaNet [Yang et al., 2024a] and Kimi Delta Attention (KDA) [Team et al., 2025a] further strengthen Euler discretization recurrences through gating, adaptive update coefficients, and richer value mixing. In contrast, our work revisits the same delta-rule dynamics from a numerical-integration perspective: under a zero-order-hold (ZOH) formulation, the delta-rule update is an explicit Euler discretization. EFLA replaces this Euler transition with the exact ZOH flow induced by the rank-1 matrix $\mathbf{A}_t = \mathbf{k}_t \mathbf{k}_t^\top$, while preserving the computational structure.

State Space Models and Continuous-Time Discretization. State Space Models (SSMs) provide another route to efficient long-sequence modeling by deriving discrete-time sequence operators from continuous-time dynamical systems. The S4 family [Gu et al., 2022a,b, 2020] uses structured state matrices and discretization schemes such as the bilinear transform and ZOH. Mamba [Gu and Dao, 2024] introduces input-dependent selectivity with a hardware-efficient scan, and Mamba-2 [Dao and Gu, 2024] connects selective SSMs with gated linear attention. EMA-based models [Fu et al., 2022, Ma et al., 2022, Sun et al., 2023] can also be viewed as scalar or diagonal state-space recurrences.

EFLA is related to SSMs in that it also starts from a continuous-time ODE and derives a discrete-time update. The key difference is the transition structure: SSMs typically rely on scalar, diagonal, or specially structured matrices, whereas delta-rule linear attention induces a token-dependent rank-1 transition matrix. This rank-1 structure makes both the matrix exponential and input integral analytically tractable, yielding an exact token-wise ZOH flow that retains the algebraic form needed for efficient WY/UT-style chunkwise parallelization.

6 Conclusion

We presented **Exact Flow Linear Attention (EFLA)**, an exact-flow formulation of delta-rule linear attention. By interpreting the Euler-style Delta-rule state update as an explicit Euler discretization of

a zero-order-hold (ZOH) continuous-time system, we identified a principled source of approximation error in the standard delta-rule recurrence. Instead of heuristically modifying this Euler-style update, EFLA derives the closed-form flow of the underlying ZOH dynamics. By exploiting the token-wise rank-1 transition matrix $\mathbf{A}_t = \mathbf{k}_t\mathbf{k}_t^\top$, the resulting update removes the Euler discretization error while preserving the algebraic structure, parameter count, and linear-time complexity of DeltaNet. Empirically, we showed that this exact-flow replacement leads to consistent gains across robustness tests, language modeling benchmarks, and the MAD synthetic benchmark. EFLA improves robustness under corrupted and high-energy inputs, achieves lower perplexity, and obtains stronger downstream performance than DeltaNet without introducing additional parameters. Moreover, its compatibility with WY/UT-style chunkwise parallelization preserves practical training efficiency. These results suggest that exact-flow integration provides a principled and scalable way to improve delta-rule linear attention, and may inspire future work on exact solvers for more general continuous-time attention-like architectures.

Acknowledgment

This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG3-GV2023-010). This work is also supported by the National Research Foundation, Singapore, under its National Large Language Models Funding Initiative (AISG Award No: AISG-NMLP-2024-005), and NTU SUG project #025628-00001: Posttraining to Improve Embodied AI Agents. This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme.

References

Christian Bischof and Charles Van Loan. The wy representation for products of householder matrices. *SIAM Journal on Scientific and Statistical Computing*, 8(1):s2–s13, 1987.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

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.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Leonhard Euler. *Institutiones calculi integralis*, volume 1. impensis Academiae imperialis scientiarum, 1792.

Daniel Y Fu, Tri Dao, Khaled K Saab, Armin W Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*, 2022.

Google. Gemini deep research, 2025. URL `https://gemini.google/overview/deep-research/`. Accessed: 2025-12-11.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Re. Hippo: Recurrent memory with optimal polynomial projections, 2020. URL https://arxiv.org/abs/2008.07669.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2022a. URL https://arxiv.org/abs/2111.00396.

Albert Gu, Ankit Gupta, Karan Goel, and Christopher Ré. On the parameterization and initialization of diagonal state space models, 2022b. URL https://arxiv.org/abs/2206.11893.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Arieh Iserles. *A first course in the numerical analysis of differential equations*. Number 44. Cambridge university press, 2009.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, 2024.

Thierry Joffrain, Tze Meng Low, Enrique S Quintana-Ortí, Robert van de Geijn, and Field G Van Zee. Accumulating householder transformations, revisited. *ACM Transactions on Mathematical Software (TOMS)*, 32(2):169–179, 2006.

Matt Gardner Johannes Welbl, Nelson F. Liu. Crowdsourcing multiple choice science questions. 2017.

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

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. An llm compiler for parallel function calling. In *Forty-first International Conference on Machine Learning*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

Hanyu Lai, Xiao Liu, Junjie Gao, Jiale Cheng, Zehan Qi, Yifan Xu, Shuntian Yao, Dan Zhang, Jinhua Du, Zhenyu Hou, et al. A survey of post-training scaling in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2771–2791, 2025.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023.

Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Luke Zettlemoyer. Mega: Moving average equipped gated attention. *arXiv preprint arXiv:2209.10655*, 2022.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

OpenAI. Introducing deep research, 2025. URL `https://openai.com/index/introducing-deep-research/`. Accessed: 2025-12-11.

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 (volume 1: Long papers)*, pages 1525–1534, 2016.

Michael Poli, Armin W Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, Ce Zhang, and Stefano Massaroli. Mechanistic design and scaling of hybrid architectures, 2024. URL `https://arxiv.org/abs/2403.17844`.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International conference on machine learning*, pages 9355–9366. PMLR, 2021.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama, 2023. URL `https://huggingface.co/datasets/cerebras/SlimPajama-627B`.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Kimi Team, Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, et al. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025a.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025b.

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.

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. *Advances in neural information processing systems*, 37: 115491–115522, 2024b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

A Limitations

EFLA is derived under a zero-order-hold (ZOH) continuous-time formulation of Euler-style delta-rule dynamics. This formulation enables an exact closed-form flow by exploiting the token-wise rank-1 transition matrix $\mathbf{A}_t = \mathbf{k}_t\mathbf{k}_t^\top$, and directly removes the Euler discretization error of the corresponding ZOH state transition. However, this does not eliminate all sources of error in practical sequence modeling. Models may still be affected by finite-precision computation, optimization dynamics, data limitations, architectural choices, and the approximation gap between linear attention and softmax attention. A more complete analysis of how exact-flow updates interact with these factors is left for future work.

Our experiments validate EFLA through controlled robustness tests, language modeling at 340M and 1.3B scales, downstream benchmarks, the MAD synthetic benchmark, and training-throughput comparisons. Nevertheless, due to the substantial computational cost of training larger long-context models from scratch, we do not yet provide exhaustive scaling experiments beyond the 1.3B regime or extensive evaluations on deployment-oriented long-context tasks. Although EFLA preserves the linear-time recurrence and WY/UT-compatible chunkwise-parallel structure of DeltaNet, further empirical characterization is needed to understand its scaling behavior under larger pretraining budgets, longer context windows, and broader downstream settings.

B Broader Impacts

This work proposes a methodological improvement to linear-time attention by deriving an exact-flow update for delta-rule linear attention. Its potential positive impact lies in improving the efficiency and robustness of long-sequence modeling, which may reduce the computational cost and energy consumption of training and deploying sequence models. More efficient recurrent attention mechanisms may also make long-context processing more accessible under limited hardware budgets.

At the same time, efficiency improvements can lower the cost of deploying large language models more broadly. As with other advances in efficient sequence modeling, this may indirectly amplify both beneficial applications and potential misuse, including unsafe generation, automated misinformation, or other harmful uses of language models. This paper does not introduce a new dataset, release a deployed model, or target a specific high-risk application domain. Therefore, we do not study application-specific societal effects, and any deployment of models using EFLA should follow standard safety evaluation and responsible-release practices.

C Experimental Setting

We used 8 A100×80G GPUs for 340M and 1.3B language modeling experiments. The random seed it set to 42. Each model uses AdamW for optimization, with a peak learning rate of 3×10^{-4}. The 340M models are trained for 8 billion tokens with a global batch size of 1M tokens, while the 1.3B models are trained for 50 billion tokens with a global batch size of 2M tokens. We use a cosine learning rate schedule, starting with a warm-up phase of 1 billion tokens for the 340M models and 2 billion tokens for the 1.3B models (corresponding to 1024 warm-up steps). Both have configurations that initial and final learning rates set at 3×10^{-5}. We apply a weight decay of 0.1 and use gradient clipping at a maximum of 1.0. The head dimension is set to 128, and the kernel size for convolution layers is set at 4. To ensure numerical stability, specifically to prevent division by zero when the key norm $\|\mathbf{k}_t\|^2$ vanishes, we clip it with a lower bound of $\epsilon = 1 \times 10^{-12}$. Additionally, we employ the *expm1* function to compute the numerator $1 - e^{-\beta_t\lambda_t}$, preserving precision for small exponents. We empirically find that directly using the projected keys, or applying L_2 normalization to the keys, leads to performance similar to DeltaNet. In contrast, scaling the keys by $\sqrt{d}$, where d is the head dimension, consistently gives better results. A possible explanation is that the difference between the exact-flow update becomes more pronounced when the effective key magnitude is larger, whereas small or normalized keys tend to place the model in a regime where the two updates behave similarly. This observation is also consistent with the role of the key norm in the exact-flow formulation, where the key magnitude directly affects the state transition and memory update. Importantly, such scaling is naturally supported by EFLA whereas applying the same scaling to the Euler-style DeltaNet update can lead to overly aggressive or unstable state transitions.

D Construction and Properties of *rank-1* Matrices

$\mathbf{A}_t$ is a *rank-1* matrix, and it satisfies:

$$\mathbf{A}_t^2 = \mathbf{k}_t\mathbf{k}_t^\top\mathbf{k}_t\mathbf{k}_t^\top = \mathbf{k}_t(\mathbf{k}_t^\top\mathbf{k}_t)\mathbf{k}_t^\top = \lambda_t\mathbf{A}_t, \tag{32}$$

Where $\lambda_t = \mathbf{k}_t^\top\mathbf{k}_t$ is scalar value.

Then it gives us a key property: $\mathbf{A}_t$ is a scaled projection matrix (i.e., $\mathbf{A}_t^2 = \lambda_t\mathbf{A}_t$).

E General Solutions of Ordinary Differential Equations

We start with a first-order linear matrix ODE:

$$\frac{d\mathbf{S}}{dt} = -A\mathbf{S} + \mathbf{b}, \tag{33}$$

Which can be rewrite as:

$$\frac{d\mathbf{S}}{dt} + A\mathbf{S} = \mathbf{b}, \tag{34}$$

For this type of differential equation, the integrating factor is:

$$e^{\int \mathbf{A}\, dt}, \tag{35}$$

Since $\mathbf{A}$ is constant, the integrating factor is simply:

$$e^{\mathbf{A}t} \tag{36}$$

Multiply the entire equation by $e^{\mathbf{A}t}$:

$$e^{\mathbf{A}t}\left(\frac{d\mathbf{S}}{dt} + A\mathbf{S}\right) = e^{\mathbf{A}t}\mathbf{b}, \tag{37}$$

Expanding the left-hand side:

$$e^{\mathbf{A}t}\frac{d\mathbf{S}}{dt} + e^{\mathbf{A}t}\mathbf{A}\mathbf{S} = e^{\mathbf{A}t}\mathbf{b}, \tag{38}$$

By the product rule for matrix-vector multiplication:

$$\frac{d}{dt}(e^{\mathbf{A}t}\mathbf{S}) = \left(\frac{d}{dt}e^{\mathbf{A}t}\right)\mathbf{S} + e^{\mathbf{A}t}\frac{d\mathbf{S}}{dt}, \tag{39}$$

and since $\frac{d}{dt}e^{\mathbf{A}t} = \mathbf{A}e^{\mathbf{A}t}$, we have:

$$\frac{d}{dt}(e^{\mathbf{A}t}\mathbf{S}) = (\mathbf{A}e^{\mathbf{A}t})\mathbf{S} + e^{\mathbf{A}t}\frac{d\mathbf{S}}{dt}, \tag{40}$$

Notice this matches exactly the left-hand side of Eq. 38 (since $\mathbf{A}$ and $e^{\mathbf{A}t}$ commute).

The equation becomes:

$$\frac{d}{dt}(e^{\mathbf{A}t}\mathbf{S}) = e^{\mathbf{A}t}\mathbf{b}, \tag{41}$$

Integrate both sides from t (initial time) to $t + \beta_t$ (final time). To avoid confusion, we use τ as the integration variable:

$$\int_t^{t+\beta_t} \frac{d}{d\tau}(e^{A\tau}\mathbf{S}(\tau))\, d\tau = \int_t^{t+\beta_t} e^{A\tau}\mathbf{b}\, d\tau, \tag{42}$$

$$\left[e^{\mathbf{A}\tau}\mathbf{S}(\tau)\right]_t^{t+\beta_t} = \int_t^{t+\beta_t} e^{\mathbf{A}\tau}\mathbf{b}\, d\tau, \tag{43}$$

Thus:

$$e^{\mathbf{A}(t+\beta_t)}\mathbf{S}(t+\beta_t) - e^{\mathbf{A}t}\mathbf{S}(t) = \int_t^{t+\beta_t} e^{\mathbf{A}\tau}\mathbf{b}\, d\tau, \tag{44}$$

Multiply both sides by $e^{-\mathbf{A}(t+\beta_t)}$:

$$\mathbf{S}(t+\beta_t) - e^{-\mathbf{A}(t+\beta_t)} e^{\mathbf{A}t} \mathbf{S}(t) = e^{-\mathbf{A}(t+\beta_t)} \int_t^{t+\beta_t} e^{\mathbf{A}\tau} \mathbf{b}\, d\tau, \tag{45}$$

Simplify using exponential properties:

$$\mathbf{S}(t+\beta_t) - e^{-\mathbf{A}\beta_t} \mathbf{S}(t) = \int_t^{t+\beta_t} e^{-\mathbf{A}(t+\beta_t-\tau)} \mathbf{b}\, d\tau, \tag{46}$$

Since $e^{-\mathbf{A}(t+\beta_t)}$ is constant, it can be moved inside the integral.

Let $s = \tau - t$. Then:

$$\begin{cases} s = 0, & \tau = t \\ s = \beta_t, & \tau = t + \beta_t \end{cases} \quad \text{and} \quad d\tau = ds, \tag{47}$$

Substitute:

$$\int_0^{\beta_t} e^{-\mathbf{A}[t+\beta_t-(s+t)]} \mathbf{b}\, ds = \int_0^{\beta_t} e^{-\mathbf{A}(\beta_t - s)} \mathbf{b}\, ds, \tag{48}$$

Rename s back to τ (dummy variable) and replace constants with $\mathbf{A}_t, \mathbf{b}_t$:

$$\int_0^{\beta_t} e^{-(\beta_t-\tau)\mathbf{A}_t} \mathbf{b}_t\, d\tau, \tag{49}$$

Combining everything, the full solution is:

$$\mathbf{S}(t+\beta_t) = e^{-\beta_t \mathbf{A}_t} \mathbf{S}(t) + \int_0^{\beta_t} e^{-(\beta_t-\tau)\mathbf{A}_t} \mathbf{b}_t\, d\tau, \tag{50}$$

F Exact-flow formulation of Gated DeltaNet.

Gated DeltaNet updates the fast-weight state by

$$\mathbf{S}_t = \alpha_t \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top. \tag{51}$$

Expanding the transition term gives

$$\alpha_t \left(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right) = \mathbf{I} - \left[(1-\alpha_t)\mathbf{I} + \alpha_t \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right]. \tag{52}$$

Therefore, the update can be written in Euler form as

$$\mathbf{S}_t = \mathbf{S}_{t-1} - \left[(1-\alpha_t)\mathbf{I} + \alpha_t \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right] \mathbf{S}_{t-1} + \beta_t \mathbf{k}_t \mathbf{v}_t^\top. \tag{53}$$

Thus, it can be interpreted as an explicit Euler discretization of

$$\frac{d\mathbf{S}(\tau)}{d\tau} = -\left[(1-\alpha_t)\mathbf{I} + \alpha_t \beta_t \mathbf{k}_t \mathbf{k}_t^\top\right] \mathbf{S}(\tau) + \beta_t \mathbf{k}_t \mathbf{v}_t^\top. \tag{54}$$

Let

$$\delta_t = 1 - \alpha_t, \qquad \rho_t = \alpha_t \beta_t, \qquad \lambda_t = \mathbf{k}_t^\top \mathbf{k}_t, \tag{55}$$

and define

$$\mathbf{A}_t = \delta_t \mathbf{I} + \rho_t \mathbf{k}_t \mathbf{k}_t^\top. \tag{56}$$

The ODE becomes

$$\frac{d\mathbf{S}(\tau)}{d\tau} = -\mathbf{A}_t \mathbf{S}(\tau) + \beta_t \mathbf{k}_t \mathbf{v}_t^\top. \tag{57}$$

Under the zero-order hold assumption, the exact solution over one update interval is

$$\mathbf{S}_t = e^{-\mathbf{A}_t} \mathbf{S}_{t-1} + \int_0^1 e^{-(1-\tau)\mathbf{A}_t} \beta_t \mathbf{k}_t \mathbf{v}_t^\top\, d\tau. \tag{58}$$

For the transition term, since:

$$\mathbf{A}_t = \delta_t \mathbf{I} + \rho_t \mathbf{k}_t \mathbf{k}_t^\top, \tag{59}$$

we have:

$$e^{-\mathbf{A}_t} = e^{-\delta_t} e^{-\rho_t \mathbf{k}_t \mathbf{k}_t^\top}. \tag{60}$$

Using the rank-one matrix exponential identity:

$$e^{-\rho_t \mathbf{k}_t \mathbf{k}_t^\top} = \mathbf{I} - \frac{1 - e^{-\rho_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{k}_t^\top, \tag{61}$$

we obtain:

$$e^{-\mathbf{A}_t} \mathbf{S}_{t-1} = e^{-\delta_t} \left[\mathbf{I} - \frac{1 - e^{-\rho_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{k}_t^\top \right] \mathbf{S}_{t-1}. \tag{62}$$

For the input term, since:

$$\mathbf{A}_t \mathbf{k}_t = (\delta_t + \rho_t \lambda_t) \mathbf{k}_t, \tag{63}$$

we have:

$$e^{-s\mathbf{A}_t} \mathbf{k}_t \mathbf{v}_t^\top = e^{-s(\delta_t + \rho_t \lambda_t)} \mathbf{k}_t \mathbf{v}_t^\top. \tag{64}$$

Let

$$\eta_t = \delta_t + \rho_t \lambda_t = (1 - \alpha_t) + \alpha_t \beta_t \lambda_t. \tag{65}$$

Then:

$$\int_0^1 e^{-(1-\tau)\mathbf{A}_t} \beta_t \mathbf{k}_t \mathbf{v}_t^\top \, d\tau = \beta_t \int_0^1 e^{-(1-\tau)\eta_t} \, d\tau \, \mathbf{k}_t \mathbf{v}_t^\top \tag{66}$$

$$= \beta_t \frac{1 - e^{-\eta_t}}{\eta_t} \mathbf{k}_t \mathbf{v}_t^\top. \tag{67}$$

Combining the transition and input terms gives the exact-flow counterpart:

$$\mathbf{S}_t = e^{-(1-\alpha_t)} \left[\mathbf{I} - \frac{1 - e^{-\alpha_t \beta_t \lambda_t}}{\lambda_t} \mathbf{k}_t \mathbf{k}_t^\top \right] \mathbf{S}_{t-1} + \beta_t \frac{1 - e^{-[(1-\alpha_t) + \alpha_t \beta_t \lambda_t]}}{(1 - \alpha_t) + \alpha_t \beta_t \lambda_t} \mathbf{k}_t \mathbf{v}_t^\top, \tag{68}$$

where $\lambda_t = \mathbf{k}_t^\top \mathbf{k}_t$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state that Exact Flow Linear Attention is an exact ODE solution within the DeltaNet rank-1 update family. We explicitly limit the claim to the chosen continuous-time delta dynamics and evaluate the method through controlled ablations, robustness tests, language modeling, and efficiency measurements.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in Appendix A. In particular, we clarify that the exactness claim is limited to the chosen continuous-time delta dynamics under the zero-order-hold assumption and does not remove floating-point error, modeling error, optimization error, or the approximation gap to softmax attention. We also discuss the experimental scope, including the evaluated model scales, and the need for further validation on larger models and broader long-context tasks.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

   Justification: The paper states the main assumptions used in the theoretical derivation, including the zero-order-hold assumption, the rank-1 dynamics matrix $\mathbf{A}_t = \mathbf{k}_t\mathbf{k}_t^\top$, and non-negative update coefficients. The main paper presents the EFLA update derivation, and the appendix provides the supporting proofs for the rank-1 matrix property and the closed-form ODE solution.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides the full mathematical specification of EFLA in Section 3 and its chunkwise parallel form in Section 3.3. The main experimental setup is described in Section 4 and Appendix C, including model scale, datasets, optimizer, learning rate, training tokens, batch size, schedule, random seed, numerical stabilization details, and evaluation benchmarks.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper uses publicly available datasets and benchmarks, including SlimPajama, Wikitext, LAMBADA, PiQA, HellaSwag, WinoGrande, ARC, BoolQ, OpenBookQA, SciQ, sMNIST, and MAD, which are cited in Section 4. However, the current submission does not yet include an anonymized code release or exact reproduction scripts, so we answer [No] while providing the algorithmic and experimental details needed to implement the method.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 4 describes the evaluated tasks, datasets, metrics, model scales, and baseline comparisons. Appendix C further reports the main training details, including GPU count, random seed, optimizer, learning rate, global batch size, number of training tokens, warm-up schedule, weight decay, gradient clipping, head dimension, convolution kernel size, and numerical stabilization choices.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper reports deterministic benchmark scores and controlled robustness curves in Section 4, but it does not currently report error bars, confidence intervals, or statistical significance tests. We leave repeated-run statistical analysis to future work due to the computational cost of training language models at the reported scales.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C reports that the 340M and 1.3B language modeling experiments use 8 A100×80G GPUs, and Section 4 reports throughput comparisons.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is methodological research on efficient sequence modeling and does not involve human subjects, private data, or safety-sensitive deployment. The experiments use standard public datasets and benchmarks, and we are not aware of any aspect of the research that conflicts with the NeurIPS Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper includes a broader-impact discussion in Appendix B. It identifies the potential positive impact of improving the efficiency and robustness of long-sequence modeling, while also noting that efficiency improvements may lower the cost of deploying language models in settings where misuse or unsafe generation remains a concern. The discussion also clarifies that the paper is methodological and does not release a deployed model or target a specific high-risk application.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a new high-risk dataset, image generator, or deployed language model. The contribution is an attention mechanism and its empirical evaluation on standard benchmarks, so release-specific safeguards for high-risk assets are not applicable.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper uses publicly available datasets and benchmarks, including SlimPajama, Wikitext, LAMBADA, PiQA, HellaSwag, WinoGrande, ARC, BoolQ, OpenBookQA, SciQ, sMNIST, and MAD. We properly credit the creators of these assets through citations in the main apaper and references, and we use these assets only for standard research evaluation following their publicly available licenses and terms of use.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset, benchmark, or released model asset. The main contribution is a new linear attention formulation, whose algorithmic details are documented in Section 3 and Section 3.3.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, user studies, human-subject experiments, or participant compensation. All experiments are conducted on standard machine learning benchmarks.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human-subject research, crowdsourcing, user studies, or collection of participant data. Therefore, IRB approval or an equivalent human-subjects review is not applicable.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core methodology does not use LLMs as an important, original, or non-standard component. Any use of LLMs, if applicable, is limited to writing, editing, or formatting assistance and does not affect the method, experiments, scientific claims, or originality of the research.