Kaczmarz Linear Attention

A normalized step for residual memory writes

Paper: Jiaxuan Zou, Ruifeng Ren, Yong Liu

Research group reading presentation

arXiv:2605.08587v1 | 13 November 2026

KLA normalizes the residual write by key energy

One-line change to the paper's GDN template

$$\beta_t = \eta_t \quad \longrightarrow \quad \boxed{\beta_t = \frac{\eta_t}{\|k_t\|_2^2 + \varepsilon}}$$

- Preserve the state, decay gate, residual direction and chunk solver.
- Derive the ideal step from a minimum-change projection.
- Test language modeling, controlled memory tasks and runtime.

Paper: arXiv v1, 9 May 2026.
Publication not found as of 2026-11-13; author page: Under Review.
Official code: public anonymized implementation.

Paper, Abstract and Secs. 1, 3; PDF pp. 1–5. Status checked 2026-11-13.

Residual writes edit an existing association

$$S_t \in \mathbb{R}^{d_k \times d_v}, \quad k_t, q_t \in \mathbb{R}^{d_k}, \quad v_t \in \mathbb{R}^{d_v}$$

$$\begin{aligned} \widetilde{S}_{t-1} &= \alpha_t S_{t-1} && \text{decay old memory} \\ e_t &= v_t - \widetilde{S}_{t-1}^{\top} k_t && \text{predict the missing value} \\ S_t &= \widetilde{S}_{t-1} + \beta_t k_t e_t^{\top} && \text{edit the active key direction} \end{aligned}$$

- Additive attention writes $k_t v_t^{\top}$ without subtracting the prediction.
- GDN uses a learned scalar $\beta_t = \eta_t$ for the residual write.
- KLA keeps the direction and changes its length.

Paper, Sec. 2, PDF pp. 2–3. Scalar-gated notation used throughout.

Key norm changes the effective correction

$$e_t^+ = v_t - S_t^\top k_t = \left(1 - \beta_t \|k_t\|_2^2\right) e_t$$

The effective correction is $\beta_t \|k_t\|_2^2$

Large key energy can cause overshoot; small energy can under-correct.

Teaching example: scalar state, $\widetilde{S} = 0$, $k = 2$, $v = 1$.

Step	β	New state S	Readout Sk
Unnormalized	1	2	4
Kaczmarz	1/4	1/2	1

Paper, Sec. 3.1, Eq. (4), PDF p. 4. Example: presentation derivation.

Exact projection makes the smallest feasible state change

$$\min_{S} \frac{1}{2}\|S - \widetilde{S}_{t-1}\|_F^2 \qquad \text{subject to} \qquad S^\top k_t = v_t$$

1. For $k_t \neq 0$, the optimal correction has form $k_t c^\top$.
2. The constraint gives $c\|k_t\|_2^2 = e_t$.
3. Hence the minimum-change update is

$$\boxed{S_t = \widetilde{S}_{t-1} + \frac{k_t e_t^\top}{\|k_t\|_2^2}}$$

It changes only the current key direction and exactly fits this key-value pair.

Paper, Sec. 3.1, Eqs. (1)–(2), PDF p. 3; Appendix A, pp. 13–14.

Three views yield the same ideal coefficient

$$\mathcal{L}_t(S) = \frac{1}{2}\|S^\top k_t - v_t\|_2^2, \qquad \nabla_S \mathcal{L}_t(\widetilde{S}_{t-1}) = -k_t e_t^\top$$

$$\mathcal{L}_t(\widetilde{S}_{t-1} + \tau k_t e_t^\top) = \frac{1}{2}(1 - \tau\|k_t\|_2^2)^2\|e_t\|_2^2$$

- Exact line search yields $\tau^* = 1/\|k_t\|_2^2$ for nonzero residual.
- The correction is orthogonal to every H with $H^\top k_t = 0$.
- The same step is normalized gradient descent on the state loss.

Exact projection: $k_t \neq 0$, $\eta_t = 1$, $\varepsilon = 0$.

Paper, Theorem 1 and Eq. (4), PDF p. 4; Appendix A.1, pp. 13–14.

Relaxation controls the remaining residual

$$\beta_t = \frac{\eta_t}{\|k_t\|_2^2 + \varepsilon}, \qquad \eta_t = \sigma(\text{input projection}), \quad \varepsilon = 10^{-6}$$

$$\boxed{e_t^+ = \left(1 - \eta_t \frac{\|k_t\|_2^2}{\|k_t\|_2^2 + \varepsilon}\right) e_t}$$

- For nonzero key and $0 < \eta_t \leq 1$, the factor is in $[0, 1)$.
- With $\varepsilon = 0$, the gate directly selects the corrected fraction.
- Comparison is against the *decayed* state $\widetilde{S}_{t-1}$.
- At zero key with positive ε, the write vanishes.

Paper, Eq. (3), Proposition 2 / Eq. (5), PDF pp. 3–5; Appendix C, p. 18.

The inference interface stays the same

1. Decay: $\widetilde{S}_{t-1} = \alpha_t S_{t-1}$.
2. Predict residual: $e_t = v_t - \widetilde{S}_{t-1}^{\top} k_t$.
3. Compute the normalized scalar β_t.
4. Write: $S_t = \widetilde{S}_{t-1} + \beta_t k_t e_t^{\top}$.
5. Read: $o_t = S_t^{\top} q_t / \|q_t\|_2$.

Implementation condition that matters

If the actual write key is already unit-normalized, the denominator is $1 + \varepsilon$ and the change nearly disappears. The paper explicitly specifies query normalization; baseline key processing needs clarification.

Paper, Sec. 3.3 and Algorithm 1, PDF p. 5. Unit-key observation: algebraic deduction.

Training preserves the GDN triangular solve

$$\gamma_i = \alpha_1 \cdots \alpha_i, \quad B = \mathrm{Diag}(\beta_i), \quad D_\gamma = \mathrm{Diag}(\gamma_i)$$
$$A_{ij} = \gamma_i / \gamma_j \ (j \le i), \qquad A^- = \text{strict lower part of } A$$

Effective residual writes $U \in \mathbb{R}^{C \times d_v}$

$$\left[I + B(A^- \odot KK^\top) \right] U = B(V - D_\gamma K S_0)$$

- $K, Q \in \mathbb{R}^{C \times d_k}$ and $V \in \mathbb{R}^{C \times d_v}$.
- Only the values in B change relative to the GDN solver.
- Fixed head dimensions and chunk length give linear sequence scaling.

Paper, Sec. 4, Eq. (7), PDF pp. 5–6. Nonzero cumulative decay assumed.

The main experiment fixes architecture and token budget

Model	0.4B parameters; 24 layers; hidden size 1024
Heads / FFN	8 heads, head dimension 128; FFN expansion $4\times$
Data / context	1B SlimPajama tokens; 4K training context
Optimizer	AdamW, $(0.9, 0.95)$; weight decay 0.1
Schedule	Cosine; 2% warmup; gradient clipping 1.0
Resources	4 A100 80GB GPUs; about 4 h per run

- Pretraining: KLA, GDN, DeltaNet, GLA, Longhorn and Mamba2.
- Other comparisons: KLA, GDN and Mamba2.
- No multi-seed pretraining; synthetic tasks use seed 42.

Paper, Sec. 5.1, PDF p. 6; Appendix C, p. 18. Configurations as reported.

KLA reduces pretraining PPL from 8.50 to 8.09

Model	KLA	DeltaNet	GLA	GDN	Longhorn	Mamba2
PPL ↓	8.09	8.26	8.42	8.50	8.79	9.41

0.4B parameters, 1B tokens, 4K training context

A decrease of 0.41 PPL relative to GDN, or about 4.82%.
The gap to the next-best baseline, DeltaNet, is 0.17 PPL.

This is the main fixed-budget language-modeling result.

Paper, Table 1, PDF p. 2; Sec. 5.2 and Fig. 1, p. 6. Relative change calculated here.

KLA has lower PPL at every tested context length

Context	1K	2K	4K	8K	16K	32K	65K
KLA	39.1	39.8	40.0	43.6	45.3	44.5	43.6
GDN	42.9	43.5	43.8	47.4	49.4	48.4	47.5
Mamba2	49.9	50.9	52.2	56.4	58.7	57.3	56.7

- PPL ↓; no long-context fine-tuning.
- GDN gap: 3.7–4.1 PPL at each tested length.
- At 65K: 43.6 versus 47.5, about 8.21% lower.

Keep this protocol separate from Table 1

At 4K, Table 3 reports 40.0 / 43.8; Table 1 reports 8.09 / 8.50. The paper does not sufficiently reconcile their evaluation pipelines.

Paper, Table 3 and Sec. 5.2, PDF p. 7. All table values are PPL.

Synthetic tasks probe distinct memory demands

Task	Memory demand	Protocol
MQAR	Multiple key-value lookups	Train 256; test to 2048
S-NIAH	One association amid distractors	Test 1K–8K
Palindrome	Ordered reverse reproduction	Test 256–2048
Stack	Repeated LIFO push/pop edits	Test 256–2048

- MQAR: 2-layer model, 10K steps.
- S-NIAH: 6K steps; Palindrome / Stack: 20K steps.
- S-NIAH specifies exact match; the other tasks report accuracy.

Paper, Sec. 5.3, PDF pp. 7–8; Appendix C, p. 18. Synthetic seed: 42.

MQAR gains emerge at the longest extrapolation

Length	256	512	1024	2048 (8×)
KLA	99.00	98.56	95.64	73.84
GDN	98.74	98.36	94.31	66.81
Mamba2	**99.08**	**99.02**	**96.54**	64.29

At 8× the training length

KLA improves over GDN by 7.03 percentage points.
Relative accuracy improvement: approximately 10.52%.

Mamba2 leads at shorter lengths; the ranking changes at 2048.

Paper, Table 4, PDF p. 8. MQAR validation accuracy (%); train length 256.

Retrieval improves; ordered reversal remains a boundary

Task	Length	KLA	GDN	Mamba2
S-NIAH	8K	100.00	98.50	0.05
Palindrome	2048	79.96	**80.74**	0.86
Stack	2048	96.94	95.64	93.82

- S-NIAH: KLA reaches 100% at all four tested lengths, 1K–8K.
- Stack: a 1.30-point gain at length 2048.
- Palindrome: KLA trails GDN by 0.78 points at length 2048.

Association editing and ordered storage impose different demands.

Paper, Table 4 and Fig. 2, PDF p. 8. All values are task accuracies (%).

Prefill matches GDN; measured decode throughput rises

Metric	KLA	GDN	Mamba2
131K prefill, ms ↓	1220	1227	1936
32K decode, token/s ↑	3.54	1.65	2.10

- Batch size 1; FLA evaluation path.
- Prefill difference versus GDN: about 0.57%.
- Decode ratio: $3.54/1.65 \approx 2.15\times$ (paper: $2.1\times$).

Interpretation

Reusing the solver explains similar dominant prefill cost. The paper does not isolate why a scalar substitution doubles measured decode throughput.

Paper, Sec. 5.4, PDF p. 7; Fig. 3, p. 9.

Normalization and gating have a joint trade-off

Variant	MQAR test ↑	MQAR 8× ↑	100M PPL ↓
KLA default	**98.39**	73.84	**753.65**
No normalization	0.03	0.01	—
Key-norm only	97.81	**77.68**	3809.71
Learned Scalar	97.77	74.46	22968.87

- Key-norm only improves extrapolation but worsens short-budget PPL.
- Learned Scalar: trainable parameter initialized at 1.0.

Configuration ambiguity

NoNorm is described as sigmoid gating, yet it collapses while the GDN reference succeeds. Additional key/module differences are not resolved.

Paper, Table 5, PDF p. 9; Appendix D, pp. 18–19. Accuracy columns in %.

Gating choices trade recall against short-budget PPL

Variant	MQAR test ↑	MQAR 8× ↑	100M PPL ↓
KLA default	98.39	73.84	753.65
Single Gate	0.01	0.03	2104.61
GDN reference	**98.72**	66.81	747.24
GLA reference	0.03	0.03	**356.01**

- GLA has the lowest PPL here, but near-zero MQAR accuracy.
- GDN also slightly beats KLA in 100M-token PPL.
- The 1B-token main result and this 100M-token ablation are separate.

Paper, Table 5, PDF p. 9; Appendix D, pp. 18–19. Accuracy columns in %.

State expansion trades recall against language modeling

Value expansion	MQAR test ↑	MQAR 8× ↑	100M PPL ↓
2	98.09	71.99	**762.33**
4	98.88	91.34	1669.35
8	95.31	**95.37**	6945.63
16	**99.18**	92.63	1383.38

- State expansion changes capacity and projection cost.
- It can strongly improve extrapolation on the synthetic task.
- It does not replace the write-strength trade-off in language modeling.

Paper, Table 5, PDF p. 9; Appendix D, pp. 18–19. Accuracy columns in %.

Normalized writes still share a fixed-capacity state

For another key k', the same write changes its readout by

$$\Delta(S^\top k') = \beta_t e_t (k_t^\top k').$$

- Orthogonal key directions avoid this write interference.
- Similar keys still interact; normalization does not add memory capacity.
- The local guarantee compares states after decay and before the next token.
- Very small keys make the stabilizer and numerical precision relevant.

Concrete task boundary

At length 2048, Palindrome accuracy is 79.96% for KLA and 80.74% for GDN.

Paper, Sec. 6 and Table 4, PDF pp. 8–9. Cross-key equation: reading derivation.

Key energy sets the scale; the gate sets write strength

1. **Mechanism:** key energy supplies the natural local step scale; η_t controls adaptive relaxation.
2. **Algorithm:** the coefficient fits the existing recurrent state and chunkwise triangular solve.
3. **Evidence:** lower 1B-token PPL and stronger long MQAR retrieval; Palindrome and short-budget ablations show concrete trade-offs.

Highest-value follow-up checks

Resolve actual key normalization, NoNorm/GDN configuration differences, and cached decode timing. Then test larger models and multiple seeds.

Paper, Secs. 3–7, PDF pp. 3–10. Follow-up priorities: reading analysis.

References and implementation

- Zou, Ren and Liu. *Kaczmarz Linear Attention*. arXiv:2605.08587v1, 2026.
- Official anonymized implementation: model components and experiment scripts.
- First author's publication page: Under Review. Publication not found as of 2026-11-13.
- Direct predecessor: Yang, Kautz and Hatamizadeh. *Gated Delta Networks*, ICLR 2025 (paper Ref. 45).
- Flash Linear Attention library (paper Ref. 42).

This presentation reads the specified arXiv v1; paper authors are listed on the title slide.

Backup: the relaxed write solves a proximal problem

$$\min_S \frac{1}{2}\|S - \widetilde{S}_{t-1}\|_F^2 + \frac{\mu_t}{2}\|S^\top k_t - v_t\|_2^2$$

$$\mu_t = \frac{\eta_t}{(1-\eta_t)\|k_t\|_2^2 + \varepsilon}$$

- Stationarity gives $S - \widetilde{S} + \mu_t k_t (S^\top k_t - v_t)^\top = 0$.
- Along $S = \widetilde{S} + \tau k_t e_t^\top$,

$$\tau = \frac{\mu_t}{1 + \mu_t \|k_t\|_2^2} = \frac{\eta_t}{\|k_t\|_2^2 + \varepsilon}.$$

At $\eta_t = 1, \varepsilon = 0$, use the limit $\mu_t \to \infty$; the displayed finite-penalty formula is singular at that endpoint.

Paper, Proposition 1, PDF p. 3; Appendix A.3, p. 14. Endpoint clarification: analysis.

Backup: a direct expansion gives the chunk equations

$$S_i = \gamma_i S_0 + \sum_{j \le i} \frac{\gamma_i}{\gamma_j} k_j u_j^\top$$

$$u_i = \beta_i \left[v_i - \gamma_i S_0^\top k_i - \sum_{j<i} \frac{\gamma_i}{\gamma_j} u_j (k_j^\top k_i) \right]$$

$$O = D_\gamma Q S_0 + (A \odot Q K^\top) U$$

$$S_{\text{out}} = \gamma_C S_0 + K^\top \operatorname{Diag}(\gamma_C/\gamma_1, \ldots, 1) U$$

- Q contains normalized read queries; γ_i is nonzero.
- The write at position j decays only at later positions.

Paper, Eqs. (7)–(9), PDF p. 5. Expansion written consistently in the S orientation.

Backup: Appendix B has inconsistent intermediate formulas

- It defines $R_0 = S_0^\top \in \mathbb{R}^{d_v \times d_k}$, then writes $P_i R_0$ with $P_i \in \mathbb{R}^{d_k \times d_k}$.
- The H_i expressions mix $v_j k_j^\top$ and $k_j u_j^\top$ orientations.
- It defines $\gamma_{j \to i} = \gamma_i / \gamma_{j-1}$ on p. 15, then substitutes γ_i / γ_j on p. 17.

How to use the result

The direct expansion on the previous slide yields the main-text chunk system. Keep the S orientation and decay only after each write; do not copy the inconsistent intermediate WY expressions.

Paper, Appendix B, Eqs. (18)–(22), PDF pp. 15–17. Inconsistency analysis.

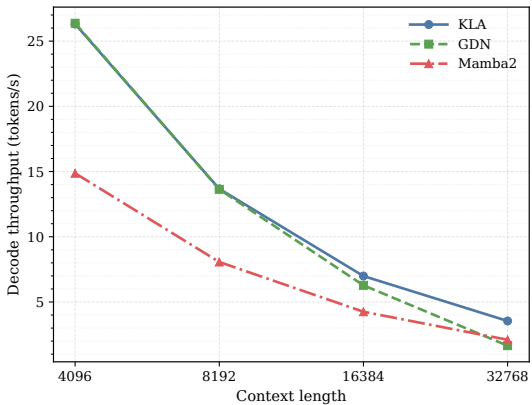
Backup: the ablations use a distinct protocol

- **MQAR:** 32 key-value pairs, length 256; 20K / 2K / 2K train / validation / test sequences.
- Batch 32; learning rate 10^{-3}; up to 10K steps; evaluation every 200 steps; early stopping patience 10; seed 42.
- **Language modeling:** 100M SlimPajama tokens; sequence length 4096; learning rate 10^{-4}.
- Validation: 20 runs, each averaging 15 iterations.
- Large state variants reduce micro-batch size to avoid OOM.

Table 4 uses MQAR validation accuracy; Table 5 reports test accuracy.

Paper, Appendix D, PDF pp. 18–19; Tables 4–5, pp. 8–9.

Backup: decode timing needs a fuller explanation



- All three curves decline strongly with context length.
- A fixed recurrent state alone does not explain that decline.
- The paper does not isolate caching, recomputation or timing scope.

The measured 2.1× ratio applies to this benchmark path.

Original Fig. 3b, PDF p. 9; caption specifies batch size 1. Timing discussion: analysis.