MesaNet: Sequence Modeling by Locally Optimal Test-Time Training

A reading of the ICLR 2026 paper

Paper: Johannes von Oswald, Nino Scherrer, et al.
Google, Google DeepMind, MIT CSAIL

arXiv:2506.05233v2, June 3, 2026
Research discussion • October 20, 2026

Solve a small regression problem inside every sequence layer.

Can a fixed memory learn the context more accurately?

The proposal

Replace one online learning step with the solution of a cumulative, regularized regression problem.

- Store two fixed-size sufficient statistics of past keys and values.
- Solve only for the current query using conjugate gradient (CG).
- Reuse gated linear attention structure for parallel training.

Three questions: Does the memory improve? What does PPL conceal? How much computation does the solver need?

Paper, Secs. 1–2, PDF pp. 1–5.

Mesa solves regression over the retained history

DeltaNet: one gradient step on the latest squared error

$$\ell_t(\Phi) = \tfrac{1}{2}\,\|v_t - \Phi k_t\|^2\,, \qquad \Phi_t = \Phi_{t-1} + \beta_t(v_t - \Phi_{t-1}k_t)k_t^\top\,.$$

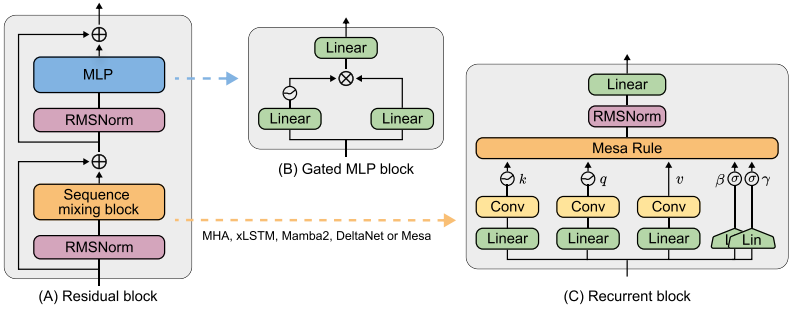
Mesa: solve a weighted regression on all retained associations

$$\widehat{\Phi}_t = \arg\min_{\Phi}\left\{\tfrac{1}{2}\sum_{i\le t}\zeta_{ti}\beta_i\,\|v_i - \Phi k_i\|^2 + \tfrac{1}{2}\,\mathrm{Tr}(\Phi\Lambda\Phi^\top)\right\}.$$

- "Local" refers to a layer/head objective; it has a unique optimum.
- Outer next-token training learns which keys, values and gates to produce.

Paper, Eqs. (3)–(8), PDF pp. 3–4; Appendix B, pp. 21–23. Input gates are written explicitly here.

The sequence mixer changes within a common backbone



- Pre-norm residual blocks alternate a sequence mixer and a SwiGLU MLP.
- Mesa produces k, q, v, input/forget gates, and normalized head outputs.
- Hawk-Mesa alternates Hawk and Mesa blocks.

Original Fig. 1, PDF p. 2; Appendix E, pp. 28–30.

A positive regularizer makes the regression well defined

$$\zeta_{ti} = \prod_{s=i+1}^{t} \gamma_s, \quad i \le t, \qquad 0 \le \gamma_s, \beta_s \le 1, \quad \Lambda \succ 0.$$

$$0 = \nabla_\Phi \mathcal{L}_t = \Phi \underbrace{\left(\sum_{i \le t} \zeta_{ti} \beta_i k_i k_i^\top + \Lambda \right)}_{H_t + \Lambda} - \underbrace{\sum_{i \le t} \zeta_{ti} \beta_i v_i k_i^\top}_{G_t}.$$

The solution

$$\widehat{\Phi}_t = G_t (H_t + \Lambda)^{-1}, \qquad o_t = \widehat{\Phi}_t q_t.$$

$k, q \in \mathbb{R}^{n_a}$, $v, o \in \mathbb{R}^{n_v}$; $G, \Phi \in \mathbb{R}^{n_v \times n_a}$, $H, \Lambda \in \mathbb{R}^{n_a \times n_a}$.

Paper, Eqs. (4)–(8), PDF pp. 3–4; derivation in Appendix D, pp. 25–26.

Two statistics replace the full history

$$
\begin{aligned}
G_t &= \gamma_t G_{t-1} + \beta_t v_t k_t^\top, \\
H_t &= \gamma_t H_{t-1} + \beta_t k_t k_t^\top, \\
(H_t + \Lambda) q_t^* &= q_t, \qquad o_t = G_t q_t^*.
\end{aligned}
$$

- G_t: associations between keys and values.
- H_t: repeated and correlated directions in key space.
- Λ: fixed over time, so forgetting does not erase regularization.

For $n_a = n_v = d$, persistent state is $2d^2$ numbers per head and sequence.
Core cost per token: $O((K+1)d^2)$ for K CG iterations.

Paper, Eqs. (7)–(8), PDF p. 4; Table 5, p. 29.

CG solves a query direction without forming an inverse

$$A_t q_t^* = q_t, \qquad A_t = H_t + \Lambda.$$

- Main cost per iteration: $A_t p$.
- Other operations: inner products and vector updates.
- Main experiments: 30 iterations.

Adaptive stopping

$$r_j = q_t - A_t x_j$$

$$\|r_j\| \leq \epsilon \|r_0\|.$$

Relative to the initial residual, with an iteration cap.

Interpretation: computation depends on the head and context through the linear system. Finite CG approximates the exact optimum.

Paper, Appendix C, Algorithm 1, PDF pp. 24–25; Sec. 5, pp. 9–10.

Each CG iteration can reuse chunked linear attention

For a chunk starting after token c, absorb $\sqrt{\beta_i}$ into k_i, v_i:

$$o_t = \zeta_{tc} G_c q_t^* + \sum_{i=c+1}^{t} \zeta_{ti} v_i (k_i^\top q_t^*).$$

The expensive solver operation has the same structure:

$$(H_t + \Lambda) p_t = \zeta_{tc} H_c p_t + \sum_{i=c+1}^{t} \zeta_{ti} k_i (k_i^\top p_t) + \Lambda p_t.$$

- Store full G, H only at chunk boundaries.
- Process token positions within a chunk using matrix multiplications.
- CG rounds remain sequential; positions within a round are parallel.

Expanded from Eq. (8), PDF p. 4; Appendix D, pp. 25–28. Cross-chunk forgetting factors are retained explicitly.

Training also needs a linear solve in the backward pass

Let $g_t = \partial \mathcal{J} / \partial o_t$ and solve

$$(H_t + \Lambda) u_t = G_t^\top g_t.$$

Then

$$\frac{\partial \mathcal{J}}{\partial q_t} = u_t, \quad \frac{\partial \mathcal{J}}{\partial G_t} = g_t q_t^{*\top}, \quad \frac{\partial \mathcal{J}}{\partial \Lambda_{ii}} = -u_{t,i} q_{t,i}^*.$$

- The paper differentiates the solution map, assuming an accurate solve.
- Key/query normalization and $\Lambda_{ii} \geq 0.25$ aid stability.
- Pallas solver matmuls use FP32; surrounding activations use BF16.

Paper, Appendix D, PDF pp. 26–28; G.5, pp. 33–34; J, p. 37. Gradient symbols renamed for clarity.

The study controls architecture, data and tokenizer

Approximate size	Blocks	Heads	Embedding size
140M	14	6	768
440M	28	8	1024
940M (“1B”)	28	12	1536

- SlimPajama: 15B / 50B tokens; GPT-2 tokenizer; train length 2048.
- Same data order and backbone; head key dimension 128.
- Learning rates tuned by model; SWA references use a fixed rate.

Comparison scope: controlled sequence mixers, with matched parameters and data. FLOPs are not matched; some baseline parameterizations are adapted.

Paper, Sec. 4, PDF pp. 6–7; Appendices E and G, pp. 28–34. Result tables use approximate 145M / 400M / 1B labels.

Average PPL reaches the Transformer baseline

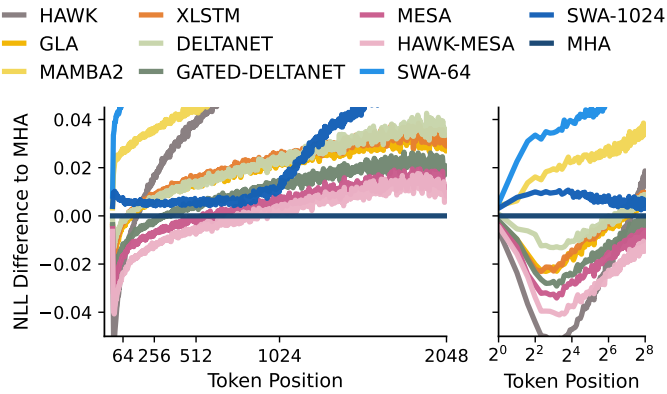
1B models, 50B training tokens, evaluation length 2048

Model	SlimPajama ↓	Qasper ↓	Six-dataset avg. ↓
Mamba2	11.39	14.02	14.58
DeltaNet	11.01	13.44	14.05
Gated DeltaNet	10.89	13.28	13.87
MesaNet	**10.83**	**13.13**	**13.79**
Hawk-Mesa	10.78	13.20	13.75
SWA-1024	11.00	13.11	13.97
Transformer	10.86	12.86	13.79

Mesa ties the average while individual datasets differ.

Paper, Table 2, PDF p. 7. Average is the arithmetic mean of six dataset PPLs, not pooled-token PPL.

Position-wise loss reveals where the advantage comes from



$\Delta\mathrm{NLL}_k = \mathrm{NLL}_k^{\mathrm{model}} - \mathrm{NLL}_k^{\mathrm{MHA}}$. Below zero is better.

Mesa's early-token advantage extends beyond 512; later tokens favor MHA.

Original Fig. 3, PDF p. 7. 1B / 50B; right panel uses a log token-position axis.

Window controls separate different benchmark sensitivities

Global subset

Lambada, HellaSwag, RACE-M/H.

Scores improve as short windows expand; short-window models do not beat MHA.

Local subset

PIQA, WinoGrande, ARC-E/C, SIQA, BoolQ, OBQA, StoryCloze.

At least one of these window-sensitivity conditions fails.

1B / 50B	Global accuracy	Local accuracy
SWA-4	31.20	49.62
MesaNet	45.03	50.49
Transformer	45.54	51.62

Paper, Table 3(b), p. 8; Appendix L.2.1, pp. 44–46. Empirical grouping for these scales; the SWA window is per layer.

Recall improves, but the gap to full attention remains

1B / 50B model	Recall avg. (%)
DeltaNet	40.46
Gated DeltaNet	39.54
MesaNet	**41.79**
SWA-1024	40.84
Transformer	52.27

Mesa relative to baselines

+2.25 points vs. Gated DeltaNet.
−10.48 points vs. Transformer.

FDA: 32.67 vs. 70.96.
SQuAD: 46.25 vs. 46.92.

- Six recall tasks; inputs truncated to 2000 tokens.
- Greedy output, at most 48 tokens; case-insensitive target containment.
- Target position helps explain why tasks show different gaps.

Paper, Table 3(b), p. 8; Appendix L.2.2, Tables 15–16, pp. 47–48.

Few-shot learning depends strongly on the task

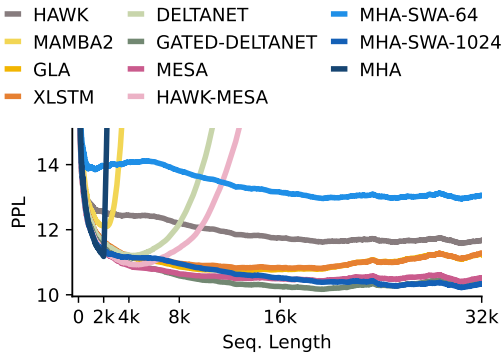
1B models trained on 50B tokens

Model	Word scrambling 100-shot accuracy (%)	Translation 50-shot BLEU-sb
Gated DeltaNet	8.90	8.53
MesaNet	**10.10**	8.17
Hawk-Mesa	8.61	8.81
SWA-1024	6.66	14.17
Transformer	6.98	13.61

- Mesa leads on the two reported word-transformation tasks.
- Translation into English favors attention; Mesa also trails Gated DeltaNet.

Paper, Table 3(b), p. 8; Tables 17–18, pp. 49–50. Two-task accuracy mean; three-language BLEU-sb mean.

Length extrapolation and long-range recall are distinct



32K average PPL, 1B / 50B

Gated DeltaNet	12.44
MesaNet	12.63
SWA-1024	12.75
Hawk-Mesa	15.09
Transformer	588.41

Mesa stays stable far beyond its 2K training length.

SWA remains competitive. The MHA failure concerns this 2K-trained baseline.

Original Fig. 4, p. 8: mean-so-far PPL on three datasets. Table 12, p. 43: four-dataset mean including PG19; these averages differ.

A strong noise-needle score can hide an essay weakness

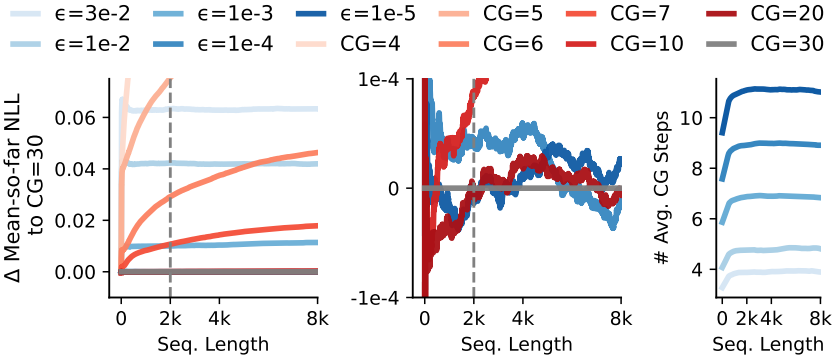
NIAH accuracy (%), 1B / 50B

Model	Noise 2K	Noise 4K	Essay 2K	Essay 4K
Gated DeltaNet	98.3	96.3	52.1	11.0
MesaNet	**99.5**	95.1	66.8	17.9
Hawk-Mesa	97.6	65.3	**90.9**	**55.5**
Transformer	99.7	0.0	98.2	0.0

- Means across 9 key/value types; 1000 samples per setting.
- Mesa Essay 2K: 66.8 ± 28.9 across types, versus Noise 99.5 ± 0.5.
- 4K is outside the training length; the haystack changes model rankings.

Paper, Appendix L.3, pp. 50–51; Table 19, p. 51. Standard deviations are across key/value types, not training runs.

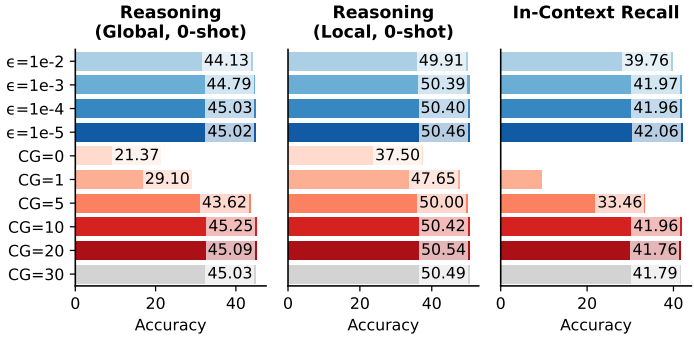
Dynamic stopping preserves the later-token predictions



- Uniformly reducing iterations hurts later positions more.
- A residual criterion allocates more work where the solve needs it.

Original Fig. 5, right side, PDF p. 10. 1B / 50B, trained with 30 CG steps. Left/middle: NLL difference from 30 steps; right: average iterations.

About 9 average iterations can retain 30-step performance



At $\epsilon = 10^{-4}$

Average CG steps:

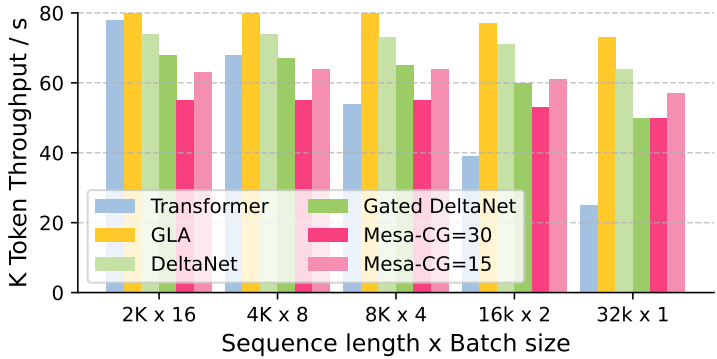
$$30 \longrightarrow \approx 9$$

Global accuracy:
$45.03 \rightarrow 45.03$

About 70% fewer iterations; end-to-end speedup is a separate measurement.

Original Fig. 5, left side, PDF p. 10, and Sec. 5. Some heads need many more iterations than others.

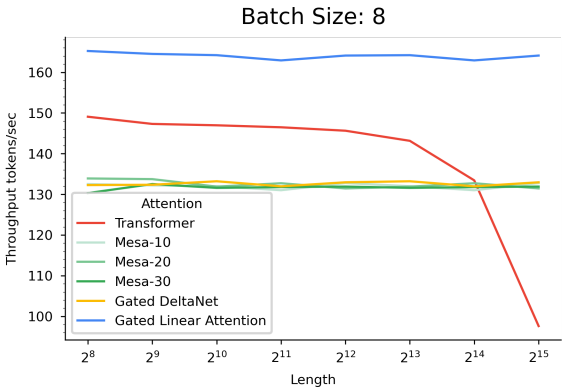
Parallel kernels make the added FLOPs more practical



- 1B models on H100; sequence length $\times$ batch size is about 32K tokens.
- Mesa is competitive, while GLA remains faster.

Original Fig. 2, right panel, PDF p. 5. FlashAttention-2 MHA and Triton recurrent implementations; fixed 15/30 CG steps.

Decode throughput depends on memory traffic and hardware



- H100, 1B models, batch size 8.
- Recurrent throughput is nearly flat as generated length increases.
- Mesa can approach Gated DeltaNet despite more arithmetic.
- GLA retains the highest throughput here.

Original Fig. 6, upper-right panel, PDF p. 35; Appendix G.7. Five timing repetitions; fixed CG budgets. TPU results have different relative timings.

The gain is better use of finite memory, at a compute cost

1. **Mechanism:** cumulative ridge regression, two sufficient statistics, and a chunk-parallel solver scale the Mesa layer.
2. **Evidence:** strong average PPL and useful recall gains coexist with a substantial gap to full attention on recall.
3. **Tradeoff:** more solver work and state; accuracy, latency and context use need separate evaluation.

A follow-up question

At equal latency or FLOPs, should extra budget buy a more accurate solve, a larger memory, or an attention component?

Paper, Secs. 4–6 and Appendices I–J. The follow-up question is reader analysis.

Source and discussion guide

Johannes von Oswald, Nino Scherrer, et al.
MesaNet: Sequence Modeling by Locally Optimal Test-Time Training.
ICLR 2026; arXiv:2506.05233v2 (June 3, 2026).

https://arxiv.org/abs/2506.05233v2

- Mechanism: Sec. 2 and Appendices C–E.
- Main evidence: Tables 2–3, Figs. 3–5.
- Reproduction details: Appendices G, L, M and N.
- Open questions: Sec. 6 and Appendices I–J.

Figures are original author assets. Selected tables were re-typeset from the paper. The following slides provide optional technical and experimental detail.

Backup: correlated keys illustrate the benefit of regression

$$k_1 = (1,0)^\top, \quad k_2 = (1,1)^\top/\sqrt{2}, \quad v_1 = 1,\ v_2 = 0.$$

No forgetting, input weights 1, $\Lambda = \lambda I$. Read both keys after two writes:

Rule	Map Φ	Read k_1	Read k_2
Additive GLA	$(1,0)$	1	$1/\sqrt{2}$
Delta, step size 1	$(1/2,-1/2)$	$1/2$	0
Mesa, $\lambda \to 0^+$	$(1,-1)$	1	0

- Correcting the latest error can change an earlier association.
- The cumulative solution accounts for key correlations.
- Positive regularization introduces shrinkage; capacity still limits recall.

Teaching example derived from the paper's updates, not a reported experiment.

Backup: why the earlier RLS implementation was restrictive

Without forgetting, set $R_0 = \Lambda$ and $R_t = R_{t-1} + k_t k_t^\top$:

$$R_t^{-1} = R_{t-1}^{-1} - \frac{R_{t-1}^{-1} k_t k_t^\top R_{t-1}^{-1}}{1 + k_t^\top R_{t-1}^{-1} k_t}.$$

- Efficient sequential inverse update, but limited training parallelism.
- Forgetting introduces inverse scaling by $1/\gamma_t$.
- A regularizer stored only in the initial state decays with the history.

The new decomposition

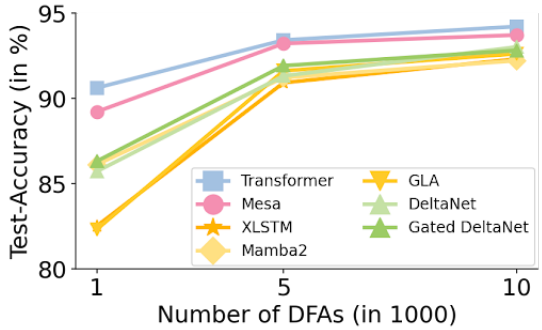
Accumulate H_t directly; add a fixed Λ at solve time.

Paper, Appendix H, PDF pp. 34–36. Warm starts and hybrid solvers remain future work.

Backup: synthetic tasks favor Mesa's context learning

MAD mean accuracy (%)	
Gated DeltaNet	71.0
xLSTM	73.2
Hawk-Mesa	73.5
Transformer	75.8
MesaNet	76.1

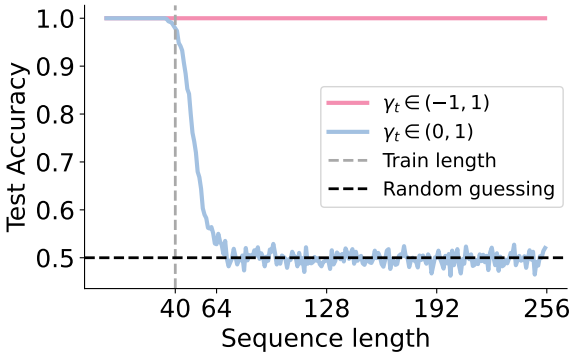
Tasks cover recall, memorization, copying and compression.



RegBench tests new automata: infer a grammar from context.

Paper, Table 9 and original Fig. 9, PDF p. 39; settings in Appendix F, pp. 30–31. These models are trained on the synthetic tasks.

Backup: parity exposes a tension with the regression view

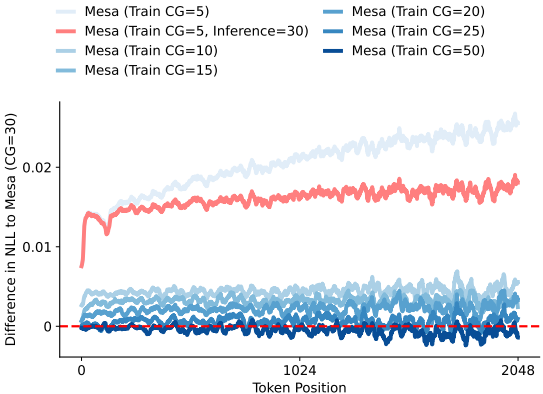


- Train on bitstrings of length 40; evaluate up to 256.
- Negative forget gates enable parity in this experiment.
- Λ starts at $50I$, with a lower bound of 49.

Large regularization makes the readout close to GLA. Negative gates also remove the usual nonnegative weighted-regression interpretation.

Original Fig. 8, right panel, PDF p. 36; Appendix I, pp. 36–37. An exploratory variant, not the main language model.

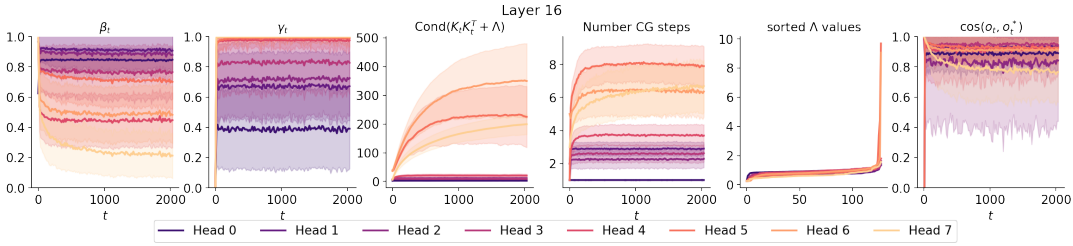
Backup: too few training iterations change the learned model



- About 400M parameters, 15B training tokens.
- 10–25 training steps cause small changes relative to 30.
- Five steps incur a clear loss penalty.
- Extra inference steps only partly recover the 5-step model.

Original Fig. 13, PDF p. 51; Appendix M. Training-budget ablation differs from reducing steps on a 30-step-trained model.

Backup: heads develop different solver requirements



- Example: layer 16 of a medium MesaNet trained on 50B tokens.
- Weak forgetting can accompany growing condition numbers and more CG steps.
- Other heads maintain lower, nearly constant solve costs.

Implementation detail: identity query transformation gives GLA; zero CG steps depend on the chosen initialization.

Original Fig. 14, layer-16 row, PDF p. 52; averaged over 500 sequences. Internal diagnostic uses tolerance 10^{-3}. Initialization caveat: Appendix C, p. 24.