LEARNABLE MULTIPLIERS: FREEING THE SCALE OF LANGUAGE MODEL MATRIX LAYERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Language-model pretraining typically combines weight decay (WD) on parameter matrices with learnable scales, such as RMSNorm gains. Scales that can be absorbed into adjacent matrices add no representational capacity, yet learning them separately often improves training. We untangle the roles of WD and separately learned scales through a single mechanism: optimizer updates can drive unchecked norm growth, WD restrains this growth but holds matrix norms near potentially suboptimal values, and separately learned scales let the effective matrix scales adapt to the data. We demonstrate this mechanism both empirically, through controlled experiments with Adam and Muon, and theoretically, using an Adam-trained Zipf model of the LM head in which second-moment normalization suppresses the signal from rare informative gradients. We introduce *learnable multipliers* (LRMs) to extend this scale adaptation systematically across model matrices. LRMs learn diverse feature scales across and within layers while remaining compatible with μP learning-rate transfer. Progressively adding scale degrees of freedom across model matrices reduces training loss. In long pretraining across architectures and optimizers, our full LRM configuration delivers substantial downstream gains comparable to the Adam-to-Muon switch.

1 INTRODUCTION

Language models commonly include dedicated parameters for learning feature scales. RMSNorm gains, for example, let the model adjust each feature's magnitude after normalization (Zhang & Sennrich, 2019). Learnable scales also appear on attention heads and projection matrices (Shleifer et al., 2021; Jordan & contributors, 2024).

Many such scales are adjacent to a linear layer, and therefore can be absorbed into its weight matrix W. This results in learning the linear layer through its effective matrix $\overline{W}$, with entries

$$\overline{W}_{ij} = sW_{ij} \qquad \text{or} \qquad \overline{W}_{ij} = r_i W_{ij} c_j, \tag{1}$$

by optimizing the bare matrix W and either a scalar s or row and/or column scales r_i, c_j. We call these *learnable multipliers* (LRMs); RMSNorm gains before a projection are column multipliers.

The representation (1) exposes a redundancy: every effective matrix can also be learned directly, without separate scales. Yet learning scales separately can improve training. This situation is clarified through weight decay (WD) (Loshchilov & Hutter, 2019). WD has proved essential for training language model matrices with both Adam (D'Angelo et al., 2024) and Muon (Liu et al., 2025), while normalization gains are typically exempt from WD (EleutherAI, 2025).

Applying WD to language model matrices leads to a peculiar norm behavior (Kosson et al., 2024; Zuo et al., 2025): their norms typically settle near a value mostly determined by learning rate η and weight decay λ, with scaling

$$\|W\| \propto \sqrt{\frac{\eta}{\lambda}}. \tag{2}$$

The scaling (2) also arises in noise-only dynamics with WD, $W_{t+1} = W_t + \eta\xi_t - \eta\lambda W_t$, $\xi_t \sim \mathcal{N}(0, \sigma^2 I)$, without any signal term (Kosson et al., 2024; Zuo et al., 2025). We therefore call (2) a *noise-WD equilibrium*; it hints at suppression of the signal controlling matrix scale, while the underlying mechanism remains open.

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

We view these observations as a three-step mechanism connecting WD and separately learned scales:

O1. Without WD, matrix norms can keep growing even as the training objective worsens.

O2. WD constrains growth but may hold norms near an equilibrium suboptimal for the task.

O3. Separately learned scales compensate for these norms, allowing the effective matrices to adapt their scales to the data. In our controlled experiments, these scale parameters can be learned without WD and do not exhibit the same harmful growth.

For scalar multipliers, the optimized matrix norm, $\|W\|$, and the effective norm, $\|\overline{W}\| = |s|\|W\|$, can therefore behave differently. WD can constrain the former while the multiplier adapts the latter. Row and column multipliers extend this freedom to individual features. When the learned scales reduce the training objective, they improve performance on the training distribution without adding representational capacity. The harder question concerns the asymmetry: why can matrix updates fail to correct harmful norm growth, while updates to separate scales succeed?

Our contribution. In Section 2, we use the LM head as a case study of observations O1–O3. We combine controlled experiments with theoretical analysis of a simple Zipf model to explain how matrix scale learning can fail while separate scales retain a useful learning signal. The case study also motivates a broader investigation of optimizer pathologies.

We then study what the additional scale freedom enables. Section 3 shows that LRMs increase the diversity of feature scales across and within layers, and examines learned width scaling and compatibility with μP learning-rate transfer. In Section 4, we conduct a comprehensive LRM placement ablation to measure the benefit of progressively adding scale freedom. Our full LRM configuration improves long-pretraining loss and downstream performance across architectures and optimizers, including dense models up to 3B parameters trained on 240B tokens.

Concurrent works (Wang et al., 2026; Lion et al., 2026; Hägele et al., 2026) also benefit from separate scales, while leaving aspects of this learning asymmetry open. We discuss their explanations and other related work in Appendix A. We also provide practical aspects of using LRMs in Appendix E.

2 LM HEAD CASE STUDY OF SCALE LEARNING

2.1 NORM GROWTH AND SCALE COMPENSATION

We begin with the LM head, where freezing the backbone lets us study the output layer independently of changes in its input features. The head maps normalized features $\mathbf{x}$, $\mathrm{mean}\{x_j^2\} = 1$, to next-token logits $z_i = \sum_j W_{ij} c_j x_j$, where W is the head matrix and c_j are final RMSNorm gains.

Harmful norm growth and weight decay. To expose norm growth without compensation from separate scales, we reset the LM head of a pretrained model to zero and fix the final RMSNorm gains to one, retaining normalization. We train with Adam in four configurations: zero or small head WD, $\lambda \approx 6 \times 10^{-3}$, with a frozen or trainable backbone. With a frozen backbone, head training is convex, excluding nonlinear interactions with backbone learning. We deliberately use a high learning rate, $\eta \approx 4 \times 10^{-2}$, to amplify the effects. Figure 1 (left) shows the resulting loss and norm trajectories.

With a frozen backbone and no WD, we observe approximately linear growth of the head norm while the loss steadily worsens (O1). The optimizer thus continues increasing the norm instead of correcting a harmful scale. This raises the question of whether Adam's processing of stochastic gradients suppresses or reverses the signal that should correct it. Adding WD bounds the norm and prevents the loss deterioration. Unfreezing the backbone also prevents this deterioration despite continued norm growth, consistent with its features adapting to compensate. Such adaptation could come at the expense of useful representations: in ordinary training, the cost may appear as poorer performance rather than a visibly increasing loss.

WD-constrained norms and scale compensation. We next test observations O2–O3 during from-scratch pretraining. We vary only the head LR and WD, changing the predicted equilibrium norm scale $\sqrt{\eta/\lambda}$ while keeping $\eta_{\text{eff}} = \sqrt{\eta\lambda}$ fixed to control the approximate relative update scale (Zuo

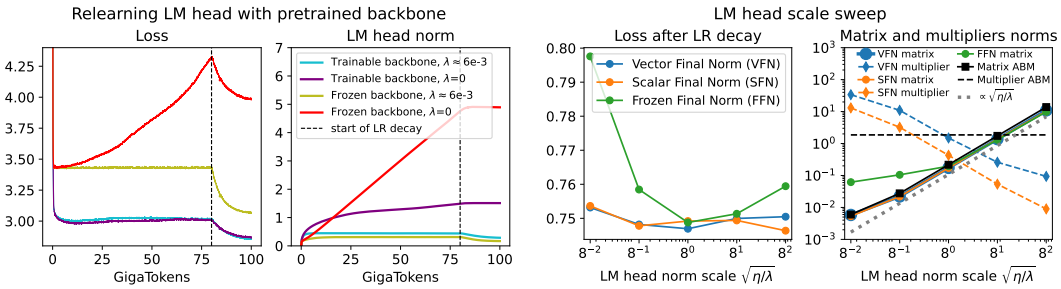


Figure 1: *Left pair:* loss and head RMS norm during continued training with the LM head reset to zero and final RMSNorm gains fixed to one. *Right pair:* loss after LR decay and matrix and multiplier RMS norms before decay, for from-scratch pretraining with fixed, scalar, or vector final RMSNorm gains. The norm scale $\sqrt{\eta/\lambda}$ uses LR and WD relative to a tuned baseline. Figure 17 compares model and Adam Brownian Motion (ABM) norm trajectories; Appendix B details the experiments.

et al., 2025). We compare three final RMSNorm configurations: fixed unit gains, $c_j = 1$ (FFN); a single learned scalar, $c_j = s$ (SFN); and learned per-feature gains, c_j (VFN). The learned gains receive no WD. Figure 1 (right) shows loss after LR decay and matrix and multiplier norms before it.

We find that matrix norms largely follow the predicted equilibrium scaling. Without learnable gains, loss worsens at both extremes (O2). Strong WD shrinks the head excessively; the upward departure from the noise-only norm prediction indicates some resistance to this shrinkage. The weak-WD side is less obvious: relaxing decay still does not let the matrix recover a useful scale. Its norm remains close to the equilibrium prediction even as loss worsens. Scalar and vector multipliers learn smaller gains for larger matrix norms, keeping loss much more stable across the sweep (O3).

To examine the contrast between matrix and gain learning, we compare both with Adam driven by independent zero-mean Gaussian gradients, using the corresponding learning-rate and WD settings. We call this noise-only reference *Adam Brownian Motion* (ABM). The matrix norms closely track ABM, especially when learnable gains are present, suggesting that their scale is largely set by noise and decay rather than by the task. The zero-WD gain reference instead undergoes unconstrained diffusion. Learned gains depart from this noise-only behavior and adapt to the matrix scale without WD. This returns us to our central question: why does the task-directed signal regulate separate gains more effectively than matrix scale? We investigate this through a simplified Zipf model.

2.2 Toy model: Adam learning a Zipf distribution

The harmful norm growth in Fig. 1 (left) occurs even with a convex loss, suggesting an interaction of Adam's normalized update, the softmax nonlinearity, and language statistics. A simple model retaining all three learns a Zipf distribution $p_i \propto i^{-\nu}$ by minimizing cross-entropy. Trainable logits z_i give probabilities $q_i(\mathbf{z}) = e^{z_i}/\sum_j e^{z_j}$. Each batch of size B contains $N_{i,k} \sim \mathrm{Binomial}(B, p_i)$ occurrences of token i, giving negative gradient $g_{i,k} = N_{i,k}/B - q_i(\mathbf{z}_k)$ and AdamW updates

$$z_{i,k+1} = z_{i,k} + \eta \frac{M_{i,k}}{\sqrt{V_{i,k}}} - \eta\lambda z_{i,k}, \qquad \begin{pmatrix} M_{i,k} \\ V_{i,k} \end{pmatrix} = (1-\beta)\sum_{j\geq 0} \beta^j \begin{pmatrix} g_{i,k-j} \\ g_{i,k-j}^2 \end{pmatrix}, \tag{3}$$

where we set $\beta_1 = \beta_2 = \beta \in (0,1)$ and $\epsilon = 0$. Dropping i and writing $R_k = M_k/\sqrt{V_k}$, for small learning rate η we approximate the dynamics by the SDE

$$\mathrm{d}z_t = [b(q(z_t)) - \lambda z_t]\,\mathrm{d}t + \sqrt{\eta\Gamma(q(z_t))}\,\mathrm{d}\xi_t, \quad b(q) = \mathbb{E}R_k, \quad \Gamma(q) = \sum_{h\in\mathbb{Z}} \mathrm{Cov}(R_0, R_h). \tag{4}$$

Here $t = \eta k$, ξ_t is Brownian motion, and the other logits form a fixed background. The drift b describes the mean optimizer response, and the integrated diffusion coefficient Γ characterizes accumulated update noise, including temporal correlations of R_k. For $b = 0$ and constant Γ, the SDE gives the noise–WD equilibrium (2), with stationary $\mathbb{E}z^2 = \eta\Gamma/(2\lambda)$.

We find three distinct types of logit dynamics, organized by Bp_i, the expected number of occurrences of token i per batch (Fig. 2). To understand these regimes, we analyze b and Γ at fixed logits with stationary optimizer moments, rather than solving the full dynamics.

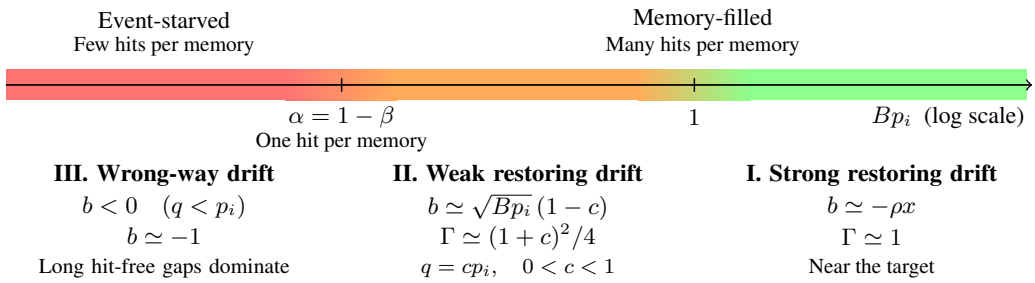


Figure 2: Local Adam responses for rare tokens along a schematic logarithmic Bp_i axis. The markers denote one occurrence per memory window (α) and per batch (1), rather than sharp boundaries. Regions I–II show leading long-memory coefficients, with $\rho = \sqrt{Bp_i(1-p_i)}$ and $x = z - z_i^*$. Region III requires sufficiently separated hits, as in Theorem 2.1.

Writing $\alpha = 1 - \beta$, Adam's exponential averaging remembers roughly $1/\alpha$ batches, containing Bp_i/α occurrences of token i on average. We call the regime *memory-filled* when $Bp_i \gg \alpha$, so many occurrences contribute within memory, and *event-starved* when $Bp_i \ll \alpha$, so a typical memory window contains none. We now examine how this distinction affects normalization and the resulting drift and diffusion, with derivations in Appendix I.

Regions I–II: strong and weak restoring drift. With many occurrences in memory, V concentrates near its expectation, so Adam's denominator varies little between updates. At fixed predicted probability q, $\mathbb{E}M = p_i - q$ and $V \simeq \mathbb{E}V = (q-p_i)^2 + p_i(1-p_i)/B$. Expanding the Adam ratio around its mean moments then gives

$$b(q) \simeq \frac{p_i - q}{\sqrt{(q-p_i)^2 + p_i(1-p_i)/B}}. \tag{5}$$

For $Bp_i(1-p_i) \gg 1$ (Region I), we obtain strong restoring drift $b \simeq -\rho x$ and noise $\Gamma \simeq 1$ near the target $q_i(z_i^*) = p_i$, with $\rho = \sqrt{Bp_i(1-p_i)}$ and $x = z - z_i^*$. For $\alpha \ll Bp_i \ll 1$ (Region II), however, the underprediction response is bounded by $O(\sqrt{Bp_i})$, while Γ remains order one. Figure 2 gives both coefficients; computing Γ requires fluctuations and correlations of M and V. For SGD, both coefficients decrease proportionally to p_i; Adam retains order-one diffusion as restoring drift weakens. For overpredicted tokens, the response strengthens once $q - p_i$ is comparable to $\sqrt{p_i/B}$.

Region III: wrong-way drift. In the event-starved regime, normalization instead depends strongly on the time since the last occurrence. For an underpredicted token ($q_i < p_i$, $Bp_i \ll 1$), most batches give the small negative gradient $-q_i$, while an occasional hit gives $g_i \simeq 1/B$ and sharply raises V. Between hits, this contribution decays. During a sufficiently long hit-free interval, the earlier positive hits fade from both moments, leaving $M \simeq -q_i$ and $V \simeq q_i^2$. Adam then turns the small negative gradients into updates near -1, even though $q_i < p_i$ calls for an increase. When such intervals dominate, the mean drift reverses, $b < 0$. The separation condition in Theorem 2.1 accounts for the time needed to forget a large hit, beyond the nominal memory length $1/\alpha$.

From local responses to the observations. In the diffusion picture, Region I keeps logits near their targets, whereas Region II permits broad excursions below them, producing one-sided fluctuations. Region III can drive logits downward even against the population gradient. These behaviors appear in Fig. 3 (left) and explain how scale regulation can fail (O1). WD restrains such excursions but can set the scale in place of the weakened task signal (O2). Where noise dominates, its balance with decay gives the noise–WD scale $\sqrt{\eta\Gamma/\lambda}$; mean drift can also shift this balance (Appendix I.5).

Why a separate scale can learn without WD. To explain O3 in this model, we multiply fixed Zipf logits $z_i = \log p_i$ by a scalar s. Then $q_i(s) \propto p_i^s$, with optimum $s = 1$ and negative batch gradient

$$\widehat{g}_s = \frac{1}{B}\sum_{b=1}^{B} \log p_{Y_b} - \sum_i q_i(s) \log p_i = -L'(s) + \xi. \tag{6}$$

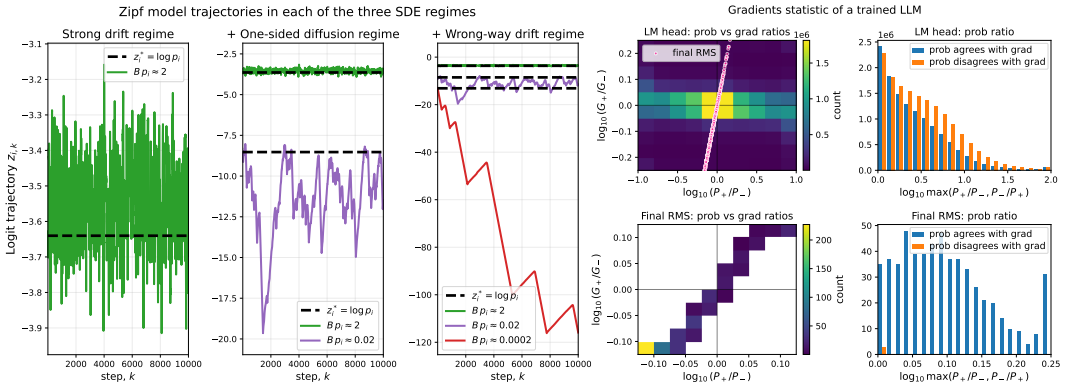


Figure 3: **Left:** simulated log-probability trajectories illustrating strong restoring drift, one-sided fluctuations, and persistent downward drift (left to right). We interpret these behaviors using the local responses in Fig. 2; Appendix I.7 gives the simulation settings and their relation to the analysis. **Right:** LLM gradient statistics (§2.3): LM-head entries (top) and final-RMSNorm gains (bottom). Heatmaps show sign-probability and signed-mass ratios, $(\log_{10}(P_+/P_-), \log_{10}\chi)$; histograms show $\max(P_+/P_-, P_-/P_+)$, split by whether the rare sign agrees with the population gradient.

Every sampled token contributes a log-probability, rather than an indicator for one particular rare token. For $\nu > 1$, the centered noise ξ has bounded standardized moments as vocabulary grows, and its distribution does not depend on s. Meanwhile, $L''(s) = \mathrm{Var}_{q(s)}(\log p) > 0$ supplies a restoring population gradient. We prove that the mean Adam update retains this restoring structure: it points toward a stable scale $s_\beta \to 1$ as $\beta \to 1$, without WD (Appendix I.8).

The contrast also holds at the same over-sharpened state $s > 1$: for large vocabularies, coordinate Adam can push the centered logit norm outward against the population gradient, while a small scalar step with sufficiently long memory decreases squared norm and loss in expectation (Theorem I.5). Separately learning the scale thus recovers a task-directed signal that coordinate updates can miss.

2.3 Adam under general rare-event gradients

The suppressed and reversed drift in Regions II–III are not specific to the Zipf model. For a fixed parameter entry, let g be its negative batch gradient and define

$$P_\pm = \mathbb{P}(\pm g > 0), \qquad G_\pm = \mathbb{E}[(\pm g)_+], \qquad \chi = G_+/G_-. \tag{7}$$

The informative sign is rare when $P_+ \ll 1$ but $\chi > 1$: rare positive gradients outweigh the frequent negative ones in the population mean. For underpredicted rare Zipf tokens, the identifications $P_+ \simeq Bp_i$ and $\chi \simeq p_i/q_i$ map Regions III and II to the two cases below, showing how both behaviors follow from more general gradient statistics.

Theorem 2.1 (Adam under rare-event gradients, informal; App. J)**.** *For Adam with equal momenta $\beta = 1 - \alpha$, consider the stationary response to rare informative gradients at fixed parameters. When $\frac{P_+}{\alpha}\log(1 + \chi^2/P_+)$ is sufficiently small, the mean update reverses the population descent direction: $b < 0 < \mathbb{E}g$. When α/P_+ is sufficiently small, the mean remains correctly directed but is suppressed relative to accumulated noise: $0 < b \lesssim \sqrt{\Gamma P_+}$, with $\Gamma = \Theta(1)$.*

We find rare-sign asymmetry in LM-head entries but more balanced signs in final-RMSNorm gains (Fig. 3, right). Appendix J.5 describes additional diagnostics for testing this connection.

2.4 Optimizer pathologies beyond the Zipf model

In the Zipf model, Adam's nonlinear normalization suppresses or reverses the restoring signal from rare gradients, providing a mechanism for unchecked norm growth (O1). We find related scale-learning failures in MLP matrices with both Adam and Muon: in an experiment analogous to the LM-head norm-scale sweep, matrix norms largely follow noise-WD scaling, large scales degrade

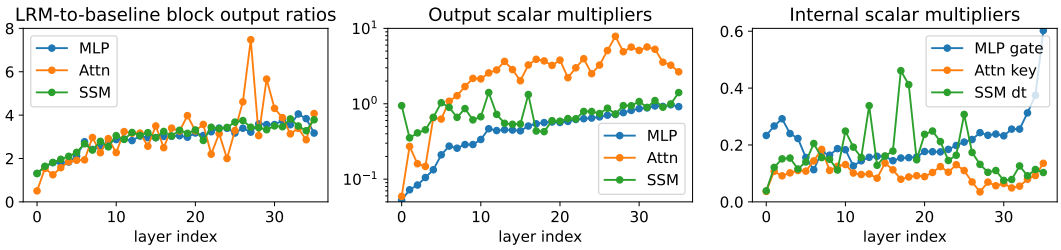


Figure 4: *(Left):* Ratio of residual-block output norms with scalar multipliers to those without. *(Middle):* Learned scalar multipliers on block outputs. *(Right):* Learned scalar multipliers at selected internal projections.

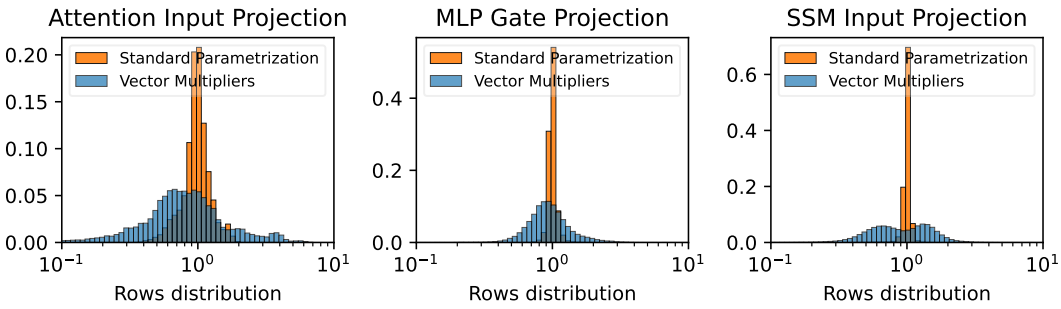


Figure 5: Distributions of row norms $\|\overline{W}_{i\bullet}\|$ of effective attention/SSM input and MLP gate matrices, corresponding to internal features of these blocks. Without multipliers (orange), the distributions are narrow; vector multipliers (blue) substantially broaden them.

performance, and separate multipliers compensate (Appendix G, Fig. 15). Yet our backbone gradient measurements lack the LM head's rare-sign asymmetry (Fig. 18). The mechanism remains open.

The Zipf analysis suggests a way to investigate such *optimizer pathologies*: examine how the nonlinear update rule interacts with gradient averaging. At fixed parameters W, let $G_1, \ldots, G_t$ be independent negative minibatch gradients. We compare the two orders of these operations through

$$\Delta_t(W) = \mathbb{E}[A_t(G_1, \ldots, G_t)] - A_t(\mathbb{E}[G_1], \ldots, \mathbb{E}[G_t]), \tag{8}$$

where A_t is the additive optimizer update before WD, and both histories start from the same optimizer state. The first term averages the response to stochastic gradients; the second processes their population mean. The question is whether this difference suppresses or reverses a useful response, such as correction of matrix scale. Together with the integrated update covariance, this comparison offers a tractable way to investigate possible optimizer pathologies in other matrices and optimizers.

3 What is learned by the multipliers

Having examined why separate scales can behave differently from matrix entries, we now characterize the structure of what they learn. We organize the discussion along three axes: variation across layers (depth-wise), variation within a layer (row/column scales), and variation with model width. For the latter, we additionally test whether learnable scale adaptation is compatible with μP learning-rate transfer.

3.1 Depth-wise scales

We add scalar multipliers to all matrix layers and examine how the resulting scales vary across model depth (Figure 4).

The block-output ratios exhibit a clear depth-dependent structure, with larger relative rescaling toward later layers. Attention outputs in the second half of the model receive particularly strong rescaling (Figure 4, left and middle). The learned scales also vary substantially across internal projections: for

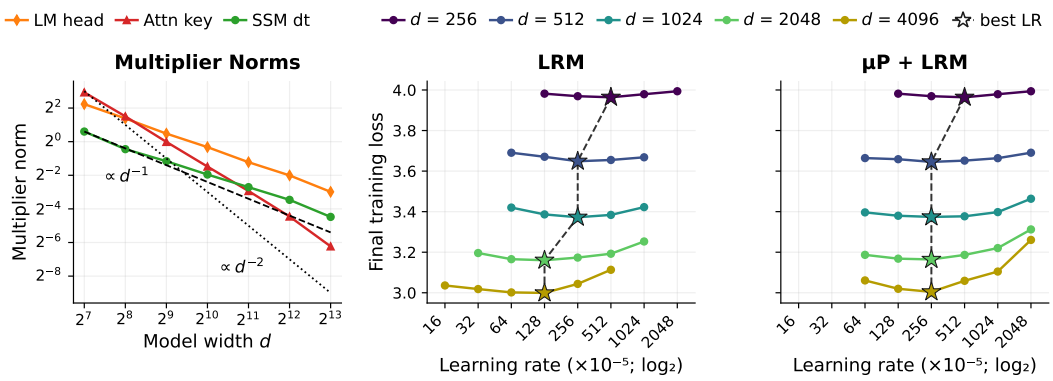


Figure 6: **Learned width scaling and learning-rate transfer.** *Left:* learned scalar multipliers across model widths, together with the corresponding μP reference trends. *Center:* global learning-rate sweep with naively parametrized LRMs. *Right:* the same sweep after applying μP width scaling to the LRM initialization and optimizer learning rate. Stars mark the best tested learning rate at each width. Additional coordinate checks, parametrizations, and experimental details are provided in Appendix F.

example, the multiplier on the SSM time-step (`dt`) projection changes considerably across layers. Thus, the learned scales are neither shared across depth nor reducible to a single global correction. Appendix H.4 describes the `dt` projection.

3.2 Within-layer scale diversity

We next examine scale diversity within individual layers by measuring row norms of the effective matrices $\overline{W}$ with and without vector LRMs (Figure 5). We consider attention/SSM input and MLP gate projections, whose outputs feed scale-sensitive nonlinearities. Without LRMs, the row norms are tightly concentrated, consistent with a common noise–WD scale. Vector LRMs substantially broaden these distributions, allowing individual output features to acquire distinct learned scales.

3.3 Width scaling and μP learning-rate transfer

In μP, width-dependent scaling can be redistributed across initialization, learning rates, and fixed forward multipliers under equivalent parametrizations (Yang & Hu, 2021; Yang et al., 2022; Yang & Littwin, 2023). In the convention used here, the relevant forward scaling is carried by fixed multipliers (Zuo et al., 2025). LRMs occupy the same position in the forward computation, but make these scales learnable. This raises two distinct questions: (i) can LRMs learn the width-dependent forward scales prescribed by μP, and (ii) can the resulting learnable parametrization preserve learning-rate transfer across widths?

Learned forward scaling. We first train LRMs without imposing any μP width dependence on their initialization or learning rate. Figure 6 (left) shows that the learned multipliers nevertheless acquire width-dependent scales close to the corresponding μP prescriptions: approximately d^{-1} for the LM head and SSM `dt`, and d^{-2} for attention `QK`. The empirical slopes are slightly shallower than these reference trends; we discuss this discrepancy and the corresponding coordinate checks in Appendix F.

Learning-rate transfer. Learning a μP-like endpoint scale does not necessarily imply that the optimization dynamics themselves transfer across width. We therefore test learning-rate transfer directly by sweeping the global learning rate across model widths. Figure 6 (center) shows that with naively parametrized LRMs, the optimal learning rate changes with width despite the μP-like endpoint scaling above.

We therefore apply the μP width-scaling rules directly to the LRM coordinates, while leaving the underlying matrix parametrization unchanged. Specifically, the prescribed width factor is applied

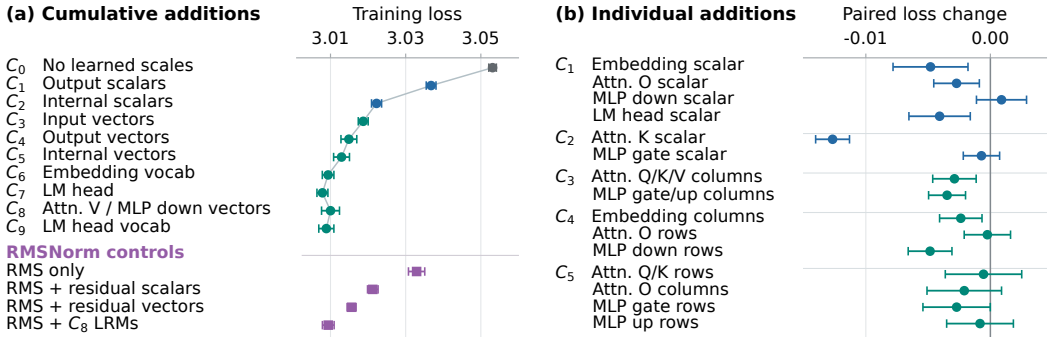


Figure 7: *(Left)* Cumulative LRM placements, starting from C_0 with no learned scales. We add output and internal scalars (C_1–C_2), then input, output, and internal vectors (C_3–C_5), replacing existing scalars. Next come embedding-vocabulary rows (C_6), LM-head columns (C_7), redundant attention-V rows and MLP-down columns (C_8), and LM-head vocabulary rows (C_9). Four learned-RMSNorm controls appear below; gains are frozen at one in all other configurations. *(Right)* Individual additions from each group C_n, $n = 1, \ldots, 5$, are tested against C_{n-1}; for example, the attention-K scalar is added alone to C_1. Some additions affect several matrices together. Every configuration is trained independently from initialization. Points show means with SE bars over 20 joint initialization/data-order seeds; the right panel shows paired child-minus-parent loss differences. See App. H.3 for placements and App. C.1 for training and uncertainty estimates.

both to the LRM initialization and to its optimizer learning rate. For row–column LRMs, we split this factor equally between the two coordinates: for example, a d^{-1} forward-scaling factor becomes $d^{-1/2}$ for each row and column multiplier, both at initialization and in their learning-rate scaling. Thus the matrices retain the same initialization and optimization rules; only the added LRM coordinates are μP-scaled.

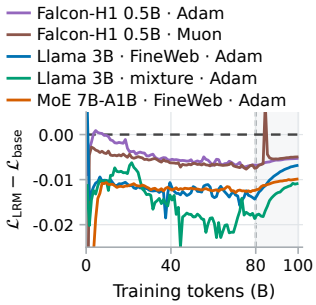
Figure 6 (right) shows that the resulting μP+LRM parametrization substantially aligns the preferred learning rates across the tested widths while retaining learnable forward scales. LRMs and μP therefore play complementary roles: LRMs learn the effective forward scales from data, whereas μP specifies how the learnable scale coordinates should be parametrized for their optimization dynamics to transfer across width.

4 Impact on Pretraining Performance

Multiplier placement. LRMs (1) can be added to any model's matrix, in either scalar or vector form (row, column, or row+column). As each of these multipliers may unlock further scale degrees of freedom, we clarify the respective impacts on the training loss in a comprehensive LRM placement ablation on Fig. 7. First, we find that adding scalar degrees of freedom has the biggest impact; further upgrading to vector LRMs also improves the loss but with diminishing returns. Figure 7 (right) reveals that each individual multiplier contributes slightly to the loss improvement, except the first scalar LRM on attention key that yields massive improvement by itself. With learned RMSNorm gains retained, C_8 also improves training loss over RMSNorm alone and over RMSNorm with residual scalar or vector multipliers. We recommend row+column LRMs across the model's matrices, except for LM-head vocabulary multipliers; App. E.5 gives the full recipe and explains this exception.

Long pretraining. Finally, we evaluate the impact of LRMs on the downstream performance of language models across five data, architecture, and optimizer choices:

- `Falcon-H1-0.5B` trained with Adam or Muon on a proprietary data mixture.
- `Llama3.2-3B` trained with Adam on Fineweb Penedo et al. (2024) or diverse math-heavy mixture of open source datasets (see App. B.2 for the full mixture details).
- `7B-A1B MoE` based on `Llama3.2` architecture trained with Adam on Fineweb.



Model / optimizer	Hella Swag	ARC-C	MMLU	MMLU Pro	BBH	GSM8K	MATH lvl 5	IFEval	Avg.
Falcon-H1 0.5B Adam	48.91	38.70	44.18	18.26	9.70	48.35	7.52		30.80
	↓+0.98	↓+0.03	↓+1.15	↓+1.66	↓+2.33	↓+0.75	↓+1.55	—	↓+1.21
	49.89	38.73	45.33	19.92	12.03	49.10	9.07		32.01
Falcon-H1 0.5B Muon	50.31	39.04	47.98	18.96	10.13	48.02	8.69		31.88
	↓+0.25	↓+0.35	↓-0.02	↓+0.36	↓+3.39	↓+2.61	↓+0.80	—	↓+1.11
	50.56	39.39	47.96	19.32	13.52	50.63	9.49		32.98
Llama 3B mixture Adam	64.12	49.42	56.74	23.33	18.75	68.85	29.38	62.03	46.58
	↓+0.65	↓-0.39	↓+0.71	↓+2.44	↓-0.01	↓+0.05	↓+1.18	↓+1.74	↓+0.79
	64.77	49.03	57.45	25.77	18.75	68.89	30.55	63.76	47.37

Figure 8: **Left:** LRM-minus-baseline training loss in 100B-token runs, averaged in 1B-token windows. We use the original 100B branches, since later restarts disrupt matched loss comparisons. Shading marks learning-rate decay over 80–100B tokens. **Right:** Benchmark scores (%) averaged over 40 checkpoints from dedicated post-decay training through approximately 200–240B tokens at minimal LR. Each cell shows baseline above LRM, with the change in percentage points between them (green/red: gain/loss). Avg. is the unweighted mean of reported benchmarks (seven for Falcon-H1, eight for Llama); dashes denote unreported IFEval. Full results, including FineWeb, and checkpoint standard deviations appear in App. B.3.

For each setting, we train a baseline–LRM pair for 200B tokens, followed by 40B at minimum LR for evaluation. App. B.1 specifies the recipes; App. E.5 gives our recommended default.

First, we look at the loss advantage of LRM configurations in Figure 8 (left), and observe consistently widening gap, suggesting that LRM improvements does not come from mere convergence acceleration. The reduction of the gap during learning rate decay is consistent with weakening of noise-WD equilibrium that LRMs are meant to correct. We also note that the loss gap scales favorably to larger models within our comparison.

Second, we evaluate models on a suite of standard pretraining benchmarks, with results depicted on Figure 8 (right). We observe overall improvement from LRMs in the considered settings, though slightly uneven across tasks: the math and reasoning focused benchmarks (`MMLU-PRO`, `BBH`, `GSM8K`, `MATH lvl5`, `IFEval`) benefit from LRMs more than knowledge-based benchmarks (`ARC-C` and `MMLU`). The pair of `Falon-H1-0.5B` experiments on Adam and Muon also suggests that the LRM improvement is of the same order as the improvement from Adam to Muon.

5 Conclusion and Discussion

We have connected the roles of weight decay and separately learned scales through a single account of scale learning. Weight decay restrains matrix-norm growth, but can leave norms poorly matched to the task; separate scales allow effective matrices to adapt beyond this constraint. Our Zipf case study explains this learning asymmetry in a simple setting: Adam can suppress or reverse the restoring signal from rare gradients, while a separate multiplier retains a useful scale-correcting response.

This view motivates LRMs as a systematic extension of scale adaptation already present in language-model architectures. LRMs learn more diverse feature scales across and within layers, remain compatible with appropriately parametrized μP learning-rate transfer, and improve long-pretraining loss and downstream performance across architectures and optimizers. These results demonstrate the practical value of learning scales separately, even when they can be absorbed into the matrix weights without changing the represented function. Hence, we believe that separately learned scales has the potential to mature into a part of state-of-the-art LLM training recipes.

The mechanism behind related scale-learning failures in backbone matrices remains open. Our analysis suggests investigating *optimizer pathologies* – how gradient statistics interact with the optimizer's update rule to suppress useful learning signals. A broader question is whether other properties of matrices, beyond their scales, are similarly difficult to learn and could benefit from targeted reparameterization.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=hmOwOZWzYE.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model, 2025. URL https://arxiv.org/abs/2502.02737.

Alexander Atanasov, Alexandru Meterez, James B Simon, and Cengiz Pehlevan. The optimization landscape of SGD across the feature learning strength. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=iEfdvDTcZg.

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. ReZero is all you need: Fast convergence at large depth. In *Conference on Uncertainty in Artificial Intelligence*, 2021.

Lukas Balles and Philipp Hennig. Dissecting Adam: The sign, magnitude and variance of stochastic gradients. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, 2018. URL https://proceedings.mlr.press/v80/balles18a.html.

Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/ae614c557843b1df326cb29c57225459-Paper.pdf.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022. URL https://arxiv.org/abs/2204.02311.

Francesco D'Angelo, Maksym Andriushchenko, Aditya Varre, and Nicolas Flammarion. Why do we need weight decay in modern deep learning? In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 23191–23223. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/29496c942ed6e08ecc469f4521ebfff0-Paper-Conference.pdf.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality, 2024. URL https://arxiv.org/abs/2405.21060.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348v1.

Nolan Dey, Gurpreet Gosal, Zhiming, Chen, Hemant Khachane, William Marshall, Ribhu Pathria, Marvin Tom, and Joel Hestness. Cerebras-gpt: Open compute-optimal language models trained on the cerebras wafer-scale cluster, 2023. URL https://arxiv.org/abs/2304.03208.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *ICML*, 2017.

EleutherAI. GPT-NeoX: Training implementation, 2025. URL https://github.com/EleutherAI/gpt-neox/blob/d12c771198388980ee054617e537665f044e0584/megatron/model/utils.py. Weight-decay parameter groups; commit d12c771, lines 30–61.

Google DeepMind. Gemma: Official model implementations. Official code at commit 0513283, 2026. Gemma 3n and Gemma 4 implementations.

Alexander Hägele, Alejandro Hernández-Cano, Atli Kosson, and Martin Jaggi. Improving neural network training by decoupling the magnitude and direction of weight vectors, 2026. URL https://arxiv.org/abs/2606.25971v2. Version 2.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm: Unveiling the potential of small language models with scalable training strategies, 2024. URL https://arxiv.org/abs/2404.06395.

Florian Hübler, Kai Lion, Antonio Orvieto, and Niao He. Muown implicitly performs angular step-size decay, 2026. URL https://arxiv.org/abs/2606.23637v1. Version 1.

Keller Jordan and contributors. modded-nanogpt: Speedrunning the NanoGPT baseline. https://github.com/KellerJordan/modded-nanogpt, 2024. GitHub repository.

Keller Jordan and contributors. modded-nanogpt: Training recipe with projection and branch multipliers, 2026. URL https://github.com/KellerJordan/modded-nanogpt/blob/bc3a0c2d640d0d73dedaef87eae26148d2e32afb/train_gpt.py. Pinned implementation, commit bc3a0c2; accessed September 2026.

Kimi Team. Kimi K3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653v1.

Atli Kosson, Bettina Messmer, and Martin Jaggi. Rotational equilibrium: How weight decay balances learning across neural networks. In *International Conference on Machine Learning*, 2024.

Atli Kosson, Jeremy Welborn, Yang Liu, Martin Jaggi, and Xi Chen. Weight decay may matter more than mup for learning rate transfer in practice, 2025. URL https://arxiv.org/abs/2510.19093.

Teven Le Scao, Angela Fan, et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv:2211.05100*, 2022.

Kai Lion, Florian Hübler, Bingcong Li, Antonio Orvieto, and Niao He. Muown: Row-norm control for Muon optimization, 2026. URL https://arxiv.org/abs/2605.10797v1. Version 1.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for llm training, 2025. URL https://arxiv.org/abs/2502.16982.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation, 2024. URL https://arxiv.org/abs/2402.09353.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. nGPT: Normalized transformer with representation learning on the hypersphere, 2024. URL https://arxiv.org/abs/2410.01131v1. Version 1.

Paulius Micikevicius, Sharan Narang, Jonah Alben, et al. Mixed precision training. *arXiv:1710.03740*, 2017.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 1310–1318. PMLR, 2013.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale, 2024. URL https://arxiv.org/abs/2406.17557.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free, 2025. URL https://arxiv.org/abs/2505.06708v1.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1904.09237.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks, 2016. URL https://arxiv.org/abs/1602.07868.

Jie Shao, Kai Hu, Changhu Wang, Xiangyang Xue, and Bhiksha Raj. Is normalization indispensable for training deep neural networks? In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/9b8619251a19057cff70779273e95aa6-Paper.pdf.

Sam Shleifer, Jason Weston, and Myle Ott. Normformer: Improved transformer pretraining with extra normalization, 2021. URL https://arxiv.org/abs/2110.09456.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *International Conference on Computer Vision*, 2021.

Twan van Laarhoven. L2 regularization versus batch and weight normalization, 2017. URL https://arxiv.org/abs/1706.05350.

Mingze Wang, Shuchen Zhu, Yuxin Fang, Binghui Li, Kai Shen, and Shu Zhong. Negligible in size, significant in effect: On scale vectors in large language models, 2026. URL https://arxiv.org/abs/2605.26895v2. Version 2.

Sifan Wang, Hanwen Wang, Jacob H. Seidman, and Paris Perdikaris. Random weight factorization improves the training of continuous neural representations, 2022. URL https://arxiv.org/abs/2210.01274v2. Version 2.

Greg Yang and Edward J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11727–11737. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/yang21c.html.

Greg Yang and Etai Littwin. Tensor programs ivb: Adaptive optimization in the infinite-width limit, 2023. URL https://arxiv.org/abs/2308.01814.

Greg Yang, Edward J. Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer, 2022. URL https://arxiv.org/abs/2203.03466.

Greg Yang, James B. Simon, and Jeremy Bernstein. A spectral condition for feature learning, 2024. URL https://arxiv.org/abs/2310.17813.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *NeurIPS*, 2019.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Residual learning without normalization via better initialization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1gsz30cKX.

Zhiming Zhou, Qingru Zhang, Guansong Lu, Hongwei Wang, Weinan Zhang, and Yong Yu. AdaShift: Decorrelation and convergence of adaptive learning rate methods. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.00143v4.

Shuchen Zhu, Yuxin Fang, Mingze Wang, and Kun Yuan. Curvature-conditioned multiscale momentum with sphere constraints for LLM pretraining, 2026. URL https://arxiv.org/abs/2608.28442v1. Version 1.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, Mugariya Farooq, Giulia Campesan, Ruxandra Cojocaru, Yasser Djilali, Shi Hu, Iheb Chaabane, Puneesh Khanna, Mohamed El Amine Seddik, Ngoc Dung Huynh, Phuc Le Khac, Leen AlQadi, Billel Mokeddem, Mohamed Chami, Abdalgader Abubaker, Mikhail Lubinets, Kacper Piskorski, and Slim Frikha. Falcon-h1: A family of hybrid-head language models redefining efficiency and performance, 2025. URL https://arxiv.org/abs/2507.22448.

Zyphra, Quentin Anthony, Yury Tokpanov, Skyler Szot, Srivatsan Rajagopal, Praneeth Medepalli, Anna Golubeva, Vasu Shyam, Robert Washbourne, Rishi Iyer, Ansh Chaurasia, Tomas Figliolia, Xiao Yang, Abhinav Sarje, Drew Thorstensen, Amartey Pearson, Zack Grossbart, Jason van Patten, Emad Barsoum, Zhenyu Gu, Yao Fu, and Beren Millidge. Training foundation models on a full-stack AMD platform: Compute, networking, and system design (ZAYA1), 2025. URL https://arxiv.org/abs/2511.17127.

CONTENTS

1 Introduction **1**

2 LM head case study of scale learning **2**

2.1 Norm growth and scale compensation 2

2.2 Toy model: Adam learning a Zipf distribution 3

2.3 Adam under general rare-event gradients 5

2.4 Optimizer pathologies beyond the Zipf model 5

3 What is learned by the multipliers **6**

3.1 Depth-wise scales 6

3.2 Within-layer scale diversity 7

3.3 Width scaling and μP learning-rate transfer 7

4 Impact on pretraining performance **8**

5 Conclusion and discussion **9**

A Related work **16**

A.1 Learnable scales in architectures and matrix reparameterizations 16

A.2 Norm growth and the role of weight decay 16

A.3 Explanations for the benefits of separately learned scales 17

A.4 Gradient statistics and adaptive-optimizer failures 18

A.5 Learned scaling and hyperparameter transfer 19

B Experiment settings **19**

B.1 Long-pretraining configurations 20

B.2 Data mixture and preparation for Llama-3.2-3B 22

B.3 Full long-pretraining evaluations 23

C Additional pretraining ablations **25**

C.1 Multiplier placement ablation 25

C.2 Multiplier tuning ablation 26

D AI-use statement **26**

E Practical considerations introduced by learnable multipliers **27**

E.1 Symmetries 27

E.2 Gradient clipping 28

E.3 Exponential parametrization 29

E.4 Learning the LM head scale 29

E.5 Recommended LRM recipe 29

F Width scaling and μP details **30**

F.1 Forward-multiplier scaling under noise–WD equilibrium 30

F.2 Endpoint coordinate checks and learned forward scales 31

F.3 RMS-normalized amplification diagnostic for learned multiplier slopes 32

F.4 Learning-rate transfer setup 32

F.5 Symmetry-affected activation norms in the width sweep 34

G MLP experiment **34**

G.1 Setup and high-level result 35

G.2 Detailed setup 35

G.3 Behavior of parameter and output norms 36

H Architecture and multiplier placement **39**

H.1 Embedding and LM head 39

H.2 Attention and gated MLP blocks 39

H.3 Configurations in the placement ablation 40

H.4 Mamba-2 and the SSM time-step projection 41

I Zipf model: frozen responses and scale dynamics **42**

I.1 Setup and the exact frozen response 42

I.2 Integrated noise and the local diffusion approximation 43

I.3 Memory-filled response: drift and linearized noise 43

I.4 Region I: dense memory-filled response 44

I.5 Region II: rare memory-filled response 45

I.6 Region III: event-starved response 45

I.7 Relation to the plotted trajectories 46

I.8 Why a separate scale can respond differently 46

J Proof of Theorem 2.1 **50**

J.1 Wrong-way drift 50

J.2 Memory-filled gradients 51

J.3 The fixed-law high-momentum limit 55

J.4 Zipf interpretation 56

J.5 Diagnostic protocol 56

K Additional plots **56**

A Related work

Separate scale parameters are used both as architectural ingredients and as optimization coordinates. We review their uses and the explanations for their benefits, with particular attention to the connection in observations O1–O3: why matrix scales can resist task-directed correction, how separate scales restore useful freedom, and why their updates can behave differently from matrix updates.

A.1 Learnable scales in architectures and matrix reparameterizations

Separately learned scales have a long history. RMSNorm learns a gain for each feature after normalization (Zhang & Sennrich, 2019), while Weight Normalization explicitly separates the magnitude and direction of each weight vector (Salimans & Kingma, 2016). Residual architectures also use scalar or vector gains: ReZero initializes a scalar on each residual branch at zero, and LayerScale introduces small channelwise gains (Bachlechner et al., 2021; Touvron et al., 2021). The residual gains were introduced to stabilize signal propagation through deep networks.

Language-model recipes place scales at several additional locations. NormFormer learns one scalar per attention head before the output projection (Shleifer et al., 2021). nGPT combines normalized weights with learned intermediate and logit scales, among other scale parameters (Loshchilov et al., 2024). The modded-nanogpt training recipe uses explicit projection and residual-branch multipliers (Jordan & contributors, 2026). Qwen's gated-attention study also includes an input-independent attention-output gain: this ablation improves perplexity over its baseline, although the input-dependent gate performs better (Qiu et al., 2025).

Recent released architectures provide further examples. The official Gemma implementations include a channelwise scale before Gemma 3n's per-layer embedding gate projection and a scalar on each Gemma 4 expert's output (Google DeepMind, 2026). DeepSeek-V4's manifold-constrained hyper-connections use learned scalars on the matrices predicting the residual connections (DeepSeek-AI, 2026). Kimi K3 places learned per-head scales on decay logits before their sigmoid transformation (Kimi Team, 2026). ZAYA1 applies channelwise affine transformations to residual and sublayer-output streams (Zyphra et al., 2025). These examples establish architectural adoption; their full-model results generally do not isolate the contribution of learning the scales.

Magnitude–direction decompositions also appear in parameter-efficient finetuning, notably DoRA (Liu et al., 2024). There a low-rank perturbation of the frozen pretrained matrix is renormalized and combined with separately learned magnitudes. Separate magnitude parameters can enlarge the permitted adaptation family, unlike adding foldable gains to the fully trainable matrices studied here.

We focus on scales that can be absorbed into adjacent weight matrices without changing the forward function. This includes normalization gains before projections, attention-output gains, expert-output scalars, and the projection gains in DeepSeek-V4. Other controls, such as gains on accumulated residual streams or input-dependent gates, need not admit this reduction. Even when absorption is possible, it changes the optimization parameterization; for explicitly normalized methods such as nGPT, the merged matrix need not satisfy the original normalization constraint. These uses motivate our study of separate scale learning, but need not share one explanation for its benefit.

A.2 Norm growth and the role of weight decay

Weight decay can affect optimization even when changing a parameter's norm leaves the represented function unchanged. For an exactly scale-invariant weight vector, the gradient is tangential. A nonzero step of ordinary gradient descent therefore increases its norm, reducing subsequent directional updates. Controlling this radius controls an effective learning rate (van Laarhoven, 2017). This explains a benefit of norm control without requiring a task-preferred value of the raw norm.

Kosson et al. (2024) develop this perspective through *rotational equilibrium*. Their noise-dominated update model balances stochastic norm growth against WD and predicts both equilibrium norms and angular update sizes. In particular, their AdamW analysis gives the $\sqrt{\eta/\lambda}$ norm scaling used in our introduction. Their extension to scale-sensitive weights explicitly includes radial task signal as a contribution to effective decay; it does not assume that all radial gradients vanish. They also construct a fixed-norm rotational optimizer with optional, foldable learned gains, including in language-model experiments. This is an important precedent for combining norm control with separately learned

functional scales. The Falcon-H1 report likewise used stochastic norm balance to explain optimizer-dependent matrix norms (Zuo et al., 2025).

The question we pursue concerns the task signal underlying this balance. Where increasing scale raises the population loss, its radial derivative supplies a restoring signal. A noise-dominated approximation describes the resulting norms once this correction is weak. We ask why the optimizer can fail to use the available restoring signal even at a harmful scale. This distinguishes the phenomenon in observation O1 from growth of a redundant, exactly invariant radius.

Other benefits of WD can coexist with this mechanism. D'Angelo et al. (2024) show that matching the norm-dependent effective learning-rate schedule reproduces the WD loss trajectory in their full-precision GPT-2 experiments. They separately find a stability benefit from WD in reduced precision. These interventions provide an optimization account of WD in language models, and show why improved loss from WD alone does not establish a harmful forward-scale mismatch. The scale interventions in Section 2.1 examine this additional question.

A.3 Explanations for the benefits of separately learned scales

Conditioning and coordinated updates. Weight Normalization (Salimans & Kingma, 2016) separates a weight vector as $w = gu$, where $u = v/\|v\|_2$. The derivatives of the loss are

$$\partial_g L = u^\top \nabla_w L, \qquad \nabla_v L = \frac{g}{\|v\|_2}(I - uu^\top)\nabla_w L.$$

The gain controls motion along the current feature, while the remaining coordinates change its direction. The authors explain the benefit through improved gradient conditioning and adaptation of the effective learning rate. Under ordinary gradient descent, growth of the raw radius $\|v\|_2$ attenuates subsequent directional updates without changing the forward scale. This is a concrete explanation of why a representationally redundant parameterization can help; unlike our unconstrained $w = sv$, its raw radius is removed from the forward function.

Random Weight Factorization (Wang et al., 2022) uses unrestricted row factors $w_i = s_i v_i$ for continuous neural representations. It explains the benefit through changed optimization geometry and effective learning rates, supported by kernel-spectrum measurements and improved learning of fine-scale structure. This provides a precedent for the unrestricted gain-times-weight construction as well as its conditioning interpretation.

Wang et al. (2026) develop a related explanation for normalization gains in LLMs. For a squared-loss linear teacher–student model with isotropic inputs, jointly learning a column w_j and its gain γ_j gives the effective column $a_j = \gamma_j w_j$ the gradient-flow dynamics

$$\dot{a}_j = (\gamma_j^2 I + w_j w_j^\top)(w_j^\star - a_j).$$

Here $w_j^\star$ is the teacher column. The additional scale coordinate changes the effective preconditioner and accelerates convergence under their initialization. Their placement and gain-reparameterization designs extend this reasoning to different coordinated matrix updates. A separate stochastic factor-balance model shows that decaying only the matrix can cause gains to grow and sharpen the objective in factor coordinates, motivating WD on certain gains. This explanation concerns faster learning in a setting where direct matrix training already converges; the factor-balance analysis takes matrix WD as a premise. It therefore addresses a different question from why matrix scale correction fails in the first place. It also shows why exemption from WD should not be assumed for every gain parameterization.

Compensation and gradient aggregation. Fixup (Zhang et al., 2019) explicitly connects a shrinking noise–decay norm equilibrium to growth of residual-branch multipliers, which compensate for weight shrinkage and modify the effective learning rate. RescaleNet (Shao et al., 2020) likewise describes compensation between decayed convolution weights and undecayed branch gains. It attributes the larger updates of scalar versus vector gains to accumulation of gradients across dimensions, allowing stronger adaptation of branch importance. These are substantive precedents for both compensation and aggregation. The remaining question is why aggregation provides a better restoring signal than matrix-coordinate updates: a larger gradient alone does not determine its direction or its balance with noise.

Separating directional motion from magnitude learning. Magnitude–Direction Decoupling (Hägele et al., 2026) identifies two interacting effects: matrix norms determine the angular size of normalized optimizer updates, while transverse updates can increase norms without an outward task signal. The authors explicitly note that an inward radial signal must counteract this growth when the loss is scale-sensitive. They fix the directional matrix's norm and learn row and column gains separately. This removes geometric growth of the directional radius and retains finer scale control than existing normalization gains provide. It explains how the intervention separates the two learning channels, while the stochastic dynamics that make the gains learn useful scales remain less understood; their Appendix I leaves the mechanism behind the gain benefit open.

Changes in stochastic optimization. Muown (Lion et al., 2026) finds that increasing row magnitudes drive much of the observed spectral-norm growth under Muon. It normalizes row directions, updates their underlying parameters with Muon, and learns row gains with Adam. Its analysis uses a sign-based gain update as a tractable proxy and derives a stochastic convergence bound in the combined gain/direction coordinates. A lower noise coefficient, also observed empirically, supplies a candidate explanation for improved optimization. The comparison includes both gain and direction noise, and measures stationarity in different coordinates. It supplies a stochastic explanation for the method, without isolating whether the gain recovers a radial task signal missed by matrix updates.

A follow-up by Hübler et al. (2026) identifies another contribution: although Muown's stored row radius cancels from the forward function, it still controls directional step sizes. Growth of that radius implicitly reduces angular updates. AngularMuown replaces this effect with an explicit directional schedule while retaining separately trained gains. This separates two possible benefits of the parameterization—learning functional magnitudes and scheduling directional motion—which should not be attributed to gain adaptation alone.

Explicit amplitudes and gain update rates. nGPT (Loshchilov et al., 2024) explains why normalized representations need additional amplitudes for logit confidence, attention sharpness, and nonlinear activation ranges. It also shows how the stored gain scale controls relative Adam updates at an unchanged initial forward function. MuonS and MuonM (Zhu et al., 2026) combine fixed directional norms with learned radii, separating magnitude adaptation from directional norm inflation; MuonM additionally uses long-memory momentum in flat directions. Its regression analysis examines how longer memory improves signal learning relative to noise accumulation, providing a momentum-based explanation distinct from the dynamics of its learned radii. These accounts explain how scale separation changes geometry and update rates. To explain the asymmetry in observations O1–O3, we additionally examine how the task signal survives the optimizer in matrix and scale coordinates. Conditioning, norm control and this statistical effect can coexist.

A.4 Gradient statistics and adaptive-optimizer failures

Earlier analyses of Adam already show that adaptive normalization can suppress informative gradients. Reddi et al. (2018) construct convex problems in which occasional large gradients determine the correct population direction, but their contribution is discounted by the adaptive denominator. Frequent smaller gradients then produce motion in the opposite direction. Their AMSGrad remedy changes the second-moment memory to prevent this failure. These examples already demonstrate that an unbiased gradient stream need not yield a correctly directed adaptive update.

AdaShift (Zhou et al., 2019) examines the accumulated influence of each observation through momentum. A large gradient raises the current and subsequent second-moment estimates, reducing the coefficients applied to its own contribution. The authors use temporally separated observations for the numerator and denominator to remove this dependence under their independence assumptions. This directly anticipates our emphasis on informative events being attenuated by second-moment normalization. Balles & Hennig (2018) provide a complementary decomposition into a sign-based direction and relative-variance adaptation, explaining why attenuation can also be beneficial when it reflects unreliable gradient signs.

We connect these statistical effects to the scale-learning problem exposed by our LM-head experiments. In the Zipf model, we characterize how event frequency, batch size and optimizer memory produce strong, suppressed or reversed drift, and compare coordinate and shared-scalar updates at the same over-sharpened state. The scalar changes the gradient random variable before Adam

processes it, rather than changing Adam's memory rule. With sufficiently long memory, the scalar retains a restoring response where coordinate updates can increase the centered logit norm against the population gradient (Appendix I.8). This connects harmful scale growth and successful scale correction through the statistics seen by the same optimizer. Section 2.4 discusses the open problem of extending this analysis to backbone matrices and other optimizers.

A.5 Learned scaling and hyperparameter transfer

The maximal-update parametrization (μP) prescribes width-dependent initialization and update scales to retain feature learning and enable hyperparameter transfer (Yang & Hu, 2021; Yang et al., 2022; Yang & Littwin, 2023). In practice, WD also affects relative updates and transfer: Kosson et al. (2025) study this interaction in finite-width language models. These results motivate distinguishing forward-scale adaptation from the scaling of optimization dynamics.

Our width experiments make this distinction explicit. LRMs learn width-dependent forward scales, but the endpoint scales alone do not ensure learning-rate transfer. Scaling the LRM initialization and learning rate according to the corresponding μP prescription substantially aligns the observed optima across widths while retaining learned scale adaptation (§3.3). We therefore view LRMs as complementary to prescribed scaling rules. The distinction is between learning a useful forward scale and choosing coordinates whose learning dynamics remain comparable as width changes.

B Experiment settings

Our experiments use several training setups. Unless stated otherwise, the controlled experiments in Sections 2–3 use the `Falcon-H1-0.5B` architecture (Zuo et al., 2025). This model provides 35 extensively tuned μP hyperparameter multipliers: 12 forward multipliers, 7 matrix learning-rate multipliers, 7 matrix weight-decay multipliers, and 9 vector learning-rate multipliers. These provide a strong reference parametrization for studying learnable scales and are also used in the tuning ablation of Appendix C.2.

The multiplier-placement ablation uses a separate dense Llama setup described in Appendix C.1. The long-pretraining experiments in Section 4 additionally include Llama-3.2-3B and a Llama-based MoE, denoted 7B-A1B (approximately 6.874B total and 0.834B active parameters per token, including the tied embedding/output matrix). Appendix B.2 describes the data mixture and preparation for the Llama-3.2-3B runs.

For the controlled Falcon-H1 experiments, the default training duration is 30GT: 25GT at constant learning rate followed by 5GT of exponential learning-rate decay by a factor of 8. This duration covers the regime in which the matrix norms approach the noise–WD equilibrium studied in Section 2. Appendix B.1 describes the longer pretraining experiments used for final validation.

For matched comparisons in this setup, including runs with versus without LRMs and the tuning configurations of Appendix C.2, we use the same initialization seed and data ordering. Thus each comparison holds the initialization and minibatch sequence fixed while changing the training configuration of interest.

Unless stated otherwise, the remaining Falcon-H1 training hyperparameters follow Zuo et al. (2025). We use a warmup duration of 0.1GT, batch-size ramp-up with square-root learning-rate scaling, and global matrix WD of 0.1, further modified by the tuned WD multipliers of the reference recipe. For the Falcon-H1 multiplier-tuning comparison, we sweep the global learning rate on a logarithmic grid with ratio $\sqrt{2}$ between adjacent values and use the best tested value. The placement ablation and width-transfer sweep use their separately specified learning-rate settings.

Depth-wise and within-layer scale experiments. Figures 4 and 5 use the default 30GT Falcon-H1 setup with AdamW and the FULL tuned-multiplier configuration defined in Appendix C.2. Figure 4 adds scalar multipliers to every matrix layer. Figure 5 compares runs differing only by the addition of vector multipliers; row norms are normalized within each layer before aggregation across layers to isolate within-layer scale diversity.

Symmetry diagnostics. The runs in Figure 10 use the default 30GT setup with vector multipliers applied throughout the architecture. The compared configurations differ only by the application of $\lambda_{\mathrm{lrm}} = 2 \times 10^{-3}$ WD to the affected multipliers.

Gradient-clipping diagnostic. Figure 11 compares two runs of the default 30GT setup with scalar multipliers on all matrix layers. They differ only in whether multiplier gradients are included when computing the global ℓ_2 norm used to determine the clipping coefficient.

LM-head experiments. The head-relearning experiment in Figure 1 (left) uses a pretrained SmolLM2-135M model (Allal et al., 2025) and FineWeb (Penedo et al., 2024). This experiment uses batches of 512 sequences of length 2,048; the head LR is 0.04096 and its WD is either zero or 0.00625, as described in Section 2.1. The remaining controlled Falcon-H1 experiments use the training mixture associated with the Falcon-H1 setup.

Zipf simulations. For the representative trajectories in Figure 3, we simulate Adam training of the Zipf model with $p_i \propto i^{-1.5}$, vocabulary size $V = 10{,}000$, batch size $B = 100$, learning rate $\eta = 4096 \times 10^{-5}$, no WD, and $\beta_1 = \beta_2 = 0.9$.

Gradient-statistics diagnostic. The gradient statistics in Figure 3 are collected from the final checkpoint after 100GT of continued pretraining of the SmolLM2 model from Figure 1 (left) on FineWeb. We use batch size $B = 64$ and sequence length 2048, corresponding to approximately 131k tokens per batch. This batch scale is relevant when interpreting rare-event probabilities such as $P_+ \sim 0.1$.

For the Falcon-H1 recipes, we use z-loss (Chowdhery et al., 2022) with coefficient 10^{-4}, except in the controlled LM-head experiments of Figure 1, where we disable it to isolate scale learning. The newer dense Llama experiments use cross-entropy without this term; the MoE training objective additionally includes a load-balancing loss. Appendix B.1 specifies which loss is plotted for each long-pretraining recipe.

The scalar width-scaling experiment in Figure 6 (left) uses a smaller, 12-layer Falcon-H1 variant to access a wider range of widths. RMSNorm gains are frozen at one to keep block-input RMS norms fixed. The learning-rate-transfer panels instead use a six-layer Llama-style model; Appendix F.4 gives this separate setup.

Notations. We use *root mean square* convention for the norms of matrices and vectors throughout the paper. For example, the norm of a matrix $W \in \mathbb{R}^{n\times m}$ is $\|W\| = \sqrt{\frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} W_{ij}^2}$. Asymptotic symbols are used informally throughout, for relations between quantities depending on a small or large parameter (most often $1-\beta \to 0$ or $P_+ \to 0$); precise statements and sufficient conditions are recorded in App. J. We use $A \simeq B$ for leading-order equality, $A \approx B$ for a weaker numerical or qualitative approximation, $A \sim B$ and $A \asymp B$ for same order of magnitude, $A \propto B$ for equality up to a constant prefactor, $A \ll B$ (resp. $A \gg B$) for A much smaller (resp. much larger) than B in the relevant limit, and $A \lesssim B$ (resp. $A \gtrsim B$) for $A = O(B)$ (resp. $\Omega(B)$); the implicit constants may absorb dependence on within-sign moment regularity. Region-defining inequalities such as $1-\beta \ll Bp_i \ll 1$ describe the asymptotic regime in which the corresponding result holds. We also write $X \sim \mathcal{D}$ for "X is distributed as $\mathcal{D}$".

B.1 Long-pretraining configurations

We compare one baseline–LRM pair for each of the five recipes in Table 1. These are individual training runs; checkpoint averaging in Appendix B.3 does not provide independent training replicates. The Llama models are initialized from scratch, using the Llama tokenizer rather than pretrained model weights.

Architecture and optimization. The dense Llama models have 28 layers, hidden size 3,072, FFN size 8,192, and 24 query heads with eight KV heads. The MoE has 16 layers, hidden size 2,048, 32 query heads with eight KV heads, and 32 SwiGLU experts per layer with top-2 routing and intermediate size 2,048 per expert. Both use a vocabulary of 128,256, tied embeddings and output weights, RMSNorm, and zero dropout. The runs use BF16 arithmetic and gradient clipping at norm

Model / optimizer	Data	Context	Global batch	Base plateau LR
Falcon-H1 0.5B / Adam	Falcon-H1 mixture	4,096	$128 \to 256$	$9.05\times10^{-4} \to 1.28\times10^{-3}$
Falcon-H1 0.5B / Muon	Falcon-H1 mixture	4,096	$128 \to 256$	$1.28\times10^{-3} \to 1.81\times10^{-3}$
Llama 3B / Adam	FineWeb	2,048	512	6.4×10^{-4}
Llama 3B / Adam	Mixture (App. B.2)	8,192	128	6.4×10^{-4}
MoE 7B-A1B / Adam	FineWeb	2,048	1,024	6.4×10^{-4}

Table 1: Long-pretraining settings. Global batch counts sequences. Falcon doubles its batch at 10B tokens and increases the base LR by $\sqrt{2}$. LRs are scheduler values before parameter-group factors. Token budgets count processed positions, including masked positions where applicable.

one. Adam components use $(\beta_1, \beta_2) = (0.9, 0.95)$, with $\epsilon = 10^{-16}$ for Falcon and 10^{-6} for Llama. Matrix WD is 0.1, modified by the reference parameter-group factors for Falcon. Falcon's Muon uses momentum 0.95, Nesterov momentum, five Newton–Schulz iterations, and the matched-AdamW-RMS setting 0.2 for the selected matrix groups; other parameters use Adam. The Falcon pairs use initialization seed 42 and the Llama pairs use seed 1.

Training schedules and branches. All five recipes warm up for approximately 0.1B tokens. The loss-gap panel uses the original 100B runs, with exponential LR decay over 80–100B tokens: a factor of 16 for Falcon and 32 for Llama/MoE. The longer Llama runs branch from checkpoints near 80B tokens, before this decay, and continue toward 240B tokens. Their intended schedule remains at peak LR until 160B, decays over 160–200B, and then holds the final LR through 240B. FineWeb uses a 32-fold decay to 2×10^{-5}; the mixture uses a 128-fold decay to 5×10^{-6}. The Falcon evaluation runs likewise decay over 160–200B, by a factor of 32, and continue at their minimum LR for checkpoint averaging (Fig. 19). Thus the 100B loss curves are not truncated versions of the longer evaluation trajectories.

LRM configurations used in these runs. The experiments predate the single recommendation in Appendix E.5 and use related, but not identical, placements. Dense Llama uses row+column factors for attention and FFN matrices and column factors for the tied embedding/output matrix. Factors initialize to one, use Adam at twice the base LR, and receive WD 0.002; existing RMSNorm gains remain separate. The decomposition is implemented in the optimizer, which clips the effective-matrix gradients before splitting them into matrix and factor gradients.

The MoE uses explicit attention factors on Q/K columns, V rows and columns, and O rows and columns, together with expert row/column factors except Up rows. Their LR is twice the base LR. The original 100B MoE run uses zero multiplier WD. Its 240B continuation switches to WD 0.002 near 80B. Falcon uses the reference architecture's explicit vector multipliers and parameter-group LR factors, generally $4\times$ for multipliers, with separate factors for the embedding and block-input scales. Its multiplier and normalization-gain groups receive zero WD, and the separate LM-head multiplier is frozen; the final RMSNorm gains remain learnable. These historical variants should be distinguished from the broad placement with light WD that we recommend as a default.

MoE continuation. The FineWeb dense continuations restore model, optimizer, dataloader, and step state, while replacing the scheduler for the longer budget. MoE continuations restore model, dataloader, and step state but reset the optimizer. The MoE LRM continuation also required restarts and a scheduler correction: its final segment resumes near 223B tokens with constant LR 2×10^{-5}. Consequently its 240B trajectory does not follow an uninterrupted nominal schedule. These changes concern the longer evaluation trajectory; the original 100B loss comparison follows the matched 80–100B decay schedule.

FineWeb data. For the dense and MoE FineWeb runs, we use the `sample-100BT` subset, shuffle documents, normalize text encoding, and tokenize with the Llama-3.2 tokenizer. We concatenate and pack the text into 2,048-token sequences with standard causal-language-model supervision. Training follows the fixed prepared order, revisiting the approximately 99.1B-token corpus to reach 240B processed tokens. Thus this budget is not a count of unique data. The mixture uses a different, fixed 240B stream described next.

Reported training losses. Figure 8 reports LRM-minus-baseline loss at matching nominal training progress, averaged over consecutive 1B-token windows. Dense Llama reports cross-entropy per unmasked target token; the MoE curves use cross-entropy excluding the load-balancing auxiliary loss used in training. Falcon retains its logged training loss, including its configured z-loss. Each curve therefore compares the two models within one recipe; the raw gap magnitudes across recipes also reflect differences in data and objectives.

B.2 Data mixture and preparation for Llama-3.2-3B

We train the Llama-3.2-3B baseline and LRM models from scratch on the same fixed mixture of web text, specialized pretraining text, and instruction/reasoning examples. All sources are mixed during pretraining. This subsection describes the 240GT stream used for the reported downstream evaluations; the 100GT loss-gap comparison shares the training trajectory up to the branch near 80GT and uses a separate learning-rate decay over 80–100GT.

Mixture composition. We allocate 20% of the token budget to FineWeb (48GT), include the retained packed instruction/reasoning data once (about 35.2GT), and distribute the remaining 156.8GT among the Nemotron Specialized components in proportion to their prepared sequence counts. Table 2 gives the resulting source contributions.

Source dataset	Sequences	GT	Share (%)
FineWeb, sample-100BT	5,859,375	48.000	20.00
Nemotron-Pretraining-Specialized-v1	16,138,079	132.203	55.08
Nemotron-Pretraining-Specialized-v1.1	537,506	4.403	1.83
Nemotron-Pretraining-Specialized-v1.2	2,460,907	20.160	8.40
Tulu 3 SFT Mixture	79,434	0.651	0.27
Nemotron-SFT-Instruction-Following-Chat-v2	517,027	4.235	1.76
Nemotron-SFT-Instruction-Following-Chat-v3	326,594	2.675	1.11
Nemotron-SFT-Math-v3	748,287	6.130	2.55
Nemotron-SFT-Math-v4	126,738	1.038	0.43
Nemotron-SFT-Science-v2	501,614	4.109	1.71
OpenScience	1,245,190	10.201	4.25
OpenScienceReasoning-2	756,273	6.195	2.58
Total	29,297,024	240.001	100.00

Table 2: Data mixture used for Llama-3.2-3B. Dataset names link to the original releases; all Nemotron and OpenScience releases are from NVIDIA. Counts refer to the selected and processed data described below. GT and shares count input positions, including masked context and padding, and are rounded.

Web and specialized text. We use FineWeb's `sample-100BT` subset (Penedo et al., 2024). The Nemotron Specialized sources supply reasoning, mathematics, science, code, rewritten knowledge text, and question-answering material. We use their released text directly for next-token prediction, including examples written in an instruction or multiple-choice format.

Instruction-following data. We use Tulu 3 after removing its NuminaMath tool-integrated-reasoning subsource, both reasoning-enabled and reasoning-disabled portions of Nemotron Chat-v2, and the chat and instruction-following portions of Chat-v3. For Chat-v3, we recover missing initial user prompts from LMSYS-Chat-1M and WildChat-1M using the released matching information, and discard conversations whose requests remain unavailable. We supervise all retained assistant turns in Tulu and Chat-v2, and use the released turn-selection metadata for Chat-v3 chat.

Mathematical and scientific reasoning. From Nemotron Math-v3/v4, we retain short chain-of-thought examples from AoPS and Math StackExchange. We also convert tool-integrated-reasoning trajectories and overlong chain-of-thought examples into direct-solution examples: we keep the original problem and the full final visible assistant response, which may itself contain a worked solution, and discard intermediate reasoning and tool interactions. We use the trajectory's response

rather than a separate reference answer. We remove examples with embedded image data or explicitly tool-dependent answers and deduplicate exact problem–response pairs within this converted data.

For Nemotron Science-v2, we retain the `rqa`, `so`, and `syn_mcq` subsets and remove conversations containing tool calls or results. Merely defining a tool schema does not cause exclusion. OpenScience contributes its `OS-Q2.5-72B-10` and `OS-Q3-235B-4` subsets, and OpenScienceReasoning-2 contributes its training split. Modified sources replace their original versions in the mixture.

Formatting and supervision. We use the Llama-3.2 tokenizer, reusing reserved vocabulary entries for chat, reasoning, and tool delimiters. We repair text-encoding artifacts and normalize Unicode before tokenization. Instruction/reasoning data uses ChatML role delimiters, with `/think` or `/no_think` marking the reasoning mode in user messages; explicit reasoning is enclosed in `<think>` tags before the answer. The loss covers selected assistant responses, including their reasoning and serialized tool calls, while prompts, tool results, and padding are masked. Web and Specialized text instead use ordinary causal-language-model loss on all non-padding targets.

Length selection, packing, and mixing. We discard malformed conversations and examples without a usable request or supervised answer. We keep instruction/reasoning examples of at most 8,192 tokens after formatting, pack complete examples into 8,192-position training sequences, and pad unused positions. Longer examples are excluded, including converted Math examples that remain overlength. Plain text is concatenated and divided into training windows. All sequences use ordinary causal attention with continuous positions, including across packed-example boundaries. We shuffle data within sources before packing, then interleave the prepared sequences according to the fixed source quotas, without repeating them. The resulting order is fixed before training, so the baseline and LRM models see identical sequences and loss masks.

Deduplication and benchmark overlap. Beyond the exact-pair filtering of converted Math examples, we apply no additional content deduplication or benchmark decontamination. Some sources contain benchmark-related training material, so we do not claim that the mixture is benchmark-decontaminated.

B.3 Full long-pretraining evaluations

We report the benchmark results for all five long-pretraining pairs in Tables 3–4, including the FineWeb runs omitted from the main evaluation display. To reduce reliance on a single final checkpoint, we average **40 late-training checkpoints per model** for Falcon-H1 and Llama 3B on the mixture. The FineWeb averages use the 39 available checkpoints in the final evaluation window. These windows cover approximately 200–240GT; Falcon-H1 uses the final 40 available evaluations independently for each run, with the Adam+LRM window starting at 198GT. Figure 19 shows the Falcon-H1 evaluation trajectories underlying these averages.

Each entry reports the mean and sample standard deviation across checkpoints. The standard deviation describes observed checkpoint-to-checkpoint variability, which can include both evaluation noise and continued model evolution. The checkpoints belong to a single training trajectory and their temporal correlation is unknown. We therefore do not convert these standard deviations into standard errors of the means or confidence intervals; they also do not measure variability across independent training runs.

We retain the benchmark-score conventions of each evaluation campaign. For the Llama runs, HellaSwag and ARC use length-normalized accuracy; MMLU uses accuracy; MMLU-Pro and BBH use chance-adjusted scores; GSM8K and MATH use exact match; and IFEval averages strict prompt- and instruction-level scores. The additional FineWeb tasks use length-normalized accuracy for PIQA, OpenBookQA, ARC-E and SciQ, and accuracy for WinoGrande and LAMBADA. FineWeb evaluations use raw prompts, whereas the mixture uses its ChatML format for GSM8K, MATH and IFEval. We therefore emphasize comparisons within each baseline–LRM pair.

Table 3: Full results for the three mixture-trained pairs: mean ± checkpoint standard deviation over 40 evaluations per model, in percentage points. The first eight rows correspond to the main-paper benchmark suite; additional recorded Falcon-H1 benchmarks appear below the rule. We omit Falcon-H1 IFEval because these models were trained without a chat template and the evaluations did not use an appropriate chat setting. Other dashes indicate unavailable evaluations.

	Falcon-H1 0.5B Adam		Falcon-H1 0.5B Muon		Llama 3B mixture Adam	
Benchmark	Baseline	LRM	Baseline	LRM	Baseline	LRM
HellaSwag	48.91 ± 0.18	49.89 ± 0.25	50.31 ± 0.22	50.56 ± 0.24	64.12 ± 0.16	64.77 ± 0.17
ARC-C	38.70 ± 0.52	38.73 ± 0.50	39.04 ± 0.39	39.39 ± 0.57	49.42 ± 0.31	49.03 ± 0.25
MMLU	44.18 ± 0.42	45.33 ± 0.52	47.98 ± 0.33	47.96 ± 0.31	56.74 ± 0.18	57.45 ± 0.14
MMLU-Pro	18.26 ± 0.49	19.92 ± 0.55	18.96 ± 0.57	19.32 ± 0.36	23.33 ± 0.24	25.77 ± 0.21
BBH	9.70 ± 0.67	12.03 ± 0.46	10.13 ± 0.78	13.52 ± 0.66	18.75 ± 0.43	18.75 ± 0.39
GSM8K	48.35 ± 1.12	49.10 ± 0.74	48.02 ± 2.03	50.63 ± 2.38	68.85 ± 0.57	68.89 ± 0.50
MATH level 5	7.52 ± 0.62	9.07 ± 0.63	8.69 ± 0.69	9.49 ± 0.59	29.38 ± 0.88	30.55 ± 0.81
IFEval	—	—	—	—	62.03 ± 0.79	63.76 ± 0.96
WinoGrande	57.30 ± 0.56	58.49 ± 0.54	57.68 ± 0.53	58.34 ± 0.54	—	—
TruthfulQA MC2	41.36 ± 0.43	41.61 ± 0.37	42.60 ± 0.47	39.50 ± 0.40	—	—
GPQA	5.14 ± 1.08	3.60 ± 1.07	7.71 ± 1.19	7.27 ± 1.64	—	—
MuSR	4.06 ± 0.47	3.02 ± 0.67	4.76 ± 1.32	1.77 ± 0.49	—	—

Table 4: Full results for the two FineWeb pairs: mean ± checkpoint standard deviation over 39 evaluations per model. Scores and standard deviations are in percentage points, except LAMBADA perplexity (PPL), for which lower is better. Dashes indicate tasks not evaluated for the MoE. The dense model's low scores on several reasoning benchmarks contrast with its stronger HellaSwag and ARC results.

	Llama 3B FineWeb Adam		MoE FineWeb Adam	
Benchmark	Baseline	LRM	Baseline	LRM
HellaSwag	72.96 ± 0.20	73.55 ± 0.23	68.11 ± 0.23	68.67 ± 0.21
ARC-C	42.13 ± 0.50	41.58 ± 0.50	36.49 ± 0.44	37.36 ± 0.52
MMLU	26.45 ± 0.47	27.00 ± 0.38	—	—
MMLU-Pro	1.56 ± 0.15	1.49 ± 0.17	—	—
BBH	4.68 ± 0.33	3.28 ± 0.29	—	—
GSM8K	2.66 ± 0.37	2.60 ± 0.32	—	—
MATH level 5	0.81 ± 0.16	1.10 ± 0.18	—	—
IFEval	11.14 ± 0.97	10.12 ± 0.73	—	—
PIQA	78.68 ± 0.32	78.81 ± 0.25	77.80 ± 0.27	77.11 ± 0.24
WinoGrande	65.19 ± 0.45	65.41 ± 0.48	61.54 ± 0.43	62.49 ± 0.42
OpenBookQA	40.87 ± 0.45	39.25 ± 0.58	38.76 ± 0.64	37.25 ± 0.49
ARC-E	65.31 ± 0.37	64.77 ± 0.37	61.04 ± 0.47	61.33 ± 0.60
SciQ	86.82 ± 0.49	86.93 ± 0.39	82.18 ± 0.41	83.72 ± 0.29
LAMBADA	60.64 ± 0.44	61.15 ± 0.50	55.95 ± 0.45	56.89 ± 0.42
LAMBADA PPL ↓	7.01 ± 0.16	6.63 ± 0.11	8.68 ± 0.14	8.49 ± 0.14

Benchmark protocol. We use the Hugging Face leaderboard task settings implemented in the Language Model Evaluation Harness. Table 5 summarizes the task-level settings shared by the Falcon and Llama evaluations. Multiple-choice tasks score answer likelihoods; BBH here uses the 24-task multiple-choice leaderboard suite, rather than free-form chain-of-thought generation. MMLU averages accuracy over examples across its 57 subjects. We apply the leaderboard chance adjustment to MMLU-Pro, $\max(0, (a - 0.1)/0.9)$, and to each BBH task using its number of choices before averaging task scores.

Benchmark	Shots	Evaluation	Maximum generated tokens
HellaSwag	10	Multiple choice	—
ARC-C	25	Multiple choice	—
MMLU	5	Multiple choice	—
MMLU-Pro	5	Multiple choice	—
BBH	3	Multiple choice	—
GSM8K	5	Greedy generation	256
MATH level 5	4	Greedy generation	1,024
IFEval	0	Greedy generation	1,280

Table 5: Leaderboard evaluation settings. IFEval is reported only for the dense Llama models. The additional FineWeb tasks (PIQA, WinoGrande, OpenBookQA, ARC-E, SciQ, and LAMBADA) use zero-shot evaluation.

For GSM8K, we use the harness's flexible numeric-answer extraction. MATH uses the leaderboard answer parser and mathematical-equivalence check against the reference solution, aggregated over examples in the seven level-5 subject subsets. IFEval averages strict prompt-level and instruction-level compliance. Generation is greedy, with sampling disabled and temperature zero. The Llama FineWeb evaluations use an input sequence limit of 2,048; the mixture evaluations use 3,072. The mixture's GSM8K, MATH, and IFEval prompts use ChatML with `/no_think`, with few-shot examples represented as turns where applicable, and stop at `<|im_end|>`. Falcon and FineWeb use raw prompts. The raw GSM8K and MATH tasks retain their task-specific stopping delimiters.

C Additional pretraining ablations

C.1 Multiplier placement ablation

Configurations. We train 29 configurations independently from initialization: ten cumulative placements, 15 individual additions, and four learned-RMSNorm controls. Appendix H.3 specifies each topology and its parent comparison, and Appendix H.2 gives the corresponding forward equations.

Training. We use a dense Llama architecture with 12 layers, hidden size 1,024, FFN size 4,096, 16 query heads, four KV heads, and an untied embedding and LM head with vocabulary size 128,256. Each run uses 40B FineWeb tokens, sequence length 2,048, global batch size 256 and BF16 arithmetic. After a 200-step warmup, the learning rate is constant through 35B tokens and decays exponentially by a factor of 32 over the final 5B. Adam uses $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-6}$. The peak learning rate is 2.56×10^{-3} for matrices, LRMs and trainable RMSNorm gains; WD is 0.1 for matrices and zero for the scales, which initialize to one. The microbatch size is 128 except at stage 9, where a compiler workaround requires 64; the global batch size is unchanged. We clip gradients at norm one. In learned-RMSNorm controls, norm-gain gradients are excluded when computing the clipping coefficient, but receive the same coefficient as all other parameters.

Replication and uncertainty. Each of the 29 configurations has 20 joint initialization/data-stream seeds, for 580 trajectories. Configurations share initialization and data stream within a seed. The first ten seeds use ten physical data-order families; the second ten use new initializations and disjoint suffix streams of the same families. Following the predeclared analysis, we treat the 20 runs as replicates without a family-level correction. The endpoint is the mean of exactly the last 1,000 recorded training losses, steps 75,295–76,294 inclusive. Error bars are sample standard deviations divided by $\sqrt{20}$. For Figure 7(b), we first compute child-minus-parent loss within each seed and then estimate the

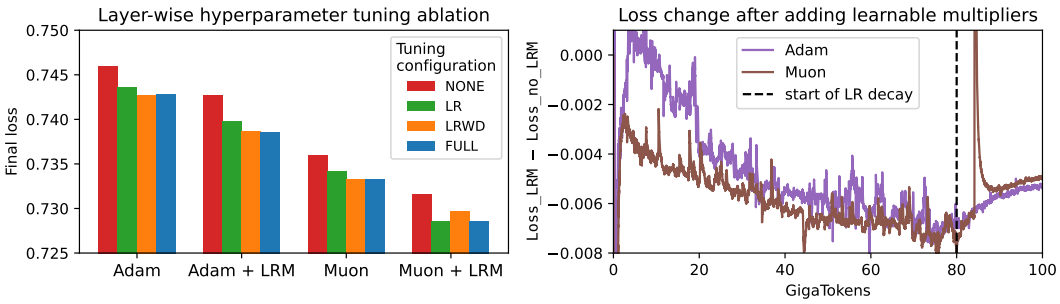


Figure 9: *(Left)* Loss after LR decay for the four tuning configurations, with and without learnable vector multipliers (+LRM). *(Right)* Loss difference between runs with and without LRMs over training.

mean and SE of these paired differences; negative values indicate improvement. These bars describe uncertainty in the means, not the full run-to-run spread or simultaneous confidence intervals.

Comparisons with learned RMSNorm and residual scales. Table 6 compares C_8 with learned RMSNorm gains against the three simpler learned-RMSNorm controls. We compute loss differences within each seed, using the same endpoint and pairing as above. As a sensitivity check, we average the two paired differences within each data-order family and estimate uncertainty across the ten family means. All three comparisons retain negative pointwise 95% Student-t confidence intervals (nine degrees of freedom).

Table 6: Training loss of C_8 with learned RMSNorm gains minus each control. Negative differences favor C_8. The primary SE uses 20 paired seeds; the family SE uses ten family means.

Control	Mean difference	SE (20 seeds)	SE (10 families)
RMSNorm only	−0.02350	0.00194	0.00241
RMSNorm + residual scalars	−0.01185	0.00188	0.00219
RMSNorm + residual vectors	−0.00621	0.00163	0.00133

C.2 Multiplier tuning ablation

Following Zuo et al. (2025), training recipes typically tune three quantities: (i) learning-rate multipliers, (ii) weight-decay multipliers, and (iii) forward multipliers. Since LRMs make the forward scales learnable, we test whether they reduce the need for explicit tuning of (ii) and (iii).

We compare four configurations: **NONE** (all multipliers set to 1), **LR** (only learning-rate multipliers tuned), **LRWD** (learning-rate and WD multipliers tuned), and **FULL** (all multipliers tuned). Tuned values are taken from Zuo et al. (2025). Each configuration is run with and without learnable vector multipliers for both Adam and Muon. When LRMs are added, tuned forward-multiplier values are used as their initialization where applicable.

Figure 9 (left) shows a consistent pattern for Adam and Muon. Untuned baselines perform worst; LR-tuned baselines and untuned LRMs form an intermediate tier; and LR-tuned LRMs achieve the best loss, while additional WD and forward-multiplier tuning provides only marginal improvement. Thus LRMs remove much of the need for explicit forward- and WD-multiplier tuning, but do not replace learning-rate tuning.

Figure 9 (right) shows that the lossadvantage of LRMs grows throughout the constant-LR stage and shrinks only slightly during LR decay. This is consistent with LRMs improving the asymptotic representation learned by the model rather than merely accelerating early optimization.

D AI-use statement

Our use of LLM-based agents spanned the five areas described below. We take full responsibility for the paper's content.

Experiment execution. Some experiments were performed when coding agents had become highly capable. Our typical workflow was to describe the experiment we wanted to run and ask an agent to prepare the configuration, launch the job, and monitor it. We designed the experiments and chose their main settings and hyperparameters.

Plotting. We described which data to plot and how the plot should be structured. The agent produced drafts, and we provided feedback until we were satisfied with the figure. As with experiment execution, we were responsible for the design and main decisions, while the agent handled implementation.

Writing. We used LLM agents to polish our writing, correct typos and language errors, and suggest effective phrasing when we struggled to express an idea. The agents were not used to draft sections or structure the arguments and exposition. We developed the paper's structure, from sections to paragraphs.

Literature review. We used LLM agents to help locate relevant works and sometimes summarize their findings and relation to our results. One useful workflow was to ask an agent to describe a wide range of related works in a single document using shared notation and terminology. This significantly reduced the overhead of adapting to each paper's notation and terminology. We nevertheless verified all key results used in our comparisons by reading the original papers.

Theoretical results. This was our most involved use of LLM agents. In intermediate versions of the paper, we had conducted experiments roughly corresponding to Sections 2.1, 3, and 4, and had identified observations O1–O3. These observations, especially O1, were counterintuitive: the optimizer failed to correct harmful matrix scales, whereas in deep learning we generally expect gradient descent and optimizers such as Adam and Muon to converge reliably to good solutions.

As an experiment in autonomous AI research, we asked LLMs (at the time, `gpt-5.5-pro`, `opus-4.6` in Claude Code, and `gpt-5.5` in Codex) to find a toy setting in which observations O1–O3 could be reproduced and explained mechanistically. Despite many rounds of small simulations and theoretical derivations, these attempts did not reproduce all three observations in a simple setting. We then conducted our own theoretical exploration. Although we did not immediately find a suitable model, we became convinced that the Zipf model with Adam was the right setting. We returned to agent-based exploration, asking the agents to reproduce the three observations within this model. This attempt also failed, but for a different reason: we had not formalized the observations precisely enough, and the agents satisfied our formulations only superficially by exploiting shortcuts.

Finally, we returned to our own investigation of the Zipf model and obtained the results in their current form. Some of our derivations used heuristic, physics-style arguments, so we asked an agent to develop more precise derivations for the paper. Most of its contribution is concentrated in Appendix J. We formulated the general rare-event gradient setting ourselves and obtained the regime boundary through an approximate derivation. Unlike the Zipf results, which we intentionally kept at the level of approximate arguments, we felt that the general rare-event gradient result could be fully formalized as a rigorous theorem. The agent (`gpt-6-astra` in Codex) then completed the proof presented in Appendix J.

E Practical considerations introduced by learnable multipliers

Learnable multipliers introduce several practical considerations for stable and effective training. We discuss four of them: architectural symmetries, gradient clipping, exponential parametrization, and learning the LM-head scale. We then summarize the LRM recipe used in our experiments.

E.1 Symmetries

In our controlled experiments, learnable multipliers can adapt their task-relevant scale without weight decay, unlike matrix norms whose scale can become dominated by the noise–WD balance. This does not imply that WD is never useful for multipliers. Adding explicit multipliers can introduce redundant scale degrees of freedom associated with architectural symmetries: transformations of the parameters that leave the represented function unchanged (Dinh et al., 2017). Drift along these symmetry

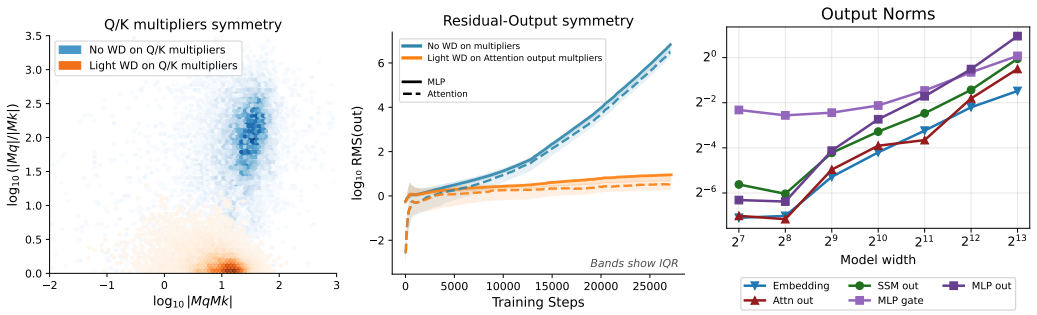


Figure 10: Effect of light WD on Q/K multiplier symmetry *(left)* and residual output symmetry *(middle)*. Light WD (orange) suppresses the drift and unbounded norm growth observed in the baseline without WD (blue). *(right):* block-component output norms (Embedding, Attn out, SSM out, MLP gate, MLP out) at end of training, across a model-width sweep $d \in \{2^7, \ldots, 2^{13}\}$. Most norms grow with d, reflecting the residual-normalization symmetry leaving the absolute output scale unconstrained at large width; the MLP gate is largely width-stable.

directions can produce increasingly large parameter or activation scales without changing the model output. While functionally harmless in exact arithmetic, such growth can become numerically problematic in low-precision formats such as `bfloat16`, where it can increase quantization error and degrade gradient estimates (Micikevicius et al., 2017; D'Angelo et al., 2024; Le Scao et al., 2022). We highlight two symmetries common in language-model architectures:

- **Multiplicative symmetry.** When two learnable factors a and b appear only through their product ab, the reparameterization $(a, b) \mapsto (sa, s^{-1}b)$ leaves the forward map unchanged for any $s \neq 0$. As a result, $\|a\|$ can grow while $\|b\|$ shrinks. An example of that is the product of queries Q and keys K in the attention computation. Figure 10 (left) illustrates this: for the baseline without intervention (blue), the product of the Q/K multipliers (x-axis) remains stable while their scale ratio (y-axis) drifts significantly.
- **Normalization symmetry.** Suppose a quantity c enters subsequent computation only through its normalized value $\hat{c} := c/\|c\|_{\text{rms}}$. Then, for any $s > 0$, the transformation $c \mapsto sc$ leaves $\hat{c}$, and hence the normalized forward computation, unchanged. Its absolute scale is therefore unconstrained along this symmetry direction. Figure 10 (middle) illustrates the resulting drift: without multiplier WD, the RMS norms of residual outputs grow strongly over training, while light WD suppresses this growth.

In standard architecture without learnable multipliers, these symmetries are essentially fixed by equilibrium norms of the weight matrices. When we first added learnable multipliers, we had a drift along symmetry directions, leading to large activations, and, eventually, divergence or NaN values.

Possible remedies include rescaling parameters along known symmetry directions or avoiding redundant factors; Appendix H.2 describes which multiplier placements share forward effects. In the Falcon-H1 experiments shown here, we instead apply a small weight decay $\lambda_{lrm} = 2 \times 10^{-3}$ to multipliers to suppress the drift. **The purpose of this WD differs from WD on matrix layers:** we use a much smaller coefficient to suppress symmetry drift, although ordinary WD also penalizes the task-relevant scales. This choice is experiment-specific: the dense placement ablation in Appendix C.1 uses zero multiplier WD.

E.2 Gradient clipping

Global ℓ_2 gradient clipping is commonly used to improve training stability (Pascanu et al., 2013). We use a clipping threshold of 1. In our initial experiments, however, adding scalar learnable multipliers produced little or even negative improvement.

Figure 11 (middle) shows that the multiplier gradients can dominate the global gradient norm early in training. As a result, the clipping coefficient is largely determined by the multiplier coordinates, even though the norm of the remaining model gradients is well below the clipping threshold. This unnecessarily shrinks the updates of all model parameters.

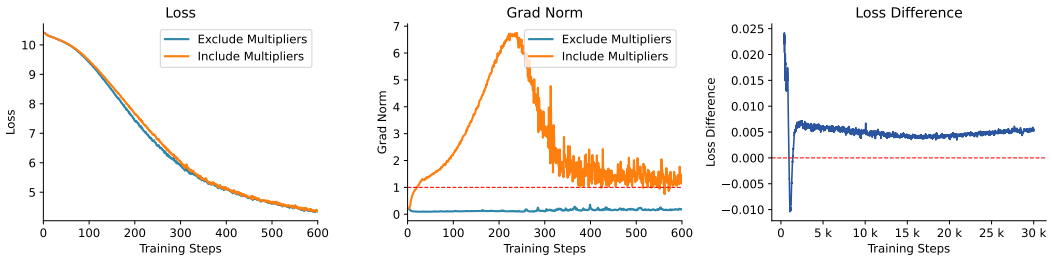


Figure 11: Training dynamics when excluding (blue) vs. including (orange) multiplier gradients from the global clip norm: (Left) initial loss, (Middle) initial gradient norm, and (Right) the long-term loss difference.

We therefore exclude multiplier gradients when computing the global gradient norm, while applying the resulting clipping coefficient to all gradients, including those of the multipliers. This mitigates the early over-clipping and improves the resulting training loss, as shown in Figure 11 (right).

E.3 Exponential parametrization

A natural idea is to use a logarithmic scale for the learnable multipliers, for example, $s \to e^s$ in the scalar case. Such reparameterization can be beneficial when different multipliers tend to have both large and small scales, making it problematic to learn with uniformly scaled optimizer updates given by the multipliers' learning rate. However, in agreement with (Salimans & Kingma, 2016), we observed that log-scale parametrization gives at most a slight performance advantage while amplifying symmetry-related instabilities discussed in sec. E.1.

E.4 Learning the LM head scale

Output scaling can affect the balance between feature learning and lazy training, where the model changes little beyond its initial linearization (Chizat et al., 2019; Yang & Hu, 2021). For the LM head, we distinguish column multipliers c_j, which scale hidden input features, from row multipliers r_v, which separately rescale each vocabulary logit. The final RMSNorm gains already provide column scales. Vocabulary multipliers add a different freedom: token-dependent rescaling of the outputs.

In our initial experiments, adding vocabulary-row multipliers on top of the final RMSNorm gains degraded performance. One possible explanation is that these factors allow output scales to adapt without corresponding improvements in shared backbone features. We did not isolate this mechanism experimentally. We also tried removing the column gains and sweeping a fixed scalar head multiplier to encourage feature learning (Atanasov et al., 2025), without observing the expected improvement.

The larger placement study gives a more modest result. Adding head vocabulary rows to C_8 produces C_9 and changes mean training loss by -0.00111 ± 0.00193 (paired mean $\pm$ SE over 20 seeds). This is a small improvement relative to its uncertainty, not a clearly resolved benefit. Together with the earlier degradation, it motivates retaining hidden-feature scale adaptation while omitting vocabulary-row multipliers from our default recipe.

E.5 Recommended LRM recipe

Placement. We recommend row and column LRMs on matrix layers, retaining the architecture's existing normalization gains, with one exception: we omit vocabulary-row multipliers on the LM head for the reasons above. For the untied dense architecture in Appendix H.3, this is the C_8 placement with learned RMSNorm gains. When embedding and head weights are tied, we apply only hidden-dimension column multipliers to their shared matrix, so vocabulary scales are absent from both uses and the effective weights remain tied.

Keeping redundant multipliers. We use a small multiplier WD of $\lambda_{\text{lrm}} = 2 \times 10^{-3}$ to suppress drift along multiplicative and normalization symmetries (Appendix E.1). This lets us retain redundant factors, such as V rows alongside O columns and Up rows alongside Down columns, and use the same

simple placement rule across architectures. Avoiding redundant factors is also possible, but requires identifying the equivalent placements in each architecture. We prefer broad placement with light WD as a practical default. The WD suppresses the observed drift; it does not remove the symmetries and also penalizes task-relevant scales. This recommendation is distinct from the controlled placement ablation, which uses zero multiplier WD; individual pretraining recipes also differ in their retained factors.

Initialization and optimization. We initialize row and column factors to one, preserving baseline forward scaling, and optimize them directly with Adam without an additional scalar factor. The matrices retain their chosen optimizer and WD. For explicit LRMs, we exclude their gradients from the global clipping norm but apply the resulting clipping coefficient to all gradients (Appendix E.2). For learning-rate transfer, we absorb fixed μP forward factors into LRM initialization and scale LRM learning rates accordingly (Section 3.3).

F WIDTH SCALING AND μP DETAILS

This appendix expands the width-scaling analysis of Section 3.3. We first derive how forward multipliers interact with the noise–WD equilibrium and the width scalings relevant to our μP parametrization. We then test whether LRMs learn these forward scales empirically, analyze the finite-width deviation of their learned slopes from the idealized μP trends, and finally test the stronger requirement of learning-rate transfer across widths. Additional diagnostics for symmetry-affected activations are reported at the end.

F.1 FORWARD-MULTIPLIER SCALING UNDER NOISE–WD EQUILIBRIUM

Scaling recipe	LR η	WD λ	Multiplier s	$\|\mathbf{y}\|$	$\frac{\|\Delta\mathbf{y}\|}{\|\mathbf{y}\|}$
LR only	d^{-1}	const	const	$d^{1/2}$	$d^{-1/2}$
LR + WD	d^{-1}	d	const	const	const
Forward multiplier	const	const	d^{-1}	const	const

Table 7: Scaling of the output norm $\|\mathbf{y}\|$ of a linear layer $\mathbf{y} = sW\mathbf{x}$, with $\mathbf{x} \in \mathbb{R}^d$ and $s \in \mathbb{R}$, and of the relative norm $\|\Delta\mathbf{y}\|/\|\mathbf{y}\|$ of the matrix-induced update $\Delta\mathbf{y} = s\,\Delta W\mathbf{x}$. The LR-only finite-width scaling recipe is taken from Yang et al. (2022); Dey et al. (2023); Hu et al. (2024); its WD-aware adjustment is discussed in Kosson et al. (2025); and an alternative parametrization placing the width dependence in forward multipliers is used in Zuo et al. (2025).

Consider a linear layer $\mathbf{y} = sW\mathbf{x}$ with $\mathbf{x} \in \mathbb{R}^d$ and normalized input norm $\|\mathbf{x}\| = \text{const}$. Assume that the matrix norm is set by the noise–WD equilibrium,

$$\|W\| \propto \sqrt{\frac{\eta}{\lambda}}. \tag{9}$$

Under an LR-only recipe with $\eta \propto d^{-1}$, $\lambda = \text{const}$, and $s = \text{const}$, we obtain $\|W\| \propto d^{-1/2}$. If the RMS-normalized amplification factor introduced in Appendix F.3 is width-independent, the output norm then scales as

$$\|\mathbf{y}\| \propto s\,d\,\|W\|\,\|\mathbf{x}\| \propto d^{1/2}, \tag{10}$$

while the relative update magnitude scales as

$$\frac{\|\Delta\mathbf{y}\|}{\|\mathbf{y}\|} \propto \frac{\|\Delta W\|}{\|W\|} \propto \frac{\eta}{\sqrt{\eta/\lambda}} = \sqrt{\eta\lambda} \propto d^{-1/2}, \tag{11}$$

where we used the scale-free update approximation $\|\Delta W\| \propto \eta$.

This calculation should be interpreted as a finite-width diagnostic of the interaction between WD and practical scaling rules, not as a shortcoming of μP theory. In particular, Yang & Littwin (2023)

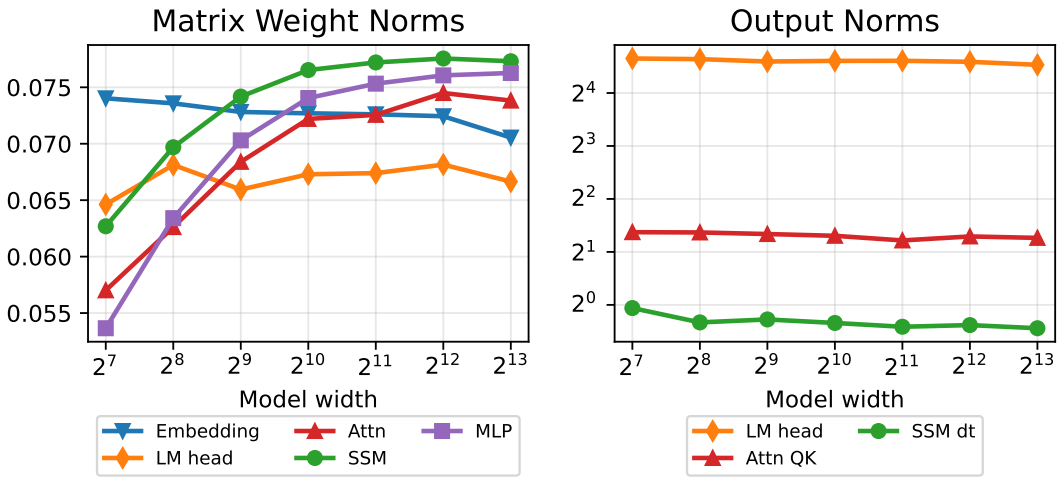


Figure 12: **Endpoint coordinate checks across model width.** *Left:* RMS norms of matrix parameters. *Right:* scale-sensitive forward quantities, including LM-head logits, attention `QK`, and SSM `dt`. Values across layers are aggregated by geometric mean.

notes that the total decay strength $\eta\lambda$ must also be scaled appropriately for an infinite-width adaptive-optimizer limit. In the simplified setting above, constant activation and relative-update scales can be restored either by scaling WD as $\lambda \propto d$ while keeping s fixed, or by keeping η, λ fixed and scaling the forward multiplier as $s \propto d^{-1}$. The latter is the forward-multiplier role that LRMs make learnable.

This gives the reference forward-scaling behavior against which we compare the scales learned by LRMs in the width sweep.

F.2 Endpoint coordinate checks and learned forward scales

We next ask whether LRMs recover the width-dependent forward scaling motivated above without imposing that scaling on the LRM coordinates. We use scalar LRMs, vary the model width d, and keep the matrix learning rate and weight decay fixed across widths. Under the forward-multiplier convention used here, the corresponding μP reference trends are d^{-1} for single linear outputs such as the LM head and SSM `dt`, and d^{-2} for attention `QK`, since both `Q` and `K` contribute a factor d^{-1}.

Matrix norms. Figure 12 (left) shows that matrix norms remain approximately constant across width. This is consistent with the noise–WD equilibrium $\|W\| \propto \sqrt{\eta/\lambda}$: when η and λ are fixed, the equilibrium itself carries no explicit width dependence. Thus the width-dependent scaling of the effective forward map must be provided elsewhere, here through the learned multipliers.

Coordinate checks. Figure 12 (right) tracks quantities whose absolute scale is directly meaningful for the forward computation. The LM-head logits, SSM `dt`, and attention `QK` remain approximately stable across width, showing that LRMs learn width-dependent forward scales that keep these coordinates controlled. We exclude quantities such as embeddings and residual-block outputs from this comparison because their absolute scale can drift along normalization symmetries without changing the model function; these are analyzed separately in Appendix F.5.

Learned scales. Figure 6 (left) compares the learned multiplier values with the corresponding μP reference trends. The learned scales follow trends close to d^{-1} for the LM head and SSM `dt`, and to d^{-2} for attention `QK`, although the empirical slopes are slightly shallower.

Overall, these endpoint checks show that LRMs recover the qualitative forward-scale behavior targeted by μP without fixing their final values by hand. The learned slopes are nevertheless systematically shallower than the corresponding reference trends. We next quantify this residual mismatch through an RMS-normalized amplification diagnostic.

F.3 RMS-NORMALIZED AMPLIFICATION DIAGNOSTIC FOR LEARNED MULTIPLIER SLOPES

We use the learned multipliers in Figure 6 (left) to quantify the residual difference between their empirical width scaling and the corresponding μP reference trends. For a scalar-multiplied linear map $\mathbf{y} = sW\mathbf{x}$, define the RMS-normalized amplification factor

$$A_{\mathrm{rms}}(W, \mathbf{x}) := \frac{\|W\mathbf{x}\|}{d\,\|W\|\,\|\mathbf{x}\|}, \tag{12}$$

where all norms follow the root-mean-square convention used throughout the paper. The forward norm can then be written as

$$\|\mathbf{y}\| = s\,d\,A_{\mathrm{rms}}(W, \mathbf{x})\,\|W\|\,\|\mathbf{x}\|. \tag{13}$$

Unlike a pure alignment measure, A_{rms} depends jointly on the singular-value distribution of W and on the orientation of $\mathbf{x}$ in its singular-vector basis. In particular, it should not be identified with the alignment notion arising from the spectral-norm analysis of feature learning in Yang et al. (2024).

Under this decomposition, a reference multiplier scaling $s \propto d^{-1}$ together with width-stable $\|W\|$, $\|\mathbf{x}\|$, and $\|\mathbf{y}\|$ would correspond to a width-independent A_{rms}.

We apply this diagnostic only to the single-linear-output cases, namely the LM head and SSM `dt`; attention `QK` is bilinear and is not covered by this decomposition. In the width sweep, the input norms, matrix norms, and selected output norms are approximately stable across width, while the learned multipliers decrease slightly more slowly than the corresponding reference trends. For the single-linear-output cases, this implies an inferred A_{rms} that decreases with width; in particular, this applies to the LM-head logits and SSM `dt`. Thus the shallower learned-multiplier slopes are consistent with LRMs compensating for the inferred width dependence of the RMS-normalized amplification factor while keeping the relevant forward coordinates controlled.

This diagnostic does not identify the source of that width dependence: it may arise from changes in the singular-value distribution of the matrix, changes in the orientation of the input features, or both. Determining these contributions separately would require additional spectral analysis, which we leave for future work.

Importantly, these results remain statements about the forward scales reached during training. Neither recovering the μP-like endpoint scales nor accounting for their finite-width deviations establishes that the optimization dynamics themselves remain comparable as width changes. We test this stronger requirement next.

F.4 LEARNING-RATE TRANSFER SETUP

Unlike the Falcon-H1 scalar-width experiment above, this sweep uses a six-layer Llama-style decoder trained from scratch on 40B FineWeb tokens. The hidden width is $d \in \{256, 512, 1024, 2048, 4096\}$, the SwiGLU intermediate size is $2d$, and attention uses head dimension 64 with a 4:1 query-to-KV-head ratio. Embedding and output weights are untied, with vocabulary size 128,256. We multiply the standard attention logits by $1/8$, giving $QK^\top/64$ before the softmax. All RMSNorm gains are frozen at one except in the RMS-gains-only control.

The batch contains 256 sequences of length 2,048. We use 200 warmup steps, constant LR until approximately 35B tokens, and 32-fold exponential decay over the final 5B. Adam uses $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-6}$, with matrix WD 0.1 and zero multiplier WD. We use BF16 and clip the effective-model gradient at norm one before splitting matrix and multiplier gradients. In the RMS-only control, gain gradients are excluded from the clipping-norm calculation but receive the resulting clipping coefficient. Each point uses initialization seed 1 and the same fixed data stream. Learning-rate grids contain five to seven points per recipe and width, with adjacent values differing by a factor of two. We report final-step training loss; averaging the final 1,000 steps selects the same best tested LR in every panel.

We compare five parametrizations:

1. **No scale adaptivity (B01):** no learnable forward-scale parameters beyond the fixed architecture.

2. **RMS gains only:** the standard learnable RMSNorm gains are retained, but no additional LRMs are introduced.
3. **LRM (B08):** the row/column topology specified below, with LRM initialization and optimizer learning rates kept width-independent.
4. **Fixed μP:** the width-dependent forward scaling prescribed by μP is applied through fixed, non-learnable forward multipliers.
5. **μP + LRM:** the same learnable forward scales as in the LRM configuration are retained, but the LRM coordinates themselves are width-scaled according to the corresponding μP parametrization.

The B08 topology uses row+column factors on the embedding, Q, K, O, MLP Up, and MLP Gate matrices; column factors on V and the LM head; and a row factor on MLP Down. Without μP scaling, all factors initialize to one and use the matrix LR. This topology differs from the broad-placement default in Appendix E.5.

For the μP+LRM configuration, let $m_\ell^{\mu\mathrm{P}}(d)$ denote the fixed forward multiplier that the corresponding μP recipe would assign to location ℓ at width d. Relative to a reference width d_0, define

$$\kappa_\ell(d; d_0) = \frac{m_\ell^{\mu\mathrm{P}}(d)}{m_\ell^{\mu\mathrm{P}}(d_0)}. \tag{14}$$

For an LRM represented by a single multiplicative coordinate u_ℓ, we apply this factor both to its initialization and to its optimizer learning rate:

$$u_{\ell,0}(d) = \kappa_\ell(d; d_0)\, u_{\ell,0}(d_0), \qquad \eta_{u_\ell}(d) = \kappa_\ell(d; d_0)\, \eta_{u_\ell}(d_0). \tag{15}$$

When the same forward factor is represented by a product of row and column LRMs, we split the factor equally between the two coordinates:

$$\begin{aligned} r_{\ell,0}(d) &= \kappa_\ell(d; d_0)^{1/2} r_{\ell,0}(d_0), \quad c_{\ell,0}(d) = \kappa_\ell(d; d_0)^{1/2} c_{\ell,0}(d_0), \\ \eta_{r_\ell}(d) &= \kappa_\ell(d; d_0)^{1/2} \eta_{r_\ell}(d_0), \quad \eta_{c_\ell}(d) = \kappa_\ell(d; d_0)^{1/2} \eta_{c_\ell}(d_0). \end{aligned} \tag{16}$$

Thus a d^{-1} forward factor is split into $d^{-1/2}$ row and column factors. We also scale the raw-matrix initialization so that fixed μP and μP+LRM start from the same effective matrices at each width. Table 8 gives the complete scaling with $m = d/256$. The raw matrices retain the swept LR and WD 0.1; their initialization is not the same as in the naive-LRM control.

Matrix group	Raw initializer std	Factor product	Effective initializer std
Embedding	σ_g	1	σ_g
Hidden Q/K/V/O, Up/Gate/Down	$\sigma_g\sqrt{m}$	$1/m$	$\sigma_g/\sqrt{m}$
LM head	σ_g	$1/m$	σ_g/m

Table 8: Initialization for fixed μP and μP+LRM in the transfer sweep. Here σ_g is the reference initializer's standard-deviation parameter. The embedding, Q/K/V, and Up/Gate matrices use a normal distribution truncated at two standard deviations with $\sigma_g = 0.02$; O and Down use $0.02/\sqrt{12}$. The LM head uses an untruncated normal with standard deviation 0.02. Truncation bounds scale with the standard-deviation parameter.

Embedding factors remain at unit initialization and unit LR ratio. Elsewhere, a single factor starts at $1/m$ and uses LR η/m; two factors each start at $1/\sqrt{m}$ and use LR $\eta/\sqrt{m}$. At the reference width, fixed μP coincides with B01 and μP+LRM with B08, so equivalent reference-width points reuse the same executions. We keep $\epsilon = 10^{-6}$ fixed for all matrix and factor coordinates. The experiment tests finite-width LR transfer, rather than exact coordinate equivalence at the numerical ϵ floor.

To compare the location of the preferred learning rate independently of the large width-dependent offset in absolute loss, we report *excess loss*

$$\Delta L_{r,d}(\eta) = L_{r,d}(\eta) - \min_{\eta' \in \mathcal{G}} L_{r,d}(\eta'), \tag{17}$$

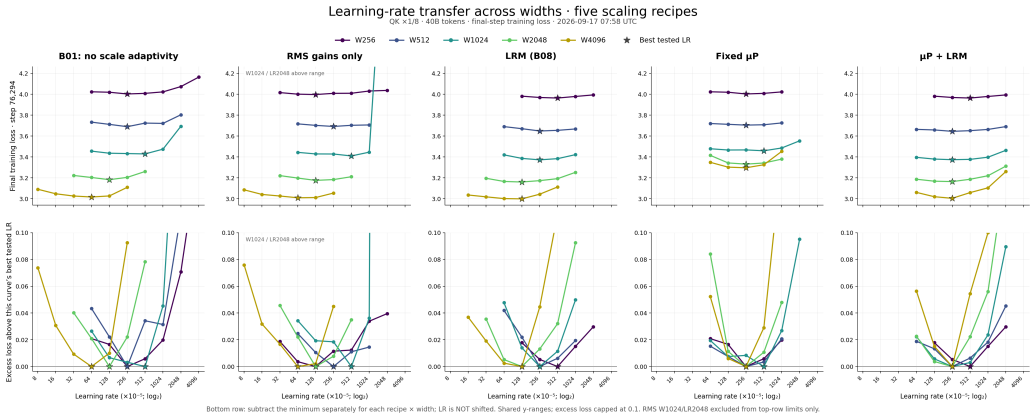


Figure 13: **Learning-rate-transfer controls across widths.** We sweep the global learning rate for the five parametrizations above. *Top:* final training loss. *Bottom:* excess loss relative to the best tested learning rate for each recipe–width pair; stars mark the best tested grid point for each curve. Naive scale-adaptive parametrizations exhibit width-dependent preferred learning rates, whereas fixed μP substantially aligns them. Applying the corresponding μP scaling to the LRM initialization and optimizer learning rate achieves comparable alignment while retaining learnable forward scales.

where r denotes the scaling recipe, d the model width, and $\mathcal{G}$ the tested learning-rate grid. Thus every curve has minimum zero, while the horizontal location of its minimum reveals the best tested learning rate. Learning-rate transfer corresponds to these minima remaining approximately aligned across widths.

Figure 13 additionally shows that the alignment is specific to the parametrization rather than merely to the presence of learnable scales: RMS gains and naively parametrized LRMs do not by themselves produce the width alignment obtained with fixed μP or μP-scaled LRM coordinates.

F.5 Symmetry-affected activation norms in the width sweep

Finally, we report width-scaling diagnostics for quantities whose absolute scale is less directly interpretable. The analysis above focuses on quantities whose scale is meaningful for the forward computation, such as LM-head logits, attention `QK`, and SSM `dt`. In contrast, embeddings and residual-block outputs can drift along residual-normalization symmetries without changing the normalized forward map.

Figure 10 (right) shows the end-of-training width dependence of these symmetry-affected quantities. Figure 14 shows their corresponding training-time trajectories, together with LM-head logits as a scale-sensitive reference.

Embedding and residual-block output norms grow with width and training time. We do not interpret this growth as evidence that LRMs fail to learn width-dependent forward scales. In pre-normalization architectures, subsequent RMSNorm layers remove the absolute residual scale, so these quantities can drift along a symmetry direction without changing the normalized forward map.

The MLP gate activation is intermediate. Its absolute scale matters because it is passed through the nonlinearity $\mathrm{SiLU}(\cdot)$. However, for large positive inputs, $\mathrm{SiLU}(\cdot)$ behaves close to the homogeneous $\mathrm{ReLU}(\cdot)$, making the gate partially subject to the same scale ambiguity as residual-stream quantities. For this reason, we treat MLP gate scaling as less clean evidence than LM-head logits, attention `QK`, and SSM `dt`.

G MLP experiment

This appendix presents the MLP experiment that complements the LM head experiment in section 2.1. We first describe the experimental setup, summarize the high-level loss behavior used in the main

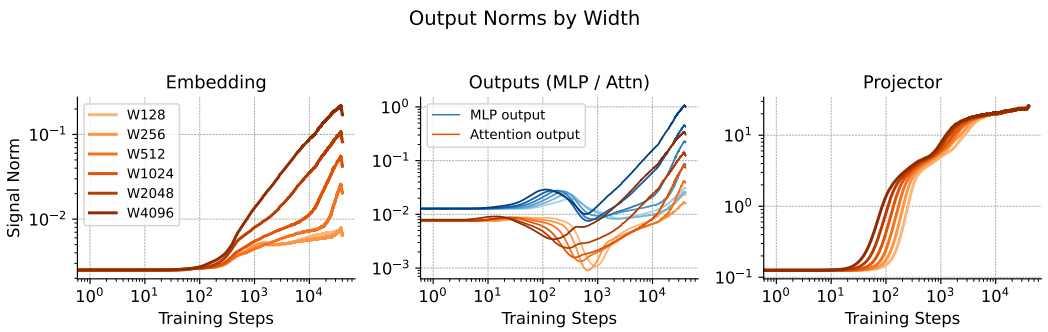


Figure 14: **Training-time width-scaling diagnostics.** *Left:* embedding output norms grow with training time and width, consistent with drift along the residual-normalization symmetry. *Center:* MLP and attention residual outputs exhibit analogous late-training growth. *Right:* LM-head logit norms initially vary with width but approach a similar scale later in training, providing a cleaner scale-sensitive coordinate check.

paper, and then present a detailed analysis of internal activation and parameter norms as we vary the norm scale $S(\eta_{MLP}, \lambda_{MLP}) = \sqrt{\frac{\eta_{MLP}}{\lambda_{MLP}}}$ of the MLP matrix layers.

G.1 Setup and high-level result

Following the same logic as the LM head experiment, we vary the norm scale $S(\eta_{MLP}, \lambda_{MLP})$ only for the 3 matrix layers of the gated MLP block. To test the ability of MLP matrix layers to adapt their scale we consider two configurations:

1. Frozen RMSNorm weights across all backbone layers to restrict the scale adaptation ability of all the backbone blocks.
2. Same as above, but with scalar learnable multipliers added to MLP block to compensate for varied norm scale $S(\eta_{MLP}, \lambda_{MLP})$ of its matrix layers.

Figure 15 (bottom right) shows that configurations without learnable multipliers suffer from loss degradation at larger norm scales $S(\eta_{MLP}, \lambda_{MLP})$ of MLP matrices. We attribute this degradation to output's magnitude mismatch between MLP and attention/SSM blocks, which we investigate in detail below. Importantly, we perform the MLP experiment for both Adam and Muon optimizers and observe essentially identical behavior between the two optimizers. This suggests that the noise-WD equilibrium trap is not a specific property of Adam but a general phenomenon persisting across different parameter update rules.

G.2 Detailed setup

Let us repeat and complement the brief description above. First, we set RMSNorm weights $c_j \to 1$ in (18) for all the MLP, attention, and SSM blocks to restrict the scale adaptation ability of these blocks. Then, we vary the norm scale of MLP block matrices in the following way: for each of $W^{\text{up}}, W^{\text{gate}}, W^{\text{down}}$ (see (19)) we change learning rate η and λ while keeping $\eta\lambda = \text{const}$. We denote $\eta_{MLP} = \eta/\eta_0$ and $\lambda_{MLP} = \lambda/\lambda_0$ the relative change of learning rate and weight decay, which are kept the same for all 3 MLP layers[1]. In Figures 15 and 16, we use these relative values to focus on the change of the matrix norms[2]. The above fully describes our baseline configuration, and in the configuration with learnable MLP multipliers, we add three scalar multipliers $s^{\text{up}}, s^{\text{gate}}, s^{\text{down}}$

[1]The current absolute values of η, λ, as well as the baseline absolute values η_0, λ_0 might be different between $W^{\text{up}}, W^{\text{gate}}$, and W^{down} projections. This is the case for our experiment, where we use tuned μP multipliers from Zuo et al. (2025) with slightly different multipliers for the three MLP projections.

[2]Same conventions apply for LM head experiment in figure 1.

to the three MLP matrix layers [3]. These scalar multipliers had a fixed learning rate $\eta = 10^{-2}$ and no weight decay. Finally, we run two versions of the experiment with Muon and Adam optimizers, while keeping all the other settings identical.

G.3 Behavior of parameter and output norms

With the experiment settings settled, we proceed to a discussion of the behavior of parameter and output norms as MLP matrix norm scaled $S(\eta_{MLP}, \lambda_{MLP})$ is varied. First, we see a clear picture on figures 15 and 16 for Adam and Muon **configurations with learnable multipliers**:

1. The norms of MLP matrices follow the equilibrium scale $S(\eta_{MLP}, \lambda_{MLP}) = \sqrt{\frac{\eta_{MLP}}{\lambda_{MLP}}}$.
2. Yet, the outputs of all residual blocks stay constant regardless of the MLP matrices scale.
3. The constant level of residual blocks output is achieved thanks to MLP multiplier compensating the change in scale of MLP matrices (bottom left plot of figure 15).

This confirms the main thesis of the current work: the ability of learnable multipliers to freely adjust their scale in order to compensate a (presumably) suboptimal scale of its respective matrix layers, which is trapped in noise-WD equilibrium (2).

Next, we look at the **configurations without learnable multipliers** which turned out to be more nuanced. As we have restrained the scale adaptation ability of residual blocks by freezing the respective RMSNorm weights, pure equilibrium norm scaling (2) would create an imbalance between scales of MLP and attention/SSM blocks. This imbalance is expected to significantly degrade the model performance, which translates in a loss gap between configuration with and without learnable multipliers, seen on figure 15 (bottom right). However, this imbalance also creates a significant optimization pressure to balance back the scales of MLP and attention/SSM blocks, and the model manages to align the scales of the residual blocks as can be seen in the top row of figure 15. This scale adaptation is achieved via accumulating several mechanism which we list below.

- **MLP.** On fig. 15 (middle left) we see that the norms of MLP matrix layers slightly deviate noise-WD scaling (2). We interpret this deviation as a result of optimization pressure to reduce the scale gap between MLP and attention/SSM blocks. This introduces a strong enough gradient signal force that modifies the equilibrium (2) which was governed by gradient noise and WD forces only.
- **Attention.** We observe on fig. 15 (top row, third column) the growth of the outputs of attention blocks with $S(\eta_{MLP}, \lambda_{MLP})$. To estimate the norm of the attention block outputs, we may ignore the attention scores and approximate the norm as $\|W^{\text{out}} W^V \mathbf{x}\|$ (see (20)). One way to increase this norm is to increase alignment between W^{out} and W^V, and also between W^V and $\mathbf{x}$. Another way is for W^{out} or W^V to escape noise-WD equilibrium, which is confirmed by the growth of the respective matrix norms on fig. 15 (middle row, third column). We expect that value matrix is W^V is more prone to escaping noise-WD equilibrium in group-query attention with large ratios of Q heads to KV heads: sharing of the KV heads results into more gradient signal coming through the matrix, resulting in higher signal-to-noise ratio. We suspect all these three mechanisms to increase the attention output to take place. Yet, from a slower growth of the attention outputs compared to SSM outputs we may conclude that employing these mechanisms negatively impacts the quality of attention blocks.
- **SSM.** Surprisingly, the outputs of SSM blocks grow at the same rate as outputs of MLP block while the norms of SSM matrices stay constant, in agreement with their noise-WD equilibrium values (fig. 15 top and middle row, second column). After noticing this, we explored components of SSM block and identified two parts: `conv1d` (see (21),(22)) and SSM skip connection scale `D` that in fact play a role of learnable multipliers as they have

[3] Although the scalar multipliers s^{up} and s^{down} are redundant due to multiplicative symmetry, we keep both for simplicity. Also, for the short training duration of MLP experiment, we did not observe any symmetry-induced instabilities discussed in appendix E.1.

[4] For simplicity, we follow the implementation in our codebase and report the norm of the merged QKV matrix. Same applies to the merged W^{XZBCdt} projection matrix of the SSM block

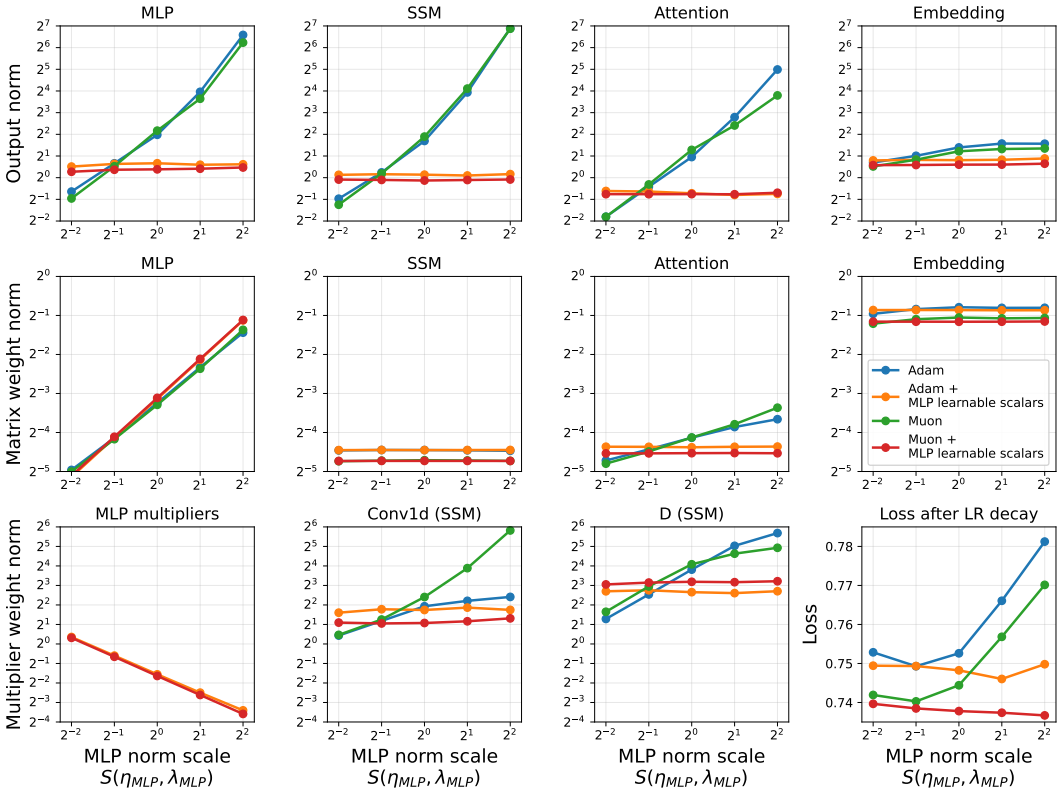


Figure 15: All the subplots, except bottom right, show behavior of different norms as the MLP norm scale $S(\eta_{\text{MLP}}, \lambda_{\text{MLP}})$ is varied. The rows contain similar types of norms and share y-axis scale for easier comparison. The bottom-right subplot shows training loss for the four configurations: Adam and Muon, each with and without learnable multipliers.

(Top row): The norms of the output of MLP, attention and SSM residual blocks, as well as the norm of the outputs of embedding layer. The norms are averaged across the tokens in a batch. Then, we apply geometric average to aggregate residual block norms across model layers.

(Middle row): The norms of matrix weights: $W^{\text{up}}, W^{\text{gate}}, W^{\text{down}}$ for MLP (see (19)), W^{QKV} and W^{out} for attention (see (20)), W^{XZBCdt} and W^{out} for SSM (see (21),(22),(23),(25),(26)), and the embeddings matrix. To aggregate the weight norms into a single metric for residual blocks, in addition to geometric averaging across the layers we further apply geometric averaging across the matrix types within the block, for example $W^{\text{up}}, W^{\text{gate}}, W^{\text{down}}$ for MLP.

(Bottom row): The norms of various non-matrix parameters that are free from the noise-WD equilibrium and can adjust their scale. For the three MLP multipliers we again apply geometric average; Conv1d acts as a row multiplier for W^{XBC} in the SSM block (see (21),(22)); D parameter scales the skip connection in the recurrent SSM computation, such that larger values of D make SSM block computation closer to gated MLP.

vector-like shapes and hence not subject to noise-WD equilibrium. Indeed on fig. 15 (bottom row, second and third columns) we observe the growth of these parameters, explaining the growth of SSM outputs at fixed norm of the respective matrix layers.

- **Embedding.** The norm of embedding matrix stays constant, following its fixed equilibrium value (fig. 15, middle right), while the embedding outputs slightly grows to catch up with residual blocks outputs. To explain this growth, we note that the embedding output norm is an average norm of token embedding vectors weighted with frequency of each token in the data. Then, most frequent tokens display vector-like behavior with high signal-to-noise ratio in the gradients which allows them to partially escape noise-WD equilibrium and adjust

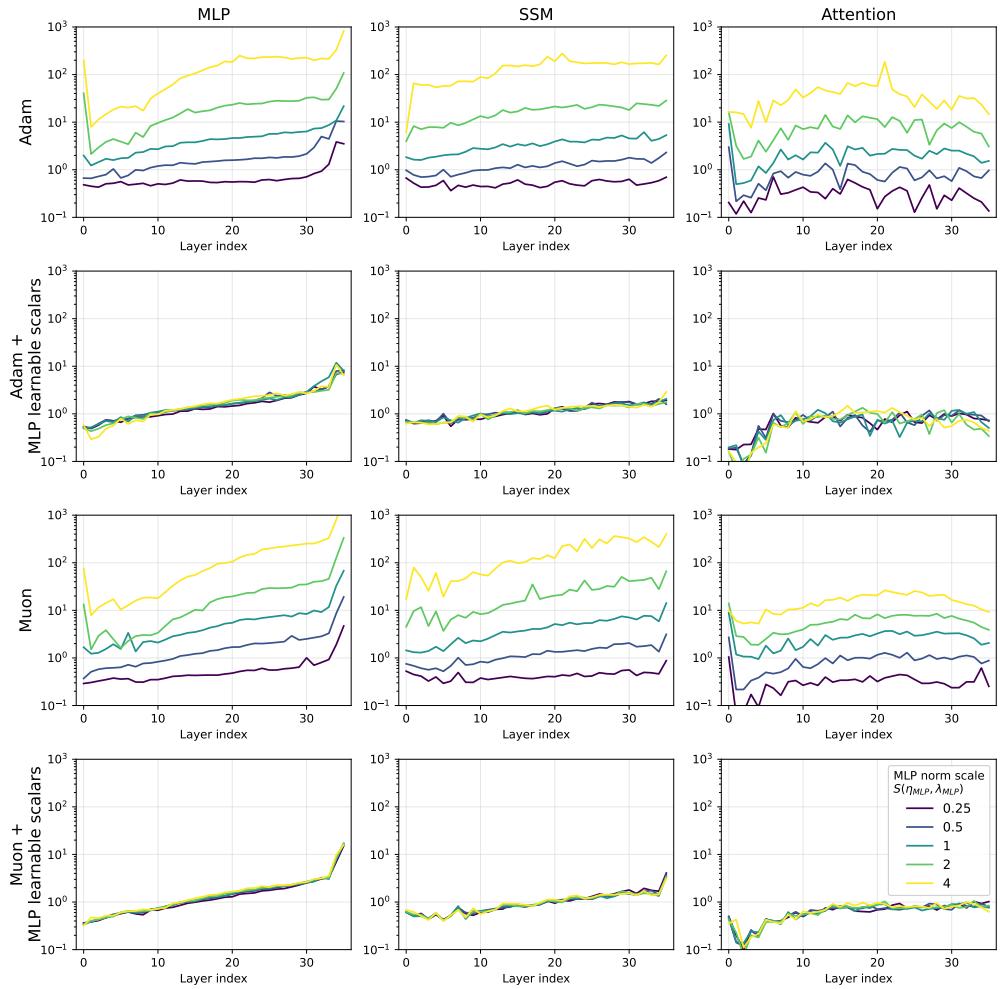


Figure 16: The behavior the norms of the MLP, attention and SSM residual blocks outputs across the model layers. This figure complements figure 15, where the norms were averaged across the layers. Overall, we observe strong layer-specific patterns, suggesting that the layer-averaged metrics in figure 15 adequately capture the behavior of the norms with respect to MLP scale $S(\eta_{MLP}, \lambda_{MLP})$.

their norm. On the other hand, low frequency tokens have low signal-to-noise ratio and therefore get trapped in the noise-WD equilibrium.

Finally, let us comment on the differences and similarities of Adam and Muon optimizers in the observed behaviors. In almost all the considered aspects, two optimizers behave identically, suggesting that noise-WD equilibrium mechanism could be a general phenomenon applicable to wide range of optimizer update rules. The cases where we observe a moderate difference between Adam and Muon include the growth of attention, `conv1d` and `D` norms.

Remark on experiment design. As we have seen above, the configuration without learnable multipliers manage to adapt the scale of attention and SSM blocks. This creates a complex picture of behaviors to ensure the growth of attention/SSM outputs, failing our original intent in restricting scale adaptation ability of these blocks to produce a clean separation between configurations with and without learnable multipliers.

A few choices in the design of this MLP experiment may help to satisfy the original intention in the restricting the attention/SSM scale adaptability. For attention, we can switch from grouped query attention (GQA) to multi-head attention (MHA): this decreases the signal-to-noise ratio in value matrices, making it harder for them to escape the equilibrium. For SSM, we can activate the internal

RMSNorm[5], as in (25), to normalize the gated branch before its output projection. This constrains its aggregate scale, while preserving the effects of `conv1d` and `D` on relative features.

H ARCHITECTURE AND MULTIPLIER PLACEMENT

We specify the multiplier locations used in the placement ablation and the SSM quantities studied in the depth and width experiments. The ablation uses a dense Llama model; the SSM discussion describes the Mamba-2 branch of our Falcon-H1 experiments. Training settings and uncertainty estimates for the ablation are in Appendix C.1.

For each matrix $W \in \mathbb{R}^{\text{out}\times\text{in}}$, we use the effective matrix $\overline{W}$ from (1). A scalar multiplies the whole projection; row and column vectors give $\overline{W} = \text{diag}(r^W)W\,\text{diag}(c^W)$. Missing factors are set to one. Rows scale output features and columns scale input features, with the embedding lookup convention described below. Multipliers at different matrices and layers are independent unless stated otherwise.

H.1 EMBEDDING AND LM HEAD

We store the untied embedding E and LM-head matrix W^{head} in vocabulary $\times$ hidden order. For token v, the embedding lookup returns $e_j = \overline{E}_{vj}$. With vector multipliers this is $r^E_v E_{vj} c^E_j$: rows scale vocabulary entries and columns scale hidden output channels. With a scalar multiplier it is $s^E E_{vj}$. The head instead maps the final hidden features x_j to logits, $\ell_v = \sum_j \overline{W}^{\text{head}}_{vj} x_j$, so its columns scale hidden inputs and its rows scale vocabulary outputs.

The final RMSNorm gain is itself a column multiplier for the head. This gain is frozen at one in the cumulative placement ablation. Stage 7 replaces the head's scalar by a hidden-input column vector, and stage 9 adds a vocabulary-output row vector. Thus these two stages adapt different dimensions. When both the final RMSNorm gain and head column multipliers are learned, their product determines the effective column scale.

H.2 ATTENTION AND GATED MLP BLOCKS

Each block receives normalized residual features. For a residual $\mathbf{z} \in \mathbb{R}^d$, we write

$$x_j = \big(\text{RMSNorm}(\mathbf{z})\big)_j = \frac{\gamma_j z_j}{\sqrt{d^{-1}\sum_k z_k^2 + \epsilon_{\text{norm}}}}. \tag{18}$$

The gain γ is shared by all projections fed by this normalization. Independent column multipliers on those projections allow different input-feature scales, and therefore provide more freedom than a shared gain. In the ablation, $\epsilon_{\text{norm}} = 10^{-5}$ and $\gamma = 1$ except in the learned-RMSNorm controls. The normalization symmetry discussed in Appendix E.1 is exact when the stabilizer is omitted and approximate when it is nonzero.

Gated MLP. The MLP contribution to the residual is

$$\mathbf{y} = \overline{W}^{\text{down}}\left[\text{SiLU}(\overline{W}^{\text{gate}}\mathbf{x}) \odot (\overline{W}^{\text{up}}\mathbf{x})\right]. \tag{19}$$

Gate and Up have shape FFN $\times$ hidden, while Down has shape hidden $\times$ FFN. Gate rows change the input to SiLU; Up rows and Down columns scale the same intermediate channels on either side of an elementwise product. With the other factors fixed, their effect depends only on $r^{\text{up}}_k c^{\text{down}}_k$. Consequently, the Down-column vector added at stage 8 can be absorbed into the Up-row vector already present at stage 5.

Attention. Let t be the query position, h a query head, and $g(h)$ its shared KV head in grouped-query attention (Ainslie et al., 2023). With R_t denoting the rotary position transformation and d_h the

[5]The reason we did not have this RMSNorm in our experiment is absence of it in Falcon-H1-0.5B architecture used throughout the work.

head dimension, the attention contribution is

$$
\begin{aligned}
Q_{t,h} &= R_t \overline{W}^{Q,h} \mathbf{x}_t, & K_{t,g} &= R_t \overline{W}^{K,g} \mathbf{x}_t, \quad V_{t,g} = \overline{W}^{V,g} \mathbf{x}_t, \\
a^h_{tt'} &= \mathrm{Softmax}_{t'}\left(\frac{Q_{t,h}^\top K_{t',g(h)}}{\sqrt{d_h}} + M_{tt'} \right), \\
\mathbf{y}_t &= \sum_h \overline{W}^{O,h} \sum_{t'} a^h_{tt'} V_{t',g(h)}.
\end{aligned}
\tag{20}
$$

Here $M_{tt'} = 0$ for $t' \le t$ and $-\infty$ otherwise. Q/K/V head matrices are row slices of their full effective projections; each projection has one column vector shared across its head slices. $W^{O,h}$ is the column slice of the output matrix acting on head h. We denote this output matrix by O in the ablation and by W^{out} elsewhere. A row multiplier on V rescales a value channel and can be absorbed into the matching columns of O. Under GQA, the same factor is repeated in O for every query head using that KV head. O columns can also learn different scales for those query heads, so they include all V-row rescalings. This is the second redundancy introduced at stage 8.

These redundancies concern the forward map at fixed bare matrices: stage 8 does not enlarge the family of maps available through the existing multipliers, but it changes their training parametrization. Earlier stages still have other symmetries, including the common reciprocal scaling of row and column factors. Q/K scalar factors also admit reciprocal rescaling. Arbitrary Q/K row vectors, however, act before RoPE and do not generally commute with its rotations; the same argument does not justify removing all K-row multipliers.

H.3 Configurations in the placement ablation

Table 9 specifies the ten cumulative configurations C_0–C_9 in Figure 7(a), where C_n denotes stage n. We write I for no multiplier, S for a scalar, R for rows, C for columns, and RC for both vectors. At a scalar-to-vector transition, the vector replaces the scalar. All factors initialize to one. A stage denotes a topology, not a continuation of a previous run: every configuration is trained independently from initialization.

Stage	Embed.	Q	K	V	O	Gate	Up	Down	Head
0	I	I	I	I	I	I	I	I	I
1	S	I	I	I	S	I	I	S	S
2	S	I	S	I	S	S	I	S	S
3	S	C	C	C	S	C	C	S	S
4	C	C	C	C	R	C	C	R	S
5	C	RC	RC	C	RC	RC	RC	R	S
6	RC	RC	RC	C	RC	RC	RC	R	S
7	RC	RC	RC	C	RC	RC	RC	R	C
8	RC	RC	RC	RC	RC	RC	RC	RC	C
9	RC	RC	RC	RC	RC	RC	RC	RC	RC

Table 9: Cumulative multiplier placements. For $W \in \mathbb{R}^{\mathrm{out} \times \mathrm{in}}$, I means no multiplier, S a scalar, R a row vector (one entry per output), C a column vector (one per input), and RC both. Embedding and head matrices have shape vocabulary $\times$ hidden, so their row and column vectors control vocabulary and hidden dimensions, respectively. Stages 0–9 correspond to configurations B01–B10 in the released data.

Stages 1–5 also have individual-addition experiments, listed in Table 10. Each starts from the preceding cumulative topology and makes only the indicated change; their union gives the next cumulative stage. A grouped addition, such as Q/K/V columns, uses a separate vector for each matrix. Scalars are also separate at each layer; in particular, the stage-2 K scalar is shared across that layer's attention heads, rather than being a per-head parameter. The ten cumulative and 15 individual-addition configurations all freeze RMSNorm gains at one.

Learned-RMSNorm controls. Four further configurations learn all RMSNorm gains: (i) gains alone, (ii) gains plus separate O and Down scalars, (iii) gains plus O and Down row vectors, and

Stage	Parent stage	Separate additions tested in Figure 7(b)
1	0	Embedding I→S; O I→S; Down I→S; Head I→S.
2	1	K I→S; Gate I→S.
3	2	Q/K/V →C together; Gate/Up →C together. Existing K and Gate scalars are replaced.
4	3	Embedding S→C; O S→R; Down S→R.
5	4	Q/K C→RC together; O R→RC; Gate C→RC; Up C→RC.

Table 10: The 15 individual additions. Semicolon-separated entries are separate training configurations; matrices joined by a slash change together within one configuration. Each comparison uses its parent stage, not the other individual additions.

(iv) gains plus the stage-8 LRMs. Configurations (ii) and (iii) use the residual-scale locations of ReZero and LayerScale, respectively, rather than reproducing their full recipes: we retain RMSNorm and initialize the added factors at one. Configuration (iv) matches stage 8, not stage 9, so it has no head vocabulary vector. Its RMSNorm gains can be absorbed into the existing projection columns, including those of the head; here again, extra parameters change the training parametrization without enlarging the available forward maps.

H.4 Mamba-2 and the SSM time-step projection

The Falcon-H1 experiments also contain Mamba-2 blocks (Dao & Gu, 2024; Zuo et al., 2025). We give their computation to identify the SSM `dt` projection and the scales discussed in Sections 3.1 and 3.3. Let $\mathbf{x}_t$ denote the normalized block input. The projected branches are

$$\mathbf{X}_t = \mathrm{SiLU}\Big(\mathrm{conv1d}(\overline{W}^{X}\mathbf{x})_t\Big), \qquad \mathbf{Z}_t = \mathrm{SiLU}(\overline{W}^{Z}\mathbf{x}_t), \tag{21}$$

$$\mathbf{B}_t = \mathrm{SiLU}\Big(\mathrm{conv1d}(\overline{W}^{B}\mathbf{x})_t\Big), \qquad \mathbf{C}_t = \mathrm{SiLU}\Big(\mathrm{conv1d}(\overline{W}^{C}\mathbf{x})_t\Big), \tag{22}$$

$$p^{dt}_{t,h} = (\overline{W}^{dt}\mathbf{x}_t)_h, \qquad \Delta_{t,h} = \mathrm{Softplus}(p^{dt}_{t,h} + b_{dt,h}). \tag{23}$$

Here `conv1d` denotes the learned depthwise causal convolution, including its bias where present. X, B, and C pass through convolution and SiLU, Z supplies an elementwise gate, and the `dt` branch directly produces one positive time step $\Delta_{t,h}$ per SSM head after bias and softplus.

For a head of width P and state size N, let $X_{t,h} \in \mathbb{R}^P$, $B_{t,g(h)}, C_{t,g(h)} \in \mathbb{R}^N$, and $H_{t,h} \in \mathbb{R}^{N\times P}$. Here $g(h)$ identifies the group of heads sharing B and C. The recurrence and its output are

$$\begin{aligned} H_{t,h} &= e^{A_h\Delta_{t,h}}H_{t-1,h} + \Delta_{t,h}B_{t,g(h)}X_{t,h}^{\top}, \qquad A_h = -e^{a_h} < 0, \\ u_{t,h} &= H_{t,h}^{\top}C_{t,g(h)} + D_hX_{t,h}. \end{aligned} \tag{24}$$

The learned D_h scales a bypass around the recurrent state. For fixed A_h, increasing $\Delta_{t,h}$ reduces retention of the previous state and increases the coefficient of the new input. Thus the `dt` projection affects both memory and input injection.

Concatenating the head outputs gives $\mathbf{u}_t$. In the Falcon-H1-0.5B experiments we use $\mathbf{F}_t = \mathbf{u}_t \odot \mathbf{Z}_t$, without internal RMSNorm. An optional normalized variant is

$$\mathbf{F}_t = \mathrm{RMSNorm}(\mathbf{u}_t \odot \mathbf{Z}_t). \tag{25}$$

Some implementations instead normalize $\mathbf{u}_t$ before gating. In each case, the block output is

$$\mathbf{y}_t = \overline{W}^{\mathrm{out}}\mathbf{F}_t. \tag{26}$$

Multiplier locations and plotted quantities. Equation (1) applies separately to each projection above. On W^{dt}, a scalar rescales the whole projected vector and a row multiplier rescales individual heads, both before bias and softplus. This controls the input-dependent part of the time step; it is distinct from learning b_{dt} or A. The SSM `dt` activation in our width diagnostics is p^{dt}, measured immediately after the effective linear projection, before bias and softplus. The `dt` multiplier curves instead report the multiplier itself, with the norm and layer aggregation specified by the respective

plots. Neither quantity is the positive time step Δ, and a larger projection or multiplier norm need not imply a larger Δ.

Other native scales help interpret the SSM measurements in Appendix G: the per-channel convolution can rescale and temporally mix the X/B/C projections before SiLU, and D can amplify the bypass. These operations remain available when no explicit LRMs are added. Their presence motivates examining the full branch computation rather than inferring its output scale from projection-matrix norms alone.

I ZIPF MODEL: FROZEN RESPONSES AND SCALE DYNAMICS

We first calculate how Adam transforms a stationary gradient stream when the model parameters are fixed. This gives rigorous statements about restoring signals, accumulated noise, and wrong-way responses. We then use these coefficients to interpret the training dynamics: the local diffusion and WD balances below are heuristic descriptions of moving parameters, supported by the simulated trajectories. This separates the part we can establish directly from the dynamical picture it suggests. Throughout, $\beta_1 = \beta_2 = \beta \in (0, 1)$, $\alpha = 1 - \beta$, and $\epsilon = 0$. In this appendix we use ordinary loss gradients, opposite to Section 2 and Appendix J. Accordingly, g, M, and $R = M/\sqrt{V}$ have the opposite signs, while the additive drift $b = -\mathbb{E}R$ and integrated covariance Γ are unchanged. The same convention applies to the scalar gradients below.

I.1 SETUP AND THE EXACT FROZEN RESPONSE

Fix a class probability $p \in (0, 1)$ and freeze its predicted probability $q \in [0, 1]$. Independent batches give $X_k = N_k/B$ with $N_k \sim \text{Binomial}(B, p)$ and $g_k = q - X_k$. The stationary optimizer moments are exponentially weighted moving averages (EWMAs) of a two-sided iid gradient stream:

$$M_k = \alpha \sum_{j \geq 0} \beta^j g_{k-j}, \qquad V_k = \alpha \sum_{j \geq 0} \beta^j g_{k-j}^2. \tag{27}$$

Define

$$A_k = \alpha \sum_{j \geq 0} \beta^j X_{k-j}, \qquad C_k = \alpha \sum_{j \geq 0} \beta^j X_{k-j}^2, \qquad \Delta_k = C_k - A_k^2 \geq 0. \tag{28}$$

The inequality follows from Jensen's inequality for the weights $\alpha\beta^j$, which sum to one. Substitution gives the exact identities

$$M_k = q - A_k, \qquad V_k = q^2 - 2qA_k + C_k = (q - A_k)^2 + \Delta_k, \tag{29}$$

so

$$R_k(q) = \frac{q - A_k}{\sqrt{(q - A_k)^2 + \Delta_k}}, \qquad |R_k(q)| \leq 1. \tag{30}$$

These identities hold at frozen q. In moving training, historical gradients contain q_{k-j}; for example, $M_k = \alpha \sum_j \beta^j q_{k-j} - A_k$, which generally differs from $q_k - A_k$.

The relevant moments are

$$\mu = q - p, \qquad \sigma^2 = \frac{p(1-p)}{B}, \qquad s_2 = \mu^2 + \sigma^2. \tag{31}$$

Write $\kappa_j = \mathbb{E}(g - \mu)^j$ and $\tau_j = \kappa_j/\sigma^j$ for the central moments and their standardized versions. In particular, κ_4 is the fourth central moment, not the fourth cumulant. The binomial formulas yield

$$\tau_3 = -\frac{1 - 2p}{\sqrt{Bp(1-p)}}, \qquad \tau_4 = 3 + \frac{1 - 6p(1-p)}{Bp(1-p)}. \tag{32}$$

The probability coordinate and softmax invariance. If all logits except the selected one are fixed, write $x = z - z^*$, where $q(z^*) = p$. The exact relation is

$$q(z) = \frac{pe^x}{1 - p + pe^x}, \qquad \frac{q - p}{\sigma} = \frac{\sqrt{Bp(1-p)}(e^x - 1)}{1 - p + pe^x}. \tag{33}$$

The simpler $q = pe^x$ holds when the total partition is treated as an external constant; it is a rare-class local approximation, not a globally valid probability model. In the full softmax, adding a common constant to every logit leaves the loss unchanged. We therefore use contrasts or centered logits when comparing norm changes with the population restoring direction (Theorem I.5).

I.2 Integrated noise and the local diffusion approximation

For a stationary frozen response with absolutely summable covariances, define

$$b(q) = -\mathbb{E}R_0(q), \qquad \Gamma(q) = \sum_{h\in\mathbb{Z}} \mathrm{Cov}(R_0(q), R_h(q)). \tag{34}$$

The variance of its accumulated response is exactly

$$\mathrm{Var}\left(\sum_{k=1}^{N} R_k\right) = \sum_{|h|<N} (N-|h|)\mathrm{Cov}(R_0, R_h), \tag{35}$$

and division by N gives Γ in the limit by dominated convergence. Thus the noise relevant over many optimizer steps is the integrated covariance, rather than the one-step variance.

Lemma I.1 (Integrated covariance of an EWMA)**.** *Let ε_k be iid, centered, with variance $v < \infty$, and let $\phi_k = \alpha \sum_{j\geq 0} \beta^j \varepsilon_{k-j}$. Then*

$$\mathrm{Cov}(\phi_0, \phi_h) = \frac{\alpha}{1+\beta} v\beta^{|h|}, \tag{36}$$

$$\sum_{h\in\mathbb{Z}} \mathrm{Cov}(\phi_0, \phi_h) = v. \tag{37}$$

The same covariance identity holds for two EWMAs of jointly iid innovations, with their innovation covariance in place of v.

Proof. Expanding the two EWMAs leaves only matching innovation times, giving (36). Summing the geometric series gives (37); the cross-covariance calculation is identical. □

The same covariance formula gives $\mathrm{Var}(\sum_{k=1}^{N} \phi_k) = Nv + O(v/\alpha)$, so the window must cover many optimizer memory lengths, $N\alpha \gg 1$, to recover the integrated noise level.

We build the local diffusion in (4) from these frozen means and integrated covariances, with time $t = \eta k$ and noise amplitude $\sqrt{\eta}$. The intended separation of scales is that we average over many optimizer memory lengths before the model changes appreciably. This provides a useful dynamical interpretation, but does not by itself prove a diffusion limit for moving Adam: that would also require control of the changing gradient law and a limit theorem for the nonlinear response. In particular, temporal correlations matter even when the one-step response variance is small.

I.3 Memory-filled response: drift and linearized noise

The memory is filled with class occurrences when $Bp/\alpha \gg 1$. Independence gives

$$\mathbb{E}V = s_2, \qquad \frac{\mathrm{Var}(V)}{s_2^2} = \frac{\alpha}{2-\alpha}\left(\frac{\mathbb{E}g^4}{s_2^2} - 1\right). \tag{38}$$

For an underpredicted rare token, write $q = cp$ with fixed $0 < c < 1$. As $Bp \to 0$, $s_2 \sim p/B$ and $\mathbb{E}g^4 \sim p/B^3$, giving

$$\mathbb{E}V \simeq \frac{p}{B}, \qquad \frac{\mathrm{Var}(V)}{(\mathbb{E}V)^2} \simeq \frac{\alpha}{(2-\alpha)Bp}. \tag{39}$$

Thus $Bp \gg \alpha$ makes V concentrate around its mean, quantifying the memory-window argument in Section 2.2. We retain its leading fluctuations when calculating the accumulated noise below.

Write $M = \mu + m$, $V = s_2 + v$, where m and v are centered EWMAs. The first-order response is

$$R_{\mathrm{lin}} = \frac{\mu}{\sqrt{s_2}} + \frac{m}{\sqrt{s_2}} - \frac{\mu v}{2s_2^{3/2}}. \tag{40}$$

Consequently, with $a = \mu/\sigma$, its additive drift is

$$b_{\mathrm{lin}}(q) = -\frac{a}{\sqrt{1+a^2}} = -\frac{q-p}{\sqrt{(q-p)^2+\sigma^2}}. \tag{41}$$

We control the error in the mean response with a bound that also applies to non-binomial gradient laws.

Lemma I.2 (Mean-response bound)**.** *For iid gradients (G_t) with mean μ, second moment $m_2 > 0$, and finite fourth moment, define*

$$M_t = (1-\beta)\sum_{j\geq 0}\beta^j G_{t-j}, \qquad V_t = (1-\beta)\sum_{j\geq 0}\beta^j G_{t-j}^2, \qquad R_t = M_t/\sqrt{V_t},$$

with $R_t = 0$ if $V_t = 0$. Then

$$\left|\mathbb{E}R_t - \frac{\mu}{\sqrt{m_2}}\right| \leq \sqrt{\frac{1-\beta}{1+\beta}\left(\frac{\mathbb{E}G_t^4}{m_2^2} - 1\right)}. \tag{42}$$

Proof. Weighted Cauchy–Schwarz gives $M_t^2 \leq V_t$, hence $|R_t| \leq 1$. Thus, also when $V_t = 0$,

$$\left|R_t - \frac{M_t}{\sqrt{m_2}}\right| = |R_t|\left|1 - \sqrt{V_t/m_2}\right| \leq |V_t/m_2 - 1|.$$

Independence gives $\mathbb{E}M_t = \mu$, $\mathbb{E}V_t = m_2$, and $\mathrm{Var}(V_t) = \frac{1-\beta}{1+\beta}(\mathbb{E}G_t^4 - m_2^2)$. Taking expectations and applying Cauchy–Schwarz proves the result, without expanding or lower-bounding the random denominator. □

Applying the lemma with $G = g$ and $m_2 = s_2$ proves $b \to b_{\mathrm{lin}}$ as $\alpha \to 0$ at fixed p, B, q. In joint rare-event limits, resolving a small nonzero drift requires the error to be smaller than that drift; the general results in App. J give a sharper criterion.

Integrated covariance of the linearization. The centered part of (40) is the EWMA of

$$\psi(g) = \frac{g-\mu}{\sqrt{s_2}} - \frac{\mu(g^2 - s_2)}{2s_2^{3/2}}.$$

Lemma I.1 therefore gives its integrated covariance exactly as $\mathrm{Var}\psi(g)$. Let $U = (g-\mu)/\sigma$. Algebra gives

$$\psi(g) = \frac{U - \frac{a}{2}(U^2-1)}{(1+a^2)^{3/2}},$$

hence

$$\Gamma_{\mathrm{lin}}(a) = \frac{1 - a\tau_3 + \frac{1}{4}a^2(\tau_4 - 1)}{(1+a^2)^3}. \tag{43}$$

Its numerator is a variance, so it is nonnegative. At $q = p$, $\Gamma_{\mathrm{lin}} = 1$. These are exact identities for the linearized response, not equalities for finite-memory Adam. Proposition J.5 gives $b \to b_{\mathrm{lin}}$ and $\Gamma \to \Gamma_{\mathrm{lin}}$ as $\alpha \to 0$ for fixed p, B, q. We use this fixed-law limit before taking the rare-class limits below. A uniform joint limit requires additional control of the nonlinear remainder and its temporal correlations; a pointwise Taylor bound alone does not suffice. Proposition J.2 establishes the coarser signal-suppression conclusion directly for the actual ratio under its stated joint-limit conditions.

I.4 Region I: dense memory-filled response

When $Bp(1-p) \gg 1$, the standardized binomial moments approach $\tau_3 = 0$, $\tau_4 = 3$. The linearized coefficient becomes

$$\Gamma_{\mathrm{lin}}^{\mathrm{G}}(a) = \frac{a^2+2}{2(1+a^2)^3}. \tag{44}$$

It is order one near $a = 0$ and decays as $1/(2a^4)$ for large $|a|$. Near the matched point, (33) gives $a \simeq \rho(z - z^*)$ with $\rho = \sqrt{Bp(1-p)}$. The drift is therefore $-\rho(z - z^*)$, and replacing the local noise by its value one gives the Ornstein–Uhlenbeck approximation

$$dZ_t = [-\rho(Z_t - z^*) - \lambda Z_t]dt + \sqrt{\eta}\, dW_t.$$

For $\rho + \lambda > 0$ its stationary distribution has

$$\mathbb{E}Z = \frac{\rho}{\rho+\lambda}z^*, \qquad \mathrm{Var}(Z) = \frac{\eta}{2(\rho+\lambda)}. \tag{45}$$

This local description requires both the mean displacement from z^* and the standard deviation to be small compared with $\min\{1, \rho^{-1}\}$. For weak WD, $\lambda \ll \rho$, the variance approaches $\eta/(2\rho)$.

I.5 Region II: rare memory-filled response

Take rare classes with $Bp \ll 1$ while the memory contains many occurrences. For $q = cp$, $0 \le c \le 1$, substitution of (32) into the linearized coefficients gives, as $Bp \to 0$,

$$b_{\mathrm{lin}} = \sqrt{Bp}(1-c)(1+o(1)), \qquad \Gamma_{\mathrm{lin}} = \frac{(1+c)^2}{4} + o(1). \tag{46}$$

At $c = 1$ the first expression is exactly zero. To obtain these formulas, use $a = -(1-c)\sqrt{Bp}(1 + o(1))$, $\tau_3 \sim -(Bp)^{-1/2}$, and $\tau_4 - 1 \sim (Bp)^{-1}$ in (41)–(43). The variance numerator tends to $1-(1-c)+(1-c)^2/4 = (1+c)^2/4$. This exposes the asymmetry: the useful mean is small, while the linearized integrated noise remains between $1/4$ and 1. For actual Adam these limits hold with $\alpha \to 0$ first at fixed p, B, q, followed by the rare-class limit. Rarity and the ordering $\alpha \ll Bp$ alone do not specify a uniform approximation error relative to a vanishing useful drift.

On the overpredicted side, $a > 0$, the normalized drift approaches -1 as a grows. The noise need not already be small where this happens: at $a = 1$,

$$\Gamma_{\mathrm{lin}}(1) \sim \frac{1}{32Bp}, \tag{47}$$

whereas far into the large-a range it decays as $1/(4Bp\,a^4)$. These follow directly from (43). Thus a restoring overprediction response does not by itself justify treating the boundary as deterministic.

From weak restoring drift to scale control. We now use these local coefficients to build a dynamical picture. On the underpredicted side, a rare coordinate receives order-one accumulated noise but only an $O(\sqrt{Bp})$ restoring drift. We therefore expect broad excursions before that weak drift can return it; the much stronger response on the overpredicted side makes those excursions asymmetric. This motivates the one-sided-diffusion description. It does not require the restoring drift to vanish, and the noise peak in (47) means that the overpredicted side should not be treated as a deterministic wall.

WD adds a restoring contribution $-\lambda Z$. To see the balance, temporarily replace $b(Z)$ and $\Gamma(Z)$ by representative constants b_0 and $\Gamma_0 > 0$ over the range explored. For $\lambda > 0$, the resulting linear diffusion has stationary moments

$$\mathbb{E}Z = \frac{b_0}{\lambda}, \qquad \mathrm{Var}(Z) = \frac{\eta\Gamma_0}{2\lambda}, \qquad \mathbb{E}Z^2 = \frac{b_0^2}{\lambda^2} + \frac{\eta\Gamma_0}{2\lambda}. \tag{48}$$

The squared norm is predominantly set by noise and WD only when $b_0^2 \ll \eta\lambda\Gamma_0$; otherwise its mean can dominate. A locally linear restoring drift instead gives the $\rho + \lambda$ balance in (45).

Applying this approximation to moving parameters requires slowly varying coefficients and decay slower than the response correlations ($\eta\lambda \ll \alpha$ for the linearized EWMA). The ordering $\alpha \ll Bp \ll 1$ alone does not ensure this. Residual drift can still help confine the trajectory, and softmax coupling changes the coefficients. The balance nevertheless explains why suppressing the task's restoring signal makes WD more influential in setting a potentially suboptimal scale; it is not a universal stationary norm formula for the coupled model.

I.6 Region III: event-starved response

For a rare underpredicted token, $0 < q < p$, a sufficiently long interval without a hit leaves $M \simeq q$ and $V \simeq q^2$. Thus $R \simeq +1$ and the additive drift is $b \simeq -1$, opposite to the population descent direction $p - q > 0$. To see when such intervals dominate, choose N with $NBp \ll 1$ and $\beta^N(p/q + \mathbb{E}X^2/q^2) \ll 1$. A union bound makes recent hits unlikely; the older contributions to A, C have means $\beta^N p$ and $\beta^N \mathbb{E}X^2$, so Markov's inequality makes A/q and C/q^2 small with high probability. Equation (30) and $|R| \le 1$ then give the stated mean saturation.

The window must therefore be long enough to forget the old moments relative to the small baseline gradient, yet short enough to contain no hit. Its required length depends logarithmically on the inverse baseline magnitude, so $Bp \ll \alpha$ alone is insufficient for arbitrarily small q. Proposition J.1 proves a general sign-reversal criterion, with the Zipf identification in Appendix J.4.

An exact example with positive momentum. Batch size one gives a short calculation showing both reversal and its limits.

Proposition I.3 (Wrong-way response with positive momentum)**.** *For $B = 1$, $0 < p < 1/2$, and $0 < \beta < 1$, the frozen drift $b(q) = -\mathbb{E}R(q)$ is continuous and strictly decreasing on $[0, 1]$. It has a unique zero $q_\beta < p$, and*

$$q_\beta < q < p \quad \Longrightarrow \quad b(q) < 0 < p - q. \tag{49}$$

Proof. Since $X^2 = X$, we have $C = A$ and

$$R(q) = \frac{q - A}{\sqrt{q^2 + (1 - 2q)A}}, \qquad \partial_q R(q) = \frac{A(1 - A)}{[q^2 + (1 - 2q)A]^{3/2}} > 0.$$

The infinite iid history contains both outcomes almost surely, so $0 < A < 1$. Thus b is strictly decreasing; bounded convergence gives continuity, $b(0) = \mathbb{E}\sqrt{A} > 0$, and $b(1) = -\mathbb{E}\sqrt{1 - A} < 0$. At $q = p$, put $f(a) = [p^2 + (1 - 2p)a]^{-1/2}$. Since $\mathbb{E}A = p$,

$$\mathbb{E}R(p) = -\mathrm{Cov}(A, f(A)) > 0.$$

Indeed, f is strictly decreasing and, for an independent copy A', $2\mathrm{Cov}(A, f(A)) = \mathbb{E}[(A - A')(f(A) - f(A'))] < 0$ because $\mathrm{Var}(A) = p(1 - p)\alpha/(1 + \beta) > 0$. Hence $b(p) < 0$ and $q_\beta < p$. □

For $p = 1/2$ the zero is $q_\beta = p$; reflecting p, q, A about $1/2$ gives the case $p > 1/2$. The positive value of $b(0)$ shows why we cannot extrapolate $b \simeq -1$ to arbitrarily small predictions: at fixed p, β, the drift eventually restores an underpredicted class. Thus $z \simeq -1/\lambda$ is not a universal consequence of rare occurrences. Nor do separate frozen roots determine the full-softmax equilibrium, where a common logit drift leaves the probabilities unchanged.

I.7 Relation to the plotted trajectories

The simulation in Fig. 3 normalizes the partition after every step, so the displayed logits are $z_i = \log q_i$. Their movement is not a common-shift artifact. For the plotted zero-WD run, the exact normalized update is

$$z_i^+ = z_i - \eta_k R_i - \log \sum_j q_j e^{-\eta_k R_j}, \tag{50}$$

where R_i is the bias-corrected response used in the simulation. Replacing the optimizer state by its stationary frozen law gives mean increment $\eta_k(b_i - \sum_j q_j b_j) + O(\eta_k^2)$. The corresponding noise involves the joint projected response, not only the marginal covariance of one coordinate. Pairwise contrasts satisfy the simpler exact update $(z_i^+ - z_j^+) - (z_i - z_j) = -\eta_k(R_i - R_j)$.

The run uses $n = 10^4$, Zipf exponent 1.5, $B = 100$, $\beta = 0.9$, $\eta = 0.04096$ after ten warmup steps, zero WD, and finite-history bias correction. We selected trajectories showing strong restoring behavior, asymmetric fluctuations, and persistent downward drift. Their Bp values are approximately 2.625, 0.0198, and 0.000200. These examples illustrate the three qualitative behaviors, rather than quantitatively testing the asymptotic boundaries. In particular, the middle trajectory has $Bp < \alpha = 0.1$ and is outside the strict memory-filled ordering $\alpha \ll Bp \ll 1$. Its similar appearance therefore does not establish that it is governed by that particular asymptotic approximation. The distinctions between a stationary frozen response and this finite-step, jointly evolving simulation include parameter motion over memory, softmax coupling, initialization, and bias correction; we do not identify a quantitative boundary shift attributable to any one of them.

I.8 Why a separate scale can respond differently

We now compare a shared scalar with coordinate-wise Adam at the same Zipf state. The scalar receives a different gradient statistic, and we show that this statistic can retain the restoring signal that individual rare-token gradients lose. Throughout this subsection, G denotes an ordinary loss gradient, so Adam subtracts its normalized ratio R. We use equal momenta $0 < \beta < 1$, $\epsilon = 0$, iid batches, and stationary optimizer histories with the model held fixed. The scalar and all coordinates may share the same batches.

What the scalar learns. For fixed nonconstant logits z_i, positive label probabilities p_i, and $q_i(s) = e^{sz_i}/\sum_j e^{sz_j}$, the loss satisfies

$$L(s) = \log \sum_i e^{sz_i} - s \sum_i p_i z_i,$$
$$\mu(s) := L'(s) = \mathbb{E}_{q(s)} z - \mathbb{E}_p z, \qquad L''(s) = \mathrm{Var}_{q(s)} z > 0. \tag{51}$$

On an iid batch $Y_1, \dots, Y_B \sim p$, the scalar gradient is

$$G_s = \mathbb{E}_{q(s)} z - \frac{1}{B} \sum_{b=1}^{B} z_{Y_b} = \mu(s) + \xi, \qquad \xi = \mathbb{E}_p z - \frac{1}{B} \sum_{b=1}^{B} z_{Y_b}, \tag{52}$$

$$\mathbb{E} G_s = \mu(s), \qquad \sigma^2 := \mathbb{E}\xi^2 = \frac{\mathrm{Var}_p z}{B}. \tag{53}$$

The noise law is independent of s. In particular, it samples the logit values z_Y, whose moments can be much better behaved than those of an individual rare-token indicator.

Lemma I.2 relates this difference in gradient statistics to Adam's mean response. For a rare-token coordinate the fourth-moment ratio can grow as $1/(Bp_i)$, so a long enough memory must see that token. The scalar statistic need not inherit this dependence on individual token probabilities.

A restoring response without weight decay.

Proposition I.4 (Scale regulation in the frozen mean response)**.** *Assume a finite vocabulary, $p_i > 0$, nonconstant z_i, and iid batches. The stationary mean Adam response $F(s) = \mathbb{E} R_s$ to* (52) *is continuous and strictly increasing. It has a unique finite zero s_β, with $-F(s)$ pointing toward that zero. As $\beta \to 1$, s_β approaches the unique minimizer s_* of $L(s)$.*

Proof. For a fixed two-sided noise history let $A = (1-\beta)\sum_{j\ge 0} \beta^j \xi_{-j}$ and $C = (1-\beta)\sum_{j\ge 0} \beta^j \xi_{-j}^2$. As a function of the mean-gradient shift u, the ratio is

$$R(u) = \frac{u + A}{\sqrt{u^2 + 2uA + C}}, \qquad R'(u) = \frac{C - A^2}{(u^2 + 2uA + C)^{3/2}} > 0 \quad \text{almost surely.}$$

Indeed, $C - A^2$ is the weighted variance of an infinite iid nondegenerate noise history, and is positive almost surely. Since $|R(u)| \le 1$, dominated convergence gives continuity of its expectation; strict monotonicity is preserved by averaging. Also, $\mu(s)$ is strictly increasing by (51). As $s \to +\infty$, gradients approach $\max_i z_i - B^{-1}\sum_b z_{Y_b} \ge 0$, which is nonzero with positive probability, and the stationary ratio has positive expectation. At $s \to -\infty$ the corresponding expectation is negative. Thus F has exactly one zero.

We give an error bound to justify convergence to the loss optimum. Set $\kappa = \mathbb{E}\xi^4/\sigma^4$, $K = 8\kappa$, and $\delta_\beta = \sqrt{\frac{1-\beta}{1+\beta}(K-1)}$. The inequality $(x+y)^4 \le 8(x^4+y^4)$ and $\kappa \ge 1$ give

$$\frac{\mathbb{E}(\mu+\xi)^4}{(\mu^2+\sigma^2)^2} \le K \qquad \text{for every } \mu.$$

When $\delta_\beta < 1$, Lemma I.2 at $F(s_\beta) = 0$ therefore yields

$$|L'(s_\beta)| \le \sigma \frac{\delta_\beta}{\sqrt{1-\delta_\beta^2}}. \tag{54}$$

Because L' is strictly increasing with unique zero s_*, this proves $s_\beta \to s_*$. More explicitly, if $L'' \ge m > 0$ on $[s_* - h, s_* + h]$ and the right-hand side above is less than mh, then s_β lies in this interval and

$$|s_\beta - s_*| \le \frac{\sigma}{m} \frac{\delta_\beta}{\sqrt{1-\delta_\beta^2}}. \tag{55}$$

□

The corresponding mean ODE

$$\dot{s} = -F(s) \tag{56}$$

has a globally attracting equilibrium at s_β, without WD: its continuous bounded vector field points toward the zero, and monotonicity of F gives uniqueness and convergence of solutions. This describes the averaged frozen response, not a convergence theorem for the finite-step Adam recursion or for joint matrix–gain training. At finite memory, skewness can shift s_β away from s_*. Symmetric noise gives $F(s_*) = 0$ exactly; the Zipf statistic below is generally asymmetric.

The scale statistic for Zipf logits. Write $a = \nu > 1$ and $n = V \geq 2$ for the Zipf exponent and vocabulary size, and set

$$p_i^{(n)} = \frac{i^{-a}}{H_{n,a}}, \qquad z_i = \log p_i^{(n)}, \qquad q_i(s) = \frac{i^{-as}}{H_{n,as}}, \qquad H_{n,c} = \sum_{i=1}^{n} i^{-c}. \tag{57}$$

Here $s_* = 1$. Up to a sign and a constant, the scalar samples $X = a \log Y$. Its moments obey vocabulary-independent bounds:

$$\mathrm{Var}_p X \geq \frac{2^{-a}(a \log 2)^2}{\zeta(a)^2} > 0, \qquad \mathbb{E}_p X^k \leq \sum_{i \geq 1} i^{-a}(a \log i)^k < \infty. \tag{58}$$

The variance bound retains the pair $(1, 2)$ in $\mathrm{Var} X = \sum_{i<j} p_i p_j (X_i - X_j)^2$; the upper bounds follow from $H_{n,a} \geq 1$ and summability. Thus the scalar's standardized fourth moment remains bounded as vocabulary grows. Writing $\kappa_X = \mathbb{E}(X - \mathbb{E}X)^4/(\mathrm{Var} X)^2$, the batch noise has

$$\sigma^2 = \mathrm{Var}_p X/B, \qquad \kappa_{\mathrm{batch}} = 3 + (\kappa_X - 3)/B. \tag{59}$$

Choose $0 < h < 1 - 1/a$. On $[1 - h, 1 + h]$, the same pair gives

$$L''(s) \geq \frac{2^{-a(1+h)}(a \log 2)^2}{\zeta(a(1-h))^2} > 0. \tag{60}$$

Together with (55), these bounds give a scale error $O(\sqrt{(1-\beta)/B})$, uniformly in vocabulary size. This is a calibration result for logits with the correct shape up to scale, not a claim that matrix training recovers that shape.

Opposite scale responses at the same state. The scalar and the matrix coordinates can therefore behave differently even when they represent the same distribution. We remove softmax's common-shift symmetry by centering the logits before comparing their norms.

Theorem I.5 (Outward coordinate response and restoring scalar response)**.** *Fix $a > 1$, $s > 1$, $B \geq 1$, and $0 < \beta < 1$. At the Zipf state* (57)*, let $\ell_i = s \log p_i^{(n)}$, let $R_{i,n}$ be the stationary coordinate-Adam ratios, and put $P_n = I - \mathbf{1}\mathbf{1}^\top/n$, $\widetilde{\ell} = P_n \ell$. There is a constant $D \in (0, \infty)$, independent of n, such that*

$$\frac{\langle \widetilde{\ell}, -\mathbb{E}R \rangle}{\log n} \longrightarrow saD > 0, \qquad \langle \widetilde{\ell}, -\nabla L(\ell) \rangle = -sL_n'(s) < 0. \tag{61}$$

For every $\eta > 0$ and all sufficiently large n, a coordinate-Adam step increases the expected squared centered norm:

$$\mathbb{E}\|P_n(\ell - \eta R)\|^2 - \|\widetilde{\ell}\|^2 > 0. \tag{62}$$

There is also $\beta_0 < 1$, depending on a, s, B but not n, such that for every fixed $\beta \in (\beta_0, 1)$ a scalar-Adam step at this same state decreases both the expected squared centered norm and expected population loss for all sufficiently small positive scalar learning rates, uniformly for large n.

Proof. Coordinate response. Write $\delta_{i,n} = 1 - \mathbb{E}R_{i,n} \in [0, 2]$. We first bound how fast the rare-coordinate ratios approach $+1$. For $i \geq 2$, write $p = p_i^{(n)} < 1/2$, $q = q_i(s)$, and choose

$$N = \left\lceil \frac{\log(1/(pq^2))}{|\log \beta|} \right\rceil, \qquad \beta^N \leq pq^2.$$

With probability at least $1 - NBp$, none of the last N batches contains token i. On this event all recent gradients equal q. Since $|G_i| \le 1$, the old histories satisfy $|M_{\text{old}}| \le 1$, $V_{\text{old}} \le 1$, and

$$M = (1-\beta^N)q + \beta^N M_{\text{old}} \ge q(1-2p), \qquad V = (1-\beta^N)q^2 + \beta^N V_{\text{old}} \le q^2(1+p).$$

Consequently $R_{i,n} \ge r_p = (1-2p)/\sqrt{1+p} \ge 0$ on this event and $R_{i,n} \ge -1$ everywhere. It follows that

$$\delta_{i,n} \le 1 - r_p + (1+r_p)NBp \le \tfrac{5}{2}p + 2NBp.$$

Since $\log(1/(pq^2)) \le \log[\zeta(a)\zeta(as)^2] + a(1+2s)\log i$, we obtain

$$\delta_{i,n} \le Ci^{-a}(1+\log i), \qquad i \ge 2, \tag{63}$$

where C depends only on the fixed parameters. Both this bound and its product with $\log i$ are summable.

For each fixed i, couple the binomial label counts using common uniforms as $p_i^{(n)} \to i^{-a}/\zeta(a)$. The bounded gradients and their EWMA sums converge almost surely. Their limiting denominator is positive almost surely because the gradient stream is nondegenerate. Since $|R_{i,n}| \le 1$, dominated convergence gives $\delta_{i,n} \to \delta_{i,\infty}$. Extending $\delta_{i,n} = 0$ for $i > n$, (63) yields

$$D_n := \sum_{i=1}^{n} \delta_{i,n} \to D := \sum_{i \ge 1} \delta_{i,\infty} < \infty, \qquad E_n := \sum_{i=1}^{n} (\log i)\delta_{i,n} \to E < \infty.$$

Token 1's limiting gradient stream is nondegenerate, so weighted Cauchy–Schwarz is strict almost surely: $R_{1,\infty} < 1$. Hence $\delta_{1,\infty} > 0$ and $D > 0$.

Set $m_n = n^{-1}\sum_{i=1}^{n} \log i$, so $m_n/\log n \to 1$ and $\widetilde{\ell}_i = sa(m_n - \log i)$. Because the centered entries sum to zero,

$$\langle \widetilde{\ell}, -\mathbb{E}R \rangle = \sum_i \widetilde{\ell}_i \delta_{i,n} = sa(m_n D_n - E_n),$$

which proves the positive limit. Meanwhile $\sum_i (q_i - p_i) = 0$ gives

$$\langle \widetilde{\ell}, -\nabla L(\ell) \rangle = -s \sum_i \log p_i (q_i - p_i) = -sL_n'(s) < 0,$$

using strict convexity and the optimum at $s = 1$. With the paper's RMS norm convention, the exact identity

$$\mathbb{E}\|P_n(\ell - \eta R)\|^2 - \|\widetilde{\ell}\|^2 = \frac{2\eta}{n}\langle \widetilde{\ell}, -\mathbb{E}R \rangle + \eta^2 \mathbb{E}\|P_n R\|^2$$

proves (62); no independence between coordinates is needed.

Scalar response. The pair $(1, 2)$ gives a vocabulary-independent restoring signal:

$$\mu_n(s) = \int_1^s L_n''(u)\, du \ge (s-1)\frac{2^{-as}(a\log 2)^2}{\zeta(a)^2} > 0.$$

Since $0 < \mu_n(s) \le \mathbb{E}_p X$, Eqs. (58) and (59) bound the scalar's second and fourth gradient moments uniformly in n. Thus its normalized mean stays positive, and Lemma I.2 gives $\mathbb{E}R_s \ge c > 0$ uniformly in n for β sufficiently close to one. A scalar step has expected squared-norm change

$$\|P_n z\|^2[-2\eta_s s\mathbb{E}R_s + \eta_s^2 \mathbb{E}R_s^2] \le \|P_n z\|^2[-2\eta_s sc + \eta_s^2] < 0 \quad \text{if } 0 < \eta_s < 2sc.$$

For the loss, choose $h > 0$ with $s - h > 1/a$. On $u \in [s-h, s+h]$,

$$L_n''(u) = \mathrm{Var}_{q(u)}(a\log i) \le M := \sum_{i \ge 1} i^{-a(s-h)}(a\log i)^2 < \infty.$$

With $L_n'(s) \ge m > 0$ as above, Taylor's theorem and $|R_s| \le 1$ give

$$\mathbb{E}[L_n(s - \eta_s R_s) - L_n(s)] \le -\eta_s mc + \tfrac{1}{2}M\eta_s^2 < 0$$

whenever $0 < \eta_s < \min(h, 2mc/M)$. These bounds are uniform in vocabulary size. □

Here the radial mismatch is task-relevant: $s > 1$ makes the Zipf distribution sharper than its optimal scale $s = 1$. The theorem compares one-step responses at a fixed state. It does not establish continued norm growth along a moving trajectory, and tangential updates can prevent coordinate Adam's full update from increasing loss even when its radial response is wrong. The scalar and coordinate results are compatible with shared batches; neither proof uses independence between their responses.

Which multipliers benefit from this argument? A scalar need not escape the rare-event problem. If $z_1 = 1$ and $z_i = 0$ for $i > 1$, then $G_s = q_1(s) - N_1/B$, exactly the original rare-token gradient. Likewise, at fixed features x, a gain on row i of an LM head has gradient

$$\partial_{r_i} \ell = (W_i x)\big(q_i - \mathbf{1}\{Y = i\}\big). \tag{64}$$

All terms share one rare-label indicator. A fixed nonzero prefactor leaves its normalized rare-event geometry unchanged, so summing many entries is not enough.

More generally, a gain producing logits $\ell_i(c) = b_i + ca_i$ at fixed features has gradient $\mathbb{E}_{q(c)}a - B^{-1}\sum_b a_{Y_b}$. The same additive-noise proof applies to any nonconstant statistic a, with controlled moments and curvature determining how close the mean-response zero lies to the loss optimum. For a nonconstant offset b_i, the mean is still strictly increasing and ranges between the same two extreme statistics. When features vary between examples, however, the q-dependent term becomes random too, so the monotonicity proof no longer applies automatically. Lemma I.2 still applies at each frozen state with iid gradients and finite fourth moment. Our final-RMSNorm measurements provide empirical evidence about the gains used in the LM head, complementing the explicit Zipf example above.

J Proof of Theorem 2.1

We prove that rare informative gradients can reverse the mean Adam update when they do not fill its memory, or suppress the positive mean relative to the integrated noise when they do. We work with stationary optimizer statistics at a fixed model.

Setup. Let $(g_t)_{t\in\mathbb{Z}}$ be iid negative batch gradients, so $\mathbb{E}g$ is the population descent direction. Assume $\mathbb{P}(g = 0) = 0$, $\mathbb{E}g^4 < \infty$, and write, within this appendix, $p = P_+ \in (0, 1/2]$ and $q = P_- = 1 - p$. Define

$$\begin{gathered} m_+ = \mathbb{E}[g \mid g > 0], \qquad m_- = \mathbb{E}[-g \mid g < 0], \qquad \chi = \frac{pm_+}{qm_-} > 1, \\ K_{\pm,r} = \frac{\mathbb{E}[|g|^r \mid \pm g > 0]}{m_\pm^r} \qquad (r = 2, 4). \end{gathered} \tag{65}$$

Thus $\mathbb{E}g = qm_-(\chi - 1) > 0$. The normalized total second moment is

$$K_2 = \frac{\mathbb{E}g^2}{m_-^2} = qK_{-,2} + \frac{\chi^2 q^2}{p} K_{+,2}. \tag{66}$$

For equal Adam momenta $\beta_1 = \beta_2 = \beta \in (0, 1)$ and zero denominator offset, let $\alpha = 1 - \beta$, $w_j = \alpha\beta^j$, and

$$M_t = \sum_{j\geq 0} w_j g_{t-j}, \qquad V_t = \sum_{j\geq 0} w_j g_{t-j}^2, \qquad Y_t = M_t/\sqrt{V_t}.$$

These stationary infinite-past sums exist almost surely, and $M_t^2 \leq V_t$ gives $|Y_t| \leq 1$. We study

$$b = \mathbb{E}Y_t, \qquad \Gamma = \sum_{h\in\mathbb{Z}} \mathrm{Cov}(Y_0, Y_h), \tag{67}$$

proving absolute convergence of the second series in the memory-filled regime. With ordinary loss gradients, the same descent drift is $-\mathbb{E}R_t$.

J.1 Wrong-way drift

Proposition J.1 (Wrong-way drift)**.** *Suppose $K_{-,2}, K_{-,4}, K_{+,2} \leq K$ for some $K \geq 1$. There are $a_K, c_K > 0$ such that*

$$\alpha \leq a_K, \qquad \frac{p}{\alpha}\log\left(1 + \frac{\chi^2}{p}\right) \leq c_K \quad \Longrightarrow \quad b < 0 < \mathbb{E}g.$$

Proof. Normalize $m_- = 1$, and set

$$a = \frac{1}{2\sqrt{K+2}}, \quad \delta = \frac{a}{4(1+a)}, \quad N = \left\lceil \frac{\log(64K_2/\delta)}{-\log\beta} \right\rceil, \quad r = \beta^N \le \frac{\delta}{64K_2}.$$

Since $K_2 \ge q \ge 1/2$, we have $r < 1/4$. Also $K_2 \le K(1+\chi^2/p)$ and $-\log\beta \ge \alpha$, so the stated conditions, with sufficiently small a_K, c_K, imply $65K\alpha \le \delta$ and $pN \le \delta$.

With probability at least $1 - pN$, the last N gradients are negative. Conditional on this event, their magnitudes are iid from the negative conditional law. Their weighted first and second sums S_N, T_N have means $1-r, K_{-,2}(1-r)$ and variances at most $\alpha(K_{-,2}-1)/(2-\alpha)$ and $\alpha K_{-,4}/(2-\alpha)$. Chebyshev bounds the failures of $S_N \ge 1-r-1/8$ and $T_N \le K_{-,2}(1-r)+1$ by a combined $65K\alpha$. The same bound applies to their intersections with the negative-window event unconditionally.

Write $M = -S_N + rM_{\rm old}$ and $V = T_N + rV_{\rm old}$. Stationarity gives $\mathbb{E}V_{\rm old} = K_2$ and $\mathbb{E}M_{\rm old}^2 = (\mathbb{E}g)^2 + \alpha{\rm Var}(g)/(2-\alpha) \le 2K_2$. Markov bounds the failures of $|rM_{\rm old}| \le 1/8$ and $rV_{\rm old} \le 1$ by $128r^2K_2$ and rK_2. Thus, outside an event of probability at most

$$d = pN + 65K\alpha + 128r^2K_2 + rK_2 \le 2\delta + \delta^2/16 + \delta/64 < 4\delta,$$

we have $M \le -1/2$, $V \le K+2$, and $Y \le -a$. Since $Y \le 1$ everywhere, $b \le -a + (1+a)d < 0$. The population descent direction is $\mathbb{E}g = q(\chi - 1) > 0$. □

The logarithm makes the event-free window long enough to remove the old moment tails; rarity relative to the nominal memory length alone is insufficient.

J.2 Memory-filled gradients

Set $\mu = \mathbb{E}g$, $s_2 = \mathbb{E}g^2$, and define the linearization quantities

$$b_\Delta = \frac{\mu}{\sqrt{s_2}}, \qquad \psi_\Delta(g) = \frac{g-\mu}{\sqrt{s_2}} - \frac{\mu(g^2 - s_2)}{2s_2^{3/2}}, \qquad \Gamma_\Delta = \mathbb{E}\psi_\Delta(g)^2. \tag{68}$$

We bound the actual covariance through its projection onto the informative sign, without requiring a uniform first-order approximation to the entire covariance. Assume the positive side contributes at least half the second moment:

$$K_{+,2}\chi^2 q \ge pK_{-,2}. \tag{69}$$

This condition is automatic for sufficiently rare informative signs under uniform conditional moment bounds. Indeed, if $K_{-,2} \le K$, then $K_{+,2} \ge 1$, $\chi > 1$, and $p \le 1/(K+1)$ imply $K_{+,2}\chi^2(1-p) \ge pK_{-,2}$. Uniform fourth-moment bounds suffice because $K_{-,2} \le \sqrt{K_{-,4}}$. We retain (69) explicitly below to state the finite-parameter range.

Normalize $X = g/\sqrt{s_2}$, $Q = X^2$, $b_0 = \mathbb{E}X = b_\Delta$, and $v = {\rm Var}Q$. We use the fourth-moment parameter

$$H = 1 + p\mathbb{E}Q^2, \qquad H \le \mathcal{K}_{\rm mf} := 1 + \frac{K_{+,4} + K_{-,4}}{K_{+,2}^2}. \tag{70}$$

Indeed, after setting $m_- = 1$ and $A = m_+ = \chi q/p$, the bound $s_2 \ge pK_{+,2}A^2$ gives

$$p\frac{\mathbb{E}g^4}{s_2^2} \le \frac{K_{+,4}}{K_{+,2}^2} + \frac{qK_{-,4}}{pK_{+,2}^2A^4} \le \frac{K_{+,4} + K_{-,4}}{K_{+,2}^2},$$

since $q/(pA^4) \le (p/q)^3 \le 1$. In particular $0 < b_0 \le \sqrt{p}$ and $v \le H/p$.

Proposition J.2 (Finite-fourth-moment bounds for actual Adam)**.** *Under the setup above and* (69)*, there are $C_H, c_H > 0$ depending only on a bound for H such that, if $\alpha/p \le c_H$, the covariance series in* (67) *is absolutely convergent, $\Gamma \le C_H$, and*

$$|b - b_\Delta| \le C_H\frac{\alpha}{\sqrt{p}}. \tag{71}$$

If, more specifically,

$$\alpha/p \le c_H(1 - 1/\chi), \tag{72}$$

with c_H chosen sufficiently small, then

$$0 < b \le 2(1-1/\chi)\sqrt{p/K_{+,2}}, \qquad \frac{1}{64K_{+,2}} \le \Gamma \le C_H, \tag{73}$$

so

$$\frac{b}{\sqrt{\Gamma}} \le 16(1-1/\chi)\sqrt{p} \le 16\sqrt{p}. \tag{74}$$

Thus α/p sufficiently small gives the memory-filled conclusion for bounded H and a fixed positive sign margin $\chi - 1$. The factor in (72) *resolves the smaller mean when χ approaches one.*

The informative sign. Direct substitution in (68) gives

$$b_\Delta = \sqrt{q}\,\frac{\chi - 1}{\sqrt{K_{-,2} + K_{+,2}\chi^2 q/p}}. \tag{75}$$

Dropping the negative second-moment contribution gives the upper bound below; (69) gives the lower bound:

$$(1-1/\chi)\sqrt{\frac{p}{2K_{+,2}}} \le b_\Delta \le (1-1/\chi)\sqrt{\frac{p}{K_{+,2}}}. \tag{76}$$

Let $S = (\mathbf{1}_{\{g>0\}} - p)/\sqrt{pq}$, so $\mathbb{E}S = 0$ and $\mathbb{E}S^2 = 1$. Writing $h = \mathbb{E}[\psi_\Delta \mid g > 0]$ and using $pK_{+,2}m_+^2 \le s_2$, we obtain

$$\begin{aligned} h &= \frac{m_+ - \mu/2}{\sqrt{s_2}} - \frac{\mu K_{+,2} m_+^2}{2 s_2^{3/2}} \\ &\ge \frac{m_+ - \mu(1+1/p)/2}{\sqrt{s_2}} = \frac{q m_+ + q(1+1/p)m_-}{2\sqrt{s_2}} \ge \frac{q m_+}{2\sqrt{s_2}}. \end{aligned}$$

Because ψ_Δ is centered and $s_2 \le 2pK_{+,2}m_+^2$ by (69),

$$\mathbb{E}[\psi_\Delta S] = \sqrt{p/q}\,h \ge \frac{\sqrt{q}}{2\sqrt{2K_{+,2}}} \ge \frac{1}{4\sqrt{K_{+,2}}}. \tag{77}$$

Thus $\Gamma_\Delta \ge 1/(16K_{+,2})$. We will establish a corresponding bound for actual Adam by controlling this single projection.

Controlling a small optimizer denominator.

Lemma J.3 (Inverse moments of a Bernoulli EWMA)**.** *Let I_j be iid Bernoulli(r), $0 < r \le 1/2$, and let $Z = \sum_{j\ge 0} \alpha(1-\alpha)^j I_j$. Let $Z^{(-\mathcal{I})}$ omit any set $\mathcal{I}$ of at most three terms. For each positive integer k, if $\alpha \le r/(8k)$, then*

$$\mathbb{E}[(Z^{(-\mathcal{I})})^{-k}] \le C_k r^{-k}$$

uniformly over $\mathcal{I}$, including $\mathcal{I} = \varnothing$, for a constant $C_k < \infty$.

Proof. All these variables are positive almost surely, because infinitely many of the independent indicators equal one. Put $h = -\log(1-\alpha) \le 2\alpha$. Independence and comparison of a decreasing sum with its integral give

$$\mathbb{E}e^{-tZ} \le \exp\left\{ -r \sum_{j\ge 0} (1 - e^{-t\alpha e^{-hj}}) \right\},$$

$$\sum_{j\ge 0} (1 - e^{-t\alpha e^{-hj}}) \ge \frac{1}{h}\int_0^{\alpha t} \frac{1-e^{-v}}{v}\,dv.$$

Use $1 - e^{-v} \ge v/2$ for $v \le 1$ and $1 - e^{-v} \ge 1/2$ for $v \ge 1$. Thus the Laplace transform is bounded by $e^{-rt/4}$ for $t \le 1/\alpha$, and by $e^{-r/(4\alpha)}(\alpha t)^{-r/(4\alpha)}$ thereafter. Deleting at most three

factors multiplies the bound by at most $(1-r)^{-3} \le 8$. Integrating against $t^{k-1}/\Gamma_{\mathrm{E}}(k)$, where Γ_{E} is Euler's gamma function, and putting $u = r/\alpha \ge 8k$, gives

$$\mathbb{E}[(Z^{(-\mathcal{I})})^{-k}] \le 8(4/r)^k + \frac{8\alpha^{-k}e^{-u/4}}{\Gamma_{\mathrm{E}}(k)(u/4-k)} \le C_k r^{-k}.$$

The last inequality uses $\sup_{u\ge 8k} u^{k-1}e^{-u/4} < \infty$; the inverse-moment integral follows from the gamma integral and Tonelli's theorem. □

By (69), $p\mathbb{E}[Q \mid g>0] \ge 1/2$. Conditional Paley–Zygmund gives

$$\theta := \mathbb{P}\left(Q \ge \tfrac{1}{2}\mathbb{E}[Q \mid g>0] \mid g>0\right) \ge \frac{\mathbb{E}[Q \mid g>0]^2}{4\mathbb{E}[Q^2 \mid g>0]} \ge \frac{1}{16H}.$$

The last inequality uses $\mathbb{E}[Q \mid g>0] \ge 1/(2p)$ and $\mathbb{E}[Q^2 \mid g>0] \le H/p^2$. On this event $Q \ge 1/(4p)$. Thus a denominator with any at most three source coordinates removed dominates $(4p)^{-1}$ times a Bernoulli(r) EWMA, with $r = p\theta \ge p/(16H)$ and $r \le 1/2$. Lemma J.3 therefore bounds every inverse moment of fixed order needed below by a constant depending only on H, for $\alpha/p \le c_H$. Fractional inverse powers follow by Hölder.

These assumptions also imply $K_{+,2} \le 4H$. Indeed, for $Z = g/m_+$ conditioned on the positive sign, Hölder gives $(\mathbb{E}Z^2)^3 \le \mathbb{E}Z^4$, so $K_{+,2} \le K_{+,4}/K_{+,2}^2$; the latter ratio is at most $4H$ by the preceding conditional moment bounds.

Weighted moments and temporal correlations.

Lemma J.4 (Inverse-weighted second moments)**.** *Let (ξ_i, Q_i) be iid with $\mathbb{E}\xi = 0$, $Q \ge 0$, $\mathbb{E}Q = 1$, $\mathbb{E}\xi^2 < \infty$, and $\mathbb{E}Q^2 < \infty$. Set $T = \sum_i w_i Q_i$ and $D = \sum_i w_i \xi_i$, allowing any one common index to be omitted. Under the preceding inverse-moment bounds, for every fixed $s > 0$,*

$$\mathbb{E}[D^2 T^{-s}] \le C_{H,s}\left[\mathbb{E}\xi^2 \sum_i w_i^2 + \left(\mathbb{E}|\xi||Q-1|\right)^2 \left(\sum_i w_i^2\right)^2\right]. \tag{78}$$

Proof. First expand a finite partial numerator sum. For a diagonal term, delete its coordinate from the denominator and use independence: $\mathbb{E}[\xi_i^2 T^{-s}] \le \mathbb{E}\xi^2\, \mathbb{E}T_{\mathrm{rest}}^{-s}$. For $i \ne j$, subtract the versions of $f(T) = T^{-s}$ in which Q_i, or Q_j, or both, are replaced by their mean one. Each subtracted term has zero expectation against $\xi_i\xi_j$, since a corresponding centered source is independent of it. The remaining double finite difference is bounded by

$$s(s+1)w_i w_j |Q_i - 1||Q_j - 1| T_{\mathrm{rest}}^{-s-2}.$$

All intermediate denominators are at least T_{rest}, because the two source values and the reference value one are nonnegative. Multiply by $|\xi_i\xi_j|$ and use independence, obtaining the square of $\mathbb{E}|\xi||Q-1|$. Sum the outer coefficients $w_i w_j$ and use the inverse-moment bounds. Let the numerator truncation grow and apply Fatou's lemma. This proves (78) without higher moments. □

Let $A = \sum_i w_i(X_i - b_0)$ and $B = \sum_i w_i(Q_i - 1)$. Since $\mathbb{E}|X - b_0||Q-1| \le \sqrt{v}$ and $\sum_i w_i^2 \le \alpha$, the lemma gives

$$\mathbb{E}[A^2 T^{-s}] \le C_H(\alpha + \alpha^2 v), \qquad \mathbb{E}[B^2 T^{-s}] \le C_H(\alpha v + \alpha^2 v^2). \tag{79}$$

These estimates also hold for the centered sums and denominator with any one source omitted. Here B denotes a fluctuation, not the batch size.

Martingale projection identity. For a stationary centered causal process H_t in the iid source with $\mathbb{E}H_0^2 < \infty$, let $\mathcal{F}_t = \sigma(g_s : s \le t)$ and $P_0 H_j = \mathbb{E}[H_j \mid \mathcal{F}_0] - \mathbb{E}[H_j \mid \mathcal{F}_{-1}]$. Let H_j^* replace g_0 by an independent copy. Then $\|P_0 H_j\|_2 \le \|H_j - H_j^*\|_2$. If $\sum_{j\ge 0}\|P_0 H_j\|_2 < \infty$, then

$$\sum_{h\in\mathbb{Z}} |\mathrm{Cov}(H_0, H_h)| \le \left(\sum_{j\ge 0}\|P_0 H_j\|_2\right)^2, \qquad \sum_h \mathrm{Cov}(H_0, H_h) = \left\|\sum_{j\ge 0} P_0 H_j\right\|_2^2. \tag{80}$$

For completeness, expand $H_t = \sum_{k \le t} P_k H_t$ in L^2; the iid past tail is trivial. Only equal projection indices survive in each covariance. After summing all lags, the pairs of nonnegative indices occur exactly once. Cauchy–Schwarz bounds the absolute double sum by the first right-hand side, justifying regrouping and proving the second identity.

Proof of Proposition J.2. **Drift.** Let $U = \sum_j w_j X_j$, $T = \sum_j w_j Q_j$, $A = U - b_0$, and $B = T - 1$. The exact decomposition is

$$\frac{U}{\sqrt{T}} = b_0 + A - \frac{b_0}{2} B + R, \qquad R = A(T^{-1/2} - 1) + b_0(T^{-1/2} - 1 + B/2).$$

For all $T > 0$, $|T^{-1/2} - 1| \le |B| T^{-1/2}$ and $0 \le T^{-1/2} - 1 + B/2 \le B^2 T^{-1/2}$. For the latter inequality, setting $s = \sqrt{T}$ makes the middle expression $(s-1)^2(s+2)/(2s)$. Therefore

$$|R| \le |AB| T^{-1/2} + b_0 B^2 T^{-1/2}.$$

Apply Cauchy–Schwarz to $AT^{-1/4}$ and $BT^{-1/4}$ and then (79) with $s = 1/2$. This gives

$$|b - b_\Delta| \le \mathbb{E}|R| \le C_H \alpha (\sqrt{v} + b_0 v)(1 + \alpha v) \le C_H \alpha / \sqrt{p}.$$

The last step uses $b_0 \le \sqrt{p}$, $v \le H/p$, and small α/p.

Covariance existence and upper bound. Replace one innovation X_0 by an independent copy X_0'; its weight at time j is $w = w_j$. Along the interpolation write

$$U_\theta = U_{\mathrm{rest}} + w X_\theta, \quad T_\theta = T_{\mathrm{rest}} + w Q_\theta, \quad X_\theta^2 \le Q_\theta,$$

where both local values are convexly interpolated. Split $U_{\mathrm{rest}} = b_0(1-w) + A_{\mathrm{rest}}$. For $f(U,T) = U/\sqrt{T}$,

$$|f_U| \le T_{\mathrm{rest}}^{-1/2}, \qquad |f_T| \le \tfrac{1}{2}(b_0 + |A_{\mathrm{rest}}|) T_{\mathrm{rest}}^{-3/2} + \tfrac{1}{2}\sqrt{w}\, T_{\mathrm{rest}}^{-1}.$$

The local term uses $w|X_\theta| \le \sqrt{w}\sqrt{T_\theta}$; this self-normalization avoids introducing a sixth-moment requirement. Conditional on the two local values, (79) implies

$$\|f_U\|_2 \le C_H, \qquad \|f_T\|_2 \le C_H \sqrt{p + \alpha + \alpha^2 v + w} \le C_H \sqrt{p}.$$

Integrate along the segment and use $\mathbb{E}(X - X')^2 \le 2$, $\mathbb{E}(Q - Q')^2 = 2v$ to obtain

$$\delta_j(Y) := \|Y_j - Y_j^*\|_2 \le C_H w_j (1 + \sqrt{pv}) \le C_H w_j.$$

Apply the projection identity (80) to $Y_t - b$. It proves absolute covariance convergence, existence of $D_Y = \sum_{j \ge 0} P_0 Y_j$, and $\Gamma = \|D_Y\|_2^2 \le C_H$.

Approximate only the sign projection. The standardized sign satisfies $\mathbb{E}S = 0$, $\mathbb{E}S^2 = 1$, and $|S| \le p^{-1/2}$. Since S is measurable at time zero and independent of the strict past,

$$\mathbb{E}[D_Y S] = \sum_{j \ge 0} \mathbb{E}[Y_j S].$$

The preceding dependence bound gives absolute summability. For each j, subtract the ratio in which the local pair (X, Q) is replaced by its deterministic mean $(b_0, 1)$; this term is independent of S. Write $\delta X = X - b_0$, $\delta Q = Q - 1$ and interpolate to the actual local pair. Then

$$\mathbb{E}[Y_j S] = w_j \mathbb{E}\left[S \int_0^1 (\delta X f_U + \delta Q f_T)\, d\theta \right].$$

At the population-moment point $(U, T) = (b_0, 1)$ the derivative is $(1, -b_0/2)$, so the corresponding linear terms sum to $\mathbb{E}[\psi_\Delta S]$.

Along this interpolation, $U_\theta - b_0 = A_{\mathrm{rest}} + w\theta\delta X$ and $T_\theta - 1 = B_{\mathrm{rest}} + w\theta\delta Q$. The elementary derivative inequalities and (79) give, conditional on the local value X,

$$\mathbb{E}_{\mathrm{rest}}|f_U - 1| \le C_H \left[\sqrt{\alpha v + \alpha^2 v^2} + w|\delta Q| \right],$$

$$\mathbb{E}_{\mathrm{rest}}|f_T + b_0/2| \le C_H \left[\sqrt{\alpha + \alpha^2 v} + b_0 \sqrt{\alpha v + \alpha^2 v^2} + w|\delta X| + b_0 w |\delta Q| \right].$$

Specifically, use $|T^{-1/2}-1| \le |T-1|T_{\mathrm{rest}}^{-1/2}$ and

$$|f_T + b_0/2| \le \tfrac{1}{2}|U - b_0|T_{\mathrm{rest}}^{-3/2} + \tfrac{3}{4}b_0|T-1|\max(1, T_{\mathrm{rest}}^{-3/2}).$$

Only inverse-weighted second moments are required.

The remaining local moments satisfy

$$\mathbb{E}|S\delta X| \le 1, \quad \mathbb{E}|S\delta Q| \le \sqrt{v}, \quad \mathbb{E}|S\delta X\delta Q| \le p^{-1/2}\sqrt{v}, \quad \mathbb{E}|S|\delta Q^2 \le p^{-1/2}v.$$

Sum with weights w_j, use $\sum_j w_j^2 \le \alpha$, and absorb the bounded factor $1+\alpha v$. We obtain

$$\begin{aligned}|\mathbb{E}[D_Y S] - \mathbb{E}[\psi_\Delta S]| &\le C_H\left[\sqrt{\alpha v} + \sqrt{\alpha}\sqrt{v} + b_0 v\sqrt{\alpha} + \frac{\alpha}{\sqrt{p}}(2\sqrt{v} + b_0 v)\right] \\ &\le C_H\sqrt{\alpha/p}.\end{aligned}$$

Hence

$$|\mathbb{E}[D_Y S] - \mathbb{E}[\psi_\Delta S]| \le C_H\sqrt{\alpha/p}. \tag{81}$$

Conclusion. The bound $K_{+,2} \le 4H$ allows us to choose c_H in (72) so that the sign-projection error is at most $1/(8\sqrt{K_{+,2}})$ and the drift error is at most half the lower bound in (76). Then (77) gives $\mathbb{E}[D_Y S] \ge 1/(8\sqrt{K_{+,2}})$. Since $\mathbb{E}S^2 = 1$, Cauchy–Schwarz yields $\Gamma = \mathbb{E}D_Y^2 \ge 1/(64K_{+,2})$. Absorbing the drift error proves (73); dividing by this covariance lower bound gives (74). □

J.3 The fixed-law high-momentum limit

Proposition J.5 (Fixed-law high-momentum limit)**.** *Let g_t be iid with a fixed distribution, $0 < s_2 = \mathbb{E}g^2 < \infty$ and $\mathbb{E}g^4 < \infty$. An atom at zero is allowed. Use stationary equal-beta, zero-offset moments, and define $b_\Delta, \psi_\Delta, \Gamma_\Delta$ by* (68)*. Then, as $\alpha \downarrow 0$,*

$$b \longrightarrow b_\Delta, \qquad \Gamma \longrightarrow \Gamma_\Delta.$$

For sufficiently small α the covariance series is absolutely convergent. No rate uniform over gradient distributions is asserted.

Proof. Normalize $X = g/\sqrt{s_2}$, $Q = X^2$, $b_0 = \mathbb{E}X$, $U = \sum_j w_j X_j$, and $T = \sum_j w_j Q_j$, so $Y = U/\sqrt{T}$. Choose $d > 0$ and a probability $r > 0$ such that $\mathbb{P}(Q \ge d) = r$. If $r > 1/2$, independently thin the successes to make their probability at most $1/2$; this preserves a lower bound by d times a Bernoulli EWMA. Every denominator with one innovation removed therefore has inverse moments of any fixed order, uniformly for sufficiently small α, by Lemma J.3.

The normalized EMAs satisfy $(U,T) \to (b_0, 1)$ in L^2 because $\mathbb{E}X^2$ and $\mathbb{E}Q^2$ are finite. Since $|Y| \le 1$, dominated convergence in probability gives $b \to b_\Delta$.

Write the ratio as $Y = b_0 + \sum_j w_j\psi_\Delta(g_{-j}) + R(U,T)$, where

$$R(U,T) = \frac{U}{\sqrt{T}} - b_0 - (U - b_0) + \frac{b_0}{2}(T-1).$$

Use the martingale projections in (80). Condition on any fixed pair of local innovation values x, x'. Uniformly over the omitted index and the interpolation parameter, the interpolated (U,T) tends in probability to $(b_0, 1)$: the omitted weight is at most α, and all remaining variance bounds tend to zero uniformly. The two remainder derivatives therefore tend to zero in conditional L^2. To justify uniform integrability, the interpolation preserves $U^2 \le T$, and hence

$$|\partial_U R| \le 1 + T_{\mathrm{rest}}^{-1/2}, \qquad |\partial_T R| \le |b_0|/2 + T_{\mathrm{rest}}^{-1}/2.$$

Higher inverse moments of T_{rest} bound the corresponding conditional moments uniformly over the index, interpolation, and local values x, x'. Thus the conditional second moments of the derivative errors are bounded by a common finite constant, and tend to zero for every fixed pair x, x'.

Multiply those errors by $(x - x')^2$ and $(x^2 - x'^2)^2$, respectively. These factors are integrable under the independent pair law because $\mathbb{E}X^4 < \infty$. Dominated convergence and integration along the interpolation therefore yield a number $e(\alpha) \to 0$ with

$$\|R_j - R_j^*\|_2 \le w_j e(\alpha) \quad \text{for every } j.$$

Summing the weights and applying (80) gives $|\sqrt{\Gamma} - \sqrt{\Gamma_\Delta}| \le e(\alpha)$. This also supplies absolute covariance convergence and completes the proof.

□

Moderate signed-mass ratios. For bounded $\chi > 1$ separated from one, wrong-way drift requires only $(p/\alpha)\log(1/p)$ sufficiently small, whereas the memory-filled condition is α/p sufficiently small. The two statements describe different sides of the memory scale, rather than a sharp common threshold.

J.4 Zipf interpretation

For a softmax logit at a frozen state, the negative gradient is $g = N/B - q_i$ with $N \sim \mathrm{Binomial}(B, p_i)$. If $0 < q_i < 1/B$, then

$$P_+ = 1 - (1 - p_i)^B, \qquad G_- = q_i(1 - p_i)^B, \qquad G_+ = p_i - q_i P_+.$$

For $Bp_i \ll 1$ and $q_i \ll p_i$, these become $P_+ \simeq Bp_i$, $G_+ \simeq p_i$, $G_- \simeq q_i$, and $\chi \simeq p_i/q_i$. Proposition J.1 then supplies wrong-way drift when $(Bp_i/\alpha)\log(1 + \chi^2/(Bp_i))$ is sufficiently small, together with the stated moment and momentum conditions. The logarithm matters when q_i is extremely small: $Bp_i \ll \alpha$ alone does not control old second moments relative to q_i^2. In the memory-filled regime $\alpha \ll Bp_i \ll 1$, and with the sign margin and regularity conditions above, Proposition J.2 gives $b/\sqrt{\Gamma} = O(\sqrt{Bp_i})$. These are frozen-coordinate statements underlying the local interpretation of Regions II and III in §2.2.

J.5 Diagnostic protocol

We can estimate the statistics in these propositions by holding model parameters fixed and sampling fresh independent batches. With the negative gradient convention, we measure (i) $P_\pm, G_\pm$ and χ, (ii) $K_{\pm,2}, K_{\pm,4}$, and (iii) their relation to the memory parameter α. The first group underlies Figs. 3 and 18. A replay through stationary or sufficiently warmed-up Adam moments can also estimate b and the lag covariance sum Γ. Opposite signs of b and $\mathbb{E}g$ indicate mean reversal; a small positive $b/\sqrt{\Gamma}$ indicates a weak mean relative to long-run noise in the frozen stream. Reliable estimation requires enough rare events and a lag window longer than the relevant correlation time. The sign plots alone do not verify the moment or memory conditions.

K Additional plots

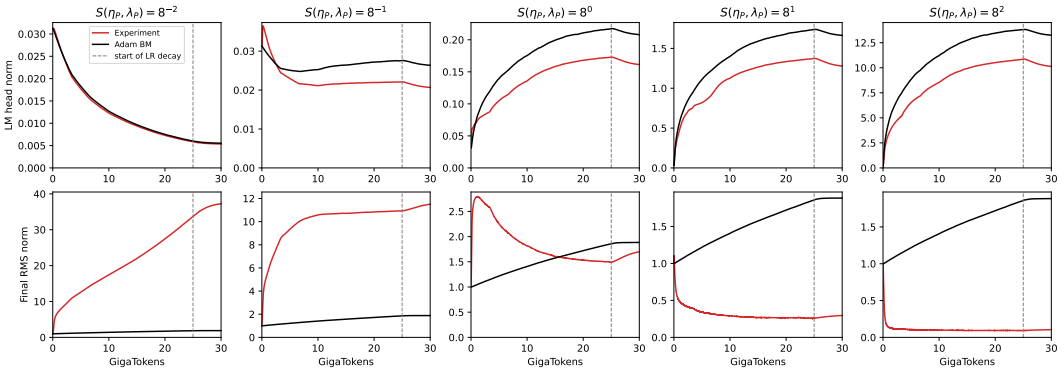


Figure 17: Detailed norm trajectories during training of the LM head experiment of Fig. 1, across five LM-head norm scales $S(\eta_P, \lambda_P) \in \{8^{-2}, 8^{-1}, 8^{0}, 8^{1}, 8^{2}\}$ (one column per scale). **Top row:** LM head matrix norm; **bottom row:** final-RMSNorm coefficient norm. Red curves are the actual training run; black curves are the Adam Brownian Motion (ABM) baseline obtained by replacing the data-driven gradient with i.i.d. noise of matching variance under the same η, λ schedules. The LM head matrix tracks ABM at every scale, confirming its norm is noise-driven; the final-RMSNorm coefficient deviates substantially from ABM, especially at large scales, indicating data-driven scale learning by the multiplier-like coordinate. Grey dashed line marks the start of learning-rate decay at 25 GT.

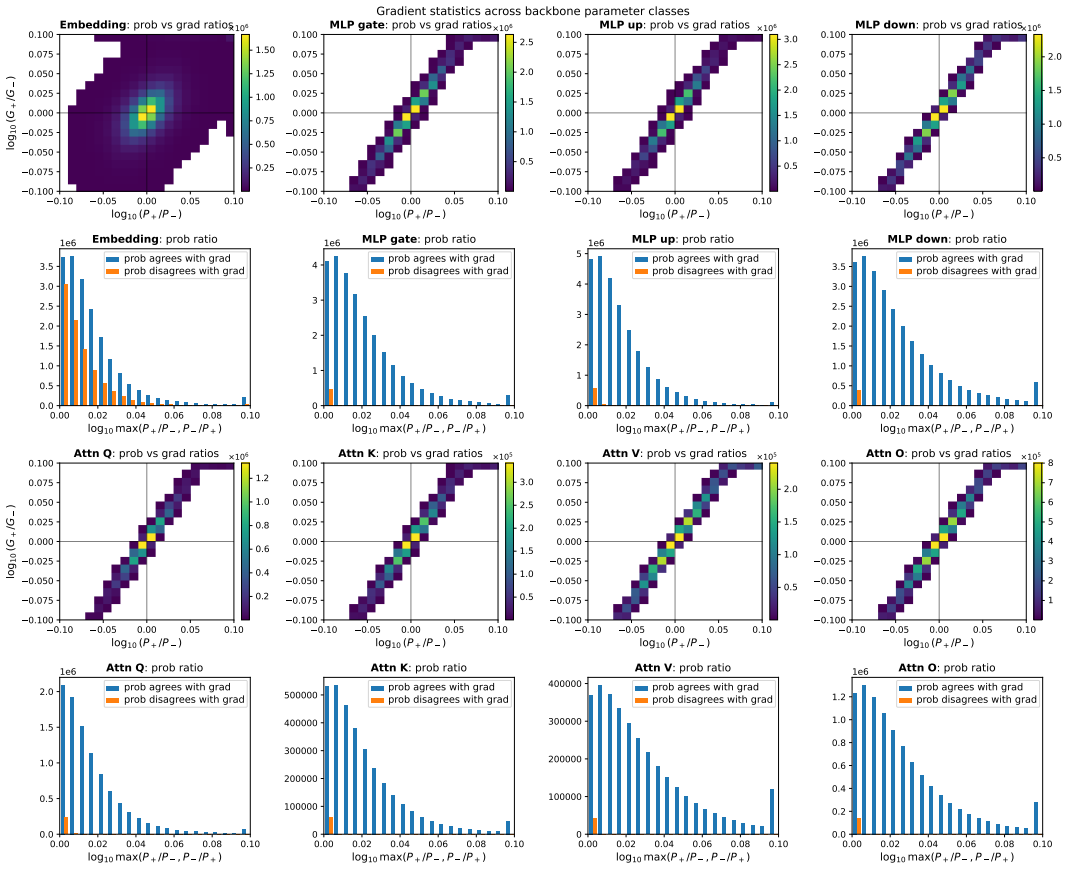


Figure 18: Sign-mass statistics of gradient streams across backbone parameter classes (Embedding; MLP gate/up/down; Attn Q/K/V/O) in a trained LLM. Each class is shown via a joint $(\log_{10}(P_+/P_-), \log_{10}\chi)$ density (heatmap rows) and a histogram of $\max(P_+/P_-, P_-/P_+)$ split by whether the rare-side direction agrees (blue) or disagrees (orange) with the population-gradient direction. All backbone classes lie on the benign diagonal ($\chi \approx P_+/P_-$, signal-side dominates) with negligible rare-event asymmetry, in marked contrast to the LM head matrix entries of Fig. 3 (right). These measurements do not show the head's rare-sign asymmetry in the backbone. They do not establish the cause of backbone norm growth (§G), which remains open.

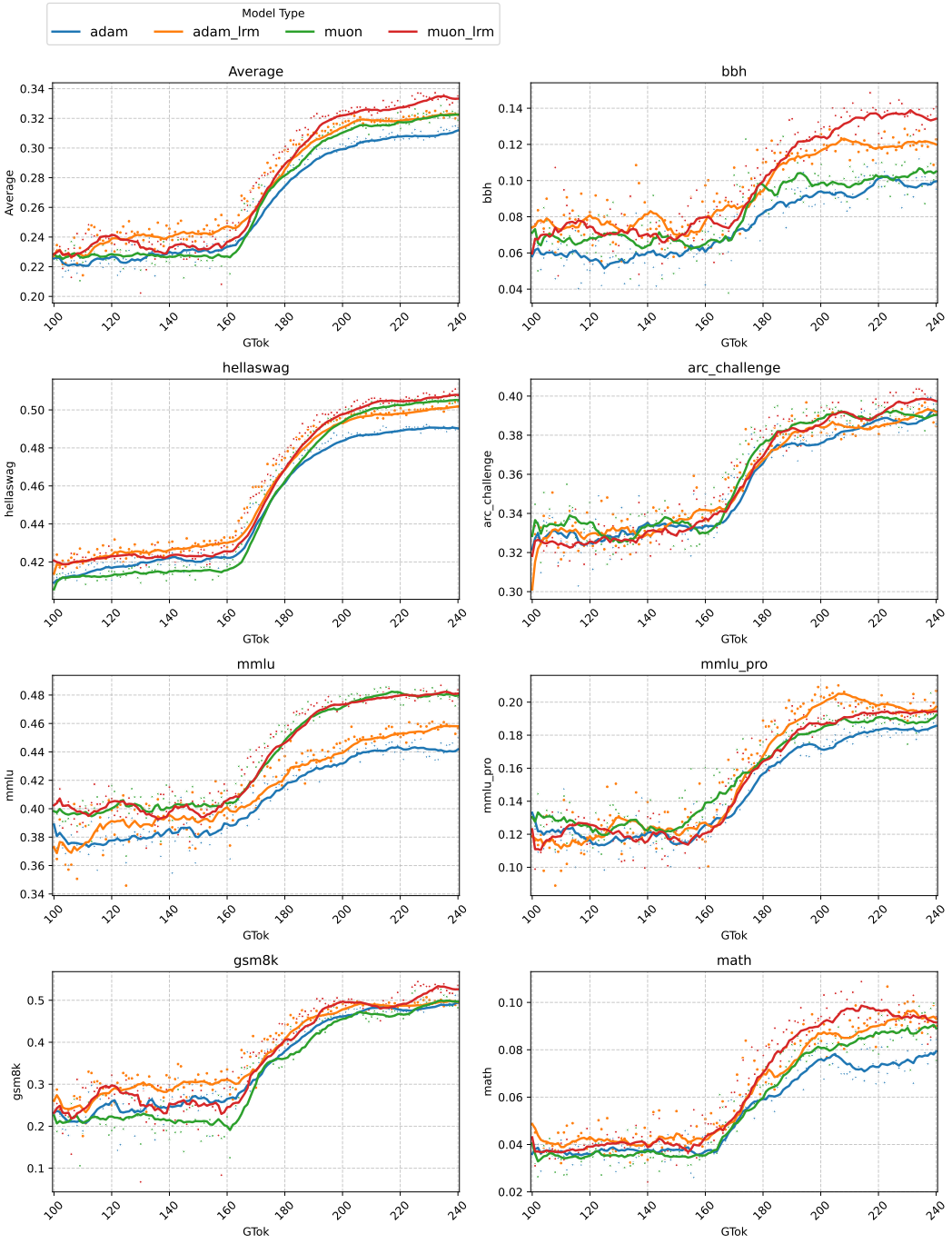


Figure 19: The detailed evaluation curves for the Falcon-H1 runs reported in Figure 8 (right). The markers correspond to actual evaluation score at a given checkpoint while solid lines denote running window average score over 20 last checkpoints. We evaluated checkpoints every gigatoken to carefully average the benchmarks stochasticity, and started this frequent evaluation only from 100GT due to compute constraints. We perform $\times 32$ exponential decay from 160GT to 200GT, which explains the growth of scores in this time window. After the end of exponential decay, the model was trained for 40 more gigatokens with minimal learning rate to obtain enough evaluation points ensuring well averaged scores reported in Figure 8 (right).