EFFECTIVE LEARNING RATE GOVERNS LOSS DYNAMICS IN LANGUAGE MODEL PRETRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

We establish **ELR collapse** as a quantitative empirical law in language model pretraining: learning rate (LR) and parameter norm govern loss dynamics primarily through their ratio, the effective learning rate (ELR). When tensor-wise ELRs are matched across runs, their loss trajectories collapse throughout training despite substantially different LRs and parameter norms. Across optimizers, architectures, datasets, and model scales, mean collapse errors are typically a few $\times 10^{-3}$, below the seed-to-seed variation measured in a representative configuration. Systematic ablations identify normalization design and the timescale of LR–norm variation as key determinants of collapse precision. In standard training, tensor-wise ELRs approximately synchronize, following a shared temporal profile up to fixed amplitudes. Controlled interventions further show that adjusting only the base LR to match an aggregate ELR reproduces the loss dynamics induced by weight decay and Hyperball to high precision. Replacing LR with ELR enables a fitted functional scaling law (FSL) to transfer across norm-control methods without refitting. The resulting FSL also explains delayed acceleration, where norm control initially raises loss but improves it later, and guides the design of an ELR schedule that further lowers final loss. Together, these findings identify ELR as an intrinsic clock rate governing loss dynamics across LR schedules and norm-control methods.

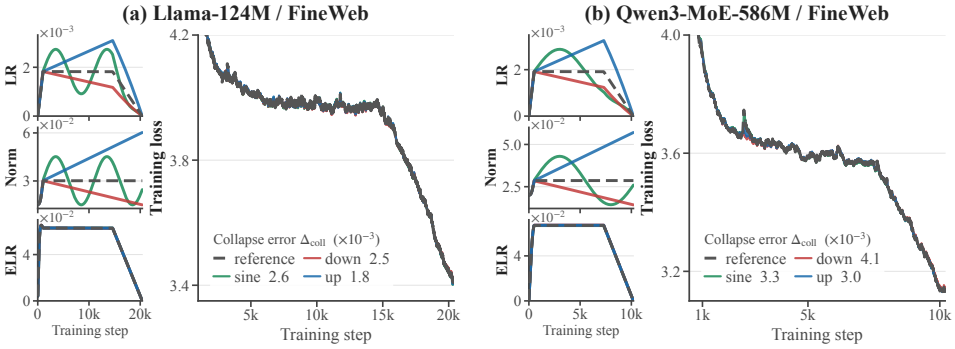


Figure 1: **ELR matching collapses loss trajectories despite different LR and norm schedules.** In each panel, three nonconstant LR schedules are paired with prescribed norm schedules to match the ELR of a reference run. Legend values report the post-warmup mean absolute loss discrepancy from the reference in units of 10^{-3}. Loss panels are clipped above 4.2 in **(a)** and 4.0 in **(b)** for display. See Section 4 and Appendices A and B.2 for experimental details and additional results.

1 INTRODUCTION

In large language model (LLM) pretraining, learning rate (LR) and batch size are standard control variables for training dynamics. Norm control also shapes these dynamics by regulating parameter-norm evolution, either implicitly through weight decay (Krogh & Hertz, 1991; Loshchilov & Hutter, 2019) or explicitly through norm constraints (Wen et al., 2026a; Loshchilov, 2023; Xie et al., 2026). It can improve training stability (D'Angelo et al., 2024), facilitate hyperparameter transfer across scales (Kosson et al., 2026), and accelerate convergence. This raises a basic question:

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.

Does norm control introduce a degree of freedom governing loss dynamics independent of LR scheduling?

ELR collapse: a precise macroscopic law. Within the regimes studied, our experiments answer this question in the negative. We construct runs with substantially different LR and parameter-norm trajectories but matching their ELR schedules, with ELR defined for each parameter matrix as $\eta_k^{\text{eff}} := \eta_k / \|\mathbf{W}_k\|_{\text{RMS}}$. Figure 1 presents the central experiment:

When ELR schedules are matched, full loss trajectories collapse with mean absolute discrepancies of only a few $\times 10^{-3}$.

We observe this precision across optimizers, architectures, datasets, and model scales. For comparison, changing only the initialization or data-order seed in a representative configuration yields mean pairwise loss discrepancies of order 10^{-2} (Appendix B.2.1). We call this phenomenon *ELR collapse*. It suggests a macroscopic picture: LR scheduling and norm control shape loss dynamics primarily through the ELRs they jointly induce, rather than acting as independent controls.

ELR collapse is nontrivial and conditional. Transformers are not exactly scale invariant: in a pre-norm residual block, $\mathbf{h}^{\ell+1} = \mathbf{h}^{\ell} + \text{FFN}(\text{RMSNorm}(\mathbf{h}^{\ell}))$, rescaling $\mathbf{h}^{\ell}$ affects the identity and residual branches differently; Appendix B.1 details further sources of scale dependence. Thus, high-precision collapse does not follow from the exact scale symmetry that motivates ELR (van Laarhoven, 2017; Li et al., 2020). Our ablations identify normalization design and the timescale of LR–norm variation as key determinants. Removing QK-Norm (Henry et al., 2020) weakens collapse even though training remains stable, and fixing the remaining RMSNorm gains weakens it further. Rapid joint variation of LR and norm also degrades collapse despite preserving ELR. Thus, precision depends on how a given ELR schedule is realized during training. These findings rule out static scale symmetry as an explanation and reveal an empirical regime for high-precision collapse, while its microscopic dynamical origin remains open.

ELR synchronization in standard training. Although ELR collapse does not require tensors to share a common ELR schedule, we observe that their ELRs approximately synchronize in standard training: they follow a shared temporal profile up to tensor-specific amplitudes. The resulting ELR matrix has a dominant rank-one component, and matching this component alone recovers the reference loss trajectory to high precision. A shared nominal LR does not impose this structure, since tensor norms can evolve differently. This synchronization suggests a shared ELR clock across tensors, expressed in different fixed units.

Beyond collapse: mediation, transfer, and explanation. We now investigate what the ELR perspective reveals about how practical norm-control methods shape loss dynamics.

- **ELR mediates the effects of norm control.** Can the effects of norm control on loss dynamics be reproduced by adjusting only a single base LR? Since this cannot independently match every tensor's ELR, we use an aggregate ELR as the matching criterion. For weight decay and Hyperball (Wen et al., 2026a), we retain each run's norm-control mechanism and adapt only the comparison run's base LR to match the reference aggregate ELR, allowing parameter norms to evolve freely. This recovers the reference loss dynamics with mean discrepancies of a few $\times 10^{-3}$ during stable training, showing that global LR scheduling can reproduce these effects without additional tensor-wise control.
- **From trajectory matching to scaling-law transfer.** We next ask whether ELR enables a predictive law to transfer across norm-control methods. We fit the functional scaling law (FSL) (Li et al., 2025a) using only non-Hyperball runs and evaluate it, without refitting, on unseen Hyperball runs. Replacing LR with ELR enables accurate prediction of these loss trajectories, whereas the LR-based FSL exhibits a large systematic bias.
- **ELR explains delayed acceleration.** We use ELR-based FSL to explain *delayed acceleration*: a norm-controlled run can initially have higher loss, yet later overtake an uncontrolled baseline (Hoffmann et al., 2022; Liu et al., 2025; Wen et al., 2026a). By restraining norm growth, norm control can sustain a larger ELR. Within FSL, this improves signal learning but also increases noise accumulation, masking the advantage until later ELR decay. The benefit is thus acquired early but revealed late, making early-loss comparisons potentially misleading. An ELR schedule optimized using the fitted FSL further lowers the final loss, supporting this explanation and illustrating ELR-first schedule design.

Together, these findings identify ELR as an intrinsic clock rate governing loss dynamics across LR schedules and norm-control methods.

2 Related Work

ELR under scale invariance. Normalization has motivated the study of scale-invariant objectives satisfying $\mathcal{L}(c\mathbf{W}) = \mathcal{L}(\mathbf{W})$ for all $c > 0$, which depend on $\mathbf{W}$ only through its direction $\widehat{\mathbf{W}} = \mathbf{W}/\|\mathbf{W}\|$. For gradient descent, the ELR governing the dynamics of $\widehat{\mathbf{W}}_k$ is $\eta_k/\|\mathbf{W}_k\|^2$ (Hoffer et al., 2018); for scale-insensitive optimizers such as signSGD and Adam, it is $\eta_k/\|\mathbf{W}_k\|$ (van Laarhoven, 2017). ELR has since been widely used to analyze optimization in normalized models (Li & Arora, 2020; Heo et al., 2021; Li et al., 2020; Wan et al., 2021; Kodryan et al., 2022). Whereas these works derive ELR from exact scale invariance, we show that ELR governs loss dynamics to high precision in LLM pretraining despite the absence of such symmetry.

Weight decay and loss dynamics. Prior experiments suggest that weight decay affects loss through ELR. D'Angelo et al. (2024) adjusted the LR of runs without weight decay to match the ELR induced by weight decay, obtaining approximate loss alignment. Li et al. (2025b) reported late-stage loss alignment when $\eta\lambda$, an equilibrium proxy for ELR, is matched. These observations suggest that ELR may explain the effect of weight decay on loss, but leave open how precisely this principle holds throughout training, how broadly it applies, and what conditions it requires.

While these studies suggest an ELR-based explanation of weight decay, we establish a quantitative relation between matched ELR schedules and full loss trajectories, test its scope beyond weight decay, and identify conditions for its precision. Additional related work is discussed in Appendix G.

Functional scaling laws. Li et al. (2025a) showed that functional scaling laws (FSL) can predict loss dynamics in LLM pretraining under general LR schedules. We show that using ELR in place of LR improves predictive accuracy and enables the fitted FSL to transfer across norm-control methods without refitting. The resulting elr-FSL also explains delayed acceleration, where norm-controlled runs initially lag behind the baseline but later overtake it.

Concurrent work. Two concurrent studies examine norm-aware effective step sizes in LLM pretraining. Yang et al. (2026) show that expressing the optimal LR in terms of ELR reduces its nonlinear scaling and improves extrapolation across model sizes and data budgets. Xiao et al. (2026) derive an angular ELR and match it between weight-decayed and Hyperball-controlled Muon runs, approximately recovering the corresponding loss dynamics. Our focus is on whether matching ELR preserves full loss trajectories across different LR and norm realizations, and on the conditions under which this agreement is precise. We also investigate whether ELR enables a fitted FSL to transfer across norm-control methods without refitting.

3 ELR, Norm Control, and Collapse Metrics

ELR and norm control. Consider a matrix-valued parameter $\mathbf{W}_k$ with learning rate η_k at training step k. We define its effective learning rate (ELR) as

$$\eta_k^{\text{eff}} := \frac{\eta_k}{r_k}, \qquad r_k := \|\mathbf{W}_k\|_{\text{RMS}} = \frac{\|\mathbf{W}_k\|_F}{\sqrt{n}}, \tag{1}$$

where n is the number of entries in $\mathbf{W}_k$. This definition applies separately to each parameter matrix, each with its own ELR schedule; we suppress the matrix index unless needed.

Decoupled weight decay updates $\mathbf{W}_k$ according to

$$\mathbf{W}_{k+1} = \big(1 - \eta_k\lambda\big)\mathbf{W}_k - \eta_k\mathbf{U}_k,$$

where $\mathbf{U}_k$ denotes the optimizer update before multiplication by the learning rate. Weight decay introduces a shrinkage factor $1 - \eta_k\lambda$ into each parameter update, counteracting norm growth and thereby tending to increase the ELR $\eta_k^{\text{eff}} = \eta_k/r_k$. At a given LR, smaller parameter norms correspond to larger ELRs. Hyperball (Wen et al., 2026a), by contrast, constrains the parameter matrix to a sphere of prescribed Frobenius radius R:

$$\mathbf{W}_{k+1} = \text{Norm}_R(\mathbf{W}_k - \eta_k\,\text{Norm}_R(\mathbf{U}_k)), \qquad \text{Norm}_R(\mathbf{Q}) := R\,\frac{\mathbf{Q}}{\|\mathbf{Q}\|_F}.$$

Rewrite the inner update as $\mathbf{W}_k - \widetilde{\eta}_k \mathbf{U}_k$, where $\widetilde{\eta}_k = \eta_k R / \|\mathbf{U}_k\|_F$. The outer projection rescales this matrix without changing its direction. Thus, using the coefficient $\widetilde{\eta}_k$ in Eq. (1) and noting that $r_k = R/\sqrt{n}$, Hyperball's ELR is $\eta_k^{\text{eff}} = \frac{\widetilde{\eta}_k}{r_k} = \frac{\eta_k \sqrt{n}}{\|\mathbf{U}_k\|_F} = \frac{\eta_k}{\|\mathbf{U}_k\|_{\text{RMS}}}$.

Quantifying ELR collapse. Each collapse experiment is a paired controlled comparison. The target and comparison runs use the same parameter initialization and data order, with all other sources of training randomness held fixed. They differ only in the LR and/or norm-control intervention under study, resulting in different LR and parameter-norm trajectories while their ELR trajectories are matched. Let $\{L_k\}$ and $\{L_k^{\text{ref}}\}$ denote the loss trajectories of a matched run and its reference, respectively, each smoothed with an exponential moving average (EMA). We define the signed loss difference and mean collapse error as

$$\delta_k \coloneqq L_k - L_k^{\text{ref}}, \qquad \Delta_{\text{coll}} \coloneqq \frac{1}{|\mathcal{T}|} \sum_{k \in \mathcal{T}} |\delta_k|, \tag{2}$$

where $\mathcal{T}$ denotes the set of training steps over which collapse is evaluated.

Calibrating the scale of collapse error. To assess how small the observed collapse errors are, we compare them with two reference scales. First, recent LLM optimizer studies use 10^{-2}-scale loss differences to distinguish optimizer and hyperparameter choices (Liu et al., 2024; 2025; Wang et al., 2026a; 2025; Chen et al., 2026), providing a practical benchmark rather than a universal threshold. Second, we measure seed-induced variation in our most extensively tested configuration, Llama-124M on FineWeb. Changing only the initialization seed or data-order seed yields mean loss discrepancies of 1.11×10^{-2} and 1.62×10^{-2}, respectively, under the same metric (Appendix B.2.1). The collapse errors of a few $\times 10^{-3}$ observed in this setting are substantially below both practically relevant loss differences and ordinary seed-to-seed variation.

4 ELR Collapse: Precision, Conditions, and Scope

Matching ELR. Our protocol constructs runs with identical tensor-wise ELR schedules but substantially different LR and norm schedules. For each parameter matrix i, let $\boldsymbol{\eta}_i = (\eta_{i,k})_k$ and $\boldsymbol{r}_i = (r_{i,k})_k$ denote its LR and RMS norm trajectories. Given a reference run A, we choose a different LR schedule $\boldsymbol{\eta}_i^B$ for run B and prescribe

$$\boldsymbol{r}_i^B = \frac{\boldsymbol{\eta}_i^B}{\boldsymbol{\eta}_i^A} \odot \boldsymbol{r}_i^A, \qquad \frac{\boldsymbol{\eta}_i^B}{\boldsymbol{r}_i^B} = \frac{\boldsymbol{\eta}_i^A}{\boldsymbol{r}_i^A}, \tag{3}$$

where multiplication and division are elementwise. We enforce these norms by rescaling each controlled matrix at every step. All runs within each comparison share the same initialization and data order. Implementation details are given in Appendix A.2.

4.1 ELR Collapse across Training Configurations

We first demonstrate ELR collapse in Llama-124M and Qwen3-MoE-586M, both trained with AdamW. For each model, we construct four runs sharing a common warmup–stable–decay (WSD) ELR schedule but using different LR schedules: WSD, linear-up, linear-down, and sinusoidal. The WSD-LR run serves as the reference; the other three use norm schedules prescribed by Eq. (3).

Figure 1 shows that the loss curves nearly coincide throughout training, despite substantial differences in LR and norm. Mean collapse errors range from 1.8×10^{-3} to 2.6×10^{-3} for Llama and from 3.0×10^{-3} to 4.1×10^{-3} for Qwen3-MoE. Figure 2 shows that the loss differences remain small throughout training, despite fluctuations and occasional transient deviations. Thus, the small mean collapse errors reflect sustained trajectory-level agreement.

We next test the same protocol across a broader range of configurations:

- **Models:** Dense (Llama and Qwen3), MoE (Qwen3-MoE), and linear-attention models based on Kimi Delta Attention (KDA) (Zhang et al., 2025), ranging from **100M** to **1B** parameters;
- **Datasets:** FineWeb, FineWeb-Edu, C4, and OpenWebText;
- **Optimizers:** AdamW, Muon, and Signum.

Across the 31 ELR-matched comparisons in Table B.1, the median collapse error is 2.5×10^{-3}, and all remain below 5×10^{-3}. Experimental details and complete results are provided in Appendices A and B.2. Comparable precision also holds for ViTs trained on ImageNet (Figure B.4(b)), extending the evidence beyond language modeling.

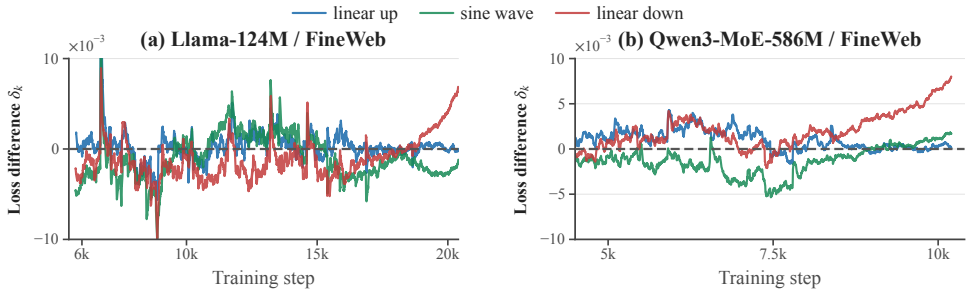


Figure 2: **ELR-matched runs maintain small loss differences throughout training.** Loss differences $\delta_k = L_k - L_k^{\text{ref}}$ for the linear-up, linear-down, and sinusoidal LR schedules over the post-warmup evaluation window. Both panels use the prescribed-ELR protocol of Figure 1; the dashed line marks zero difference.

Heterogeneous tensor-wise schedules. As a stress test, we also examine heterogeneous tensor-wise schedules in two settings: (i) all tensors share a common ELR schedule but use different LR–norm realizations; and (ii) tensors follow different ELR schedules, which are matched tensor by tensor between the two runs. Both settings retain comparable collapse precision (Appendix B.3). Thus, ELR collapse does not require either a shared LR–norm realization or a common ELR schedule across tensors.

4.2 When Is ELR Collapse Precise?

Transformers are not exactly scale invariant (Appendix B.1), so high-precision ELR collapse does not follow from an exact architectural symmetry. We therefore examine how its precision depends on model parameterization and training configuration.

QK-Norm and learnable RMSNorm gains improve collapse precision. To assess the role of normalization, we repeat the four-schedule ELR-matching experiment in Figures 1 and 2 under three configurations: the baseline with QK-Norm and learnable RMSNorm gains, without QK-Norm, and without QK-Norm with RMSNorm gains fixed. Training remains stable without QK-Norm, allowing us to distinguish its effect on collapse precision from its role in training stability.

Figure 3(a) shows that removing QK-Norm increases the mean collapse error from 2.3×10^{-3} to 5.2×10^{-3}. Further fixing the RMSNorm gains raises it to 1.84×10^{-2}, nearly an order of magnitude above the baseline. This is counterintuitive: *fixing the gains makes the parameterization more scale invariant, yet worsens collapse precision*.

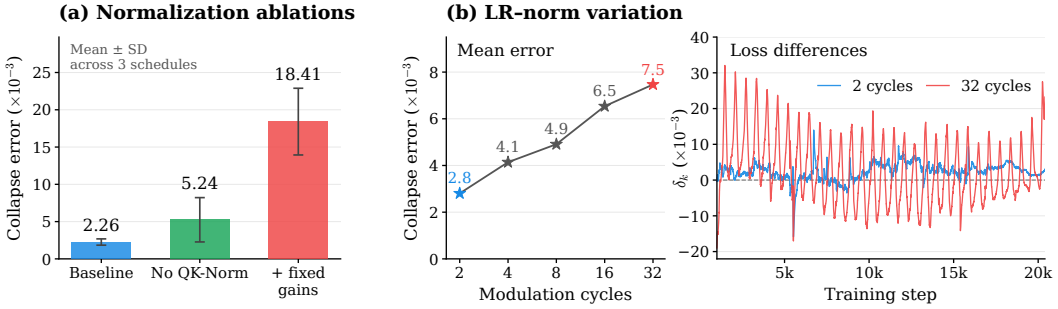


Figure 3: **Normalization and LR–norm variation affect ELR collapse precision.** Experiments use Llama-124M on FineWeb with AdamW. **(a)** Removing QK-Norm and fixing RMSNorm gains increase collapse error; bars show mean ± std across three matching runs. **(b)** Faster LR-norm variation increases mean collapse error (left) and oscillatory loss differences (right). LR and norms receive identical modulation, preserving ELRs.

Fast LR–norm variation degrades ELR collapse. Starting from a WSD LR schedule, we multiply both the LR and norm by the same sinusoidal factor $s_m(k) = 1 + 0.5 \sin(2\pi mk/N)$, where N is the training horizon and m determines the number of modulation cycles. Increasing m accelerates LR–norm variation while leaving the ELRs unchanged. Figure 3(b, left) shows that the mean collapse

error increases monotonically from 2.8×10^{-3} at 2 cycles to 7.5×10^{-3} at 32 cycles. Figures 3(b, right) and B.7 show that loss discrepancy oscillates at the imposed frequency, with larger amplitudes under faster modulation.

Implications. Together, these ablations show that ELR is an accurate but not exact coordinate for loss dynamics: collapse precision depends on normalization design and the timescale of LR–norm variation. These dependencies point to ELR collapse as an emergent property of training dynamics, rather than a consequence of static scale invariance. Identifying the mechanism behind this emergence is an important direction for future work.

5 ELR Synchronization in Standard Training

Section 4 establishes ELR collapse even under heterogeneous tensor-wise schedules. We now examine their structure in standard training, where tensors share a base LR schedule but their norms evolve freely (Appendix C).

A dominant rank-one structure. We collect the post-warmup tensor-wise ELRs into a matrix $\mathbf{E} = (\eta_{i,k}^{\text{eff}}) \in \mathbb{R}^{K \times N}$, where K is the number of parameter tensors and N is the number of post-warmup steps. Let σ_j denote its singular values in decreasing order. Across eight Llama-124M runs combining two optimizers, two LR schedules, and two weight-decay coefficients, the leading component captures $\rho_1 = \sigma_1^2 / \sum_j \sigma_j^2 = 97.59\%$–$99.61\%$ of the matrix energy; see Figure 4(a). Writing $\mathbf{a} = \sigma_1 \mathbf{u}_1$ and $\mathbf{s} = \mathbf{v}_1$, where $\mathbf{u}_1$ and $\mathbf{v}_1$ are the leading left and right singular vectors, gives $\mathbf{E} \approx \mathbf{a}\mathbf{s}^\top$, or

$$\eta_{i,k}^{\text{eff}} \approx \bar{\eta}_{i,k}^{\text{eff}} := a_i s_k, \tag{4}$$

where a_i is a tensor-specific amplitude and s_k is a shared temporal profile.

A shared ELR clock. Under this approximation, the trajectories coincide after rescaling by their fixed amplitudes: each tensor follows a shared ELR clock in a different unit. We call this approximate proportionality *ELR synchronization*. A shared nominal LR does not impose it, since tensor norms can evolve differently. It is an empirical property of the standard training examined here, rather than a requirement for collapse, which also holds under heterogeneous schedules.

Matching the synchronized component. A large fraction of ELR matrix energy need not imply that the leading component preserves loss dynamics. We therefore construct a comparison run whose post-warmup tensor-wise ELRs match the rank-one approximation $\bar{\eta}_{i,k}^{\text{eff}}$ of a reference AdamW run. The comparison preserves the shared profile and tensor-specific amplitudes while leaving the remaining components unmatched. Its loss trajectory closely follows the reference, with mean collapse error $\Delta_{\text{coll}} = 6.82 \times 10^{-3}$; see Figure 4(b). Thus, matching the synchronized component approximately recovers the reference loss dynamics in this setting, although less precisely than full tensor-wise ELR matching.

(a) Rank-one ELR structure

Leading-component energy ρ_1 (%)

Optimizer	$\lambda = 0$		$\lambda = 0.1$	
	WSD	Cosine	WSD	Cosine
AdamW	99.58	99.60	99.51	99.51
MuonW	99.61	99.58	97.59	98.00

(b) Rank-one ELR matching



Figure 4: **Approximate synchronization of tensor-wise ELRs. (a)** Fraction of ELR matrix energy captured by the leading singular component across optimizers, LR schedules, and weight-decay coefficients. **(b)** Loss trajectories of the reference run and a comparison run matching the reference's rank-one ELR approximation.

6 Weight Decay and Hyperball Act through ELR

We now test whether the effects of weight decay and Hyperball on loss dynamics can be reproduced by changing only a global LR schedule. Restricting the intervention to a single base LR avoids introducing additional tensor-wise preconditioning. Since this intervention cannot independently

match each tensor's ELR, we use an aggregate ELR as the matching criterion. Specifically, we consider the parameter-count-weighted ELR

$$\gamma_k := \sum_i w_i \eta_{i,k}^{\text{eff}}, \qquad w_i = \frac{n_i}{\sum_j n_j}, \tag{5}$$

where n_i is the number of parameters in tensor i. This choice is supported by its lower matching error than an unweighted mean in Appendix D.1; it is not derived from the rank-one structure.

In each experiment, we record the reference run's aggregate ELR schedule and adjust only the comparison run's base LR to match it. Both experiments use Llama-124M trained on FineWeb; matching details are given in Appendix A.2.

Weight decay. An AdamW run with $\lambda = 0.1$ serves as the reference. Removing weight decay ($\lambda = 0$) changes norm evolution and produces a distinct loss trajectory under the original LR schedule. ELR matching restores close loss alignment, with a mean collapse error of 7.64×10^{-3}; see Figure 5(a).

Hyperball. We use Muon with Hyperball (MuonH) as the reference and Muon with weight decay (MuonW) for comparison. Under its original LR schedule, MuonW follows a substantially different loss trajectory. Aggregate ELR matching recovers the MuonH loss trajectory after an initial period of instability, when the adapted LR is largest (Figure 5(b)). Once training stabilizes, the mean collapse error is 1.45×10^{-3}, evaluated from step 7,500 onward.

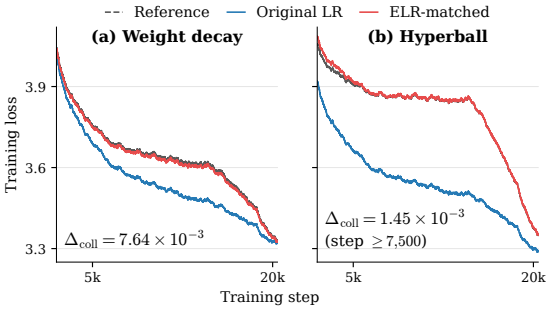


Figure 5: **LR-only ELR matching across norm-control methods.** Dashed gray: reference; blue: comparison with its original LR; red: comparison with its LR adapted to match the reference aggregate ELR. **(a)** AdamW without weight decay matches a reference with $\lambda = 0.1$. **(b)** MuonW matches a MuonH reference.

Appendix D.2 provides additional loss-trajectory diagnostics and tests the reverse interventions: the weight-decayed AdamW run matches the run without weight decay, and MuonH matches MuonW. Both again yield close loss collapse.

Together, these experiments show that, apart from the initial matching transient, the effects of weight decay and Hyperball on loss dynamics can be fully reproduced to high precision by adjusting only the base LR schedule to match their induced aggregate ELRs.

7 ELR Enables Scaling-Law Transfer

The LR-only interventions support aggregate ELR (5) as an accurate empirical coordinate for loss dynamics across norm-control methods. We now test its predictive value: can a scaling law fitted without Hyperball predict Hyperball loss trajectories when expressed in this coordinate?

Functional scaling laws. To model the effect of LR schedules on loss dynamics, we adopt the functional scaling law (FSL) proposed in Li et al. (2025a). Given an LR schedule $\boldsymbol{\eta} = (\eta_0, \ldots, \eta_N)$, define the intrinsic time $t_k := \sum_{j=0}^{k} \eta_j$. FSL models the loss at step k as

$$L_k = L_\infty + S(t_k) + N_k, \qquad N_k = \sum_{j=0}^{k} K(t_k - t_j)\eta_j^2, \tag{6}$$

where L_∞ is the irreducible loss. We parameterize the signal term and forgetting kernel as

$$S(t) = C_1(B_1 + t)^{-a_1}, \qquad K(t) = C_2(B_2 + t)^{-a_2}, \tag{7}$$

with seven fitted parameters $\Theta = (L_\infty, C_1, B_1, a_1, C_2, B_2, a_2)$. The signal term decreases with accumulated intrinsic time, whereas N_k captures the effect of gradient noise accumulated during training. Increasing the LR therefore accelerates signal learning but also increases noise injection through the factor η_j^2. We call this formulation **lr-FSL**.

We define **elr-FSL** by replacing the LR input η_j in Eq. (6) with a scalar ELR schedule, including in both the intrinsic time and the noise scale, while leaving the functional form unchanged. Standard

runs use the parameter-count-weighted ELR in Eq. (5); Hyperball runs use a fixed reference-scale calibration, detailed in Appendix E. Transfer without refitting therefore tests whether this empirical scalar description supports a common predictive law across norm-control methods. We use elr-FSL again in Section 8 to analyze delayed acceleration.

Transfer across norm-control methods. We train Llama-135M on FineWeb-Edu under ten configurations: four without weight decay, four with $\lambda = 0.1$, and two with Hyperball. For each FSL variant, we jointly fit a single parameter vector Θ to the same four non-Hyperball trajectories by minimizing mean squared prediction error. We then evaluate without refitting: four held-out non-Hyperball runs test generalization to new LR schedules under familiar norm-control methods (ID), while two Hyperball runs test transfer to an unseen norm-control method (OOD). Appendix E details the data split and fitting procedure.

Figure 6 summarizes the results. On the two unseen Hyperball runs, elr-FSL remains accurate without refitting, with a mean RMSE of 0.0172, whereas lr-FSL develops a large systematic bias and reaches 0.2301, a $13.40\times$ larger error. ELR also improves prediction on the held-out ID runs, reducing the mean RMSE from 0.0240 to 0.0119. Thus, the advantage of ELR is not limited to pairwise trajectory alignment: it enables a scaling law fitted under one set of norm-control methods to transfer to another.

(a) Prediction error (RMSE)

Evaluation	# Runs	lr-FSL ↓	elr-FSL ↓
Fit	4	0.0174	**0.0122**
ID: held-out	4	0.0240	**0.0119**
OOD: Hyperball	2	0.2301	**0.0172**

(b) Transfer to Hyperball

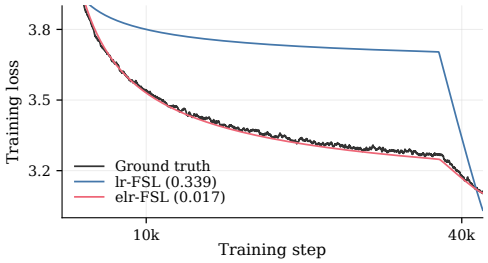


Figure 6: **ELR enables FSL to transfer to unseen Hyperball runs.** Both FSL variants are fitted on the same four non-Hyperball trajectories and evaluated without refitting. Experiments use Llama-135M trained on FineWeb-Edu with Adam, with no weight decay, decoupled weight decay, or Hyperball. **(a)** Mean RMSE across four fitting runs, four held-out ID runs, and two OOD-Hyperball runs. **(b)** Predictions for a representative Hyperball run: elr-FSL closely tracks the observed loss, whereas lr-FSL exhibits a large systematic deviation. Legend values report RMSE for this run.

8 Delayed Acceleration: Acquired Early, Revealed Late

Weight decay has been found important for optimization efficiency in large-scale LLM pretraining, yet its benefits can emerge only late: weight-decayed runs initially have higher loss than runs without weight decay, but eventually overtake them (Hoffmann et al., 2022; Liu et al., 2025; Wen et al., 2026b). This reversal makes early-loss comparisons misleading, as a beneficial norm-control method may initially appear detrimental. Figure 7(b) reproduces this behavior, and Hyperball exhibits a similar pattern (Appendix F). We refer to this late-emerging gain as **delayed acceleration** and use elr-FSL to explain it through the ELR schedule induced by norm control.

Norm growth prematurely suppresses ELR. Figure 7(a) reveals the origin of the reversal. Without weight decay, the parameter norm grows steadily, causing the ELR to decay substantially faster than the nominal LR. Weight decay restrains this norm growth and thereby sustains a larger ELR for longer. Thus, the two runs follow markedly different effective optimization trajectories despite sharing the same LR schedule. From the ELR perspective, the role of weight decay in loss dynamics is to prevent premature ELR decay.

The gain is acquired early and revealed late. Within elr-FSL, a larger ELR accelerates effective training time $t_k^{\text{eff}} = \sum_{j=0}^{k} \gamma_j$, reducing the signal-learning term $S(t_k^{\text{eff}})$, but also increases noise injection. Weight decay thus builds a signal-learning advantage that can initially be masked by noise. As ELR decays, noise injection diminishes and accumulated noise is forgotten, revealing the earlier advantage as lower loss. This account predicts that the decay window affects whether the advantage becomes visible before training ends. Consistent with this prediction, the weight-decayed run remains above the baseline when WSD decay occupies the final 10% of training, but overtakes it when this window extends to 30% (Appendix F.2).

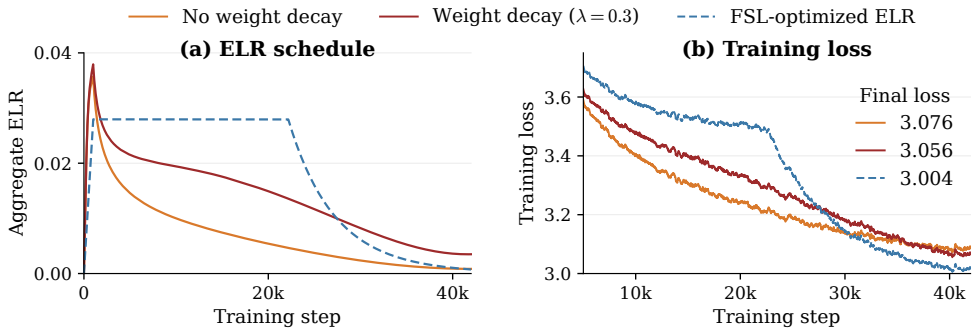


Figure 7: **ELR explains delayed acceleration and guides schedule design.** Experiments use Llama-135M trained on FineWeb-Edu. **(a)** Aggregate ELR schedules for AdamW with $\lambda = 0$ and $\lambda = 0.3$, and an FSL-optimized schedule realized with Hyperball. Weight decay sustains a larger ELR, qualitatively approaching the optimized schedule obtained from elr-FSL fitted to $\lambda = 0.3$ runs (Appendix F.1). **(b)** Observed training losses under these schedules. The weight-decayed run initially has higher loss but eventually overtakes the run without weight decay; the FSL-optimized schedule further lowers the final loss despite a larger initial disadvantage. The annotated final losses are averages over the last 50 raw training losses.

From explanation to ELR-first schedule design. To further test the elr-FSL explanation, we fit it to four AdamW trajectories with $\lambda = 0.3$, obtaining parameters $\widehat{\Theta}$, and optimize the ELR schedule:

$$\boldsymbol{\gamma}^\star \in \arg\min_{\boldsymbol{\gamma} \in \mathcal{A}} \mathcal{F}_N(\boldsymbol{\gamma}; \widehat{\Theta}), \tag{8}$$

where $\mathcal{F}_N$ is the final loss predicted by elr-FSL and $\mathcal{A}$ is the set of admissible schedules with a fixed training horizon, prescribed warmup, and peak-ELR constraint. We solve this problem using the variational approach of Li et al. (2026); see Appendix F.1.

Figure 7(a) shows that the optimized schedule $\boldsymbol{\gamma}^*$ sustains a high ELR before decaying; weight decay likewise sustains a larger ELR by counteracting norm growth. Realizing the optimized schedule with Hyperball yields a larger initial loss but a lower final loss than the plotted weight-decay baseline (Figure 7(b)), improving on our best Adam baseline by approximately 0.04 (Table F.1). These results support the elr-FSL account of delayed acceleration and demonstrate its utility for ELR-schedule design.

Appendix F provides additional experiments, including analogous delayed acceleration under Hyperball and its explanation through elr-FSL. Beyond delayed acceleration, elr-FSL also offers a qualitative interpretation of prior findings on weight-decay scaling and scheduling; see Appendix G.

9 Conclusion and Discussion

We establish ELR collapse as a quantitative empirical law within the regimes studied: matching tensor-wise ELR schedules aligns loss trajectories to within a few $\times 10^{-3}$ despite substantial differences in LR and norm evolution. ELR captures the joint effects of LR and norm on loss dynamics. Aggregate ELR matching further reproduces the effects of weight decay and Hyperball, while elr-FSL supports prediction across norm-control methods and explains delayed acceleration.

Mechanism. Collapse precision depends on normalization design and the timescale of LR–norm variation, pointing to training dynamics beyond static scale invariance. Standard training also exhibits approximate ELR synchronization, whose origin is unknown. Why do tensor-wise ELRs evolve proportionally, and why can a single aggregate accurately describe loss dynamics across runs? Rank-one structure alone does not determine the appropriate aggregation or guarantee that aggregate matching preserves loss. Explaining these observations and their limits is a central theoretical challenge. Matched losses need not imply matched representations or downstream performance.

Design and control. These findings suggest an ELR-first approach to pretraining: design an ELR schedule, then realize it through LR and norm control subject to stability constraints. Our FSL-optimized schedule provides an initial demonstration. A key next step is to determine how the desired ELR schedule should change with model size, batch size, and data budget, and whether this yields more reliable hyperparameter transfer.

AI USE STATEMENT

We used generative AI tools only to assist with literature searches and language editing. The authors read the original papers identified through these searches and reviewed the edited text to ensure that it preserved the intended technical meaning. The authors take responsibility for the final content of this paper, including all AI-assisted text.

REFERENCES

Anuj Apte. Scale weight decay and train better. *arXiv preprint arXiv:2607.23777*, 2026. (cited on page 32)

Shane Bergsma, Nolan Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. Power lines: Scaling laws for weight decay and batch size in LLM pre-training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 125153–125188, 2025. (cited on page 32)

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pp. 560–569, 2018. (cited on page 15)

Johan Bjorck, Alon Benhaim, Vishrav Chaudhary, Furu Wei, and Xia Song. Scaling optimal LR across token horizons. In *International Conference on Learning Representations*, 2025. (cited on page 32)

Lizhang Chen, Jonathan Li, Kaizhao Liang, Baiyu Su, Cong Xie, Nuo Wang Pierse, Chen Liang, Ni Lao, and Qiang Liu. Cautious weight decay. In *International Conference on Learning Representations*, 2026. (cited on page 4)

Jason Chuan-Chih Chou. Correction of decoupled weight decay. *arXiv preprint arXiv:2512.08217*, 2025. (cited on page 32)

Francesco D'Angelo, Maksym Andriushchenko, Aditya Varre, and Nicolas Flammarion. Why do we need weight decay in modern deep learning? In *Advances in Neural Information Processing Systems*, volume 37, pp. 23191–23223, 2024. (cited on pages 1 and 3)

Aaron Defazio. Why gradients rapidly increase near the end of training. *arXiv preprint arXiv:2506.02285*, 2025. (cited on page 32)

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. (cited on page 16)

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. (cited on page 16)

Aaron Gokaslan and Vanya Cohen. OpenWebText corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019. (cited on page 15)

Tessa Han, Sebastian Bordt, Hanlin Zhang, and Sham Kakade. Weight decay improves language model plasticity. In *International Conference on Machine Learning*, 2026. (cited on page 32)

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4246–4253, 2020. (cited on page 2)

Byeongho Heo, Sanghyuk Chun, Seong Joon Oh, Dongyoon Han, Sangdoo Yun, Gyuwan Kim, Youngjung Uh, and Jung-Woo Ha. AdamP: Slowing down the slowdown for momentum optimizers on scale-invariant weights. In *International Conference on Learning Representations*, 2021. (cited on page 3)

Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: Efficient and accurate normalization schemes in deep networks. In *Advances in Neural Information Processing Systems*, volume 31, pp. 2160–2170, 2018. (cited on page 3)

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030, 2022. (cited on pages 2 and 8)

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. In *Conference on Language Modeling*, 2024. (cited on page 16)

Maxim Kodryan, Ekaterina Lobacheva, Maksim Nakhodnov, and Dmitry P. Vetrov. Training scale-invariant neural networks on the sphere can happen in three regimes. In *Advances in Neural Information Processing Systems*, volume 35, pp. 14058–14070, 2022. (cited on page 3)

Atli Kosson, Bettina Messmer, and Martin Jaggi. Rotational equilibrium: How weight decay balances learning across neural networks. In *International Conference on Machine Learning*, pp. 25333–25369, 2024. (cited on page 32)

Atli Kosson, Jeremy Welborn, Yang Liu, Martin Jaggi, and Xi Chen. Weight decay may matter more than μP for learning rate transfer in practice. In *International Conference on Learning Representations*, 2026. (cited on page 1)

Anders Krogh and John A. Hertz. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, volume 4, pp. 950–957, 1991. (cited on page 1)

Binghui Li, Fengling Chen, Zixun Huang, Lean Wang, and Lei Wu. Functional scaling laws in kernel regression: Loss dynamics and learning rate schedules. In *Advances in Neural Information Processing Systems*, volume 38, pp. 101211–101269, 2025a. (cited on pages 2, 3, and 7)

Binghui Li, Zilin Wang, Fengling Chen, Shiyang Zhao, Ruiheng Zheng, and Lei Wu. Optimal learning rate schedules under functional scaling laws: Power decay and warmup-stable-decay. *arXiv preprint arXiv:2602.06797*, 2026. (cited on pages 9 and 29)

Bingrui Li, Jiaxin Wen, Zhanpeng Zhou, Jun Zhu, and Jianfei Chen. Efficient hyperparameter tuning via trajectory invariance principle. *arXiv preprint arXiv:2509.25049*, 2025b. (cited on page 3)

Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. In *International Conference on Learning Representations*, 2020. (cited on page 3)

Zhiyuan Li, Kaifeng Lyu, and Sanjeev Arora. Reconciling modern deep learning with traditional optimization analyses: The intrinsic learning rate. In *Advances in Neural Information Processing Systems*, volume 33, pp. 14544–14555, 2020. (cited on pages 2 and 3)

Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training. In *International Conference on Learning Representations*, 2024. (cited on page 4)

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. *arXiv preprint arXiv:2502.16982*, 2025. (cited on pages 2, 4, 8, and 15)

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. (cited on page 15)

Ilya Loshchilov. Weight norm control. *arXiv preprint arXiv:2311.11446*, 2023. (cited on page 1)

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. (cited on page 1)

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. In *Advances in Neural Information Processing Systems*, volume 37, pp. 30811–30849, 2024. (cited on page 15)

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. (cited on page 15)

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. (cited on page 15)

Twan van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017. (cited on pages 2 and 3)

Ruosi Wan, Zhanxing Zhu, Xiangyu Zhang, and Jian Sun. Spherical motion dynamics: Learning dynamics of normalized neural network using SGD and weight decay. In *Advances in Neural Information Processing Systems*, volume 34, pp. 6380–6391, 2021. (cited on page 3)

Jinbo Wang, Mingze Wang, Zhanpeng Zhou, Junchi Yan, Weinan E, and Lei Wu. The sharpness disparity principle in transformers for accelerating language model pre-training. In *International Conference on Machine Learning*, pp. 64859–64879, 2025. (cited on page 4)

Jinbo Wang, Mingze Wang, Jiaqi Zhang, Wei Wang, Peng Pei, Xunliang Cai, Weinan E, and Lei Wu. GradPower: Powering gradients for faster language model pre-training. In *International Conference on Machine Learning*, 2026a. (cited on page 4)

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. *arXiv preprint arXiv:2605.26895*, 2026b. (cited on page 25)

Xi Wang and Laurence Aitchison. How to set AdamW's weight decay as you scale model and dataset size. In *International Conference on Machine Learning*, pp. 62222–62250, 2025. (cited on page 32)

Kaiyue Wen, Xingyu Dang, Kaifeng Lyu, Tengyu Ma, and Percy Liang. Fantastic pretraining optimizers and where to find them II: Hyperball optimization. *arXiv preprint arXiv:2606.16899*, 2026a. (cited on pages 1, 2, 3, and 31)

Kaiyue Wen, David Hall, Tengyu Ma, and Percy Liang. Fantastic pretraining optimizers and where to find them. In *International Conference on Learning Representations*, 2026b. (cited on page 8)

Mitchell Wortsman, Peter J. Liu, Lechao Xiao, Katie E. Everett, Alexander A. Alemi, Ben Adlam, John D. Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, Jeffrey Pennington, Jascha Sohl-Dickstein, Kelvin Xu, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Small-scale proxies for large-scale transformer training instabilities. In *International Conference on Learning Representations*, 2024. (cited on page 32)

Yihao Xiao, Jialong Sun, Zitian Gao, Zeming Wei, Chutian Wang, Ran Tao, Jiaye Teng, and Bryan Dai. Hyperball may not be a free lunch. *arXiv preprint arXiv:2607.22444*, 2026. (cited on page 3)

Tian Xie, Haoming Luo, Haoyu Tang, Yiwen Hu, Jason Klein Liu, Qingnan Ren, Yang Wang, Wayne Xin Zhao, Rui Yan, Bing Su, Chong Luo, and Baining Guo. Controlled LLM training on spectral sphere. *arXiv preprint arXiv:2601.08393*, 2026. (cited on page 1)

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. (cited on page 15)

Zaiwen Yang, Huaqing Zhang, Jing Xu, and Jingzhao Zhang. On the nonlinearity of learning rate scaling for LLM training. *arXiv preprint arXiv:2606.29158*, 2026. (cited on page 3)

Kazuki Yano, Shun Kiyono, Sosuke Kobayashi, Sho Takase, and Jun Suzuki. Pre-training LLM without learning rate decay enhances supervised fine-tuning. In *International Conference on Learning Representations*, 2026. (cited on page 33)

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. (cited on page 15)

Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger Grosse. Three mechanisms of weight decay regularization. In *International Conference on Learning Representations*, 2019. (cited on page 32)

Yu Zhang, Zongyu Lin, Xingcheng Yao, Jiaxi Hu, Fanqing Meng, Chengyin Liu, Xin Men, Songlin Yang, Zhiyuan Li, Wentao Li, Enzhe Lu, Weizhou Liu, Yanru Chen, Weixin Xu, Longhui Yu, Yejie Wang, Yu Fan, Longguang Zhong, Enming Yuan, Dehao Zhang, Yizhi Zhang, T. Y. Liu, Haiming Wang, Shengjun Fang, Weiran He, Shaowei Liu, Yiwei Li, Jianlin Su, Jiezhong Qiu, Bo Pang, Junjie Yan, Zhejun Jiang, Weixiao Huang, Bohong Yin, Jiacheng You, Chu Wei, Zhengtao Wang, Chao Hong, Yutian Chen, Guanduo Chen, Yucheng Wang, Huabin Zheng, Feng Wang, Yibo Liu, Mengnan Dong, Zheng Zhang, Siyuan Pan, Wenhao Wu, Yuhao Wu, Longyu Guan, Jiawen Tao, Guohong Fu, Xinran Xu, Yuzhi Wang, Guokun Lai, Yuxin Wu, Xinyu Zhou, Zhilin Yang, and Yulun Du. Kimi Linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025. (cited on pages 4 and 15)

Appendix

Table of Contents

A Experimental Details **15**
- A.1 Training Configurations . . . 15
- A.2 ELR-Matching Protocols . . . 16

B ELR Collapse: Nontriviality, Scope, and Conditions (Supplement to Section 4) **17**
- B.1 Why Scale Invariance Does Not Explain ELR Collapse . . . 17
- B.2 Quantitative Evidence across Training Configurations . . . 18
- B.3 ELR Collapse under Tensor-wise Heterogeneity . . . 21
- B.4 Conditions for Precise ELR Collapse . . . 22

C ELR Synchronization and Rank-One Matching (Supplement to Section 5) **23**

D LR-only Interventions across Norm-Control Methods (Supplement to Section 6) **24**
- D.1 Choice of Aggregate ELR . . . 24
- D.2 Weight Decay and Hyperball . . . 25

E Functional Scaling Laws (Supplement to Section 7) **25**

F Delayed Acceleration (Supplement to Section 8) **28**
- F.1 Optimal ELR Schedule . . . 28
- F.2 Acquired Early, Revealed Late: A Controlled WSD Test . . . 30
- F.3 Delayed Acceleration under Hyperball . . . 31

G Additional Related Work: Weight Decay and ELR **32**

A Experimental Details

This appendix specifies the shared training configurations and the ELR matching protocols used throughout the paper. Appendix B.2 then reports the corresponding trajectory-level results.

A.1 Training Configurations

Tables A.1 and A.2 summarize the language model training configurations. Below we provide further details, followed by the ViT configuration.

Language models. We consider several Transformer architectures in our pretraining experiments.

- **Llama.** Llama (Touvron et al., 2023) is a popular dense decoder-only Transformer architecture, incorporating Rotary Positional Encoding (RoPE), Swish-Gated Linear Units (SwiGLU), and root mean square layer normalization (RMSNorm) (Zhang & Sennrich, 2019). Our 135M implementation follows this and is augmented with QK-Norm. Our 124M and 1B parameter implementations are adapted from Modded-NanoGPT[1]. They use tied embeddings, QK-Norm, and a GPT-2 tokenizer.
- **Qwen3.** Qwen3 (Yang et al., 2025) is a family of decoder-only language models that includes both dense and mixture-of-experts (MoE) variants. Our Qwen3 Dense and Qwen3-MoE implementations are adapted from the official Qwen3 release. Compared with Llama, Qwen3 uses Grouped-Query Attention and QK-Norm. Our Qwen3-MoE model replaces the dense MLP with a sparse 32-expert MoE layer using top-4 routing.
- **Kimi Delta Attention (KDA).** KDA is a gated linear-attention mechanism introduced in Zhang et al. (2025). Our local implementation replaces three out of every four standard self-attention layers with KDA, giving a 3:1 KDA to standard attention ratio.

Datasets. Models are pretrained on the following datasets:

- **FineWeb and FineWeb-Edu.** FineWeb (Penedo et al., 2024) is a large-scale English web dataset derived from Common Crawl. FineWeb-Edu is a subset of FineWeb filtered for educational quality.
- **Colossal Clean Crawled Corpus (C4)** (Raffel et al., 2020). It is a large-scale public language dataset widely used for LLM pretraining, including T5 (Raffel et al., 2020).
- **OpenWebText** (Gokaslan & Cohen, 2019). It is an open-source recreation of the WebText corpus and is extensively used for LLM pretraining, including RoBERTa (Liu et al., 2019) and nanoGPT[2].

Table A.1: **Dense model configurations and peak learning rates used in the ELR-collapse experiments.** Q/KV lists the numbers of query and key/value heads.

Model	size	d_{model}	d_{FF}	Q/KV	depth	peak lr	final lr
Llama-124M	124M	768	3072	6	12	1.8×10^{-3}	0
Llama-135M	135M	576	1536	9/3	30	6×10^{-4}	6×10^{-5}
Llama-1B	1.07B	2560	10240	20	12	3×10^{-4}	0
Qwen3-145M	145M	768	2304	12/6	12	1.8×10^{-3}	0
KDA-127M	127M	768	3072	6	12	1.8×10^{-3}	0

Language model training recipe. Unless stated otherwise, we use the following configuration for language model experiments. All runs within each comparison share the same initialization and data-order seeds, eliminating the seed-induced variability discussed in Appendix B.2.1. We use AdamW with $\beta_1 = 0.9$ and $\beta_2 = 0.95$, the MoonShot variant of Muon (Liu et al., 2025) with momentum coefficient $\beta = 0.95$, and Signum (Bernstein et al., 2018) with momentum coefficient $\beta = 0.95$. Unless otherwise specified, weight decay is set to zero; nonzero weight decay is used

[1] https://github.com/KellerJordan/modded-nanogpt
[2] https://github.com/karpathy/nanoGPT

Table A.2: **Qwen3-MoE configuration and peak learning rate used in the ELR-collapse experiment.** Q/KV lists the numbers of query and key/value heads; top-k gives the number of active experts per token.

Model	size	d_{model}	d_{FF}	Q/KV	depth	n_{experts}	top-k	peak lr	final lr
Qwen3-MoE	586M	768	576	12/4	12	32	4	1.8×10^{-3}	0

only when explicitly indicated. The default LR schedule is warmup–stable–decay (WSD) (Hu et al., 2024). Unless otherwise specified, all language model experiments use batch size $B = 128$ and sequence length 1024. All language model experiments use BF16 precision.

Vision Transformer experiments. We train a 9.8M-parameter pre-norm Vision Transformer (ViT) (Dosovitskiy et al., 2021) on ImageNet-1K (Deng et al., 2009) with 64×64 inputs. The model has 12 layers, hidden size 256, four attention heads, an MLP expansion ratio of 4, and 8×8 patches; it uses RMSNorm, QK-Norm, and ReLU activations. We train in FP32 for 50,000 steps using AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$, no weight decay, and a global batch size of 512. The reference run uses a WSD LR schedule that warms up to 6×10^{-4} over 1,250 steps, remains constant until step 45,000, and decays to 6×10^{-5} over the remaining steps.

Loss smoothing. For all collapse and ELR-matching experiments, we apply an exponential moving average with $\alpha_{\text{EMA}} = 0.99$ to the raw loss at each training step. The smoothing is performed independently for each run before computing the loss difference δ_k and mean collapse error Δ_{coll}. Both the plotted loss curves and the reported collapse errors are therefore based on the same smoothed losses. Unless otherwise stated, $\mathcal{T}$ comprises all post-warmup training steps.

A.2 ELR-Matching Protocols

The conversion from LR to ELR depends on how the optimizer parameterizes its update. We write $\mathbf{W}_i(k)$ for a matrix-valued parameter immediately before training step k, $\mathbf{U}_i(k)$ for its unscaled optimizer update, and $\eta_i(k)$ for its LR. Let n_i denote the number of parameters in tensor i. When $\eta_i(k)$ directly multiplies $\mathbf{U}_i(k)$, the ELR is

$$\eta_i^{\text{eff}}(k) = \frac{\eta_i(k)}{\|\mathbf{W}_i(k)\|_{\text{RMS}}}.$$

However, Hyperball requires a different conversion because it normalizes the update:

$$\mathbf{W}_i(k+1) = \texttt{Norm}_{R_i}(\mathbf{W}_i(k) - \eta_i(k)\texttt{Norm}_{R_i}(\mathbf{U}_i(k))), \qquad \texttt{Norm}_{R_i}(\mathbf{Q}) \coloneqq R_i \frac{\mathbf{Q}}{\|\mathbf{Q}\|_F}.$$

The inner update can be written as

$$\mathbf{W}_i(k) - \eta_i(k)\texttt{Norm}_{R_i}(\mathbf{U}_i(k)) = \mathbf{W}_i(k) - \frac{\eta_i(k) R_i}{\|\mathbf{U}_i(k)\|_F}\mathbf{U}_i(k).$$

Because Hyperball maintains $\|\mathbf{W}_i(k)\|_F = R_i$, its ELR is therefore

$$\eta_i^{\text{eff}}(k) = \frac{\eta_i(k) R_i / \|\mathbf{U}_i(k)\|_F}{R_i/\sqrt{n_i}} = \frac{\eta_i(k)\sqrt{n_i}}{\|\mathbf{U}_i(k)\|_F} = \frac{\eta_i(k)}{\|\mathbf{U}_i(k)\|_{\text{RMS}}}.$$

We use these optimizer-specific definitions in both matching protocols below.

Intervention timing. Let k_0 denote the training step at which control begins. Unless otherwise stated, k_0 is the first step after LR warmup. Before k_0, the compared runs follow the same configuration and enter the controlled phase from the same parameter state. All ELR schedules below refer to the controlled phase $k \geqslant k_0$.

We use two matching protocols. The first explicitly controls the LR and parameter norm schedule to realize a prescribed ELR schedule. The second adapts only the LR to reproduce the ELR induced by a reference run.

Matching a prescribed ELR schedule. Given a target ELR schedule $\{\eta_k^{\text{eff}}\}_{k \geqslant k_0}$, we construct multiple runs with different LR and norm schedules. For an optimizer whose LR directly multiplies

its update, we choose an LR schedule $\{\eta_k\}_{k\geqslant k_0}$ and define the corresponding norm schedule by

$$\rho_k := \frac{\eta_k}{\eta_k^{\text{eff}}}.$$

We scale the LR schedule so that $\rho_i(k_0) = \|\mathbf{W}_i(k_0)\|_{\text{RMS}}$, ensuring continuity at the start of the controlled phase. At step k, we first apply the underlying optimizer update with LR $\eta_i(k)$, obtaining a provisional parameter $\widetilde{\mathbf{W}}_i(k+1)$, and then rescale it to the next target RMS:

$$\mathbf{W}_i(k+1) \leftarrow \rho_i(k+1)\frac{\widetilde{\mathbf{W}}_i(k+1)}{\|\widetilde{\mathbf{W}}_i(k+1)\|_{\text{RMS}}}.$$

Consequently, $\|\mathbf{W}_i(k)\|_{\text{RMS}} = \rho_i(k)$ and

$$\frac{\eta_i(k)}{\|\mathbf{W}_i(k)\|_{\text{RMS}}} = \frac{\eta_i(k)}{\rho_i(k)} = \eta_i^{\text{eff}}(k)$$

throughout the controlled phase. Repeating this construction with different LR schedules produces distinct LR and RMS trajectories that realize the same prescribed tensor-wise ELR schedules.

For a Hyperball run, its prescribed radius supplies the norm control. After computing $\mathbf{U}_i(k)$, we instead set

$$\eta_i(k) = \eta_i^{\text{eff}}(k)\|\mathbf{U}_i(k)\|_{\text{RMS}},$$

which realizes the same prescribed ELR under the Hyperball definition above.

Reference-run ELR matching. This protocol does not prescribe the norm trajectory of the matched run. Its optimizer and norm-control mechanism are left unchanged, and only a single shared LR is adapted. This allows us to test whether a norm-control method affects loss dynamics through the aggregate ELR it induces.

Let run A be the reference run and run B the matched run. We first train run A and record its parameter-count-weighted aggregate ELR:

$$\eta_k^{\text{eff,agg},A} = \begin{cases} \eta_k^A \displaystyle\sum_i \frac{w_i}{\|\mathbf{W}_i^A(k)\|_{\text{RMS}}}, & \text{if run A uses a direct-update optimizer,} \\ \eta_k^A \displaystyle\sum_i \frac{w_i}{\|\mathbf{U}_i^A(k)\|_{\text{RMS}}}, & \text{if run A uses Hyperball.} \end{cases}$$

Here, w_i are the parameter-count weights defined in Equation (5), and the sums run over the controlled tensors. When training run B, we set

$$\eta_k^B = \begin{cases} \eta_k^{\text{eff,agg},A}\left(\displaystyle\sum_i \frac{w_i}{\|\mathbf{W}_i^B(k)\|_{\text{RMS}}}\right)^{-1}, & \text{if run B uses a direct-update optimizer,} \\ \eta_k^{\text{eff,agg},A}\left(\displaystyle\sum_i \frac{w_i}{\|\mathbf{U}_i^B(k)\|_{\text{RMS}}}\right)^{-1}, & \text{if run B uses Hyperball.} \end{cases}$$

Thus, the matched run tracks the reference aggregate ELR while retaining its own norm evolution and norm-control mechanism.

B ELR Collapse: Nontriviality, Scope, and Conditions (Supplement to Section 4)

B.1 Why Scale Invariance Does Not Explain ELR Collapse

Our ELR is defined using the norm of the matrix-valued parameters. Transformers, however, also contain trainable vector-valued parameters, most notably the biases of FFN layers and gains in normalization layers. Let $\mathcal{L}(\mathbf{W}, \boldsymbol{\phi})$ denote the loss, where $\mathbf{W}$ collects the matrix-valued parameters and $\boldsymbol{\phi}$ the vector-valued parameters. The strongest relevant symmetry one might posit is matrix-only scale invariance:

$$\mathcal{L}(c\mathbf{W}, \boldsymbol{\phi}) = \mathcal{L}(\mathbf{W}, \boldsymbol{\phi}), \qquad \forall\, c > 0. \tag{B.1}$$

However, even if this symmetry held exactly, matching the ELR would not by itself imply matching loss trajectories since the loss is sensitive to the scale of ϕ.

More seriously, the matrix-only symmetry in Eq. (B.1) also fails for several architectural reasons:

- **Pre-norm residual blocks.** A pre-norm residual block updates the hidden state as
$$\mathbf{h}^{\ell+1} = \mathbf{h}^{\ell} + F_{\ell}\big(\mathrm{RMSNorm}_{\boldsymbol{\gamma}_{\ell}}(\mathbf{h}^{\ell}); \mathbf{W}_{\ell}\big)\,. \tag{B.2}$$
The identity path preserves the scale of $\mathbf{h}^{\ell}$, whereas RMSNorm removes it before the residual branch. Thus, rescaling the upstream matrices that determine $\mathbf{h}^{\ell}$ affects the two paths differently. Rescaling $\mathbf{W}_{\ell}$ likewise changes the residual branch relative to the identity path. Hence, the block is not invariant under a rescaling of its matrix parameters.
- **Value and output projections.** The attention value–output path contains the product
$$(\mathbf{H}\mathbf{W}_V)\mathbf{W}_O.$$
Jointly rescaling $\mathbf{W}_V$ and $\mathbf{W}_O$ by c scales this path by c^2, even if the attention weights remain fixed. Its magnitude relative to the residual stream therefore changes.
- **SwiGLU feed-forward networks.** The SwiGLU FFN combines several matrix scales:
$$\mathrm{MLP}(\mathbf{H}) = [\mathrm{SiLU}(\mathbf{H}\mathbf{W}_G) \odot (\mathbf{H}\mathbf{W}_U)]\,\mathbf{W}_D. \tag{B.3}$$
Its multiplicative gate couples the scales of multiple projections and moreover, SiLU is not homogeneous. Consequently, a common rescaling of $\mathbf{W}_G$, $\mathbf{W}_U$, and $\mathbf{W}_D$ cannot be canceled by a single global scale factor.
- **Token embeddings and output logits.** Rescaling the token embeddings changes the scale of the initial residual stream, which is not eliminated by subsequent residual blocks. Rescaling the head layer changes the logit scale and hence the effective softmax temperature, to which cross-entropy is not invariant. When the embedding and output matrices are tied, the same parameter introduces scale dependence at both ends of the network.

B.2 Quantitative Evidence across Training Configurations

We first calibrate the magnitude of collapse errors against seed-induced loss variation in a representative configuration. We then report additional validation across language model configurations and extend the experiments to vision transformers. Training configurations and ELR-matching protocols are given in Appendix A.

B.2.1 Interpreting Collapse Errors at the 10^{-3} Scale

To put the observed collapse errors into context, we measure how much training-loss trajectories differ when only the initialization seed or data-order seed changes. This provides a reference scale for the representative Llama-124M/FineWeb configuration, rather than a universal threshold for collapse. Fix a source of stochasticity c, and let $L_{i,k}$ denote the smoothed loss of run i at step k. To match the definition of collapse error, we measure run-to-run variation using the pairwise mean absolute discrepancy

$$d_{ij} \coloneqq \frac{1}{|\mathcal{T}|}\sum_{k\in\mathcal{T}} |L_{i,k} - L_{j,k}|\,, \qquad \text{for } 1 \leqslant i < j \leqslant n,$$

where $\mathcal{T}$ is the set of evaluation steps. We then define the stochastic variation associated with c by averaging over all unordered pairs:

$$\Delta_{\mathrm{stoch}}(c) \coloneqq \frac{2}{n(n-1)} \sum_{1\leqslant i<j\leqslant n} d_{ij}.$$

This definition uses the same mean absolute trajectory discrepancy as Δ_{coll}, allowing a direct comparison between ELR-collapse errors and ordinary run-to-run variation.

Figure B.1 reports run-to-run variation for Llama-124M trained on FineWeb with AdamW. For each source of stochasticity, we perform four runs with the same configuration, varying only the

initialization seed or only the data-order seed. Averaging over the six unordered run pairs in each group gives

$$\Delta_{\text{stoch}}(\text{init}) = 1.11 \times 10^{-2}, \qquad \Delta_{\text{stoch}}(\text{data-order}) = 1.62 \times 10^{-2}.$$

For comparison, the three ELR-matched runs in Figure 1 under the same training configurations, which keep both seeds fixed, have collapse errors between 1.8×10^{-3} and 2.6×10^{-3}. Thus, even the largest collapse error is more than four times smaller than the lower stochastic-variation baseline; across the reported values, the difference is a factor of 4.3–9.0.

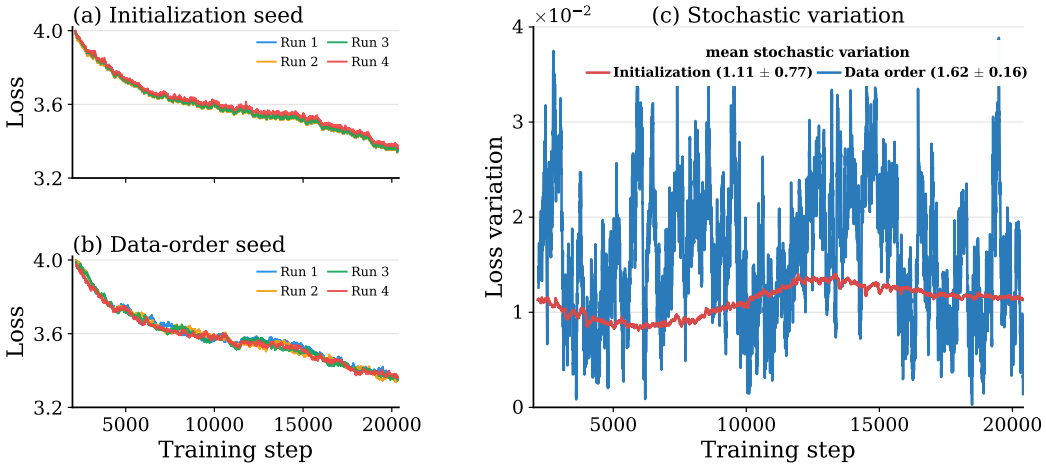


Figure B.1: **Run-to-run loss variation substantially exceeds ELR-collapse error. (a,b)** Training-loss trajectories from repeated runs differing only in initialization seed (a) or data-order seed (b). **(c)** The step-wise mean pairwise loss discrepancy over training. Legend values report the mean and standard deviation of the pairwise time-averaged discrepancies across unordered run pairs, in units of 10^{-2}. The resulting time-averaged variations are $\Delta_{\text{stoch}}(\text{init}) = 1.11 \times 10^{-2}$ and $\Delta_{\text{stoch}}(\text{data-order}) = 1.62 \times 10^{-2}$. By comparison, the three ELR-matched Llama-124M runs in Figures 1 and 2, with both seeds fixed, have collapse errors of 1.8×10^{-3}, 2.5×10^{-3}, and 2.6×10^{-3}.

B.2.2 Broader Validation of ELR Collapse

Table B.1 summarizes the prescribed-ELR validation experiments for language models, including those presented in Section 4.1, using the mean collapse error defined in Section 3. The table reports the principal configurations, spanning model architectures, model sizes, pretraining datasets, and optimizers. Unless otherwise noted, each setting compares three runs using linear-up, linear-down, and sinusoidal LR schedules against a reference run. Their target-norm schedules are adjusted so that the ELR of each controlled tensor matches that of the corresponding tensor in the reference run. Below, we provide trajectory diagnostics for additional language model experiments and the extension to vision transformers. The heterogeneous-schedule comparisons included in the table are described separately in Appendix B.3.

1B-scale stress test. As a scale stress test, we repeat the prescribed-ELR experiment with a Llama-1B model trained on FineWeb. Figure B.2 reports the full LR, ELR, loss, and loss difference trajectories for the linear-up and linear-down realizations. Each yields a mean collapse error of 2.0×10^{-3}, showing that high-precision ELR collapse persists at the 1B scale.

Table B.1: **Collapse errors across prescribed-ELR validation settings.** Each row compares runs with different LR and parameter-norm schedules that realize the same prescribed ELR schedule. Entries report the mean collapse error Δ_{coll}, in units of 10^{-3}, for linear-up, linear-down, and sinusoidal LR schedules. An em dash denotes a schedule that was not evaluated.

No.	Validation setting	Model	Dataset	Δ_{coll} $(\times 10^{-3})$			Figure
				Up	Down	Sine	
Experiments reported in the main text							
1	Dense Llama	Llama-124M	FineWeb	1.8	2.5	2.6	Fig. 1
2	MoE	Qwen3-MoE 586M	FineWeb	3.0	4.1	3.3	Fig. 1
Additional experiments reported in the appendix							
3	1B scale	Llama-1B	FineWeb	2.0	2.0	—	Fig. B.2
4	Qwen3 Dense	Qwen3-145M	FineWeb	1.2	1.9	2.7	Fig. B.3(a)
5	Linear attention	KDA-127M	FineWeb	0.9	1.7	3.2	Fig. B.3(b)
6	C4	Qwen3-145M	C4	2.1	2.1	2.5	Fig. B.3(c)
7	OpenWebText	Qwen3-145M	OpenWebText	3.5	3.4	3.5	Fig. B.3(d)
8	Signum	Llama-124M	FineWeb	4.8	1.6	4.9	Fig. B.3(e)
9	Muon	Llama-124M	FineWeb	1.3	1.8	2.5	Fig. B.3(f)
10	FineWeb-Edu	Llama-135M	FineWeb-Edu	3.4	3.3	4.4	Fig. B.4(a)
Tensor-wise heterogeneity							
11	Heterogeneous norms	Llama-124M	FineWeb		0.91		Fig. B.5
12	Heterogeneous ELRs	Llama-124M	FineWeb		1.77		Fig. B.5

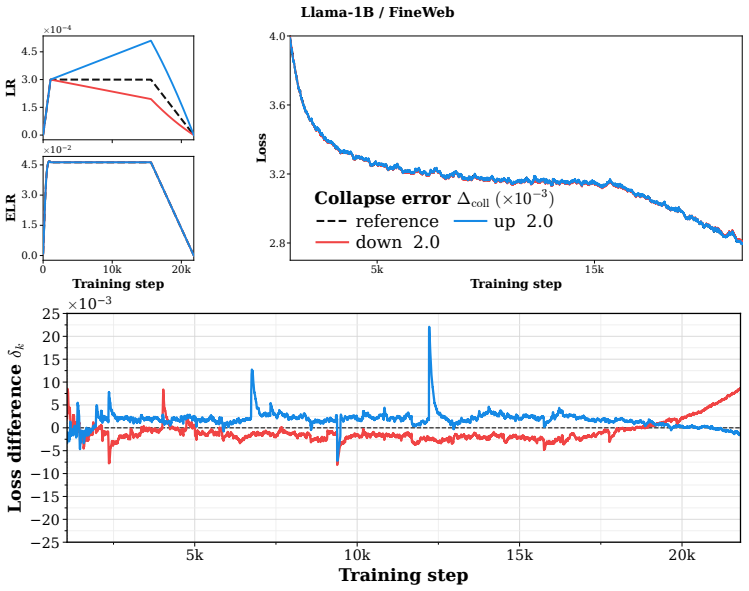


Figure B.2: **ELR collapse in Llama-1B on FineWeb.** The compact panels show LR and ELR. The loss panel compares the two varied schedules with the constant reference and reports their mean absolute collapse errors.

Broader validation. Figure B.3 reports trajectory-level diagnostics for the additional experiments. These extend the validation to dense Qwen3 and KDA, to C4 and OpenWebText, and to the Signum and Muon optimizers. Each panel follows the LR–ELR–loss layout of Figure 1.

Figure B.4(a) shows ELR collapse in Llama-135M trained on FineWeb-Edu, the configuration used in our FSL experiments in Sections 7 and 8. For Llama-135M, we use batch size $B = 64$ and sequence length 1024.

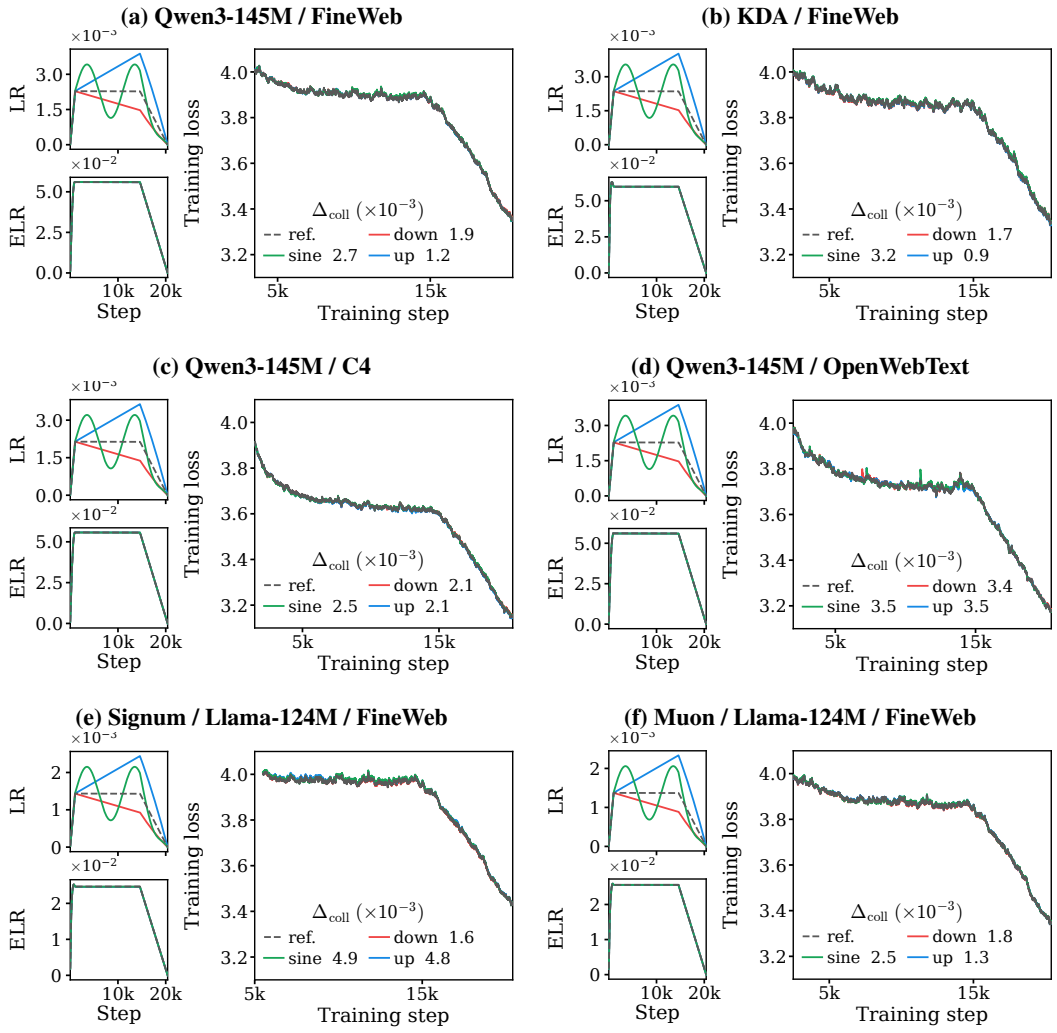


Figure B.3: **Additional validation of ELR collapse across training configurations.** **(a)** Replicates the prescribed-ELR experiment with Qwen3-145M on FineWeb. **(b)** Applies the same protocol to KDA. **(c,d)** Repeat the Qwen3-145M experiment on C4 and OpenWebText, respectively. **(e,f)** Evaluate Signum and Muon with Llama-124M, respectively. Together, these experiments test ELR collapse across model architectures, pretraining datasets, and optimizers. In each subfigure, the loss-panel legend reports the mean absolute collapse error relative to the constant-schedule reference run.

Extension to vision transformers. We extend the prescribed-ELR experiment to image classification using the ViT configuration in Appendix A. The controlled parameter collections comprise the patch-projection affine parameters, the class-token and positional embeddings, and the affine parameters in all attention and MLP blocks. The RMSNorm gains and classifier head retain their base learning rates and are excluded from the prescribed-norm projection.

Figure B.4(b) shows that the three nonconstant branches remain aligned with the reference run, with mean collapse errors $\Delta_{\text{coll}} \in [2.9, 3.0] \times 10^{-3}$. This extends the 10^{-3}-scale ELR collapse observed in language model pretraining to this ViT setting.

B.3 ELR Collapse under Tensor-wise Heterogeneity

We next test whether collapse depends on using a common schedule across tensors. We introduce heterogeneity first in the norm schedules and then in the ELR targets, while matching the ELR of each corresponding tensor between the two runs.

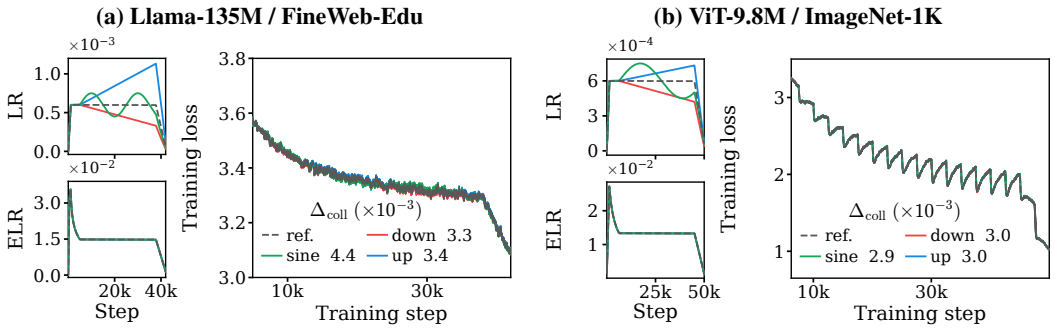


Figure B.4: **ELR collapse in Llama-135M and ViT.** Both experiments branch after LR warmup and a further period of shared training: at step 5,000 in **(a)** and 6,250 in **(b)**. Within each experiment, the branches use different LR and parameter-norm schedules but matched ELR schedules, yielding closely aligned loss trajectories. The loss-panel legends report the mean collapse errors relative to the reference run over the post-branch phase.

Experimental setup. For each test, we construct two runs, A and B, using different assignments of prescribed norm schedules to the controlled tensors. These schedules are selected randomly from four families: constant, linear-up, linear-down, and sinusoidal. The assignments are fixed before training, so different tensors can follow different norm trajectories within each run.

Heterogeneous norm schedules. We first vary the norm schedules across tensors while keeping a common WSD ELR target for all controlled tensors. Runs A and B use different assignments of norm schedules, but match the ELR of each corresponding tensor. Despite their different tensor-wise norm trajectories, the two runs have closely aligned losses, with $\Delta_{\text{coll}} = 0.91 \times 10^{-3}$ (Figure B.5(a)).

Heterogeneous ELR targets. We next allow ELR targets to vary across tensor types, in addition to using heterogeneous norm schedules. Attention tensors use cosine ELR schedules, whereas MLP tensors and the tied embedding/head use WSD schedules, with the peak values listed in Table B.2. Runs A and B again use different norm-schedule assignments, while matching the ELR target of each corresponding tensor. Their loss trajectories still closely align, with $\Delta_{\text{coll}} = 1.77 \times 10^{-3}$ (Figure B.5(b)).

Table B.2: **Prescribed tensor-wise ELR profile.** Each controlled tensor uses the peak and schedule assigned to its tensor type.

Tensor type	Count	Peak ELR	ELR schedule
Attention Q/K/V	36	0.05	cosine
Attention output	12	0.07	cosine
MLP up	12	0.05	WSD
MLP down	12	0.07	WSD
Tied embedding/head	1	0.05	WSD

Together, these experiments show that the observed collapse does not rely on all tensors sharing a common norm schedule or ELR target. They provide further evidence that matching ELR tensor by tensor preserves loss dynamics even under the heterogeneous schedules tested here.

B.4 Conditions for Precise ELR Collapse

We supplement Section 4.2 with normalization ablations in ViTs and a trajectory-level comparison of fast LR–norm variation.

Normalization also affects collapse precision in ViTs. We test whether the normalization dependence identified in Section 4.2 also appears in ViTs. Using the ViT configuration in Appendix A, we consider three nested variants: (i) QK-Norm with learnable gains, (ii) no QK-Norm with learnable gains, and (iii) no QK-Norm with all remaining RMSNorm gains fixed at initialization. For

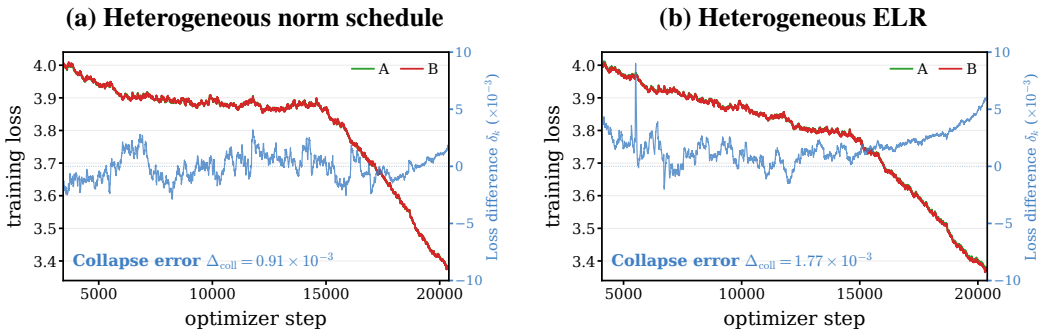


Figure B.5: **ELR collapse under tensor-wise heterogeneity.** Runs A and B use different tensor-wise norm schedules while matching the ELR of each corresponding tensor. **(a)** Norm schedules vary across tensors, while all controlled tensors share a common WSD ELR target. **(b)** ELR targets also vary in amplitude and schedule family across tensor types. Blue curves show the loss difference δ_k on the right axis, and annotations report the mean collapse error Δ_{coll}.

each variant, we repeat the prescribed-ELR protocol using the reference, linear-up, linear-down, and sinusoidal schedules.

Figure B.6 shows the same qualitative ordering observed in language models. Averaged over the three nonreference schedules, the mean collapse error increases from 3.0×10^{-3} in the default configuration to 3.9×10^{-3} after removing QK-Norm, and further to 6.5×10^{-3} after fixing the RMSNorm gains. This result shows that the normalization dependence of ELR collapse is not confined to our language model experiments, but also appears in the ViT/ImageNet setting studied here.

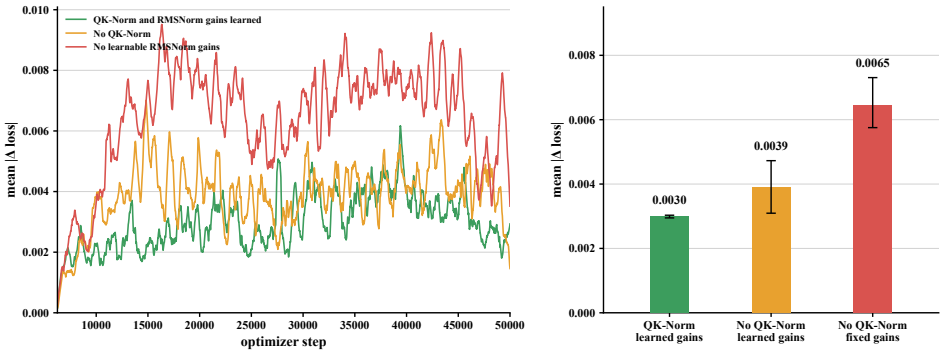


Figure B.6: **Normalization improves the precision of ELR collapse in ViTs. Left:** Mean absolute loss difference across the linear-up, linear-down, and sine-wave ELR-matched schedules relative to the reference run during training. **Right:** Corresponding trajectory-averaged collapse error. Removing QK-Norm weakens the collapse, while additionally fixing the remaining gains produces the largest discrepancy.

Fast LR–norm variation. Figure B.7 pairs the loss differences for the 2- and 32-cycle runs in Figure 3(b) with their LR schedules on aligned time axes. The lower panel shows the prescribed LR schedules.

C ELR Synchronization and Rank-One Matching (Supplement to Section 5)

We provide details of the SVD analysis and rank-one ELR-matching experiment reported in Figure 4.

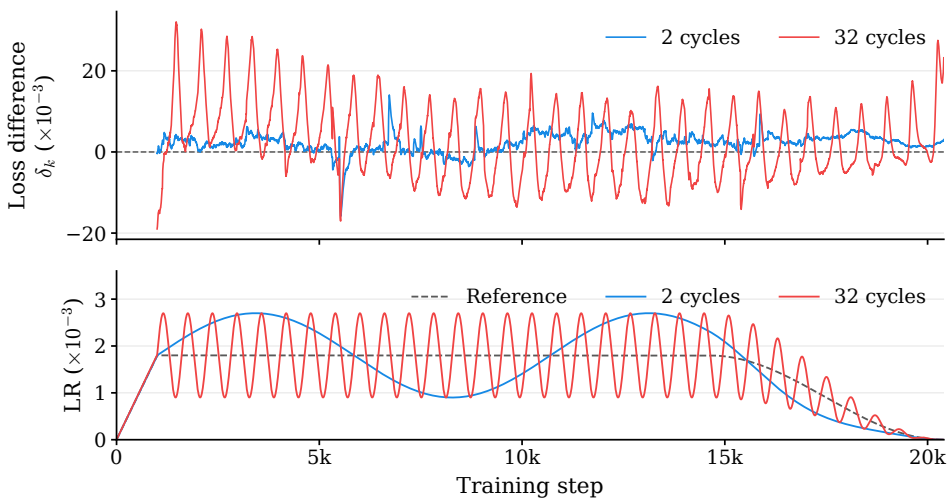


Figure B.7: **LR schedules and loss differences under sinusoidal modulation. Top:** signed loss differences for 2 and 32 modulation cycles. **Bottom:** reconstructed LR schedules, with the unmodulated reference shown as a dashed gray curve. Loss differences are shown after warmup.

The statistics in Figure 4(a) are computed from eight standard-training runs of Llama-124M on FineWeb, combining AdamW or MuonW, WSD or cosine LR schedules, and weight decay $\lambda \in \{0, 0.1\}$. The remaining training settings follow Appendix A. We compute ELR for all two-dimensional parameter tensors and construct the ELR matrix and perform SVD as described in the main text.

For the rank-one matching experiment in Figure 4(b), we use an AdamW run with a WSD LR schedule and zero weight decay as the reference. We extract the leading singular component of its ELR matrix to obtain the tensor-wise targets $\bar{\eta}^{\text{eff}}_{i,k}$. The comparison run shares the same warmup trajectory as the reference. Starting from the 1,001st update, we change its weight decay to 0.1 and set each two-dimensional parameter tensor's LR to $\eta^{\text{match}}_{i,k} = \bar{\eta}^{\text{eff}}_{i,k} \|\mathbf{W}^{\text{match}}_{i,k}\|_{\text{RMS}}$ so that its ELR matches the corresponding rank-one target. Parameter norms evolve freely during training without additional projection, while RMSNorm gains retain their original WSD LR schedule.

D LR-ONLY INTERVENTIONS ACROSS NORM-CONTROL METHODS (SUPPLEMENT TO SECTION 6)

D.1 CHOICE OF AGGREGATE ELR

Section 6 uses a parameter-count-weighted aggregate of tensor-wise ELRs for LR-only matching. Here, we compare this choice with an unweighted mean through matching experiments.

Let $\mathcal{I}$ be the set of controlled tensors, with $m = |\mathcal{I}|$. For each tensor $\mathbf{W}_i$, let $n_i = |\mathbf{W}_i|$. At training step k, write $\eta^{\text{eff}}_{i,k} = \eta_k / \|\mathbf{W}_{i,k}\|_{\text{RMS}}$ where η_k is the base LR.

We consider the following aggregation rules:

$$\begin{aligned} \text{Mean ELR:} \qquad & \gamma^A_k = \frac{1}{m} \sum_i \eta^{\text{eff}}_{i,k}, \\ \text{Parameter-count-weighted ELR:} \qquad & \gamma^G_k = \sum_i w_i \eta^{\text{eff}}_{i,k}. \end{aligned} \tag{D.1}$$

Here, $w_i = n_i / \sum_j n_j$. The two rules average tensor-wise ELRs with equal or parameter-count weights.

We use the same Llama-124M/FineWeb setup as the weight-decay experiment in Section 6, with an AdamW run using $\lambda = 0.1$ as the reference and an AdamW run using $\lambda = 0$ as the comparison.

For each aggregation rule, we adjust only the comparison run's base LR after warmup to match the corresponding reference aggregate ELR schedule. All other settings are held fixed between the two tests. The resulting mean collapse errors are 1.26×10^{-2} for mean ELR, and 7.64×10^{-3} for parameter-count-weighted ELR. The lower error supports the choice of parameter-count weighting in the main text. It also falls below the seed-to-seed variation measured in Appendix B.2.1.

D.2 Weight Decay and Hyperball

This section reports the loss trajectories underlying the intervention experiments in Section 6 and repeats each intervention with the matching direction reversed. All adapted LRs follow the reference-run ELR-matching protocol in Appendix A.2.

Weight decay with AdamW. Figure D.1(a) supplements Figure 5(a) with corresponding loss residuals. The run with $\lambda = 0.1$ serves as the target, while the LR of the run without weight decay is adapted to track its ELR schedule. Figure D.1(b) reverses the direction: the $\lambda = 0$ run serves as the target, and the LR of the $\lambda = 0.1$ run is adapted accordingly. The loss trajectories remain closely aligned in both directions.

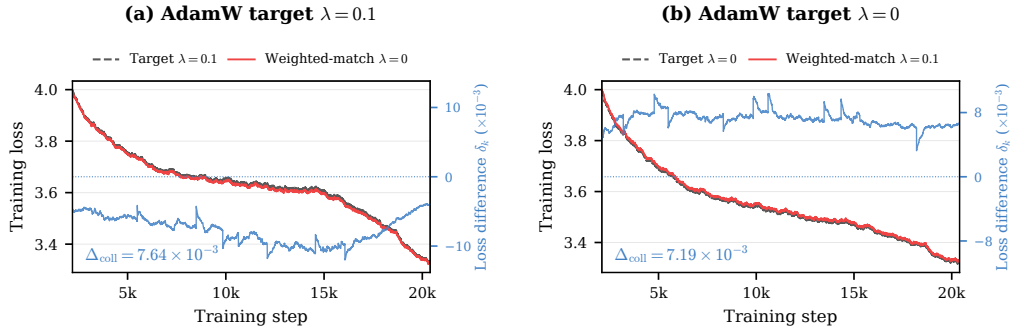


Figure D.1: **LR-only ELR matching recovers loss dynamics across AdamW weight-decay settings.** Experiments use Llama-124M trained on FineWeb with AdamW. In each subfigure, only the LR of the comparison run is adapted to match the target ELR, while its parameter norm evolves freely. The panel compares the losses and reports the mean collapse error. **(a)** The run with weight decay $\lambda = 0.1$ serves as the target, and the LR of the $\lambda = 0$ run is adapted. **(b)** The matching direction is reversed: the $\lambda = 0$ run serves as the target, and the LR of the $\lambda = 0.1$ run is adapted. The loss trajectories remain closely aligned in both directions.

Hyperball with Muon. Figure D.2(a) supplements Figure 5(b) with the corresponding loss residuals. MuonH, which uses Hyperball, serves as the target, while the LR of MuonW, which uses weight decay, is adapted to track its ELR schedule. Figure D.2(b) reverses the direction: MuonW serves as the target, and the LR of MuonH is adapted to track its ELR. The resulting loss trajectories remain closely aligned, with a mean collapse error of 2.5×10^{-3}. Thus, the agreement is not specific to the direction of ELR matching.

Learnable RMSNorm gains remain important under LR-only intervention. Figure D.3 repeats the AdamW LR-only intervention in Figure D.1, with the RMSNorm gains fixed throughout training as the only change. As in the prescribed-ELR experiments, fixing these gains substantially weakens the collapse and increases the collapse error to the 10^{-2} scale. Thus, the dependence on learnable gains is not specific to prescribed norm schedules; it persists when the parameter norm evolves freely and only the LR is adapted to match ELR. This complements the finding of Wang et al. (2026b) that removing learnable RMSNorm gains degrades LLM pretraining even after retuning the peak LR.

E Functional Scaling Laws (Supplement to Section 7)

We provide the FSL specification and per-trajectory results complementing Figure 6 in Section 7.

Trajectories and evaluation split. We train Llama-135M on FineWeb-Edu with batch size $B = 64$ and sequence length 1024 for 42,000 optimizer steps, including 1,000 warmup steps. ELR collapse

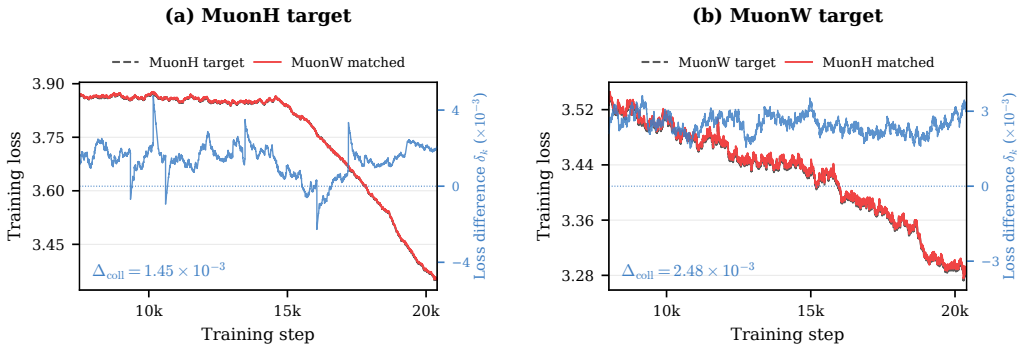


Figure D.2: **LR-only ELR matching aligns MuonW and MuonH loss dynamics.** Experiments use Llama-124M trained on FineWeb. MuonW denotes Muon with weight decay, whereas MuonH uses Hyperball. In each subfigure, only the LR of the comparison run is adapted to match the target ELR, while its norm-control method remains unchanged. The panel compares the resulting losses. **(a)** MuonH serves as the target, and the LR of MuonW is adapted. **(b)** The matching direction is reversed: MuonW serves as the target, and the LR of MuonH is adapted. For MuonH, we use the optimizer-specific ELR defined in Appendix A.

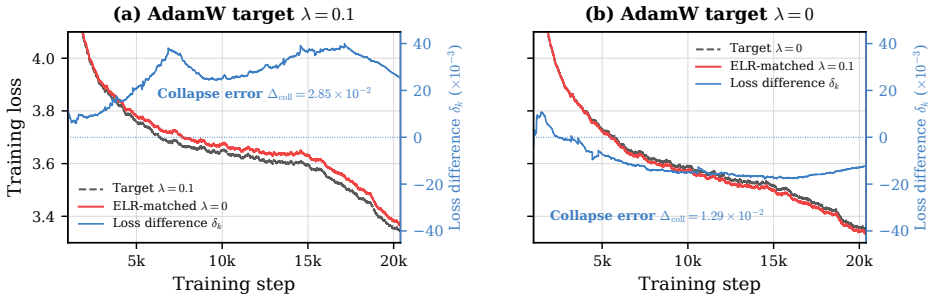


Figure D.3: **Fixing RMSNorm gains degrades LR-only ELR matching.** Dashed black curves show the target losses, red curves show the comparison runs whose LRs are adapted to match the target ELR schedules, and blue curves show the loss differences δ_k on the right axis. **(a)** The run with $\lambda = 0.1$ serves as the target, and the LR of the $\lambda = 0$ run is adapted. **(b)** The matching direction is reversed. In both panels, all RMSNorm gains are fixed and no norm trajectory is prescribed. Compared with the learnable-gain results in Figure D.1, fixing the gains produces sustained 10^{-2}-scale loss differences in both directions.

for this configuration is validated in Figure B.4(a). Standard training uses weight decay $\lambda \in \{0, 0.1\}$ and a peak learning rate of 6×10^{-4}, whereas Hyperball directly controls the update norm and uses a peak learning rate of 2×10^{-3}. We consider four LR schedules. WSD-10% and WSD-30% decay LR over the final 10% and 30% of training, respectively. The 7111 schedule divides the post-warmup steps into four consecutive phases containing 70%, 10%, 10%, and 10% of the steps, with LR reduced by a factor of $10^{1/3}$ at each phase boundary. The complete assignment of trajectories to the fitting, in-distribution (ID), and out-of-distribution (OOD) splits is reported in Table E.1.

ELR schedules. We use the aggregate ELR from Equation (5). For standard AdamW trajectories, this gives

$$\gamma_k^{\text{WD}} = \sum_i w_i \frac{\eta_{i,k}}{\|\mathbf{W}_{i,k}\|_{\text{RMS}}}. \tag{E.1}$$

For Hyperball, the instantaneous aggregate ELR is $\gamma_k^{\text{H}} = \eta_k^{\text{H}} \sum_i w_i / \|\mathbf{U}_{i,k}\|_{\text{RMS}}$, where η_k^{H} is the shared Hyperball LR and $\mathbf{U}_{i,k}$ is the unscaled Adam update. However, this quantity fluctuates with the update norms. For FSL prediction and schedule design, we replace the time-varying update norms with fixed reference scales. Specifically, we compute the mean RMS norm of the Adam

update $\bar{u}_i = \frac{1}{28000}\sum_{k=10001}^{38000} \|\mathbf{U}_{i,k}^{\text{reference}}\|_{\text{RMS}}$ from a reference AdamW run with $\lambda = 0.1$ and WSD-10%. This averaging window lies predominantly within the constant-LR phase, where the update norms are relatively stable. Then the calibrated aggregate ELR for every Hyperball run is

$$\widetilde{\gamma}_k^{\text{H}} := \sum_i w_i \frac{\eta_k^{\text{H}}}{\bar{u}_i} = \frac{\eta_k^{\text{H}}}{d_H}, \qquad d_H = \left(\sum_i \frac{w_i}{\bar{u}_i}\right)^{-1}, \tag{E.2}$$

where d_H is a fixed scale shared across Hyperball runs. This fixed calibration preserves the shape of the prescribed LR schedule without the additional step-to-step fluctuations from update norms; it does not assume that the instantaneous update norms are constant. We use the same reference scales for all Hyperball trajectories here and for schedule design in Appendix F.1, so the input ELR schedule does not require update statistics from a Hyperball run. Figure E.1 shows that these aggregate ELR schedules place the different norm-control regimes on a comparable scale.

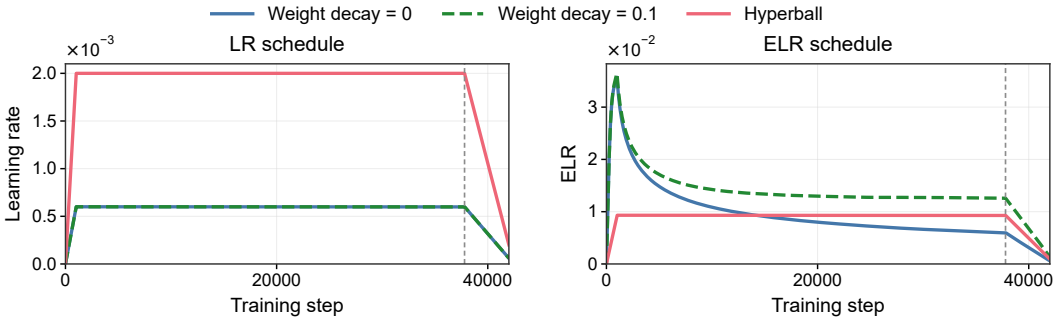


Figure E.1: **LR and ELR schedules under distinct norm-control regimes.** All curves use WSD-10%. Standard training uses weight decay with $\lambda \in \{0, 0.1\}$. Standard runs use the parameter-count-weighted aggregate ELR; Hyperball uses the calibrated aggregate ELR in Equation (E.2), with fixed reference update scales.

FSL parameterization and fitting. Given a nominal LR schedule $\boldsymbol{\eta} = (\eta_0, \ldots, \eta_N)$, we define the intrinsic time as $t_k = \sum_{j=0}^{k} \eta_j$. The lr-FSL model predicts the loss trajectory as

$$\widehat{L}_k = L_\infty + C_1(B_1 + t_k)^{-a_1} + C_2 \sum_{j=0}^{k} (B_2 + t_k - t_j)^{-a_2} \eta_j^2. \tag{E.3}$$

The elr-FSL model replaces η_j with the aggregate ELR input γ_j in both the intrinsic time and the noise term. Writing $t_k^{\text{eff}} = \sum_{j=0}^{k} \gamma_j$, its loss functional is

$$\mathcal{F}_k(\boldsymbol{\gamma}; \Theta) = L_\infty + C_1(B_1 + t_k^{\text{eff}})^{-a_1} + C_2 \sum_{j=0}^{k} (B_2 + t_k^{\text{eff}} - t_j^{\text{eff}})^{-a_2} \gamma_j^2. \tag{E.4}$$

For AdamW, we use γ_j^{WD} as input; for Hyperball, we use the calibrated schedule $\widetilde{\gamma}_j^{\text{H}}$ from Equation (E.2).

Following the FSL specification in the main text, we jointly fit the seven parameters

$$\Theta = (L_\infty, C_1, B_1, a_1, C_2, B_2, a_2)$$

by minimizing the mean squared error between the model predictions and the four smoothed loss trajectories. For both variants, we smooth the loss trajectories using an exponential moving average with decay 0.995, exclude the first 2,000 steps from the MSE, and use the same parameter initialization. We optimize Θ using Adam and L-BFGS.

Prediction Results. Table E.1 reports every trajectory, and Figure E.2 shows the corresponding predictions. elr-FSL has lower RMSE on nine of ten trajectories; under Hyperball, the improvements reach 6.86× for cosine and 20.34× for WSD-10%.

Table E.1: **Per-trajectory prediction error.** Improvement = $\mathrm{RMSE}_{lr}/\mathrm{RMSE}_{elr}$; values above one favor elr-FSL. Bold marks the lower error.

Split	Trajectory	lr-FSL ↓	elr-FSL ↓	Improvement ↑
Fit	wd = 0, WSD-30%	0.0258	**0.0135**	1.91×
	wd = 0, cosine	**0.0067**	0.0129	0.52×
	wd = 0.1, cosine	0.0177	**0.0119**	1.49×
	wd = 0.1, WSD-10%	0.0192	**0.0104**	1.84×
ID	wd = 0, WSD-10%	0.0347	**0.0132**	2.63×
	wd = 0, 7111	0.0233	**0.0134**	1.74×
	wd = 0.1, WSD-30%	0.0189	**0.0098**	1.92×
	wd = 0.1, 7111	0.0190	**0.0114**	1.67×
OOD	Hyperball, cosine	0.1213	**0.0177**	6.86×
	Hyperball, WSD-10%	0.3389	**0.0167**	20.34×

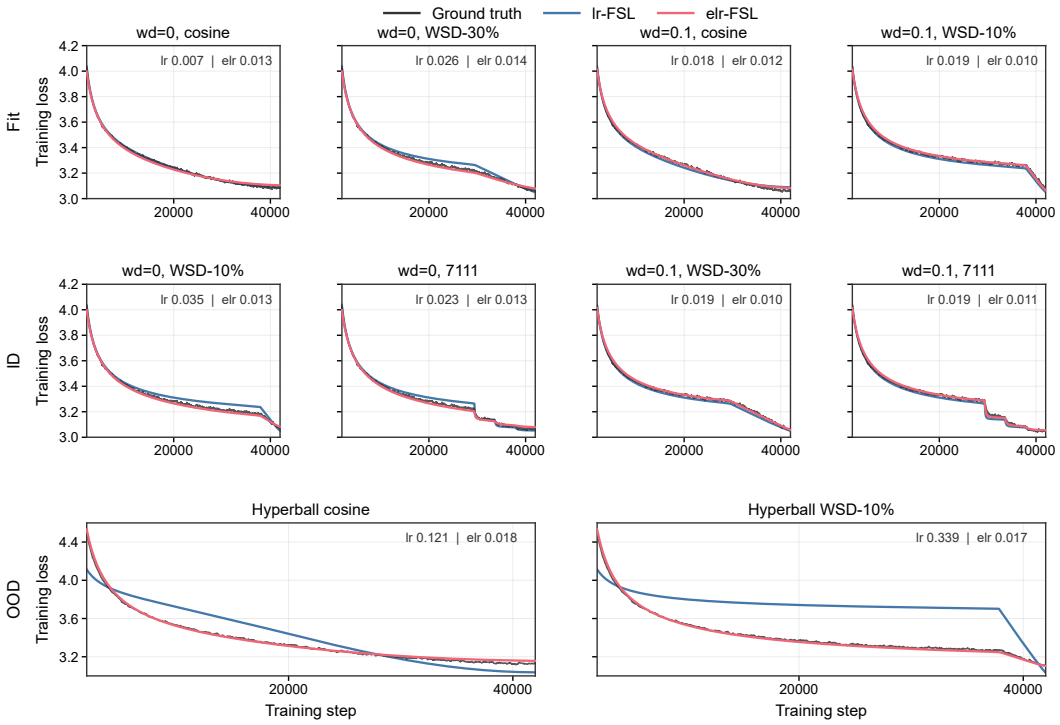


Figure E.2: **Predictions for all fitted and held-out trajectories.** Rows show fit, ID schedule transfer, and OOD Hyperball trajectories. Black denotes ground truth; blue and red denote lr-FSL and elr-FSL. We see that elr-FSL is substantially more accurate on both held-out ID and OOD schedules.

F Delayed Acceleration (Supplement to Section 8)

F.1 Optimal ELR Schedule

We describe how the fitted elr-FSL is used to design the Hyperball schedule in Figure 7. The objective is to minimize the final loss predicted by the fitted model, subject to a fixed training horizon, warmup, and peak ELR. The resulting schedule is then evaluated in an actual training run without refitting the model.

Fitting trajectories and model. We use the same Llama-135M training setup as Appendix E and fit four AdamW trajectories with weight decay $\lambda = 0.3$: cosine, WSD-10%, WSD-30%, and 7111. We choose these trajectories because they sustain relatively large aggregate ELRs, providing fitting

data in the higher-ELR regime relevant to schedule optimization. We minimize the mean squared error between the model predictions and the four EMA-smoothed training-loss trajectories (decay 0.995), excluding the first 2,000 steps from the MSE. We use Adam and L-BFGS to fit the seven parameters

$$\Theta = (L_\infty, C_1, B_1, a_1, C_2, B_2, a_2).$$

We denote the fitted parameter vector by $\widehat{\Theta}$.

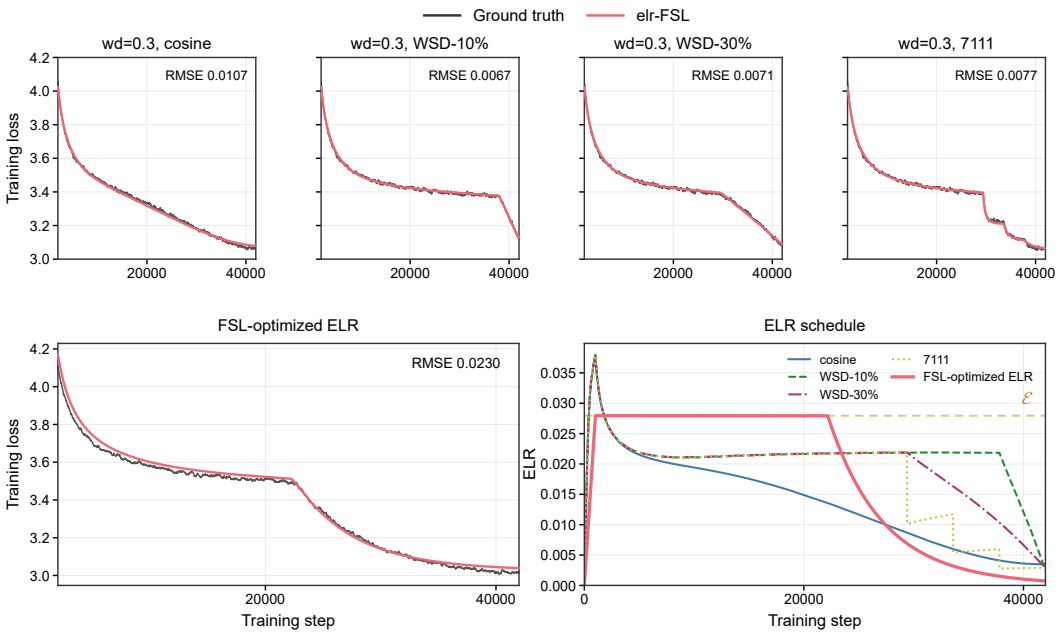


Figure F.1: **From fitted loss dynamics to an optimized ELR schedule. Top:** elr-FSL jointly fitted to the four AdamW trajectories with $\lambda = 0.3$. **Bottom left:** the predicted and observed loss of the Hyperball run using the derived schedule. **Bottom right:** the FSL-optimized ELR and the aggregate ELRs of the four fitting runs. The horizontal dashed line marks the chosen upper bound $\mathcal{E}$.

Variational schedule design. We fix the peak Hyperball LR $\eta^{\mathrm{H}}_{\max}$ and use the calibration in Equation (E.2) to set the ELR upper bound $\mathcal{E} = \eta^{\mathrm{H}}_{\max}/d_H$. The chosen bound, shown as a horizontal line in Figure F.1 (bottom right), exceeds the aggregate ELRs of the $\lambda = 0.3$ fitting runs during their stable training phase. The admissible set $\mathcal{A}$ in Equation (8) consists of schedules $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_N)$ over $N = 42{,}000$ training steps, with nonnegative ELRs bounded above by $\mathcal{E}$ and a linear warmup starting from zero during the first $W = 1{,}000$ steps. After warmup, the schedule is otherwise unrestricted within this bound. Following Li et al. (2026), we solve the continuous counterpart of this schedule-design problem. Let $\gamma(z)$ denote the ELR at continuous training step $z \in [0, N]$, and use $\mathcal{A}$ also for the corresponding class of continuous schedules. Define $t^{\mathrm{eff}}(z) = \int_0^z \gamma(s)\,\mathrm{d}s$. With L_∞, S, and K evaluated at the fitted parameters $\widehat{\Theta}$, the optimization problem is

$$\begin{aligned} \gamma^\star &\in \arg\min_{\gamma \in \mathcal{A}} \mathcal{F}_N[\gamma; \widehat{\Theta}], \\ \mathcal{F}_N[\gamma; \widehat{\Theta}] &= L_\infty + S\big(t^{\mathrm{eff}}(N)\big) + \int_0^N K\big(t^{\mathrm{eff}}(N) - t^{\mathrm{eff}}(z)\big)\gamma(z)^2\,\mathrm{d}z. \end{aligned} \tag{F.1}$$

For our fitted kernel, $a_2 > 2$ and the peak constraint is active at the optimum. The resulting schedule consists of linear warmup, a constant-ELR phase, and inverse-power decay:

$$\gamma^\star(z) = \begin{cases} \mathcal{E} z/W, & 0 \leqslant z < W, \\ \mathcal{E}, & W \leqslant z \leqslant z_s, \\ \mathcal{E}\left(1 + \dfrac{z - z_s}{h}\right)^{-\nu}, & z_s < z \leqslant N, \end{cases} \qquad \nu = \frac{a_2}{a_2 - 2}. \tag{F.2}$$

Here, the decay onset $z_s \in [W, N]$ and timescale $h > 0$ are determined by minimizing $\mathcal{F}_N[\gamma; \widehat{\Theta}]$, rather than by prescribing a decay ratio. The stable phase promotes signal learning, while the subsequent decay reduces noise accumulation near the end of training. We realize the resulting schedule with Hyperball by setting $\eta^{\mathrm{H}}(z) = d_H \gamma^\star(z)$, so that its calibrated aggregate ELR satisfies $\widetilde{\gamma}^{\mathrm{H}}(z) = \gamma^\star(z)$. Figure F.1 shows the fitted loss trajectories for the four AdamW runs with $\lambda = 0.3$ (top) and the predicted and observed loss of the Hyperball run using the resulting schedule (bottom left).

Final-loss comparison across Adam experiments. Table F.1 reports the final training and validation losses for the fifteen Adam-based runs used in the FSL and ELR schedule-design experiments: twelve AdamW runs combining $\lambda \in \{0, 0.1, 0.3\}$ with four LR schedules, two Hyperball baselines, and the Hyperball run using the optimal ELR schedule. Final training loss is the mean of the last 50 raw training losses. We evaluate all final checkpoints on the same held-out, in-distribution subset of FineWeb-Edu, containing 0.15% as many tokens as the full training budget. Hyperball with the optimal ELR schedule achieves the lowest loss on both metrics, improving over the best AdamW run by approximately 0.04 and the better Hyperball baseline by approximately 0.08 in both training and validation loss. These results support the use of elr-FSL to guide schedule design beyond describing existing training-loss trajectories.

Table F.1: **Final training and validation loss across all Adam-based runs with Llama-135M on FineWeb-Edu.** Training loss is averaged over the last 50 steps without smoothing. Validation loss is evaluated at the final checkpoint.

Optimizer	Schedule	Final training loss ↓	Final validation loss ↓
AdamW, $\lambda = 0$	Cosine	3.0760	3.1058
	WSD-10%	3.0585	3.0874
	WSD-30%	3.0517	3.0818
	7111	3.0571	3.0885
AdamW, $\lambda = 0.1$	Cosine	3.0531	3.0830
	WSD-10%	3.0650	3.0936
	WSD-30%	3.0427	3.0728
	7111	3.0402	3.0713
AdamW, $\lambda = 0.3$	Cosine	3.0555	3.0846
	WSD-10%	3.0998	3.1276
	WSD-30%	3.0655	3.0942
	7111	3.0514	3.0809
Adam with Hyperball	Cosine	3.1192	3.1489
	WSD-10%	3.0885	3.1174
	FSL-optimized ELR	**3.0039**	**3.0352**

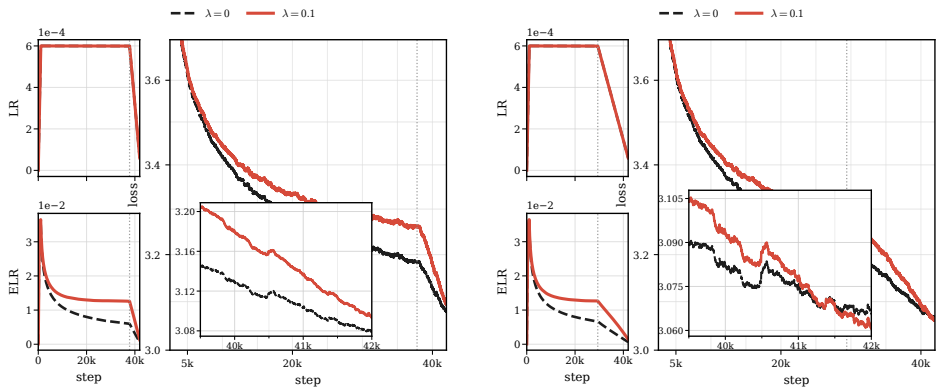
F.2 Acquired Early, Revealed Late: A Controlled WSD Test

The AdamW comparison under cosine LR schedules in the main text establishes delayed acceleration. However, under cosine decay, the learning rate decreases continuously, making it difficult to vary the duration of the late low-ELR phase independently. We therefore conduct a complementary experiment using WSD, whose terminal decay window can be controlled explicitly. Holding the peak LR and training budget fixed, we vary the WSD decay ratio from 0.1 to 0.3. This provides a targeted test of the temporal mechanism predicted by FSL.

FSL predicts that the benefit of weight decay is *acquired early but revealed late*. By suppressing norm growth, weight decay sustains a larger ELR during the early phase. The resulting faster accumulation of effective training time improves the signal-learning term, but the larger ELR simultaneously produces greater noise accumulation. Consequently, the signal-learning advantage need not be visible immediately in the observed loss. As ELR decays, new noise injection falls while previously accumulated noise continues to be forgotten, revealing the signal-learning advantage acquired earlier. The decisive prediction is therefore that a short decay phase may end before this advantage becomes visible, whereas a longer decay phase should reveal it and allow the weight-decayed run to overtake the unregularized baseline earlier.

Figure F.2 confirms this prediction. With a WSD decay ratio of 0.1, the $\lambda = 0.1$ run maintains a larger ELR than the $\lambda = 0$ run, but the terminal decay phase is too short to reveal its signal-learning advantage: it remains above the unregularized baseline at the end of training. When the decay ratio is increased to 0.3, the longer low-ELR phase changes the outcome. The $\lambda = 0.1$ run overtakes $\lambda = 0$ during decay and attains a lower final loss. In other words, the gain is accumulated before the crossover but becomes observable only during a sufficiently long decay phase.

The predicted crossover under a longer decay window supports the FSL account: an early signal-learning gain is initially masked by noise and revealed only late.



(a) WSD decay ratio 0.1 **(b)** WSD decay ratio 0.3

Figure F.2: **A longer WSD decay phase reveals the signal-learning gain acquired earlier.** Using Llama-135M trained on FineWeb-Edu, we explicitly vary the duration of the terminal WSD decay phase. Each panel compares $\lambda \in \{0, 0.1\}$ at a fixed WSD decay ratio; black and red denote $\lambda = 0$ and 0.1, respectively. With a decay ratio of 0.1 (left), the decay phase is too short for the $\lambda = 0.1$ run to overtake the unregularized baseline. Increasing the decay ratio to 0.3 (right) extends the low-ELR phase, allowing the accumulated noise to be forgotten and revealing the earlier signal-learning advantage: the $\lambda = 0.1$ run overtakes $\lambda = 0$ and attains a lower final loss.

F.3 Delayed Acceleration under Hyperball

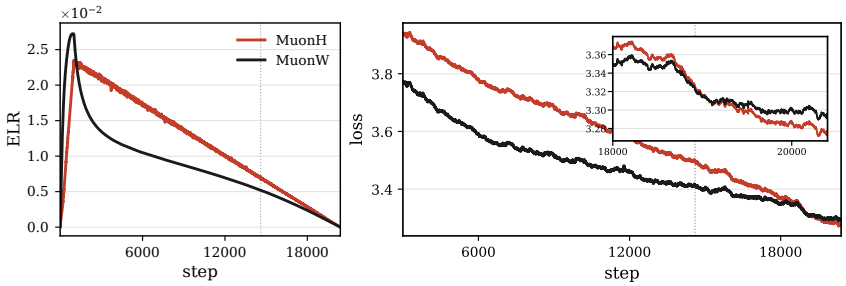


Figure F.3: **Delayed acceleration under Hyperball.** On Llama-124M trained on FineWeb, MuonH initially has higher loss than MuonW, but later overtakes it and reaches a lower late-stage loss. Both runs use linear schedules.

The same qualitative phenomenon also appears under Hyperball. A delayed gain is already visible in the MuonH trajectory reported by Wen et al. (2026a, Figure 3, right). In our experiment, Figure F.3 compares MuonH with MuonW when both use linear LR decay. MuonH has higher loss in early training, but subsequently catches up with and overtakes MuonW, reaching a lower loss near the end of training. Delayed acceleration is therefore not specific to weight decay; it also arises when the parameter norm is controlled explicitly by Hyperball.

The same ELR mechanism explains this reversal: Hyperball's norm control slows ELR decay relative to MuonW, maintaining a larger ELR before the terminal decay. This accelerates signal learning

but also increases noise accumulation, initially masking the advantage. During the terminal decay, the ELR decreases and the accumulated noise is gradually forgotten, revealing the advantage acquired earlier. This is consistent with Figure 5(b), where matching the aggregate ELR of MuonH brings the MuonW loss trajectory into close alignment, with a mean collapse error of 1.45×10^{-3} from step 7.5k onward.

G Additional Related Work: Weight Decay and ELR

Prior work has shown that weight decay can increase effective learning rates in batch-normalized networks (Zhang et al., 2019) and balance angular updates across layers and neurons (Kosson et al., 2024). Weight decay has also been linked to LR sensitivity in Transformer training (Wortsman et al., 2024).

Beyond these effects on optimization dynamics, the preferred weight-decay strength varies with the training budget, LR schedule, and evaluation objective. ELR collapse and elr-FSL offer complementary perspectives: the former relates loss dynamics to the induced ELR schedule, while the latter describes how this schedule balances signal learning and noise accumulation. We discuss these scaling, scheduling, and adaptation findings below, distinguishing pretraining loss from downstream adaptation.

Training horizon and batch size. Wang & Aitchison (2025) propose choosing AdamW's weight decay through a timescale normalized by the training horizon. Keeping this timescale fixed implies weaker decay for longer runs at fixed peak LR and batch size. Bergsma et al. (2025) refine this picture: the preferred timescale follows a power law in the tokens-per-parameter ratio rather than remaining constant. They also find that the preferred weight decay increases with batch size below the critical batch size, with this trend no longer holding beyond that regime.

These observations are consistent with the signal–noise trade-off in elr-FSL. Longer training allows signal learning to proceed with a smaller ELR, reducing the cost of noise accumulation. Weaker weight decay can realize such a reduction by allowing larger parameter norms. A related preference for smaller nominal LRs at longer horizons is reported by Bjorck et al. (2025). Conversely, at a fixed token budget, a larger batch size reduces both the number of updates and the gradient noise per update. This can favor larger effective steps, which stronger weight decay can produce by limiting norm growth. The connection between weight-decay scaling and ELR is also discussed by Bergsma et al. (2025). Our interpretation is qualitative, not a derivation of the reported scaling exponents: elr-FSL takes the realized ELR schedule as input and does not by itself determine parameter-norm dynamics.

Weight-decay scheduling during LR decay. A fixed weight-decay coefficient can cause parameter norms to shrink during LR decay, partly offsetting the intended decrease in ELR. Defazio (2025) analyzes the associated late-stage rise in gradient norms and introduces AdamC, which corrects the LR dependence of weight decay. Related corrections have also been studied for Scion (Chou, 2025) and Muon (Apte, 2026). The shared LR scaling can be written as

$$\lambda(k) = \lambda_0 \frac{\eta(k)}{\eta_{\max}}, \qquad \eta(k)\lambda(k) = \frac{\lambda_0}{\eta_{\max}} \eta(k)^2,$$

where $\eta_{\max}$ is the peak LR and λ_0 is the decay coefficient at that LR. Thus, the shrinkage applied in each update decreases quadratically rather than linearly with the LR.

These methods connect late-stage weight-decay reduction to norm dynamics, rather than treating it as an isolated scheduling choice. By keeping norms more stable as the LR falls, the correction can produce a more pronounced ELR decay. In elr-FSL, this reduces new noise injection near the end of training, while the accumulated noise is forgotten through the remaining intrinsic time. The benefit therefore depends on the full ELR trajectory: an overly aggressive reduction can also limit signal progress and leave too little intrinsic time to forget earlier noise. This gives a common interpretation of why coordinating LR and weight decay can improve final loss, without implying that a particular decay schedule is universally optimal.

Pretraining loss and downstream adaptation. The weight decay that minimizes pretraining loss need not maximize performance after fine-tuning. Han et al. (2026) show that stronger pretraining weight decay can improve downstream performance even when pretraining performance is worse. A

related trade-off appears in LR scheduling: Yano et al. (2026) find that keeping the LR constant after warmup can improve performance after supervised fine-tuning despite worse pretraining metrics.

Together, these findings motivate investigating whether sustaining a higher ELR during pretraining can improve downstream performance even at the cost of higher pretraining loss.