

LazyMuon: Exploiting Angular Stability to Reduce Newton-Schulz Overhead in Muon for Efficient Large Model Training

Kainan Lin^{2,*}, Rongwei Lu^{1,*}, Haotian Dong^{1,*}, Le Chen²

Jingyan Jiang¹, Huan Yang², Zhi Wang^{1,†}

¹Tsinghua Shenzhen International Graduate School, Tsinghua University

²South China Normal University

knlin@m.scnu.edu.cn wangzhi@sz.tsinghua.edu.cn

Abstract

Recently, the Muon optimizer has demonstrated strong efficiency in large model training through matrix orthogonalization. At each optimization step, Muon orthogonalizes the raw update matrix via Newton-Schulz (NS) iterations before applying it to model parameters. However, orthogonalization requires repeated matrix multiplications whose computational cost scales with $O(n^3)$ for an $n \times n$ matrix, making the overhead prohibitive at large model scales. In this paper, we observe that the geometric angle derived from cosine similarity between consecutive orthogonal update matrices remains highly stable during training, a phenomenon that we partially explain theoretically and support empirically. Based on this angular stability, we propose LazyMuon, which skips computationally expensive NS iterations through intermittent linear interpolation and employs a residual correction mechanism to mitigate accumulated approximation errors. This design ensures fast iteration while preserving Muon’s orthogonal manifold constraints. Empirical evaluations on language and vision tasks demonstrate that LazyMuon achieves speedups of up to $1.34\times$ and $1.31\times$ over Muon, respectively, with negligible degradation in convergence.

1 Introduction

For a long time, element-wise optimizers like AdamW (Loshchilov and Hutter, 2019) have been considered the standard baseline for training large models (Zhang et al., 2022; DeepSeek-AI et al., 2024). As a promising alternative, matrix-wise optimizers (Gupta et al., 2018; Vyas et al., 2025) have been developed to capture structural information. Building on this, the recently proposed Muon (Jordan et al., 2024) has challenged this dominance

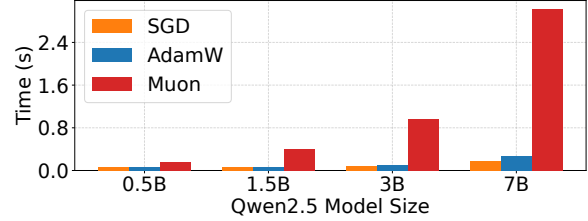


Figure 1: Training time per step across optimizers, including forward, backward, and optimizer update stages. Muon shows higher cost due to expensive optimizer updates dominated by NS iterations.

by delivering superior performance across diverse tasks. In particular, Muon has demonstrated substantial advantages and scalability in pre-training (Team et al., 2025; Zeng et al., 2025).

At each optimization step, Muon orthogonalizes the raw update matrix before applying it to model parameters. This orthogonalization is implemented using Newton-Schulz (NS) iterations, which are more efficient but less precise than Singular Value Decomposition (SVD) (Björck and Bowie, 1971; Cesista et al., 2025). However, the enormous weight matrices involved in large model training constrain the widespread adoption of Muon due to intensive matrix multiplication operations in the NS iterations. As illustrated in Figure 1, the training overhead per step of Muon increases sharply with model scale. Specifically, when scaling the Qwen2.5 model (Qwen et al., 2025) from 0.5B to 7B, Muon’s per-step training time overhead grows by $18.7\times$, consistent with observations reported in (Zhang et al., 2026) on NS implementations for Muon, highlighting the substantial computational burden introduced by the NS iterations on extremely large weight matrices.

Previous work has explored various avenues to address this efficiency bottleneck, providing valuable progress from different perspectives (Boissin et al., 2025; Luo et al., 2026; Huang et al., 2025; He et al., 2026). Nevertheless, attempts to directly

*Equal contribution.

†Corresponding author.

[‡]Kainan Lin conducted this work while interning at Tsinghua Shenzhen International Graduate School.

accelerate the orthogonalization step still entail sub-optimal trade-offs. Existing methods either break the constraints of the orthogonal manifold, leading to compromised convergence, or still require explicit orthogonalization at every step to strictly guarantee the manifold constraints, leaving the fundamental computational burden largely unresolved.

In this paper, we uncover a novel geometric property of the optimization trajectory to overcome this challenge: the angle derived from cosine similarity between consecutive orthogonal update matrices remains highly stable during the steady phase of training. We partially explain this phenomenon theoretically and support it empirically, revealing that it originates from the fundamental synergy between the norm constraint of orthogonal matrices and the inherent long-tail distribution of singular values in actual training gradients.

Based on this observation of angular stability, we propose LazyMuon, an optimizer that achieves high computational efficiency without sacrificing the geometric integrity of the updates. Instead of performing expensive NS iterations at every step, LazyMuon skips these iterations during intermediate steps within each period, relying instead on linear interpolation guided by the pre-measured angle of the orthogonalized state. To mitigate the approximation errors that inevitably accumulate during these skipped steps, LazyMuon incorporates a period-end residual correction mechanism, ensuring that the optimization trajectory remains in the orthogonalization manifold while substantially mitigating computational overhead. Empirical evaluations on language and vision tasks demonstrate that LazyMuon substantially reduces wall-clock training time without sacrificing convergence, achieving up to $1.34\times$ speedup over the standard Muon baseline.

In summary, our contributions include:

- We empirically observe that during stable loss descent, the angles between orthogonalized update matrices remain highly stable over time. We further analyze the underlying causes of this phenomenon through a combination of theoretical analysis and empirical validation.
- We propose LazyMuon, an optimizer that periodically skips the NS iterations by leveraging this stable angle property to approximate the orthogonalization results, and incorporates a residual correction at the end of each period to guarantee the manifold constraints.
- We compare LazyMuon with other baselines by evaluating their convergence and time efficiency on large model pretraining tasks. Results show that LazyMuon requires significantly less wall-clock time while maintaining competitive performance.

2 Preliminaries

2.1 Muon

Muon (Jordan et al., 2024) is a matrix-wise optimizer that orthogonalizes gradients to maintain consistent step lengths across different optimization directions. Formally, let $f(\mathbf{W}_{t-1}, \xi_t)$ denote the objective function evaluated on the weight matrix $\mathbf{W}_{t-1} \in \mathbb{R}^{m \times n}$ and a stochastic mini-batch ξ_t . The standard momentum $\mathbf{M}_t$ is updated with a coefficient $\beta \in (0, 1)$ as follows:

$$\mathbf{M}_t = \beta \mathbf{M}_{t-1} + \nabla f(\mathbf{W}_{t-1}, \xi_t) \quad (1)$$

Recent empirical studies suggest that directly utilizing $\mathbf{M}_t$ in the original Muon formulation can yield suboptimal performance in large scale settings (Sato et al., 2025). To address this, we adopt a robust variant that incorporates a Nesterov-style momentum term and an explicit magnitude alignment factor. The complete update rule of Muon employed in our work is formulated as:

$$\mathbf{N}_t = \beta \mathbf{M}_t + \nabla f(\mathbf{W}_{t-1}, \xi_t) \quad (2)$$

$$\mathbf{O}_t = \text{NewtonSchulz5}(\mathbf{N}_t) \quad (3)$$

$$\mathbf{W}_t = \mathbf{W}_{t-1} - \eta \cdot 0.2 \sqrt{\max(m, n)} \cdot \mathbf{O}_t \quad (4)$$

where $\mathbf{N}_t$ incorporates the Nesterov lookahead, and $\eta > 0$ is the learning rate. The scaling factor $0.2 \sqrt{\max(m, n)}$ is crucial for aligning the update magnitudes between Muon and AdamW.

The NewtonSchulz5 algorithm is employed to bypass the computationally expensive SVD. Through polynomial iterations, it approximates the orthogonal matrix $\mathbf{U}_t \mathbf{V}_t^\top$ derived from the exact SVD $\mathbf{N}_t = \mathbf{U}_t \mathbf{\Sigma}_t \mathbf{V}_t^\top$. As is standard practice (Jordan et al., 2024; Liu et al., 2025), vector-like parameters, including 1D normalization weights, the input embedding layer, and the language modeling head, are excluded from Muon and optimized conventionally via AdamW.

2.2 NS Orthogonalization and Directional Alignment

The core intuition behind Muon is to orthogonalize the update matrix, effectively normalizing its

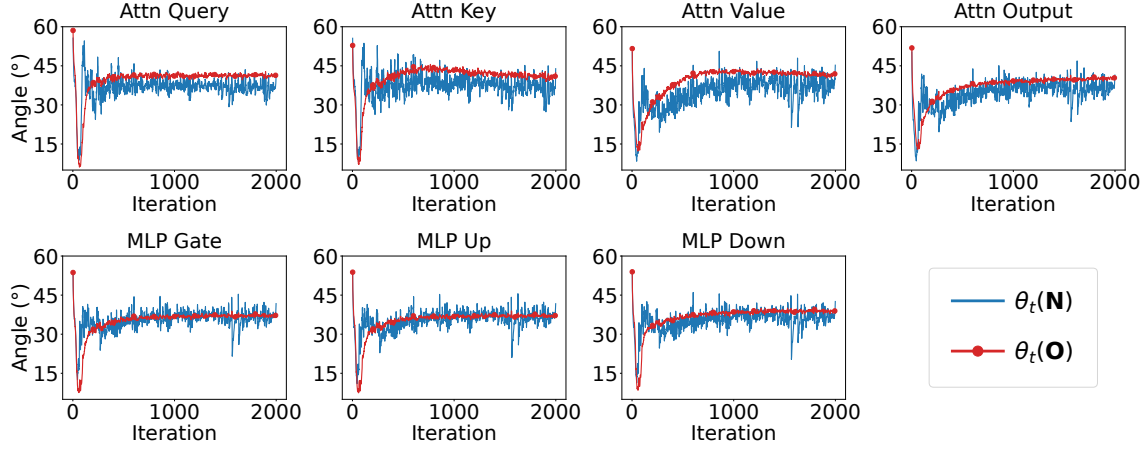


Figure 2: Evolution of angles between consecutive steps in the Nesterov momentum and orthogonalized update sequences for parameter matrices in Layer 11 of Qwen2.5-1.5B.

singular values to a uniform scale. This mechanism perfectly balances the update step sizes across different singular vector directions of the weight matrix, thereby optimizing the convergence trajectory. Mathematically, given a momentum matrix $\mathbf{N}_t = \mathbf{U}\Sigma\mathbf{V}^T$ (where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices), Muon seeks the strict orthogonal projection $\mathbf{Q}^* = \mathbf{U}\mathbf{V}^T$ as the ideal update direction.

However, due to the prohibitive computational overhead of exact SVD or polar decomposition, NS iterations are widely adopted as an efficient approximation. Although NS iterations yield an inexact orthogonalization, the original Muon work (Jordan et al., 2024) shows that Muon exhibits strong tolerance to such approximation: The inexact orthogonalization produced by NS iterations only leaves the singular values bounded within a relaxed range (e.g. $[0.7, 1.3]$), yet this approximation error does not degrade the convergence performance.

Building upon this intrinsic tolerance, recent work (Liu et al., 2026) argues that the conventional distance metric $\|\mathbf{Q}^T\mathbf{Q} - \mathbf{I}\|$ widely adopted in prior studies (Boissin et al., 2025; Sato et al., 2025) may not be a practically effective measure of orthogonalization quality. Under this strict metric, benign global scaling of singular values directly penalizes the orthogonal accuracy. Instead, they propose evaluating the update quality via cosine similarity, defined as $\frac{\langle \mathbf{Q}, \mathbf{Q}^* \rangle_F}{\|\mathbf{Q}\|_F \|\mathbf{Q}^*\|_F}$, which is inherently invariant to global scaling.

This scale-invariant perspective provides a crucial inspiration for our work: the superiority of an update matrix is not strictly determined by its rigid mathematical orthogonality, but rather dictated by its angular similarity to the ideal update direction. This profound geometric observation naturally mo-

tivates us to investigate the directional stability across consecutive steps, directly paving the theoretical groundwork for our angle-based LazyMuon algorithm in the subsequent sections.

3 Angular Stability Analysis.

Observation of Angular Stability. Although the existing literature on the Muon optimizer is predominantly theory-driven (Boissin et al., 2025; Khaled et al., 2025; Si et al., 2025), our proposed method is based on key empirical insights regarding the geometric effect of orthogonalization. To investigate this phenomenon, we train a Qwen2.5-1.5B model on the OpenWebText (Gokaslan et al., 2019). Following the standard Muon application protocol (Jordan et al., 2024; Liu et al., 2025), the optimizer is applied exclusively to parameter tensors with two or more dimensions, excluding the embedding layer and the language model head. In this setting, we systematically track the training dynamics of all seven parameter matrices using Muon within an intermediate Transformer block (Layer 11 of 24).

Specifically, we evaluate the geometric relationship of these matrices between consecutive optimization steps. Let $\theta_t(\mathbf{X})$ denote the angle between consecutive steps for a given matrix sequence $\mathbf{X}$, defined by the Frobenius inner product as $\theta_t(\mathbf{X}) = \arccos\left(\frac{\langle \mathbf{X}_{t-1}, \mathbf{X}_t \rangle_F}{\|\mathbf{X}_{t-1}\|_F \|\mathbf{X}_t\|_F}\right)$. As illustrated in Figure 2, a striking contrast emerges before and after the orthogonalization operation. While the angular trajectory of the pre-orthogonalized matrix $\theta_t(\mathbf{N})$ exhibits severe fluctuations, the orthogonalized update matrix $\theta_t(\mathbf{O})$ demonstrates a remarkably smooth and stable progression throughout the steady phase of loss descent.

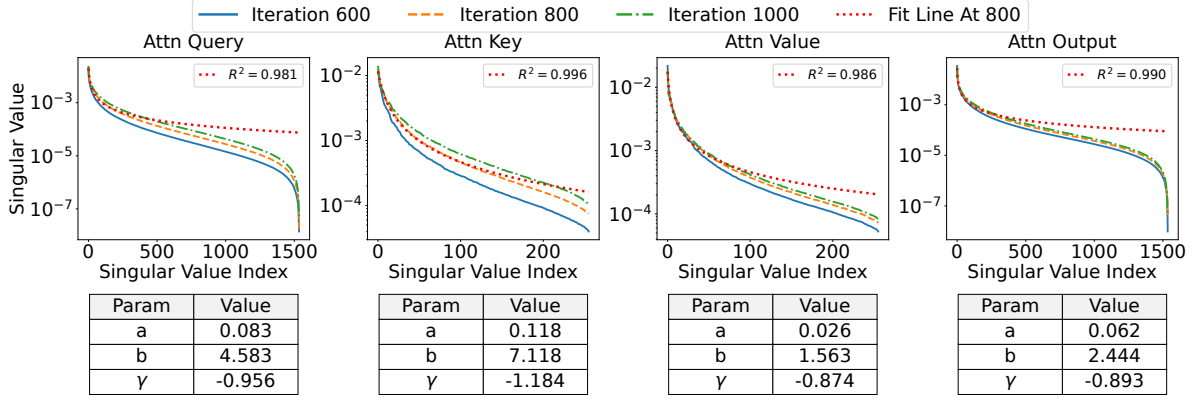


Figure 3: Evolution of singular value distributions for four attention matrices in Layer 11 of Qwen2.5-1.5B across iterations, with distribution fitting at iteration 800.

Variance Decomposition Analysis. To further explore the underlying mechanism of this phenomenon, we rewrite $\theta_t(\mathbf{X})$ by expanding the inner product formula:

$$\theta_t(\mathbf{X}) = \arccos \left(\underbrace{\frac{\|\mathbf{X}_t\|_F}{2\|\mathbf{X}_{t-1}\|_F}}_{Y_1(\mathbf{X})} + \underbrace{\frac{\|\mathbf{X}_{t-1}\|_F}{2\|\mathbf{X}_t\|_F}}_{Y_2(\mathbf{X})} - \underbrace{\frac{\|\mathbf{X}_t - \mathbf{X}_{t-1}\|_F^2}{2\|\mathbf{X}_{t-1}\|_F\|\mathbf{X}_t\|_F}}_{Y_3(\mathbf{X})} \right) \quad (5)$$

Our empirical analysis reveals that the overall stability of $\theta_t(\mathbf{O})$ is strictly driven by its decomposed components. Appendix 2 summarizes the variances of Y_1 , Y_2 , and Y_3 for both $\theta_t(\mathbf{N})$ and $\theta_t(\mathbf{O})$ across attention layers. The results clearly show that $\text{Var}(Y_i(\mathbf{O})) < \text{Var}(Y_i(\mathbf{N}))$ holds universally for all $i \in \{1, 2, 3\}$. The stabilization of $Y_1(\mathbf{O})$ and $Y_2(\mathbf{O})$ is theoretically expected; since the NS orthogonalization in a few-steps bounds $\|\mathbf{O}_t\|_F$ to a value near-constant, both terms are constrained near $\frac{1}{2}$, resulting in negligible variance.

Unlike the first two terms, rigorously proving $\text{Var}(Y_3(\mathbf{O})) < \text{Var}(Y_3(\mathbf{N}))$ mathematically is highly challenging. We thus turn to an empirical methodology by simulating Muon updates on artificial gradient sequences. We observe that for purely random sequences, $\text{Var}(Y_3(\mathbf{O}))$ does not exhibit a clear reduction compared with $\text{Var}(Y_3(\mathbf{N}))$. This strongly implies that the superior stability of $Y_3(\mathbf{O})$ is intrinsically coupled with the specific structural nature of actual training gradients. Therefore, prior to the analysis of Y_3 , we investigate the properties of the true gradient to guide the construction of an artificial sequence.

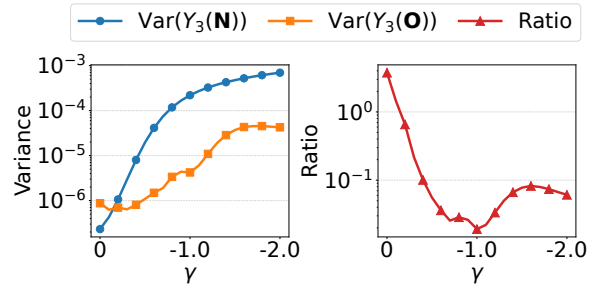


Figure 4: Variance and variance ratio of $Y_3(\mathbf{N})$ and $Y_3(\mathbf{O})$ under varying γ , where a larger absolute value of γ causes the variance of $Y_3(\mathbf{O})$ to grow slower than that of $Y_3(\mathbf{N})$ and their ratio to decrease.

Empirical Verification via Simulation. Given that the core objective of orthogonalization is to flatten the singular value distribution, it naturally motivates us to investigate the spectral properties of the raw gradients. Taking the four projection matrices within the Attention layer of Qwen2.5-1.5B as an example, we visualize their singular value distributions at iterations 600, 800, and 1000. As illustrated in Figure 3, these distributions exhibit a pronounced long-tail effect, consistent with prior findings (Sonthalia et al., 2026). Such a distribution can be accurately modeled by the function $y = a(x + b)^\gamma$. In the figure, we demonstrate this curve fitting specifically on the 800th iteration, yielding an $R^2 > 0.98$. Notably, since the fundamental long-tail characteristic remains unaffected by the scaling factor a and the shift factor b , the exponent γ intrinsically dictates the degree of imbalance within this distribution.

Building on this observation, we conduct a controlled simulation to examine how the spectral decay rate γ affects $\text{Var}(Y_3(\mathbf{O}))$, $\text{Var}(Y_3(\mathbf{N}))$, and their ratio. Taking the Query projection matrix in the

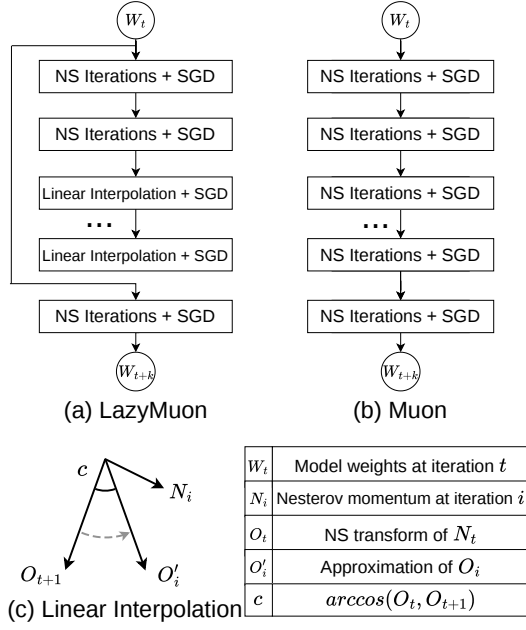


Figure 5: Overview of the optimizer frameworks and mechanisms. (a) The proposed LazyMuon framework incorporating linear interpolation and residual correction. (b) The baseline Muon optimizer framework. (c) Schematic representation of the linear interpolation mechanism, where $i \in \{t+3, \dots, t+k\}$.

Attention layer as a representative example, we fix the fitted scaling and shift parameters a and b , and vary only γ to construct singular value distributions of the form $y = a(x+b)^\gamma$. For simplicity, we assume that the singular value distribution remains approximately unchanged over short sequences of iterations. We then replace the original singular value distribution of the raw gradients with these synthesized distributions while retaining the corresponding singular vectors, thereby isolating the effect of the spectral decay rate.

As shown in Figure 4, as the absolute value of γ increases, the singular value distribution becomes increasingly long-tailed. Under this condition, $\text{Var}(Y_3(\mathbf{N}))$ surges rapidly and becomes highly unstable, while $\text{Var}(Y_3(\mathbf{O}))$ grows at a much slower rate. This difference in growth rates leads to a continuous decrease in the variance ratio $\frac{\text{Var}(Y_3(\mathbf{O}))}{\text{Var}(Y_3(\mathbf{N}))}$, demonstrating that $Y_3(\mathbf{O})$ is significantly more robust than $Y_3(\mathbf{N})$ against severe long-tail spectral imbalances. These results suggest that the long-tailed singular value distribution of raw gradients is the primary cause of the stark difference in stability between $Y_3(\mathbf{O})$ and $Y_3(\mathbf{N})$.

These findings demonstrate that the remarkable

Algorithm 1 LazyMuon

Require: 2D weights $\mathbf{W}_t \in \mathbb{R}^{n \times m}$, loss function $\mathcal{L}$, momentum μ , period $k \in \mathbb{Z}$, $k > 3$, learning rate η , weight decay λ

- 1: Initialize $\mathbf{O}_{\text{last}} \leftarrow \mathbf{0}$, $\Delta \leftarrow \mathbf{0}$, $\mathbf{M}_0 \leftarrow \mathbf{0}$
- 2: **for** $t = 1, \dots$ **do**
- 3: Compute gradient $\mathbf{G}_t \leftarrow \nabla_{\mathbf{W}} \mathcal{L}_t(\mathbf{W}_{t-1})$
- 4: $\mathbf{M}_t \leftarrow \mu \mathbf{M}_{t-1} + \mathbf{G}_t$
- 5: $\mathbf{N}_t \leftarrow \mu \mathbf{M}_t + \mathbf{G}_t$
- 6: **if** $t \bmod k \in \{1, 2\}$ **then**
- 7: $\mathbf{O}_t \leftarrow \text{NewtonSchulz5}(\mathbf{N}_t)$
- 8: $\mathbf{U}_t \leftarrow 0.2 \cdot \sqrt{\max(n, m)} \cdot \eta \mathbf{O}_t$
- 9: $c \leftarrow \text{cosine}(\mathbf{O}_t, \mathbf{O}_{\text{last}})$, $\mathbf{O}_{\text{last}} \leftarrow \mathbf{O}_t$
- 10: **else**
- 11: $\mathbf{O}'_t \leftarrow \text{LinearInterpolation}(\mathbf{N}_t, \mathbf{O}_{\text{last}}, c)$
- 12: $\mathbf{U}_t \leftarrow 0.2 \cdot \sqrt{\max(n, m)} \cdot \eta \mathbf{O}'_t$
- 13: **end if**
- 14: $\mathbf{W}_t \leftarrow (1 - \eta\lambda) \cdot \mathbf{W}_{t-1} - \mathbf{U}_t$
- 15: $\Delta \leftarrow \Delta + \mathbf{N}_t$
- 16: **if** $t \bmod k = 0$ **then**
- 17: $\mathbf{O} = \text{NewtonSchulz5}(\Delta)$
- 18: $\mathbf{U} \leftarrow \frac{\|\mathbf{W}_t - \mathbf{W}_{t-k}\|}{\|\mathbf{O}\|} \mathbf{O}$
- 19: $\mathbf{W}_t \leftarrow (1 - \eta\lambda) \cdot \mathbf{W}_{t-k} - \mathbf{U}$
- 20: $\Delta \leftarrow \mathbf{0}$
- 21: **end if**
- 22: **end for**
- 23: **return** $\mathbf{W}_t$

angular stability of orthogonalized update matrices originates from a fundamental synergy: the norm constraint of orthogonal matrices guarantees the stability of Y_1 and Y_2 , while the inherent long-tail distribution of singular values in gradient sequence enables the suppression of volatility in Y_3 .

4 LazyMuon

While the Muon optimizer utilizes NS iterations to avoid expensive matrix inversions, the repetitive matrix multiplications still impose a significant computational bottleneck in large model training. Inspired by our empirical findings on angular stability, we propose LazyMuon, which mitigates the frequent overhead of exact orthogonalization through a periodic residual correction strategy. The framework is illustrated in Figure 5 and its detailed procedure is given in algorithm 1 and algorithm 2.

Given a 2D weight matrix $\mathbf{W} \in \mathbb{R}^{n \times m}$, LazyMuon optimizes in a periodic manner with period length k . During the first two steps of each period, we apply full NS iterations on Nesterov momentum

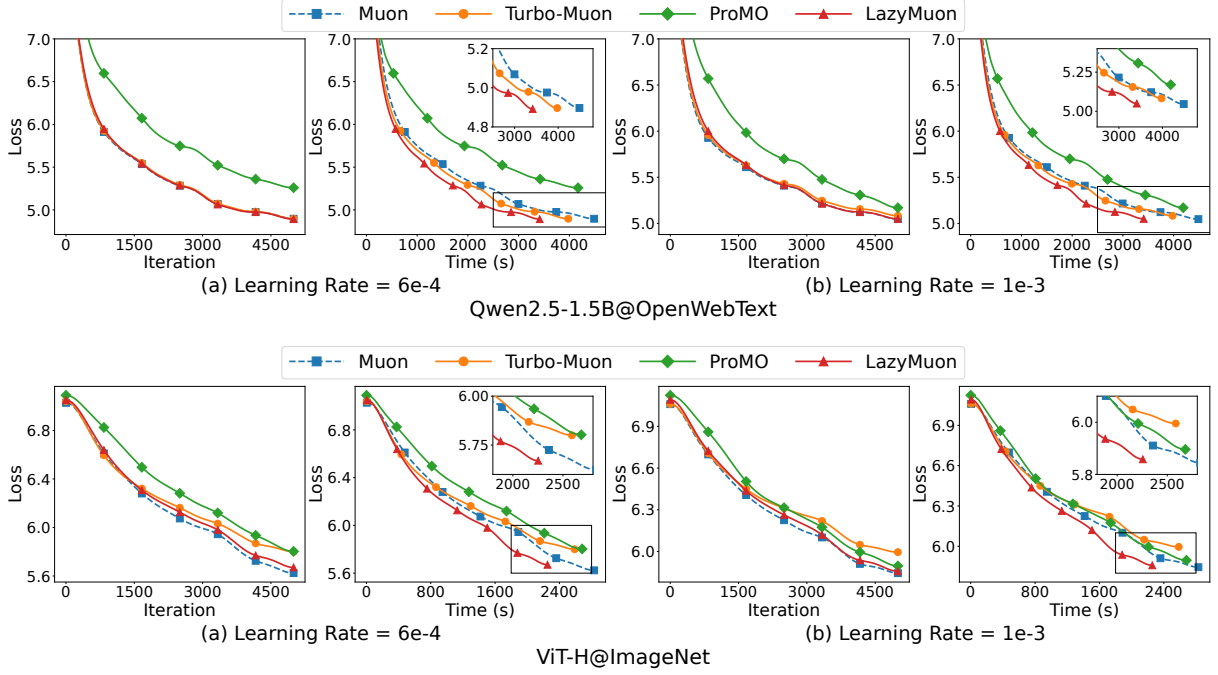


Figure 6: Training curves on Qwen2.5-1.5B@OpenWebText and ViT-H@ImageNet across learning rates (Left: Loss vs. Iteration; Right: Loss vs. Time). LazyMuon consistently achieves faster convergence than all baselines.

to obtain two orthogonal updates; the latter is kept as $\mathbf{O}_{\text{last}}$, and c is calculated as their cosine similarity. For the subsequent skipped steps within the period, we bypass high-frequency NS iterations and rely on linear interpolation for fast updates. At step t , guided by the estimated c , this interpolation effectively deflects $\mathbf{O}_{\text{last}}$ toward the current Nesterov momentum $\mathbf{N}_t$, thereby yielding $\mathbf{O}'_t$ as an approximation of the full orthogonal update $\mathbf{O}_t$. Following (Liu et al., 2025), each step incorporates decoupled weight decay λ and scale the learning rate by a dimension-dependent factor of $0.2\sqrt{\max(n, m)}$.

Although linear interpolation ensures high efficiency, approximation biases inevitably accumulate over the period. To rectify this, inspired by residual mechanisms (He et al., 2016; Zhang et al., 2019), we maintain an accumulator Δ to aggregate the unorthogonalized Nesterov momentum $\mathbf{N}_t$ across the period, and trigger a residual correction at the k -th step of the period. Specifically, we apply the NS operation to the accumulated residual Δ , yielding a manifold-constrained correction direction $\mathbf{O}$. We scale the orthogonalized direction $\mathbf{O}$ by $\frac{\|\mathbf{W}_t - \mathbf{W}_{t-k}\|}{\|\mathbf{O}\|}$ to preserve the exact magnitude of the accumulated update. This ensures that the residual correction only modifies the update direction to satisfy the orthogonal manifold constraints, leaving the overall step norm unchanged. The scaled update is applied to the weight-decayed anchor $\mathbf{W}_{t-k}$

to execute the final parameter update. This residual correction ensures the optimization trajectory remains in the orthogonalization manifold while substantially mitigating computational overhead.

5 Experiment

5.1 Experimental Environment

The experiments were conducted on a server with an Ubuntu 22.04.4 LTS system, an Intel Xeon Gold 6530 CPU, and four Nvidia L20 GPUs, each with 48GB of memory. The Python version is 3.12.6, and the other libraries used are all based on this Python version. We use PyTorch 2.4.1 with CUDA 12.1 as the machine learning toolkit, and NCCL as the communication backend for distributed training.

5.2 Experimental Settings

Tasks. To evaluate the effectiveness and versatility of our proposed method across different modalities, we conduct experiments on two representative and widely adopted setups: Qwen2.5 (Qwen et al., 2025) trained on the OpenWebText (Gokaslan et al., 2019) dataset (denoted as Qwen2.5@OpenWebText) and ViT (Dosovitskiy et al., 2021; Zhai et al., 2022) trained on ImageNet-1K (Deng et al., 2009) (denoted as ViT@ImageNet). All experiments are executed using Data Parallelism (DP) (Li et al., 2020) across four GPUs. For the

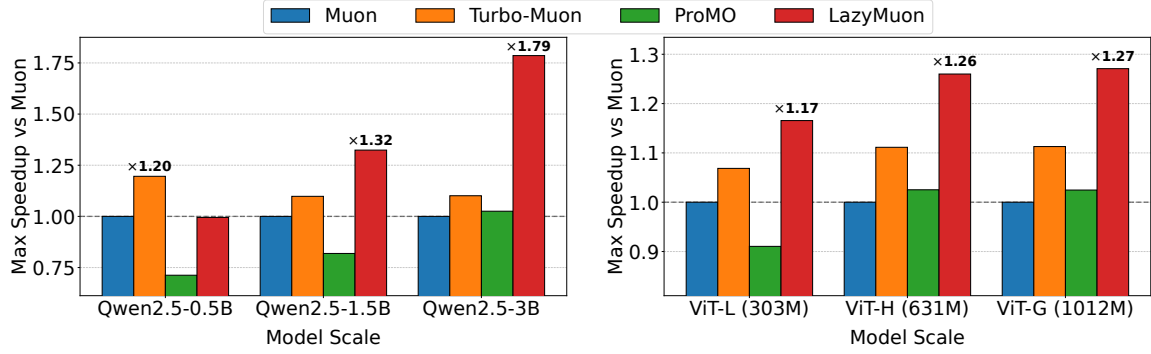


Figure 7: Maximum speedup of evaluated optimizers relative to Muon across varying model sizes. The experiments are conducted under a learning rate of 1×10^{-3} . Each bar indicates the highest acceleration ratio achieved by the algorithm when reaching the same target loss as Muon.

language modeling task Qwen2.5@OpenWebText, we set the sequence length to 512 tokens with a total batch size of 8, following the implementation configurations in the repository proposed by (Liu et al., 2025). Furthermore, considering that recent large models training heavily focuses on linguistic data, we intentionally incorporate the ViT@ImageNet benchmark to ensure a comprehensive evaluation in the vision domain. For this visual task, we adopt a total batch size of 16 and resize all input images to the standard 224×224 resolution. To rigorously verify the generalization capabilities of our method across different hyperparameter regimes—a criterion emphasized in (Si et al., 2025)—both tasks are independently evaluated under a smaller learning rate of 6×10^{-4} and larger learning rate of 1×10^{-3} .

Baselines. We compare our proposed LazyMuon against three representative optimizers. First, we adopt the foundational Muon optimizer from the official Kimi repository (Liu et al., 2025), which adapts the original Muon design (Jordan et al., 2024) for large language models (LLMs) by incorporating weight decay and per-parameter update scaling factors. Second, we evaluate ProMO (Luo et al., 2026), a dynamic hybrid optimizer that accelerates early training primarily with SGD and smoothly transitions to Muon in later stages via an adaptive, sine-based scheduling strategy tied to the training progress. Third, we benchmark against Turbo-Muon (Boissin et al., 2025), which mitigates the computational overhead of orthogonalization through Approximate Orthogonal (AOL) preconditioning; following their original configuration, we set the number of NS iterations to 4. Our LazyMuon employs period length $k = 10$ as the default setting for the main evaluations, with a detailed sensitivity analysis of k provided in subsequent sec-

tions. Across all baselines, we employ a Nesterov momentum of $\beta = 0.9$ for orthogonalized updates, whereas vector-like parameters such as embedding, normalization, and LM head layers are optimized via AdamW with $\beta_1 = 0.9$ and $\beta_2 = 0.95$.

5.3 Evaluation on Language and Vision Tasks

As shown in Figure 6, we evaluate LazyMuon on both Qwen2.5-1.5B and ViT-Huge under two learning rate settings. Across all configurations, LazyMuon consistently achieves superior wall-clock convergence efficiency compared with Muon, Turbo-Muon, and ProMO. Taking ViT@ImageNet with a learning rate of 6×10^{-4} as an example, when the loss reaches 6.0, LazyMuon obtains $1.15\times$, $1.25\times$, and $1.39\times$ speedups over Muon, Turbo-Muon, and ProMO, respectively, which further increase to $1.21\times$, $1.42\times$, and $1.49\times$ when the loss decreases to 5.8. The advantage of LazyMuon becomes increasingly pronounced at larger model scales, where existing methods either incur substantial overhead from per-step orthogonalization or damage convergence by intermittently switching to updates that violate the orthogonal manifold constraints. In contrast, LazyMuon reduces orthogonalization cost through angle-guided interpolation while preserving optimization stability via period-end residual correction, thereby achieving a better balance between convergence and training efficiency.

5.4 Scaling Behavior Across Model Sizes

In this section, we investigate how model scale affects the acceleration efficiency of LazyMuon. For Qwen@OpenWebText, we evaluate Qwen2.5-0.5B, Qwen2.5-1.5B, and Qwen2.5-3B, while for ViT@ImageNet we evaluate ViT-Large/16 (ViT-L), ViT-Huge/14 (ViT-H), and ViT-Giant/14 (ViT-G).

Table 1: Sensitivity analysis of the period length k . The table reports the wall-clock time required for LazyMuon with $k \in [4, 12]$ and the Muon baseline to reach specific target losses across two different tasks (LR denotes the learning rate). The relative speedup ratios of LazyMuon compared to Muon are provided in parentheses.

Setting	Qwen2.5-1.5B @ OpenWebText				ViT-H @ ImageNet			
	LR = 6×10^{-4}		LR = 1×10^{-3}		LR = 6×10^{-4}		LR = 1×10^{-3}	
	Loss = 5.5	Loss = 5.25	Loss = 5.5	Loss = 5.25	Loss = 6.0	Loss = 5.75	Loss = 6.0	Loss = 5.75
k=4	1441.10 (1.09 $\times$)	2197.57 (1.12 $\times$)	1515.55 (1.19 $\times$)	2451.76 (1.17 $\times$)	1121.64 (1.22 $\times$)	1715.42 (1.12 $\times$)	1376.21 (1.19 $\times$)	1909.26 (1.14 $\times$)
k=5	1332.31 (1.18 $\times$)	2027.71 (1.21 $\times$)	1405.79 (1.29 $\times$)	2271.90 (1.26 $\times$)	1084.97 (1.26 $\times$)	1533.85 (1.25$\times$)	1285.26 (1.28 $\times$)	1836.40 (1.19 $\times$)
k=6	1276.21 (1.23 $\times$)	1891.86 (1.30 $\times$)	1391.07 (1.30 $\times$)	2233.19 (1.28 $\times$)	1105.13 (1.23 $\times$)	1661.03 (1.15 $\times$)	1288.89 (1.27 $\times$)	1813.64 (1.20 $\times$)
k=7	1238.91 (1.27 $\times$)	1912.12 (1.29 $\times$)	1369.66 (1.32$\times$)	2161.15 (1.32 $\times$)	1142.67 (1.19 $\times$)	1606.57 (1.19 $\times$)	1326.34 (1.24 $\times$)	1793.54 (1.21 $\times$)
k=8	1204.67 (1.31 $\times$)	1863.21 (1.32$\times$)	1408.10 (1.29 $\times$)	2172.43 (1.32 $\times$)	1165.78 (1.17 $\times$)	1612.04 (1.19 $\times$)	1314.41 (1.25 $\times$)	1760.23 (1.24 $\times$)
k=9	1194.01 (1.32$\times$)	1886.86 (1.30 $\times$)	1405.14 (1.29 $\times$)	2138.18 (1.34$\times$)	1107.76 (1.23 $\times$)	1608.17 (1.19 $\times$)	1254.85 (1.31$\times$)	1760.65 (1.24 $\times$)
k=10	1201.80 (1.31 $\times$)	1881.70 (1.31 $\times$)	1591.10 (1.14 $\times$)	2388.44 (1.20 $\times$)	1066.41 (1.28$\times$)	1550.69 (1.24 $\times$)	1270.63 (1.29 $\times$)	1713.99 (1.27$\times$)
k=11	1226.25 (1.29 $\times$)	1913.80 (1.28 $\times$)	1600.87 (1.13 $\times$)	2298.49 (1.24 $\times$)	1073.20 (1.27 $\times$)	1538.22 (1.25$\times$)	1394.55 (1.18 $\times$)	1734.18 (1.26 $\times$)
k=12	1226.41 (1.29 $\times$)	1878.66 (1.31 $\times$)	1800.28 (1.01 $\times$)	2537.27 (1.13 $\times$)	1144.53 (1.19 $\times$)	1603.95 (1.19 $\times$)	1392.61 (1.18 $\times$)	1785.68 (1.22 $\times$)
Muon	1576.02	2459.03	1810.50	2858.58	1364.36	1915.52	1639.18	2176.59

We compare LazyMuon with all baselines across these model scales. The results under a learning rate of 1×10^{-3} are summarized in Figure 7, and those for 6×10^{-4} in Appendix 9. As model size increases, the speed advantage of LazyMuon becomes increasingly significant. On Qwen2.5-0.5B, the orthogonalization overhead remains relatively small, resulting in only limited improvement over Muon. However, when scaling to Qwen2.5-3B, LazyMuon achieves up to 1.79 $\times$ speedup over Muon. A similar trend is observed in the vision experiments, where the speedup increases from 1.17 $\times$ on ViT-L to 1.27 $\times$ on ViT-G. These results demonstrate that LazyMuon benefits more from larger models, since it replaces computationally expensive NS iterations with lightweight element-wise operations whose overhead grows much more slowly with model scale. Related loss curves are in Appendix 8.

5.5 Sensitivity Analysis of Hyperparameter

The period k in LazyMuon plays an important role in determining its training efficiency. A larger k enables faster execution by skipping more NS iterations, but may lead to more aggressive updates, potentially hurting convergence. In contrast, a smaller k makes LazyMuon behave more similarly to the baseline Muon optimizer, resulting in a slower training process. Nevertheless, we find that LazyMuon is remarkably robust to the choice of k within a reasonable range. We study the effect of k on the end-to-end training time and the results are reported in Table 1, while details are provided in Appendix 10. We sweep k from 4 to 12 and observe that LazyMuon consistently outperforms Muon across all settings. In particular, when k lies in the range of [7, 11], LazyMuon achieves the best overall performance, reaching up to a 1.34 $\times$

speedup. Based on these results, we consider [7, 11] to be a safe and effective range for choosing k in LazyMuon.

5.6 Ablation Study

We ablate the two core components of LazyMuon: interpolation-based deflection approximation and residual correction. We construct two variants: LazyMuon w/o Deflection Approximation (w/o-DA), which removes linear interpolation and directly applies raw momentum updates during skipped steps; and LazyMuon w/o Residual Correction (w/o-RC), which disables period-end correction while retaining interpolation. As shown in Appendix 11, removing either component consistently degrades performance in all tasks.

6 Conclusion

In this paper, we address the computational bottleneck of the Muon optimizer when training large models. We discover that the geometric angle between consecutive orthogonal update matrices remains highly stable during the training process. Leveraging this inherent property, we propose LazyMuon, an efficient matrix-level optimizer that skips computationally expensive NS iterations through intermittent linear interpolation. To strictly preserve the orthogonal manifold constraints, we incorporate a period-end residual correction mechanism. Extensive experiments on both language and vision tasks demonstrate that LazyMuon significantly accelerates training process and reduces computational overhead while maintaining competitive convergence performance. Furthermore, our approach exhibits superior scalability, making it a practical solution for massive deep learning models.

Limitations

This work focuses on accelerating large language model training via the LazyMuon optimizer, but several limitations should be noted. First, due to constraints in available computational resources, empirical validation on larger-scale models is currently lacking. Second, a rigorous theoretical analysis of convergence is not provided, as it diverges from the scope of our discussion.

Acknowledgement

This work is supported in part by National Natural Science Foundation of China(Grant No. 62472249 and 92467204), and Shenzhen Science and Technology Program (Grant No. KJZD20240903102300001). The work of Jingyan Jiang is supported by the Shenzhen Science and Technology Program (Grant No. JCYJ20250604145014018) and Natural Science Foundation of Top Talent of SZTU(Grant No. GDRC202413).

References

- Åke Björck and Clazett Bowie. 1971. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364.
- Thibaut Boissin, Thomas Massena, Franck Mamalet, and Mathieu Serrurier. 2025. Turbo-muon: Accelerating orthogonality-based optimization with preconditioning. *arXiv preprint arXiv:2512.04632*.
- Franz Louis Cesista, You Jiacheng, and Keller Jordan. 2025. *Squeezing 1-2% efficiency gains out of Muon by optimizing the Newton-Schulz Coefficients*.
- DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiusi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, and 69 others. 2024. *Deepseek llm: Scaling open-source language models with longtermism*. *Preprint*, arXiv:2401.02954.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. *Imagenet: A large-scale hierarchical image database*. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.
- 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. 2021. *An image is worth 16x16 words: Transformers for image recognition at scale*. *Preprint*, arXiv:2010.11929.
- Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. 2019. *Openwebtext corpus*.
- Vineet Gupta, Tomer Koren, and Yoram Singer. 2018. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pages 1842–1850. PMLR.
- Chuan He, Zhanwang Deng, and Zhaosong Lu. 2026. *Low-rank orthogonalization for large-scale matrix optimization with applications to foundation model training*. *Preprint*, arXiv:2509.11983.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Feihu Huang, Yuning Luo, and Songcan Chen. 2025. Limuon: Light and fast muon optimizer for large models. *arXiv preprint arXiv:2509.14562*.
- Keller Jordan and 1 others. 2024. Muon: An optimizer for hidden layers in neural networks. <https://kellerjordan.github.io/posts/muon/>.
- Ahmed Khaled, Kaan Ozkara, Tao Yu, Mingyi Hong, and Youngsuk Park. 2025. Muonbp: Faster muon via block-periodic orthogonalization. *arXiv preprint arXiv:2510.16981*.
- Jiaxiang Li and Mingyi Hong. 2025. *A note on the convergence of muon*. *Preprint*, arXiv:2502.02900.
- Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. 2020. *Pytorch distributed: Experiences on accelerating data parallel training*. *Proc. VLDB Endow.*, 13(12):3005–3018.
- Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, and 1 others. 2025. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*.
- Ziyue Liu, Ruijie Zhang, Zhengyang Wang, Yequan Zhao, Yupeng Su, Zi Yang, and Zheng Zhang. 2026. Muon²: Boosting muon via adaptive second-moment preconditioning. *arXiv preprint arXiv:2604.09967*.
- Ilya Loshchilov and Frank Hutter. 2019. *Decoupled weight decay regularization*. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.
- Jiaan Luo, Ziyi Tang, Feng Hong, Qiang Hu, Shuai Xiao, and Jiangchao Yao. 2026. *Long-tailed learning with muon optimizer*.
- James Martens and Roger Grosse. 2015. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR.

- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, and 25 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Naoki Sato, Hiroki Naganuma, and Hideaki Iiduka. 2025. Convergence bound and critical batch size of muon optimizer. *arXiv preprint arXiv:2507.01598*.
- Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International conference on machine learning*, pages 4596–4604. PMLR.
- Chongjie Si, Debing Zhang, and Wei Shen. 2025. Adamuon: Adaptive muon optimizer. *arXiv preprint arXiv:2507.11005*.
- Rishi Sonthalia, Michael Murray, and Guido Montúfar. 2026. Low rank gradients and where to find them. *Advances in Neural Information Processing Systems*, 38:152341–152385.
- Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, and 1 others. 2025. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*.
- Nikhil Vyas, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. 2025. [Soap: Improving and stabilizing shampoo using adam for language modeling](#). In *International Conference on Learning Representations*, volume 2025, pages 93423–93444.
- Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.
- Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2022. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113.
- Jack Zhang, Noah Amsel, Berlin Chen, and Tri Dao. 2026. [Gram newton-schulz](#).
- Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. 2019. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32.
- Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. [Opt: Open pre-trained transformer language models](#). *Preprint*, arXiv:2205.01068.

Appendix

A Related Works

A.1 Matrix-wise Optimization

Traditional adaptive optimizers such as AdamW (Loshchilov and Hutter, 2019) perform parameter updates in an element-wise manner, ignoring structural correlations within weight matrices. Matrix-wise optimization addresses this by explicitly exploiting these structures to enhance training efficiency and stability. Early explorations primarily focused on matrix factorization techniques: Shampoo (Gupta et al., 2018) and K-FAC (Martens and Grosse, 2015) utilize Kronecker-factored approximations for scalable second-order preconditioning, while AdaFactor (Shazeer and Stern, 2018) factorizes second-moment statistics into row and column components to reduce memory overhead. Building on these foundations, recent advancements propose more sophisticated matrix-structured updates. For instance, SOAP (Vyas et al., 2025) integrates adaptive preconditioning with matrix-wise statistics. Meanwhile, Muon (Jordan et al., 2024) employs Newton-Schulz iterations to orthogonalize update matrices, effectively extracting matrix-level directional features. Supported by solid theoretical guarantees (Li and Hong, 2025) and successful deployments in large model pre-training (Team et al., 2025; Zeng et al., 2025), Muon has demonstrated exceptional scaling behavior.

A.2 Muon and its variants

Building on this foundation, further studies have sought to extend its optimization capabilities. For instance, AdaMuon (Si et al., 2025) and Muon² (Liu et al., 2026) introduce adaptive mechanisms and refined update rules to enhance convergence robustness. Despite these empirical and algorithmic successes, Muon’s orthogonalization relies on multiple matrix multiplications per step, introducing substantial computational overhead that severely limits its practical training throughput on massive models. To mitigate this efficiency bottleneck, recent works have focused on accelerating the training process. For instance, Turbo-Muon (Boissin et al., 2025) aims to optimize the computational cost of the orthogonalization steps directly; however, as a per-step approach, it still incurs heavy computational overhead at scale. On the other hand, ProMO (Luo et al., 2026) attempts to balance efficiency and structural optimization by introducing a progres-

Table 2: Variances of Y_1, Y_2, Y_3 for the updates before (**N**) and after (**O**) orthogonalization and their ratios across Attention parameter matrices. $\text{Var}(Y_i(\mathbf{O})) < \text{Var}(Y_i(\mathbf{N}))$ for all $i \in \{1, 2, 3\}$.

Term	Attn Query			Attn Key			Attn Value			Attn Output		
	X = N	X = O	Ratio	X = N	X = O	Ratio	X = N	X = O	Ratio	X = N	X = O	Ratio
$\text{Var}(Y_1(\mathbf{X}))$	2.3e-3	9.1e-5	4%	4.3e-3	4.5e-5	1%	6.2e-3	7.2e-5	1%	4.0e-3	1.7e-4	4%
$\text{Var}(Y_2(\mathbf{X}))$	1.8e-3	9.4e-5	5%	3.1e-3	4.5e-5	1%	3.6e-3	7.2e-5	2%	2.5e-3	1.9e-4	7%
$\text{Var}(Y_3(\mathbf{X}))$	4.0e-3	2.0e-4	5%	7.8e-3	4.9e-4	6%	1.0e-2	9.7e-4	9%	7.0e-3	5.4e-4	8%

sive hybrid framework that dynamically alternates between standard momentum and Muon updates via a sinusoidal schedule. Nevertheless, this hybrid strategy damages convergence by intermittently switching to updates that violate orthogonal constraints.

B Extended Empirical Evaluations

This section provides extended empirical results across model architectures and scales. Table 2 presents the detailed variances and ratios of three terms of $\theta_t(\mathbf{X})$ for the updates before ($\mathbf{X} = \mathbf{N}$) and after ($\mathbf{X} = \mathbf{O}$) orthogonalization across the evaluated attention parameter matrices within an intermediate Transformer block. Meanwhile, Figure 8 shows the complete training curves under various learning rates across scaled model sizes to demonstrate scalability, while Figure 9 supplements the wall-clock speedup results specifically under an alternative learning rate of 6×10^{-4} . Furthermore, expanding upon the specific metric values reported in Table 1, Figure 10 provides the complete loss curves across all evaluated period lengths k for hyperparameter sensitivity analysis, while Figure 11 illustrates the complete loss curves for our ablation studies, validating the robustness and individual necessity of each component in LazyMuon.

C Extended Observation of Angular Stability

To further justify the foundation of our LazyMuon framework, we extend the empirical observation of angular stability from adjacent steps to larger step intervals. In the main text, our observation is focused on the temporal angle between adjacent updates, denoted as $\theta_t(\mathbf{O}) = \arccos\left(\frac{\langle \mathbf{O}_{t-1}, \mathbf{O}_t \rangle_F}{\|\mathbf{O}_{t-1}\|_F \|\mathbf{O}_t\|_F}\right)$, which corresponds to a step interval of $j = 1$. While this confirms that the orthogonalized update matrix at step $t - 1$ provides a highly reliable insight for predicting step t , the LazyMuon algorithm practically utilizes the orthogonalized update from a single step to approximate and correct subsequent

Algorithm 2 LinearInterpolation

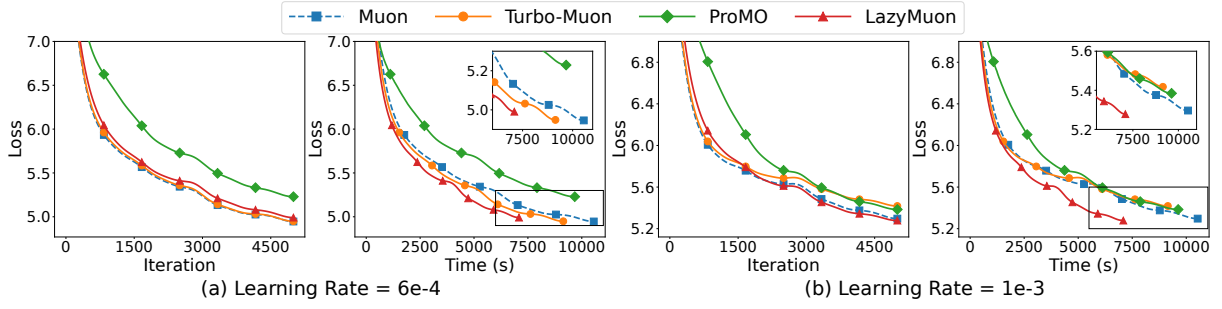
Require: Current Nesterov momentum buffer $\mathbf{N}_t$, cached direction $\mathbf{O}_{\text{last}}$, cached cosine c , and numerical constant ε

- 1: $\theta_1 \leftarrow \arccos\left(\frac{\langle \mathbf{N}_t, \mathbf{O}_{\text{last}} \rangle_F}{\|\mathbf{N}_t\|_F \|\mathbf{O}_{\text{last}}\|_F}\right)$
- 2: $\theta_2 \leftarrow \arccos(c)$
- 3: $w_t \leftarrow \frac{\|\mathbf{O}_{\text{last}}\|_F \sin \theta_2}{\|\mathbf{O}_{\text{last}}\|_F \sin \theta_2 + \|\mathbf{N}_t\|_F \sin(\theta_1 - \theta_2)}$
- 4: $\tilde{\mathbf{O}}_t \leftarrow w_t \mathbf{N}_t + (1 - w_t) \mathbf{O}_{\text{last}}$
- 5: $\mathbf{O}'_t \leftarrow \frac{\|\mathbf{O}_{\text{last}}\|_F}{\|\tilde{\mathbf{O}}_t\|_F + \varepsilon} \tilde{\mathbf{O}}_t$
- 6: **return** $\mathbf{O}'_t$

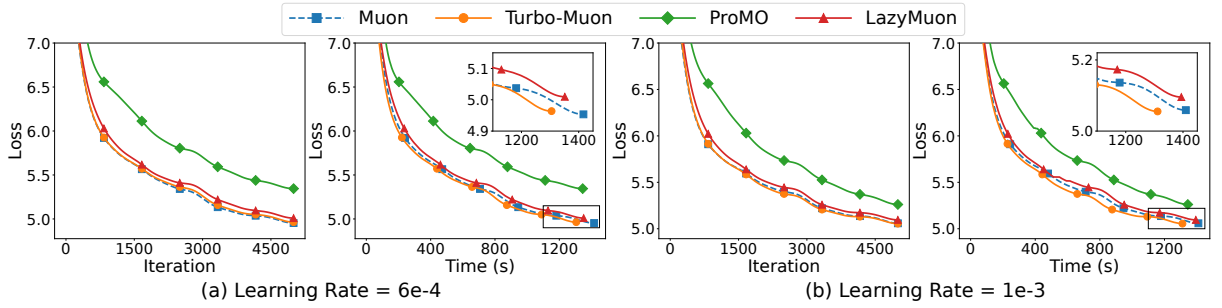
updates across an entire period (i.e., predicting steps $t + 1, t + 2, \dots, t + j$). To bridge this gap, we introduce a 3D visualization in Appendix 12 that systematically tracks the generalized interval angle, defined as $\theta_{t,j}(\mathbf{O}) = \arccos\left(\frac{\langle \mathbf{O}_{t-j}, \mathbf{O}_t \rangle_F}{\|\mathbf{O}_{t-j}\|_F \|\mathbf{O}_t\|_F}\right)$.

The 3D plot reveals several profound insights that strongly support our approach. First, the remarkable geometric stability of the orthogonalized updates is not restricted to adjacent steps ($j = 1$); rather, it universally holds true across larger step intervals ($j > 1$), exhibiting an identical smooth and flat progression throughout the steady training phase. Second, the angles change slowly as j increases during the steady phase of loss descent, demonstrating a strong continuity where the values for neighboring intervals remain close to one another. Although the baseline angle grows marginally and gradually as the step interval j widens, its high predictability and low variance are strictly preserved. This continuous and regular trend along the j axis guarantees that an orthogonal update computed at a given step remains an exceptionally robust anchor. Provided that j does not take an excessively large value, this anchor can reliably generate and calibrate updates several steps into the future with low approximation error.

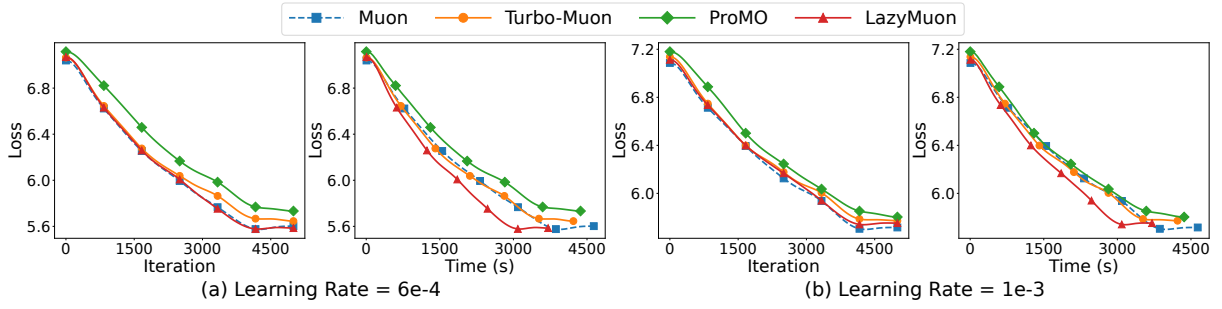
Figure 8: Results of Scaling Behavior Across Model Sizes



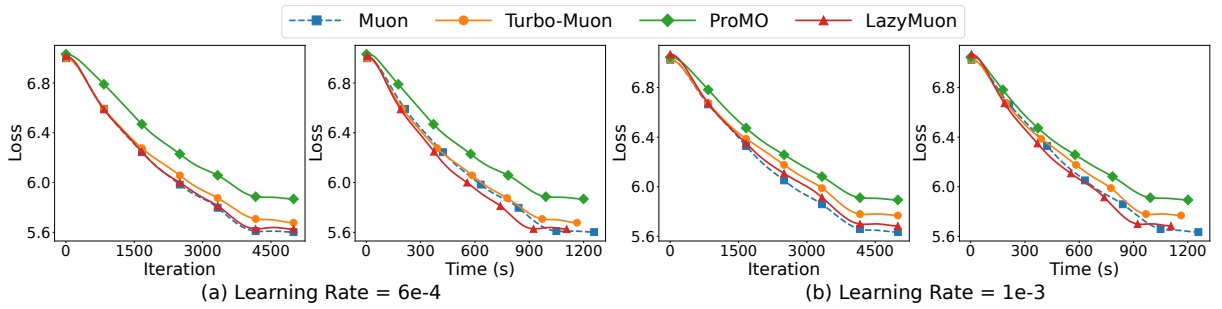
(A) Training curves on Qwen2.5-3B@OpenWebText across learning rates.



(B) Training curves on Qwen2.5-0.5B@OpenWebText across learning rates.



(C) Training curves on ViT-G@ImageNet across learning rates.



(D) Training curves on ViT-L@ImageNet across learning rates.

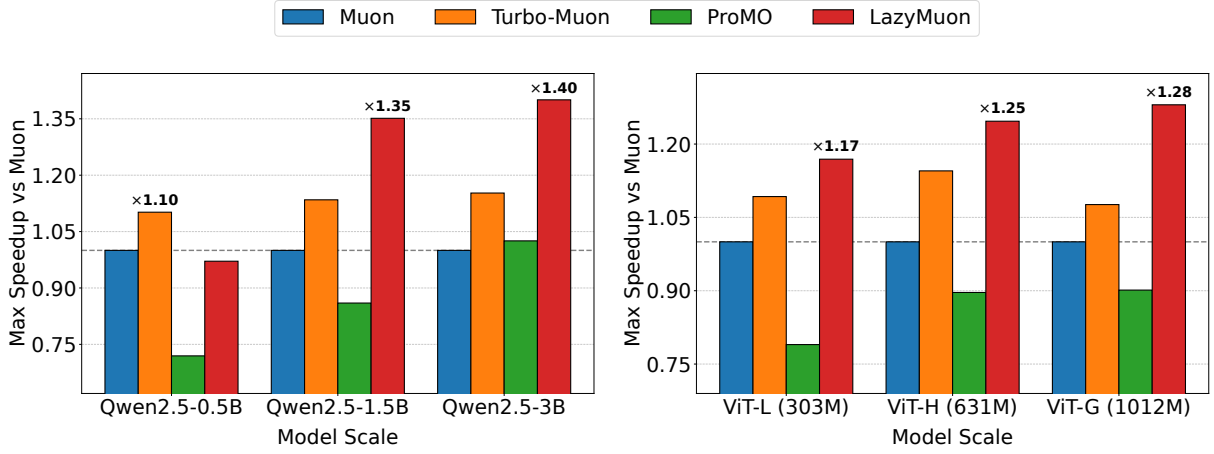
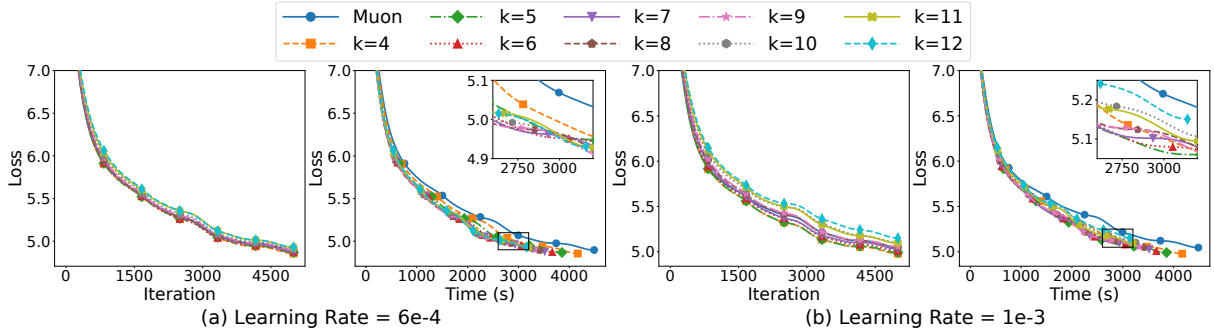
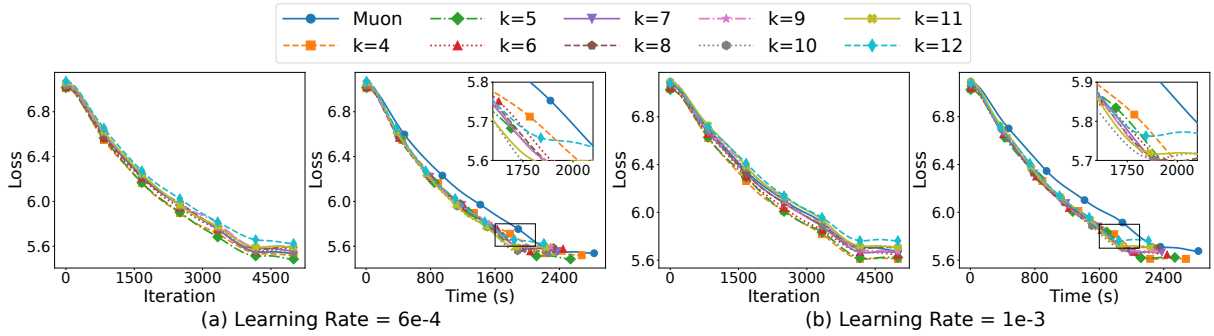


Figure 9: Maximum speedup of evaluated optimizers relative to Muon across varying model sizes. The experiments are conducted under a learning rate of 6×10^{-4} (Left: Qwen. Right: ViT.). Each bar indicates the highest acceleration ratio achieved by the algorithm when reaching the same target loss as Muon.

Figure 10: Results of Sensitivity Analysis of Hyperparameter

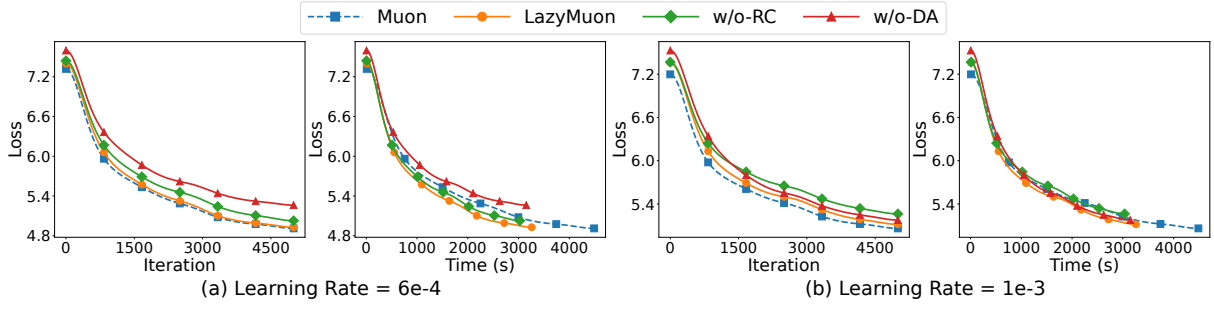


(A) Sensitivity analysis of the period length k on Qwen2.5-1.5B@OpenWebText. The plots compare the training loss curves of LazyMuon ($k \in [4, 12]$) and the Muon baseline across different learning rate settings.

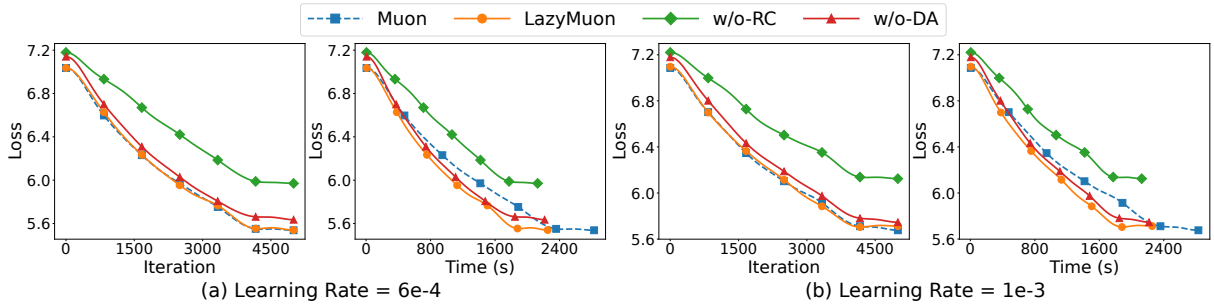


(B) Sensitivity analysis of the period length k on ViT-H@ImageNet. The plots compare the training loss curves of LazyMuon ($k \in [4, 12]$) and the Muon baseline across different learning rate settings.

Figure 11: Results of Ablation Study



(A) Ablation study on Qwen2.5-1.5B@OpenWebText. Convergence comparison of LazyMuon versus w/o-DA and w/o-RC under different learning rates (Left: Loss vs. Iteration; Right: Loss vs. Time).



(B) Ablation study on ViT-H@ImageNet. Convergence comparison of LazyMuon versus w/o-DA and w/o-RC under different learning rates (Left: Loss vs. Iteration; Right: Loss vs. Time).

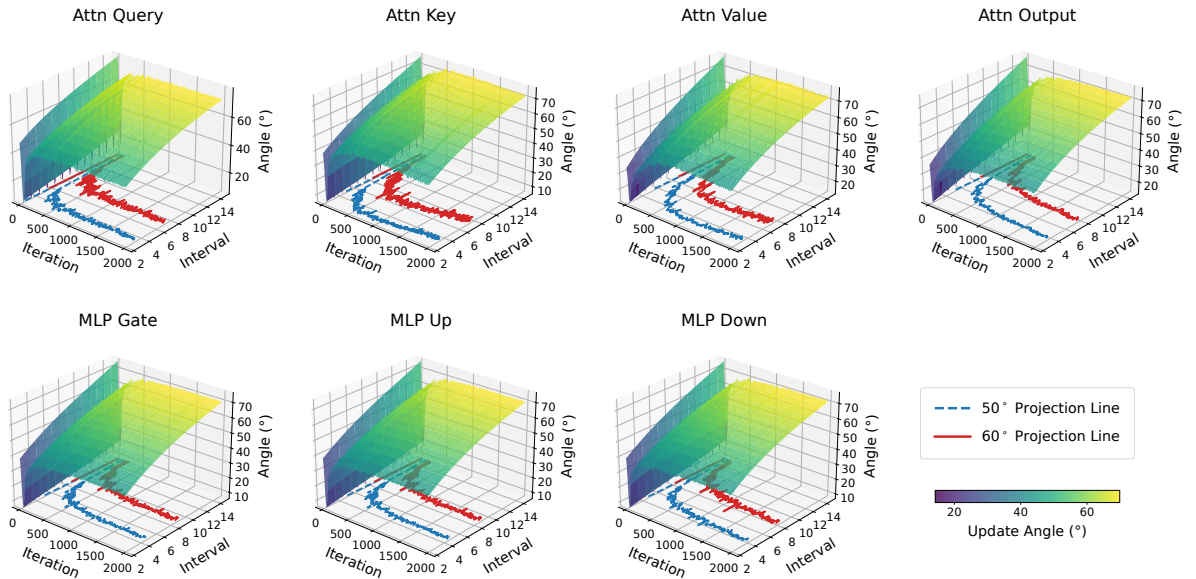


Figure 12: Evolution of the generalized interval angle $\theta_{t,j}(\mathbf{O})$ for orthogonalized updates across varying step intervals j . The horizontal axis denotes the training iteration t , the depth axis represents the step interval j , and the vertical axis indicates the resulting angle $\theta_{t,j}$.