# TMUON: TANGENT MOMENTUM IMPROVES MUON FOR LLM TRAINING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Geometry-aware optimization has recently advanced large-scale model pretraining by more effectively leveraging gradient structure and spectral information. Muon, as a representative example, uses Newton–Schulz orthogonalization to construct update directions from momentum matrices. However, we hypothesize that the standard momentum signal used in Muon contains substantial components that are geometrically aligned with the current model weights. In this work, we propose *TMuon*, a simple yet effective optimizer that incorporates tangent-projected momentum as input to Newton–Schulz orthogonalization within the Muon framework. Tangent filtering suppresses locally redundant components in the update signal, yielding a more informative momentum matrix for spectral normalization. We further show that norm-dependent tangent filtering leverages local geometry along the optimization trajectory and improves the input to Newton–Schulz orthogonalization. Across pretraining experiments on LLaMA and Qwen3 models, TMuon consistently improves optimization performance over Muon and recent variants at negligible additional cost. In particular, on Qwen3-1.7B trained with a $150$B-token budget, TMuon improves average benchmark accuracy over Muon by $2.3$ percentage points, suggesting a promising direction for LLM optimizers.

## 1 INTRODUCTION

Over the past decade, scaling neural networks has been a central engine of progress in deep learning (Kaplan et al., 2020). In particular, large language models (LLMs) trained on massive text corpora have exhibited emergent capabilities across a wide range of downstream tasks, including reasoning, knowledge acquisition, and code generation, as model parameters, data volume, and compute budgets continue to grow (Hoffmann et al., 2022; Yang et al., 2025; DeepSeek-AI, 2026). However, this trend toward ever-larger scale also raises a fundamental challenge: the rapidly increasing cost of training, which places substantial pressure on computational infrastructure, memory capacity, energy consumption, and hardware availability (Patterson et al., 2021; Chowdhury et al., 2025). Moreover, as the supply of high-quality text data approaches practical limits, there is a growing demand to extract maximum value from the available data (Villalobos et al., 2022). Thus, finding training algorithms that are both effective and efficient is critical.

Since the early development of transformer-based LLMs, AdamW (Loshchilov & Hutter, 2018) has been the dominant optimization method due to its robustness, simplicity, and strong performance across diverse neural architectures. By combining momentum, adaptive second-moment preconditioning, and decoupled weight decay, AdamW proved particularly effective for large-batch transformer training and facilitated the design of scalable distributed systems such as ZeRO (Rajbhandari et al., 2020; Wang et al., 2024). However, as model scale continued to grow, the limitations of purely element-wise adaptive methods became increasingly apparent, motivating interest in matrix-based optimizers such as Shampoo (Gupta et al., 2018) and SOAP (Vyas et al., 2024), which better exploit the structure of two-dimensional transformer parameters. More recently, Muon has attracted significant attention by applying Newton–Schulz iteration to momentum matrices to produce spectrally normalized updates (Jordan et al., 2024). Empirical studies show that Muon can outperform AdamW in transformer pretraining, suggesting that structured matrix optimization may offer a stronger paradigm for scaling neural networks (Liu et al., 2025; Shah et al., 2025). This progress has further stimulated the development of geometric optimizers and numerous Muon variants (Zhang et al., 2026c).



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.

Mano introduces a complementary optimization perspective by performing updates along tangent directions of constrained parameter manifolds (Gu & Xie, 2026). Mano demonstrates that local parameter geometry can be used to filter or redirect optimization signals, implying that not all components of a gradient or momentum matrix contribute equally to functional progress. Motivated by this perspective, we hypothesize that the momentum matrices used in Muon may contain substantial components aligned with the current model weights. Such directions may correspond to magnitude-changing or locally redundant updates, contributing limited objective improvement while consuming update capacity. Consequently, suppressing these components before orthogonalization may accelerate convergence and allow spectral normalization to make better use of more informative update directions.

To mitigate the redundant momentum components in Muon, we propose **TMuon**, a simple yet effective optimizer that combines tangent projection with Newton–Schulz orthogonalization. We further generalize this design under a product-manifold formulation that unifies row-wise, column-wise, and blockwise geometric constraints. Unlike Mano's standard orthogonal projection with normalized weight slices, TMuon contracts momentum against the unnormalized slices, yielding norm-dependent filtering of the weight-aligned component. Across LLaMA and Qwen3 pretraining experiments, TMuon improves optimization efficiency over Muon and recent variants with negligible additional cost. These gains in loss and benchmark accuracy extend consistently to Qwen3-0.6B and 1.7B models trained with budgets of up to 150B tokens. These results suggest that tangent-projected momentum is a promising new direction for efficient large-scale model optimization.

1. We introduce **TMuon**, a novel optimizer that applies tangent projection to filter less functionally effective momentum directions before Newton–Schulz orthogonalization.
2. Unlike standard tangent projection, TMuon uses norm-dependent tangent filtering to preserve local weight-norm information, exploit local optimization geometry, and provide a more informative input to spectral normalization.
3. Extensive experiments on LLaMA and Qwen3 models show that TMuon outperforms Muon with negligible computational overhead, including a 2.3-percentage-point gain in average benchmark accuracy for Qwen3-1.7B trained on 150B tokens.

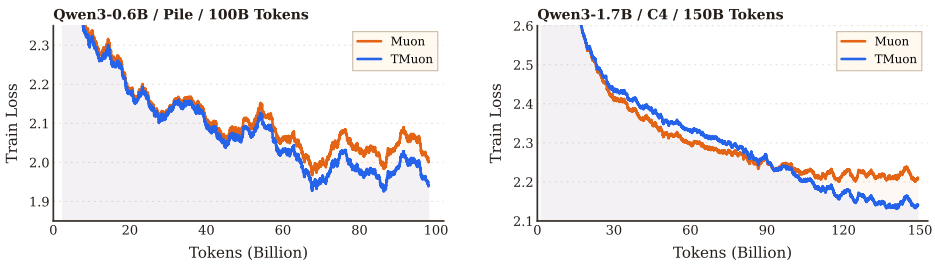



Figure 1: Training curves for Qwen3-0.6B and 1.7B models on the `Pile` and `C4` datasets, using 100B and 150B tokens with global batch sizes of 4.2M and 8.4M, respectively. TMuon reduces the final training loss by 0.07 and 0.08 compared with Muon and improves average benchmark accuracy by 1.24 and 2.29 percentage points, respectively, with little additional optimizer overhead. These results indicate that tangent-projected momentum delivers gains at scale.

## 2 Related Works

Beyond coordinate-wise adaptive methods such as AdamW, a long line of work has exploited matrix and tensor structure during optimization, demonstrating the benefits of spectral geometry for matrix-valued parameters (Carlson et al., 2015; Gupta et al., 2018; Vyas et al., 2024). Muon extends this broader context to momentum orthogonalization for transformer training by applying Newton–Schulz iteration to matrix gradients or momentum buffers (Jordan et al., 2024). Practical follow-up studies showed that, with suitable update scaling and momentum settings, Muon can train LLMs efficiently and outperform AdamW under matched budgets (Liu et al., 2025). Muon now serves as a baseline optimizer for pretraining frontier LLMs (Team et al., 2025; Zeng et al.,

2025; Tencent, 2026; DeepSeek-AI, 2026), and several theoretical analyses have sought to explain its empirical success (Su, 2025; Yu et al., 2026; Ma et al., 2026).

Several Muon variants modify different stages of the update pipeline to improve convergence and throughput. AdaMuon (Si et al., 2025), NAMO (Zhang et al., 2026b), and Muon$^2$ (Liu et al., 2026) introduce adaptive second-order statistics, while NorMuon (Li et al., 2025) and Muon+ (Zhang et al., 2026c) modify normalization before or after orthogonalization. Newton–Schulz has also been revisited from numerical and implementation perspectives. Polar Express (Amsel et al., 2025) and Dion2 (Ahn et al., 2025) leverage minimax-optimized polynomials and row-shrinking techniques to accelerate polar decomposition, while Gram Newton–Schulz iterates on the smaller symmetric Gram matrix using high-performance GEMM kernels (Zhang et al., 2026a). Additionally, MuonBP (Khaled et al., 2025) addresses distributed training bottlenecks through periodic blockwise updates, and Turbo-Muon (Boissin et al., 2025) accelerates Newton–Schulz convergence via preconditioning. These studies focus on efficiency and reduced systems overhead in large-scale training.

Related methods include HTMuon (Pang et al., 2026), which reshapes the singular-value spectrum through heavy-tailed spectral correction, and PolarGrad (Lau et al., 2025), which provides a unified preconditioned optimization framework with efficient algorithms for polar decomposition. Our method is complementary to these Muon variants. Rather than primarily modifying normalization, adaptive statistics, or the orthogonalization solver, we intervene in the geometry of the momentum matrix before orthogonalization by filtering components aligned with the current weights. Newton-Muon (Du & Su, 2026) also incorporates parameter curvature; however, it follows a derivation based on Newton's method, whereas TMuon integrates manifold-aware filtering with Muon's structured spectral updates.

# 3 METHODOLOGY

## 3.1 TANGENT MOMENTUM ORTHOGONALIZED BY NEWTON–SCHULZ

Motivated by the observation that certain momentum components induce limited functional change under the local parameter geometry, we improve Muon's Newton–Schulz input by filtering directions that produce magnitude drift rather than angular change. Before matrix orthogonalization, TMuon attenuates subdirections aligned with the current parameters. This geometry-aware preprocessing preserves Muon's efficiency while reducing reliance on low-impact directions.

Given a parameter matrix $W_{t-1} \in \mathbb{R}^{m\times n}$, momentum coefficient $\mu$, and objective $\mathcal{L}_t$ at iteration $t$, Muon first constructs a first-order momentum estimate

$$M_t = \mu M_{t-1} + \nabla \mathcal{L}_t(W_{t-1}), \tag{1}$$

and feeds it directly into Newton–Schulz iteration for approximate matrix orthogonalization (Jordan et al., 2024). In practical settings, a Nesterov-accelerated gradient (NAG) is computed with an additional lookahead step:

$$G_t = \mu M_t + \nabla \mathcal{L}_t(W_{t-1}), \tag{2}$$

which replaces $M_t$ as the Newton–Schulz input to accelerate convergence (Liu et al., 2025).

TMuon modifies this pipeline before orthogonalization. Because Newton–Schulz is a nonlinear matrix polynomial, local filtering and spectral normalization do not commute in general: $\mathrm{NS}(\mathcal{P}(G_t; W_{t-1})) \neq \mathcal{P}(\mathrm{NS}(G_t); W_{t-1})$. Filtering first therefore determines which local, weight-aligned components enter the global spectral operation while leaving the Newton–Schulz recurrence unchanged. Motivated by the row- and column-wise geometry of Mano (Gu & Xie, 2026), we define TMuon's unnormalized filters as

$$\mathcal{P}_{\mathrm{col}}(G_t; W_{t-1}) = \frac{V_t}{\|V_t\|^2_{\mathrm{col}} + \epsilon}, \quad V_t = G_t - W_{t-1} \odot \langle G_t, W_{t-1}\rangle_{\mathrm{col}}, \tag{3}$$

where $\|\cdot\|^2_{\mathrm{col}}$ denotes the column-wise norm, $\langle\cdot,\cdot\rangle_{\mathrm{col}}$ denotes the column-wise inner product, and $\odot$ denotes element-wise multiplication. Rescaling balances contributions across local geometric directions, and $\epsilon > 0$ provides numerical stability. Similarly,

$$\mathcal{P}_{\mathrm{row}}(G_t; W_{t-1}) = \frac{V_t}{\|V_t\|^2_{\mathrm{row}} + \epsilon}, \quad V_t = G_t - W_{t-1} \odot \langle G_t, W_{t-1}\rangle_{\mathrm{row}}. \tag{4}$$

These operations filter momentum independently over the active row or column slices, while rescaling balances their contributions before matrix orthogonalization. Unlike exact tangent projection, the retained aligned component depends on the local weight norm, as analyzed in Section 3.3. To incorporate both geometries, we alternate the active filter across successive optimization steps:

$$\mathcal{P}(G_t; W_{t-1}) = \begin{cases} \mathcal{P}_{\text{col}}(G_t; W_{t-1}), & \text{if column update,} \\ \mathcal{P}_{\text{row}}(G_t; W_{t-1}), & \text{if row update,} \end{cases}. \tag{5}$$

The resulting tangent momentum is

$$U_t = \mathcal{P}(G_t; W_{t-1}). \tag{6}$$

Its Newton–Schulz transform is

$$O_t = \text{Newton–Schulz}(U_t, T), \tag{7}$$

where $T$ is the number of Newton–Schulz iterations. This stage preserves the conditioning benefits of Muon's matrix orthogonalization while steering the update toward directions that are less redundant with the current parameter geometry. Compared with applying Newton–Schulz directly to $G_t$ or $M_t$, filtering first reduces the influence of weight-aligned subdirections before spectral rescaling. Following the practical Muon implementation, the final update is

$$W_t = W_{t-1} - \eta_t(0.2 \cdot O_t \cdot \sqrt{\max(m, n)} + \lambda W_{t-1}), \tag{8}$$

where $\eta_t$ is the learning rate and $\lambda$ is the weight decay. Algorithm 1 summarizes **TMuon**, or *Tangent MomentUm Orthogonalized by Newton–Schulz*. Because tangent filtering uses only row- or column-wise reductions and element-wise operations, the dominant optimizer cost remains the Newton–Schulz iteration; Section 3.4 quantifies this overhead.

**Algorithm 1:** The *TMuon* Optimizer

**Require:** Initial 2D weight $W_0 \in \mathbb{R}^{m \times n}$, initial momentum $M_0 = \mathbf{0}$, objective $\mathcal{L}$, learning rate schedule $\{\eta_t\}$, momentum coefficient $\mu$, weight decay $\lambda$, and number of Newton–Schulz steps $T$.
**for** $t = 1, 2, \ldots$ **do**
  $M_t = \mu M_{t-1} + \nabla \mathcal{L}_t(W_{t-1})$
  $G_t = \mu M_t + \nabla \mathcal{L}_t(W_{t-1})$
  $U_t = \mathcal{P}(G_t; W_{t-1})$
  $O_t = \text{Newton–Schulz}(U_t, T)$
  $W_t = W_{t-1} - \eta_t(0.2 \cdot O_t \cdot \sqrt{\max(m, n)} + \lambda W_{t-1})$
**end for**

### 3.2 TMuon on a General Product Manifold of Spheres

Mano interprets column- or row-wise tangent projection through an *oblique manifold*, a product of unit-norm vector spheres. TMuon extends this view from individual vectors to arbitrary matrix blocks, allowing the locality of the reference geometry to vary without imposing a single matrix-wide constraint. Specifically, we partition $W_{t-1}$ and $G_t$ into corresponding blocks $\Theta_{i,j}, \Gamma_{i,j} \in \mathbb{R}^{r_m \times r_n}$, where $i = 1, \ldots, m/r_m$ and $j = 1, \ldots, n/r_n$. The product manifold and blockwise filter are

$$\begin{aligned} \mathcal{M}_{\text{prod}} &= \left\{ W \in \mathbb{R}^{m \times n} : \|\Theta_{i,j}\|_F = 1,\ \forall i, j \right\}, \\ V_{i,j} &= \Gamma_{i,j} - \langle \Gamma_{i,j}, \Theta_{i,j} \rangle_F \Theta_{i,j}, \\ \left[\mathcal{P}_{\text{prod}}(G_t; W_{t-1})\right]_{i,j} &= \frac{V_{i,j}}{\|V_{i,j}\|_F^2 + \epsilon}. \end{aligned} \tag{9}$$

Here $\langle \Gamma_{i,j}, \Theta_{i,j} \rangle_F = \text{tr}(\Gamma_{i,j}^\top \Theta_{i,j})$ measures local alignment, $V_{i,j}$ filters that direction, and the final rescaling balances the contributions of different blocks. Importantly, $\mathcal{M}_{\text{prod}}$ serves only as a reference geometry for filtering momentum: TMuon neither constrains nor retracts the weights onto this manifold. The block shape instead determines the spatial scale at which parameter-aligned momentum is filtered.

Substituting Eq. 9 into Eq. 6 yields the following geometries:

- TMuon$_{\text{col}}$: column-wise projection on the oblique manifold with $(r_m, r_n) = (m, 1)$.
- TMuon$_{\text{row}}$: row-wise projection on the oblique manifold with $(r_m, r_n) = (1, n)$.
- TMuon$_{\text{iter}}$: stepwise alternation between column- and row-wise projection (Eq. 5).
- TMuon$_{\text{sphere}}$: matrix-wide projection on the sphere manifold with $(r_m, r_n) = (m, n)$.
- TMuon$_{\text{block}=B}$: blockwise projection with square block partitions when $r_m = r_n = B$.[1]

These choices span neuron-wise, local-tile, and matrix-wide geometries. Finer partitions provide more localized filtering but retain less cross-block structure, whereas coarser partitions preserve broader matrix coupling but apply fewer independent geometric constraints. TMuon$_{\text{iter}}$ alternates the two neuron-wise views across steps, incorporating both row and column geometry without composing both operations within a single update.

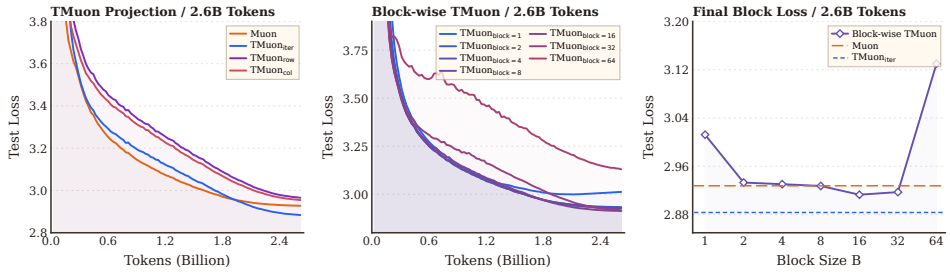



Figure 2: Projection ablations for LLaMA-130M trained on 2.6B C4 tokens. The left panel compares Muon with alternating, sphere-wide, row-wise, and column-wise TMuon; the center panel shows the training trajectories of blockwise variants; and the right panel summarizes their final losses against Muon and TMuon$_{\text{iter}}$. Alternating row- and column-wise projection achieves the best final loss, while block size $B = 16$ is the strongest blockwise configuration but remains behind TMuon$_{\text{iter}}$.

Table 1: Final test loss for the projection ablations on LLaMA-130M trained on 2.6B C4 tokens. Muon and TMuon$_{\text{iter}}$ use their tuned learning rates; the sphere-, row-, column-, and blockwise ablations use $\eta = 1.5 \times 10^{-3}$, $\alpha = 1$, and update RMS 0.2. Row- and column-wise alternation attains the lowest loss and is used as the default variant in subsequent experiments.

| Model | Tokens | Optimizer | Test Loss | Optimizer | Test Loss |
|---|---|---|---|---|---|
| LLaMA-130M | 2.6B | TMuon$_{sphere}$ | 2.9780 | TMuon$_{block=1}$ | 3.0123 |
| | | TMuon$_{row}$ | 2.9654 | TMuon$_{block=4}$ | 2.9304 |
| | | TMuon$_{col}$ | 2.9545 | TMuon$_{block=16}$ | **2.9130** |
| | | Muon | 2.9277 | TMuon$_{block=32}$ | 2.9173 |
| | | **TMuon$_{iter}$** | **2.8836** | TMuon$_{block=64}$ | 3.1299 |

To evaluate these geometries, we pretrain LLaMA-130M on 2.6B C4 tokens under the Chinchilla-optimal budget. Figure 2 and Table 1 show that TMuon$_{\text{iter}}$ attains the lowest final loss. The blockwise variants exhibit a non-monotonic dependence on manifold resolution: $B = 16$ performs best among the tested block sizes, whereas both finer and coarser partitions increase the loss, and every blockwise configuration trails alternation. The non-monotonic block-size trend shows that overly fine or coarse manifold constraints increase loss, whereas an appropriate granularity balances local information filtering with preservation of broader matrix structure.

The individual row- and column-wise filters remain competitive with the tuned Muon baseline, while alternating between the two geometries yields an additional gain. We therefore use TMuon$_{\text{iter}}$ as the default configuration in subsequent experiments. Together, these results indicate that projection granularity trades off localized geometric filtering against retention of broader matrix structure.

[1] At $B = 1$, the manifold normalization step recovers the sign operator in signSGD (Bernstein et al., 2018).

### 3.3 NORM-GATED TANGENT FILTERING

TMuon's tangent filtering differs from the standard tangent projection used by Mano (Gu & Xie, 2026). For a nonzero active row or column vector $w$ and the corresponding Nesterov-momentum vector $g$, let $s = \|w\|_2^2$, $g_\parallel = \langle g, w\rangle w/s$, and $g_\perp = g - g_\parallel$. Before row- or column-wise normalization, the two filters satisfy

$$\mathcal{F}_{\mathrm{std}}(g; w) = g_\perp, \qquad \mathcal{F}_{\mathrm{TMuon}}(g; w) = g - \langle g, w\rangle w = g_\perp + (1 - s)g_\parallel. \tag{10}$$

Thus, standard tangent projection removes the weight-aligned component exactly and independently of the scale of $w$, whereas TMuon retains it with the norm-dependent coefficient $1 - s$. When $s \geq 2$, the residual is reversed, and the filter actively suppresses the current weight direction, acting as a local regularizer on oversized blocks. Rather than deleting the radial signal unconditionally, TMuon performs norm-dependent tangent filtering, relaxing the constraint on small slices and counteracting growth in large slices, thereby redistributing update energy across the product of hyperspheres.

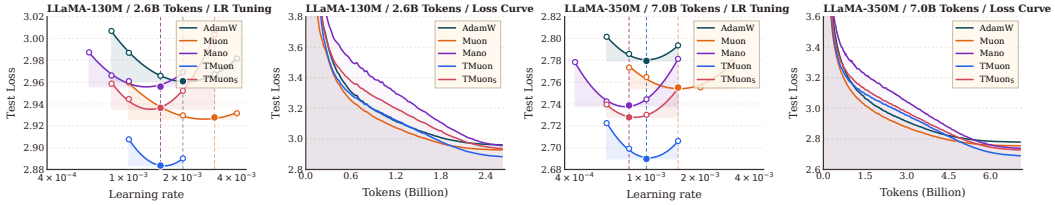



Figure 3: Learning rate sweeps and test-loss trajectories for LLaMA-130M and 350M trained on C4, comparing TMuon with TMuon$_S$, the standard-tangent-momentum variant. TMuon reaches lower final losses at both scales, indicating that retaining local weight-norm information in the tangent filter is more effective than fully normalizing the projection reference.

We evaluate this design choice by comparing TMuon with TMuon$_S$, which uses standard tangent momentum, in Figure 3. Both variants alternate row- and column-wise operations across steps and feed the resulting momentum into the same Newton–Schulz procedure, isolating norm-gated from standard tangent filtering. TMuon outperforms TMuon$_S$ when each uses its best tuned learning rate, suggesting that local parameter magnitudes provide useful information for tangent filtering.

### 3.4 COMPUTATIONAL OVERHEAD OF TMUON

We analyze TMuon's computational overhead relative to momentum SGD and conventional Muon. Let $W \in \mathbb{R}^{m\times n}$, $d = \min(m, n)$, and $D = \max(m, n)$, and count a multiply–add as two FLOPs. The leading optimizer costs in FLOPs are

$$F_{\mathrm{SGD}} \approx 2dD, \qquad F_{\mathrm{NS}} \approx T(4d^2D + 2d^3), \qquad F_{\mathrm{proj}} \approx 7dD, \tag{11}$$

where $T$ is the number of Newton–Schulz steps. Normalizing $W$ instead of caching its norms increases $F_{\mathrm{proj}}$ to at most $10dD$. The projection therefore adds only lower-order reductions and element-wise operations, and the corresponding overhead ratios are

$$\frac{F_{\mathrm{proj}}}{F_{\mathrm{NS}}} \approx \frac{7D}{2Td(2D + d)}, \qquad \frac{F_{\mathrm{TMuon}} - F_{\mathrm{SGD}}}{F_{\mathrm{SGD}}} \approx \frac{7}{2} + T\left(2d + \frac{d^2}{D}\right). \tag{12}$$

For square matrices, the first ratio is $7/(6Td)$ and is below $0.02\%$ for the tested shapes with $T = 5$. Thus, TMuon has the same asymptotic complexity as Muon, and the relative projection cost decreases as $O(1/d)$ with matrix dimension. Table 2 measures a larger but still small $0.4\%$–$3.5\%$ latency increase over Muon because reductions are memory-bandwidth bound and introduce synchronization and kernel-launch costs.

## 4 EXPERIMENTS

In this section, we compare TMuon with AdamW, Muon, and Mano across four LLaMA model scales pretrained on C4 (Raffel et al., 2020), a mixture-of-experts backbone pretrained on the Pile (Gao et al.,

Table 2: Runtime comparison of Muon and TMuon on LLaMA architectures. Mean runtime over 1,000 runs is reported for BFloat16 tensors on an RTX 4090 GPU. Row- and column-wise TMuon results are reported separately; both add little overhead relative to Muon. The measurements confirm that tangent filtering preserves Muon's asymptotic complexity, in line with the FLOP analysis in Eq. 12.

| Parameter | Attention | MLP | Attention | MLP | Attention | MLP |
|---|---|---|---|---|---|---|
| Models | **LLaMA-3B** | | **LLaMA-7B** | | **LLaMA-70B** | |
| Shape $(m, n)$ | (2560, 2560) | (2560, 6848) | (4096, 4096) | (4096, 11008) | (8192, 8192) | (8192, 16384) |
| Muon | 3.810 (ms) | 8.120 (ms) | 14.955 (ms) | 30.484 (ms) | 111.288 (ms) | 184.677 (ms) |
| TMuon$_{row}$ | 3.825 (ms) | 8.170 (ms) | 15.256 (ms) | 31.528 (ms) | 112.824 (ms) | 187.649 (ms) |
| TMuon$_{col}$ | 3.825 (ms) | 8.197 (ms) | 15.266 (ms) | 31.565 (ms) | 112.823 (ms) | 187.633 (ms) |

2020), and long-run Qwen3 experiments on both C4 and Pile. We tune the learning rate separately for each optimizer in the compute-optimal LLaMA and MoE experiments. For the long-run Qwen3 experiments, we follow the equal-update-RMS protocol of Liu et al. (2025) and use a shared learning rate for Muon and TMuon at each model scale. All other hyperparameters, including weight decay $\lambda = 0.1$, 1000 warmup steps, a minimum learning rate ratio of 0.1, and gradient clipping of 1.0, follow standard practices in previous studies (Raffel et al., 2020; Zhao et al., 2024). All experiments are conducted in *BFloat16* precision on H20 and H100 GPUs.

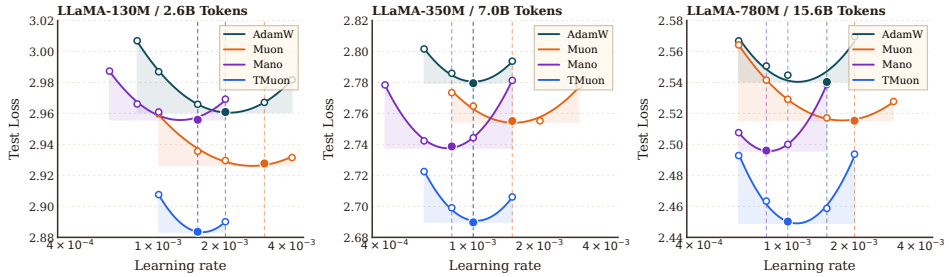



Figure 4: Optimizer-specific learning rate sweeps for LLaMA-130M, 350M, and 780M models on C4 under Chinchilla-optimal token budgets.

**Learning rate sweep.** Following Chinchilla-optimal scaling laws, we pretrain LLaMA models with 130M, 350M, 780M, and 1.3B parameters on C4 (Raffel et al., 2020), using 2.6B, 7.0B, 15.6B, and 26.8B tokens, respectively. We first identify each optimizer's preferred learning rate, since a single shared value would otherwise confound the optimizer comparison. Figure 4 reports the dedicated per-scale sweeps. Across scales, the optima are well separated: AdamW peaks near $1$–$2 \times 10^{-3}$, Muon near $1.5$–$3 \times 10^{-3}$, and Mano at $8 \times 10^{-4}$–$1.5 \times 10^{-3}$, while TMuon prefers a slightly lower rate around $1 \times 10^{-3}$ at 130M–780M. TMuon achieves the lowest test loss when each optimizer is evaluated at its best completed learning rate. For LLaMA-1.3B, we evaluate selected learning rates for each optimizer rather than complete sweeps due to computational constraints.

**Training dynamics.** Using each optimizer's best completed learning rate, Figure 5 shows the corresponding test-loss trajectories, and Table 3 lists the final test losses. TMuon reaches the lowest final loss at all four scales. Its margins over Muon are 0.044, 0.065, 0.065, and 0.069 from 130M to 1.3B, respectively, while its corresponding margins over Mano are 0.072, 0.049, 0.046, and 0.055. Across scales, TMuon progresses more slowly than Muon early in training but overtakes it during the middle-to-late stage and sustains stronger loss reduction toward the end. This recurring crossover suggests that TMuon's gains arise from sustained late-stage progress, consistent with norm-dependent tangent filtering becoming more effective as the local parameter geometry evolves.

**Mixture-of-Experts (MoE).** We further evaluate TMuon on a mixture-of-experts backbone whose architecture is borrowed from the Qwen3-8B-A1B design and rescaled to a 1B-parameter budget with roughly 200M parameters activated per token, denoted Qwen3-1B-A200M. Each optimizer is tuned

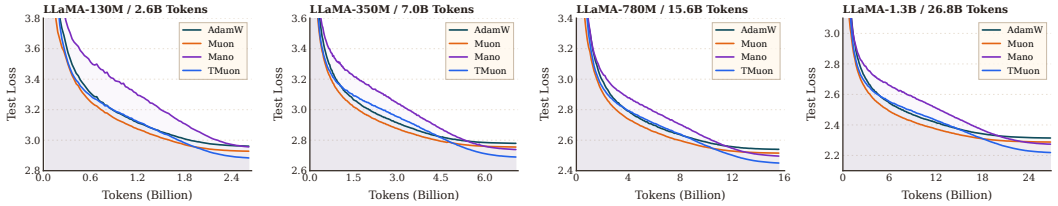



Figure 5: Test-loss trajectories for LLaMA-130M, 350M, 780M, and 1.3B models trained on C4 under Chinchilla-optimal token budgets. Each trajectory uses the best available optimizer-specific learning rate from the runs in Figure 4 and Table 3. TMuon reaches the lowest final loss at all four scales, with its advantage over Muon emerging later in training.

Table 3: Final test loss for LLaMA models trained on C4 under Chinchilla-optimal token budgets and Qwen3-1B-A200M pretrained on the Pile. Each entry reports the best completed optimizer-specific run, with TMuon denoting `proj_norm` in every setting. TMuon attains the lowest loss in all five settings, including margins of 0.069 over Muon on LLaMA-1.3B and 0.037 on the MoE backbone.

| Model | Batch Size | AdamW | Muon | Mano | TMuon (vs. Muon) |
|---|---|---|---|---|---|
| LLaMA-130M | 0.26M | 2.9609 | 2.9277 | 2.9559 | **2.8836** (-0.0441) |
| LLaMA-350M | 0.26M | 2.7795 | 2.7551 | 2.7387 | **2.6897** (-0.0654) |
| LLaMA-780M | 1.0M | 2.5403 | 2.5153 | 2.4960 | **2.4502** (-0.0651) |
| LLaMA-1.3B | 1.0M | 2.3143 | 2.2878 | 2.2738 | **2.2191** (-0.0687) |
| Qwen3-1B-A200M | 4.2M | 2.2568 | 2.1630 | 2.2443 | **2.1263** (-0.0367) |

separately for this backbone. Figure 6 shows the corresponding test loss together with gradient and parameter norms recorded during 50.4B tokens of pretraining on the Pile. TMuon attains the lowest final test loss of 2.1263, improving over Muon (2.1630), Mano (2.2443), and AdamW (2.2568). The norm trajectories also reveal different optimization dynamics: TMuon maintains substantially smaller parameter norms than Muon despite exhibiting larger gradient norms. Its lower loss is therefore not accompanied by uniform attenuation of the gradient signal; instead, this pattern is consistent with norm-dependent tangent filtering selectively modulating weight-aligned momentum before orthogonalization.

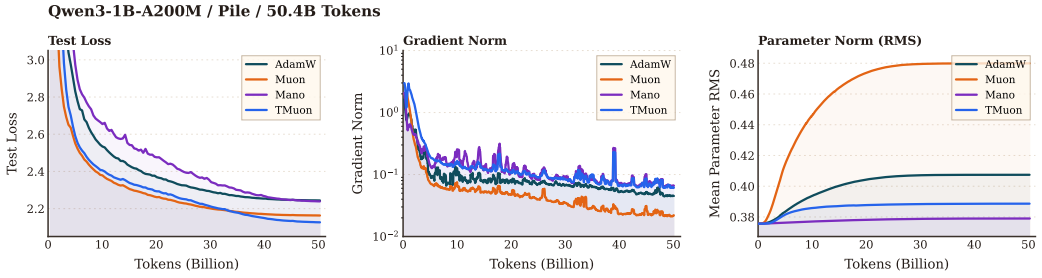



Figure 6: Test loss (left), gradient norm (middle), and mean parameter RMS (right) for Qwen3-1B-A200M pretrained on 50.4B Pile tokens with each optimizer at its own tuned learning rate. TMuon achieves the lowest test loss and maintains lower parameter RMS than Muon despite its larger gradient norms, a pattern consistent with norm-dependent filtering of weight-aligned momentum.

**Scalability.** To further validate TMuon's scalability, we compare it with Muon in practical long-run pretraining scenarios with much larger batch sizes and token budgets. We pretrain Qwen3-0.6B and Qwen3-1.7B on 100B tokens from the Pile and 150B tokens from C4, using global batch sizes of 4.2M and 8.4M tokens, respectively. Following the equal-update-RMS convention of Liu et al. (2025), we use the same learning rate for both optimizers at each scale: $6.0 \times 10^{-4}$ for the 0.6B model and $1.0 \times 10^{-3}$ for the 1.7B model. As shown in Figure 1, TMuon achieves lower late-stage

and final training loss at both scales. On Qwen3-1.7B, TMuon initially trails Muon but overtakes it in the second half of training, a late-stage crossover that we leave for further investigation.

Table 4 shows that this training-loss advantage transfers to downstream tasks: TMuon improves the average accuracy despite regressions on several individual benchmarks. Average benchmark accuracy increases by 1.24 and 2.29 percentage points on Qwen3-0.6B and Qwen3-1.7B, respectively, indicating that TMuon scales to large batch sizes and extensive data while improving overall model quality. The late crossover on Qwen3-1.7B mirrors the LLaMA trajectories, suggesting that TMuon's advantage becomes more pronounced after extended optimization rather than during the initial rapid-loss-reduction phase. This behavior may be especially relevant under large token budgets, where sustained late-stage improvement determines final model quality.

Table 4: Evaluation results on standard world knowledge, language reasoning, and science question-answering benchmarks under 0-shot settings. The Qwen3-0.6B and 1.7B model pairs are trained on 100B Pile tokens and 150B C4 tokens, respectively. TMuon improves average accuracy over Muon by 1.24 and 2.29 percentage points in the two settings, demonstrating that its training-loss advantage translates into gains in overall downstream performance.

| | **Benchmark** | **Qwen3-0.6B** (100B Tokens / Pile) | | **Qwen3-1.7B** (150B Tokens / C4) | |
|---|---|---|---|---|---|
| | **Optimizer** | Muon | TMuon | Muon | TMuon |
| **World Knowl.** | ARC-Easy | 48.48 | 50.63 (+2.15) | 52.40 | 56.19 (+3.79) |
| | ARC-Challenge | 20.05 | 21.76 (+1.71) | 26.37 | 25.77 (-0.60) |
| | BoolQ | 61.31 | 60.76 (-0.55) | 57.65 | 58.07 (+0.42) |
| | MMLU | 23.91 | 23.13 (-0.78) | 23.38 | 23.23 (-0.15) |
| | LogiQA | 21.81 | 21.04 (-0.77) | 23.04 | 28.57 (+5.53) |
| **Lang. & Reas.** | CommonsenseQA | 18.67 | 20.48 (+1.81) | 19.98 | 19.98 (+0.00) |
| | OpenBookQA | 16.00 | 17.40 (+1.40) | 20.20 | 25.20 (+5.00) |
| | HellaSwag | 31.98 | 33.49 (+1.51) | 41.76 | 45.76 (+4.00) |
| | LAMBADA | 31.17 | 36.11 (+4.94) | 33.57 | 39.41 (+5.84) |
| | WinoGrande | 50.36 | 53.12 (+2.76) | 57.14 | 59.51 (+2.37) |
| **Math & Sci.** | MathQA | 23.25 | 22.55 (-0.70) | 22.75 | 22.11 (-0.64) |
| | MedMCQA | 30.74 | 32.06 (+1.32) | 29.14 | 30.12 (+0.98) |
| | SciQ | 81.30 | 81.90 (+0.60) | 79.10 | 82.30 (+3.20) |
| | PIQA | 64.74 | 66.70 (+1.96) | 72.31 | 74.59 (+2.28) |
| | **Average** | **37.41** | **38.65 (+1.24)** | **39.91** | **42.20 (+2.29)** |

## 5 Conclusion

In this study, we introduced TMuon, which combines local parameter geometry with global spectral normalization by applying norm-dependent tangent filtering before Muon's Newton–Schulz orthogonalization. Unlike standard tangent projection, this filter retains local weight-scale information and adaptively modulates weight-aligned momentum using only row- or column-wise reductions and element-wise operations. Our product-manifold analysis and projection ablations further characterize how geometric granularity balances localized filtering with preservation of broader matrix structure. Across LLaMA and Qwen3 models, TMuon reduces pretraining loss relative to Muon and recent variants, while the long-run Qwen3 experiments show that these gains extend to large token budgets and average downstream performance. Together, these results establish tangent-filtered momentum as a practical direction for improving matrix-based LLM optimizers at negligible additional cost.

However, this study has several limitations. Although the norm-dependent decomposition explains how TMuon modulates weight-aligned momentum, its broader optimization mechanism is not yet fully understood, and we do not provide a formal convergence analysis or a complete theoretical account of its interaction with Newton–Schulz dynamics. Moreover, the large-scale Qwen3 experiments follow the equal-update-RMS convention and use a shared learning rate for Muon and TMuon rather than optimizer-specific learning rate tuning. Broader tuning and deeper theoretical study remain important directions for future work.

## AI USE STATEMENT

Generative AI tools were used to refine the presentation of the paper, polish the wording, and help review the interpretation of the experimental results. The authors examined every AI-assisted suggestion, and the scientific claims, figures, tables, and conclusions remain entirely the authors' responsibility.

## ETHICS STATEMENT

The pretraining runs use the public C4 and Pile corpora, which may contain personally identifiable information, biased content, or other sensitive material. This paper does not release checkpoints, generated samples, or new datasets and therefore does not introduce direct deployment-related risks. The contribution is a purely methodological study, so we do not identify ethical risks beyond those associated with the existing corpora. Improving pretraining efficiency may also reduce the compute and energy needed to train future LLMs, providing a modest sustainability benefit.

## REPRODUCIBILITY STATEMENT

Algorithm 1 gives the complete TMuon update used in our code, including tangent momentum and the Newton–Schulz step. Section 3 specifies the projection variants, the norm-gated filter, its standard-tangent ablation, and the FLOP analysis. Section 4 records the corpora, model scales, token budgets, shared hyperparameters, learning rate selection, and evaluation protocol, with numerical results in Tables 1–4.

## REFERENCES

Kwangjun Ahn, Noah Amsel, and John Langford. Dion2: A simple method to shrink matrix in muon. *arXiv preprint arXiv:2512.16928*, 2025.

Noah Amsel, David Persson, Christopher Musco, and Robert M Gower. The polar express: Optimal matrix sign methods and their application to the muon algorithm. *arXiv preprint arXiv:2505.16932*, 2025.

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. PMLR, 2018.

Thibaut Boissin, Thomas Massena, Franck Mamalet, and Mathieu Serrurier. Turbo-Muon: Accelerating Orthogonality-Based Optimization with Pre-Conditioning. working paper or preprint, December 2025. URL `https://hal.science/hal-05390446`.

David E Carlson, Edo Collins, Ya-Ping Hsieh, Lawrence Carin, and Volkan Cevher. Preconditioned spectral descent for deep learning. *Advances in neural information processing systems*, 28, 2015.

Md Naseef Ur Rahman Chowdhury, Ahshanul Haque, and Hamdy Soliman. The hidden cost of ai: Unraveling the power-hungry nature of large language models. 2025.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026.

Zhehang Du and Weijie Su. The newton-muon optimizer. *arXiv preprint arXiv:2604.01472*, 2026.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Yufei Gu and Zeke Xie. Mano: Restriking manifold optimization for llm training. *arXiv preprint arXiv:2601.23000*, 2026.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, pp. 1842–1850. PMLR, 2018.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL `https://kellerjordan.github.io/posts/muon/`.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Ahmed Khaled, Kaan Ozkara, Tao Yu, Mingyi Hong, and Youngsuk Park. Muonbp: Faster muon via block-periodic orthogonalization. *arXiv preprint arXiv:2510.16981*, 2025.

Tim Tsz-Kit Lau, Qi Long, and Weijie Su. Polargrad: A class of matrix-gradient optimizers from a unifying preconditioning perspective. *arXiv preprint arXiv:2505.21799*, 2025.

Zichong Li, Liming Liu, Chen Liang, Weizhu Chen, and Tuo Zhao. Normuon: Making muon more efficient and scalable. *arXiv preprint arXiv:2510.05491*, 2025.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Ziyue Liu, Ruijie Zhang, Zhengyang Wang, Yequan Zhao, Yupeng Su, Zi Yang, and Zheng Zhang. Muon$^2$: Boosting muon via adaptive second-moment preconditioning. *arXiv preprint arXiv:2604.09967*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

Jianhao Ma, Yu Huang, Yuejie Chi, and Yuxin Chen. Preconditioning benefits of spectral orthogonalization in muon. *arXiv preprint arXiv:2601.13474*, 2026.

Tianyu Pang, Yujie Fang, Zihang Liu, Shenyang Deng, Lei Hsiung, Shuhua Yu, and Yaoqing Yang. Htmuon: Improving muon via heavy-tailed spectral correction. *arXiv preprint arXiv:2603.10067*, 2026.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*, 2021.

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.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: international conference for high performance computing, networking, storage and analysis*, pp. 1–16. IEEE, 2020.

Ishaan Shah, Anthony M Polloreno, Karl Stratos, Philip Monk, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, Anil Thomas, Ashish Tanwer, Darsh J Shah, et al. Practical efficiency of muon for pretraining. *arXiv preprint arXiv:2505.02222*, 2025.

Chongjie Si, Debing Zhang, and Wei Shen. Adamuon: Adaptive muon optimizer. *arXiv preprint arXiv:2507.11005*, 2025.

Weijie Su. Isotropic curvature model for understanding deep learning optimization: Is gradient orthogonalization optimal? *arXiv preprint arXiv:2511.00674*, 2025.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Tencent. Hy3-preview, 2026. URL https://huggingface.co/tencent/Hy3-preview.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Will we run out of data? limits of llm scaling based on human-generated data. *arXiv preprint arXiv:2211.04325*, 2022.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Mujin Kwun, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. Soap: Improving and stabilizing shampoo using adam. *arXiv preprint arXiv:2409.11321*, 2024.

Guanhua Wang, Heyang Qin, Sam Ade Jacobs, Xiaoxia Wu, Connor Holmes, Zhewei Yao, Samyam Rajbhandari, Olatunji Ruwase, Feng Yan, Lei Yang, et al. Zero++: Extremely efficient collective communication for large model training. In *The Twelfth International Conference on Learning Representations*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Dingzhi Yu, Hongyi Tao, Yuanyu Wan, Luo Luo, and Lijun Zhang. Sign-based optimizers are effective under heavy-tailed noise. *arXiv preprint arXiv:2602.07425*, 2026.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Jack Zhang, Noah Amsel, Berlin Chen, and Tri Dao. Gram newton-schulz, 2026a. URL https://dao-ailab.github.io/blog/2026/gram-newton-schulz/.

Minxin Zhang, Yuxuan Liu, and Hayden Schaeffer. Adam improves muon: Adaptive moment estimation with orthogonalized momentum. *arXiv preprint arXiv:2602.17080*, 2026b.

Ruijie Zhang, Yequan Zhao, Ziyue Liu, Zhengyang Wang, and Zheng Zhang. Muon+: Towards better muon via one additional normalization step. *arXiv preprint arXiv:2602.21545*, 2026c.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.