Controlled LLM Training on Spectral Sphere

Tian Xie[1 *] **Haoming Luo**[2] **Haoyu Tang**[2] **Yiwen Hu**[2] **Jason Klein Liu** **Qingnan Ren**[1]

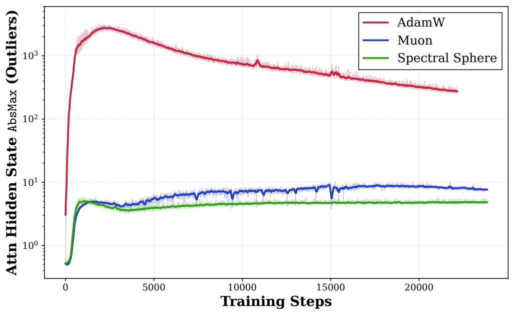
Yang Wang[1] **Wayne Xin Zhao**[2] **Rui Yan**[3] **Bing Su**[2] **Chong Luo**[1] **Baining Guo**[1]

[1]Microsoft Research Asia [2]Renmin University [3]Wuhan University

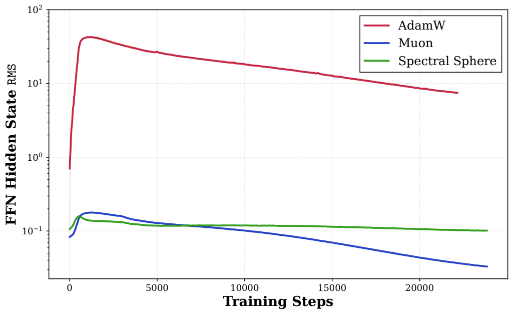
Project Page: Spectral-Sphere-Optimizer

Abstract

Scaling large models requires optimization strategies that ensure rapid convergence grounded in stability. Maximal Update Parametrization (μP) provides a theoretical safeguard for width-invariant $\Theta(1)$ activation control, whereas emerging optimizers like Muon are only "half-aligned" with these constraints: they control updates but allow weights to drift. To address this limitation, we introduce the **Spectral Sphere Optimizer (SSO)**, which enforces strict module-wise spectral constraints on both weights and their updates. By deriving the steepest descent direction on the spectral sphere, SSO realizes a fully μP-aligned optimization process. To enable large-scale training, we implement SSO as an efficient parallel algorithm within Megatron. Through extensive pretraining on diverse architectures, including Dense 1.7B, MoE 8B-A1B, and 200-layer DeepNet models, SSO consistently outperforms AdamW and Muon. Furthermore, we observe significant practical stability benefits, including improved MoE router load balancing, suppressed outliers, and strictly bounded activations.



(a) `AbsMax` (outliers) of Attention Activations



(b) `RMS` of FFN Activations

Figure 1: **Training dynamics of Dense-1.7B activations (log-scaled cross-layer averages).** Our Spectral Sphere maintains constant activation magnitudes throughout training because its μP-metrized constraints on the spectral manifold ensure that the activation `RMS` remains at $\Theta(1)$ scale. Muon activations show a mild drift due to learning rate decay and weight decay. By contrast, AdamW proves the most unstable, generating significantly larger activations, with attention `AbsMax` and FFN `RMS` reaching $\sim 100\times$ magnitude compared to those spectral optimizers.

[*]Corresponding to unakar666@gmail.com

Contents

1 Introduction **3**

2 Preliminary **3**

2.1 Maximal Update Parametrization (μP) 3

2.2 Steepest Descent under Different Norms 4

2.3 Muon Optimizer 4

3 Method **5**

3.1 Optimization Target Formulation 5

3.2 First-Order Tangent Space Constraint 6

3.3 Second-Order Manifold Constraint 7

3.4 Overall Algorithm & Interpretation 7

4 Algorithm Details **8**

4.1 Spectral Radius Scale 8

4.2 Learning Rate Scaler 8

4.3 Module Granularity 9

5 Infrastructure Design **10**

5.1 Bottleneck Analysis 10

5.2 Optimization Pipeline 10

5.3 Future Improvements 11

6 Scaling Experiments **12**

6.1 Experimental Setup 12

6.2 Dense 1.7B 12

6.3 MoE 8B-A1B 12

6.4 DeepNet 200-Layer 13

7 Discussion **14**

A Appendix **19**

A.1 Duality with Spectral Norm 19

A.2 Proofs: Localization of the Root of $h(\lambda)$ 19

A.3 Dynamic Spectral Weight Decay 20

A.4 MoE Routing Scaling Factor 21

A.5 μP Width Scaling 22

1 Introduction

LLM training is, at its core, a pursuit of **convergence speed** grounded in the necessity of **stability**. While the community has explored various optimization strategies, we argue that the essential principle governing training stability is the *Maximal Update Parametrization* (**μ**P) [Yang et al., 2023]. By mandating that the spectral norms of weights and updates scale as $\Theta(\sqrt{d_{\text{out}}/d_{\text{in}}})$ to ensure width-invariant activations remains $\Theta(1)$ scale, **μ**P serves as the mathematical safeguard against activation explosions [Takase et al., 2025]. However, the current landscape is saturated with methods that fail to satisfy these fundamental conditions. Conventional soft regularization methods, such as decoupled weight decay or initialization strategies, prove insufficient over long training horizons [Kosson et al., 2025]. This unconstrained weight drift destabilizes the *effective step size* (update-to-weight ratio) and degrades feature learning.

On the other side of the spectrum lies the pursuit of optimal convergence. The recent Muon optimizer [Jordan et al., 2024] has demonstrated remarkable efficiency, often interpreted as steepest descent under the spectral norm. In analyzing Muon, we uncover a surprising insight: it acts as a **"half-aligned"** solution to the **μ**P constraints. However, maintaining stable features requires constraining not only the updates but also the weights themselves. Unstable activations like attention logits explosion were still observed in Muon training [Kimi Team et al., 2025]. Consequently, practitioners are forced to rely on "non-essential" architectural patches to artificially force stability — ranging from aggressive normalization schemes like SandwichNorm [Ding et al., 2021] and QK-Norm [Henry et al., 2020], to ad-hoc fixes like logit softcapping [Kimi Team et al., 2025] — often at the cost of model expressivity and requiring extensive hyperparameter tuning. This observation motivates a fundamental question:

> *What if an optimizer could simultaneously satisfy the steepest descent property for* ***convergence speed*** *and the strict **μ**P constraints for* ***fundamental stability****?*

To answer this, we propose the mathematically unique solution that unifies these two objectives. By identifying the spectral sphere as the natural manifold for stable feature learning, SSO derives the steepest descent direction constrained within this geometry. Unlike heuristic manifold projection methods [Xie et al., 2025, Pethick et al., 2025], SSO solves a constrained optimization problem in the tangent space via a Lagrange multiplier search, followed by a retraction step to map the trajectory back onto the spectral sphere.

To enable large-scale training, we offer a systematic implementation in Megatron. We provide principled guidelines for spectral preconditioned optimization, specifically deriving the optimal *learning rate scaler*, determining the critical *atomic module granularity*, and identifying the optimal *spectral radius* to control activation at optimal scales precisely. These offer a robust recipe for large-scale training. Specifically to mitigate the overhead of the iterative root solver, we utilize a novel distributed strategy centered on atomic module sharding[Nvidia, 2025]. This technique partitions fused params into independent spectral units, enabling communication-free local updates. We further address solver-induced workload imbalance through a size-aware ping-pong placement strategy, and accelerate matrix operations using adaptive kernel dispatcher, alongside multi-stream execution and singular vector caching.

Empirically, we validate SSO through extensive pretraining experiments across various scales, including Dense 1.7B, MoE 8B-A1B, and 200-layer DeepNet models. SSO consistently outperforms AdamW and Muon while uniquely preserving stable **μ**P learning rate transfer. Notably, it yields superior training dynamics: it significantly improves MoE router load balancing, suppresses outliers in deep networks, and strictly bounds activations within a tunable scale.

2 Preliminary

2.1 Maximal Update Parametrization (μP)

μP prescribes how activations and weight updates should scale with width to preserve feature learning [Yang et al., 2023]. Ideal feature learning requires the scale of activations to remain as invariant as possible. We use **operator norm** to characterize how the norm of activations changes through a linear layer.

Considering a linear layer $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ with $\boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, $\boldsymbol{x} \in \mathbb{R}^{d_{\text{in}}}$, the RMS norm is defined as

$$\|\boldsymbol{x}\|_{\text{rms}} := \frac{\|\boldsymbol{x}\|_2}{\sqrt{d_{\text{in}}}}, \tag{1}$$

while the RMS-to-RMS operator norm is defined as

$$\|\boldsymbol{W}\|_{\text{rms}\to\text{rms}} := \sup_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{W}\boldsymbol{x}\|_{\text{rms}}}{\|\boldsymbol{x}\|_{\text{rms}}}. \tag{2}$$

$\boldsymbol{\mu}$P scale invariance requires maintaining $\|\boldsymbol{y}\|_{\text{rms}} = \|\boldsymbol{x}\|_{\text{rms}} = \Theta(1)$, which is equivalent to enforcing the RMS-to-RMS condition $\|\boldsymbol{W}\|_{\text{rms}\to\text{rms}} = \|\boldsymbol{W}\|_2\sqrt{d_{\text{in}}/d_{\text{out}}} = \Theta(1)$. This induces the following spectral norm constraint on the weight matrix $\|\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{\text{out}}/d_{\text{in}}})$. A similar requirement applies to parameter updates; together, we refer to these as the spectral $\boldsymbol{\mu}$P condition below.

Spectral μP Condition

Yang et al. [2023] shows that preserving *scale-invariant activations* for feature learning requires the same law to hold for both weights and their updates:

$$\|\boldsymbol{W}\|_2 = \Theta\left(\sqrt{\frac{d_{\text{out}}}{d_{\text{int}}}}\right), \qquad \|\boldsymbol{\Phi}\|_2 = \Theta\left(\sqrt{\frac{d_{\text{out}}}{d_{\text{int}}}}\right).$$

2.2 Steepest Descent under Different Norms

Following metrized deep learning [Bernstein and Newhouse, 2024], many successful optimizers (e.g. AdamW [Loshchilov and Hutter, 2019], Shampoo [Gupta et al., 2018], Prodigy [Mishchenko and Defazio, 2024]) can be interpreted as **first-order** methods without convexity assumptions. Specifically, after switching off exponential moving averages, these algorithms reduce to instances of steepest descent governed by distinct norms. Under this framework, an optimizer is fundamentally defined by a weight update $\boldsymbol{\Phi}$ that minimizes a quadratic model of the loss:

$$\boldsymbol{\Phi} := \operatorname*{argmin}_{\boldsymbol{\Phi}} \left\{ \mathcal{L}(\boldsymbol{W}) + \langle \boldsymbol{G}, \boldsymbol{\Phi} \rangle + \frac{\lambda}{2}\|\boldsymbol{\Phi}\|^2 \right\}. \tag{3}$$

The update is thus determined by two priors: a **norm** $\|\cdot\|$ assigned according to the specific *functional role* of the module, and a **sharpness** λ that governs the update scale. The solution is as follows:

Definition 1: Steepest Descent Update

Given a norm $\|\cdot\|$ that endows the parameter space with a geometry, the **steepest descent update** is

$$\boldsymbol{\Phi} = -\eta \cdot \boldsymbol{\Phi}_{\text{unit}}, \quad \text{where} \quad \boldsymbol{\Phi}_{\text{unit}} := \operatorname*{arg\,max}_{\|\boldsymbol{T}\|\leq 1} \langle \boldsymbol{G}, \boldsymbol{T} \rangle, \quad \text{and} \quad \eta := \frac{\|\boldsymbol{G}\|_\dagger}{\lambda}. \tag{4}$$

Here, $\|\boldsymbol{G}\|_\dagger := \max_{\|\boldsymbol{T}\|\leq 1} \langle \boldsymbol{G}, \boldsymbol{T} \rangle$ is the **dual norm** of the gradient $\boldsymbol{G}$ induced by $\|\cdot\|$.

This perspective unifies diverse algorithms by mapping them to their underlying geometries: SGD corresponds to Frobenius norm $\|\cdot\|_F$, Adam to l_∞ norm, and Shampoo to spectral norm $\|\cdot\|_2$.

2.3 Muon Optimizer

Following the framework of metrized deep learning (Section 2.2), Muon [Jordan et al., 2024] can be interpreted as steepest descent under the **spectral norm**. For a 2D weight matrix $\boldsymbol{W}$, the spectral norm $\|\cdot\|_2$ [1] gives the tightest bound on the matrix's input-output gain:

$$\|\boldsymbol{W}\|_2 := \sup_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{W}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}, \tag{5}$$

[1]For vectors, $\|\boldsymbol{v}\|_2$ denotes the ℓ_2 norm; for matrices, $\|\boldsymbol{W}\|_2$ denotes the (induced $\ell_2 \to \ell_2$) spectral norm.

By choosing the spectral norm to constrain the update direction $\mathbf{\Phi}$, the steepest descent direction is uniquely given by the **matrix sign function (msign)** (Appendix A.1):

$$\operatorname{msign}(\boldsymbol{G}) = \boldsymbol{U}\operatorname{sign}(\boldsymbol{\Sigma})\boldsymbol{V}^\top = \boldsymbol{U}_{[:,:r]}\boldsymbol{V}_{[:,:r]}^\top, \tag{6}$$

where $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ is the singular value decomposition (SVD). The msign operation orthogonalizes the gradient, equalizing all active singular directions and yielding a spectrum isotropic update. A key contribution of Muon is the efficient approximation of $\operatorname{msign}(\boldsymbol{G})$ via Newton–Schulz iterations which can be implemented on GPUs.

However, Muon constrains only the backward update $\mathbf{\Phi}$, leaving the forward weights $\boldsymbol{W}$ unconstrained. This often leads to unstable $\boldsymbol{\mu}$P feature learning in hidden states rms, motivating our development of **a fully aligned approach that constrains both $\boldsymbol{W}$ and $\mathbf{\Phi}$**.

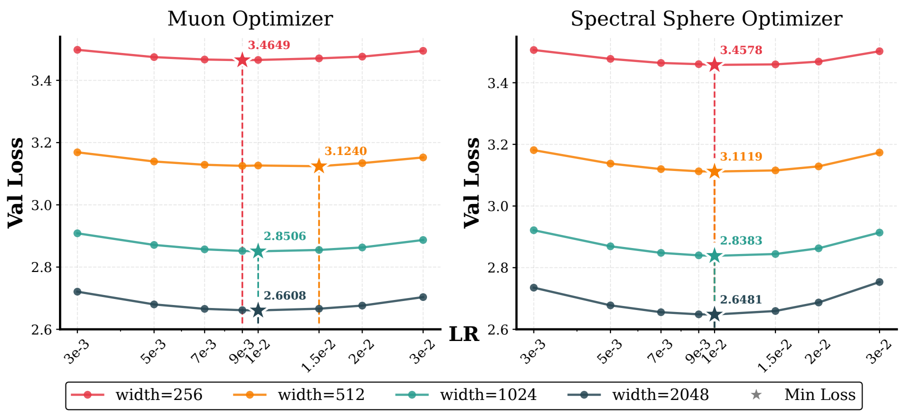


Figure 2: **$\boldsymbol{\mu}$P width scaling across 25× model size (70M to 1.8B).** Although $\boldsymbol{\mu}$P aims for width-invariant scaling, Muon still exhibits notable optimal learning-rate drift. In contrast, our Spectral Sphere achieves stable LR transfer, while also obtaining **lower optimal loss than** Muon. More details and related experiments are provided in Appendix A.5.

3 Method

3.1 Optimization Target Formulation

We start by focusing on a hidden layer matrix $\boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. To satisfy the spectral $\boldsymbol{\mu}$P scaling condition in Section 2.1, we set the spectral scale to target radius R:

$$R = \Theta\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right). \tag{7}$$

Following metrized deep learning (Section 2.2), we define the unit update $\mathbf{\Phi}$ as the solution to a constrained optimization problem:

Formulation: Steepest Descent on the Spectral Sphere

We perform **steepest descent** under the **spectral norm**, constraining both the **weights** and the **updates** to a spectral sphere of radius R. Specifically, we parameterize the update step as $\Delta\boldsymbol{W} = \eta R\mathbf{\Phi}$, where η is the base learning rate. The update direction $\mathbf{\Phi}$ is the solution to:

$$\begin{aligned} \max_{\mathbf{\Phi}} \quad & \langle \boldsymbol{G}, \mathbf{\Phi} \rangle \\ \text{s.t.} \quad & \|\mathbf{\Phi}\|_2 = 1, \\ & \|\boldsymbol{W} - \eta R\mathbf{\Phi}\|_2 = \|\boldsymbol{W}\|_2 = R. \end{aligned} \tag{8}$$

3.2 First-Order Tangent Space Constraint

Assisted by the uniqueness of the top singular value[2], the spectral norm $\|\boldsymbol{W}\|_2$ is differentiable with gradient $\boldsymbol{\Theta} := \nabla_{\boldsymbol{W}}\|\boldsymbol{W}\|_2 = \boldsymbol{u}_1\boldsymbol{v}_1^\top$, where $(\boldsymbol{u}_1, \boldsymbol{v}_1)$ are the principal left and right singular vectors [Watson, 1992]. We consider a first-order Taylor Expansion of the spectral norm around $\boldsymbol{W}$:

$$\|\boldsymbol{W} - \eta R\boldsymbol{\Phi}\|_2 = \|\boldsymbol{W}\|_2 - \eta R\langle\boldsymbol{\Theta}, \boldsymbol{\Phi}\rangle + \mathcal{O}(\eta^2 R^2 \|\boldsymbol{\Phi}\|_2^2). \tag{9}$$

To enforce the invariance condition $\|\boldsymbol{W} - \eta R\boldsymbol{\Phi}\|_2 = \|\boldsymbol{W}\|_2$, the first-order term must vanish, which implies the tangent constraint: $\langle\boldsymbol{\Theta}, \boldsymbol{\Phi}\rangle = 0$. Equation (8) thus reduces to

$$\max_{\boldsymbol{\Phi}} \langle\boldsymbol{G}, \boldsymbol{\Phi}\rangle \quad \text{s.t.} \quad \|\boldsymbol{\Phi}\|_2 = 1, \ \langle\boldsymbol{\Theta}, \boldsymbol{\Phi}\rangle = 0. \tag{10}$$

We then introduce a *Lagrange multiplier* λ and maximize *Lagrangian* $\mathcal{L}(\boldsymbol{\Phi}, \lambda) = \langle\boldsymbol{G} + \lambda\boldsymbol{\Theta}, \boldsymbol{\Phi}\rangle$ under constraint $\|\boldsymbol{\Phi}\|_2 = 1$. The analytical solution and numerical method are summarized below.

Solution: Tangent Space Update via λ-Search

For a fixed λ, the steepest descent direction is (proof in Appendix A.1):

$$\boldsymbol{\Phi}^\star(\lambda) = \operatorname{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta}), \tag{11}$$

where $\lambda^\star$ is the unique root of the constraint function:

$$h(\lambda^\star) := \langle\boldsymbol{\Theta}, \operatorname{msign}(\boldsymbol{G} + \lambda^\star\boldsymbol{\Theta})\rangle = 0. \tag{12}$$

Theoretical Properties (proof in Appendix A.2, visualized in Figure 3):

i) **Monotonicity:** The function $h(\lambda)$ is monotonically non-decreasing and transitions from -1 to $+1$ as λ varies from $-\infty$ to $+\infty$.

ii) **Root Localization:** The solution $\lambda^\star$ is strictly bounded within the interval $[-2\|\boldsymbol{G}\|_*, 2\|\boldsymbol{G}\|_*]$, providing a finite search space [a].

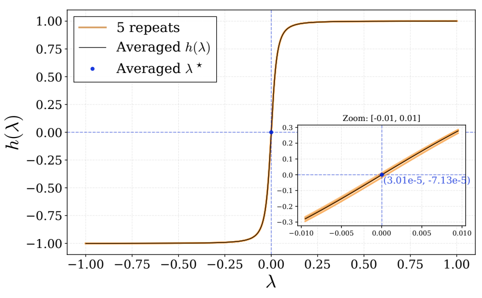
Numerical Algorithm (overhead analysis in Section 5):

Given the monotonic nature of $h(\lambda)$, we locate $\lambda^\star$ efficiently:

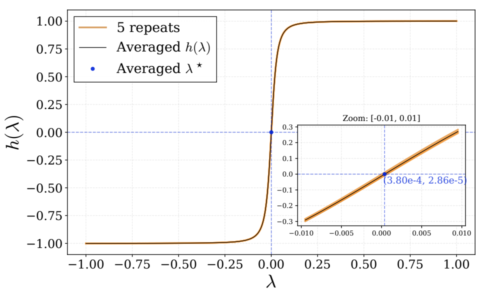
▷ **Bracketing:** Leveraging monotonicity, we start from $\lambda = 0$ and exponentially expand the search bracket in the opposite direction of the sign of $h(0)$ until the root is enclosed.

▷ **Bisection:** We isolate $\lambda^\star$ via standard bisection within the bracketed interval.

[a]Nuclear norm $\|\cdot\|_*$ is the sum of singular values.



(a) 1024×3072 matrix



(b) 4096×1024 matrix

Figure 3: **Empirical curves of $h(\lambda) = \langle\boldsymbol{\Theta}, \operatorname{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})\rangle$ for random matrices**. $h(\lambda)$ is monotonic non-decreasing in λ, and its root $\lambda^\star$ lies close to zero (proof in Appendix A.2). Here, each curve is obtained by averaging over 5 repeats, and each matrix is initialized from $\mathcal{N}(0, 0.02^2)$.

[2]Numerical coincidence of singular values is a measure-zero event for random matrices [Tao and Vu, 2014]. Quantitatively, it occurs with probability at most $\exp(-c\max(d_{\text{in}}, d_{\text{out}}))$ [Han, 2025].

3.3 Second-Order Manifold Constraint

Note that the remainder $\mathcal{O}(\eta^2 R^2 \|\mathbf{\Phi}\|_2^2)$ in Equation (9) may accumulate over iterations, causing gradual drift off the spectral sphere. To enforce the exact constraint $\|\boldsymbol{W}\|_2 = R$ throughout training, we apply a retraction step that projects the weights back onto the manifold:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} \cdot \frac{R}{\|\boldsymbol{W}\|_2}. \tag{13}$$

While the retraction is conceptually a post-update projection, we implement it as a pre-update correction for efficiency, which is operationally equivalent. This reordering allows us to invoke the computationally expensive Power Iteration only once per step, reusing the resulting singular triplet for both the manifold retraction and the tangent projector $\mathbf{\Theta}$ (Lines 6–9 in Algorithm 1).

The retraction strictly constrains $\|\boldsymbol{W}\|_2 = R$, which via Equation (14), automatically bounds the weight magnitudes. As a result, weight decay, which is primarily introduced to limit the weight scale, becomes redundant. We therefore **eliminate weight decay** in hidden 2D weights[3], removing a sensitive hyperparameter from training. More details can be found in Appendix A.3.

$$\|\boldsymbol{W}\|_F \le \sqrt{rank(\boldsymbol{W})}\, \|\boldsymbol{W}\|_2 \le \sqrt{\min(d_{\text{out}}, d_{\text{in}})}\, R. \tag{14}$$

3.4 Overall Algorithm & Interpretation

Algorithm 1 Spectral Sphere Optimizer (SSO)

```
Require: Initial 2D weights $\boldsymbol{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, spectral $\mu$P scaler $R = \sqrt{d_{\text{out}}/d_{\text{in}}}$, learning rate $\eta$,
    momentum coefficient $\beta$, precision tolerance $\epsilon$
 1: Initialize: $\boldsymbol{W}_0 \leftarrow R \cdot \boldsymbol{W}_0/\|\boldsymbol{W}_0\|_2$, $\boldsymbol{M}_0 \leftarrow 0$
 2: for $t = 0, 1, \ldots$ do
 3:     $\boldsymbol{G}_t \leftarrow \nabla_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}_t)$
 4:     $\boldsymbol{M}_t \leftarrow \beta \boldsymbol{M}_t + (1-\beta)\boldsymbol{G}_t$
 5:     $\widehat{\boldsymbol{M}}_t \leftarrow \boldsymbol{M}_t / \|\boldsymbol{M}_t\|_F$                    ▷ Normalize for Numerical Stability
    // 1. Spectral Geometry Analysis
 6:     $(\sigma_t, \boldsymbol{u}_t, \boldsymbol{v}_t) \leftarrow \text{PowerIteration}(\boldsymbol{W}_t)$          ▷ Top Singular Value & Vectors
 7:     $\mathbf{\Theta}_t \leftarrow \boldsymbol{u}_t \boldsymbol{v}_t^\top$                                    ▷ Tangent Space Projector
    // 2. Retraction to Spectral Sphere
 8:     $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_t \cdot R/\sigma_t$
    // 3. Steepest Descent Lagrange Solver
 9:     Define $h(\lambda) := \langle \mathbf{\Theta}_t, \text{msign}(\widehat{\boldsymbol{M}}_t + \lambda \mathbf{\Theta}_t) \rangle$
10:     $\lambda_t^* \leftarrow \text{Bisection}(h, \text{tolerance} = \epsilon)$                ▷ Find root of $h(\lambda) = 0$
    // 4. $\mu$P-Scaled Update
11:     $\mathbf{\Phi}_t \leftarrow \text{msign}(\widehat{\boldsymbol{M}}_t + \lambda_t^* \mathbf{\Theta}_t)$
12:     $\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t - \eta \cdot R \cdot \mathbf{\Phi}_t$                       ▷ $\mu$P Style Update
13: end for
```

In this paper, we set AdamW and Muon as baselines. Additionally, we introduce MuonSphere, a variant similar to Scion [Pethick et al., 2025][4], which can be viewed as Spectral Sphere with $\lambda = 0$. While Spectral Sphere follows a steepest-descent approach, MuonSphere simply normalizes 2D hidden weights onto the spectral sphere before each update. Figure 4 provides a sketch of the latter three spectral-preconditioned optimizers' update.

[3]We still apply weight decay to 1D params (e.g. `Embedding`, `RMSNorm`) for possible scaling stability, although our ablations on 1.7B model suggest that disabling 1D params may actually be slightly better.

[4]Unlike Scion,which applies ColNorm→Spectral→Sign (l_∞) norm chain throughout the network, MuonSphere retains Sign→Spectral→Sign norm scheme. We found ColNorm input hurts performance.

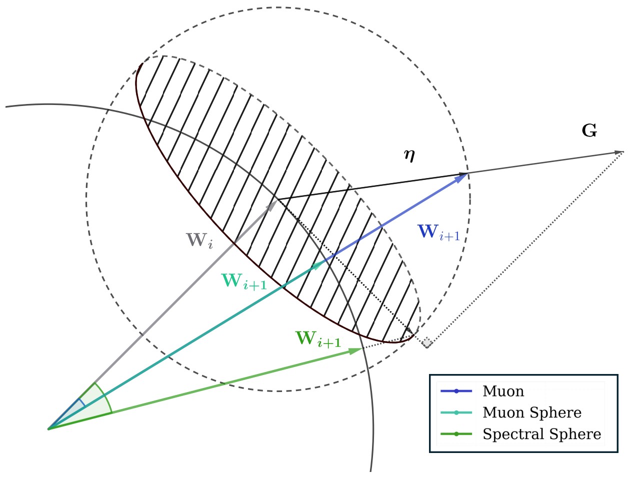


Figure 4: **Geometry of Steepest Descent Update Directions.** The left solid arc denotes the $\boldsymbol{W}$ sphere, while the right dotted arc denotes the $\Delta\boldsymbol{W}$ sphere (unit $\boldsymbol{\Phi}$ scaled by η). The shaded region represents the feasible set within the *tangent space* of the $\boldsymbol{W}$ sphere at step $\boldsymbol{W}_i$. Under weight constraint, projecting $\boldsymbol{G}$ onto the tangent space (Spectral Sphere) yields the largest update angle.

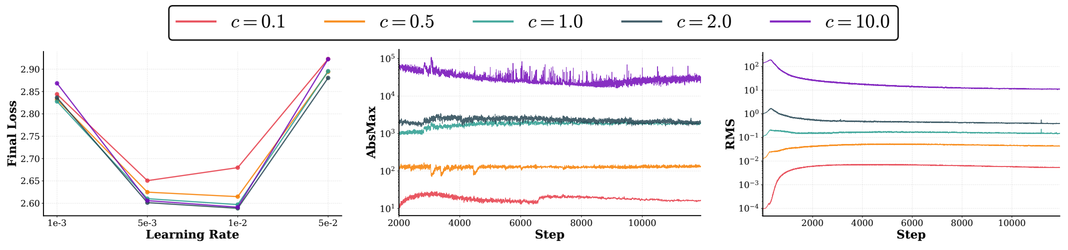
4 Algorithm Details

4.1 Spectral Radius Scale

Given the target spectral radius

$$R = \Theta\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right) = c\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right)$$

the constant c serves as a scalar that sets the branch output magnitude relative to the residual stream. By tuning c, one can precisely control the signal-to-noise ratio along the deep residual path, balancing the contributions of the Attention/FFN blocks against the skip connection. Properly choosing c is therefore essential for stabilizing depth-wise signal propagation in Transformers.



(a) Radius scale search. (b) `AbsMax` of FFN activations. (c) `RMS` of FFN activations.

Figure 5: **Ablation of radius scaling on *optimization* and *activation*.** (a) Final loss vs. learning rate for varying radius scales c. A moderate scale (e.g. $c = 2.0$) achieves the best performance. (b) AbsMax and (c) RMS of FFN activations during training. AbsMax monotonically follows the radius scale, whereas RMS follows a clear power-law scaling with c.

4.2 Learning Rate Scaler

We propose a unified view in which each learning-rate scaler enforces a consistent **effective step size** under a chosen **norm metric** and **initialization scheme**.

In the update rule $\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta R \boldsymbol{\Phi}$, R scales the update size. To avoid instability caused by heterogeneous layer shapes, we select R to maintain a constant *effective step size* — defined as the ratio of update-to-weight magnitude under a norm metric $\|\cdot\|$:

$$\frac{\|\Delta \boldsymbol{W}\|}{\|\boldsymbol{W}\|} = \frac{\|\eta R \boldsymbol{\Phi}\|}{\|\boldsymbol{W}\|} \approx \eta. \tag{15}$$

We evaluate three common learning rate scalers below:

$$R = \begin{cases} \sqrt{d_{\text{out}}/d_{\text{in}}}, & \textit{(Spectral } \boldsymbol{\mu}\textit{P Scaler)} \\ \sqrt{\max(d_{\text{out}}, d_{\text{in}})} \cdot 0.2, & \textit{(Align-Adam-RMS Scaler)} \\ \sqrt{\max(1, d_{\text{out}}/d_{\text{in}})}, & \textit{(Spectral Kaiming Scaler)} \end{cases} \tag{16}$$

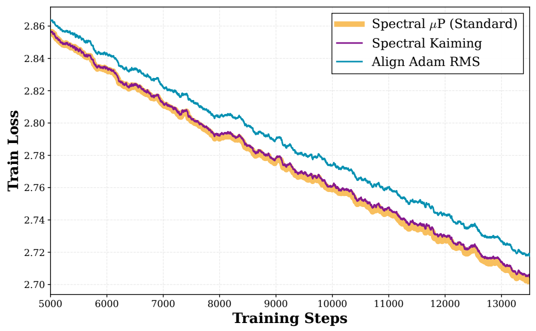


Figure 6: **Ablation of learning rate scalers.** Each curve represents the optimal validation loss obtained via a learning rate grid search. **Spectral $\boldsymbol{\mu}$P** (yellow) outperforms **Align-Adam-RMS** (blue), validating the optimality of $\boldsymbol{\mu}$P-aligned scaling under the Spectral $\boldsymbol{\mu}$P condition.

- **Spectral $\boldsymbol{\mu}$P Scaler.** Enforces the *RMS-to-RMS operator norm* invariance from Section 2.1. It ensures that both $\boldsymbol{W}$ and $\Delta\boldsymbol{W}$ satisfy the $\boldsymbol{\mu}$P scaling $\|\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{\text{out}}/d_{\text{in}}})$, forming the geometric basis of our Spectral Sphere Optimizer.
- **Align-Adam-RMS Scaler.** A heuristic for consistent relative learning rates in the *RMS norm* under fixed standard-deviation initialization. Empirically, it aligns per-layer update RMSnorm to AdamW, enabling direct transfer of AdamW-tuned hyperparameters (e.g. learning rate, weight decay) to the spectral method [Liu et al., 2025].
- **Spectral Kaiming Scaler.** Targets the *spectral norm* under Kaiming initialization ($\boldsymbol{W} \sim \mathcal{N}(0, 1/d_{\text{in}})$) [He et al., 2015]. Random matrix theory establishes that such matrices satisfy $\|\boldsymbol{W}\|_2 \approx 1 + \sqrt{d_{\text{out}}/d_{\text{in}}}$. This scaling prevents vanishing pre-activations in bottleneck layers where $d_{\text{out}} \ll d_{\text{in}}$ [Pethick et al., 2025].

Experimental results (Figure 6) favor the **Spectral $\boldsymbol{\mu}$P** scaler, supporting the theoretical scaling condition in Section 2.1. This demonstrates that *optimizing on a spectral manifold requires a scaler explicitly calibrated to the spectral norm.*

4.3 Module Granularity

To maximize computational efficiency, modern Transformer implementations such as Megatron-LM [Shoeybi et al., 2019] fuse several matrices into a single physical tensor (e.g. the QKV projections or the SwiGLU gate/up matrices). However, since these modules have distinct functional roles, imposing a unified constraint on the fused tensor is suboptimal as shown in Figure 7. Instead, we decompose fused tensors and treat each submatrix as an independent module. We apply **per module spectral initialization and optimization** at this finer granularity by default (e.g. splitting attention QKV per head and separating FFN gate/up). Note that granularity is tunable depending on infra speed.

Under our spectral $\boldsymbol{\mu}$P initialization scheme (also discussed in Section 4.2), the weight matrix is constructed by first sampling $\boldsymbol{W}_k \sim \mathcal{N}(0, \sigma^2)$ and subsequently projecting it onto the spectral sphere:

$$\boldsymbol{W} = c\sqrt{\frac{d_{\text{out}}}{d_{\text{in}}}} \cdot \frac{\boldsymbol{W}_k}{\|\boldsymbol{W}_k\|_2},$$

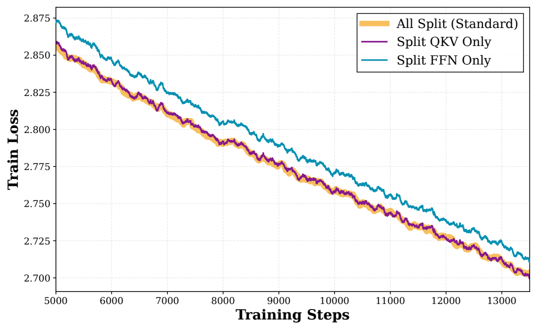


Figure 7: **Ablation of module granularity for *initialization* and *optimization*.** Splitting QKV per head alone yields the most significant performance gain. Although splitting the FFN gate/up weights produces nearly the same loss curve as no-split, we maintain this split to respect their distinct functional roles; the overlapping curve is omitted from the figure.

5 Infrastructure Design

5.1 Bottleneck Analysis

The main challenge in **SSO** implementation comes from the **bracket-and-bisect** root solver that runs at every update. To satisfy the tangent-space constraint, we must find a Lagrange multiplier λ such that $h(\lambda) = 0$. We first expand the search interval by **bracketing**, and then we run **bisection** inside the bracket to obtain a $\lambda^\star$ that meets a target tolerance. This solver introduces non-trivial overhead:

1. **Workload Imbalance:** different bracketing range and tolerance settings can change the number of search and bisection steps, leading to unstable runtime and workload imbalance between devices.
2. **Computational Cost:** Each step in the search also evaluates $h(\lambda)$, which calls $\mathrm{msign}(\widehat{\boldsymbol{M}} + \lambda \boldsymbol{\Theta})$; these extra matrix computations accumulate and add noticeable cost to every optimizer step.
3. **Synchronization Overhead:** Iterative search creates frequent synchronization between the GPU and the CPU, because the algorithm must finish each evaluation and then check a scalar condition before choosing the next λ and continuing.

5.2 Optimization Pipeline

We introduce a holistic optimization pipeline designed to mitigate these overheads while preserving numerical precision. **All performance statistics are collected from Dense 1.7B model pretraining.**

Atomic Module Sharding. We employ a fine-grained, parameter-wise sharding strategy [Nvidia, 2025]. As noted by [Liu et al., 2025], while standard ZeRO-1 is efficient for element-wise optimizers (e.g. AdamW), its flat-buffer sharding approach is incompatible with spectral methods that require full gradient matrices to compute updates. To reconcile this, we shard parameters as *atomic modules* rather than flattened buffers. An atomic module is defined as the minimal independent weight matrix required to remain intact for spectral operations (see Section 4.3).

Load Balancing Strategy. To resolve the workload imbalance caused by variable solver depths, we employ a "ping-pong" load balancing strategy adapted from [Nvidia, 2025]. Empirical results indicate this method outperforms both greedy size-descent sorting and default round-robin allocation. We sort atomic modules by size and assign them to DP ranks in an alternating zigzag pattern. This heuristic effectively balancing the solver workload without complex runtime scheduling. Finally, we synchronize the updated params using iterative All-Gather collective from Nvidia [2025].

Adaptive Kernel Selection. We observe that the optimal implementation for Matrix Sign computation is highly sensitive to matrix dimensions. As shown in Table 1, applying specialized kernels indiscriminately can degrade performance. We therefore implement an Adaptive Dispatcher:

- **Small Matrices** (< 512): We use a JIT-compiled PyTorch implementation built on `torch.addmm`. This avoids the launch overhead of specialized kernels.
- **Large Matrices** (≥ 512): We dispatch to a custom Triton kernel implementing the SYmmetric Rank-K (SYRK) Nvidia [2025] update, which exploits the symmetry of Newton–Schulz iterations to halve memory reads.

Multi-Stream Parallelism. For layers composed of many small independent matrices (e.g. per head attention split), single-stream execution suffers from kernel launch latency bubbles. We exploit this independence by dispatching spectral updates across multiple CUDA streams.

Mixed-Precision. The Power Iteration for spectral norm estimation is performed in BFloat16, while the sensitive **msign remains in FP32 with 8 iterations**.

Cache Top Singular Vectors. The singular vectors of model weights evolve slowly during training. Leveraging this temporal locality, we initialize the current Power Iteration using the cached singular vectors u and v from the previous step. This reuse mechanism drastically accelerates convergence, requiring only a few iterations to maintain approximation accuracy.

Table 1: Optimization breakdown on end-to-end latency for 4M tokens/step on NVIDIA B200. ↓ denotes improvement, while ↑ denotes regression. Note there is still room for improvement.

	Time (ms)	Δ vs Baseline	Δ vs Prev.
Naive Baseline (No opt.)	10928.5	-	-
+ Load balance & All Gather	9365.5	-1563.0 (-14.3%) ↓	-1563.0 (-14.3%) ↓
+ Triton SYRK Kernel	10284.2	-644.3 (-5.9%) ↓	+918.7 (+9.8%) ↑
+ Adaptive & Multi-stream	9383.4	-1545.1 (-14.2%) ↓	-900.8 (-8.8%) ↓
+ BF16 & Torch.compile (**Final**)	**7666.3**	**-3262.2 (-29.9%)** ↓	**-1717.1 (-18.3%)** ↓

Table 2: End-to-end per-step latency for 4M tokens/step on NVIDIA B200. Muon as baseline.

	AdamW	Muon	MuonSphere	Spectral Sphere
Time (ms)	6734.15 (-2.10%)	6878.83	6949.85 (+1.03%)	7666.32 (+11.45%)

5.3 Future Improvements

- **GPU-Native Solver.** Profiling indicates that the current CPU-based bisection solver introduces latency due to frequent device-host synchronization. Although the bracketing phase converges rapidly (< 2 steps), the bisection phase averages 5–7 steps, creating synchronization bubbles. Future work will prioritize implementing a pure GPU-native solver to eliminate these overheads. Additionally, we plan to adopt higher convergence algorithms, such as Brent's method or n-section search, and optimize initial bracketing intervals to minimize msign calls.
- **Kernel Optimization.** The theoretical advantage of SSO relies on the accuracy of the tangent space projection; when errors are high, the update direction may degrade to the "worst-case" Muon update. Currently, we ensure precision via 8 iterations of msign in FP32. To follow standard Muon practices, we plan to use msign in BFloat16 within 5 steps. Furthermore, we intend to replace current JIT-compiled operations with custom, fully optimized kernels (e.g. for batched msign and Power Iterations) to better exploit the hardware features of next-generation GPUs.
- **Low-precision Training.** While weight matrices are spectral constrained, we found the residual stream remains the primary source of outliers. We aim to explore "fully manifold constrained" architectures, i.e. using mHC [Xie et al., 2025]. Additionally, given SSO's demonstrated stability, we plan to explore low-precision training (e.g. FP8/NVFP4) to leverage the high throughput of low-bit arithmetic for better training efficiency.

6 Scaling Experiments

6.1 Experimental Setup

Hyperparameters. Following the μP protocol, we perform a learning rate sweep on the 1.7B model with AdamW between $[10^{-3}, 10^{-2}]$ and found 5×10^{-3} to be the optimal value. Training uses 500 warmup steps, a global batch size of $\sim$4M tokens (1024 sequences $\times$ 4096 tokens each), and a cosine learning rate decay reduced to 10% peak. We use BF16 mixed-precision training.

Following Kimi Moonlight [Liu et al., 2025], all optimizers use a weight decay of 0.1. However, distinct from their approach of aligning updates to AdamW RMS (intended to reuse current scaling laws), we find that spectral μP LR scaler outperforms the uniform 0.2 update-rms alignment (Section 4.2). For msign coefficients, we use the Polar Express method [Amsel et al., 2025] with 8 Newton–Schulz iterations[5]. We employ Nesterov momentum and, by default, split attention heads and FFN gate/up projections for separate initialization and optimization. For Spectral Sphere, we set the λ-solver's maximum iterations to 20, Lagrange solver precision tolerance to 2e-4, and remove weight decay for all hidden 2D weights, as retraction maintains the weight constraint (Section 3.3).

Training Data We use the OLMo-Mix-1124 dataset [Team OLMo et al., 2025], tokenized with the OLMo-2 tokenizer. We randomly sample 100 billion tokens for training and reserve 1 billion tokens for validation. Data indices are built offline to guarantee a deterministic training order.

Benchmarks. We primarily evaluate on the following downstream tasks: ARC [Clark et al., 2018], BoolQ [Clark et al., 2019], CSQA [Talmor et al., 2019], HellaSwag [Zellers et al., 2019], PIQA [Bisk et al., 2020], WinoGrande [Sakaguchi et al., 2021] and LAMBADA [Paperno et al., 2016].

6.2 Dense 1.7B

We adopt the architecture configuration of Qwen3-1.7B [Qwen-Team et al., 2025], replacing its original tokenizer with that of OLMo-2 [Team OLMo et al., 2025]. The core architecture utilizes Grouped Query Attention (GQA), QK-Norm, SwiGLU activations, Rotary Positional Embeddings (RoPE), and pre-normalization RMSNorm. We do not tie the embedding and the head.

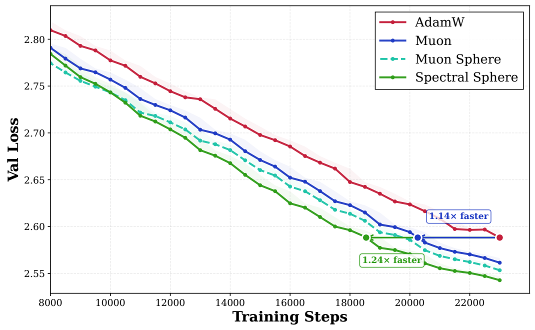


Figure 8: **Validation loss of training dense 1.7B model on 100B tokens.** As a reference point, AdamW attains a final validation loss of 2.588 at 23k steps. The overall setup favors AdamW, since the learning rate is set for it (5e-3), rather than the higher optimal rate (1e-2) for Muon and Spectral Sphere. Even under this setting, spectral-based optimizers exhibit higher efficiency: Muon reaches the same validation loss level in 12% fewer steps, while Spectral Sphere does so in 19% fewer steps.

6.3 MoE 8B-A1B

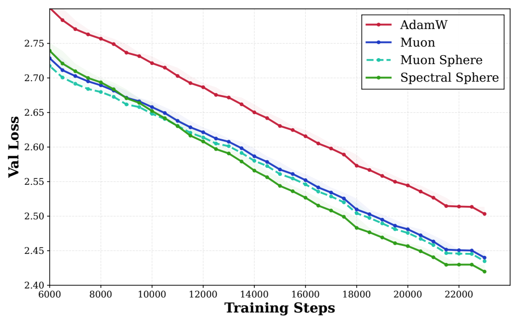
The configuration largely follows DeepSeek-V3 [DeepSeek-AI et al., 2025]. The model has 27 layers: the first is a standard dense FFN, followed by 26 MoE layers. We utilize 64 experts in total, with a top-4 routing expert plus 1 shared expert.

[5]We tested 5 and 8 iterations as well as different msign coefficients and observed negligible differences in training loss (< 1e-3). We retain 8 iterations for improved numerical accuracy.

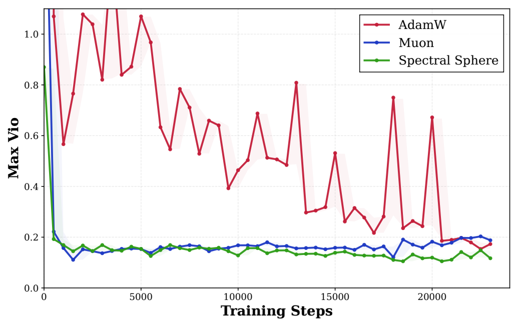
Table 3: Performance comparison of different optimizers on Dense 1.7B models.

Optimizer	LMB. (PPL) ↓	LMB. (Acc) ↑	CSQA (Acc) ↑	PIQA (Acc) ↑	Hella. (Acc) ↑	Wino. (Acc) ↑	ARC-e (Acc) ↑	ARC-c (Acc) ↑	BoolQ (Acc) ↑	Avg. (Acc) ↑
AdamW	5.40	63.71	19.66	74.70	47.90	62.59	68.81	37.37	63.24	54.75
Muon	5.05	65.19	19.00	75.35	48.91	61.72	70.24	37.46	64.22	55.26
MuonSphere	**4.87**	**65.55**	20.07	74.97	49.20	62.83	71.51	**38.40**	**66.97**	56.19
Spectral Sphere	5.00	65.07	**21.05**	**75.95**	**49.25**	**63.77**	**71.80**	38.31	65.57	**56.35**

Router and Load Balancing. The router is implemented in FP32 precision. We adopt the auxiliary-loss-free strategy [Wang et al., 2024], which demonstrated superior expert loading balance compared to global auxiliary loss [Qiu et al., 2025] in our ablations. We enable expert bias with an update rate of 0.001. The sequence-level auxiliary loss is removed, as we found it redundant when expert bias is enabled. we use a sigmoid gate with top-k scores renormalized and scaled by 2. (see Appendix A.4). To evaluate router load balancing, we employ Maximal Violation (MaxVio) [Wang et al., 2024], where 0 indicates perfect balance. As shown in Figure 9, weight spectral normalization effectively promotes balanced routing, leading to superior validation loss.



(a) Validation Loss across four optimizers.



(b) Max Vio as a MoE load-balance metric.

Figure 9: **MoE 8B-A1B training.** Spectral Sphere achieves the lowest validation loss while maintaining the best expert load balance. In contrast, AdamW exhibits substantially larger MaxVio with frequent spikes, indicating unstable routing and poorer utilization of effective model capacity. Compared to Muon, constraining each expert on the spectral sphere further improves load balance.

6.4 DeepNet 200-Layer

To evaluate the stability of different optimizers under extreme depth, we extended the baseline's 28 layers to 200 layers. This deep and narrow serves as a stress test for stability. The training loss in Figure 10 shows that Spectral Sphere outperforms baselines with lower loss and higher stability.

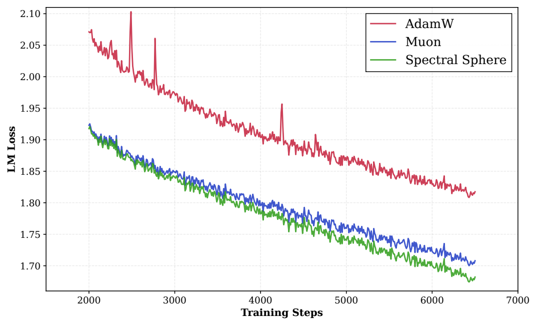


Figure 10: **Deepnet 200 layers training loss.** AdamW shows pronounced instability, characterized by frequent loss spikes and a growing gap in performance relative to spectral-based optimizers. Spectral Sphere attains both the lowest loss and the highest stability.

7 Discussion

In this work, we propose a novel perspective on optimizer design grounded in spectral μP principles. As detailed in Section 2.1, μP provides a mathematical safeguard that strictly controls hidden layer activations at the desired scale. By identifying the spectral sphere as the natural geometry for stable feature learning, we derive the Spectral Sphere Optimizer (SSO) — the unique solution for steepest descent constrained within both weight and update manifolds (Section 3). This formulation effectively achieves rapid convergence grounded in fundamental training stability. Empirically, SSO consistently outperforms AdamW and Muon, while uniquely preserving stable μP learning rate transfer. Notably, it yields superior training dynamics: it significantly improves MoE router load balancing, suppresses outliers in deep networks, and strictly bounds activations within a tunable scale.

A critical distinction exists between our approach and emerging works on Stiefel manifold optimization [Bernstein, 2025]: while Stiefel manifold requires *all* singular values to be exactly 1, SSO constrains only the *maximal* singular value. This relaxation allows the internal spectrum to evolve freely below the bound, avoiding the overly rigid isotropy of the Stiefel manifold.

Beyond the theoretical contribution, we provide a complete and systematic recipe (Section 4) implemented in Megatron-LM. Our guidelines on atomic granularity, "ping-pong" load balancing strategy, learning rate scaler, and spectral radius scale serve as a robust empirical practice for the broader class of spectral optimizers. Finally, while we acknowledge that the current root solver introduces non-trivial latency, we have outlined concrete pathways to mitigate this overhead in Section 5.3. For scenarios prioritizing infra cost, we recommend MuonSphere — a variant that retains equivalent activation control with minimal overhead.

Acknowledgements

We sincerely thank Jianlin Su for his excellent blog, Scientific Spaces.

References

Noah Amsel, David Persson, Christopher Musco, and Robert M. Gower. The polar express: Optimal matrix sign methods and their application to the muon algorithm, 2025. URL https://arxiv.org/abs/2505.16932.

Jeremy Bernstein. Modular manifolds. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20250926. https://thinkingmachines.ai/blog/modular-manifolds/.

Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology, 2024. URL https://arxiv.org/abs/2409.20325.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*, pages 7432–7439, 2020. doi: 10.1609/AAAI.V34I05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, 2019. doi: 10.18653/v1/N19-1300.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. Cogview: Mastering text-to-image generation via transformers, 2021. URL https://arxiv.org/abs/2105.13290.

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

Yi Han. Repeated singular values of a random symmetric matrix and decoupled singular value estimates, 2025. URL https://arxiv.org/abs/2504.15992.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

Alex Henry, Prudhvi Raj Dachapally, Shubham Pawar, and Yuxuan Chen. Query-key normalization for transformers, 2020. URL https://arxiv.org/abs/2010.04245.

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

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, Chenzhuang Du, Dikang Du, Yulun Du, Yu Fan, Yichen Feng, Kelin Fu, Bofei Gao, Hongcheng Gao, Peizhong Gao, Tong Gao, Xinran Gu, Longyu Guan, Haiqing Guo, Jianhang Guo, Hao Hu, Xiaoru Hao, Tianhong He, Weiran He, Wenyang He, Chao Hong, Yangyang Hu, Zhenxing Hu, Weixiao Huang, Zhiqi Huang, Zihao Huang, Tao Jiang, Zhejun Jiang, Xinyi Jin, Yongsheng Kang, Guokun Lai, Cheng Li, Fang Li, Haoyang Li, Ming Li, Wentao Li, Yanhao Li, Yiwei Li, Zhaowei Li, Zheming Li, Hongzhan Lin, Xiaohan Lin, Zongyu Lin, Chengyin Liu, Chenyu Liu, Hongzhang Liu, Jingyuan Liu, Junqi Liu, Liang Liu, Shaowei Liu, T. Y. Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yibo Liu, Yiping Liu, Yue Liu, Zhengying Liu, Enzhe Lu, Lijun Lu, Shengling Ma, Xinyu Ma, Yingwei Ma, Shaoguang Mao, Jie Mei, Xin Men, Yibo Miao, Siyuan Pan, Yebo Peng, Ruoyu Qin, Bowen Qu, Zeyu Shang, Lidong Shi, Shengyuan Shi, Feifan Song, Jianlin Su, Zhengyuan Su, Xinjie Sun, Flood Sung, Heyi Tang, Jiawen Tao, Qifeng Teng, Chensi Wang, Dinglu Wang, Feng Wang, Haiming Wang, Jianzhou Wang, Jiaxing Wang, Jinhong Wang, Shengjie Wang, Shuyi Wang, Yao Wang, Yejie Wang, Yiqin Wang, Yuxin Wang, Yuzhi Wang, Zhaoji Wang, Zhengtao Wang, Zhexu Wang, Chu Wei, Qianqian Wei, Wenhao Wu, Xingzhe Wu, Yuxin Wu, Chenjun Xiao, Xiaotong Xie, Weimin Xiong, Boyu Xu, Jing Xu, Jinjing Xu, L. H. Xu, Lin Xu, Suting Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ziyao Xu, Junjie Yan, Yuzi Yan, Xiaofei Yang, Ying Yang, Zhen Yang, Zhilin Yang, Zonghan Yang, Haotian Yao, Xingcheng Yao, Wenjie Ye, Zhuorui Ye, Bohong Yin, Longhui Yu, Enming Yuan, Hongbang Yuan, Mengjie Yuan, Haobing Zhan, Dehao Zhang, Hao Zhang, Wanlu Zhang, Xiaobin Zhang, Yangkun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Haotian Zhao, Yikai Zhao, Huabin Zheng, Shaojie Zheng, Jianren Zhou, Xinyu Zhou, Zaida Zhou, Zhen Zhu, Weiyu Zhuang, and Xinxing Zu. Kimi k2: Open agentic intelligence, 2025. URL https://arxiv.org/abs/2507.20534.

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

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

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

Konstantin Mishchenko and Aaron Defazio. Prodigy: An expeditiously adaptive parameter-free learner. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=JJpOssn0uP.

Nvidia. Emerging optimizers, 2025. URL https://docs.nvidia.com/nemo/emerging-optimizers/latest/apidocs/orthogonalized-optimizers.html.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, 2016. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

Thomas Pethick, Wanyun Xie, Kimon Antonakopoulos, Zhenyu Zhu, Antonio Silveti-Falls, and Volkan Cevher. Training deep learning models with norm-constrained lmos, 2025. URL https://arxiv.org/abs/2502.07529.

Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models, 2025. URL https://arxiv.org/abs/2501.11873.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019. URL https://arxiv.org/abs/1909.08053.

Jianlin Su. A preliminary exploration of mup: Cross-model scale migration rules of hyperparameters, Mar 2025a. URL https://kexue.fm/archives/10770.

Jianlin Su. Higher-order mup: simpler but wiser spectral conditioning scaling, Mar 2025b. URL https://kexue.fm/archives/10795.

Jianlin Su. Moe travelogue: 5. reflections on uniform distribution, May 2025c. URL https://kexue.fm/archives/10945.

Jianlin Su. Steepest decent on manifolds: 4. muon + spectral sphere, Aug 2025d. URL https://kexue.fm/archives/11241.

Sho Takase, Shun Kiyono, Sosuke Kobayashi, and Jun Suzuki. Spike no more: Stabilizing the pre-training of large language models, 2025. URL https://arxiv.org/abs/2312.16903.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019.

Terence Tao and Van Vu. Random matrices have simple spectrum, 2014. URL https://arxiv.org/abs/1412.1438.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, Michal Guerquin, David Heineman, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Jake Poznanski, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali

Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts, 2024. URL https://arxiv.org/abs/2408.15664.

G Alistair Watson. Characterization of the subdifferential of some matrix norms. *Linear Algebra Appl*, 170(1):33–45, 1992.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mhc: Manifold-constrained hyper-connections, 2025. URL https://arxiv.org/abs/2512.24880.

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

Greg Yang, James B. Simon, and Jeremy Bernstein. A spectral condition for feature learning, 2023. URL https://arxiv.org/abs/2310.17813.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

A Appendix

A.1 Duality with Spectral Norm

We may first prove Theorem A.1, and we can then directly derive Equation (11).

Theorem A.1. *Nuclear norm $\|\cdot\|_*$ is the dual norm of spectral norm $\|\cdot\|_2$, which means*

$$\|\boldsymbol{G}\|_* = \max_{\|T\|_2=1} \langle \boldsymbol{G}, \boldsymbol{T} \rangle.$$

$\operatorname{msign}(\cdot)$ *is the dual map based on $\|\cdot\|_2$, which means*

$$\operatorname{msign}(\boldsymbol{G}) = \underset{\|\boldsymbol{T}\|_2=1}{\operatorname{argmax}} \langle \boldsymbol{G}, \boldsymbol{T} \rangle.$$

Proof. Since $\boldsymbol{G} \in \mathbb{R}^{n\times m}$ has Singular Vector Decomposition $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \sum_{i=1}^r \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top$ where $\boldsymbol{u}_i \in \mathbb{R}^n$ and $\boldsymbol{v}_i \in \mathbb{R}^m$ are left and right singular vectors, we have

$$\langle \boldsymbol{G}, \boldsymbol{T} \rangle = \operatorname{tr}(\boldsymbol{G}^\top \boldsymbol{T}) = \operatorname{tr}(\sum_{i=1}^r \sigma_i \boldsymbol{v}_i \boldsymbol{u}_i^\top \boldsymbol{T}) = \sum_{i=1}^r \sigma_i \boldsymbol{u}_i^\top \boldsymbol{T} \boldsymbol{v}_i. \tag{17}$$

With $\|\boldsymbol{T}\|_2 = 1$ and $\|\boldsymbol{u}_i\|_2 = \|\boldsymbol{v}_i\|_2 = 1$, we have $\boldsymbol{u}_i^\top \boldsymbol{T}\boldsymbol{v}_i \le \|\boldsymbol{T}\|_2 \|\boldsymbol{u}_i\|_2 \|\boldsymbol{v}_i\|_2 = 1$, hence

$$\langle \boldsymbol{G}, \boldsymbol{T} \rangle = \sum_{i=1}^r \sigma_i \boldsymbol{u}_i^\top \boldsymbol{T} \boldsymbol{v}_i \le \sum_{i=1}^r \sigma_i = \|\boldsymbol{G}\|_*. \tag{18}$$

Equality is attained when $\boldsymbol{u}_i^\top \boldsymbol{T}\boldsymbol{v}_i = 1$ for all $i = 1, \dots, r$, that is when

$$\boldsymbol{T} = \sum_{i=1}^r \boldsymbol{u}_i \boldsymbol{v}_i^\top = \boldsymbol{U}_{[:,:r]} \boldsymbol{V}_{[:,:r]}^\top = \operatorname{msign}(\boldsymbol{G}). \tag{19}$$

Note that for this $\boldsymbol{T}$ we indeed have $\|\boldsymbol{T}\|_2 = 1$ (its nonzero singular values are all equal to 1), so $\boldsymbol{T}$ is feasible and achieves the upper bound. Therefore, we have

$$\underset{\|\boldsymbol{T}\|_2=1}{\operatorname{argmax}} \langle \boldsymbol{G}, \boldsymbol{T} \rangle = \operatorname{msign}(\boldsymbol{G}), \tag{20}$$

and according to Equation (18), the maximum value equals $\|\boldsymbol{G}\|_*$. □

A.2 Proofs: Localization of the Root of $h(\lambda)$

In this section, we first prove the localization of the root $\lambda^\star$ to $h(\lambda) = 0$, which is required in Section 3.2. We then present experiments showing that the computational overhead of the proposed λ-solver is negligible compared to the whole training process.

We first prove Theorem A.2, from which Theorem A.3 follows.

Theorem A.2. *The function*

$$h(\lambda) = \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda) \rangle = \langle \boldsymbol{\Theta}, \operatorname{msign}(\boldsymbol{G} + \lambda \boldsymbol{\Theta}) \rangle$$

is monotonic non-decreasing in λ.

Proof. Recall that the matrix sign function can be equivalently defined as the solution of the spectral-norm constrained maximization as in Theorem A.1:

$$\operatorname{msign}(\boldsymbol{G}) = \underset{\|T\|_2=1}{\operatorname{argmax}} \langle \boldsymbol{G}, \boldsymbol{T} \rangle. \tag{21}$$

Consider two values $\lambda_2 > \lambda_1$. By the definition of $\boldsymbol{\Phi}^\star(\lambda)$ as the maximizer of $\langle \boldsymbol{G} + \lambda\boldsymbol{\Theta}, \cdot \rangle$, we obtain

$$\begin{aligned}
\langle \boldsymbol{G} + \lambda_1 \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle \ge& \langle \boldsymbol{G} + \lambda_1 \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle \\
&\Rightarrow \quad \langle \boldsymbol{G}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle + \lambda_1 \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle \ge \langle \boldsymbol{G}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle + \lambda_1 \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle, \quad (22)\\
\langle \boldsymbol{G} + \lambda_2 \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle \ge& \langle \boldsymbol{G} + \lambda_2 \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle \\
&\Rightarrow \quad \langle \boldsymbol{G}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle + \lambda_2 \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2) \rangle \ge \langle \boldsymbol{G}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle + \lambda_2 \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1) \rangle. \quad (23)
\end{aligned}$$

Summing Equation (22) and Equation (23), we can derive

$$\lambda_1\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1)\rangle + \lambda_2\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2)\rangle \geq \lambda_1\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2)\rangle + \lambda_2\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1)\rangle \tag{24}$$

$$\Rightarrow \quad (\lambda_2-\lambda_1)\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2)\rangle \geq (\lambda_2-\lambda_1)\langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1)\rangle \quad \Rightarrow \quad \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_2)\rangle \geq \langle \boldsymbol{\Theta}, \boldsymbol{\Phi}^\star(\lambda_1)\rangle, \tag{25}$$

with $\lambda_2 - \lambda_1 > 0$. Hence, $h(\lambda) = \langle \boldsymbol{\Theta}, \mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})\rangle$ is monotonic non-decreasing in λ. □

Theorem A.3. *There exists some $\lambda^\star \in \mathbb{R}$ such that $h(\lambda^\star) = 0$, and any such root satisfies*

$$|\lambda^\star| \leq 2\|\boldsymbol{G}\|_*.$$

Proof. We first prove the existence of a root $\lambda^\star$. Since $\boldsymbol{\Theta} = \boldsymbol{u}_1\boldsymbol{v}_1^\top$ is a rank-one matrix with unit-norm singular vectors, its nuclear norm satisfies $\|\boldsymbol{\Theta}\|_* = 1$. Moreover, by construction, we have

$$\|\boldsymbol{\Phi}^\star(\lambda)\|_2 = \|\,\mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})\|_2 = 1. \tag{26}$$

By Theorem A.1, letting $\boldsymbol{T}_1 = \frac{\boldsymbol{\Phi}^\star(\lambda)}{\|\boldsymbol{\Phi}^\star(\lambda)\|_2}, \boldsymbol{T}_2 = -\frac{\boldsymbol{\Phi}^\star(\lambda)}{\|\boldsymbol{\Phi}^\star(\lambda)\|_2}$, we have

$$\|\boldsymbol{T}_1\|_2 = \|\boldsymbol{T}_2\|_2 = 1 \quad \text{and thus} \quad \langle \boldsymbol{\Theta}, \boldsymbol{T}_1\rangle \leq \|\boldsymbol{\Theta}\|_* \text{ and } \langle \boldsymbol{\Theta}, \boldsymbol{T}_2\rangle \leq \|\boldsymbol{\Theta}\|_*. \tag{27}$$

Therefore, we have

$$|\langle \boldsymbol{\Theta}, \mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})\rangle| \leq \|\boldsymbol{\Theta}\|_*\,\|\,\mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})\|_2 = 1, \tag{28}$$

which implies $|h(\lambda)| \leq 1$ for all $\lambda \in \mathbb{R}$. Moreover, since $h(\lambda)$ is monotonic non-decreasing by Theorem A.2, the limits $\lim_{\lambda\to-\infty} h(\lambda)$ and $\lim_{\lambda\to+\infty} h(\lambda)$ exist. Using the property $\mathrm{msign}(\lambda\boldsymbol{T}) = \mathrm{msign}(\boldsymbol{T})$, we have

$$\lim_{\lambda\to+\infty} \mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta}) = \lim_{\lambda\to+\infty} \mathrm{msign}\left[\lambda\left(\boldsymbol{\Theta} + \tfrac{1}{\lambda}\boldsymbol{G}\right)\right] = \mathrm{msign}(\boldsymbol{\Theta}) = \boldsymbol{\Theta}, \tag{29}$$

where the last equality is satisfied by definition of $\mathrm{msign}(\cdot)$ and $\boldsymbol{\Theta}$. Consequently,

$$\lim_{\lambda\to+\infty} h(\lambda) = \langle \boldsymbol{\Theta}, \boldsymbol{\Theta}\rangle = \mathrm{tr}(\boldsymbol{v}_1\boldsymbol{u}_1^\top\boldsymbol{u}_1\boldsymbol{v}_1^\top) = 1. \tag{30}$$

Similarly, we can also obtain $\lim_{\lambda\to-\infty} h(\lambda) = -1$. Since $\mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})$ is a subgradient of a convex function $\|\boldsymbol{G}+\lambda\boldsymbol{\Theta}\|_*$ with respect to λ according to Watson [1992], monotonic non-decreasing $h(\lambda)$ satisfies the intermediate value property. Consequently, there exists at least one $\lambda^\star \in \mathbb{R}$ such that $h(\lambda^\star) = 0$.

We next localize the root. For any $\lambda > 2\|\boldsymbol{G}\|_* > 0$ and any matrix $\boldsymbol{T}$ with $\|\boldsymbol{T}\|_2 = 1$, we have

$$\lambda\langle \boldsymbol{\Theta}, \boldsymbol{T}\rangle = \langle \boldsymbol{G} + \lambda\boldsymbol{\Theta}, \boldsymbol{T}\rangle - \langle \boldsymbol{G}, \boldsymbol{T}\rangle. \tag{31}$$

By Equation (27), $\langle \boldsymbol{G}, \boldsymbol{T}\rangle \leq \|\boldsymbol{G}\|_*$. Let $\boldsymbol{T} = \mathrm{msign}(\boldsymbol{G} + \lambda\boldsymbol{\Theta})$, using Theorem A.1, we can derive

$$\lambda h(\lambda) = \|\boldsymbol{G} + \lambda\boldsymbol{\Theta}\|_* - \langle \boldsymbol{G}, \boldsymbol{T}\rangle \geq \big|\|\boldsymbol{G}\|_* - \lambda\|\boldsymbol{\Theta}\|_*\big| - \|\boldsymbol{G}\|_* \geq \lambda - 2\|\boldsymbol{G}\|_* > 0, \tag{32}$$

and this implies $h(\lambda) > 0$. Similarly, if $\lambda < -2\|\boldsymbol{G}\|_*$, $h(\lambda) < 0$. Since $h(\lambda)$ is monotonic non-decreasing, any root $\lambda^\star$ satisfying $h(\lambda^\star) = 0$ must therefore lie in the interval $[-2\|\boldsymbol{G}\|_*, 2\|\boldsymbol{G}\|_*]$. □

Theorem A.3 actually provides a theoretical guarantee that the correctness of using the bisection algorithm to solve the root $\lambda^\star$. To further validate this theoretical result, we randomly generate several zero-centered matrices with varying dimensions, construct $\boldsymbol{G}$ and $\boldsymbol{\Theta}$ as described in this section, and plot the curve of $h(\lambda)$ together with its root. As shown in Figure 3, $f(\lambda)$ is indeed monotonic in λ, and its root lies close to $\lambda = 0$, which is consistent with our theoretical analysis.

A.3 Dynamic Spectral Weight Decay

In this section, we introduce spectral retraction mechanisms to counteract the accumulation of higher-order errors that may gradually drift the weights off the spectral sphere. As noted in Section 3.3, the remainder term $\mathcal{O}(\eta^2 R^2\|\boldsymbol{\Phi}\|_2^2)$ in the Taylor expansion can accumulate over iterations. To enforce the exact constraint $\|\boldsymbol{W}\|_2 = R$ throughout training, we apply Equation (13) that projects the weights back onto the spectral manifold.

In practice, we consider two retraction variants that differ in how strictly the spectral constraint is enforced. The hard variant applies an explicit projection to maintain $\|\boldsymbol{W}\|_2 = R$ at every step, while the dynamic variant replaces the exact projection with a soft, learning-rate-scaled radial correction that steers $\|\boldsymbol{W}\|_2$ toward the target radius. These two variants correspond to enforcing the spectral constraint exactly or approximately, and lead to different interactions with conventional weight decay.

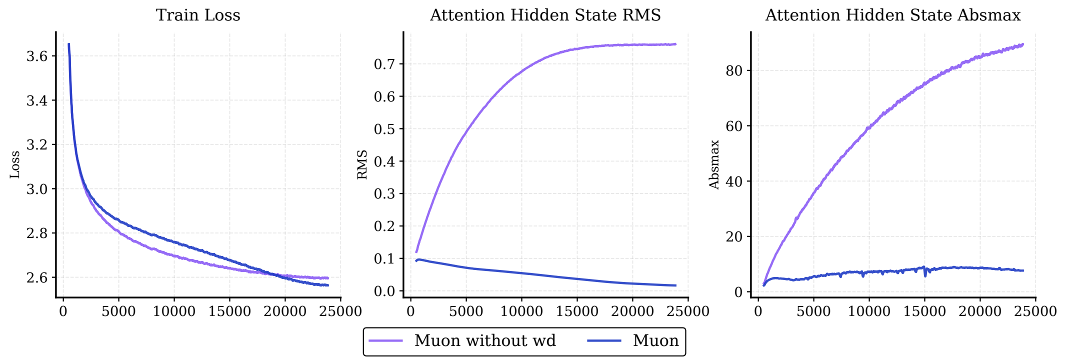


Figure 11: **Abaltion of weight decay in Muon.** In contrast with Spectral Sphere (Figure 1), without weight constraint, Muon training dynamics become instable, which in-turn would hurt performance.

Hard retraction (exact projection). The hard variant applies the retraction map explicitly using an estimated top singular value $\sigma \approx \|\boldsymbol{W}\|_2$ which is computed via Power Iteration.

$$\boldsymbol{W} \leftarrow \frac{R}{\sigma}\boldsymbol{W}. \tag{33}$$

This enforces the exact constraint $\|\boldsymbol{W}\|_2 = R$ at every step, with deviations arising solely from the approximation error in σ. Under this setting, Spectral Sphere is typically used together with standard decoupled weight decay as in AdamW: weight decay shrinks the full parameter vector, while the spectral retraction constrains the dominant singular value.

Dynamic retraction (soft spectral decay). The dynamic variant replaces the exact projection with a small, sign-controlled radial adjustment with hyperparameter λ.

$$\boldsymbol{W} \leftarrow \big(1 + \lambda\,\eta\,\mathrm{sign}(R - \sigma)\big)\boldsymbol{W}. \tag{34}$$

Instead of strictly enforcing the constraint $\|\boldsymbol{W}\|_2 = R$, this update gently adjusts the spectral norm toward the target radius R based on σ. Since the correction is proportional to η, the adjustment strength diminishes automatically as learning rate schedules. Under this formulation, dynamic retraction functions as a spectrally aligned, layer-wise analogue to decoupled weight decay. Consequently, we tie λ to the AdamW weight decay coefficient and employ the dynamic variant as a drop-in replacement for conventional weight decay.

From a geometric perspective, the two variants above can both be viewed as approximate projections onto the spectral sphere. As discussed in Section 3.3, applying retraction at the beginning of iteration $t+1$ instead of after the update at iteration t only introduces $\mathcal{O}(\eta^2)$ discrepancies. Consequently, the update remains first-order equivalent to steepest descent on the spectral manifold.

A.4 MoE Routing Scaling Factor

Following [Su, 2025c], in our MoE architecture, the output $\boldsymbol{y}$ is the sum of the shared experts and the routed experts. A critical issue arises from the magnitude mismatch between the two parts:

$$\boldsymbol{y} = \underbrace{\sum_{i=1}^{N_{\text{shared}}} \boldsymbol{e}_{\text{shared},i}}_{\text{Weight} \approx 1} + \underbrace{\sum_{j \in \text{TopK}} g_j \boldsymbol{e}_{\text{routed},j}}_{\text{Weight } g_j \ll 1} \tag{35}$$

The shared experts are directly added to the residual stream, effectively having a coefficient of 1. In contrast, the routed experts are multiplied by sigmoid probabilities g_j, which are typically small. Consequently, the variance (signal energy) of the routed experts is significantly lower than that of the shared experts, causing the optimizer to neglect the routed part.

To balance the contributions, we compute a scaling factor M to the routed experts:

$$\boldsymbol{y} = \sum_{i=1}^{N_{\text{shared}}} \boldsymbol{e}_{\text{shared},i} + M \sum_{j \in \text{TopK}} g_j \boldsymbol{e}_{\text{routed},j} \tag{36}$$

We aim to choose M such that the expected variance of the routed term matches that of the shared term, under assumption that experts have unit variance:

$$M \approx \sqrt{\frac{N_{\text{shared}}}{\mathbb{E}[\sum g_j^2]}} \tag{37}$$

Using numerical simulation (Listing 1) with $N_{\text{shared}} = 1$ and Top-4 sigmoid routing, we find $M \approx 2.0$. We find this scaling factor is crucial for MoE training stability.

Listing 1: Python implementation for estimating the MoE scaling factor M.

```python
import numpy as np

def estimate_scaling_factor(n_total=64, k_routed=4, n_shared=1):
    factors = []
    for _ in range(10000):
        # 1. Simulate logits
        logits = np.random.randn(n_total - n_shared)

        # 2. Get Sigmoid Scores
        scores = 1 / (1 + np.exp(-logits))

        # 3. Select Top-K routed scores
        scores = np.sort(scores)[::-1][:k_routed]

        # 3. Normalize scores
        scores /= scores.sum()

        # 4. Calculate Magnitude
        magnitude = np.sum(scores**2)**0.5

        factors.append(n_shared**0.5 / magnitude)

    return np.mean(factors)
```

A.5 μP Width Scaling

In Figure 2, we conduct experiment to validate μP width scaling across different model sizes from 70M to 1.8B. Additional AdamW results are included in Figure 12. We scale the hidden size, intermediate size, and number of attention heads (while fixing head dimensions). The models are trained on 30B tokens sampled from the Olmo2-mix124 dataset [Team OLMo et al., 2025]. All optimizers use the Spectral μP LR Scaler.

- The LR is swept across {1e-3, 3e-3, 5e-3, 7e-3, 9e-3, 1e-2, 1.5e-2, 2e-2, 3e-2}.
- Hidden size is swept across {256, 512, 1024, 2048}.

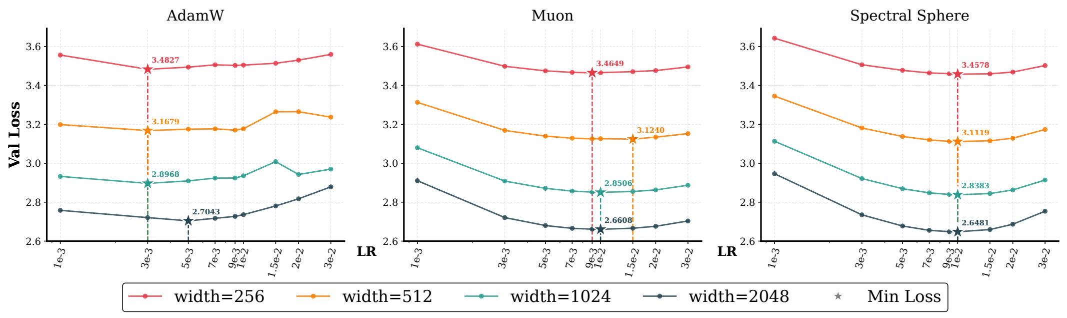


Figure 12: **μP LR grid search with AdamW, Muon and Spectral Sphere.** AdamW shows even worse validation loss, with drifting optimal LR.