Pion: A Spectrum-Preserving Optimizer via Orthogonal Equivalence Transformation

Kexuan Shi[1,†] **Hanxuan Li**[1,†] **Zeju Qiu**[1,2] **Yandong Wen**[3] **Simon Buchholz**[2] **Weiyang Liu**[1,*]
[1]The Chinese University of Hong Kong [2]Max Planck Institute for Intelligent Systems [3]Westlake University

Abstract

We introduce Pion, a spectrum-preserving optimizer for large language model (LLM) training based on orthogonal equivalence transformation. Unlike additive optimizers such as Adam and Muon, Pion updates each weight matrix through left and right orthogonal transformations, preserving its singular values throughout training. This yields an optimization mechanism that modulates the geometry of weight matrices while keeping their spectral norm fixed. We derive the Pion update rule, systematically examine its design choices, and analyze its convergence behavior along with several key properties. Empirical results show that Pion offers a stable and competitive alternative to standard optimizers for both LLM pretraining and finetuning.

1 Introduction

As large language models (LLMs) continue to scale, the difficulty of training them also increases significantly. One of the most critical challenges today is designing optimizers that are both efficient and stable. Training stability can be partially characterized by the Maximal Update Parameterization (μP) [90], where spectral norms of weights and updates are constrained such that width-invariant activations are of constant scale and hence prevent explosions. By performing the steepest descent under the spectral norm through update orthogonalization, Muon [37] has emerged as a competitive alternative to AdamW [38, 51]. Although Muon's orthogonalization ensures that each update is easily μP-compatible, the spectral norms of the weight matrices themselves may still drift throughout training. Built upon Muon, recent work addresses this issue either by introducing normalization [21, 32, 43, 78] or by incorporating spectral retraction directly into the update rule [85]. Rather than adapting Muon to achieve μP, we introduce Pion, a fundamentally different optimizer that constrains the spectral norm of both weights and updates through its optimization dynamics. Specifically, Pion is derived from orthogonal equivalence transformations of weight matrices, updating each matrix via coupled left and right orthogonal transformations. Its design is guided by the following principles:

- **Algorithmic spectrum control**: Pion derives the update rule directly on the iso-spectral manifold, eliminating the need for explicit normalization while preserving the weight spectrum throughout optimization. This property is particularly desirable, as the upper-bounded spectral norm of the weight is closely linked to stronger generalization [5, 36, 57, 91]. Moreover, the update's spectral norm is also guaranteed to be upper bounded, making Pion easily compatible with μP.
- **Minimum energy training**: Pion updates weight matrices via orthogonal equivalence transformations, which inherently preserve hyperspherical energy [47, 49]. This energy quantifies how uniformly normalized neurons are distributed on the hypersphere, and lower energy has been shown to correlate with better generalization [44, 47, 48]. Because zero-mean Gaussian weight initialization yields a minimum-energy configuration, Pion provably preserves this configuration throughout training, maintaining a uniform hyperspherical distribution of normalized neurons.

Pion is inspired by orthogonal pretraining [48] and POET [63, 64], which reparameterizes each weight matrix as a left orthogonal matrix, a randomly initialized base weight, and a right orthogonal matrix, learning only the two orthogonal factors. This reparameterization enforces spectrum preservation by construction, but recasts the optimization variables from the weights themselves to auxiliary orthogonal

[†]Equal contribution [*]Corresponding author Project page: spherelab.ai/pion

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

parameters. While this shift enables greater memory efficiency [64], it also complicates training dynamics, giving rise to issues such as loss spikes and the need for careful momentum design. Pion, short for POET-induced optimizer with no reparameterization, removes this auxiliary parameterization and instead turns the same principle into a direct optimizer. Specifically, Pion updates each weight matrix through left and right orthogonal transformations, preserving its singular-value spectrum throughout training while operating directly on the model weights. This yields a spectrum-preserving optimization dynamics that retains the geometric inductive bias of POET in a simpler and more stable optimizer form.

2 Pion: A Spectrum-Preserving Optimizer for LLM Training

2.1 Background and Preliminaries

POET [63, 64] reparameterizes each weight matrix as $\boldsymbol{W}_{RP} = \boldsymbol{R}\boldsymbol{W}_0\boldsymbol{P}$, where $\boldsymbol{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is fixed at random initialization, and $\boldsymbol{R} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$, $\boldsymbol{P} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ are trainable orthogonal matrices. This corresponds to an orthogonal equivalence transformation that acts on both sides of $\boldsymbol{W}_0$, yielding the forward pass weight $\boldsymbol{R}\boldsymbol{W}_0\boldsymbol{P}$. After training, $\boldsymbol{R}$ and $\boldsymbol{P}$ are merged into the weight matrix, so POET incurs no additional inference overhead. However, since optimization is performed over two orthogonal matrices while the weight matrix remains fixed throughout training, this reparameterization poses non-trivial challenges in both training stability and cross-architecture adaptability. Motivated by this, Pion introduces a novel update rule that fully preserves the weight spectrum without reparameterization.



(a) POET's update (b) Pion's update

Figure 1: Comparison of POET and Pion (Green: learnable).

2.2 Spectrum-Preserving Update Rule

We begin with the intuition behind Pion's update rule. Consider a general weight matrix $\boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. At iteration t, we can trivially write $\boldsymbol{W}_t$ as $\boldsymbol{W}_t = \boldsymbol{I}_{d_{\text{out}}} \boldsymbol{W}_t \boldsymbol{I}_{d_{\text{in}}}$ where $\boldsymbol{I}_{d_{\text{out}}}$ and $\boldsymbol{I}_{d_{\text{in}}}$ are identity matrices of the corresponding sizes. Geometrically, each identity matrix is the neutral element of an orthogonal group, representing a zero rotation. Pion leverages this observation by *updating the identity factors directly on the orthogonal group* without an explicit reparameterization like $\boldsymbol{RWP}$. This induces left and right orthogonal transformations of $\boldsymbol{W}_t$, preserving its spectrum. See the comparison in Figure 1.

The challenge is to update the identity factors on the orthogonal group. Since the orthogonal group is a compact Lie group, we use standard techniques from Lie group optimization [12, 41, 56]. Let $\boldsymbol{G}_t \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ denote the gradient of $\boldsymbol{W}_t$. Pion updates $\boldsymbol{W}_t$ in a spectrum-preserving manner:

$$\boldsymbol{G}_t^{\text{in}} = \boldsymbol{W}_t^\top \boldsymbol{G}_t - (\boldsymbol{W}_t^\top \boldsymbol{G}_t)^\top, \quad \boldsymbol{G}_t^{\text{out}} = \boldsymbol{G}_t \boldsymbol{W}_t^\top - (\boldsymbol{G}_t \boldsymbol{W}_t^\top)^\top, \quad \boxed{\boldsymbol{W}_{t+1} = \exp(-\eta \boldsymbol{G}_t^{\text{out}}) \boldsymbol{W}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}})} \tag{1}$$

where $\exp(\cdot)$ denotes the matrix exponential. The spectrum-preserving update rule can be understood in three steps; its full derivation is given in Appendix A.1. First, we apply the chain rule to the two identity factors $\boldsymbol{I}_{d_{\text{out}}}$ and $\boldsymbol{I}_{d_{\text{in}}}$, computing the corresponding gradients $\boldsymbol{G}_t \boldsymbol{W}_t^\top$ and $\boldsymbol{W}_t^\top \boldsymbol{G}_t$. Second, we enforce skew-symmetry to project these gradients onto the Lie algebra, producing Lie algebra elements from the two identity factors. Finally, we map these elements back to the Lie group via the matrix exponential, which produces valid orthogonal transformations.

2.3 Properties of Pion's Update Rule

Pion's update rule can be written as $\boldsymbol{W}_{t+1} = \boldsymbol{R}_t \boldsymbol{W}_t \boldsymbol{P}_t$ with $\boldsymbol{R}_t = \exp(-\eta \boldsymbol{G}_t^{\text{out}})$ and $\boldsymbol{P}_t = \exp(-\eta \boldsymbol{G}_t^{\text{in}})$, where both $\boldsymbol{R}_t$ and $\boldsymbol{P}_t$ are orthogonal. Hence Pion transforms the row and column subspaces of $\boldsymbol{W}_t$ while preserving its singular values. We start with Pion's geometric structures.

Proposition 2.1 (Geometric structure of Pion's update)**.** *Let $\Delta\boldsymbol{W} = \boldsymbol{W}_{t+1} - \boldsymbol{W}_t$. Pion's update preserves the singular values of $\boldsymbol{W}_t$ and only changes its row and column subspaces through orthogonal transformations, in which positive determinant leads to rotation. Consequently, $\|\Delta\boldsymbol{W}\|_F$ characterizes the total rotational strength applied to $\boldsymbol{W}_t$. At a finer level, the in-side and out-side updates decompose into independent planar rotations on orthogonal 2D invariant subspaces induced by $\boldsymbol{G}_t^{in}$ and $\boldsymbol{G}_t^{out}$. The quantities $\frac{1}{\sqrt{d_{in}}}\|\boldsymbol{G}_t^{in}\|_F$ and $\frac{1}{\sqrt{d_{out}}}\|\boldsymbol{G}_t^{out}\|_F$ characterize the average rotation magnitudes on the two sides, while $\|\boldsymbol{G}_t^{in}\|_2$ and $\|\boldsymbol{G}_t^{out}\|_2$ control the maximum rotation angles.*

Because $\boldsymbol{R}_t$ and $\boldsymbol{P}_t$ are orthogonal, they preserve the ℓ_2 norms of the rows and columns of $\boldsymbol{W}_t$. Hence, $\Delta\boldsymbol{W}$ reflects angular deviation rather than rescaling. The update norm is therefore directly interpretable as rotational motion, unlike vanilla gradient descent, which generally entangles changes in magnitude and direction. Appendix A.2 provides a detailed derivation of the planar-rotation view.

The following theorem shows that Pion's spectrum-preserving update admits first-order convergence guarantees on the isospectral manifold under standard assumptions. The proof is given in Appendix B.

Theorem 2.2 (Pion's Convergence on the Isospectral Manifold)**.** *Assume that $f(\boldsymbol{W})$ is L-smooth and lower bounded by $f_{\inf}$. Let the stochastic gradient be $\tilde{\boldsymbol{G}}_t = \nabla f(\boldsymbol{W}_t) + \boldsymbol{\xi}_t$, where $\mathbb{E}_t[\boldsymbol{\xi}_t] = 0$ and $\mathbb{E}_t[\|\boldsymbol{\xi}_t\|_F^2] \le \sigma^2$. Assume the iterates remain on the iso-spectral manifold induced by $\boldsymbol{W}_0$, so that $\|\boldsymbol{W}_t\|_2 = \gamma$ for all t. Define $\boldsymbol{G}_t^{\text{in}} = \boldsymbol{W}_t^\top \nabla f(\boldsymbol{W}_t) - \nabla f(\boldsymbol{W}_t)^\top \boldsymbol{W}_t$ and $\boldsymbol{G}_t^{\text{out}} = \nabla f(\boldsymbol{W}_t)\boldsymbol{W}_t^\top - \boldsymbol{W}_t \nabla f(\boldsymbol{W}_t)^\top$. Assume the updates are performed for $T+1$ iterations with step size $\eta = C/\sqrt{T+1}$, where $C > 0$ is sufficiently small such that the one-step descent coefficient is positive. Then we have*

$$\min_{0 \le t \le T} \mathbb{E}\left[\|\boldsymbol{G}_t^{\text{in}}\|_F^2 + \|\boldsymbol{G}_t^{\text{out}}\|_F^2\right] \le \frac{1}{\sqrt{T+1}}(C_1 + C_2) = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right), \tag{2}$$

where C_1 depends on the initial optimality gap $f(\boldsymbol{W}_0) - f_{\inf}$, and C_2 depends on L, γ, σ^2, and C.

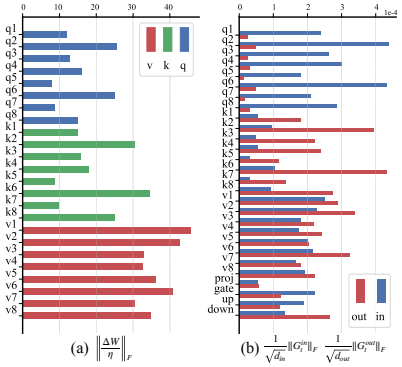
Remark 2.3. Theorem 2.2 establishes first-order stationarity on the isospectral manifold. However, it does not guarantee stationarity in the ambient space, as a critical point of the restricted problem may still admit a nonzero ambient gradient that is normal to the manifold. Appendix B.3 gives more details.

2.4 Design Principles for Stable Training and Convergence

While Pion's update rule offers a simple and functional approach to spectrum-preserving optimization, training practical LLMs demands additional design choices for greater stability. To this end, we explore the following design principles. We note that our exploration is by no means comprehensive, but rather represents an initial yet principled effort toward building a stable spectrum-preserving optimizer. For rapid prototyping, we perform all design explorations using a 60M-parameter LLaMA-based model [73, 80, 95], a common setup for ablation [26, 63, 96]. All the models in this section are trained on C4 [67] with sequence length 256 for 9.6B tokens, ensuring sufficient training.

2.4.1 Consistent Update

To train deep neural networks effectively, prior work [4, 24, 35, 86, 90] has sought to keep network components operating under stable input/output distributions and receiving consistent feature updates. This consistency principle has also guided the scaling of modern optimizers [8, 46, 85] to large models. In particular, optimizer-induced parameter updates $\frac{1}{\eta}\Delta\boldsymbol{W}$ are expected to be scale-consistent, *i.e.*, their norms should grow proportionally with the size of the corresponding parameter space. We analyze the training dynamics of Pion and identify two notable violations of this principle. First, the original update produces substantial heterogeneity in the normalized update magnitude, $\|\frac{1}{\eta}\Delta\boldsymbol{W}\|_F$, across identically sized parameter matrices within the same layer, as shown in Figure 2(a). Second, for individual parameter matrices, the average bilateral rotation angles, which are measured by $\frac{1}{\sqrt{d_{\text{in}}}}\|\boldsymbol{G}_t^{\text{in}}\|_F$ and $\frac{1}{\sqrt{d_{\text{out}}}}\|\boldsymbol{G}_t^{\text{out}}\|_F$, show a pronounced imbalance between the input and output sides, as shown in Figure 2(b). These inconsistencies stem from a geometric mismatch: the naive update's transformation dynamics are neither scale-consistent across the full parameter feature space nor balanced between each matrix's input and output feature spaces. We resolve these mismatches by controlling update magnitude in the Lie algebra. Specifically, we normalize the in-side and out-side skew-symmetric gradients:



(a) $\left\|\frac{\Delta W}{\eta}\right\|_F$ (b) $\frac{1}{\sqrt{d_{\text{in}}}}\|G_t^{\text{in}}\|_F$ $\frac{1}{\sqrt{d_{\text{out}}}}\|G_t^{\text{out}}\|_F$

Figure 2: Inconsistent updates in Pion.

$$\boldsymbol{G}_t^{\text{out}} \leftarrow \sqrt{d_{\text{out}}} \cdot \boldsymbol{G}_t^{\text{out}} / \|\boldsymbol{G}_t^{\text{out}}\|_F, \quad \boldsymbol{G}_t^{\text{in}} \leftarrow \sqrt{d_{\text{in}}} \cdot \boldsymbol{G}_t^{\text{in}} / \|\boldsymbol{G}_t^{\text{in}}\|_F. \tag{3}$$

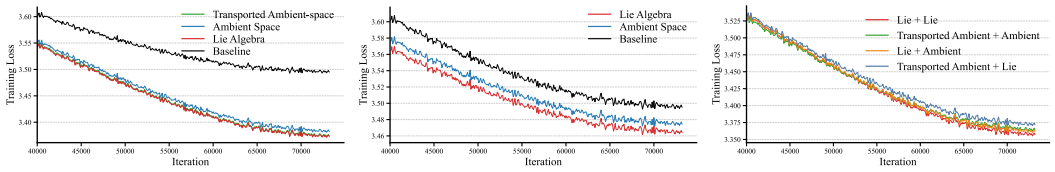
To enforce scale consistency across weight matrices, we introduce a per-weight coefficient α_t below:

$$\boldsymbol{W}_{t+1} = \exp(-\eta\alpha_t \boldsymbol{G}_t^{\text{out}})\boldsymbol{W}_t \exp(-\eta\alpha_t \boldsymbol{G}_t^{\text{in}}), \quad \text{s.t. RMS}(\Delta\boldsymbol{W}/\eta) \approx \text{const}. \tag{4}$$

Directly computing $\frac{1}{\eta}\Delta\boldsymbol{W}$ under the exponential map would nearly double the computational cost, so we instead use a first-order approximation to compute α:

$$\alpha_t \approx \frac{c\sqrt{d_{\text{out}} d_{\text{in}}}}{\|\Delta\boldsymbol{W}/\eta\|_F + \epsilon}, \quad \text{where } \Delta\boldsymbol{W}/\eta \approx -\boldsymbol{G}_t^{\text{out}}\boldsymbol{W}_t - \boldsymbol{W}_t\boldsymbol{G}_t^{\text{in}}. \tag{5}$$

Such per-weight scaling in Equation (4) makes the effective rotational update magnitude scale with the size of each weight matrix, so that the rotational strengths across different matrices are approximately



(a) First-Order Momentum with RMS Scaling (b) Second-Order Momentum with RMS Scaling (c) First- and Second-Order Momentum with RMS Scaling

Figure 4: Training loss curves of momentum designs. Figures (a) and (b) show first-order-only and second-order-only momentum, both with RMS scaling. Figure (c) combines the two momentum techniques. "Lie+Lie" denotes Lie-algebra first- and second-order momentum. "Transported Ambient + Ambient" denotes transported ambient-space first-order momentum with ambient-space second-order momentum.

proportional to the square root of the ratio of their parameter counts. Such RMS scaling is also used to ensure proportionally consistent updates in Euclidean space [26, 46].

Results in Figure 3 show that the naive update rule performs well only under small learning rates and diverges when the learning rate becomes large. In contrast, the RMS-controlled scale-consistent update ensures consistent update magnitudes across matrices and can effectively utilize larger learning rates to accelerate convergence. Applying bilateral normalization alone, or combining RMS control with bilateral normalization, does not further improve training stability or final performance. Taking a step further, these results suggest that scale-consistent rotation update across parameter matrices is key to stable spectrum-preserving optimization. Due to its empirical effectiveness, we therefore adopt RMS-controlled scale consistency as a core component of Pion. Because the bilateral normalization does not show obvious benefits, it is not adopted in Pion.

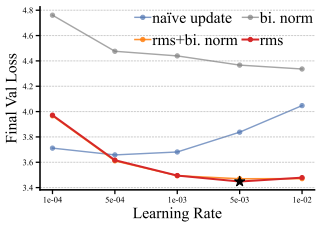


Figure 3: Validation loss comparison of different consistent update strategies. "★" denotes the achieved minimum validation loss.

2.4.2 Momentum Design

A key ingredient to accelerate gradient-based iterative optimization is momentum [59, 62], which uses accumulated gradient information to adapt the current update direction. This technique has inspired a range of highly effective first-order optimizers for deep learning [22, 38, 68, 93]. In this section, we explore how to integrate momentum into the Pion's update rule in Equation (1).

Transported ambient-space first-order momentum. We note that the update trajectory $\{\boldsymbol{W}_t\}_{t=1}^{\infty}$ evolves on a smooth iso-spectral manifold $\mathcal{M}$ with nonzero curvature (we assume the singular values of $\boldsymbol{W}_0$ are distinct and nonzero). Hence, momentum should be expressed in a consistent tangent space. A natural approach is to parallel-transport the historical momentum to the current tangent space [6, 75]. Following this idea, we derive a transported first-order momentum update below. After the t-th step, $\exp(-\eta \boldsymbol{G}_t^{\text{out}})$ and $\exp(-\eta \boldsymbol{G}_t^{\text{in}})$ used in the update are reused to transport $\boldsymbol{m}_t$ to the tangent space:

$$\boldsymbol{m}_t = \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{G}_t, \quad \boldsymbol{G}_t^{\text{in}} = \boldsymbol{W}_t^\top \boldsymbol{m}_t - (\boldsymbol{W}_t^\top \boldsymbol{m}_t)^\top, \quad \boldsymbol{G}_t^{\text{out}} = \boldsymbol{m}_t \boldsymbol{W}_t^\top - (\boldsymbol{m}_t \boldsymbol{W}_t^\top)^\top,$$
$$\underbrace{\boldsymbol{W}_{t+1} = \exp(-\eta \boldsymbol{G}_t^{\text{out}}) \boldsymbol{W}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}})}_{\text{Update rule with first-order mementum}}, \qquad \underbrace{\boldsymbol{m}_t \leftarrow \exp(-\eta \boldsymbol{G}_t^{\text{out}}) \boldsymbol{m}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}})}_{\text{Transported to } T_{\boldsymbol{W}_{t+1}}\mathcal{M}}. \tag{6}$$

where $T_{\boldsymbol{W}_{t+1}}\mathcal{M}$ denote the tangent space at $\boldsymbol{W}_{t+1} \in \mathcal{M}$. The next gradient $\boldsymbol{G}_{t+1}$ is also in $T_{W_{t+1}}\mathcal{M}$. This parallel transport improves the accuracy of first-order gradient estimation. For comparison, we also consider a first-order momentum variant without transport, referred to as ambient-space momentum. As shown in Figure 4(a), we empirically compare the new update rule above with or without momemtum transportation, and the transported version consistently achieves faster convergence.

Lie algebra first-order momentum. Because Pion's update is performed in the Lie algebra, *i.e.*, within a single tangent space. This property provides another way to accumulate first-order momentum, in which the accumulation is performed directly in the Lie algebra. Let $\boldsymbol{m}_t^{\text{in}} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ and $\boldsymbol{m}_t^{\text{out}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ denote the momentum variables associated with the in-side and out-side gradient update, respectively. Given the same $\boldsymbol{G}_t^{\text{in}}$ and $\boldsymbol{G}_t^{\text{out}}$ in Equation (1), the modified update rule becomes

$$\underbrace{\boldsymbol{m}_t^{\text{in}} = \beta_1 \boldsymbol{m}_{t-1}^{\text{in}} + (1-\beta_1)\boldsymbol{G}_t^{\text{in}}}_{\text{Input-side Lie algebra momentum}}, \quad \underbrace{\boldsymbol{m}_t^{\text{out}} = \beta_1 \boldsymbol{m}_{t-1}^{\text{out}} + (1-\beta_1)\boldsymbol{G}_t^{\text{out}}}_{\text{Output-side Lie algebra momentum}}, \quad \underbrace{\boldsymbol{W}_{t+1} = \exp(-\eta \boldsymbol{m}_t^{\text{out}}) \boldsymbol{W}_t \exp(-\eta \boldsymbol{m}_t^{\text{in}})}_{\text{Update rule with first-order momentum}}. \tag{7}$$

As shown in Figure 4, the Lie algebra momentum achieves the fastest convergence, slightly outperforming the transported ambient-space momentum. All the first-order momentum formulations are spectrum-preserving by construction, as they generate updates via skew-symmetric operators followed by the matrix exponential map. Ambient-space momentum is the most efficient in compute and memory,

but produces biased estimates due to tangent space mismatch. Transported ambient-space momentum corrects this bias via parallel transport at added computational cost, with no extra memory overhead. Lie algebra momentum achieves exact, geometrically consistent estimation at comparable compute cost, but requires additional memory for separate in- and out-side variables.

Second-order momentum. Second-order momentum tracks the running average of squared gradients, serving as an adaptive normalization factor. Unlike its first-order counterpart, it accumulates no directional information across tangent spaces and therefore does not require parallel transport in manifold optimization [6]. Guided by this observation and the design principles established for first-order momentum, we consider two natural implementations of second-order momentum: (1) estimating second-order momentum in the ambient space using the full gradient $\boldsymbol{G}_t$; and (2) modeling second-order statistics separately for the in-side and out-side gradients. Specifically, we use the standard second-order momentum formulation in AdamW. For example, the second-order momentum for the in-side update in Equation (7) is $\boldsymbol{v}_t^{\text{in}} = \beta_2 \boldsymbol{v}_{t-1}^{\text{in}} + (1-\beta_2)(\boldsymbol{G}_t^{\text{in}} \odot \boldsymbol{G}_t^{\text{in}})$ where $\odot$ is the element-wise multiplication. The complete algorithms are given in Algorithm 1 and Appendix G.1.

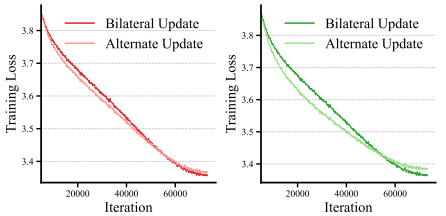
From Figure 4(b), we observe that the Lie Algebra variants consistently outperform the ambient-space variants. We then examine their combined behavior in Figure 4(c). The Lie+Lie variant performs best, consistent with the trends observed for each momentum order individually. In contrast, mixed variants perform slightly worse, suggesting that mismatched deployments hinder this complementarity. This observation further suggests that, under our spectrum-preserving update rule, accumulating momentum in the Lie algebra provides a more natural and effective formulation. Thus, we adopt the Lie+Lie and Transported-Ambient+Ambient variants as two functional implementations of momentum in Pion, representing two effective design choices identified from our exploration.

2.4.3 Alternate Update

The bilateral update in Equation (1) applies both input-side and output-side orthogonal transformations at every iteration. We propose a more computationally efficient Pion variant that alternates between the input-side and output-side updates across successive iterations:

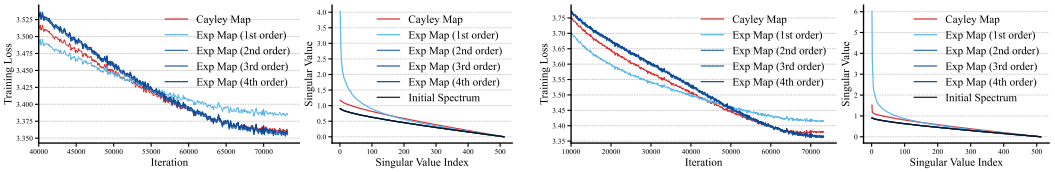
$$\boldsymbol{W}_{t+1} = \exp(-\eta(1-\psi)\boldsymbol{G}_t^{\text{out}}) \cdot \boldsymbol{W}_t \cdot \exp(-\eta\psi\boldsymbol{G}_t^{\text{in}}), \quad \text{with } \psi = 1 \text{ if } t \text{ is odd, } \psi = 0 \text{ if } t \text{ is even.} \tag{8}$$

The alternate update remains spectrum-preserving and also decouples the two orthogonal transformations across iterations, largely reducing per-step computation. Moreover, the alternation can be naturally extended to occur every few steps rather than every step. From Figure 5, we observe that alternate update achieves performance close to bilateral update for both Lie+Lie and Transported-Ambient+Ambient variants. For Lie+Lie, the final loss of alternate update is 3.3654, only about 0.23% higher than the 3.3575 achieved by bilateral update. This small gap suggests that updating the two transformations simultaneously is not always necessary to obtain strong optimization performance, and that much of the benefit can be retained through a more lightweight alternating scheme. Alternate updates are slightly faster early in training, while bilateral updates overtake them near convergence, reflecting a tradeoff between efficiency and refinement. Overall, the alternate update is a strong compromise, as it preserves spectral structure, lowers computational cost, and achieves nearly the same final performance as the more expensive bilateral update.



(a) Lie+Lie (b) Transported-Ambient+Ambient

Figure 5: Training loss of bilateral and alternate update.



(a) "Lie+Lie" with different approximations of Exp. (b) "Transported-Ambient+Ambient" with different approximations of Exp.

Figure 6: Comparison of different approximation schemes for $\exp(\cdot)$. Left panels: training loss curves under each scheme for the Lie+Lie and Transported-Ambient+Ambient configurations. Right panels: final singular value spectra alongside the initial spectra for reference.

2.4.4 Efficient Approximation to Matrix Exponential Map

All Pion variants require computing the matrix exponential. Since exact evaluation of $\exp(\boldsymbol{A})$ is generally expensive, we consider two efficient approximations. The first is the Cayley transform [31,

42, 48]: $\exp(\boldsymbol{A}) \approx \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{A}\right)^{-1}\left(\boldsymbol{I} + \frac{1}{2}\boldsymbol{A}\right)$, which agrees with $\exp(\boldsymbol{A})$ up to second order with error $\mathcal{O}(\|\boldsymbol{A}\|^3)$ and strictly preserves orthogonality for skew-symmetric $\boldsymbol{A}$. In practice, the matrix inverse can be further cheapened via low-order series expansions [63, 65]. The second is a truncated power series: $\exp(\boldsymbol{A}) \approx \sum_{k=0}^{L} \frac{\boldsymbol{A}^k}{k!}$, whose truncation error decays rapidly with L when $\|\boldsymbol{A}\|_F$ is small.

Figure 6 compares first- to fourth-order power-series approximations with the Cayley transform. The first-order approximation degrades both convergence and singular-value preservation, while the Cayley variant offers only modest gains. The second-order approximation preserves the spectrum nearly as well as higher-order variants. Unlike conventional orthogonal-group optimization where errors in $\exp(\boldsymbol{A}_t)$ can accumulate through updates, Pion's update always starts from the identity matrix (Equation 1). This prevents repeated error compounding and improves numerical robustness, making higher-order approximations unnecessary. Pion therefore adopts a second-order exponential approximation.

2.5 Detailed Implementation and Computational Complexity

The previous exploration and experiments motivate four design choices. First, scale-consistent rotational scaling is essential for stable training and large learning rates, whereas bilateral rotation balancing has a much weaker empirical effect. Second, Lie-algebra momentum better aligns with the geometry of spectrum-preserving updates than ambient-space accumulation. Third, alternate update retains most benefits of bilateral updates at substantially lower computational cost, though bilateral updates offer slightly better refinement near the final convergence. Empirically, we find that a second-order approximation of $\exp(\cdot)$ works sufficiently well for both optimization and spectrum preservation. Based on these observations, the final Pion optimizer combines RMS-controlled scaling, Lie-algebra first-order and second-order momentum, and a second-order truncated approximation to the matrix exponential. Algorithm 1 gives the resulting optimizer steps. Appendix G.1 presents an alternative implementation that uses transported ambient-space first-order momentum together with ambient-space second-order momentum.

Algorithm 1: The Pion Optimizer (Lie Algebra)

Input: Learning rate η, momentum coefficients β_1, β_2, RMS constant c, stability constant ϵ, alternating flag, initial weight matrix $\boldsymbol{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$

Output: Optimized parameter $\boldsymbol{W}_t$

Initialize $\boldsymbol{m}_0^{\text{in}}, \boldsymbol{v}_0^{\text{in}} \leftarrow \boldsymbol{0} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$, $\boldsymbol{m}_0^{\text{out}}, \boldsymbol{v}_0^{\text{out}} \leftarrow \boldsymbol{0} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$;

Define $\mathcal{E}_2(\boldsymbol{A}, \alpha) \leftarrow \boldsymbol{I} + \eta\alpha\boldsymbol{A} + \frac{1}{2}(\eta\alpha\boldsymbol{A})^2$;

for $t = 1, 2, \ldots$ **do**

- $\boldsymbol{G}_t \leftarrow \nabla_{\boldsymbol{W}} f(\boldsymbol{W}_{t-1})$;
- $\boldsymbol{G}_t^{\text{in}} \leftarrow \boldsymbol{W}_{t-1}^\top \boldsymbol{G}_t - \boldsymbol{G}_t^\top \boldsymbol{W}_{t-1}$, $\boldsymbol{G}_t^{\text{out}} \leftarrow \boldsymbol{G}_t \boldsymbol{W}_{t-1}^\top - \boldsymbol{W}_{t-1} \boldsymbol{G}_t^\top$;
- $\boldsymbol{m}_t^{\text{in}} \leftarrow \beta_1 \boldsymbol{m}_{t-1}^{\text{in}} + (1-\beta_1)\boldsymbol{G}_t^{\text{in}}$, $\boldsymbol{m}_t^{\text{out}} \leftarrow \beta_1 \boldsymbol{m}_{t-1}^{\text{out}} + (1-\beta_1)\boldsymbol{G}_t^{\text{out}}$;
- $\boldsymbol{v}_t^{\text{in}} \leftarrow \beta_2 \boldsymbol{v}_{t-1}^{\text{in}} + (1-\beta_2)(\boldsymbol{G}_t^{\text{in}} \odot \boldsymbol{G}_t^{\text{in}})$, $\boldsymbol{v}_t^{\text{out}} \leftarrow \beta_2 \boldsymbol{v}_{t-1}^{\text{out}} + (1-\beta_2)(\boldsymbol{G}_t^{\text{out}} \odot \boldsymbol{G}_t^{\text{out}})$;
- $\boldsymbol{A}_t^{\text{in}} \leftarrow -\boldsymbol{m}_t^{\text{in}} / (\sqrt{\boldsymbol{v}_t^{\text{in}}} + \epsilon)$, $\boldsymbol{A}_t^{\text{out}} \leftarrow -\boldsymbol{m}_t^{\text{out}} / (\sqrt{\boldsymbol{v}_t^{\text{out}}} + \epsilon)$;
- **if** *alternative update is used* **then**
  - **if** *t is even* **then**
    - $\alpha_t \leftarrow \frac{c\sqrt{d_{\text{out}} d_{\text{in}}}}{\|\boldsymbol{A}_t^{\text{out}} \boldsymbol{W}_{t-1}\|_F + \epsilon}$;
    - $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{W}_{t-1}$;
  - **else**
    - $\alpha_t \leftarrow \frac{c\sqrt{d_{\text{out}} d_{\text{in}}}}{\|\boldsymbol{W}_{t-1} \boldsymbol{A}_t^{\text{in}}\|_F + \epsilon}$;
    - $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_{t-1} \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$;
  - **end if**
- **else**
  - $\alpha_t \leftarrow \frac{c\sqrt{d_{\text{out}} d_{\text{in}}}}{\|\boldsymbol{A}_t^{\text{out}} \boldsymbol{W}_{t-1} + \boldsymbol{W}_{t-1} \boldsymbol{A}_t^{\text{in}}\|_F + \epsilon}$;
  - $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{W}_{t-1} \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$;
- **end if**

end for

return $\boldsymbol{W}_t$;

Computation Complexity. For a weight matrix $\boldsymbol{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, the main overhead is constructing the input- and output-side Lie algebra gradients and applying the second-order exponential approximation. Computing $\boldsymbol{G}^{\text{in}} = \boldsymbol{W}^\top \boldsymbol{G} - \boldsymbol{G}^\top \boldsymbol{W}$ costs $4d_{\text{out}} d_{\text{in}}^2$ FLOPs, and computing $\boldsymbol{G}^{\text{out}} = \boldsymbol{G}\boldsymbol{W}^\top - \boldsymbol{W}\boldsymbol{G}^\top$ costs $4d_{\text{out}}^2 d_{\text{in}}$ FLOPs. Element-wise momentum and second-moment updates are lower-order. RMS scaling requires evaluating $\boldsymbol{A}^{\text{out}}\boldsymbol{W} + \boldsymbol{W}\boldsymbol{A}^{\text{in}}$, costing another $2d_{\text{out}}^2 d_{\text{in}} + 2d_{\text{out}} d_{\text{in}}^2$ FLOPs. Applying the second-order update contributes $\mathcal{O}(d_{\text{out}}^3 + d_{\text{in}}^3)$ FLOPs for squared Lie matrices and $2d_{\text{out}}^2 d_{\text{in}} + 2d_{\text{out}} d_{\text{in}}^2$ FLOPs for left and right multiplication with $\boldsymbol{W}$. Thus, the dominant additional cost of one bilateral Pion update is $\mathcal{O}(d_{\text{out}}^2 d_{\text{in}} + d_{\text{out}} d_{\text{in}}^2 + d_{\text{out}}^3 + d_{\text{in}}^3)$. Alternate update can reduce the dominant update-side cost by roughly half. Compared with the baseline cost $\mathcal{O}(B d_{\text{out}} d_{\text{in}})$ for the forward and backward passes of a linear layer with batch-token size B, the relative overhead of Pion is approximately $\mathcal{O}(\frac{d_{\text{out}} + d_{\text{in}}}{B} + \frac{d_{\text{out}}^3 + d_{\text{in}}^3}{B d_{\text{out}} d_{\text{in}}})$. In LLM pretraining, B is typically large because it equals the number of tokens processed by the layer in one optimization step. Consequently, the optimizer-side matrix multiplications are amortized over a large token batch, and the overhead remains small relative to forward-backward computation. More detailed analysis and memory overhead are in Appendix C.

2.6 Compatibility with Maximal Update Parametrization

Maximal update parametrization [88–90] provides a principled framework for understanding training stability and enabling optimizer hyperparameter transfer across model scales. A central insight of μP is

that stable and effective training requires appropriate spectral scaling of both model weights and their updates. It identifies the following two spectral scaling conditions as essential for training stability:

$$\text{Forward spectral condition: } \underbrace{\|\boldsymbol{W}\|_2 = \Theta\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right)}_{\text{This is inherently satisfied by Pion.}} \quad \text{Update spectral condition: } \underbrace{\|\Delta \boldsymbol{W}\|_2 = \Theta\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right)}_{\text{This is easily satisfied by Muon.}}. \tag{9}$$

Existing μP-compatible optimizers [46, 78, 85] are built on Muon, which inherently satisfies the update condition. As a result, prior work focuses on modifying Muon to also satisfy the forward condition. Pion takes the opposite route: it inherently satisfies the forward condition, so our goal is to make it satisfy the update condition. To this end, we approximate Pion's weight-update magnitude as $\|\Delta \boldsymbol{W}_t\|_2 \approx \eta \|\boldsymbol{W}_t\|_2 (\|\boldsymbol{G}_t^{\text{out}}\|_2 + \|\boldsymbol{G}_t^{\text{in}}\|_2)$, where $\|\boldsymbol{W}_t\|_2$ naturally satisfies μP's spectral condition $\Theta(\sqrt{d_{\text{out}}/d_{\text{in}}})$. Thus, it suffices to maintain $\|\boldsymbol{G}_t^{\text{out}}\|_2 = \Theta(1)$ and $\|\boldsymbol{G}_t^{\text{in}}\|_2 = \Theta(1)$ for Pion to be μP-compatible. This can be achieved by normalizing the spectral norms of both factors, or alternatively by orthogonalizing $\boldsymbol{G}_t^{\text{out}}$ and $\boldsymbol{G}_t^{\text{in}}$. We give detailed derivation and full results in Appendix F.

To empirically assess the μP-compatibility of Pion, we conduct a hyperparameter transfer experiment across model scales. Specifically, we consider the Pion variant that applies simple normalization to both $\|\boldsymbol{G}_t^{\text{out}}\|_2$ and $\|\boldsymbol{G}_t^{\text{in}}\|_2$. We defer the detailed experimental settings and extended experimental results for the $\boldsymbol{G}_t$-orthogonalized variant to Appendix F. We train two representative LLM architectures (Llama and Qwen) over a range of model widths using the μP-compatible Pion, and examine whether optimal hyperparameters identified at small scale transfer reliably to larger models. Figure 7 validates that Pion has robust hyperparameter transferability, *i.e.*, the optimal learning rate is invariant to model scales. In fact, a broader class of μP-compatible Pion variants remain unexplored, and our results represent only an initial step.

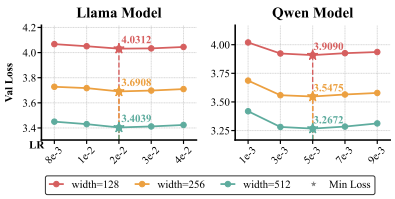


Figure 7: μP learning rate transfer across model scales.

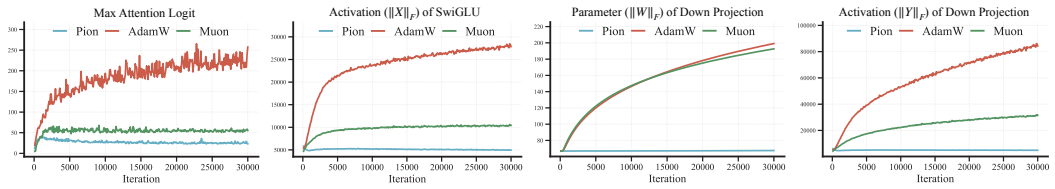


Figure 8: Dynamics of four diagnostic indicators for monitoring pretraining stability. From left to right, the four panels show, respectively, the maximum attention logit within the attention block of Layer 12, the norm of the input to the down-projection layer (equivalent to the SwiGLU activation output), the norm of the down-projection layer parameters, and the norm of its output. All indicators validate the strong stability of Pion.

3 Experiments and Results

We evaluate Pion's performance and compare it with current standard optimizers (*e.g.*, AdamW [51], Muon [37, 46]) across diverse training scenarios, including both pretraining and post-training (supervised finetuning and reinforcement learning). All experimental details can be found in Appendix D.

3.1 Stable LLM Pretraining

Main results on pretraining. We first investigate the stability of Pion in LLM pretraining. Specifically, we pretrain a LLaMA-based 1.3B model (same as [26, 63, 96]) on 54B tokens, 2× the Chinchilla-optimal data budget [33]. We ensure that the number of training tokens are sufficient for the model to converge. These tokens are sampled from the C4 corpus [67] and preprocessed using the T5-base tokenizer with sequence length 256. This setup corresponds to roughly 400K optimization iterations, effectively simulating practice long-horizon training regimes. We compare Pion (with alternate update and Lie-Lie momentum) against two most widely used optimizers, AdamW and Muon, under the same training configuration. Main results are given in Table 1. Pion achieves the best generalization performance across the evaluated benchmarks. Its validation loss is comparable to that of Muon, and both Pion and Muon yield substantially lower validation losses than AdamW. These results demonstrate that Pion not only scales effectively to large language model pretraining but also provides a superior trade-off between optimization stability and downstream capabilities.

Method	ARC-C	ARC-E	BoolQ	Hella.	PIQA	SciQ	TriviaQA	Wino.	Avg	Val Loss
AdamW	25.94	45.96	46.30	45.10	71.27	70.80	1.06	51.46	44.74	2.7700
Muon	25.34	47.94	51.56	46.70	**72.20**	71.60	1.64	**53.75**	46.34	**2.7225**
Pion	**26.79**	**49.41**	**57.58**	**47.34**	71.27	**73.40**	**2.17**	53.59	**47.69**	2.7350

Table 1: Benchmark performance and validation loss on LLaMA-1.3B. The best results are in **Bold**.

Besides validation loss, we also monitor several indicators of training stability in Figure 8. Specifically, we track the maximum attention logit [46], the norm of the SwiGLU activation $\|\boldsymbol{X}\|_F$ (also the input of the down-projection layer), the norm of down-projection weights $\|\boldsymbol{W}\|_F$, and the norm of down-projection outputs $\|\boldsymbol{Y}\|_F$, following [20, 70, 84]. In particular, the norm of the SwiGLU activation and the maximum attention logit have been generally recognized as an important stability indicator in large-scale pretraining [1, 19, 45, 77]. These results reveal a clear separation among optimizers. AdamW exhibits continuously growing attention logits and rapidly amplified activation magnitude. Muon substantially suppresses attention-logit growth, but its activations and down-projection norms keep increasing throughout training. In contrast, Pion keeps all monitored quantities nearly flat and stable. This distinctive behavior highlights Pion's potential for stable large-scale training. Pion's training stability stems from its well-preserved weight matrix spectra throughout the optimization process, as verified in Figure 9. Full results are provided in Appendix G.2.

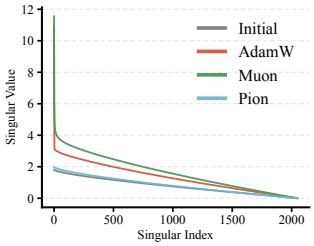


Figure 9: Weight spectrum comparison.

Normalization-free pretraining. To stress-test Pion's training stability, we remove all normalization layers from the 60M LLaMA-based model. Because normalization layers [4, 95] are widely regarded as essential for controlling activation scales and stabilizing gradient back-propagation, this challenging setting can effectively probe whether an optimizer alone can provide adequate scale regulation during pretraining. Under this setting, both AdamW and Muon can make initial progress but soon fail due to gradient overflow, producing NaNs. In contrast, Pion remains stable throughout the full 9.6B-token training run and converges successfully. These results show that spectrum-preserving updates can limit signal amplification even when explicit normalization mechanisms are removed. They further suggest that spectrum-preserving optimization can partially replace architectural scale-control mechanisms, providing an optimizer-level source of training stability.

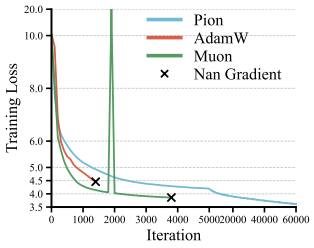


Figure 10: Normalization-free pretraining.

Pretraining on ultra-deep architectures. We further stress-test stability under LLMs with extreme depth. Training such networks often leads to severe optimization instabilities, including vanishing gradients and representation collapse [76]. To examine this setting, we scale the depth of a LLaMA 60M baseline from 8 to 200 layers and train each model on a 50B-token subset of the C4 dataset. For visual clarity, Fig. 11 reports the training loss after applying an N-step moving average, $Y_t = \frac{1}{N}\sum_{i=0}^{N-1} X_{t-i}$, where X_t and Y_t denote the raw and smoothed losses, respectively. We quantify stability by the mean standard deviation of the local loss trajectory, visualized as the integrated area of the shaded band. AdamW shows the largest loss spikes and the worst overall stability. The mean standard deviations are 0.0931 for AdamW, 0.0927 for Muon, and 0.0892 for Pion, making Pion the most stable optimizer under this setting. Pion also decreases loss more rapidly in the intermediate training stages, exhibiting an effective optimization behavior even under extreme depth.

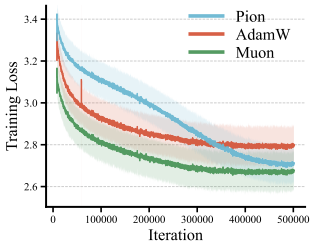


Figure 11: Training Loss of DeepNet.

To further justify our results, we analyze the layer-wise expressivity induced by different optimizers. Following the anlysis in [58, 76], we quantify each layer's local geometry using a shape-normalized Frobenius distance $\|\boldsymbol{J}_\ell - \boldsymbol{I}\|_F$, where $\boldsymbol{J}_\ell$ is the Jacobian matrix of layer ℓ. A larger Jacobian norm signifies a greater deviation from identity-like transport, thereby reflecting more effective expressivity. As shown in Figure 12, all optimizers exhibit a substantial deviation at the first layer, reflecting the architectural necessity of transforming static embeddings into highly contextualized representations. However, as the network enters the deep iterative refinement phase, the expressivity induced by different optimizers diverges significantly. AdamW exhibits a sharp Jacobian-norm drop in the middle layers, while Muon's norm steadily decays with depth. In contrast, Pion leverages the full network depth, maintaining a notably uniform norm profile across layers and consistently dominating the interior. These results suggest that Pion preserves layer-wise balanced expressivity in deep networks, avoiding the expressivity degradation observed in AdamW and Muon.

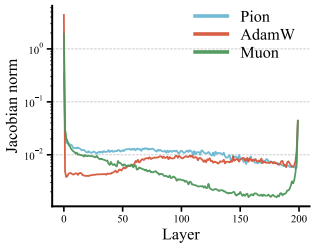


Figure 12: Jacobian Norm in DeepNet.

Method	Qwen3-1.7B						DeepSeek-R1-Distill-Qwen-1.5B					
	AIME24 (avg@32)	AIME25 (avg@32)	AMC23 (avg@8)	Minerva Math (avg@4)	Olympiad Bench (avg@8)	Avg	AIME24 (avg@32)	AIME25 (avg@32)	AMC23 (avg@8)	Minerva Math (avg@4)	Olympiad Bench (avg@8)	Avg
Base	4.06	10.10	30.27	16.27	23.67	16.87	20.52	20.83	54.06	19.39	36.20	30.20
AdamW	22.71	20.94	58.43	25.91	46.09	34.82	25.42	23.94	62.65	23.16	44.69	35.97
Muon	20.42	19.27	54.22	24.08	42.41	32.08	29.06	23.33	66.72	22.89	44.61	37.32
Pion	**25.42**	**21.98**	**59.94**	**26.84**	**46.43**	**36.12**	**30.00**	**24.38**	**66.87**	**23.90**	**46.43**	**38.32**

Table 3: Performance comparison of Pion and other baseline optimizers on RLVR tasks (training with GRPO). The metric avg@K denotes the average accuracy across K generated samples per problem. **Bold** values indicate the best overall average results.

3.2 Efficient LLM Post-training

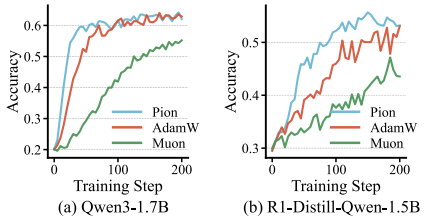
Supervised finetuning. In this section, we study the effectiveness of the Pion optimizer in the LLM supervised finetuning scenarios. Specifically, we conduct full-parameter finetuning experiments utilizing the LLaMA-Factory [97] framework. We employ Qwen2.5-1.5B [79] and Llama-3.2-3B [25] as base models, fine-tuning them on the MetaMathQA [92] and Magicoder-Evol-Instruct-110K [83] datasets. For our evaluation protocol, following the setups in [9], we analyze the stability-plasticity tradeoff [55] inherent in model adaptation. We adopt GSM8K [18] and HumanEval [16] as in-domain (ID) benchmarks for mathematical and code generation tasks respectively. Concurrently, out-of-domain (OOD) performance is assessed using ARC-Easy, ARC-Challenge [17], Winogrande [71], PIQA [10] and Hellaswag [94] benchmarks. All evaluations are conducted using the LM Evaluation Harness [23] framework with its default generation parameters. More experiment details can be found in Appendix D.3. The experimental results are summarized in Table 2. Overall, Pion achieves a highly competitive stability-plasticity tradeoff compared to established baselines such as AdamW and Muon. In particular, it shows a clear performance advantage on code generation, achieving the highest ID and OOD scores across both base models. On mathematical finetuning, Pion matches the ID performance of competing optimizers while more effectively preserving OOD capabilities, highlighting its robustness against catastrophic forgetting. This also validates the effectiveness of orthogonal finetuning [50, 65, 66] in knowledge preservation.

Method	Qwen2.5-1.5B				Llama3.2-3B			
	Math		Code		Math		Code	
	ID (%)	OOD (%)	ID (%)	OOD (%)	ID (%)	OOD (%)	ID (%)	OOD (%)
Base	59.81	64.83	35.98	63.99	25.47	67.59	26.22	53.08
AdamW	**65.88**	62.13	51.83	62.64	**59.87**	60.86	46.95	58.64
Muon	65.27	61.22	50.00	62.41	57.77	**61.20**	46.34	58.88
Pion	65.76	**62.16**	**53.05**	**63.21**	58.83	60.44	**47.19**	**59.74**

Table 2: Performance comparison of Pion and baseline optimizers on finetuning tasks. ID (in-domain) evaluates downstream capability, while OOD (out-of-domain) measures the model's robustness against forgetting. **Bold** values indicate the best results.

Reinforcement learning with verifiable reward (RLVR). We further investigate Pion as an optimizer for RLVR. Our motivation comes from recent observations [34, 98] that RLVR updates largely preserve the spectral structure of pretrained weight matrices. This finding suggests that RLVR may particularly benefit from optimizers whose update geometry is aligned with the intrinsic spectral structure of the pretrained weights. Since Pion explicitly preserves the weight spectrum throughout optimization, its spectrum-preserving update mechanism provides a natural inductive bias for RLVR. Motivated by this connection, we compare Pion with AdamW and Muon to evaluate its effectiveness in RLVR training.

We implement the RLVR training pipeline with the VeRL framework [74] and adopt Group Relative Policy Optimization (GRPO) [72] as our RL objective. Experiments are conducted on two base models, Qwen3-1.7B [87] and DeepSeek-R1-Distill-Qwen-1.5B [27], using DeepMath [30] as the training dataset. We evaluate the trained models on five mathematical reasoning benchmarks, including AIME24 [53], AIME25 [54], AMC23 [52], Minerva Math [40], and OlympiadBench [29]. The maximum generation length is set to 4096 tokens for Qwen3-1.7B and 8192 tokens for DeepSeek-R1-Distill-Qwen-1.5B. All evaluations are performed using POLARIS [3]. Detailed settings and more results are provided in Appendix D.4. As shown in Table 3, Pion achieves the best performance in both settings. These results provide empirical support for the hypothesis that spectrum-preserving optimization is particularly well suited to RLVR, where parameter updates tend to preserve the spectral structure of pretrained weights. Notably, Pion's improvements are consistent across multiple evaluation metrics and checkpoints, suggesting that its spectrum-preserving constraint provides a systematic advantage rather than an isolated performance gain. Furthermore, Figure 13 demonstrates that Pion can achieve significantly faster convergence in validation accuracy.



(a) Qwen3-1.7B (b) R1-Distill-Qwen-1.5B

Figure 13: Training dynamics of evaluation accuracy. Pion demonstrates the fastest convergence rate among all.

4 Balanced Rotations in Pion via Matrix Sign Normalization

We explore applying the matrix sign normalization (msign normalization, $\mathrm{msign}(\cdot)$) to Pion while keeping its spectrum-preserving construction. We apply msign to the skew-symmetric update generators on both sides of Pion. This keeps the generator skew-symmetric while giving it approximately unit singular values, so every active rotation plane turns by approximately the same angle η, mirroring Muon's equal-singular-value updates. The update no longer depends on the gradient scale, so the step size is set by η alone, and the RMS scaling and second-moment statistics are not needed. The variant keeps one ambient momentum buffer per matrix. We visualize the training dynamics of Pion optimizer (msign) in Fig. 14. Msign normalization is also compatible with Pion and achieves better convergence speed. This shows a promising direction for more advanced spectrum-preserving optimizer. A detailed algorithm in Appendix H.

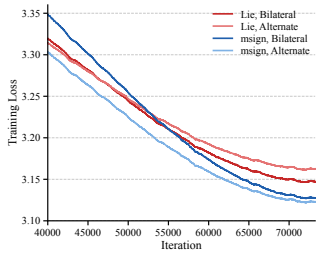


Figure 14: Performance of Pion with Lie algebra and $\mathrm{msign}(\cdot)$ on Fine-web Edu [60].

5 Related Work and Concluding Remarks

Related work. Stable LLM training relies on a delicate interplay between optimization dynamics and scale control. A common recipe combines adaptive optimizers such as AdamW [38] with normalization layers [4, 32] or scale-aware parameterizations such as maximal update parameterization [89, 90]. Recent work has further shown that matrix-aware optimization can substantially improve training stability by exploiting the structure of weight matrices [7, 8, 13, 14, 28, 37, 61, 81, 82]. In particular, Muon [37] and its variants [2, 15, 43, 46, 78, 85] have attracted considerable attention for their empirical effectiveness. Our work shares the goal of stable training, but approaches it from a distinct geometric perspective. Building on ideas from orthogonal group optimization [41, 56] and the geometric inductive biases of POET [63, 64], Pion elevates spectrum preservation to an optimizer-level principle. Rather than enforcing orthogonality through reparameterization, Pion directly preserves the spectral geometry of weight matrices during optimization. This view also offers a natural path to minimum hyperspherical energy [47–49], while retaining sufficient flexibility for effective training.

Concluding remarks. We introduce Pion, a spectrum-preserving optimizer for stable training via orthogonal equivalence transformations. Pion provably preserves the weight spectrum and maintains minimal hyperspherical energy throughout training without relying on explicit reparameterization. Empirical results have demonstrated Pion's effectiveness, making it competitive with standard optimizers across diverse settings, including pretraining, supervised finetuning, and reinforcement learning.

Acknowledgments

The work described in this paper was supported by the General Research Fund and Early Career Scheme by the Research Grants Council of Hong Kong (Project Number: 24211626). The authors declare no competing interests.

References

[1] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025. 8

[2] 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. 10

[3] Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. 9, 25

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3, 8, 10

[5] Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *NIPS*, 2017. 1

[6] Gary Bécigneul and Octavian-Eugen Ganea. Riemannian adaptive optimization methods. *arXiv preprint arXiv:1810.00760*, 2018. 4, 5

[7] Jeremy Bernstein and Laker Newhouse. Modular duality in deep learning. *arXiv preprint arXiv:2410.21265*, 2024. 10

[8] Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology. *arXiv preprint arXiv:2409.20325*, 2024. 3, 10

[9] Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. Lora learns less and forgets less. *arXiv preprint arXiv:2405.09673*, 2024. 9

[10] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. In *AAAI*, 2020. 9

[11] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018. 24

[12] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023. 2

[13] David Carlson, Volkan Cevher, and Lawrence Carin. Stochastic spectral descent for restricted boltzmann machines. In *AISTATS*, 2015. 10

[14] David Carlson, Ya-Ping Hsieh, Edo Collins, Lawrence Carin, and Volkan Cevher. Stochastic spectral descent for discrete graphical models. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):296–311, 2016. 10

[15] Lizhang Chen, Jonathan Li, and Qiang Liu. Muon optimizes under spectral norm constraints. *arXiv preprint arXiv:2506.15054*, 2025. 10

[16] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 9, 25

[17] 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. *arXiv preprint arXiv:1803.05457*, 2018. 9

[18] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. 9, 25

[19] DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. 8

[20] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *ICML*, 2023. 8

[21] Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. In *NeurIPS*, 2021. 1

[22] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011. 4

[23] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. 9, 25

[24] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, 2010. 3

[25] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 9, 24

[26] Yufei Gu and Zeke Xie. Mano: Restriking manifold optimization for llm training. *arXiv preprint arXiv:2601.23000*, 2026. 3, 4, 7

[27] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 9, 25

[28] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *ICML*, 2018. 10

[29] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, 2024. 9, 25

[30] Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025. 9, 25

[31] Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled cayley transform. In *ICML*, 2018. 5

[32] Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of EMNLP*, 2020. 1, 10

[33] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, DDL Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 10, 2022. 7

[34] Yangyi Huang, Ruotian Peng, Zeju Qiu, Jiale Kang, Yandong Wen, Bernhard Schölkopf, and Weiyang Liu. Peft-arena: Understanding parameter-efficient finetuning from a stability-plasticity perspective. *arXiv preprint arXiv:2605.28819*, 2026. 9

[35] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 3

[36] Haoming Jiang, Zhehui Chen, Minshuo Chen, Feng Liu, Dingding Wang, and Tuo Zhao. On computation and generalization of generative adversarial networks under spectrum control. In *ICLR*, 2019. 1

[37] 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. 1, 7, 10

[38] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1, 4, 10

[39] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023. 25

[40] Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022. 9, 25

[41] Mario Lezcano-Casado and David Martınez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. In *ICML*, 2019. 2, 10

[42] Jun Li, Li Fuxin, and Sinisa Todorovic. Efficient riemannian optimization on the stiefel manifold via the cayley transform. *arXiv preprint arXiv:2002.01113*, 2020. 6

[43] Zichong Li, Liming Liu, Chen Liang, Weizhu Chen, and Tuo Zhao. Normuon: Making muon more efficient and scalable. *arXiv preprint arXiv:2510.05491*, 2025. 1, 10

[44] Rongmei Lin, Weiyang Liu, Zhen Liu, Chen Feng, Zhiding Yu, James M. Rehg, Li Xiong, and Le Song. Regularizing neural networks via minimizing hyperspherical energy. In *CVPR*, 2020. 1

[45] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024. 8

[46] 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. 3, 4, 7, 8, 10

[47] Weiyang Liu, Rongmei Lin, Zhen Liu, Lixin Liu, Zhiding Yu, Bo Dai, and Le Song. Learning towards minimum hyperspherical energy. In *NeurIPS*, 2018. 1, 10

[48] Weiyang Liu, Rongmei Lin, Zhen Liu, James M. Rehg, Liam Paull, Li Xiong, Le Song, and Adrian Weller. Orthogonal over-parameterized training. In *CVPR*, 2021. 1, 6

[49] Weiyang Liu, Rongmei Lin, Zhen Liu, Li Xiong, Bernhard Schölkopf, and Adrian Weller. Learning with hyperspherical uniformity. In *AISTATS*, 2021. 1, 10

[50] Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and Bernhard Schölkopf. Parameter-efficient orthogonal finetuning via butterfly factorization. In *ICLR*, 2024. 9

[51] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 1, 7

[52] MAA. American mathematics contest 12 (amc 12), November 2023. 9, 25

[53] MAA. American invitational mathematics examination (aime), February 2024. 9, 25

[54] MAA. American invitational mathematics examination (aime), February 2025. 9, 25

[55] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, 4:504, 2013. 9

[56] Zakaria Mhammedi, Andrew Hellicar, Ashfaqur Rahman, and James Bailey. Efficient orthogonal parametrisation of recurrent neural networks using householder reflections. In *ICML*, 2017. 2, 10

[57] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. 1

[58] Dilxat Muhtar, Xinyuan Song, Sebastian Pokutta, Max Zimmer, Nico Pelleriti, Thomas Hofmann, and Shiwei Liu. When does sparsity mitigate the curse of depth in llms. *arXiv preprint arXiv:2603.15389*, 2026. 8

[59] Yurii Nesterov. A method for solving the convex programming problem with convergence rate o (1/k2). In *Dokl akad nauk Sssr*, volume 269, page 543, 1983. 4

[60] Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. In *NeurIPS*, 2024. 10

[61] Thomas Pethick, Wanyun Xie, Kimon Antonakopoulos, Zhenyu Zhu, Antonio Silveti-Falls, and Volkan Cevher. Training deep learning models with norm-constrained lmos. *arXiv preprint arXiv:2502.07529*, 2025. 10

[62] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964. 4

[63] Zeju Qiu, Simon Buchholz, Tim Z Xiao, Maximilian Dax, Bernhard Schölkopf, and Weiyang Liu. Reparameterized llm training via orthogonal equivalence transformation. In *NeurIPS*, 2025. 1, 2, 3, 6, 7, 10

[64] Zeju Qiu, Lixin Liu, Adrian Weller, Han Shi, and Weiyang Liu. Poet-x: Memory-efficient llm training by scaling orthogonal transformation. In *ICML*, 2026. 1, 2, 10

[65] Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. In *NeurIPS*, 2023. 6, 9

[66] Zeju Qiu, Weiyang Liu, Adrian Weller, and Bernhard Schölkopf. Orthogonal finetuning made scalable. In *EMNLP*, 2025. 9

[67] 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. 3, 7

[68] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019. 4

[69] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 24

[70] Oleg Rybakov, Mike Chrzanowski, Peter Dykas, Jinze Xue, and Ben Lanir. Methods of improving llm training stability. *arXiv preprint arXiv:2410.16682*, 2024. 8

[71] 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. 9

[72] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. 9, 25

[73] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020. 3

[74] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024. 9, 25

[75] Steven Thomas Smith. Optimization techniques on riemannian manifolds. *arXiv preprint arXiv:1407.5965*, 2014. 4

[76] Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. *arXiv preprint arXiv:2502.05795*, 2025. 8

[77] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. 8

[78] Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025. 1, 7, 10

[79] Qwen Team. Qwen2.5: A party of foundation models, 2024. 9, 24

[80] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 3

[81] Mark Tuddenham, Adam Prügel-Bennett, and Jonathan Hare. Orthogonalising gradients to speed up neural network optimisation. *arXiv preprint arXiv:2202.07052*, 2022. 10

[82] 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. 10

[83] Yuxiang Wei, Zhe Wang, Jiawei Liu, Yifeng Ding, and Lingming Zhang. Magicoder: Empowering code generation with oss-instruct. In *ICML*, 2024. 9, 24

[84] Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie Everett, Alex Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, et al. Small-scale proxies for large-scale transformer training instabilities. *arXiv preprint arXiv:2309.14322*, 2023. 8

[85] Tian Xie, Haoming Luo, Haoyu Tang, Yiwen Hu, Jason Klein Liu, Qingnan Ren, Yang Wang, Wayne Xin Zhao, Rui Yan, Bing Su, et al. Controlled llm training on spectral sphere. *arXiv preprint arXiv:2601.08393*, 2026. 1, 3, 7, 10

[86] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *ICML*, 2020. 3

[87] 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. 9, 25

[88] Ge Yang, Edward Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tuning large neural networks via zero-shot hyperparameter transfer. In *NeurIPS*, 2021. 6, 26

[89] Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020. 10, 26

[90] Greg Yang, James B Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023. 1, 3, 6, 10, 26

[91] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017. 1

[92] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023. 9, 24

[93] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 4

[94] 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*, 2019. 9

[95] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *NeurIPS*, 2019. 3, 8

[96] 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. 3, 7

[97] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *ACL*, 2024. 9, 24

[98] Hanqing Zhu, Zhenyu Zhang, Hanxian Huang, DiJia Su, Zechun Liu, Jiawei Zhao, Igor Fedorov, Hamed Pirsiavash, Zhizhou Sha, Jinwon Lee, et al. The path not taken: Rlvr provably learns off the principals. *arXiv preprint arXiv:2511.08567*, 2025. 9

Appendix

Table of Contents

A Geometric Structure of Pion's Update **17**
- A.1 Derivation of the Update Rule . . . 17
- A.2 Singular-Value Preservation and Planar Rotations . . . 17

B Convergence Analysis **19**
- B.1 Simplified Single-Side Analysis . . . 20
- B.2 Deterministic Bilateral Update . . . 20
- B.3 Ambient-Space Behavior of the Deterministic Update . . . 21
- B.4 Stochastic Bilateral Update . . . 22

C Additional Discussion on Computation Complexity **23**

D Experimental Details **24**
- D.1 Experiments for Design Principles . . . 24
- D.2 Pretraining . . . 24
- D.3 Supervised Fine-tuning . . . 24
- D.4 Reinforcement Learning with Verifiable Reward . . . 25

E Limitations **25**

F Compatibility with Maximal Update Parametrization **26**

G Additional Results **27**
- G.1 Another variant of Pion . . . 27
- G.2 Pretraining . . . 27
- G.3 Supervised Fine-tuning . . . 27
- G.4 Reinforcement Learning with Verifiable Reward . . . 28

H Balanced Rotations in Pion via Matrix Sign Normalization **33**

A Geometric Structure of Pion's Update

A.1 Derivation of the Update Rule

Let the objective be $f(\boldsymbol{W})$ and denote $\boldsymbol{G}_t = \nabla f(\boldsymbol{W}_t)$. Starting from $\boldsymbol{W}_t = \boldsymbol{I}_{d_{\text{out}}} \boldsymbol{W}_t \boldsymbol{I}_{d_{\text{in}}}$, we temporarily regard the two identity matrices as independent variables $\boldsymbol{Q}_{\text{out}}$ and $\boldsymbol{Q}_{\text{in}}$, and define $\widetilde{\boldsymbol{W}}(\boldsymbol{Q}_{\text{out}}, \boldsymbol{Q}_{\text{in}}) = \boldsymbol{Q}_{\text{out}} \boldsymbol{W}_t \boldsymbol{Q}_{\text{in}}$. The gradients are evaluated at $\boldsymbol{Q}_{\text{out}} = \boldsymbol{I}$ and $\boldsymbol{Q}_{\text{in}} = \boldsymbol{I}$. For infinitesimal perturbations $\Delta\boldsymbol{Q}_{\text{out}}$ and $\Delta\boldsymbol{Q}_{\text{in}}$, the corresponding perturbation of the weight matrix is $\Delta\boldsymbol{W} = \Delta\boldsymbol{Q}_{\text{out}} \boldsymbol{W}_t + \boldsymbol{W}_t \Delta\boldsymbol{Q}_{\text{in}}$. By the chain rule and the Frobenius inner product,

$$\begin{aligned} df &= \langle \boldsymbol{G}_t, \Delta\boldsymbol{W} \rangle_F = \langle \boldsymbol{G}_t, \Delta\boldsymbol{Q}_{\text{out}} \boldsymbol{W}_t \rangle_F + \langle \boldsymbol{G}_t, \boldsymbol{W}_t \Delta\boldsymbol{Q}_{\text{in}} \rangle_F \\ &= \langle \boldsymbol{G}_t \boldsymbol{W}_t^\top, \Delta\boldsymbol{Q}_{\text{out}} \rangle_F + \langle \boldsymbol{W}_t^\top \boldsymbol{G}_t, \Delta\boldsymbol{Q}_{\text{in}} \rangle_F. \end{aligned}$$

Therefore, the Euclidean gradients with respect to the two identity factors are

$$\nabla_{\boldsymbol{Q}_{\text{out}}} f = \boldsymbol{G}_t \boldsymbol{W}_t^\top, \qquad \nabla_{\boldsymbol{Q}_{\text{in}}} f = \boldsymbol{W}_t^\top \boldsymbol{G}_t. \tag{10}$$

Next, we impose the orthogonality constraint. Since $\boldsymbol{Q}_{\text{out}} \in \mathrm{O}(d_{\text{out}})$ and $\boldsymbol{Q}_{\text{in}} \in \mathrm{O}(d_{\text{in}})$, the Euclidean gradients must be projected onto the tangent spaces at the identity. The tangent space of $\mathrm{O}(k)$ at the identity is the Lie algebra $\mathfrak{so}(k) = \{\boldsymbol{\Omega} \in \mathbb{R}^{k \times k} : \boldsymbol{\Omega}^\top = -\boldsymbol{\Omega}\}$ (cf. Lemma B.2). Under the Frobenius inner product, the orthogonal projection of a square matrix $\boldsymbol{A}$ onto $\mathfrak{so}(k)$ is $\mathrm{skew}(\boldsymbol{A}) = \frac{1}{2}(\boldsymbol{A} - \boldsymbol{A}^\top)$. This gives the Lie-algebra directions $\mathrm{skew}(\boldsymbol{G}_t \boldsymbol{W}_t^\top)$ and $\mathrm{skew}(\boldsymbol{W}_t^\top \boldsymbol{G}_t)$. Eq. (1) uses the unnormalized forms

$$\boldsymbol{G}_t^{\text{out}} = \boldsymbol{G}_t \boldsymbol{W}_t^\top - (\boldsymbol{G}_t \boldsymbol{W}_t^\top)^\top = 2\mathrm{skew}(\boldsymbol{G}_t \boldsymbol{W}_t^\top), \qquad \boldsymbol{G}_t^{\text{in}} = \boldsymbol{W}_t^\top \boldsymbol{G}_t - (\boldsymbol{W}_t^\top \boldsymbol{G}_t)^\top = 2\mathrm{skew}(\boldsymbol{W}_t^\top \boldsymbol{G}_t), \tag{11}$$

which differ from the standard projections only by the constant factor 2, absorbed into the learning rate η.

Finally, $\boldsymbol{G}_t^{\text{out}} \in \mathfrak{so}(d_{\text{out}})$ and $\boldsymbol{G}_t^{\text{in}} \in \mathfrak{so}(d_{\text{in}})$, so $\exp(-\eta \boldsymbol{G}_t^{\text{out}}) \in \mathrm{O}(d_{\text{out}})$ and $\exp(-\eta \boldsymbol{G}_t^{\text{in}}) \in \mathrm{O}(d_{\text{in}})$. Applying these as the new identity factors gives

$$\boldsymbol{W}_{t+1} = \exp(-\eta \boldsymbol{G}_t^{\text{out}}) \boldsymbol{W}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}}), \tag{12}$$

which is Eq. (1). Since the identity factors are reset to $\boldsymbol{I}$ at every iteration, each step is a Riemannian gradient step on $\mathrm{O}(d_{\text{out}}) \times \mathrm{O}(d_{\text{in}})$, with the exponential map as the retraction.

A.2 Singular-Value Preservation and Planar Rotations

We provide the detailed derivation of Proposition 2.1. Recall that Pion updates a matrix parameter as

$$\boldsymbol{W}_{t+1} = \boldsymbol{R}_t \boldsymbol{W}_t \boldsymbol{P}_t, \qquad \boldsymbol{R}_t = \exp(-\eta \boldsymbol{G}_t^{\text{out}}), \quad \boldsymbol{P}_t = \exp(-\eta \boldsymbol{G}_t^{\text{in}}). \tag{13}$$

Since $\boldsymbol{G}_t^{\text{out}}$ and $\boldsymbol{G}_t^{\text{in}}$ are skew-symmetric, $\boldsymbol{R}_t$ and $\boldsymbol{P}_t$ are orthogonal. Therefore, if $\boldsymbol{W}_t = \boldsymbol{U}_t \boldsymbol{\Sigma}_0 \boldsymbol{V}_t^\top$ is a singular value decomposition, then

$$\boldsymbol{W}_{t+1} = (\boldsymbol{R}_t \boldsymbol{U}_t) \boldsymbol{\Sigma}_0 (\boldsymbol{P}_t^\top \boldsymbol{V}_t)^\top. \tag{14}$$

This is again a valid singular value decomposition up to orthogonal basis changes, so the singular values $\boldsymbol{\Sigma}_0$ are preserved. The update hence only rotates the left and right singular subspaces of $\boldsymbol{W}_t$.

This also explains why $\|\Delta\boldsymbol{W}\|_F$, with $\Delta\boldsymbol{W} = \boldsymbol{W}_{t+1} - \boldsymbol{W}_t$, measures the total strength of the update. Orthogonal multiplication preserves vector norms, so the update does not scale the row or column vectors of $\boldsymbol{W}_t$; it changes only their directions. Thus, the Frobenius norm of the displacement reflects the aggregate angular movement induced by the two rotations.

We next make the planar-rotation structure explicit. For simplicity, consider the one-sided update

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}}). \tag{15}$$

Because $\boldsymbol{G}_t^{\text{in}}$ is real skew-symmetric, there exists an orthogonal matrix $\boldsymbol{Q} \in O(d_{\text{in}})$ such that

$$\boldsymbol{G}_t^{\text{in}} = \boldsymbol{Q} \mathrm{diag}\left(\begin{pmatrix} 0 & \theta_1 \\ -\theta_1 & 0 \end{pmatrix}, \cdots, \begin{pmatrix} 0 & \theta_m \\ -\theta_m & 0 \end{pmatrix}, \cdots \right) \boldsymbol{Q}^\top = \boldsymbol{Q} \boldsymbol{\Sigma} \boldsymbol{Q}^\top. \tag{16}$$

Using $\boldsymbol{W}_t = \boldsymbol{U}_t \boldsymbol{\Sigma}_0 \boldsymbol{V}_t^\top$, we obtain

$$\boldsymbol{W}_t \exp(-\eta \boldsymbol{G}_t^{\text{in}}) = \boldsymbol{U}_t \boldsymbol{\Sigma}_0 \boldsymbol{V}_t^\top \boldsymbol{Q} \exp(-\eta \boldsymbol{\Sigma}) \boldsymbol{Q}^\top. \tag{17}$$

The matrix $\exp(-\eta\boldsymbol{\Sigma})$ is block diagonal, and each 2×2 block has the form

$$\exp\left(-\eta \begin{pmatrix} 0 & \theta_j \\ -\theta_j & 0 \end{pmatrix} \right), \tag{18}$$

which is a planar rotation with angle determined by $-\eta\theta_j$. Hence the right-side update first represents the right singular space in the orthogonal basis induced by $\boldsymbol{G}_t^{\text{in}}$, and then applies independent planar rotations within the corresponding 2D invariant subspaces.

The Frobenius and spectral norms of the Lie algebra element quantify these rotations:

$$\|-\eta \boldsymbol{G}_t^{\mathrm{in}}\|_F = \eta \sqrt{2 \sum_{j=1}^{\lfloor d_{\mathrm{in}}/2 \rfloor} \theta_j^2}, \qquad \|-\eta \boldsymbol{G}_t^{\mathrm{in}}\|_2 = \eta \max_j |\theta_j|. \tag{19}$$

Thus, $\|\boldsymbol{G}_t^{\mathrm{in}}\|_F/\sqrt{d_{\mathrm{in}}}$ characterizes the average rotation magnitude on the input side up to a constant factor, while $\|\boldsymbol{G}_t^{\mathrm{in}}\|_2$ controls the maximum angle. The same argument applies to $\boldsymbol{G}_t^{\mathrm{out}}$ for the output-side update.

B Convergence Analysis

Before presenting the convergence analysis, we first introduce several basic characterizations of the geometry induced by spectrum-preserving updates in Equation 1. As this update preserves the spectrum of the weight matrices, idealized trajectory stays on the set of matrices sharing the same singular values as the initialization. We denote this set by $\mathcal{M}_{\boldsymbol{W}_0}$.

Lemma B.1 (Characterization of the Isospectral Manifold)**.** *Let $\boldsymbol{W}_0 \in \mathbb{R}^{m\times n}$ be the initial weight matrix. The isospectral manifold passing through $\boldsymbol{W}_0$ is*

$$\mathcal{M}_{\boldsymbol{W}_0} = \{\boldsymbol{U}\boldsymbol{W}_0\boldsymbol{V}^\top \mid \boldsymbol{U} \in \mathrm{O}(m), \boldsymbol{V} \in \mathrm{O}(n)\}. \tag{20}$$

Proof. Let $\boldsymbol{W}_0 = \boldsymbol{U}_0\boldsymbol{\Sigma}\boldsymbol{V}_0^\top$ be the singular value decomposition of $\boldsymbol{W}_0$.

First, if $\boldsymbol{W} \in \mathcal{M}_{\boldsymbol{W}_0}$, then $\boldsymbol{W}$ has the same singular values as $\boldsymbol{W}_0$. Hence we can write $\boldsymbol{W} = \boldsymbol{U}_w\boldsymbol{\Sigma}\boldsymbol{V}_w^\top$. Substituting $\boldsymbol{\Sigma} = \boldsymbol{U}_0^\top\boldsymbol{W}_0\boldsymbol{V}_0$ gives

$$\boldsymbol{W} = \boldsymbol{U}_w\boldsymbol{U}_0^\top\boldsymbol{W}_0\boldsymbol{V}_0\boldsymbol{V}_w^\top = \boldsymbol{U}\boldsymbol{W}_0\boldsymbol{V}^\top, \tag{21}$$

where $\boldsymbol{U} = \boldsymbol{U}_w\boldsymbol{U}_0^\top \in \mathrm{O}(m)$ and $\boldsymbol{V} = \boldsymbol{V}_w\boldsymbol{V}_0^\top \in \mathrm{O}(n)$.

Conversely, if $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{W}_0\boldsymbol{V}^\top$ with orthogonal $\boldsymbol{U}$ and $\boldsymbol{V}$, then

$$\boldsymbol{W} = (\boldsymbol{U}\boldsymbol{U}_0)\boldsymbol{\Sigma}(\boldsymbol{V}\boldsymbol{V}_0)^\top, \tag{22}$$

which is a valid SVD with the same singular values as $\boldsymbol{W}_0$. Thus $\boldsymbol{W} \in \mathcal{M}_{\boldsymbol{W}_0}$. □

Lemma B.2 (Tangent Space of the Isospectral Manifold)**.** *For any $\boldsymbol{W} \in \mathcal{M}_{\boldsymbol{W}_0}$, the tangent space is*

$$T_{\boldsymbol{W}}\mathcal{M}_{\boldsymbol{W}_0} = \{\boldsymbol{G}_{\text{out}}\boldsymbol{W} + \boldsymbol{W}\boldsymbol{G}_{\text{in}} \mid \boldsymbol{G}_{\text{out}} \in \mathfrak{so}(m), \boldsymbol{G}_{\text{in}} \in \mathfrak{so}(n)\}, \tag{23}$$

where $\mathfrak{so}(k) = \{\boldsymbol{G} \in \mathbb{R}^{k\times k} \mid \boldsymbol{G}^\top = -\boldsymbol{G}\}$.

Lemma B.3 (First-Order Stationarity on $\mathcal{M}_{\boldsymbol{W}_0}$)**.** *Let $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ be smooth and let $\boldsymbol{G} = \nabla f(\boldsymbol{W})$. A point $\boldsymbol{W} \in \mathcal{M}_{\boldsymbol{W}_0}$ is a first-order critical point of f restricted to $\mathcal{M}_{\boldsymbol{W}_0}$ if and only if*

$$\boldsymbol{G}_{\text{in}} = \boldsymbol{W}^\top\boldsymbol{G} - \boldsymbol{G}^\top\boldsymbol{W} = \boldsymbol{0}, \qquad \boldsymbol{G}_{\text{out}} = \boldsymbol{G}\boldsymbol{W}^\top - \boldsymbol{W}\boldsymbol{G}^\top = \boldsymbol{0}. \tag{24}$$

Proof. By the first-order optimality condition on a smooth embedded manifold, $\boldsymbol{W}$ is stationary if and only if

$$\langle \boldsymbol{G}, \delta\boldsymbol{W}\rangle = 0, \qquad \forall \delta\boldsymbol{W} \in T_{\boldsymbol{W}}\mathcal{M}_{\boldsymbol{W}_0}. \tag{25}$$

Using Lemma B.2, any tangent vector can be written as $\delta\boldsymbol{W} = \boldsymbol{G}_m\boldsymbol{W} + \boldsymbol{W}\boldsymbol{G}_n$, where $\boldsymbol{G}_m \in \mathfrak{so}(m)$ and $\boldsymbol{G}_n \in \mathfrak{so}(n)$. Therefore,

$$\mathrm{Tr}\big(\boldsymbol{G}^\top(\boldsymbol{G}_m\boldsymbol{W} + \boldsymbol{W}\boldsymbol{G}_n)\big) = 0 \tag{26}$$

for all skew-symmetric $\boldsymbol{G}_m$ and $\boldsymbol{G}_n$.

Since $\boldsymbol{G}_m$ and $\boldsymbol{G}_n$ vary independently, this is equivalent to

$$\mathrm{Tr}(\boldsymbol{W}\boldsymbol{G}^\top\boldsymbol{G}_m) = 0, \quad \forall \boldsymbol{G}_m \in \mathfrak{so}(m), \tag{27}$$

and

$$\mathrm{Tr}(\boldsymbol{G}^\top\boldsymbol{W}\boldsymbol{G}_n) = 0, \quad \forall \boldsymbol{G}_n \in \mathfrak{so}(n). \tag{28}$$

The orthogonal complement of skew-symmetric matrices under the Frobenius inner product is the space of symmetric matrices. Hence $\boldsymbol{W}\boldsymbol{G}^\top$ and $\boldsymbol{G}^\top\boldsymbol{W}$ must be symmetric, which gives

$$\boldsymbol{G}\boldsymbol{W}^\top - \boldsymbol{W}\boldsymbol{G}^\top = \boldsymbol{0}, \qquad \boldsymbol{W}^\top\boldsymbol{G} - \boldsymbol{G}^\top\boldsymbol{W} = \boldsymbol{0}. \tag{29}$$

This proves the claim. □

Lemma B.3 shows that it is sufficient to prove

$$\|\boldsymbol{G}_t^{\text{in}}\|_F \to 0, \qquad \|\boldsymbol{G}_t^{\text{out}}\|_F \to 0. \tag{30}$$

For the bilateral update, we therefore use the combined stationarity measure

$$\mathcal{S}_t = \|\boldsymbol{G}_t^{\text{in}}\|_F^2 + \|\boldsymbol{G}_t^{\text{out}}\|_F^2. \tag{31}$$

Showing $\mathcal{S}_t \to 0$ directly implies first-order convergence on the isospectral manifold.

Assumption B.4 (L-smoothness)**.** The objective function f is L-smooth in the Euclidean sense, i.e.,

$$f(\boldsymbol{W}_2) \le f(\boldsymbol{W}_1) + \langle \nabla f(\boldsymbol{W}_1), \boldsymbol{W}_2 - \boldsymbol{W}_1\rangle + \frac{L}{2}\|\boldsymbol{W}_2 - \boldsymbol{W}_1\|_F^2. \tag{32}$$

Assumption B.5 (Lower boundedness)**.** There exists $f_{\text{inf}} \in \mathbb{R}$ such that $f(\boldsymbol{W}) \ge f_{\text{inf}}$ for all $\boldsymbol{W}$.

Assumption B.6 (Boundedness along the trajectory)**.** Along the trajectory, there exist constants $\gamma, G, B > 0$ such that

$$\|\boldsymbol{W}_t\|_2 \le \gamma, \qquad \|\nabla f(\boldsymbol{W}_t)\|_F \le G, \qquad \|\boldsymbol{G}_t^{\text{in}}\|_F, \|\boldsymbol{G}_t^{\text{out}}\|_F \le B. \tag{33}$$

Assumptions B.4 and B.5 are standard in first-order convergence analysis. Assumption B.6 is mild in our setting because the ideal spectrum-preserving dynamics remains on a compact isospectral manifold, and the truncated update stays in a bounded neighborhood for sufficiently small step size.

B.1 Simplified Single-Side Analysis

We first briefly revisit the single-side update, since it provides the key descent identity used later for the bilateral update. Consider the in-side update

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t \exp^{(2)}(-\eta \boldsymbol{G}_t^{\text{in}}), \qquad \boldsymbol{G}_t^{\text{in}} = \boldsymbol{W}_t^\top \boldsymbol{G}_t - \boldsymbol{G}_t^\top \boldsymbol{W}_t. \tag{34}$$

Expanding the truncated exponential gives

$$\boldsymbol{W}_{t+1} - \boldsymbol{W}_t = -\eta \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} + \frac{\eta^2}{2} \boldsymbol{W}_t (\boldsymbol{G}_t^{\text{in}})^2. \tag{35}$$

Let $\boldsymbol{S}_t = \boldsymbol{W}_t^\top \boldsymbol{G}_t$. Since $\boldsymbol{G}_t^{\text{in}} = \boldsymbol{S}_t - \boldsymbol{S}_t^\top$, we have

$$\left\langle \boldsymbol{G}_t, \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} \right\rangle = \operatorname{Tr}(\boldsymbol{G}_t^\top \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}}) = \frac{1}{2} \|\boldsymbol{G}_t^{\text{in}}\|_F^2. \tag{36}$$

Therefore, the first-order part of the update gives the descent term

$$\left\langle \boldsymbol{G}_t, -\eta \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} \right\rangle = -\frac{\eta}{2} \|\boldsymbol{G}_t^{\text{in}}\|_F^2. \tag{37}$$

The out-side update gives the analogous identity

$$\left\langle \boldsymbol{G}_t, -\eta \boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t \right\rangle = -\frac{\eta}{2} \|\boldsymbol{G}_t^{\text{out}}\|_F^2. \tag{38}$$

Thus, both in-side and out-side rotations are aligned with the Riemannian descent direction. The remaining second-order terms can be controlled by smoothness and boundedness, yielding convergence for the alternating version. Since the bilateral update simply applies both descent directions in the same step, we next analyze it directly.

B.2 Deterministic Bilateral Update

We consider the simplified bilateral Pion update

$$\boldsymbol{W}_{t+1} = \exp^{(2)}(-\eta \boldsymbol{G}_t^{\text{out}}) \boldsymbol{W}_t \exp^{(2)}(-\eta \boldsymbol{G}_t^{\text{in}}). \tag{39}$$

Expanding the update gives

$$\boldsymbol{W}_{t+1} - \boldsymbol{W}_t = -\eta \left(\boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t + \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} \right) + \boldsymbol{R}_t, \tag{40}$$

where the remainder $\boldsymbol{R}_t$ contains all terms of order η^2 and higher:

$$\begin{aligned} \boldsymbol{R}_t = {} & \frac{\eta^2}{2} (\boldsymbol{G}_t^{\text{out}})^2 \boldsymbol{W}_t + \eta^2 \boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} + \frac{\eta^2}{2} \boldsymbol{W}_t (\boldsymbol{G}_t^{\text{in}})^2 \\ & - \frac{\eta^3}{2} (\boldsymbol{G}_t^{\text{out}})^2 \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} - \frac{\eta^3}{2} \boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t (\boldsymbol{G}_t^{\text{in}})^2 + \frac{\eta^4}{4} (\boldsymbol{G}_t^{\text{out}})^2 \boldsymbol{W}_t (\boldsymbol{G}_t^{\text{in}})^2. \end{aligned} \tag{41}$$

Using Assumption B.6, there exists a constant $K_\eta = O(\eta^2)$ such that

$$\|\boldsymbol{R}_t\|_F \le K_\eta \mathcal{S}_t, \qquad K_\eta = \gamma \left(\eta^2 + \frac{B\eta^3}{2} + \frac{B^2 \eta^4}{8} \right). \tag{42}$$

By L-smoothness,

$$f(\boldsymbol{W}_{t+1}) - f(\boldsymbol{W}_t) \le \langle \boldsymbol{G}_t, \boldsymbol{W}_{t+1} - \boldsymbol{W}_t \rangle + \frac{L}{2} \|\boldsymbol{W}_{t+1} - \boldsymbol{W}_t\|_F^2. \tag{43}$$

Substituting the bilateral expansion and using the descent identities from the single-side analysis, we obtain

$$\left\langle \boldsymbol{G}_t, -\eta (\boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t + \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}}) \right\rangle = -\frac{\eta}{2} \mathcal{S}_t. \tag{44}$$

The remainder satisfies

$$\langle \boldsymbol{G}_t, \boldsymbol{R}_t \rangle \le \|\boldsymbol{G}_t\|_F \|\boldsymbol{R}_t\|_F \le G K_\eta \mathcal{S}_t. \tag{45}$$

For the quadratic term,

$$\begin{aligned} \|\boldsymbol{W}_{t+1} - \boldsymbol{W}_t\|_F^2 &\le 2\eta^2 \left\| \boldsymbol{G}_t^{\text{out}} \boldsymbol{W}_t + \boldsymbol{W}_t \boldsymbol{G}_t^{\text{in}} \right\|_F^2 + 2 \|\boldsymbol{R}_t\|_F^2 \\ &\le 4\eta^2 \gamma^2 \mathcal{S}_t + 2 K_\eta^2 \mathcal{S}_t^2 \\ &\le \left(4\eta^2 \gamma^2 + 4 K_\eta^2 B^2 \right) \mathcal{S}_t, \end{aligned} \tag{46}$$

where the last inequality uses $\mathcal{S}_t \le 2B^2$. Combining the above inequalities yields

$$f(\boldsymbol{W}_{t+1}) - f(\boldsymbol{W}_t) \le -c_\eta \mathcal{S}_t, \tag{47}$$

where

$$c_\eta = \frac{\eta}{2} - GK_\eta - 2L\eta^2\gamma^2 - 2LK_\eta^2 B^2. \tag{48}$$

Since $K_\eta = O(\eta^2)$, we have $c_\eta > 0$ for sufficiently small η.

Summing over $t = 0,\ldots,T-1$ gives

$$f(\boldsymbol{W}_T) - f(\boldsymbol{W}_0) \le -c_\eta \sum_{t=0}^{T-1} \mathcal{S}_t. \tag{49}$$

Using $f(\boldsymbol{W}_T) \ge f_{\inf}$, we obtain

$$\sum_{t=0}^{T-1} \mathcal{S}_t \le \frac{f(\boldsymbol{W}_0) - f_{\inf}}{c_\eta}. \tag{50}$$

Therefore,

$$\min_{0\le t<T} \left(\|\boldsymbol{G}_t^{\text{in}}\|_F^2 + \|\boldsymbol{G}_t^{\text{out}}\|_F^2 \right) \le \frac{f(\boldsymbol{W}_0) - f_{\inf}}{c_\eta T}. \tag{51}$$

By Lemma B.3, this implies convergence to a first-order stationary point on the isospectral manifold.

Theorem B.7 (Deterministic Convergence of Simplified Bilateral Pion)**.** *Under Assumptions B.4–B.6, suppose the learning rate η is sufficiently small such that $c_\eta > 0$. Then the simplified bilateral Pion update satisfies*

$$\min_{0\le t<T} \left(\|\boldsymbol{G}_t^{\text{in}}\|_F^2 + \|\boldsymbol{G}_t^{\text{out}}\|_F^2 \right) \le \frac{f(\boldsymbol{W}_0) - f_{\inf}}{c_\eta T}. \tag{52}$$

In particular, the stationarity measure converges at rate $\mathcal{O}(\frac{1}{T})$.

B.3 Ambient-Space Behavior of the Deterministic Update

Theorems 2.2 and B.7 control the Lie-algebra gradients, which measure stationarity on $\mathcal{M}_{\boldsymbol{W}_0}$ only. This subsection records what the deterministic analysis implies for the ambient gradient $\nabla f(\boldsymbol{W}_t)$. Here L-smoothness is understood in the usual Lipschitz-gradient sense, $\|\nabla f(\boldsymbol{X}) - \nabla f(\boldsymbol{Y})\|_F \le L\|\boldsymbol{X} - \boldsymbol{Y}\|_F$, which implies the quadratic upper bound in Assumption B.4.

Proposition B.8 (Ambient-space behavior of deterministic Pion)**.** *Consider the simplified deterministic bilateral update of Appendix B.2 under Assumptions B.5 and B.6. Assume ∇f is L-Lipschitz and η is small enough that $c_\eta > 0$, and let $\mathcal{S}_t = \|\boldsymbol{G}_t^{\text{in}}\|_F^2 + \|\boldsymbol{G}_t^{\text{out}}\|_F^2$. Then*

1. $\sum_{t=0}^{\infty} \mathcal{S}_t \le \dfrac{f(\boldsymbol{W}_0) - f_{\inf}}{c_\eta} < \infty$, *and hence* $\mathcal{S}_t \to 0$.
2. $\|\nabla f(\boldsymbol{W}_{t+1}) - \nabla f(\boldsymbol{W}_t)\|_F \le L\left(\sqrt{2}\eta\gamma\sqrt{\mathcal{S}_t} + K_\eta \mathcal{S}_t\right) \to 0$.
3. *The gradient itself need not vanish: there are instances satisfying all the assumptions with $\mathcal{S}_t = 0$ for all t but $\|\nabla f(\boldsymbol{W}_t)\|_F \ne 0$ for all t (Remark B.10).*

Proof. (i) The descent inequality $f(\boldsymbol{W}_{t+1}) - f(\boldsymbol{W}_t) \le -c_\eta \mathcal{S}_t$ was established in the proof of Theorem B.7. Summing over $t = 0,\ldots,T-1$ and using $f(\boldsymbol{W}_T) \ge f_{\inf}$ gives $\sum_{t=0}^{T-1} \mathcal{S}_t \le (f(\boldsymbol{W}_0) - f_{\inf})/c_\eta$ for every T. Letting $T \to \infty$ proves (i).

(ii) The expansion in Appendix B.2 gives $\boldsymbol{W}_{t+1} - \boldsymbol{W}_t = -\eta(\boldsymbol{G}_t^{\text{out}}\boldsymbol{W}_t + \boldsymbol{W}_t\boldsymbol{G}_t^{\text{in}}) + \boldsymbol{R}_t$ with $\|\boldsymbol{R}_t\|_F \le K_\eta \mathcal{S}_t$. Since $\|\boldsymbol{W}_t\|_2 \le \gamma$,

$$\|\boldsymbol{G}_t^{\text{out}}\boldsymbol{W}_t + \boldsymbol{W}_t\boldsymbol{G}_t^{\text{in}}\|_F \le \gamma\left(\|\boldsymbol{G}_t^{\text{out}}\|_F + \|\boldsymbol{G}_t^{\text{in}}\|_F\right) \le \sqrt{2}\gamma\sqrt{\mathcal{S}_t},$$

so $\|\boldsymbol{W}_{t+1} - \boldsymbol{W}_t\|_F \le \sqrt{2}\eta\gamma\sqrt{\mathcal{S}_t} + K_\eta \mathcal{S}_t$. The Lipschitz property of ∇f gives the bound in (ii), whose right-hand side tends to zero by (i).

(iii) This is the example in Remark B.10. □

Remark B.9 (Stochastic setting). Part (i) relies on the deterministic telescoping. For the stochastic update, Theorem B.11 only gives $\min_{0\le t<T} \mathbb{E}[\mathcal{S}_t] = \mathcal{O}(T^{-1/2})$, which does not imply summability of $\mathcal{S}_t$. Hence the statement that the variation of the ambient gradient vanishes is claimed only for the deterministic update.

Remark B.10 (The ambient gradient need not vanish). Let $f(\boldsymbol{W}) = \frac{1}{2}\|\boldsymbol{W} - 2\boldsymbol{I}_n\|_F^2$ and $\boldsymbol{W}_0 = \boldsymbol{I}_n$, so that $\mathcal{M}_{\boldsymbol{W}_0} = \mathrm{O}(n)$ by Lemma B.1. The gradient $\nabla f(\boldsymbol{W}) = \boldsymbol{W} - 2\boldsymbol{I}_n$ is 1-Lipschitz and $f \geq 0$. At $\boldsymbol{W} = \boldsymbol{I}_n$ we have $\nabla f = -\boldsymbol{I}_n$, so $\boldsymbol{G}^{\mathrm{in}} = \boldsymbol{W}^\top \nabla f - \nabla f^\top \boldsymbol{W} = \boldsymbol{0}$ and $\boldsymbol{G}^{\mathrm{out}} = \boldsymbol{0}$. Hence $\boldsymbol{W}_1 = \exp^{(2)}(\boldsymbol{0})\boldsymbol{I}_n \exp^{(2)}(\boldsymbol{0}) = \boldsymbol{I}_n$, and by induction $\boldsymbol{W}_t = \boldsymbol{I}_n$, $\mathcal{S}_t = 0$ and $\|\nabla f(\boldsymbol{W}_t)\|_F = \sqrt{n}$ for all t. By Lemma B.3, $\boldsymbol{W}_0$ is a first-order critical point of f on $\mathcal{M}_{\boldsymbol{W}_0}$. It is in fact the global minimizer there: for $\boldsymbol{W} \in \mathrm{O}(n)$ we have $\|\boldsymbol{W}\|_F^2 = n$, so $f(\boldsymbol{W}) = \frac{1}{2}(5n - 4\mathrm{tr}\boldsymbol{W})$, and $\mathrm{tr}\boldsymbol{W} \leq n$ with equality only at $\boldsymbol{W} = \boldsymbol{I}_n$, since the eigenvalues of an orthogonal matrix lie on the unit circle. Nevertheless $\nabla f(\boldsymbol{W}_0) \neq \boldsymbol{0}$, so $\boldsymbol{W}_0$ is not a critical point of f on the full space. The unconstrained minimizer $2\boldsymbol{I}_n$ has singular values $2 \neq 1$, so it lies outside $\mathcal{M}_{\boldsymbol{W}_0}$ and is never reached.

B.4 Stochastic Bilateral Update

We now consider the stochastic setting. Let

$$\tilde{\boldsymbol{G}}_t = \boldsymbol{G}_t + \boldsymbol{\xi}_t \tag{53}$$

be an unbiased mini-batch gradient estimator, where $\mathbb{E}_t[\boldsymbol{\xi}_t] = \boldsymbol{0}$ and $\mathbb{E}_t[\|\boldsymbol{\xi}_t\|_F^2] \leq \sigma^2$. Define

$$\tilde{\boldsymbol{G}}_t^{\mathrm{in}} = \boldsymbol{W}_t^\top \tilde{\boldsymbol{G}}_t - \tilde{\boldsymbol{G}}_t^\top \boldsymbol{W}_t, \qquad \tilde{\boldsymbol{G}}_t^{\mathrm{out}} = \tilde{\boldsymbol{G}}_t \boldsymbol{W}_t^\top - \boldsymbol{W}_t \tilde{\boldsymbol{G}}_t^\top. \tag{54}$$

By linearity and unbiasedness,

$$\mathbb{E}_t[\tilde{\boldsymbol{G}}_t^{\mathrm{in}}] = \boldsymbol{G}_t^{\mathrm{in}}, \qquad \mathbb{E}_t[\tilde{\boldsymbol{G}}_t^{\mathrm{out}}] = \boldsymbol{G}_t^{\mathrm{out}}. \tag{55}$$

Moreover, the induced stochastic noise satisfies

$$\begin{aligned}\mathbb{E}_t\Big[\|\tilde{\boldsymbol{G}}_t^{\mathrm{in}} - \boldsymbol{G}_t^{\mathrm{in}}\|_F^2 + \|\tilde{\boldsymbol{G}}_t^{\mathrm{out}} - \boldsymbol{G}_t^{\mathrm{out}}\|_F^2\Big] \\ \leq 8\gamma^2\sigma^2 \triangleq \sigma_\Omega^2.\end{aligned} \tag{56}$$

Therefore,

$$\mathbb{E}_t\Big[\|\tilde{\boldsymbol{G}}_t^{\mathrm{in}}\|_F^2 + \|\tilde{\boldsymbol{G}}_t^{\mathrm{out}}\|_F^2\Big] \leq \mathcal{S}_t + \sigma_\Omega^2. \tag{57}$$

Applying the deterministic bilateral descent argument to the stochastic update and taking conditional expectation gives

$$\mathbb{E}_t[f(\boldsymbol{W}_{t+1})] - f(\boldsymbol{W}_t) \leq -\frac{\eta}{2}\mathcal{S}_t + a_\eta(\mathcal{S}_t + \sigma_\Omega^2), \tag{58}$$

where

$$a_\eta = GK_\eta + 2L\eta^2\gamma^2 + 2LK_\eta^2 B^2. \tag{59}$$

Equivalently,

$$\mathbb{E}_t[f(\boldsymbol{W}_{t+1})] - f(\boldsymbol{W}_t) \leq -c_\eta \mathcal{S}_t + a_\eta\sigma_\Omega^2, \qquad c_\eta = \frac{\eta}{2} - a_\eta. \tag{60}$$

Taking total expectation and summing over $t = 0, \ldots, T-1$, we obtain

$$c_\eta \sum_{t=0}^{T-1} \mathbb{E}[\mathcal{S}_t] \leq f(\boldsymbol{W}_0) - f_{\inf} + T a_\eta \sigma_\Omega^2. \tag{61}$$

Thus,

$$\min_{0 \leq t < T} \mathbb{E}[\mathcal{S}_t] \leq \frac{f(\boldsymbol{W}_0) - f_{\inf}}{c_\eta T} + \frac{a_\eta}{c_\eta}\sigma_\Omega^2. \tag{62}$$

Choosing $\eta = C/\sqrt{T}$ with sufficiently small C gives the standard stochastic nonconvex rate

$$\min_{0 \leq t < T} \mathbb{E}\big[\|\boldsymbol{G}_t^{\mathrm{in}}\|_F^2 + \|\boldsymbol{G}_t^{\mathrm{out}}\|_F^2\big] = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right). \tag{63}$$

Theorem B.11 (Stochastic Convergence of Simplified Bilateral Pion)**.** *Under Assumptions B.4–B.6, assume the stochastic gradient is unbiased and has bounded variance. Let $\eta = C/\sqrt{T}$ with sufficiently small $C > 0$. Then*

$$\min_{0 \leq t < T} \mathbb{E}\big[\|\boldsymbol{G}_t^{\mathrm{in}}\|_F^2 + \|\boldsymbol{G}_t^{\mathrm{out}}\|_F^2\big] = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right). \tag{64}$$

By Lemma B.3, this implies convergence to a stochastic first-order stationary neighborhood on the isospectral manifold.

C Additional Discussion on Computation Complexity

Alternate Update. When alternate update is enabled, only one exponential map is applied at each step. On an output-side step, the update-side cost becomes $2d_{\text{out}}^2 d_{\text{in}}$ for RMS scaling, $\mathcal{O}(d_{\text{out}}^3)$ for forming the second-order term, and $2d_{\text{out}}^2 d_{\text{in}}$ for multiplying the exponential approximation with $\boldsymbol{W}$. On an input-side step, the corresponding cost is $2d_{\text{out}} d_{\text{in}}^2 + \mathcal{O}(d_{\text{in}}^3) + 2d_{\text{out}} d_{\text{in}}^2$. Averaged over two consecutive steps, the dominant update-side cost is therefore reduced from $4d_{\text{out}}^2 d_{\text{in}} + 4d_{\text{out}} d_{\text{in}}^2 + \mathcal{O}(d_{\text{out}}^3 + d_{\text{in}}^3)$ to approximately $2d_{\text{out}}^2 d_{\text{in}} + 2d_{\text{out}} d_{\text{in}}^2 + \mathcal{O}\left(\frac{d_{\text{out}}^3 + d_{\text{in}}^3}{2}\right)$. Thus, alternate update reduces the update-side computation by roughly a factor of two.

Memory Analysis. The persistent optimizer states consist of the first- and second-moment buffers on both Lie algebras: $\boldsymbol{m}^{\text{in}}, \boldsymbol{v}^{\text{in}} \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}}$ and $\boldsymbol{m}^{\text{out}}, \boldsymbol{v}^{\text{out}} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$. Thus, the persistent memory overhead is $2(d_{\text{in}}^2 + d_{\text{out}}^2)$ floating-point numbers, excluding the parameter matrix and its backpropagated gradient. In comparison, Adam-style optimizers store $2d_{\text{out}} d_{\text{in}}$ floating-point numbers for the same weight matrix. Therefore, the relative persistent-state overhead of Pion compared with Adam is $(d_{\text{out}}^2 + d_{\text{in}}^2)/(d_{\text{out}} d_{\text{in}})$. For nearly square matrices this is a small constant factor, while for highly rectangular matrices it increases with the aspect ratio.

The transient memory is dominated by the Lie algebra gradients, the normalized algebra directions, the squared Lie matrices used in $\mathcal{E}_2$, and temporary products with $\boldsymbol{W}$. With buffer reuse or in-place construction of $\boldsymbol{A}^{\text{in}}$ and $\boldsymbol{A}^{\text{out}}$, the additional peak temporary memory is $\mathcal{O}(d_{\text{in}}^2 + d_{\text{out}}^2 + d_{\text{out}} d_{\text{in}})$. A naive implementation that materializes both sides of the RMS-scaling term $\boldsymbol{A}^{\text{out}}\boldsymbol{W}$ and $\boldsymbol{W}\boldsymbol{A}^{\text{in}}$ may require two extra $d_{\text{out}} \times d_{\text{in}}$ buffers, but this can be reduced by accumulating the Frobenius norm with buffer reuse.

Practical Cost. We further report the practical memory and runtime cost under the same training configuration. All experiments are conducted on 8 NVIDIA H100 GPUs connected with NVLink. We use distributed data parallelism (DDP) without gradient accumulation, and report the peak memory usage per GPU together with the wall-clock time per training step.

Method	**AdamW**	**Muon**	**Pion**	**Pion w/o 2nd Moment**
Memory Usage per GPU	51593 MB	47251 MB	59839 MB	45289 MB
Time Usage per Step	0.3932 s	0.5505 s	0.5679 s	0.5678 s

Table 4: Practical memory and runtime cost of AdamW, Muon, and Pion. All experiments are run on 8 H100 GPUs with NVLink using DDP, without gradient accumulation. Memory is reported as peak per-GPU memory usage.

The measured results are consistent with the theoretical analysis. Full Pion uses 59,839 MB per GPU, which is about 16.0% higher than AdamW and 26.6% higher than Muon. This increase mainly comes from the additional input- and output-side Lie algebra states and temporary matrix products. However, the variant without second-moment buffers reduces the memory usage to 45289 MB per GPU, which is 24.3% lower than full Pion, 12.2% lower than AdamW, and 4.2% lower than Muon. This shows that the second-moment Lie algebra buffers are the major source of Pion's persistent memory overhead. Fortunately, Pion without second-order moment achieves only slightly worse performance compared to the full Pion.

In terms of runtime, Pion takes 0.5679 seconds per step, compared with 0.3932 seconds for AdamW and 0.5505 seconds for Muon. Thus, Pion is about 44.4% slower than AdamW, but only about 3.2% slower than Muon. Removing the second-moment buffers does not noticeably change the step time in this setting, indicating that the runtime is dominated by matrix multiplications and the second-order exponential approximation rather than element-wise moment updates. Overall, Pion introduces moderate additional runtime over Muon while providing a structured bilateral Lie-algebra update, and its memory footprint can be substantially reduced by removing the second-moment buffers.

D Experimental Details

In this section, we provide the detailed experimental setups and configurations for each experiment presented in the main manuscript.

D.1 Experiments for Design Principles

All experiments are conducted in the Megatron-LM codebase with bf16 mixed-precision.

Shared settings. Unless otherwise specified, we adopt a second-order approximation to the matrix exponential due to its strong spectrum-preserving property. We use the T5-base tokenizer for text tokenization. We use a batch size of 512 for all experiments. We use a cosine decay schedule for the learning rate, decaying to 0.01 times the initial value. The initial learning rate is selected from $\{$1e-4,5e-4,1e-3,5e-3,1e-2$\}$, and we observe that lr $=$ 1e-3 consistently achieves the best performance. Therefore, results in the main text are reported with this value. In implementation, the spectrum-preserving updates are applied in a per-head manner (e.g., per attention head). For the language modeling embedding and head layers, we employ AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay$= 0.1$) for updates. Pion is applied exclusively to two-dimensional parameter matrices, and no weight decay is imposed on these Pion-updated parameters.

Consistent Update. For bilateral normalization, independently normalizing G_t^{out} and G_t^{in} in Eq. (3) alters the overall update norm and would require additional learning-rate tuning. To isolate the effect of normalization itself, we rescale the resulting Lie algebra update to approximately match the norm of the original update, thereby avoiding extra hyperparameter adjustments.

Momentum. We observe that directly applying momentum without RMS scaling is only stable under extremely small learning rates. Therefore, we apply RMS scaling to all variants and evaluate them under a unified setting. The RMS scaling coefficient is set to 0.2. For all variants, we set the first-order momentum coefficient to $\beta_1 = 0.9$ and the second-order momentum coefficient to $\beta_2 = 0.95$.

Alternate Update. The settings follow those of the momentum experiments. For the Lie+Lie variant, we find that disabling momentum accumulation on the non-updated side degrades convergence, despite reducing gradient computation. This aligns with classical stochastic optimization theory, where reducing the effective sample size for first-order moment estimation increases variance [11, 69]. Therefore, even under alternating updates, we maintain momentum accumulation on both sides and alternate only the parameter updates.

Approximation of $\exp(\cdot)$. The settings follow the shared configuration above. For the Cayley transform, we adopt the standard formulation $(\boldsymbol{I} - \frac{1}{2}\boldsymbol{S})^{-1}(\boldsymbol{I} + \frac{1}{2}\boldsymbol{S})$, where $\mathbf{S}$ is a square matrix. In implementation, computations are performed in float32 for numerical stability, and the corresponding linear system is solved via `torch.linalg.solve`.

D.2 Pretraining

The pretraining experiments are conducted in the Megatron-LM codebase using bfloat16 precision. We use the T5-base tokenizer for text tokenization. We follow standard settings and adopt a cosine decay schedule for the learning rate, decaying to 0.01 times the maximum learning rate. The maximum learning rate is set to 5e-4 for all methods, which is chosen based on the use of the AdamW optimizer. For both Muon and Pion, we set the RMS scaling coefficient to 0.2. For optimizer hyperparameters, we use $\beta_1 = 0.9$ and $\beta_2 = 0.95$ for AdamW and Pion. For Muon, we set $\beta_1 = 0.95$. For attention updates, Muon adopts a split-head strategy, where each attention head is updated independently. For Pion, due to memory constraints, we apply updates separately to query, key and value matrices. We use a batch size of 512 for all experiments. For both Muon and Pion, the embedding layers, output head, and bias vectors are updated using AdamW. We set a unified weight decay coefficient of 0.1 for AdamW and Muon. In contrast, for the parameters updated by Pion, we do not apply weight decay.

D.3 Supervised Fine-tuning

Models and Datasets. We employ Qwen2.5-1.5B [79] and Llama-3.2-3B [25] as our base models, fine-tuning them on the MetaMathQA [92] and Magicoder-Evol-Instruct-110K [83] datasets. To maintain a uniform computational budget across both domains, we limit each dataset to 50K samples.

Training Details. All training experiments are conducted within the LLaMA-Factory [97] framework and run on NVIDIA H200 GPUs. To ensure a fair comparison, all methods are trained for 3 epochs with a global batch size of 64, a learning rate of 1×10^{-5}, a cutoff length of 4096 tokens, and using

the default FP32 numerical precision for optimizer updates. Our implementation of Pion optimizer deliberately omits the second-order momentum and employ a bilateral update strategy. Furthermore, we split the attention heads and the FFN gate/up projections for separate optimization.

Evaluation Protocols. All evaluations are conducted using the LM Evaluation Harness [23] framework with its default generation parameters. Specifically, we evaluate GSM8K [18] in a 5-shot setting, while all other benchmarks are evaluated zero-shot. For generation-based tasks, we apply deterministic greedy decoding. Performance on the HumanEval [16] benchmark is reported using the pass@1 metric.

D.4 Reinforcement Learning with Verifiable Reward

Models and Datasets. Experiments are conducted on two base models, Qwen3-1.7B [87] and DeepSeek-R1-Distill-Qwen-1.5B [27], using DeepMath [30] as the training dataset. The maximum context length is set to 4096 for Qwen3-1.7B and 8192 for DeepSeek-R1-Distill-Qwen-1.5B.

Training Details. We implement our RLVR training pipeline using the VeRL [74] framework, utilizing vLLM [39] for efficient rollouts and adopting GRPO [72] algorithm. All experiments run on NVIDIA H200 GPUs. To ensure a fair comparison across all methods, we fix the learning rate at 1×10^{-6}, sample 12 rollouts per prompt, and use the default FP32 numerical precision for optimizer updates. Regarding model-specific configurations, Qwen3-1.7B is trained for 400 steps with both the global batch size and on-policy minibatch size set to 128. In contrast, DeepSeek-R1-Distill-Qwen-1.5B is trained for 781 steps, with both batch sizes configured to 64. Our Pion implementation operates without second-order momentum and uses an alternating update strategy. Furthermore, we split the attention heads and the FFN gate/up projections for separate optimization.

Evaluation Protocols. We evaluate our models on five widely used benchmarks: AIME24 [53], AIME25 [54], AMC23 [52], Minerva Math [40], and OlympiadBench [29]. The maximum generation length is set to 4,096 tokens for Qwen3-1.7B and 8,192 tokens for DeepSeek-R1-Distill-Qwen-1.5B. To ensure robust evaluation, we report the averaged accuracy across multiple independent samples for each benchmark. Specifically, we average the accuracy over 32 samples for AIME24 and AIME25, 8 samples for AMC23 and OlympiadBench, and 4 samples for Minerva Math. All evaluations are conducted within the POLARIS [3] framework with a decoding temperature of 1.0, top-k sampling with $k = 20$, and top-p sampling with $p = 0.8$.

E Limitations

While Pion demonstrates exceptional stability and competitive performance, it introduces certain computational and memory overheads. First, computing the Lie-algebra gradients and applying the truncated matrix exponential mapping requires additional FLOPs. However, this overhead is largely amortized in standard LLM pretraining regimes where large token batches are used, making the practical impact on wall-clock time minimal. Second, accumulating momentum directly in the Lie algebra moderately increases the optimizer's memory footprint. Nevertheless, our framework is highly flexible: in strictly memory-constrained settings, components like second-order momentum can be seamlessly omitted while still retaining the core benefits of spectrum preservation. Finally, scaling Pion's empirical evaluation to larger models remains an important direction for future work.

F Compatibility with Maximal Update Parametrization

In this section, we study whether the maximal update parametrization (μP) framework [88, 89] holds under the Pion optimizer. [90] shows that μP can be expressed through strict spectral norm constraints on the weight matrices and their updates: $\|\boldsymbol{W}\|_2 = \Theta\left(\sqrt{\frac{d_{\text{out}}}{d_{\text{in}}}}\right)$ and $\|\Delta\boldsymbol{W}\|_2 = \Theta\left(\sqrt{\frac{d_{\text{out}}}{d_{\text{in}}}}\right)$, where $\Delta\boldsymbol{W}$ is the weight increment induced by a single optimization step. We refer to these as the Forward Spectral Condition and the Update Spectral Condition, respectively.

In standard optimization frameworks, both spectral conditions require active maintenance throughout training. The Pion optimizer, however, operates via left and right orthogonal transformations on the weight matrix. This unique structure inherently preserves the singular values of $\boldsymbol{W}$, making its spectral norm strictly invariant. Consequently, as long as the initialization scheme properly enforces the Forward Spectral Condition, this stability holds automatically at every subsequent step. Therefore, the remaining critical task is to determine whether the update matrix $\Delta\boldsymbol{W}$ induced by Pion can satisfy the corresponding Update Spectral Condition.

To address this, recall the Pion update takes the form

$$\boldsymbol{W}_{t+1} = \exp(-\eta\boldsymbol{G}_t^{\text{out}})\boldsymbol{W}_t\exp(-\eta\boldsymbol{G}_t^{\text{in}}). \tag{65}$$

First-order Taylor expansion around $\eta = 0$ yields the weight increment

$$\Delta\boldsymbol{W}_t := \boldsymbol{W}_{t+1} - \boldsymbol{W}_t \approx -\eta\left(\boldsymbol{G}_t^{\text{out}}\boldsymbol{W}_t + \boldsymbol{W}_t\boldsymbol{G}_t^{\text{in}}\right). \tag{66}$$

By applying the triangle inequality and the submultiplicative property of the spectral norm, we can bound the update magnitude as

$$\|\Delta\boldsymbol{W}_t\|_2 \lesssim \eta\|\boldsymbol{W}_t\|_2\left(\|\boldsymbol{G}_t^{\text{out}}\|_2 + \|\boldsymbol{G}_t^{\text{in}}\|_2\right). \tag{67}$$

Since the Pion optimizer inherently preserves the Forward Spectral Condition, the norm $\|\boldsymbol{W}_t\|_2 = \Theta\left(\sqrt{d_{\text{out}}/d_{\text{in}}}\right)$ remains strictly invariant throughout training, provided it is satisfied at initialization. Consequently, fulfilling the Update Spectral Condition reduces to controlling the spectral scales of the two generator matrices. Specifically, enforcing $\|\boldsymbol{G}_t^{\text{out}}\|_2 = \Theta(1)$ and $\|\boldsymbol{G}_t^{\text{in}}\|_2 = \Theta(1)$ ensures that $\|\Delta\boldsymbol{W}_t\|_2$ inherits the identical Θ-order as $\|\boldsymbol{W}_t\|_2$, thereby satisfying the μP requirements. To practically enforce this $\Theta(1)$ spectral norm condition on the Lie-algebra generators, we consider two distinct methodological schemes:

Scheme I: Spectral Norm Scaling. The most straightforward approach is to directly constrain the spectral norm of the generators. By computing the maximum singular values of the projected gradients, we can explicitly normalize $\|\boldsymbol{G}_t^{\text{out}}\|_2$ and $\|\boldsymbol{G}_t^{\text{in}}\|_2$ to exactly $\Theta(1)$.

Scheme II: Explicit Orthogonalization. Alternatively, inspired by Muon's principle, we can explicitly orthogonalize the gradients on the Lie algebra. By applying an orthogonalization procedure via Newton-Schulz iteration to $\boldsymbol{G}_t^{\text{out}}$ and $\boldsymbol{G}_t^{\text{in}}$, we push their non-zero singular values toward 1. This structurally guarantees the strictly bounded $\Theta(1)$ spectral norm condition while uniformly maintaining the update magnitude across all active spectral directions.

A key implication of satisfying the μP condition is that hyperparameters become transferable under width scaling. To verify this property for both proposed schemes, we perform experiments on a LLaMA-based architecture. Specifically, we scale the hidden size, intermediate size, and number of attention heads while keeping the head dimension fixed. For each configuration, we sweep the learning rate and identify the value that yields the lowest validation loss. The hidden size is varied over $\{128, 256, 512\}$. Regarding the specific experimental details, for Scheme I, we directly normalize the spectral norm to 1.0. For Scheme II, to prevent the large μP-derived learning rates from destabilizing the parameters optimized by AdamW, we explicitly set $\alpha = 10.0$ in Eq. 4. As illustrated in Fig. 15, the validation loss curves across all three hidden sizes demonstrate a precise alignment of their optimal learning rates under both Scheme I and Scheme II, confirming that both approaches successfully achieve hyperparameter transferability.

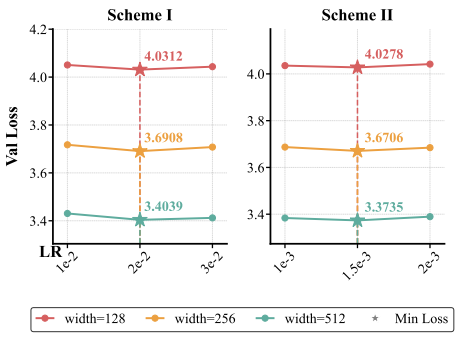


Figure 15: μP learning rate grid search.

G Additional Results

G.1 Another variant of Pion

Here, we provide another variant of Pion, *i.e.*, Pion with Transported Ambient-space Momentum.

Algorithm 2: Pion with Transported Ambient-Space Momentum

Input: Learning rate η, momentum coefficients β_1,β_2, RMS constant c, stability constant ϵ, alternating flag, initial weight $\boldsymbol{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$

Output: Optimized parameter $\boldsymbol{W}_t$

Initialize $\boldsymbol{m}_0, \boldsymbol{v}_0 \leftarrow \boldsymbol{0} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$;

Define $\mathcal{E}_2(\boldsymbol{A},\alpha) \leftarrow \boldsymbol{I} + \eta\alpha\boldsymbol{A} + \frac{1}{2}(\eta\alpha\boldsymbol{A})^2$;

for $t = 1,2,\ldots$ **do**

  $\boldsymbol{G}_t \leftarrow \nabla_{\boldsymbol{W}} f(\boldsymbol{W}_{t-1})$;

  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{G}_t$, $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)(\boldsymbol{m}_t \odot \boldsymbol{m}_t)$;

  $\widetilde{\boldsymbol{G}}_t \leftarrow \boldsymbol{m}_t / (\sqrt{\boldsymbol{v}_t} + \epsilon)$;

  $\boldsymbol{S}_t^{\text{in}} \leftarrow \boldsymbol{W}_{t-1}^\top \widetilde{\boldsymbol{G}}_t - \widetilde{\boldsymbol{G}}_t^\top \boldsymbol{W}_{t-1}$, $\boldsymbol{S}_t^{\text{out}} \leftarrow \widetilde{\boldsymbol{G}}_t \boldsymbol{W}_{t-1}^\top - \boldsymbol{W}_{t-1} \widetilde{\boldsymbol{G}}_t^\top$;

  $\boldsymbol{A}_t^{\text{in}} \leftarrow -\boldsymbol{S}_t^{\text{in}}$, $\boldsymbol{A}_t^{\text{out}} \leftarrow -\boldsymbol{S}_t^{\text{out}}$;

  if *alternate update is used* **then**

    if t *is odd* **then**

      $\alpha_t \leftarrow c\sqrt{d_{\text{out}} d_{\text{in}}} / \left(\|\boldsymbol{W}_{t-1} \boldsymbol{A}_t^{\text{in}}\|_F + \epsilon \right)$;

      $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_{t-1} \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$, $\boldsymbol{m}_t \leftarrow \boldsymbol{m}_t \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$;

    else

      $\alpha_t \leftarrow c\sqrt{d_{\text{out}} d_{\text{in}}} / \left(\|\boldsymbol{A}_t^{\text{out}} \boldsymbol{W}_{t-1}\|_F + \epsilon \right)$;

      $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{W}_{t-1}$, $\boldsymbol{m}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{m}_t$;

    end if

  else

    $\alpha_t \leftarrow c\sqrt{d_{\text{out}} d_{\text{in}}} / \left(\|\boldsymbol{A}_t^{\text{out}} \boldsymbol{W}_{t-1} + \boldsymbol{W}_{t-1} \boldsymbol{A}_t^{\text{in}}\|_F + \epsilon \right)$;

    $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{W}_{t-1} \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$, $\boldsymbol{m}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\text{out}}, \alpha_t) \boldsymbol{m}_t \mathcal{E}_2(\boldsymbol{A}_t^{\text{in}}, \alpha_t)$;

  end if

end for

return $\boldsymbol{W}_t$;

G.2 Pretraining

In this section, we further provide additional results on spectrum evolution and training stability indicators. Specifically, we monitor the Frobenius norms of representative weight matrices together with their corresponding input and output activations, as well as the maximum attention logits throughout training. These quantities serve as practical indicators for characterizing optimization stability and activation amplification during large-scale training.

Figure 17 shows that Pion preserves the spectrum of weight matrices remarkably well throughout optimization, remaining highly consistent with the initialization. In contrast, both AdamW and Muon substantially distort the original spectra as training proceeds. This observation is consistent with the spectrum-preserving design of Pion and highlights its fundamentally different optimization dynamics.

A clear separation among optimizers can also be observed from the stability indicators (Figure 16, 18, 19, 20. AdamW exhibits continuously growing attention logits together with rapidly amplified activation magnitudes. Muon effectively suppresses the growth of attention logits, but the activations and down-projection norms still increase steadily during training. In contrast, Pion keeps nearly all monitored quantities flat and stable throughout optimization. Such distinctive behavior demonstrates the exceptional stability of Pion's spectrum-preserving updates and suggests strong potential for stable large-scale pretraining.

G.3 Supervised Fine-tuning

To complement the aggregated results in the main text, we provide a detailed breakdown of scores for each individual benchmark in Table 5 and Table 6. Furthermore, we investigate the alternating update strategy of the Pion optimizer. Empirical results indicate that while the bilateral update consistently outperforms its alternating counterpart, the alternating strategy still achieves competitive performance that is highly comparable to established baselines like AdamW and Muon. We hypothesize that in

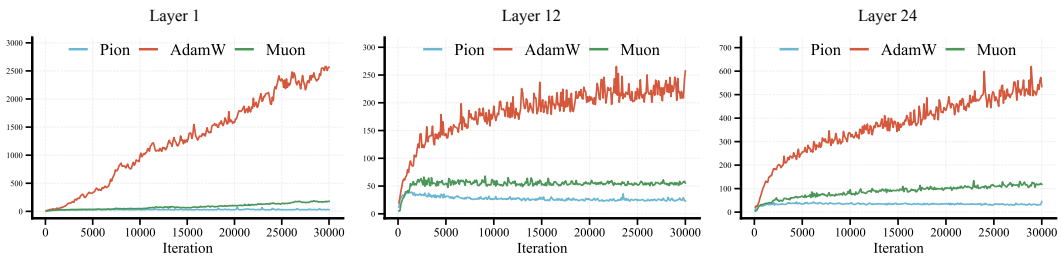


Figure 16: Indicators for stable pretraining. These figures show the maximum attention logit in the attention block of Layer 1, 12, 24.

the supervised fine-tuning phase, which requires precise fitting to deterministic supervision signals, simultaneous double-side updates provide a more synchronous and direct gradient descent path.

Method		MetaMath-50K						Magicoder-50K						
		ARC_C	ARC_E	Hella.	PIQA	Wino.	GSM8K	ARC_C	ARC_E	Hella.	PIQA	Wino.	GSM8K	Human eval
Base		45.22	71.76	67.83	75.95	63.38	59.81	45.22	71.76	67.83	75.95	63.38	59.81	35.98
AdamW		41.46	63.46	66.88	75.35	63.52	65.88	43.71	66.62	67.57	74.64	64.03	59.28	51.83
Muon		40.69	59.55	67.16	74.86	63.85	65.27	42.83	65.02	68.35	75.13	64.16	58.98	50.00
Pion	bilateral update	42.38	63.59	66.64	75.80	62.40	65.76	44.71	66.29	66.56	74.59	63.54	63.54	53.05
	alternating update	41.21	62.96	66.25	75.14	62.50	64.67	43.34	65.44	66.41	74.70	63.85	62.02	52.61

Table 5: Performance comparison of Pion and baseline optimizers on fine-tuning task using Qwen2.5-1.5B base model.

Method		MetaMath-50K						Magicoder-50K						
		ARC_C	ARC_E	Hella	PIQA	Wino	GSM8K	ARC_C	ARC_E	Hella	PIQA	Wino	GSM8K	Human eval
Base		45.82	71.68	73.63	77.58	69.22	25.47	45.82	71.68	73.63	77.58	69.22	25.47	26.22
AdamW		37.03	57.20	68.49	76.17	65.43	59.87	45.39	69.65	72.02	76.33	66.06	22.37	46.95
Muon		38.40	58.33	68.74	76.22	64.33	57.77	44.97	67.76	72.42	76.22	65.04	26.84	46.34
Pion	bilateral update	37.09	56.30	67.70	76.57	64.54	58.83	44.14	68.38	71.94	76.77	67.69	29.49	47.19
	alternating update	35.41	53.37	67.15	75.40	63.93	57.09	43.92	67.13	70.96	76.52	67.25	28.81	45.12

Table 6: Performance comparison of Pion and baseline optimizers on fine-tuning task using Llama-3.2-3B base model.

G.4 Reinforcement Learning with Verifiable Reward

While the main text reports the RLVR performance of AdamW, Muon, and the Pion optimizer utilizing the alternating update strategy, Table 7 supplements these findings by providing the ablation results for Pion's bilateral update counterpart. Interestingly, contrary to the observations in the supervised fine-tuning phase, the alternating strategy proves superior in the RL setting. Nevertheless, the bilateral approach remains highly robust, delivering overall performance that is closely comparable to the established baselines. While yet to be rigorously verified through targeted experiments, we hypothesize that the superiority of the alternating strategy in RL stems from its implicit promotion of exploration within the parameter space. Unlike the deterministic supervision in SFT, RL relies on discovering optimal reasoning paths through sparse and noisy reward landscapes. In such settings, rigid bilateral updates might cause the policy to greedily over-optimize toward early and potentially sub-optimal reward signals, leading to premature convergence. Conversely, we conjecture that the decoupled nature of alternating updates introduces a beneficial exploratory variance into the optimization trajectory. Theoretically, this prevents the model from rapidly collapsing into local optima, thereby maintaining a healthier exploration-exploitation balance essential for robust policy learning.

Method		Qwen3-1.7B						DeepSeek-R1-Distill-Qwen-1.5B					
		AIME24 (avg@32)	AIME25 (avg@32)	AMC23 (avg@8)	Minerva Math (avg@4)	Olympiad Bench (avg@8)	Avg	AIME24 (avg@32)	AIME25 (avg@32)	AMC23 (avg@8)	Minerva Math (avg@4)	Olympiad Bench (avg@8)	Avg
Base		4.06	10.10	30.27	16.27	23.67	16.87	20.52	20.83	54.06	19.39	36.20	30.20
AdamW		22.71	20.94	58.43	25.91	46.09	34.82	25.42	23.94	62.65	23.16	44.69	35.97
Muon		20.42	19.27	54.22	24.08	42.41	32.08	29.06	23.33	66.72	22.89	44.61	37.32
Pion	bilateral update	23.44	17.40	54.07	26.47	44.33	33.14	25.04	23.73	64.76	22.70	43.98	36.04
	alternate update	25.42	21.98	59.94	26.84	46.43	**36.12**	30.00	24.38	66.87	23.90	46.43	**38.32**

Table 7: Performance Comparison of Pion and Baseline Optimizers on RLVR Tasks. The metric (avg@K) denotes the average accuracy over K generated samples per problem. **Bold** values indicate the best overall average results.

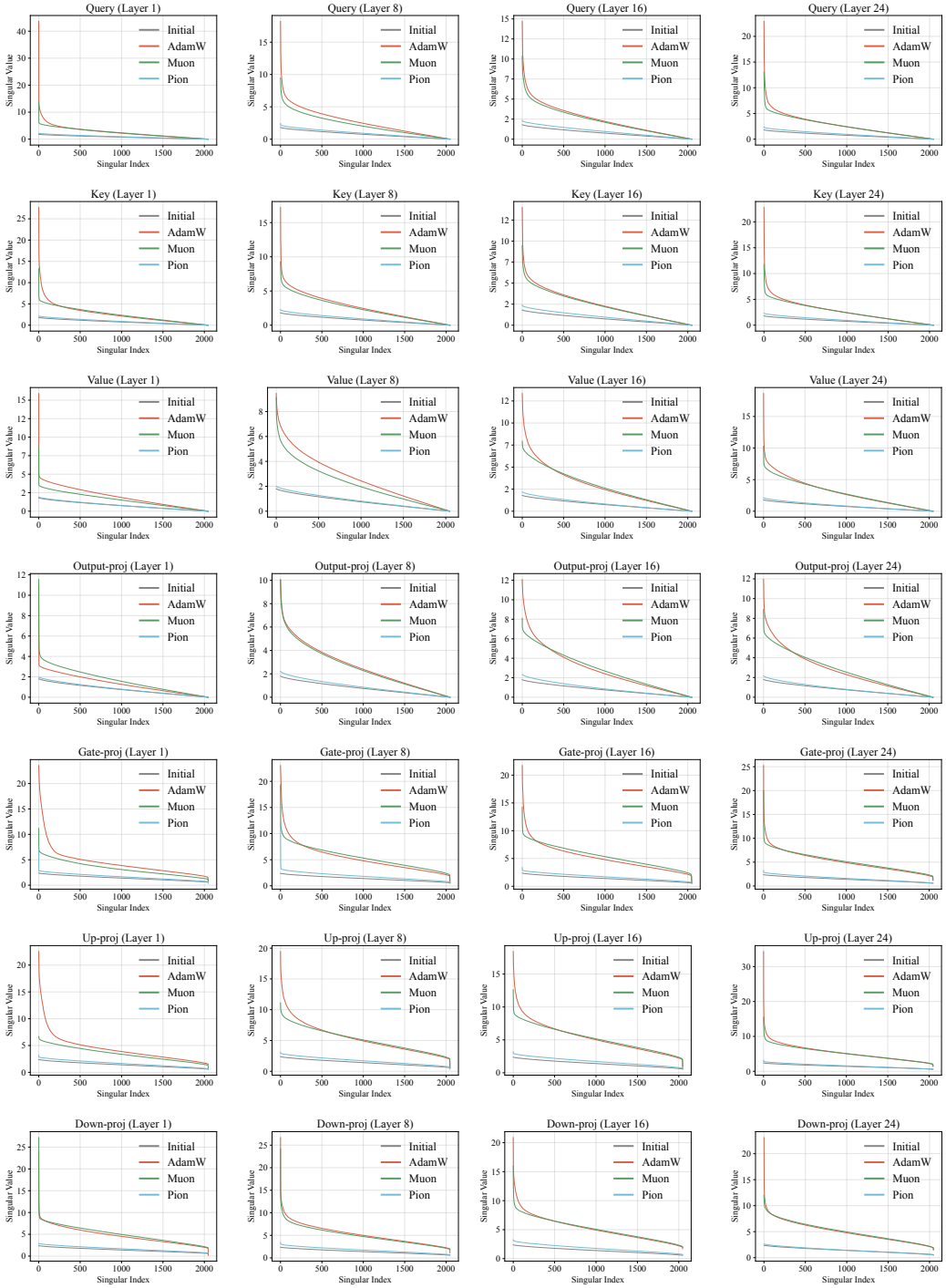


Figure 17: Comparison of final spectrum with the initial spectrum.

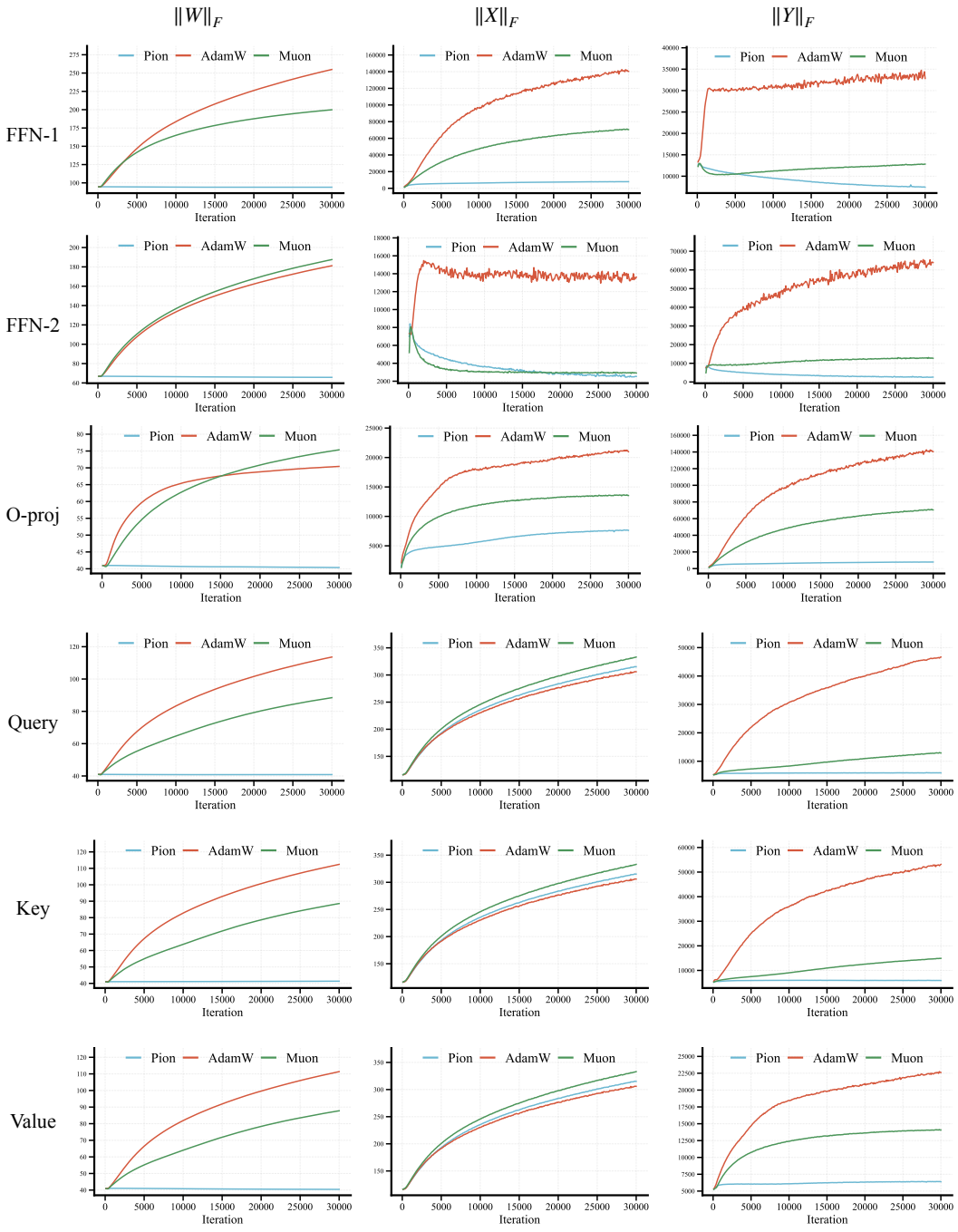


Figure 18: Stable indicators for Layer 1. “FFN-1” denotes the combined gate and up projections in the LLaMA-style feed-forward network, where we compute the norm using their overall input and output; “FFN-2” denotes the down projection matrix; “O-proj” denotes the output projection in the attention module; “Query” denotes the query projection; “Key” denotes the key projection; and “Value” denotes the value projection.

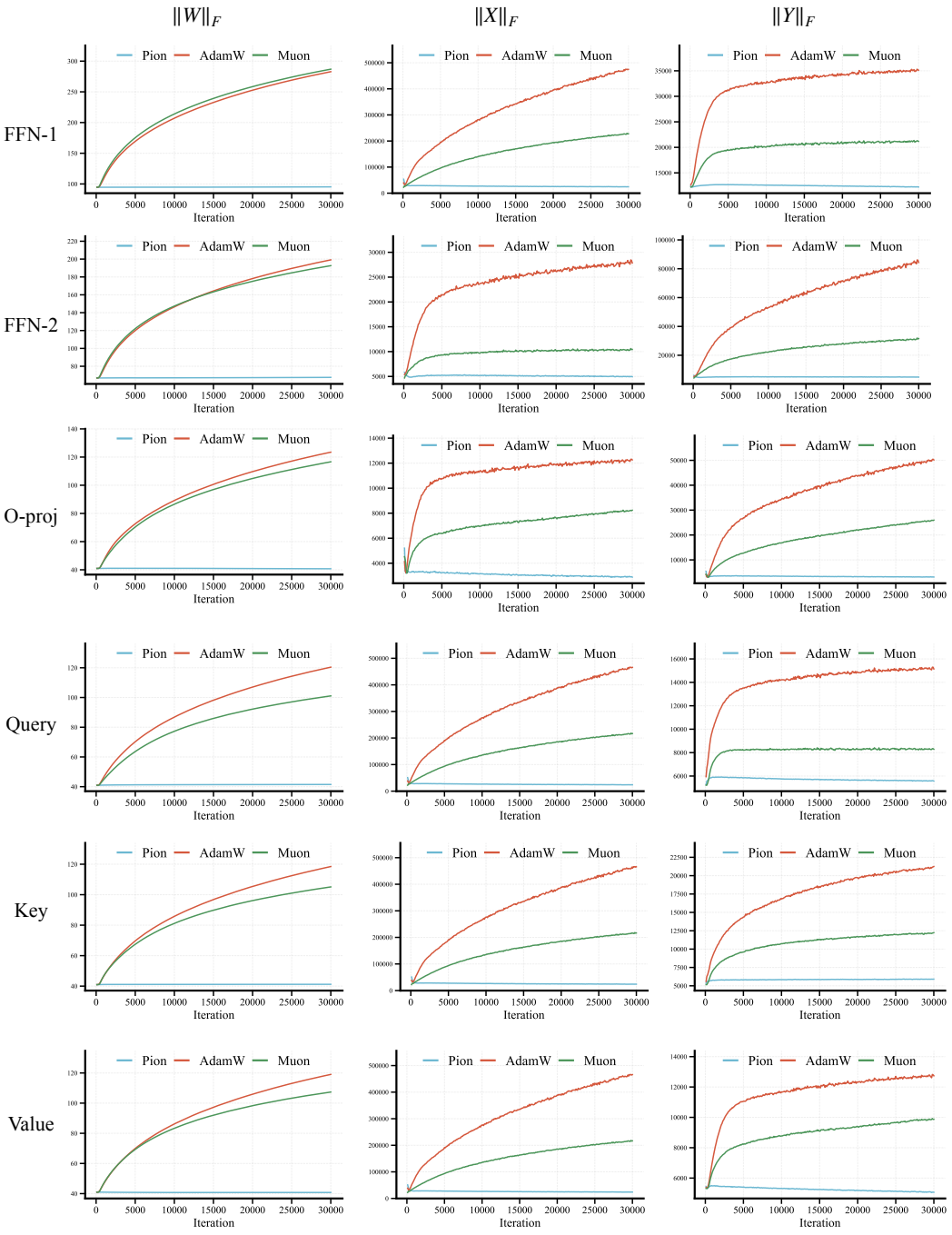


Figure 19: Stable indicators for Layer 12. "FFN-1" denotes the combined gate and up projections in the LLaMA-style feed-forward network, where we compute the norm using their overall input and output; "FFN-2" denotes the down projection matrix; "O-proj" denotes the output projection in the attention module; "Query" denotes the query projection; "Key" denotes the key projection; and "Value" denotes the value projection.

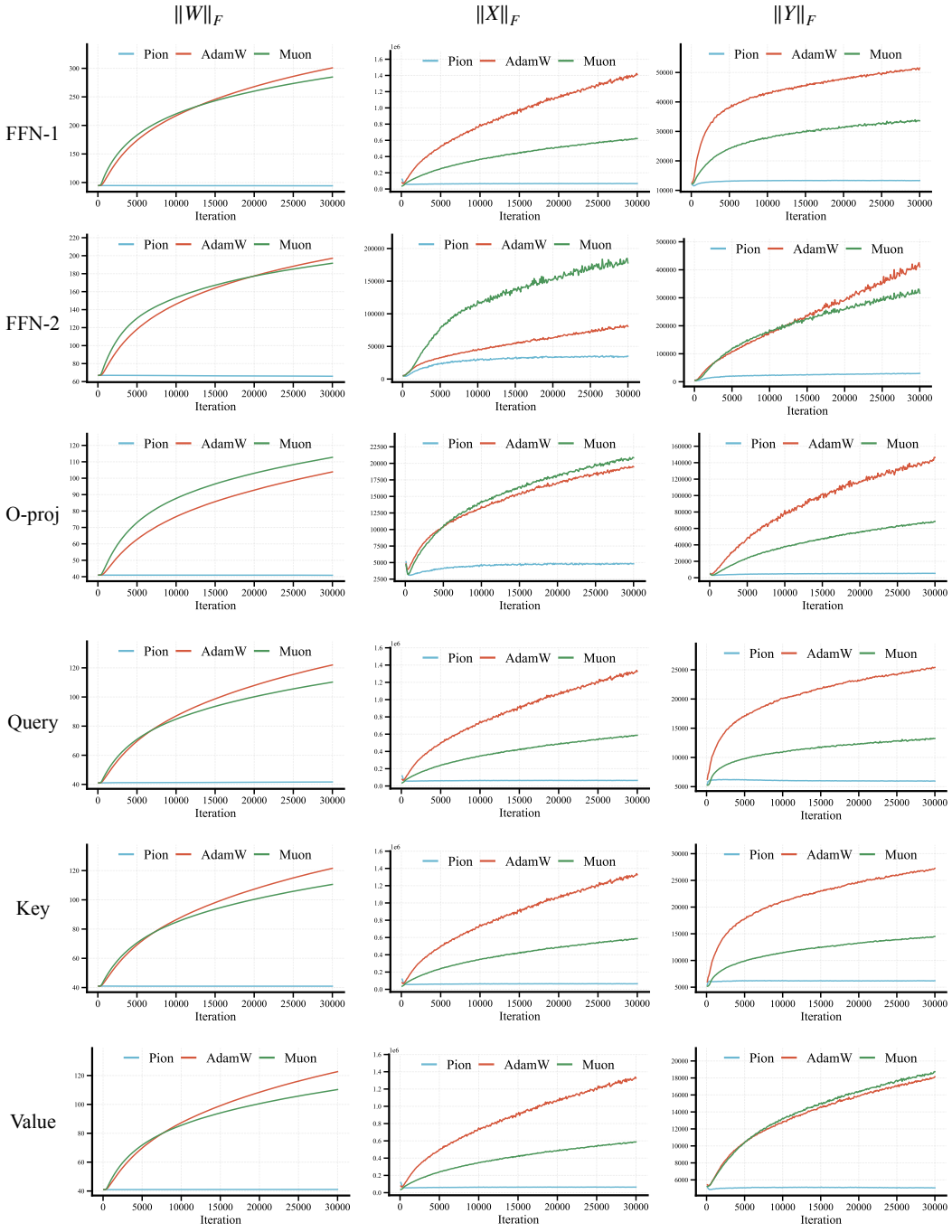


Figure 20: Stable indicators for Layer 24. "FFN-1" denotes the combined gate and up projections in the LLaMA-style feed-forward network, where we compute the norm using their overall input and output; "FFN-2" denotes the down projection matrix; "O-proj" denotes the output projection in the attention module; "Query" denotes the query projection; "Key" denotes the key projection; and "Value" denotes the value projection.

H Balanced Rotations in Pion via Matrix Sign Normalization

We use the 60M LLaMA-based design-exploration setting of Section 2.4, with FineWeb-Edu corpus. The sequence length and training budget (256 and 9.6B tokens) are as in Section 2.4, and the batch size, learning-rate schedule, and learning-rate grid are as in Appendix D.1. The detailed implementation is in Algorithm 3. Here, $\mathrm{msign}(\cdot)$ denotes the matrix sign. Since msign fixes the scale of each generator, $\alpha = 1$ in $\mathcal{E}_2$.

Algorithm 3: Pion Optimizer (msign)

Input: Learning rate η, momentum coefficient β_1, alternating flag, initial weight matrix $\boldsymbol{W}_0$
Output: Optimized parameter $\boldsymbol{W}_t$
Initialize $\boldsymbol{M}_0 \leftarrow \boldsymbol{0}$;
Define $\mathcal{E}_2(\boldsymbol{A},\alpha) \leftarrow \boldsymbol{I} + \eta\alpha\boldsymbol{A} + \frac{1}{2}(\eta\alpha\boldsymbol{A})^2$;
for $t = 1,2,\ldots$ **do**
  $\boldsymbol{G}_t \leftarrow \nabla_{\boldsymbol{W}} f(\boldsymbol{W}_{t-1})$;
  $\boldsymbol{M}_t \leftarrow \beta_1 \boldsymbol{M}_{t-1} + (1-\beta_1)\boldsymbol{G}_t$;
  $\boldsymbol{G}_t^{\mathrm{in}} \leftarrow \boldsymbol{W}_{t-1}^\top \boldsymbol{M}_t - \boldsymbol{M}_t^\top \boldsymbol{W}_{t-1}$;
  $\boldsymbol{G}_t^{\mathrm{out}} \leftarrow \boldsymbol{M}_t \boldsymbol{W}_{t-1}^\top - \boldsymbol{W}_{t-1} \boldsymbol{M}_t^\top$;
  if *alternative update is used* **then**
    if t *is even* **then**
      $\boldsymbol{A}_t^{\mathrm{out}} \leftarrow -\mathrm{msign}(\boldsymbol{G}_t^{\mathrm{out}})$;
      $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\mathrm{out}},1)\boldsymbol{W}_{t-1}$;
    else
      $\boldsymbol{A}_t^{\mathrm{in}} \leftarrow -\mathrm{msign}(\boldsymbol{G}_t^{\mathrm{in}})$;
      $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_{t-1}\mathcal{E}_2(\boldsymbol{A}_t^{\mathrm{in}},1)$;
    end if
  else
    $\boldsymbol{A}_t^{\mathrm{out}} \leftarrow -\mathrm{msign}(\boldsymbol{G}_t^{\mathrm{out}})$;
    $\boldsymbol{A}_t^{\mathrm{in}} \leftarrow -\mathrm{msign}(\boldsymbol{G}_t^{\mathrm{in}})$;
    $\boldsymbol{W}_t \leftarrow \mathcal{E}_2(\boldsymbol{A}_t^{\mathrm{out}},1)\boldsymbol{W}_{t-1}\mathcal{E}_2(\boldsymbol{A}_t^{\mathrm{in}},1)$;
  end if
end for
return $\boldsymbol{W}_T$;

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Please refer to our paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to our Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Please refer to our paper and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides comprehensive details necessary for reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce

the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The experiments in this paper were implemented using custom code based on standard deep learning frameworks. While the code is not publicly released at submission time, the method is described in sufficient detail in Appendix to enable reproducibility. We intend to release the code upon acceptance to facilitate further research.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed experimental settings are presented in the main text and further expanded upon in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The experiments in this work report average performance metrics over multiple runs (e.g., with different random seeds), but do not include explicit error bars, confidence intervals, or formal statistical significance tests. While the main conclusions are supported by consistent trends across datasets and settings, we acknowledge that a more rigorous statistical analysis would strengthen the empirical validation. This is a limitation of the current submission.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to our Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: The paper does not include a discussion of broader societal impacts. This is foundational research on a generic optimization algorithm for deep neural networks. Because the proposed method is domain-agnostic and highly theoretical, its potential societal impacts depend entirely on the specific downstream applications and datasets it is applied to, making direct impact assessment highly speculative and not directly applicable to this work.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [No]

Justification: The paper introduces a mathematical optimizer and does not release pre-trained language models, image generators, or scraped datasets that carry a high risk for direct misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Please refer to our paper.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: This paper does not release any new datasets, code, or trained models. Therefore, this question is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work is purely algorithmic and theoretical, involving no human participants, user studies, or crowdsourced data collection. All experiments are conducted on standard benchmark datasets using automated evaluation protocols

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer:[N/A]

    Justification: This work does not involve any human subjects, user studies, surveys, interviews, or data collected from human participants. All experiments are conducted using existing benchmark datasets and automated evaluation procedures, and therefore IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology of this work is developed independently without reliance on large language models (LLMs).

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.