RETHINKING MUON BEYOND PRETRAINING: SPECTRAL FAILURES AND HIGH-PASS REMEDIES FOR VLA AND RLVR

Anonymous authors
Paper under double-blind review

ABSTRACT

Muon (MomentUm Orthogonalized by Newton–Schulz) is a matrix-aware optimizer that leverages Newton–Schulz (NS) iterations to enforce spectral gradient orthogonalization by driving all singular values of the momentum matrix toward 1. While this *uniform spectral whitening* enhances exploration and outperforms AdamW in LLM pretraining, we show it could lead to fundamental limitations beyond pretraining in two increasingly important regimes: (i) cross-modality *vision-language-action* (VLA) training, where inherently low-rank action-module gradients cause amplification of noisy tail directions, and (ii) *reinforcement learning with verifiable rewards* (RLVR), where low-SNR gradients and the need to preserve per-head specialization inherited from prior training make whitening unstable. To address these challenges, we propose **Pion** (sPectral hIgh-pass Optimization on momeNtum), a drop-in replacement for Muon that preserves its computational efficiency while replacing uniform spectral whitening with a two-stage *Promotion+Suppression* mechanism, which we call the *high-pass NS* iteration. This design induces a sharp spectral high-pass effect, anchoring dominant singular values at 1 while suppressing noisy tail components toward 0, with controllable filter strength. To preserve pretrained per-head heterogeneity, Pion also supports a *per-head* mode that applies updates independently across attention heads via a simple reshape, at no extra cost. Extensive experiments demonstrate consistent gains over Muon and AdamW across both VLA and RLVR regimes. In VLA training on LIBERO and LIBERO-Plus, Pion consistently outperforms both baselines across ℓ_1-regression (VLA-Adapter) and flow-matching (VLANeXt) architectures, *e.g.*, reaching 100% success rate on LIBERO Object after $1{,}500$ training steps with VLA-Adapter, vs. 97.0% for Muon and only 32.2% for AdamW. The advantage of Pion further extends to a real Franka Research 3 robot with a $\pi_{0.5}$ backbone under the DROID setup on three grasp-and-place tasks. In RLVR post-training on Qwen3-1.7B/4B with GRPO and GMPO, Pion also outperforms AdamW on MATH and GSM8K while Muon collapses to zero.

1 INTRODUCTION

AdamW has been the dominant optimizer for deep learning. A recent line of *matrix-aware* optimizers (Gupta et al., 2018; Vyas et al., 2024; Jordan et al., 2024; Liu et al., 2025a) departs from this element-wise paradigm by exploiting the spectral geometry of weight matrices. Among them, **Muon** (Jordan et al., 2024; Liu et al., 2025a) approximates steepest descent under the spectral norm via multi-step Newton–Schulz (NS) iterations that orthogonalize the momentum matrix. This design has achieved consistent gains in large language model (LLM) pretraining and inspired a family of variants (Li et al., 2025; Si et al., 2025; He et al., 2025b; Amsel et al., 2025; Ahn et al., 2025; Wang et al., 2026c).

Despite this progress, Muon's effectiveness *beyond* pretraining remains underexplored. In this work, we ask whether its core mechanism, the matrix sign operation (*i.e.*, gradient orthogonalization that drives all singular values toward 1), remains a desirable inductive bias in non-pretraining regimes.

Inspired by this, we study two representative paradigms beyond pretraining: (i) *multimodal training*, which adapts a base model to new modalities, with our focus on vision-language-action (**VLA**)

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

models (Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025; Wang et al., 2026b; Kim et al., 2025); and (ii) *reinforcement-learning-based post-training*, with our focus on RL with verifiable rewards (**RLVR**) (Shao et al., 2024; Guo et al., 2025; Zhang et al., 2025a).

Therefore, the key research question we address in this work is:

> ***(Q)** Does Muon exhibit promise or limitations in underexplored training paradigms such as VLA and RLVR? If limitations arise, what are the causes and remedies?*

To address (Q), we attribute Muon's limitations in both VLA and RLVR to a shared *spectral mismatch*. In VLA, the action gradient is highly low-rank, while in RLVR the policy gradient is low-SNR. In both cases, informative directions concentrate in a few leading singular values, with the remaining tail dominated by noise (*e.g.*, spectral floor or stochastic estimation noise). Muon's NS iteration uniformly whitens this spectrum, elevating noisy tail directions to the same magnitude as the informative head and thereby corrupting the update. In addition, Muon applies NS to each weight matrix as a *single* block, ignoring the per-head specialization in attention projections inherited from pretraining. This prevents Muon from respecting the heterogeneous update scales required across heads during post-training. The closest related line of work is *Low-Rank Muon* (He et al., 2025a; Pan et al., 2025; Lang et al., 2026), which projects the momentum onto a top-k subspace (via SVD or random sketching) before applying NS. However, it (i) has been studied primarily in LLM pretraining rather than regimes such as VLA or RLVR; (ii) relies on a fixed rank k that cannot adapt across layers or training steps; and (iii) incurs non-trivial per-step SVD or sketching overhead, resulting in significantly poorer scalability than NS iterations in standard Muon.

We exploit the structure of NS to design a direct drop-in alternative to Muon, avoiding computationally intensive spectral operations such as SVD or sketching. Since each NS step reshapes normalized singular values via a scalar polynomial, improving NS reduces to redesigning this polynomial map. Building on this view, we propose **Pion** (sPectral hIgh-pass Optimization on momeNtum), which splits the NS iterations into a two-stage *Promotion+Suppression* sequence. The polynomial coefficients are determined by constraints that first promote dominant singular values and then suppress the tail. This yields a *soft* high-pass filter that anchors leading singular values at 1 while driving the tail toward 0, with per-step cost identical to Muon. We further introduce a *per-head* mode that reshapes each attention projection along its head dimension and applies the high-pass NS independently per head, thereby respecting the heterogeneous update scales required across heads beyond pretraining.

- We identify fundamental limitations of Muon in VLA and RLVR (beyond pretraining) for the first time, arising from its uniform spectral whitening, which amplifies noise in low-rank gradients (*e.g.*, VLA action heads) or low-SNR gradients (*e.g.*, RLVR).

- We propose *Pion*, which redesigns NS into a two-stage *Promotion+Suppression* polynomial iteration (termed *high-pass NS*) that preserves leading singular directions while suppressing noise, at per-step cost identical to Muon. Pion further supports a *per-head mode* that applies the iteration independently across attention heads via a simple reshape, incurring no additional cost.

- On VLA training with ℓ_1-regression and flow-matching heads over LIBERO and LIBERO-Plus as well as on a real Franka Research 3 robot using a $\pi_{0.5}$ backbone (Intelligence et al., 2025), and on RLVR post-training with GRPO and GMPO using Qwen3-1.7B/4B on MATH and GSM8K, Pion consistently outperforms AdamW and Muon while matching Muon's computational efficiency.

2 Related Work

Muon and matrix-aware optimizers. *Matrix-aware* optimizers exploit the spectral geometry of weights: Shampoo/SOAP (Gupta et al., 2018; Vyas et al., 2024) use Kronecker-factored preconditioners at high memory cost, while Muon (Jordan et al., 2024; Liu et al., 2025a) orthogonalizes momentum via NS iterations. Variants improve Muon's per-parameter LR (Li et al., 2025; Si et al., 2025), noise robustness (He et al., 2025b), NS coefficients (Amsel et al., 2025), distributed orthonormalization (Ahn et al., 2025), and low-rank momentum (Wang et al., 2026c; He et al., 2025a), but all retain its *uniform* whitening or rely on costly SVD/sketching. Pion replaces uniform whitening with a polynomial-iteration spectral *high-pass* at no additional overhead.

Vision-language-action models. VLA models turn pretrained vision-language models (VLMs) into closed-loop robot policies (Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025), differing

mainly in the action head – ℓ_1-regression (Wang et al., 2026b; Kim et al., 2025; Wu et al., 2026; Goyal et al., 2025), flow-matching (Lipman et al., 2022; Black et al., 2024), tokenization (Pertsch et al., 2025), and discrete/diffusion decoders (Liang et al., 2025; Wen et al., 2025b; Li et al., 2024a) – with further work on compactness (Shukor et al., 2025; Wen et al., 2025a), prompting (Zheng et al., 2024; Zhang et al., 2026), and benchmarks (Liu et al., 2023; Mees et al., 2022; O'Neill et al., 2024; Li et al., 2024b). The cross-modal VLA *optimizer* is overlooked; we show its action-module gradient is low-rank and calls for a rank-adaptive optimizer.

RLVR and policy optimization for LLM reasoning. RLVR (Shao et al., 2024; Guo et al., 2025; Yang et al., 2025; Zhang et al., 2025a) turns programmatic verifiers into a post-training reward, building on classical policy gradients (Williams, 1992; Schulman et al., 2015; 2017) and RLHF (Ouyang et al., 2022; Bai et al., 2022; Ethayarajh et al., 2024; Li et al., 2023). Subsequent work mostly refines the GRPO (Shao et al., 2024) *objective* – importance-ratio normalization (Zhao et al., 2025; Zheng et al., 2025a), clipping/IS (Yu et al., 2025; Wang et al., 2025; Mao et al., 2025; Liu et al., 2026; Su et al., 2025), critic-free advantage (Hu et al., 2025), KL (Zhang et al., 2025b), exploration (Li et al., 2026; Fan et al., 2026), off-policy stability (Zheng et al., 2025b; Roux et al., 2025), and infra/dynamics (Sheng et al., 2025; Kwon et al., 2023; Liu et al., 2025b; Zhu et al., 2025; Yue et al., 2025). Orthogonal to these, we target the *optimizer*: per-head Pion yields stable, AdamW-matching gains where Muon collapses on the low-SNR RLVR gradient.

3 Muon and Two Underexplored Training Regimes: VLA and RLVR

Muon as spectral optimization. Muon (Jordan et al., 2024) is a matrix-aware optimizer whose core principle is to update a weight matrix $\mathbf{\Theta} \in \mathbb{R}^{m\times n}$ along the *steepest descent direction under the spectral norm*. Given a stochastic gradient $\mathbf{G}_t$ at iteration t as well as a momentum buffer $\mathbf{M}_t = \mu\mathbf{M}_{t-1} + \mathbf{G}_t$ (with μ denoting the momentum coefficient), Muon updates the weight as

$$\mathbf{\Theta}_t = \mathbf{\Theta}_{t-1} - \eta\,\mathrm{msign}(\mathbf{M}_t), \tag{1}$$

where $\eta > 0$ is the step size, and $\mathrm{msign}(\cdot)$ denotes a matrix sign operator, also known as *gradient orthogonalization*, which transforms the momentum $\mathbf{M}_t$ in the spectral domain by mapping its singular values to 1 while preserving the singular vectors. This gives rise to

$$\mathrm{msign}(\mathbf{M}) = \mathbf{U}\mathrm{sign}(\mathbf{\Sigma})\mathbf{V}^\top = \mathbf{U}\mathbf{V}^\top \tag{2}$$

where the iteration index t is omitted for brevity. Here, $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ denotes the *compact* singular value decomposition (SVD) of $\mathbf{M}$, where $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vector matrices, and $\mathbf{\Sigma}$ is the $r \times r$ diagonal matrix collecting the $r = \mathrm{rank}(\mathbf{M})$ strictly positive singular values. The sign operator then yields $\mathrm{sign}(\mathbf{\Sigma}) = \mathbf{I}_r$, returning 1 for every (strictly positive) singular value.

Newton–Schulz (NS) iterations in Muon. As shown in (2), Muon induces a spectrally isotropic update by assigning equal magnitude to all singular directions, which promotes strong exploration during training. However, computing $\mathrm{msign}(\mathbf{M})$ via SVD incurs significant computational overhead and is impractical for large model training. In practice, Muon instead approximates the matrix sign operator using a small number of NS (Newton–Schulz) iterations.

The rationale behind the NS iteration is based on the equivalent form $\mathrm{msign}(\mathbf{M}) = \mathbf{M}(\mathbf{M}^\top\mathbf{M})^{-\frac{1}{2}}$, which reduces the problem to computing $(\mathbf{M}^\top\mathbf{M})^{-\frac{1}{2}}$. This inverse square root is then approximated via a polynomial iteration derived from a local Taylor expansion around the identity. As a result, NS iteratively applies low-order matrix polynomials to approximate $(\mathbf{M}^\top\mathbf{M})^{-1/2}$, and thus $\mathrm{msign}(\mathbf{M})$, without requiring explicit matrix decomposition. Specifically, for a general matrix $\mathbf{X}$, the matrix sign operator $\mathrm{msign}(\mathbf{X})$ is approximated via NS iteration of the following form (Jordan et al., 2024)

$$\mathbf{X} \;\leftarrow\; a\,\mathbf{X} + b\,\mathbf{X}\mathbf{X}^\top\mathbf{X} + c\,\mathbf{X}(\mathbf{X}^\top\mathbf{X})^2, \;\text{ with }\; (a,b,c) = (3.4445, -4.7750, 2.0315), \tag{3}$$

where the input is pre-normalized as $\mathbf{X} \leftarrow \mathbf{X}/(\|\mathbf{X}\|_{\mathrm{F}} + \epsilon)$ (with small $\epsilon \geq 0$) to bound all singular values within $[0, 1]$, and $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. Setting $\mathbf{X} = \mathbf{M}_t$, the NS iterations are used in place of (2) to approximate the msign operation in the Muon update (1).

Underexplored regimes for Muon beyond LLM pretraining. Muon is widely used for LLM pretraining. We show that Muon-type optimizers also hold significant potential beyond this setting. However, the conventional Muon design exhibits important limitations in these settings (as will be

shown in Sec. 4), leading to suboptimal performance and hindering its broader adoption. Throughout our work, we focus on two underexplored training regimes for Muon: (i) multimodal training of VLA (vision-language-action) models, and (ii) post-training via RLVR (reinforcement learning with verifiable rewards), where Muon remains less explored than AdamW.

(i) VLA trains a policy on offline demonstrations $\mathcal{D} = \{(\mathbf{x}, \mathbf{c}, \mathbf{a})\}$ to map visual observations $\mathbf{x}$ and language instructions $\mathbf{c}$ to continuous robot actions $\mathbf{a}$. Internally, the policy is factorized into a VLM (vision-language model) backbone and an action head, parameterized as $\mathbf{\Theta} = \{\mathbf{\Theta}_{\text{VLM}}, \mathbf{\Theta}_{\text{action}}\}$. We consider two representative designs for the action head $\mathbf{\Theta}_{\text{action}}$ (training losses detailed in **Appendix A.1**): a ℓ_1-*regression head* (Wang et al., 2026b; Kim et al., 2025), and a *flow-matching head* (Lipman et al., 2022; Black et al., 2024; Wu et al., 2026).

(ii) RLVR is a *post-training* paradigm in which the supervised fine-tuning (SFT)-initialized policy is further updated by policy gradient against a verifiable reward (Shao et al., 2024). Unlike SFT, which matches token-level teacher signals on offline demonstrations, RLVR alternates between three stages at every iteration: *rollout*, *scoring*, and *policy update*. We instantiate the policy update via two algorithms, GRPO (Shao et al., 2024) and GMPO (Zhao et al., 2025) (objectives in **Appendix A.2**).

4 Rethinking Muon in Heterogeneous and Noisy Training Regimes

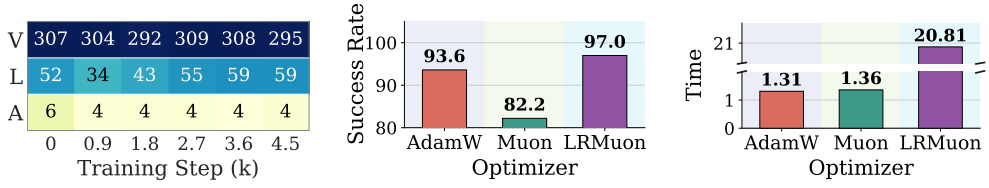
In this section, we show that the default Muon design exhibits fundamental limitations in VLA and RLVR, revealing opportunities for improved optimizer design.

Rank adaptiveness in cross-modality VLA training. VLA models jointly train three heterogeneous modules, a vision encoder, a language backbone, and an action head (Kim et al., 2024; Black et al., 2024), whose gradients can differ significantly in their intrinsic dimensionality. To quantify this heterogeneity, we use the *effective rank* (*erank*) (Roy & Vetterli, 2007) of a gradient matrix $\mathbf{G} \in \mathbb{R}^{m \times n}$ (*w.l.o.g.*, $n \leq m$), defined via the entropy of its singular value spectrum:

$$\text{erank}(\mathbf{G}) := \exp\big(H(\mathbf{p})\big), \quad H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i, \quad p_i = \frac{\sigma_i(\mathbf{G})}{\sum_{j=1}^{n} \sigma_j(\mathbf{G})}, \tag{4}$$

where $\mathbf{p} = [p_1, \ldots, p_n]^\top$, and $\sigma_i(\mathbf{G})$ denotes the i-th singular value of $\mathbf{G}$. A higher erank indicates that the gradient energy is distributed across many directions.

Fig. 1-(a) reports the average per-module erank along the trajectory of training VLA-Adapter on LIBERO Object. The vision module maintains the highest erank, the language module is intermediate, and the action module consistently exhibits the *lowest* erank. This ordering is stable across training steps, with intra-module variance (column-wise) much smaller than inter-module variance (row-wise). It also aligns with the information capacity of each modality: vision inputs encode rich pixel-level statistics, language tokens use high-dimensional embeddings to disambiguate a large vocabulary, while each action is just a seven-dimensional vector encoding the incremental end-effector translation, rotation, and a binary gripper command. Given this low-rank structure, applying Muon uniformly to the action module inflates every normalized singular value toward 1, making Muon ill-suited for the action module despite its effectiveness on the higher-rank vision and language modules.



(a) Per-module gradient erank (b) Test success rate (c) Total training time (hrs)

Figure 1: Limitations of Muon in VLA training (VLA-Adapter on LIBERO Object). (a) Average per-module gradient erank (V/L/A) along the training trajectory of 4.5k steps, recorded every 900 steps. (b)–(c) Test success rates on LIBERO Object for models trained for 4.5k steps, along with total training time (hours), under three optimizer configurations: AdamW / Muon / LRMuon for the action module, with AdamW fixed for VL modules.

Can existing Muon variants address the limitation in VLA training? A natural candidate is *Low-rank Muon* (**LRMuon**) (He et al., 2025a; Pan et al., 2025; Lang et al., 2026), which projects the momentum onto a low-rank subspace (via SVD or Gaussian sketching) prior to gradient orthogonalization. This

approach can adapt to the low-rank structure of the action-module gradients. However, both SVD and Gaussian sketching incur substantially higher computational cost than NS, leading to slower training. To validate this, **Fig. 1-(b,c)** reports the success rate on the LIBERO Object evaluation set together with the total training time, under three optimizer configurations that share the same AdamW updates on the vision and language modules and *differ only in the action module:* (i) **AdamW**, (ii) **Muon**, and (iii) **LRMuon** (see **Alg. 1** in **Appendix B** for details). We deliberately fix the V/L optimizer to AdamW, so that *the comparison isolates the effect of the action-module optimizer*. As shown, Muon underperforms AdamW, as expected from the rank heterogeneity shown in Fig. 1-(a). In addition, LRMuon achieves the highest success rate, confirming the benefit of rank-aware optimization for the action module; however, it incurs about $15\times$ higher training cost than AdamW and Muon.

Motivated by the above, we summarize the first limitation of Muon below.

> ***(Limitation 1)*** *Lack of rank adaptiveness: Conventional Muon is not adaptive to rank heterogeneity across modules, leading to suboptimal performance, while explicit low-rank projection introduces significant computational overhead, limiting scalability.*

SNR tolerance for RLVR post-training. Despite recent progress applying Muon to SFT-based (pre-)training (Liu et al., 2025a; Si et al., 2025; Li et al., 2025), its effectiveness in post-training, particularly for RLVR, remains largely unexplored. To understand this gap, we examine how SFT and RLVR, as two post-training paradigms, differ in terms of gradient signal-to-noise ratio (SNR). Unlike LLM pretraining, post-training typically requires only moderate modifications to weights (Gan & Isola, 2026), making optimization more sensitive to noise. Meanwhile, as discussed in **Sec. 3**, a key characteristic of Muon is its strong exploration behavior induced by the uniform spectral sign function (2), which can amplify noise during training.

Motivated by the above, we analyze the per-step gradient SNR of a layer's weight matrix, defined as

$$\mathrm{SNR}(\mathbf{G}) := \frac{\| \mathbb{E}[\mathbf{G}] \|_{\mathrm{F}}^2}{\mathbb{E}\big[\|\mathbf{G} - \mathbb{E}[\mathbf{G}]\|_{\mathrm{F}}^2 \big]}, \tag{5}$$

where the expectation is over the batch; a higher SNR means a cleaner gradient signal. We use GRPO (Shao et al., 2024) as the representative RLVR algorithm, train Qwen3-1.7B on MATH levels 3–5 (Liu et al., 2025b), and evaluate on MATH500. **Fig. 2-(a)** compares the gradient SNR of SFT and GRPO, both optimized with AdamW. GRPO exhibits a much *lower* SNR than SFT throughout training. We attribute this gap to two primary sources of additional noise in GRPO. First, GRPO has coarser *supervision granularity*: SFT receives token-level teacher signals, whereas GRPO relies on trajectory-level rewards, yielding a much sparser per-token signal. Second, GRPO relies on *stabilization mechanisms*: Importance sampling, clipping, and group-relative normalization in (A3) reweight or suppress portions of per-token gradients, thereby further increasing gradient variance. As a result, GRPO gradients exhibit a *low-SNR* structure, a regime in which Muon's spectral whitening becomes counterproductive. See **Appendix C** for a derivation.

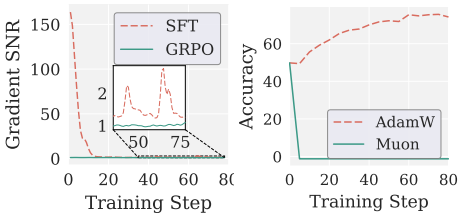


Figure 2: (a) Gradient SNR of SFT vs. GRPO (AdamW, Qwen3-1.7B) on MATH levels 3–5. (b) MATH500 accuracy of Qwen3-1.7B via GRPO (AdamW vs. Muon).

Fig. 2-(b) shows that GRPO with AdamW steadily improves, whereas with Muon it exhibits a *model collapse*: accuracy collapses to near-zero from initialization. This confirms that Muon's uniform spectral whitening amplifies noisy directions in gradients to the same magnitude as informative ones, corrupting the policy. Moreover, Muon's msign treats each weight matrix as a *single* block, ignoring the *per-head specialization* established during pretraining in attention projections.

We summarize the above limitation of Muon as evidenced in RLVR post-training below.

> ***(Limitation 2)*** *Lack of noise adaptiveness: Muon's uniform spectral whitening amplifies noisy directions in low-SNR gradients, making it ill-suited for noise-sensitive post-training regimes.*

Both Limitations 1 and 2 stem from the inappropriate spectral exploration induced by the msign operator (*i.e.*, via NS iterations). This motivates us to improve the design of NS iterations in the next section to enhance Muon's adaptiveness to rank heterogeneity and resilience to low-SNR gradients.

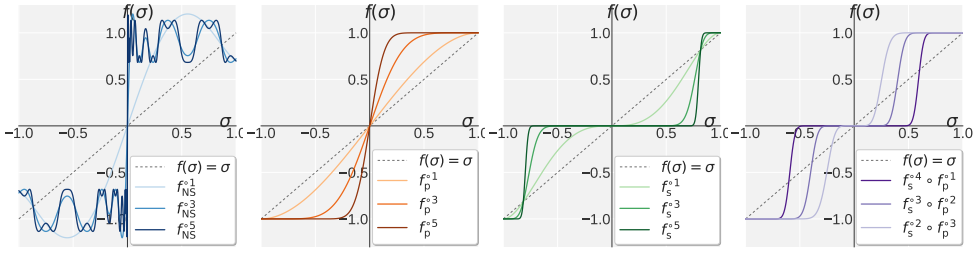
5 PION: SPECTRAL HIGH-PASS OPTIMIZATION ON MOMENTUM

A unifying spectral view of Muon's limitations: informative head vs. noisy tail. Although the two limitations of Sec. 4 originate from different sources (low erank for VLA, low SNR for RLVR), they share a *common spectral signature*: in the SVD of $\mathbf{M}_t$, the few *leading* singular values carry the informative descent direction, while the long *tail* of small singular values is dominated by noise (spectral floor for low erank, stochastic estimation noise for low SNR). In **Appendix D**, we also use the directional SNR to show that this tail is mostly noise. Muon's msign, by driving *every* σ_i to 1, lifts this tail to the same magnitude as the head and corrupts the update in both regimes. This motivates a single remedy, a *spectral high-pass* that *retains large singular values* (anchoring them near 1) and *suppresses small singular values* (contracting them toward 0), in contrast to Muon's uniform whitening (**Fig. 3-(a)**). We realize this with **Pion**, which inherits Muon's control flow and per-step cost and differs only in the coefficients of its NS iteration.

A two-stage high-pass NS mechanism as a remedy. A single NS step (3) on $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ factors through the SVD as $\mathbf{U}\big(a\mathbf{\Sigma} + b\mathbf{\Sigma}^3 + c\mathbf{\Sigma}^5\big)\mathbf{V}^\top$ via the identity $\mathbf{X}(\mathbf{X}^\top\mathbf{X})^j = \mathbf{U}\mathbf{\Sigma}^{2j+1}\mathbf{V}^\top$. Hence the NS step preserves $(\mathbf{U}, \mathbf{V})$ and *independently* reshapes each $\sigma_i \in [0, 1]$ through the polynomial

$$f(\sigma;\, a, b, c) := a\sigma + b\sigma^3 + c\sigma^5. \tag{6}$$

Thus, designing an NS iteration reduces to designing f on $[0, 1]$ (see **Appendix E** for the full derivation). A single polynomial f in (6) is insufficient to produce a sharp high-pass profile, so we split the NS iteration (with $k = 5$ steps by default) into two stages: an early-stage **Promotion** polynomial f_{p} (**Fig. 3-(b)**) applied for k_{p} steps to reinforce dominant singular values, and a late-stage **Suppression** polynomial f_{s} (**Fig. 3-(c)**) applied for $k_{\mathrm{s}} = k - k_{\mathrm{p}}$ steps to attenuate smaller components, each with its own coefficients (a, b, c).



(a) Muon NS (b) Promotion f_{p} (c) Suppression f_{s} (d) High-pass NS

Figure 3: Visualization of $f(\sigma)$ in (6) over $\sigma \in [0, 1]$, with $f(\sigma) = \sigma$ shown as the identity reference. (a) f_{NS}^t denotes Muon's NS iteration applied t times. (b) f_{p}^t denotes the Promotion polynomial f_{p} (7) applied t times. (c) f_{s}^t denotes the Suppression polynomial f_{s} (8) applied t times. (d) Pion's high-pass NS iteration (Alg. 2): $f_{\mathrm{s}}^{k_{\mathrm{s}}} \circ f_{\mathrm{p}}^{k_{\mathrm{p}}}$ applies k_{p} Promotion steps followed by $k_{\mathrm{s}} = 5 - k_{\mathrm{p}}$ Suppression steps.

The Promotion stage $f_{\mathrm{p}} := f(\,\cdot\,;\, a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}})$ monotonically amplifies all singular values σ, so as to (i) lift as many singular values as possible above the subsequent suppression threshold and (ii) preserve their relative ordering, ensuring that the later Suppression eventually removes only the smallest. The three coefficients in (6) are pinned by two equality constraints and one inequality: **(P1)** *fixed point* $f_{\mathrm{p}}(1) = 1$ and **(P2)** *first-order stationarity* $f_{\mathrm{p}}'(1) = 0$ (both shared with Suppression) anchor any direction already at $\sigma = 1$; **(P3)** *boundary concavity* $f_{\mathrm{p}}''(1) \le 0$, together with (P2), ensures that $\sigma = 1$ is a maximum, *i.e.*, prevents the Promotion from curving upward near $\sigma = 1$. See **Fig. 3-(b)** for illustration. As derived in **Appendix F**, conditions (P1)–(P3) directly carve out the upper bound $a_{\mathrm{p}} \le 1.875$, and additionally requiring f_{p} to be monotonically non-decreasing on $[0, 1]$ (so that the relative ordering of singular values is preserved across each Promotion step) tightens the lower bound to $a_{\mathrm{p}} \ge 0$, yielding $a_{\mathrm{p}} \in [0, 1.875]$. Since $f_{\mathrm{p}}'(0) = a_{\mathrm{p}}$ determines the slope at the origin, we set $a_{\mathrm{p}} = 1.875$ to maximize promotion, thereby amplifying small singular values as strongly as possible. This choice uniquely determines the polynomial coefficients for the Promotion stage:

$$f_{\mathrm{p}}(\sigma) = a_{\mathrm{p}}\,\sigma + b_{\mathrm{p}}\,\sigma^3 + c_{\mathrm{p}}\,\sigma^5, \quad \text{with } (a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}}) = (1.875, -1.25, 0.375). \tag{7}$$

With these coefficients, the derivative becomes a perfect square, $f_{\mathrm{p}}'(\sigma) = 1.875\,(1 - \sigma^2)^2 \ge 0$, ensuring monotonicity on $[0, 1]$, as shown in Fig. 3-(b).

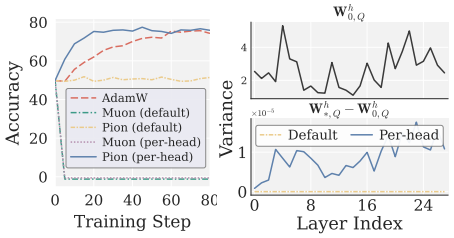
The Suppression stage $f_{\mathrm{s}} := f(\,\cdot\,; a_{\mathrm{s}}, b_{\mathrm{s}}, c_{\mathrm{s}})$ pins large singular values at 1 while contracting smaller ones toward 0 (**Fig. 3-(c)**). It inherits $f_{\mathrm{s}}(1) = 1$ and $f'_{\mathrm{s}}(1) = 0$, and adds the *spectral filtering* condition $f'_{\mathrm{s}}(0) = 0$, which removes the linear term near the origin so that small singular values are driven to 0 by higher-order terms. These constraints give the *Suppression polynomial*:

$$f_{\mathrm{s}}(\sigma) = a_{\mathrm{s}}\,\sigma + b_{\mathrm{s}}\,\sigma^3 + c_{\mathrm{s}}\,\sigma^5, \;\; \text{with} \;\; (a_{\mathrm{s}}, b_{\mathrm{s}}, c_{\mathrm{s}}) = (0, 2.5, -1.5). \tag{8}$$

The Pion optimizer and its two application modes. Chaining k_{p} Promotion steps with k_{s} ($= k - k_{\mathrm{p}}$) Suppression steps yields a **high-pass NS** iteration; the resulting Muon variant is termed **Pion** (see the full algorithm in **Appendix G**). Fixing $k = 5$ preserves Muon's per-step cost, and $k_{\mathrm{p}} \in \{0, 1, \ldots, 5\}$ becomes the single hyperparameter that controls the high-pass cutoff: **Fig. 3-(d)** shows that Pion exhibits a sharp transition between the pinned region ($\sigma \mapsto 1$) and the filtered region ($\sigma \mapsto 0$). Experiments in **Appendix H** show that Muon's NS raises the erank of its input, whereas Pion's high-pass NS can shrink it. Empirically, Suppression-dominant allocations with $k_{\mathrm{s}} \geq 3$ consistently perform best for VLA and RLVR training, as they more aggressively suppress noisy tail while preserving the informative head.

The high-pass NS admits *two modes*: (i) the **default mode** applies the iteration to each weight matrix as a single block, mirroring Muon; (ii) the **per-head mode** first reshapes each attention projection along its head dimension into multiple per-head sub-matrices and runs the iteration independently on each. We use the default mode for VLA training (**Sec. 6.2**) and the per-head mode for RLVR post-training (**Sec. 6.3**), as explained next.

Why per-head high-pass NS is needed for RLVR. RLVR starts from an already-pretrained model whose attention layers exhibit heterogeneous per-head weight norms. Such heterogeneity is functionally meaningful: per-head norms govern attention sharpness and gradient magnitudes (**Appendix I**), so different heads naturally require updates at different scales. However, both default-mode Pion and Muon apply NS iterations to each projection *as a whole*, ignoring this per-head heterogeneity. As a result, training becomes less effective, as shown in **Fig. 4-(a)**, where default-mode Pion underperforms AdamW and (default-mode) Muon collapses. We also observe that enabling the per-head mode for Muon does not improve performance, since the lack of noise adaptiveness (Limitation 2) remains the primary cause of its ineffectiveness in RLVR. The superior performance of per-head Pion suggests that spectral high-pass filtering is the primary driver of RLVR stability, while the per-head reshape serves as an auxiliary mechanism that preserves pretrained head structure. To further justify per-head awareness in Pion, we analyze the Q projection sub-blocks $\{\mathbf{W}_Q^h\}_{h=1}^H$ across H attention heads (**Fig. 4-(b)**). Let $\mathbf{W}_0$ and $\mathbf{W}_*$ denote the weights before and after RLVR, respectively. We measure per-head heterogeneity via the cross-head variance $\mathrm{Var}(\|\mathbf{W}_{0,Q}^h\|_{\mathrm{F}})$. Prior to RLVR, this variance is non-trivial across all 28 layers of Qwen3-1.7B (top). However, the update variance $\mathrm{Var}(\|\mathbf{W}_{*,Q}^h - \mathbf{W}_{0,Q}^h\|_{\mathrm{F}})$ under default-mode Pion is nearly flat (bottom), indicating uniform updates across heads that fail to reflect heterogeneity. In contrast, the *per-head* mode reshapes projections along the head dimension, enabling heterogeneous updates.



(a) (b)

Figure 4: Effect of per-head high-pass NS on RLVR (Qwen3-1.7B, GRPO on MATH levels 3–5). (a) MATH500 accuracy of AdamW, Muon (default vs. per-head), and Pion (default vs. per-head). (b) Cross-head Q-projection variance: before-RLVR weight $\mathrm{Var}(\|\mathbf{W}_{0,Q}^h\|_{\mathrm{F}})$ (top) and after-RLVR update $\mathrm{Var}(\|\mathbf{W}_{*,Q}^h - \mathbf{W}_{0,Q}^h\|_{\mathrm{F}})$ for default vs. per-head Pion (bottom).

6 Experiments

6.1 Experiment setups

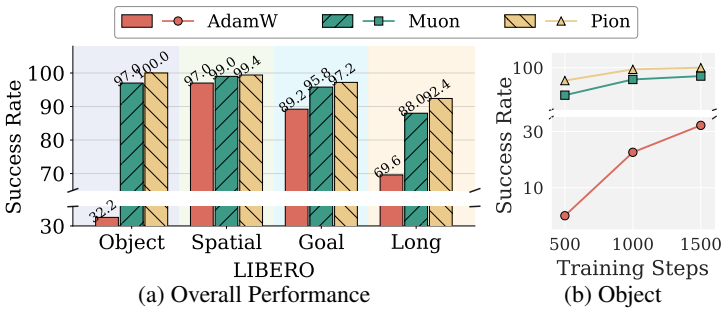
VLA setups. Two models are assessed: ℓ_1-regression-based **VLA-Adapter** (Wang et al., 2026b) and flow-matching-based **VLANeXt** (Wu et al., 2026). Both are trained and tested on the four **LIBERO** suites (Liu et al., 2023), with VLANeXt additionally evaluated on **LIBERO-Plus** (Fei et al., 2025). We further include a real-robot evaluation by finetuning $\pi_{0.5}$ (Intelligence et al., 2025) under the **DROID** setup (Khazatsky et al., 2025) on three grasp-and-place tasks. We compare three optimizers:

(i) **AdamW** globally; (ii) **Muon** on all 2D matrices (excluding embeddings/output layer), with AdamW elsewhere; and (iii) **Pion**, applying Pion to the action 2D matrices, Muon to vision/language 2D matrices, and AdamW elsewhere. Performance is measured by *success rate* (%).

RLVR setups. Experiments utilize **Qwen3-1.7B** and **Qwen3-4B** (Yang et al., 2025) optimized via **GRPO** (Shao et al., 2024) and **GMPO** (Zhao et al., 2025). Models are trained on **GSM8K** (training split) and **MATH** levels 3–5, and evaluated on the GSM8K test split (Cobbe et al., 2021) and MATH500 (Hendrycks et al., 2021), respectively. Optimizer configurations mirror the VLA setups: (i) **AdamW**, (ii) **Muon**, and (iii) **Pion**, which adopts the per-head mode introduced in **Sec. 5**. The evaluation metric is *accuracy* (%). See **Appendix J** for details.

6.2 VLA EXPERIMENT RESULTS

Advantages of Pion over Muon and AdamW for VLA-Adapter on LIBERO. **Fig. 5** presents final success rates of VLA-Adapter on the four LIBERO task suites using AdamW, Muon, and Pion under a fixed budget per suite, along with learning curves for LIBERO Object. As shown in **Fig. 5-(a)**, Muon already outperforms AdamW on all four tasks, indicating that spectral steepest descent benefits multimodal training. Pion further improves over Muon on every task. This aligns with the spectral analysis in Fig. 1: the action module exhibits near-low-rank gradients, so Pion's high-pass filter preserves informative singular directions while suppressing tail noise that Muon would otherwise amplify. Furthermore, **Fig. 5-(b)** shows that Pion reaches 95.4% success at 500 steps and saturates at 100% by $1{,}500$ steps, while AdamW requires substantially more steps and Muon consistently lags behind, indicating that Pion's spectral high-pass yields faster convergence on the action module.



(a) Overall Performance (b) Object

Figure 5: AdamW, Muon and Pion for VLA-Adapter on LIBERO. (a) Test success rates on LIBERO Object, Spatial, Goal and Long at the same training budget ($1{,}500$ steps for Object and $15{,}000$ steps for others). (b) Test success rates vs. training steps on Object.

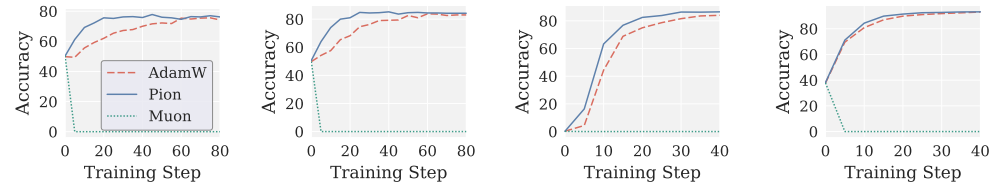
Table 1: AdamW, Muon and Pion for VLANeXt on LIBERO and LIBERO-Plus. Columns 2–3: average test success rate on LIBERO/LIBERO-Plus; Columns 4–10: test success rate on LIBERO-Plus under different perturbations. Best score in each column is in **bold**.

Optimizer	LIBERO	LIBERO-Plus	Background	Camera	Language	Layout	Light	Noise	Robot
AdamW	79.45	64.57	68.97	70.38	54.50	61.80	76.35	66.37	47.04
Muon	93.65	72.34	82.72	68.00	77.53	76.21	86.17	69.98	57.36
Pion (Ours)	**96.35**	**75.93**	**84.53**	**70.88**	**86.93**	**76.71**	**90.67**	**76.09**	**63.18**

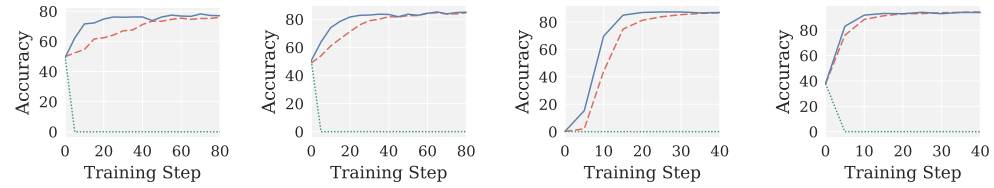
The Pion advantage extends to flow-matching VLAs and perturbed scenes. To validate that Pion's benefit is not architecture-specific, we evaluate **VLANeXt** (Wu et al., 2026), a flow-matching VLA. **Table 1** reports success rates on LIBERO and LIBERO-Plus. Pion outperforms Muon and AdamW in all settings, confirming that the high-pass mechanism is *model-agnostic* across both regression-based and flow-matching VLAs. Moreover, its advantage is *preserved and amplified* on the more challenging LIBERO-Plus split, notably under Language (+9%), Noise (+6%), and Robot (+6%) perturbations. This suggests that Pion yields more *robust* policies under distribution shifts, tackling the limitation that Muon-style whitening could over-amplify non-generalizable noise directions. Qualitative rollouts on the four LIBERO suites are provided in **Appendix K**.

Additional results. On a real robot ($\pi_{0.5}$ under DROID), Pion reaches an 85.6% average success rate versus 31.1% (AdamW) and 38.9% (Muon) (**Appendix L**). Three studies on VLA-Adapter (**Appendix M**) show that (i) Pion outperforms LRMuon across all top-k ranks while matching Muon's total training time (**Fig. A1**); (ii) per-head Pion on the action head also beats Muon and AdamW but underperforms the default mode (**Table A8**); and (iii) a modality-wise optimizer sweep prefers Muon on V/L and Pion on action (**Table A9**).

6.3 RLVR EXPERIMENT RESULTS



(a) GRPO, 1.7B, MATH (b) GRPO, 4B, MATH (c) GRPO, 1.7B, GSM8K (d) GRPO, 4B, GSM8K



(e) GMPO, 1.7B, MATH (f) GMPO, 4B, MATH (g) GMPO, 1.7B, GSM8K (h) GMPO, 4B, GSM8K

Figure 6: AdamW, Muon and Pion on RLVR: validation accuracy vs. training step across eight settings, spanning two algorithms (GRPO, GMPO), two model sizes (Qwen3-1.7B, Qwen3-4B), and two benchmarks (MATH: train on levels 3–5 / evaluate on MATH500; GSM8K: train/test splits).

Pion succeeds while Muon collapses. Across all eight settings in **Fig. 6**, Muon fails: accuracy stays near zero and often falls below the initial checkpoint, and this collapse persists across different learning rate setups (**Appendix N**). This aligns with our Limitation 2 analysis in Sec. 4: under low-SNR RLVR gradients, Muon's uniform whitening amplifies noisy directions to the same magnitude as informative ones, leading to rapid policy collapse. In contrast, Pion recovers a meaningful training signal and *outperforms* AdamW, converging faster in all settings, which shows that spectral high-pass filtering is key to stable RLVR. This contrast with Muon is not due to update norm: Pion's update norm stays close to Muon's (**Appendix O**), as tail singular values contribute little to the norm. **Fig. 7** further shows that Pion achieves higher SNR than AdamW throughout training.

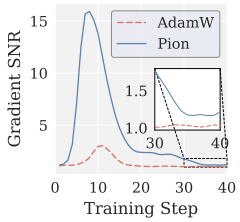
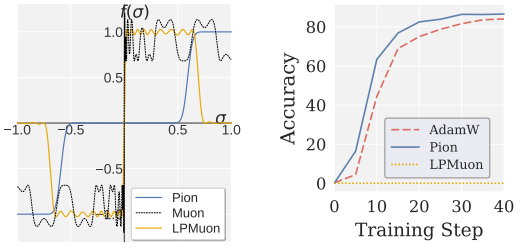


Figure 7: Gradient SNR of Pion vs. AdamW (Qwen3-1.7B, GRPO on GSM8K).

A reverse ablation: flipping the filter direction collapses on RLVR. To isolate the role of the *high-pass* design, we construct **Low-pass Muon** (LPMuon), a mirror of Pion that keeps its NS structure and per-step cost but flips the coefficients to induce a low-pass mapping (contracting large singular values and amplifying small ones) (**Appendix P**). **Fig. 8-(a)** confirms the resulting low-pass profile. Yet LPMuon fails to train: its accuracy stays at the initial checkpoint (**Fig. 8-(b)**). Together with Muon's failure (no filtering) in Fig. 6, this reverse ablation isolates the *direction* of spectral shaping as the key factor: Pion's gains arise specifically from high-pass filtering.



(a) Low-pass NS (b) GSM8K accuracy

Figure 8: (a) Scalar map $f(\sigma)$ of LPMuon for $\sigma \in [0, 1]$. (b) Accuracy of AdamW, Pion, and LPMuon (Qwen3-1.7B, GRPO on GSM8K).

7 CONCLUSION

We identified two limitations of Muon beyond LLM pretraining: lack of rank adaptiveness in cross-modality VLA training, and lack of noise adaptiveness in RLVR post-training. To address them, we proposed Pion, a drop-in replacement for Muon's NS iteration that uses a high-pass NS to preserve leading singular directions while suppressing the noisy tail, at the same per-step cost as Muon. Pion consistently outperforms AdamW and Muon across VLA training on LIBERO/LIBERO-Plus and RLVR post-training on Qwen3-1.7B/4B over MATH and GSM8K, including settings where Muon collapses. We discuss Pion's limitations (**Appendix Q**) and broader impacts (**Ethics statement**).

AI USE STATEMENT

In this work, large language models were used solely to polish the writing and check grammar. They did not contribute to research ideation, methods, experiments, analysis, or code. The authors reviewed all edited text and take full responsibility for the content of this paper.

ETHICS STATEMENT

This work studies optimizers and evaluates them on publicly available benchmarks (LIBERO, LIBERO-Plus, MATH, and GSM8K) and pretrained models. It involves no human-subject research, and the teleoperated demonstrations used for the real-robot experiments record only object-manipulation tasks and contain no personal data. On the positive side, Pion lowers the cost of training capable VLA policies and RLVR-tuned reasoning LLMs by stabilizing post-training under the same compute budget as Muon, which can broaden access to embodied agents and reasoning models. On the negative side, more capable VLA policies and reasoning LLMs carry the standard dual-use risks of robotic and language-based agents, including unsafe deployment and misuse for harmful content. We hope our work encourages further study of matrix-aware optimization beyond LLM pretraining alongside the safety practices already established for VLA and RLVR systems.

REFERENCES

Kwangjun Ahn, Byron Xu, Natalie Abreu, Ying Fan, Gagik Magakyan, Pratyusha Sharma, Zheng Zhan, and John Langford. Dion: Distributed orthonormalized updates. *arXiv preprint arXiv:2504.05295*, 2025.

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

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

Chongyu Fan, Yihua Zhang, Jinghan Jia, Alfred O. Hero, and Sijia Liu. Cyclicreflex: Improving reasoning models via cyclical reflection token scheduling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=4oOF4J2xSy.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Yulu Gan and Phillip Isola. Neural thickets: Diverse task experts are dense around pretrained weights. *arXiv preprint arXiv:2603.12228*, 2026.

Ankit Goyal, Hugo Hadfield, Xuning Yang, Valts Blukis, and Fabio Ramos. Vla-0: Building state-of-the-art vlas with zero modification. *arXiv preprint arXiv:2510.13054*, 2025.

Rongzhi Gu, Lianwu Chen, Shi-Xiong Zhang, Jimeng Zheng, Yong Xu, Meng Yu, Dan Su, Yuexian Zou, and Dong Yu. Neural spatial filter: Target speaker speech separation assisted with directional information. In *Proc. Interspeech 2019*, pp. 4290–4294, 2019.

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.

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

Chuan He, Zhanwang Deng, and Zhaosong Lu. Low-rank orthogonalization for large-scale matrix optimization with applications to foundation model training. *arXiv preprint arXiv:2509.11983*, 2025a.

Wei He, Kai Han, Hang Zhou, Hanting Chen, Zhicheng Liu, Xinghao Chen, and Yunhe Wang. Root: Robust orthogonalized optimizer for neural network training. *arXiv preprint arXiv:2511.20626*, 2025b.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: Stabilizing critic-free policy optimization with global advantage normalization. *arXiv preprint arXiv:2501.03262*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

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

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

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*, pp. 611–626, 2023.

Yicheng Lang, Changsheng Wang, Yihua Zhang, Mingyi Hong, Zheng Zhang, Wotao Yin, and Sijia Liu. Powering up zeroth-order training via subspace gradient orthogonalization. *arXiv preprint arXiv:2602.17155*, 2026.

Pengyi Li, Elizaveta Goncharova, Andrey Kuznetsov, and Ivan Oseledets. Back to basics: Revisiting exploration in reinforcement learning for llm reasoning via generative probabilities. *arXiv preprint arXiv:2602.05281*, 2026.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024a.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024b.

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

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Liuao Pei, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Fanfan Liu, Youyang Yin, Peng Shi, Siqi Yang, Zhixiong Zeng, and Haibo Qiu. Length-unbiased sequence policy optimization: Revealing and controlling response length variation in rlvr. *arXiv preprint arXiv:2602.05261*, 2026.

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*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Hanyi Mao, Quanjia Xiao, Lei Pang, and Haixiao Liu. Clip your sequences fairly: Enforcing length fairness for sequence-level rl. *arXiv preprint arXiv:2509.09177*, 2025.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Rui Pan, Yang Luo, Yuxing Liu, Yang You, and Tong Zhang. Unbiased gradient low-rank projection. *arXiv preprint arXiv:2510.17802*, 2025.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Nicolas Le Roux, Marc G Bellemare, Jonathan Lebensold, Arnaud Bergeron, Joshua Greaves, Alex Fréchette, Carolyne Pelletier, Eric Thibodeau-Laufer, Sándor Toth, and Sam Work. Tapered off-policy reinforce: Stable and efficient reinforcement learning for llms. *arXiv preprint arXiv:2503.14286*, 2025.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pp. 606–610. IEEE, 2007.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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.

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. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

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

Zhenpeng Su, Leiyu Pan, Xue Bai, Dening Liu, Guanting Dong, Jiaming Huang, Wenping Hu, Fuzheng Zhang, Kun Gai, and Guorui Zhou. Klear-reasoner: Advancing reasoning capability via gradient-preserving clipping policy optimization. *arXiv preprint arXiv:2508.07629*, 2025.

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.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Yuzhang Shang, Ramana Rao Kompella, and Gaowen Liu. Real-time robot execution with masked action chunking. In *International Conference on Learning Representations (ICLR)*, 2026a.

Jiakang Wang, Runze Liu, Lei Lin, Wenping Hu, Xiu Li, Fuzheng Zhang, Guorui Zhou, and Kun Gai. Aspo: Asymmetric importance sampling policy optimization. *arXiv preprint arXiv:2510.06062*, 2025.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pp. 18638–18646, 2026b.

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. Taming momentum: Rethinking optimizer states through low-rank approximation. In *The Fourteenth International Conference on Learning Representations*, 2026c. URL https://openreview.net/forum?id=9Q0dNBYeEY.

Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025a.

Yuqing Wen, Hebei Li, Kefan Gu, Yucheng Zhao, Tiancai Wang, and Xiaoyan Sun. Llada-vla: Vision language diffusion action models. *arXiv preprint arXiv:2509.06932*, 2025b.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Xiao-Ming Wu, Bin Fan, Kang Liao, Jian-Jian Jiang, Runze Yang, Yihang Luo, Zhonghua Wu, Wei-Shi Zheng, and Chen Change Loy. Vlanext: Recipes for building strong vla models. *arXiv preprint arXiv:2602.18532*, 2026.

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.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Jianke Zhang, Xiaoyu Chen, Qiuyue Wang, Mingsheng Li, Yanjiang Guo, Yucheng Hu, Jiajun Zhang, Shuai Bai, Junyang Lin, and Jianyu Chen. Vlm4vla: Revisiting vision-language-models in vision-language-action models. *arXiv preprint arXiv:2601.03309*, 2026.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025a.

Yifan Zhang, Yifeng Liu, Huizhuo Yuan, Yang Yuan, Quanquan Gu, and Andrew Chi-Chih Yao. On the design of kl-regularized policy gradient algorithms for llm reasoning. *arXiv preprint arXiv:2505.17508*, 2025b.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, et al. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Haizhong Zheng, Jiawei Zhao, and Beidi Chen. Prosperity before collapse: How far can off-policy rl reach with stale data on llms? *arXiv preprint arXiv:2510.01161*, 2025b.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.

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.

APPENDIX

A Additional Preliminaries: VLA Training and RLVR Training **16**
A.1 VLA action heads and training objectives 16
A.2 RLVR training: GRPO and GMPO 16

B Low-rank Muon (LRMuon) Algorithm **17**

C SNR Analysis for SFT and RLVR **18**
C.1 Setup and gradient estimators 18
C.2 SFT variance and SNR 18
C.3 GRPO variance and SNR (on-policy) 19
C.4 On-policy SNR comparison 20
C.5 Additional SNR degradation in GRPO 20
C.6 Combined bound 20

D Directional SNR of Singular Directions **21**

E SVD Factorization of Newton–Schulz Polynomial Iteration **22**

F Derivation of the Promotion and Suppression Polynomials **23**
F.1 Promotion polynomial f_{p} 23
F.2 Suppression polynomial f_{s} 24

G The Pion Optimizer: Full Algorithmic Description **25**

H Effective Rank after Muon NS and Pion's High-Pass NS **26**

I Per-Head Norm Heterogeneity Affects Forward and Backward Computation **27**

J Detailed Training Setups for VLA and RLVR Experiments **28**

K Qualitative rollouts **30**
K.1 LIBERO Object 30
K.2 LIBERO Spatial 31
K.3 LIBERO Goal 32
K.4 LIBERO Long 33

L Real-Robot Evaluation **34**
L.1 Quantitative results 34
L.2 Visualization of real-robot rollouts 34

M Additional VLA Experiments **36**
M.1 Pion vs. LRMuon for VLA training 36
M.2 Per-head vs. default Pion on VLA 36
M.3 Modality-wise optimizer assignment on VLA 36

N Learning-Rate Sweep for RLVR **38**

O Update Norm of Muon and Pion **39**

P Low-pass Muon (LPMuon): Coefficient Design via Constrained Polynomial Fitting **40**

Q Limitations **43**

A Additional Preliminaries: VLA Training and RLVR Training

This appendix provides the formal definitions deferred from Sec. 3: the two representative VLA action heads (ℓ_1-regression and flow-matching) used as our cross-modality testbeds, and the two representative RLVR algorithms (GRPO and GMPO) used for our post-training experiments.

A.1 VLA action heads and training objectives

We consider two representative designs for the action head of a VLA policy, each instantiating a different way of modeling the action distribution conditioned on the multimodal input $(\mathbf{x}, \mathbf{c})$.

- *ℓ_1-regression head* (Wang et al., 2026b; Kim et al., 2025): A deterministic transformer maps the multimodal input $(\mathbf{x}, \mathbf{c})$ to a single action prediction $f_{\boldsymbol{\Theta}}(\mathbf{x}, \mathbf{c})$, trained with

$$\mathcal{L}_{\text{reg}}(\boldsymbol{\Theta}) \;=\; \mathbb{E}_{(\mathbf{x},\mathbf{c},\mathbf{a})\sim\mathcal{D}}\big[\, \| f_{\boldsymbol{\Theta}}(\mathbf{x}, \mathbf{c}) - \mathbf{a} \|_1 \,\big], \tag{A1}$$

  where $\| \cdot \|_1$ denotes the ℓ_1 norm.
- *Flow-matching head* (Lipman et al., 2022; Black et al., 2024): Rather than producing a single point estimate, the action head models the conditional distribution $p(\mathbf{a}|\mathbf{x}, \mathbf{c})$ via a continuous-time generative process that transports a Gaussian prior to the data action. Concretely, let $\mathbf{a}_1 := \mathbf{a}$ be the ground-truth action drawn from $\mathcal{D}$ and $\mathbf{a}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ be a noise sample. Along the linear interpolation path $\mathbf{a}_t = t\,\mathbf{a}_1 + (1-t)\,\mathbf{a}_0$ for $t \in [0, 1]$, the target velocity field is the constant displacement $\frac{d\mathbf{a}_t}{dt} = \mathbf{a}_1 - \mathbf{a}_0$. The action head parameterizes a conditional velocity field $v_{\boldsymbol{\Theta}_{\text{action}}}(\mathbf{a}_t, t|\mathbf{x}, \mathbf{c})$ that predicts this velocity, and is trained to regress the target via

$$\mathcal{L}_{\text{FM}}(\boldsymbol{\Theta}) \;=\; \mathbb{E}_{t\sim\mathcal{U}(0,1),\, \mathbf{a}_0\sim\mathcal{N}(\mathbf{0},\mathbf{I}),\, (\mathbf{x},\mathbf{c},\mathbf{a}_1)\sim\mathcal{D}}\Big[\, \big\| \, v_{\boldsymbol{\Theta}_{\text{action}}}(\mathbf{a}_t, t|\mathbf{x}, \mathbf{c}) - (\mathbf{a}_1 - \mathbf{a}_0) \big\|_2^2 \,\Big], \tag{A2}$$

  where $t \sim \mathcal{U}(0, 1)$ denotes the uniform distribution over the interpolation timestep.

In our experiments (**Sec. 6**), the ℓ_1-regression head is instantiated by VLA-Adapter (Wang et al., 2026b) and the flow-matching head by VLANeXt (Wu et al., 2026).

A.2 RLVR training: GRPO and GMPO

We expand here on the three-stage RLVR loop sketched in Sec. 3. At each iteration, RLVR alternates between (a) *rollout*: for each prompt $\mathbf{q}$, a group of g responses $\{\mathbf{o}_i\}_{i=1}^{g}$ is sampled from the old policy π_{old}; (b) *scoring*: each response $\mathbf{o}_i$ is assigned a scalar reward by a programmatic verifier, and the rewards within a group are normalized into a *group-relative advantage* $\hat{a}_i \in \mathbb{R}$; (c) *policy update*: $\pi_{\boldsymbol{\Theta}}$ is optimized through a clipped importance-ratio objective. We study two representative policy-gradient algorithms, GRPO and GMPO, which differ in how they aggregate the per-token importance ratio $r_{i,t}$. Throughout, we denote by $\text{clip}(x, l, u) := \min\big(\max(x, l),\, u\big)$ the standard clipping operator that confines a scalar $x \in \mathbb{R}$ to the interval $[l, u]$.

- *GRPO* (Shao et al., 2024) aggregates the ratio at the token level via arithmetic averaging $r_{i,t}(\boldsymbol{\Theta}) := \pi_{\boldsymbol{\Theta}}(o_{i,t}|\mathbf{q}, \mathbf{o}_{i,<t}) \,/\, \pi_{\text{old}}(o_{i,t}|\mathbf{q}, \mathbf{o}_{i,<t})$, where $o_{i,t}$ denotes the t-th token of $\mathbf{o}_i$ and $\mathbf{o}_{i,<t}$ its preceding prefix:

$$\mathcal{J}_{\text{GRPO}}(\boldsymbol{\Theta}) = \mathbb{E}_{\mathbf{q},\,\{\mathbf{o}_i\}}\left[\frac{1}{g}\sum_{i=1}^{g}\frac{1}{|\mathbf{o}_i|}\sum_{t=1}^{|\mathbf{o}_i|}\min\Big(\, r_{i,t}(\boldsymbol{\Theta})\,\hat{a}_i,\;\; \text{clip}\big(r_{i,t}(\boldsymbol{\Theta}),\; 1-\epsilon,\; 1+\epsilon\big)\,\hat{a}_i \,\Big)\right]. \tag{A3}$$

- *GMPO* (Zhao et al., 2025) replaces the token-level arithmetic mean with the geometric mean of the per-token clipped objectives of each response (clipping per token, in log space):

$$\mathcal{J}_{\text{GMPO}}(\boldsymbol{\Theta}) = \mathbb{E}_{\mathbf{q},\{\mathbf{o}_i\}}\left[\frac{1}{g}\sum_{i=1}^{g}\left(\prod_{t=1}^{|\mathbf{o}_i|}\Big|\min\big(r_{i,t}(\boldsymbol{\Theta})\hat{a}_i,\; \text{clip}(r_{i,t}(\boldsymbol{\Theta}), e^{-\epsilon}, e^{\epsilon})\,\hat{a}_i\big)\Big|\right)^{1/|\mathbf{o}_i|}\cdot \text{sign}(\hat{a}_i)\right]. \tag{A4}$$

B LOW-RANK MUON (LRMUON) ALGORITHM

We provide the full pseudocode for **Low-rank Muon** (LRMuon), used as a baseline in Sec. 4 and Sec. 6.2. LRMuon follows the standard Muon optimization loop (1), but replaces the NS approximation to $\mathrm{msign}(\mathbf{M}_t)$ (2) with an exact SVD-based **top-k polar factor**. Concretely, given the compact SVD $\mathbf{M}_t = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, LRMuon truncates to the top-k singular subspace $(\mathbf{U}_k, \mathbf{V}_k)$ and uses the partial-isometry update $\mathbf{U}_k\mathbf{V}_k^\top$. The full procedure is summarized in Alg. 1.

Algorithm 1 LRMuon Optimizer

Require: Learning rate η, momentum coefficient μ, target rank k

$\mathbf{M}_0 \leftarrow \mathbf{0}$
for $t = 1, 2, \ldots$ **do**
  $\mathbf{G}_t \leftarrow \nabla_{\mathbf{\Theta}}\mathcal{L}_t(\mathbf{\Theta}_{t-1})$; $\mathbf{M}_t \leftarrow \mu\,\mathbf{M}_{t-1} + \mathbf{G}_t$ $(\mathbf{M}_t \in \mathbb{R}^{m\times n})$
  $\mathbf{U},\ \mathbf{\Sigma},\ \mathbf{V}^\top \leftarrow \mathrm{SVD}(\mathbf{M}_t)$
  $k_{\text{eff}} \leftarrow \min\big(k,\ \mathrm{rank}(\mathbf{M}_t)\big)$
  $\mathbf{U}_k \leftarrow \mathbf{U}_{:,\,1:k_{\text{eff}}}$; $\mathbf{V}_k^\top \leftarrow (\mathbf{V}^\top)_{1:k_{\text{eff}},\,:}$
  $\mathbf{X} \leftarrow \mathbf{U}_k\mathbf{V}_k^\top$
  $\mathbf{\Theta}_t \leftarrow \mathbf{\Theta}_{t-1} - \eta\,\mathbf{X}$
end for
return $\mathbf{\Theta}_t$

C SNR ANALYSIS FOR SFT AND RLVR

This appendix justifies the empirical observation in Sec. 4 that RLVR has a much lower gradient SNR than SFT. We derive closed-form expressions for the per-step SNR of both estimators under matched batch size, and then account for the additional noise sources that are unique to RLVR.

C.1 SETUP AND GRADIENT ESTIMATORS

We adopt the notation of Sec. 3 and Appendix A.2. For a prompt $\mathbf{q}$, the old policy π_{old} produces a group of g responses $\{\mathbf{o}_i\}_{i=1}^{g}$ with lengths $|\mathbf{o}_i|$, and a verifier assigns binary rewards $R_i \in \{0,1\}$. Throughout, denote

$$\ell_{i,t}(\boldsymbol{\Theta}) = \log \pi_{\boldsymbol{\Theta}}(o_{i,t} \mid \mathbf{q}, \mathbf{o}_{i,<t}), \qquad r_{i,t}(\boldsymbol{\Theta}) = \frac{\pi_{\boldsymbol{\Theta}}(o_{i,t} \mid \mathbf{q}, \mathbf{o}_{i,<t})}{\pi_{\text{old}}(o_{i,t} \mid \mathbf{q}, \mathbf{o}_{i,<t})}, \qquad \hat{a}_i = \frac{R_i - \bar{R}}{\text{std}(R) + \epsilon}, \tag{A5}$$

where $\bar{R} = \frac{1}{g}\sum_j R_j$ and $\text{std}(R)^2 = \frac{1}{g}\sum_j (R_j - \bar{R})^2$. Throughout this appendix, for a random vector $\mathbf{X}$ we write $\text{Var}(\mathbf{X}) := \mathbb{E}\|\mathbf{X} - \mathbb{E}[\mathbf{X}]\|^2 = \text{tr}(\text{Cov}(\mathbf{X}))$ for its total scalar variance, which coincides with the Frobenius-based denominator of the main-text SNR (5) once the gradient matrix is vectorized. Two identities that we invoke repeatedly follow directly from (A5):

$$\sum_{i=1}^{g} \hat{a}_i \;=\; 0, \qquad \frac{1}{g}\sum_{i=1}^{g} \hat{a}_i^2 \;=\; \frac{\text{std}(R)^2}{(\text{std}(R)+\epsilon)^2} \;\approx\; 1. \tag{A6}$$

SFT estimator. On a labelled pair $(\mathbf{q}, \mathbf{o}^\star)$ of length T, the per-sample loss is $-\log \pi_{\boldsymbol{\Theta}}(\mathbf{o}^\star \mid \mathbf{q}) = -\sum_{t=1}^{T} \ell_t(\boldsymbol{\Theta})$, so the batch estimator over g i.i.d. examples is

$$\hat{\mathbf{g}}_{\text{SFT}} \;=\; -\frac{1}{g}\sum_{j=1}^{g}\sum_{t=1}^{T} \nabla_{\boldsymbol{\Theta}} \ell_{j,t}(\boldsymbol{\Theta}), \tag{A7}$$

with a *deterministic* coefficient -1 on every token.

GRPO estimator. Differentiating the GRPO objective (A3) inside the unclipped branch of the $\min$, and applying the log-derivative identity $\nabla_{\boldsymbol{\Theta}} r_{i,t}(\boldsymbol{\Theta}) = r_{i,t}(\boldsymbol{\Theta})\, \nabla_{\boldsymbol{\Theta}} \ell_{i,t}(\boldsymbol{\Theta})$, gives

$$\nabla_{\boldsymbol{\Theta}} \mathcal{J}_{\text{GRPO}}(\boldsymbol{\Theta}) \;=\; \mathbb{E}_{\mathbf{q},\{\mathbf{o}_i\}}\left[\frac{1}{g}\sum_{i=1}^{g} \frac{1}{|\mathbf{o}_i|}\sum_{t=1}^{|\mathbf{o}_i|} \mathbb{1}\!\!\!\!\mathbb{1}_{i,t}\, \hat{a}_i\, r_{i,t}(\boldsymbol{\Theta})\, \nabla_{\boldsymbol{\Theta}} \ell_{i,t}(\boldsymbol{\Theta})\right], \tag{A8}$$

where $\mathbb{1}\!\!\!\!\mathbb{1}_{i,t} \in \{0,1\}$ is the active indicator picking out tokens for which the $\min$ in (A3) selects the unclipped branch. In the *on-policy regime* $\boldsymbol{\Theta} = \boldsymbol{\Theta}_{\text{old}}$ we have $r_{i,t} \equiv 1$ and $\mathbb{1}\!\!\!\!\mathbb{1}_{i,t} \equiv 1$, so (A8) reduces to

$$\hat{\mathbf{g}}_{\text{GRPO}} \;=\; \frac{1}{g}\sum_{i=1}^{g} \hat{a}_i\, \bar{\mathbf{S}}_i, \qquad \bar{\mathbf{S}}_i \;:=\; \frac{1}{|\mathbf{o}_i|}\sum_{t=1}^{|\mathbf{o}_i|} \nabla_{\boldsymbol{\Theta}} \ell_{i,t}(\boldsymbol{\Theta}). \tag{A9}$$

Regularity assumptions. For notational simplicity we treat all responses as having a common representative length T (the lengths $|\mathbf{o}_i|$ are replaced by T throughout; the analysis goes through when T is interpreted as the average length, as long as $\|\bar{\mathbf{s}}\|^2 \ll \sigma_s^2$). We assume throughout that (i) per-token scores have constant variance $\text{Var}(\nabla_{\boldsymbol{\Theta}} \ell_{i,t}) = \sigma_s^2$ across token positions and are uncorrelated across time steps; (ii) the rewards are i.i.d., $R_i \sim \text{Bern}(p)$ with $p = p(\mathbf{q}) \in (0,1)$; and (iii) conditional on $\{R_i\}$, the trajectories $\{\mathbf{o}_i\}$ are independent and the residual $\bar{\mathbf{S}}_i - \mathbb{E}[\bar{\mathbf{S}}_i \mid R_i]$ has mean zero with second moment σ_s^2/T.

C.2 SFT VARIANCE AND SNR

Signal. Taking expectation in (A7) with $\bar{\mathbf{s}} := \mathbb{E}[\nabla_{\boldsymbol{\Theta}} \ell_{i,t}]$,

$$\mathbb{E}[\hat{\mathbf{g}}_{\text{SFT}}] \;=\; -\frac{1}{g} \cdot g \cdot T\, \bar{\mathbf{s}} \;=\; -T\, \bar{\mathbf{s}}, \qquad \|\mathbb{E}[\hat{\mathbf{g}}_{\text{SFT}}]\|^2 \;=\; T^2 \|\bar{\mathbf{s}}\|^2. \tag{A10}$$

Variance. Samples are independent and tokens within a sample are uncorrelated, so

$$\mathrm{Var}(\hat{\mathbf{g}}_{\mathrm{SFT}}) \;=\; \frac{1}{g^2}\sum_{j=1}^{g}\mathrm{Var}\Big(\sum_{t=1}^{T}\nabla_{\boldsymbol{\Theta}}\ell_{j,t}\Big) \;=\; \frac{1}{g}\cdot\sum_{t=1}^{T}\mathrm{Var}(\nabla_{\boldsymbol{\Theta}}\ell_{1,t}) \;=\; \frac{T\,\sigma_s^2}{g}, \tag{A11}$$

where the last equality treats σ_s^2 as the per-token noise scale. Combining (A10)–(A11),

$$\mathrm{SNR}_{\mathrm{SFT}} \;:=\; \frac{\|\mathbb{E}[\hat{\mathbf{g}}_{\mathrm{SFT}}]\|^2}{\mathrm{Var}(\hat{\mathbf{g}}_{\mathrm{SFT}})} \;=\; \frac{T^2\|\bar{\mathbf{s}}\|^2}{T\sigma_s^2/g} \;=\; g\,T\,\frac{\|\bar{\mathbf{s}}\|^2}{\sigma_s^2}. \tag{A12}$$

C.3 GRPO VARIANCE AND SNR (ON-POLICY)

To isolate the reward-dependent part of $\bar{\mathbf{S}}_i$, decompose

$$\bar{\mathbf{S}}_i \;=\; \underbrace{\mathbb{E}[\bar{\mathbf{S}}_i \mid R_i]}_{\mathbf{u}_i} \;+\; \underbrace{\bar{\mathbf{S}}_i - \mathbb{E}[\bar{\mathbf{S}}_i \mid R_i]}_{\mathbf{v}_i}, \qquad \mathbf{u}_i = \boldsymbol{\mu}_S^- + R_i\,\boldsymbol{\Delta}, \tag{A13}$$

where $\boldsymbol{\mu}_S^+ := \mathbb{E}[\bar{\mathbf{S}}_i \mid R_i = 1]$, $\boldsymbol{\mu}_S^- := \mathbb{E}[\bar{\mathbf{S}}_i \mid R_i = 0]$, and $\boldsymbol{\Delta} := \boldsymbol{\mu}_S^+ - \boldsymbol{\mu}_S^-$ is the expected score gap between successful and failed trajectories. Assumption (iii) gives $\mathbb{E}\|\mathbf{v}_i\|^2 = \sigma_s^2/T$.

Signal. Substituting (A13) into (A9),

$$\hat{\mathbf{g}}_{\mathrm{GRPO}} \;=\; \frac{1}{g}\sum_{i=1}^{g}\hat{a}_i\,\mathbf{u}_i \;+\; \frac{1}{g}\sum_{i=1}^{g}\hat{a}_i\,\mathbf{v}_i. \tag{A14}$$

Using $\sum_i \hat{a}_i = 0$ from (A6), the reward-dependent term becomes

$$\frac{1}{g}\sum_{i=1}^{g}\hat{a}_i\mathbf{u}_i \;=\; \left(\frac{1}{g}\sum_{i=1}^{g}\hat{a}_i R_i\right)\boldsymbol{\Delta}. \tag{A15}$$

For finite group size, this coefficient depends on the number of successful responses $K := \sum_{i=1}^{g} R_i$. Ignoring the small ϵ in the normalization, degenerate groups with $K \in \{0, g\}$ have zero advantage and hence contribute no signal. For non-degenerate groups, $1 \le K \le g-1$,

$$\frac{1}{g}\sum_{i=1}^{g}\hat{a}_i R_i \;=\; \sqrt{\frac{K}{g}\left(1-\frac{K}{g}\right)}. \tag{A16}$$

Thus, with $K \sim \mathrm{Binomial}(g, p)$ and

$$q_{\mathrm{nd}} := \Pr(0 < K < g) = 1 - p^g - (1-p)^g, \qquad \rho_g(p) := \mathbb{E}\left[\sqrt{\frac{K}{g}\left(1-\frac{K}{g}\right)}\,\middle|\, 0 < K < g\right], \tag{A17}$$

the first term has expectation $q_{\mathrm{nd}}\rho_g(p)\boldsymbol{\Delta}$, while the second term has zero expectation by assumption (iii). In the large-g regime with p bounded away from 0 and 1, $q_{\mathrm{nd}} \to 1$ and $\rho_g(p) \to \sqrt{p(1-p)}$, recovering the simpler approximation used in the main text. Therefore

$$\mathbb{E}[\hat{\mathbf{g}}_{\mathrm{GRPO}}] \;\approx\; q_{\mathrm{nd}}\rho_g(p)\,\boldsymbol{\Delta}, \qquad \|\mathbb{E}[\hat{\mathbf{g}}_{\mathrm{GRPO}}]\|^2 \;\approx\; q_{\mathrm{nd}}^2\rho_g(p)^2\,\|\boldsymbol{\Delta}\|^2. \tag{A18}$$

Variance. The first term in (A14) contributes $O(\|\boldsymbol{\Delta}\|^2/g)$, while the second contributes $O(\sigma_s^2/(gT))$; in the low-SNR regime $T\|\boldsymbol{\Delta}\|^2 \ll \sigma_s^2$ the second term dominates. Conditioning on $\{R_i\}$ (which fixes the $\hat{a}_i$) and using assumption (iii),

$$\mathbb{E}\left\|\frac{1}{g}\sum_{i=1}^{g}\hat{a}_i\,\mathbf{v}_i\right\|^2 \;=\; \frac{1}{g^2}\sum_{i=1}^{g}\hat{a}_i^2\,\mathbb{E}\|\mathbf{v}_i\|^2 \;=\; \frac{\sigma_s^2}{gT}\cdot \underbrace{\frac{1}{g}\sum_{i=1}^{g}\hat{a}_i^2}_{\approx 1 \text{ for non-degenerate groups, } 0 \text{ otherwise}} \;\approx\; \frac{q_{\mathrm{nd}}\,\sigma_s^2}{gT}. \tag{A19}$$

Combining (A18) and (A19),

$$\mathrm{SNR}_{\mathrm{GRPO}} \;=\; \frac{\|\mathbb{E}[\hat{\mathbf{g}}_{\mathrm{GRPO}}]\|^2}{\mathrm{Var}(\hat{\mathbf{g}}_{\mathrm{GRPO}})} \;\approx\; g\,T\,\frac{\kappa_g(p)\,\|\boldsymbol{\Delta}\|^2}{\sigma_s^2}, \qquad \kappa_g(p) := q_{\mathrm{nd}}\rho_g(p)^2. \tag{A20}$$

C.4 On-policy SNR comparison

Dividing (A12) by (A20),

$$\frac{\mathrm{SNR}_{\mathrm{SFT}}}{\mathrm{SNR}_{\mathrm{GRPO}}} \;\approx\; \frac{\|\bar{\mathbf{s}}\|^2}{\kappa_g(p)\,\|\boldsymbol{\Delta}\|^2}. \tag{A21}$$

Two regimes drive this ratio large: (i) extreme difficulty $p \to 0$ or $p \to 1$, where the effective reward signal $\kappa_g(p)$ vanishes because many groups become degenerate and the within-group success/failure contrast disappears; and (ii) low distinctiveness $\|\boldsymbol{\Delta}\| \ll \|\bar{\mathbf{s}}\|$, where successful and failed rollouts produce nearly identical score directions. In the large-g non-degenerate approximation, $\kappa_g(p) \approx p(1-p)$, recovering the simpler ratio $\|\bar{\mathbf{s}}\|^2/(p(1-p)\|\boldsymbol{\Delta}\|^2)$. These are exactly the failure modes that dynamic sampling (Yu et al., 2025) and mean-only normalization (Liu et al., 2025b) are designed to mitigate.

C.5 Additional SNR degradation in GRPO

The on-policy bound (A19) is optimistic: practical GRPO runs deviate from on-policy and lose signal through clipping and degenerate reward groups, neither of which has an SFT counterpart.

Importance-sampling amplification. When $\boldsymbol{\Theta} \neq \boldsymbol{\Theta}_{\mathrm{old}}$, each token gradient in (A8) is weighted by $r_{i,t}$. Assuming that magnitudes and directions of $r_{i,t}$ and $\nabla_{\boldsymbol{\Theta}}\ell_{i,t}$ factorize in second moment,

$$\mathbb{E}\|r_{i,t}\,\nabla_{\boldsymbol{\Theta}}\ell_{i,t}\|^2 \;=\; \mathbb{E}_{\pi_{\mathrm{old}}}[r_{i,t}^2]\cdot\mathbb{E}\|\nabla_{\boldsymbol{\Theta}}\ell_{i,t}\|^2 \;=\; (1+\chi^2)\,\sigma_s^2, \qquad \chi^2 \;:=\; \mathbb{E}_{\pi_{\mathrm{old}}}[r_{i,t}^2]-1, \tag{A22}$$

where χ^2 is the per-token chi-squared divergence between $\pi_{\boldsymbol{\Theta}}$ and π_{old} (equivalently, $e^{D_2(\pi_{\boldsymbol{\Theta}}\|\pi_{\mathrm{old}})}-1$, where D_2 denotes the Rényi-2 divergence); it equals zero on-policy and grows with every inner gradient step. Thus, off-policy updates multiply the variance term in (A19) by $(1+\chi^2)$.

Clipping-induced signal loss. Let $\alpha = \Pr(\mathbb{1}_{i,t} = 0)$ be the clip fraction. Modeling $\mathbb{1}_{i,t}$ as a Bernoulli$(1-\alpha)$ mask independent of the per-token score (valid in the mean-field sense), under random masking the conditional expectation of the per-response score $\bar{\mathbf{S}}_i$ scales by $(1-\alpha)$ while its variance scales by $(1-\alpha)$ as well, so signal-squared contributes a factor $(1-\alpha)^2$ and variance contributes $(1-\alpha)$, giving a net $(1-\alpha)$ attenuation on the GRPO SNR (equivalently, substituting the effective length $T \to (1-\alpha)T$ into (A20)). This attenuation appears in the SFT/GRPO SNR ratio as

$$\frac{1}{1-\alpha}. \tag{A23}$$

Degenerate reward groups. For binary rewards, group normalization provides a useful advantage only when a group contains both successes and failures. As reflected in $\kappa_g(p)$ above, the probability of such a non-degenerate group is

$$q_{\mathrm{nd}} \;=\; 1 - p^g - (1-p)^g. \tag{A24}$$

When $p \to 0$ or $p \to 1$, q_{nd} becomes small: many groups have zero reward variance, hence zero normalized advantage and no learning signal. This reduces the effective batch size and weakens the GRPO signal, beyond the large-g approximation $p(1-p)$.

C.6 Combined bound

Combining the on-policy variance (A19) with the importance-sampling factor (A22) and the clipping attenuation (A23), the off-policy variance and SNR ratio satisfy

$$\mathrm{Var}(\hat{\mathbf{g}}^{\mathrm{full}}_{\mathrm{GRPO}}) \gtrsim \frac{q_{\mathrm{nd}}\sigma_s^2}{gT}\cdot\underbrace{(1+\chi^2)}_{\text{IS}}\cdot\underbrace{(1-\alpha)}_{\text{clip}}, \qquad \|\mathbb{E}[\hat{\mathbf{g}}^{\mathrm{full}}_{\mathrm{GRPO}}]\|^2 \approx (1-\alpha)^2 q_{\mathrm{nd}}^2\rho_g(p)^2\|\boldsymbol{\Delta}\|^2. \tag{A25}$$

Importance weights leave the signal unchanged ($\mathbb{E}_{\mathrm{old}}[r]=1$), while clipping enters the squared signal twice and the variance once (modeling the clip mask as independent of r); dividing gives (A26).

$$\frac{\mathrm{SNR}_{\mathrm{SFT}}}{\mathrm{SNR}^{\mathrm{full}}_{\mathrm{GRPO}}} \;\gtrsim\; \underbrace{\frac{\|\bar{\mathbf{s}}\|^2}{\kappa_g(p)\,\|\boldsymbol{\Delta}\|^2}}_{\text{credit assignment}}\cdot(1+\chi^2)\cdot\frac{1}{1-\alpha}. \tag{A26}$$

D DIRECTIONAL SNR OF SINGULAR DIRECTIONS

The SNR (5) used in Fig. 2-(a) and Fig. 7 aggregates signal and noise over the whole gradient matrix. To resolve how the noise is distributed across singular directions, we additionally measure a *directional* SNR (dSNR) that quantifies the strength of the optimization signal relative to stochastic sampling noise along each singular direction (Gu et al., 2019).

Definition. Let $\mathbf{M}_{t-1} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be the SVD of the momentum before the current optimizer step, with left and right singular vectors $\mathbf{u}_i$ and $\mathbf{v}_i$, and let $\mathbf{D}_i := \mathbf{u}_i\mathbf{v}_i^\top$ denote the i-th singular direction. We project each of B independently sampled stochastic gradients $\{\mathbf{G}_b\}_{b=1}^{B}$ onto $\mathbf{D}_i$,

$$a_{b,i} := \langle \mathbf{G}_b, \mathbf{D}_i \rangle_{\mathrm{F}} = \mathbf{u}_i^\top \mathbf{G}_b \mathbf{v}_i, \qquad b = 1, \ldots, B, \tag{A27}$$

with $B = 64$ in our experiments, and compute

$$\bar{a}_i = \frac{1}{B}\sum_{b=1}^{B} a_{b,i}, \qquad s_i^2 = \frac{1}{B-1}\sum_{b=1}^{B}\big(a_{b,i} - \bar{a}_i\big)^2, \qquad \mathrm{dSNR}_i := \frac{\max\big(\bar{a}_i^2 - s_i^2/B,\, 0\big)}{s_i^2 + 10^{-12}}. \tag{A28}$$

Subtracting s_i^2/B corrects the finite-sample upward bias of $\bar{a}_i^2$ as an estimate of the squared mean projection. A smaller dSNR_i indicates that the i-th singular direction is more strongly dominated by noise.

Head vs. tail directions. **Table A1** reports the ratio between the dSNR of the head (top-$p\%$) and the tail (bottom-$p\%$) singular directions of $\mathbf{M}_{t-1}$ at training steps 20, 40, 60, and 80 of GRPO training of Qwen3-1.7B on MATH (setup of Fig. 6-(a)). The head dSNR exceeds the tail dSNR by three to four orders of magnitude throughout training, so the small singular directions are overwhelmingly noise-dominated. This refines the matrix-level SNR gap between SFT and GRPO in Fig. 2-(a): within the low-SNR GRPO gradient, the noise concentrates in the spectral tail, which Muon's NS lifts to unit magnitude and Pion's Suppression stage (8) drives toward 0, consistent with the higher gradient SNR of Pion in Fig. 7.

Table A1: Ratio of head (top-$p\%$) to tail (bottom-$p\%$) dSNR (A28) across training steps for Qwen3-1.7B trained with GRPO on MATH (setup of Fig. 6-(a)).

p	**Step 20**	**Step 40**	**Step 60**	**Step 80**
1%	1.1×10^4	4.0×10^4	2.4×10^4	5.8×10^4
10%	5.0×10^3	2.0×10^4	1.8×10^3	5.6×10^3

E SVD FACTORIZATION OF NEWTON–SCHULZ POLYNOMIAL ITERATION

This appendix provides the detailed derivation behind the claim in Sec. 5 that designing Pion's spectral high-pass at the matrix level reduces, via the SVD, to designing a scalar polynomial f on $[0, 1]$. We show that the odd matrix polynomial used by a single Newton–Schulz (NS) step factors through the SVD as a scalar polynomial acting entrywise on the singular values, so that designing the matrix filter is equivalent to designing three scalar coefficients (a, b, c). The chaining of multiple NS steps further composes these scalar polynomials, while leaving the singular vectors $(\mathbf{U}, \mathbf{V})$ unchanged throughout.

Setup. Let $\mathbf{X} \in \mathbb{R}^{m\times n}$ with $r := \mathrm{rank}(\mathbf{X})$, and let its *compact* singular value decomposition (consistent with (2)) be

$$\mathbf{X} \;=\; \mathbf{U}\,\mathbf{\Sigma}\,\mathbf{V}^\top, \qquad \mathbf{U} \in \mathbb{R}^{m\times r},\;\; \mathbf{V} \in \mathbb{R}^{n\times r},\;\; \mathbf{U}^\top\mathbf{U} = \mathbf{V}^\top\mathbf{V} = \mathbf{I}_r, \tag{A29}$$

where $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r) \succ 0$ collects the strictly positive singular values.

Polynomial iteration factors through the SVD. Consider the odd matrix polynomial used by a single quintic NS step:

$$\mathcal{P}(\mathbf{X};\, a, b, c) \;:=\; a\mathbf{X} + b\,\mathbf{X}\mathbf{X}^\top\mathbf{X} + c\,\mathbf{X}(\mathbf{X}^\top\mathbf{X})^2. \tag{A30}$$

Using the SVD (A29), we have $\mathbf{X}^\top\mathbf{X} = \mathbf{V}\mathbf{\Sigma}^2\mathbf{V}^\top$. Since the thin right singular vector matrix satisfies $\mathbf{V}\mathbf{V}^\top \neq \mathbf{I}_n$ in general, the Gram-power identity should be read for positive powers:

$$(\mathbf{X}^\top\mathbf{X})^k \;=\; \mathbf{V}\,\mathbf{\Sigma}^{2k}\,\mathbf{V}^\top \quad \text{for all } k \in \mathbb{N}_{\geq 1}. \tag{A31}$$

For $k \geq 1$, left-multiplying the Gram power by $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and using $\mathbf{V}^\top\mathbf{V} = \mathbf{I}_r$ yields $\mathbf{X}(\mathbf{X}^\top\mathbf{X})^k = \mathbf{U}\mathbf{\Sigma}^{2k+1}\mathbf{V}^\top$; the same identity is immediate for $k = 0$. Hence the key identity is

$$\mathbf{X}(\mathbf{X}^\top\mathbf{X})^k \;=\; \mathbf{U}\,\mathbf{\Sigma}^{2k+1}\,\mathbf{V}^\top. \tag{A32}$$

Substituting (A32) into (A30), the matrix iteration collapses to

$$\mathcal{P}(\mathbf{X};\, a, b, c) \;=\; \mathbf{U}\,\underbrace{\left(a\,\mathbf{\Sigma} + b\,\mathbf{\Sigma}^3 + c\,\mathbf{\Sigma}^5\right)}_{f(\mathbf{\Sigma};\, a,b,c)}\,\mathbf{V}^\top \;=\; \mathbf{U}\,f(\mathbf{\Sigma};\, a, b, c)\,\mathbf{V}^\top, \tag{A33}$$

where $f(\sigma; a, b, c) = a\sigma + b\sigma^3 + c\sigma^5$ is the scalar polynomial from (6) and $f(\mathbf{\Sigma})$ is understood as applying f entrywise to the diagonal of $\mathbf{\Sigma}$. Equation (A33) has three important consequences:

- **Per-singular-value control.** The matrix map $\mathbf{X} \mapsto \mathcal{P}(\mathbf{X})$ is exactly equivalent to the scalar map $\sigma_i \mapsto f(\sigma_i)$ applied independently to each singular value.
- **Invariance of singular vectors.** The left and right singular vectors $\mathbf{U}$ and $\mathbf{V}$ are preserved unchanged; only the singular values are reshaped.
- **Reduction to a 3-dim. coefficient design.** Specifying the full matrix-level filter reduces to specifying the three scalar coefficients (a, b, c) that encode the desired shape of f on $[0, 1]$.

Composition of NS steps. Composing t NS steps $\mathcal{P}_t \circ \cdots \circ \mathcal{P}_1$ simply composes the scalar polynomials. If step $\mathcal{P}_i$ uses coefficients (a_i, b_i, c_i) and induces the scalar map f_i, then by repeatedly applying (A33),

$$\big(\mathcal{P}_t \circ \cdots \circ \mathcal{P}_1\big)(\mathbf{X}) \;=\; \mathbf{U}\,\big(f_t \circ \cdots \circ f_1\big)(\mathbf{\Sigma})\,\mathbf{V}^\top. \tag{A34}$$

This is exactly the chaining mechanism exploited by Pion to compose Promotion (7) for k_p steps and Suppression (8) for k_s steps into a single composite high-pass $f_\mathrm{s}^{\circ k_\mathrm{s}} \circ f_\mathrm{p}^{\circ k_\mathrm{p}}$ acting entrywise on $\mathbf{\Sigma}$, while leaving $(\mathbf{U}, \mathbf{V})$ untouched throughout.

Conclusion. The SVD factorization (A33) reduces the problem of designing a matrix-level spectral filter to the problem of designing a scalar polynomial f on $[0, 1]$. This justifies the treatment in Sec. 5, where the entire Pion design (Promotion plus Suppression) is specified through scalar coefficients $(a_\mathrm{p}, b_\mathrm{p}, c_\mathrm{p})$ and $(a_\mathrm{s}, b_\mathrm{s}, c_\mathrm{s})$ acting on the normalized singular spectrum, with the singular vectors $(\mathbf{U}, \mathbf{V})$ of the gradient preserved exactly throughout the iteration.

F Derivation of the Promotion and Suppression Polynomials

Setup. Recall from (6) the odd quintic scalar map that any single NS step induces on each normalized singular value $\sigma \in [0, 1]$:

$$f(\sigma;\, a, b, c) \;=\; a\,\sigma + b\,\sigma^3 + c\,\sigma^5, \qquad f'(\sigma) \;=\; a + 3b\,\sigma^2 + 5c\,\sigma^4, \qquad f''(\sigma) \;=\; 6b\,\sigma + 20c\,\sigma^3. \tag{A35}$$

The Pion design problem is to choose two sets of coefficients $(a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}})$ and $(a_{\mathrm{s}}, b_{\mathrm{s}}, c_{\mathrm{s}})$ such that the chained iteration $f_{\mathrm{s}}^{\circ k_{\mathrm{s}}} \circ f_{\mathrm{p}}^{\circ k_{\mathrm{p}}}$ realizes a high-pass on $[0, 1]$.

F.1 Promotion polynomial f_{p}

Design constraints. The Promotion stage must satisfy three constraints:

- **(P1)** *Fixed point*: $f_{\mathrm{p}}(1) = 1$, i.e., any singular value already at 1 is left unchanged.
- **(P2)** *First-order stationarity*: $f'_{\mathrm{p}}(1) = 0$, so that small perturbations around the fixed point $\sigma = 1$ are not amplified.
- **(P3)** *Boundary concavity*: $f''_{\mathrm{p}}(1) \le 0$, which prevents the Promotion map from curving upward near the anchored fixed point and pushing nearby singular values outside the normalized spectral range.

We motivate (P3) as follows. Since (P2) makes $\sigma = 1$ a stationary point of f_{p}, the sign of $f''_{\mathrm{p}}(1)$ controls the local shape of f_{p} near $\sigma = 1$. If $f''_{\mathrm{p}}(1) > 0$, then f'_{p} is strictly increasing through 0 at $\sigma = 1$ and hence strictly negative just to the left of $\sigma = 1$; consequently f_{p} is locally decreasing as $\sigma \uparrow 1$, so values σ slightly below 1 are mapped to $f_{\mathrm{p}}(\sigma) > f_{\mathrm{p}}(1) = 1$, leaving the spectral budget $[0, 1]$. Imposing $f''_{\mathrm{p}}(1) \le 0$ rules out this upward curving; the strict case $f''_{\mathrm{p}}(1) < 0$ already gives a local maximum at $\sigma = 1$ via the standard second-derivative test, while the boundary case $f''_{\mathrm{p}}(1) = 0$ is degenerate at the second order and its consequences are pinned down by the global monotonicity analysis below. As we verify below, restricted to the one-parameter family fixed by (P1)–(P2), the boundary concavity (P3) together with a matching lower bound is in fact equivalent to global monotonicity of f_{p} on $[0, 1]$, so it preserves the relative ordering of singular values throughout. As an immediate corollary, the Promotion stage stays inside the spectral budget: $f_{\mathrm{p}}(\sigma) \le f_{\mathrm{p}}(1) = 1$ for all $\sigma \in [0, 1]$.

Step 1: reduction to a one-parameter family via (P1)–(P2). By (A35), conditions (P1) and (P2) yield

$$a_{\mathrm{p}} + b_{\mathrm{p}} + c_{\mathrm{p}} \;=\; 1, \qquad a_{\mathrm{p}} + 3b_{\mathrm{p}} + 5c_{\mathrm{p}} \;=\; 0. \tag{A36}$$

Solving for a_{p} and b_{p} in terms of c_{p},

$$b_{\mathrm{p}} \;=\; -\,\frac{1 + 4c_{\mathrm{p}}}{2}, \qquad a_{\mathrm{p}} \;=\; \frac{3 + 2c_{\mathrm{p}}}{2}. \tag{A37}$$

Step 2: applying (P3) to obtain feasible ranges of $(a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}})$. Substituting (A37) into the second-order derivative gives

$$f''_{\mathrm{p}}(1) \;=\; 6b_{\mathrm{p}} + 20c_{\mathrm{p}} \;=\; -\,3 + 8c_{\mathrm{p}}, \tag{A38}$$

so (P3) is equivalent to

$$c_{\mathrm{p}} \;\le\; 0.375. \tag{A39}$$

Next, we derive the conditions that ensure f_{p} is monotonically non-decreasing on $[0, 1]$. Setting $u := \sigma^2 \in [0, 1]$, define

$$g(u) \;:=\; f'_{\mathrm{p}}(\sigma) \;=\; a_{\mathrm{p}} + 3b_{\mathrm{p}}\,u + 5c_{\mathrm{p}}\,u^2. \tag{A40}$$

Then g is a quadratic in u with $g(1) = a_{\mathrm{p}} + 3b_{\mathrm{p}} + 5c_{\mathrm{p}} = 0$ by (P2). For $c_{\mathrm{p}} \neq 0$, this lets us factor g as

$$g(u) \;=\; 5\,c_{\mathrm{p}}\,(u - 1)(u - r), \qquad r \;=\; \frac{a_{\mathrm{p}}}{5\,c_{\mathrm{p}}} \;=\; \frac{3 + 2c_{\mathrm{p}}}{10\,c_{\mathrm{p}}}. \tag{A41}$$

Since $u - 1 \le 0$ for all $u \in [0, 1]$, the inequality $g(u) \ge 0$ is equivalent to $5c_{\mathrm{p}}\,(u - r) \le 0$ on $[0, 1]$. We split on the sign of c_{p}:

- If $c_{\mathrm{p}} > 0$, we need $u \le r$ for all $u \in [0,1]$, i.e. $r \ge 1$. From (A41), $r \ge 1 \iff 3 + 2c_{\mathrm{p}} \ge 10c_{\mathrm{p}} \iff c_{\mathrm{p}} \le 0.375$.
- If $c_{\mathrm{p}} < 0$, we need $u \ge r$ for all $u \in [0,1]$, i.e. $r \le 0$. From (A41), $r \le 0 \iff 3 + 2c_{\mathrm{p}} \ge 0 \iff c_{\mathrm{p}} \ge -1.5$.
- If $c_{\mathrm{p}} = 0$, then $g(u) = \frac{3}{2} - \frac{3}{2}u \ge 0$ on $[0,1]$.

Combining the three cases yields the feasible range

$$-1.5 \;\le\; c_{\mathrm{p}} \;\le\; 0.375 \quad\Longrightarrow\quad g(u) \ge 0 \text{ for all } u \in [0,1]. \tag{A42}$$

The upper bound in (A42) coincides with the local condition (A39) from (P3), and the lower bound corresponds (via (A37)) exactly to $a_{\mathrm{p}} \ge 0$. Hence, within the family pinned by (P1)–(P2), the boundary concavity (P3) together with $a_{\mathrm{p}} \ge 0$ is necessary and sufficient for global monotonicity of f_{p} on $[0,1]$.

Combining (A42) with (A37), we obtain the feasible coefficient ranges

$$0 \;\le\; a_{\mathrm{p}} \;\le\; 1.875, \qquad -1.25 \;\le\; b_{\mathrm{p}} \;\le\; 2.5, \qquad -1.5 \;\le\; c_{\mathrm{p}} \;\le\; 0.375. \tag{A43}$$

Step 3: choosing the largest feasible slope at the origin. The slope $a_{\mathrm{p}} = f_{\mathrm{p}}'(0)$ controls how aggressively a single Promotion step lifts small singular values $\sigma \approx 0$ into the regime where Suppression eventually anchors them at 1: since $f_{\mathrm{p}}(\sigma) \approx a_{\mathrm{p}}\,\sigma$ near the origin, small singular values are amplified by a factor of approximately a_{p} per step. We therefore choose a_{p} at its maximal feasible value, $a_{\mathrm{p}} = 1.875$, to promote rapid growth under a fixed budget of $k = 5$ NS iterations. This achieves equality in (A39), yielding $c_{\mathrm{p}} = 0.375$ and, by (A38), $f_{\mathrm{p}}''(1) = 0$. Substituting back into (A37) fixes

$$(a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}}) \;=\; (1.875,\, -1.25,\, 0.375), \tag{A44}$$

which recovers exactly (7). At these coefficients, the derivative simplifies to a perfect square,

$$f_{\mathrm{p}}'(\sigma) \;=\; 1.875 - 3.75\,\sigma^2 + 1.875\,\sigma^4 \;=\; 1.875\,\big(1-\sigma^2\big)^2 \;\ge\; 0 \qquad \forall\,\sigma \in [0,1], \tag{A45}$$

making f_{p} monotone non-decreasing on $[0,1]$ with f_{p}' vanishing only at the boundary $\sigma = 1$.

F.2 Suppression polynomial f_{s}

Design constraints. The Suppression stage inherits the fixed-point and first-order stationarity conditions at $\sigma = 1$ from Promotion in order to anchor the leading singular values at 1. In addition, it imposes a *spectral filtering* condition at the origin that strips the linear term, so that small singular values are driven toward 0 by the higher-order (σ^3, σ^5) terms. Concretely:

- **(S1)** *Fixed point*: $f_{\mathrm{s}}(1) = 1$.
- **(S2)** *First-order stationarity*: $f_{\mathrm{s}}'(1) = 0$.
- **(S3)** *Spectral filtering at the origin*: $f_{\mathrm{s}}'(0) = 0$, eliminating the linear term so that small singular values $\sigma \approx 0$ are pushed toward 0 by the higher-order terms.

By (A35), (S3) is equivalent to $a_{\mathrm{s}} = 0$. Substituting into (S1) and (S2) gives a 2×2 linear system in $(b_{\mathrm{s}}, c_{\mathrm{s}})$:

$$b_{\mathrm{s}} + c_{\mathrm{s}} \;=\; 1, \qquad 3b_{\mathrm{s}} + 5c_{\mathrm{s}} \;=\; 0, \tag{A46}$$

whose unique solution is $b_{\mathrm{s}} = 2.5$ and $c_{\mathrm{s}} = -1.5$. Combined with $a_{\mathrm{s}} = 0$, this yields

$$(a_{\mathrm{s}}, b_{\mathrm{s}}, c_{\mathrm{s}}) \;=\; (0,\, 2.5,\, -1.5), \tag{A47}$$

which recovers exactly (8). Unlike the Promotion stage, the Suppression coefficients are determined uniquely by (S1)–(S3) and admit no remaining degree of freedom. At these coefficients, the derivative factors as

$$f_{\mathrm{s}}'(\sigma) \;=\; 7.5\,\sigma^2 - 7.5\,\sigma^4 \;=\; 7.5\,\sigma^2\big(1-\sigma^2\big) \;\ge\; 0 \qquad \forall\,\sigma \in [0,1], \tag{A48}$$

so f_{s} is monotone non-decreasing on $[0,1]$ with f_{s}' vanishing only at the endpoints $\sigma \in \{0,1\}$. Hence Suppression also preserves the relative ordering of singular values, and the chained iteration $f_{\mathrm{s}}^{\circ k_{\mathrm{s}}} \circ f_{\mathrm{p}}^{\circ k_{\mathrm{p}}}$ is monotone on $[0,1]$.

G THE PION OPTIMIZER: FULL ALGORITHMIC DESCRIPTION

We provide the full pseudocode for Pion deferred from Sec. 5. Pion is a drop-in replacement for Muon: the only change is that the per-step Newton–Schulz orthogonalization (3) is replaced by our **high-pass NS**, which chains the Promotion polynomial f_{p} (7) and the Suppression polynomial f_{s} (8). The total iteration count is fixed to $k = 5$, split by $k_{\mathrm{p}} \in \{0, 1, \ldots, 5\}$ with $k_{\mathrm{s}} = k - k_{\mathrm{p}}$. The high-pass NS has two modes: a **default** mode applied to each weight matrix $\mathbf{M}_t \in \mathbb{R}^{m\times n}$ as a whole (Alg. 2), used for VLA training, and a **per-head** mode that splits each attention projection along the head dimension into sub-blocks $\{\mathbf{M}_t^h\}_{h=1}^H$ and runs the iteration independently per head (Alg. 3), used for RLVR post-training; the per-head mode adds only a single reshape on top of the default mode.

Algorithm 2 Pion Optimizer (**default mode**: high-pass NS on the whole matrix)

Require: Learning rate η, momentum coefficient μ, promotion steps k_{p}

$k_{\mathrm{s}} \leftarrow 5 - k_{\mathrm{p}}$; $\mathbf{M}_0 \leftarrow \mathbf{0}$ — Total iterations strictly fixed to $k = 5$
for $t = 1, 2, \ldots$ **do**
  $\mathbf{G}_t \leftarrow \nabla_{\mathbf{\Theta}} \mathcal{L}_t(\mathbf{\Theta}_{t-1})$; $\mathbf{M}_t \leftarrow \mu\, \mathbf{M}_{t-1} + \mathbf{G}_t$
  $\mathbf{X} \leftarrow \mathbf{M}_t / (\|\mathbf{M}_t\|_{\mathrm{F}} + \epsilon)$ — Spectral pre-normalization, cf. (3)
  for $i = 1, \ldots, k_{\mathrm{p}}$ **do** — **Stage 1: Promotion** (7), $(a_{\mathrm{p}}, b_{\mathrm{p}}, c_{\mathrm{p}}) = (1.875, -1.25, 0.375)$
    $\mathbf{X} \leftarrow a_{\mathrm{p}}\, \mathbf{X} + b_{\mathrm{p}}\, \mathbf{X}\mathbf{X}^\top \mathbf{X} + c_{\mathrm{p}}\, \mathbf{X}(\mathbf{X}^\top \mathbf{X})^2$
  end for
  for $j = 1, \ldots, k_{\mathrm{s}}$ **do** — **Stage 2: Suppression** (8), $(a_{\mathrm{s}}, b_{\mathrm{s}}, c_{\mathrm{s}}) = (0, 2.5, -1.5)$
    $\mathbf{X} \leftarrow a_{\mathrm{s}}\, \mathbf{X} + b_{\mathrm{s}}\, \mathbf{X}\mathbf{X}^\top \mathbf{X} + c_{\mathrm{s}}\, \mathbf{X}(\mathbf{X}^\top \mathbf{X})^2$
  end for
  $\mathbf{\Theta}_t \leftarrow \mathbf{\Theta}_{t-1} - \eta\, \mathbf{X}$
end for
return $\mathbf{\Theta}_t$

Algorithm 3 Pion Optimizer (**per-head mode**: per-head high-pass NS on attention projections)

Require: Learning rate η, momentum coefficient μ, promotion steps k_{p}, number of heads H

$k_{\mathrm{s}} \leftarrow 5 - k_{\mathrm{p}}$; $\mathbf{M}_0 \leftarrow \mathbf{0}$ — Total iterations strictly fixed to $k = 5$
for $t = 1, 2, \ldots$ **do**
  $\mathbf{G}_t \leftarrow \nabla_{\mathbf{\Theta}} \mathcal{L}_t(\mathbf{\Theta}_{t-1})$; $\mathbf{M}_t \leftarrow \mu\, \mathbf{M}_{t-1} + \mathbf{G}_t$
  $\{\mathbf{M}_t^h\}_{h=1}^H \leftarrow \mathrm{Reshape}(\mathbf{M}_t)$ — Split the attention projection along the head dim
  $\mathbf{X}^h \leftarrow \mathbf{M}_t^h / (\|\mathbf{M}_t^h\|_{\mathrm{F}} + \epsilon),\ \ \forall\, h \in \{1, \ldots, H\}$ — Per-head pre-normalization
  for $i = 1, \ldots, k_{\mathrm{p}}$ **do** — **Stage 1: Promotion** (7), batched over H
    $\mathbf{X}^h \leftarrow a_{\mathrm{p}}\, \mathbf{X}^h + b_{\mathrm{p}}\, \mathbf{X}^h (\mathbf{X}^h)^\top \mathbf{X}^h + c_{\mathrm{p}}\, \mathbf{X}^h \big((\mathbf{X}^h)^\top \mathbf{X}^h\big)^2,\ \ \forall\, h \in \{1, \ldots, H\}$
  end for
  for $j = 1, \ldots, k_{\mathrm{s}}$ **do** — **Stage 2: Suppression** (8), batched over H
    $\mathbf{X}^h \leftarrow a_{\mathrm{s}}\, \mathbf{X}^h + b_{\mathrm{s}}\, \mathbf{X}^h (\mathbf{X}^h)^\top \mathbf{X}^h + c_{\mathrm{s}}\, \mathbf{X}^h \big((\mathbf{X}^h)^\top \mathbf{X}^h\big)^2,\ \ \forall\, h \in \{1, \ldots, H\}$
  end for
  $\mathbf{X} \leftarrow \mathrm{Reshape}^{-1}\big(\{\mathbf{X}^h\}_{h=1}^H\big) \in \mathbb{R}^{m\times n}$
  $\mathbf{\Theta}_t \leftarrow \mathbf{\Theta}_{t-1} - \eta\, \mathbf{X}$
end for
return $\mathbf{\Theta}_t$

H EFFECTIVE RANK AFTER MUON NS AND PION'S HIGH-PASS NS

This appendix quantifies how the two orthogonalization maps reshape the spectrum of the update in practice: Muon's NS iteration (3), which drives every normalized singular value toward 1, and Pion's high-pass NS (Promotion (7) followed by Suppression (8)), which anchors the leading singular values at 1 and contracts the tail toward 0. We measure the erank (4) before and after each NS map over all 2D matrices of the *entire* VLA-Adapter model, along the training trajectory on LIBERO Object used in Fig. 1-(a) (4.5k steps, recorded every 900 steps). **Table A2** reports the erank ratio, *i.e.*, the erank after applying Muon NS or Pion's high-pass NS divided by the erank before processing, averaged over all 2D matrices. A ratio above 1 means that the map spreads the update energy over more singular directions, and a ratio below 1 means that it concentrates the energy on fewer directions.

Table A2: Erank ratio (after vs. before processing) of Muon NS and Pion's high-pass NS, computed over all 2D matrices of the entire VLA-Adapter model on LIBERO Object, along the training trajectory of Fig. 1-(a).

Transform	**Step 900**	**Step 1800**	**Step 2700**	**Step 3600**	**Step 4500**
Muon NS	20.9	18.6	14.2	14.2	13.0
Pion high-pass NS	0.070	0.073	0.058	0.046	0.039

Muon NS increases the erank by 13.0–20.9$\times$ throughout training, confirming that uniform spectral whitening lifts the low-energy tail of the spectrum to the same magnitude as its informative head (cf. Limitation 1 in Sec. 4). In contrast, Pion's high-pass NS reduces the erank to 3.9–7.3% of its input value, *i.e.*, it concentrates the update on the few leading singular directions, which is the intended spectral high-pass behavior (Fig. 3-(d)).

I PER-HEAD NORM HETEROGENEITY AFFECTS FORWARD AND BACKWARD COMPUTATION

We analyze how per-head norm heterogeneity, an empirical property of trained transformers (Fig. 4-(b)), affects both forward computation and gradient flow. This motivates per-head spectral filtering in place of whole-matrix filtering.

Notation. For clarity, we write the analysis for a standard multi-head attention layer. For grouped-query or multi-query attention, the same argument applies to each Q, K, and V projection along its own head dimension. Let $\mathbf{X} \in \mathbb{R}^{n\times d}$ denote the input sequence and let d_k be the head dimension. For head h, define $\mathbf{W}_Q^h, \mathbf{W}_K^h, \mathbf{W}_V^h \in \mathbb{R}^{d\times d_k}$ and $\mathbf{W}_O^h \in \mathbb{R}^{d_k\times d}$. The head computes $\mathbf{S}^h = \mathbf{X}\mathbf{W}_Q^h(\mathbf{X}\mathbf{W}_K^h)^\top/\sqrt{d_k}$, $\mathbf{A}^h = \mathrm{softmax}(\mathbf{S}^h)$ row-wise, $\mathbf{O}^h = \mathbf{A}^h\mathbf{X}\mathbf{W}_V^h$, and the layer output is $\mathbf{Z} = \sum_h \mathbf{O}^h\mathbf{W}_O^h$.

Proposition I.1 (Per-head norms modulate attention and gradients)**.** *For each head h, the following forward and backward norm couplings hold.*

(a) ***Forward.*** *The Q/K norms control attention* sharpness*: the logits admit the factorization*

$$\mathbf{S}^h = \underbrace{\frac{\|\mathbf{W}_Q^h\|_F\,\|\mathbf{W}_K^h\|_F}{\sqrt{d_k}}}_{\text{effective inverse temperature } \beta_h} \cdot \mathbf{X}\,\widetilde{\mathbf{W}}^h\,\mathbf{X}^\top, \qquad \widetilde{\mathbf{W}}^h := \frac{\mathbf{W}_Q^h(\mathbf{W}_K^h)^\top}{\|\mathbf{W}_Q^h\|_F\,\|\mathbf{W}_K^h\|_F}, \tag{A49}$$

so at fixed normalized shape $\widetilde{\mathbf{W}}^h$, larger $\|\mathbf{W}_Q^h\|_F\|\mathbf{W}_K^h\|_F$ gives a larger softmax inverse temperature and a sharper attention pattern. The V/O norms control the head's output magnitude*:*

$$\|\mathbf{O}^h\mathbf{W}_O^h\|_F \;\le\; \|\mathbf{A}^h\|_2\,\|\mathbf{X}\|_F\,\|\mathbf{W}_V^h\|_2\,\|\mathbf{W}_O^h\|_2, \tag{A50}$$

so heads with larger $\|\mathbf{W}_V^h\|_2\|\mathbf{W}_O^h\|_2$ tend to contribute more to the layer output.

(b) ***Backward.*** *Let $\mathbf{G} = \partial\mathcal{L}/\partial\mathbf{Z}$. Then*

$$\left\|\frac{\partial\mathcal{L}}{\partial\mathbf{W}_O^h}\right\|_F \le \|\mathbf{A}^h\|_2\,\|\mathbf{X}\|_2\,\|\mathbf{W}_V^h\|_2\,\|\mathbf{G}\|_F, \tag{A51}$$

$$\left\|\frac{\partial\mathcal{L}}{\partial\mathbf{W}_V^h}\right\|_F \le \|\mathbf{A}^h\|_2\,\|\mathbf{X}\|_2\,\|\mathbf{W}_O^h\|_2\,\|\mathbf{G}\|_F, \tag{A52}$$

$$\left\|\frac{\partial\mathcal{L}}{\partial\mathbf{W}_Q^h}\right\|_F \le C_X\,\|\mathbf{W}_K^h\|_2\,\|\mathbf{W}_V^h\|_2\,\|\mathbf{W}_O^h\|_2\,\|\mathbf{G}\|_F, \tag{A53}$$

$$\left\|\frac{\partial\mathcal{L}}{\partial\mathbf{W}_K^h}\right\|_F \le C_X\,\|\mathbf{W}_Q^h\|_2\,\|\mathbf{W}_V^h\|_2\,\|\mathbf{W}_O^h\|_2\,\|\mathbf{G}\|_F, \tag{A54}$$

where $C_X := 2\|\mathbf{X}\|_2^3/\sqrt{d_k}$.

Proof. The logit factorization follows by substituting the definition of $\widetilde{\mathbf{W}}^h$. The sharpness claim is the standard temperature-scaling property of softmax: for non-constant $\boldsymbol{\ell}$, the entropy of $\mathrm{softmax}(\beta\boldsymbol{\ell})$ decreases with $\beta > 0$. The output bound (A50) follows from $\mathbf{O}^h\mathbf{W}_O^h = \mathbf{A}^h\mathbf{X}\mathbf{W}_V^h\mathbf{W}_O^h$ and submultiplicativity ($\|MN\|_F \le \|M\|_2\|N\|_F$ applied left-to-right, then $\|MN\|_F \le \|M\|_F\|N\|_2$ on $\mathbf{X}\mathbf{W}_V^h$).

For the backward bounds, the chain rule gives $\partial\mathcal{L}/\partial\mathbf{W}_O^h = (\mathbf{X}\mathbf{W}_V^h)^\top(\mathbf{A}^h)^\top\mathbf{G}$ and $\partial\mathcal{L}/\partial\mathbf{W}_V^h = \mathbf{X}^\top(\mathbf{A}^h)^\top\mathbf{G}(\mathbf{W}_O^h)^\top$, which imply (A51) and (A52). For Q and K, first note that $\partial\mathcal{L}/\partial\mathbf{A}^h = \mathbf{G}(\mathbf{W}_O^h)^\top(\mathbf{W}_V^h)^\top\mathbf{X}^\top$, so $\|\partial\mathcal{L}/\partial\mathbf{A}^h\|_F \le \|\mathbf{G}\|_F\|\mathbf{W}_O^h\|_2\|\mathbf{W}_V^h\|_2\|\mathbf{X}\|_2$. The row-wise softmax Jacobian has spectral norm at most 2, giving $\|\partial\mathcal{L}/\partial\mathbf{S}^h\|_F \le 2\|\partial\mathcal{L}/\partial\mathbf{A}^h\|_F$. Combining this with $\partial\mathcal{L}/\partial\mathbf{W}_Q^h = \mathbf{X}^\top(\partial\mathcal{L}/\partial\mathbf{S}^h)\mathbf{X}\mathbf{W}_K^h/\sqrt{d_k}$ yields (A53); (A54) follows symmetrically. □

Remark I.2 (Implications for optimizer design). Proposition I.1 shows that the per-head norms inherited from prior training modulate both attention behavior and gradient scale. Since these norms vary substantially across heads in trained models (Fig. 4-(b)), different heads naturally receive updates of different magnitudes. A whole-matrix spectral optimizer applies one Newton-Schulz

orthogonalization to a concatenated projection matrix, which tends to equalize update scale across heads and mix head-specific directions. Per-head spectral filtering avoids this by filtering each head independently.

J Detailed Training Setups for VLA and RLVR Experiments

In this section, we report the hyperparameter configurations for the VLA and RLVR experiments in Sec. 6. Within each setting, the three optimizer configurations (AdamW, Muon, and Pion) share identical training setups, hardware, and evaluation protocols; the only altered variable is the optimizer assignment. For Pion, we use Suppression-dominant high-pass NS schedules with $k_{\mathrm{s}} \geq 3$ (equivalently, $k_{\mathrm{p}} \leq 2$ under the fixed total $k = 5$). **Table A3** lists the VLA training hyperparameters for VLA-Adapter (Wang et al., 2026b) and VLANeXt (Wu et al., 2026) on LIBERO (Liu et al., 2023), with VLANeXt additionally evaluated on the perturbed LIBERO-Plus split (Fei et al., 2025); the *Object* suite converges faster and is allocated fewer training steps. **Table A4** summarizes the RLVR hyperparameters, reused across both RL algorithms (GRPO/GMPO) and both model scales (Qwen3-1.7B/4B); only the prompt/response length, train batch, rollout group size, and total steps differ between MATH and GSM8K. **Table A5** summarizes the real-robot setup, where $\pi_{0.5}$ (Intelligence et al., 2025) is finetuned under the DROID hardware platform (Khazatsky et al., 2025; Wang et al., 2026a) and evaluated on three grasp-and-place tasks.

Table A3: Training hyperparameters for the VLA experiments on the LIBERO benchmark. The three optimizer configurations (i)–(iii) are applied identically to both models, and share all other hyperparameters listed in this table.

Item	**VLA-Adapter**	**VLANeXt**
Backbone	Prismatic-Qwen2.5-0.5B	Qwen3-VL-2B-Instruct
Train dataset	LIBERO	LIBERO
Test dataset	LIBERO	LIBERO and LIBERO-Plus
Global batch size	64	256
Learning rate	1×10^{-4}	1×10^{-4}
Weight decay	1×10^{-2}	1×10^{-2}
Max steps (*Object*)	1,500	4,000
Max steps (*Spatial* / *Goal* / *Long*)	15,000	10,000
Compute	$8 \times$ NVIDIA RTX A6000	$8 \times$ NVIDIA RTX A6000

Optimizer configurations † (applied to action (A), vision (V), and language (L) modules):
(i) **AdamW** on all modules.
(ii) **Muon** on the 2D matrices of A, V, and L; AdamW on all remaining parameters.
(iii) **Pion** on the 2D matrices of A, **Muon** on those of V and L; AdamW elsewhere.
† The 2D weight matrices exclude token embeddings and the output (LM-head) layer.

Table A4: Training and rollout hyperparameters for the RLVR experiments. The three optimizer configurations (i)–(iii) are reused across the two RL algorithms (GRPO and GMPO) and the two model scales (Qwen3-1.7B and Qwen3-4B), and share all other hyperparameters listed in this table within each benchmark.

Item	**MATH**	**GSM8K**
Base model	Qwen3-1.7B and Qwen3-4B	Qwen3-1.7B and Qwen3-4B
Algorithm	GRPO and GMPO	GRPO and GMPO
Train dataset	MATH levels 3–5	GSM8K (train split)
Test dataset	MATH500	GSM8K (test split)
Max prompt / response length	1,024 / 3,000	512 / 1,024
Train batch (prompts)	128	1,024
Rollout group size n	8	5
Rollout temperature / Top-p	1.0 / 1.0	1.0 / 1.0
Learning rate	1×10^{-6}	1×10^{-6}
Total training steps	80	40
Compute	$2 \times$ NVIDIA H100	$2 \times$ NVIDIA H100

Optimizer configurations †:
(i) **AdamW** on all parameters.
(ii) **Muon** on all 2D weight matrices; AdamW elsewhere.
(iii) **Pion** (per-head mode) on all 2D weight matrices; AdamW elsewhere.
† The 2D weight matrices exclude token embeddings and the output (LM-head) layer.

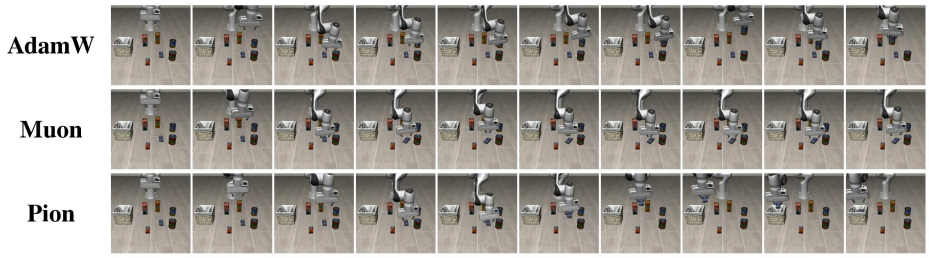
Table A5: Hardware, training, and rollout configuration for the real-robot evaluation. The three optimizer configurations (i)–(iii) share all other settings listed in this table; the only altered variable is the optimizer assignment.

Item	**Real-robot ($\pi_{0.5}$ on three grasp-and-place tasks)**
Backbone VLA	$\pi_{0.5}$
Robot platform	Franka Research 3 (7-DoF)
Hardware setup	DROID setup
Cameras (input)	one third-view camera + one wrist-mounted camera
Tasks	*Cucumber* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Bowl*
Demonstrations	200 teleoperated trajectories
Total training steps	20,000
Trials per (optimizer, task)	30 (randomized initial pose), $\leq$ 300 control steps each
Evaluation metric	trial-level success rate (#successes / 30)

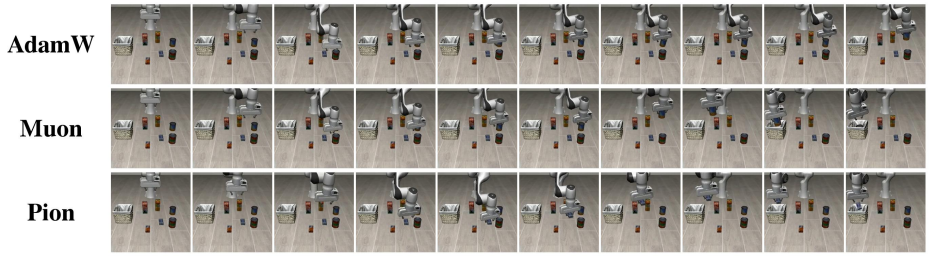
Optimizer configurations † (applied to action (A), vision (V), and language (L) modules):
(i) **AdamW** on all parameters.
(ii) **Muon** on the 2D matrices of A, V, and L; AdamW on all remaining parameters.
(iii) **Pion** on the 2D matrices of A, **Muon** on those of V and L; AdamW elsewhere.
† The 2D weight matrices exclude token embeddings and the output (LM-head) layer.

K QUALITATIVE ROLLOUTS

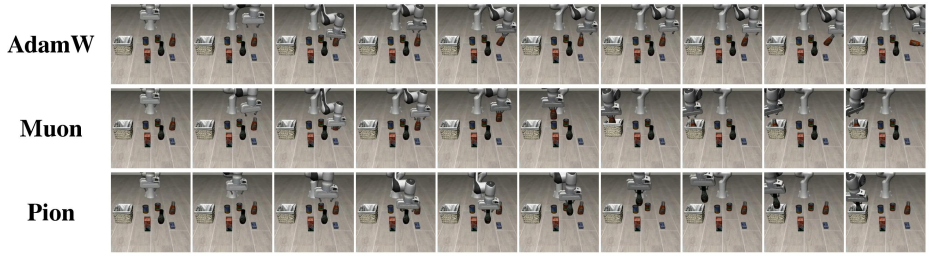
K.1 LIBERO OBJECT



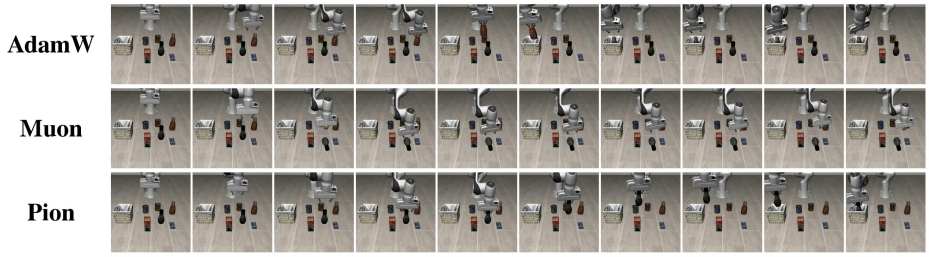
LIBERO-Object: Lift the spread often used for bagels and add it to the container meant for holding items during shopping



LIBERO-Object: Pick up the spread used on bagels and place it in the woven container for carrying items

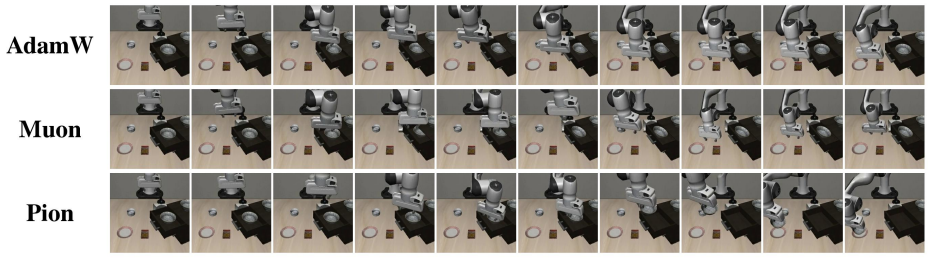


LIBERO-Object: Grasp the container for enhancing the flavor of greens and relocate it to the woven holder for carrying items

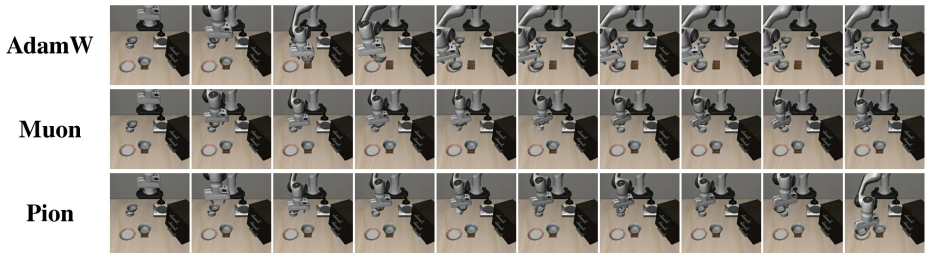


LIBERO-Object: Pick up the liquid condiment for greens and place it in the woven carrier

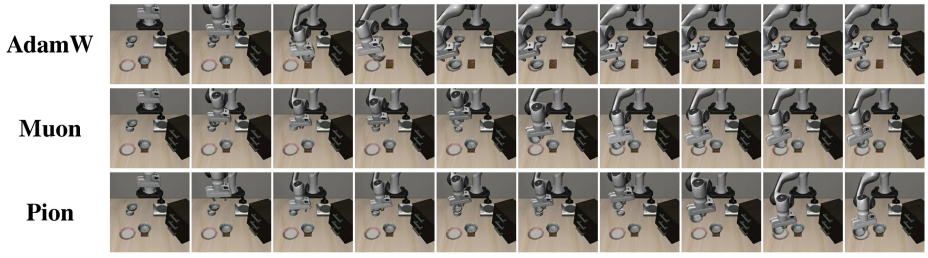
K.2 LIBERO SPATIAL



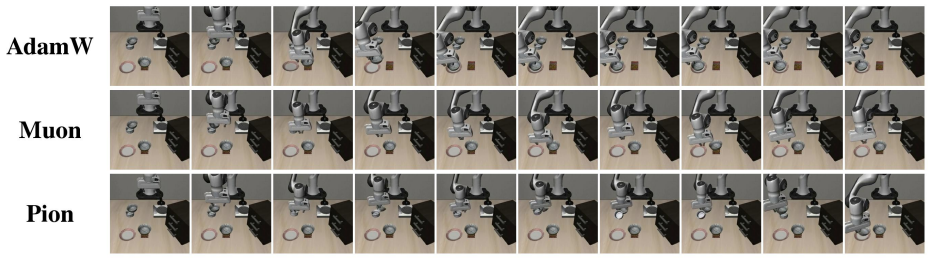
LIBERO-Spatial: Pick up the darkhued curved vessel located in the uppermost sliding storage compartment of the timber storage unit and place it on the flat ceramic dish



LIBERO-Spatial: Grasp the dark bowl atop the small dish and set it on the plate

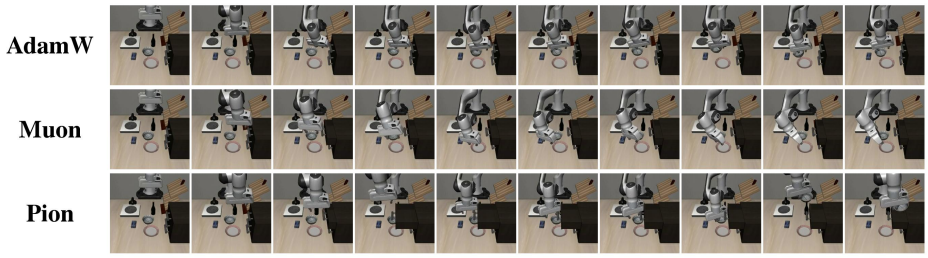


LIBERO-Spatial: Pick up the darkhued circular container on the small glazed ceramic dish and place it on the flat dishware for serving food

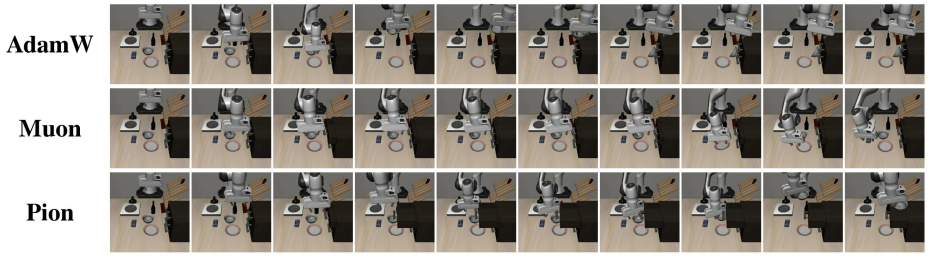


LIBERO-Spatial: Lift the darkcolored round vessel located atop the small glazed ceramic dish and set it down on the flat dish used for serving meals

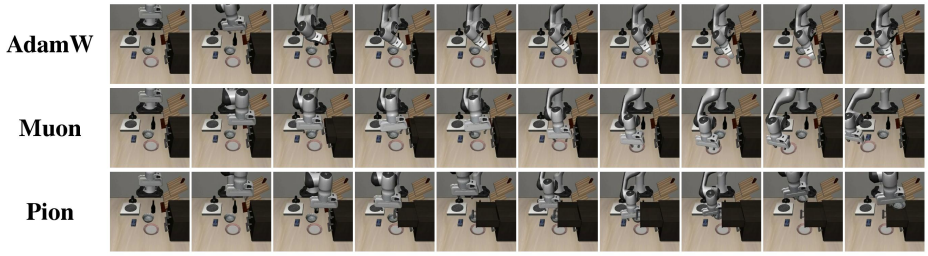
K.3 LIBERO GOAL



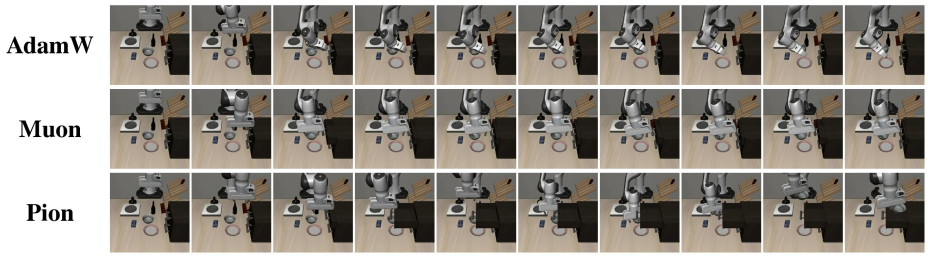
LIBERO-Goal: Pull open the uppermost sliding storage compartment and place the rounded dish inside



LIBERO-Goal: After washing the bowl remember to open the top drawer and put it inside immediately

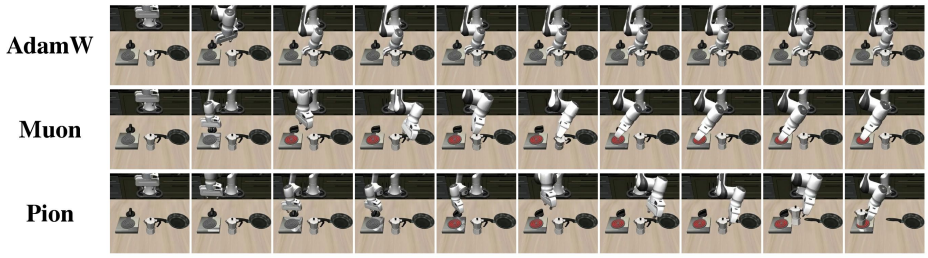


LIBERO-Goal: Open the highest sliding storage compartment and place the round dish used for eating or serving food within it

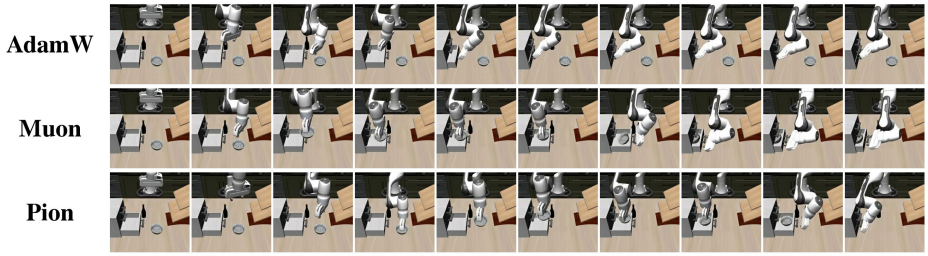


LIBERO-Goal: Open the uppermost storage compartment and place the round dish used for eating or serving food within

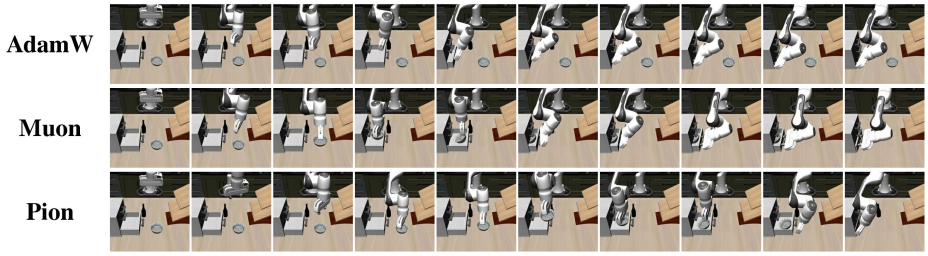
K.4 LIBERO LONG



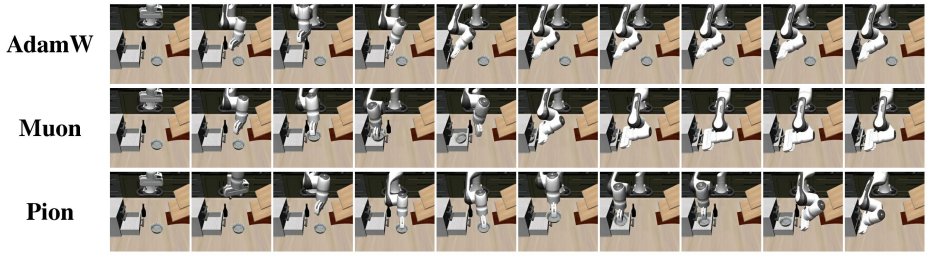
LIBERO-Long: Activate the kitchen appliance used for cooking and place the Italian coffee maker on its surface



LIBERO-Long: Place the dark bowl into the lower drawer of the cupboard and shut it



LIBERO-Long: Make sure the black bowl is inside the bottom drawer of the cabinet and that the drawer is closed



LIBERO-Long: On this lovely afternoon could you please place the black bowl in the bottom drawer of the cabinet and make sure to close it afterwards

L Real-Robot Evaluation

L.1 Quantitative results

We further validate Pion on a physical robot by finetuning $\pi_{0.5}$ (Intelligence et al., 2025) under the DROID setup (Khazatsky et al., 2025) on three grasp-and-place tasks (*Cucumber* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Bowl*). All three optimizers are trained for the same $20{,}000$ steps under the same training dataset. **Table A6** reports the trial-level success rate over 30 randomized trials per task. Pion sharply outperforms both baselines on every task, lifting the average success rate from 31.1% (AdamW) and 38.9% (Muon) to 85.6%. Crucially, this substantial performance gain over AdamW and Muon is achieved under a low-budget VLA training regime consisting of only 20,000 training steps, which is much fewer than those typically used in standard AdamW-based VLA training. This step-efficiency advantage mirrors the margin observed in simulation (Fig. 5-(b)), confirming that Pion's training-efficiency gain carries over from simulation to real hardware. We attribute this to Pion's high-pass spectral filtering on the action module, whose benefit is further amplified under the tighter precision tolerances of physical manipulation. Qualitative rollouts are visualized in **Appendix L.2** (**Table A7**).

Table A6: AdamW, Muon, and Pion on real-robot grasp-and-place tasks using $\pi_{0.5}$ (Intelligence et al., 2025) backbone under the DROID setup (Khazatsky et al., 2025). Each entry reports the success rate (%) over 30 randomized initial configurations. The best score in each column is in **bold**.

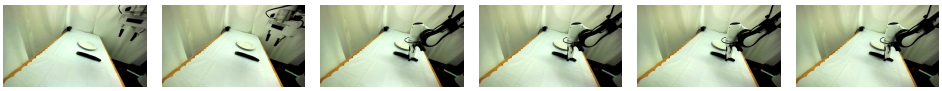
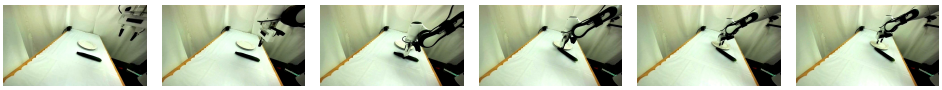
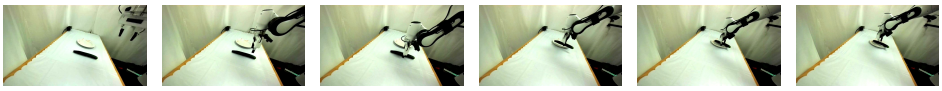
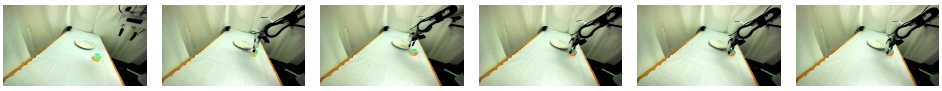
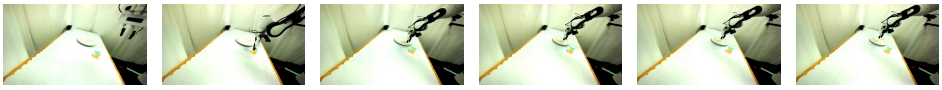
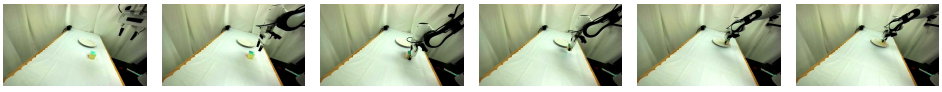
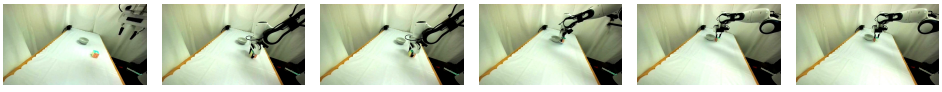
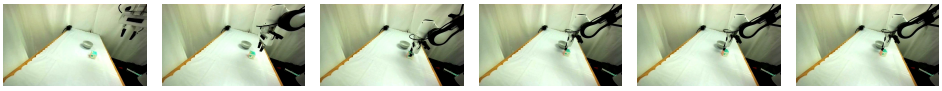
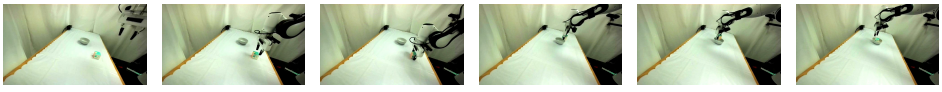
Optimizer	Cucumber $\rightarrow$ Plate	Cube $\rightarrow$ Plate	Cube $\rightarrow$ Bowl	Average
AdamW	40.0	33.3	20.0	31.1
Muon	56.7	33.3	26.7	38.9
Pion (Ours)	**93.3**	**83.3**	**80.0**	**85.6**

L.2 Visualization of real-robot rollouts

Table A7 compares a single rollout of $\pi_{0.5}$ trained with AdamW, Muon, and Pion on each of the three tasks (*Cucumber* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Plate*, *Cube* $\rightarrow$ *Bowl*, top to bottom). Each row shows 6 frames uniformly sampled along that rollout, from approach to placement.

Cucumber $\rightarrow$ *Plate* (Table A7, top): AdamW repeatedly attempts to grasp the cucumber but never lifts it off the table (frame 5); Muon grasps it but opens the gripper prematurely, dropping the cucumber mid-transport (frame 3); Pion grasps and places cleanly. *Cube* $\rightarrow$ *Plate* (Table A7, middle): both AdamW and Muon open the gripper prematurely before reaching the plate (frame 3 in either row), so the cube is released mid-air rather than on the plate, while Pion grasps and places the cube accurately. *Cube* $\rightarrow$ *Bowl* (Table A7, bottom): on the hardest task, AdamW lifts the cube but not high enough to clear the rim of the bowl (frame 3), and Muon misaligns the gripper with the cube and fails to establish a stable grasp (frame 3); Pion deposits the cube inside the bowl, corroborating the quantitative gains in Table A6.

Table A7: Real-robot rollouts of $\pi_{0.5}$ trained with AdamW, Muon, and Pion on the three grasp-and-place tasks. Each row shows 6 frames uniformly sampled along a single rollout, from approach to placement. The natural-language task prompt is shown above each task block (in gray).

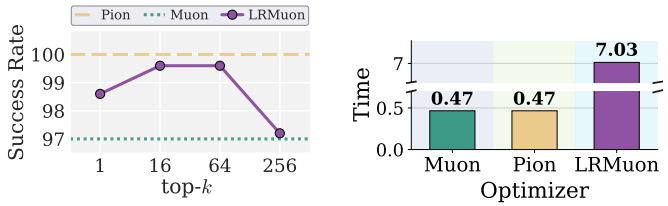
Optimizer	**Frame index**					
	0	**1**	**2**	**3**	**4**	**5**
Prompt: *"Pick up the cucumber and place it on the plate."*						
AdamW						
Muon						
Pion (Ours)						
Prompt: *"Pick up the cube and place it on the plate."*						
AdamW						
Muon						
Pion (Ours)						
Prompt: *"Pick up the cube and place it in the bowl."*						
AdamW						
Muon						
Pion (Ours)						

M ADDITIONAL VLA EXPERIMENTS

This appendix expands the ablation summary in Sec. 6.2 with the full setups, figures/tables, and per-row analysis of three studies on VLA-Adapter (Wang et al., 2026b): (i) Pion vs. LRMuon for action-module training, (ii) per-head vs. default Pion on the action head, and (iii) modality-wise optimizer assignment across the Vision, Language, and Action branches.

M.1 PION VS. LRMUON FOR VLA TRAINING

Pion outperforms LRMuon for VLA training with near-Muon cost. Fig. A1 compares Pion with LRMuon (Low-rank Muon) for training VLA-Adapter on LIBERO Object. LRMuon computes an exact SVD of the momentum at each step, retains the top-k singular subspace, and applies the corresponding top-k polar factor $\mathbf{U}_k\mathbf{V}_k^\top$ (He et al., 2025a), as used in Fig. 1. We can observe from Fig. A1-(a) that LRMuon improves over Muon across all top-k ranks $k \in \{1, 16, 64, 256\}$, confirming the benefit of low-rank spectral filtering, but underperforms Pion at every k. This gap arises for two reasons. First, LRMuon uses a *fixed* top-k rank that cannot adapt to the per-step and per-layer rank of the momentum, whereas Pion applies a *soft* spectral filter via high-pass NS. Second, Fig. A1-(b) shows that the per-step exact SVD computation significantly increases total training time, whereas Pion matches Muon's cost almost exactly.



(a) Success rate vs. top-k rank (b) Total training time (hrs)

Figure A1: Muon, Pion and LRMuon for VLA-Adapter on LIBERO Object for 1,500 steps. (a) Test success rate as the top-k rank of LRMuon sweeps $k \in \{1, 16, 64, 256\}$; Pion and Muon are shown as horizontal references. (b) Total training time (hours).

M.2 PER-HEAD VS. DEFAULT PION ON VLA

Sec. 5 introduces two application modes of high-pass NS, the default mode and the per-head mode. Table A8 reports both modes on VLA-Adapter across the four LIBERO task suites. The two modes perform on par, with the default mode marginally ahead on three of four suites (Object 100.0 vs. 99.6, Spatial 99.4 vs. 98.8, Long 92.4 vs. 91.6; only Goal slightly favors per-head, 97.4 vs. 97.2), yielding a $+0.4$ gap on the four-suite average. This is consistent with the intuition of Sec. 5: unlike the LLM backbone in RLVR, the VLA action head is trained from scratch and carries no per-head heterogeneity for the per-head reshape to preserve, so the default whole-matrix mode already suffices. We therefore use default Pion on the action head throughout Sec. 6.2.

Table A8: AdamW, Muon, and Pion (default vs. per-head) for VLA-Adapter on LIBERO. Test success rates on LIBERO Object, Spatial, Goal, and Long at the same training budget (1,500 steps for Object and 15,000 steps for others). The best results in each column are in **bold**.

Optimizer	Object	Spatial	Goal	Long	Average
AdamW	32.2	97.0	89.2	69.6	72.00
Muon	97.0	99.0	95.8	88.0	94.95
Pion (per-head)	99.6	98.8	**97.4**	91.6	96.85
Pion (default)	**100.0**	**99.4**	97.2	**92.4**	**97.25**

M.3 MODALITY-WISE OPTIMIZER ASSIGNMENT ON VLA

The VLA configuration used throughout Sec. 6.2, namely Muon on V/L and Pion on the action head, is one of several plausible assignments. To check whether it is the right one, we sweep the optimizer

of each branch independently on VLA-Adapter/LIBERO Object at $1{,}500$ steps, indexing the resulting nine settings as **S1**–**S9** in Table A9. **S1** is the all-AdamW reference; **S2–S3**, **S4–S5**, **S6–S7** perturb only the Action, Language, and Vision modules away from **S1** respectively; and **S8–S9** contrast the all-Muon configuration with our final "Muon on V/L + Pion on action" design. Three observations follow:

Table A9: Modality-wise optimizer ablation for VLA-Adapter on LIBERO Object. Test success rates at $1{,}500$ training steps. AdamW, Muon, and Pion are ablated across Vision, Language, and Action modules. The best result is in **bold**.

Setting	Optimizer			Success Rate (%)
	Vision	Language	Action	
S1	AdamW	AdamW	AdamW	32.3
S2	AdamW	AdamW	Muon	28.0
S3	AdamW	AdamW	Pion	73.6
S4	AdamW	Muon	AdamW	94.6
S5	AdamW	Pion	AdamW	73.8
S6	Muon	AdamW	AdamW	96.8
S7	Pion	AdamW	AdamW	17.8
S8	Muon	Muon	Muon	97.0
S9	Muon	Muon	Pion	**100.0**

(i) *Action head wants Pion, not Muon.* With V/L fixed at AdamW, switching the action head from AdamW (**S1**, 32.3) to Muon (**S2**) *drops* accuracy to 28.0, while switching it to Pion (**S3**) *lifts* it to 73.6. This confirms the spectral diagnosis of Sec. 4: the low-erank action gradient is mismatched with Muon's uniform whitening, but well-suited to Pion's high-pass.

(ii) *Vision and Language want Muon, not Pion.* Symmetrically, with the other two branches fixed at AdamW, switching Language to Muon (**S4**) improves accuracy from **S1** (32.3) to 94.6, while switching it to Pion (**S5**) only reaches 73.8; switching Vision to Muon (**S6**) improves accuracy to 96.8, while switching it to Pion (**S7**) collapses to 17.8. The high-rank V/L modules thus genuinely benefit from Muon's uniform spectral updates, and applying a high-pass there discards informative tail components.

(iii) *The chosen assignment is optimal.* Combining the two findings, "Muon on V/L + Pion on action" (**S9**) reaches 100.0% success, strictly above all-Muon (**S8**, 97.0%) and any single-module configuration in **S2–S7**. **S9** is therefore not an arbitrary engineering choice but the assignment that respects the spectral structure of each modality.

N LEARNING-RATE SWEEP FOR RLVR

To test whether Muon's collapse in RLVR (Fig. 6) is an artifact of the learning rate, we sweep the learning rate $\eta \in \{2\times10^{-7}, 5\times10^{-7}, 1\times10^{-6}, 2\times10^{-6}, 5\times10^{-6}\}$ for Qwen3-1.7B trained with GRPO on MATH, keeping all other settings of Fig. 6-(a); the main experiments use $\eta = 1\times10^{-6}$ (Table A4). **Table A10** reports the best MATH500 accuracy for each learning rate. Muon collapses at every learning rate, including those $5\times$ smaller than the default, whereas Pion trains stably across the whole range, reaching 73.8–79.0%. Muon's failure is therefore not a tuning artifact, consistent with the spectral explanation of Limitation 2 (Sec. 4).

Table A10: Best MATH500 accuracy (%) of Muon and Pion under different learning rates for Qwen3-1.7B trained with GRPO on MATH (setup of Fig. 6-(a)).

Learning rate	**Muon**	**Pion**
2×10^{-7}	0.0	75.8
5×10^{-7}	0.0	79.0
1×10^{-6}	0.0	78.8
2×10^{-6}	0.0	76.8
5×10^{-6}	0.0	73.8

O UPDATE NORM OF MUON AND PION

Since the Suppression stage drives tail singular values toward 0, a natural question is whether Pion simply takes much smaller steps than Muon. **Table A11** reports the normalized Frobenius norm of the update produced by Muon and Pion at layers 5, 10, 15, 20, and 25 of Qwen3-1.7B trained with GRPO on MATH (setup of Fig. 6-(a)). Pion's update norm is 72–80% of Muon's across layers, so the two remain of the same order of magnitude: the suppressed tail accounts for only a modest fraction of the update norm, while the retained leading directions carry most of it. The contrast between Muon and Pion in RLVR is therefore not explained by a large difference in step size; this is further supported by the learning-rate sweep in Appendix N, where Muon collapses even at learning rates $5\times$ smaller than the default.

Table A11: Normalized Frobenius norm of the update at different layers of Qwen3-1.7B trained with GRPO on MATH under Muon and Pion (setup of Fig. 6-(a)), together with the Pion-to-Muon ratio.

Layer	**Muon**	**Pion**	**Pion / Muon**
5	0.149	0.107	0.721
10	0.148	0.108	0.727
15	0.145	0.113	0.782
20	0.139	0.108	0.777
25	0.137	0.109	0.798

P LOW-PASS MUON (LPMUON): COEFFICIENT DESIGN VIA CONSTRAINED POLYNOMIAL FITTING

This appendix details the coefficient design of **Low-pass Muon** (LPMuon), the reverse-ablation baseline of Sec. 6.3 (Fig. 8). Unlike Pion, whose Promotion and Suppression polynomials admit closed-form solutions from analytic constraints at $\sigma=0$ and $\sigma=1$ (Sec. 5), the LPMuon target profile is a sharp band indicator whose quality depends on the *whole composition* across σ, and the $t=5$ steps exchange degrees of freedom (e.g., scaling by p_1 can be partially absorbed into p_2). We therefore treat all 15 coefficients as free variables and fit them numerically via a multi-start L-BFGS-B procedure.

Target filter and matrix-level update. LPMuon composes $t = 5$ odd quintic polynomials $p_k(\sigma) = a_{1,k}\sigma + a_{3,k}\sigma^3 + a_{5,k}\sigma^5$:

$$\tilde{f}_{\boldsymbol{\theta}}(\sigma) := (p_5 \circ p_4 \circ p_3 \circ p_2 \circ p_1)(\sigma), \quad \boldsymbol{\theta} = \{(a_{1,k}, a_{3,k}, a_{5,k})\}_{k=1}^{5} \in \mathbb{R}^{15}, \tag{A55}$$

to approximate an odd extension of the band indicator on $\sigma \in [-1, 1]$. The actual normalized singular values of the pre-normalized momentum are nonnegative and lie in $[0, 1]$; the negative half-axis is included only to define and visualize the odd scalar extension:

$$\tilde{f}_{\boldsymbol{\theta}}(\sigma) \approx \operatorname{sign}(\sigma) \cdot \mathbb{1}\big[|\sigma| \le \tau\big], \quad \tau \in (0, 1) \text{ is the cutoff.} \tag{A56}$$

Since each p_k is odd, $\tilde{f}_{\boldsymbol{\theta}}$ is automatically antisymmetric and we only need to fit on $\sigma \ge 0$. By the SVD factorization (A33), applying p_k at the matrix level on $\mathbf{M}_t \in \mathbb{R}^{m\times n}$,

$$\mathbf{M}_t \leftarrow a_{1,k}\,\mathbf{M}_t + a_{3,k}\,(\mathbf{M}_t\mathbf{M}_t^\top)\mathbf{M}_t + a_{5,k}\,(\mathbf{M}_t\mathbf{M}_t^\top)^2\mathbf{M}_t, \tag{A57}$$

is equivalent to applying $\tilde{f}_{\boldsymbol{\theta}}$ entry-wise to every singular value of $\mathbf{M}_t$, so LPMuon preserves Muon's per-step 5-matmul cost and requires no explicit SVD.

Discretized fitting objective. Given a cutoff τ, we discretize the positive half-axis into a pass band $\mathcal{S}_{\mathrm{p}}^{+} \subset [0.01,\ \tau-\Delta]$ and a stop band $\mathcal{S}_{\mathrm{s}}^{+} \subset [\tau+\Delta,\ 1]$ separated by a transition half-width $\Delta = 0.03$ (up to 250 samples per band per side, reduced to 50 when τ is close to 1); these are mirrored to the negative half-axis for notational symmetry in the odd-extension loss, forming $\mathcal{S}_{\mathrm{p}}, \mathcal{S}_{\mathrm{s}}$. Intermediate iterates are clipped to $[-10^3, 10^3]$ to avoid early-iteration overflow. The fitting loss combines a pass-band, stop-band, overshoot, and non-negativity term:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta};\tau) = \lambda_{\mathrm{p}} \underbrace{\frac{1}{|\mathcal{S}_{\mathrm{p}}|}\sum_{\sigma\in\mathcal{S}_{\mathrm{p}}}\big(\tilde{f}_{\boldsymbol{\theta}}(\sigma) - \operatorname{sign}(\sigma)\big)^2}_{\mathcal{L}_{\mathrm{pass}}} + \lambda_{\mathrm{s}} \underbrace{\frac{1}{|\mathcal{S}_{\mathrm{s}}|}\sum_{\sigma\in\mathcal{S}_{\mathrm{s}}}\tilde{f}_{\boldsymbol{\theta}}(\sigma)^2}_{\mathcal{L}_{\mathrm{stop}}} \\ + \lambda_{\mathrm{o}} \underbrace{\frac{1}{|\mathcal{S}_{\mathrm{p}}\cup\mathcal{S}_{\mathrm{s}}|}\sum_{\sigma}\big(\max(|\tilde{f}_{\boldsymbol{\theta}}(\sigma)| - 1.02,\ 0)\big)^2}_{\mathcal{L}_{\mathrm{over}}} + \lambda_{\mathrm{nn}} \underbrace{\sum_{\mathcal{S}\in\{\mathcal{S}_{\mathrm{p}}^{+},\mathcal{S}_{\mathrm{s}}^{+}\}}\frac{1}{|\mathcal{S}|}\sum_{\sigma\in\mathcal{S}}\big(\max(-\tilde{f}_{\boldsymbol{\theta}}(\sigma),\ 0)\big)^2}_{\mathcal{L}_{\mathrm{nn}}},\end{aligned} \tag{A58}$$

with weights $(\lambda_{\mathrm{p}}, \lambda_{\mathrm{s}}, \lambda_{\mathrm{o}}, \lambda_{\mathrm{nn}}) = (3,\ 8,\ 30,\ 30)$. Here $\mathcal{L}_{\mathrm{pass}}$ anchors the pass band at ± 1; $\mathcal{L}_{\mathrm{stop}}$ drives the stop band to 0; $\mathcal{L}_{\mathrm{over}}$ keeps intermediate iterates bounded so the 5-step composition does not blow up; $\mathcal{L}_{\mathrm{nn}}$ enforces non-negativity on $\sigma > 0$ (without it the fit admits sign-flipping solutions that would invert the gradient direction for a subset of singular components). The stop-band and overshoot terms are weighted more heavily because residual energy or overshoot compounds multiplicatively across the 5 compositions.

Warm-start initialization, multi-start solver, and aggregation. To escape the many spurious local minima of quintic compositions, we use a structured warm start with random restarts. The first polynomial is initialized as identity ($p_1^{(0)} = (1, 0, 0)$) and the remaining four are initialized to Pion's Promotion coefficients (7), $p_{2:5}^{(0)} = (1.875, -1.25, 0.375)$. Trial $m = 1$ uses $\boldsymbol{\theta}^{(0)}$ directly; trials $m = 2, \ldots, M$ ($M = 8$) use $\boldsymbol{\theta}^{(0)} + \boldsymbol{\varepsilon}^{(m)}$ with $\boldsymbol{\varepsilon}^{(m)} \sim \mathcal{N}(\mathbf{0},\ 0.25^2\mathbf{I}_{15})$. Each trial is solved by `scipy.optimize.minimize` with L-BFGS-B (maximum 2,000 iterations, $f_{\mathrm{tol}} = 10^{-12}$, $g_{\mathrm{tol}} = 10^{-9}$, finite-difference gradients); divergent restarts are discarded. The final solution is

$\hat{\boldsymbol{\theta}}(\tau) = \boldsymbol{\theta}_{\infty}^{(m^\star)}$ with $m^\star = \arg\min_m \mathcal{L}(\boldsymbol{\theta}_{\infty}^{(m)}; \tau)$. We sweep $\tau \in \{0.1, 0.2, \ldots, 0.9\}$, and use $\tau = 0.5$ inside the RLVR optimization loop in Fig. 8. The full procedure is summarized in Alg. 4.

Algorithm 4 LPMuon Coefficient Fitting (L-BFGS-B)

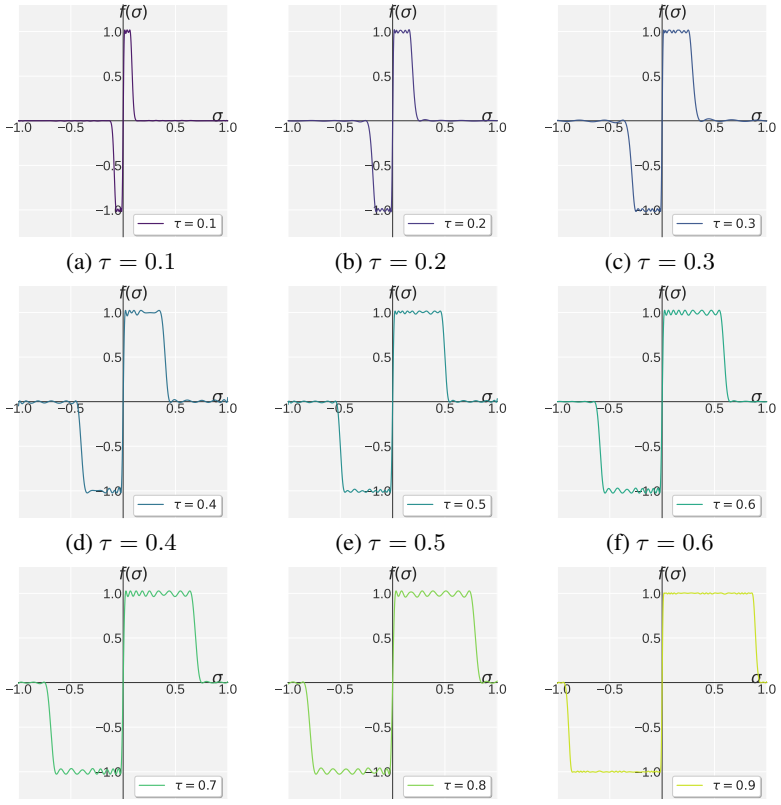
Require: Cutoff τ, transition half-width Δ, steps $t = 5$, restarts M, weights $(\lambda_{\mathrm{p}}, \lambda_{\mathrm{s}}, \lambda_{\mathrm{o}}, \lambda_{\mathrm{nn}})$

1: Build discretizations $\mathcal{S}_{\mathrm{p}} \subset [-(\tau-\Delta),\ -0.01] \cup [0.01,\ \tau-\Delta]$, $\mathcal{S}_{\mathrm{s}} \subset [-1,\ -(\tau+\Delta)] \cup [\tau+\Delta,\ 1]$
2: Warm start $\boldsymbol{\theta}^{(0)}$: $p_1 = (1, 0, 0)$ and $p_{2:t} = (1.875, -1.25, 0.375)$ {Pion Promotion}
3: **for** $m = 1, \ldots, M$ **do**
4: $\quad \boldsymbol{\theta}_{\mathrm{init}}^{(m)} \leftarrow \boldsymbol{\theta}^{(0)}$ if $m = 1$, else $\boldsymbol{\theta}^{(0)} + \boldsymbol{\varepsilon}^{(m)}$ with $\boldsymbol{\varepsilon}^{(m)} \sim \mathcal{N}(\mathbf{0}, 0.25^2\, \mathbf{I}_{15})$
5: $\quad \boldsymbol{\theta}_{\infty}^{(m)} \leftarrow \mathrm{LBFGSB}\big(\mathcal{L}(\,\cdot\,; \tau),\ \boldsymbol{\theta}_{\mathrm{init}}^{(m)}\big)$; $\quad \mathcal{L}^{(m)} \leftarrow \mathcal{L}(\boldsymbol{\theta}_{\infty}^{(m)}; \tau)$
6: **end for**
7: **return** $\hat{\boldsymbol{\theta}}(\tau) = \boldsymbol{\theta}_{\infty}^{(m^\star)}$ with $m^\star = \arg\min_m \mathcal{L}^{(m)}$, and the corresponding scalar filter $\tilde{f}_{\hat{\boldsymbol{\theta}}(\tau)}$ (A55)

Resulting filters and reverse-ablation evidence. **Fig. A2** visualizes $\tilde{f}_{\hat{\boldsymbol{\theta}}(\tau)}$ across the full sweep $\tau \in \{0.1, 0.2, \ldots, 0.9\}$: as τ grows, the transition shifts rightward while the pass band ($|\sigma| \le \tau$) stays anchored at ± 1 and the stop band ($|\sigma| > \tau$) is driven to 0, confirming that Alg. 4 consistently recovers the desired low-pass profile. Numerical coefficients are listed in **Table A12** and can be plugged directly into (A57). At the matrix level, this gives the LPMuon optimizer used in Fig. 8-(b); its flat accuracy curve (LPMuon fails to train at all) provides the reverse-ablation evidence of Sec. 6.3: retaining the small singular values while discarding the large ones destroys the learning signal, isolating that Pion's gains arise from *high-pass* filtering rather than the iteration form, per-head reshape, or generic spectral transformation.

Table A12: Fitted coefficients $\hat{\boldsymbol{\theta}}(\tau) = \{(a_{1,k}, a_{3,k}, a_{5,k})\}_{k=1}^{5}$ of the 5-step odd-quintic composition (A55) solved by Alg. 4 for the cutoff sweep $\tau \in \{0.1, \ldots, 0.9\}$. Each row corresponds to one τ; the 15 entries are read step by step ($k = 1, \ldots, 5$). At the matrix level, step k applies (A57). The last column reports the converged loss $\mathcal{L}(\hat{\boldsymbol{\theta}}(\tau); \tau)$ of (A58).

τ	Step $k=1$			Step $k=2$			Step $k=3$			Step $k=4$			Step $k=5$			$\mathcal{L}(\hat{\boldsymbol{\theta}}; \tau)$
	$a_{1,1}$	$a_{3,1}$	$a_{5,1}$	$a_{1,2}$	$a_{3,2}$	$a_{5,2}$	$a_{1,3}$	$a_{3,3}$	$a_{5,3}$	$a_{1,4}$	$a_{3,4}$	$a_{5,4}$	$a_{1,5}$	$a_{3,5}$	$a_{5,5}$	
0.1	+4.753	−10.636	+7.172	+2.414	−2.282	+0.877	+2.589	−1.202	+0.245	+1.999	−1.774	+0.525	+2.131	−1.530	+0.274	0.00070
0.2	+3.104	−3.578	+1.639	+2.844	−2.041	+0.616	+2.577	−2.567	+0.639	+2.807	−1.811	+0.113	+1.877	−1.153	+0.292	0.00105
0.3	+2.547	−1.190	+0.122	+3.202	−1.581	+0.326	+3.100	−1.684	+0.229	+2.289	−1.547	+0.342	+2.185	−1.841	+0.645	0.00278
0.4	+2.624	−1.021	−0.555	+2.762	−1.221	+0.238	+2.682	−1.486	+0.293	+2.021	−1.724	+0.483	+2.154	−1.565	+0.283	0.00412
0.5	+2.461	−0.443	−0.811	+3.084	−1.139	+0.188	+2.612	−1.453	+0.220	+2.057	−1.837	+0.558	+2.043	−1.355	+0.224	0.00263
0.6	+2.313	−0.434	−0.335	+2.913	−0.920	+0.130	+2.751	−1.493	+0.217	+1.939	−1.683	+0.470	+2.253	−1.784	+0.353	0.00316
0.7	+1.636	−0.310	−0.039	+3.286	−1.566	+0.333	+3.036	−1.962	+0.338	+2.004	−1.730	+0.476	+2.204	−1.663	+0.313	0.00524
0.8	+1.743	−0.247	+0.015	+2.990	−0.933	+0.130	+2.656	−1.371	+0.189	+2.069	−1.663	+0.423	+2.054	−1.345	+0.220	0.00843
0.9	+3.009	−1.041	−0.021	+2.796	−2.170	+0.433	+3.051	−2.487	+0.507	+2.128	−2.166	+0.772	+2.174	−1.889	+0.709	0.00043



(a) $\tau = 0.1$ (b) $\tau = 0.2$ (c) $\tau = 0.3$

(d) $\tau = 0.4$ (e) $\tau = 0.5$ (f) $\tau = 0.6$

(g) $\tau = 0.7$ (h) $\tau = 0.8$ (i) $\tau = 0.9$

Figure A2: Odd scalar extension $\sigma_{\text{in}} \mapsto \sigma_{\text{out}}$ fitted for LPMuon at nine thresholds $\tau \in \{0.1, 0.2, \ldots, 0.9\}$ obtained from Alg. 4. In the actual SVD update only the nonnegative half $\sigma_{\text{in}} \in [0, 1]$ is applied to singular values; the plotted negative half visualizes the antisymmetric extension. Each panel anchors the pass band ($|\sigma| \leq \tau$) at ± 1 and contracts the stop band ($|\sigma| > \tau$) toward 0.

Q LIMITATIONS

Pion is designed for regimes where the informative descent direction concentrates in a few leading singular values, which is not the case for LLM pretraining: pretraining benefits from Muon's uniform whitening, which lifts every singular value to 1 and maximizes spectral exploration, whereas Pion's high-pass NS attenuates the tail and discards potentially useful directions. We therefore expect Pion to underperform Muon on LLM pretraining, and we leave to future work the question of how to adapt the high-pass cutoff to recover Muon's exploration behavior in pretraining while retaining Pion's noise robustness in VLA and RLVR.