COST-FREE SPECTRAL ESTIMATION FOR ADAPTIVE NEWTON–SCHULZ IN MATRIX OPTIMIZERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Matrix optimizers such as Muon transform each momentum matrix through an approximate orthogonalization, typically implemented by a small number of Newton–Schulz matrix multiplications. The quality and cost of this approximation depend strongly on the singular-value spectrum of its input, yet existing implementations use the same fixed polynomial routine for every layer and throughout training. We show that this uniform treatment is unnecessary: the computations in the Newton–Schulz method already reveal enough information to make the method adaptive. The Gram matrices formed inside Newton–Schulz iterations yield spectral moments through inexpensive scalar reductions, requiring no additional matrix multiplications. From these moments, we recover an estimate of the empirical singular-value distribution and use it to select a polynomial routine specialized to the current matrix. This turns Newton–Schulz orthogonalization into a spectrum-adaptive procedure that responds to differences across both layers and training time. On saved momentum matrices, spectral estimation substantially reduces orthogonalization error at a fixed iteration budget or reaches the same accuracy with fewer iterations, and in GPT pretraining up to 1B parameters it lowers the validation loss of two matrix optimizers. Our results suggest that matrix-function operations inside optimizers need not be designed for a conservative worst-case spectrum: they can cheaply measure the spectrum they are already processing and specialize computations accordingly.

1 INTRODUCTION

Matrix optimizers are an increasingly practical alternative to elementwise optimization for neural-network training. Methods such as Shampoo (Gupta et al., 2018; Anil et al., 2020), SOAP (Vyas et al., 2025), Dion (Ahn et al., 2025), and Muon (Jordan et al., 2024; Liu et al., 2025) exploit matrix structure in gradients, momentum, or preconditioning statistics rather than treating every parameter independently. Muon has attracted substantial attention for language-model pretraining, where it approximately orthogonalizes momentum matrices before using them as updates. Recent large-scale systems have adopted Muon or closely related matrix-optimization ideas, including DeepSeek-V4, GLM-5, and Kimi K3 (DeepSeek-AI, 2026; GLM-5 Team, 2026; Kimi Team, 2026), and practical studies show that Muon is competitive at language-model scale (Essential AI, 2025; Liu et al., 2025).

This growing use of matrix optimizers makes the cost of their internal matrix functions increasingly important. Muon approximately computes the polar factor of each momentum matrix with a small number of Newton–Schulz iterations, replacing an SVD with hardware-efficient matrix multiplications (Higham, 2008; Amsel et al., 2026a). Online KL Shampoo (OKLS) (Zhang et al., 2026a) similarly computes the inverse square roots of its preconditioner statistics with a coupled Newton–Schulz iteration. Recent work improves these iterations through better polynomial design and scaling (Grishina et al., 2025; Amsel et al., 2026a), but typically applies the same routine across matrices. This can be wasteful because the difficulty of the approximation depends strongly on the input spectrum, and momentum spectra vary across matrix types, layers, and training time, with attention often substantially broader than MLPs. A conservative routine that handles the hardest cases can spend unnecessary iterations on easier ones.

Several recent methods have begun to exploit this heterogeneity. AMO adapts orthogonalization at the matrix-type level using an observe-then-commit strategy (Zhuang et al., 2026), while PRISM adapts

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.

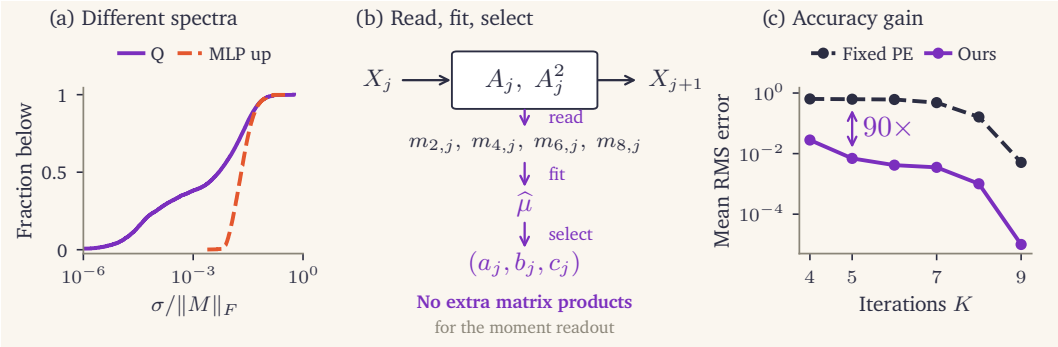

Figure 1: **Newton–Schulz can measure its input and specialize its own computation.** **(a)** Full singular-value CDFs show different momentum spectra for attention Q and MLP-up at layer 5, step 16k. **(b)** Read moments from existing matrix products, fit the spectrum, and select a routine for subsequent steps. **(c)** On 540 saved MLP matrices, spectral estimation reduces mean polar RMS error by $90\times$ relative to fixed Polar Express (PE) at $K = 5$ iterations, scored above 10^{-5} in Frobenius-normalized coordinates.

matrix-function computation using randomized spectral information during the iterations (Yang et al., 2026). Section G reviews related work. The best matrix-function routine depends on the spectrum it is applied to, so exploiting this during training requires spectral information that is cheap to obtain. Newton–Schulz already contains such information. Its intermediate matrices encode measurements of the input singular-value distribution that can be extracted through inexpensive scalar reductions, without additional matrix multiplications. We use them to estimate enough of the spectrum to select a better-matched routine, a procedure we call *spectral estimation* (Figure 1).

We make three contributions.

- We show that the intermediate matrices of Newton–Schulz yield spectral measurements at negligible cost. They need no additional matrix multiplications, and fitting a spectrum to them costs the same at every matrix size, unlike an SVD.
- We use these measurements to build a spectrum-adaptive Newton–Schulz procedure that selects a polynomial routine for each matrix instead of one conservative choice.
- We evaluate spectral estimation both offline on saved language-model momentum matrices and end-to-end in GPT pretraining. Offline, it substantially improves polar approximation at a fixed iteration budget or reaches the same accuracy with fewer Newton–Schulz iterations. In pretraining, it improves two different matrix optimizers, Muon and OKLS.

2 Matrix Functions in Optimization

2.1 From Matrix Updates to Spectral Transformations

Matrix optimizers act on gradients, momentum, or preconditioning statistics through matrix-valued transformations. Examples include Muon (Jordan et al., 2024; Liu et al., 2025), Dion (Ahn et al., 2025), AdaMuon (Si et al., 2025), and Shampoo-style methods (Gupta et al., 2018; Anil et al., 2020; Vyas et al., 2025). Muon is our main example. For a momentum matrix $M = U\Sigma V^\top$, its ideal update is the polar factor $\mathrm{polar}(M) = UV^\top$, which preserves singular-vector directions while mapping every nonzero singular value to one (Jordan et al., 2024).

Other matrix optimizers apply different spectral transformations. Shampoo and SOAP construct dense preconditioners from matrix-valued second-moment statistics, typically using eigendecomposition or related dense linear algebra (Gupta et al., 2018; Anil et al., 2020; Vyas et al., 2025). OKLS instead computes inverse square roots of those statistics using a coupled Newton–Schulz iteration (Zhang et al., 2026a). Both orthogonalization and dense preconditioning reduce to evaluating matrix functions such as the polar factor, inverse roots, or related spectral powers. Direct SVDs or eigendecompositions make these functions easy to define but expensive to evaluate repeatedly during training. Newton–

Schulz and related polynomial iterations replace them with short sequences of matrix multiplications that map efficiently to GPUs (Higham, 2008; Amsel et al., 2026a). We focus on polar approximation, while coupled variants provide analogous routes to inverse roots and fractional powers (Higham, 1997; Qi et al., 2026).

2.2 Newton–Schulz and optimal polynomial iterations

Newton–Schulz orthogonalization applies a sequence of low-degree polynomials to the singular values of a matrix. The classical update $x \mapsto (3x - x^3)/2$ drives normalized positive singular values toward one (Björck & Bowie, 1971; Kovarik, 1970). Modern variants accelerate this process through improved scaling and higher-degree polynomial iterations (Chen & Chow, 2014; Amsel et al., 2026a; Grishina et al., 2025). Orient $M \in \mathbb{R}^{r\times n}$ so that $r \le n$, and let X_0 be a rescaling whose singular values γ_i lie in $[0, 1]$. A quintic iteration takes the form

$$A_j = X_j X_j^\top, \qquad X_{j+1} = a_j X_j + (b_j A_j + c_j A_j^2) X_j. \tag{1}$$

If $p_j(x) = a_j x + b_j x^3 + c_j x^5$, then after K iterations,

$$X_K = U \operatorname{diag}(P(\gamma_i)) V^\top, \qquad P = p_{K-1} \circ \cdots \circ p_0. \tag{2}$$

The matrix iteration is therefore determined by its scalar response P, and the coefficients set how quickly singular values move toward one without changing an iteration's matrix-multiplication pattern.

Polar Express (PE) chooses these coefficients by minimizing worst-case approximation error over a prescribed spectral interval (Amsel et al., 2026a). For an iteration with design interval $[\ell_j, u_j]$,

$$(a_j^\star, b_j^\star, c_j^\star) \in \operatorname*{arg\,min}_{a,b,c\in\mathbb{R}} \max_{x\in[\ell_j,u_j]} |1 - (ax + bx^3 + cx^5)|. \tag{3}$$

If the resulting error is e_j, the next iteration is designed for the output interval $[1 - e_j, 1 + e_j]$. This greedy construction is optimal among compositions of the same depth and degree (Amsel et al., 2026a, Theorem 3.1), and CANS similarly uses Chebyshev-type design (Grishina et al., 2025).

This optimality is relative to the chosen spectral interval. For an initial design interval $[\ell_0, u_0]$, rescaling the upper endpoint to one leaves the ratio $\rho = \ell_0/u_0$ as the only relevant quantity. A broader interval therefore defines a harder problem.

3 Spectrum-dependent cost of Newton–Schulz

3.1 Broader design intervals require more computation

The following result shows how widening the design interval degrades the best error at fixed depth.

Theorem 1 (Interval width and minimax error)**.** *Let $\varepsilon_K^\star(\rho)$ be the minimum uniform error on $[\rho, 1]$ over all compositions of K odd polynomials of degree at most five, and write $e(\rho) = \varepsilon_1^\star(\rho)$ for the one-iteration error. In exact arithmetic, ideal PE attains this minimum, and for $K \ge 1$*

$$\varepsilon_{K+1}^\star(\rho) = e\left(\frac{1 - \varepsilon_K^\star(\rho)}{1 + \varepsilon_K^\star(\rho)}\right). \tag{4}$$

For $K \ge 1$ and $0 < \rho_1 < \rho_2 < 1$,

$$\varepsilon_K^\star(\rho_1) > \varepsilon_K^\star(\rho_2), \qquad \varepsilon_K^\star(\rho) \ge \left(\frac{1-\rho}{1+\rho}\right)^{5^K}, \quad 0 < \rho < 1. \tag{5}$$

By Theorem 1, a broader interval gives strictly larger minimax error at every fixed depth. As $\rho \to 0$, the error approaches one, so achieving a fixed target accuracy eventually requires more iterations. The proof and corresponding depth bound are in Section A.2. Figure 2 shows the effect directly. Broader intervals leave more error at the same depth because the polynomial must amplify small singular values while remaining bounded on the upper end. Narrower ones reach low error sooner.

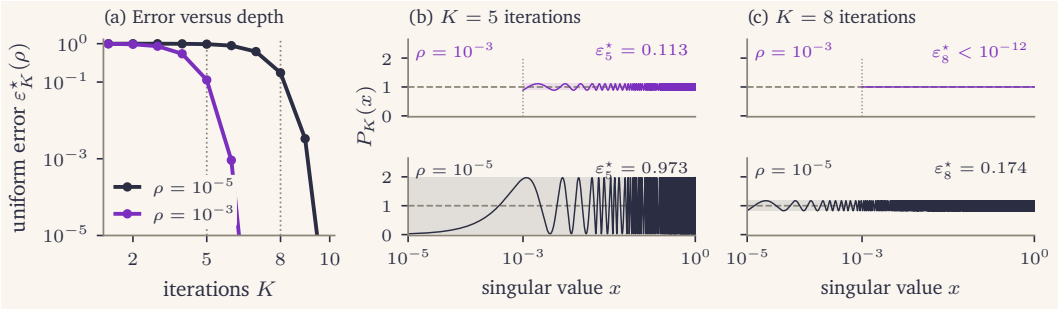

Figure 2: **Wider design intervals require more iterations.** Ideal PE on $[\rho, 1]$. **(a)** Optimal uniform error versus depth. **(b,c)** Composed responses at $K = 5$ and $K = 8$. Shading marks $1 \pm \varepsilon_K^\star(\rho)$.

3.2 Training produces different spectra across layers and time

Orthogonalization difficulty varies across matrix types, layers, and training time (Zhuang et al., 2026). Figure 3 shows this for attention-query and MLP-up matrices of the 160M Muon baseline of our pretraining experiments. Attention spectra are broader and MLP spectra more concentrated, and both vary across layers and training. A single conservative interval must cover all of these cases, even where it contains little spectral mass for a given matrix.

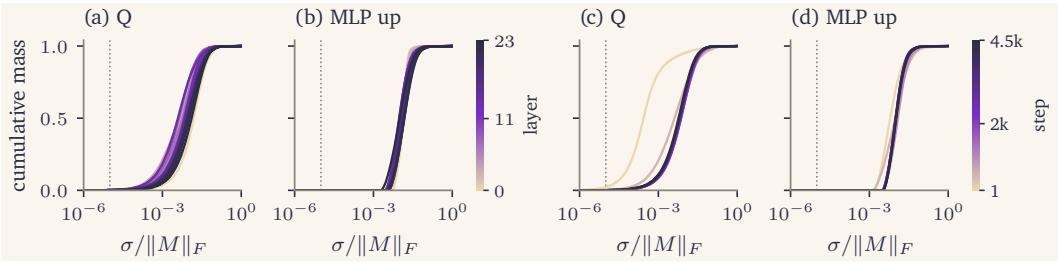

Figure 3: **Spectra vary across layers and training.** CDFs of the 160M Muon baseline. **(a,b)** Q and MLP-up across layers at step 2k. **(c,d)** Layer 11 across checkpoints. The dotted line marks 10^{-5}.

3.3 Spectral information identifies unnecessary work

An oracle that knows the exact nonzero spectral range can design PE for $[\sigma_{\min}, \sigma_{\max}]$. Figure 4 compares this with a common conservative interval. Attention still requires a relatively deep iteration, but the narrower MLP spectrum can reach the same target in far fewer iterations. Endpoints determine

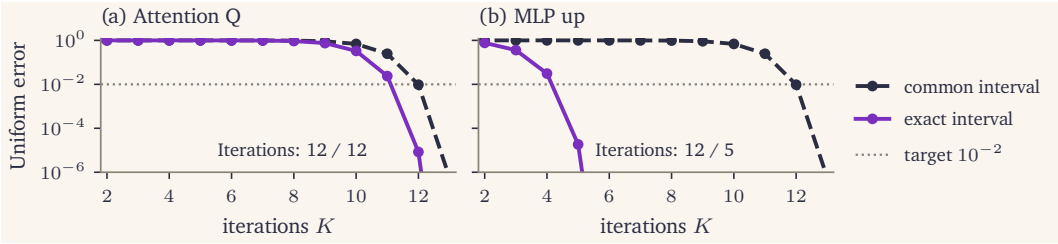

Figure 4: **A conservative interval spends unnecessary work on narrow spectra.** Q and MLP-up at layer 5, step 16k. The common interval is $[10^{-7}, 1]$, and the exact interval is each matrix's nonzero spectral range, as used by an oracle. Labels show the first depth reaching 10^{-2}.

the best uniform guarantee, but average error depends on where the spectral mass lies. For normalized

nonzero singular values γ_i, define

$$E_{\mathrm{RMS}}(P)^2 = \frac{1}{r_+}\sum_{i=1}^{r_+} |1 - P(\gamma_i)|^2, \qquad r_+ = \mathrm{rank}(M). \tag{6}$$

This is the squared relative Frobenius error to the polar factor. Minimizing worst-case error need not minimize this average, since allowing larger error in a sparse tail can improve accuracy over the bulk (Fan et al., 2020). We therefore use spectral information beyond the endpoints to score candidate routines. Obtaining this information is usually expensive. An exact SVD of $M \in \mathbb{R}^{r\times n}$ costs $O(r^2 n)$ in sequential steps that map poorly to GPUs, and this cost grows with the matrix size. Newton–Schulz provides this information without extra matrix products, at a cost that stays nearly constant as matrices grow (Section 4).

4 Spectral estimation from Newton–Schulz

Choosing the best routine requires information about the input spectrum, and computing it with an SVD would defeat the purpose. If an SVD were affordable at every step, we could compute the polar factor directly. Instead, Newton–Schulz itself lets us estimate the spectrum cheaply. We collect the measurements during a measurement pass and reuse the chosen routine over subsequent steps. For nonzero M, let $x_i = \sigma_i(M)/\|M\|_F$ and $X_0 = M/(s\|M\|_F)$, where s upper-bounds $\max_i x_i$. Then $\gamma_i = x_i/s \in [0,1]$ and $\mu = \frac{1}{r}\sum_{i=1}^{r}\delta_{\gamma_i}$ is the empirical scaled spectrum, including zeros.

4.1 Moments from the Newton–Schulz computation

Measurement pass

Require: X_0, coefficients (a_j, b_j, c_j)

for $j = 0, \ldots, K_{\mathrm{meas}} - 1$ **do**
  $A_j \leftarrow X_j X_j^\top$ (1)
  $B_j \leftarrow A_j A_j$ (2)
  $(m_{2,j}, m_{4,j}) \leftarrow (\mathrm{tr}\, A_j, \|A_j\|_F^2)/r$
  $(m_{6,j}, m_{8,j}) \leftarrow (\langle A_j, B_j\rangle_F, \|B_j\|_F^2)/r$
  $C_j \leftarrow b_j A_j + c_j B_j$
  $X_{j+1} \leftarrow a_j X_j + C_j X_j$ (3)
end for
Read $\|X_{K_{\mathrm{meas}}}\|_F^2/r$

(1)–(3) are existing matrix products.
Purple marks added scalar reductions.

Newton–Schulz already forms the matrices needed to read spectral moments. At iteration j, the intermediate matrices $A_j = X_j X_j^\top$ and $B_j = A_j^2$ expose the even moments of orders two through eight using only scalar reductions. No additional matrix multiplication is required. Because each iteration applies a different polynomial transform, successive readouts measure different views of the same input spectrum (Weiße et al., 2006; Lin et al., 2016). To see this, write $X_0 = U\,\mathrm{diag}(\gamma_i)V^\top$ and define $z_0(x) = x$, $z_{j+1}(x) = p_j(z_j(x))$, and $Z_j = \mathrm{diag}(z_j(\gamma_i))$. Then Equation (1) gives

$$X_j = UZ_jV^\top, \quad A_j = UZ_j^2U^\top, \quad B_j = UZ_j^4U^\top. \tag{7}$$

Since A_j is symmetric, $\langle A_j, B_j\rangle_F = \mathrm{tr}(A_j^3)$ and $\|B_j\|_F^2 = \mathrm{tr}(A_j^4)$, giving

$$m_{2,j} = \mathrm{tr}(A_j)/r,\ m_{4,j} = \|A_j\|_F^2/r,\ m_{6,j} = \langle A_j, B_j\rangle_F/r,\ m_{8,j} = \|B_j\|_F^2/r. \tag{8}$$

The cost of these reductions grows only with the smaller matrix dimension and is a vanishing fraction of each iteration (Section D).

Proposition 1 (Spectral moments from the iteration)**.** *In exact arithmetic,*

$$m_{2k,j} = \frac{\mathrm{tr}\, A_j^k}{r} = \frac{1}{r}\sum_{i=1}^{r} z_j(\gamma_i)^{2k} = \int z_j(x)^{2k}\,\mathrm{d}\mu(x), \qquad k = 1, \ldots, 4. \tag{9}$$

For $j = 0, \ldots, K_{\mathrm{meas}} - 1$, this gives four measurements per iteration. The final output contributes

$$m_{2,K_{\mathrm{meas}}} = \|X_{K_{\mathrm{meas}}}\|_F^2/r = \int z_{K_{\mathrm{meas}}}(x)^2\,\mathrm{d}\mu(x),$$

for $4K_{\mathrm{meas}} + 1$ scalar measurements in total.

Each pair (j,k) therefore defines a known measurement function $\phi_q(x) = z_j(x)^{2k}$ with observation $y_q = \int \phi_q\,\mathrm{d}\mu$. These functions constrain several transformed views of the same spectrum from one pass. We choose s using the CANS Gram-power upper bound (Grishina et al., 2025, Section 3.3), reusing the first-iteration products, which safely bounds the spectrum from above.

4.2 Fitting a spectrum to the measured moments

We recover a discrete spectral measure from the measurement functions using a maximum-entropy moment fit (Mead & Papanicolaou, 1984; Silver & Röder, 1997). Place masses w_b on grid points $g_b \in [0, 1]$, and define $\Phi_{qb} = \phi_q(g_b)$. Then Φw gives the predicted measurements. We enforce normalization and the known second moment by restricting w to

$$\Delta = \left\{ w \geq 0 : \mathbf{1}^\top w = 1, \qquad \sum_b w_b g_b^2 = \frac{1}{rs^2} \right\}. \tag{10}$$

Because discretization and measurement error may preclude an exact match, we first minimize the weighted residual, then select the maximum-entropy fit within tolerance κ. With measured values $\widehat{y}_q$, row scales d_q that put them on a common scale, and $D = \mathrm{diag}(1/d_q)$,

$$\xi^\star = \min_{w \in \Delta} \|D(\Phi w - \widehat{y})\|_2, \qquad \widehat{w} = \arg\max_{w \in \Delta} \Big\{ -\sum_b w_b \log w_b : \|D(\Phi w - \widehat{y})\|_2 \leq \xi^\star + \kappa \Big\}. \tag{11}$$

Both problems are convex. When several grid distributions match the measurements, the entropy objective selects a unique representative. Figure 5 compares the fitted measures with the SVD spectra. The fit captures broad attention and concentrated MLP spectra, with close agreement in cumulative mass.

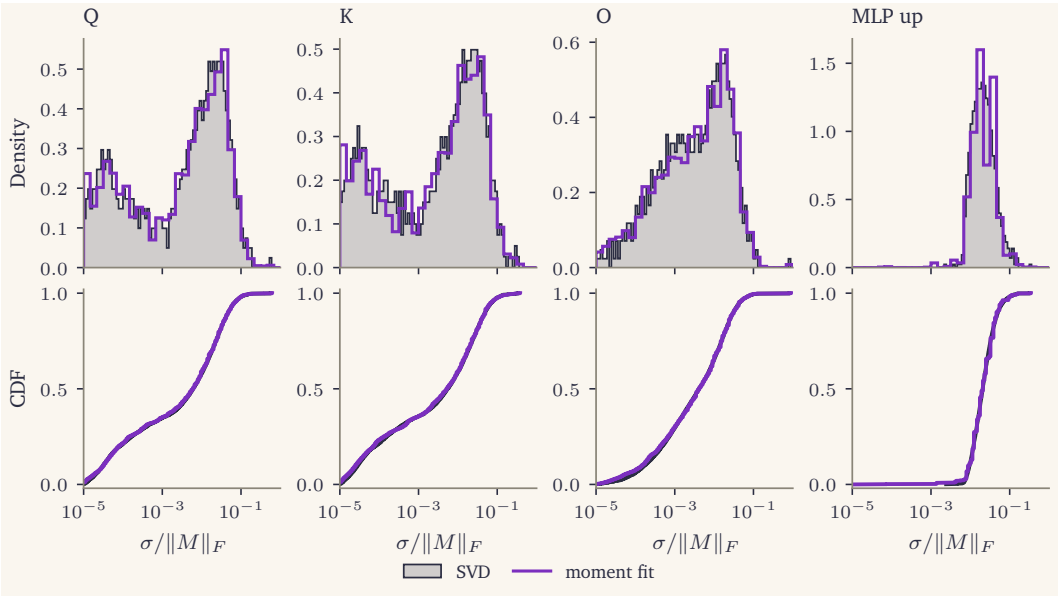

Figure 5: **Moment fits recover useful spectral structure.** Four matrices at layer 5, step 16k. The top row shows the density in $\log_{10}(\sigma/\|M\|_F)$ and the bottom row the CDF, both conditioned on $\sigma/\|M\|_F \geq 10^{-5}$.

4.3 What the fitted spectrum can predict

We do not need to recover the spectrum uniquely, only to predict the spectral quantities used to score candidate routines. Although finitely many moments may not identify a distribution (Krein & Nudel'man, 1977), they can determine a particular spectral average when its integrand is well represented by the measurements (Braverman et al., 2022).

Theorem 2 (Prediction from the fitted spectrum)**.** *Let μ be the true spectral measure on $[0,1]$, and let $\phi_1, \dots, \phi_L$ be the measurement functions with observations $\widehat{y}_q$ satisfying $|\widehat{y}_q - \int \phi_q \, \mathrm{d}\mu| \le \omega_q$. Let $\widehat{w}$ solve Equation (11) and define $\widehat{\mu} = \sum_b \widehat{w}_b \delta_{g_b}$. For any bounded measurable f and any affine combination of the measurement functions $\psi_\alpha = \alpha_0 + \sum_{q=1}^{L} \alpha_q \phi_q$, define $\beta_\alpha(f) = \sup_{x\in[0,1]} |f(x) - \psi_\alpha(x)|$. Then, with $\alpha = (\alpha_1, \dots, \alpha_L)$ and D, $\xi^\star$ as in Equation (11),*

$$\left| \int f \, \mathrm{d}\widehat{\mu} - \int f \, \mathrm{d}\mu \right| \le 2\beta_\alpha(f) + \sum_{q=1}^{L} |\alpha_q| \omega_q + (\xi^\star + \kappa) \|D^{-1}\alpha\|_2. \tag{12}$$

The three terms capture how well f is represented by the measured functions, error in the observed values, and the fit residual. In exact arithmetic the reductions are exact, so $\omega_q = 0$ and the middle term covers only numerical error. If f lies in their affine span and the fitted moments are exact, its spectral average is recovered exactly even when the spectrum is not uniquely determined.

4.4 Selecting a Polar Express routine

The fitted spectrum thus predicts the averages that determine candidate quality, and we use it to score a precomputed set of PE routines. Each entry $\pi = (\rho_\pi, K_\pi) \in \mathcal{C}$ is a complete PE sequence with response P_π, designed for $[\rho_\pi, 1]$. Here ρ_π only sets the routine's design interval. All candidates are scored on the same spectral interval $\mathcal{R} = [x_\star/s, 1]$, where $x_\star$ is fixed in Frobenius-normalized coordinates. Their true and predicted squared RMS errors are

$$E_{\mathcal{R}}(P_\pi)^2 = \frac{\int_{\mathcal{R}} (1 - P_\pi(x))^2 \, \mathrm{d}\mu(x)}{\mu(\mathcal{R})}, \qquad \widehat{E}_\pi^2 = \frac{\sum_{g_b \in \mathcal{R}} \widehat{w}_b (1 - P_\pi(g_b))^2}{\sum_{g_b \in \mathcal{R}} \widehat{w}_b}. \tag{13}$$

Since candidate responses are precomputed, scoring needs only scalar operations on the fitted spectrum, with no candidate matrix runs. The same scores support two objectives. The *fixed-depth rule* picks the best routine at a given depth K, $\widehat{\pi}_K \in \arg\min_{\pi\in\mathcal{C},\, K_\pi = K} \widehat{E}_\pi$. The *target rule* picks the shortest routine predicted to meet an accuracy τ, $\widehat{\pi}_\tau \in \arg\min_{\pi\in\mathcal{C},\, \widehat{E}_\pi \le \tau} K_\pi$, and falls back to fixed PE at the largest allowed depth if no candidate meets τ. Figure 6 illustrates both choices.

These three steps, measure, fit, and select, make up spectral estimation. Beyond early training, spectra change gradually, so we measure every T steps and reuse the selected routine in between. Offline, we instead fit each saved checkpoint independently.

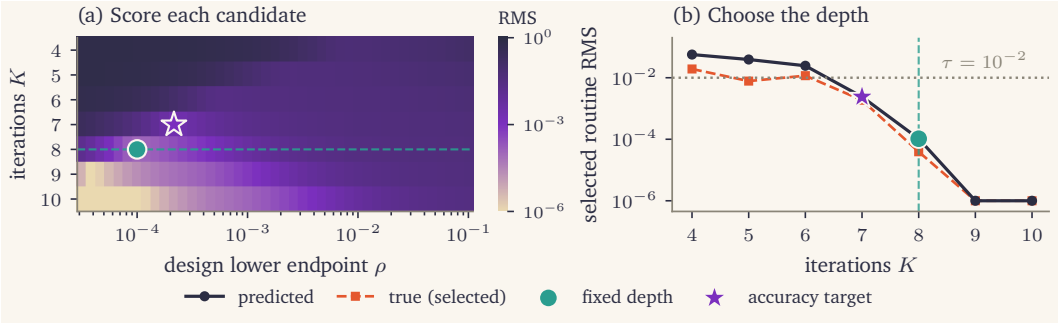

Figure 6: **One set of predicted scores supports both objectives.** MLP-up, layer 5, step 16k. **(a)** Predicted errors, with the best depth-8 endpoint (circle) and the shortest routine meeting $\tau = 10^{-2}$ (star). **(b)** Predicted and SVD-evaluated errors of the selected routines. Values below 10^{-6} are shown at that floor.

5 Experiments

5.1 Orthogonalization of real spectra

We evaluate saved Muon momentum spectra from a FineWeb run (Penedo et al., 2024). Baselines are fixed conservative PE on $[10^{-5}, 1]$ (Amsel et al., 2026a) and a minimax oracle. Using the exact

spectrum, the oracle selects from the same candidate set the routine with the smallest worst-case error $E_{\max}(P) = \max_{x\in[\gamma_{\min},\gamma_{\max}]} |1 - P(x)|$, where $[\gamma_{\min}, \gamma_{\max}]$ spans the scored singular values. Spectral estimation instead minimizes predicted RMS error, which the fitted spectrum predicts more reliably than worst-case error, so we report both errors. Both share normalization, candidate set, and scoring interval. Offline moments are computed exactly from the saved spectra, and bf16 measurements give nearly the same selections (Section F.4).

At fixed depth, selection reduces RMS error most strongly on MLP matrices (Figure 7 and Table 1). Because it targets RMS error, spectral estimation has lower RMS error than the minimax oracle on attention through $K = 9$ and on MLP through $K = 6$, while the oracle has lower worst-case error at every depth.

With a target accuracy, spectral estimation picks shorter routines than fixed PE, especially for MLP matrices (Figure 7c–d, Table 2).

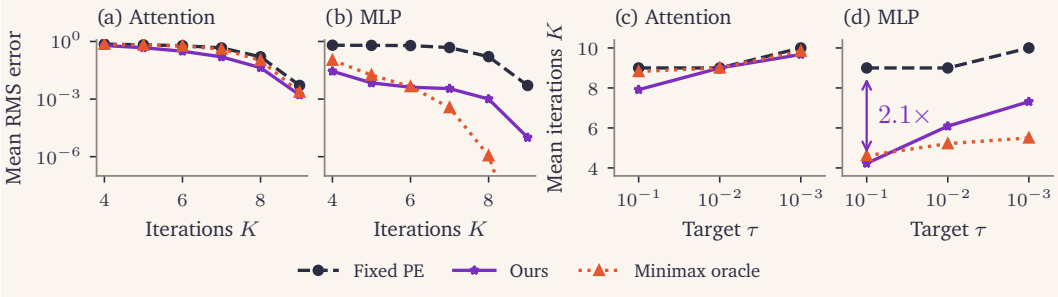

Figure 7: **Offline accuracy and depth.** Means over attention and MLP matrices, scored above 10^{-5} in Frobenius-normalized coordinates. **(a–b)** Polar RMS error at fixed depth (Table 1). **(c–d)** Iterations needed to meet a target τ (Table 2). Ours meets τ in predicted RMS error, the oracle in worst-case error, and fixed PE is the uniform depth that meets τ.

Table 1: **Fixed-depth polar error.** Mean per-matrix RMS and worst-case error (minimax oracle).

		RMS error			Worst-case error		
	K	Fixed PE	Ours	Oracle	Fixed PE	Ours	Oracle
Attention	5	0.667	**0.472**	0.653	0.976	0.982	**0.966**
	8	0.159	**0.0431**	0.104	0.222	0.353	**0.161**
MLP	5	0.626	**0.00694**	0.0179	0.976	0.0643	**0.0266**
	8	0.162	0.00101	$\mathbf{1.13\times10^{-6}}$	0.222	0.00153	$\mathbf{1.78\times10^{-6}}$

Table 2: **Mean depth K at target τ** (ours in predicted RMS error, oracle in worst-case error).

	Attention			MLP		
τ	10^{-1}	10^{-2}	10^{-3}	10^{-1}	10^{-2}	10^{-3}
Ours	**7.91**	**9.00**	**9.68**	**4.23**	6.09	7.31
Oracle	8.82	**9.00**	9.84	4.61	**5.21**	**5.50**

5.2 LLM PRETRAINING

We pretrain GPT-style decoder-only language models (Radford et al., 2019) with 160M, 300M, and 1B non-embedding parameters on 20B tokens of Nemotron-CC v2 High-Quality (NVIDIA, 2025a;b), using the experimental setup of Zhang et al. (2026a) with minor changes (Section F.1). Muon and OKLS need different matrix functions. Muon needs the polar factor of each momentum matrix, and OKLS the inverse square roots of Kronecker-factored second-moment statistics, computed by a coupled Newton–Schulz iteration. Our method handles both with the same measure, fit, and select

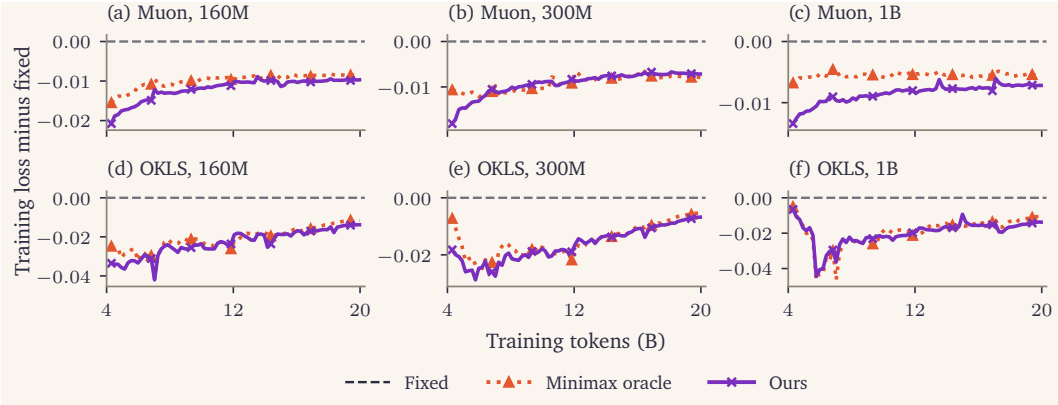

Figure 8: **Training loss relative to the fixed baseline.** Training loss minus the fixed baseline at $K = 5$ in 50-step windows, where negative is better.

procedure, changing only the measured products and candidate routines (Section E), so it is not specific to orthogonalization. Each run fixes the depth K of ordinary (non-measurement) steps and varies only the routine. With the fixed-depth RMS rule of Section 4.4, our method selects among K-iteration routines that differ only in their endpoint ρ, namely $\mathrm{PE}[\rho, 1]$ with $\rho \in [10^{-5}, 10^{-1}]$ for Muon and cubic inverse-root routines with $\rho \in [10^{-7}, 1]$ for OKLS. The oracle selects from the same set by the minimax rule every 500 steps. The fixed baseline is $\mathrm{PE}[10^{-5}, 1]$ for Muon and, for OKLS, the cubic routine for eigenvalues above $10^{-7}\lambda_{\max}$.

Table 3: **Pretraining across optimizers and sizes.** Final validation loss, $K = 5$, 20B tokens, with $T = 200$ for ours.

	Muon			OKLS		
Size	Fixed	Oracle	Ours	Fixed	Oracle	Ours
160M	2.761	2.752	**2.751**	2.687	2.675	**2.673**
300M	2.662	**2.654**	2.655	2.597	2.591	**2.590**
1B	2.455	2.450	**2.449**	2.449	2.439	**2.436**

Table 4: **Muon 160M across depths.** Final validation loss.

K	Fixed	Ours
4	2.808	**2.791**
5	2.761	**2.751**
6	2.754	**2.752**
8	**2.752**	**2.752**

Spectral estimation improves on the fixed baseline for both optimizers at every model size, and it matches or improves on periodic selection from an exact SVD or eigendecomposition (Table 3 and Figure 8).

We also vary the depth K of ordinary Muon steps at 160M (Table 4). The fixed baseline saturates quickly. With 6 and 8 iterations it reaches the same validation loss, so more accurate orthogonalization brings no further gain. With only five iterations, our method reaches a lower loss than the fixed baseline at any depth, so adaptation can replace extra iterations.

6 Conclusion

The Newton–Schulz method does not need to treat the spectrum it acts on as unknown, because its iterations already produce measurements that can be reused to specialize its own computations. We use these measurements to select spectrum-adapted polynomial routines, improving approximation at fixed depth or reducing the iterations needed to reach a target accuracy. We evaluate the method on saved momentum spectra and in end-to-end GPT pretraining. Useful adaptation does not require recovering the spectrum exactly, only predicting which candidate routine will work well. Because the measurements come from matrices the iteration already forms and the fit works on a fixed number of scalars, this adaptation stays cheap at the matrix sizes of current frontier models, where computing exact spectra becomes expensive.

AI USE STATEMENT

We used generative AI tools to assist with coding, writing and editing, and checking the manuscript for mistakes and inconsistencies. The authors reviewed and verified all final content.

REPRODUCIBILITY STATEMENT

All hyperparameters are in the appendix. The supplementary material provides saved spectra together with code that reproduces our results on them.

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. URL https://arxiv.org/abs/2504.05295.

Noah Amsel, David Persson, Christopher Musco, and Robert M. Gower. The Polar Express: Optimal matrix sign methods and their application to the Muon algorithm. In *International Conference on Learning Representations*, 2026a. URL https://arxiv.org/abs/2505.16932.

Noah Amsel, Jack Zhang, Kwangjun Ahn, Ali Naeimi, Austin Feng, Berlin Chen, Tri Dao, and John Langford. Dion3: Full-stack orthogonal updates. *arXiv preprint arXiv:2608.11612*, 2026b. URL https://arxiv.org/abs/2608.11612.

Rohan Anil, Vineet Gupta, Tomer Koren, Kevin Regan, and Yoram Singer. Scalable second order optimization for deep learning. *arXiv preprint arXiv:2002.09018*, 2020. URL https://arxiv.org/abs/2002.09018.

Åke Björck and C. Bowie. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364, 1971. doi: 10.1137/0708036. URL https://doi.org/10.1137/0708036.

Vladimir Braverman, Aditya Krishnan, and Christopher Musco. Sublinear time spectral density estimation. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1144–1157, 2022. doi: 10.1145/3519935.3520009. URL https://arxiv.org/abs/2104.03461.

Jie Chen and Edmond Chow. A stable scaling of Newton–Schulz for improving the sign function computation of a Hermitian matrix. Technical Report ANL/MCS-P5059-0114, Argonne National Laboratory, 2014. URL https://faculty.cc.gatech.edu/~echow/pubs/chen-chow-2014.pdf.

Tyler Chen, Thomas Trogdon, and Shashanka Ubaru. Analysis of stochastic Lanczos quadrature for spectrum approximation. In *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1728–1739, 2021. URL https://proceedings.mlr.press/v139/chen21s.html.

Sayantan Choudhury, Xiaoran Cheng, Martin Takáč, Sen Na, and Mladen Kolar. Muon with Nesterov momentum: Heavy-tailed noise and (randomized) inexact polar decomposition. *arXiv preprint arXiv:2605.06884*, 2026. URL https://arxiv.org/abs/2605.06884.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Bishnu Dev, Sushil Bohara, Martin Takáč, and Samuel Horváth. CacheMuon: Using temporal preconditioning to approximate polar factor. *arXiv preprint arXiv:2606.16371*, 2026. URL https://arxiv.org/abs/2606.16371.

Thang Do, Steffen Dereich, and Arnulf Jentzen. On MUON optimization: From non-convergence to an error analysis with Polar Express and the Newton–Schulz polynomial from implementations. *arXiv preprint arXiv:2608.04607*, 2026. URL https://arxiv.org/abs/2608.04607.

Essential AI. Practical efficiency of Muon for pretraining. *arXiv preprint arXiv:2505.02222*, 2025. URL https://arxiv.org/abs/2505.02222.

Tiffany Fan, David I. Shuman, Shashanka Ubaru, and Yousef Saad. Spectrum-adapted polynomial approximation for matrix functions with applications in graph signal processing. *Algorithms*, 13(11):295, 2020. doi: 10.3390/a13110295. URL https://www.mdpi.com/1999-4893/13/11/295.

GLM-5 Team. GLM-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026. URL https://arxiv.org/abs/2602.15763.

Ekaterina Grishina, Matvey Smirnov, and Maxim Rakhuba. Accelerating Newton–Schulz iteration for orthogonalization via Chebyshev-type polynomials. *arXiv preprint arXiv:2506.10935*, 2025. URL https://arxiv.org/abs/2506.10935v2. Version 2.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1842–1850, 2018. URL https://proceedings.mlr.press/v80/gupta18a.html.

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*, 2025. URL https://arxiv.org/abs/2511.20626.

Nicholas J. Higham. Stable iterations for the matrix square root. *Numerical Algorithms*, 15(2):227–242, 1997. doi: 10.1023/A:1019150005407. URL https://eprints.maths.manchester.ac.uk/367/.

Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, 2008. doi: 10.1137/1.9780898717778. URL https://doi.org/10.1137/1.9780898717778.

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 and implementation.

Ahmed Khaled, Kaan Ozkara, Tao Yu, Mingyi Hong, and Youngsuk Park. MuonBP: Faster Muon via block-periodic orthogonalization. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/1d485d337d5622bdb1462333f4247522-Abstract-Conference.html. arXiv:2510.16981.

Gyu Yeol Kim and Min-hwan Oh. Convergence of Muon with Newton–Schulz. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2601.19156.

Kimi Team. Kimi K3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026. URL https://arxiv.org/abs/2607.24653.

Zdislav Kovarik. Some iterative methods for improving orthonormality. *SIAM Journal on Numerical Analysis*, 7 (3):386–389, 1970. doi: 10.1137/0707031. URL https://doi.org/10.1137/0707031.

Mark G. Krein and Adolf A. Nudel'man. *The Markov Moment Problem and Extremal Problems*, volume 50 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 1977. doi: 10.1090/mmono/050.

Lin Lin, Yousef Saad, and Chao Yang. Approximating spectral densities of large matrices. *SIAM Review*, 58(1): 34–65, 2016. doi: 10.1137/130934283. URL https://doi.org/10.1137/130934283.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Lawrence R. Mead and N. Papanicolaou. Maximum entropy in the problem of moments. *Journal of Mathematical Physics*, 25(8):2404–2417, 1984. doi: 10.1063/1.526446. URL https://bayes.wustl.edu/Manual/MeadPapanicolaou.pdf.

NVIDIA. Nemotron-CC-v2. Dataset, https://huggingface.co/datasets/nvidia/Nemotron-CC-v2, 2025a. Released August 18, 2025.

NVIDIA. NVIDIA Nemotron Nano 2: An accurate and efficient hybrid Mamba-Transformer reasoning model. *arXiv preprint arXiv:2508.14444*, 2025b. URL https://arxiv.org/abs/2508.14444.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 37, pp. 30811–30849, 2024. URL https://arxiv.org/abs/2406.17557.

Xianbiao Qi, Marco Chen, Jiaquan Ye, Yelin He, and Rong Xiao. Delving into Muon and beyond: Deep analysis and extensions. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.04669.

Xun Qian and Peter Richtárik. Convergence analysis of Muon-type methods with inexact LMO in the degenerate case. *arXiv preprint arXiv:2606.21581*, 2026. URL https://arxiv.org/abs/2606.21581.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

Egor Shulgin, Sultan AlRashed, Francesco Orabona, and Peter Richtárik. Beyond the ideal: Analyzing the inexact Muon update. In *International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 5167–5175, 2026. URL https://proceedings.mlr.press/v300/shulgin26a.html.

Chongjie Si, Debing Zhang, and Wei Shen. AdaMuon: Adaptive Muon optimizer. *arXiv preprint arXiv:2507.11005*, 2025. URL https://arxiv.org/abs/2507.11005.

R. N. Silver and H. Röder. Calculation of densities of states and spectral functions by Chebyshev recursion and maximum entropy. *Physical Review E*, 56(4):4822–4829, 1997. doi: 10.1103/PhysRevE.56.4822. URL https://doi.org/10.1103/PhysRevE.56.4822.

Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of $\mathrm{tr}(f(A))$ via stochastic Lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017. doi: 10.1137/16M1104974. URL https://doi.org/10.1137/16M1104974.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. SOAP: Improving and stabilizing Shampoo using Adam for language modeling. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=IDxZhXrpNf.

Alexander Weiße, Gerhard Wellein, Andreas Alvermann, and Holger Fehske. The kernel polynomial method. *Reviews of Modern Physics*, 78(1):275–306, 2006. doi: 10.1103/RevModPhys.78.275. URL https://doi.org/10.1103/RevModPhys.78.275.

Shenghao Yang, Zhichao Wang, Oleg Balabanov, N. Benjamin Erichson, and Michael W. Mahoney. PRISM: Distribution-free adaptive computation of matrix functions for accelerating neural network training. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2601.22137.

Ashley Zhang, Ben Keigwin, Dhruv Pai, and Alec Dewulf. Online KL Shampoo. Tilde Research blog, https://blog.tilderesearch.com/blog/online-kl-shampoo, 2026a. Code: https://github.com/tilde-research/online-kl-shampoo-release.

Jack Zhang, Noah Amsel, Berlin Chen, and Tri Dao. Gram Newton–Schulz: A fast, hardware-aware Newton–Schulz algorithm for Muon. Dao AI Lab blog, 2026b. URL https://dao-lab.ai/blog/2026/gram-newton-schulz/. Code: https://github.com/Dao-AILab/gram-newton-schulz.

Xinlin Zhuang, Panyi Ouyang, Yichen Li, Jiangming Shi, Yizhang Chen, Shuman Liu, Ying Qian, Weiyang Liu, Haibo Zhang, and Imran Razzak. AMO: Adaptive Muon orthogonalization. *arXiv preprint arXiv:2605.17806*, 2026. URL https://arxiv.org/abs/2605.17806.

A PROOFS

A.1 PROOF OF PROPOSITION 1

For a thin SVD $X = U\,\mathrm{diag}(x_i)V^\top$ with orthonormal columns in U and V, $(XX^\top)^k X = U\,\mathrm{diag}(x_i^{2k+1})V^\top$. Applying this identity iteration by iteration gives Equation (2).

Proof. The eigenvalues of A_j are $z_j(\gamma_i)^2$, so $\mathrm{tr}\,A_j^k = \sum_i z_j(\gamma_i)^{2k}$. For symmetric A_j,

$$\mathrm{tr}\,A_j^2 = \|A_j\|_F^2, \qquad \mathrm{tr}\,A_j^3 = \langle A_j, A_j^2\rangle_F, \qquad \mathrm{tr}\,A_j^4 = \|A_j^2\|_F^2.$$

The final iterate satisfies $\|X_{K_{\mathrm{meas}}}\|_F^2 = \sum_i z_{K_{\mathrm{meas}}}(\gamma_i)^2$. This proves Equation (9). □

Singular values below the scoring cutoff. Let $x_\star$ be the scoring cutoff, $\mathcal{R} = [x_\star/s, 1]$ the scoring interval, and $m_{\mathcal{R}} = \mu(\mathcal{R})$ the fraction of singular values in it. For a full-row-rank input, the output is $X_K = U\,\mathrm{diag}(P(\gamma_i))V^\top$ and the polar factor is $UV^\top$, so

$$\frac{\|X_K - UV^\top\|_F^2}{r} = m_{\mathcal{R}} E_{\mathcal{R}}(P)^2 + \frac{1}{r}\sum_{\gamma_i \notin \mathcal{R}} (1 - P(\gamma_i))^2. \tag{14}$$

Our score covers only the first term. In the offline data, on average 3.8% of attention singular values lie below the cutoff (median 2.0%, at most 24%), and no MLP singular values do. For rank-deficient inputs the same decomposition holds with the partial polar factor, whose response is zero on zero singular values.

A.2 PROOF OF THEOREM 1

Proof. We first derive the recursion using the composition-optimality result of Amsel et al. (2026a, Theorem 3.1). The change of variables $x = ut$ maps a quintic on $[\ell, u]$ to one on $[\ell/u, 1]$, so the one-iteration optimum depends only on $\rho = \ell/u$. If an optimal iteration has error δ, its image is $[1-\delta, 1+\delta]$ by continuity and equioscillation. The next optimal error is therefore $e((1-\delta)/(1+\delta))$, which gives Equation (4).

For strict monotonicity, $\mathrm{span}\{x, x^3, x^5\}$ is a Haar space on every positive interval, so the minimax residual has four alternating extrema. Restricting a minimizer from $[\rho_1, 1]$ to $[\rho_2, 1]$ with $\rho_1 < \rho_2$ removes the leftmost extremum, so it cannot remain minimax with the same error. Hence the one-iteration error is strictly decreasing in ρ. Since $t \mapsto (1-t)/(1+t)$ is strictly decreasing, induction on K gives strict monotonicity at every depth.

For the lower bound, let P be an odd polynomial of degree at most d and $E = \max_{x\in[\rho,1]} |1 - P(x)|$. The case $E \geq 1$ is immediate. Otherwise $P(x) \in [1-E, 1+E]$, and $R(y) = 1 - P(\sqrt{y})^2/(1+E^2)$ is a polynomial of degree at most d with $R(0) = 1$ and $\max_{y\in[\rho^2,1]} |R(y)| \leq 2E/(1+E^2)$. Mapping $[\rho^2, 1]$ to $[-1, 1]$ and applying the Chebyshev extremal bound at the image of $y = 0$ gives

$$\frac{2E}{1+E^2} \geq \frac{1}{\cosh(dL)}, \qquad L = \log\frac{1+\rho}{1-\rho}.$$

Because $e \mapsto 2e/(1+e^2)$ is increasing on $[0, 1]$ and equals $1/\cosh(dL)$ at $e = \exp(-dL)$, we get $E \geq \exp(-dL) = ((1-\rho)/(1+\rho))^d$. A composition of K quintics has degree at most 5^K, which proves Equation (5). □

Consequently, reaching a uniform error τ on $[\rho, 1]$ requires

$$5^K \geq \frac{\log(1/\tau)}{\log((1+\rho)/(1-\rho))}. \tag{15}$$

A.3 Proof of Theorem 2 and accuracy of selection

Proof. The set Δ is a compact convex subset of the simplex, so the best-fit minimum in Equation (11) is attained, and its minimizer is feasible for the maximum-entropy problem. The feasible set of the latter is compact and convex, and entropy is continuous and strictly concave, so the fitted measure exists and is unique.

Insert $\psi_\alpha = \alpha_0 + \sum_q \alpha_q \phi_q$ between the two integrals. The constant α_0 cancels because both measures have unit mass, so

$$\int f \,\mathrm{d}\widehat{\mu} - \int f \,\mathrm{d}\mu = \int (f - \psi_\alpha)\,\mathrm{d}\widehat{\mu} - \int (f - \psi_\alpha)\,\mathrm{d}\mu + \alpha^\top(\Phi\widehat{w} - \widehat{y}) + \alpha^\top\left(\widehat{y} - \int \phi \,\mathrm{d}\mu\right).$$

The first two terms are at most $2\beta_\alpha(f)$ in absolute value, and the last is at most $\sum_q |\alpha_q|\omega_q$. Finally, $|\alpha^\top(\Phi\widehat{w} - \widehat{y})| = |(D^{-1}\alpha)^\top D(\Phi\widehat{w} - \widehat{y})| \le \|D^{-1}\alpha\|_2(\xi^\star + \kappa)$. Adding the terms gives Equation (12). □

From integrals to selection. The score in Equation (13) is the ratio of the error integral $J_\pi = \int_{\mathcal{R}}(1 - P_\pi)^2 \,\mathrm{d}\mu$ to the scored mass $m_{\mathcal{R}}$. Theorem 2 applies to each. If the estimates satisfy $|\widehat{J}_\pi - J_\pi| \le \delta_{J_\pi}$, $|\widehat{m}_{\mathcal{R}} - m_{\mathcal{R}}| \le \delta_m$, and $\widehat{m}_{\mathcal{R}} \ge m_0 > 0$, and $(1 - P_\pi)^2 \le C_\pi$ on $\mathcal{R}$, then

$$\left|\frac{\widehat{J}_\pi}{\widehat{m}_{\mathcal{R}}} - \frac{J_\pi}{m_{\mathcal{R}}}\right| \le \frac{\delta_{J_\pi} + C_\pi \delta_m}{m_0}. \tag{16}$$

Accurate scores give accurate selection. Write $E_\pi = E_{\mathcal{R}}(P_\pi)$, let $\widehat{\pi}$ minimize the predicted squared error $\widehat{E}^2_\pi$ and $\pi^\star$ the true one over the same candidates. If $|\widehat{E}^2_\pi - E^2_\pi| \le \delta_\pi$ for every candidate, then

$$E^2_{\widehat{\pi}} - E^2_{\pi^\star} \le \delta_{\widehat{\pi}} + \delta_{\pi^\star}, \tag{17}$$

since $E^2_{\widehat{\pi}} \le \widehat{E}^2_{\widehat{\pi}} + \delta_{\widehat{\pi}} \le \widehat{E}^2_{\pi^\star} + \delta_{\widehat{\pi}} \le E^2_{\pi^\star} + \delta_{\pi^\star} + \delta_{\widehat{\pi}}$.

B Method details

B.1 Input normalization

For $\bar{X} = M/\|M\|_F$ and $\bar{A} = \bar{X}\bar{X}^\top$, the Gram-power bound of Grishina et al. (2025, Section 3.3) is

$$s_8 = \|\bar{A}^2\|_F^{1/4} = \left(\sum_i x_i^8\right)^{1/8}, \qquad \max_i x_i \le s_8 \le r^{1/8} \max_i x_i, \tag{18}$$

and $s_8 \le 1$ because $\sum_i x_i^2 = 1$. Rescaling does not change the polar factor. The first-iteration products give both the scale and the moments. If $X_0 = \bar{X}/s$, then $A_0 = \bar{A}/s^2$ and $A_0^2 = \bar{A}^2/s^4$, so each iteration-zero moment is a rescaled reduction of $\bar{A}$ or $\bar{A}^2$. The same scale is used for the measurement pass, the spectral grid, and scoring. Figure 9 shows that this bound is nearly tight for every matrix type, unlike the Frobenius norm, and that better normalization alone accounts for only part of the gain from spectral selection.

B.2 Candidates and fitting

Each candidate is a complete PE routine given by a design lower endpoint and a depth. Candidates are checked for positive iteration responses. For an iteration $p(x) = x(a + bx^2 + cx^4)$ on $[0, u]$, positivity reduces to the quadratic $a + bt + ct^2$ on $[0, u^2]$, and the extreme outputs occur at the endpoints or at roots of $a + 3bx^2 + 5cx^4$. Candidates that fail are removed.

The convex programs in Equation (11) use the row scaling $D = \mathrm{diag}(1/d_q)$ with $d_q = \max\{\max_b |\Phi_{qb}|, |\widehat{y}_q|, d_{\min}\}$, which puts measurements of different magnitudes on a common scale. Each matrix is fitted independently. Table 5 lists all settings.

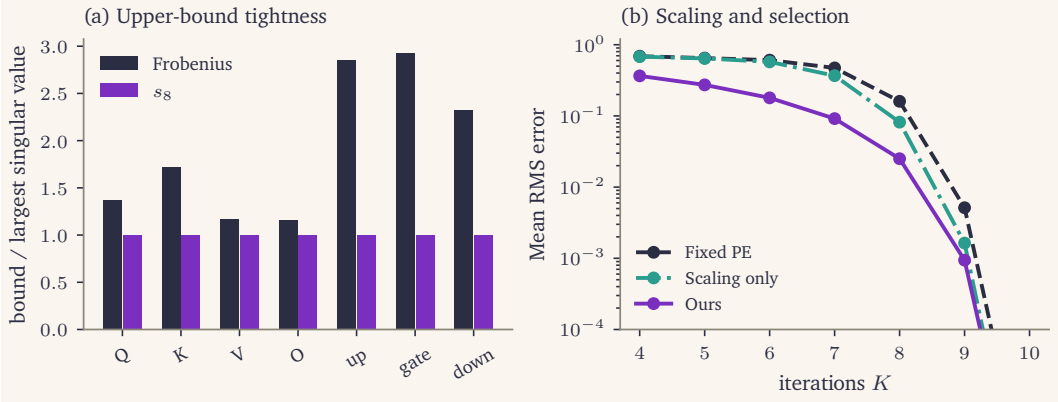

Figure 9: **Normalization explains part of the gain. (a)** Ratio of each upper bound to the largest singular value, by matrix type. **(b)** Fixed PE, fixed PE with the Gram-power scale (scaling only), and spectral estimation.

Ambiguity of the fitted spectrum. Finitely many measurements can admit many spectral measures. Figure 10 constructs two grid measures that match the same measurements to numerical tolerance, the maximum-entropy fit and an extreme point of the feasible set. Their point masses differ substantially, but their cumulative distributions nearly coincide, so spectral averages such as the RMS score are similar under both (Theorem 2). Maximum entropy selects one representative among the feasible measures.

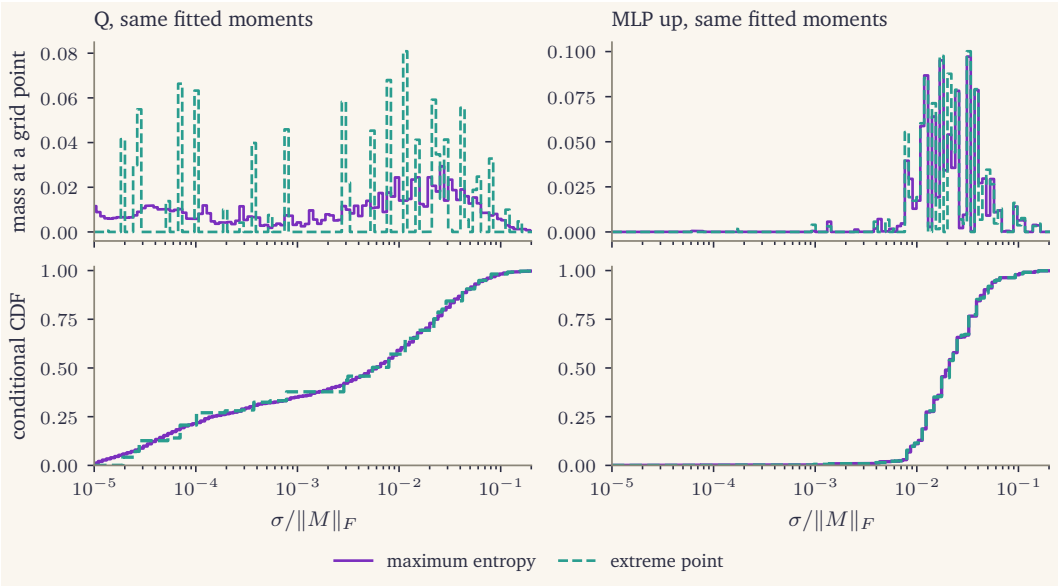

Figure 10: **Different spectral measures can match the same measurements.** Maximum-entropy and extreme-point grid measures for representative attention and MLP spectra. The top row shows the mass on the estimator grid and the bottom row the conditional CDF on the scoring interval.

B.3 Periodic measurement

Periodic recomputation in the background is a standard way to amortize preconditioner updates (Anil et al., 2020). Algorithm 1 applies it here. A measurement pass produces the current update and the measurements, and fitting and selection run in the background while the previous routine stays in use.

Table 5: **Estimator and candidate settings.**

Component	Setting
Scale	$1.005\, s_8$, rounded up to the grid $10^{-j/24}$, $j = 0, \ldots, 15$
Scoring cutoff	$x_\star = 10^{-5}$ in Frobenius-normalized coordinates
Spectral grid	zero, 24 points below the cutoff, 120 in the scoring interval
Measurements	four moments per iteration (0–9) and the final second moment
Entropy tolerance	$\kappa = 3 \times 10^{-4}$
Row-scale floor	$d_{\min} = 10^{-10}$
Solver	CVXPY with Clarabel
Design lower endpoints	49 log-spaced values in $[10^{-5}, 10^{-1}]$
Candidate depths	$K = 4, \ldots, 10$
Remez tolerance / iteration cap	10^{-15} / 300
Endpoint cushion	$\ell \leftarrow \max(\ell, 0.02u)$
Coefficient safety scale	$(1.01, 1.01^3, 1.01^5)$ before the final iteration
Positivity margin	10^{-9}

Algorithm 1 Periodic spectral estimation.

Require: Period T, measurement routine $\mathcal{S}_{\text{meas}}$ of depth K_{meas}, candidate set, selection rule
Use $\mathcal{S}_{\text{meas}}$ for every matrix
for optimizer steps $t = 0, 1, \ldots$ **do**
  Form the current momenta and their normalization
  if a finished fit is available **then**
    Switch to the newly selected routines
  end if
  if a measurement is due and no fit is running **then**
    Run $\mathcal{S}_{\text{meas}}$, record the moments, and use its output for the update
    Start fitting and selection in the background
  else
    Apply the current routines
  end if
end for

If steps between measurements use mean depth $\bar{K}$, the average number of iterations per step is

$$K_{\text{eff}} = \bar{K} + \frac{K_{\text{meas}} - \bar{K}}{T}. \tag{19}$$

C Offline experiments

C.1 Data

We use Muon momentum matrices saved during a FineWeb run of a small GPT model (Table 6). The held-out run (Section C.4) and Figure 3 use the 160M Muon baseline of our pretraining experiments (Section F).

Table 6: **Offline data.**

	Primary run	Held-out run
Dataset	FineWeb	Nemotron-CC v2 High-Quality
Model	12 layers, width 768	24 layers, width 768 (160M Muon baseline)
Checkpoints	every 2,000 steps, 2k–30k	step 1, then every 500 steps to 4,500
Matrices	720 attention, 540 MLP	1,680
Matrix types	Q, K, V, O, MLP up, MLP gate, MLP down	
Muon momentum	0.95	0.9586

C.2 Fitted spectra across layers and training

Figures 11 and 12 compare the fitted and SVD spectra for every matrix type, across layers and across training. The fits follow the SVD spectra closely throughout.

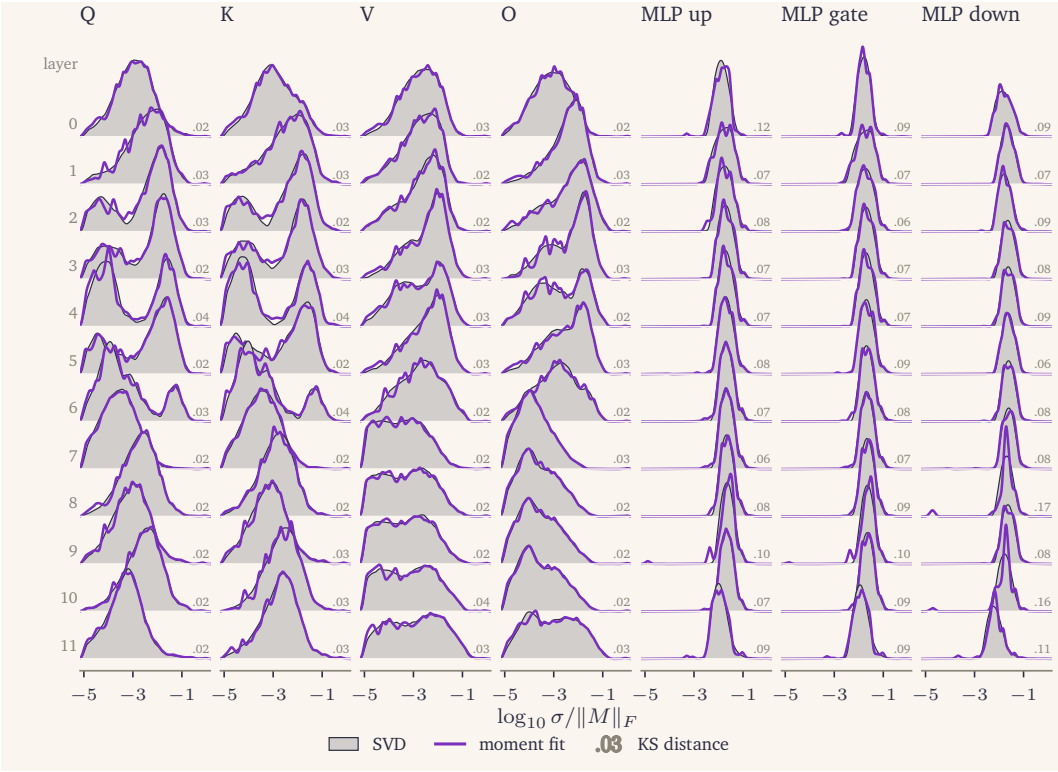

Figure 11: **Fitted and SVD spectra across layers** (step 16k), with the SVD in grey and the moment fit in purple. Both are restricted to $\sigma/\|M\|_F \geq 10^{-5}$ and smoothed in $\log_{10} \sigma$. The numbers give the largest gap between the two CDFs.

C.3 Fixed-depth errors and the choice of score

Table 8 extends Table 1 to all depths and adds the RMS oracle, which selects by exact RMS error. The minimax oracle, used as the oracle throughout the paper, selects by exact worst-case error. On attention matrices spectral estimation nearly matches the RMS oracle, and on MLP matrices it is close at shallow depths. It has lower RMS error than the minimax oracle on attention through $K = 9$ and on MLP through $K = 6$, and higher worst-case error.

We score candidates by RMS error because the fitted spectrum can predict it. RMS error is an average over the spectrum, which Theorem 2 covers. Worst-case error depends on the extreme scored singular values, which finitely many moments do not determine. Table 7 instead applies the minimax rule to the fitted spectrum, on the interval that keeps all but $0.5/r$ of the fitted mass in each tail. Against its own oracle, it has much larger regret than the RMS rule has against the RMS oracle, and it misses about a fifth of the MLP accuracy targets, compared with at most 6% for the RMS rule.

C.4 Held-out run

We apply the method unchanged to the held-out run (Table 9). On attention matrices spectral estimation stays close to the RMS oracle and well below fixed PE, and on this run it also has lower worst-case error than fixed PE. On MLP matrices it stays far below fixed PE, while the minimax oracle is lower in both errors.

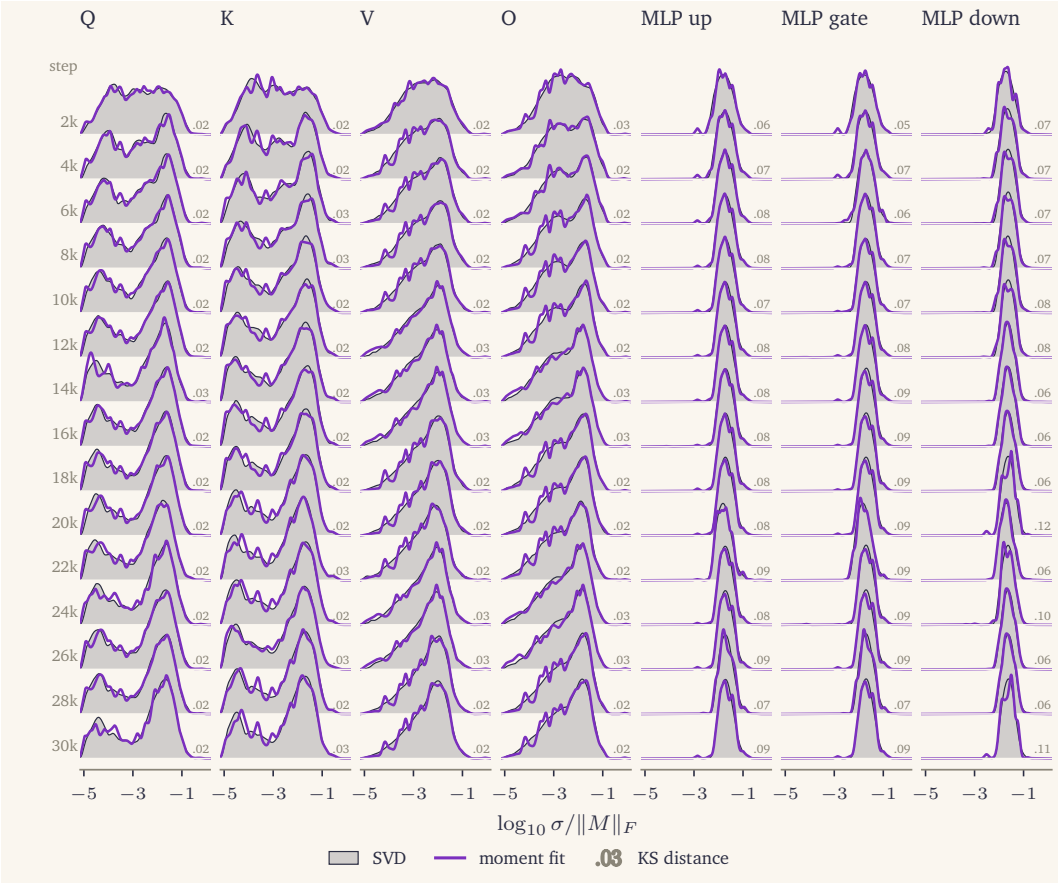

Figure 12: **Fitted and SVD spectra across training** (layer 5). Same conventions as Figure 11.

Table 7: **Scoring rules on the fitted spectrum.** Regret is the mean error of the selected routine minus that of the matching oracle, in the rule's own criterion (worst-case error for minimax, RMS error for ours). Misses are the share of MLP matrices whose true error exceeds τ under the target rule.

	Regret				MLP misses (%)		
	Attention		MLP		τ		
Rule on the fit	$K=5$	$K=8$	$K=5$	$K=8$	10^{-1}	10^{-2}	10^{-3}
Minimax	0.0014	0.0038	0.283	0.0069	20.0	18.7	21.1
RMS (ours)	$\mathbf{3\times10^{-5}}$	**0.00068**	**0.0036**	**0.0010**	**0.0**	**5.7**	**5.4**

C.5 Ablations

Measurements. Figure 13 fits the spectrum from fewer measurements, either from only the first iterations of the measurement pass or with fewer moment orders per iteration. More iterations and more orders both help, mostly on MLP matrices.

Scoring cutoff. Figure 14 repeats the offline comparison with lower scoring cutoffs. A lower cutoff includes more of the small-singular-value tail and makes the problem harder for every method.

Estimator settings. Table 10 changes one setting at a time. Attention results are insensitive to all of them. On MLP matrices at $K = 5$, a small positive entropy tolerance, $\kappa \in [10^{-4}, 3 \times 10^{-4}]$, gives the lowest error, and we use $\kappa = 3 \times 10^{-4}$ for all experiments.

Table 8: **Fixed-depth RMS and worst-case error.** Means over attention and MLP matrices. Oracle is the minimax oracle, and the RMS oracle selects by exact RMS error. All methods use the same candidates and scoring interval.

	RMS error				Worst-case error		
K	Fixed PE	Ours	Oracle	RMS oracle	Fixed PE	Ours	Oracle
Attention							
4	0.729	**0.618**	0.718	**0.618**	0.994	0.995	**0.992**
5	0.667	**0.472**	0.653	**0.472**	0.976	0.982	**0.966**
6	0.604	**0.311**	0.583	**0.311**	0.904	0.934	**0.868**
7	0.465	0.158	0.402	**0.157**	0.665	0.774	**0.582**
8	0.159	0.0431	0.104	**0.0424**	0.222	0.353	**0.161**
9	0.00515	0.00164	0.00229	**0.00144**	0.00753	0.0121	**0.00496**
10	3.86×10^{-7}	1.55×10^{-7}	1.52×10^{-7}	$\mathbf{1.49\times10^{-7}}$	$\mathbf{5.56\times10^{-7}}$	1.15×10^{-6}	1.14×10^{-6}
MLP							
4	0.638	0.0281	0.106	**0.0252**	0.994	0.329	**0.157**
5	0.626	0.00694	0.0179	**0.00330**	0.976	0.0643	**0.0266**
6	0.612	0.00416	0.00449	$\mathbf{7.65\times10^{-4}}$	0.904	0.0207	**0.00647**
7	0.482	0.00352	3.60×10^{-4}	$\mathbf{1.08\times10^{-4}}$	0.665	0.00968	$\mathbf{5.48\times10^{-4}}$
8	0.162	0.00101	1.13×10^{-6}	$\mathbf{7.03\times10^{-7}}$	0.222	0.00153	$\mathbf{1.78\times10^{-6}}$
9	0.00511	1.00×10^{-5}	$\mathbf{2.06\times10^{-12}}$	$\mathbf{2.06\times10^{-12}}$	0.00753	1.55×10^{-5}	$\mathbf{2.53\times10^{-12}}$
10	3.86×10^{-7}	3.99×10^{-10}	1.30×10^{-16}	$\mathbf{2.79\times10^{-18}}$	5.56×10^{-7}	5.16×10^{-10}	$\mathbf{2.26\times10^{-18}}$

Table 9: **Held-out run.** Mean errors over all checkpoints with unchanged settings. Oracle is the minimax oracle.

	RMS error				Worst-case error		
K	Fixed PE	Ours	Oracle	RMS oracle	Fixed PE	Ours	Oracle
Attention							
5	0.640	0.153	0.348	**0.149**	0.976	0.550	**0.516**
8	0.161	0.0123	0.0288	**0.00977**	0.222	0.126	**0.0444**
MLP							
5	0.623	0.0315	0.0247	**0.0214**	0.976	0.0539	**0.0373**
8	0.162	0.00232	4.79×10^{-6}	$\mathbf{1.87\times10^{-6}}$	0.222	0.00339	$\mathbf{7.60\times10^{-6}}$

Table 10: **Estimator settings.** Mean RMS error of the selected routine, changing one setting at a time. RMS-oracle rows use the corresponding candidate set.

Setting	Value	Attn K=5	Attn K=8	MLP K=5	MLP K=8
Default	Table 5	0.472	0.0431	0.00694	0.00101
Entropy tolerance κ	0 (best fit only)	0.472	0.0437	0.00729	0.000622
	10^{-4}	0.472	0.0430	0.00684	0.000777
	10^{-3}	0.472	0.0434	0.00862	0.00140
	3×10^{-3}	0.472	0.0439	0.0131	0.00216
Grid, scoring interval	60	0.472	0.0435	0.0199	0.00223
	240	0.472	0.0437	0.00870	0.00161
Grid, below cutoff	12	0.472	0.0434	0.00696	0.00112
	48	0.472	0.0430	0.00692	0.000978
Endpoints per decade	6	0.473	0.0445	0.00672	0.00105
	24	0.472	0.0429	0.00704	0.000986
RMS oracle	6 per decade	0.473	0.0437	0.00346	1.17×10^{-6}
	12 per decade	0.472	0.0424	0.00330	7.03×10^{-7}
	24 per decade	0.472	0.0422	0.00326	6.59×10^{-7}

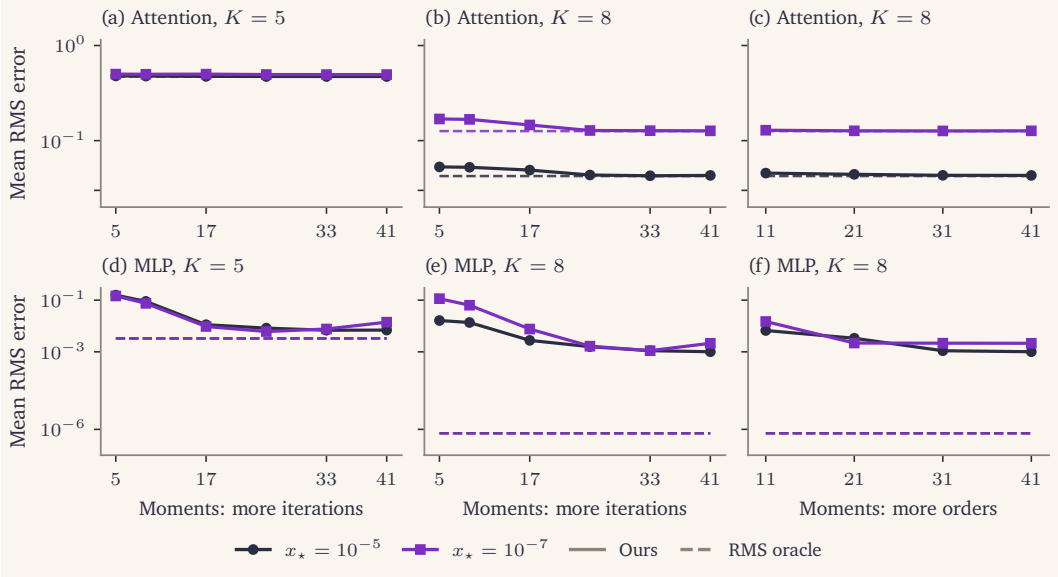

Figure 13: **Fewer measurements.** The left and center columns use measurements from only the first iterations of the measurement pass, and the right column fewer moment orders at every iteration. Colors give $x_\star$, the scoring cutoff and smallest design endpoint. Solid lines show spectral estimation and dashed lines the RMS oracle.

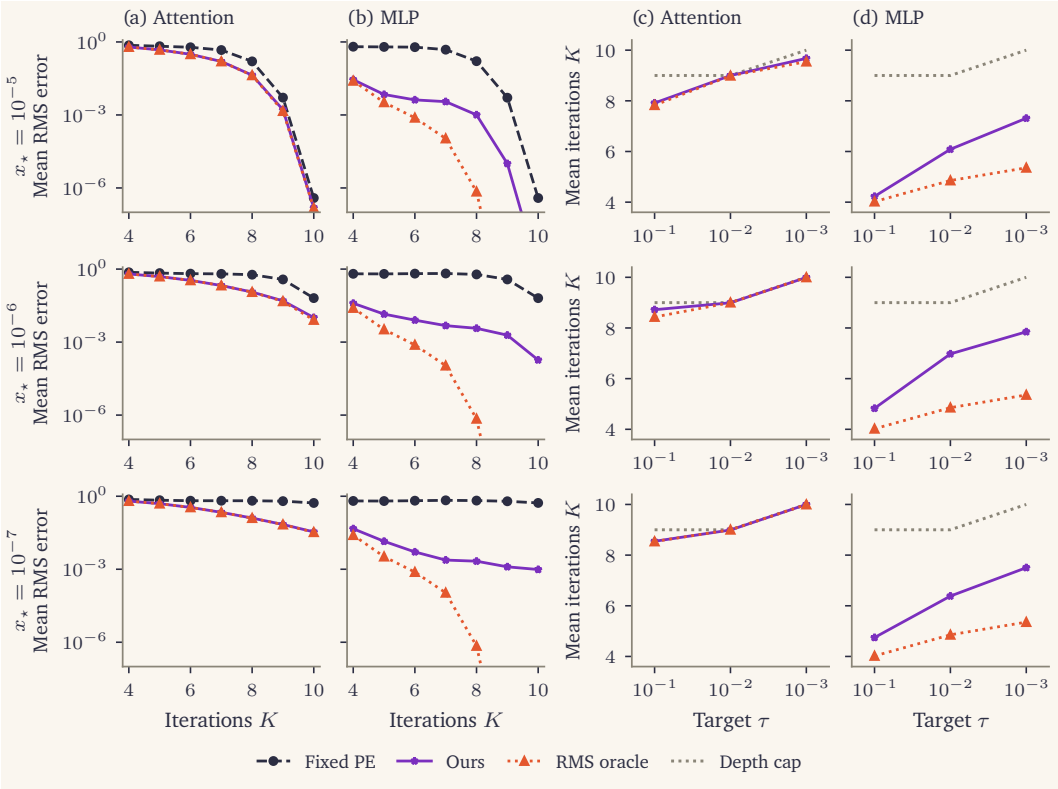

Figure 14: **Lower scoring cutoffs.** Each row repeats Figure 7 with the RMS oracle at the cutoff $x_\star$ shown on the left. Panels **(a–b)** show RMS error at fixed depth and **(c–d)** the mean depth at target τ, capped at the largest allowed depth (dotted). At the lower cutoffs, almost no attention matrix meets $\tau \leq 10^{-2}$ within the cap, for either method.

C.6 TARGET RULE

The target rule picks the shortest candidate whose predicted RMS error is at most τ. Table 11 adds a safety margin m by requiring the prediction to be at most τ/m. On attention matrices the predictions are accurate, and a small margin removes every miss at little extra depth. On MLP matrices a small fraction of matrices miss the target even with a large margin. These are MLP-down matrices with a few isolated singular values far below the rest of the spectrum. Each such value adds a large error, and the moments do not detect it.

Table 11: **Target rule with a safety margin m.** A candidate is accepted if its predicted RMS error is at most τ/m. Misses are the fraction of matrices whose true RMS error exceeds τ.

		Mean depth		Misses (%)		
τ	m	Attn	MLP	Attn	MLP	Max error
10^{-1}	1	7.91	4.23	1.9	0.0	0.124
	1.5	8.19	4.44	0.0	0.0	0.0780
	10	9.00	6.09	0.0	0.0	0.0567
	RMS oracle	7.83	4.02	0.0	0.0	0.0995
10^{-2}	1	9.00	6.09	0.0	5.7	0.0567
	1.5	9.00	6.27	0.0	4.4	0.0567
	10	9.00	7.31	0.0	2.4	0.0389
	RMS oracle	8.99	4.85	0.0	0.0	0.00991
10^{-3}	1	9.68	7.31	0.1	5.4	0.0389
	1.5	9.77	7.52	0.0	5.0	0.0384
	10	9.91	7.97	0.0	1.9	0.0326
	RMS oracle	9.55	5.36	0.0	0.0	0.000998

C.7 REUSE AND PER-TYPE SELECTION

Table 12 evaluates each saved matrix with a design endpoint chosen at an earlier checkpoint, while recomputing its normalization. Reusing older choices changes the error very little, and every variant stays far below fixed PE.

Table 13 compares per-matrix selection with one fixed endpoint per matrix type, chosen with exact SVD errors on early checkpoints, in the spirit of AMO (Zhuang et al., 2026). The per-type choice needs exact SVDs from the same run and depends on which checkpoints are used to choose it. Spectral estimation needs neither, has lower error at both depths, and gains most at larger depth, where layers of the same type differ. Grouping would also help in mixture-of-experts models with many expert matrices. Because the moments are averages over the spectrum, averaging the measurements of a group gives the moments of its pooled spectrum, so one fit can serve the whole group.

Table 12: **Reusing earlier choices.** Mean RMS error on checkpoints from step 4k. Columns give the checkpoint whose choice is used, namely the current one, the previous one, the first one, or one updated only at every fourth checkpoint. Oracle is the minimax oracle.

K	Current	Previous	First	Every fourth	RMS oracle	Oracle	Fixed PE
5	0.275	0.275	0.278	0.275	**0.273**	0.381	0.650
8	0.0254	0.0257	0.0261	0.0256	**0.0246**	0.0601	0.160

Table 13: **Per-type versus per-matrix selection.** Mean RMS error on the later checkpoints (16k–30k), with the per-type endpoints chosen on the earlier ones. Oracle is the minimax oracle.

K	Per type	Ours	RMS oracle	Oracle
5	0.286	0.283	**0.282**	0.383
8	0.0290	0.0273	**0.0266**	0.0639

D COST

For $M \in \mathbb{R}^{r\times n}$ with $r \le n$, one quintic iteration costs $2r^2n + r^3$ multiply–adds. The reductions of A_j and A_j^2 add about $3r^2$, a fraction $3/(2n + r)$ of an iteration, and use only matrices the iteration already holds. The fit and the scoring work on a fixed number of measurements and a fixed grid, so their cost does not depend on the matrix size, and they run on the CPU. Table 14 compares these costs with those of exact spectra.

Figure 15 and Table 15 report GPU times on one H200 for random square and wide matrices and for matrix shapes of recent models trained with Muon. The reductions take between 1 and 3 ms at every tested size, and the fit takes 4 ms of CPU time per matrix. The Newton–Schulz pass and the exact spectrum grow with the matrix.

The reductions and the fit run once every T steps, and the K_{meas}-iteration measurement pass replaces one K-iteration pass (Equation (19)). For a model with N hidden matrices, this adds Nt_{red}/T of GPU time, $(K_{\text{meas}} - K)/T$ iterations, and Nt_{fit}/T of CPU time per step, where t_{red} and t_{fit} are the per-matrix times above. For our 1B Muon model ($N = 168$, $K = 5$, $T = 200$, about 40 s per step), this is under 1 ms of reductions, 0.025 iterations, and 4 ms of CPU time per step.

Table 14: **Added cost of each spectrum estimator** for $M \in \mathbb{R}^{r\times n}$, $r \le n$. One Newton–Schulz iteration costs $2r^2n + r^3$.

Method	Extra products	Work that grows with M	Fixed work
Ours	none	$\approx 3r^2$ per iteration	fit, scoring
SVD (singular values)	–	$O(r^2n)$	–
Gram eigenvalues	$MM^\top$	$O(r^2n + r^3)$	–

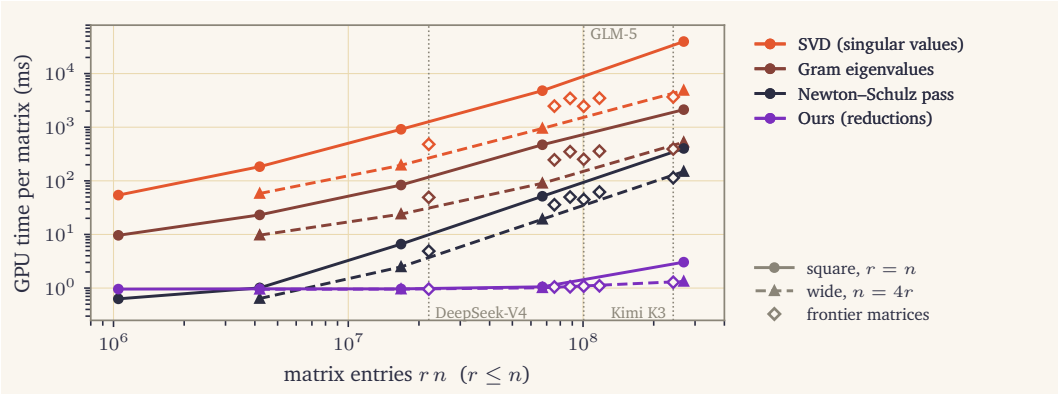

Figure 15: **GPU time per matrix versus matrix size** (one H200, median). Ours denotes the compiled reductions. Circles mark square matrices, triangles wide matrices, and diamonds the matrices of Table 15. The Newton–Schulz pass runs in bf16, and the SVD and the Gram eigenvalues in fp32. Matrices are random Gaussian.

Table 15: **Matrix shapes of recent models trained with Muon** (DeepSeek-AI, 2026; GLM-5 Team, 2026; Kimi Team, 2026) and GPU time per matrix in milliseconds (one H200, median). The NS pass runs ten bf16 iterations, ours is the compiled reductions, and the last two columns compute the eigenvalues of $MM^\top$ and the singular values. The DeepSeek-V4 attention output is the second matrix of its grouped output projection.

Model	Matrix	Shape	NS pass	Ours	Gram eig.	SVD
DeepSeek-V4	routed expert	3072×7168	4.9	**0.96**	49	480
	attention output	7168×16384	62	**1.1**	358	3481
GLM-5	attention output	6144×16384	45	**1.1**	252	2461
	dense FFN	6144×12288	36	**1.1**	246	2473
Kimi K3	attention output	7168×12288	50	**1.1**	349	3446
	dense FFN	7168×33792	114	**1.3**	392	3667

E INVERSE SQUARE ROOTS

For a positive-definite matrix S scaled to have spectrum in $(0, 1]$, consider the coupled iteration (Higham, 1997; Anil et al., 2020)

$$Y_0 = S, \quad Z_0 = I, \quad H_j = Z_j Y_j, \quad Q_j = a_j I + b_j H_j, \quad Y_{j+1} = Y_j Q_j, \quad Z_{j+1} = Q_j Z_j, \tag{20}$$

which computes $S^{-1/2}$ and is used in dense preconditioners, including the scaled CANS construction of Online KL Shampoo (Grishina et al., 2025; Zhang et al., 2026a).

Proposition 2 (Scalar recursion of the coupled iteration)**.** *For an eigenvalue $\lambda > 0$ of S, let ζ_j be the corresponding eigenvalue of Z_j and $z_j = \sqrt{\lambda}\,\zeta_j$. In exact arithmetic,*

$$z_0 = \sqrt{\lambda}, \qquad z_{j+1} = a_j z_j + b_j z_j^3, \tag{21}$$

and the eigenvalue of H_j is z_j^2. Thus z_j follows the same scalar recursion as in the polar case, and $z_j - 1$ is the relative inverse-root error.

Proof. All iterates are polynomials in S and commute, and induction gives $Y_j = SZ_j$. The eigenvalue of $H_j = Z_j Y_j$ is therefore $\lambda\zeta_j^2 = z_j^2$, so $\zeta_{j+1} = (a_j + b_j z_j^2)\zeta_j$, and multiplying by $\sqrt{\lambda}$ gives Equation (21). □

With $\gamma = \sqrt{\lambda}$, reductions of matrices the iteration already forms measure the functions in Table 16, so the method of the polar case carries over. For $0 < \ell < 1$, the odd cubic minimizing $\max_{x\in[\ell,1]} |ax + bx^3 - 1|$ is

$$h = 1 + \ell + \ell^2, \qquad \bar{x} = \sqrt{h/3}, \qquad b = \frac{2}{1 - h - (2h/3)\bar{x}}, \qquad a = -bh, \tag{22}$$

where $\bar{x}$ is the interior extremum of the error. This is the cubic iteration behind scaled Newton–Schulz and CANS (Chen & Chow, 2014; Grishina et al., 2025), and it defines our cubic candidates.

Table 16: **Measurements from the coupled iteration.** Each function is averaged over the distribution of γ.

Reduction	Function
$\operatorname{tr} H_j/n$, $\|H_j\|_F^2/n$	z_j^2, z_j^4
$\operatorname{tr} Y_j/n$, $\|Y_j\|_F^2/n$	γz_j, $\gamma^2 z_j^2$
$\langle Y_j, H_j\rangle_F/n$, $\|Y_j H_j\|_F^2/n$	γz_j^3, $\gamma^2 z_j^6$
$\operatorname{tr} Z_j/n$, $\|Z_j\|_F^2/n$	z_j/γ, z_j^2/γ^2
$\langle Z_j, H_j\rangle_F/n$, $\|H_j Z_j\|_F^2/n$	z_j^3/γ, z_j^6/γ^2

We test the extension on regularized momentum Gram matrices $S = \bar{M}\bar{M}^\top + \varepsilon I$ from the primary run (Table 17). Figure 16 shows that selecting a cubic routine from the fitted spectrum improves on the fixed cubic routine at every depth, with the largest gains on the more concentrated spectra.

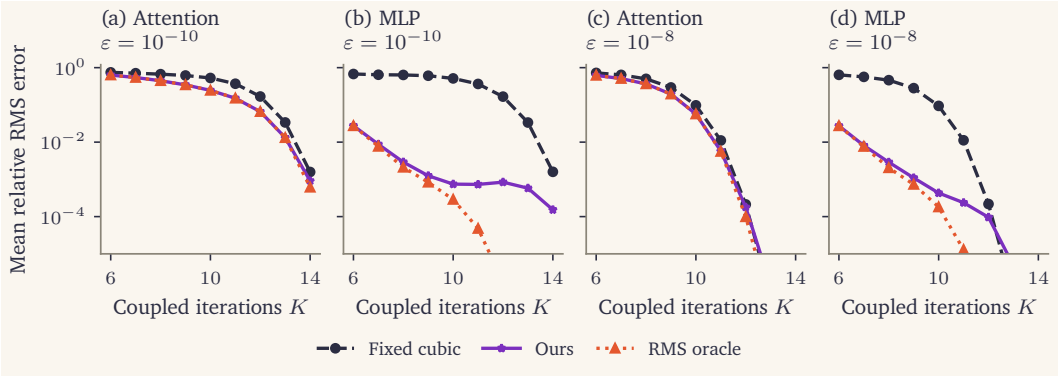

Figure 16: **Inverse square roots.** Mean relative RMS error on regularized momentum Gram matrices for the fixed cubic routine, spectral estimation, and the RMS oracle over the same candidates.

Table 17: **Inverse-root test settings.**

Component	Setting
Test matrices	$\bar{M}\bar{M}^\top + \varepsilon I, \varepsilon \in \{10^{-10}, 10^{-8}\}$
Endpoints	49, log-spaced in $[10^{-5}, 10^{-1}]$
Candidate depths	$K = 4, \ldots, 18$
Grid	one low point, 150 log-spaced
Entropy tolerance	$\kappa = 3 \times 10^{-4}$
Measurement depth	15 ($\varepsilon = 10^{-10}$) or 13 ($\varepsilon = 10^{-8}$)

F Pretraining Details

F.1 Setup

The model and optimizer follow Zhang et al. (2026a). Table 18 lists the model, data, and training settings, and Table 19 the hidden-matrix optimizer settings, which are shared across model sizes. The hidden matrices of each layer (attention Q, K, V, O and MLP up, gate, down) use Muon or OKLS, while embeddings, the output head, normalization gains, and attention-gate projections use AdamW (Loshchilov & Hutter, 2019). The data order is the same for every run. Each evaluation reads the next window of a held-out shard, the same window for every run.

A hidden weight of shape $m \times n$ is initialized as $W_{ij} \sim \mathcal{N}\big(0, [\sigma_* \sqrt{m/n}/(\sqrt{m}+\sqrt{n})]^2\big)$, embeddings as $\mathcal{N}(0,1)$, and the untied output head at zero, so hidden gradients vanish at the first step. The learning rate warms up linearly, stays constant, and decays linearly to zero over the final fraction p_{dec} of training.

For Muon, with gradient $G_t \in \mathbb{R}^{m \times n}$,

$$\mathcal{B}_t = \beta \mathcal{B}_{t-1} + G_t, \qquad N_t = \beta \mathcal{B}_t + G_t,$$

$$U_t = \sqrt{m/n}\, \frac{\sqrt{\min(m,n)}}{\|\operatorname{polar}(N_t)\|_F} \operatorname{polar}(N_t), \qquad \theta_{t+1} = (1 - \lambda_{\mathrm{wd}} \eta_t^2/\eta_{\mathrm{peak}})\theta_t - \eta_t U_t,$$

where $\operatorname{polar}(N_t)$ is computed by the routine under study. Because the output is rescaled to a fixed norm, routines differ only in the direction of the update.

For OKLS, with the previous inverse-root factors $R_{\mathrm{L}}, R_{\mathrm{R}}$,

$$\mathcal{B}_t = \beta_1 \mathcal{B}_{t-1} + (1-\beta_1) G_t, \qquad N_t = (1-\beta_1) G_t + \beta_1 \mathcal{B}_t,$$

$$S_{\mathrm{L}}^+ = \beta_2 S_{\mathrm{L}} + \tfrac{1-\beta_2}{n} (G_t R_{\mathrm{R}})(G_t R_{\mathrm{R}})^\top + \epsilon I, \quad S_{\mathrm{R}}^+ = \beta_2 S_{\mathrm{R}} + \tfrac{1-\beta_2}{m} (R_{\mathrm{L}} G_t)^\top (R_{\mathrm{L}} G_t) + \epsilon I.$$

The factors are symmetrized, their inverse roots $R_{\mathrm{L}}^+, R_{\mathrm{R}}^+$ are recomputed every step, and

$$\theta_{t+1} = (1 - \lambda_{\mathrm{wd}} \eta_t^2/\eta_{\mathrm{peak}})\theta_t - \eta_t \chi R_{\mathrm{L}}^+ N_t R_{\mathrm{R}}^+, \qquad \chi = \frac{\sqrt{m/n}}{\sqrt{m}+\sqrt{n}} \left[\frac{1-\beta_1}{1+\beta_1}(1 + 2\beta_1 - 2\beta_1^3)\right]^{-1/2}.$$

The factors start from scaled $GG^\top$ and $G^\top G$ of the first nonzero gradient. Compared with the OKLS setup, we hold the tokenizer, head dimension, MLP expansion ratio, attention gate, initialization, and AdamW parameter group fixed across model sizes, train for slightly fewer steps, and use Polar Express instead of CANS for Muon.

Table 18: **Model and training settings.**

Setting	160M	300M	1B
Layers / width	24 / 768	24 / 1024	24 / 2048
Query heads / KV heads	12 / 3	16 / 4	32 / 8
Head dimension / MLP width	64 / 2304	64 / 3072	64 / 6144
Non-embedding / total params	163.1M / 262.9M	289.9M / 423.0M	1159.3M / 1425.6M
Attention / MLP	grouped-query attention with a per-head sigmoid gate / SwiGLU		
Normalization / positions	pre-RMSNorm and final RMSNorm ($\epsilon = 10^{-6}$) / RoPE ($\theta = 10^4$)		
Data	Nemotron-CC v2 High-Quality (NVIDIA, 2025a;b)		
Tokenizer / vocabulary	`tiiuae/falcon-7b` / 65,024		
Context / sequences per step	4096 / 1024 (4,194,304 tokens)		
Steps / warmup	4769 (20B tokens) / 250		
Validation	4,194,304 held-out tokens every 250 steps		
AdamW (non-hidden params)	lr 3×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.95)$, $\epsilon = 10^{-8}$, no weight decay		
Precision	bf16 autocast, fp32 weights and optimizer state		
Clipping / dropout / z-loss	none		

Table 19: **Hidden-matrix optimizer settings.** The decay coefficient λ_{wd} enters as $\theta \leftarrow (1 - \lambda_{\mathrm{wd}}\eta_t^2/\eta_{\mathrm{peak}})\theta$.

Setting	Muon	OKLS
Peak learning rate	0.01202	0.09434
Momentum β or β_1	0.9586	0.9684
Factor decay β_2	–	0.9482
Factor diagonal shift ϵ	–	10^{-9}
Weight decay λ_{wd}	1.008×10^{-4}	0.0303
Initialization scale σ_*	0.1877	0.07539
Decay fraction p_{dec}	0.7677	0.7319
Matrix-function precision	bf16	fp32

F.2 Matrix-function routines

Muon. The routines use the normalization, candidates, and fitting of Sections B.1 and B.2 with the settings in Tables 5 and 20. The fixed baseline is $\mathrm{PE}[10^{-5}, 1]$ at the run's depth. The oracle recomputes the minimax choice from an exact SVD every 500 steps and uses the fixed baseline until its first choice. Spectral estimation also starts from the fixed baseline. Every T steps it runs the measurement pass, which also produces that step's update, fits the moments on the CPU, and uses the selected routine until the next measurement. If a fit fails, the previous routine is kept. Our runs also record exact spectra at each measurement to measure selection quality (Section F.4), and these records never enter selection.

OKLS. The inverse roots use the coupled iteration of Section E, scaled by an upper bound on the largest eigenvalue, in fp32 because the coupled iteration drifts in bf16. The measurements include all traces, squared norms, and inner products of H_j, Y_j, Z_j and of the products the iteration already forms (Table 16), and those involving Z_j add information about small eigenvalues. The fixed baseline uses the cubic routine designed for eigenvalues above $10^{-7}\lambda_{\max}$. An exact eigendecomposition every 500 steps only updates $\lambda_{\max}$, which, like the Frobenius norm in PE, bounds the scaled spectrum by one, while the relative lower endpoint stays at 10^{-7}.

Comparisons. We use $T = 200$ throughout, and Section F.3 also reports $T = 500$. Figure 8 averages the training-loss gaps over short windows.

Table 20: **Pretraining routine settings.**

	Muon	OKLS
Scoring cutoff	10^{-5}	10^{-7} (relative)
Candidates	49 PE endpoints in $[10^{-5}, 10^{-1}]$ at the run's depth K	cubic, 12 endpoints per decade in $[10^{-7}, 1]$, at the run's depth K
Measurement pass	10 iterations, 41 measurements	10 iterations, 94 measurements
Measurement period T	200 (500 also at 160M and 300M)	200 (500 also at 160M and 300M)
Fixed baseline	PE$[10^{-5}, 1]$	cubic routine for eigenvalues above $10^{-7}\lambda_{\max}$
Oracle	minimax choice from the exact spectrum every 500 steps	
Solver	Clarabel in float64 on CPU, tolerance 10^{-8}, one retry at 10^{-11}	

F.3 Additional pretraining results

Measurement period. Table 21 compares $T = 200$, used throughout, with $T = 500$, which we also ran at 160M and 300M. Both periods improve on the fixed baseline in every configuration, and neither is better in all of them.

Table 21: **Measurement period.** Final validation loss of our method with $T = 200$ and $T = 500$.

Optimizer	Size	K	Fixed	$T = 200$	$T = 500$
Muon	160M	5	2.761	2.751	**2.748**
Muon	300M	5	2.662	2.655	**2.651**
OKLS	160M	5	2.687	2.673	**2.671**
OKLS	300M	5	2.597	**2.590**	2.591

Practitioner endpoint. The default Polar Express routine (Amsel et al., 2026a) is designed for $[10^{-3}, 1]$, a narrower interval than the $[10^{-5}, 1]$ of our fixed baseline. Table 22 compares both fixed routines with our method at $K = 5$. The less conservative endpoint improves slightly on the fixed baseline at 160M and matches it at 300M, and per-matrix selection improves on both.

Table 22: **Practitioner endpoint.** Muon, $K = 5$, final validation loss.

Size	PE$[10^{-5}, 1]$ (baseline)	PE$[10^{-3}, 1]$ (default)	Ours
160M	2.761	2.765	**2.751**
300M	2.662	2.669	**2.655**

F.4 Selection quality during training

At every measurement we also recorded exact spectra and compared the chosen routine with the RMS oracle on the same spectra (Table 23). The error of spectral estimation is close to the oracle's and far below the fixed baseline's. Figure 17 breaks this down by matrix type. On attention Q and O, the minimax rule picks much smaller endpoints than the RMS rule, because it protects the worst direction. On the other matrix types spectral estimation picks smaller, more conservative endpoints than the RMS oracle.

Table 24 repeats the selection on saved training matrices and varies two things separately. Across columns, measured moments, in bf16 for Muon and fp32 for OKLS, give nearly the same selection quality as exact ones. Across rows, the OKLS reductions involving Y_j and Z_j matter, since traces and norms of H_j alone give clearly worse choices.

Table 23: **Selection quality during training.** Agreement is the share of measurements where spectral estimation picks the RMS oracle's endpoint, distance the median gap between the two endpoints in decades, regret our RMS error minus the RMS oracle's, and N the number of matrix or factor measurements. 160M runs.

				Endpoint		RMS error			
Optimizer	K	T	N	Agreement (%)	Distance	Fixed	RMS oracle	Ours	Regret
Muon	4	200	3,864	58.6	0.00	0.6681	0.2226	0.2254	0.0028
Muon	4	500	1,512	61.1	0.00	0.6603	0.2194	0.2219	0.0025
Muon	5	200	3,864	27.3	0.25	0.6359	0.0967	0.1066	0.0099
Muon	5	500	1,512	28.2	0.25	0.6322	0.0724	0.0828	0.0104
OKLS	5	200	4,704	77.8	0.00	0.6267	0.2390	0.2394	0.0004
OKLS	5	500	2,016	73.8	0.00	0.6149	0.2226	0.2232	0.0006

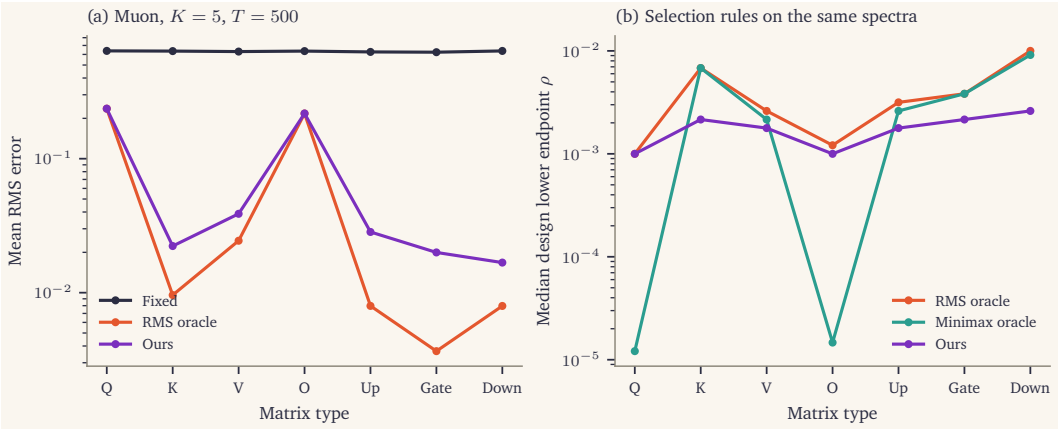

Figure 17: **Selection by matrix type** (Muon 160M, $K = 5$, $T = 500$). **(a)** Mean RMS error on the recorded exact spectra. **(b)** Median design endpoint chosen by spectral estimation and by both oracles on the same spectra.

Table 24: **Moment precision and measurement set.** Mean regret against the RMS oracle on saved training matrices. Columns compare moments computed exactly with moments measured during the pass (bf16 for Muon, fp32 for OKLS), and rows the reductions used. The exact columns cover 504 Muon matrices and 1008 OKLS factors, and the measured ones 504 and 56.

Optimizer	Reductions	Count	Exact	Measured
Muon	all	41	0.0078	0.0090
OKLS	all	94	0.0004	0.0008
OKLS	H_j only	20	0.0019	0.0044

G Related work

Polynomial iterations for matrix functions. Newton–Schulz iterations compute the polar factor using only matrix products (Björck & Bowie, 1971; Kovarik, 1970; Higham, 2008), and coupled versions compute inverse square roots (Higham, 1997). Later work improves the polynomials. Scaled Newton–Schulz speeds up convergence (Chen & Chow, 2014), Polar Express designs each iteration by minimax approximation on a spectral interval (Amsel et al., 2026a), and CANS uses Chebyshev-type designs (Grishina et al., 2025). These routines need the spectral interval in advance and typically use the same one for every matrix. Our method keeps these polynomial families and chooses the interval and depth from the spectrum of the matrices being processed.

Other work makes each orthogonalization cheaper. Gram Newton–Schulz iterates on the smaller Gram matrix with symmetric products (Zhang et al., 2026b), Dion and Dion3 reduce communication

in distributed training (Ahn et al., 2025; Amsel et al., 2026b), MuonBP orthogonalizes blocks between full steps (Khaled et al., 2026), and CacheMuon reuses work from earlier steps (Dev et al., 2026). These methods lower the cost of an iteration or of a step, whereas spectral estimation chooses which iterations to run and can be combined with them.

Adapting the iteration to the matrix. ROOT tailors coefficients to the matrix dimensions (He et al., 2025). AMO observes spectra early in training and then fixes one routine per matrix type (Zhuang et al., 2026), and Section C.7 compares per-matrix selection with this kind of per-type choice. PRISM fits the polynomial to the current spectrum at every iteration using randomized sketches (Yang et al., 2026). Unlike these works, we read the spectral information from products that Newton–Schulz already forms, without random probes, and use it to choose a complete routine for later steps.

Inexact orthogonalization. Analyses of Muon with inexact orthogonalization relate the approximation error to convergence and to the choice of learning rate and momentum (Shulgin et al., 2026; Kim & Oh, 2026; Choudhury et al., 2026; Qian & Richtárik, 2026; Do et al., 2026). Spectral estimation reduces this error at a fixed number of iterations.

Spectral density estimation from moments. Estimating a spectral density from traces of matrix polynomials is standard in numerical linear algebra. The kernel polynomial method (Weiße et al., 2006), stochastic Lanczos quadrature (Ubaru et al., 2017; Chen et al., 2021), and related methods (Lin et al., 2016) estimate these traces with random probe vectors, and maximum-entropy fits turn a few moments into a density (Mead & Papanicolaou, 1984; Silver & Röder, 1997). Fan et al. (2020) use such estimates to adapt polynomial approximations to the spectrum, which is close in spirit to our selection step. Finitely many moments do not identify a distribution (Krein & Nudel'man, 1977), but they determine averages of functions that the measured polynomials approximate well (Braverman et al., 2022), and our prediction theorem applies this to the moments that Newton–Schulz produces. Here the moments are traces of matrices the iteration already forms, so they need neither probe vectors nor extra matrix products.