THE ROW NORMALIZATION PUZZLE IN MUON

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper examines how row-wise renormalization affects Muon, focusing on the gap between worst-case guarantees and practical performance in NorMuon [54]. Despite its growing adoption and promising performance in large language model (LLM) pretraining, NorMuon's worst-case guarantees remain poorly understood. One fundamental question is: *does row normalization yield provable convergence gains, potentially through its interaction with approximated polar computation and exponential moving-average momentum?* Our results show that row normalization introduces a dimension-dependent factor in the worst-case iteration complexity under the operator-norm geometry, which persists even with exact polar computation and any fixed momentum parameters. Indeed, we establish an *algorithm-dependent* lower bound and a matching upper bound in deterministic settings, and extend our upper bound analysis to stochastic settings. Our upper bound covers approximated polar computations. Experiments show that NorMuon is slower than Muon on synthetic problems inspired by our worst-case construction, yet outperforms Muon in LLM pretraining. These findings sharpen the puzzle of why row normalization helps in practice and complement the recent findings in Dewulf et al. [22].

1 INTRODUCTION

Large-scale neural network training has driven major advances in vision and language [50, 37, 87], with large language model (LLM) pretraining placing increasing demands on optimization efficiency [85]. Progress depends not only on data and computational resources, but also on optimizers that exploit the structure of neural network parameters. Among recent methods, Muon [42] has attracted attention for its training efficiency and scalability [56]. Rather than applying coordinate-wise rescaling, Muon updates weight matrices using the polar factor of gradient momentum. Together with related methods such as Scion [68], it reflects a shift toward optimizer design based on input-output matrix geometry. More recently, NorMuon [54] introduces row-wise second-moment normalization to balance update magnitudes across neurons and reports improved performance. Its practical relevance extends beyond optimizer benchmarks: NorMuon was used to train Datadog's Toto 2.0 forecasting models [44], while similar normalization has been incorporated into the nanochat LLM training [43]. NVIDIA's Megatron Core also supports NorMuon-style updates as an optimizer option [66]. These empirical gains motivate a closer examination of the convergence consequences of row normalization.

This design philosophy has a long history in optimization. Indeed, the simplex method exploits the polyhedral geometry of linear optimization [19], while gradient descent applies carefully designed step sizes in nonlinear optimization [69]. Row normalization follows a related principle by respecting the neuron-wise organization of matrix parameters. Yet, geometric insights and practical success do not necessarily imply favorable worst-case guarantees. The simplex method is effective in practice, but needs exponentially many iterations in the worst case [47]. A related tension appears in gradient-based optimization. Step-size scheduling improves practical neural network training, from cosine annealing with warm restarts [58] to warmup-stable-decay schedules for LLM pretraining [90]. In theory, step-size scheduling alone can accelerate gradient descent beyond its classical $O(T^{-1})$ rate [2, 3]. However, recent work shows that optimally chosen schedules cannot attain the algorithm-independent lower bound of $\Omega(T^{-2})$ [92]. We continue this line of thoughts for matrix optimizers, investigating whether the practical benefits of row normalization are reflected in its worst-case guarantees.

The benefits of row normalization have been systematically studied in the recent work [22], which identified a mechanism behind the imbalance in Muon's neuron-wise updates: neurons with low row leverage can receive persistently small updates and will become effectively inactive. NorMuon

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.

mitigates this issue by using exponential moving averages of squared row norms to rebalance update magnitudes across neurons. Their findings provide empirical evidence and alternative explanations for the practical value of balanced neuron updates, motivating a rigorous analysis of the worst-case convergence consequences of row normalization.

In this paper, we investigate row normalization through gradient-update alignment in operator-norm geometry. Our key construction uses a rank-one gradient with unequal row magnitudes: normalizing its polar factor balances the rows but weakens alignment with the gradient. We embed this mechanism in a smooth objective whose gradient remains constant along the iterates, isolating a directional distortion that persists under fixed momentum parameters and deterministic adaptive step sizes. A complementary descent analysis controls the same alignment together with momentum error and polar approximation. Importantly, balanced neuron updates alone do not guarantee faster convergence. Our experimental results illustrate this distinction: NorMuon is slower than Muon on synthetic problems, yet outperforms it in LLM pretraining. Our findings sharpen the row normalization puzzle and further call for a deeper understanding of which key features of practical training make normalization beneficial despite its worst-case cost.

Contributions. We focus on one fundamental question: does row normalization yield provable convergence gains, potentially through its interaction with approximated polar computation and exponential moving-average momentum? Our contributions can be summarized as follows:

1. We establish an algorithm-dependent lower bound of $\Omega(mL\epsilon^{-2})$ for NorMuon on smooth optimization over $\mathbb{R}^{m\times n}$, compared with Muon's $O(L\epsilon^{-2})$ rate in operator-norm geometry, where $L > 0$ is the smoothness parameter. We prove an upper bound of $O(mL\epsilon^{-2})$ in the deterministic setting and that of $O(mL\epsilon^{-2} + m^2L\sigma^2\epsilon^{-4})$ in the stochastic setting, where σ^2 is noise variance in nuclear norm. Our analysis accommodates polar approximation.
2. We conduct the experiments on synthetic problems and real LLM pretraining. The contrasting results separate the theoretical cost of row normalization from its practical benefits, complementing the findings of Dewulf et al. [22] and showing that improved neuron balance alone does not imply uniformly faster optimization.

Related works. Our work is mostly related to the literature on structured matrix-aware optimization and algorithm-dependent convergence analysis. Due to space limitations, we defer our comments on other relevant topics to Appendix A. Muon exploits matrix structure through polar-factor updates of gradient momentum, connecting spectral descent to efficient LLM pretraining [42, 7, 56]. However, controlling singular values does not generally ensure balanced update magnitudes across neurons. To address this issue, NorMuon introduces row-wise second-moment normalization [54], while Muon+ applies stateless row/column normalization after orthogonalization [99]. Their ablation studies provide evidence of lower validation loss and highlight the importance of normalization direction. MuonEq instead normalizes momentum before orthogonalization, reporting lower validation loss and improved robustness [14]. Dion3 incorporates row normalization into a variant with selective orthogonalization [5], while Aurora addresses low-leverage neurons in tall matrices, reporting reduced neuron death and lower validation loss [22]. These empirical findings motivate our analysis of the worst-case convergence cost of row normalization in NorMuon.

Algorithm-dependent convergence analysis studies specified update rules rather than oracle classes. Classical works have established sharp worst-case analyses for line search methods [12, 13] and many other first-order methods [27, 84]. Recent works characterized the pros and cons of predetermined step-size schedules in gradient descent [33, 2, 3, 92]. For heavy-ball methods, the convergence analyses and cycling counterexamples [30, 53] were followed by provable non-acceleration results for smooth strongly convex objectives [32], while acceleration becomes possible through over-relaxed recurrences [89] or randomized scheduling on smooth convex objectives [36]. Our work establishes the dimension-dependent lower bound specifically for NorMuon with exact polar computation under any deterministic adaptive nonnegative step-size policy and any fixed momentum parameters.

2 Preliminaries and Technical Background

We review the setup for LLM pretraining and the update rules of Muon and NorMuon, followed by the assumptions and basic properties used in our analysis.

Algorithm 1 NorMuon

Input: $X_0 \in \mathbb{R}^{m\times n}$, $\ell, \delta, \beta_1, \beta_2 \in [0,1)$, $\eta_t, \alpha \geq 0$, $v_0 = 0 \in \mathbb{R}^m$, and $M_0 = 0 \in \mathbb{R}^{m\times n}$.
for $t = 0, \ldots, T-1$ **do**
  $G_t \leftarrow G(X_t, \xi_t)$, where ξ_t is independent of $\{X_0, \xi_0, \ldots, \xi_{t-1}\}$.
  $M_{t+1} \leftarrow \beta_1 M_t + (1-\beta_1) G_t$.
  $O_{t+1} \leftarrow \mathrm{Polar}_{\ell,\delta}(\frac{M_{t+1}}{\|M_{t+1}\|_F})$.
  $v_{t+1} \leftarrow \beta_2 v_t + \frac{1-\beta_2}{n} \mathrm{diag}(O_{t+1}O_{t+1}^\top)$.
  $[\overline{O}_{t+1}]_{i,:} \leftarrow \frac{[O_{t+1}]_{i,:}}{\sqrt{v_{t+1,i}+\alpha}}$ for all $i \in \{1, \ldots, m\}$.
  $D_{t+1} \leftarrow \frac{0.2\sqrt{mn}}{\|\overline{O}_{t+1}\|_F} \overline{O}_{t+1}$.
  $X_{t+1} \leftarrow X_t - \eta_t D_{t+1}$.
return $\widetilde{X} = X_{\widetilde{T}}$ where $\widetilde{T}$ is sampled independently and uniformly from $\{0, \ldots, T-1\}$.

2.1 LLM PRETRAINING AND MATRIX OPTIMIZATION

We consider an autoregressive language model with parameters θ and vocabulary $\mathcal{V}$. Given a token sequence $\mathbf{s} = (s_1, \ldots, s_{|\mathbf{s}|}) \in \mathcal{V}^{|\mathbf{s}|}$, the model assigns probability $p_\theta(\mathbf{s}) = \prod_{k=1}^{|\mathbf{s}|} p_\theta(s_k \mid s_{<k})$, where $s_{<k}$ denotes the first $k-1$ tokens. For a data distribution $\mathcal{D}$ over such sequences, the goal of pretraining is to minimize the expected next-token prediction loss as follows,

$$\mathcal{L}(\theta) = -\mathbb{E}_{\mathbf{s}\sim\mathcal{D}} \left[\frac{1}{|\mathbf{s}|} \sum_{k=1}^{|\mathbf{s}|} \log p_\theta(s_k \mid s_{<k}) \right].$$

where the model parameters θ consist of weight matrices in attention and feed-forward layers [87]. Muon and its variants apply their updates to each weight matrix separately. As such, we consider a single matrix variable in our theoretical analysis. Formally, we have

$$\min_{X\in\mathbb{R}^{m\times n}} F(X),$$

where $F : \mathbb{R}^{m\times n} \to \mathbb{R}$ is differentiable but not necessarily convex.

Muon [42] updates a weight matrix using the polar factor of its gradient momentum. For a nonzero matrix $Z \in \mathbb{R}^{m\times n}$ with compact singular value decomposition (SVD) $Z = U\Sigma V^\top$, where $\Sigma \in \mathbb{R}^{r\times r}$ and $r = \mathrm{rank}(Z)$, we define $\mathrm{Polar}(Z) = UV^\top$, and set $\mathrm{Polar}(0) = 0$. Let $\|\cdot\|_F$ denote the Frobenius norm. Starting from $M_0 = 0$, the exact Muon recursion is

$$M_{t+1} = \beta M_t + (1-\beta) G(X_t, \xi_t), \quad O_{t+1} = \mathrm{Polar}(\tfrac{M_{t+1}}{\|M_{t+1}\|_F}), \quad X_{t+1} = X_t - \eta_t O_{t+1},$$

where $G(X_t, \xi_t) \in \mathbb{R}^{m\times n}$ is a stochastic gradient at X_t, $\beta \in [0,1)$ is the momentum parameter, and $\eta_t > 0$ is the step size. In practice, Newton-Schulz iterations [42] or PolarExpress [4] can approximate the polar factor using the normalized momentum $\frac{M_{t+1}}{\|M_{t+1}\|_F}$ as input.

NorMuon [54] adds row-wise normalization to this update. Given the possibly approximate polar factor O_{t+1}, it maintains an exponential moving average of the mean squared entries of each row and rescales that row accordingly:

$$v_{t+1,i} = \beta_2 v_{t,i} + (1-\beta_2) \frac{\|[O_{t+1}]_{i,:}\|_2^2}{n}, \quad [\overline{O}_{t+1}]_{i,:} = \frac{[O_{t+1}]_{i,:}}{\sqrt{v_{t+1,i}+\alpha}}, \quad \text{for each } i = 1, \ldots, m,$$

where $[O]_{i,:}$ denotes the ith row of O, $\|\cdot\|_2$ denotes the Euclidean norm, $v_0 = 0$, $\beta_2 \in [0,1)$, and $\alpha \geq 0$ is a stabilization constant. Then, following the rescaling of [56, 54], we set

$$D_{t+1} = \frac{0.2\sqrt{mn}}{\|\overline{O}_{t+1}\|_F} \overline{O}_{t+1}, \quad X_{t+1} = X_t - \eta_t D_{t+1}.$$

We see that row normalization changes the relative magnitudes of the rows, while the final rescaling fixes the Frobenius norm of every nonzero update. The full scheme is summarized in Algorithm 1, including approximated polar computation as specified in Assumption 2.3.

2.2 Assumptions and basic properties

Throughout, we assume that $F^\star = \inf_{X\in\mathbb{R}^{m\times n}} F(X) > -\infty$ and let $\Delta \geq 0$ be an upper bound of $F(X_0) - F^\star$. For $X \in \mathbb{R}^{m\times n}$, we define $\|X\|_{\text{op}} = \max_{\|z\|_2\leq 1} \|Xz\|_2$, and $\|X\|_{\text{nuc}}$ as the sum of its singular values. These norms are dual under the Frobenius inner product $\langle X, Y\rangle = \text{tr}(X^\top Y)$. Indeed, we have

$$\max_{\|Y\|_{\text{op}}\leq 1} \langle X, Y\rangle = \langle X, \text{Polar}(X)\rangle = \|X\|_{\text{nuc}}.$$

This implies that the exact Muon recursion minimizes a linear model over an operator-norm ball centered at X_t [68]. In other words, we have $X_{t+1} \in \arg\min_{\|X-X_t\|_{\text{op}}\leq\eta_t} \langle X - X_t, M_{t+1}\rangle$. This motivates the operator-norm geometry used in our analysis. In particular, we impose the following assumptions on the objective and stochastic gradients.

Assumption 2.1. *There exists $L \geq 0$ such that $\|\nabla F(Y) - \nabla F(X)\|_{\text{nuc}} \leq L\|Y - X\|_{\text{op}}$ for all $X, Y \in \mathbb{R}^{m\times n}$.*

Assumption 2.2. *There exist $\sigma \geq 0$ and a stochastic gradient oracle $G : \mathbb{R}^{m\times n} \times \Xi \to \mathbb{R}^{m\times n}$ such that $\mathbb{E}[G(X, \xi) \mid X] = \nabla F(X)$ and $\mathbb{E}[\|G(X, \xi) - \nabla F(X)\|^2_{\text{nuc}} \mid X] \leq \sigma^2$ for all $X \in \mathbb{R}^{m\times n}$.*

A consequence of Assumption 2.1 is the descent inequality in the form of

$$F(Y) \leq F(X) + \langle \nabla F(X), Y - X\rangle + \tfrac{L}{2}\|Y - X\|^2_{\text{op}}, \quad \text{for all } X, Y \in \mathbb{R}^{m\times n}. \tag{2.1}$$

Our analysis uses this inequality to relate the decrease in the objective to the alignment between the gradient and the update direction. In the deterministic setting, we have $G_t = \nabla F(X_t)$ and $\sigma = 0$. To cover approximated polar computation, we impose a condition on the singular values of its output.

Assumption 2.3. *There exist $\ell, \delta \in [0, 1)$ and $\text{Polar}_{\ell,\delta} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ with $\text{Polar}_{\ell,\delta}(0) = 0$ such that the following statement holds. For every nonzero $Z \in \mathbb{R}^{m\times n}$ with $\|Z\|_F \leq 1$, $r = \text{rank}(Z)$ and $Z = U\Sigma V^\top$ being a compact SVD, we assume that $\text{Polar}_{\ell,\delta}(Z) = U\widetilde{\Sigma}V^\top$ for some diagonal $\widetilde{\Sigma} \in \mathbb{R}^{r\times r}$ satisfying (i) $\widetilde{\Sigma}_{jj} \in [0, 1 + \delta]$ and (ii) $\widetilde{\Sigma}_{jj} \in [1 - \delta, 1 + \delta]$ whenever $\Sigma_{jj} \geq \ell$, for all j.*

Assumption 2.3 allows an approximate polar computation. If an input singular value is at least ℓ, the output singular value must be within δ of 1. For smaller input singular values, the output lies between 0 and $1 + \delta$. As such, a smaller ℓ requires accurate approximation for more singular values, while a smaller δ requires greater accuracy. PolarExpress [4] uses polynomial iterations to adjust the singular values without changing the singular vectors and guarantees $\delta \leq (1 - \ell^2)^{3^K}$ after K iterations.

We use $\min_{0\leq t<T} \|\nabla F(X_t)\|_{\text{nuc}} \leq \epsilon$ and $\mathbb{E}[\|\nabla F(X_{\widehat{T}})\|_{\text{nuc}}] \leq \epsilon$ to measure the convergence in the deterministic and stochastic settings, and study the number of iterations that NorMuon requires to meet these criteria. In particular, we establish matching lower and upper bounds in the deterministic setting (Theorems 3.1 and 3.2), showing that row normalization incurs an extra factor of m compared with Muon. We extend the upper bound to the stochastic setting (Theorem 3.3), allowing approximated polar computation in both upper bound analyses.

3 Main Results

We establish matching upper and lower bounds for NorMuon in the deterministic setting and extend the upper bound to stochastic settings. We also explain the main ideas behind our analysis, with full proofs deferred to Appendix B.

3.1 Deterministic setting

We establish an algorithm-dependent lower bound with exact polar computation. This isolates the effect of row normalization from errors in computing the polar factor.

Theorem 3.1. *Let $m \geq 2$, $n \geq 1$, $\Delta, L > 0$, and $\epsilon \in (0, \frac{1}{4}\sqrt{\Delta L})$. For any deterministic step-size rule $\{\eta_t\}_{t\geq 0}$ based on the observed history $\{(X_s, F(X_s), \nabla F(X_s))\}_{s\leq t}$, there exists $F : \mathbb{R}^{m\times n} \to \mathbb{R}$ satisfying Assumption 2.1 and $F(X_0) - F^\star \leq \Delta$ such that Algorithm 1 with any fixed $\beta_1, \beta_2 \in [0, 1)$, exact gradients, and $\ell = \delta = \alpha = 0$ satisfies*

$$\|\nabla F(X_t)\|_{\text{nuc}} > \epsilon \text{ for all } 0 \leq t \leq T_0 = \left\lfloor \tfrac{m\Delta L}{32\epsilon^2} \right\rfloor.$$

Muon achieves a dimension-independent rate of $O(\Delta L\epsilon^{-2})$ under the same smoothness assumption and stationarity criterion [77]. Theorem 3.1 therefore shows that row normalization can introduce an extra factor of m, even when the momentum parameters and the adaptive step-size rule are chosen.

We next show that this dimension dependence is attainable. Indeed, we define

$$\kappa = \frac{(1-\delta)(1-\ell\min\{m,n\})\sqrt{1-\beta_2}}{(1+\delta)\sqrt{m}},$$

which quantifies the alignment retained after approximate polar computation and row normalization.

Theorem 3.2. *Suppose that Assumptions 2.1 and 2.3 hold with $L > 0$ and let $\ell < \frac{c}{\min\{m,n\}}$ where $c \in (0,1)$ is a universal constant. Then, there exists some $T > 0$ such that Algorithm 1 with $\beta_1 = 0$, $\beta_2 \in [0,1)$, $\alpha \geq 0$, $\eta_t \equiv \eta := \frac{\kappa\epsilon}{0.2L\sqrt{mn}}$ and exact gradients satisfies $\min_{0\leq t<T} \|\nabla F(X_t)\|_{\text{nuc}} \leq \epsilon$ and the total number of calls of the gradient oracle is bounded by*

$$O\left(\frac{m\Delta L}{\epsilon^2}\right),$$

where $\Delta > 0$ is an upper bound for the initial objective function gap $F(X_0) - F^\star$.

Theorem 3.2 shows that the upper bound of Algorithm 1 matches the lower bound for exact polar computation with $\alpha = 0$, and this bound is attained without momentum. The same $O(m\Delta L\epsilon^{-2})$ rate continues to hold under approximate polar computation, provided that $\ell < \frac{c}{\min\{m,n\}}$ for a universal constant $c \in (0,1)$.

3.2 Stochastic Setting

We proceed to the stochastic setting, where Algorithm 1 uses stochastic gradients.

Theorem 3.3. *Suppose that Assumptions 2.1, 2.2 and 2.3 hold with $L > 0$ and let $\ell < \frac{c}{\min\{m,n\}}$ where $c \in (0,1)$ is a universal constant and $\epsilon \in (0, \sqrt{\Delta L})$. Then, there exists some $T > 0$ such that Algorithm 1 with $\beta_1 = 1 - \min\{1, \frac{\kappa^2\epsilon^2}{64\sigma^2}\}$, $\beta_2 \in [0,1)$, $\alpha \geq 0$ and $\eta_t \equiv \eta := \frac{\kappa(1-\beta_1)\epsilon}{3.2L\sqrt{mn}}$ satisfies $\mathbb{E}[\|\nabla F(X_{\widetilde{T}})\|_{\text{nuc}}] \leq \epsilon$ and the total number of calls of the stochastic gradient oracle is bounded by*

$$O\left(\frac{m\Delta L}{\epsilon^2} + \frac{m^2\Delta L\sigma^2}{\epsilon^4}\right),$$

where $\Delta > 0$ is an upper bound for the initial objective function gap $F(X_0) - F^\star$.

Fixing $\sigma > 0$ and sufficiently small $\epsilon > 0$, Muon has an upper bound of $O(\min\{m,n\}\Delta L\sigma^2\epsilon^{-4})$ which is shown to be optimal under the same smoothness and noise assumptions [97]. Theorem 3.3 shows that there exists an additional dimension-dependent factor of $m\max\{1, m/n\}$ for NorMuon in the stochastic setting. We believe it is promising to either improve the upper bound of NorMuon or show the impossibility by establishing a lower bound in the stochastic setting.

3.3 Proof Sketches

Lower bound. The proof is divided into (i) constructing a function $F(X)$ and (ii) proving it is hard for NorMuon within T_0 iterations. Indeed, we define $\lambda = \langle H_\star, D_\star\rangle = \frac{2m-1}{\sqrt{m(m^2+m-1)}}$, where $e_1^{(n)}$ is the first basis of $\mathbb{R}^n$ and

$$H_\star = \frac{(m,1,\ldots,1)^\top}{\sqrt{m^2+m-1}}(e_1^{(n)})^\top \in \mathbb{R}^{m\times n}, \quad D_\star = \frac{1}{\sqrt{m}}(1,1,\ldots,1)^\top(e_1^{(n)})^\top \in \mathbb{R}^{m\times n}.$$

We construct two sequences $\{q_t\}_{t\leq T_0}$ and $\{f_t\}_{t\leq T_0}$, starting from $q_0 = f_0 = 0$. For $0 \leq t < T_0$, we feed the deterministic step-size rule $\{(X_0 + q_s D_\star, f_s, 2\epsilon H_\star)\}_{s\leq t}$, which returns $\eta_t \geq 0$. We set $q_{t+1} := q_t - \eta_t'$ with $\eta_t' := 0.2\sqrt{mn}\eta_t$ and $f_{t+1} := f_t - \int_0^{\eta_t'} \max\{2\epsilon\lambda - L\min\{z, \eta_t' - z\}, 0\}dz$. Given $\{q_t\}_{t\leq T_0}$ and $P = H_\star - \lambda D_\star$, we define

$$\begin{aligned} f(q) &= \int_0^q \max_{0\leq j\leq T_0} \max\{2\epsilon\lambda - L|z - q_j|, 0\}dz, \\ F(X) &= f(\langle D_\star, X - X_0\rangle) + \frac{L}{2\|P\|_F^2}\left(\langle P, X - X_0\rangle + \frac{2\epsilon\|P\|_F^2}{L}\right)^2 - \frac{2\epsilon^2\|P\|_F^2}{L}. \end{aligned}$$

We show that Algorithm 1 applied to F observes exactly $\{(X_0 + q_s D_\star, f_s, 2\epsilon H_\star)\}_{s \le T_0}$, that is,

$$X_t = X_0 + q_t D_\star, \quad F(X_t) = f_t, \quad \nabla F(X_t) = 2\epsilon H_\star, \quad \text{for all } 0 \le t \le T_0,$$

which implies $\|\nabla F(X_t)\|_{\text{nuc}} = 2\epsilon > \epsilon$ for all $t \le T_0$. Indeed, we have $F(X_0 + q_t D_\star) = f_t$ and $\nabla F(X_0 + q_t D_\star) = 2\epsilon\lambda D_\star + 2\epsilon P = 2\epsilon H_\star$ for all $t \le T_0$. It remains to show that $X_t = X_0 + q_t D_\star$, which we prove by induction on t. The case $t = 0$ is trivial. Suppose that $X_s = X_0 + q_s D_\star$ for all $s \le t$, then $G_s = 2\epsilon H_\star$ for all $s \le t$, and hence M_{t+1} is a positive multiple of $H_\star$ and $O_{t+1} = H_\star$. Since every row of $H_\star$ is a positive multiple of $e_1^{(n)}$, row normalization equalizes the rows and gives $D_{t+1} = 0.2\sqrt{mn} D_\star$ for any fixed $\beta_1, \beta_2 \in [0, 1)$. Therefore, the step-size rule observes $\{(X_0 + q_s D_\star, f_s, 2\epsilon H_\star)\}_{s \le t}$ and returns the same η_t, which gives $X_{t+1} = X_0 + q_{t+1} D_\star$.

Since $F(X_0) - F^\star \le \frac{2\epsilon^2}{L} + \frac{4T_0\epsilon^2\lambda^2}{L}$, $\lambda^2 < \frac{4}{m}$, and $\epsilon < \frac{1}{4}\sqrt{\Delta L}$, we have $F(X_0) - F^\star \le \Delta$ whenever $T_0 \le \frac{m\Delta L}{32\epsilon^2}$. Hence, our construction allows $T_0 = \Omega(m\Delta L\epsilon^{-2})$.

Upper bound. The key is to apply the descent inequality with $X = X_t$ and $Y = X_{t+1}$ as follows,

$$F(X_{t+1}) - F(X_t) \le -\eta_t \langle \nabla F(X_t), D_{t+1} \rangle + \frac{L\eta_t^2}{2} \|D_{t+1}\|_{\text{op}}^2. \tag{3.1}$$

By definition, $\|D_{t+1}\|_{\text{op}} \le 0.2\sqrt{mn}$. It suffices to bound $\langle \nabla F(X_t), D_{t+1} \rangle$ from below, and this is where row normalization matters. Lemma B.2 shows that

$$\langle M_{t+1}, D_{t+1} \rangle \ge 0.2\kappa\sqrt{mn} \|M_{t+1}\|_{\text{nuc}} \tag{3.2}$$

for any $\alpha \ge 0$ under approximate polar computation, where $\kappa = \Theta(m^{-1/2})$.

In the deterministic setting with $\beta_1 = 0$, we have $M_{t+1} = \nabla F(X_t)$. With η defined in Theorem 3.2, Eq. (3.1) becomes $F(X_{t+1}) - F(X_t) \le -\frac{\kappa^2\epsilon}{L} \|\nabla F(X_t)\|_{\text{nuc}} + \frac{\kappa^2\epsilon^2}{2L}$, which guarantees a decrease of at least $\frac{\kappa^2\epsilon^2}{2L}$ whenever $\|\nabla F(X_t)\|_{\text{nuc}} > \epsilon$. Telescoping Eq. (3.1) yields the desired result.

In the stochastic setting, we decompose $M_{t+1} = \nabla F(X_t) + E_t + B_t$ using a drift term E_t and a noise term B_t. Using Eq. (3.2) and the definition of κ, $\langle \nabla F(X_t), D_{t+1} \rangle \ge 0.2\sqrt{mn}(\kappa \|\nabla F(X_t)\|_{\text{nuc}} - 2\|E_t\|_{\text{nuc}} - 2\|B_t\|_F)$. Here, $\|E_t\|_{\text{nuc}}$ is controlled by Assumption 2.1, and $\|B_t\|_F$ is controlled by the martingale structure of gradient noises and Assumption 2.2. Choosing β_1 and η and telescoping Eq. (3.1) yields the desired result.

4 Experiments

We compare Muon and NorMuon on synthetic least-squares problems motivated by Theorem 3.1, image classification with a convolutional neural network (CNN), and LLM pretraining. We run the synthetic experiments on an AMD Ryzen 7 8845HS CPU, and the CNN and LLM experiments on 1 and 8 NVIDIA H200 GPUs, respectively. In all neural network experiments, we use the practical variant of NorMuon [54], which replaces the update direction $D_{t+1} = \frac{0.2\sqrt{mn}}{\|\overline{O}_{t+1}\|_F} \overline{O}_{t+1}$ in Algorithm 1 with

$$D_{t+1} = \frac{\|O_{t+1}\|_F}{\|\overline{O}_{t+1}\|_F} \overline{O}_{t+1}, \quad X_{t+1} \leftarrow X_t - \eta_t D_{t+1}, \tag{4.1}$$

so that the update has the same Frobenius norm as the Muon update, which simplifies parameter tuning. Since this change only rescales the step size, our lower bound still applies, and our upper bounds hold once the rescaling is absorbed into the step size.

4.1 Synthetic experiment

The hard function in Theorem 3.1 depends on the step-size rule, and hence it cannot serve as a fixed benchmark for candidate methods. We instead minimize a least-squares objective with imbalanced rows as follows,

$$F(X) = \tfrac{1}{2}\|X - \|C\|_F^{-1} C\|_F^2, \text{ where } C \in \mathbb{R}^{m \times n}, \ [C]_{i,:} = \tfrac{1}{i} u_i^\top,$$

and $u_1, \ldots, u_m$ are drawn independently and uniformly from the unit sphere in $\mathbb{R}^n$. We set $n = p = 8$ and $m \in \{8, 32, 128, 512, 2048\}$. We run Muon, NorMuon (Algorithm 1), and NorMuon with Eq. (4.1) from $X_0 = M_0 = v_0 = 0$ with $\alpha = 0$ for $T = 1000$ iterations, and tune a constant step size and the momentum parameters for each method and each m. See Appendix C for more details.

We consider the reduction $\log_{10}(F(X_0)) - \log_{10}(\min_{0\le t\le T} F(X_t))$, where a larger value is better. We tune the hyperparameters on 50 random seeds and report the results over 100 other random seeds. We compare Muon and NorMuon on the hard instances and report the results in Table 1. As m grows from 8 to 2048, the loss reduction of NorMuon drops, while that of Muon is similar (see Appendix C for the loss curves). The two versions of NorMuon perform almost identically. Since NorMuon with Eq. (4.1) has the same update norm as Muon, the gap can be attributed to row normalization.

Table 1: Mean $\pm$ 2$\times$standard error of the loss reduction $(\log_{10} F(X_0) - \log_{10} \min_{0\le t\le T} F(X_t))$ over 100 random seeds. A larger value means better performance.

m	8	32	128	512	2048
Muon	$5.70_{\pm 0.03}$	$5.70_{\pm 0.02}$	$5.70_{\pm 0.02}$	$5.72_{\pm 0.03}$	$5.72_{\pm 0.03}$
NorMuon (Algorithm 1)	$5.70_{\pm 0.03}$	$5.46_{\pm 0.02}$	$5.00_{\pm 0.02}$	$4.12_{\pm 0.01}$	$3.15_{\pm 0.01}$
NorMuon with Eq. (4.1)	$5.70_{\pm 0.03}$	$5.46_{\pm 0.02}$	$5.00_{\pm 0.01}$	$4.13_{\pm 0.01}$	$3.15_{\pm 0.01}$

4.2 Image Classification

Training. We train CIFARNET [41], a CNN with 2M parameters, on CIFAR-10 [48]. CIFARNET has 3 groups of convolutional layers with widths 64, 256, and 256. Each group applies a 3×3 convolution, max pooling, BatchNorm, GELU, a second 3×3 convolution, BatchNorm, and GELU. Flattening each convolution into a matrix in $\mathbb{R}^{d_{\text{out}} \times 9 d_{\text{in}}}$, where d_{in} and d_{out} are its numbers of input and output channels, gives six matrices of sizes 64×216, 64×576, 256×576, and three of sizes 256×2304. We update these matrix-valued parameters with AdamW [46, 59], Muon, or NorMuon with row or column normalization, and all other parameters with the same SGD optimizer, following Jordan et al. [42]. We use 50 epochs and batch size 512. See Appendix C for tuning details.

Evaluation. We select hyperparameters using the mean final validation accuracy over 20 seeds on a fixed 45,000/5,000 training/validation split of the training set. We retrain each selected configuration on all 50,000 training images with 50 new seeds and evaluate it on the test set.

Results. We compare AdamW, Muon, and both NorMuon variants in Table 2. All three Muon-family optimizers outperform AdamW. NorMuon with column normalization, which normalizes along the larger dimension of every weight matrix, attains the highest mean test accuracy by more than a standard error, while Muon attains the lowest mean test loss.

Table 2: Mean $\pm$ standard error of the test loss and accuracy over 50 seeds for CIFARNET.

	AdamW	Muon	NorMuon (column)	NorMuon (row)
Test loss	$0.4298_{\pm 0.0005}$	$\mathbf{0.3884}_{\pm 0.0004}$	$0.3889_{\pm 0.0005}$	$0.3900_{\pm 0.0004}$
Test accuracy	$92.33_{\pm 0.02}\%$	$93.79_{\pm 0.03}\%$	$\mathbf{93.87}_{\pm 0.02}\%$	$93.83_{\pm 0.02}\%$

4.3 LLM Pretraining

Training. We pretrain decoder-only transformers with 286M, 1.38B, and 6.44B total parameters using the nanochat codebase [43] on the Nemotron-CLIMB dataset [23] for 2.20B, 14.6B, and 19.7B tokens, respectively. Following the practice for Muon [42], Muon or NorMuon trains only the transformer matrices, which have about 84.9M, 679M, and 1.61B parameters, and AdamW trains all other parameters. The token budget of the 6.44B model follows the compute-optimal setting of nanochat, which allocates about 10.5 tokens per parameter in the transformer matrices and the language model head. The 286M and 1.38B models use 20 tokens per parameter in the transformer matrices and the language model head, which exceeds this setting. See Appendix C for more details.

We compare NorMuon with Muon and include AdamW with default hyperparameters, i.e., step size 0.001 and momentum factors $(0.9, 0.999)$, as a reference for the 1.38B and 6.44B models. As in Karpathy [43], the query, key, value, and output projections are separate square matrices. NorMuon

normalizes these square matrices along their rows and the MLP matrices along their larger dimension. The decoupled weight decay is 0.02 throughout. For Muon and NorMuon, the step size is set as $\eta\sqrt{2^{-19}B}$ [49], where η is the base step size and B is the batch size in tokens. We select $\eta = 0.03$ and $\eta = 0.01$ by a grid search on the 286M and the 1.38B models respectively (Section 4.4) and reuse $\eta = 0.01$ for the 6.44B model. Both optimizers use Nesterov-type momentum [24] with $\beta_1 = 0.95$ and 5 PolarExpress iterations [4], and NorMuon uses $\beta_2 = 0.95$. We perform a grid search to select the best step size for AdamW on the 286M model from $\{3 \times 10^{-4}, 10^{-3}, 3 \times 10^{-3}\}$ and pick 10^{-3}.

Evaluation. We hold out 41.94M tokens to compute the validation loss and 167.77M tokens to compute the test loss of the final model. Following OLMES [34], we report accuracy on 10 knowledge and reasoning benchmarks: MMLU [38], HellaSwag (HSwag) [94], PIQA [9], WinoGrande [72], ARC-Challenge (ARC-C) and ARC-Easy (ARC-E) [18], BoolQ [17], CommonsenseQA (CSQA) [82], Social IQa (SIQA) [73], and OpenBookQA (OBQA) [63].

Results. We show that NorMuon achieves lower validation loss than Muon for most of training at all 3 model sizes in Figure 1. We show that NorMuon improves upon Muon's test loss and average downstream accuracy for the 1.38B and 6.44B models in Table 3. For the 1.38B model, we run Muon and NorMuon with 3 random seeds and report the mean. See the standard deviations in Appendix C. All other runs use a single seed. Both Muon and NorMuon outperform AdamW. The 6.44B model uses a different tokenizer, and its loss is not directly comparable to those of the other models.

Table 3: LLM pretraining downstream accuracies and test losses.

		MMLU	HSwag	PIQA	Wino.	ARC-C	ARC-E	BoolQ	CSQA	SIQA	OBQA	Avg	Loss
1.38B	AdamW	31.9	54.9	74.6	54.2	45.1	72.3	57.9	56.6	46.2	48.2	54.2	2.3553
	Muon	**33.3**	60.3	75.9	55.5	43.8	73.6	**64.4**	**60.0**	49.1	47.1	56.3	2.2914
	NorMuon	**33.3**	**60.9**	**76.1**	**58.4**	**45.4**	**75.0**	62.0	59.6	**50.2**	**49.8**	**57.1**	**2.2810**
6.44B	AdamW	34.2	60.8	76.4	57.2	43.5	73.1	66.9	60.2	48.8	46.4	56.8	2.3748
	Muon	36.2	67.3	**78.5**	**61.4**	**48.7**	**79.6**	65.7	63.1	**53.0**	50.4	60.4	2.2803
	NorMuon	**36.4**	**68.2**	78.0	60.5	47.9	76.5	**72.8**	**64.5**	52.5	**51.6**	**60.9**	**2.2674**

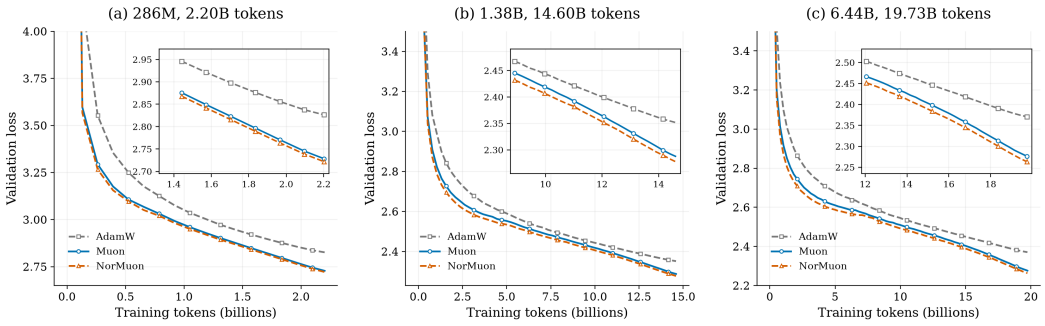

Figure 1: Validation loss in LLM pretraining for the (a) 286M, (b) 1.38B, and (c) 6.44B models.

4.4 Ablation Studies

Step size sensitivity. We test how sensitive Muon and NorMuon are to step sizes. We train the 286M model on 2.20B tokens with step sizes in $\{0.01, 0.02, 0.03, 0.04\}$ and the 1.38B model on 14.6B tokens with step sizes in $\{0.005, 0.01, 0.02\}$, and compare NorMuon with Muon at each step size. Figure 2(a) shows the final validation loss. NorMuon has lower loss than Muon at every step size, and both reach their lowest loss at 0.03 for the 286M model and 0.01 for the 1.38B model.

In what follows, we instead train the 1.38B model in the compute-optimal setting of 7.66B tokens.

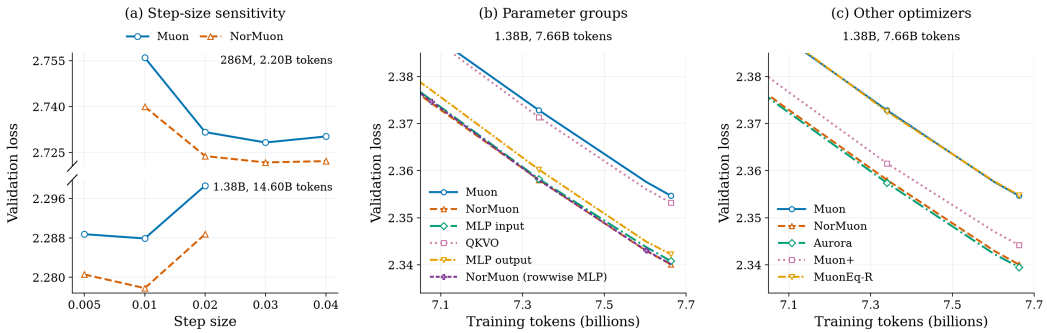

Figure 2: Ablation studies. (a) Final validation loss versus step size. (b) NorMuon applied to different groups of matrices. (c) Comparison with other Muon variants that use normalization.

Parameter groups. We test which weight matrices benefit from normalization. We apply NorMuon only to the attention query/key/value/output projections (QKVO), the MLP input projections, or the MLP output projections, with Muon optimizing the remaining matrices. We compare these variants with Muon, NorMuon on all transformer matrices, and NorMuon (rowwise MLP), which normalizes the MLP matrices along their rows instead of their larger dimension. As shown in Figure 2(b) and the left half of Table 4, normalizing only the QKVO matrices gives nearly the same validation and test losses as Muon, while normalizing only the MLP input or only the MLP output projections recovers most of the gain of NorMuon, and NorMuon (rowwise MLP) nearly matches NorMuon. Thus, the gain of NorMuon comes mainly from the MLP matrices. This aligns with Dewulf et al. [22].

Other matrix optimizers with normalization. We compare NorMuon with other Muon variants that use normalization. The baselines include Muon, Aurora [22], Muon+ [99], and MuonEq-R [14]. As shown in Figure 2(c) and the right half of Table 4, Aurora and NorMuon reach almost the same validation and test losses with Aurora being slightly lower, followed by Muon+, while MuonEq-R nearly overlaps with Muon. With the same step size and 5 PolarExpress iterations, Aurora, NorMuon, and Muon+, which all normalize the orthogonalized update, improve on Muon.

Table 4: Test loss for parameter group and optimizer ablations.

NorMuon parameter groups	286M	1.38B	Other optimizers	286M	1.38B
Muon	2.7311	2.3587	Muon	2.7311	2.3587
NorMuon	**2.7248**	**2.3442**	NorMuon	2.7248	2.3442
QKVO only	2.7319	2.3572	Aurora	**2.7237**	**2.3435**
MLP input only	2.7258	2.3448	Muon+	2.7265	2.3483
MLP output only	2.7260	2.3463	MuonEq-R	2.7305	2.3587
NorMuon (rowwise MLP)	2.7260	**2.3442**			

5 Conclusion

We studied the effect of row normalization on Muon through the worst-case convergence of NorMuon in operator-norm geometry. Indeed, we established matching deterministic lower and upper bounds of $\Theta(mL\epsilon^{-2})$, revealing an extra factor of m compared with Muon. The lower bound holds with exact polar computation, any fixed momentum parameters and any adaptive step sizes. We also extended the upper bound analysis to stochastic settings, allowing approximate polar computation in both upper bound analyses. Synthetic experiments show a slowdown for NorMuon, whereas LLM pretraining experiments demonstrate improved performance over Muon. These findings highlight that balanced neuron-wise updates alone do not guarantee faster optimization in worst cases. Future directions include identifying structural properties of LLM training that explain when row normalization is beneficial and developing the theory that captures these benefits.

AI USE STATEMENT

In this work, we used generative AI tools for aiding and polishing the writing and delivery of the paper content. We have not used generative AI tools for other source of disclosures listed by ICLR. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

[1] K. Ahn, B. Xu, N. Abreu, Y. Fan, G. Magakyan, P. Sharma, Z. Zhan, and J. Langford. Dion: Distributed orthonormalized updates. *ArXiv Preprint: 2504.05295*, 2025.

[2] J. M. Altschuler and P. A. Parrilo. Acceleration by stepsize hedging: Multi-step descent and the silver stepsize schedule. *Journal of the ACM*, 72(2):1–38, 2025.

[3] J. M. Altschuler and P. A. Parrilo. Acceleration by stepsize hedging: Silver stepsize schedule for smooth convex optimization. *Mathematical Programming*, 213(1):1105–1118, 2025.

[4] N. Amsel, D. Persson, C. Musco, and R. M. Gower. The polar express: Optimal matrix sign methods and their application to the Muon algorithm. In *ICLR*, volume 2026, pp. 138323–138360, 2026.

[5] N. Amsel, J. Zhang, K. Ahn, A. Naeimi, A. Feng, B. Chen, T. Dao, and J. Langford. Dion3: Full-stack orthogonal updates. *ArXiv Preprint: 2608.11612*, 2026.

[6] Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, and B. Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1):165–214, 2023.

[7] J. Bernstein and L. Newhouse. Old optimizer, new norm: An anthology. In *NeurIPS Workshop on Optimization for Machine Learning*, 2024. URL https://openreview.net/forum?id=ux18f5nOpD.

[8] J. Bernstein and L. Newhouse. Modular duality in deep learning. In *ICML*, 2025. URL https://openreview.net/forum?id=hErdffTsLu.

[9] Y. Bisk, R. Zellers, R. Le Bras, J. Gao, and Y. Choi. PIQA: Reasoning about physical commonsense in natural language. In *AAAI*, volume 34, pp. 7432–7439, 2020.

[10] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points I. *Mathematical Programming*, 184(1-2):71–120, 2020.

[11] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points II: First-order methods. *Mathematical Programming*, 185(1-2):315–355, 2021.

[12] C. Cartis, N. I. M. Gould, and P. L. Toint. On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization problems. *SIAM Journal on Optimization*, 20(6):2833–2852, 2010.

[13] C. Cartis, N. I. M. Gould, and P. L. Toint. Worst-case evaluation complexity and optimality of second-order methods for nonconvex smooth optimization. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro*, pp. 3711–3750. World Scientific, 2018.

[14] D. Chang, Q. Shi, L. Zhang, Y. Li, R. Zhang, Y. Lu, Y. Liu, and G. Yuan. MuonEq: Balancing before orthogonalization with lightweight equilibration. *ArXiv Preprint: 2603.28254*, 2026.

[15] L. Chen, J. Li, and Q. Liu. Muon optimizes under spectral norm constraints. *Transactions on Machine Learning Research (TMLR)*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=Blz4hjxLwU.

[16] S. Choudhury, X. Cheng, M. Takáč, S. Na, and M. Kolar. Muon with Nesterov momentum: Heavy-tailed noise and (randomized) inexact polar decomposition. *ArXiv Preprint: 2605.06884*, 2026.

[17] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

[18] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *ArXiv Preprint: 1803.05457*, 2018.

[19] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.

[20] D. Davis and D. Drusvyatskiy. When do spectral gradient updates help in deep learning? *ArXiv Preprint: 2512.04299*, 2025.

[21] E. De Klerk, F. Glineur, and A. B. Taylor. On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions. *Optimization Letters*, 11 (7):1185–1199, 2017.

[22] A. Dewulf, D. Pai, L. Yang, A. Zhang, and B. Keigwin. Aurora: A leverage-aware spectral optimizer. *ArXiv Preprint: 2606.27715*, 2026.

[23] S. Diao, Y. Yang, Y. Fu, X. Dong, D. Su, M. Kliegl, Z. Chen, P. Belcak, Y. Suhara, H. Yin, et al. Nemotron-CLIMB: Clustering-based iterative data mixture bootstrapping for language model pre-training. In *NeurIPS Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=aBlqKPkc4a.

[24] T. Dozat. Incorporating Nesterov momentum into Adam. In *ICLR*, 2016. URL https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ.

[25] S. Dragutinović and R. Ranganath. To use or not to use Muon: How simplicity bias in optimizers matters. In *ICLR Workshop on Scientific Methods for Understanding Deep Learning*, 2026. URL https://openreview.net/forum?id=GsZtgQf3IM.

[26] Y. Drori and O. Shamir. The complexity of finding stationary points with stochastic gradient descent. In *ICML*, pp. 2658–2667. PMLR, 2020.

[27] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: A novel approach. *Mathematical Programming*, 145(1):451–482, 2014.

[28] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

[29] C. Fan, M. Schmidt, and C. Thrampoulidis. Implicit bias of spectral descent and Muon on multiclass separable data. In *NeurIPS*, pp. 39622–39669, 2025.

[30] E. Ghadimi, H. R. Feyzmahdavian, and M. Johansson. Global convergence of the heavy-ball method for convex optimization. In *ECC*, pp. 310–315. IEEE, 2015.

[31] A. Gonon, A.-A. Muşat, and N. Boumal. Insights on Muon from simple quadratics. *ArXiv Preprint: 2602.11948*, 2026.

[32] B. Goujaud, A. Taylor, and A. Dieuleveut. Provable non-accelerations of the heavy-ball method. *Mathematical Programming*, pp. 1–59, 2025.

[33] B. Grimmer. Provably faster gradient descent via long steps. *SIAM Journal on Optimization*, 34(3):2588–2608, 2024.

[34] Y. Gu, O. Tafjord, B. Kuehl, D. Haddad, J. Dodge, and H. Hajishirzi. OLMES: A standard for language model evaluations. In *Findings of ACL: NAACL*, pp. 5020–5048, 2025.

[35] V. Gupta, T. Koren, and Y. Singer. Shampoo: Preconditioned stochastic tensor optimization. In *ICML*, pp. 1842–1850. PMLR, 2018.

[36] C. He and S. Zhang. Heavy-ball method under randomized schedules. *ArXiv Preprint: 2609.09743*, 2026.

[37] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778. IEEE, 2016.

[38] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

[39] A. Henry, P. R. Dachapally, S. S. Pawar, and Y. Chen. Query-key normalization for transformers. In *Findings of the ACL: EMNLP*, pp. 4246–4253, 2020.

[40] F. Hübler, K. Lion, A. Orvieto, and N. He. Muown implicitly performs angular step-size decay. *ArXiv Preprint: 2606.23637*, 2026.

[41] K. Jordan. 94% on CIFAR-10 in 3.29 seconds on a single GPU. *ArXiv Preprint: 2404.00498*, 2024.

[42] K. Jordan, Y. Jin, V. Boza, J. You, F. Cesista, L. Newhouse, and J. Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/.

[43] A. Karpathy. Nanochat: The best ChatGPT that $100 can buy, 2025. URL https://github.com/karpathy/nanochat.

[44] E. Khwaja, C. Lettieri, G. Woo, E. Belouadah, M. Cenac, G. Jarry, E. Paquin, X. Zhao, V. Zhukov, O. Abou-Amal, et al. Toto 2.0: Time series forecasting enters the scaling era. *ArXiv Preprint: 2605.20119*, 2026.

[45] G. Y. Kim and M. Oh. Convergence of Muon with Newton-Schulz. In *ICLR*, 2026. URL https://openreview.net/forum?id=lJSfxtLpLm.

[46] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://openreview.net/forum?id=8gmWwjFyLj.

[47] V. Klee and G. J. Minty. How good is the simplex algorithm? *Inequalities*, 3(3):159–175, 1972.

[48] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[49] A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *ArXiv Preprint: 1404.5997*, 2014.

[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, pp. 1097–1105, 2012.

[51] T. Large, Y. Liu, M. Huh, H. Bahng, P. Isola, and J. Bernstein. Scalable optimization in the modular norm. In *NeurIPS*, pp. 73501–73548, 2024.

[52] T. T. Lau, Q. Long, and W. Su. PolarGrad: A class of matrix-gradient optimizers from a unifying preconditioning perspective. *ArXiv Preprint: 2505.21799*, 2025.

[53] L. Lessard, B. Recht, and A. Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

[54] Z. Li, L. Liu, C. Liang, W. Chen, and T. Zhao. NorMuon: Making Muon more efficient and scalable. In *ICML*, 2026. URL https://openreview.net/forum?id=m1IRWFAMsa.

[55] K. Lion, F. Hübler, B. Li, A. Orvieto, and N. He. Muown: Row-norm control for Muon optimization. *ArXiv Preprint: 2605.10797*, 2026.

[56] J. Liu, J. Su, X. Yao, Z. Jiang, G. Lai, Y. Du, Y. Qin, W. Xu, E. Lu, J. Yan, et al. Muon is scalable for LLM training. *ArXiv Preprint: 2502.16982*, 2025.

[57] Z. Liu, R. Zhang, Z. Wang, Y. Zhao, Y. Su, Z. Yang, and Z. Zhang. Muon^2: Boosting Muon via adaptive second-moment preconditioning. *ArXiv Preprint: 2604.09967*, 2026.

[58] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

[59] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[60] C. Ma, W. Gong, M. Scetbon, and E. Meeds. SWAN: SGD with normalization and whitening enables stateless LLM training. In *ICML*, 2025. URL https://openreview.net/forum?id=CQZXGmw5vO.

[61] J. Ma, Y. Huang, Y. Chi, and Y. Chen. Preconditioning benefits of spectral orthogonalization in Muon. *ArXiv Preprint: 2601.13474*, 2026.

[62] J. Martens and R. Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *ICML*, pp. 2408–2417. PMLR, 2015.

[63] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *EMNLP*, pp. 2381–2391, 2018.

[64] A. S. Nemirovski and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience, 1983.

[65] Y. Nesterov. *Lectures on Convex Optimization*, volume 137. Springer, 2018.

[66] NVIDIA. Megatron Core API documentation: Emerging optimizers, n.d. URL https://docs.nvidia.com/megatron-core/developer-guide/latest/apidocs/core/core.optimizer.emerging_optimizers.html.

[67] T. Parshakova, A. Khaled, M. Crawshaw, G. Garrigos, and R. M. Gower. Muon does not converge on convex Lipschitz functions. *ArXiv Preprint: 2605.08980*, 2026.

[68] T. Pethick, W. Xie, K. Antonakopoulos, Z. Zhu, A. Silveti-Falls, and V. Cevher. Training deep learning models with norm-constrained LMOs. In *ICML*, pp. 49069–49104. PMLR, 2025.

[69] B. T. Polyak. Gradient methods for the minimisation of functionals. *USSR Computational Mathematics and Mathematical Physics*, 3(4):864–878, 1963.

[70] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of Adam and beyond. In *ICLR*, 2018. URL https://openreview.net/forum?id=ryQu7f-RZ.

[71] I. Safran and O. Shamir. How good is SGD with random shuffling? In *COLT*, pp. 3250–3284. PMLR, 2020.

[72] K. Sakaguchi, R. Le Bras, C. Bhagavatula, and Y. Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[73] M. Sap, H. Rashkin, D. Chen, R. Le Bras, and Y. Choi. Social IQa: Commonsense reasoning about social interactions. In *EMNLP-IJCNLP*, pp. 4463–4473, 2019.

[74] M. Scetbon, C. Ma, W. Gong, and E. Meeds. Gradient multi-normalization for efficient LLM training. In *NeurIPS*, 2025. URL https://openreview.net/forum?id=oanhUGY6un.

[75] M. Sfyraki and J. Wang. Lions and Muons: Optimization via stochastic Frank-Wolfe under heavy-tailed noise. In *ICML*, 2026. URL https://openreview.net/forum?id=gvroXZ0HS8.

[76] N. Shazeer and M. Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *ICML*, pp. 4596–4604. PMLR, 2018.

[77] W. Shen, R. Huang, M. Huang, C. Shen, and J. Zhang. On the convergence analysis of Muon. *Transactions on Machine Learning Research (TMLR)*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=4nH4CulGaP. J2C Certification.

[78] C. Si, D. Zhang, and W. Shen. AdaMuon: Adaptive Muon optimizer. *ArXiv Preprint: 2507.11005*, 2025.

[79] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[80] R. Sun and Y. Ye. Worst-case complexity of cyclic coordinate descent: O(n^2) gap with randomized version. *Mathematical Programming*, 185(1):487–520, 2021.

[81] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, pp. 1139–1147. PMLR, 2013.

[82] A. Talmor, J. Herzig, N. Lourie, and J. Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, 2019.

[83] H. Tao, D. Yu, and L. Zhang. When and why SignSGD outperforms SGD: A theoretical study based on ℓ_1-norm lower bounds. *ArXiv Preprint: 2605.06615*, 2026.

[84] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming*, 161(1):307–345, 2017.

[85] Kimi Team, Y. Bai, Y. Bao, Y. Charles, C. Chen, G. Chen, H. Chen, H. Chen, J. Chen, N. Chen, et al. Kimi K2: Open agentic intelligence. *ArXiv Preprint: 2507.20534*, 2025.

[86] T. Tieleman and G. E. Hinton. Neural networks for machine learning, Lecture 6.5 - RMSProp, 2012. COURSERA: Neural Networks for Machine Learning.

[87] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, pp. 6000–6010, 2017.

[88] N. Vyas, D. Morwani, R. Zhao, I. Shapira, D. Brandfonbrener, L. Janson, and S. Kakade. SOAP: Improving and stabilizing Shampoo using Adam for language modeling. In *ICLR*, volume 2025, pp. 93423–93444, 2025.

[89] J. Wei and L. Chen. Accelerated over-relaxation heavy-ball method: Achieving global accelerated convergence with broad generalization. In *ICLR*, 2025. URL https://openreview.net/forum?id=SWEqzy7IQB.

[90] K. Wen, Z. Li, J. Wang, D. Hall, P. Liang, and T. Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape view. In *ICLR*, 2025. URL https://openreview.net/forum?id=m51BgoqvbP.

[91] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. In *NeurIPS*, pp. 4151–4161, 2017.

[92] Y. Ye and K. Liu. Silver rate is (almost) optimal for gradient descent. *ArXiv Preprint: 2609.09152*, 2026.

[93] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *ICLR*, 2020. URL https://openreview.net/forum?id=Syx4wnEtvH.

[94] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? In *ACL*, pp. 4791–4800, 2019.

[95] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023.

[96] B. Zhang and R. Sennrich. Root mean square layer normalization. In *NeurIPS*, volume 32, 2019.

[97] J. Zhang and T. Lin. Scale-invariant neural network optimization: Norm geometry and heavy-tailed noise. *ArXiv Preprint: 2605.18528*, 2026.

[98] M. Zhang, Y. Liu, and H. Schaeffer. Adam improves Muon: Adaptive moment estimation with orthogonalized momentum. *ArXiv Preprint: 2602.17080*, 2026.

[99] R. Zhang, Y. Zhao, Z. Liu, Z. Wang, Y. Su, L. Tan, and Z. Zhang. MUON+: Towards more effective Muon via one additional normalization step for LLM pre-training. *ArXiv Preprint: 2602.21545*, 2026.

[100] Y. Zhang, C. Chen, N. Shi, R. Sun, and Z. Luo. Adam can converge without any modification on update rules. In *NeurIPS*, pp. 28386–28399, 2022.

[101] Y. Zhang, C. Chen, Z. Li, T. Ding, C. Wu, D. Kingma, Y. Ye, Z. Luo, and R. Sun. Adam-mini: Use fewer learning rates to gain more. In *ICLR*, 2025. URL https://openreview.net/forum?id=iBExhaU3Lc.

[102] Z. Zhou, T. Wu, Z. Jiang, F. Obeid, and Z. Lan. Value residual learning. In *ACL (Volume 1: Long Papers)*, pp. 28341–28356, 2025.

A Further Related Work

We make some comments on other topics, including more discussions on matrix-aware optimization methods, the theoretical analysis of matrix-aware optimization methods, lower bound analysis, and other optimizers and analysis. For an overview of neural network optimization methods and analysis, we refer to the monographs [65, 95].

More discussion on matrix-aware optimization methods. Earlier matrix optimizers exploit the structure of weight matrices through Kronecker-factored or second-moment preconditioners [62, 35, 88]. Muon [42] and Scion [68] depart from preconditioning and instead construct updates by dualizing the gradient under the operator norm. Several other Muon variants modify how the update is scaled or preconditioned. AdaMuon [78] applies element-wise second-moment normalization to the orthogonalized update. NAMO [98] scales the orthogonalized momentum by a single adaptive step size, while its diagonal extension NAMO-D applies column-wise adaptive scaling. Muon^2 [57] instead preconditions the momentum before orthogonalization, which improves the conditioning of the input to the polar computation. Muown [55, 40] controls row magnitudes through reparameterization. Stateless normalization has also been studied: SWAN [60] applies row normalization followed by orthogonalization, and SinkGD [74] replaces the orthogonalization with alternating row and column normalization. Other extensions target distributed efficiency [1, 5], step size scaling based on polar decomposition [52], and more accurate polar approximation [4].

Analysis of matrix-aware optimization methods. A conceptual basis for Muon is that the gradient is a dual vector and should be mapped back to the primal space before being used as an update. Bernstein & Newhouse [7] identify the orthogonalized update as steepest descent under the spectral norm, and modular dualization [8, 51] extends this view by assigning each layer an operator norm according to its input-output structure; see also [68, 15]. Within this operator-norm geometry, exact polar analyses establish a dimension-free $O(\Delta L\epsilon^{-2})$ rate for Muon [77], and Muon with weight decay converges as a stochastic Frank-Wolfe method [75]. Kim & Oh [45] prove that finite Newton-Schulz steps only introduce a multiplicative factor that approaches one doubly exponentially, and Choudhury et al. [16] further handle Nesterov momentum and heavy-tailed noise. Other works study implicit and simplicity biases, local quadratic models, structured problems, and nonsmooth settings [29, 25, 20, 61, 31, 67].

Lower bound analysis. Classical first-order complexity analysis apply to every gradient-based optimization method and therefore gives no specific treatment to individual update rules [64, 10, 11, 6], leaving open the best rate that a particular algorithm can achieve. Algorithm-dependent convergence analysis instead analyzes a fixed update rule and seeks the best convergence result attainable by tuning it. Additional examples include tight stationary-point lower bounds for SGD [26], explicit dimension-dependent lower bounds for cyclic coordinate descent relative to randomized coordinate descent [80], lower bounds distinguishing random-shuffling and incremental-gradient schemes [71], and exact worst-case constructions for gradient descent with exact line search [21]. More recently, related lower-bound analyses have been developed for sign-based methods [83].

Other optimizers and analysis. Classical optimizers treat weight matrices as vectors, from SGD with momentum [81] to adaptive methods that rescale each coordinate by accumulated or exponentially averaged squared gradients [28, 86, 46, 59]. Other works rescale updates at a coarser granularity to exploit parameter structure [93]. Adafactor [76] factorizes the second moment of a weight matrix into row and column statistics to save memory. Adam-mini [101] uses a single second-moment estimate for each Hessian-informed block. NorMuon adopts this normalization, but applies it to the row-wise norms of the orthogonalized update rather than to gradient coordinates.

Our analysis is also motivated by prior works that study adaptive gradient methods. Wilson et al. [91] construct a linear classification problem on which adaptive methods converge to poorly generalizing solutions, whereas SGD finds the minimum norm solution. Reddi et al. [70] construct simple convex problems on which Adam fails to converge. Zhang et al. [100] show that Adam converges when β_2 is chosen large enough after the problem is fixed. In contrast, we show that the worst-case cost of row normalization in NorMuon is not non-convergence but an extra dimension-dependent factor in the iteration complexity, and this cost persists under any fixed momentum parameters.

B MISSING PROOFS

Throughout this section, we let $d = \min\{m, n\}$ and $e_i^{(n)} \in \mathbb{R}^n$ be the ith basis vector of $\mathbb{R}^n$.

B.1 PROOF OF THEOREM 3.1

We define $\lambda = \langle H_\star, D_\star \rangle = \frac{2m-1}{\sqrt{m(m^2+m-1)}}$, where

$$H_\star := \tfrac{(m,1,\dots,1)^\top}{\sqrt{m^2+m-1}}(e_1^{(n)})^\top \in \mathbb{R}^{m\times n}, \quad D_\star := \tfrac{1}{\sqrt{m}}(1,1,\dots,1)^\top (e_1^{(n)})^\top \in \mathbb{R}^{m\times n}. \tag{B.1}$$

We construct two sequences $\{q_t\}_{t=0}^{T_0}$ and $\{f_t\}_{t=0}^{T_0}$ by setting $q_0 = 0$ and $f_0 = 0$. For $0 \le t < T_0$, we feed the deterministic adaptive step size rule the history $\{(X_0 + q_s D_\star, f_s, 2\epsilon H_\star)\}_{s\le t}$, which returns a fixed $\eta_t \ge 0$. We let $\eta_t' = 0.2\sqrt{mn}\eta_t$ and define

$$q_{t+1} = q_t - \eta_t', \quad f_{t+1} = f_t - \int_0^{\eta_t'} \max\{2\epsilon\lambda - L\min\{z, \eta_t' - z\}, 0\} dz. \tag{B.2}$$

We also define

$$f(q) = \int_0^q \max_{0\le j\le T_0} \max\{2\epsilon\lambda - L|z - q_j|, 0\} dz, \tag{B.3}$$

$$F(X) = f(\langle D_\star, X - X_0\rangle) + \tfrac{L}{2\|H_\star - \lambda D_\star\|_F^2}(\langle H_\star - \lambda D_\star, X - X_0\rangle + \tfrac{2\epsilon\|H_\star - \lambda D_\star\|_F^2}{L})^2 - \tfrac{2\epsilon^2\|H_\star - \lambda D_\star\|_F^2}{L}. \tag{B.4}$$

We claim that, when applied to F with any fixed $\beta_1, \beta_2 \in [0, 1)$, exact gradients and $\ell = \delta = \alpha = 0$, Algorithm 1 satisfies

$$X_t = X_0 + q_t D_\star, \quad F(X_t) = f(q_t) = f_t, \quad \nabla F(X_t) = 2\epsilon H_\star, \quad \text{for all } t = 0, 1, \dots, T_0. \tag{B.5}$$

In other words, Algorithm 1 observes $\{(X_0 + q_s D_\star, f_s, 2\epsilon H_\star)\}_{s\le T_0}$ and selects the same $\{\eta_s\}_{s<T_0}$. Since $\|2\epsilon H_\star\|_{\text{nuc}} = 2\epsilon > \epsilon$, we obtain the desired result in Theorem 3.1. Thus, it suffices to show that (i) Eq. (B.5) holds; (ii) F satisfies Assumption 2.1; and (iii) $F(X_0) - F^\star \le \Delta$.

We prove (i). The sequence $\{q_t\}_{t\le T_0}$ is nonincreasing, which implies that the distance from any $q \in [q_{t+1}, q_t]$ to $\{q_0, \dots, q_{T_0}\}$ is $\min\{q - q_{t+1}, q_t - q\}$. Therefore, we have

$$f(q_t) - f(q_{t+1}) = \int_0^{\eta_t'} \max\{2\epsilon\lambda - L\min\{z, \eta_t' - z\}, 0\} dz = f_t - f_{t+1}.$$

Since $f(q_0) = f_0 = 0$, we have $f(q_t) = f_t$ for all $0 \le t \le T_0$. By definition, we have

$$\nabla F(X) = f'(\langle D_\star, X - X_0\rangle) D_\star + (\tfrac{L\langle H_\star - \lambda D_\star, X - X_0\rangle}{\|H_\star - \lambda D_\star\|_F^2} + 2\epsilon)(H_\star - \lambda D_\star).$$

Since $j = t$ attains the maximum in Eq. (B.3), we have $f'(q_t) = 2\epsilon\lambda$. Putting these pieces together with $\langle H_\star - \lambda D_\star, D_\star\rangle = 0$ yields that $F(X_0 + q_t D_\star) = f_t$ and $\nabla F(X_0 + q_t D_\star) = 2\epsilon H_\star$ for all $0 \le t \le T_0$.

It remains to show $X_t = X_0 + q_t D_\star$ for all $t \le T_0$. Indeed, we prove by induction. The case of $t = 0$ follows from $q_0 = 0$. Suppose that $X_s = X_0 + q_s D_\star$ for all $s \le t < T_0$. Then, we have $G_s = \nabla F(X_s) = 2\epsilon H_\star$ and $F(X_s) = f_s$ for $s \le t$. Thus, the deterministic adaptive step size rule observes $\{(X_0 + q_\tau D_\star, f_\tau, 2\epsilon H_\star)\}_{\tau\le t}$ and selects η_t. Since $\operatorname{rank}(H_\star) = \|H_\star\|_F = 1$, Algorithm 1 gives $M_{s+1} = 2\epsilon(1 - \beta_1^{s+1})H_\star$ and $O_{s+1} = \operatorname{Polar}(H_\star) = H_\star$ for all $s \le t$. By the definition of $H_\star$ (see Eq. (B.1)), we have

$$v_{t+1} = \sum_{s=0}^{t} \beta_2^{t-s}\tfrac{1-\beta_2}{n}\operatorname{diag}(O_{s+1}O_{s+1}^\top) = \tfrac{1-\beta_2^{t+1}}{n(m^2+m-1)}(m^2, 1, \dots, 1)^\top, \tag{B.6}$$

Since $\alpha = 0$, we have

$$[\overline{O}_{t+1}]_{i,:} = \tfrac{1}{\sqrt{[v_{t+1}]_i}}[O_{t+1}]_{i,:} = \tfrac{1}{\sqrt{[v_{t+1}]_i}}[H_\star]_{i,:} = \begin{cases} \frac{m}{\sqrt{(m^2+m-1)[v_{t+1}]_1}} e_1^{(n)}, & i = 1, \\ \frac{1}{\sqrt{(m^2+m-1)[v_{t+1}]_i}} e_1^{(n)}, & i = 2, \dots, m. \end{cases}$$

This together with Eq. (B.6) yields that $[\overline{O}_{t+1}]_{i,:} = \sqrt{\frac{n}{1-\beta_2^{t+1}}} e_1^{(n)}$ for all $i = 1, \ldots, m$. In addition, by the definition of $D_\star$ (see Eq. (B.1)), we have

$$\overline{O}_{t+1} = \sqrt{\frac{mn}{1-\beta_2^{t+1}}} D_\star,$$

which further implies

$$\|\overline{O}_{t+1}\|_F = \sqrt{\frac{mn}{1-\beta_2^{t+1}}}, \quad D_{t+1} = 0.2\sqrt{mn}\frac{\overline{O}_{t+1}}{\|\overline{O}_{t+1}\|_F} = 0.2\sqrt{mn} D_\star.$$

By induction, we have $X_{t+1} = X_t - \eta_t D_{t+1} = X_0 + q_t D_\star - \eta_t D_{t+1} = X_0 + q_{t+1} D_\star$.

We prove (ii). Since f' is a maximum of L-Lipschitz functions, we obtain that f' is L-Lipschitz. Since $\nabla F(X)$ and $\nabla F(Y)$ are supported in the first column for all $X, Y \in \mathbb{R}^{m\times n}$, we have

$$\begin{aligned}
&\|\nabla F(X) - \nabla F(Y)\|_{\text{nuc}}^2 = \|\nabla F(X) - \nabla F(Y)\|_F^2 \\
&= |f'(\langle D_\star, X - X_0\rangle) - f'(\langle D_\star, Y - X_0\rangle)|^2 + \tfrac{L^2 |\langle H_\star - \lambda D_\star, X - Y\rangle|^2}{\|H_\star - \lambda D_\star\|_F^2} \\
&\le L^2 \left(|\langle D_\star, X - Y\rangle|^2 + \tfrac{|\langle H_\star - \lambda D_\star, X - Y\rangle|^2}{\|H_\star - \lambda D_\star\|_F^2} \right) \\
&\le L^2 \|(X - Y) e_1^{(n)}\|_2^2 \le L^2 \|X - Y\|_{\text{op}}^2,
\end{aligned}$$

where we used the fact that $D_\star$ and $\frac{H_\star - \lambda D_\star}{\|H_\star - \lambda D_\star\|_F}$ are orthonormal and supported in the first column.

We prove (iii). Since $f' \ge 0$, it follows from Eq. (B.3) that

$$\begin{aligned}
f(0) - \inf_{q\in\mathbb{R}} f(q) &= \int_{-\infty}^{0} \max_{0\le j\le T_0} \max\{2\epsilon\lambda - L|z - q_j|, 0\} dz \\
&\le \int_{-\infty}^{0} \max\{2\epsilon\lambda - L|q|, 0\} dq + \sum_{j=1}^{T_0} \int_{-\infty}^{+\infty} \max\{2\epsilon\lambda - L|q - q_j|, 0\} dq \\
&= \tfrac{2\epsilon^2\lambda^2}{L} + \tfrac{4T_0\epsilon^2\lambda^2}{L}.
\end{aligned}$$

which implies that f is bounded below. By definition, F is bounded below and $F^\star$ is finite.

Since $F(X_0) = f(0) = 0$, Eq. (B.4) gives

$$F(X_0) - F^\star \le \tfrac{2\epsilon^2\lambda^2}{L} + \tfrac{4T_0\epsilon^2\lambda^2}{L} + \tfrac{2\epsilon^2\|H_\star - \lambda D_\star\|_F^2}{L} = \tfrac{2\epsilon^2}{L} + \tfrac{4T_0\epsilon^2\lambda^2}{L} \le \tfrac{2\epsilon^2}{L} + \tfrac{16T_0\epsilon^2}{mL} < \tfrac{5\Delta}{8} < \Delta,$$

where we used $\lambda^2 < \frac{4}{m}$, $T_0 \le \frac{m\Delta L}{32\epsilon^2}$, and $\epsilon < \frac{1}{4}\sqrt{\Delta L}$.

B.2 Proof of Theorems 3.2 and 3.3

We first prove the descent inequality in Eq. (2.1) for the sake of completeness.

Lemma B.1. *If Assumption 2.1 holds with $L > 0$, we have*

$$F(Y) \le F(X) + \langle \nabla F(X), Y - X\rangle + \tfrac{L}{2}\|Y - X\|_{\text{op}}^2 \text{ for any } X, Y \in \mathbb{R}^{m\times n}. \tag{B.7}$$

In addition, $F(X) - F^\star \le \Delta$ implies $\|\nabla F(X)\|_{\text{nuc}} \le \sqrt{2\Delta L}$.

Proof. For any $X, Y \in \mathbb{R}^{m\times n}$, we have

$$F(Y) - F(X) = \int_0^1 \langle \nabla F((1-s)X + sY), Y - X\rangle ds \le \langle \nabla F(X), Y - X\rangle + \tfrac{L}{2}\|Y - X\|_{\text{op}}^2.$$

Suppose that $\nabla F(X) = U\Sigma V^\top$ is a compact SVD. Then, we let $Y = X - \frac{1}{L}\|\nabla F(X)\|_{\text{nuc}} UV^\top$ and obtain that $F^\star \le F(Y) \le F(X) - \frac{\|\nabla F(X)\|_{\text{nuc}}^2}{2L}$. □

We then bound $\langle M_{t+1}, D_{t+1}\rangle$ in the following lemma.

Lemma B.2. *If Assumption 2.3 holds and $\ell \in [0, \frac{1}{d})$, we have $\langle M_{t+1}, D_{t+1}\rangle \geq 0.2\kappa\sqrt{mn}\|M_{t+1}\|_{\text{nuc}}$ and $\|D_{t+1}\|_{\text{op}} \leq \|D_{t+1}\|_F = 0.2\sqrt{mn}$ for any $\alpha \geq 0$ and any $t \geq 0$.*

Proof. If $M_{t+1} = 0$, we have $D_{t+1} = \overline{O}_{t+1} = O_{t+1} = 0$ which yields the desired result. Otherwise, by the update rule of Algorithm 1, we have

$$\langle M_{t+1}, D_{t+1}\rangle = \frac{0.2\sqrt{mn}\langle M_{t+1}, \overline{O}_{t+1}\rangle}{\|\overline{O}_{t+1}\|_F}. \tag{B.8}$$

First, we upper bound $\|\overline{O}_{t+1}\|_F$. By Assumption 2.3, we have $O_{t+1} = U\widetilde{\Sigma}V^\top$ where $M_{t+1} = U\Sigma V^\top$ is the compact SVD. For all t, the largest singular value of $\frac{M_{t+1}}{\|M_{t+1}\|_F}$ is at least $\frac{1}{\sqrt{d}} > \ell$, which implies that $1 - \delta \leq \|O_{t+1}\|_{\text{op}} \leq 1 + \delta$. Since $[v_{t+1}]_i = \beta_2[v_t]_i + \frac{1-\beta_2}{n}\|[O_{t+1}]_{i,:}\|_2^2$ and $v_0 = 0$, we have

$$\tfrac{1-\beta_2}{n}\|[O_{t+1}]_{i,:}\|_2^2 \leq [v_{t+1}]_i \leq \tfrac{(1+\delta)^2}{n}, \quad \text{for all } i.$$

By the definition of $\overline{O}_{t+1}$, we have

$$\|[\overline{O}_{t+1}]_{i,:}\|_2 = \tfrac{\|[O_{t+1}]_{i,:}\|_2}{\sqrt{[v_{t+1}]_i}+\alpha} \leq \tfrac{\|[O_{t+1}]_{i,:}\|_2}{\sqrt{\frac{1-\beta_2}{n}\|[O_{t+1}]_{i,:}\|_2}+\alpha} \leq \tfrac{1+\delta}{\sqrt{\frac{1-\beta_2}{n}(1+\delta)}+\alpha}, \quad \text{for all } i,$$

where the last inequality follows from monotonicity and $\|[O_{t+1}]_{i,:}\|_2 \leq \|O_{t+1}\|_{\text{op}} \leq 1 + \delta$. Thus, we have

$$\|\overline{O}_{t+1}\|_F \leq \frac{(1+\delta)\sqrt{m}}{\sqrt{\frac{1-\beta_2}{n}(1+\delta)}+\alpha}. \tag{B.9}$$

Second, we lower bound $\langle M_{t+1}, \overline{O}_{t+1}\rangle$. Since $M_{t+1}O_{t+1}^\top = U\Sigma\widetilde{\Sigma}U^\top \succeq 0$, for $i = 1, \ldots, m$, we have $\langle [M_{t+1}]_{i,:}, [O_{t+1}]_{i,:}\rangle \geq 0$. Then, we have

$$\begin{aligned} \langle M_{t+1}, \overline{O}_{t+1}\rangle &= \textstyle\sum_{i=1}^m \langle [M_{t+1}]_{i,:}, [\overline{O}_{t+1}]_{i,:}\rangle = \sum_{i=1}^m \langle [M_{t+1}]_{i,:}, \frac{1}{\sqrt{[v_{t+1}]_i}+\alpha}[O_{t+1}]_{i,:}\rangle \\ &\geq \textstyle\sum_{i=1}^m \frac{1}{\frac{1+\delta}{\sqrt{n}}+\alpha}\langle [M_{t+1}]_{i,:}, [O_{t+1}]_{i,:}\rangle = \frac{1}{\frac{1+\delta}{\sqrt{n}}+\alpha}\langle M_{t+1}, O_{t+1}\rangle. \end{aligned} \tag{B.10}$$

By Assumption 2.3, we obtain from $\Sigma_{jj} \geq \ell\|M_{t+1}\|_F$ that $\widetilde{\Sigma}_{jj} \geq 1 - \delta$. It follows that

$$\begin{aligned} \langle M_{t+1}, O_{t+1}\rangle &\geq (1-\delta)\textstyle\sum_{j:\,\Sigma_{jj} \geq \ell\|M_{t+1}\|_F} \Sigma_{jj} \geq (1-\delta)(\|M_{t+1}\|_{\text{nuc}} - d\ell\|M_{t+1}\|_F) \\ &\geq (1-\delta)(1-d\ell)\|M_{t+1}\|_{\text{nuc}}, \end{aligned} \tag{B.11}$$

where the last inequality uses $\|M_{t+1}\|_F \leq \|M_{t+1}\|_{\text{nuc}}$. Combining Eq. (B.8), Eq. (B.9), Eq. (B.10), and Eq. (B.11) yields

$$\tfrac{\langle M_{t+1}, D_{t+1}\rangle}{0.2\sqrt{mn}} \geq \tfrac{\sqrt{\frac{1-\beta_2}{n}}(1+\delta)+\alpha}{(1+\delta)\sqrt{m}(\frac{1+\delta}{\sqrt{n}}+\alpha)}(1-\delta)(1-d\ell)\|M_{t+1}\|_{\text{nuc}} \geq \kappa\|M_{t+1}\|_{\text{nuc}}.$$

By definition, we have $\|D_{t+1}\|_{\text{op}} \leq \|D_{t+1}\|_F = 0.2\sqrt{mn}$. □

Proof of Theorem 3.2. Since $\beta_1 = 0$ and $G_t = \nabla F(X_t)$, we have $M_{t+1} = \nabla F(X_t)$. By applying Lemmas B.1 and B.2 and using $X_{t+1} = X_t - \eta D_{t+1}$, we have

$$\begin{aligned} F(X_{t+1}) - F(X_t) &\leq -\eta\langle \nabla F(X_t), D_{t+1}\rangle + \tfrac{L\eta^2}{2}\|D_{t+1}\|_{\text{op}}^2 \\ &\leq -0.2\eta\kappa\sqrt{mn}\|\nabla F(X_t)\|_{\text{nuc}} + \tfrac{L\eta^2}{2}(0.2\sqrt{mn})^2 \\ &= -\tfrac{\kappa^2\epsilon}{L}\|\nabla F(X_t)\|_{\text{nuc}} + \tfrac{\kappa^2\epsilon^2}{2L}. \end{aligned}$$

Summing the above inequality over $t = 0, 1, \ldots, T-1$ and using $F(X_0) - F^\star \leq \Delta$ yields

$$\min_{0\leq t<T}\|\nabla F(X_t)\|_{\text{nuc}} \leq \tfrac{1}{T}\sum_{t=0}^{T-1}\|\nabla F(X_t)\|_{\text{nuc}} \leq \tfrac{\Delta L}{\kappa^2\epsilon T} + \tfrac{\epsilon}{2}.$$

We choose $T := \lceil \frac{2\Delta L}{\kappa^2\epsilon^2}\rceil$. By the definition of κ, we have $\min_{0\leq t<T}\|\nabla F(X_t)\|_{\text{nuc}} \leq \epsilon$ and $T = O(\frac{m\Delta L}{\epsilon^2})$. Since Algorithm 1 calls the gradient oracle once per iteration, the total number of calls of the gradient oracle is bounded by $O(\frac{m\Delta L}{\epsilon^2})$. □

Proof of Theorem 3.3. By applying Lemma B.1 and using $X_{t+1} = X_t - \eta D_{t+1}$, we have

$$F(X_{t+1}) - F(X_t) \leq -\eta\langle \nabla F(X_t), D_{t+1}\rangle + \tfrac{L\eta^2}{2}\|D_{t+1}\|_{\text{op}}^2. \tag{B.12}$$

We write $M_{t+1} = \nabla F(X_t) + E_t + B_t$, where

$$\begin{aligned} E_t &= -\beta_1^{t+1}\nabla F(X_0) + \textstyle\sum_{s=1}^{t} \beta_1^{t-s+1}(\nabla F(X_{s-1}) - \nabla F(X_s)), \\ B_t &= (1-\beta_1)\textstyle\sum_{s=0}^{t} \beta_1^{t-s}(G_s - \nabla F(X_s)). \end{aligned}$$

By using $\|D_{t+1}\|_{\text{op}} \leq \|D_{t+1}\|_F = 0.2\sqrt{mn}$ (see Lemma B.2) and $\kappa \leq \frac{1}{\sqrt{d}}$, we have

$$\begin{aligned} \tfrac{\langle \nabla F(X_t), D_{t+1}\rangle}{0.2\sqrt{mn}} &\geq \kappa\|M_{t+1}\|_{\text{nuc}} - \|E_t\|_{\text{nuc}} - \|B_t\|_F \\ &\geq \kappa\|\nabla F(X_t)\|_{\text{nuc}} - (1+\kappa)\|E_t\|_{\text{nuc}} - \|B_t\|_F - \kappa\|B_t\|_{\text{nuc}} \\ &\geq \kappa\|\nabla F(X_t)\|_{\text{nuc}} - 2\|E_t\|_{\text{nuc}} - 2\|B_t\|_F. \end{aligned}$$

Combining with Eq. (B.12), $\|D_{t+1}\|_{\text{op}} \leq 0.2\sqrt{mn}$ (see Lemma B.2) and the definition of η yields

$$F(X_{t+1}) - F(X_t) \leq -\tfrac{\kappa(1-\beta_1)\epsilon}{16L}\big(\kappa\|\nabla F(X_t)\|_{\text{nuc}} - 2\|E_t\|_{\text{nuc}} - 2\|B_t\|_F\big) + \tfrac{\kappa^2(1-\beta_1)^2\epsilon^2}{512L}.$$

Summing the above inequality over $t = 0, \ldots, T-1$ and using $F(X_0) - F^\star \leq \Delta$ yields

$$\kappa\textstyle\sum_{t=0}^{T-1}\|\nabla F(X_t)\|_{\text{nuc}} \leq \frac{16\Delta L}{\kappa(1-\beta_1)\epsilon} + 2\sum_{t=0}^{T-1}\|E_t\|_{\text{nuc}} + 2\sum_{t=0}^{T-1}\|B_t\|_F + \frac{\kappa(1-\beta_1)\epsilon T}{32}. \tag{B.13}$$

First, we bound $\sum_{t=0}^{T-1}\|E_t\|_{\text{nuc}}$. Indeed, we have $\|\nabla F(X_0)\|_{\text{nuc}} \leq \sqrt{2\Delta L}$ (see Lemma B.1) and obtain from Assumption 2.1 and $\|X_s - X_{s-1}\|_{\text{op}} \leq 0.2\eta\sqrt{mn}$ that $\|\nabla F(X_s) - \nabla F(X_{s-1})\|_{\text{nuc}} \leq \frac{\kappa(1-\beta_1)\epsilon}{16}$ for $1 \leq s \leq T$. This implies

$$\|E_t\|_{\text{nuc}} \leq \beta_1^{t+1}\sqrt{2\Delta L} + \tfrac{\kappa(1-\beta_1)\epsilon}{16}\textstyle\sum_{s=1}^{t}\beta_1^{t-s+1} \leq \beta_1^{t+1}\sqrt{2\Delta L} + \frac{\kappa\epsilon}{16},$$

and

$$\tfrac{1}{T}\sum_{t=0}^{T-1}\|E_t\|_{\text{nuc}} \leq \tfrac{\sqrt{2\Delta L}}{(1-\beta_1)T} + \tfrac{\kappa\epsilon}{16}.$$

Second, we bound $\mathbb{E}[\|B_t\|_F]$. Indeed, we let $\mathcal{F}_s$ be the sigma algebra generated by $X_0, \xi_0, \ldots, \xi_{s-1}$. The independence of ξ_s and $\mathcal{F}_s$ and Assumption 2.2 give $\mathbb{E}[G_s - \nabla F(X_s)|\mathcal{F}_s] = 0$ and $\mathbb{E}[\|G_s - \nabla F(X_s)\|_{\text{nuc}}^2|\mathcal{F}_s] \leq \sigma^2$. Since $G_s - \nabla F(X_s)$ is $\mathcal{F}_j$-measurable and $\mathbb{E}[\langle G_s - \nabla F(X_s), G_j - \nabla F(X_j)\rangle] = 0$ for $s < j$, we have

$$\begin{aligned} \mathbb{E}[\|B_t\|_F^2] &= (1-\beta_1)^2\textstyle\sum_{s=0}^{t}\beta_1^{2(t-s)}\mathbb{E}[\|G_s - \nabla F(X_s)\|_F^2] \leq (1-\beta_1)^2\sigma^2\sum_{s=0}^{t}\beta_1^{2(t-s)} \\ &\leq (1-\beta_1)\sigma^2. \end{aligned}$$

Taking expectations in Eq. (B.13) and putting these pieces together, we have

$$\tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(X_t)\|_{\text{nuc}}] \leq \tfrac{16\Delta L}{\kappa^2(1-\beta_1)\epsilon T} + \tfrac{2\sqrt{2\Delta L}}{\kappa(1-\beta_1)T} + \tfrac{5\epsilon}{32} + \tfrac{2\sigma\sqrt{1-\beta_1}}{\kappa}. \tag{B.14}$$

We choose $T := \lceil \frac{64\Delta L}{\kappa^2(1-\beta_1)\epsilon^2}\rceil$. By the definition of β_1 and $\epsilon \leq \sqrt{\Delta L}$, we have

$$\mathbb{E}[\|\nabla F(\widetilde{X})\|_{\text{nuc}}] = \tfrac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla F(X_t)\|_{\text{nuc}}] \leq \left(\tfrac{1}{4} + \tfrac{\sqrt{2}}{32} + \tfrac{5}{32} + \tfrac{1}{4}\right)\epsilon < \epsilon.$$

In addition, we have $\frac{1}{1-\beta_1} = \max\{1, \frac{64\sigma^2}{\kappa^2\epsilon^2}\} \leq 1 + \frac{64\sigma^2}{\kappa^2\epsilon^2}$. By the definition of κ, we have

$$T = \left\lceil \tfrac{64\Delta L}{\kappa^2(1-\beta_1)\epsilon^2}\right\rceil \leq 1 + \tfrac{64\Delta L}{\kappa^2\epsilon^2} + \tfrac{4096\Delta L\sigma^2}{\kappa^4\epsilon^4} = O\left(\tfrac{m\Delta L}{\epsilon^2} + \tfrac{m^2\Delta L\sigma^2}{\epsilon^4}\right).$$

Since Algorithm 1 calls the stochastic gradient oracle once per iteration, the total number of calls of the stochastic gradient oracle is bounded by $O(\frac{m\Delta L}{\epsilon^2} + \frac{m^2\Delta L\sigma^2}{\epsilon^4})$. □

C ADDITIONAL EXPERIMENTS

We present detailed setups and additional results for our experiments.

Table 5: Selected hyperparameters $(\gamma, \beta_1, \beta_2)$ for the synthetic experiment.

m	Muon	NorMuon (Algorithm 1)	NorMuon with Eq. (4.1)
8	$(10^{-3}, 0, -)$	$(10^{-3}, 0, 0.95)$	$(10^{-3}, 0, 0)$
32	$(10^{-3}, 0, -)$	$(10^{-2.5}, 0.5, 0.99)$	$(10^{-2.5}, 0.5, 0.99)$
128	$(10^{-3}, 0, -)$	$(10^{-2.5}, 0.5, 0.9)$	$(10^{-2.5}, 0.5, 0.9)$
512	$(10^{-3}, 0, -)$	$(10^{-2}, 0.5, 0.9)$	$(10^{-2}, 0.5, 0.9)$
2048	$(10^{-3}, 0, -)$	$(10^{-1.5}, 0.5, 0.9)$	$(10^{-1.5}, 0.5, 0.9)$

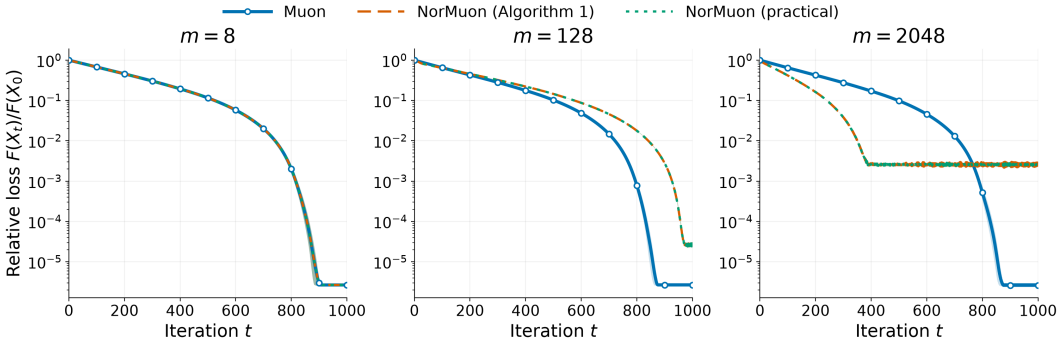

Figure 3: Loss curves for the synthetic experiment.

C.1 Setup for synthetic experiment

We run Muon, NorMuon (Algorithm 1), and practical NorMuon with the scaling in Eq. (4.1) on the same 150 fixed random seeds. We use 50 of these seeds to tune the constant step size and momentum parameters of each method for each m. For both Muon and NorMuon, we search over $\beta_1 \in \{0, 0.5, 0.9, 0.95\}$, and for NorMuon we additionally search over $\beta_2 \in \{0, 0.9, 0.95, 0.99\}$. We search over the effective step sizes $\gamma_j = 10^{-5+0.5j}$ for $j = 0, \ldots, 10$. To match the global update scale across methods, we set $\eta = \gamma_j$ for Muon and NorMuon with Eq. (4.1), and $\eta = \frac{\gamma_j}{0.2\sqrt{m}}$ for NorMuon (Algorithm 1). For each method and m, we select the hyperparameters that achieve the smallest minimum loss $\min_{0 \le t \le T} F(X_t)$ over $T = 1000$ iterations. We then evaluate all methods with their selected hyperparameters on the remaining 100 seeds. The selected hyperparameters are reported in Table 5, and the loss curves are shown in Figure 3. As m grows, NorMuon selects larger step sizes. Hence, its loss decreases faster initially but stalls later.

C.2 Setup for image classification

For Muon, NorMuon (with both column and row normalization), and AdamW, we use a step-size schedule with 5% linear warmup followed by cosine decay to 0. Muon and NorMuon use 5 PolarExpress iterations [4]. For AdamW, we sweep the step size over $\{0.0003, 0.001, 0.003, 0.01\}$ with momentum factors $(\beta_1, \beta_2) = (0.9, 0.999)$. For Muon and NorMuon, we sweep the step size over $\{0.03, 0.05, 0.10, 0.15\}$ and the momentum factor over $\{0.5, 0.6, 0.7, 0.8, 0.9, 0.95\}$. NorMuon additionally uses $\beta_2 = 0.9$. For all methods, we sweep the decoupled weight decay over $\{0, 0.01\}$. We use label smoothing with factor 0.2 and clip the Euclidean norm of the gradient to at most 1.

C.3 Setup and additional results for LLM pretraining

We pretrain decoder-only transformers with the nanochat codebase [43] at depths 12, 24, and 32. All models use ReLU^2 activations, root mean square (RMS) normalization [96], rotary positional embeddings [79], query-key normalization [39], untied token embeddings and language-model (LM) heads, and separate value embeddings [102]. Following the standard design of Muon [42], Muon or NorMuon updates only the transformer's weight matrices, while AdamW updates the embeddings, LM head, and other trainable parameters. Table 6 summarizes the model configurations and training token

Table 6: Model configurations and training budgets for LLM pretraining.

Model	Depth	Hidden dim.	MLP dim.	Vocab. size	Seq. length	Matrix params	Batch size (tokens)	Training tokens
286M	12	768	3072	32,768	2048	84.9M	524,288	2.20B
1.38B	24	1536	6144	32,768	2048	679M	1,048,576	14.6B
6.44B	32	2048	8192	131,072	4096	1.61B	1,048,576	19.7B

Table 7: Mean $\pm$ standard deviation of LLM pretraining downstream accuracies and test losses.

Benchmark	Muon	NorMuon
MMLU	**33.3**$_{\pm 0.3}$	**33.3**$_{\pm 0.06}$
HellaSwag	60.3$_{\pm 0.2}$	**60.9**$_{\pm 0.4}$
PIQA	75.9$_{\pm 0.3}$	**76.1**$_{\pm 0.4}$
WinoGrande	55.5$_{\pm 0.4}$	**58.4**$_{\pm 1.5}$
ARC-C	43.8$_{\pm 0.8}$	**45.4**$_{\pm 0.3}$
ARC-E	73.6$_{\pm 0.3}$	**75.0**$_{\pm 0.8}$
BoolQ	**64.4**$_{\pm 2.6}$	62.0$_{\pm 1.7}$
CSQA	**60.0**$_{\pm 1.0}$	59.6$_{\pm 0.5}$
SIQA	49.1$_{\pm 0.1}$	**50.2**$_{\pm 0.03}$
OBQA	47.1$_{\pm 2.8}$	**49.8**$_{\pm 2.1}$
Avg	56.3$_{\pm 0.4}$	**57.1**$_{\pm 0.2}$
Test loss	2.2914$_{\pm 0.0006}$	**2.2810**$_{\pm 0.0009}$

budgets for the experiments in Table 3 and Figure 1. We follow the step-size schedule of nanochat [43]: the step size is warmed up linearly over the first 40 updates, held constant until 35% of training, and then decayed linearly to 5% of its peak value over the remaining 65% of training. The original NorMuon [54] rescales each row as $[\overline{O}_{t+1}]_{i,:} = [O_{t+1}]_{i,:}/(\sqrt{v_{t+1,i}} + \alpha)$ with $\alpha = 10^{-10}$. See Algorithm 1 for details. In our experiments, we instead use $[\overline{O}_{t+1}]_{i,:} = [O_{t+1}]_{i,:}/\sqrt{\max\{v_{t+1,i}, \alpha\}}$ with the same α, following [43].

Random seeds. For the 1.38B model, we run Muon and NorMuon with 3 random seeds and report the mean $\pm$ standard deviation in Table 7. The average improvement of NorMuon over Muon exceeds one standard deviation. Due to space constraints, Table 3 reports only the mean.

Setup for ablation studies. We conduct ablation studies on the 1.38B model trained on the compute-optimal budget of 7.66B tokens. Unless otherwise stated, we use a base step size of 0.02, 5 PolarExpress iterations, a weight decay of 0.02, and Nesterov-type momentum with $\beta_1 = 0.95$.

We next describe the setups for the other optimizers considered in Section 4. For Aurora [22], we use the $K = 2$ variant and apply diagonal refinement to tall matrices. For Muon+, we use the column–row variant, which normalizes columns and then rows after the polar approximation. MuonEq-R instead equilibrates rows before the polar approximation. For the 286M model, we sweep the base step size over $\{0.02, 0.03, 0.04\}$ for all methods, and all of them select 0.03. For the 1.38B model, we use a common base step size of 0.02 for the reasons given below.

Step-size sensitivity under the compute-optimal setting. We compare base step sizes in $\{0.01, 0.02, 0.03, 0.04\}$ for the 1.38B model in Figure 4(a) and Table 8. NorMuon attains lower validation and test losses than Muon at every step size tested. For NorMuon, step sizes 0.01 and 0.02 give nearly identical losses, and the loss increases for larger step sizes. Muon is nearly insensitive to step sizes between 0.01 and 0.03, attains its lowest loss at 0.03, and degrades at 0.04. Even when each optimizer uses its own best step size, NorMuon still attains a lower loss. We therefore use 0.02 as the common step size for the ablation studies, since it lies in a good regime for both optimizers and enables a comparison at a matched step size.

Accuracy of polar approximation. We test whether the benefit of NorMuon persists across polar approximation accuracies by running Muon and NorMuon with 3, 5, or 7 PolarExpress (PE) steps

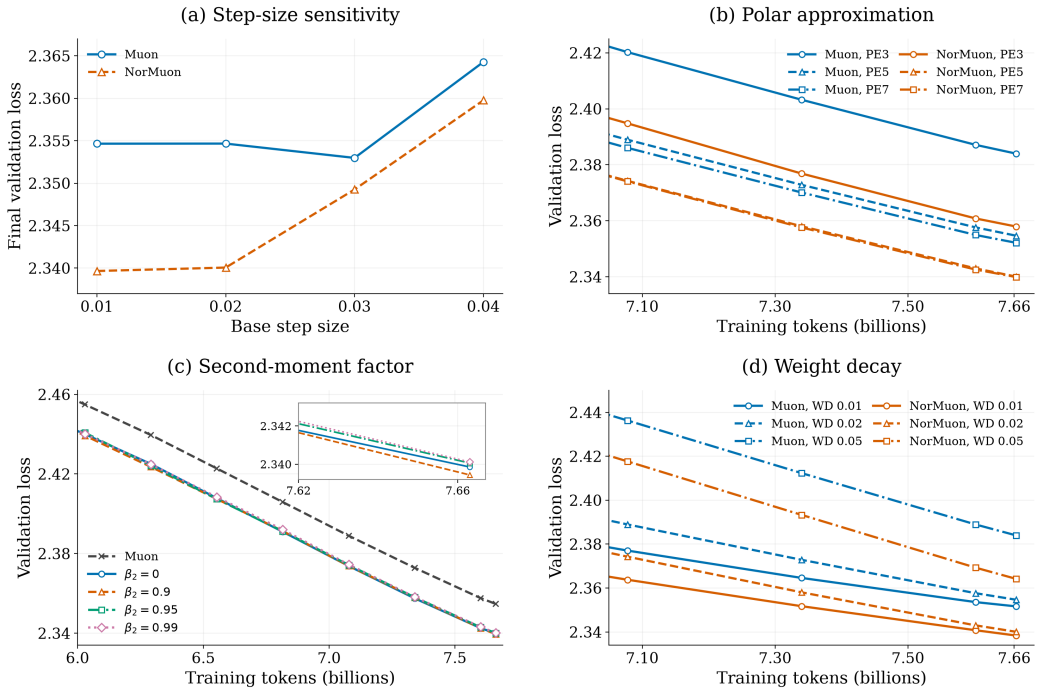

Figure 4: Ablation studies on the 1.38B model trained on 7.66B tokens. (a) Final validation loss versus step size. (b) Muon and NorMuon with 3, 5, and 7 PolarExpress (PE) iterations. (c) NorMuon with different second-moment factors β_2. (d) Muon and NorMuon with different weight decay (WD).

Table 8: Test loss versus base step size for the 1.38B model trained on 7.66B tokens.

Step size	0.01	0.02	0.03	0.04
Muon	2.3587	2.3587	**2.3571**	2.3684
NorMuon	**2.3436**	2.3442	2.3533	2.3637

Table 9: Test loss versus the number of PolarExpress (PE) steps.

PE steps	3	5	7
Muon	2.3882	2.3587	2.3562
NorMuon	2.3620	2.3442	2.3440

Table 10: Test loss versus the second-moment factor β_2.

NorMuon β_2	0	0.9	0.95	0.99	Muon
Test loss	2.3440	2.3435	2.3442	2.3439	2.3587

in Figure 4(b) and Table 9. NorMuon achieves lower validation and test losses than Muon in every setting. Increasing the number of PE steps from 3 to 5 substantially reduces the test loss of both optimizers, while increasing it further to 7 has little effect. The gap between Muon and NorMuon persists across all settings and is largest with 3 PE steps.

Second-moment factor. We compare second-moment factors $\beta_2 \in \{0, 0.9, 0.95, 0.99\}$ for NorMuon in Figure 4(c) and Table 10. The validation and test losses are nearly identical across all values of β_2, outperforming Muon.

Table 11: Test loss versus weight decay.

Weight decay	0.01	0.02	0.05
Muon	2.3557	2.3587	2.3880
NorMuon	2.3424	2.3442	2.3685

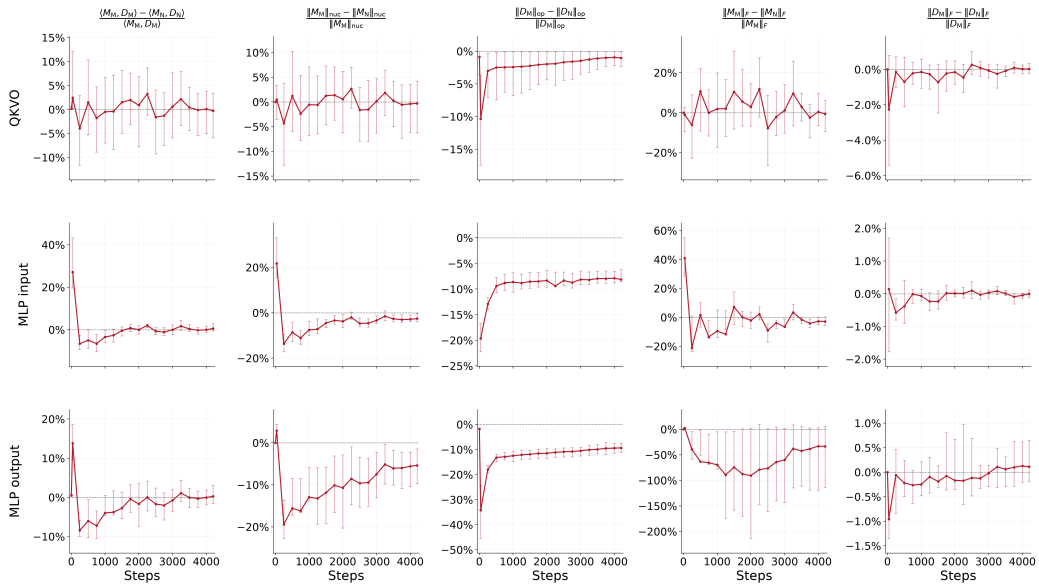

Figure 5: Relative differences between Muon and NorMuon in the alignment and in the norms of the momentum and update direction, for different parameter groups. Error bars show the 10%–90% range across the matrices in each group.

Weight decay. We compare decoupled weight decay factors of 0.01, 0.02, and 0.05 for both Muon and NorMuon in Figure 4(d) and Table 11. NorMuon achieves lower test loss than Muon at every weight decay tested.

Alignment. We compare the alignment $\langle M_{t+1}, D_{t+1}\rangle$ from Lemma B.2 of Muon and NorMuon with Eq. (4.1) on the 286M model. At a fixed time step $t+1$, let M_{M} and M_{N} denote the gradient momenta M_{t+1} of Muon and NorMuon, respectively. The corresponding update directions are

$$D_{\mathrm{M}} = \mathrm{Polar}_{\ell,\delta}\left(\frac{M_{\mathrm{M}}}{\|M_{\mathrm{M}}\|_F}\right), \qquad D_{\mathrm{N}} = \frac{\|O\|_F}{\|\overline{O}\|_F}\overline{O} \ \text{ with } \ O = \mathrm{Polar}_{\ell,\delta}\left(\frac{M_{\mathrm{N}}}{\|M_{\mathrm{N}}\|_F}\right),$$

where $\overline{O}$ denotes the row-normalized version of O, computed as in Algorithm 1. We compare the relative difference in alignment, $\frac{\langle M_{\mathrm{M}}, D_{\mathrm{M}}\rangle - \langle M_{\mathrm{N}}, D_{\mathrm{N}}\rangle}{\langle M_{\mathrm{M}}, D_{\mathrm{M}}\rangle}$, together with the analogous relative differences in $\|M\|_{\mathrm{nuc}}$, $\|D\|_{\mathrm{op}}$, $\|M\|_F$, and $\|D\|_F$ in Figure 5. A negative value indicates that the corresponding quantity is larger for NorMuon. On the MLP matrices, NorMuon tends to attain a slightly larger alignment $\langle M_{\mathrm{N}}, D_{\mathrm{N}}\rangle$ after the first few hundred steps, along with a larger $\|M_{\mathrm{N}}\|_{\mathrm{nuc}}$, while on the QKVO matrices, the two optimizers have similar alignment. This is consistent with the observation of Dewulf et al. [22] that NorMuon mainly affects the MLP matrices. Finally, by the design of Eq. (4.1), the update directions of Muon and NorMuon have similar Frobenius norms, although NorMuon's updates have larger operator norms on the MLP matrices.