An Exploration of Non-Euclidean Gradient Descent: Muon and its Many Variants

Michael Crawshaw
George Mason University

April 22, 2026

Joint work with: Chirag Modi, Mingrui Liu, Robert M. Gower

Bird's Eye View

"Muon is so fast for language model training, but it's so hard to tune!" (Rob, Fall 2024)

Bird's Eye View

"Muon is so fast for language model training, but it's so hard to tune!" (Rob, Fall 2024)

Our motivating idea: if Muon is just GD in another norm, can we apply Polyak-type stepsizes to make Muon less sensitive to the choice of learning rate?

Bird's Eye View

"Muon is so fast for language model training, but it's so hard to tune!" (Rob, Fall 2024)

Our motivating idea: if Muon is just GD in another norm, can we apply Polyak-type stepsizes to make Muon less sensitive to the choice of learning rate?

This led us down a dark rabbit hole containing many variations of the Muon algorithm.

Bird's Eye View

"Muon is so fast for language model training, but it's so hard to tune!" (Rob, Fall 2024)

Our motivating idea: if Muon is just GD in another norm, can we apply Polyak-type stepsizes to make Muon less sensitive to the choice of learning rate?

This led us down a dark rabbit hole containing many variations of the Muon algorithm.

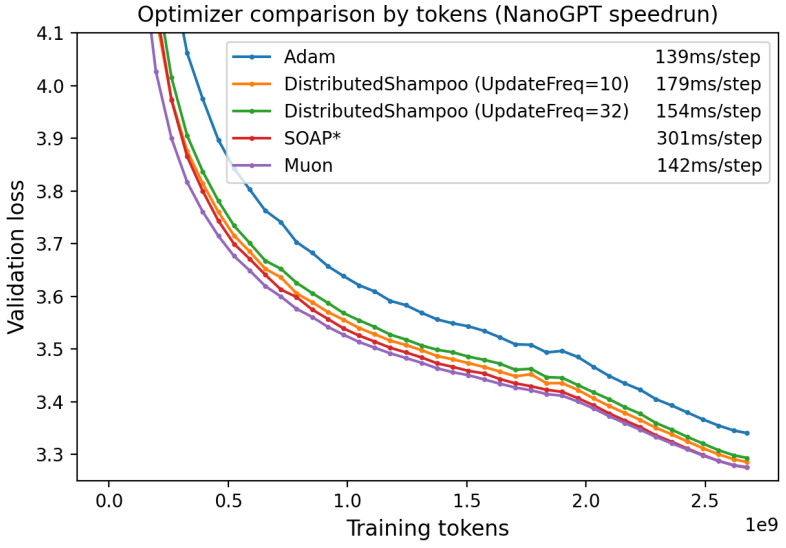
We emerged with:

- A clarified formulation of Muon as a type of steepest descent.
- Polyak-type stepsizes for Muon and its friends.
- A new variant of Muon that is significantly easier to tune.

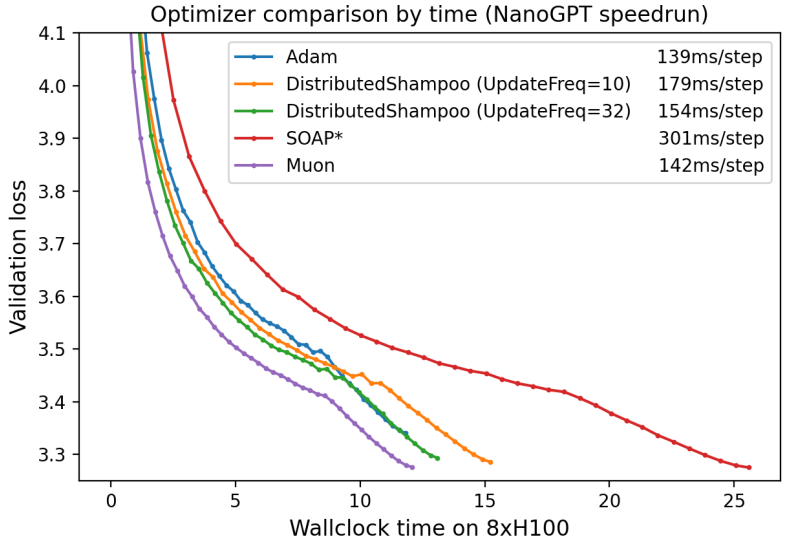
First Look at Muon

What is Muon?

A neural network optimization algorithm (Jordan et al., 2024), quite efficient for training language models.



*SOAP is under active development. Future versions will significantly improve the wallclock overhead.



*SOAP is under active development. Future versions will significantly improve the wallclock overhead.

Muon Definition

Algorithm Muon (for a single weight matrix $\boldsymbol{W}$)

Input: Step size η, momentum β

$\boldsymbol{M}_0 \leftarrow \boldsymbol{0}$
for $t = 0, 1, \ldots, T-1$ **do**
  $\boldsymbol{G}_t \leftarrow$ Stochastic gradient at $\boldsymbol{W}_t$
  $\boldsymbol{M}_t \leftarrow \beta \boldsymbol{M}_{t-1} + (1-\beta)\boldsymbol{G}_t$
  $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V} \leftarrow \mathrm{SVD}(\boldsymbol{M}_t)$
  $\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t - \eta \boldsymbol{U}\boldsymbol{V}^\top$
end for

(Polar factor is approximated by evaluating a certain polynomial at $\boldsymbol{M}_t$ (Jordan et al., 2024; Amsel et al., 2025))

Muon Definition

Algorithm Muon (for a single weight matrix $\boldsymbol{W}$)

Input: Step size η, momentum β

$\boldsymbol{M}_0 \leftarrow \boldsymbol{0}$
for $t = 0, 1, \ldots, T-1$ **do**
  $\boldsymbol{G}_t \leftarrow$ Stochastic gradient at $\boldsymbol{W}_t$
  $\boldsymbol{M}_t \leftarrow \beta \boldsymbol{M}_{t-1} + (1-\beta)\boldsymbol{G}_t$
  $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V} \leftarrow \mathrm{SVD}(\boldsymbol{M}_t)$
  $\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t - \eta \boldsymbol{U}\boldsymbol{V}^\top$
end for

(Polar factor is approximated by evaluating a certain polynomial at $\boldsymbol{M}_t$ (Jordan et al., 2024; Amsel et al., 2025))

Can be interpreted as steepest descent w.r.t. the spectral norm:

$$\begin{aligned}\boldsymbol{W}_{t+1} &= \underset{\|\boldsymbol{W}-\boldsymbol{W}_t\|_2 \le \eta}{\arg\min} \{f(\boldsymbol{W}_t) + \langle \boldsymbol{M}_t, \boldsymbol{W} - \boldsymbol{W}_t \rangle\} \\ &= \boldsymbol{W}_t - \eta \underset{\|\Delta\|_2 \le 1}{\arg\max} \langle \boldsymbol{M}_t, \Delta \rangle.\end{aligned}$$

(using momentum as estimate of full batch gradient $\nabla f(\boldsymbol{W}_t)$)

The Practical Muon

A neural network has many weight matrices and possibly some non-matrix parameters.

The Practical Muon

A neural network has many weight matrices and possibly some non-matrix parameters.

Algorithm 1 MuonAdam: where $\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L$ are the weight matrices, and $\boldsymbol{\theta}$ are all other parameters flattened into a vector.

Inputs: $\boldsymbol{W}_0^1, \ldots, \boldsymbol{W}_0^L, \boldsymbol{\theta}_0$, learning rates η_b, η_m, EMA parameters β, β_1, β_2

for $t = 0, 1, \ldots, T-1$ **do**
  $(\boldsymbol{G}_t^1, \ldots, \boldsymbol{G}_t^L, \boldsymbol{g}_t^\theta) \leftarrow \text{backward}(\boldsymbol{W}_t^1, \ldots \boldsymbol{W}_t^L, \boldsymbol{\theta}_t)$
  for $\ell = 1, \ldots, L$ **do**
    $\boldsymbol{M}_t^\ell = \beta \boldsymbol{M}_{t-1}^\ell + (1-\beta)\boldsymbol{G}_t^\ell$
    $\boldsymbol{W}_{t+1}^\ell \leftarrow \boldsymbol{W}_t^\ell - \eta_m \text{polar}(\boldsymbol{M}_t^\ell)$
  end for
  $\boldsymbol{m}_t^\theta = \beta_1 \boldsymbol{m}_{t-1}^\theta + (1-\beta_1)\boldsymbol{g}_t^\theta$
  $\boldsymbol{v}_t^\theta = \beta_2 \boldsymbol{v}_{t-1}^\theta + (1-\beta_2)\boldsymbol{g}_t^\theta \odot \boldsymbol{g}_t^\theta$
  $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_b \frac{\boldsymbol{m}_t^\theta}{\sqrt{\boldsymbol{v}_t^\theta + \epsilon}}$
end for

The Practical Muon

A neural network has many weight matrices and possibly some non-matrix parameters.

Algorithm 1 MuonAdam: where $\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L$ are the weight matrices, and $\boldsymbol{\theta}$ are all other parameters flattened into a vector.

Inputs: $\boldsymbol{W}_0^1, \ldots, \boldsymbol{W}_0^L, \boldsymbol{\theta}_0$, learning rates η_b, η_m, EMA parameters β, β_1, β_2

for $t = 0, 1, \ldots, T-1$ **do**
  $(\boldsymbol{G}_t^1, \ldots, \boldsymbol{G}_t^L, \boldsymbol{g}_t^\theta) \leftarrow \text{backward}(\boldsymbol{W}_t^1, \ldots \boldsymbol{W}_t^L, \boldsymbol{\theta}_t)$
  for $\ell = 1, \ldots, L$ **do**
    $\boldsymbol{M}_t^\ell = \beta \boldsymbol{M}_{t-1}^\ell + (1-\beta)\boldsymbol{G}_t^\ell$
    $\boldsymbol{W}_{t+1}^\ell \leftarrow \boldsymbol{W}_t^\ell - \eta_m \text{polar}(\boldsymbol{M}_t^\ell)$
  end for
  $\boldsymbol{m}_t^\theta = \beta_1 \boldsymbol{m}_{t-1}^\theta + (1-\beta_1)\boldsymbol{g}_t^\theta$
  $\boldsymbol{v}_t^\theta = \beta_2 \boldsymbol{v}_{t-1}^\theta + (1-\beta_2)\boldsymbol{g}_t^\theta \odot \boldsymbol{g}_t^\theta$
  $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_b \frac{\boldsymbol{m}_t^\theta}{\sqrt{\boldsymbol{v}_t^\theta + \epsilon}}$
end for

Things that bother me compared to "steepest descent" perspective:

1. Why normalize?
2. Jump from one matrix to many matrices.
3. Running side-by-side with Adam (with a different learning rate!)

What We'll Do Today

1. Incorporate the three nuisances into steepest descent (and evaluate alternatives).
2. Use this clarified perspective to port Polyak-type stepsizes.
3. Demonstrate easier tuning for language modeling and image classification.

The Many Variants

Normalize or Not?

Remember our steepest descent formulation:

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \underset{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta}{\arg\min} \{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle\} \\ &= \boldsymbol{w}_t - \eta \underset{\|\Delta\|\leq 1}{\arg\max} \langle \boldsymbol{m}_t, \Delta\rangle. \end{aligned}$$

We enforced $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}_t\| \leq \eta$ to ensure an good linear approximation of f around $\boldsymbol{w}_t$.

Normalize or Not?

Remember our steepest descent formulation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta}{\arg\min}\ \{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle\} \\ &= \boldsymbol{w}_t - \eta \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

We enforced $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}_t\| \leq \eta$ to ensure an good linear approximation of f around $\boldsymbol{w}_t$.

Why not regularize instead?

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\boldsymbol{w}}{\arg\min}\left\{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle + \frac{1}{2\eta}\|\boldsymbol{w}-\boldsymbol{w}_t\|^2\right\} \\ &= \boldsymbol{w}_t - \eta\|\boldsymbol{m}_t\|_* \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

Normalize or Not?

Remember our steepest descent formulation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta}{\arg\min}\ \{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle\} \\ &= \boldsymbol{w}_t - \eta \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

We enforced $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}_t\| \leq \eta$ to ensure an good linear approximation of f around $\boldsymbol{w}_t$.

Why not regularize instead?

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\boldsymbol{w}}{\arg\min}\left\{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle + \frac{1}{2\eta}\|\boldsymbol{w}-\boldsymbol{w}_t\|^2\right\} \\ &= \boldsymbol{w}_t - \eta\|\boldsymbol{m}_t\|_* \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

Choice #1: **Constrain or regularize?**

Normalize or Not?

Remember our steepest descent formulation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta}{\arg\min}\ \{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle\} \\ &= \boldsymbol{w}_t - \eta \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

We enforced $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}_t\| \leq \eta$ to ensure an good linear approximation of f around $\boldsymbol{w}_t$.

Why not regularize instead?

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\boldsymbol{w}}{\arg\min}\left\{f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle + \frac{1}{2\eta}\|\boldsymbol{w}-\boldsymbol{w}_t\|^2\right\} \\ &= \boldsymbol{w}_t - \eta\|\boldsymbol{m}_t\|_* \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle.\end{aligned}$$

Choice #1: **Constrain or regularize?**

$\|\boldsymbol{m}_t\|_* = \max_{\|\Delta\|\leq 1}\langle \boldsymbol{m}_t, \Delta\rangle$ is easy to compute if you know $\arg\max_{\|\Delta\|\leq 1}\langle \boldsymbol{m}_t, \Delta\rangle$.

Beyond a Single Matrix

Neural network weights: $\boldsymbol{W} = (\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L)$

Beyond a Single Matrix

Neural network weights: $\boldsymbol{W} = (\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L)$

Want to minimize $f : \mathcal{W} \to \mathbb{R}$, where $\mathcal{W} = \mathcal{W}^1 \times \ldots \times \mathcal{W}^L$ and $\mathcal{W}^\ell = \mathbb{R}^{h_\ell \times h_{\ell+1}}$

Beyond a Single Matrix

Neural network weights: $\boldsymbol{W} = (\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L)$

Want to minimize $f : \mathcal{W} \to \mathbb{R}$, where $\mathcal{W} = \mathcal{W}^1 \times \ldots \times \mathcal{W}^L$ and $\mathcal{W}^\ell = \mathbb{R}^{h_\ell \times h_{\ell+1}}$

Same formulation with full domain $\mathcal{W}$:

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \eta \arg\max_{\|\boldsymbol{\Delta}\| \leq 1} \langle \boldsymbol{M}_t, \boldsymbol{\Delta} \rangle.$$

Now $\boldsymbol{\Delta} \in \mathcal{W}$.

Beyond a Single Matrix

Neural network weights: $\boldsymbol{W} = (\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L)$

Want to minimize $f : \mathcal{W} \to \mathbb{R}$, where $\mathcal{W} = \mathcal{W}^1 \times \ldots \times \mathcal{W}^L$ and $\mathcal{W}^\ell = \mathbb{R}^{h_\ell \times h_{\ell+1}}$

Same formulation with full domain $\mathcal{W}$:

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \eta \arg\max_{\|\boldsymbol{\Delta}\| \leq 1} \langle \boldsymbol{M}_t, \boldsymbol{\Delta} \rangle.$$

Now $\boldsymbol{\Delta} \in \mathcal{W}$.

Need a norm on $\mathcal{W}$!

Beyond a Single Matrix

Neural network weights: $\boldsymbol{W} = (\boldsymbol{W}^1, \ldots, \boldsymbol{W}^L)$

Want to minimize $f : \mathcal{W} \to \mathbb{R}$, where $\mathcal{W} = \mathcal{W}^1 \times \ldots \times \mathcal{W}^L$ and $\mathcal{W}^\ell = \mathbb{R}^{h_\ell \times h_{\ell+1}}$

Same formulation with full domain $\mathcal{W}$:

$$\boldsymbol{W}_{t+1} = \boldsymbol{W}_t - \eta \arg\max_{\|\boldsymbol{\Delta}\| \leq 1} \langle \boldsymbol{M}_t, \boldsymbol{\Delta} \rangle.$$

Now $\boldsymbol{\Delta} \in \mathcal{W}$.

Need a norm on $\mathcal{W}$!

Choice #2: **A norm on the product space $\mathcal{W}$.**

Outer Norm Examples

Some natural choices:

$$\|\boldsymbol{W}\| = \max_{\ell \in [L]} \|\boldsymbol{W}\|_2, \quad \|\boldsymbol{W}\| = \sqrt{\sum_{\ell=1}^{L} \|\boldsymbol{W}\|_2^2}.$$

Outer Norm Examples

Some natural choices:

$$\|\boldsymbol{W}\| = \max_{\ell \in [L]} \|\boldsymbol{W}\|_2, \quad \|\boldsymbol{W}\| = \sqrt{\sum_{\ell=1}^{L} \|\boldsymbol{W}\|_2^2}.$$

Leads to different update scaling across layers:

$\|\cdot\|_\infty$, Constrained:	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \mathsf{polar}(\boldsymbol{M}_t^\ell)$
$\|\cdot\|_\infty$, Regularized:	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\mathsf{nuc}} \right) \mathsf{polar}(\boldsymbol{M}_t^\ell)$
$\|\cdot\|_2$, Constrained:	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \frac{\|\boldsymbol{M}_t^\ell\|_{\mathsf{nuc}}}{\sqrt{\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\mathsf{nuc}}^2}} \mathsf{polar}(\boldsymbol{M}_t^\ell)$
$\|\cdot\|_2$, Regularized:	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \|\boldsymbol{M}_t^\ell\|_{\mathsf{nuc}} \mathsf{polar}(\boldsymbol{M}_t^\ell)$

Computational Issues of Non-Separable Updates

Design Choices	Layer ℓ Update	Separable
$\|\cdot\|_\infty$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\text{nuc}} \right) \text{polar}(\boldsymbol{M}_t^\ell)$	✗
$\|\cdot\|_2$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \|\boldsymbol{M}_t^\ell\|_{\text{nuc}} \text{polar}(\boldsymbol{M}_t^\ell)$	✓

Computational Issues of Non-Separable Updates

Design Choices	Layer ℓ Update	Separable
$\|\cdot\|_\infty$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\text{nuc}} \right) \text{polar}(\boldsymbol{M}_t^\ell)$	✗
$\|\cdot\|_2$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \|\boldsymbol{M}_t^\ell\|_{\text{nuc}} \text{polar}(\boldsymbol{M}_t^\ell)$	✓

If separable, can execute update with a single pass over layers without any extra storage.

Computational Issues of Non-Separable Updates

Design Choices	Layer ℓ Update	Separable
$\|\cdot\|_\infty$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\text{nuc}} \right) \text{polar}(\boldsymbol{M}_t^\ell)$	✗
$\|\cdot\|_2$, Regularized	$\boldsymbol{W}_{t+1}^\ell = \boldsymbol{W}_t^\ell - \eta \|\boldsymbol{M}_t^\ell\|_{\text{nuc}} \text{polar}(\boldsymbol{M}_t^\ell)$	✓

If separable, can execute update with a single pass over layers without any extra storage.

If non-separable, need two passes over layers and one of the following:

- Store polar factors across loop steps (extra memory).
- Recompute polar factors during second pass (extra time).

A Simple Trick: Stale Dual Norms

Approximation: store nuclear norms from previous step, use this to compute step size.

$$\boldsymbol{W}_{t+1}^{\ell} = \boldsymbol{W}_{t}^{\ell} - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_{t}^{j}\|_{\mathsf{nuc}} \right) \mathsf{polar}(\boldsymbol{M}_{t}^{\ell})$$

$$\downarrow$$

$$\boldsymbol{W}_{t+1}^{\ell} = \boldsymbol{W}_{t}^{\ell} - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_{t-1}^{j}\|_{\mathsf{nuc}} \right) \mathsf{polar}(\boldsymbol{M}_{t}^{\ell})$$

A Simple Trick: Stale Dual Norms

Approximation: store nuclear norms from previous step, use this to compute step size.

$$\boldsymbol{W}_{t+1}^{\ell} = \boldsymbol{W}_t^{\ell} - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_t^j\|_{\text{nuc}} \right) \text{polar}(\boldsymbol{M}_t^{\ell})$$

$$\downarrow$$

$$\boldsymbol{W}_{t+1}^{\ell} = \boldsymbol{W}_t^{\ell} - \eta \left(\sum_{j=1}^{L} \|\boldsymbol{M}_{t-1}^j\|_{\text{nuc}} \right) \text{polar}(\boldsymbol{M}_t^{\ell})$$

- Similar time/memory overhead as original Muon.
- Very small change in performance compared to non-stale version.

Incorporating Non-Matrix Parameters

Initially, two side-by-side optimizers seems to break our steepest descent formulation.

Incorporating Non-Matrix Parameters

Initially, two side-by-side optimizers seems to break our steepest descent formulation.

Actually, even this can be written as steepest descent for a certain norm.

Incorporating Non-Matrix Parameters

Initially, two side-by-side optimizers seems to break our steepest descent formulation.

Actually, even this can be written as steepest descent for a certain norm.

Let $\boldsymbol{W} = (\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L, \boldsymbol{\theta})$ be the parameters of a neural network, where $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$ are weight matrices, and $\boldsymbol{\theta} \in \mathbb{R}^k$ holds all other parameters.

MuonAdam (Algorithm 1) is constrained steepest descent with stepsize η_m w.r.t.

$$\|\boldsymbol{W}\|_{\mathsf{muon}} := \max\left(\max_{1\le\ell\le L} \|\boldsymbol{W}^\ell\|_{2\to 2}, \frac{\eta_m}{\eta_b}\|\boldsymbol{\theta}\|_{\mathsf{ada}\infty}\right)$$

$$\|\boldsymbol{\theta}\|_{\mathsf{ada}\infty} := \left\| \frac{\sqrt{\boldsymbol{v}_t} + \epsilon}{|\boldsymbol{m}_t|} \odot \boldsymbol{\theta} \right\|_\infty .$$

Incorporating Non-Matrix Parameters

Initially, two side-by-side optimizers seems to break our steepest descent formulation.

Actually, even this can be written as steepest descent for a certain norm.

Let $\boldsymbol{W} = (\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L, \boldsymbol{\theta})$ be the parameters of a neural network, where $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$ are weight matrices, and $\boldsymbol{\theta} \in \mathbb{R}^k$ holds all other parameters.

MuonAdam (Algorithm 1) is constrained steepest descent with stepsize η_m w.r.t.

$$\|\boldsymbol{W}\|_{\text{muon}} := \max\left(\max_{1\le\ell\le L} \|\boldsymbol{W}^\ell\|_{2\to 2}, \frac{\eta_m}{\eta_b}\|\boldsymbol{\theta}\|_{\text{ada}\infty}\right)$$

$$\|\boldsymbol{\theta}\|_{\text{ada}\infty} := \left\| \frac{\sqrt{\boldsymbol{v}_t} + \epsilon}{|\boldsymbol{m}_t|} \odot \boldsymbol{\theta} \right\|_\infty .$$

Other variations: Can use $\|\cdot\|_\infty$ (sign GD), $\|\cdot\|_2$ (GD), or other norms for $\boldsymbol{\theta}$.

Incorporating Non-Matrix Parameters

Initially, two side-by-side optimizers seems to break our steepest descent formulation.

Actually, even this can be written as steepest descent for a certain norm.

Let $\boldsymbol{W} = (\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L, \boldsymbol{\theta})$ be the parameters of a neural network, where $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$ are weight matrices, and $\boldsymbol{\theta} \in \mathbb{R}^k$ holds all other parameters.

MuonAdam (Algorithm 1) is constrained steepest descent with stepsize η_m w.r.t.

$$\|\boldsymbol{W}\|_{\text{muon}} := \max\left(\max_{1\le\ell\le L} \|\boldsymbol{W}^\ell\|_{2\to 2}, \frac{\eta_m}{\eta_b}\|\boldsymbol{\theta}\|_{\text{ada}\infty}\right)$$

$$\|\boldsymbol{\theta}\|_{\text{ada}\infty} := \left\| \frac{\sqrt{\boldsymbol{v}_t} + \epsilon}{|\boldsymbol{m}_t|} \odot \boldsymbol{\theta} \right\|_\infty .$$

Other variations: Can use $\|\cdot\|_\infty$ (sign GD), $\|\cdot\|_2$ (GD), or other norms for $\boldsymbol{\theta}$.

Design Choice #3: **Choose a norm for non-matrix parameters $\boldsymbol{\theta}$.**

The Story So Far

Optimizer design choices:

1. Constrained or regularized steepest descent.
2. Outer norm to aggregate norms of each layer.
3. Norm for non-matrix parameters.

The Story So Far

Optimizer design choices:

1. Constrained or regularized steepest descent.
2. Outer norm to aggregate norms of each layer.
3. Norm for non-matrix parameters.

We have removed the nuisances, and written MuonAdam as a steepest descent!

The Story So Far

Optimizer design choices:

1. Constrained or regularized steepest descent.
2. Outer norm to aggregate norms of each layer.
3. Norm for non-matrix parameters.

We have removed the nuisances, and written MuonAdam as a steepest descent!

...but why?

The Story So Far

Optimizer design choices:

1. Constrained or regularized steepest descent.
2. Outer norm to aggregate norms of each layer.
3. Norm for non-matrix parameters.

We have removed the nuisances, and written MuonAdam as a steepest descent!

...but why?

Allows for systematic evaluation of Muon variants (numerics later).

The Story So Far

Optimizer design choices:

1. Constrained or regularized steepest descent.
2. Outer norm to aggregate norms of each layer.
3. Norm for non-matrix parameters.

We have removed the nuisances, and written MuonAdam as a steepest descent!

...but why?

Allows for systematic evaluation of Muon variants (numerics later).

Can return to our original motivation: extending GD techniques (particularly Momo (Schaipp et al., 2024)) for Muon and friends.

Model Truncation

Adaptive Stepsizes with Model Truncation

Momo (Schaipp et al., 2024) is a type of stochastic Polyak stepsize, which utilizes momentum and knowledge of a loss lower bound f_*.

Adaptive Stepsizes with Model Truncation

Momo (Schaipp et al., 2024) is a type of stochastic Polyak stepsize, which utilizes momentum and knowledge of a loss lower bound f_*.

In a nutshell: Do not trust gradients that promise to make loss smaller than f_*.

Adaptive Stepsizes with Model Truncation

Momo (Schaipp et al., 2024) is a type of stochastic Polyak stepsize, which utilizes momentum and knowledge of a loss lower bound f_*.

In a nutshell: Do not trust gradients that promise to make loss smaller than f_*.

At each step, locally minimize a truncated linear approximation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \operatorname*{arg\,min}_{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta} \{\max\left(f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle, f_*\right)\} \\ &= \boldsymbol{w}_t - \min\left(\eta, \frac{f(\boldsymbol{w}_t)-f_*}{\|\boldsymbol{m}_t\|_*}\right) \operatorname*{arg\,max}_{\|\Delta\|\leq 1} \langle \boldsymbol{m}_t, \Delta\rangle\end{aligned}$$

Adaptive Stepsizes with Model Truncation

Momo (Schaipp et al., 2024) is a type of stochastic Polyak stepsize, which utilizes momentum and knowledge of a loss lower bound f_*.

In a nutshell: Do not trust gradients that promise to make loss smaller than f_*.

At each step, locally minimize a truncated linear approximation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \underset{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta}{\arg\min}\ \{\max\left(f(\boldsymbol{w}_t)+\langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle, f_*\right)\} \\ &= \boldsymbol{w}_t - \min\left(\eta, \frac{f(\boldsymbol{w}_t)-f_*}{\|\boldsymbol{m}_t\|_*}\right) \underset{\|\Delta\|\leq 1}{\arg\max}\langle \boldsymbol{m}_t, \Delta\rangle\end{aligned}$$

If learning rate η is too big, update size automatically adapts!

Adaptive Stepsizes with Model Truncation

Momo (Schaipp et al., 2024) is a type of stochastic Polyak stepsize, which utilizes momentum and knowledge of a loss lower bound f_*.

In a nutshell: Do not trust gradients that promise to make loss smaller than f_*.

At each step, locally minimize a truncated linear approximation:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \operatorname*{arg\,min}_{\|\boldsymbol{w}-\boldsymbol{w}_t\|\leq\eta} \{\max\left(f(\boldsymbol{w}_t) + \langle \boldsymbol{m}_t, \boldsymbol{w}-\boldsymbol{w}_t\rangle, f_*\right)\} \\ &= \boldsymbol{w}_t - \min\left(\eta, \frac{f(\boldsymbol{w}_t)-f_*}{\|\boldsymbol{m}_t\|_*}\right) \operatorname*{arg\,max}_{\|\Delta\|\leq 1} \langle \boldsymbol{m}_t, \Delta\rangle\end{aligned}$$

If learning rate η is too big, update size automatically adapts!

Can't just drop $f(\boldsymbol{w}_t)$ anymore. But we can estimate it similarly as momentum estimates gradient.

Adaptive Stepsizes with Model Truncation

With momentum, $\boldsymbol{m}_t$ is a convex combination of previous stochastic gradients:

$$\boldsymbol{m}_t = \sum_{i=1}^{t} \rho_{t,i} \boldsymbol{g}_i, \quad \sum_{i=1}^{t} \rho_{t,i} = 1.$$

Adaptive Stepsizes with Model Truncation

With momentum, $\boldsymbol{m}_t$ is a convex combination of previous stochastic gradients:

$$\boldsymbol{m}_t = \sum_{i=1}^{t} \rho_{t,i} \boldsymbol{g}_i, \quad \sum_{i=1}^{t} \rho_{t,i} = 1.$$

Can similarly estimate $f(\boldsymbol{w}_t)$ as a convex combination of first-order approximations:

$$f(\boldsymbol{w}_t) \approx \tilde{f}_t := \sum_{i=0}^{t} \rho_{t,i} \left(f_i(\boldsymbol{w}_i) + \langle \boldsymbol{g}_i, \boldsymbol{w}_t - \boldsymbol{w}_i \rangle \right).$$

Adaptive Stepsizes with Model Truncation

With momentum, $\boldsymbol{m}_t$ is a convex combination of previous stochastic gradients:

$$\boldsymbol{m}_t = \sum_{i=1}^{t} \rho_{t,i} \boldsymbol{g}_i, \quad \sum_{i=1}^{t} \rho_{t,i} = 1.$$

Can similarly estimate $f(\boldsymbol{w}_t)$ as a convex combination of first-order approximations:

$$f(\boldsymbol{w}_t) \approx \tilde{f}_t := \sum_{i=0}^{t} \rho_{t,i} \left(f_i(\boldsymbol{w}_i) + \langle \boldsymbol{g}_i, \boldsymbol{w}_t - \boldsymbol{w}_i \rangle \right).$$

Resulting update:

$$\tilde{F}_t = \beta \tilde{F}_{t-1} + (1-\beta)(f_t(\boldsymbol{w}_t) - \langle \boldsymbol{g}_t, \boldsymbol{w}_t \rangle), \quad \tilde{f}_t = \tilde{F}_t + \langle \boldsymbol{m}_t, \boldsymbol{w}_t \rangle$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \min\left(\eta, \frac{\tilde{f}_t - f_*}{\|\boldsymbol{m}_t\|_*}\right) \arg\max_{\|\Delta\| \leq 1} \langle \boldsymbol{m}_t, \Delta \rangle.$$

(originally designed for SGD/Adam, but extends naturally to any norm)

Momo for Muon

Now we have Momo for any steepest descent!

$$\text{Constrained :} \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \min\left(\eta, \frac{\tilde{f}_t - f_*}{\|\boldsymbol{m}_t\|_*}\right) \arg\max_{\|\Delta\| \leq 1} \langle \boldsymbol{m}_t, \Delta \rangle$$

$$\text{Regularized :} \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \min\left(\eta, \frac{\tilde{f}_t - f_*}{\|\boldsymbol{m}_t\|_*^2}\right) \|\boldsymbol{m}_t\|_* \arg\max_{\|\Delta\| \leq 1} \langle \boldsymbol{m}_t, \Delta \rangle.$$

Momo for Muon

Now we have Momo for any steepest descent!

$$\text{Constrained :} \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \min\left(\eta, \frac{\tilde{f}_t - f_*}{\|\boldsymbol{m}_t\|_*}\right) \arg\max_{\|\Delta\| \leq 1} \langle \boldsymbol{m}_t, \Delta \rangle$$

$$\text{Regularized :} \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \min\left(\eta, \frac{\tilde{f}_t - f_*}{\|\boldsymbol{m}_t\|_*^2}\right) \|\boldsymbol{m}_t\|_* \arg\max_{\|\Delta\| \leq 1} \langle \boldsymbol{m}_t, \Delta \rangle.$$

Using that MuonAdam is constrained steepest descent w.r.t. $\|\cdot\|_{\text{muon}}$, we finally have MuonAdam-Momo:

$$\begin{aligned}
d_t &= \textstyle\sum_{\ell=1}^{L} \|\boldsymbol{M}_t^\ell\|_{\text{nuc}} + \frac{\eta_b}{\eta_m} \left\| \frac{|\boldsymbol{m}_t|}{\sqrt{\boldsymbol{v}_t}+\epsilon} \odot \boldsymbol{m}_t^\theta \right\|_1 \\
\eta_m' &= \min\left\{\eta_m, \tfrac{\tilde{f}_t - f_*}{d_t}\right\}, \;\; \eta_b' = \min\left\{\eta_b, \tfrac{\eta_b}{\eta_m} \tfrac{\tilde{f}_t - f_*}{d_t}\right\} \\
\boldsymbol{W}_{t+1}^\ell &= \boldsymbol{W}_t^\ell - \eta_m' \,\text{polar}(\boldsymbol{M}_t^\ell) \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta_b' \frac{\boldsymbol{m}_t^\theta}{\sqrt{\boldsymbol{v}_t^\theta}+\epsilon}.
\end{aligned}$$

A New Algorithm

A new norm falls from the sky:

$$\|\boldsymbol{W}\|_{\mathsf{MM}} = \sqrt{\left(\max_{\ell \in [L]} \|\boldsymbol{W}^{\ell}\|_2^2\right)^2 + \frac{\eta_m}{\eta_b} \left\| \sqrt{\sqrt{\boldsymbol{v}_t} + \epsilon} \odot \boldsymbol{\theta} \right\|_2^2}.$$

(we originally liked this one for the aesthetics of the resulting steepest descent)

A New Algorithm

A new norm falls from the sky:

$$\|\boldsymbol{W}\|_{\mathsf{MM}} = \sqrt{\left(\max_{\ell\in[L]} \|\boldsymbol{W}^\ell\|_2^2\right)^2 + \frac{\eta_m}{\eta_b}\left\|\sqrt{\sqrt{\boldsymbol{v}_t}+\epsilon}\odot\boldsymbol{\theta}\right\|_2^2}.$$

(we originally liked this one for the aesthetics of the resulting steepest descent)

MuonMax-Momo (regularized steepest descent w.r.t. $\|\cdot\|_{\mathsf{MM}}$, plus Momo):

$$\begin{aligned}
d_t &= \sqrt{\Big(\textstyle\sum_{\ell=1}^L \|\boldsymbol{M}_t^\ell\|_{\mathsf{nuc}}\Big)^2 + \frac{\eta_b}{\eta_m}\Big\|\frac{\boldsymbol{m}_t^\theta}{\sqrt{\sqrt{\boldsymbol{v}_t^\theta}+\epsilon}}\Big\|_2^2}\\
\eta_m' &= \min\left\{\eta_m, \tfrac{\tilde{f}_t - f_*}{d_t^2}\right\},\ \eta_b' = \min\left\{\eta_b, \tfrac{\eta_b}{\eta_m}\tfrac{\tilde{f}_t - f_*}{d_t^2}\right\}\\
\boldsymbol{W}_{t+1}^\ell &= \boldsymbol{W}_t^\ell - \eta_m'\Big(\textstyle\sum_{\ell=1}^L \|\boldsymbol{M}_t^\ell\|_{\mathsf{nuc}}\Big)\mathsf{polar}(\boldsymbol{M}_t^\ell)\\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta_b'\frac{\boldsymbol{m}_t^\theta}{\sqrt{\boldsymbol{v}_t^\theta}+\epsilon}.
\end{aligned}$$

Experiments

Testing Many Variations

Three design choices in our framework lead to many variations, all based on the same principles. Which ones work well for deep learning?

Testing Many Variations

Three design choices in our framework lead to many variations, all based on the same principles. Which ones work well for deep learning?

Training GPT-2 (124M params) for language modeling with FineWeb data (1B tokens):

(SD type, Outer Norm, Backup Norm)	Muon LR	Other LR	Final Loss	Name
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-3	1e-5	3.783	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.599	Scion
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-6	4.179	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-2	3.712	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-3	1e-5	3.826	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.610	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	1e-3	1e-5	3.859	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	1e-2	1e-3	3.604	Muon
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-4	4.229	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-2	3.748	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	1e-3	1e-4	3.917	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	1e-2	1e-2	3.628	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-3	1e-4	3.757	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-3	3.701	-
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	1e-1	1e-3	4.049	PolarGrad
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	1e-1	1e-2	3.664	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-3	1e-3	3.791	MuonMax
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-2	3.585	-

Testing Many Variations

Three design choices in our framework lead to many variations, all based on the same principles. Which ones work well for deep learning?

Training GPT-2 (124M params) for language modeling with FineWeb data (1B tokens):

(SD type, Outer Norm, Backup Norm)	Muon LR	Other LR	Final Loss	Name
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-3	1e-5	3.783	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.599	Scion
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-6	4.179	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-2	3.712	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-3	1e-5	3.826	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.610	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	1e-3	1e-5	3.859	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	1e-2	1e-3	3.604	Muon
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-4	4.229	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-2	3.748	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	1e-3	1e-4	3.917	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	1e-2	1e-2	3.628	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-3	1e-4	3.757	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-3	3.701	-
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	1e-1	1e-3	4.049	PolarGrad
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	1e-1	1e-2	3.664	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-3	1e-3	3.791	MuonMax
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-2	3.585	-

Regularized variants lag behind. Aligns with convention: need normalization or clipping.

Testing Many Variations (w/ Momo)

Same setup, using Momo:

(SD type, Outer Norm, Backup Norm)	Muon LR	Other LR	Final Loss	Name
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-4	3.627	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.592	Scion-Momo
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1	1e-5	3.728	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-2	3.843	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-2	1e-4	3.628	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	3e-2	3e-3	3.604	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-4	3.578	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-3	3.551	Muon-Momo
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1	1e-3	3.719	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-2	3.737	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-3	3.584	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-2	3.607	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	3e-3	3e-4	3.662	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-3	3.701	-
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	3	3e-2	3.613	PolarGrad-Momo
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	3e-1	3e-2	3.602	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-2	3.576	MuonMax-Momo
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	3e-2	3e-2	3.553	-

Testing Many Variations (w/ Momo)

Same setup, using Momo:

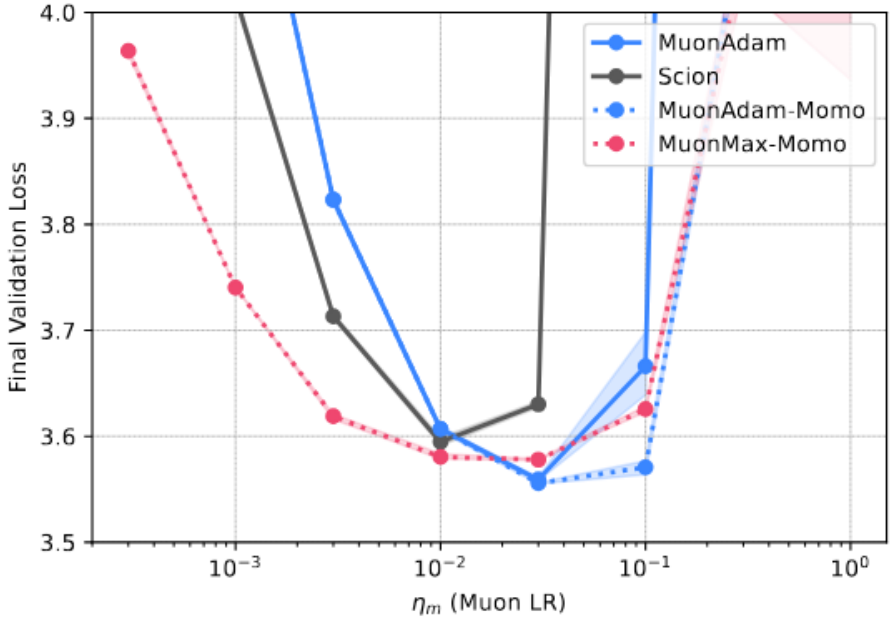
(SD type, Outer Norm, Backup Norm)	Muon LR	Other LR	Final Loss	Name
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-4	3.627	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_\infty$)	1e-2	1e-3	3.592	Scion-Momo
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1	1e-5	3.728	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_\infty$)	1e-1	1e-2	3.843	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	1e-2	1e-4	3.628	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_\infty$)	3e-2	3e-3	3.604	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-4	3.578	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-3	3.551	Muon-Momo
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1	1e-3	3.719	-
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada}\infty}$)	1e-1	1e-2	3.737	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-3	3.584	-
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada}\infty}$)	3e-2	3e-2	3.607	-
(Regularized, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	3e-3	3e-4	3.662	-
(Constrained, $\|\cdot\|_\infty$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-3	3.701	-
(Regularized, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	3	3e-2	3.613	PolarGrad-Momo
(Constrained, $\|\cdot\|_2$, $\|\cdot\|_{\text{ada2}}$)	3e-1	3e-2	3.602	-
(Regularized, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	1e-2	1e-2	3.576	MuonMax-Momo
(Constrained, $\|\cdot\|_{\text{hyb}}$, $\|\cdot\|_{\text{ada2}}$)	3e-2	3e-2	3.553	-

Momo greatly helps regularized variants.

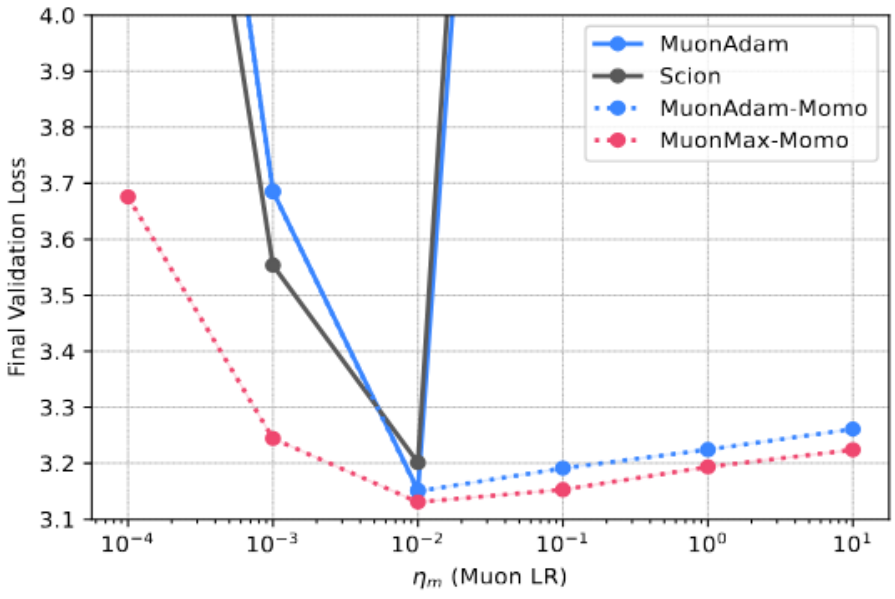
Muon-Momo achieves lowest loss (small margin).

Tuning Sensitivity

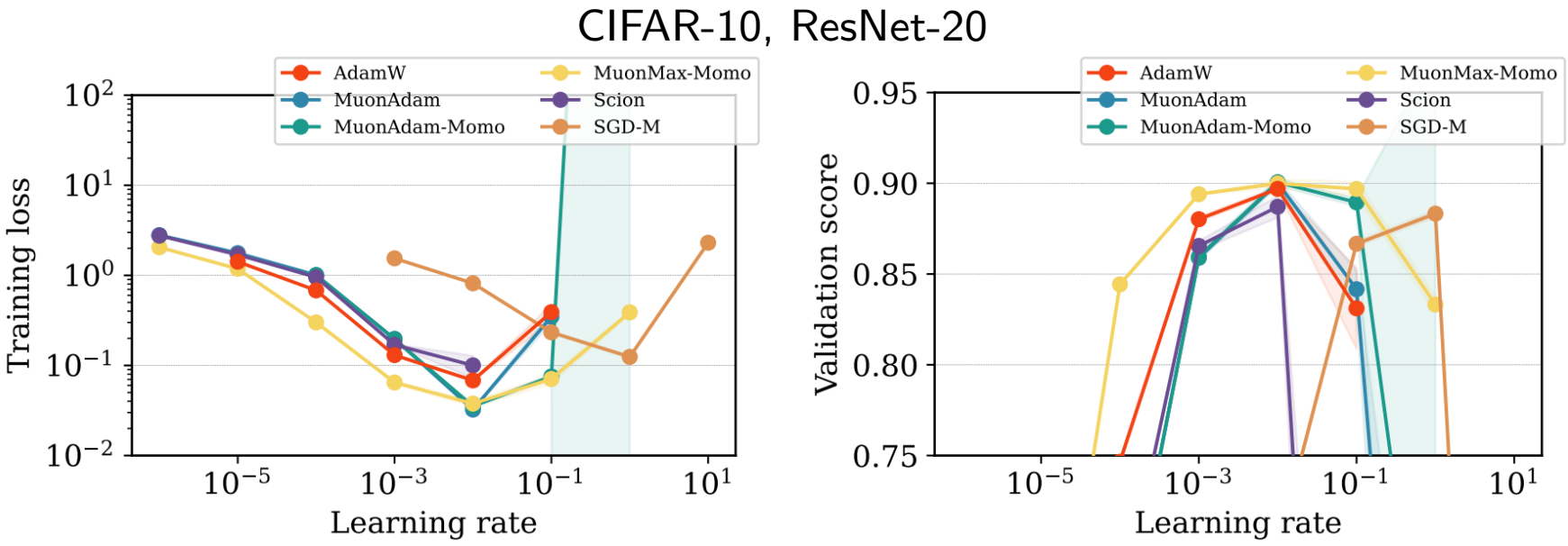
FineWeb1B, GPT2-Small



SlimPajama1B, GPT2-Large



Tuning Sensitivity



Probing Practical Behavior

Wall-clock time (GPT2-Small, FineWeb1B)

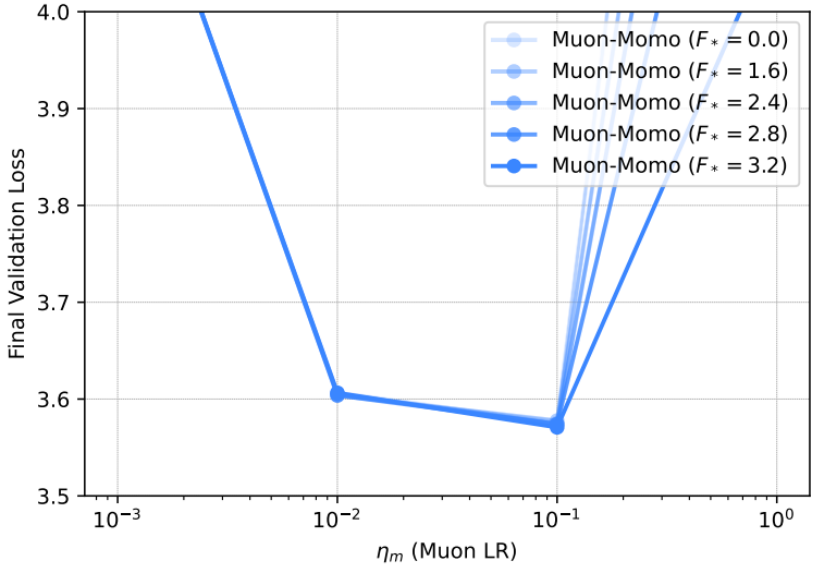
	MuonAdam	MuonMax	MuonAdam-Momo	Scion-Momo	MuonMax-Momo
Original	3.604 (1×)	3.791 (1.09×)	3.551 (1.10×)	3.592 (1.08×)	3.576 (1.11×)
Stale	-	3.768 (1.04×)	3.554 (1.04×)	3.590 (1.02×)	3.580 (1.05×)

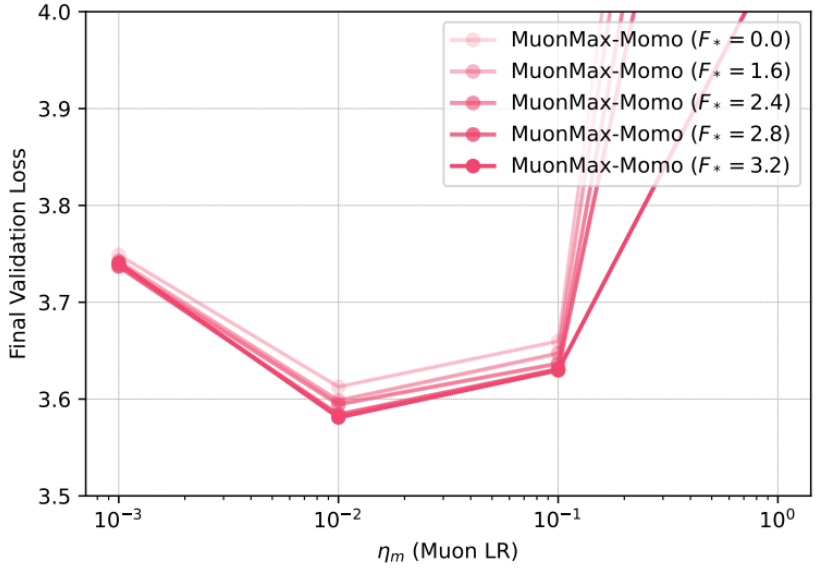
Probing Practical Behavior

Wall-clock time (GPT2-Small, FineWeb1B)

	MuonAdam	MuonMax	MuonAdam-Momo	Scion-Momo	MuonMax-Momo
Original	3.604 (1×)	3.791 (1.09×)	3.551 (1.10×)	3.592 (1.08×)	3.576 (1.11×)
Stale	-	3.768 (1.04×)	3.554 (1.04×)	3.590 (1.02×)	3.580 (1.05×)

Sensitivity to choice of f_* (GPT2-Small, FineWeb1B):





Conclusion

Conclusion

Usual derivation of Muon leaves out a lot of details which point in new directions.

Conclusion

Usual derivation of Muon leaves out a lot of details which point in new directions.

We formalized MuonAdam as a steepest descent on the space of all neural network parameters.

Conclusion

Usual derivation of Muon leaves out a lot of details which point in new directions.

We formalized MuonAdam as a steepest descent on the space of all neural network parameters.

This formalization enables immediate application of GD techniques to Muon and friends, e.g. Momo stepsizes for easier tuning.

Conclusion

Usual derivation of Muon leaves out a lot of details which point in new directions.

We formalized MuonAdam as a steepest descent on the space of all neural network parameters.

This formalization enables immediate application of GD techniques to Muon and friends, e.g. Momo stepsizes for easier tuning.

Future directions:

- Why do some of our variations work better than others?
- Why does Muon work better than SGD and Adam???

Thanks!

References

Amsel, N., Persson, D., Musco, C., and Gower, R. (2025). The polar express: Optimal matrix sign methods and their application to the muon algorithm. *arXiv preprint arXiv:2505.16932*.

Jordan, K., Jin, Y., Boza, V., You, J., Cesista, F., Newhouse, L., and Bernstein, J. (2024). Muon: An optimizer for hidden layers in neural networks.

Schaipp, F., Ohana, R., Eickenberg, M., Defazio, A., and Gower, R. M. (2024). Momo: Momentum models for adaptive learning rates. In *International Conference on Machine Learning*, pages 43542–43570. PMLR.