The Ky Fan Norms and Beyond: Dual Norms and Combinations for Matrix Optimization

Introduction

Training large language models requires optimizers more scalable and efficient than Adam/AdamW. The recent Muon [2] optimizer shows great promise, sometimes doubling AdamW's efficiency, due to its unique design:

- **Matrix-Aware Design:** It is constructed specifically for weight matrices, not generic vectors.
- **Principled Update Rule:** Muon's update is not merely heuristic. It is derived as the solution to a linear minimization problem constrained by the operator (spectral) norm, giving it a solid mathematical foundation.

Muon's success has inspired variants like Dion [1] algorithm and frameworks of Scion and Gluon [4]. We build on this work by asking a fundamental question:

> In deriving Muon's update step, why constrain by the spectral norm? How would alternative norms affect performance and computational cost?

To investigate this, we first analyze the link between Muon-like algorithms and their defining norms. We then propose **F-Fanions**, a novel family of optimizers derived from exotic, mixed norms. Finally, we present an empirical comparison between Muon and F-Fanions across various numerical experiments.

How Norms Shape the Update Step

Many optimizers are defined by a norm-constrained Linear Minimization Oracle (LMO) [3], which computes the update $\Delta\mathbf{X}^t$ based on the gradient (possibly stochastic / with momentum) $\mathbf{G}^t$:

$$\Delta X^t = \mathbf{X}^{t+1} - \mathbf{X}^t \in \eta \arg\min_{\|\mathbf{X}\|\leq 1} \langle \mathbf{G}^t, \mathbf{X} \rangle .$$

The choice of norm dictates the algorithm. For matrix methods, the update often uses the Singular Value Decomposition (SVD) of the gradient, $\mathbf{G}^t = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$.

Case	Method	Constraining Norm	Update Formula
Vec.	Normalized SGD	ℓ_2	$-\eta g/\|g\|_2$
	SignSGD	ℓ_∞	$-\eta\,\mathbf{sign}(g)$
	Coordinate Descent	ℓ_1	$-\eta e_{\text{argmax}\|g_i\|}$
Mat.	Normalized SGD	$\|\cdot\|_{\text{F}}$	$-\eta\mathbf{G}^t/\|\mathbf{G}^t\|_{\text{F}}$
	Muon	$\|\cdot\|_{\text{op}}$	$-\eta\mathbf{U}\mathbf{V}^\top$
	Dion (no feedback)	$\|\cdot\|^\dagger_{KF-k}$	$-\eta\mathbf{U}_k\mathbf{V}_k^\top$

We introduce a generalized norm that unifies and interpolates between these methods:

$$\|\mathbf{X}\|_{\text{gen}} = \left(\alpha\|\mathbf{X}\|_{\text{KF}-k} + (1-\alpha)\|\mathbf{X}\|_{\text{F}}\right)^\dagger .$$

This defines the **F-Fanions**, a new family of optimizers parameterized by $\alpha \in [0,1]$ and k, with the update rule:

$$\Delta\mathbf{X}^t = \eta\alpha\mathbf{U}_k\mathbf{V}_k^\top + \eta(1-\alpha)\frac{\mathbf{G}^t}{\|\mathbf{G}^t\|_{\text{F}}}.$$

This framework recovers existing algorithms as special cases:

- **Normalized SGD:** $\alpha = 0$.
- **Dion** (no feedback): $\alpha = 1$, $k < \min\{m,n\}$.
- **Muon:** $\alpha = 1$, $k = \min\{m,n\}$.

We also identify and analyze **F-Neon**, a computationally efficient member of this family where $k = 1$.

Efficiently Computing the Update

Computing the F-Fanion update is efficient at the extremes of the rank parameter k. We propose using the GPU-friendly **Thick-Restarted Lanczos (TRLan)** method in two distinct ways:

- **For small k (e.g., F-Neon):** TRLan is used directly to find the few leading singular vectors.
- **For large k (e.g., Muon):** TRLan is used to compute trailing singular values, they are substracted from a full-rank term, obtained through Newton-Shulz iterations.

The table below demonstrates the speed of the direct approach for finding k leading singular vectors, from a 5000×5000 matrix.

Method	k=1	k=10	k=100
TRLan (our choice)	0.18s	0.47s	1.96s
Randomized SVD (RSVD)	1.15s	19.4s	170.0s
Power Iterations	7.70s	—	—

TRLan's dramatic speed advantage in this direct computation makes low-rank F-Fanions, like F-Neon, highly efficient.

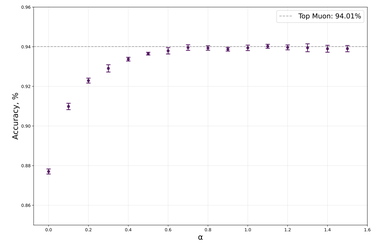
Modded NanoGPT Speedrun

We benchmarked our method against Muon on the `modded_nanogpt_2024` speedrun, a time-constrained training challenge. Below are the results after 1750 steps,

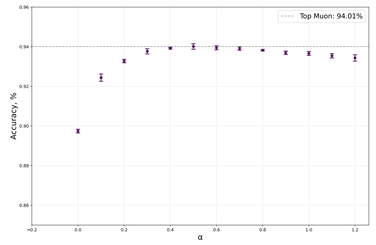
Method	LR	Momentum	Final Loss
Muon (Baseline)	0.05	0.95	3.279 (Pass)
F-Muon ($\alpha = 0.5$)	0.07	0.95	3.281 (Near Miss)

CIFAR-10 Airbench Experiments

We evaluate F-Muon against Muon by training a Convolutional Neural Network (CNN) on the CIFAR-10 Airbench. All results are averaged over multiple runs after 8 epochs.



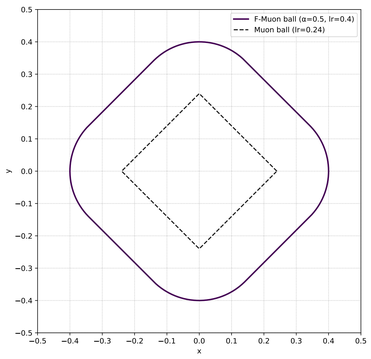
With parameters from tuned version of Muon



With parameters tuned for F-Muon, $\alpha = 0.5$

We can see that Properly tuned F-Muon can perform on par with Muon, achieving the same 94% accuracy.

Another Interesting observation here is that tuned F-Muon has much larger trust region then Muon, and thus makes larger step sizes (figure to the right).



Conclusion & Outlook

- We introduced the **F-Fanion framework**, unifying LMO-based optimizers like Muon and Dion via mixed norms.
- The choice of norm is a flexible design parameter, not a fixed rule. Our results show alternatives to the spectral norm can be equally effective.
- This work opens the core question of how to theoretically guide norm selection. Future work could also explore adapting the norm (via α and k) dynamically during training.

Key References

[1] Ahn, K. & Xu, B. *Dion*. arXiv:2504.05295, 2025.

[2] Bernstein, J. *Deriving muon*. 2025. URL: jeremybernste.in/writing/deriving-muon

[3] Bernstein, J. & Newhouse, L. *Old optimizer, new norm*. arXiv:2409.20325, 2024.

[4] Riabinin, A. et al. *Gluon: Making muon & scion great again!* arXiv:2505.13416, 2025.

Alexey Kravatskiy[1] kravtskii.aiu@phystech.edu

Ivan Kozyrev[1] kozyrev.in@phystech.edu

Nikolai Kozlov[1] kozlov.na@phystech.edu

Alexander Vinogradov[1] vinogradov.am@phystech.edu

Daniil Merkulov[1,2,3,4] daniil.merkulov@phystech.edu

Ivan Oseledets[5,2] i.oseledets@skoltech.ru

[1]Moscow Institute of Physics and Technology
[2]Skolkovo Institute of Science and Technology
[3]HSE [4]AI4Science [5]AIRI