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

Alexey Kravatskiy
Moscow Institute of Physics and Technology (MIPT)
kravtskii.aiu@phystech.edu

Ivan Kozyrev
Moscow Institute of Physics and Technology (MIPT)
kozyrev.in@phystech.edu

Nikolai Kozlov
Moscow Institute of Physics and Technology (MIPT)
kozlov.na@phystech.edu

Alexander Vinogradov
Moscow Institute of Physics and Technology (MIPT)
vinogradov.am@phystech.edu

Daniil Merkulov
Moscow Institute of Physics and Technology (MIPT), Skoltech, HSE, AI4Science
daniil.merkulov@phystech.edu

Ivan Oseledets
AIRI, Skoltech
i.oseledets@skoltech.ru

Abstract

In this article, we explore the use of various matrix norms for optimizing functions of weight matrices, a crucial problem in training large language models. Moving beyond the spectral norm that underlies the Muon update, we leverage the duals to the Ky Fan k-norms to introduce a family of Muon-like algorithms we name *Fanions*, which happen to be similar to Dion. Subsequently, we construct a second family of *F-Fanions*, which are based on the duals of convex combinations of Ky Fan k-norms and the Frobenius norm. One prominent member of this family is *F-Muon*. We complement our theoretical analysis with an extensive empirical study of the algorithms across a wide range of tasks and settings, from which it follows that F-Muon is on par with Muon, which questions the exclusivity of a spectral norm.

1 Introduction

Minimizing loss functions in unprecedentedly high-dimensional spaces has recently become an integral and crucial part in training large language models. Hence, new scalable, time- and memory-efficient algorithms have been demanded. Besides well-known Adam (Kingma & Ba, 2014) and AdamW (Loshchilov & Hutter, 2017), recently proposed Muon (Jordan et al., 2024b) has shown promising results on training very large models (Liu et al., 2025). Its key difference from Adam and AdamW is that it has been constructed specifically for optimizing functions of weight matrices, which are common in deep learning.

That is what can be said from a practical point of view. From a theoretical perspective, a key innovation of Muon was its principled derivation of the update rule, which emerges as the solution to an optimization problem constrained by the RMS-to-RMS norm (scaled version of spectral norm) (Bernstein, 2025)

Motivated by the success of Muon, many generalizations and variations of it were proposed. Among the notable ones are Scion (Pethick et al., 2025b), Dion (Ahn & Xu, 2025) and Gluon (Riabinin et al., 2025). Those researches target Muon's efficiency and establish convergence bounds. One central question, however, remains unanswered:

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

In this article, we tackle this question by first showing the connection between matrix norms and corresponding existing algorithms and discuss the theoretic bounds derived for those algorithms. We then leverage the family of norms dual to Ky Fan k-norms to derive a new class of algorithms with low-rank updates, which we call Fanions. Subsequently, we create a second, hybrid family named F-Fanions by constructing convex combinations of the Ky Fan norms with the Frobenius norm and taking dual of that composite norm. Working within the linear minimization oracle (LMO) framework we derive the explicit update formulas for both algorithm families. As it was done with Muon, we stipulate our algorithms to be fast to approximate, for which we utilize the Lanczos algorithm as described in section 3.

We then compare the performance of the algorithm families across various benchmarks (section 4):

- Synthetic least squares experiment,
- CIFAR-10 airbench,
- Pre-training NanoGPT on FineWeb dataset.

Our experiments reveal important insights about the role of matrix norms in optimization. On a synthetic least squares problem, we observe a striking discrepancy: while some algorithms converge slowly in terms of loss, they may converge quickly in terms of gradient norm, and vice versa. This suggests that existing theoretical guarantees may not fully explain practical algorithm performance.

Most notably, our experiments on real-world tasks demonstrate that the choice of underlying matrix norm is remarkably flexible. On CIFAR-10 airbench, properly-tuned F-Muon achieves $94.01 \pm 0.14\%$ accuracy, essentially matching Muon's $94.00 \pm 0.13\%$ performance. On NanoGPT pre-training, F-Muon achieves 3.281 cross-entropy loss, only marginally worse than Muon's 3.279. These results show that Muon-like algorithms can maintain competitive performance even when the underlying norm constraint is significantly modified, answering affirmatively the central question posed above. This flexibility suggests potential for developing easier-to-compute variants of successful algorithms like Muon.

2 Update Step as Constrained Optimization

2.1 Linear Minimization Oracle Framework

Training a neural network is essentially an optimization of a function of several weight matrices. Let us start by considering the problem of minimizing a differentiable function of a *single* matrix:

$$F(\cdot)\colon \mathbb{R}^{m\times n} \to \mathbb{R}\,, \qquad F(\boldsymbol{X}) \to \min_{\boldsymbol{X}\in\mathbb{R}^{m\times n}} \tag{1}$$

We equip the matrix space $\mathbb{R}^{m\times n}$ with a standard dot product $\langle\cdot,\cdot\rangle \to \mathbb{R}$ and a norm $\|\cdot\|\colon \mathbb{R}^{m\times n} \to \mathbb{R}_+$, which does not have to coincide with the Frobenius norm $\|\cdot\|_{\mathrm{F}}$. The dual norm $\|\cdot\|^\dagger\colon \mathbb{R}^{m\times n} \to \mathbb{R}_+$ associated with $\|\cdot\|$ is defined as

$$\|\boldsymbol{G}\|^\dagger = \sup_{\boldsymbol{D}\in\mathbb{R}^{m\times n}:\|\boldsymbol{D}\|\leq 1} \langle \boldsymbol{G}, \boldsymbol{D}\rangle \,. \tag{2}$$

Such problems can be solved with an iterative algorithm based on the Linear Minimization Oracle (LMO):

$$\mathrm{LMO}(\boldsymbol{G}) \in \operatorname*{arg\,min}_{\boldsymbol{D}\in\mathcal{S}} \langle \boldsymbol{G}, \boldsymbol{D}\rangle \,, \tag{3}$$

where $\boldsymbol{G}$ is typically a gradient (possibly with momentum) of F and $\mathcal{S} \subset \mathbb{R}^{m\times n}$ is some set. The simple update formula then can look like

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k + \gamma_k \mathrm{LMO}\left(\nabla F\left(\boldsymbol{X}^k\right)\right) \,. \tag{4}$$

In a typical neural network, the objective function F depends on a set of weight matrices $\{\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots\}$. The optimization framework we have described is applied in a layer-wise fashion. At each iteration k, a stochastic gradient $g(\boldsymbol{X}^k, \xi^k)$ is computed using a mini-batch of data with the ξ^k noise via backpropagation. This yields a separate gradient component, $\boldsymbol{G}_i^k$, for each matrix $\boldsymbol{X}_i$. The LMO-based update rule is then applied to each matrix $\boldsymbol{X}_i$ using its corresponding gradient component $\boldsymbol{G}_i^k$.

The update rule used in Muon optimizer is uSCG (Pethick et al., 2025b). In the most general case, which involves momentum, it can be written as

$$\begin{aligned} \boldsymbol{M}^k &= \alpha_k g(\boldsymbol{X}^k, \xi^k) + (1-\alpha_k)\boldsymbol{M}^{k-1} \,, \\ \boldsymbol{X}^{k+1} &= \boldsymbol{X}^k + \gamma_k \mathrm{LMO}(\boldsymbol{M}^k) \,. \end{aligned} \tag{5}$$

Here we are particularly interested in the case when $\mathcal{S}$ is a ball in the $\|\cdot\|$ norm:

$$\mathcal{S} = \mathcal{B}_\eta = \{\boldsymbol{D} \in \mathbb{R}^{m\times n} \mid \|\boldsymbol{D}\| \leq \eta\} \,. \tag{6}$$

In this case it can be easily shown that

$$\operatorname*{arg\,min}_{\boldsymbol{D}\in\mathcal{S}} \langle \boldsymbol{G}, \boldsymbol{D}\rangle = -\eta\{\boldsymbol{D} \in \mathcal{B}_1 \mid \langle \boldsymbol{G}, \boldsymbol{D}\rangle = \|\boldsymbol{G}\|^\dagger\} \,. \tag{7}$$

And update for $\boldsymbol{X}$ in eq. (5) becomes

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \gamma_k\eta\{\boldsymbol{D} \in \mathcal{B}_1 \mid \langle \boldsymbol{M}^k, \boldsymbol{D}\rangle = \|\boldsymbol{M}^k\|^\dagger\} \,. \tag{8}$$

This formula will later allow us to easily compute updates for various norms.

2.2 Different Norms $\|\cdot\|$ imply different updates

In this subsection we will derive and discuss updates induced by the choice of the norm in eq. (8). To describe those updates, the singular value decomposition (SVD) of $\boldsymbol{M}^k$ is required. We denote SVD of $\boldsymbol{M}^k$ as $\boldsymbol{M}^k = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} = [u_1, u_2, \ldots, u_r]$, $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$, and $\boldsymbol{V} = [v_1, v_2, \ldots, v_r]$.

2.3 $\|\boldsymbol{M}^k\|_{\mathrm{F}}$ AND NORMALIZED SGD

Lemma 1. *When* $\|\cdot\| = \|\cdot\|_{\mathrm{F}}$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \eta \frac{\boldsymbol{M}^k}{\|\boldsymbol{M}^k\|_{\mathrm{F}}} \tag{9}$$

The result is the same as in (Table 1, Pethick et al. (2025b)), but here we use the Euclidean norm as $\|\boldsymbol{M}^k\|_{\mathrm{F}} = \sqrt{\sum_{i=1}^{\min\{m,n\}} \sigma_i^2}$, which clearly is a matrix norm.

2.4 $\|\boldsymbol{M}^k\|_{\mathrm{op}}$ AND MUON

Lemma 2. *When* $\|\cdot\| = \|\cdot\|_{\mathrm{op}}$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \eta U V^\top \tag{10}$$

Though the update is well-known, we provide a much simpler proof in the appendix, when compared to Bernstein & Newhouse (2024).

2.5 $\|\boldsymbol{M}^k\|_{\mathrm{nuc}}$ AND NEON

Lemma 3. *When* $\|\cdot\| = \|\cdot\|_{\mathrm{nuc}}$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \eta u_1 v_1^\top \tag{11}$$

We name the derived algorithm *Neon*. In the section *The Matrix side of updates*, we will discuss how to compute the update efficiently.

2.6 $\|\boldsymbol{M}^k\|_{\mathrm{F}*}^\dagger$ AND F-MUON

We define $\|\cdot\|_{\mathrm{F}*}$ as a convex combination of $\|\cdot\|_{\mathrm{nuc}}$ and $\|\cdot\|_{\mathrm{F}}$:

$$\|\boldsymbol{X}\|_{\mathrm{F}*} = \alpha \|\boldsymbol{X}\|_{\mathrm{nuc}} + (1-\alpha)\|\boldsymbol{X}\|_{\mathrm{F}}, \tag{12}$$

where $\alpha \in [0, 1]$ defines a specific norm from the $F*$-family. $\|\cdot\|_{\mathrm{F}*}^\dagger$ can be expressed as in eq. (19). However, its norm ball is a simple Minkowski sum of $\alpha\|\cdot\|_{\mathrm{nuc}}$ and $(1-\alpha)\|\cdot\|_{\mathrm{F}}$ balls fig. 5a.

Lemma 4. *When* $\|\cdot\| = \|\cdot\|_{\mathrm{F}*}^\dagger$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \eta(\alpha U V^\top + (1-\alpha)\frac{\boldsymbol{M}^k}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}) \tag{13}$$

We name the derived algorithm *F-Muon*. It turns out that F-Muon is a convex combination of Normalized SGD and Muon. The implications are significant and discussed in the following sections.

2.7 $\|\boldsymbol{M}^k\|^\dagger_{\mathrm{F2}}$ AND F-NEON

We define $\|\cdot\|_{\mathrm{F2}}$ as a convex combination of $\|\cdot\|_{\mathrm{op}}$ and $\|\cdot\|_{\mathrm{F}}$:

$$\|\boldsymbol{X}\|_{\mathrm{F2}} = \alpha\|\boldsymbol{X}\|_{\mathrm{op}} + (1-\alpha)\|\boldsymbol{X}\|_{\mathrm{F}}, \tag{14}$$

where $\alpha \in [0,1]$ defines a specific norm from the F2-family.$\|\cdot\|^\dagger_{\mathrm{F2}}$ can be expressed as in eq. (20). However, its norm ball is a simple Minkowski sum of $\alpha\|\cdot\|_{\mathrm{op}}$ and $(1-\alpha)\|\cdot\|_{\mathrm{F}}$ balls fig. 5b.

Lemma 5. *When* $\|\cdot\| = \|\cdot\|^\dagger_{\mathrm{F2}}$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - \eta(\alpha u_1 v_1^\top + (1-\alpha)\frac{\boldsymbol{M}^k}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}) \tag{15}$$

We name the derived algorithm *F-Neon*. It turns out that F-Neon is a convex combination of Normalized SGD and Neon. The implications are significant and discussed in the following sections.

2.8 $\|\boldsymbol{M}^k\|^\dagger_{\mathrm{KF-k}}$ AND MUON, NEON, AND CENTRALIZED DION WITHOUT ERROR FEEDBACK

We remind the reader that the Ky Fan k-norm (Bhatia (2013), p. 92), which we denote as $\|\cdot\|_{\mathrm{KF-k}}$, is the sum of the k largest singular values of the matrix. It can be proved that $\|\cdot\|^\dagger_{\mathrm{KF-k}} = \max\{\frac{1}{k}\|\cdot\|_{\mathrm{nuc}}, \|\cdot\|_{\mathrm{op}}\}$ (see Bhatia (2013), p. 96). Special cases of the Ky Fan k-norm are the Ky Fan 1-norm, which is the sprectral norm, and the Ky Fan $\min\{m,n\}$-norm, which is the nuclear norm.

Lemma 6. *When* $\|\cdot\| = \|\boldsymbol{M}^t\|^\dagger_{\mathrm{KF-k}}$*, eq.* (8) *turns into:*

$$\boldsymbol{X}^{t+1} = \boldsymbol{X}^t - \eta\sum_{i=1}^{k} u_i v_i^\top \tag{16}$$

In the section *The Matrix side of updates*, we will discuss how to compute the updates efficiently.

We introduce the family of *Fanions*, which consists of *Fanion-k*, lmo-based algorithms under the $\|\boldsymbol{M}^t\|^\dagger_{\mathrm{KF-k}}$ norms. By this terminology and the lemma, Muon is a Fanion-$\min\{n,m\}$, while Neon is an Fanion-1. Moreover, the centralized version of rank-r Dion (Ahn & Xu, 2025) without the error feedback and without scaling of the update, from the perspective of lmo, is actually a Fanion-r.

In addition, one can consider F-KF-k-norm: $\|\cdot\|_{\mathrm{F-KF-k}} = \alpha\|\cdot\|_{\mathrm{KF-k}} + (1-\alpha)\|\cdot\|_{\mathrm{F}}$, the dual to which will produce algorithms like F-Dion without the error feedback. The resulting family can be named *F-Fanions*.

2.9 ALGORITHMS FOR MATRICES ↔ ALGORITHMS FOR VECTORS

lmo optimizers in Schatten S_p norms and in l_p norms with common p may be analogous to each other, as is illustrated by table 1. The analogies go beyond similarities in the updates. SignSGD is very close to Adam, as noted in Bernstein & Newhouse (2024), and both Adam and Muon

Table 1: lmo optimizers in Schatten S_p norms and in l_p norms. g is the gradient. When it is a matrix, $g = \boldsymbol{U\Sigma V}^\top$

Method	lmo constraint set $\mathcal{D}$	lmo	Reference
Normalized SGD	l_2-ball, S_2-ball	$-\eta\frac{g}{\|g\|_2} = -\eta\frac{g}{\|g\|_F}$	(Hazan et al., 2015)
Momentum Normalized SGD	Ball in l_2, or Ball in S_2	$-\eta\frac{g}{\|g\|_2} = -\eta\frac{g}{\|g\|_F}$	(Cutkosky et al., 2020)
SignSGD	Ball in Max-norm l_∞	$-\eta\,\mathrm{sign}(g)$	(Bernstein et al., 2018, Thm. 1)
Signum	Ball in Max-norm l_∞	$-\eta\,\mathrm{sign}(g)$	(Bernstein et al., 2018, Thm. 3)
Muon	Ball in Spectral S_∞	$-\eta UV^\top$	(Jordan et al., 2024b)
Gauss-Southwell Coordinate Descent	Ball in l_1	$-\eta\sum_{i\in\arg\max\|g_i^t\|}\mathrm{sign}(g_i^t)e_i$	(Shi et al., 2016, p. 19)
Neon	Ball in Nuclear S_1	$-\eta u_1 v_1^\top$	This work

perform well in training large models. NSGD is the same for both matrix and vector cases. Greedy Coordinate Descent methods are not applied to high-dimensional problems, from this perspective, it is not surprising that Neon underperforms on such problems.

3 Matrix side of updates

To compute the algorithms' updates, we use thick-restarted Lanczos method for singular value problem (TRLan) on $\boldsymbol{M}^{k\top}\boldsymbol{M}^k$ or $\boldsymbol{M}^k\boldsymbol{M}^{k\top}$ matrices (the one with less size is picked), implemented in CuPy lSibrary (Preferred Infrastructure & Developers, 2025) and described in Simonz (1998).

This method is designed for efficiently approximating the largest singular values and vectors of large matrices. Its thick-restart strategy retains the most informative Ritz vectors at each cycle, which accelerates convergence while avoiding excessive memory growth. We are specifically interested in this algorithm because it allows us to extract several largest singular values and related singular vectors of the matrix to make a Neon step. Moreover, TRLan is stable GPU-friendly algorithm because it mainly operates with matrix-vector multiplications (MVs), which are higly-parallel, and does not require full reorthogonalization against the whole Krylov basis by managing short recurrent formulas and incorparating thick restarting.

Per-cycle complexity is $O(mn^2 + n^2k + nk^2)$, where m and n are dimensions of the target matrix and n is the smaller one, k is retained subspace's size.

In table 2, we compare performance of TRLan, RSVD, and power iterations on calculating k-rank update, which is used in k-Fanion. The results highlight that TrLan is much faster that its competitors.

During the research it was noted that RSVD can give good and fast approximation for singular values, but the matrix of approximation is far from the one given by truncated SVD, while TR-Lan gives good and fast approximation of a matrix, but not so good approximation for singular values. That means that TRLan may be not a perfect choice for algorithms like Dion, where σ_i are required for error feedback.

The practical drawback of the TRLAn is the absence of its implementation in PyTorch.

Method	rtol	k	time,s
Power Iterations	0.01	1	7.7
SVDS (thick-restart Lanczos method)	0.01	1	0.18
PCA Low Rank (RSVD)	0.01	1	1.15
SVDS (thick-restart Lanczos method)	0.01	10	0.47
PCA Low Rank (RSVD)	0.01	10	19.4
SVDS (thick-restart Lanczos method)	0.01	100	1.96
PCA Low Rank (RSVD)	0.01	100	170

Table 2: k-rank updated comparison
Comparison of different numerical methods to calculate k-rank update on 5000×5000 matrix of real numbers, rtol is an error in Frobenius norm relative to the k-rank approximation.

4 Experiments

4.1 Randomized Linear Least Squares

We first compare F-Fanions on the following L-smooth problem:

$$F(\boldsymbol{X}) = \frac{1}{2}\langle(\boldsymbol{X} - \boldsymbol{S}), \boldsymbol{M}(\boldsymbol{X} - \boldsymbol{S})\boldsymbol{N}\rangle \to \min_{\boldsymbol{X} \in \mathbb{R}^{m \times n}} \tag{17}$$

where $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, $m = 500$ and $n = 500$ are typical dimensions of a neural network weight matrix, $S \in \mathbb{R}^{m \times n}$, $\boldsymbol{M} \in \mathbb{S}_+^m$ and $\boldsymbol{N} \in \mathbb{S}_+^n$ are positive-semidefinite matrices. The spectra of $\boldsymbol{M}$ and $\boldsymbol{N}$ are uniformly distributed in the (0, 1) interval. We set $\boldsymbol{S} = 0$, and $\boldsymbol{X}^0 \sim 0.1\mathcal{N}(0, \boldsymbol{I})$.

We run different Fanions and their respective F-Fanions with $\alpha = 1/2$: Neon (Fanion-1), Fanion-2, Fanion-5, Fanion-100, and Muon (Fanion-500). We test them against Normalized SGD, which is also a F-Fanion with $\alpha = 0$ and an arbitrary k.

Since theoretical bounds (Kovalev, 2025; Riabinin et al., 2025) rely on a very loose norm bound $\|\cdot\| \leq \rho\|\cdot\|_{\mathrm{F}}$, we do not derive learning rate from smoothness. Rather, we set the learning rate of $1/\sqrt{T}$ for each optimizer and the momentum of 0.6. 12 000 iterations for each algorithm ensure that the rate at some point gets low enough for each algorithm to converge. The results are presented in fig. 1 and section D.

In terms of the loss minimization, F-Muon had the fastest convergence. In terms of the Frobenius norm of the gradient, the lowest was observed for NSGD. Then go F-Neon (F-Fanion-1), F-Fanion-2, F-Fanion-5, F-Fanion-100, and F-Muon (Fanion-500). It is noticeable that all F-Fanions have lower Frobenius norms of the gradients than their respective Fanioins. The same behavior will be recorded in the next section.

4.2 CIFAR-10 airbench

We adapt Keller Jordan's code to test F-Muon, Neon, and F-Neon on the CIFAR-10 airbench (Keller, 2023). First, we run F-Muon for different α with the same `lr=0.24(1 - step/total_steps), momentum=0.6, nesterov=True`, as have been finetuned by Jordan, 10 repetitions for each α. We record the accuracy after 8 epochs of training. The results are in fig. 2a.

Then we tune F-Muon with $\alpha = 0.5$. Tuned parameters are `lr=0.4(1 - step/total_steps), momentum=0.65, nesterov = True`. While Muon reaches $94.00 \pm 0.13\%$ accuracy after 8 epochs, tuned F-Muon reaches $94.01 \pm 0.14\%$ after 8 epochs (average was done by 200 iterations).

Finally, we take this set of tuned parameters and test on different α, 5 times for each α. The results are in fig. 2b. We notice that even when $\alpha = 0.1$, the accuracy is much higher than in case of the pure NSGD.

The results are curious and could be represented by fig. 3: lmo ball, which we plotted in a 2D space of singular values, has drastically changed, but the observed convergence after tuning has not degraded. These observations raise the question of how much lmo-based algorithms are sensitive to the constraint area, i.e. what will happen if the ball is corrupted. In this particular example, we have had a blurred ball, which proved as robust as the original ball.

The most pathological case is $\alpha > 1$, which corresponds to the lmo with an area that is not a norm ball! Despite this violation, the mixture of algorithms reaches almost the same accuracy as vanilla Muon.

4.3 NanoGPT speedrun

We test F-Muon on NanoGPT speedrun (Jordan et al., 2024a). For $\alpha = 0.5$, the optimal parameters are `lr=0.07, momentum=0.95`, while for Muon they were `lr=0.05, momentum=0.95`. After testing for 1750 steps, as it has been done on the speedruns, we get 3.281 cross-entropy loss, while on Muon, it falls below the target threshold 3.28 reaching 3.279. However, this difference is negligible, if one looks at fig. 4. It is even more striking, considering the fact that F-Muon is an average of Muon and NSGD, and the later performed quite poorly.

Again, as on the CIFAR airbench, if we set $\alpha = -0.1$, F* to which is not a norm, we get a 3.2818 loss, which is not higher than for $\alpha = 0.3$, for example.

5 Related Work

As Muon (Jordan et al., 2024b) is a very successful and popular optimizer for functions of weight matrices, a lot of reseach has been put into, first, further improving its performance, and, second, in explaining its success.

Improvements of Muon. Regarding the first point, in less than a year, a large number of applications and improvements of Muon has been proposed. Liu et al. (2025) adapted the algorthm for training language models larger than NanoGPT. Shah et al. (2025) organized efficient hyperparameter transfer by combining Muon with maximal update parametrization. To construct their COSMOS optimizer, Chen et al. (2025) applied computationally intensive updates of SOAP optimizer to a low-dimensional ”leading eigensubspace” while using memory-efficient methods like Muon for the remaining parameters. Amsel et al. (2025) proposed a more efficient alternative to Newton-Schulz operations. Si et al. (2025) introduced AdaMuon which combines element-wise adaptivity with orthogonal updates. We suppose that the described above or similar techniques can be applied to our optimizers as well. For example, F-Muon also benefits from faster alternatives to Newton-Schulz iterations, and Neon may be a great substitute to Muon in COSMOS, because, as we have shown in *the Matrix side of the updates*, Lanczos algorithms is much faster than Newton-Schulz iterations on large matrices.

Theory behind Muon. Regarding the second point, there has been a prolonged gap in theory behind Muon, simplistic derivation of Bernstein (2025) based on Bernstein & Newhouse (2024) excluded. This gap, as it seems to us, is not even now completely closed. For example, Kovalev (2025) has provided convergence guarantees of Muon in various settings, from which, however, Muon's supremacy cannot be recovered. Indeed, although the obtained bounds depend on the norm choice, the asymptotics of the convergence remain the same as for NSGD and other optimizers, $K = \mathcal{O}(\varepsilon^{-4})$ in a L-smooth stochastic case.

Similar drawback has a recent article Riabinin et al. (2025), where L-smoothness assumption is replaced with a more practical (L_0, L_1)-smoothness. The authors derived from their theorems optimal stepsizes for Muon and Scion that match fine-tuned stepsizes reported by Pethick et al. (2025b). But still, they did not explain why, for instance, NSGD is inferior to Muon in training large-language models.

We suppose that the reason for the recorded by us discrepancy between Neon and Muon performance, both of which are described by Scion or Gluon frameworks, lies in the structure of the norm ball, which must be an object of further research.

The nuclear norm in lmo. As we found out only when writing this article, the nuclear norm has been already explored in the context of the linear minimization oracle. Pethick et al. (2025a) applied it to create νSAM, a new sharpness-aware minimization technique. The update from their Lemma 3.1 coincides with the update of Neon, but is used for completely different purposes. In addition, Pethick et al. (2025a) use power iterations to find u_1 and $v_1^\top$, while we suggest utilizing much more efficient and precise Lanczos algorithm.

The Ky Fan Norm and Dion. Rank-k Centralized Dion, Algorithm 1 from Ahn & Xu (2025), without an error feedback and scaling of the update, turns out to be an lmo-based algorithm under the $\|\cdot\|_{\mathrm{KF-k}}^{\dagger}$ norm, which we described in *Different norms imply different updates*. Reported by the authors necessity of using error feedback to obtain satisfactory accuracy may take place in the cases of our algorithms as well, for example, F-Neon. This we leave to future research.

6 Conclusion

In this article, we have generalized several successful algorithms, like Muon and Dion, to the lmo-based algorithms in the $\|\cdot\|_{\mathrm{KF-k}}^{\dagger}$ norm. Also we have proposed the technique of "regularizing" the updates with NSGD, a trick to increase the robustness of the algorithms, which is motivated by the consideration of the $\|\cdot\|_{\mathrm{F*}}$, $\|\cdot\|_{\mathrm{F2}}$, and general $\|\cdot\|_{\mathrm{F-KF-k}}$ norms. Generalizations of well-known norms and subsequent combinations of them may further improve performance of lmo-based algorithms. If a theory is developed that explains the discrepancy between performance of different algorithms based on the matrix norms, one will probably be able to intentionally construct norms optimal to given architectures and probably even their parameters.

7 Author Contributions

IO suggested using the nuclear norm in the Bernstein & Newhouse (2024) framework. DM supervised the project and helped with editing the article. IK, NK, and AV participated mainly on the first stage of research, when it has been a project in the optimization course at MIPT. IK suggested using composite norms (though not F2 and F*) and KKT conditions to find the resulting updates. NK suggested Lanczos algorithm as the fastest tool to compute Neon's updates and conducted experiments to prove it. AV tested Neon on the finetuning of NanoGPT. All other work was done by AK.

REFERENCES

Kwangjun Ahn and Byron Xu. Dion: A communication-efficient optimizer for large models, 2025. URL `https://arxiv.org/abs/2504.05295`.

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

Jeremy Bernstein. Deriving muon, 2025. URL `https://jeremybernste.in/writing/deriving-muon`.

Jeremy Bernstein and Laker Newhouse. Old optimizer, new norm: An anthology. *arXiv preprint arXiv:2409.20325*, 2024.

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International conference on machine learning*, pp. 560–569. PMLR, 2018.

Rajendra Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

Weizhu Chen, Chen Liang, Tuo Zhao, Zixuan Zhang, Hao Kang, Liming Liu, Zichong Li, and Zhenghao Xu. Cosmos: A hybrid adaptive optimizer for memory-efficient training of llms, 2025. URL `https://arxiv.org/abs/2502.17410`.

Ashok Cutkosky, Harsh Mehta, and Francesco Orabona. Momentum-based variance reduction in nonconvex sgd. *Advances in Neural Information Processing Systems*, 2020.

Elad Hazan, Kfir Levy, and Shai Shalev-Shwartz. Beyond convexity: Stochastic quasi-convex optimization. *Advances in neural information processing systems*, 28, 2015.

Keller Jordan, Jeremy Bernstein, Brendan Rappazzo, @fernbear.bsky.social, Boza Vlado, You Jiacheng, Franz Cesista, Braden Koszarsky, and @Grad62304977. modded-nanogpt: Speedrunning the nanogpt baseline, 2024a. URL `https://github.com/KellerJordan/modded-nanogpt`.

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

Jordan Keller. cifar10-airbench, 2023. URL `https://github.com/KellerJordan/cifar10-airbench`.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dmitry Kovalev. Understanding gradient orthogonalization for deep learning via non-euclidean trust-region optimization. *arXiv preprint arXiv:2503.12645*, 2025.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Thomas Pethick, Parameswaran Raman, Lenon Minorics, Mingyi Hong, Shoham Sabach, and Volkan Cevher. νSAM: Memory-efficient sharpness-aware minimization via nuclear norm constraints. *Transactions on Machine Learning Research*, 2025a. ISSN 2835-8856. URL `https://openreview.net/forum?id=V6ia5hWIMD`.

Thomas Pethick, Wanyun Xie, Kimon Antonakopoulos, Zhenyu Zhu, Antonio Silveti-Falls, and Volkan Cevher. Training deep learning models with norm-constrained lmos. *arXiv preprint arXiv:2502.07529*, 2025b.

Inc. Preferred Infrastructure and CuPy Developers. CuPy: `cupyx.scipy.sparse.linalg.svds` — api reference. `https://docs.cupy.dev/en/stable/reference/generated/cupyx.scipy.sparse.linalg.svds.html`, 2025. Accessed: 2025-08-24.

Artem Riabinin, Egor Shulgin, Kaja Gruntkowska, and Peter Richtárik. Gluon: Making muon & scion great again!(bridging theory and practice of lmo-based optimizers for llms). *arXiv preprint arXiv:2505.13416*, 2025.

Ishaan Shah, Anthony M Polloreno, Karl Stratos, Philip Monk, Adarsh Chaluvaraju, Andrew Hojel, Andrew Ma, Anil Thomas, Ashish Tanwer, Darsh J Shah, et al. Practical efficiency of muon for pretraining. *arXiv preprint arXiv:2505.02222*, 2025.

Hao-Jun Michael Shi, Shenyinying Tu, Yangyang Xu, and Wotao Yin. A primer on coordinate descent algorithms. *arXiv preprint arXiv:1610.00040*, 2016.

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

Kesheng Wuzand Horst Simonz. Thick-restart lanczos method for symmetric eigenvalue problemsy. 1998.

Yao-Liang Yu. Arithmetic duality for norms, 2012. URL `https://cs.uwaterloo.ca/~y328yu/notes/normduality.pdf`.

A NORMS $\|\cdot\|_{\mathrm{F}*}^{\dagger}$ AND $\|\cdot\|_{\mathrm{F}2}^{\dagger}$

First, we need a well-known fact mentioned, for example, in Yu (2012, Table 1). For the sake of completeness, we provide a proof of the fact.

Lemma 7. *Let $\|\cdot\|_{(1)}$ and $\|\cdot\|_{(2)}$ be norms on a finite-dimensional Euclidean space, and let $\alpha, \beta \geq 0$. Define*

$$\|x\| := \alpha \|x\|_{(1)} + \beta \|x\|_{(2)}.$$

Then the dual unit ball of $\|\cdot\|$ satisfies

$$B_{\|\cdot\|^{\dagger}} = \alpha B_{\|\cdot\|_{(1)}^{\dagger}} + \beta B_{\|\cdot\|_{(2)}^{\dagger}},$$

where $+$ denotes the Minkowski sum and $B_{\|\cdot\|_{(i)}^{\dagger}}$ is the unit ball of the dual norm $\|\cdot\|_{(i)}^{\dagger}$.

Proof. Write $f(x) = \alpha\|x\|_{(1)}$ and $g(x) = \beta\|x\|_{(2)}$, so

$$\|x\| = f(x) + g(x).$$

Recall two standard facts:

1. For any norm $\|\cdot\|$ and $\lambda > 0$,

$$(\lambda\|\cdot\|)^*(y) = \sup_x \bigl(\langle y, x\rangle - \lambda\|x\|\bigr) = \delta_{\lambda B_{\|\cdot\|^\dagger}}(y),$$

   i.e., the indicator function of the scaled dual ball.

2. The Fenchel conjugate of a sum satisfies

$$(f+g)^*(y) = \inf_{u+v=y} \bigl(f^*(u) + g^*(v)\bigr).$$

Applying these to f and g, we have

$$f^*(u) = \delta_{\alpha B_{\|\cdot\|^\dagger_{(1)}}}(u), \quad g^*(v) = \delta_{\beta B_{\|\cdot\|^\dagger_{(2)}}}(v).$$

Thus

$$\|\cdot\|^*(y) = (f+g)^*(y) = \inf_{u+v=y} \bigl(\delta_{\alpha B_{\|\cdot\|^\dagger_{(1)}}}(u) + \delta_{\beta B_{\|\cdot\|^\dagger_{(2)}}}(v)\bigr) = \delta_{\alpha B_{\|\cdot\|^\dagger_{(1)}} + \beta B_{\|\cdot\|^\dagger_{(2)}}}(y).$$

But by definition, the conjugate of a norm is exactly the indicator of its dual unit ball:

$$\|\cdot\|^*(y) = \delta_{B_{\|\cdot\|^\dagger}}(y).$$

Therefore,

$$B_{\|\cdot\|^\dagger} = \alpha B_{\|\cdot\|^\dagger_{(1)}} + \beta B_{\|\cdot\|^\dagger_{(2)}}.$$

□

Consequently,

$$\|y\|^\dagger = \inf\bigl\{t \geq 0 : y \in t\bigl(\alpha B_{\|\cdot\|^\dagger_{(1)}} + \beta B_{\|\cdot\|^\dagger_{(2)}}\bigr)\bigr\} = \inf_{z,t}\bigl\{t \geq 0 : z \in t\alpha B_{\|\cdot\|^\dagger_{(1)}}, y - z \in t\beta B_{\|\cdot\|^\dagger_{(1)}}\bigr\} \tag{18}$$

Thus, we immediately find $\|\cdot\|^\dagger_{\mathrm{F}*}$, which is related to F-Muon update. Indeed, after setting $\beta = 1 - \alpha$ and remembering that for smooth and bounded cases we can use $\min$ instead of $\inf$, we get

$$\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}*} = \min_{\boldsymbol{Z}} \min_t \{t, s.t. \|\boldsymbol{Z}\|_{\mathrm{op}} \leq \alpha t, \|\boldsymbol{Y} - \boldsymbol{Z}\|_{\mathrm{F}} \leq (1-\alpha)t\} \tag{19}$$

If $\alpha = 1$, then $\boldsymbol{Z} = \boldsymbol{Y}$, and we get $\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}*} = \|\boldsymbol{Y}\|_{\mathrm{op}}$. If $\alpha = 0$, then $\boldsymbol{Z} = 0$, and we get $\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}*} = \|\boldsymbol{Y}\|_{\mathrm{F}}$.

Similarly, we find $\|\cdot\|^\dagger_{\mathrm{F}2}$, which is related to F-Neon update:

$$\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}2} = \min_{\boldsymbol{Z}} \min_t \{t, s.t. \|\boldsymbol{Z}\|_{\mathrm{nuc}} \leq \alpha t, \|\boldsymbol{Y} - \boldsymbol{Z}\|_{\mathrm{F}} \leq (1-\alpha)t\} \tag{20}$$

If $\alpha = 1$, then $\boldsymbol{Z} = \boldsymbol{Y}$, and we get $\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}2} = \|\boldsymbol{Y}\|_{\mathrm{nuc}}$. If $\alpha = 0$, then $\boldsymbol{Z} = 0$, and we get $\|\boldsymbol{Y}\|^\dagger_{\mathrm{F}2} = \|\boldsymbol{Y}\|_{\mathrm{F}}$.

B VISUALIZATION OF DIFFERENT MATRIX NORMS

B.1 DUALS TO F* AND F2 NORMS

It follows from lemma 7 that the norm ball in $\|\cdot\|_{\mathrm{F}*}^{\dagger}$ is the Minkowski sum of the norm ball in $\alpha\|\cdot\|_{\mathrm{nuc}}$ and $(1-\alpha)\|\cdot\|_{\mathrm{F}}$ and the norm ball in $\|\cdot\|_{\mathrm{F2}}^{\dagger}$ is the Minkowski sum of the norm ball in $\alpha\|\cdot\|_{\mathrm{op}}$ and $(1-\alpha)\|\cdot\|_{\mathrm{F}}$.

In Fig. fig. 5 we plot these norms. On x-axis and y-axis, there are singular values σ_1, σ_2 respectively of a matrix from $\mathbb{R}^{m\times n}$ with $\min\{m,n\}=2$.

B.2 THE KY FAN NORM AND ITS DUAL

1-balls in l_∞, l_1 and l_2 norms are well-known from textbooks. But what about the Ky Fan k-norm? How can it be represented?

To showcase the complex structure of the Ky Fan k-norm and its dual, we suggest the illustrations fig. 6 with the ball in the Ky Fan 2-norm in fig. 6a and its dual in fig. 6b. On x-, y-, and z-axes, there are singular values σ_1, σ_2, and σ_3 respectively of a matrix from $\mathbb{R}^{m\times n}$ with $\min\{m,n\}=3$. In this particular case, we do not sort the singular values. In the proposed representation, we actually plot balls in the Top-2 norm $\max\{|x|+|y|,|x|+|z|,|y|+|z|\}$ and its dual norm $\max\{\max(|(|x),|y|,|z|),\frac{1}{2}(|x|+|y|+|z|)\}$. The resulting balls are much more complex than balls in l_∞, l_1 and l_2 norms.

In fact, those balls can be described easier if we use the results from Yu (2012). The Ky Fan 2-norm ball is an intersection of three l_1 balls in (x,y), (x,z), and (y,z) spaces. The 1-ball in the dual Ky Fan 2-norm is an intersection of 1-ball the in l_∞ norm and $\frac{1}{2}$-ball in the l_1 norm.

C UPDATES DERIVATIONS

Proof of lemma 2 follows from eq. (13) with $\alpha=1$. Indeed, $\|\cdot\|_{\mathrm{op}}^{\dagger}=1\cdot\|\cdot\|_{\mathrm{nuc}}+0\cdot\|\cdot\|_{\mathrm{F}}$.

Proof of lemma 4: Since $\|\cdot\|^{\dagger}=\|\cdot\|_{\mathrm{F}*}^{\dagger\dagger}=\|\cdot\|_{\mathrm{F}*}$, the goal is to reach $\|\boldsymbol{M}^k\|_{\mathrm{F}*}=\alpha\operatorname{tr}\boldsymbol{\Sigma}+(1-\alpha)\|\boldsymbol{M}^k\|_{\mathrm{F}}$.

Let us note that $\Delta=\alpha\boldsymbol{U}\boldsymbol{V}^\top+(1-\alpha)\frac{\boldsymbol{M}^k}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}$ delivers this value. Indeed, by the trace property, $\langle\boldsymbol{M}^k,\Delta\rangle=\langle\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top,\alpha\boldsymbol{U}\boldsymbol{V}^\top+(1-\alpha)\frac{\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}\rangle=\alpha\operatorname{tr}\boldsymbol{\Sigma}+(1-\alpha)\|\boldsymbol{M}^k\|_{\mathrm{F}}=\|\boldsymbol{M}^k\|_{\mathrm{F}*}$, which completes the proof.

Proof of lemma 3 follows from eq. (15) with $\alpha=1$. Indeed, $\|\cdot\|_{\mathrm{nuc}}^{\dagger}=1\cdot\|\cdot\|_{\mathrm{op}}+0\cdot\|\cdot\|_{\mathrm{F}}$.

Proof of lemma 5: Since $\|\cdot\|^{\dagger}=\|\cdot\|_{\mathrm{F2}}^{\dagger\dagger}=\|\cdot\|_{\mathrm{F2}}$, the goal is to reach $\|\boldsymbol{M}^k\|_{\mathrm{F2}}=\alpha\sigma_1+(1-\alpha)\|\boldsymbol{M}^k\|_{\mathrm{F}}$.

Let us note that $\Delta=\alpha(u_1v_1^\top)+(1-\alpha)\frac{\boldsymbol{M}^k}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}$ delivers this value. Indeed, by the trace property and singular vectors orthogonality, $\langle\boldsymbol{M}^k,\Delta\rangle=\langle\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top,\alpha u_1v_1^\top+(1-\alpha)\frac{\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top}{\|\boldsymbol{M}^k\|_{\mathrm{F}}}\rangle=\alpha\operatorname{tr}\operatorname{diag}(\sigma_1,0,\ldots,0)+(1-\alpha)\|\boldsymbol{M}^k\|_{\mathrm{F}}=\|\boldsymbol{M}^k\|_{\mathrm{F2}}$, which completes the proof.

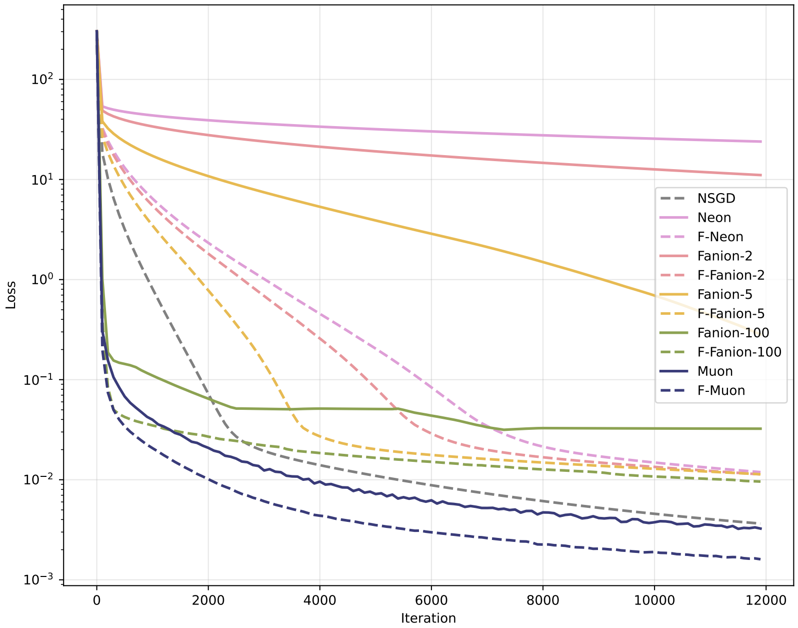
Proof of lemma 6: Since $\|\cdot\|^{\dagger}=\|\cdot\|_{\mathrm{KF-k}}^{\dagger\dagger}=\|\cdot\|_{\mathrm{KF-k}}$, the goal to reach $\|\boldsymbol{M}^t\|_{\mathrm{KF-k}}$.

Let us note that $\Delta = \sum_{i=1}^{k} u_i v_i^\top$ delivers the value. Indeed, $\langle \boldsymbol{M}^t, \Delta \rangle = \langle \boldsymbol{U\Sigma V}^\top, \sum_{i=1}^{k} u_i v_i^\top \rangle = \sum_{i,j=1}^{r,k} \langle u_i \sigma_i v_i^\top, u_j v_j^\top \rangle = \sum_{i=1}^{k} \sigma_i = \|\boldsymbol{M}^t\|_{\mathrm{KF-k}}$, which completes the proof.

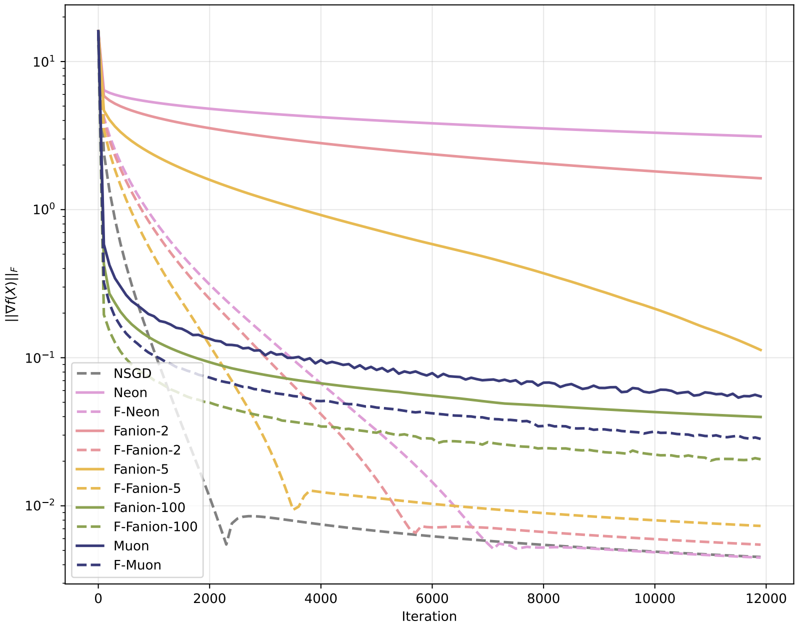
D More plots for Linear Least Squares

E Technical details of the experiments

We compared TRLand with other methods in Google Colab. We used RTX A4000 for CIFAR airbench.

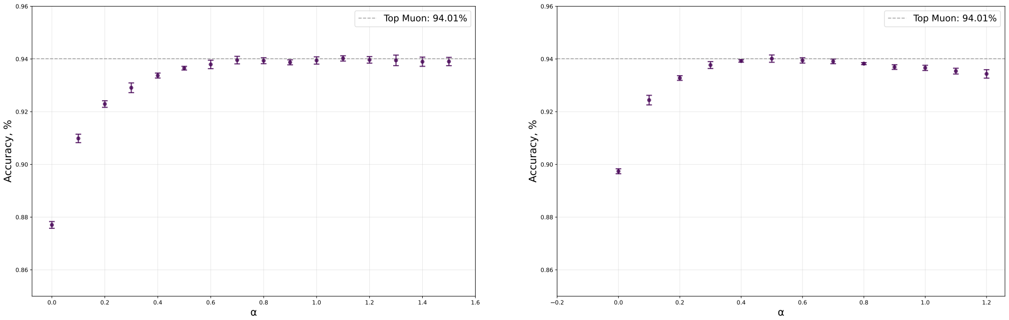


(a) The loss



(b) The Frobenius norm of the gradient

Figure 1: Linear least squares problem for a 500x500 matrix



(a) With parameters tuned for Muon

(b) With parameters tuned for F-Muon

Figure 2: Mean accuracies for different α of F-Muon.

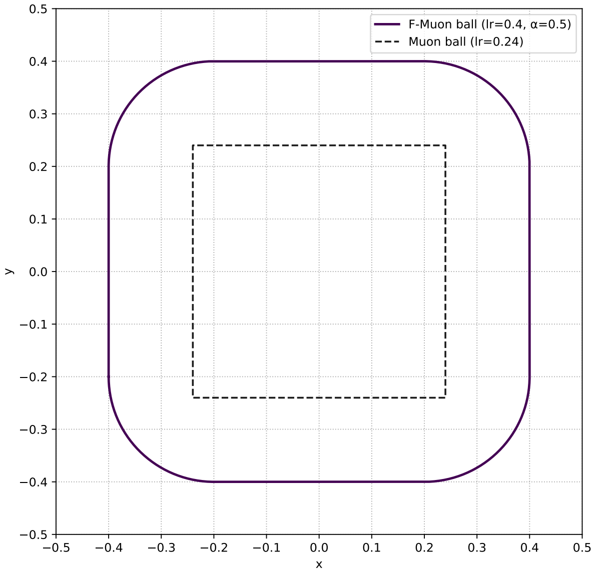


Figure 3: Visualization of lmo balls for Muon and F-Muon.

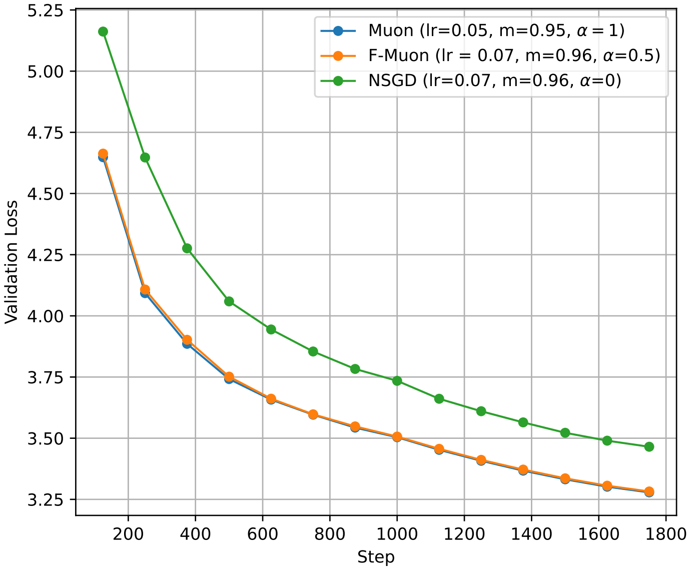
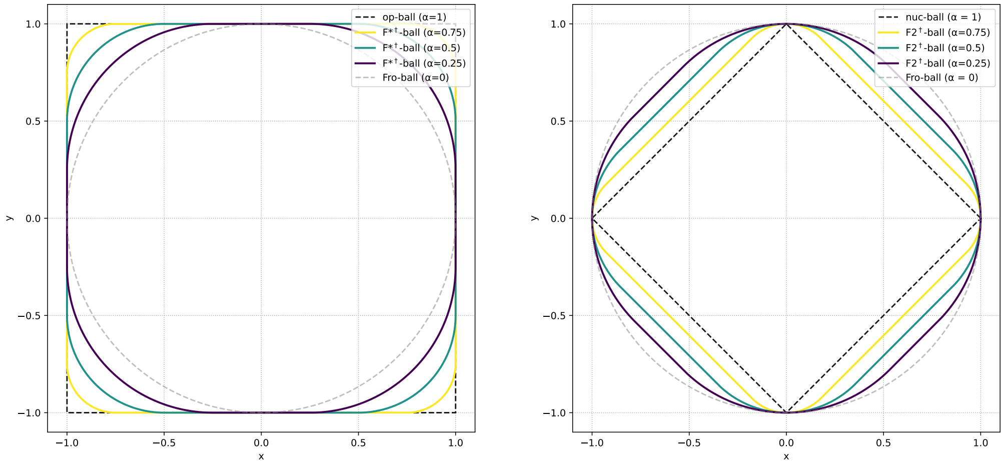


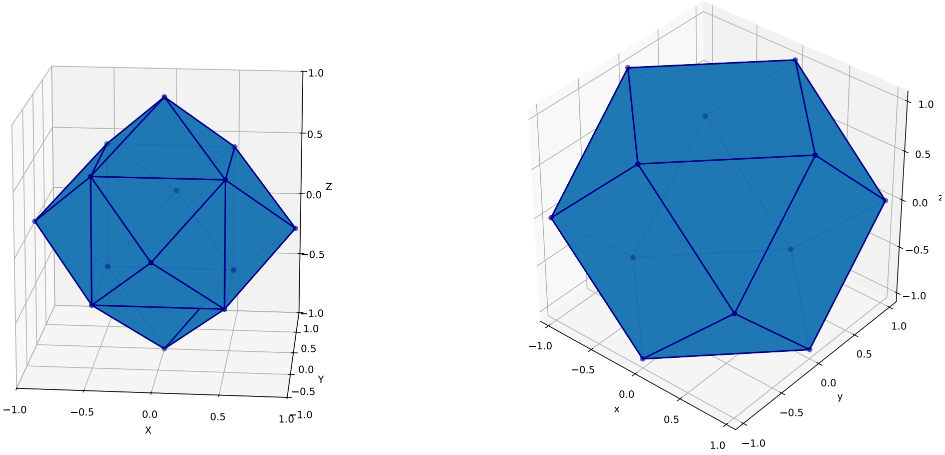
Figure 4: The validation loss for NanoGPT



(a) lmo balls for F-Muon for different α

(b) lmo ball for F-Neon for different α

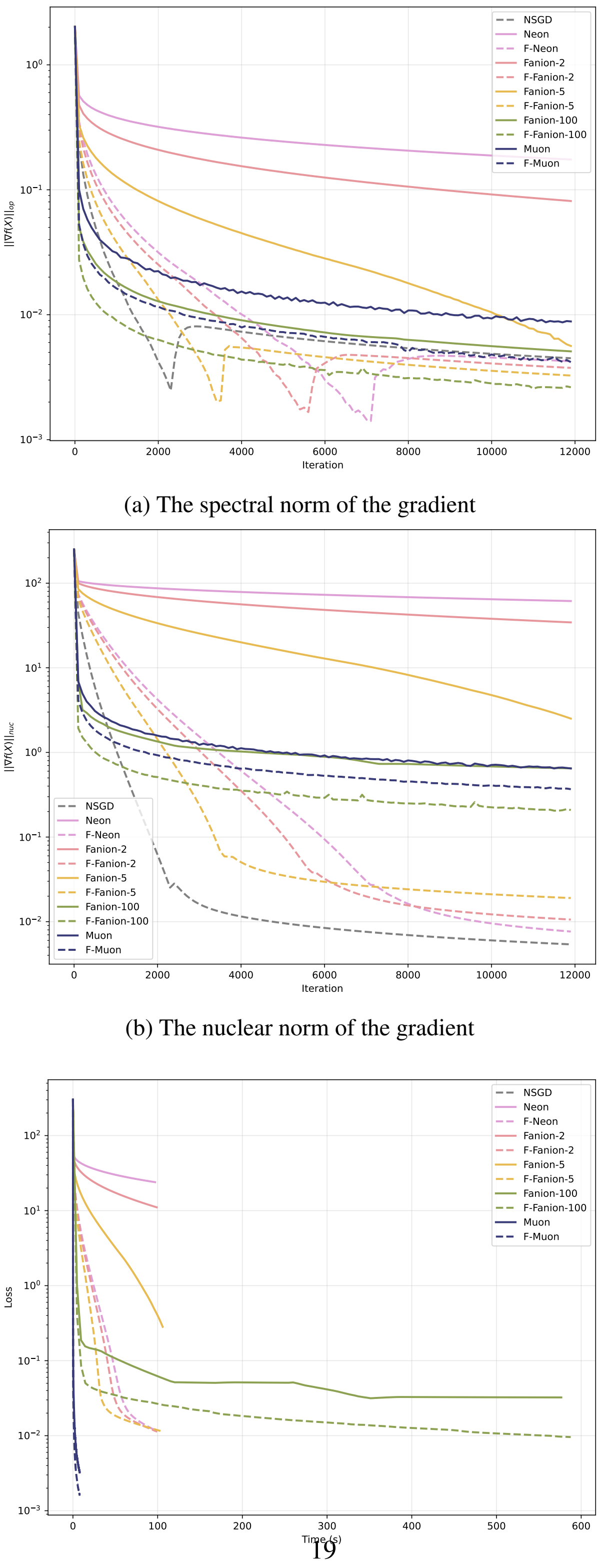
Figure 5: Balls in the duals to F* and F2 norms for different α



(a) Ky Fan 2-norm

(b) Dual to Ky Fan 2-norm

Figure 6: Ky Fan 2-norm and its dual



(a) The spectral norm of the gradient

(b) The nuclear norm of the gradient

(c) The loss over time

Figure 7: More images for Linear least squares problem for a 500x500 matrix