Muon Operates Beyond the Edge of Stability

Niko Čović [*1] **Baraq Lipshitz** [*1] **Alexander Padula** [*1] **Stefan Višnjić** [*1]

Abstract

We study the Muon optimizer using sharpness and provide the first empirical evidence that the Flat Minima Hypothesis also applies to Muon. Further, we derive Muon's implicit preconditioner to analyze its potential Edge of Stability behavior. Experiments on CIFAR-10 show sharpness strongly correlates with generalization gap within each optimizer (Pearson $r = 0.45 - 0.69, p \leq 10^{-4}$), confirming the hypothesis holds for Muon similarly to Adam and SGD. However, Muon operates above the classical sharpness-based Adaptive Edge of Stability threshold that bounds diagonally-preconditioned optimizers like Adam and RMSprop, yet remains stable. Our findings suggest Muon operates in a fundamentally different regime than diagonally-adaptive methods. Defining the exact stability boundary for Muon remains an open question. All code is available at `https://github.com/NikoCovic/DL-project`.

1. Introduction

Muon is a matrix-structured optimizer recently introduced by Jordan et al. (2024). Recent work has empirically observed that it converges in fewer steps than first-order optimizers such as Adam (Kingma, 2014) and Stochastic Gradient Descent (SGD) while incurring less computational overhead than related second-order optimizers like Shampoo (Gupta et al., 2018). While Muon has generated excitement for converging to high accuracy solutions in less wall-clock time than competing methods, the mechanisms underlying this behavior remain a subject of ongoing research.

In this paper, we aim to characterize Muon's behavior in terms of the sharpness of its solutions. We are the first to empirically observe the relationship between sharpness and generalization predicted by the Flat Minima Hypothesis for Muon's solutions. We are also the first to attempt to verify whether Muon is bounded by an Edge of Stability in the same way as first-order optimizers such as Gradient Descent (GD) and Adam. We present a derivation for Muon's preconditioner and empirically show that Muon operates outside of the Edge of Stability as found in diagonally preconditioned optimizers such as Adam.

1.1. Flat Minima Hypothesis

The Flat Minima Hypothesis (Hochreiter & Schmidhuber, 1997) (Keskar et al., 2017) predicts that sharpness should correlate with the model's generalization gap — the difference between training and validation accuracy. It relies on the intuition that the validation loss landscape can be approximated as a stochastically perturbed version of the training landscape. Subsequent work has highlighted the limitations of this hypothesis. Most notably, Dinh et al. (2017) demonstrated that raw sharpness is not scale-invariant; one can alter it via re-parametrization without changing the model's function. This occurs frequently when comparing models across different optimizers. In response, methods such as Adaptive Sharpness-Aware Minimization (ASAM) (Kwon et al., 2021) have been proposed to define a scale-invariant geometric measure of a minimum's size. By validating the Flat Minima Hypothesis for Muon's solutions, we investigate whether there exist sharpness-hacking reparametrizations beyond scale invariance that a novel optimizer like Muon might inadvertently favor.

1.2. Edge of Stability

Although much work exists formalizing Muon's convergence (Shen et al., 2025; Chen et al., 2025; Sfyraki & Wang, 2025; Sato et al.; Chang et al., 2025), none attempts to look for an explanation using the Edge of Stability, a common phenomenon present in state-of-the-art optimizers such as Adam. This phenomenon was originally formalized in (Cohen et al., 2021) for full-batch GD. The authors identify that, throughout training, GD's sharpness seems to hover around a stability threshold value. They reason that divergence does not occur due to higher order Taylor terms which implicitly regularize the sharpness.

In (Cohen et al., 2024) the authors extend the classical EoS to adaptive gradient methods (i.e. algorithms with implicit

[*]Equal contribution; alphabetical order [1]Department of Informatics, ETH Zurich, Switzerland. Correspondence to: Alexander G. Padula <apadula@ethz.ch>.

Submission to the Deep Learning 2025 class., Zurich, Switzerland. 2025. Copyright 2025 by the author(s).

adaptive preconditioners) and coin the term Adaptive EoS (AEoS). Their work finds that the phenomenon still occurs in the sharpness of the effective Hessian $\lambda_{\max}(P_t H_t)$, where P_t is the implicit preconditioner at time t. Another important contribution of the paper is that they reason that the adaptive preconditioner allows the optimizer to reach sharper regions of the loss landscape. However, this work focuses only on diagonally preconditioned algorithms such as Adam and RMSprop (Tieleman & Hinton, 2017).

In our experiments, breaking this assumption using Muon's non-diagonal preconditioner eliminates the classical AEoS phenomenon. While the mechanism that keeps Muon in a stable regime remains unclear, we hypothesize that AEoS disappears because Muon's updates are poorly aligned with the eigenbasis of the effective Hessian $P_t H_t$.

2. Models and Methods

2.1. Optimizers

We formalize all of the optimizers we use in our experiments in Appendix E.

To determine the quality of sharpness as a generalization metric, we use DECOUPLEDMUON, NORMALIZEDMUON, COUPLEDSGD and COUPLEDADAM.

To quantify the AEoS, we compare VANILLAMUON to VANILLAADAM and RMSPROP.

2.2. Sharpness

For parameters w_t, let $H_t := \nabla_w^2 \mathcal{L}(w_t)$ denote the Hessian of the training loss.

Raw sharpness: We define raw (Hessian) sharpness as

$$\mathcal{S}_{\text{raw}}(w_t) := \lambda_{\max}(H_t). \tag{1}$$

Raw sharpness suffers from sensitivity to loss-preserving parameter rescaling, which is particularly an issue when comparing sharpness across solutions derived from different optimizers (Kwon et al., 2021). Two parameterizations with identical functions can yield arbitrarily different sharpness values.

Adaptive sharpness: To avoid conflating optimizer behavior with irrelevant scaling effects, we additionally report adaptive sharpness (Kwon et al., 2021), defined as

$$\mathcal{S}_{\text{adapt}}(w) := \max_{\|T_w^{-1}\epsilon\| \le \rho} \big[\mathcal{L}(w+\epsilon) - \mathcal{L}(w)\big], \tag{2}$$

where T_w is a normalization operator depending on the current parameters. This defines a perturbation set in scale-normalized space, making the measure invariant to loss-preserving parameter rescaling. Unlike classical sharpness it should be comparable across different optimizers.

Effective sharpness: To measure the AEoS, raw sharpness is not sufficient except in the GD setting. In adaptive gradient optimizers the dynamics are governed by the effective hessian $P_t H_t$, where P_t is the implicit preconditioner of the optimizer. We define the effective sharpness as:

$$\mathcal{S}_{\text{eff}}(w_t) = \lambda_{\max}(P_t H_t) \tag{3}$$

2.3. Generalization Benchmark Methodology

We base our first experimental setup on a variant of Airbench (Jordan, 2024), an optimized script that trains a VGG-like CNN to 94% accuracy on the CIFAR-10 (Krizhevsky et al.) image classification dataset.

We modify the Airbench script by (i) adding a callback function called after each epoch to measure sharpness, (ii) supporting training with a fixed learning rate in addition to the original Linear Decay Scheduler (LDS), and (iii) implementing DECOUPLEDMUON, COUPLEDADAM, and COUPLEDSGD in addition to the NORMALIZEDMUON implementation from the original script.

For each optimizer, we perform an extensive hyperparameter sweep using Bayesian optimization with a Gaussian-process (GP) surrogate model. We consider both a fixed learning rate and an LDS schedule. Each sweep consists of 128-512 trials where the GP posterior selects hyperparameters to maximize validation accuracy. All reported results are using the best hyperparameters found by this procedure.

Starting with highly optimized architectures and hyperparameters lends weight to the empirical observations we make regarding Muon's performance compared to Adam and SGD, and allows us to measure sharpness in a practical setting.

We train each run for 16 epochs to ensure convergence. After the final epoch, we report per-optimizer mean validation accuracy, generalization gap, and both raw and adaptive sharpness. For each optimizer, we compute correlation between the sharpness and the generalization gap.

2.4. Edge of Stability

To investigate whether Muon operates at the AEoS, we conduct experiments analogous to those in (Cohen et al., 2024). VANILLAMUON(η, μ) (and other gradient-based algorithms) constructs an implicit adaptive preconditioner P_t as we formalize in Appendix A, differing from VANILLAADAM(η, β_1, β_2) in that the preconditioner of VANILLAMUON(η, μ) is non-diagonal. Using this we compute the sharpness of the effective Hessian $P_t H_t$. Note that this matrix has the same eigenvalues as the symmetric matrix

$P_t^{1/2} H_t P_t^{1/2}$, so we opt to compute the sharpness of this matrix instead.

We measure $\mathcal{S}_{\text{raw}}$ and $\mathcal{S}_{\text{eff}}$ in each epoch of multiple full batch training runs for VANILLAMUON(η, μ), VANILLAADAM(η, β_1, β_2) and RMSPROP(η, α), each with fixed hyperparameters and multiple fixed learning rates η. Note that we restrict our analysis to algorithms without weight decay. This is because weight decay only introduces a slight difference in the constructed preconditioner when coupled, and only a slight difference in the loss when decoupled. As for NORMALIZEDMUON(η, μ), the update operates in a different regime, where the VANILLAMUON(η, μ) updates are projected onto a unit Frobenius ball. We believe the analysis of this algorithm would yield similar results, so we keep it out for simplicity.

We use a fully connected network with the tanh activation, a width of 200, and 5 hidden layers. Each run was executed for 1000 epochs on a subset of the CIFAR-10 dataset, which contained 200 samples per class and was preprocessed with mean normalization, using a train-validation split of 0.8-0.2. Finally, each model was trained on the MSE loss, as it demonstrates the AEoS behavior more clearly (Cohen et al., 2021).

To construct the preconditioner of VANILLAMUON, we use our derivation from Appendix A. Every epoch we store the new and previous weights $W_{t+1}^{(l)}$ and $W_t^{(l)}$ for each layer. This allows us to compute the update matrix as $U\Sigma' V^T = (W_t^{(l)} - W_{t+1}^{(l)})/\eta$ and construct the respective block of the preconditioner as $D_t^{(l)} = U\Sigma^{-1}\Sigma' U^T$, where $M_t^{(l)} = U\Sigma V^T$ is the SVD of the momentum. The preconditioner for the other optimizers is simpler, as it only requires a diagonal equal to the update at that step.

3. Results

3.1. Generalization Benchmark Results

First, Figure 1 shows that under a fixed learning rate, sharpness strongly correlates with generalization gap within each optimizer. For raw sharpness, Pearson correlations range from 0.27 (Adam) to 0.49 (SGD), all statistically significant. Using adaptive sharpness further strengthens this relationship, with correlations increasing to 0.45–0.69 across optimizers (all $p < 10^{-4}$). However, these trends do not transfer across optimizers: raw sharpness varies substantially between methods without corresponding changes in gap, indicating weak cross-optimizer predictiveness.

Beyond correlations, we observe a new pattern in sharpness magnitude. Under a fixed learning rate, Muon exhibits substantially lower adaptive sharpness (ASAM = 0.82 ± 0.229) than SGD (2.06 ± 0.911) and Adam (1.93 ± 0.626). Interestingly, this trend reverses under LDS, where Muon becomes sharper (0.71 ± 0.097) than SGD (0.32 ± 0.072) and Adam (0.56 ± 0.074).

Turning to performance, our results align with prior work: Muon variants consistently achieve higher validation accuracy. Under LDS, Normalized Muon reaches 0.94 ± 0.002, outperforming SGD and Adam (both 0.92 ± 0.002). In contrast, fixed learning rates degrade performance for all optimizers (e.g., Normalized Muon: 0.94 → 0.90; SGD: 0.92 → 0.80).

Finally, despite these differences, the mean generalization gap remains stable across schedulers, staying within 0.04–0.08 for all optimizers.

For completeness, we report the means and standard deviations of the aforementioned accuracy and sharpness measures in Table 1 in the Appendix. We additionally note that, under a scheduled learning rate, we do not observe the correlation predicted by the Flat Minima Hypothesis; see B in the Appendix.

3.2. Edge of Stability

In Figure 2 it is clearly visible that VANILLAMUON(η, μ) seems to operate beyond the classical stability threshold. However, it does not seem to diverge as apparent from the loss trajectory. Interestingly, the raw sharpness appears to exhibit similar behavior to ADAM(η, β_1, β_2) and RMSPROP(η, α) up to convergence (see Figure C), however past convergence the sharpness of VANILLAMUON(η, μ) starts dropping rapidly, a phenomenon not present in the other algorithms.

4. Discussion

4.1. Flat Minima Hypothesis

In aggregate, we find that the Flat Minima Hypothesis holds for solutions obtained with Muon to a similar extent as for those obtained with Adam and SGD. The fact that, in Figure 1, runs across different optimizers remain broadly correlated with generalization suggests that scaling bias largely explains the distinct clustering observed in the Raw Sharpness plot. Nevertheless, the Adaptive Sharpness plot continues to exhibit optimizer-specific clusters. If Adaptive Sharpness fully accounted for all parametrization bias, no such clustering should be expected, as the sharpness measure would depend solely on the generalization gap and noise. Muon's cluster, for instance, exhibits lower Adaptive Sharpness, yet this does not correspond to a comparably reduced generalization gap. Assuming that the Flat Minima Hypothesis holds for some (as yet unidentified) flatness measure, our results indicate that, while Adaptive Sharpness is effective at mitigating scale-related bias, it is not a completely parametrization-invariant measure of flatness.

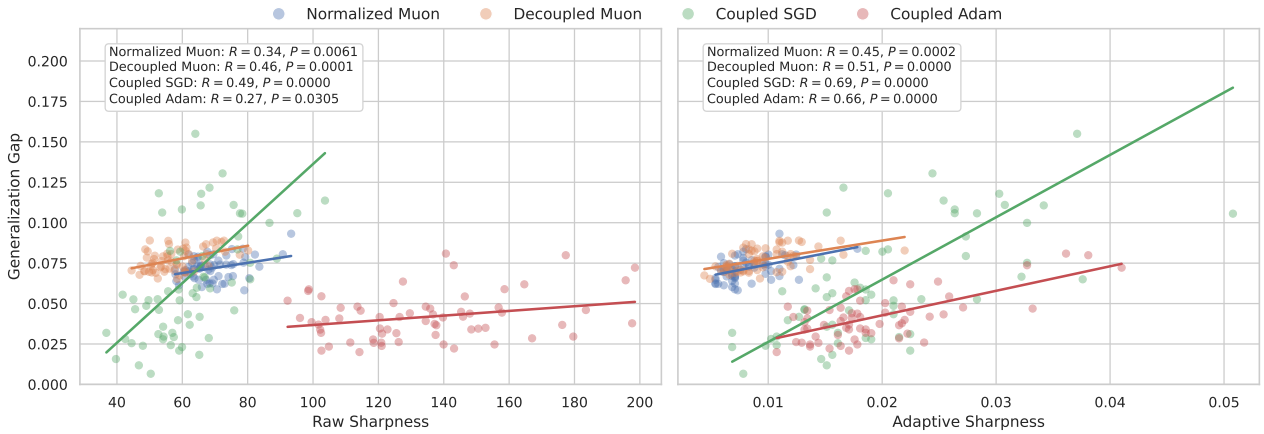

Figure 1. Per-optimizer correlation of raw/adaptive sharpness with generalization gap using a fixed LR.

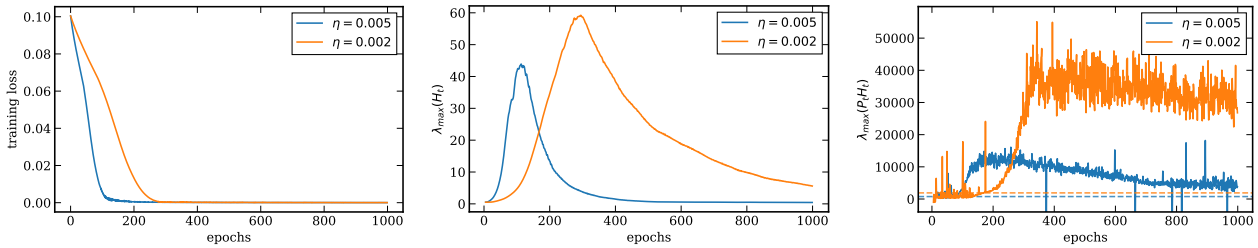

Figure 2. The training loss (left), raw sharpness (middle) and Effective Sharpness (right) of VANILLAMUON(η, μ) with $\mu = 0.9$ and two different η. The dashed lines in the bottom plot represent the threshold where it is expected for Muon to oscillate, computed as $(2 + 2\mu)/\eta$ (Goh, 2017; Cohen et al., 2024). Clearly, VANILLAMUON(η, μ) operates beyond this threshold.

4.2. Edge of Stability

As apparent from Figure 2, VANILLAMUON seems to operate beyond the classical AEoS, but still does not diverge. Empirically, VANILLAMUON may operate within a different regime compared to ADAM and RMSPROP. We hypothesize that this is caused by those algorithms constructing a diagonal preconditioner, while VANILLAMUON constructs a block-diagonal one. Since stability dynamics are controlled by $P_t H_t$, we speculate that the large sharpness of Muon stems from the misalignment of the update directions with the eigenbasis of $P_t H_t$. This is not a problem for diagonal optimizers, as the alignment is expected to be high. Sharpness could therefore be overestimating the "stability" of VANILLAMUON and should be replaced with a different metric. However, we do not explore this in our work so it remains an open question.

A surprising observation also comes from the raw sharpness found in VANILLAMUON's trajectory, seen in the middle plot of Figure 2. Like the raw sharpness of VANILLAADAM and RMSprop, it rises until convergence, as expected given the assumptions of (Cohen et al., 2024). However, post convergence VANILLAMUON seems to drastically reduce its raw sharpness. This does not happen with VANILLAADAM and RMSPROP, where the raw sharpness seems to plateau post convergence. This is another strong hint that VANILLAMUON operates within a different regime.

5. Summary

We were able to reproduce the Flat Minima Hypothesis within optimizers under a realistic setting and show that it also applies to Muon. We show, however, that even using Adaptive Sharpness the hypothesis does not explain all the variance in the generalization gap.

We show that the AEoS does not appear with Muon in terms of sharpness as it does with diagonally preconditioned optimizers such as Adam and RMSprop. We hypothesize that this stems from the misalignment of the preconditioners eigenvalues with the update directions, however we do not explore this in our work and leave this as an open question.

References

Chang, D., Liu, Y., and Yuan, G. On the convergence of muon and beyond. *arXiv preprint arXiv:2509.15816*, 2025.

Chen, L., Li, J., and Liu, Q. Muon optimizes under spectral norm constraints. *arXiv preprint arXiv:2506.15054*, 2025.

Cohen, J. M., Kaur, S., Li, Y., Kolter, J. Z., and Talwalkar, A. Gradient descent on neural networks typically occurs at the edge of stability. *arXiv preprint arXiv:2103.00065*, 2021.

Cohen, J. M., Ghorbani, B., Krishnan, S., Agarwal, N., Medapati, S., Badura, M., Suo, D., Cardoze, D., Nado, Z., Dahl, G. E., and Gilmer, J. Adaptive gradient methods at the edge of stability, 2024. URL https://arxiv.org/abs/2207.14484.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning (ICML)*, pp. 1019–1028. PMLR, 2017.

Goh, G. Why momentum really works. *Distill*, 2(4):e6, 2017.

Gupta, V., Koren, T., and Singer, Y. Shampoo: Preconditioned stochastic tensor optimization, 2018. URL https://arxiv.org/abs/1802.09568.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Jordan, K. 94URL https://arxiv.org/abs/2404.00498.

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

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.

Kingma, D. P. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krizhevsky, A., Nair, V., and Hinton, G. Cifar-10 (canadian institute for advanced research). URL http://www.cs.toronto.edu/~kriz/cifar.html.

Kwon, J., Kim, J., Park, H., and Choi, I. K. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International conference on machine learning*, pp. 5905–5914. PMLR, 2021.

Liu, J., Su, J., Yao, X., Jiang, Z., Lai, G., Du, Y., Qin, Y., Xu, W., Lu, E., Yan, J., et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Sato, N., Naganuma, H., and Iiduka, H. Convergence bound and critical batch size of muon optimizer. 2025. *URL https://api. semanticscholar. org/CorpusID*, 280140878.

Sfyraki, M.-E. and Wang, J.-K. Lions and muons: Optimization via stochastic frank-wolfe. *arXiv preprint arXiv:2506.04192*, 2025.

Shen, W., Huang, R., Huang, M., Shen, C., and Zhang, J. On the convergence analysis of muon. *arXiv preprint arXiv:2505.23737*, 2025.

Tieleman, T. and Hinton, G. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. In *Technical report*. University of Toronto, 2017.

Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. W. Pyhessian: Neural networks through the lens of the hessian. In *2020 IEEE international conference on big data (Big data)*, pp. 581–590. IEEE, 2020.

A. Muon Preconditioner Derivation

In this section, we formally derive the implicit preconditioner of VANILLAMUON(η, μ). We aim to rewrite the update in terms of the collected vectorized network weights $w_t = \begin{bmatrix}\text{vec}(W_t^{(1)})^T & \dots & \text{vec}(W_t^{(L)})^T\end{bmatrix}^T$ and a preconditioner P_t as $w_{t+1} = w_t - \eta P_t m_t$, where $m_t = \begin{bmatrix}\text{vec}(M_t^{(1)})^T & \dots & \text{vec}(M_t^{(L)})^T\end{bmatrix}^T$ is the vectorized concatenated momentum. We begin by deriving a per-layer preconditioner and extend it to the entire network. Let $\mathcal{W}_t = \{W_t^{(l)} | 1 \leq l \leq L\}$ represent the set of all rectangular weight matrices optimized by Muon at time t. The authors in (Liu et al., 2025) have already shown that Muon uses Newton-Schulz to approximate UV^T where $M_t^{(l)} = U\Sigma V^T$ is the SVD of the momentum. The implicit preconditioner then becomes $(M_t^{(l)} M_t^{(l)T})^{-1/2}$. Here, we derive the preconditioner assuming the approximate orthogonalization using Newton-Schulz.

Proposition A.1. *The Newton-Schulz update of* VANILLAMUON(η, μ) $O_t^{(l)}$ *can be written as* $O_t^{(l)} = U\Sigma' V^T$*, where* $M_t^{(l)} = U\Sigma V^T$ *is the SVD of the corresponding momentum matrix and* Σ' *is some diagonal matrix.*

Proof. Let $t > 0$ and $1 \leq l \leq L$ be an arbitrary time and weight matrix index. Let $M_t^{(l)} = U\Sigma V^T$ be the singular value decomposition of the momentum matrix.

The update step of VANILLAMUON(η, μ) is $W_{t+1}^{(l)} = W_t^{(l)} - \eta O_t^{(l)}$, where η is the learning rate and $O_t^{(l)} = \text{Newton-Schulz5}(M_t^{(l)}; a, b, c)$ for some coefficients a, b, c.

Let $X_{k+1} = aX_k + b(X_k X_k^T)X_k + c(X_k X_k^T)^2 X_k$ be a single Newton-Schulz5 iteration for some matrix X_k. By setting $X_0 = M_t^{(l)}$ and inserting the SVD of $M_t^{(l)}$ into the iteration, we can easily obtain $X_1 = U(a\Sigma + b\Sigma^3 + c\Sigma^5)V^T$. Notice that $a\Sigma + b\Sigma^3 + c\Sigma^5$ is also a diagonal matrix, and is in fact the diagonal singular value matrix of X_1. So, by applying this step multiple times, only the diagonal matrix of the SVD of X_0 changes, while U and V stay the same. Therefore, after applying 5 steps, we obtain a matrix UDV^T, where D is some diagonal matrix. So, $O_t^{(l)} = \text{Newton-Schulz5}(M_t^{(l)}; a, b, c) = U\Sigma' V^T$ □

Proposition A.2. *For a single rectangular weight matrix* $W_t^{(l)}$ *at time* t*, the non-vectorized preconditioner of* VANILLAMUON(η, μ) *is* $D_t^{(l)} = US^{-1}U^T$*, where* $M_t^{(l)} = U\Sigma V^T$ *is the SVD of the corresponding momentum matrix and* $S = \Sigma\Sigma'^{-1}$ *for* $O_t^{(l)} = U\Sigma' V^T$*. being the SVD of the orthogonalized Momentum using Newton-Schulz.*

Proof. Let $t > 0$ and $1 \leq l \leq L$ be an arbitrary time and weight matrix index. Let $M_t^{(l)} = U\Sigma V^T$ be the singular value decomposition of the momentum matrix.

The update step of VANILLAMUON(η, μ) is $W_{t+1}^{(l)} = W_t^{(l)} - \eta U\Sigma' V^T$, where η is the learning rate. Let $S = \Sigma\Sigma'^{-1}$ and $D_t^{(l)} = US^{-1}U^T$. We then have:

$$D_t^{(l)} M_t^{(l)} = US^{-1}U^T U\Sigma V^T = US^{-1}\Sigma V^T = U\Sigma' V^T \tag{4}$$

□

Proposition A.3. *Given the non-vectorized preconditioner* $D_t^{(l)}$ *of* VANILLAMUON(η, μ)*, the preconditioner of the vectorized block* $w_t^{(l)} = vec(W_t^{(l)})$ *is* $P_t^{(l)} = I \otimes D_t^{(l)}$*.*

Proof.

$$\text{vec}(W_{t+1}^{(l)}) = \text{vec}(W_t^{(l)} - \eta D_t^{(l)} M_t^{(l)}) = w_t^{(l)} - \eta \text{vec}(D_t^{(l)} M_t^{(l)}) = w_t^{(l)} - \eta(I \otimes D_t^{(l)}) m_t^{(l)} \tag{5}$$

□

Proposition A.4. *Given the vectorized preconditioner* $P_t^{(l)}$ *of* VANILLAMUON*, the preconditioner combined for all the weights* $\mathcal{W}$ *is* $P_t = \bigoplus_{l=1}^{L} P_t^{(l)}$

Proof.

$$
\begin{aligned}
w_{t+1} &= \begin{bmatrix} w_{t+1}^{(1)T} & \dots & w_{t+1}^{(L)T} \end{bmatrix}^T = \begin{bmatrix} w_t^{(1)T} & \dots & w_t^{(L)T} \end{bmatrix}^T - \eta \begin{bmatrix} (P_t^{(1)} m_t^{(1)})^T & \dots & (P_t^{(L)} m_t^{(L)})^T \end{bmatrix}^T \\
&= w_t - \eta \begin{bmatrix} P_t^{(1)} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & P_t^{(L)} \end{bmatrix} m_t = w_t - \eta \left(\bigoplus_{l=1}^{L} P_t^{(l)} \right) m_t
\end{aligned} \tag{6}
$$

□

It is also relevant to show that raising the preconditioner P_t to a power p can be done by simply raising the non-vectorized preconditioner matrices $D_t^{(l)}$ to p. This is important for computing the Effective Sharpness of the matrix $P_t^{1/2} H_t P_t^{1/2}$.

Proposition A.5. *Computing P_t^p can be done by simply taking the power of each individual non-vectorized matrix $D_t^{(l)}$.*

Proof.

$$
P_t^p = \left(\bigoplus_{l=1}^{L} P_t^{(l)} \right)^p = \bigoplus_{l=1}^{L} P_t^{(l)p} = \bigoplus_{l=1}^{L} (I \otimes D_t^{(l)})^p = \bigoplus_{l=1}^{L} (I \otimes D_t^{(l)p}) \tag{7}
$$

□

B. Flat Minima Hypothesis

C. Edge of Stability Graphs

In this section, we provide the plots for the AEoS of VANILLAADAM and RMSPROP.

D. Sharpness Computation Details

To compute the raw and Effective Sharpness in our experiments we utilize a power iteration approach inspired by PyHessian (Yao et al., 2020). In addition to their implementation, we add support for a preconditioner to compute the effective Hessian sharpness. The algorithm can be found in 1

Note that in Algorithm 1, it is not relevant to store the entire Hessian H_t and preconditioner P_t, as $H_t v$ can be computed efficiently via the Pearlmutter trick and $P_t^{1/2} v$ can be computed very efficiently if it is diagonal and quite efficiently if it is block diagonal. To compute the raw sharpness, all that has to be done is to pass $P_t = I$. In our computations, we used a tolerance of $\tau = 10^{-9}$ and a maximum number of iterations $n_{\text{max}} = 200$.

Table 1. For each Learning Rate scheduler (LR) and Optimizer pairing, we report Validation Accuracy (Val Acc), Generalization Gap (Gap), Raw Sharpness (Raw), and Adaptive Sharpness (ASAM) scores. Each entry aggregates the mean and standard deviation across 64 runs.

LR	OPTIMIZER	TRAIN ACC	VAL ACC	ACC GAP	RAW ($\times 10^2$)	ASAM ($\times 10^{-2}$)
FIXED	NORMALIZED MUON	0.97 ± 0.005	0.90 ± 0.007	0.07 ± 0.007	0.69 ± 0.075	0.82 ± 0.229
FIXED	DECOUPLED MUON	0.98 ± 0.004	0.90 ± 0.006	0.08 ± 0.007	0.58 ± 0.080	0.96 ± 0.307
FIXED	COUPLED SGD	0.87 ± 0.012	0.80 ± 0.057	0.07 ± 0.051	0.62 ± 0.135	2.06 ± 0.911
FIXED	COUPLED ADAM	0.91 ± 0.007	0.87 ± 0.012	0.04 ± 0.014	1.34 ± 0.266	1.93 ± 0.626
LDS	NORMALIZED MUON	1.00 ± 0.001	0.94 ± 0.002	0.06 ± 0.002	0.45 ± 0.078	0.71 ± 0.097
LDS	DECOUPLED MUON	1.00 ± 0.001	0.93 ± 0.002	0.07 ± 0.002	0.13 ± 0.076	0.70 ± 0.283
LDS	COUPLED SGD	0.98 ± 0.003	0.92 ± 0.002	0.06 ± 0.003	0.03 ± 0.005	0.32 ± 0.072
LDS	COUPLED ADAM	0.99 ± 0.002	0.92 ± 0.002	0.07 ± 0.003	1.05 ± 0.112	0.56 ± 0.074

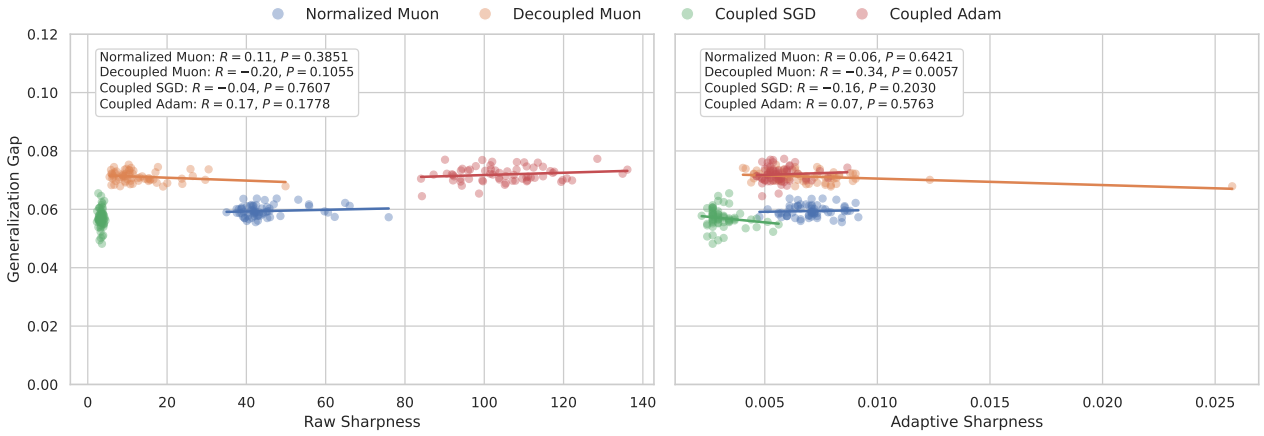

Figure 3. Per-optimizer correlation of raw/adaptive sharpness with generalization gap using a LDS.

Algorithm 1 Sharpness Power Iteration

Input: preconditioner P_t, Hessian H_t, maximum iterations n_{max}, tolerance τ
Initialize $v_0 \sim \mathcal{N}(0, I_p)$.
Normalize $v_1 = v_0 / \|v_0\|_2$
Initialize $\lambda_{\text{max}} = \text{none}$
for $i = 1$ **to** n_{max} **do**
  $Mv_i = P_t^{1/2} H_t P_t^{1/2} v_i$
  $\lambda_{\text{temp}} = v_i^T M v_i$
  $v_{i+1} = Mv_i / \|Mv_i\|_2$
  if $\lambda_{\text{max}} = \text{none}$ **then**
    $\lambda_{\text{max}} = \lambda_{\text{temp}}$
  else if $|\lambda_{\text{max}} - \lambda_{\text{temp}}| / |\lambda_{\text{max}} + 10^{-7}| < \tau$ **then**
    End for loop
  end if
  $\lambda_{\text{max}} = \lambda_{\text{temp}}$
end for
return λ_{max}

E. Optimizer Definitions

E.1. Muon Optimizer

The Muon optimizer (Jordan et al., 2024) is defined as follows: At time t, for a weight matrix $W_t^{(l)}$ with index l of a neural network with weights $w_t \in \mathbb{R}^p$ and loss function $\mathcal{L} : \mathbb{R}^p \to \mathbb{R}$, the update is computed as:

$$\begin{aligned} M_t^{(l)} &= \mu M_{t-1}^{(l)} + \nabla_{W_t^{(l)}} \mathcal{L}(w_t) \\ O_t^{(l)} &= \text{Newton-Schulz5}(M_t^{(l)}; a, b, c) \\ W_{t+1}^{(l)} &= W_t^{(l)} - \eta_t O_t^{(l)} \end{aligned} \tag{8}$$

We refer to this rule as VANILLAMUON(η_t, μ). In Liu et al. (2025), the authors extend Muon by adding decoupled weight decay. The update rule then becomes $W_{t+1}^{(l)} = W_t^{(l)} - \eta_t (O_t^{(l)} + \lambda W_t^{(l)})$, where λ is the weight decay constant. We refer to this rule as DECOUPLEDMUON(η_t, μ, λ) since it uses decoupled weight decay. The final update used in Airbench Jordan (2024) involves performing a normalization step before the update. I.e., before the new weights at $t+1$ are computed, the weights are normalized as $W_t^{(l)} = \sqrt{m_l} W_t^{(l)} / \|W_t^{(l)}\|_F$, where $W_t^{(l)} \in \mathbb{R}^{m_l \times n_l}$. We refer to this update rule as NORMALIZEDMUON(η_t, μ).

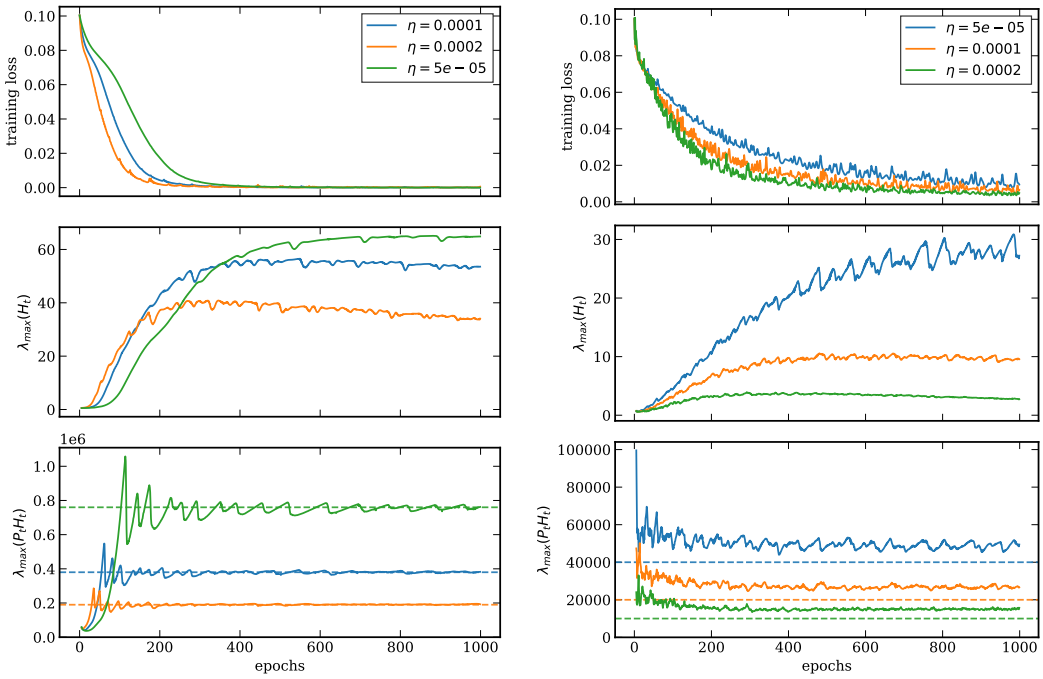

Figure 4. The training loss (top), raw sharpness (middle) and Effective Sharpness (bottom) of VANILLAADAM(η, β_1, β_2) with $\beta_1 = 0.9, \beta_2 = 0.999$ and three different η (left) and RMSPROP(η, α) (right) with $\alpha = 0.99$ and three different η. The dashed lines in the bottom plots represent the threshold where it is expected for the algorithms to oscillate, computed as $2/\eta$ for RMSPROP(η, α) and $(2 + 2\beta_1)/\eta$ for VANILLAADAM(η, β_1, β_2) (Goh, 2017). VANILLAADAM oscillates around the threshold and RMSPROP slightly above the threshold, exactly as observed in previous work (Cohen et al., 2024)

E.2. Other Optimizers

We compare Muon to several other optimizers, including a non-adaptive optimizer and a diagonally preconditioned optimizer. We define Gradient Descent with weight decay as COUPLEDGD(η_t, λ) with coupled weight decay.

The second algorithm we use is Adam, which we refer to as VANILLAADAM$(\eta_t, \beta_1, \beta_2)$ with learning rate η_t at time t and the momentum (β_1) and squared exponential average (β_2) coefficients. As with Muon, we also use a coupled weight decay version of Adam in our experiments, named COUPLEDADAM$(\eta_t, \beta_1, \beta_2, \lambda)$ with the weight decay constant λ.

Finally, we use RMSprop, named RMSPROP(η_t, α) with learning rate η_t at time t and squared gradient exponential average coefficient α. Since this algorithm is only used for AEoS analysis, we do not use a weight decay alternative.