Beyond MuP: 4. Preserving Parameter Stability

Su Jianlin

2026-04-24

Through derivations and calculations in previous articles, we can observe that the three stability metrics proposed in the first article *Beyond MuP: 1. Three Characteristics of a Good Model* can generally be divided into *parameter stability* and *increment stability*. In *Beyond MuP: 2. Linear Layers and Steepest Descent* and *Beyond MuP: 3. Special Cases Require Special Treatment*, we demonstrated the process of integrating increment stability with steepest descent to obtain new update rules (optimizers).

However, for parameter stability, we previously only considered initialization. The task of this article is to explore how to maintain parameter stability throughout the entire training process, thus completing the practical application of the theory.

1 Background

Taking *Beyond MuP: 2. Linear Layers and Steepest Descent* as an example, the three stability metrics are:

$$\text{Forward Stability:} \quad \max_{\|\boldsymbol{x}\|_{RMS}=1} \|\boldsymbol{x}\boldsymbol{W}\|_{RMS} = \sqrt{\frac{d_{in}}{d_{out}}}\|\boldsymbol{W}\|_2 \tag{1}$$

$$\text{Dependency Stability:} \quad \max_{\|\boldsymbol{x}_1\|_{RMS}=\|\boldsymbol{x}_2\|_{RMS}=1} \|\boldsymbol{x}_1\boldsymbol{W} - \boldsymbol{x}_2\boldsymbol{W}\|_{RMS} = 2\sqrt{\frac{d_{in}}{d_{out}}}\|\boldsymbol{W}\|_2 \tag{2}$$

$$\text{Update Stability:} \quad \max_{\|\boldsymbol{x}\|_{RMS}=1} \|\boldsymbol{x}(\boldsymbol{W}+\Delta\boldsymbol{W}) - \boldsymbol{x}\boldsymbol{W}\|_{RMS} = \sqrt{\frac{d_{in}}{d_{out}}}\|\Delta\boldsymbol{W}\|_2 \tag{3}$$

where $\boldsymbol{W} \in \mathbb{R}^{d_{in}\times d_{out}}$ is the parameter of a linear layer. We desire these three metrics to be $\Theta(1)$, which means we want the norm of the parameter and its increment to satisfy $\|\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{out}/d_{in}})$ and $\|\Delta\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{out}/d_{in}})$, respectively. In *Beyond MuP: 3. Special Cases Require Special Treatment*, we performed similar calculations for layers such as Embedding and LM Head, with analogous conclusions, though the corresponding norms differ.

We used the increment condition as a stability metric and, based on the principle of steepest descent under "stable acceleration," derived theoretically optimal update rules. For linear layers, this leads to the Muon optimizer:

$$\operatorname{argmin}_{\|\Delta\boldsymbol{W}\|_2\leq\eta\sqrt{\frac{d_{out}}{d_{in}}}} \operatorname{tr}(\boldsymbol{G}^\top\Delta\boldsymbol{W}) \qquad \Rightarrow \qquad \Delta\boldsymbol{W} = -\eta\sqrt{\frac{d_{out}}{d_{in}}}\operatorname{msign}(()\boldsymbol{G}) \tag{4}$$

For parameter stability, we previously only required that initialization satisfies $\|\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{out}/d_{in}})$. How to ensure such parameter stability is maintained throughout the entire training process remains unclear.

2 Preliminary Thoughts

How can we ensure that $\boldsymbol{W}$ maintains $\|\boldsymbol{W}\|_2 = \Theta(\sqrt{d_{out}/d_{in}})$? More generally, given a parameter $\boldsymbol{\omega}$, which could be a vector, matrix, or even a higher-order tensor, a norm $\|\cdot\|$, typically induced by forward or dependency stability, and a scale τ, how can we ensure that $\boldsymbol{\omega}$ maintains $\|\boldsymbol{\omega}\| = \Theta(\tau)$ during training?

A naive idea is to directly enforce $\|\boldsymbol{\omega}\| = \tau$ (where τ can be replaced by a constant multiple, but this does not affect the following discussion). The simplest implementation is to re-scale the norm to τ after each optimization step via normalization (e.g., Hyperball and Nemotron-Flash). Another approach is to replace the original model parameterization with normalized ones, i.e., change $\boldsymbol{f}(\boldsymbol{x};\boldsymbol{\omega})$ to $\boldsymbol{f}(\boldsymbol{x};\tau\boldsymbol{\omega}/\|\boldsymbol{\omega}\|)$, which theoretically achieves a similar effect.

A more advanced approach combines this with the steepest descent idea to adjust the update rule, as discussed in articles *Steepest Descent on Manifolds: 1. SGD + Hypersphere* and *Steepest Descent on Manifolds: 4. Muon + Spectral Sphere*, and in the paper *Controlled LLM Training on Spectral Sphere*. This approach is more elegant in principle but more complex in practice, often requiring solving a nonlinear equation to obtain the exact update.

However, should we really insist on strictly controlling a parameter's norm to a fixed value? Intuitively, the parameter norm should be determined by the training process itself; at best, we set a prior range for it. Although some work shows that setting an appropriate fixed norm does not harm performance, it still disrupts the original training dynamics and may require extra effort to adapt.

Therefore, the view proposed in this article is that we only need to ensure $\|\boldsymbol{\omega}\| = \mathcal{O}(\tau)$. Specifically, we aim to guarantee $\|\boldsymbol{\omega}\| \leq \tau$ at every step, while leaving the exact valueincluding whether it reaches $\Theta(\tau)$to be determined by the training algorithm, without further intervention.

3 Post Clip

The next question is: how to achieve $\|\boldsymbol{\omega}\| \leq \tau$? More specifically, suppose the original update rule for $\boldsymbol{\omega}$ is

$$\boldsymbol{\omega}_t = \boldsymbol{\omega}_{t-1} - \eta\boldsymbol{\phi}_t \tag{5}$$

How should we modify it to ensure $\|\boldsymbol{\omega}_t\| \leq \tau$? There are many possible methods; for instance, normalization as mentioned earlier is one option. Among them, we aim to select the one that causes the *least disruption* to the optimization process. That is, given $\boldsymbol{\omega}$ and a norm $\|\cdot\|$, we aim to minimize the modification to make its norm at most τ. Formally, this is defined as:

$$\lfloor\boldsymbol{\omega}\rfloor_{\|\cdot\|\leq\tau} = \mathrm{argmin}_{\|\tilde{\omega}\|\leq\tau} \|\boldsymbol{\omega} - \tilde{\boldsymbol{\omega}}\|_{RMS} \tag{6}$$

Familiar readers of convex optimization will recognize this as the projection of $\boldsymbol{\omega}$ into a ball of radius τ under the specified norm. The key idea is to achieve the goal of bounded norm while minimizing the perturbation to the original parameter $\boldsymbol{\omega}$, thus inducing a specific projection or clipping operation.

The exact computation of $\lfloor\boldsymbol{\omega}\rfloor_{\|\cdot\|\leq\tau}$ depends on the specific norm and will be detailed later. With this operator, one possible scheme is to apply it after each update:

$$\boldsymbol{\omega}_t = \lfloor\boldsymbol{\omega}_{t-1} - \eta\boldsymbol{\phi}_t\rfloor_{\|\cdot\|\leq\tau} \tag{7}$$

We call this scheme "Post Clip," which is simple and intuitive. However, it may feel "non-smooth": if initialization starts below τ, the parameter norm increases slowly until reaching τ, at which point clipping "suddenly" activates. While continuous, this is not differentiable at the boundarysimilar to the $\max(x, 0)$ function.

4 Pre Decay

If this non-smoothness is undesirable, one can imitate weight decay by distributing the penalty across updates. Still starting from update rule (5), suppose $\|\boldsymbol{\phi}_t\| \leq \tau$, then by the triangle inequality:

$$\|\boldsymbol{\omega}_t\| = \|\boldsymbol{\omega}_{t-1} - \eta\boldsymbol{\phi}_t\| \leq \|\boldsymbol{\omega}_{t-1}\| + \eta\tau, \tag{8}$$

meaning the norm could grow by up to $\eta\tau$ per step, accumulating out of control over time.

To prevent this, we can pre-process $\boldsymbol{\omega}_{t-1}$ before subtracting $\eta\boldsymbol{\phi}_t$, reducing its norm just enough to counteract the potential increase. Following the weight decay analogy:

$$\boldsymbol{\omega}_t = \lfloor\boldsymbol{\omega}_{t-1}\rfloor_{\|\cdot\|\leq(1-\eta)\|\boldsymbol{\omega}_{t-1}\|} - \eta\boldsymbol{\phi}_t \tag{9}$$

That is, first reduce $\boldsymbol{\omega}_{t-1}$'s norm to $(1-\eta)$ times its original value, then update. This gives:

$$\|\boldsymbol{\omega}_t\| \leq (1-\eta)\|\boldsymbol{\omega}_{t-1}\| + \eta\tau \leq \max(\|\boldsymbol{\omega}_{t-1}\|, \tau),$$

and by induction, $\|\boldsymbol{\omega}_t\| \leq \max(\|\boldsymbol{\omega}_0\|, \tau)$. Thus, if initialization satisfies $\|\boldsymbol{\omega}_0\| \leq \tau$, then $\|\boldsymbol{\omega}_t\| \leq \tau$ holds throughout training. This result is independent of the choice of normit only relies on the triangle inequality. The minimal-modification operation to reduce norm is precisely the clipping operator in (6), making its use natural here.

We call this "Pre Decay." Unlike Post Clip, which uses a static threshold τ, Pre Decay uses a dynamic threshold $(1-\eta)\|\boldsymbol{\omega}_{t-1}\|$ and always triggers a reduction, making the process smoother. Hence, we call it "decay" instead of "clip," generalizing conventional weight decay.

5 Simple Example

So far, we have established a general framework for constraining parameter norms, with Post Clip and Pre Decay as two schemes, centered on the clipping operator $\lfloor\boldsymbol{\omega}\rfloor_{\|\cdot\|\leq\tau}$ defined in (6). While formally defined, practical implementation depends on the norm.

Consider a simple example using $\|\cdot\|_{RMS}$. For vectors, this equals the L2 norm; for matrices, it equals the Frobenius norm. We have:

$$\lfloor\boldsymbol{\omega}\rfloor_{\|\cdot\|_{RMS}\leq\tau} = \mathrm{argmin}_{\|\tilde{\boldsymbol{\omega}}\|_{RMS}\leq\tau} \|\boldsymbol{\omega} - \tilde{\boldsymbol{\omega}}\|_{RMS} = \min\left(1, \frac{\tau}{\|\boldsymbol{\omega}\|_{RMS}}\right)\boldsymbol{\omega} \tag{10}$$

The proof is left to the reader (if stuck, consult Kimi). Substituting $\tau = (1-\eta)\|\boldsymbol{\omega}\|_{RMS}$ yields:

$$\lfloor\boldsymbol{\omega}\rfloor_{\|\cdot\|_{RMS}\leq(1-\eta)\|\boldsymbol{\omega}\|_{RMS}} = \min\left(1, \frac{(1-\eta)\|\boldsymbol{\omega}\|_{RMS}}{\|\boldsymbol{\omega}\|_{RMS}}\right)\boldsymbol{\omega} = (1-\eta)\boldsymbol{\omega} \tag{11}$$

Plugging into (9) gives:

$$\boldsymbol{\omega}_t = (1-\eta)\boldsymbol{\omega}_{t-1} - \eta\boldsymbol{\phi}_t \tag{12}$$

This is precisely conventional weight decay. Thus, Pre Decay under RMS norm reduces to standard weight decay, the minimal modification scheme to preserve RMS (equivalently, L2 or Frobenius) norm bounds.

6 Singular Clipping

We now turn to the main eventmatrix parameters and Muon. Recall the Muon update:

$$\boldsymbol{W}_t = \boldsymbol{W}_{t-1} - \eta\lambda\boldsymbol{\Phi}_t, \quad \boldsymbol{\Phi}_t = \frac{1}{\lambda}\sqrt{\frac{d_{out}}{d_{in}}}\,\mathrm{msign}(()\boldsymbol{G}_t) \tag{13}$$

Let $\tau = \frac{1}{\lambda}\sqrt{\frac{d_{out}}{d_{in}}}$, so $\|\boldsymbol{\Phi}_t\|_2 = \tau$. To ensure $\|\boldsymbol{W}_t\|_2 \le \tau$, two schemes are:

$$\text{Post Clip:} \quad \boldsymbol{W}_t = \lfloor \boldsymbol{W}_{t-1} - \eta\lambda\boldsymbol{\Phi}_t \rfloor_{\|\cdot\|_2 \le \tau} \tag{14}$$

$$\text{Pre Decay:} \quad \boldsymbol{W}_t = \lfloor \boldsymbol{W}_{t-1} \rfloor_{\|\cdot\|_2 \le (1-\eta\lambda)\|\boldsymbol{W}_{t-1}\|_2} - \eta\lambda\boldsymbol{\Phi}_t \tag{15}$$

Now compute $\lfloor \boldsymbol{W} \rfloor_{\|\cdot\|_2 \le \tau}$, which is:

$$\lfloor \boldsymbol{W} \rfloor_{\|\cdot\|_2 \le \tau} = \text{argmin}_{\|\tilde{\boldsymbol{W}}\|_2 \le \tau} \|\boldsymbol{W} - \tilde{\boldsymbol{W}}\|_F \tag{16}$$

This problem's optimal solution is known as "Singular Value Clipping (SVC)" or mclip(; :)

$$\lfloor \boldsymbol{W} \rfloor_{\|\cdot\|_2 \le \tau} = \text{mclip}((;)W; \tau) = \boldsymbol{U} \min(\boldsymbol{\Sigma}, \tau)\boldsymbol{V}^\top \tag{17}$$

where $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ is the SVD of $\boldsymbol{W}$ and $\min(\boldsymbol{\Sigma}, \tau)$ truncates singular values to at most τ. With this notation:

$$\text{Post Clip:} \quad \boldsymbol{W}_t = \text{mclip}((;)W_{t-1} - \eta\lambda\boldsymbol{\Phi}_t; \tau) \tag{18}$$

$$\text{Pre Decay:} \quad \boldsymbol{W}_t = \text{mclip}((;)W_{t-1}; (1-\eta\lambda)\|\boldsymbol{W}_{t-1}\|_2) - \eta\lambda\boldsymbol{\Phi}_t \tag{19}$$

7 Derivation

We now prove (17). Let $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the SVD. Then:

$$\|\boldsymbol{W} - \tilde{\boldsymbol{W}}\|_F = \|\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top - \tilde{\boldsymbol{W}}\|_F \tag{20}$$

$$= \|\boldsymbol{U}^\top(\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top - \tilde{\boldsymbol{W}})\boldsymbol{V}\|_F \tag{21}$$

$$= \|\boldsymbol{\Sigma} - \boldsymbol{U}^\top\tilde{\boldsymbol{W}}\boldsymbol{V}\|_F \tag{22}$$

since orthogonal transformations preserve Frobenius norm. Let $\tilde{\boldsymbol{\Sigma}} = \boldsymbol{U}^\top\tilde{\boldsymbol{W}}\boldsymbol{V}$. The objective becomes:

$$\text{argmin}_{\|\tilde{\boldsymbol{\Sigma}}\|_2 \le \tau} \|\boldsymbol{\Sigma} - \tilde{\boldsymbol{\Sigma}}\|_F \tag{23}$$

Since $\|\tilde{\boldsymbol{\Sigma}}\|_2 \le \tau$ and the minimum of $\|\boldsymbol{\Sigma} - \tilde{\boldsymbol{\Sigma}}\|_F$ occurs when off-diagonal elements of $\tilde{\boldsymbol{\Sigma}}$ are zero and diagonal elements are $\tilde{\sigma}_i = \min(\sigma_i, \tau)$, we obtain:

$$\tilde{\boldsymbol{\Sigma}}^* = \min(\boldsymbol{\Sigma}, \tau) \tag{24}$$

Thus, $\tilde{\boldsymbol{W}}^* = \boldsymbol{U}\min(\boldsymbol{\Sigma}, \tau)\boldsymbol{V}^\top$, proving (17).

8 Main Component Clipping

How to efficiently compute mclip(; ?) Performing SVD every step is expensive. As discussed in *Computing mclip via msign (Part I)* and *Computing mclip via msign (Part II)*, one can use msign() but it requires multiple passes. An alternative uses iterative top-singular-value estimation (SVD1). Since mclip(; c)lips all singular values above τ to τ, repeatedly clipping the top singular value achieves the same effect. Assuming training smoothness, one clipping per step suffices. Thus:

$$\text{Post Clip:} \quad \boldsymbol{W}_t = \tilde{\boldsymbol{W}}_t - \max(\sigma_1 - \tau, 0)\boldsymbol{u}_1\boldsymbol{v}_1^\top, \quad \sigma_1, \boldsymbol{u}_1, \boldsymbol{v}_1 = \text{SVD1}(\tilde{\boldsymbol{W}}_t), \quad \tilde{\boldsymbol{W}}_t = \boldsymbol{W}_{t-1} - \eta\boldsymbol{\Phi}_t \tag{25}$$

$$\text{Pre Decay:} \quad \boldsymbol{W}_t = \boldsymbol{W}_{t-1} - \lambda\eta\sigma_1\boldsymbol{u}_1\boldsymbol{v}_1^\top - \eta\boldsymbol{\Phi}_t, \quad \sigma_1, \boldsymbol{u}_1, \boldsymbol{v}_1 = \text{SVD1}(\boldsymbol{W}_{t-1}) \tag{26}$$

The Pre Decay version matches the spectral weight decay introduced in *From Spectral Gradient to New Weight Decay*.

9 Other Details

For greater accuracy, approximate k largest singular values and clip up to k per step. This requires QR during power iteration (see *Streamed Power Iteration for Muon: Part I*).

Other norms appear in *Beyond MuP: 3*, e.g., row/column RMS for Embedding/LM Head and ℓ_∞ norm for gamma in RMS Norm. Their clipping operators are straightforward: - Embedding: clip each row's RMS to $\leq \tau$ - LM Head: same for columns - gamma: clip element-wise to $[-\tau, \tau]$, i.e., $\max(\min((,), \gamma); \tau, -\tau)$

These are intuitive and left as exercises.

10 Necessary Justification

Some may ask: why not use ordinary weight decay as in *Training Deep Learning Models with Norm-Constrained LMOs*:

$$\boldsymbol{W}_t = (1 - \eta\lambda)\boldsymbol{W}_{t-1} - \eta\sqrt{\frac{d_{out}}{d_{in}}}\,\mathrm{msign}(()\boldsymbol{G}_t) \tag{27}$$

to limit spectral norm?

The issue is over-intervention. The clipping operator (6) minimizes perturbation. Multiplying by $1-\eta\lambda$ reduces spectral norm but is not minimal; hence, it over-intervenes. Consequences: - Small λ: insufficient control (τ too large) - Large λ: too much damping

For example, to bound spectral norm by 5 with $d_{in} = d_{out}$, $\lambda = 0.2$ gives excessive 0.2 weight decay (typical is 0.01).

We stress "guarantee": small models may stay below 5 with decay 0.01, but large models can reach theoretical limits (here 100). Maintaining a tight theoretical bound is essential for safetythis is the "stable" in "stable acceleration". The clipping operator (6) provides the lightest intervention that guarantees bounded norms, thus minimizing performance loss.

11 Summary

This article, based on minimal-perturbation principles, proposes a general framework for maintaining parameter stability during training, including Post Clip and Pre Decay. Under spectral norm, they induce Singular Value Clipping and Spectral Weight Decay. These operations aim to bound key parameter norms while minimizing interference with training dynamics.