https://pubsonline.informs.org/journal/moor

MATHEMATICS OF OPERATIONS RESEARCH

Vol. 00, No. 0, Xxxxx 0000, pp. 000–000

Submitted to *Mathematics of Operations Research*

ISSN 0364-765X, EISSN 1526-5471

On the Width Scaling of Neural Optimizers Under Matrix Operator Norms I: Making Operators Play Nice Together

Ruihan Xu
University of Chicago, ruihanx@uchicago.edu, https://ruihanxx.github.io/

Jiajin Li
Sauder School of Business, University of British Columbia, jiajin.li@sauder.ubc.ca, https://gerrili1996.github.io/

Yiping Lu
Industrial Engineering & Management Science, Northwestern University, yiping.lu@northwestern.edu, https://2prime.github.io/

Abstract. A central question in modern deep learning and language model is how to design optimizers whose performance scales favorably with network width. We address this question by viewing neural-network optimizers such as `AdamW` and `Muon` through a unified lens as instances of steepest descent under matrix operator norms. Within this framework, we align the optimizer's geometry with the Lipschitz structure of the network's forward map, require layerwise composability, and show that standard $p \to q$ norms steepest-descent rules fail to compose across layers. To overcome this, we introduce a family of matrix operator norm geometries $(p, \mathtt{mean}) \to (q, \mathtt{mean})$ that admit closed-form layerwise descent directions and yield practical optimizers such as a rescaled `AdamW`, row normalization, and column normalization. By construction, our rescaling recovers μP-style [51] width scaling as a special case and provides predictable cross-width learning-rate transfer across a broader class of optimizers.

We further prove that the induced descent direction preserves standard convergence guarantees and achieves near width-insensitive smoothness for mappings $(1, \mathtt{mean}) \to (q, \mathtt{mean})$ with $q \geq 2$ and $(p, \mathtt{mean}) \to \ell_\infty$, where smoothness is measured in the corresponding matrix-norm geometry. Building on this, we formalize a local optimization–approximation trade-off: enlarging the operator-norm unit ball improves approximation of the forward dynamics but increases the associated smoothness constant, whereas shrinking it reduces smoothness at the cost of larger bias. This trade-off identifies the $(p, \mathtt{mean}) \to \ell_\infty$ geometry with $p \geq 2$ as particularly favorable and motivates a simple yet effective optimizer in which gradients are normalized row-wise.

Finally, we show that this optimizer achieves improved width scaling compared with Muon, and that Muon in turn outperforms AdamW, suggesting a principled and practical route for mitigating dimensional dependence in large-scale optimization.

To sum up, these findings point to a principled, practical way to mitigate dimensional dependence in optimization.

Key words: Neural-network optimization; steepest descent; width scaling; learning-rate transfer; gradient normalization; row normalization; smoothness and Lipschitz analysis

MSC2000 subject classification: 90C30

OR/MS subject classification: Primary: ; secondary:

History:

1. Introduction

Recent neural scaling laws [2, 13, 20, 22] offer a clear narrative: Model performance improves predictably with scale, suggesting that larger architectures should yield better results. Yet putting these laws into practice exposes a gap: They say little about how the optimal learning rate varies with width. In practice, for mainstream optimizers such as `AdamW` [51–53] and `Muon` [16], the optimal learning rate is strongly width dependent: The optimal rate tuned for a network with 512 hidden units can diverge or slow markedly when width increases to 2048. This sensitivity indicates that standard optimizers do not naturally respect architectural scaling. To realize the gains promised by scaling laws, we therefore aim for optimizers whose tuning varies only weakly with width. Naturally, we ask:

How can we decouple the optimal learning rate from network width, and do algorithms that succeed at small scale remain effective as models grow?

We answer this question in the affirmative by viewing existing neural network optimizers, including `SignSGD` [7], `AdamW` [23, 30], `GradPower` [48], and `Muon` [9, 21], in a unified framework as instances of steepest descent under matrix operator norms [15, 32]. Specifically, we consider the optimization problem $\min_{\boldsymbol{W}} f(\boldsymbol{W})$, where $\boldsymbol{W}$ denotes the network parameters. Classical steepest descent in the Euclidean geometry (Frobenius geometry for matrices) picks the unit–norm direction that yields the largest instantaneous decrease of the first–order model:

$$\boldsymbol{D}_k = \operatorname*{arg\,min}_{\|\boldsymbol{D}\|_F=1} \langle \nabla f(\boldsymbol{W}_k),\, \boldsymbol{D}\rangle = -\frac{\nabla f(\boldsymbol{W}_k)}{\|\nabla f(\boldsymbol{W}_k)\|_F}.$$

Hence, the negative gradient is the steepest descent direction under the Euclidean/Frobenius norm. With a step size $\eta_k > 0$, the update rule follows $\boldsymbol{W}_{k+1} = \boldsymbol{W}_k - \eta_k \nabla f(\boldsymbol{W}_k)$. More generally, steepest descent can be defined with respect to an arbitrary norm $\|\cdot\|$ with dual $\|\cdot\|_*$. At iterate $\mathbf{W}_k$, a steepest–descent direction is any

$$\boldsymbol{D}_k \in \operatorname*{arg\,min}_{\|\boldsymbol{D}\|\le 1} \langle \nabla f(\boldsymbol{W}_k), \mathbf{D}\rangle \;=\; -\partial \|\nabla f(\boldsymbol{W}_k)\|_*\,,$$

where $\partial\|\cdot\|_*$ denotes the subdifferential of the dual norm. This formulation recovers classical Euclidean steepest descent when $\|\cdot\| = \|\cdot\|_F$, while alternative norms induce different update rules that exploit problem-specific geometry. For example, when $\|\cdot\| = \|\cdot\|_p$ with $p \ge 1$, the induced ℓ_{p^*} normalization, where $\frac{1}{p} + \frac{1}{p^*} = 1$, yields the `GradPower` family of methods [48]. When $\|\cdot\| = \|\cdot\|_\infty$, the steepest descent direction simplifies to the coordinate-wise sign of the gradient, yielding the well-known `SignSGD` algorithm [7]. `AdamW` [23, 30] can in fact be viewed as a smoothed or adaptive variant of `SignSGD`, see Remark 1. Extending the choice of norm from vectors to matrices naturally introduces matrix norm–based scaling, which corresponds to "whitened" or geometry-aware descent directions implemented through row/column normalization or related preconditioning schemes [5, 9, 21, 28, 35, 44]. In this way, the choice of norm acts as a versatile design knob, offering a unified geometric framework that encompasses and connects a wide range of first-order optimization algorithms.

A natural next question is how to select a matrix norm that faithfully characterizes the objective landscape. To this end, we follow the modular framework of [4, 25], which prescribes selecting the matrix norm that renders the forward map well-conditioned with respect to weight perturbations, thereby inducing an M-Lipschitz constant under the chosen geometry. Building on this intuition, [4] identifies the operator norm as a particularly natural choice; we adopt it as the starting point for our analysis.

Definition 1 (Operator Norm). Let $\boldsymbol{D} \in \mathbb{R}^{m\times n}$, $\|\cdot\|_{\text{in}}$ and $\|\cdot\|_{\text{out}}$ be norms on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. The operator norm of $\boldsymbol{D}$ induced by these norms is defined as

$$\|\boldsymbol{D}\|_{\text{in}\to\text{out}} := \sup_{\boldsymbol{x}\in\mathbb{R}^n, \boldsymbol{x}\neq 0} \frac{\|\boldsymbol{D}\boldsymbol{x}\|_{\text{out}}}{\|\boldsymbol{x}\|_{\text{in}}} = \sup_{\|\boldsymbol{x}\|_{\text{in}}=1} \|\boldsymbol{D}\boldsymbol{x}\|_{\text{out}}.$$

While operator norms provide a unifying geometric framework, not every such norm leads to a practical optimizer. From a computational standpoint, the chosen geometry must admit an efficient steepest-descent oracle—that is, the direction $\boldsymbol{D}_k$ should be obtainable either in closed form or via a fast iterative approximation. Otherwise, the elegance of the operator-norm formulation offers little algorithmic value. To this end, [5] propose simple oracles for several families of operator norms: Newton–Schulz iteration for $2 \to 2$ the norm, column normalization for $1 \to q$, and row normalization for $p \to \infty$. In what follows, we restrict attention to these computationally tractable cases (i.e., $p \to q$ where $p \leq q$), and examine whether their induced geometries truly preserve the M-Lipschitz property of the network mapping.

With Definition 1 in place, a natural follow-up question arises: How do operator norms interact when stacked across multiple layers of a neural network? We seek conditions under which the global Lipschitz constant satisfies a dimension-free multiplicative bound. The directional gradient of the $(i+2)$-th layer output with respect to $\boldsymbol{W}_i$ is obtained by a standard chain–rule calculation, and its norm decomposes into three factors: the operator norm of the perturbation $\boldsymbol{\Delta W}_i$, the operator norm of the next–layer weight matrix $\boldsymbol{W}_{i+1}$, and the cross–layer mismatch $\|\boldsymbol{I}_d\|_{i,\text{out}\to i+1,\text{in}}$. When this mismatch coefficient is strictly larger than 1, even tiny perturbations of $\boldsymbol{W}_\ell$ are amplified purely due to the change of geometry between consecutive layers. This phenomenon is precisely what motivates our requirement that adjacent norms "play nicely together". More precisely, an operator norm $\|\cdot\|_{\text{in}\to\text{out}}$ is said to *play nicely together* if $\|\cdot\|_{\text{in}} \leq \|\cdot\|_{\text{out}}$, which ensures that the identity map between consecutive spaces is non-expansive. Under this condition, the global Lipschitz behavior is width-independent, and no dimension-dependent correction terms arise. Nevertheless, the standard selection of operator norms [5] $1 \to q$ and $p \to \infty$ violate this property. Indeed, for $p < q$, we have $\|\boldsymbol{x}\|_p \leq n^{1/p-1/q}\|\boldsymbol{x}\|_q$, or equivalently $\|\boldsymbol{I}_n\|_{\ell_q\to\ell_p} = n^{1/p-1/q} > 1$. Such a mismatch leads to multiplicative amplification of order $n^{1/p}$ in the $p \to \infty$ geometry, and of order $n^{1-1/p}$ in the $1 \to q$ geometry, when layers are stacked.

These observations motivate introducing a new, width-aware geometry via the mean-normalized norms $\|\cdot\|_{(p,\texttt{mean})}$:

$$\|\boldsymbol{x}\|_{(p,\texttt{mean})} := \left(\frac{1}{n}\sum_{i=1}^{n}|\boldsymbol{x}_i|^p\right)^{1/p} = n^{-1/p}\|\boldsymbol{x}\|_p.$$

The factor $n^{-1/p}$ precisely cancels the dimensional scaling of the ℓ_p embeddings, ensuring that ensuring that the operator norms *play nicely together*: $\|\mathbf{I}_n\|_{(q,\texttt{mean})\to(p,\texttt{mean})} \leq 1$, for all $1 \leq p \leq q \leq \infty$. In effect, this normalization provides a width-aware rescaling that keeps feature magnitudes stable under layer composition, yielding a dimension-free Lipschitz constant. We formalize these results in Section 2.2. The transition from $p \to q$ geometry to $(p,\texttt{mean}) \to (q,\texttt{mean})$ geometry induces a layer-wise scaling of the optimization update, as detailed in Section 2.3.

Having established that the $(p,\texttt{mean})$ geometry yields width-free M-Lipschitz bounds for the network map, we now examine the *sensitivity of gradients* under the same geometry—namely, its L-smoothness. Whereas the M-Lipschitz property controls the magnitude of function variations, L-smoothness characterizes local curvature and thus the stability of gradient-based updates. Rather than requiring strict width-independence, we quantify how the smoothness constant scales with the layer width w and seek geometries that make this scaling as slow as possible—ideally constant. Theorem 5 shows that, in the $(p,\texttt{mean})\to(q,\texttt{mean})$ geometry, the smoothness constant is *width-insensitive* precisely when $q \geq 2p$. Otherwise, any residual width dependence is governed by the term $\frac{1}{q} - \frac{1}{2p}$, which determines the rate at which the smoothness constant increases with the layer width w. In particular, $(1,\texttt{mean})\to(q,\texttt{mean})$ with $q \geq 2$ and $(p,\texttt{mean})\to\infty$ both yield smoothness bounds that are independent of the network width. By contrast, $(2,\texttt{mean})\to(2,\texttt{mean})$ (Muon) exhibits L-smoothness scaling as $\sqrt{w}$.

To summarize, the two geometries $(1,\texttt{mean})\to(q,\texttt{mean})$ with $q \geq 2$ and $(p,\texttt{mean})\to\infty$ yield width-independent behavior for both the M-Lipschitz and L-smoothness properties. However, our L-smoothness bounds are established within the unit ball of the chosen geometry. Since different norms are ordered in strength, the optimization constraint changes with the geometry, and so does the approximation power of the network class. Specifically, if $\|\cdot\|_{\boldsymbol{A}}$ is stronger than $\|\cdot\|_{\boldsymbol{B}}$, then $\{\boldsymbol{W} : \|\boldsymbol{W}\|_{\boldsymbol{A}} \leq 1\} \subset \{\boldsymbol{W} : \|\boldsymbol{W}\|_{\boldsymbol{B}} \leq 1\}$, i.e., a stronger norm induces a smaller feasible set (better smoothness control but less expressivity), while a weaker norm enlarges the feasible set (greater expressivity but potentially worse smoothness). In the final step, we analyze this approximation–smoothness trade-off to identify geometries that balance expressivity with training stability. For the $(1, \texttt{mean}) \to (q, \texttt{mean})$ family ($q \geq 2$), the mean normalization already absorbs width in the smoothness bound; the residual width dependence appears in the approximation bound through the unit-ball size: $\|\boldsymbol{W}\|_{(1,\texttt{mean})\to(p,\texttt{mean})} = w^{1-1/p} \|\boldsymbol{W}\|_{1\to q}$. Hence we prefer exponents that keep the growth in w as slow as possible. For the $(p, \texttt{mean}) \to \infty$ family, the corresponding scaling factor is $w^{1/q}$. Comparing $w^{1-1/p}$ (column normalization) with $w^{1/q}$ (row normalization) for $p, q \geq 2$, the latter grows no faster and is strictly smaller unless $p = q = 2$. Consequently, row normalization attains a given smoothness target with smaller norms (i.e., a larger feasible set for the same smoothness budget). Guided by this insight, we advocate gradient–descent algorithms with row-normalized updates under the $(p, \texttt{mean}) \to \infty$ geometry with $p \geq 2$. Concretely, each step applies the element-wise sign-preserving power transform $\varphi_p(\cdot) = \mathrm{sign}(\cdot) \odot |\cdot|^p$, followed by row-wise p-norm normalization. As a by-product, since `AdamW`/`SignSGD` corresponds to steepest descent under $(1, \texttt{mean}) \to \ell_\infty$, the unit-ball scaling is linear in width (i.e., $O(w)$). Consequently, for a fixed smoothness budget, this geometry yields the poorest approximation guarantee among those we consider.

Our Contributions Our central premise is that the optimization geometry should be aligned with the Lipschitz geometry of the network's forward map. This perspective naturally leads to mean-normalized operator norms, which compose across layers and yield width-invariant optimization dynamics. Under this framework, μP-scaled `Adam` emerges as a special case, while our analysis reveals a broader family of compatible optimizers. Our specific contributions are listed as follows:

(i) We show that although the $1 \to q$ (column normalization) and $p \to \infty$ (row normalization) geometries admit closed-form steepest descent directions, they inherently break the cross-layer composability of stability estimates and fail to yield dimension-independent M-Lipschitz bounds. This motivate us to use $(p, \texttt{mean}) \to (q, \texttt{mean})$ operator norms instead to propagate stability bound across layers and resulting a width-aware scaling to each layer's updates. Under `Adam`, this scaling matches the μP-scaling rule [52], which ensures that each layer is updated on the same order of magnitude during training—regardless of network width—a property known as learning-rate (hyperparameter) transfer.

(ii) Beyond [4, 25], who only study the M-Lipschitz property of the forward map, we further analyze the gradient sensitivity, known as L-smoothness, which governs the optimal learning rate and optimization difficulty. Performing steepest descent under $(1, \mathtt{mean}) \to (q, \mathtt{mean})$ ($p \geq 2$, corresponding to column normalization under the p^*-norm) or $(p, \mathtt{mean}) \to \infty$ (row normalization under the p-norm) yields width independent L-smoothness constants estimate. As a result, we achieve optimal learning rates that are independent of network width.

(iii) Previous analyses assume network weights lie in the unit ball, revealing a trade-off between optimization ease and approximation power. However, this unit-norm constraint limits the representable function space. Under width-independent L-smoothness, we find that row normalization surprisingly offers greater approximation capacity than column normalization.

Notation. Let $\boldsymbol{x} \in \mathbb{R}^n$ be a column vector, and denote its i-th entry by x_i, $i = 1, \ldots, n$. Let $\mathbf{G} \in \mathbb{R}^{C \times R}$ be a matrix. We write $\mathbf{G}_{c,:}$ for the c-th row of $\mathbf{G}$ and $\mathbf{G}_{:,r}$ for its r-th column, for $c = 1, \ldots, C$ and $r = 1, \ldots, R$. For two vectors $x, y \in \mathbb{R}^n$, their elementwise product is defined as $(x \odot y)_i := x_i y_i, i = 1, \ldots, n$. Let X and Y be Banach spaces, and let $f : X \to Y$ be a mapping. For $x \in X$ and a direction $h \in X$, the *directional gradient* of f at x in the direction h is defined as $\nabla_x f(x)[h] := \lim_{t \to 0} \frac{f(x+th)-f(x)}{t}$, whenever this limit exists. We say that f is (Gâteaux) differentiable at x if the above limit exists for all directions $h \in X$. Similarly, the directional Hessian is defined as $\nabla^2 f(x)[h,k] := \lim_{t \to 0} \frac{1}{t}\Big(\nabla_x f(x+tk)[h] - \nabla_x f(x)[h]\Big)$.

2. Matrix Thinking: Unifying Optimizers via Matrix Operator Norm

We consider the optimization problem of minimizing a neural-network-based loss

$$\min_{\boldsymbol{W}_{1:\ell}, \boldsymbol{b}_{1:\ell}} f(\boldsymbol{W}_{1:\ell}, \boldsymbol{b}_{1:\ell}) := \mathcal{L}\big(\boldsymbol{y}_\ell(\boldsymbol{x})\big),$$

where $\boldsymbol{y}_\ell(\boldsymbol{x}) \in \mathbb{R}$ denotes the output of a ℓ-layer feedforward neural network with parameters $(\boldsymbol{W}_{1:\ell}, \boldsymbol{b}_{1:\ell})$, evaluated at input $\boldsymbol{x} \in \mathbb{R}^d$. Later on, we write $\Theta := \{\boldsymbol{W}_{1:\ell}, \boldsymbol{b}_{1:\ell}\}$ for simplicity.

Definition 2 (Feedforward Neural Network). Let $\boldsymbol{y}_0(\boldsymbol{x}) := \boldsymbol{x}$. For $i = 1, \ldots, K$, define recursively

$$\boldsymbol{y}_i(\boldsymbol{x}) := \sigma\left(\boldsymbol{W}_i \boldsymbol{y}_{i-1}(\boldsymbol{x}) + \boldsymbol{b}_i\right),$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{w \times d}$, $\boldsymbol{W}_i \in \mathbb{R}^{w \times w}$ for $i = 2, \ldots, K-1$, and $\boldsymbol{W}_K \in \mathbb{R}^{1 \times w}$. The bias vectors satisfy $\boldsymbol{b}_i \in \mathbb{R}^w$ for $i = 1, \ldots, K-1$ and $b_\ell \in \mathbb{R}$. The hidden states satisfy $\boldsymbol{y}_i(\boldsymbol{x}) \in \mathbb{R}^w$ for $i < K$, and the network output is $\boldsymbol{y}_\ell(\boldsymbol{x}) \in \mathbb{R}$. Here, σ denotes a activation function, which represents both a mapping $\sigma : \mathbb{R} \to \mathbb{R}$ and its natural extension $\sigma : \mathbb{R}^n \to \mathbb{R}^n$, where it is applied pointwise to each component, i.e., $\sigma(\boldsymbol{x})_i = \sigma(\boldsymbol{x}_i)$ for each coordinate i.

Throughout this paper, we assume that the loss function $\mathcal{L}$ and all activation functions σ_i are twice continuously differentiable with uniformly bounded first- and second-order derivatives.

ASSUMPTION 1. *Bounded derivatives of activation functions There exist constants $L_\sigma, M_\sigma >$ 0 such that*

$$|\sigma'(z)| \le L_\sigma, \quad \textit{and} \quad |\sigma''(z)| \le M_\sigma, \qquad \forall z \in \mathbb{R}.$$

ASSUMPTION 2. *Bounded derivatives of the loss function There exist constants $L_J, M_J > 0$ such that*

$$|\mathcal{L}'(z)| \le L_J, \quad \textit{and} \quad |\mathcal{L}''(z)| \le M_J, \qquad \forall z \in \mathbb{R}.$$

With the network architecture and smoothness assumptions in place, we now turn to the central question of this section: how to design and compare first-order optimizers through the geometry they induce. We adopt a matrix operator-norm perspective, viewing many modern optimization methods as instances of steepest-descent updates under appropriately chosen operator norms. This viewpoint encompasses a variety of recent optimizers, including `SignSGD` [7], `Lion` [12], and `Muon` [21], which have demonstrated strong empirical performance in large-scale learning. From this perspective, the choice of geometry plays a dual role. On the one hand, it governs the computational cost of each update through the tractability of the associated steepest-descent direction; on the other hand, it determines the convergence behavior of the resulting optimization method. A well-chosen operator norm must therefore strike a balance between per-iteration computational efficiency and favorable convergence properties.

Computability and per-iteration cost. The steepest-descent direction induced by a general $\|\cdot\|_{p\to q}$ operator norm does not necessarily admit a closed-form expression. Beyond `Muon` [21], which employs a Newton–Schulz iteration to approximate steepest descent under the $2 \to 2$ operator norm, prior work [5] shows that steepest descent under the $1 \to q$ and $p \to \infty$ norms admits efficient column-wise and row-wise closed-form computations, respectively. In this paper, we therefore restrict attention to operator norms for which the induced steepest-descent directions are computationally tractable; see Section 2.1 for details.

Convergence speed. Among computable operator norms, a natural question is which (p,q) geometries render the optimization problem well posed and lead to favorable convergence guarantees. We address this question by analyzing the M-Lipschitz continuity (Section 2.2) and L-smoothness (Section 2.4) of neural networks under various $p \to q$ operator norms.

Our analysis reveals a fundamental limitation of classical $p \longrightarrow q$ operator norms: Both Lipschitz and smoothness constants deteriorate with increasing network width, indicating a structural mismatch between these geometries and the compositional nature of deep neural networks. This degradation arises because standard operator norms fail to propagate stability estimates across layers, thereby distorting the effective geometry of the network's forward map.

To overcome this issue, we introduce *mean-normalized operator norms*, which are specifically designed to preserve composability under width scaling. These norms yield width-independent first- and second-order estimates and align the optimization geometry with the network's forward-map geometry. Building on this geometric insight, we develop the `MOGA` (**M**atrix **O**perator **G**eometry **A**ware) optimizer and show that μP-Adam arises as a special case within our framework.

2.1. From `AdamW`, `Muon` to Row and Column Normalization

In this subsection, we show that many state-of-the-art optimizers can be interpreted as instances of steepest descent under suitable matrix $p \longrightarrow q$ operator norms, with several such instances admitting implementable per-iteration updates [5]. To this end, recall that the steepest-descent direction under a matrix $p \longrightarrow q$ operator norm is defined as the solution to

$$\boldsymbol{D}^\star := \operatorname*{arg\,min}_{\|\boldsymbol{D}\|_{p\to q}\leq 1} \langle \nabla f(\boldsymbol{W}), \boldsymbol{D}\rangle. \tag{1}$$

We begin with three representative optimizers: `SignSGD` [7], `Lion` [12], and `AdamW` [23, 30]. Although these methods were originally developed for optimization over vector-valued parameters, they admit natural interpretations when applied to matrix-valued parameters. At the vector level, both `SignSGD` and `AdamW` can be viewed as steepest-descent methods under the ℓ_∞ geometry [3]. When such vector updates are applied entrywise to matrix parameters, the induced geometry corresponds implicitly to the matrix operator norm $\ell_1 \longrightarrow \ell_\infty$ [5]. This connection is formalized by the following fact.

FACT 1 (**[5], Proposition 3**). *Let $\boldsymbol{D} \in \mathbb{R}^{m\times n}$. Then the element-wise ℓ_∞ norm of $\boldsymbol{D}$ coincides with its $\ell_1 \longrightarrow \ell_\infty$ operator norm, that is,*

$$\|\boldsymbol{D}\|_{1\to\infty} := \sup_{\boldsymbol{x}\neq 0} \frac{\|\boldsymbol{D}\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_1} = \max_{i,j} |\boldsymbol{D}_{ij}| =: \|\boldsymbol{D}\|_{\max}.$$

Then, the steepest-descent subproblem (1) *admits the closed-form solution $\boldsymbol{D}^\star = -sign\big(\nabla f(\boldsymbol{W})\big)$, where $sign(\cdot)$ is applied entrywise. Consequently, the induced update direction recovers the `SignSGD` algorithm.*

REMARK 1. We briefly clarify the relationship between Adam, RMSprop, and sign-based methods at the level of update directions. Consider the sign descent update $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\,\mathrm{sign}(\tilde{\mathbf{g}}_t)$, where $\tilde{\mathbf{g}}_t$ denotes a (possibly stochastic) gradient estimate and $\alpha > 0$ is the step size. This update corresponds to steepest descent under the ℓ_∞ geometry, as discussed above. The optimizer RMSprop maintains an exponential moving average of squared gradients,

$$\boldsymbol{v}_{t+1} = \beta_2 \boldsymbol{v}_t + (1-\beta_2)\tilde{\mathbf{g}}_t^2, \qquad \mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\,\frac{\tilde{\mathbf{g}}_t}{\sqrt{\boldsymbol{v}_{t+1} + \epsilon}},$$

where $\beta_2 \in [0,1)$ and $\epsilon > 0$ ensure numerical stability. At the level of update directions, RMSprop (and equivalently Adam with $\beta_1 = 0$) can be viewed as a smoothed variant of sign descent, where the discontinuous sign operator is replaced by a normalized gradient with adaptive, coordinate-wise scaling. Formally, in the limiting case $\beta_2 \to 0$ and $\epsilon \to 0$, the RMSprop update reduces to

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\,\frac{\tilde{\mathbf{g}}_t}{\sqrt{\tilde{\mathbf{g}}_t^2}} = \mathbf{w}_t - \alpha\,\mathrm{sign}(\tilde{\mathbf{g}}_t),$$

recovering the SignSGD direction. This observation places RMSprop, Adam and AdamW within a broader family of sign-based and geometry-aware first-order methods.

The second example is Muon [21], which can be interpreted as performing steepest descent under the matrix $2 \to 2$ operator norm. Since the spectral norm of a matrix coincides with its $\ell_2 \to \ell_2$ operator norm, the steepest-descent subproblem (1) admits the closed-form solution $\boldsymbol{D}^\star = -\mathrm{matrixsign}\big(\nabla f(\boldsymbol{W})\big)$, where the matrix sign is defined via the singular value decomposition: if $\nabla f(\boldsymbol{W}) = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, then $\mathrm{matrixsign}(\nabla f(\boldsymbol{W})) = \boldsymbol{U}\boldsymbol{V}^\top$. This characterization recovers the update direction of Muon, which whitens or orthogonalizes the gradient matrix. Empirically, Muon has been shown to be an effective alternative to AdamW for large-scale language-model pretraining [28].

The final class of examples consists of the operator norms $\|\cdot\|_{1\to q}$ and $\|\cdot\|_{p\to\infty}$ with $p, q \geq 1$, whose associated steepest-descent subproblems admit closed-form solutions [5, 17]. These geometries induce simple column-wise and row-wise update rules on matrix parameters, see Proposition 1 for details.

PROPOSITION 1. *Consider the steepest-descent subproblem* (1) *with gradient* $\boldsymbol{G} = \nabla f(\boldsymbol{W}) \in \mathbb{R}^{m\times n}$. *For the operator norms* $\|\cdot\|_{1\to q}$ *and* $\|\cdot\|_{p\to\infty}$, *the corresponding steepest-descent directions admit the following closed-form expressions:*

(i) ***(Column-wise update,** $\|\cdot\|_{1\to q}$)*

$$\boldsymbol{D}^\star = \texttt{colnorm}_q(\boldsymbol{G}) \quad \textit{and} \quad \texttt{colnorm}_q(\boldsymbol{G})_{:,c} := \frac{\operatorname{sign}(\mathbf{G}_{:,c}) \odot |\mathbf{G}_{:,c}|^{q^*-1}}{\|\mathbf{G}_{:,c}\|_{q^*}^{q^*-1}},$$

where $q^* = q/(q-1)$ is the dual exponent.

(ii) ***(Row-wise update,** $\|\cdot\|_{p\to\infty}$)*

$$\boldsymbol{D}^\star = \texttt{rownorm}_p(\boldsymbol{G}) \quad \textit{and} \quad \texttt{rownorm}_p(\mathbf{G})_{r,:} := \frac{\operatorname{sign}(\mathbf{G}_{r,:}) \odot |\mathbf{G}_{r,:}|^{p-1}}{\|\mathbf{G}_{r,:}\|_{p}^{p-1}}.$$

Here $\odot$ denotes elementwise multiplication.

Proof of Proposition 1. By [5, Proposition 8], the operator norms admit the representations

$$\|\boldsymbol{D}\|_{1\to q} = \max_{c\in[n]} \|\boldsymbol{D}_{:,c}\|_q, \qquad \|\boldsymbol{D}\|_{p\to\infty} = \max_{r\in[m]} \|\boldsymbol{D}_{r,:}\|_{p^*},$$

where $p^* = p/(p-1)$ denotes the dual exponent. As a consequence, the steepest-descent subproblem (1) under the $\|\cdot\|_{1\to q}$ and $\|\cdot\|_{p\to\infty}$ operator norms decouples across columns and rows, respectively.

We focus on the case $\|\cdot\|_{1\to q}$ for illustration; the proof for $\|\cdot\|_{p\to\infty}$ follows by an analogous argument and is therefore omitted. For the $\|\cdot\|_{1\to q}$ case, the steepest-descent subproblem (1) can be written as

$$\min_{\boldsymbol{D}\in\mathbb{R}^{m\times n}} \langle \boldsymbol{G}, \boldsymbol{D}\rangle \quad \text{s.t.} \quad \|\boldsymbol{D}_{:,c}\|_q \le 1,\ \forall c \in [n].$$

Since the objective decomposes as $\langle \boldsymbol{G}, \boldsymbol{D}\rangle = \sum_{c=1}^n \langle \boldsymbol{G}_{:,c}, \boldsymbol{D}_{:,c}\rangle$, the problem decouples across columns. For each column $c \in [n]$, we obtain the subproblem $\min_{\|\boldsymbol{d}\|_q\le 1} \langle \boldsymbol{G}_{:,c}, \boldsymbol{d}\rangle$. By Hölder's inequality, an optimal solution is given by

$$\boldsymbol{d}^\star = -\frac{\operatorname{sign}(\mathbf{G}_{:,c}) \odot |\mathbf{G}_{:,c}|^{q^*-1}}{\|\mathbf{G}_{:,c}\|_{q^*}^{q^*-1}},$$

where $q^* = q/(q-1)$ is the dual exponent. Collecting the solutions for all columns yields $\boldsymbol{D}^\star = \texttt{colnorm}_q(\boldsymbol{G})$, which completes the proof.

Message: Although `SignSGD/AdamW` is typically formulated under the vector ℓ_∞ norm, it can also be interpreted through the lens of a matrix $\ell_1 \to \ell_\infty$ norm. In a similar vein, `Muon` [9, 21] performs steepest descent under the $\ell_2 \to \ell_2$ operator norm. More generally, our framework considers the full families of operator norms $\ell_1 \to \ell_q$ and $\ell_p \to \ell_\infty$ for arbitrary $p, q \geq 1$, which induce column-wise and row-wise normalization updates, respectively. Several existing methods studied in the literature can be recovered as special cases corresponding to particular choices of (p, q); for instance, [17, 41] focus on the $\ell_1 \to \ell_2$ case, while [5, 31] consider the $\ell_2 \to \ell_\infty$ geometry. This perspective highlights the underlying **matrix thinking** behind a broad class of first-order optimizers. Notably, the update rules induced by the $\ell_1 \to \ell_q$ and $\ell_p \to \ell_\infty$ families admit efficient, vectorized implementations.

SignSGD/AdamW	Row Normalization	Column Normalization	Muon
$\|\cdot\|_{1\to\infty}$	$\|\cdot\|_{1\to q}$	$\|\cdot\|_{p\to\infty}$	$\|\cdot\|_{2\to 2}$

In summary, all matrix $p \to q$ operator norms that admit implementable per-iteration updates fall within the regime $p \leq q$. Accordingly, we focus on the case $p \to q$ with $p \leq q$ in the remainder of the paper.

2.2. Why mean-normalized operator norms are needed

Section 2.1 shows that a broad class of first-order optimizers can be interpreted as steepest-descent methods under matrix $p \to q$ operator norms ($p \leq q$) with computable update rules. A natural next question is whether these classical operator-norm geometries are well aligned with the stability properties of deep neural networks. In particular, the chosen matrix norm should reflect the model's sensitivity to weight perturbations so that the loss remains width-independent Lipschitz continuous [4, 6, 25]. However, despite their computational tractability and clean geometric form, we show that classical $p \to q$ operator norms with $p \leq q$ generally fail to yield width-independent Lipschitz bounds for neural networks. As a result, the induced Lipschitz bounds deteriorate with network width under these norms. This structural limitation motivates a refined geometry. In the remainder of this subsection, we introduce mean-normalized operator norms and show that they do yield width-independent Lipschitz bounds, see Theorem 3 for details.

To quantify the Lipschitz behavior induced by a given operator norm, we start to analyze the network in a layer-wise manner and track how Lipschitz bounds compose across layers. We model the network as a composition of linear maps and nonlinear activation functions:

$$\boldsymbol{y}_{\ell+1} = \sigma(\boldsymbol{z}_{\ell+1}), \quad \text{and} \qquad \boldsymbol{z}_{\ell+1} = \boldsymbol{W}_{\ell+1}\boldsymbol{y}_\ell + \boldsymbol{b}_{\ell+1},$$

where $\boldsymbol{z}_{\ell+1}$ denotes the pre-activation.

Each linear layer $\boldsymbol{W}_\ell$ maps features from one normed space to the next pre-activation space. When $\boldsymbol{W}_\ell$ is endowed with the operator norm $\|\cdot\|_{\text{in}\to\text{out}}$, Lipschitz stability propagates layer by layer. In particular, if σ is 1-Lipschitz, then the sensitivity of $\boldsymbol{y}_{\ell+1}$ to perturbations in an earlier weight matrix $\boldsymbol{W}_i$ ($i < \ell + 1$) satisfies

$$\left\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{l+1}\right\|_{\text{out}} \leq \left\|\nabla_{\boldsymbol{W}_i}\boldsymbol{z}_{l+1}\right\|_{\text{out}} = \left\|\boldsymbol{W}_{l+1}\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{l}\right\|_{\text{out}} \leq \|\boldsymbol{W}_{l+1}\|_{\text{in}\to\text{out}}\left\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{l}\right\|_{\text{in}},$$

where $\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_\ell$ denotes the Jacobian of $\boldsymbol{y}_\ell$ with respect to $\boldsymbol{W}_i$. Similarly, by the definition of the induced operator norm of $W_{\ell+2}$, we have $\left\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{l+2}\right\|_{\text{out}} \leq \|\boldsymbol{W}_{l+2}\|_{\text{in}\to\text{out}}\left\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{l+1}\right\|_{\text{in}}$. To compose this bound with the previous layer-wise estimate, the output norm of layer $\ell + 1$ must be compatible with the input norm of layer $\ell + 2$. That is, we impose the following norm-compatibility condition between consecutive layers:

$$\|\cdot\|_{\text{in}} \leq \|\cdot\|_{\text{out}}. \tag{2}$$

Under the compatibility condition (2), the layer-wise bounds compose and yield

$$\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{\ell+2}\|_{\text{in}} \leq \|\boldsymbol{W}_{\ell+2}\|_{\text{in}\to\text{out}}\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_{\ell+1}\|_{\text{out}} \leq \left(\prod_{k=i+1}^{\ell+2}\|\boldsymbol{W}_k\|_{\text{in}\to\text{out}}\right)\|\nabla_{\boldsymbol{W}_i}\boldsymbol{y}_i\|_{\text{in}}.$$

From the layer-wise bounds derived above, a width-independent cross-layer stability estimate based on operator norms is valid only if the input and output norms across consecutive layers satisfy the compatibility condition (2). This ensures that the sensitivity bounds can be composed across layers without distortion and therefore imposes a structural constraint on the admissible operator-norm geometries. However, classical matrix-induced norm families—such as the $1 \longrightarrow q$ and $p \longrightarrow \infty$ cases that admit closed-form steepest-descent directions—do not meet this compatibility requirement in a dimension-independent manner. The difficulty is that the underlying feature norms are not uniformly comparable across layers. Indeed, ℓ_p norms are not uniformly equivalent: for $\boldsymbol{x} = (1, \ldots, 1) \in \mathbb{R}^n$ and any $p > 1$, $\|\boldsymbol{x}\|_\infty = 1 < \|\boldsymbol{x}\|_p = n^{1/p} < \|\boldsymbol{x}\|_1 = n$, so the norm ratios grow with dimension.

To remove this dimension dependence, we introduce a mean-normalized variant of the p-power norm and work with the $(p, \texttt{mean}) \longrightarrow (q, \texttt{mean})$ operator-norm geometry instead of the standard $p \longrightarrow q$ setting. Formally, for $\boldsymbol{x} \in \mathbb{R}^n$ and $1 \leq p < \infty$, we define the (p, mean) norm by

$$\|\boldsymbol{x}\|_{(p,\texttt{mean})} := \left(\frac{1}{n}\sum_{i=1}^{n}|x_i|^p\right)^{1/p}. \tag{3}$$

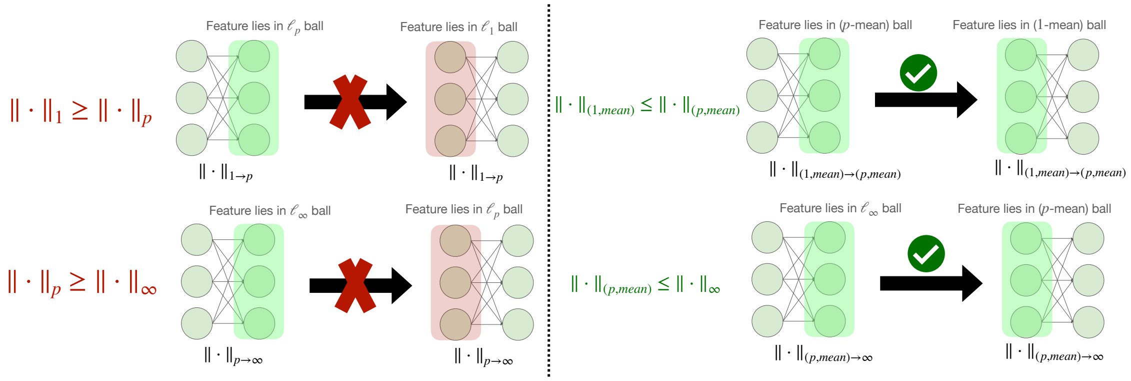
$\|\cdot\|_1 \geq \|\cdot\|_p$
Feature lies in ℓ_p ball
$\|\cdot\|_{1\to p}$
Feature lies in ℓ_1 ball
$\|\cdot\|_{1\to p}$
$\|\cdot\|_p \geq \|\cdot\|_\infty$
Feature lies in ℓ_∞ ball
$\|\cdot\|_{p\to\infty}$
Feature lies in ℓ_p ball
$\|\cdot\|_{p\to\infty}$
$\|\cdot\|_{(1,mean)} \leq \|\cdot\|_{(p,mean)}$
Feature lies in (p-mean) ball
$\|\cdot\|_{(1,mean)\to(p,mean)}$
Feature lies in (1-mean) ball
$\|\cdot\|_{(1,mean)\to(p,mean)}$
$\|\cdot\|_{(p,mean)} \leq \|\cdot\|_\infty$
Feature lies in ℓ_∞ ball
$\|\cdot\|_{(p,mean)\to\infty}$
Feature lies in (p-mean) ball
$\|\cdot\|_{(p,mean)\to\infty}$

FIGURE 1. **Operators Should Play Nice Together.** Chaining layer-wise stability bounds under $\|\cdot\|_{p\to q}$ requires $\|\cdot\|_p \leq \|\cdot\|_q$. This fails for classical $p \to q$ norms when $p \leq q$ but holds for $(p,\texttt{mean}) \to (q,\texttt{mean})$ norms, yielding dimension-independent bounds.

This family recovers several familiar quantities as special cases: $\|\cdot\|_{(1,\text{mean})}$ equals the mean absolute value (MAV), $\|\cdot\|_{(2,\text{mean})}$ equals the root mean square (RMS), and $\|\cdot\|_{(\infty,\text{mean})}$ reduces to the maximum norm, i.e., $\|\boldsymbol{x}\|_\infty = \max_i |x_i|$. This normalization rescales the classical ℓ_p norm by the width factor and removes dimension-dependent growth. The key property of the (p,mean) norms is that they remain uniformly comparable across widths.

FACT 2 (**Monotonicity of** (p,mean) **norms**). *Let* $\boldsymbol{x} \in \mathbb{R}^n$ *and* $1 \leq p < q \leq \infty$. *Then*

$$\|\boldsymbol{x}\|_{(p,\text{mean})} \leq \|\boldsymbol{x}\|_{(q,\text{mean})}.$$

Proof of Fact 2. The claim follows from the generalized mean inequality: For any $a_i \geq 0$ and $1 \leq p < q$,

$$\left(\frac{1}{n}\sum_{i=1}^{n} a_i^p\right)^{1/p} \leq \left(\frac{1}{n}\sum_{i=1}^{n} a_i^q\right)^{1/q}.$$

Apply this inequality with $a_i = |x_i|$ yields the result.

We now establish a width-independent Lipschitz bound for neural networks under the general $(p,\texttt{mean}) \to (q,\texttt{mean})$ operator-norm geometry.

THEOREM 3 (**Width-independent Lipschitz bound under mean-normalized geometry**). *Consider a* K*-layer neural network defined in Definition 2 with activation function* σ *and loss* $\mathcal{L}$ *satisfying Assumptions 1–2. Let* $1 \leq p \leq q < \infty$, *and suppose the input satisfies* $\|\boldsymbol{x}\|_2 \leq C$ *for some* $C > 1$. *Define the parameter set as* $\Omega_C := \{\Theta : \|\Theta\|_{\text{block}} \leq C\}$, *where the block norm is*

ATTENTION: The following displayed equation, in its current form, exceeds the column width that will be used in the published edition of your article. Please break or rewrite this equation to fit, including the equation number, within a column width of 470 pt / 165.81 mm / 6.53 in (the width of this red box).

$$\|\Theta\|_{\text{block}} := \max\left\{\|\boldsymbol{W}_1\|_{1\to(q,\text{mean})}, \max_{2\le i\le K-1}\|\boldsymbol{W}_i\|_{(p,\text{mean})\to(q,\text{mean})}, \|\boldsymbol{W}_K\|_{(p,\text{mean})\to\infty}, \max_i \|\boldsymbol{b}_i\|_\infty\right\}.$$

Then the loss function is Lipschitz continuous on Θ_C, i.e.,

$$|f(\Theta) - f(\Theta')| \le M_{\text{net}}\,\|\Theta - \Theta'\|_{\text{block}} \quad \forall\,\Theta,\Theta' \in \Omega_C,$$

where $M_{\text{net}} > 0$ depends only on C, K, and the constants in Assumptions 1–2, and is independent of all layer widths.

REMARK 2 (WHY THE BOUNDED PARAMETER SET IS NATURAL). The restriction $\Theta \in \Omega_C$ is natural in practice, as standard training procedures implicitly control parameter norms. Under the standard decoupled weight-decay update,

$$\Theta_{t+1} = (1-\eta\lambda)\Theta_t + \eta\boldsymbol{D}_t,$$

where $\eta > 0$, $0 < \lambda < 1$ and $\|\boldsymbol{D}_t\| \le 1$, we have $\|\Theta_{t+1}\| \le (1-\eta\lambda)\|\Theta_t\| + \eta$. Unrolling the recursion gives $\|\Theta_t\| \le (1-\eta\lambda)^t\|\Theta_0\| + 1/\lambda$, and hence $\sup_t \|\Theta_t\| \lesssim O(1/\lambda)$. Thus, weight decay together with bounded update directions keeps the parameters in a uniformly bounded ball. Moreover, several modern optimizers admit equivalent formulations as optimization under explicit norm constraints [10, 11, 35], further supporting the bounded-parameter assumption.

We now turn to the proof of Theorem 3. We begin with a standard lemma relating Lipschitz continuity to bounded directional derivatives, which will be used later in the argument.

LEMMA 1. *Let $\Omega \subset \mathbb{R}^{m\times n}$ be a convex norm ball, and let $f : \Omega \to \mathbb{R}$ be differentiable on Ω. Fix any norm $\|\cdot\|$ on $\mathbb{R}^{m\times n}$. Suppose there exists a constant $M > 0$ such that*

$$\sup_{\|\Delta\boldsymbol{Z}\|\le 1} \left|\nabla f(\boldsymbol{Z})[\Delta\boldsymbol{Z}]\right| \le M, \qquad \forall\,\boldsymbol{Z} \in \Omega.$$

Then f is M-Lipschitz continuous on Ω with respect to $\|\cdot\|$, i.e.,

$$|f(\boldsymbol{Z}) - f(\boldsymbol{Z}')| \le M\|\boldsymbol{Z} - \boldsymbol{Z}'\|, \qquad \forall\,\boldsymbol{Z},\boldsymbol{Z}' \in \Omega.$$

Proof of Lemma 1. Fix $\boldsymbol{Z}, \boldsymbol{Z}' \in \Omega$ and define the line segment $\boldsymbol{Z}(t) := (1-t)\boldsymbol{Z}' + t\boldsymbol{Z} = \boldsymbol{Z}' + t(\boldsymbol{Z} - \boldsymbol{Z}'), t \in [0,1]$. Since Ω is a norm ball, it is convex, and hence $\boldsymbol{Z}(t) \in \Omega$ for all $t \in [0,1]$. By the chain rule, we have $\frac{\mathrm{d}}{\mathrm{d}t} f(\boldsymbol{Z}(t)) = \nabla f(\boldsymbol{Z}(t))[\boldsymbol{Z} - \boldsymbol{Z}']$. Applying the Fundamental Theorem of Calculus yields

$$f(\boldsymbol{Z}) - f(\boldsymbol{Z}') = \int_0^1 \nabla f(\boldsymbol{Z}(t))[\boldsymbol{Z} - \boldsymbol{Z}']\, dt.$$

If $\boldsymbol{Z} = \boldsymbol{Z}'$, the claim is trivial. Otherwise let $\Delta \boldsymbol{Z} := \frac{\boldsymbol{Z} - \boldsymbol{Z}'}{\|\boldsymbol{Z} - \boldsymbol{Z}'\|}$, so that $\|\Delta \boldsymbol{Z}\| = 1$. Then, for all $t \in [0,1]$, the assumed bound implies

$$\left|\nabla f(\boldsymbol{Z}(t))[\boldsymbol{Z} - \boldsymbol{Z}']\right| = \|\boldsymbol{Z} - \boldsymbol{Z}'\| \left|\nabla f(\boldsymbol{Z}(t))[\Delta \boldsymbol{Z}]\right| \le M\|\boldsymbol{Z} - \boldsymbol{Z}'\|.$$

Therefore,

$$|f(\boldsymbol{Z}) - f(\boldsymbol{Z}')| \le \int_0^1 M\|\boldsymbol{Z} - \boldsymbol{Z}'\|\, dt = M\|\boldsymbol{Z} - \boldsymbol{Z}'\|.$$

This proves that f is M-Lipschitz continuous on Ω with respect to $\|\cdot\|$.

Proof of Theorem 3. we bound this dual norm using induction and backpropagation.

To bound the directional gradient of loss function $\mathcal{L}$, i.e., $|\nabla_{\mathbf{W}_{1:K}} \mathcal{L}[\mathbf{\Delta W}_{1:K}]|$ and $|\nabla_{\boldsymbol{b}_{1:K}} \mathcal{L}[\boldsymbol{\delta b}_{1:K}]|$, we first bound the directional gradient of y_K with respect to parameter matrix $\mathbf{W}_j$ and bias $\boldsymbol{b}_j$, i.e.,$\left|\nabla_{\mathbf{W}_j} y_K[\mathbf{\Delta W}]\right|$ and $\left|\nabla_{\boldsymbol{b}_j} \boldsymbol{y}_i[\boldsymbol{\delta b}]\right|$ for all $\|\Delta W\|_{(p,\mathtt{mean})\to(q,\mathtt{mean})} \le 1$ and $\|\boldsymbol{b}\|_\infty \le 1$.

To start with, we first establish a width-independent norm bound for the feature vectors $\boldsymbol{y}_i$. Since $\sigma(0) = 0$ and $|\sigma'(z)| \le L_\sigma$ for all $z \in \mathbb{R}$, the mean-value theorem implies $|\sigma(z)| \le L_\sigma |z|$. Applying it componentwise, we obtain $\|\sigma(\boldsymbol{x}\|_{q,\mathtt{mean}} \le L_\sigma \|\boldsymbol{x}\|_{q,\mathtt{mean}}, \forall \boldsymbol{x} \in \mathbb{R}^d$. Also Recall that by assumptions we have $\|\boldsymbol{x}\|_2 \le C$ and $\max\left\{\|\boldsymbol{b}_i\|_\infty,\ \|\mathbf{W}_1\|_{1\to(q,\mathtt{mean})},\ \max_{2\le i\le K-1} \|\mathbf{W}_i\|_{(p,\mathtt{mean})\to(q,\mathtt{mean})},\ \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty}\right\} \le C$. We then show $\|\boldsymbol{y}_i\|_{(p,\mathtt{mean})} \le (2L_\sigma)^i C^{i+1}$ for $1 \le i \le K-1$ by induction. First note that $\|\boldsymbol{y}_1\|_{(p,\mathtt{mean})} = \|\sigma(\mathbf{W}_1\boldsymbol{x} + \boldsymbol{b}_1)\|_{(p,\mathtt{mean})} \le L_\sigma(\|\mathbf{W}_1\|_{1\to(p,\mathtt{mean})}\|\boldsymbol{x}\|_1 + \|\boldsymbol{b}\|_\infty) \le 2L_\sigma C^2$, and for $2 \le i \le K-1$ we have

$$\|\boldsymbol{y}_i\|_{(q,\mathtt{mean})} = \|\sigma(\mathbf{W}_i\boldsymbol{y}_{i-1} + \boldsymbol{b}_i)\|_{(p,\mathtt{mean})} \le L_\sigma(\|\mathbf{W}_i\|_{(p,\mathtt{mean})\to(q,\mathtt{mean})}\|\boldsymbol{y}_{i-1}\|_{(p,\mathtt{mean})} + \|\boldsymbol{b}_i\|_\infty)$$

holds for $2 \le i \le K-1$. This leads to the bound $\|\boldsymbol{y}_i\|_{(p,\mathtt{mean})} \le (2L_\sigma)^i C^{i+1}$.Importantly, these upper bounds are independent of the network width w.

With the bound on $\|\boldsymbol{y}_i(\mathbf{W}_{1:i};\boldsymbol{x})\|_{(p,\texttt{mean})}$, we now turn to bounding the directional gradient $\nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]$ for $\|\boldsymbol{\Delta}\mathbf{W}\|_{(p,\texttt{mean})\to(q,\texttt{mean})} \le 1$ ($\|\boldsymbol{\Delta}\mathbf{W}\|_{1\to(q,\texttt{mean})} \le 1$ when $p=1$ and $\|\boldsymbol{\Delta}\mathbf{W}\|_{(p,\texttt{mean})\to\infty} \le 1$ when $p=K$) and $\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i[\boldsymbol{\delta b}]$, for $\|\boldsymbol{\delta b}\|_\infty \le 1$. By standard chain rule, we obtain

$$\nabla_{\mathbf{W}_j}\boldsymbol{y}_i = \begin{cases} \mathbf{D}_i \otimes (\boldsymbol{y}_{i-1})^\top, & j=i, \\ \\ \mathbf{D}_i\mathbf{W}_i\,\nabla_{\mathbf{W}_j}\boldsymbol{y}_{i-1}, & j<i, \end{cases} \quad \text{and} \quad \nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i = \begin{cases} \mathbf{D}_i, & j=i, \\ \\ \mathbf{D}_i\mathbf{W}_i\,\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}, & j<i, \end{cases},$$

where $\mathbf{D}_i = \texttt{diag}(\sigma'(\boldsymbol{z}_i))$ with $\boldsymbol{z}_i$ the pre-activation. We then consider three separate cases for p and i. For simplicity, we let

$$\boldsymbol{\Delta}^{\mathbf{W}}_{i,j} := \nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}], \quad \boldsymbol{\delta}^{\boldsymbol{b}}_{i,j} := \nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i[\boldsymbol{\delta b}]$$

- **Case 1** ($j=i$)**.** For $2 \le i \le K-1$, since $\nabla_{\mathbf{W}_i}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}] = \mathbf{D}_i(\boldsymbol{\Delta}\mathbf{W}\boldsymbol{y}_{i-1})$ and $\nabla_{\boldsymbol{b}_i}\boldsymbol{y}_i[\boldsymbol{\delta b}] = \mathbf{D}_i\boldsymbol{\delta b}$, we have

$$\begin{aligned} \left\|\nabla_{\mathbf{W}_i}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]\right\|_{(p,\texttt{mean})} &\le \left\|\nabla_{\mathbf{W}_i}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]\right\|_{(q,\texttt{mean})} \le \|\mathbf{D}_i\|_\infty\,\|\boldsymbol{\Delta}\mathbf{W}\boldsymbol{y}_{i-1}\|_{(q,\texttt{mean})} \\ &\le L_\sigma\|\boldsymbol{\Delta}\mathbf{W}\|_{(p,\texttt{mean})\to(q,\texttt{mean})}\,\|\boldsymbol{y}_{i-1}\|_{(p,\texttt{mean})} \le 2^{i-1}L_\sigma^i C^{i+1}, \\ \|\nabla_{\boldsymbol{b}_i}\boldsymbol{y}_i[\boldsymbol{\delta b}]\|_{(p,\texttt{mean})} &\le \|\nabla_{\boldsymbol{b}_i}\boldsymbol{y}_i[\boldsymbol{\delta b}]\|_\infty \le \|\mathbf{D}_i\|_\infty\|\boldsymbol{\delta b}\|_\infty \le L_\sigma. \end{aligned}$$

  Similarly for $i=1,K$, we have $\left\|\nabla_{\mathbf{W}_1}\boldsymbol{y}_1[\boldsymbol{\Delta}\mathbf{W}]\right\|_{(p,\texttt{mean})} \le \|\mathbf{D}_1\|_\infty\|\boldsymbol{\Delta}\mathbf{W}\|_{1\to(q,\texttt{mean})}\|\boldsymbol{x}\|_1 \le L_\sigma C^2$, $|\nabla_{\mathbf{W}_K}y_K[\boldsymbol{\Delta}\mathbf{W}]| \le L_\sigma\|\boldsymbol{\Delta}\mathbf{W}\|_{(p,\texttt{mean})\to\infty}\|\boldsymbol{y}_{K-1}\|_\infty \le 2^{K-1}L_\sigma^K C^{K+1}$ and $\left\|\nabla_{\boldsymbol{b}_{1,K}}\boldsymbol{y}_{1,K}[\boldsymbol{\delta b}]\right\|_{(p,\texttt{mean})} \le \|\mathbf{D}_{1,K}\|_\infty\|\boldsymbol{\delta b}\|_\infty \le L_\sigma$.

- **Case 2** ($j<i, i \ne K$)**.** Since $\nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}] = \mathbf{D}_i(\mathbf{W}_i\nabla_{\mathbf{W}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{W}])$ and $\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i[\boldsymbol{\delta b}] = \mathbf{D}_i\mathbf{W}_i\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\delta b}]$, we have

$$\begin{aligned} \left\|\nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]\right\|_{(p,\texttt{mean})} &\le \left\|\boldsymbol{\Delta}^{\mathbf{W}}_{i,j}\right\|_{(q,\texttt{mean})} \le L_\sigma\|\mathbf{W}_i\|_{(p,\texttt{mean})\to(q,\texttt{mean})}\left\|\boldsymbol{\Delta}^{\mathbf{W}}_{i-1,j}\right\|_{(p,\texttt{mean})} \\ &\le L_\sigma C\left\|\boldsymbol{\Delta}^{\mathbf{W}}_{i-1,j}\right\|_{(p,\texttt{mean})} \le (L_\sigma C)^{i-j}\left\|\boldsymbol{\Delta}^{\mathbf{W}}_{j,j}\right\|_{(p,\texttt{mean})} \le 2^{j-1}L_\sigma^i C^{i+1}, \\ \left\|\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i[\boldsymbol{\delta b}]\right\|_{(p,\texttt{mean})} &\le \left\|\mathbf{D}_i\mathbf{W}_i\boldsymbol{\delta}^{\boldsymbol{b}}_{i-1,j}\right\|_{(q,\texttt{mean})} \le L_\sigma\|\mathbf{W}_i\|_{(p,\texttt{mean})\to(q,\texttt{mean})}\left\|\boldsymbol{\delta}^{\boldsymbol{b}}_{i-1,j}\right\|_{(p,\texttt{mean})} \\ &\le (L_\sigma C)^{i-j}\left\|\boldsymbol{\delta}^{\boldsymbol{b}}_{j,j}\right\|_{(p,\texttt{mean})} \le L_\sigma^{i-j+1}C^{i-j}. \end{aligned}$$

- **Case 3** $(j < i = l)$**.** In this case, y_K is a scalar and we have

$$\begin{aligned}\left|\nabla_{\mathbf{W}_j} y_K[\boldsymbol{\Delta}\mathbf{W}]\right| &= \left|\mathbf{D}_i \mathbf{W}_i \boldsymbol{\Delta}^{\mathbf{W}}_{K-1,j}\right| \le L_\sigma \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\Delta}^{\mathbf{W}}_{K-1,j}\right\|_{(p,\mathtt{mean})} \\ &\le L_\sigma C \left\|\boldsymbol{\Delta}^{\mathbf{W}}_{K-1,j}\right\|_{(p,\mathtt{mean})} \le (L_\sigma C)^{K-j} \left\|\boldsymbol{\Delta}^{\mathbf{W}}_{j,j}\right\|_{(p,\mathtt{mean})} \le 2^{j-1} L_\sigma^K C^{K+1}, \\ \left\|\nabla_{\boldsymbol{b}_j} \boldsymbol{y}_K[\boldsymbol{\delta b}]\right\|_{(p,\mathtt{mean})} &\le \left\|\mathbf{D}_K \mathbf{W}_K \boldsymbol{\delta}^{\boldsymbol{b}}_{K-1,j}\right\|_{(q,\mathtt{mean})} \le L_\sigma \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\delta}^{\boldsymbol{b}}_{K-1,j}\right\|_{(p,\mathtt{mean})} \\ &\le (L_\sigma C)^{K-j} \left\|\boldsymbol{\delta}^{\boldsymbol{b}}_{j,j}\right\|_{(p,\mathtt{mean})} \le L_\sigma^{K-j+1} C^{K-j}.\end{aligned}$$

This indicates that all $\nabla_{\mathbf{W}_j} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]$ can be bounded by $(2L_\sigma)^i C^{i+1}$, and all $\nabla_{\boldsymbol{b}_j} \boldsymbol{y}_i[\boldsymbol{\delta b}]$ can be bounded by $L_\sigma^i C^{i-1}$, which are constants independent of the width w.

Note that $\nabla_{\mathbf{W}_{1:K}} \mathcal{L}[\boldsymbol{\Delta}\mathbf{W}_{1:K}] = \mathcal{L}'(y_K) \nabla_{\mathbf{W}_{1:K}} y_K[\boldsymbol{\Delta}\mathbf{W}_{1:K}] = \mathcal{L}'(y_K) \sum_{j=1}^K \nabla_{\mathbf{W}_j} y_K[\boldsymbol{\Delta}\mathbf{W}_j]$ and $\nabla_{\boldsymbol{b}_{1:K}} \mathcal{L}[\boldsymbol{\delta b}_{1:K}] = \mathcal{L}'(y_K) \nabla_{\boldsymbol{b}_{1:K}} y_K[\boldsymbol{\delta b}_{1:K}] = \mathcal{L}'(y_K) \sum_{j=1}^K \nabla_{\boldsymbol{b}_j} \boldsymbol{y}_K[\boldsymbol{\delta b}_j]$. By assumption $\mathcal{L}'(y_K) \le L_J$, we have

$$\begin{aligned}|\nabla_{\mathbf{W}_{1:K}} \mathcal{L}[\boldsymbol{\Delta}\mathbf{W}_{1:K}]| &\le L_J \sum_{j=1}^K \left|\nabla_{\mathbf{W}_j} y_K[\boldsymbol{\Delta}\mathbf{W}_j]\right| \lesssim_K L_J (2L_\sigma)^K C^{K+1}, \\ |\nabla_{\boldsymbol{b}_{1:K}} \mathcal{L}[\boldsymbol{\delta b}_{1:K}]| &\le L_J \sum_{j=1}^K \left|\nabla_{\boldsymbol{b}_j} \boldsymbol{y}_K[\boldsymbol{\delta b}_j]\right| \lesssim_K L_J L_\sigma^K C^{K-1}.\end{aligned}$$

Finally it follows by Lemma 1 that

$$\begin{aligned}|\mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}^1_{1:K}) - \mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}^2_{1:K})| &\lesssim \|\mathbf{W}^1_{1:K} - \mathbf{W}^2_{1:K}\|_{\text{block}}, \\ |\mathcal{L}_{\boldsymbol{b}_{1:K}}(\boldsymbol{b}^1_{1:K}) - \mathcal{L}_{\boldsymbol{b}_{1:K}}(\boldsymbol{b}^2_{1:K})| &\lesssim \|\boldsymbol{b}^1_{1:K} - \boldsymbol{b}^2_{1:K}\|_\infty.\end{aligned}$$

2.3. MOGA Optimizer: Fixing the Scaling Problems in $p \to q$ Geometry

In this section, we show that moving from the classical $p \to q$ geometry to the $(p, \mathtt{mean}) \to (q, \mathtt{mean})$ geometry requires only a corresponding width aware rescaling of the learning rate, since $(p, \mathtt{mean})$ and ℓ_p differ only by a $d^{1/p}$ factor on $\mathbb{R}^d$. Moreover, this rescaling coincides exactly with the μP scaling [52] in the special cases of `Adam` and `SignSGD`. Precisely, we have

FACT 4 **($(p,\texttt{mean}) \to (q,\texttt{mean})$ Geometry as Width-aware Scaling of $p \to q$ Geometry).** *For a matrix $\mathbf{D} \in \mathbb{R}^{\mathit{fan\_out} \times \mathit{fan\_in}}$ and $1 \le p \le \infty$, then we have*

$$\|\mathbf{D}\|_{(1,\mathit{mean}) \to (p,\mathit{mean})} = \frac{\mathit{fan\_in}}{\mathit{fan\_out}^{1/p}} \|\mathbf{D}\|_{1\to p}, \quad \|\mathbf{D}\|_{(p,\mathit{mean}) \to \infty} = \mathit{fan\_in}^{1/p} \|\mathbf{D}\|_{p\to\infty}. \tag{4}$$

Thus the steepest descent direction associated with the above two norms is steepest update in Proposition 1 with an appropriate width-dependent rescaling:

- $\arg\max_{\|X\|_{(1,\mathit{mean})\to(p,\mathit{mean})} \le 1} \langle X, \mathbf{G} \rangle = \frac{\mathit{fan\_out}^{1/p}}{\mathit{fan\_in}}\, \mathit{rownorm}_p(\mathbf{G})$,
- $\arg\max_{\|X\|_{(p,\mathit{mean})\to\infty} \le 1} \langle X, \mathbf{G} \rangle = \frac{1}{\mathit{fan\_in}^{1/p}}\, \mathit{colnorm}_p(\mathbf{G})$.

Proof of Fact 4. By definition of induced operator norm, we have $\|\mathbf{D}\|_{(1,\texttt{mean})\to(p,\texttt{mean})} = \sup_{\boldsymbol{x}\neq 0} \frac{\|\mathbf{D}\boldsymbol{x}\|_{(p,\texttt{mean})}}{\|\boldsymbol{x}\|_{(1,\texttt{mean})}} = \frac{\texttt{fan\_in}}{\texttt{fan\_out}^{1/p}} \sup_{\boldsymbol{x}\neq 0} \frac{\|\mathbf{D}\boldsymbol{x}\|_p}{\|\boldsymbol{x}\|_1} = \frac{\texttt{fan\_in}}{\texttt{fan\_out}^{1/p}} \|\mathbf{D}\|_{1\to p}$, and $\|\mathbf{D}\|_{(p,\texttt{mean})\to\infty} = \sup_{\boldsymbol{x}\neq 0} \frac{\|\mathbf{D}\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_{(p,\texttt{mean})}} = \texttt{fan\_in}^{1/p} \sup_{\boldsymbol{x}\neq 0} \frac{\|\mathbf{D}\boldsymbol{x}\|_\infty}{\|\boldsymbol{x}\|_p}. = \texttt{fan\_in}^{1/p} \|\mathbf{D}\|_{p\to\infty}$

The updates induced by the $(p,\texttt{mean}) \to (q,\texttt{mean})$ geometry can be viewed as a rescaled version of the updates under the standard $p \to q$ geometry. The associated scaling factor, highlighted in red in Fact 4, is referred to as MOGA (Matrix Operator Geometry Aware) scaling. **Somewhat surprisingly, we find that, in the case of Adam, MOGA scaling exactly recovers the μP scaling.** Intuitively, they are doing the same thing: rescaling the update direction by a $1/\text{fan}_\text{i}\text{n}$ factor. In Maximal Update Parametrization (μP) [51], parameters and learning rates are scaled to keep feature-level update magnitudes bounded as width grows, ensuring non-degenerate feature learning and width-independent optimization dynamics. Consequently, learning rates tuned on small proxy networks transfer robustly to much wider models. In Figure 2, we demonstrate learning-rate transfer across a broader family of MOGA optimizers. In Section 2.6, we further compare MOGA scaling with the μP implementation for Transformers and show that the two coincide exactly for the Adam optimizer.

By switching from $p \to q$ geometry to the $(p,\texttt{mean}) \to (q,\texttt{mean})$ geometry, we introduce a width-aware MOGA scaling for row/column normalization. Remarkably, under the $(1,\texttt{mean}) \to \ell_\infty$ operator norm, this MOGA scaling exactly recovers the μP scaling for Adam:

- If we use MAV $\to \ell_\infty$ norm do gradient descent, one need to scale the Adam/SignSgd updates by $\frac{1}{\texttt{fan\_in}}$ in each layer. Interestingly, our derived worst-case scaling coincides with the parameter choice in the μP parameterization [52, Table 3], which scales the update by $\frac{1}{\mathrm{fan}_{\mathrm{i}}\mathrm{n}}$ to achieve the same optimal convergence rate across different network widths, as predicted by an average-case analysis near initialization.

- If we do column-wise ℓ_2 normalization [17], the learning rate should scale $\frac{\mathtt{fan\_out}^{1/2}}{\mathtt{fan\_in}}$ in each layer. If we do row-wise ℓ_2 normalization, the learning rate should scale $\sqrt{\frac{1}{\mathtt{fan\_in}}}$ in each layer.

Remark 3. [53] proposed a spectral condition requiring the update direction to have spectral norm $\sqrt{\mathtt{fan\_out/fan\_in}}$, which recovers the μP scaling [52] for SGD. In our notation, this spectral condition is equivalent to demanding that the update direction has a width-independent $\|\cdot\|_{(2,\mathtt{mean})\to(2,\mathtt{mean})}$ norm. We first verify that our scaling for `Adam`/`SignSGD` satisfies the spectral condition of [53]. Since the mean–scaled norms obey $\|\cdot\|_{(1,\mathtt{mean})} \leq \|\cdot\|_{(2,\mathtt{mean})} \leq \|\cdot\|_{\infty}$, their induced operator norms satisfy $\|\cdot\|_{(1,\mathtt{mean})\to\infty} \leq \|\cdot\|_{(2,\mathtt{mean})\to(2,\mathtt{mean})}$. Consequently, any update direction normalized to have unit $\|\cdot\|_{(1,\mathtt{mean})\to\infty}$ norm automatically has a width–independent $\|\cdot\|_{(2,\mathtt{mean})\to(2,\mathtt{mean})}$ operator norm, and therefore satisfies the required spectral condition.

However, our scaling also reveals meaningful regimes in which the updates can remain stable even when the spectral condition is violated. To see this, consider the $(3,\text{mean})\to\ell_\infty$ operator norm of the update and let the descent direction $\mathbf{D}_n \in \mathbb{R}^{n\times n}$ be the rank-one matrix $(\mathbf{D}_n)_{i1} = n^{-1/3}$, $(\mathbf{D}_n)_{ij} = 0$ for $j \neq 1$. that satisfies $\|\mathbf{D}_n\|_{(3,\text{mean})\to\ell_\infty} = 1$. On the other hand, its spectral norm is $\|\mathbf{D}_n\|_{(2,\text{mean})\to(2,\text{mean})} = \|\mathbf{D}_n\|_{2\to2} = \left\|n^{-1/3}\mathbf{1}\right\|_2 = n^{-1/3}\sqrt{n} = n^{1/6}$, which grows with n and therefore does not satisfy the spectral condition.

Considering this scaling, we propose a simple yet general family of algorithms, which we refer to as `MOGA` (**M**atrix-**O**perator-**G**eometry-**A**ware):

2.4. How to select p, q ? Achieving Width-Independent Hessian Estimate

While [4, 25] suggest choosing norms that reflect the output's sensitivity to input and weight perturbations, M-Lipschitz functions provide limited structure and may not adequately capture deep learning dynamics. In this section, we also consider how the sensitivity of gradient changes that characterize the oscillates along the optimization trajectory, a property known as L-smoothness in optimization. L-smoothness means that a function's gradient changes in a controlled way—specifically, the gradient doesn't vary too quickly between nearby points. This property is important for optimization because it ensures that gradient-based methods can take stable, predictable steps and allows us to bound how much the function value decreases at each iteration.

Algorithm 1 Matrix-Operator-Geometry-Aware Steepest Descent

```
Require: Learning rate sequence $\{\eta^{(t)}\}$, momentum parameters $\alpha_1, \alpha_2$, initial parameter $\Theta^{(0)}$
$M^{(0)} \leftarrow \mathbf{0}$
for $t = 1, 2, \ldots$ do
    Sample randomized gradient: $\mathbf{G} \leftarrow \nabla L(\Theta^{(t-1)}; \xi),\ \xi \sim \mathcal{P}$
    Exponential moving average: $M^{(t)} \leftarrow \alpha_1 M^{(t-1)} + (1-\alpha_1)\mathbf{G}$
    Nesterov accelerated gradient: $\tilde{M}^{(t)} \leftarrow \alpha_2 M^{(t-1)} + (1-\alpha_2)\mathbf{G}$
    if Descent under $(1, \texttt{mean}) \to (q, \texttt{mean})$ then
        if $\Theta^{(t-1)} \in \mathbb{R}^{\texttt{fan\_out} \times \texttt{fan\_in}}$ is a weight matrix then
            for $r = 1$ to $n$ do
                $\Theta^{(t)}_{:,c} \leftarrow \Theta^{(t-1)}_{:,c} - \eta^{(t)} \frac{\texttt{fan\_out}^{1/q}}{\texttt{fan\_in}} \underbrace{\frac{\texttt{sign}(\tilde{M}_{:,c}) \odot |\tilde{M}_{:,c}|^{q^*-1}}{\|\tilde{M}_{:,c}\|_{q^*}^{q^*-1}}}_{\texttt{colnorm}_{q^*}(\mathbf{G})}$    ▷ We need $q \geq 2$.
            end for
        else if $\Theta^{(t-1)} \in \mathbb{R}^d$ is a bias vector then
            $\Theta^{(t)} \leftarrow \Theta^{(t-1)} - \eta^{(t)} \operatorname{sign}(\tilde{M})$
            or $\Theta^{(t)} \leftarrow \Theta^{(t-1)} - \eta^{(t)} \frac{\operatorname{sign}(\tilde{M}) \odot |\tilde{M}|^{q^*-1}}{\|\tilde{M}\|_{(q^*, \mathrm{mean})}^{q^*-1}}$,    ▷ Steepest descent under $(q, \texttt{mean})$ norm
        end if
    else if Descent under $(p, \texttt{mean}) \to \infty$ then
        if $\Theta^{(t-1)} \in \mathbb{R}^{\texttt{fan\_out} \times \texttt{fan\_in}}$ is a weight matrix then
            for $c = 1$ to $m$ do
                $\Theta^{(t)}_{r,:} \leftarrow \Theta^{(t-1)}_{r,:} - \eta^{(t)} \texttt{fan\_in}^{-1/p} \underbrace{\frac{\texttt{sign}(\tilde{M}_{r,:}) \odot |\tilde{M}_{r,:}|^{p-1}}{\|\tilde{M}_{r,:}\|_p^{p-1}}}_{\texttt{rownorm}_p(\mathbf{G})}$
            end for
        else if $\Theta^{(t-1)} \in \mathbb{R}^d$ is a bias vector then
            $\Theta^{(t)} \leftarrow \Theta^{(t-1)} - \eta^{(t)} \operatorname{sign}(\tilde{M})$
        end if
    end if
end for
```

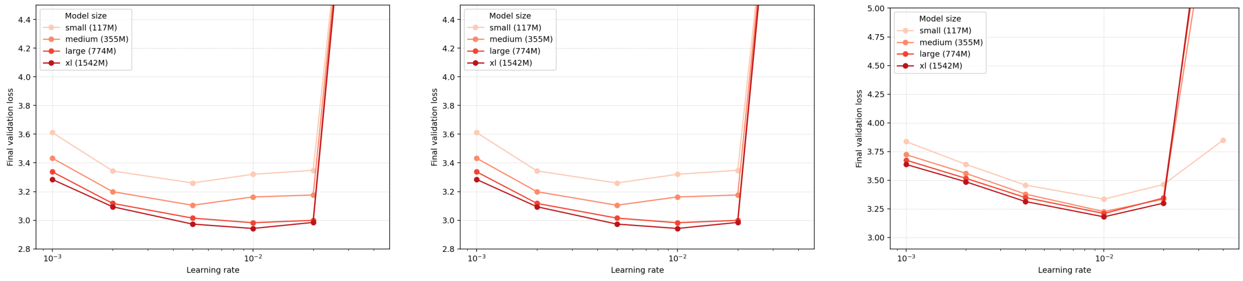

FIGURE 1. (a) MOGA (p=1.5)

FIGURE 1. (b) MOGA (p=2)

FIGURE 1. (c) MOGA (p=3)

FIGURE 2. **Learning Rate Transfer.** Performance of MOGA from GPT-2 Small to GPT-XL. The optimal learning rate of MOGA is invariant to width.

DEFINITION 3 (L-SMOOTHNESS, [26, 35, 43]). Let $f : \mathbf{B} \to \mathbb{R}$. We say that f is *L-smooth* with respect to the norm $\|\cdot\|_{\mathbf{B}}$ if, for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbf{B}$,

$$\|\nabla f(\boldsymbol{x}_1) - \nabla f(\boldsymbol{x}_2)\|_{\mathbf{B}^*} \le L\,\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_{\mathbf{B}},$$

where $\|\cdot\|_{\mathbf{B}^*}$ denotes the dual norm of $\|\cdot\|_{\mathbf{B}}$.

REMARK 4. Recent theoretical works [17, 24, 26, 35, 40, 43] have shown that, under the L-smoothness assumption, convergence results can be independent of the dimension, depending only on L.

Optimal Learning Rate Depends on the L-smoothness The optimal learning rate is determined by the L-smoothness of the objective function. This is because L-smoothness bounds how fast the gradient can change, which controls the largest step size that ensures stable convergence [8, 34]. This condition ensures that the curvature of f is controlled in the geometry induced by $\|\cdot\|_B$, implying the following quadratic upper bound:

$$\begin{aligned} f(\boldsymbol{y}) - f(\boldsymbol{x}) &= \langle \nabla f(\boldsymbol{x}), \boldsymbol{y}-\boldsymbol{x}\rangle + \int_0^1 \langle \nabla f(\boldsymbol{x} + t(\boldsymbol{y}-\boldsymbol{x})) - \nabla f(\boldsymbol{x}), \boldsymbol{y}-\boldsymbol{x}\rangle \, dt \\ &\le \langle \nabla f(\boldsymbol{x}), \boldsymbol{y}-\boldsymbol{x}\rangle + \int_0^1 \|\nabla f(\boldsymbol{x} + t(\boldsymbol{y}-\boldsymbol{x})) - \nabla f(\boldsymbol{x})\|_{B^*} \, \|\boldsymbol{y}-\boldsymbol{x}\|_B \, d \\ &\le \langle \nabla f(\boldsymbol{x}), \boldsymbol{y}-\boldsymbol{x}\rangle + \int_0^1 Lt\|\boldsymbol{y}-\boldsymbol{x}\|_B^2 \, d = \langle \nabla f(\boldsymbol{x}), \boldsymbol{y}-\boldsymbol{x}\rangle + \frac{L}{2}\|\boldsymbol{y}-\boldsymbol{x}\|_B^2. \end{aligned} \tag{5}$$

In this setting, the steepest descent direction under $\|\cdot\|_B$ is defined by $\mathbf{D}_t = \arg\min_{\|d\|_B \le 1} \langle \nabla f(\boldsymbol{x}_t), d\rangle = -\frac{\nabla f(\boldsymbol{x}_t)}{\|\nabla f(\boldsymbol{x}_t)\|_{B^*}}$, and the update rule becomes $\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta_t \mathbf{D}_t$. Based on the fact that $\langle \nabla f(\boldsymbol{x}_t), d_t\rangle = \|\nabla f(\boldsymbol{x}_t)\|_{B^*}$, under the L-smooth condition, we can bound the objective decrease as $f(\boldsymbol{x}_{t+1}) \le f(\boldsymbol{x}_t) - \eta_t \|\nabla f(\boldsymbol{x}_t)\|_{B^*} + \frac{L}{2}\eta_t^2$. Choosing the optimal step size $\eta_t = \|\nabla f(\boldsymbol{x}_t)\|_{B^*}/L$ yields $f(\boldsymbol{x}_{t+1}) \le f(\boldsymbol{x}_t) - \frac{1}{2L}\|\nabla f(\boldsymbol{x}_t)\|_{B^*}^2$. Hence, the L-smoothness condition ensures that the gradient field varies smoothly in the B-geometry, allowing the algorithm to take a large, stable step of order $1/L$ and achieve faster progress along the steepest descent direction.

Goal: Scale Independent Optimizer

Is it possible to define a norm under which the optimal learning rate for training a neural network becomes independent of its width? In other words, can both the M-Lipschitz, L-smoothness constant be made independent of network size?

We estimated the $\mathbf{L}$-smoothness of a neural network under the $(p, \texttt{mean}) \to (q, \texttt{mean})$ geometry (where $q > p$). Our results show that the $\mathbf{L}$-smoothness coefficient scales with the network width w as $w^{\max\{0,2/q-1/p\}}$.

Why L-Smooth is Harder We now intuitively explain why the feature-wise nonlinearities cause the L-smoothness coefficient to scale with the network width d as $d^{\max\{0,2/q-1/p\}}$. For a twice-differentiable function, the $\mathbf{L}$-smoothness coefficient is related to the magnitude of its Hessian matrix under $(p, \texttt{mean}) \to (q, \texttt{mean})$ geometry. (Lemma 2) Now let's consider the Hessian of the simplest neuron model $\sigma(\mathbf{W}\boldsymbol{x})$, where $\mathbf{W}$ is the weight, $\boldsymbol{x}$ is the input data and σ is a feature-wise nonlinearity. Since the activation functions σ in a neural network act feature-wise, the Taylor expansion also decomposes feature-wise, yielding a polynomial in each coordinate. For a perturbation $\mathbf{W} \mapsto \mathbf{W} + \Delta\mathbf{W}$, we obtain

$$\sigma((\mathbf{W} + \boldsymbol{\Delta}\mathbf{W})\boldsymbol{x}) \approx \sigma(\mathbf{W}X) + \operatorname{diag}(\sigma'(\mathbf{W}\boldsymbol{x}))\,\boldsymbol{\Delta}\mathbf{W}\boldsymbol{x} + \frac{1}{2}\operatorname{diag}(\sigma''(\mathbf{W}\boldsymbol{x}))\;\boldsymbol{\Delta}\mathbf{W}\boldsymbol{x} \odot \boldsymbol{\Delta}\mathbf{W}\boldsymbol{x}\; + \cdots$$

where $\odot$ denotes element-wise multiplication (i.e., the Hadamard product), such that $(\boldsymbol{x} \odot \boldsymbol{y})_i = x_i y_i$. To estimate the L-smoothness of the map $\boldsymbol{x} \mapsto \sigma(\mathbf{W}\boldsymbol{x})$ under the in $\to$ out geometry, we must control the directional Hessian along perturbations $\boldsymbol{\Delta}\mathbf{W}$, namely the second-order term in the Taylor expansion of $\sigma((\mathbf{W} + \boldsymbol{\Delta}\mathbf{W})\boldsymbol{x})$. Since the second-order term takes the explicit form $\operatorname{diag}(\sigma''(\mathbf{W}x))\,(\boldsymbol{\Delta}\mathbf{W}x \odot \boldsymbol{\Delta}\mathbf{W}x)$, we must obtain a $\|\cdot\|_{\text{in}}$ bound on the quadratic perturbation $\boldsymbol{\Delta}\mathbf{W}x \odot \boldsymbol{\Delta}\mathbf{W}x$. However, because $\Delta\mathbf{W}$ is measured in the operator norm $\|\cdot\|_{\text{in}\to\text{out}}$, the only information we directly control is $\|\boldsymbol{\Delta}\mathbf{W}x\|_{\text{out}}$. To translate this into a bound in the in-norm, the input and output norms must be compatible with the Hadamard product. This leads naturally to the norm-compatibility condition $\|\mathbf{u} \odot \boldsymbol{v}\|_{\text{in}} \leq \|\mathbf{u}\|_{\text{out}}\,\|\boldsymbol{v}\|_{\text{out}}, \forall u, v \in \mathbb{R}^d$, which ensures that the second-order perturbation is controlled and yields a width-independent L-smoothness estimate under the chosen geometry.

Precisely, based on the fact that $\|\boldsymbol{x} \odot \boldsymbol{y}\|_{(p,\texttt{mean})} \leq d^{\max\left(0,2/q-1/p\right)} \|\boldsymbol{x}\|_{(q,\texttt{mean})} \|\boldsymbol{y}\|_{(q,\texttt{mean})}$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$, we can show that the $\mathbf{L}$-smoothness coefficient of a neural network with width w scales as $w^{\max\{0,2/q-1/p\}}$ under $(p, \texttt{mean}) \to (q, \texttt{mean})$ geometry.

THEOREM 5. *Consider a feedforward neural network $\boldsymbol{y}_i(\mathbf{W}_{1:K}, \boldsymbol{b}_{1:K}; \boldsymbol{x})$ with K-1 hidden layer, and a learning objective function $f(\mathbf{W}) = \mathcal{L}(y_l(\mathbf{W}; \boldsymbol{x}))$, where the activation function σ and the loss function $\mathcal{L}$ satisfy Assumption 1 and 2. We further assume that the input satisfies $\|x\|_2 \leq C$ and that*

$$\max\left\{\|\boldsymbol{b}_i^j\|_\infty, \|\mathbf{W}_1^j\|_{1\to(q,mean)}, \max_{2\leq i\leq K-1} \|\mathbf{W}_i^j\|_{(p,mean)\to(q,mean)}, \|\mathbf{W}_K^j\|_{(p,mean)\to\infty}\right\} \leq C, j = 1, 2$$

where $0 < p < q$ and $C > 1$. Then the $L-$smoothness estimation has the following width dependency

$$\|\nabla\mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}_{1:K}^1) - \nabla\mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}_{1:K}^2)\|_{block,*} \lesssim w^{\max\{0,2/q-1/p\}}\|\mathbf{W}_{1:K}^1 - \mathbf{W}_{1:K}^2\|_{block}$$
$$\|\nabla_{\boldsymbol{b}_{1:K}}\mathcal{L}(\boldsymbol{b}_{1:K}^1) - \nabla_{\boldsymbol{b}_{1:K}}\mathcal{L}(\boldsymbol{b}_{1:K}^2)\|_1 \lesssim w^{\max\{0,2/q-1/p\}}\|\boldsymbol{b}_{1:K}^1 - \boldsymbol{b}_{1:K}^2\|_\infty$$

where the block norm is defined as

$$\|\mathbf{W}_{1:K}\|_{block} := \max\{\|\mathbf{W}_1\|_{1\to(q,mean)}, \max_{2\leq i\leq K-1} \|\mathbf{W}_i\|_{(p,mean)\to(q,mean)}, \|\mathbf{W}_K\|_{(p,mean)\to\infty}\}$$

and $\|\cdot\|_{block,*}$ is the dual norm of the Block norm.

Proof of Theorem 5. We first establish several Lemmas that will be useful later. The first Lemma connects the Lipschitz constant of the gradient of the loss function $\nabla\mathcal{L}$ to the magnitude of the directional Hessian. The second Lemma introduces a key norm inequality.

LEMMA 2 **(Directional Hessian Lipschitz Bound).** *Let $f : \mathbb{R}^{m\times n} \to \mathbb{R}$ be a twice differentiable function, and $\|\cdot\|_B$ be any matrix norm. Suppose that for all $\mathbf{Z}$ on the line segment between $\mathbf{Z}^1$ and $\mathbf{Z}^2$, and for all matrices $\mathbf{\Delta U}, \mathbf{\Delta V} \in \mathbb{R}^{m\times n}$ with $\|\mathbf{\Delta U}\|_B \leq 1$ and $\|\mathbf{\Delta V}\|_B \leq 1$, the directional Hessian satisfies $|\nabla^2 f(\mathbf{Z})[\mathbf{\Delta U}, \mathbf{\Delta V}]| \leq L$. Then, the gradient of f satisfies $\|\nabla f(\mathbf{Z}^1) - \nabla f(\mathbf{Z}^2)\|_{B^*} \leq L\|\mathbf{Z}^1 - \mathbf{Z}^2\|_B$, where $\|\cdot\|_{B^*}$ is the dual norm of $\|\cdot\|_B$.*

Proof of Lemma 2. By the fundamental theorem of calculus in Banach spaces, we have

$$\nabla f(\mathbf{Z}^1) - \nabla f(\mathbf{Z}^2) = \int_0^1 \nabla^2 f(\mathbf{Z}^2 + t(\mathbf{Z}^1 - \mathbf{Z}^2))[\mathbf{Z}^1 - \mathbf{Z}^2]\, dt.$$

Taking the dual norm (nuclear norm) and using the definition,

$$\begin{aligned}\|\nabla f(\mathbf{Z}^1) - \nabla f(\mathbf{Z}^2)\|_{\mathrm{B}^*} &= \sup_{\|\mathbf{\Delta U}\|_{\mathrm{B}} \le 1} \int_0^1 \nabla^2 f(\mathbf{Z}^2 + t(\mathbf{Z}^1 - \mathbf{Z}^2))[\mathbf{\Delta U}, \mathbf{Z}^1 - \mathbf{Z}^2]\, dt \\ &\le \int_0^1 \sup_{\|\mathbf{\Delta U}\|_{\mathrm{B}} \le 1} \nabla^2 f(\mathbf{Z}^2 + t(\mathbf{Z}^1 - \mathbf{Z}^2))[\mathbf{\Delta U}, \mathbf{Z}^1 - \mathbf{Z}^2]\, dt.\end{aligned}$$

Now, for each $t \in [0,1]$, let $\mathbf{\Delta V} = \frac{\mathbf{Z}^1 - \mathbf{Z}^2}{\|\mathbf{Z}^1 - \mathbf{Z}^2\|_{\mathrm{B}}}$ so that $\|\mathbf{\Delta V}\|_{\mathrm{B}} = 1$. By the directional Hessian bound,

$$\begin{aligned}\nabla^2 f(\mathbf{Z}^2 + t(\mathbf{Z}^1 - \mathbf{Z}^2))[\mathbf{\Delta U}, \mathbf{Z}^1 - \mathbf{Z}^2] &= \|\mathbf{Z}^1 - \mathbf{Z}^2\|_{\mathrm{B}} \cdot |\nabla^2 f(\mathbf{Z}^2 + t(\mathbf{Z}^1 - \mathbf{Z}^2))[\mathbf{\Delta U}, \mathbf{\Delta V}] \\ &\le L\, \|\mathbf{Z}^1 - \mathbf{Z}^2\|_{\mathrm{B}}.\end{aligned}$$

Integrating over $t \in [0,1]$ yields $\|\nabla f(\mathbf{Z}^1) - \nabla f(\mathbf{Z}^2)\|_{\mathrm{B}^*} \le \int_0^1 L\, \|\mathbf{Z}^1 - \mathbf{Z}^2\|_{\mathrm{B}}\, dt = L\, \|\mathbf{Z}^1 - \mathbf{Z}^2\|_{\mathrm{B}}$. This completes the proof.

Before estimating the directional Hessian of the neural network, we first record two basic properties of the $(p, \mathtt{mean})$ norm.

FACT 6. *Let $z \in \mathbb{R}^n$, then for any $s, t > 0$, $\|z\|_{(s,mean)} \le n^{\max(0, \frac{1}{t} - \frac{1}{s})} \|z\|_{(t,mean)}$.*

Proof of Fact 6. Note that $\|z\|_{(p,\mathtt{mean})} = n^{-1/p}\|z\|_p$. It is standard that for $p \ge q > 0$, $\|z\|_p \le \|z\|_q \le n^{1/q - 1/p}\|z\|_p$. Now set $p = s$ and $q = t$. If $s \ge t$ then

$$\|z\|_{(s,\mathtt{mean})} = n^{-1/s}\|z\|_s \le n^{-1/s}\, n^{1/t - 1/s}\|z\|_t = n^{1/t - 1/s}\|z\|_{(t,\mathtt{mean})}.$$

If $s \le t$, the monotonicity of the power means gives $\|z\|_{(s,\mathtt{mean})} \le \|z\|_{(t,\mathtt{mean})}$, thus we have

$$\|z\|_{(s,\mathtt{mean})} \le n^{\max(0, 1/t - 1/s)}\|z\|_{(t,\mathtt{mean})},$$

which is the claimed inequality.

LEMMA 3. *Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. For all $p, q > 0$, we have $\|\boldsymbol{x} \odot \boldsymbol{y}\|_{(p,mean)} \le n^{\max\left(0, 2/q - 1/p\right)}\, \|\boldsymbol{x}\|_{(q,mean)}\, \|\boldsymbol{y}\|_{(q,mean)}$.*

Proof of Lemma 3. By applying Cauchy–Schwarz we have $\|\boldsymbol{x} \odot \boldsymbol{y}\|_{(p,\texttt{mean})} \leq \|\boldsymbol{x}\|_{(2p,\texttt{mean})} \|\boldsymbol{y}\|_{(2p,\texttt{mean})}$. Applying Fact 6 with $s = 2p$ and $t = q$ to each of the two factors above gives

$$\|\boldsymbol{x}\|_{(2p,\texttt{mean})} \leq n^{\max\left(0,\,1/q-1/2p\right)} \|\boldsymbol{x}\|_{(q,\texttt{mean})},$$
$$\|\boldsymbol{y}\|_{(2p,\texttt{mean})} \leq n^{\max\left(0,\,1/q-1/2p\right)} \|\boldsymbol{y}\|_{(q,\texttt{mean})}.$$

Thus we have $\|\boldsymbol{x} \odot \boldsymbol{y}\|_{(p,\texttt{mean})} \leq \|\boldsymbol{x}\|_{(2p,\texttt{mean})}\|\boldsymbol{y}\|_{(2p,\texttt{mean})} \leq n^{\max\left(0,\,2/q-1/p\right)} \|\boldsymbol{x}\|_{(q,\texttt{mean})} \|\boldsymbol{y}\|_{(q,\texttt{mean})}$.

In Theorem 3, we already provide a dimension-independent norm estimate of the feature $\boldsymbol{y}_i$ and directional gradient $\nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\cdot]$. Next, we bound the directional Hessian of the neural network. As a first step, we compute the full Hessian of the network. Following the Definition 2 and the standard computation, and set $\boldsymbol{y}_0 = x$, then for $j \leq l$ we have

$$\nabla^2_{\mathbf{W}_{j,l}}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}] = \begin{cases} \mathbf{D}_i^{(2)}\Big((\boldsymbol{\Delta}\mathbf{U}\boldsymbol{y}_{i-1}) \odot (\boldsymbol{\Delta}\mathbf{V}\boldsymbol{y}_{i-1})\Big), & j = l = i, \\ \mathbf{D}_i^{(2)}\Big((\mathbf{W}_i \nabla_{\mathbf{W}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{U}]) \odot (\boldsymbol{\Delta}\mathbf{V}\boldsymbol{y}_{i-1})\Big) & j < l = i, \\ \qquad +\mathbf{D}_i^{(1)} \boldsymbol{\Delta}\mathbf{V} \nabla_{\mathbf{W}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{U}], & \\ \mathbf{D}_i^{(2)}\Big((\mathbf{W}_i \nabla_{\mathbf{W}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{U}]) \odot (\mathbf{W}_i \nabla_{\mathbf{W}_l}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{V}])\Big) & l < i. \\ \qquad +\mathbf{D}_i^{(1)}\mathbf{W}_i \nabla^2_{\mathbf{W}_{j,l}}\boldsymbol{y}_{i-1}[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}], & \end{cases}$$

$$\nabla^2_{\boldsymbol{b}_{j,l}}\boldsymbol{y}_i[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta v}] = \begin{cases} \mathbf{D}_i^{(2)}\big(\boldsymbol{\delta}\mathbf{u} \odot \boldsymbol{\delta v}\big), & j = l = i, \\ \mathbf{D}_i^{(2)}\big(\mathbf{W}_i \nabla_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\delta}\mathbf{u}] \odot \boldsymbol{\delta v}\big), & j < l = i, \\ \mathbf{D}_i^{(2)}\Big((\mathbf{W}_i \nabla_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\delta}\mathbf{u}]) \odot (\mathbf{W}_i \nabla_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\delta v}])\Big) & l < i. \\ \qquad +\mathbf{D}_i^{(1)}\mathbf{W}_i \nabla^2_{\boldsymbol{b}_j}\boldsymbol{y}_{i-1}[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta v}], & \end{cases}$$

for $1 \leq i \leq K$, where $\mathbf{D}_i^{(1)} = \texttt{diag}\left(\sigma'(z_i)\right), \mathbf{D}_i^{(2)} = \texttt{diag}\left(\sigma''(z_i)\right)$.

We then consider three separate cases for p and i, where we use the previous bounds $\|\boldsymbol{y}_i\| \leq 2^{i-1}L_\sigma^i C^{i+1}$, $\left\|\nabla_{\mathbf{W}_j}\boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{W}]\right\|_{(p,\texttt{mean})} \leq (2L_\sigma)^i C^{i+1}$ and $\left\|\nabla_{\boldsymbol{b}_j}\boldsymbol{y}_i[\boldsymbol{\delta b}]\right\|_{(p,\texttt{mean})} \leq L_\sigma^i C^{i-1}$. For simplicity, we let

$$
\begin{aligned}
&\boldsymbol{\Delta}_{i,j}^{\mathbf{U}} := \nabla_{\mathbf{W}_j} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}], \quad \boldsymbol{\Delta}_{i,j}^{\mathbf{V}} := \nabla_{\mathbf{W}_j} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{V}], \quad \boldsymbol{\Delta}_{j,l,i}^{\mathbf{U},\mathbf{V}} := \nabla^2_{\mathbf{W}_{j,l}} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}] \\
&\boldsymbol{\delta}_{i,j}^{\mathbf{u}} := \nabla_{\boldsymbol{b}_j} \boldsymbol{y}_i[\boldsymbol{\delta}\mathbf{u}], \qquad \boldsymbol{\delta}_{i,j}^{\boldsymbol{v}} := \nabla_{\boldsymbol{b}_j} \boldsymbol{y}_i[\boldsymbol{\delta}\boldsymbol{v}], \qquad \boldsymbol{\delta}_{j,l,i}^{\mathbf{u},\boldsymbol{v}} := \nabla^2_{\boldsymbol{b}_{j,l}} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]
\end{aligned}
$$

- **Case 1** ($j = l = i$) For $2 \le i \le K-1$, we have

$$
\begin{aligned}
&\left\|\nabla^2_{\mathbf{W}_{i,i}} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]\right\|_{(p,\texttt{mean})} \le \|\mathbf{D}_i^{(2)}\|_\infty \|(\boldsymbol{\Delta}\mathbf{U}\boldsymbol{y}_{i-1}) \odot (\boldsymbol{\Delta}\mathbf{V}\boldsymbol{y}_{i-1})\|_{(p,\texttt{mean})} \\
&\le M_\sigma w^{\max\left(0,2/q-1/p\right)} \|\boldsymbol{\Delta}\mathbf{U}\boldsymbol{y}_{i-1}\|_{(q,\texttt{mean})} \|\boldsymbol{\Delta}\mathbf{V}\boldsymbol{y}_{i-1}\|_{(q,\texttt{mean})} \quad \text{(Lemma 3)} \\
&\le M_\sigma w^{\max\left(0,2/q-1/p\right)} \|\boldsymbol{\Delta}\mathbf{U}\|_{(p,\texttt{mean})\to(q,\texttt{mean})} \|\boldsymbol{y}_{i-1}\|_{(p,\texttt{mean})} \|\boldsymbol{\Delta}\mathbf{V}\|_{(p,\texttt{mean})\to(q,\texttt{mean})} \|\boldsymbol{y}_{i-1}\|_{(p,\texttt{mean})} \\
&\le M_\sigma w^{\max\left(0,2/q-1/p\right)} \|\boldsymbol{y}_{i-1}\|^2_{(q,\texttt{mean})} \le w^{\max\left(0,2/q-1/p\right)} 2^{2i-2} M_\sigma L_\sigma^{2i-2} C^{2i}, \\
&\left\|\nabla^2_{\boldsymbol{b}_{i,i}} \boldsymbol{y}_i[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_\infty \le \|\mathbf{D}_i^{(2)}\|_\infty \|\boldsymbol{\delta}\mathbf{u} \odot \boldsymbol{\delta}\boldsymbol{v}\|_\infty \le M_\sigma
\end{aligned}
$$

Similarly for $i = 1, K$, by replacing $\|\cdot\|_{(p,\texttt{mean})\to(q,\texttt{mean})}$ with $\|\cdot\|_{1\to(q,\texttt{mean})}$ and $\|\cdot\|_{(p,\texttt{mean})\to\infty}$ respectively, we have

$$
\begin{aligned}
&\left\|\nabla^2_{\mathbf{W}_{1,1}} \boldsymbol{y}_1[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]\right\|_{(p,\texttt{mean})} \le M_\sigma w^{\max\left(0,2/q-1/p\right)} C^2 \\
&\left\|\nabla^2_{\mathbf{W}_{K,K}} y_K[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]\right\|_{(p,\texttt{mean})} \le M_\sigma w^{\max\left(0,2/q-1/p\right)} 2^{2K-2} L_\sigma^{2K-2} C^{2K}
\end{aligned}
$$

and

$$
\begin{aligned}
&\left\|\nabla^2_{\boldsymbol{b}_{1,1}} \boldsymbol{y}_1[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_{(p,\texttt{mean})} \le \left\|\nabla^2_{\boldsymbol{b}_{1,1}} \boldsymbol{y}_1[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_\infty \le \|\mathbf{D}_1^{(2)}\|_\infty \|\boldsymbol{\delta}\mathbf{u} \odot \boldsymbol{\delta}\boldsymbol{v}\|_\infty \le M_\sigma \\
&\left\|\nabla^2_{\boldsymbol{b}_{K,K}} \boldsymbol{y}_K[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_{(p,\texttt{mean})} \le \left\|\nabla^2_{\boldsymbol{b}_{K,K}} \boldsymbol{y}_K[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_\infty \le \|\mathbf{D}_K^{(2)}\|_\infty \|\boldsymbol{\delta}\mathbf{u} \odot \boldsymbol{\delta}\boldsymbol{v}\|_\infty \le M_\sigma.
\end{aligned}
$$

- **Case 2** ($j < l = i$) For $2 \le i \le K - 1$, we have

$$\left\|\nabla^2_{\mathbf{W}_{j,i}} \boldsymbol{y}_i [\mathbf{\Delta U}, \mathbf{\Delta V}]\right\|_{(p,\texttt{mean})} = \left\|\mathbf{D}_i^{(2)} \left((\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}) \odot (\mathbf{\Delta V} \boldsymbol{y}_{\boldsymbol{i-1}})\right) + \mathbf{D}_i^{(1)} \mathbf{\Delta V} \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\right\|_{(p,\texttt{mean})}$$
$$\le \|\mathbf{D}_i^{(2)}\|_\infty \left\|\left((\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}) \odot (\mathbf{\Delta V} \boldsymbol{y}_{\boldsymbol{i-1}})\right)\right\|_{(p,\texttt{mean})} + \left\|\mathbf{D}_i^{(1)} \mathbf{\Delta V} \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\right\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \|\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\|_{(q,\texttt{mean})} \|\mathbf{\Delta V} \boldsymbol{y}_{i-1}\|_{(q,\texttt{mean})}$$
$$+ L_\sigma \|\mathbf{\Delta V}\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \|\mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\|_{(p,\texttt{mean})} \quad \text{(Lemma 3)}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \|\mathbf{W}_i\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \|\mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\|_{(p,\texttt{mean})} \|\mathbf{\Delta V}\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \|\boldsymbol{y}_{i-1}\|_{(q,\texttt{mean})}$$
$$+ L_\sigma \|\mathbf{\Delta V}\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \|\mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} 2^{2i-3} L_\sigma^{2i-2} C^{2i+1} + 2^{i-1} L_\sigma^i C^i$$

$$\left\|\nabla^2_{\boldsymbol{b}_{j,i}} \boldsymbol{y}_i [\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta v}]\right\|_{(p,\texttt{mean})} \le \|\mathbf{D}_i^{(2)}\|_\infty \|(\mathbf{W}_i \nabla_{\boldsymbol{b}_j} \boldsymbol{y}_{i-1} [\boldsymbol{\delta}\mathbf{u}]) \odot \boldsymbol{\delta v}\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \|W_i\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \|\nabla_{\boldsymbol{b}_j} \boldsymbol{y}_{i-1} [\boldsymbol{\delta}\mathbf{u}]\|_{(p,\texttt{mean})} \|\boldsymbol{\delta v}\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} L_\sigma^{i-1} C^{i-1}$$

- **Case 3** ($l < i, i \ne K$) By the computation of Hessian we have

$$\left\|\nabla^2_{\mathbf{W}_{j,l}} \boldsymbol{y}_i [\mathbf{\Delta U}, \mathbf{\Delta V}]\right\|_{(p,\texttt{mean})} = \left\|\mathbf{D}_i^{(2)} \left((\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}) \odot (\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{V}}_{i-1,l})\right) + \mathbf{D}_i^{(1)} \mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,i-1}\right\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \left\|\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\right\|_{(q,\texttt{mean})} \left\|\mathbf{W}_i \, \mathbf{\Delta}^{\mathbf{V}}_{i-1,l}\right\|_{(q,\texttt{mean})}$$
$$+ L_\sigma \|\mathbf{W}_i\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \left\|\mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,i-1}\right\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \|\mathbf{W}_i\|^2_{(p,\texttt{mean}) \to (q,\texttt{mean})} \left\|\mathbf{\Delta}^{\mathbf{U}}_{i-1,j}\right\|_{(p,\texttt{mean})} \left\|\mathbf{\Delta}^{\mathbf{V}}_{i-1,l}\right\|_{(p,\texttt{mean})}$$
$$+ L_\sigma \|\mathbf{W}_i\|_{(p,\texttt{mean}) \to (q,\texttt{mean})} \left\|\mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,i-1}\right\|_{(p,\texttt{mean})}$$
$$\le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} (2L_\sigma)^{2i-2} C^{2i+2} + L_\sigma C \left\|\mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,i-1}\right\|_{(p,\texttt{mean})}$$

Thus

$$\left\|\mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,i}\right\|_{(p,\texttt{mean})} \le M_\sigma w^{\max\left(0, 2/q - 1/p\right)} \sum_{k=0}^{i-l-1} 2^{2i-2-2k} L_\sigma^{2i-2-k} C^{2i+2-k} + (L_\sigma C)^{i-l} \left\|\mathbf{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,l}\right\|_{(p,\texttt{mean})}$$
$$\le i \, w^{\max\left(0, 2/q - 1/p\right)} M_\sigma (2L_\sigma)^{2i-2} C^{2i+2} + w^{\max\left(0, 2/q - 1/p\right)} M_\sigma (2L_\sigma)^{2i-2} C^{2i+2}$$
$$\lesssim w^{\max\left(0, 2/q - 1/p\right)} M_\sigma (2L_\sigma)^{2i-2} C^{2i+2}.$$

On the other hand,

$$\begin{aligned}
&\left\|\nabla^2_{\boldsymbol{b}_{j,l}} \boldsymbol{y}_i[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta v}]\right\|_{(p,\mathtt{mean})} = \left\|\mathbf{D}_i^{(2)}\left((\mathbf{W}_i\,\boldsymbol{\delta}^{\mathbf{u}}_{i-1,j}) \odot (\mathbf{W}_i\,\boldsymbol{\delta}^{\mathbf{u}}_{i-1,l})\right) + \mathbf{D}_i^{(1)}\mathbf{W}_i\,\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,i-1}\right\|_{(p,\mathtt{mean})} \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \left\|\mathbf{W}_i\,\boldsymbol{\delta}^{\mathbf{u}}_{i-1,j}\right\|_{(q,\mathtt{mean})} \left\|\mathbf{W}_i\,\boldsymbol{\delta}^{\boldsymbol{v}}_{i-1,l}\right\|_{(q,\mathtt{mean})} + L_\sigma \left\|\mathbf{W}_i\,\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,i-1}\right\|_{(q,\mathtt{mean})} \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \|\mathbf{W}_i\|^2_{(p,\mathtt{mean})\to(q,\mathtt{mean})} \left\|\boldsymbol{\delta}^{\mathbf{u}}_{i-1,j}\right\|_{(p,\mathtt{mean})} \left\|\boldsymbol{\delta}^{\boldsymbol{v}}_{i-1,l}\right\|_{(p,\mathtt{mean})} \\
&+ L_\sigma \|\mathbf{W}_i\|_{(p,\mathtt{mean})\to(q,\mathtt{mean})} \left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,i-1}\right\|_{(p,\mathtt{mean})} \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} (L_\sigma C)^{2i-2} + L_\sigma C \left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,i-1}\right\|_{(p,\mathtt{mean})}
\end{aligned}$$

Thus

$$\begin{aligned}
\left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,i}\right\|_{(p,\mathtt{mean})} &\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \sum_{j=0}^{i-j-1} (L_\sigma C)^{2i-2-j} + (L_\sigma C)^{i-j} \left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,l}\right\|_{(p,\mathtt{mean})} \\
&\le i\, w^{\max\left(0,\,2/q-1/p\right)} M_\sigma (L_\sigma C)^{2i-2} + w^{\max\left(0,\,2/q-1/p\right)} M_\sigma (L_\sigma C)^{i} \\
&\lesssim w^{\max\left(0,\,2/q-1/p\right)} M_\sigma (L_\sigma C)^{2i-2}.
\end{aligned}$$

- **Case 4** ($l < i = K$) With a similar computation as Case 2, we have

$$\begin{aligned}
&\left|\nabla^2_{\mathbf{W}_{j,l}} y_K[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]\right| = \left|\mathbf{D}_K^{(2)}\left((\mathbf{W}_K\,\boldsymbol{\Delta}^{\mathbf{U}}_{K-1,j}) \odot (\mathbf{W}_K\,\boldsymbol{\Delta}^{\mathbf{V}}_{K-1,l})\right) + \mathbf{D}_K^{(1)}\mathbf{W}_K\,\boldsymbol{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,K-1}\right| \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \left|\mathbf{W}_K\,\boldsymbol{\Delta}^{\mathbf{U}}_{K-1,j}\right| \left|\mathbf{W}_K\,\boldsymbol{\Delta}^{\mathbf{V}}_{K-1,l}\right| + L_\sigma \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,K-1}\right\|_{(p,\mathtt{mean})} \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \|\mathbf{W}_i\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\Delta}^{\mathbf{U}}_{K-1,j}\right\|_{(p,\mathtt{mean})} \|\mathbf{W}_i\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\Delta}^{\mathbf{V}}_{K-1,l}\right\|_{(p,\mathtt{mean})} \\
&+ L_\sigma C \left\|\boldsymbol{\Delta}^{\mathbf{U},\mathbf{V}}_{j,l,K-1}\right\|_{(p,\mathtt{mean})} \\
&\lesssim M_\sigma w^{\max\left(0,\,2/q-1/p\right)} (2L_\sigma)^{2K-2} C^{2K+2}.
\end{aligned}$$

and

$$\begin{aligned}
&\left|\nabla^2_{\boldsymbol{b}_{j,l}} y_K[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta v}]\right| = \left|\mathbf{D}_K^{(2)}\left((\mathbf{W}_K\,\boldsymbol{\delta}^{\mathbf{u}}_{K-1,j}) \odot (\mathbf{W}_K\,\boldsymbol{\delta}^{\boldsymbol{v}}_{K-1,l})\right) + \mathbf{D}_K^{(1)}\mathbf{W}_K\,\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,K-1}\right| \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \left|\mathbf{W}_K\,\boldsymbol{\delta}^{\mathbf{u}}_{K-1,j}\right| \left|\mathbf{W}_K\,\boldsymbol{\delta}^{\boldsymbol{v}}_{K-1,l}\right| + L_\sigma \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,K-1}\right\|_{(p,\mathtt{mean})} \\
&\le M_\sigma w^{\max\left(0,\,2/q-1/p\right)} \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\delta}^{\mathbf{u}}_{K-1,j}\right\|_{(p,\mathtt{mean})} \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\delta}^{\boldsymbol{v}}_{K-1,l}\right\|_{(p,\mathtt{mean})} \\
&+ L_\sigma \|\mathbf{W}_K\|_{(p,\mathtt{mean})\to\infty} \left\|\boldsymbol{\delta}^{\mathbf{u},\boldsymbol{v}}_{j,l,K-1}\right\|_{(p,\mathtt{mean})} \\
&\lesssim M_\sigma w^{\max\left(0,\,2/q-1/p\right)} (L_\sigma C)^{2K-2}
\end{aligned}$$

Thus all $\left\|\nabla^2_{\mathbf{W}_{j,l}} \boldsymbol{y}_i[\boldsymbol{\Delta}\mathbf{U}, \boldsymbol{\Delta}\mathbf{V}]\right\|_{(p,\texttt{mean})}$ can be bounded by $w^{\max\left(0,\, 2/q-1/p\right)} M_\sigma (2L_\sigma)^{2i-2} C^{2i+2}$ up to a constant and all $\left\|\nabla^2_{\boldsymbol{b}_j} \boldsymbol{y}_i[\boldsymbol{\delta}\mathbf{u}, \boldsymbol{\delta}\boldsymbol{v}]\right\|_{(p,\texttt{mean})}$ can be bounded by $w^{\max\left(0,\, 2/q-1/p\right)} M_\sigma (L_\sigma C)^{2i-2}$ up to a constant, which indicates the dependence of width w is $w^{\max\left(0,\, 2/q-1/p\right)}$.

Note that

$$\begin{aligned}
\nabla^2_{\mathbf{W}_{1:K}} \mathcal{L}[\boldsymbol{\Delta}\mathbf{U}_{1:K}, \boldsymbol{\Delta}\mathbf{V}_{1:K}] &= \mathcal{L}''(y_K) \nabla_{\mathbf{W}_{1:K}} y_K[\boldsymbol{\Delta}\mathbf{U}_{1:K}] \cdot \nabla_{\mathbf{W}_{1:K}} y_K[\boldsymbol{\Delta}\mathbf{V}_{1:K}] \\
&\quad + \mathcal{L}'(y_K) \nabla^2_{\mathbf{W}_{1:K}} y_K[\boldsymbol{\Delta}\mathbf{U}_{1:K}, \boldsymbol{\Delta}\mathbf{V}_{1:K}] \\
&= \mathcal{L}''(y_K) (\sum_{j=1}^{K} \nabla_{\mathbf{W}_j} y_K[\boldsymbol{\Delta}\mathbf{U}_j]) \cdot (\sum_{j=1}^{K} \nabla_{\mathbf{W}_j} y_K[\boldsymbol{\Delta}\mathbf{V}_j]) \\
&\quad + \mathcal{L}'(y_K) \sum_{1 \le j,k \le K} \nabla^2_{\mathbf{W}_{j,l}} y_K[\boldsymbol{\Delta}\mathbf{U}_j, \boldsymbol{\Delta}\mathbf{V}_l] \\
\nabla^2_{\boldsymbol{b}_{1:K}} \mathcal{L}[\boldsymbol{\delta}\mathbf{u}_{1:K}, \boldsymbol{\delta}\boldsymbol{v}_{1:K}] &= \mathcal{L}''(y_K) \nabla_{\boldsymbol{b}_{1:K}} y_K[\boldsymbol{\delta}\mathbf{u}_{1:K}] \cdot \nabla_{\boldsymbol{b}_{1:K}} y_K[\boldsymbol{\delta}\boldsymbol{v}_{1:K}] \\
&\quad + \mathcal{L}'(y_K) \nabla^2_{\boldsymbol{b}_{1:K}} y_K[\boldsymbol{\delta}\mathbf{u}_{1:K}, \boldsymbol{\delta}\boldsymbol{v}_{1:K}] \\
&= \mathcal{L}''(y_K) (\sum_{j=1}^{K} \nabla_{\boldsymbol{b}_j} y_K[\boldsymbol{\delta}\mathbf{u}_j]) \cdot (\sum_{j=1}^{K} \nabla_{\boldsymbol{b}_j} y_K[\boldsymbol{\delta}\boldsymbol{v}_j]) \\
&\quad + \mathcal{L}'(y_K) \sum_{1 \le j,l \le K} \nabla^2_{\boldsymbol{b}_{j,l}} y_K[\boldsymbol{\delta}\mathbf{u}_j, \boldsymbol{\delta}\boldsymbol{v}_j]
\end{aligned}$$

. By assumption $\mathcal{L}''(y_K) \le M_J$, $\mathcal{L}'(y_K) \le L_J$ and $\|\boldsymbol{\Delta}\mathbf{U}_{1:K}\|_{\text{block}} \le 1, \|\boldsymbol{\Delta}\mathbf{V}_{1:K}\|_{\text{block}} \le 1$, $\|\boldsymbol{\delta}\mathbf{u}_{1:K}\|_\infty \le 1, \|\boldsymbol{\delta}\boldsymbol{v}_{1:K}\|_\infty \le 1$, we have

$$\begin{aligned}
\nabla^2_{\mathbf{W}_{1:K}} \mathcal{L}[\boldsymbol{\Delta}\mathbf{U}_{1:K}, \boldsymbol{\Delta}\mathbf{V}_{1:K}] &\lesssim M_J (\sum_{j=1}^{K} (2L_\sigma)^K C^{K+1})^2 \\
&\quad + L_J \sum_{1 \le j,l \le K} w^{\max\{0, 2/q-1/p\}} M_\sigma (2L_\sigma)^{2K-2} C^{2K+2} \\
&\lesssim_K w^{\max\{0, 2/q-1/p\}} (L_J + M_J) M_\sigma (2L_\sigma)^{2K} C^{2K+2} \\
\nabla^2_{\boldsymbol{b}_{1:K}} \mathcal{L}[\boldsymbol{\delta}\mathbf{u}_{1:K}, \boldsymbol{\delta}\boldsymbol{v}_{1:K}] &\lesssim M_J (\sum_{j=1}^{K} L_\sigma^K C^{K-1})^2 + L_J \sum_{1 \le j,l \le K} w^{\max\{0, 2/q-1/p\}} M_\sigma L_\sigma^{2K-2} C^{2K-2} \\
&\lesssim_K w^{\max\{0, 2/q-1/p\}} (L_J + M_J) M_\sigma L_\sigma^{2K} C^{2K-2}
\end{aligned}$$

Thus by Lemma 2 we have

$$\begin{aligned}
&\|\nabla \mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}^1_{1:K}) - \nabla \mathcal{L}_{\mathbf{W}_{1:K}}(\mathbf{W}^2_{1:K})\|_{\text{block},*} \lesssim w^{\max\{0,2/q-1/p\}} \|\mathbf{W}^1_{1:K} - \mathbf{W}^2_{1:K}\|_{\text{block}} \\
&\|\nabla_{\boldsymbol{b}_{1:K}} \mathcal{L}(\boldsymbol{b}^1_{1:K}) - \nabla_{\boldsymbol{b}_{1:K}} \mathcal{L}(\boldsymbol{b}^2_{1:K})\|_1 \lesssim w^{\max\{0,2/q-1/p\}} \|\boldsymbol{b}^1_{1:K} - \boldsymbol{b}^2_{1:K}\|_\infty .
\end{aligned}$$

As shown in the previous theorem, the L-smoothness exhibits a $d^{\max\{0,2/q-1/p\}}$ dependence on the network width d under the $(p, \mathtt{mean}) \to (q, \mathtt{mean})$ geometry. To achieve width-independent L-smoothness, the neural network should be considered in a $(p, \mathtt{mean}) \to (q, \mathtt{mean})$ geometry with $q > 2p$. Combined with the requirement that the steepest descent admits a closed-form update, our analysis focuses on the $(1, \mathtt{mean}) \to (p, \mathtt{mean})$ $(p \geq 2)$ and $(p, \mathtt{mean}) \to \ell_\infty$ geometries.

Message: Although the operator norms may play nicely together, allowing the M-Lipschitz constant to be well controlled, the L-smoothness coefficient can still exhibit diverse behavior:

- **`Muon`:** Under the $\|\cdot\|_{\mathrm{RMS}\to\mathrm{RMS}}$ geometry, the L-smoothness coefficient scales as $O(\sqrt{\text{width}})$.
- **Width-independent Smoothness is Possible:** The gradient of a neural network enjoys a width-independent L-smoothness coefficient in the $\|\cdot\|_{1,\mathtt{mean}\to(p,\mathtt{mean})}$ or $\|\cdot\|_{(p,\mathtt{mean})\to\ell_\infty}$ geometry. Under the $\|\cdot\|_{1,\mathtt{mean}\to\ell_\infty}$ geometry of `Adam`/`SignSGD`, the neural network enjoys a width-independent L-smoothness coefficient.

2.5. Good Norms, Bad Norms: Which Factors Matter?

In Theorem 5, both the M-Lipschitz and L-smoothness estimates rely on the assumption that the neural network admits a width-independent norm bound under the selected geometry. In this section, we demonstrate that this assumption imposes constraints on the network's approximation capacity. In supervised learning, let f^* denote the target function and $\hat{f}_{\mathcal{F}}$ the model learned by an algorithm. The total expected risk $\mathcal{R} := \mathbb{E}_{\mathcal{P}}[\ell(\hat{f}_{\mathcal{F}}(\boldsymbol{x}), \boldsymbol{y})] - \mathbb{E}_{\mathcal{P}}[\ell(f^*(\boldsymbol{x}), \boldsymbol{y})]$ can be decomposed as

$$\begin{aligned}
\mathcal{R} \leq\ & \underbrace{\inf_{f\in\mathcal{F}} \mathbb{E}_{\mathcal{P}}[\ell(f(\boldsymbol{x}), \boldsymbol{y})] - \mathbb{E}_{\mathcal{P}}[\ell(f^*(\boldsymbol{x}), \boldsymbol{y})]}_{\text{approximation error}} + \underbrace{\mathbb{E}_{\mathcal{P}_n}[\ell(\hat{f}_{\mathcal{F}}(\boldsymbol{x}), \boldsymbol{y})] - \mathbb{E}_{\mathcal{P}}[\ell(f^*_{\mathcal{F}}(\boldsymbol{x}), \boldsymbol{y})]}_{\text{optimization error}} \\
& + 2 \sup_{f\in\mathcal{F}} \underbrace{\left| \mathbb{E}_{\mathcal{P}_n}[\ell(f(\boldsymbol{x}), \boldsymbol{y})] - \mathbb{E}_{\mathcal{P}}[\ell(f(\boldsymbol{x}), \boldsymbol{y})] \right|}_{\text{generalization error}} .
\end{aligned} \tag{6}$$

Here $\mathbb{E}_{\mathcal{P}}[\cdot]$ denotes expectation with respect to the true (unknown) data distribution $\mathcal{P}$, $\mathbb{E}_{\mathcal{P}_n}[\cdot]$ denotes the empirical expectation over the n observed samples $\mathbb{E}_{\mathcal{P}_n}[g] = \frac{1}{n}\sum_{i=1}^n g(x_i, y_i)$, $\mathcal{F}$ is the hypothesis space, $f^*_{\mathcal{F}} = \arg\min_{f\in\mathcal{F}} \mathbb{E}_{\mathcal{P}}[\ell(f(\boldsymbol{x}),\boldsymbol{y})]$ is the best approximation to f^* inside $\mathcal{F}$, and $\hat{f}_{\mathcal{F}} = \arg\min_{f\in\mathcal{F}} \mathbb{E}_{\mathcal{P}_n}[\ell(f(\boldsymbol{x}),\boldsymbol{y})]$ denotes the empirical risk minimizer. In this decomposition, the **approximation error** quantifies the expressiveness of the function class $\mathcal{F}$, the **optimization error** measures the suboptimality of the learning algorithm, and the **generalization error** captures the gap between the performance of the training and the unseen data.

The norm selection influence all three components: they govern the complexity and smoothness of the hypothesis space in approximation, affect gradients, condition numbers, and convergence in optimization, and control the effective capacity of the model in generalization. In the following, we analyze the role of norms in each of these aspects.

- ***Optimization***: A stronger norm corresponds to larger norm values and thus a smaller unit ball. Under such a geometry, both the M-Lipschitz and L-smooth constants of the objective function tend to decrease. Intuitively, the smaller ball contracts the variations of gradients and function values, making the function appear smoother in this space. Consequently, optimization algorithms such as gradient descent can safely adopt a larger learning rate, leading to faster convergence.
- ***Approximation and Generalization***: All M-Lipschitz and L-smoothness bounds assume unit norm constraints. A stronger norm enables larger learning rates through improved smoothness, but at the cost of restricting the weights to a smaller feasible set, which reduces the representational capacity of the network. A norm constraint on the weights of a neural network effectively restricts the set of functions the network can represent. Intuitively, the stronger the norm constraint (i.e., the smaller the allowed norm), the smaller the "size" of the weight space, which limits the flexibility of the network. Tight constraints reduce the network's ability to approximate highly complex or rapidly varying functions, while looser constraints allow larger weights and a richer function space, improving approximation power. If $\|\cdot\|_{\mathsf{A}}$ is stronger than $\|\cdot\|_{\mathsf{B}}$, then $\{\mathbf{W} : \|\mathbf{W}\|_{\mathsf{A}} \le 1\} \subset \{\mathbf{W} : \|\mathbf{W}\|_{\mathsf{B}} \le 1\}$, i.e., a stronger norm induces a smaller feasible set (better smoothness control but less expressivity), while a weaker norm enlarges the feasible set (greater expressivity but potentially worse smoothness).

  At the same time, the norm constraint also affects generalization. By limiting the magnitude of weights, strong norm constraints prevent the network from fitting overly complex or noisy patterns in the training data, which reduces overfitting and improves generalization. Therefore, there is a trade-off: tighter norms can improve generalization but may hurt approximation power, while looser norms enhance approximation at the potential cost of overfitting.

We now examine the performance of different normalization schemes with respect to this approximation-optimization trade-off:

- **`Muon` under `RMS` → `RMS` Geometry.** Although `Muon` exhibits larger L-smoothness that scales as $O(\sqrt{\text{Width}})$, meaning its optimization landscape becomes increasingly rough as the network widens. However, its unit ball size remains $O(1)$ and does not shrink as the network becomes wider, maintaining a consistent representational capacity regardless of width.
- **Column normalization under $(1, \mathtt{mean}) \to (p, \mathtt{mean})$ Geometry.** $(1, \mathtt{mean}) \to (p, \mathtt{mean})$ geometry achieves $O(1)$ smoothness when $p \geq 2$. For the $\|\cdot\|_{(1,\mathtt{mean})\to\ell_p\text{-mean}}$ operator norm, the input ℓ_1-mean scales as $O(\text{Width}^{-1})$, while the output ℓ_p-mean scales as $O(\text{Width}^{-1/p})$. Since the operator norm measures the maximum amplification ratio between output and input norms, maintaining the operator norm requires scaling by the ratio $O(\text{Width}^{-1/p})/O(\text{Width}^{-1}) = O(\text{Width}^{(p-1)/p})$. Consequently, the unit ball size scales as $O(\text{Width}^{-(p-1)/p})$. This unit ball size $O(\text{Width}^{-(p-1)/p})$ shrinks more aggressively with width, particularly for larger p values, leading to more severe constraints on representational capacity.
- **Row normalization under $(p, \mathtt{mean}) \to \ell_\infty$ Geometry.** The $\|\cdot\|_{\ell_p\text{-mean}\to\ell_\infty}$ operator norm has unit ball size $O(\text{Width}^{-1/p})$ because the ℓ_p-mean input norm scales as $O(\text{Width}^{-1/p})$ while the ℓ_∞ output norm is $O(1)$. To maintain unit operator norm, the weight matrix must scale as $O(1)/O(\text{Width}^{-1/p}) = O(\text{Width}^{1/p})$. This implies that the unit ball diameter scales as $O(\text{Width}^{-1/p})$, which shrinks more slowly than column normalization's $O(\text{Width}^{-(p-1)/p})$ as width increases. This is crucial because when $p \geq 2$, we have $\frac{1}{p} \leq \frac{1}{2} < \frac{q-1}{q}$ for all $q \geq 2$, and moreover $\frac{q-1}{q} \to 1$ as $p \to \infty$ (which leads to `SignSgd/Adam`). Therefore, row normalization surprisingly preserves a larger unit ball—and thus greater approximation capacity—compared to column normalization for the same network width.

The row normalization family enjoys a better optimization-approximation trade-off because it simultaneously achieves $O(1)$ L-smoothness (ensuring efficient optimization) while maintaining a larger unit ball size that shrinks more slowly with width (preserving better approximation capacity). This favorable balance makes row normalization particularly attractive for wide networks. Therefore, we recommend steepest descent in the $\|\cdot\|_{\ell_p\text{-mean}\to\ell_\infty}$ norm for practical applications.

Message: Using a stronger norm constrains the weights to a smaller set, thereby limiting the class of functions that can be approximated. On the other hand, a stronger norm improves the smoothness of the optimization landscape by reducing the Lipschitz constant of the gradient. Consequently, there exists a trade-off between optimization and approximation.

	Row Normalization	Column Normalization	Muon
Norm	$\|\cdot\|_{(p,\texttt{mean})\to\ell_\infty}$	$\|\cdot\|_{(1,\texttt{mean})\to(q,\texttt{mean})}$	$\|\cdot\|_{\texttt{RMS}\to\texttt{RMS}}$
L-Smoothness	$O(1)$	$O(1)(\text{when } p \geq 2)$	$O(\sqrt{\text{Width}})$
Unit Ball Size	$O(\text{Width}^{1/p})$	$O(\text{Width}^{(p-1)/p})$	$O(1)$

While Muon faces larger L-smoothness of $O(\sqrt{\text{Width}})$, its unit ball size remains $O(1)$ and does not shrink as the network widens. In contrast, when $p \geq 2$, row normalization achieves both $O(1)$ smoothness and a unit ball size of $O(\text{Width}^{1/p})$, which shrinks more slowly than column normalization's $O(\text{Width}^{(q-1)/q})$ since $1/p \leq 1/2 < (q-1)/q$ for $p, q \geq 2$. This means row normalization preserves greater approximation capacity while maintaining efficient optimization, yielding a superior optimization-approximation trade-off. Therefore, we recommend steepest descent in the $\|\cdot\|_{\ell_p\text{-mean}\to\ell_\infty}$ norm for practical applications.

2.6. Adapting to Transformer To ground Algorithm 1 in a concrete Transformer architecture [45], we walk through the parameters of a standard Transformer block and specify, component by component, how each parameter should be normalized and parametrized under our framework:

Input Word Embedding and Positional Embeddings Input word embeddings convert each token into a continuous vector that represents its meaning, while positional embeddings encode the token's location in the sequence so the Transformer understands order. The input word embedding matrix has size $d_{\texttt{model}} \times \texttt{vocabsize}$ and the (absolute or relative) positional embedding matrix has size $d_{\texttt{model}} \times \texttt{contextsize}$ where vocabsize and contextsize is the fan_in and $d_{\texttt{model}}$ is the fan_out. [52] consider fan_in of the embedding and attention projections, controlled by the vocabsize and contextsize, is independent of model width for Adam. Consequently, the update magnitude is kept at a constant scale of 1 during training.

MOGA assigns the embedding layer a geometry consistent with its role as the linear operator that initializes token representations. In a Transformer, both token and positional embeddings are simply matrix operators that act on one-hot basis vectors, *i.e.* $x_i = W_{\texttt{tok}} e_i$ and $p_j = W_{\texttt{pos}} e_j$, mapping discrete indices into $d_{\texttt{model}}$-dimensional continuous representations. Since one-hot vectors lie in the ℓ_1 unit ball, and our previous analysis models hidden features in the $(p, \texttt{mean})$ geometry, the embedding layer naturally corresponds to an $\ell_1 \to (p, \texttt{mean})$ operator. Under this geometry, we find that the following two optimizers perform particularly well.

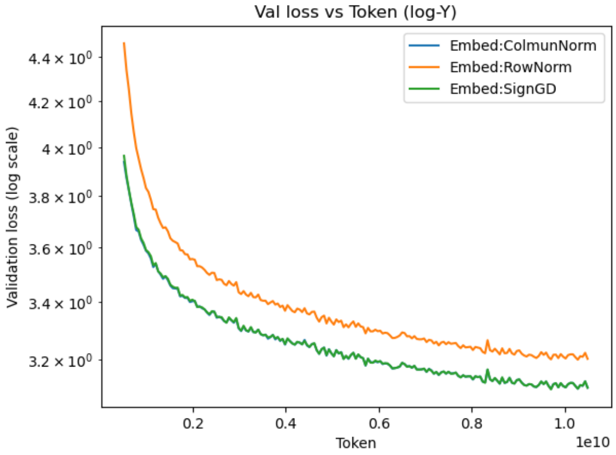

FIGURE 3. **Embedding layer geometry.** One-hot inputs place embedding training in the $\ell_1! \to !(q, \texttt{mean})$ regime, where both column normalization and `SignGD` are effective.

- **SignSGD**. Our first choice is to model the embedding layer under the $1 \to \infty$ operator geometry, which naturally reduces to applying SIGNSGD (Proposition 1) on the embedding parameters.
- **Scaled Column Normalization** Our second choice is to model the embedding layer under the $1 \to (p, \texttt{mean})$ operator geometry, where the corresponding steepest–descent update becomes $\texttt{fan\_out}^{1/p}\, \texttt{colnorm}_p(\cdot)$. The $\texttt{fan\_out}^{1/p}$ scaling arises because the $(p, \texttt{mean})$ geometry corresponds to a rescaled ℓ_p-norm, defined by $\|x\|_{(p,\texttt{mean})} = d^{-1/p}\|x\|_p$ for vectors in $\mathbb{R}^d$, where $d = \texttt{fan\_out}$ for the embedding layer since each embedding vector lives in a `fan_out`-dimensional output space.

One may also view a one-hot vector as a unit vector in ℓ_p for any p. However, our experiments show that modeling the embedding layer under the $p \to \infty$ geometry—corresponding to a $\texttt{rownorm}_p(\cdot)$ update—does not perform well in practice. We hypothesize that the ℓ_1 norm provides the tightest and most faithful characterization of one-hot vectors, making the $1 \to (p, \texttt{mean})$ geometry more appropriate for embedding layers.

Bias Since the bias directly perturbs the feature, whose representation lives in the $(p, \texttt{mean})$ geometry, we require the bias step to remain safely within this constraint set. Although the LayerNorm bias can, in principle, be updated under the $(q, \texttt{mean})$ geometry (for column normalization), we use the ℓ_∞–norm ball for simplicity. Since the ℓ_∞ ball is smaller (and thus more conservative) than the $(q, \texttt{mean})$ ball, the update collapses to taking only the sign of the gradient—effectively `signSGD` without any scaling on LayerNorm parameters. In short, all one-dimensional parameters (LayerNorm weight and bias) end up using a pure `SignSGD` update. All one-dimensional parameters (LayerNorm weight, LayerNorm bias, and MLP bias) end up using `signSGD`-style updates. This choice aligns with the scaling rule adopted by `Adam` under the μP framework [52].

Layernorm Weights and Biases Layernorm weights w^{LN} and biases b^{LN} both are vectors of shape $\mathbb{R}^{d_{\texttt{model}}}$. Since the LayerNorm weight is a diagonal matrix, in the $((q, \texttt{mean}) \to (q, \texttt{mean}))$ geometry (for row normalization $q = \infty$) its update simplifies to only keeping the sign of the gradient. In other words, LayerNorm parameters behave like `SignSGD` but without applying any scaling. In line with the treatment of other bias parameters, we update the LayerNorm bias in the ℓ_∞ geometry. This results in a `signSGD`-type update with no scaling applied to the LayerNorm bias. This scaling matches the `Adam` parametrization recommended by the μP theory [52].

Self-Attention In a Transformer MLP block, there are four weight matrices $W^q, W^k, W^v \in \mathbb{R}^{(d_v n_{\texttt{head}}) \times d_{\texttt{model}}}, W^o \in \mathbb{R}^{d_{\texttt{model}} \times (d_v n_{\texttt{head}})}$,. In standard multi-head self-attention [45], the input $X \in \mathbb{R}^{\texttt{contextsize} \times d_{\texttt{model}}}$ is projected into queries, keys, and values by $Q = XW^q,\ K = XW^k,\ V = XW^v$ with $W^q, W^k, W^v \in \mathbb{R}^{(d_v n_{\texttt{head}}) \times d_{\texttt{model}}}$; As discussed above, we train the query, key, and value projection matrices W^q, W^k, W^v under the $(1, \texttt{mean}) \to (p, \texttt{mean})$ geometry or the $(p, \texttt{mean}) \to \ell_\infty$ geometry.

While performing multi-head attnetion [45], we separate query Q, key K and value V to $n_{\texttt{head}}$ heads, *i.e.* reshape to a $(n_{\texttt{head}}, \texttt{contextsize}, d_v)$ tensor and each head performs

$$\mathrm{head}_i = \mathrm{softmax}\left(\frac{Q_i K_i^\top}{\sqrt{d_v}}\right) V_i,$$

and the final output is obtained by concatenating all heads and applying the output projection

$$\mathrm{MultiHead}(X) = \mathrm{concat}(\mathrm{head}_1, \ldots, \mathrm{head}_{n_{\texttt{head}}})\, W^o, \qquad W^o \in \mathbb{R}^{d_{\texttt{model}} \times (d_v n_{\texttt{head}})}.$$

However in μP-parametrization [52], authors use $\frac{1}{d}$ normalization rather than the $\frac{1}{\sqrt{d}}$ normalization, *i.e.* $\mathrm{head}_i = \mathrm{softmax}\big(Q_i K_i^\top / d_v\big)\, V_i$. We now give an alternative explanation from the perspective of M-Lipschitzness and L-smoothness. To ensure that the softmax operator remains L-smooth, the magnitude of the attention logits must be controlled so that it does not scale with the network width. Lemma 4 shows that when both the query and key vectors are $(p, \texttt{mean})$ vectors, the growth of the attention logit can be bounded by $q^\top k = O\Big(d^{\max\{1, 2/p\}}\Big)$. Thus, under **row normalization** (i.e., the geometry $(p, \texttt{mean}) \to \infty$), every query and key becomes an ℓ_∞–bounded vector. The inner product $q^\top k$ therefore scales at most like d, and we apply a normalization factor of $1/d$. In contrast, under **column normalization** (i.e., the geometry $(1, \texttt{mean}) \to (p, \texttt{mean})$), every query and key becomes a $(p, \mathtt{mean})$–bounded vector. By Lemma 4, the inner product $q^\top k$ scales as $d^{\max\{1, 2/p\}}$ and therefore we apply a normalization factor of $1/d^{\max\{1, 2/p\}}$.

- For row normalization (i.e., the geometry $(p, \mathtt{mean}) \to \infty$), we use $1/d$ normalization, *i.e.* $\text{head}_i = \text{softmax}\left(\frac{Q_i K_i^\top}{d_v}\right) V_i$.
- For column normalization (i.e., the geometry $(1, \mathtt{mean}) \to (p, \mathtt{mean})$), we use $1/d^{\max\{1, 2/p\}}$ normalization, *i.e.* $\text{head}_i = \text{softmax}\left(\frac{Q_i K_i^\top}{d_v^{\max\{1, 2/p\}}}\right) V_i$.
- For `Adam/SignSgd` case, we recover the normalization in μP parameterization [52].

LEMMA 4 **(Inner product bound under mean norms).** *Let $p, q \in [1, \infty]$ and $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$. Then*

$$|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leq d^{\max\{1, 1/p+1/q\}} \, \|\boldsymbol{x}\|_{(p, mean)} \, \|\boldsymbol{y}\|_{(q, mean)}. \tag{7}$$

Moreover, the exponent on d is optimal in general (sharp up to a universal constant).

Proof of Lemma 4. Let p' be the conjugate of p (i.e., $1/p + 1/p' = 1$, with the usual conventions). By Hölder on this probability space, $\frac{1}{d} |\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leq \|\boldsymbol{x}\|_{(p, \mathtt{mean})} \|\boldsymbol{y}\|_{(p', \mathtt{mean})}$. We now compare mean norms. Since we have $\|z\|_{(p', \mathtt{mean})} \leq d^{\left(\frac{1}{q} - \frac{1}{p'}\right)_+} \|z\|_{(q, \mathtt{mean})}$, combining with Hölder's Inequality, we have

$$|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leq d^{1+\left(\frac{1}{q} - \frac{1}{p'}\right)_+} \|\boldsymbol{x}\|_{(p, \mathtt{mean})} \|\boldsymbol{y}\|_{(q, \mathtt{mean})}.$$

Since $1/p' = 1 - 1/p$, we have $1 + \left(\frac{1}{q} - \frac{1}{p'}\right)_+ = \max\left\{1, \frac{1}{p} + \frac{1}{q}\right\}$, which yields (7).

MLP In a Transformer MLP block, there are two weight matrices $W^1 \in \mathbb{R}^{d_{\mathtt{ffn}} \times d_{\mathtt{model}}}, W^2 \in \mathbb{R}^{d_{\mathtt{model}} \times d_{\mathtt{ffn}}}$, where $d_{\mathtt{ffn}}$ is typically set to $4d_{\mathtt{model}}$. For W^1, we set `fan_in` to $d_{\mathtt{model}}$ and `fan_out` to $d_{\mathtt{ffn}}$. For W^2, we set `fan_in` to $d_{\mathtt{ffn}}$ and `fan_out` to $d_{\mathtt{model}}$. In line with the treatment of other bias parameters, we update the MLP bias in the ℓ_∞ geometry. This results in a `signSGD`-type update with no scaling applied to the MLP bias.

Word Unembeddings Following the μP scaling rule [52], we note that the word *unembedding* matrix is always tied to the word *embedding* matrix. Consequently, we scale the unembedding parameters in exactly the same way as the embedding parameters—namely, we do not scale down the learning rate for

3. Experiments

We evaluate the `MOGA` family of optimizers for the task of LLM pre-training. The main experimental configurations are summarized below.

Large Token Budget Experiment. For large-token-budget pretraining, we use approximately $\sim 8\times$ the Chinchilla-optimal number of training tokens. Details are as follows:

- **Models [37] .** We conduct our experiments using GPT-2 small architectures, which contains approximately **124 million** parameters, with 12 Transformer blocks, each configured with a hidden size of 768 and 12 attention heads. We use the standard GPT-2 tokenizer with a vocabulary of 50,257 tokens and are trained with a context window of 1,024 tokens. These configurations follow the canonical GPT-2 design choices while offering substantial model capacity for large-scale language modeling experiments.
- **Datasets.** We conduct experiments on the Openwebtext [18] dataset, which contains approximately 8.5 billion tokens. OpenWebText is a community-created replication of the web text portion of OpenAI's GPT-2 training data and is widely used in language modeling benchmarks.
- **LR schedulers.** We adopt a learning rate schedule consisting of a **linear warm-up of 2.5% of the total steps** followed by a **cosine decay**. The learning rate increases linearly from zero to the peak value `lr_max`, after which it follows a cosine annealing schedule down to the terminal learning rate `lr_min` = `lr_max`/10.
- **Training configurations.** All experiments are conducted on four NVIDIA H100 GPUs. We use a micro-batch size of 64 per GPU and apply gradient accumulation with a factor of 4. During training, the context length (block size) is set to 512 tokens. Consequently, each optimization step processes a total of $4 \times 64 \times 4 \times 512 = 524,288$ tokens, which is consistent with the widely adopted configuration for GPT-2 small training.

Standard Token Budget Experiment. We follow [47] and pretrain with the $1\times$ Chinchilla-optimal token budget. Details are as follows:

- **Models.** We conduct our experiments on the LLaMA series (130M, 350M). LLaMA-130M has 12 Transformer blocks, each configured with a hidden size of 768 and 12 attention heads. LLaMA-350M has 24 Transformer blocks, each configured with a hidden size of 1024 and 16 attention heads. The tokenizer is configured with a vocabulary size of 32,000 and a maximum sequence length of 1024.
- **Datasets.** we pretrain the LLaMA series models from scratch on the C4 dataset [**?**]. C4 dataset is a colossal, cleaned version of Common Crawl's web crawl corpus, which is mainly intended to pre-train language models and word representations.
- **LR schedulers.** We adopt a learning rate schedule consisting of a **linear warm-up of 10% of the total steps** followed by a **cosine decay**. The learning rate increases linearly from zero

to the peak value `lr_max`, after which it follows a cosine annealing schedule down to the terminal learning rate `lr_min` = `lr_max`/10.

- **Training configurations.** All experiments are conducted on four NVIDIA H100 GPUs. The batch size is set to 512. During training, the context length (block size) is set to 256 tokens. Consequently, each optimization step processes a total of $512 \times 256 = 131,072$ tokens.

`AdamW` and `Muon` Baselines. We evaluate our method against `AdamW` and `Muon`, which serve as strong baseline optimizers in all experiments. For `AdamW`, we use the standard configuration with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay $\lambda = 0.1$. For `Muon`, we adopt settings from [29], with momentum parameter $\beta = 0.95$ and weight decay $\lambda = 0.1$. For each optimizer, we tune the peak learning rate `lr_max` via a logarithmic sweep during the first training epoch. Specifically, we search over `lr_max` $\in [8 \times 10^{-5}, 8 \times 10^{-4}]$ for `AdamW` and `lr_max` $\in [4 \times 10^{-4}, 4 \times 10^{-3}]$ for `Muon`. All baseline models are subsequently trained using the optimizer-specific `lr_max` that achieves the best validation performance.

Learning Rate Transfer Following the zero-shot hyperparameter transfer framework of [52], we study whether the optimal learning rate remains invariant across model widths. Empirically, we train architectures with widths ranging from small to large, while keeping all other hyperparameters fixed. For each width, we sweep the maximal learning rate `lr_max` and record the value that produces the best validation performance during the early training. We extend the μP scaling rule to a broader family of optimizers, enabling consistent learning-rate transfer across model widths. Models with radically different parameter counts converge fastest when using nearly the same `lr_max`. This supports the idea that, when the parameterization follows a width-consistent scaling rule, the optimization geometry of the loss landscape remains stable as width grows. As a consequence, learning rates tuned on a small model can be directly transferred to large models without additional search. This property significantly reduces the cost of hyperparameter tuning and enables efficient scaling experiments.

Performance on GPT-2 Small/Large/XL

4. Discussion and Conclusion

4.1. Related Work

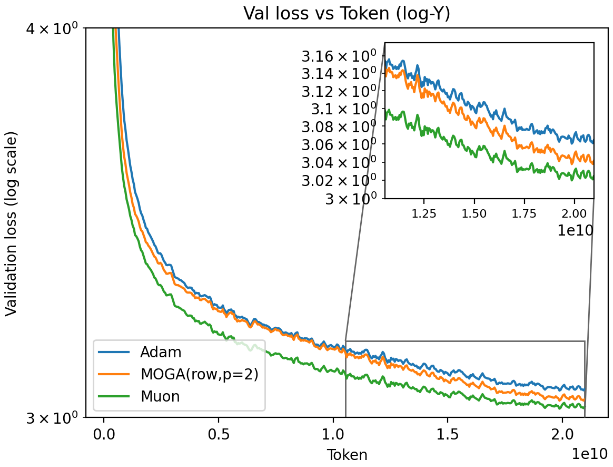

FIGURE 4. Comparison between Adam, MUON and MOGA.

Modern Language Model Optimizer Modern large-scale neural networks and language models commonly rely on variants and approximations of the AdaGrad algorithm [14, 49]. These methods typically employ diagonal preconditioning, as exemplified by Adam [23, 30, 38], or block-wise approximations such as Adafactor [42], LAMB [54], and Adam-Mini [55]. More sophisticated approaches use structured tensor-based preconditioning, including Shampoo [19] and SOAP [46]. Another line of work treats the Gauss-Newton method as the gold standard for optimization. For instance, [33, 36, 39] approximate the Fisher information matrix to implement natural gradient methods, while [27, 46] employ SOAP-like methodologies for similar approximations. More recently, [1, 50] introduced ASGO (One-Sided Shampoo), which only applies a one-sided preconditioner within the Shampoo.

Gradient Normalization Starting from SignSGD [7], researchers began to connect gradient normalization with steepest descent under the corresponding norms. Recently, Muon [21] formulates optimization as steepest descent under the spectral norm, which naturally leads to whitening the gradient matrix. [17] investigates steepest descent under the $\ell_1 \to \ell_2$ norm, which results in column normalization of the gradient. The most closely related works are [31, 41], both of which employ row normalization to normalize and whiten the gradient. Specifically, [31] combines row normalization with the matrix sign function to normalize and whiten the data, while [41] applies the Sinkhorn algorithm to simultaneously normalize the gradient row-wise and column-wise. Different from [31, 41], we consider $\mathrm{RMS} \to \ell_\infty$ norm, thus we need to further scale the $\frac{1}{\sqrt{\mathtt{fan\_in}}}$. [55] proposed adam-mini, which carefully partitions the parameters into blocks based on our new principle of Hessian structure, and assigns a single, well-chosen learning rate to each block.

4.2. Discussion

Conclusion Do Transformers at different widths share the same optimal learning rate for a given optimizer? And does an algorithm that performs well on small models remain effective as the model scales up? In this paper, we study the width-scaling behavior of neural optimizers through the lens of matrix–operator–norm steepest descent, aiming to understand how optimization geometry determines scalable learning-rate rules. We understand many popular optimizer, including AdamW, Muon, Lion and row/column normalization, can be unified as instances of steepest descent under different matrix operator norms. However, a neural network is not dimension-independently Lipschitz under arbitrary matrix operator norms. In particular, classical $p \to q$ operator norms fail to propagate stability across layers: their Lipschitz estimates do not compose cleanly, causing width-dependent distortions in the overall Lipschitz constant and leading to unstable learning-rate scaling.

To resolve this issue, we introduce the **mean operator norm** $(p, \texttt{mean}) \to (q, \texttt{mean})$, which rescales the geometry to match neural-network width. This makes operator norms *play nicely together*: the network forward map becomes M-Lipschitz with no width dependence, enabling stable layerwise composition. We further analyze gradient sensitivity and prove that under this geometry, the objective becomes L-smooth with *width-independent optimal learning rates*. Our theory identifies two width-independent geometries:

$$(1, \texttt{mean}) \to (p, \texttt{mean}) \quad (p \geq 2), \qquad \text{and} \qquad (p, \texttt{mean}) \to \infty,$$

and shows that the latter enlarges the feasible approximation set, yielding better optimization–approximation trade-offs. This motivates a new optimizer: **row-normalized steepest descent**, which applies a power transform followed by per-row normalization.

Empirically, the proposed MOGA optimizer exhibits better width-scaling behavior than AdamW, validating our theoretical predictions without requiring any learning-rate retuning across widths in GPT pretraining. Moreover, MOGA attains the same token-wise convergence rate as MUON while avoiding any spectral computations. Overall, our results yield a geometry-aware principle for constructing neural optimizers that remain stable under width scaling and preserve consistent learning dynamics as model size grows.

Acknowledgments

We thank Jose Blanchet, Jorge Nocedal, Lexing Ying, Weijie Su, Fanghui Liu, Lei Wu, Yinuo Ren and Yinyu Ye for their valuable feedback.

References

[1] An K, Liu Y, Pan R, Ren Y, Ma S, Goldfarb D, Zhang T (2025) Asgo: Adaptive structured gradient optimization. *arXiv preprint arXiv:2503.20762* .

[2] Bahri Y, Dyer E, Kaplan J, Lee J, Sharma U (2024) Explaining neural scaling laws. *Proceedings of the National Academy of Sciences* 121(27):e2311878121.

[3] Balles L, Hennig P (2018) Dissecting adam: The sign, magnitude and variance of stochastic gradients. *International Conference on Machine Learning*, 404–413 (PMLR).

[4] Bernstein J, Newhouse L (2024) Modular duality in deep learning. *arXiv preprint arXiv:2410.21265* .

[5] Bernstein J, Newhouse L (2024) Old optimizer, new norm: An anthology. *arXiv preprint arXiv:2409.20325* .

[6] Bernstein J, Vahdat A, Yue Y, Liu MY (2020) On the distance between two neural networks and the stability of learning. *Advances in Neural Information Processing Systems* 33:21370–21381.

[7] Bernstein J, Wang YX, Azizzadenesheli K, Anandkumar A (2018) signsgd: Compressed optimisation for non-convex problems. *International conference on machine learning*, 560–569 (PMLR).

[8] Bottou L, Curtis FE, Nocedal J (2018) Optimization methods for large-scale machine learning. *SIAM review* 60(2):223–311.

[9] Carlson DE, Collins E, Hsieh YP, Carin L, Cevher V (2015) Preconditioned spectral descent for deep learning. *Advances in neural information processing systems* 28.

[10] Chen L, Li J, Liu Q (2025) Muon optimizes under spectral norm constraints. *arXiv preprint arXiv:2506.15054* .

[11] Chen L, Liu B, Liang K, Liu Q (2023) Lion secretly solves constrained optimization: As lyapunov predicts. *arXiv preprint arXiv:2310.05898* .

[12] Chen X, Liang C, Huang D, Real E, Wang K, Pham H, Dong X, Luong T, Hsieh CJ, Lu Y, et al. (2023) Symbolic discovery of optimization algorithms. *Advances in neural information processing systems* 36:49205–49233.

[13] Cortes C, Jackel LD, Solla S, Vapnik V, Denker J (1993) Learning curves: Asymptotic values and rate of convergence. *Advances in neural information processing systems* 6.

[14] Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12(7).

[15] Flynn T (2017) The duality structure gradient descent algorithm: analysis and applications to neural networks. *arXiv preprint arXiv:1708.00523* .

[16] Ghosh N, Wu D, Bietti A (2025) Understanding the mechanisms of fast hyperparameter transfer. *arXiv preprint arXiv:2512.22768* .

[17] Glentis A, Li J, Han A, Hong M (2025) A minimalist optimizer design for llm pretraining. *arXiv preprint arXiv:2506.16659* .

[18] Gokaslan A, Cohen V, Pavlick E, Tellex S (2019) Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

[19] Gupta V, Koren T, Singer Y (2018) Shampoo: Preconditioned stochastic tensor optimization. *International Conference on Machine Learning*, 1842–1850 (PMLR).

[20] Hestness J, Narang S, Ardalani N, Diamos G, Jun H, Kianinejad H, Patwary MMA, Yang Y, Zhou Y (2017) Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409* .

[21] Jordan K, Jin Y, Boza V, You J, Cesista F, Newhouse L, Bernstein J (2024) Muon: An optimizer for hidden layers in neural networks. *Cited on* 10.

[22] Kaplan J, McCandlish S, Henighan T, Brown TB, Chess B, Child R, Gray S, Radford A, Wu J, Amodei D (2020) Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361* .

[23] Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

[24] Kovalev D (2025) Understanding gradient orthogonalization for deep learning via non-euclidean trust-region optimization. *arXiv preprint arXiv:2503.12645* .

[25] Large T, Liu Y, Huh M, Bahng H, Isola P, Bernstein J (2024) Scalable optimization in the modular norm. *Advances in Neural Information Processing Systems* 37:73501–73548.

[26] Li J, Hong M (2025) A note on the convergence of muon. *arXiv preprint arXiv:2502.02900* .

[27] Lin W, Dangel F, Eschenhagen R, Bae J, Turner RE, Makhzani A (2024) Can we remove the square-root in adaptive gradient methods? a second-order perspective. *arXiv preprint arXiv:2402.03496* .

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

[29] Liu Y, Yuan A, Gu Q (2025) Mars-m: When variance reduction meets matrices. *arXiv preprint arXiv:2510.21800* .

[30] Loshchilov I, Hutter F (2017) Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* .

[31] Ma C, Gong W, Scetbon M, Meeds E (2024) Swan: Sgd with normalization and whitening enables stateless llm training. *arXiv preprint arXiv:2412.13148* .

[32] Maddison CJ, Paulin D, Teh YW, Doucet A (2021) Dual space preconditioning for gradient descent. *SIAM Journal on Optimization* 31(1):991–1016.

[33] Martens J, Grosse R (2015) Optimizing neural networks with kronecker-factored approximate curvature. *International conference on machine learning*, 2408–2417 (PMLR).

[34] Nocedal J, Wright SJ (2006) *Numerical optimization* (Springer).

[35] Pethick T, Xie W, Antonakopoulos K, Zhu Z, Silveti-Falls A, Cevher V (2025) Training deep learning models with norm-constrained lmos. *arXiv preprint arXiv:2502.07529* .

[36] Pooladzandi O, Li XL (2024) Curvature-informed sgd via general purpose lie-group preconditioners. *arXiv preprint arXiv:2402.04553* .

[37] Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I, et al. (2019) Language models are unsupervised multitask learners. *OpenAI blog* 1(8):9.

[38] Reddi SJ, Kale S, Kumar S (2019) On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237* .

[39] Ren Y, Goldfarb D (2021) Tensor normal training for deep learning models. *Advances in Neural Information Processing Systems* 34:26040–26052.

[40] Riabinin A, Shulgin E, Gruntkowska K, Richtárik P (2025) Gluon: Making muon & scion great again!(bridging theory and practice of lmo-based optimizers for llms). *arXiv preprint arXiv:2505.13416* .

[41] Scetbon M, Ma C, Gong W, Meeds E (2025) Gradient multi-normalization for stateless and scalable llm training. *arXiv preprint arXiv:2502.06742* .

[42] Shazeer N, Stern M (2018) Adafactor: Adaptive learning rates with sublinear memory cost. *International Conference on Machine Learning*, 4596–4604 (PMLR).

[43] Shen W, Huang R, Huang M, Shen C, Zhang J (2025) On the convergence analysis of muon. *arXiv preprint arXiv:2505.23737* .

[44] Tuddenham M, Prügel-Bennett A, Hare J (2022) Orthogonalising gradients to speed up neural network optimisation. *arXiv preprint arXiv:2202.07052* .

[45] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. *Advances in neural information processing systems* 30.

[46] Vyas N, Morwani D, Zhao R, Kwun M, Shapira I, Brandfonbrener D, Janson L, Kakade S (2024) Soap: Improving and stabilizing shampoo using adam. *arXiv preprint arXiv:2409.11321* .

[47] Wang J, Zhou P, Dong Y, Li H, Li J, Zhou X, Lao Q, Fang C, Lin Z (2025) Conda: Column-normalized adam for training large language models faster. *arXiv preprint arXiv:2509.24218* .

[48] Wang M, Wang J, Zhang J, Wang W, Pei P, Cai X, Wu L, et al. (2025) Gradpower: Powering gradients for faster language model pre-training. *arXiv preprint arXiv:2505.24275* .

[49] Ward R, Wu X, Bottou L (2020) Adagrad stepsizes: Sharp convergence over nonconvex landscapes. *Journal of Machine Learning Research* 21(219):1–30.

[50] Xie S, Wang T, Reddi S, Kumar S, Li Z (2025) Structured preconditioners in adaptive optimization: A unified analysis. *arXiv preprint arXiv:2503.10537* .

[51] Yang G, Hu EJ (2021) Tensor programs iv: Feature learning in infinite-width neural networks. *International Conference on Machine Learning*, 11727–11737 (PMLR).

[52] Yang G, Hu EJ, Babuschkin I, Sidor S, Liu X, Farhi D, Ryder N, Pachocki J, Chen W, Gao J (2022) Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer. *arXiv preprint arXiv:2203.03466* .

[53] Yang G, Simon JB, Bernstein J (2023) A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813* .

[54] You Y, Li J, Reddi S, Hseu J, Kumar S, Bhojanapalli S, Song X, Demmel J, Keutzer K, Hsieh CJ (2019) Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962* .

[55] Zhang Y, Chen C, Li Z, Ding T, Wu C, Kingma DP, Ye Y, Luo ZQ, Sun R (2024) Adam-mini: Use fewer learning rates to gain more. *arXiv preprint arXiv:2406.16793* .