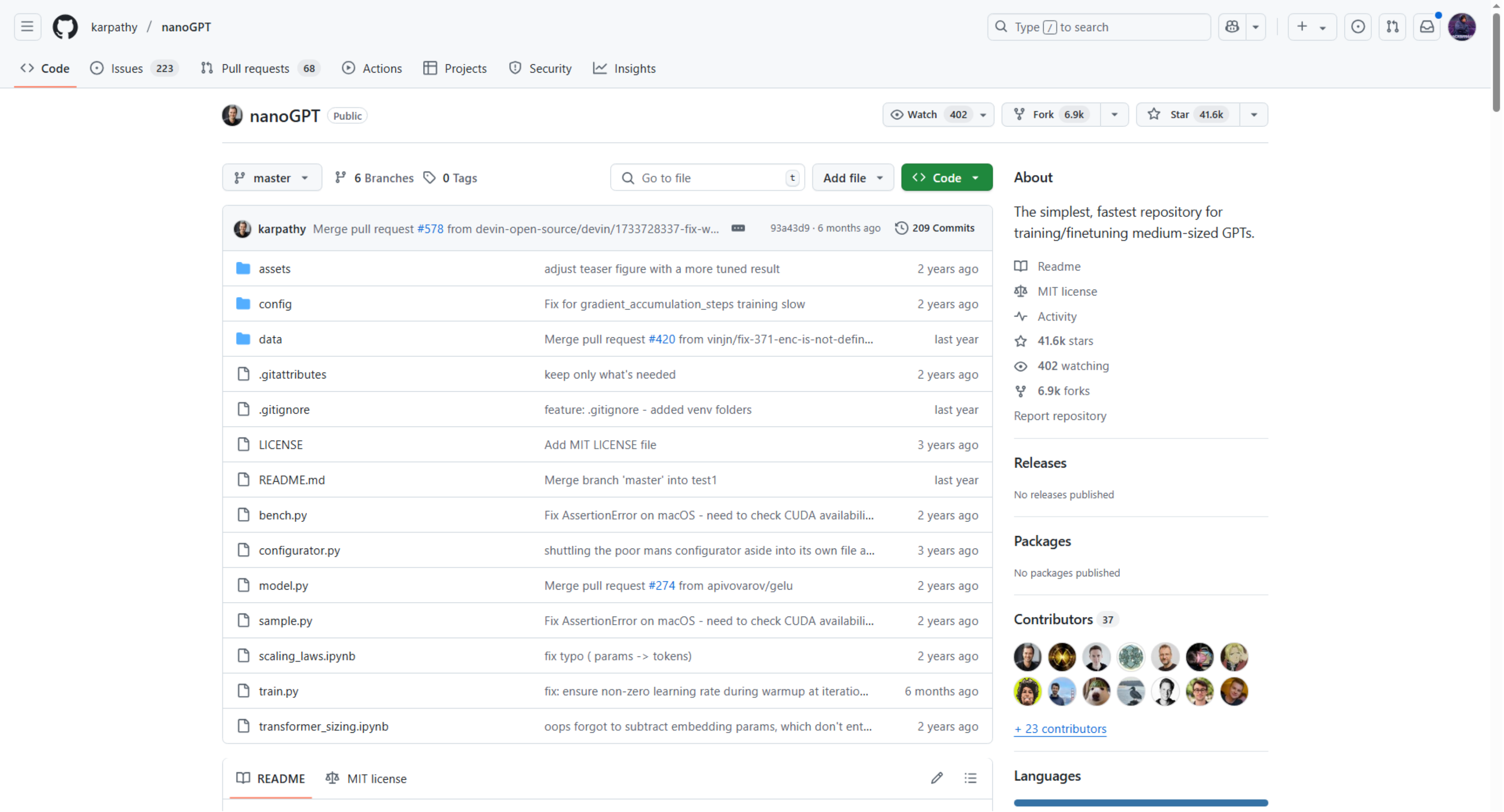
Modular Optimization
The Great Mind of Jeremy Bernstein

Donghu Kim

NanoGPT Speedrun

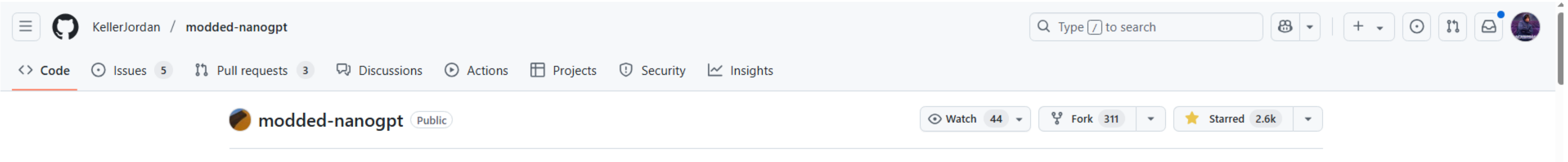
NanoGPT (124M) by Andrej Karpathy

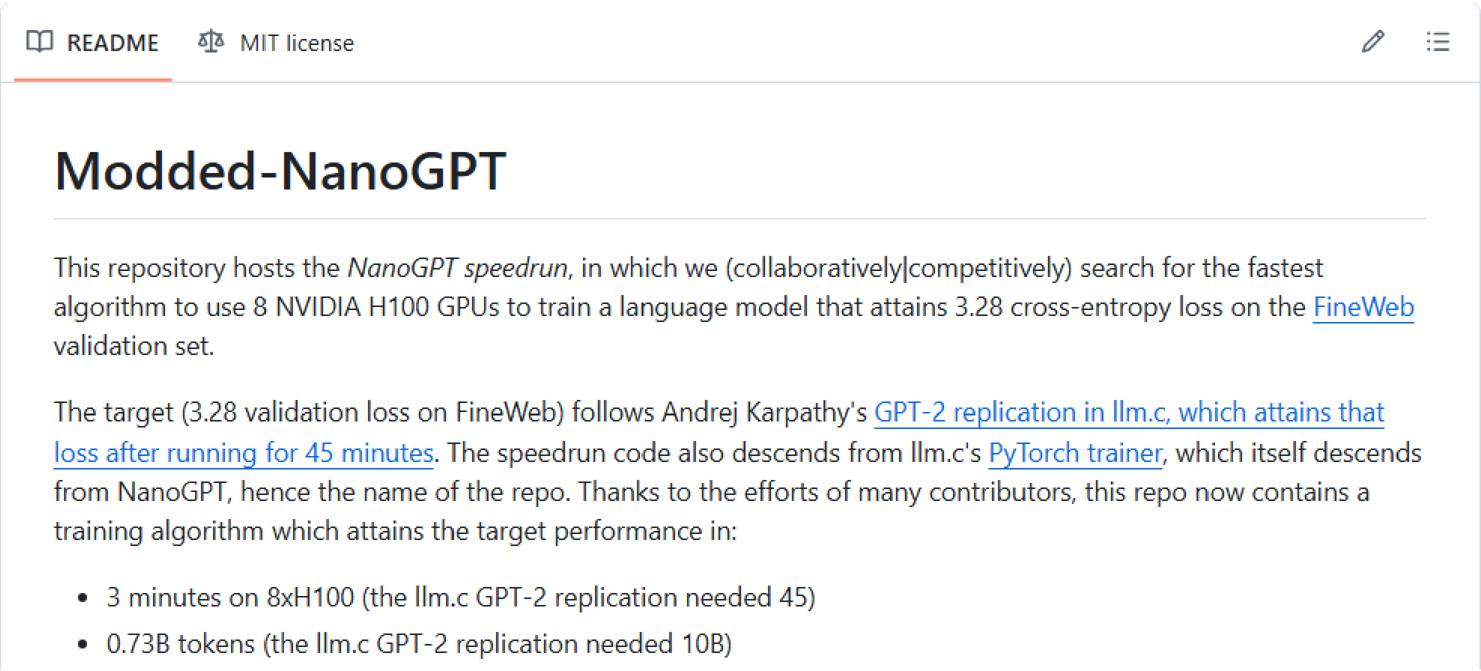


https://github.com/karpathy/nanoGPT

NanoGPT Speedrun

A man simply must go *fast*





README MIT license

Modded-NanoGPT

This repository hosts the *NanoGPT speedrun*, in which we (collaboratively|competitively) search for the fastest algorithm to use 8 NVIDIA H100 GPUs to train a language model that attains 3.28 cross-entropy loss on the FineWeb validation set.

The target (3.28 validation loss on FineWeb) follows Andrej Karpathy's GPT-2 replication in llm.c, which attains that loss after running for 45 minutes. The speedrun code also descends from llm.c's PyTorch trainer, which itself descends from NanoGPT, hence the name of the repo. Thanks to the efforts of many contributors, this repo now contains a training algorithm which attains the target performance in:

- 3 minutes on 8xH100 (the llm.c GPT-2 replication needed 45)
- 0.73B tokens (the llm.c GPT-2 replication needed 10B)

*But also for a reason: [1]

NanoGPT Speedrun

A man simply must go *fast*

#	Record time	Description	Date	Log	Contributors
1	45 minutes	llm.c baseline	05/28/24	log	@karpathy, llm.c contributors
2	31.4 minutes	Tuned learning rate & rotary embeddings	06/06/24	log	@kellerjordan0
⋮					
22	2.990 minutes	Faster gradient all-reduce	05/24/25	log	@KonstantinWilleke, @alexrgilbert, @adricarda, @tuttyfrutyee, @vdlad; The Enigma project
23	2.979 minutes	Overlap computation and gradient communication	05/25/25	log	@ryanyang0

https://github.com/KellerJordan/modded-nanogpt

NanoGPT Speedrun

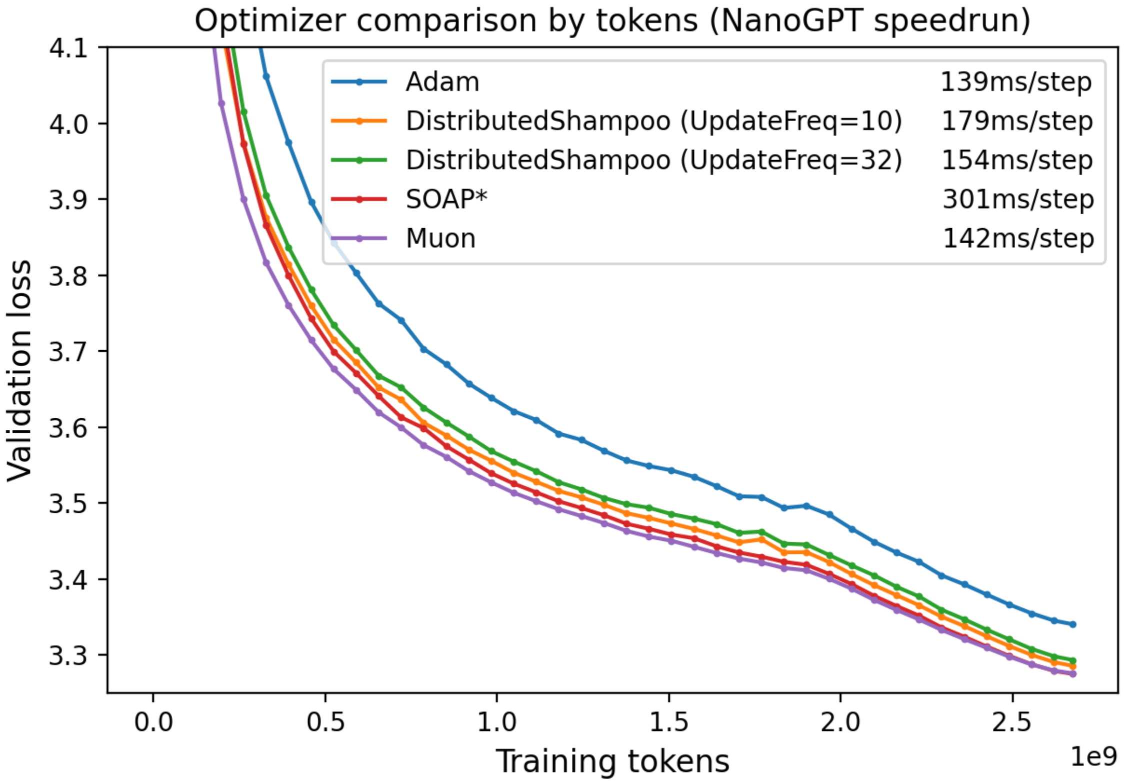
What's that?

#	Record time	Description	Date	Log	Contributors
1	45 minutes	llm.c baseline	05/28/24	log	@karpathy, llm.c contributors
2	31.4 minutes	Tuned learning rate & rotary embeddings	06/06/24	log	@kellerjordan0
3	24.9 minutes	Introduced the Muon optimizer	10/04/24	none	@kellerjordan0, @jxbz
4	22.3 minutes	Muon improvements	10/11/24	log	@kellerjordan0, @bozavlado
5	15.2 minutes	Pad embeddings, ReLU^2, zero-init projections, QK-norm	10/14/24	log	@Grad62304977, @kellerjordan0
6	13.1 minutes	Distributed the overhead of Muon	10/18/24	log	@kellerjordan0

~11min speedup

Faster Optimization

More sample-efficient than Adam but at the same speed?



https://kellerjordan.github.io/posts/muon/#results

Hyperparameter Transfer

Somehow makes scaling easier?

(Cousin of Muon) (Muon)

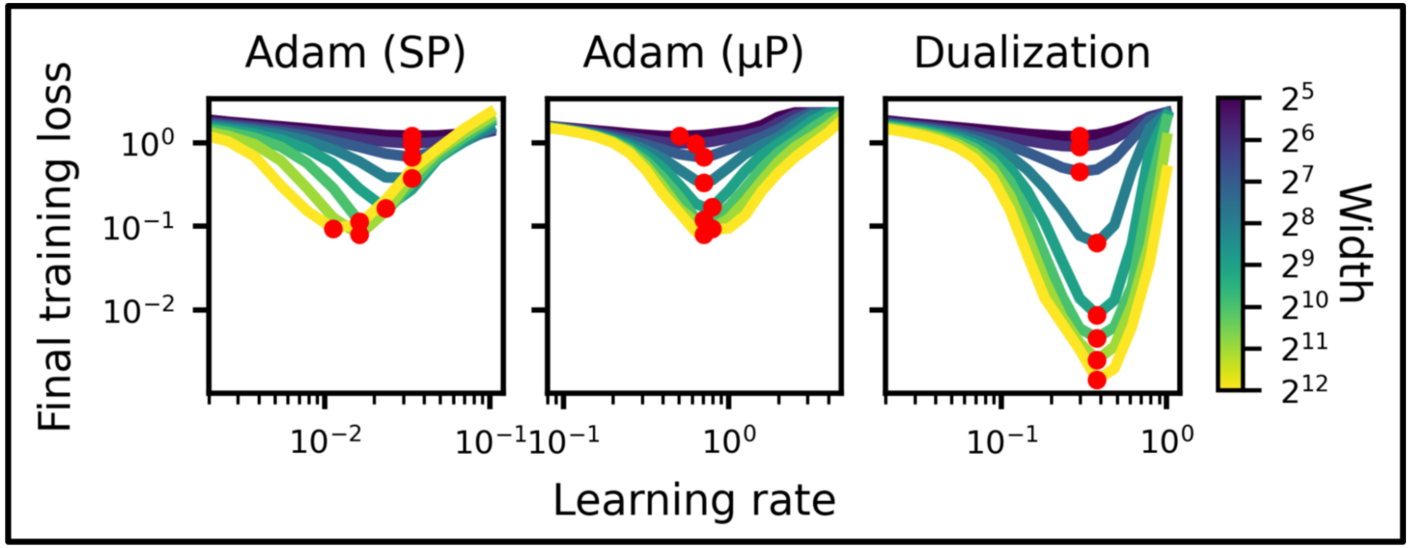


Table 1: **Hyperparameters That Can Be μTransferred, Not μTransferred, or μTransferred Across,** with a few caveats discussed in Section 6.1. * means *empirically validated only* on Transformers, while all others additionally have theoretical justification.

μTransferable	Not μTransferable	μTransferred *Across*
optimization related, init, parameter multipliers, etc	regularization (dropout, weight decay, etc)	width, depth*, batch size*, training time*, seq length*

🌐 https://kellerjordan.github.io/posts/muon/#results

📗 Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer., Yang et al.

Muon

Momentum + Orthogonal Gradients

1. Compute gradient $G_t = \nabla_\theta L$
2. Update momentum $B_t = \mu B_{t-1} + G_t$
3. Orthogonalize $B_t = U \Sigma V^T \rightarrow O_t = UV^T$
4. Update $\theta_t = \theta_{t-1} - \eta O_t$

Why would orthogonal gradients help?

Today's Goal

Understand that

muon is a steepest descent under spectral norm

and why it's a good idea.

Today's Goal

Understand that

muon is a steepest descent under spectral norm

and why it's a good idea.

I. Preliminary

Some Norms Are Induced by Others

Consider the linear transformation:

$$\boxed{\mathbf{x} \xrightarrow{A} \mathbf{Ax}}$$

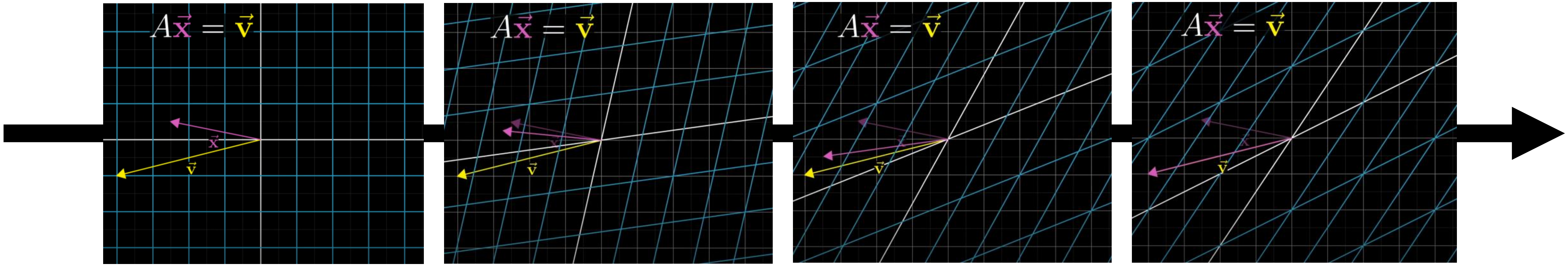
How do we measure the 'size' of $\mathbf{A}$?

Some Norms Are Induced by Others

Operator Norm (Induced Norm)

$$\boxed{\mathbf{x} \xrightarrow{A} \mathbf{Ax}}$$

$\mathbf{A}$ is an operator. $\mathbf{A}$ is defined by how it changes $\mathbf{x}$



The ‘size’ of $\mathbf{A}$ should thus be defined by how much it changes $\mathbf{x}$.

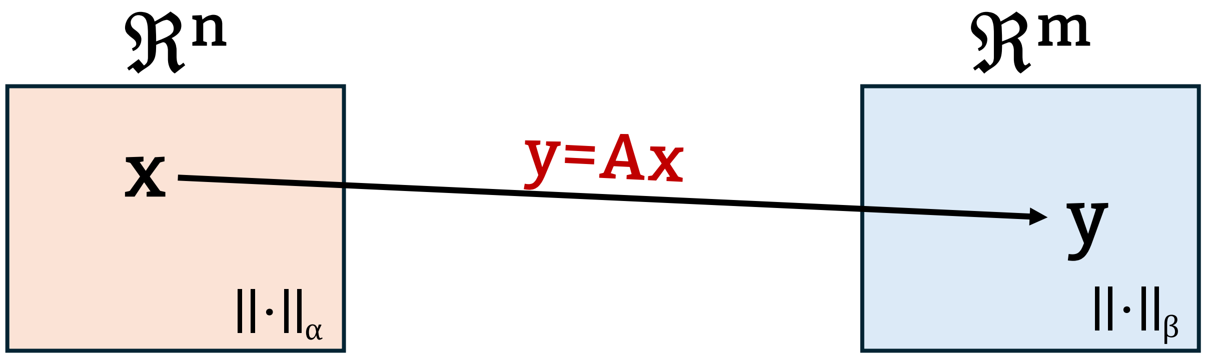
https://youtu.be/uQhTuRlWMxw?si=uCGiRToj25Pdm3dT&t=172

Some Norms Are Induced by Others

Operator Norm (Induced Norm)

A is a linear operator that maps one space (equipped with $||\cdot||_\alpha$) to other space (with $||\cdot||_\beta$).

α and β can be any type of norm of our choice.



Then the operator norm **A** is defined by the maximum norm growth from input to output:

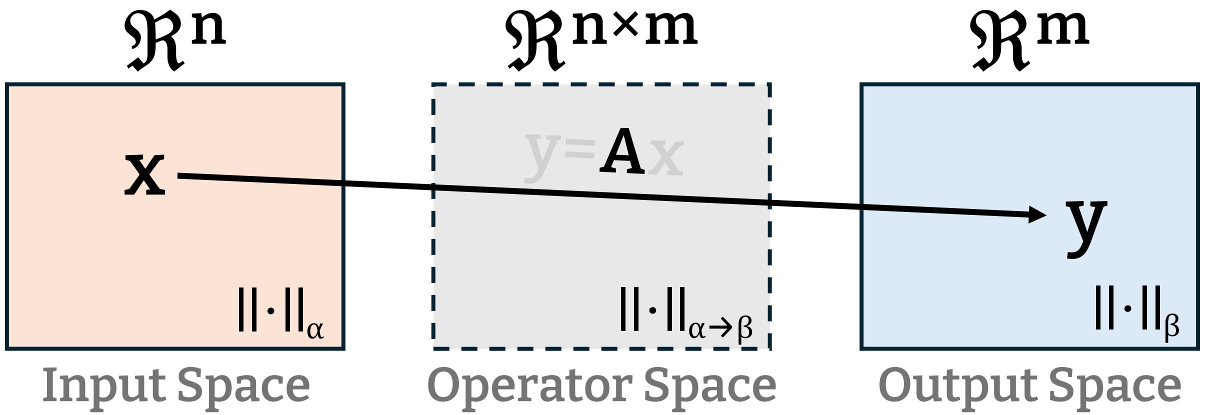
$$||A||_{\alpha\to\beta} \triangleq \sup_{x\neq 0} \frac{||Ax||_\beta}{||x||_\alpha} = \sup_{||x||_\alpha=1} ||Ax||_\beta$$

Foreshadowing: we should use this to measure the norm of weight matrices in ML/DL!

Some Norms Are Induced by Others

Operator Norm (Induced Norm)

Now we can draw the full diagram:



Example: Mapping from Euclidean to Euclidean (ℓ_2-to-ℓ_2)

Then the operator norm of A is the spectral norm:

$$||A||_{\ell_2\to\ell_2} \triangleq \sup_{x\neq 0} \frac{||Ax||_2}{||x||_2} = (\text{largest singular value of } A) = ||A||_* = ||A||_{S_\infty}$$

* ℓ_2-to-ℓ_2 operator norm = Spectral norm = Schatten-∞ norm

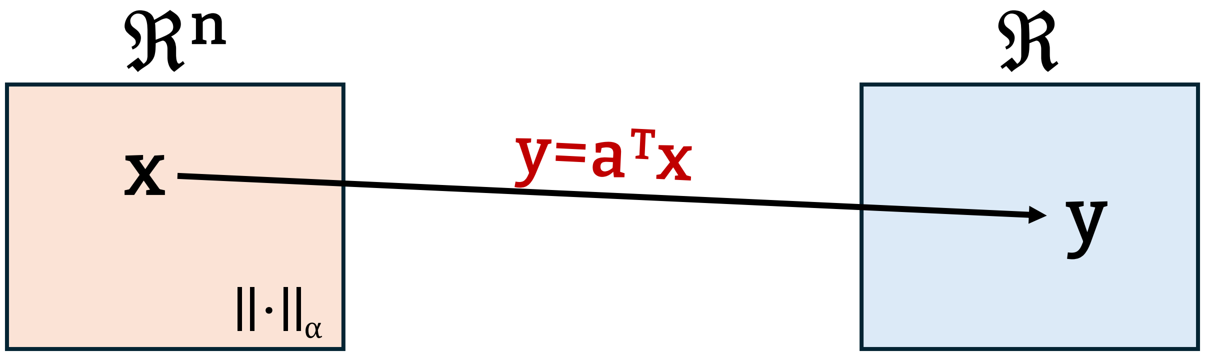
Some Norms Are Induced by Others

Dual Norm

A special case of operator norm, where $\mathbf{A}$ is a vector rather than a matrix.

$$\boxed{\mathbf{x} \xrightarrow{a} \mathbf{a}^T\mathbf{x}}$$

Still an operator, but maps to a scalar rather than another vector space!

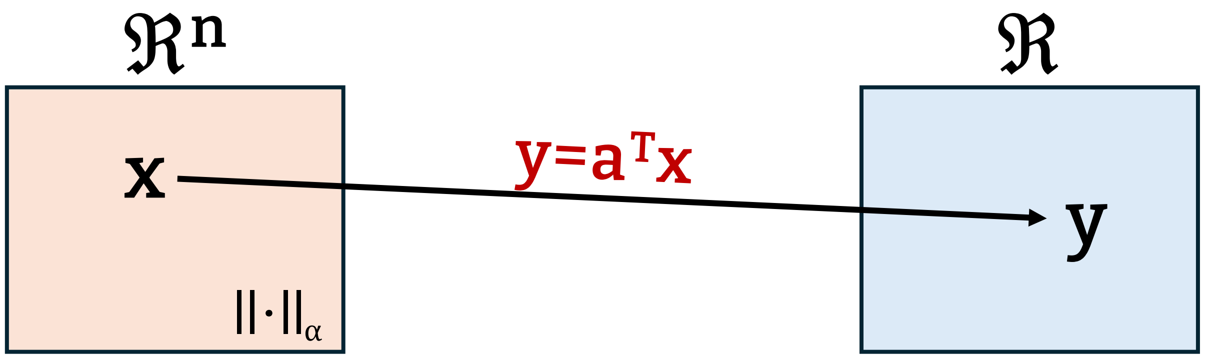


Some Norms Are Induced by Others

Dual Norm

a is a linear operator that maps a vector space (equipped with $||\cdot||_\alpha$) to a scalar.

α can be any type of norm of our choice.



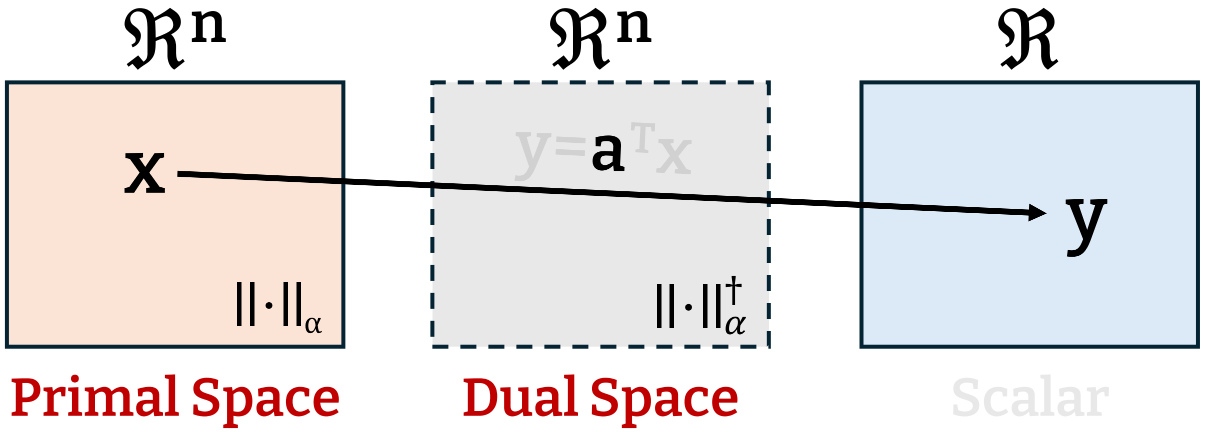
This defines the dual norm of α, indicated by the dagger(†):

$$||a||_\alpha^\dagger \triangleq \sup_{x\neq 0} \frac{a^T x}{||x||_\alpha} = \sup_{||x||_\alpha=1} a^T x$$

Some Norms Are Induced by Others

Dual Norm

We also have a special name for the spaces in this case:



Important note: Dual of ℓ_p is ℓ_q, where $1/p + 1/q = 1$

Example: Dual of ℓ_∞ is ℓ_1, dual of ℓ_1 is ℓ_∞.

Gradients Live in Dual Space

When I was a wee little undergraduate…

"Gradients tell us how fast the loss changes…
isn't it so weird that we directly subtract it from the parameter?"

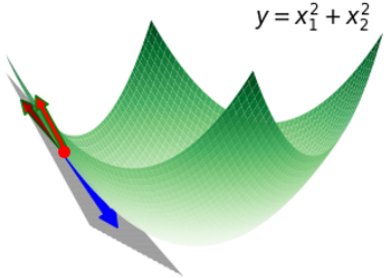
Intentional confusion

On higher dimension: moving x to minimize $y = f(x),\ x \in \mathbb{R}^2$

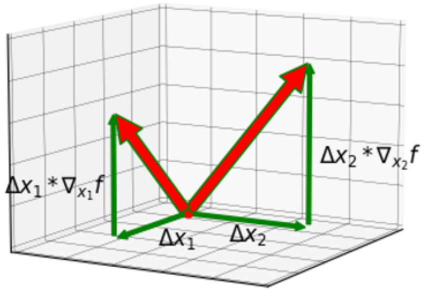
If x is currently at a, how much should x move? $-\nabla_x f(a)$!

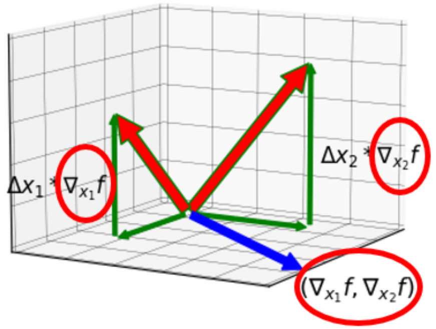
Gradient $\nabla_x f(a) = \begin{bmatrix} \nabla_{x_1} f(a) \\ \nabla_{x_2} f(a) \end{bmatrix}$ consists of 2 partial derivatives to corresponding axes

Seems pretty "descent-y"



…but the blue arrow still seems to be out of nowhere!



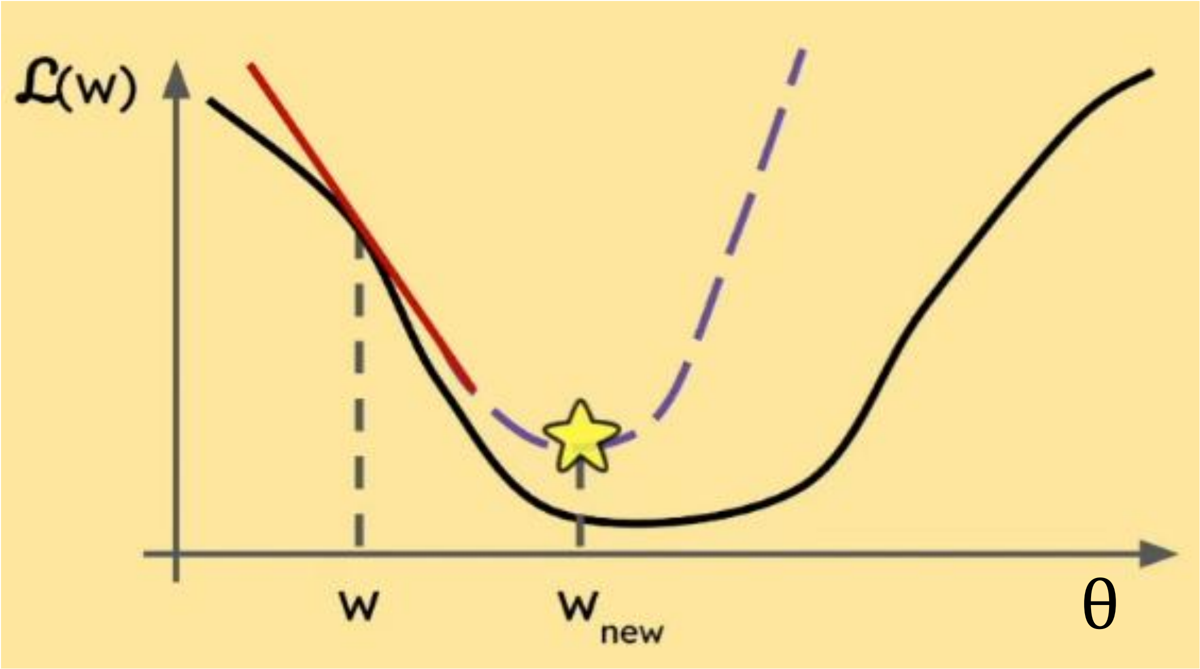


Gradients Live in Dual Space

↑↑↑↑↑ That's Because ↑↑↑↑↑

The gradient $\nabla_\theta \mathbf{L}$ is a first-order approximation on how fast the loss $\mathbf{L}$ changes (near $\mathbf{w}$).

$\boxed{L(w+\Delta w) \approx L(w) + g^T \Delta w}$ ---→ $\boxed{g^T \Delta w \approx L(w+\Delta w) - L(w) = \Delta L}$



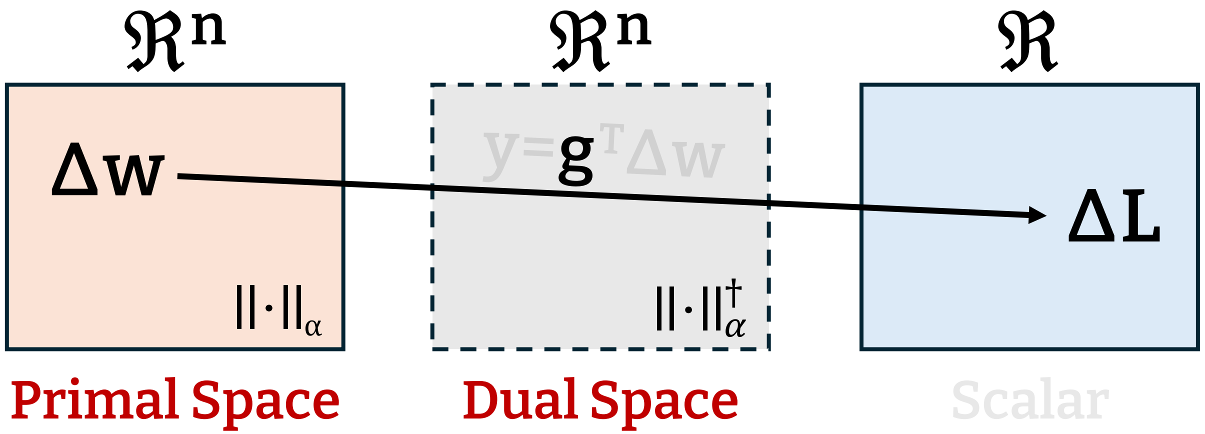
Gradients Live in Dual Space

↑↑↑↑↑ That's Because ↑↑↑↑↑

i.e., the gradient is a linear operator on Δw (that approximates ΔL).

$\boxed{\Delta w \xrightarrow{g} g^T \Delta w}$ $\boxed{g = \nabla_\theta L}$ $\boxed{g^T \Delta w \approx \Delta L}$

So even though Δw and g are both $\mathfrak{R}^n$, they in fact live in different spaces: primal and dual!



Foreshadowing: everything gradient related will involve the dualization: $\boxed{\sup_{\|x\|_\alpha = 1} g^T x}$

Gradients Live in Dual Space

Why didn't we care about this the entire time?

Because we don't need to *if* we're using Euclidean norm!

1. Recall that: dual of ℓ_p is ℓ_q, where $1/p + 1/q = 1$

2. So, if we're using Euclidean norm (ℓ_2) on the parameters (primal space):

$$\|g\|_2^\dagger \triangleq \sup_{\|x\|_2=1} g^T x = \|g\|_2$$

The gradients (dual space) are also measured by Euclidean norm!

3. More importantly (and jumping a bit ahead):

$$\text{Steepest Descent} = \frac{\|g\|_2^\dagger}{\lambda} \operatorname*{argmax}_{\|x\|_2=1} g^T x = -\frac{\|g\|_2}{\lambda} \frac{g}{\|g\|_2} = -\frac{1}{\lambda} g$$

Gradient IS the steepest descent under Euclidean norm!

Summary

Operator norm $\alpha \rightarrow \beta$ is defined by the maximum norm growth from input to output:

$$||A||_{\alpha\rightarrow\beta} \triangleq \sup_{x\neq 0} \frac{||Ax||_\beta}{||x||_\alpha} = \sup_{||x||_\alpha = 1} ||Ax||_\beta$$

We will use this to measure the norm of weight matrices in neural networks.

The dual norm of α is a special case of operator norm where the output is a scalar

$$||a||_\alpha^\dagger \triangleq \sup_{x\neq 0} \frac{a^T x}{||x||_\alpha} = \sup_{||x||_\alpha = 1} a^T x$$

Gradients live in dual space (of parameters), so it will always involve the dualization: $\boxed{\sup_{||x||_\alpha = 1} g^T x}$

So far...

Understand that

muon is a steepest descent under spectral norm

and why it's a good idea.

II. Steepest Descent

Optimization Algorithms

Gradient-based optimizers are basically...

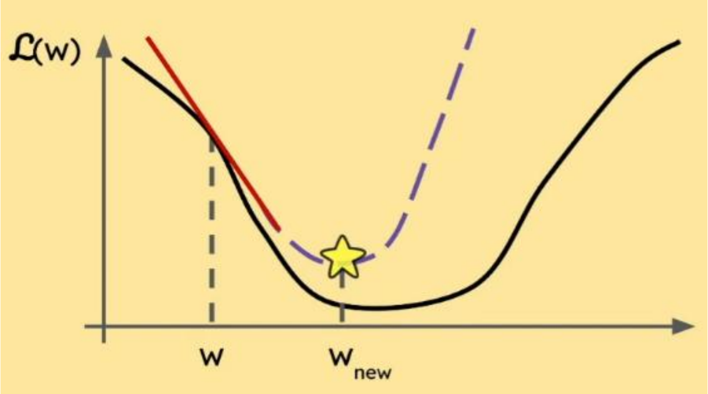
1. Taylor Expansion on the loss function

$$\boxed{L(w+\Delta w)} = \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\Delta w^T H \Delta w} + \boxed{\dots}$$

2. Approximate high-order terms

$$\boxed{L(w+\Delta w)} \approx \boxed{L(w) + g^T\Delta w} + \boxed{D(w, w+\Delta w)}$$

3. Minimize (find Δw that minimize RHS)



Optimization Algorithms

Different higher-order modeling = Different optimization

$$\boxed{L(w+\Delta w)} = \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\Delta w^T H \Delta w} + \boxed{\ldots}$$

ex1) Euclidean Norm → Gradient Descent (1st order method)

$$\approx \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\lambda^* \|\Delta w\|_2^2}$$

Optimization Algorithms

Different higher-order modeling = Different optimization

$$\boxed{L(w+\Delta w)} = \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\Delta w^T H \Delta w} + \boxed{\dots}$$

ex1) Euclidean Norm → Gradient Descent (1^{st} order method)

$$\approx \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\lambda^* \|\Delta w\|_2^2}$$

ex2) Hessian Matrix → Newton's Method (2^{nd} order)

$$\approx \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\, \Delta w^T H \Delta w}$$

Optimization Algorithms

Different higher-order modeling = Different optimization

$$L(w+\Delta w) = L(w) + g^T\Delta w + \tfrac{1}{2}\Delta w^T H \Delta w + \ldots$$

ex1) Euclidean Norm → Gradient Descent (1st order method)

$$\approx L(w) + g^T\Delta w + \tfrac{1}{2}\lambda^* \|\Delta w\|_2^2$$

ex2) Hessian Matrix → Newton's Method (2nd order)

$$\approx L(w) + g^T\Delta w + \tfrac{1}{2}\Delta w^T H \Delta w$$

ex3) Some distance function **D** on output **f** (e.g., TRPO: KL divergence)

→ Natural Gradient Descent (2nd order)

$$\approx L(w) + g^T\Delta w + D(f, f+\Delta f)$$

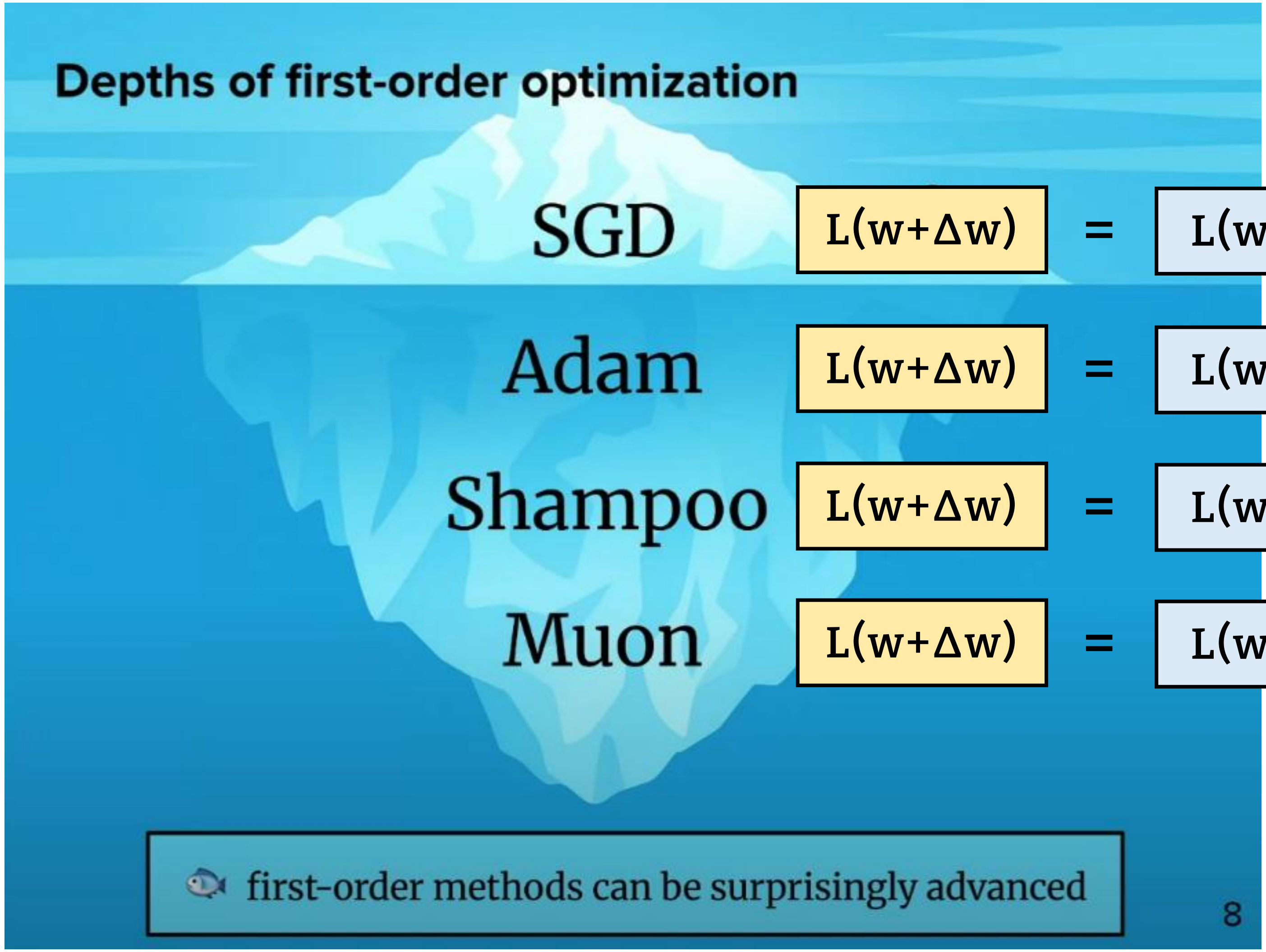
$$\approx L(w) + g^T\Delta w + \tfrac{1}{2}\Delta w^T (\nabla_w f^T \nabla_f^2 D \nabla_w f)\Delta w$$

Optimization Algorithms

A lot to argue about which is better, but let's just say that...

Same Steepest Descent, Different Norms

...and we will see that these all use the same formula as SGD



...but with different norms!

$L(w+\Delta w)$	=	$L(w) + g^T\Delta w$	+	$\tfrac{1}{2}\lambda^* \lVert \Delta w \rVert_2^2$
$L(w+\Delta w)$	=	$L(w) + g^T\Delta w$	+	$\tfrac{1}{2}\lambda^* \lVert \Delta w \rVert_\infty^2$
$L(w+\Delta w)$	=	$L(w) + g^T\Delta w$	+	$\tfrac{1}{2}\lambda^* \lVert \Delta w \rVert_{\ell_2\to\ell_2}^2$
$L(w+\Delta w)$	=	$L(w) + g^T\Delta w$	+	$\tfrac{1}{2}\lambda^* \lVert \Delta w \rVert_{\ell_2\to\ell_2}^2$

Same Steepest Descent, Different Norms

Same Steepest Descent, Different Norms

The first-order method:

$$\boxed{L(w+\Delta w)} = \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\Delta w^T H \Delta w} + \boxed{\ldots}$$

$$\leq \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\lambda^* \|\Delta w\|^2}$$

Sharpness parameter

Square of *any* norm

e.g., L2 → Gradient descent

This is also called a **steepest descent**

Proposition 1 (Steepest descent) *For any* $\boldsymbol{g} \in \mathbb{R}^n$ *thought of as "the gradient" and any* $\lambda \geqslant 0$ *thought of as "the sharpness", and for any norm* $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ *with dual norm* $\|\cdot\|^\dagger$*:*

$$\underset{\Delta \boldsymbol{w} \in \mathbb{R}^n}{\arg\min} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda} \cdot \underset{\|\boldsymbol{t}\|=1}{\arg\max}\, \boldsymbol{g}^\top \boldsymbol{t}. \tag{1}$$

📗 Old Optimizer, New Norm: An Anthology., Bernstein & Newhouse.

Note: You can actually derive the $\frac{1}{2}\lambda^* \|\Delta w\|$ under Lipschitz condition on the gradient (which is indeed related to sharpness); see Appendix B.1 and B.3.

Same Steepest Descent, Different Norms

Simple intuition: λ decides step size, ||·|| decides direction

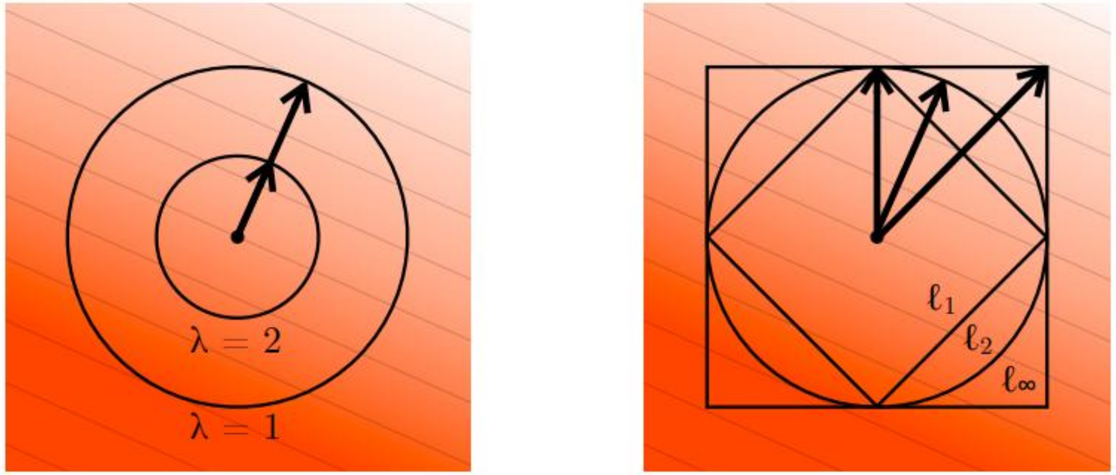
$$\underset{\Delta \boldsymbol{w} \in \mathbb{R}^n}{\arg\min} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = \underbrace{-\frac{\|\boldsymbol{g}\|^\dagger}{\lambda}}_{\text{Step size}} \cdot \underbrace{\underset{\|\boldsymbol{t}\|=1}{\arg\max}\, \boldsymbol{g}^\top \boldsymbol{t}.}_{\text{Step direction}}$$

Step size

If the landscape is sharp, take smaller steps

Step direction

1. Draw a unit ball of selected norm
2. Find the direction that changes L the most



a) varying sharpness λ   b) varying choice of norm ‖·‖

Same Steepest Descent, Different Norms

Extra intuition: gradients are always 'dualized'

$$\operatorname*{arg\,min}_{\Delta \boldsymbol{w} \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = \underbrace{-\frac{\|\boldsymbol{g}\|^\dagger}{\lambda}}_{\text{Step size}} \cdot \underbrace{\operatorname*{arg\,max}_{\|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}}_{\text{Step direction}}.$$

Step size which uses dual norm on g

Step direction also called a duality map on g

Recall: g is an operator on w, so any operation on g must depend on w (or t).

Definition 1 (Dual norm). *Given a norm $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$, the dual norm $\|\cdot\|^\dagger$ of a vector $\boldsymbol{g} \in \mathbb{R}^n$ is given by:*

$$\|\boldsymbol{g}\|^\dagger := \max_{\boldsymbol{t} \in \mathbb{R}^n : \|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}. \tag{5}$$

Definition 2 (Duality map based on a norm). *Given a norm $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$, we consider the duality map:*

$$\operatorname{dualize}_{\|\cdot\|} \boldsymbol{g} := \operatorname*{arg\,max}_{\boldsymbol{t} \in \mathbb{R}^n : \|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}, \tag{6}$$

where, if the arg max *is not unique,* $\operatorname{dualize}_{\|\cdot\|}$ *returns any maximizer.*

Same Steepest Descent, Different Norms

Example: Vanilla Gradient Descent is steepest under ℓ_2

$$\operatorname*{arg\,min}_{\Delta \boldsymbol{w} \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda} \cdot \operatorname*{arg\,max}_{\|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}.$$

$$||g||_2^\dagger \triangleq \sup_{||x||_2=1} g^T x = ||g||_2$$

$$\operatorname*{argmax}_{||x||_2=1} g^T x = \frac{g}{||g||_2}$$

$$\frac{||g||_2^\dagger}{\lambda} \operatorname*{argmax}_{||x||_2=1} g^T x = \frac{||g||_2}{\lambda} \frac{g}{||g||_2} = \frac{1}{\lambda} g$$

$$w_{t+1} = w_t - \frac{1}{\lambda} g$$

Same Steepest Descent, Different Norms

Example: Adam without EMA is steepest under ℓ_∞

Adam's update rule:

$$\begin{aligned} \boldsymbol{m}_t &= \beta_1 \cdot \boldsymbol{m}_{t-1} + (1-\beta_1) \cdot \boldsymbol{g}_t, \\ \boldsymbol{v}_t &= \beta_2 \cdot \boldsymbol{v}_{t-1} + (1-\beta_2) \cdot \boldsymbol{g}_t^2, \\ \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \eta \cdot \boldsymbol{m}_t / \sqrt{\boldsymbol{v}_t}, \end{aligned}$$

Adam without EMA ($\beta_1 = \beta_2 = 0$) is a sign descent:

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \eta \cdot \boldsymbol{g}_t / \sqrt{\boldsymbol{g}_t^2} \\ &= \boldsymbol{w}_t - \eta \cdot \operatorname{sign}(\boldsymbol{g}_t). \end{aligned}$$

Note: Same applies to any sign descent algorithms such as RMSProp and Lion.
Note: "EMA can then be thought of as "smoothing out" the algorithm, or making it more robust to mini-batch noise, although nailing down the precise role of EMA is perhaps still an open problem"
📗 Old Optimizer, New Norm: An Anthology., Bernstein & Newhouse.

Same Steepest Descent, Different Norms

Example: Adam without EMA is steepest under ℓ_∞

$$\operatorname*{arg\,min}_{\Delta w \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda} \cdot \operatorname*{arg\,max}_{\|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}.$$

$$\|g\|_\infty^\dagger \triangleq \sup_{\|x\|_\infty=1} g^\top x = \|g\|_1$$

$$\operatorname*{argmax}_{\|x\|_\infty=1} g^\top x = \operatorname{sign}(g)$$

$$\frac{\|g\|_\infty^\dagger}{\lambda} \operatorname*{argmax}_{\|x\|_\infty=1} g^\top x = \frac{\|g\|_1}{\lambda} \operatorname{sign}(g)$$

$$w_{t+1} = w_t - \frac{\|g\|_1}{\lambda} \operatorname{sign}(g)$$

Summary

The first-order method:

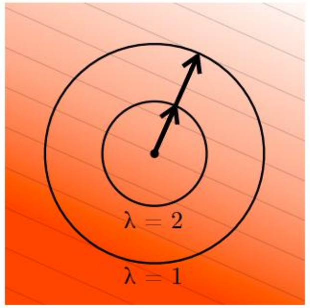
$$\boxed{L(w+\Delta w)} = \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\Delta w^T H \Delta w} + \boxed{\ldots}$$

$$\leq \boxed{L(w) + g^T\Delta w} + \boxed{\tfrac{1}{2}\lambda^* \|\Delta w\|^2}$$

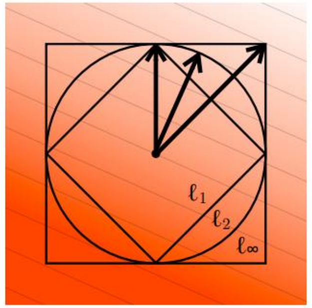
Square of *any* norm

$$\operatorname*{arg\,min}_{\Delta \boldsymbol{w} \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda} \cdot \operatorname*{arg\,max}_{\|\boldsymbol{t}\|=1} \boldsymbol{g}^\top \boldsymbol{t}.$$

and the choice of norm results in a completely different algorithm!



a) varying sharpness λ



b) varying choice of norm $\|\cdot\|$

Next Question

Which norm should be use?

$$\underset{\Delta \boldsymbol{w} \in \mathbb{R}^n}{\arg\min} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda} \cdot \underset{\|\boldsymbol{t}\|=1}{\arg\max}\, \boldsymbol{g}^\top \boldsymbol{t}.$$

$\ell_1, \ell_2, \ldots, \ell_\infty$?

Hold on…

w is not a vector, they are matrices with structure…

Can’t we use matrix norms here?

So far...

Understand that

muon is a steepest descent under spectral norm

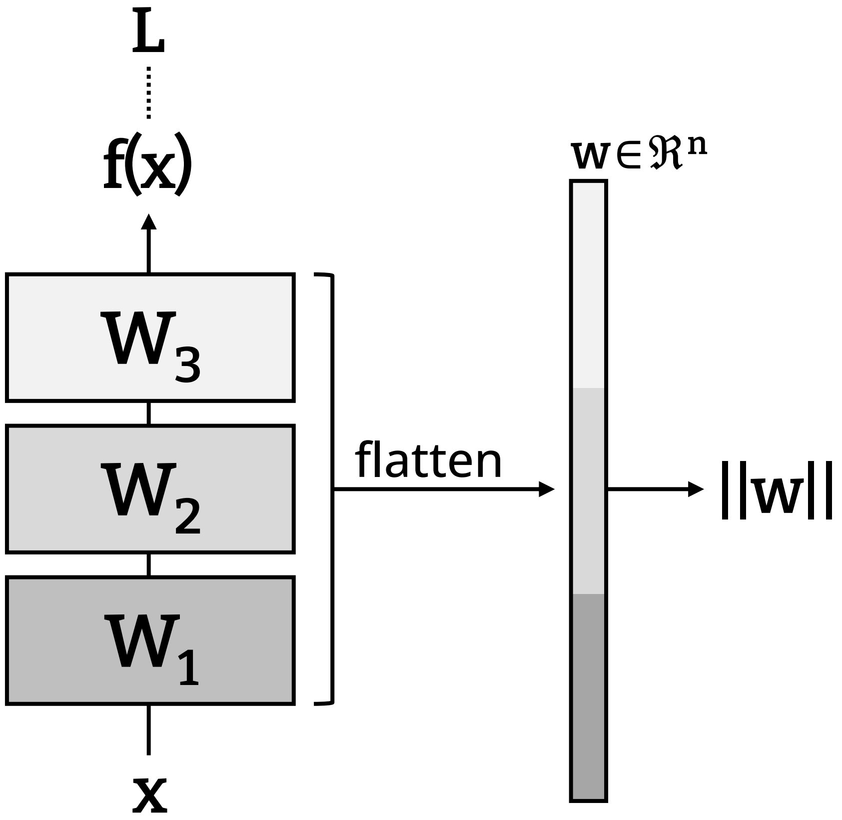
and why it's a good idea.

III. Modular Steepest Descent

Norms Should be Defined Layerwise

What we've been doing so far:

$$\underset{\Delta \boldsymbol{w}\in\mathbb{R}^n}{\arg\min}\left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2}\|\Delta \boldsymbol{w}\|^2\right] = -\frac{\|\boldsymbol{g}\|^\dagger}{\lambda}\cdot \underset{\|\boldsymbol{t}\|=1}{\arg\max}\, \boldsymbol{g}^\top \boldsymbol{t}.$$



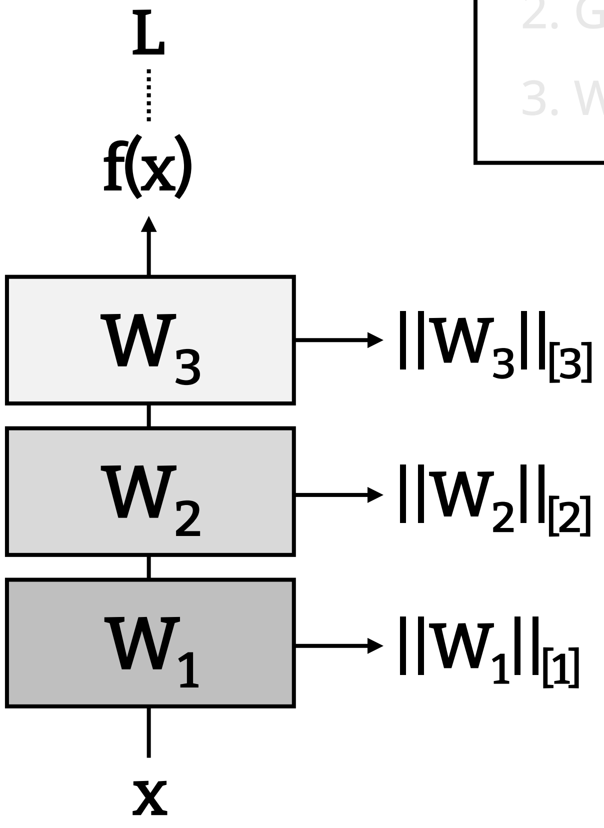
Flattening completely erases the **structure** of the network!

- Number of layers (Depth)
- Size of each layer (layerwise width)
- Layer type (CNN, MLP, Attention, …)
- Weights are matrices, not vectors!
- Weights are operators, not just matrices!

Norms Should be Defined Layerwise

What we should be doing:

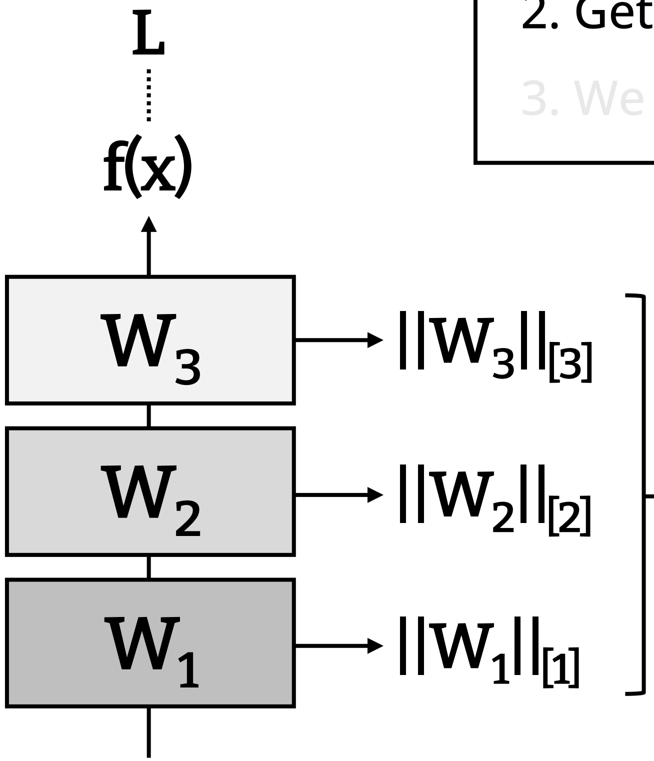
1. Measure each matrix individually: $||\cdot||_{[1]}, ||\cdot||_{[2]}, \ldots, ||\cdot||_{[L]}$
2. Get total norm of the network $||w||$ by weighted max.
3. We can go back to the same steepest descent.



Norms Should be Defined Layerwise

What we should be doing:

1. Measure each matrix individually: $||\cdot||_{[1]}, ||\cdot||_{[2]}, \ldots, ||\cdot||_{[L]}$
2. Get total norm of the network $||\mathbf{w}||$ by weighted max.
3. We can go back to the same steepest descent.



$$||\mathbf{w}||^2 \triangleq \max \mathbf{s}_l^2 \, ||W_l||_{[l]}^2$$

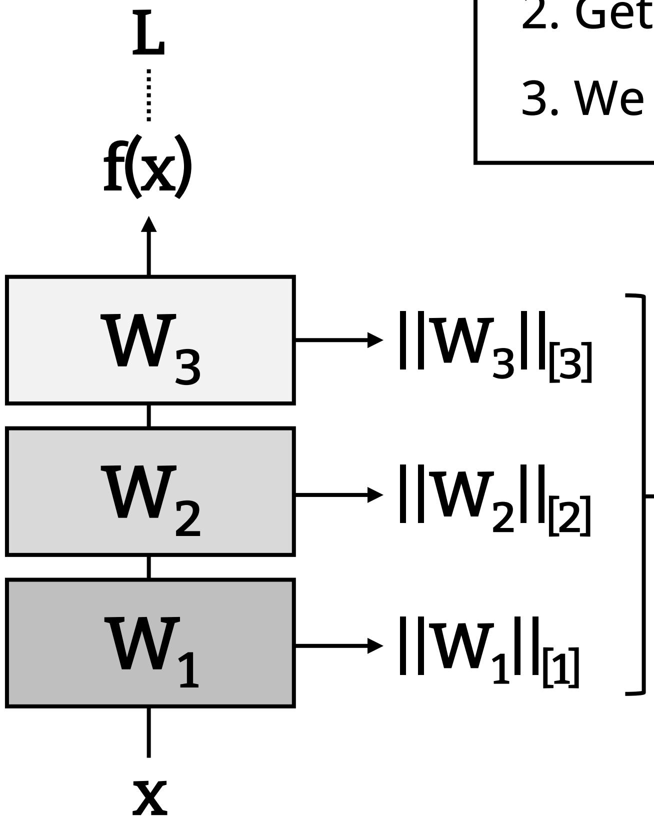
Note: The weights **s** *seems* to connect to 'sensitivity' in later works (for now they're just set to 1).
Note: The max operation kinda makes sense when you actually derive the results (although I'm still unsure of the intuition). See Appendix C.

📗 Scalable Optimization in the Modular Norm., Large et al.
📗 Modular Duality in Deep Learning., Bernstein & Newhouse.

Norms Should be Defined Layerwise

What we should be doing:

1. Measure each matrix individually: $\|\cdot\|_{[1]}, \|\cdot\|_{[2]}, \ldots, \|\cdot\|_{[L]}$
2. Get total norm of the network $\|w\|$ by weighted max.
3. We can go back to the same steepest descent.



$$\|w\|^2 \triangleq \max s_l^2 \, \|W_l\|_{[l]}^2$$

Steepest Descent

$$\operatorname*{arg\,min}_{\Delta w \in \mathbb{R}^n} \left[g^\top \Delta w + \frac{\lambda}{2} \|\Delta w\|^2 \right]$$

Modular Steepest Descent

$$\min_{\Delta W_1, \ldots, \Delta W_L} \left[\sum_{l=1}^{L} \langle G_l, \Delta W_l \rangle + \frac{\lambda}{2} \max_{l=1}^{L} s_l^2 \|\Delta W_l\|_l^2 \right]$$

Norms Should be Defined Layerwise

No big difference other than "is it layerwise or not".

Steepest Descent	Modular Steepest Descent
$\arg\min_{\Delta \boldsymbol{w} \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right]$	$\min_{\Delta \boldsymbol{W}_1, \ldots, \Delta \boldsymbol{W}_L} \left[\sum_{l=1}^{L} \langle \boldsymbol{G}_l, \Delta \boldsymbol{W}_l \rangle + \frac{\lambda}{2} \max_{l=1}^{L} s_l^2 \|\Delta \boldsymbol{W}_l\|_l^2 \right]$
$\Delta w = -\frac{\lvert\lvert g \rvert\rvert^\dagger}{\lambda} \operatorname{argmax}_{\lvert\lvert t \rvert\rvert = 1} g^T t$	$\Delta W_i = -\frac{\eta}{s_l} \operatorname{argmax}_{\lvert\lvert T_i \rvert\rvert = 1} < G_i, T_i >$
Norm on flattened vector	Norm on each matrix
Can use only vector norm	Can use matrix norm
All layers are jointly solved	Each layer is individually solved

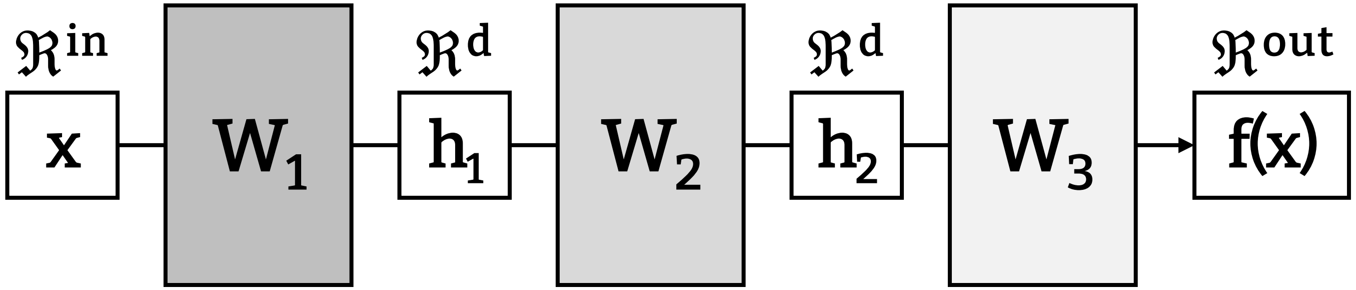
Now the question is: "Which matrix norm do we use?"

Note: For the full solution and derivation of modular steepest descent, see Appendix C.

Which Norm Should We Use?

We need to first ask: what are the matrices doing?

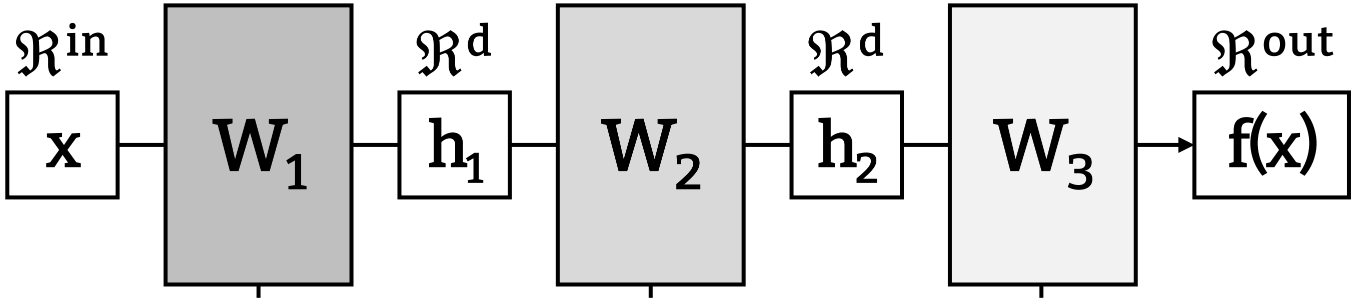
They are linear operators that map (each of their) input space to output space!



Which Norm Should We Use?

We need to first ask: what are the matrices doing?

They are linear operators that map (each of their) input space to output space!



Which means their norms are induced by the norm of the features they work on!

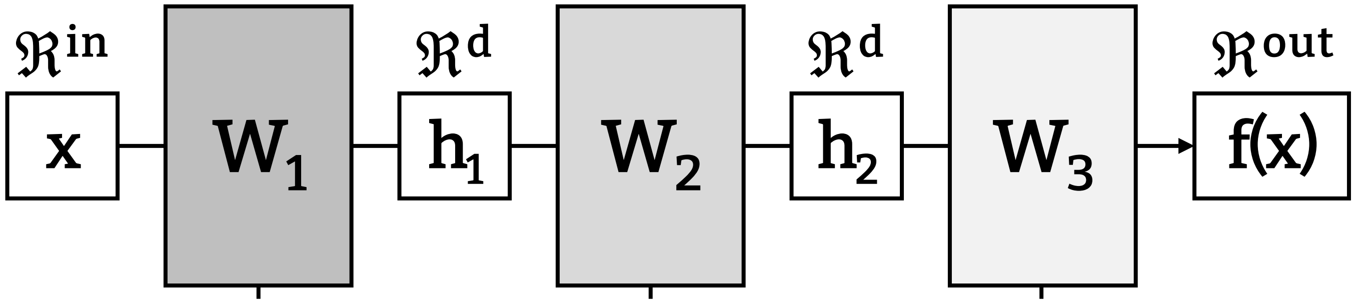
$||W_1||_{x \to h_1}$ $\quad$ $||W_2||_{h_1 \to h_2}$ $\quad$ $||W_3||_{h_2 \to y}$

Recall: $$||A||_{\alpha \to \beta} \triangleq \sup_{x \neq 0} \frac{||Ax||_\beta}{||x||_\alpha} = \sup_{||x||_\alpha = 1} ||Ax||_\beta$$

Which Norm Should We Use?

We need to first ask: what are the matrices doing?

They are linear operators that map (each of their) input space to output space!



Which means their norms are induced by the norm of the features they work on!

$||W_1||_{x \to h_1}$ $||W_2||_{h_1 \to h_2}$ $||W_3||_{h_2 \to y}$

Now the question becomes: “Which feature norm do we use?”

Which Feature Norm Do We Use?

Example: Adam without EMA is steepest under max $\ell_1 \to \ell_\infty$ norm.

Interestingly, Adam can also be thought as using $\ell_1 \to \ell_\infty$ norm on every weight matrix.

First, the $\ell_1 \to \ell_\infty$ norm is simply the largest entry of the matrix.

$$\boxed{||A||_{\ell_1 \to \ell_\infty} \triangleq \sup_{x \neq 0} \frac{||Ax||_\infty}{||x||_1} = \sup_{||x||_1 = 1} ||Ax||_\infty = \max_{i,j} |A_{i,j}|}$$

*Maximized when **x=one-hot(j)** where max entry is at $A_{i,j}$

Then, ℓ_∞ is (coincidentally) the maximum of $\ell_1 \to \ell_\infty$ norms (a.k.a. max-of-max norm)!

$$\underbrace{\|\boldsymbol{w}\|_\infty}_{\uparrow} = \max_l \max_r \|\mathrm{row}_r(\boldsymbol{W}_l)\|_\infty = \underbrace{\max_l \|\boldsymbol{W}_l\|_{\ell_1 \to \ell_\infty}}_{\uparrow}$$

$$\boxed{\operatorname*{arg\,min}_{\Delta \boldsymbol{w} \in \mathbb{R}^n} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \|\Delta \boldsymbol{w}\|^2 \right]} \qquad \boxed{\min_{\Delta \boldsymbol{W}_1, \ldots, \Delta \boldsymbol{W}_L} \left[\sum_{l=1}^{L} \langle \boldsymbol{G}_l, \Delta \boldsymbol{W}_l \rangle + \frac{\lambda}{2} \max_{l=1}^{L} s_l^2 \|\Delta \boldsymbol{W}_l\|_l^2 \right]}$$

Which Feature Norm Do We Use?

Example: Adam without EMA is steepest under max $\ell_1 \rightarrow \ell_\infty$ norm.

Is $\ell_1 \rightarrow \ell_\infty$ natural? Probably not...

Still, they ARE doing modular steepest descent, which may explain why they work so well.

$$\Delta \boldsymbol{W}_l = -\eta \cdot \operatorname{sign}(\boldsymbol{G}_l) \qquad \textit{for each layer } l = 1, ..., L. \tag{11}$$

In words, the matrix-aware steepest descent problem of Equation (10) is solved by layerwise sign descent as given in Equation (11). This observation—that sign descent updates are implicitly doing *per-matrix gradient normalization*—may be a major reason that Adam, sign descent and Lion (Chen et al., 2023) outperform vanilla gradient descent in large language model training (Zhao et al., 2024; Large et al., 2024). The proof is given in Appendix B.

Still still, we can probably do better than this!

📗 Old Optimizer, New Norm: An Anthology., Bernstein & Newhouse.

Summary

Norms should be measured layer-by-layer

Steepest Descent	Modular Steepest Descent
$\underset{\Delta \boldsymbol{w} \in \mathbb{R}^n}{\arg\min} \left[\boldsymbol{g}^\top \Delta \boldsymbol{w} + \frac{\lambda}{2} \lVert \Delta \boldsymbol{w} \rVert^2 \right]$	$\min_{\Delta \boldsymbol{W}_1, \ldots, \Delta \boldsymbol{W}_L} \left[\sum_{l=1}^{L} \langle \boldsymbol{G}_l, \Delta \boldsymbol{W}_l \rangle + \frac{\lambda}{2} \max_{l=1}^{L} s_l^2 \lVert \Delta \boldsymbol{W}_l \rVert_l^2 \right]$
$\Delta w = -\frac{\lvert\lvert g \rvert\rvert^\dagger}{\lambda} \underset{\lvert\lvert t \rvert\rvert = 1}{\operatorname{argmax}} \, g^T t$	$\Delta W_i = -\frac{\eta}{s_l} \underset{\lvert\lvert T_i \rvert\rvert = 1}{\operatorname{argmax}} < G_i, T_i >$

Question: Which matrix norm should we use?

Answer: Matrices in NNs are operators, so it should be an operator norm ($\alpha \to \beta$).

New Question: Which feature norm (α, β) then?

$$\lvert\lvert A \rvert\rvert_{\alpha \to \beta} \triangleq \sup_{x \neq 0} \frac{\lvert\lvert Ax \rvert\rvert_\beta}{\lvert\lvert x \rvert\rvert_\alpha} = \sup_{\lvert\lvert x \rvert\rvert_\alpha = 1} \lvert\lvert Ax \rvert\rvert_\beta$$

So far...

Understand that

muon is a steepest descent under modular norm

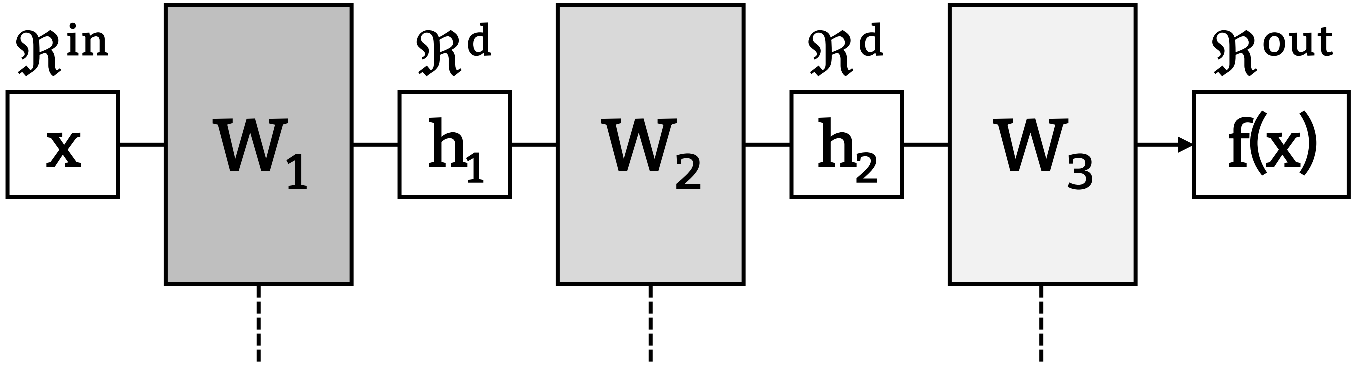
and why it's a good idea.

IV. Deriving Muon

Which Feature Norm Does Muon Use?

Muon asks: which feature norm are we using?

Since we love LayerNorm sooo much, we're using ℓ_2 norm* almost everywhere!



So except for input and output, Muon wants to follow exactly that!

$||W_1||_{x \to \ell_2}$ $\quad$ $||W_2||_{\ell_2 \to \ell_2}$ $\quad$ $||W_3||_{\ell_2 \to y}$

We can define input and output norms, but let's only think about hidden features.

In fact, Muon just uses Adam on input and output layers.

Note: Technically it's ℓ_{RMS} which scales ℓ_2 by sqrt(dim), so it affects step size (but not the direction). This opens a whole another story; see spectral condition and muP.

Muon

Muon w/o momentum is steepest under $\ell_2 \to \ell_2$ norm

1. Compute gradient $G_t = \nabla_\theta L$
2. Update momentum $B_t = \mu B_{t-1} + G_t$
3. Orthogonalize $B_t = U\Sigma V^T \rightarrow O_t = UV^T$
4. Update $\theta_t = \theta_{t-1} - \eta O_t$

$$\min_{\Delta \boldsymbol{W}_1,\dots,\Delta \boldsymbol{W}_L} \left[\sum_{l=1}^{L} \langle \boldsymbol{G}_l, \Delta \boldsymbol{W}_l \rangle + \frac{\lambda}{2} \max_{l=1}^{L} s_l^2 \|\Delta \boldsymbol{W}_l\|_l^2 \right]$$

$$\Delta W_i = -\frac{\eta}{s_l} \underset{||T_i||=1}{\operatorname{argmax}} < G_i, T_i >$$

$$\underset{||T||_{\ell 2 \to \ell 2}=1}{\operatorname{argmax}} <G,T> = UV^T$$

Note: I intentionally hid the dual norm once we moved to matrix norms to only focus on the orthogonalization part. See Appendix C.3 for full derivation.

Muon

Muon w/o momentum is steepest under $\ell_2 \to \ell_2$ norm

$$\underset{||T||_{\ell 2 \to \ell 2}=1}{\operatorname{argmax}} <G,T> = UV^T$$

$$\Delta W_i = -\frac{\eta}{s_l} \underset{||T_i||=1}{\operatorname{argmax}} < G_i, T_i >$$

$$\max_{||X||_{l_2 \to l_2}=1} < G, X > = \max_{||X||_{l_2 \to l_2}=1} tr(G^T X)$$

$$= \max_{||X||_{l_2 \to l_2}=1} tr(V \Sigma U^T X)$$ (Spectral decomposition $G = U \Sigma V^T$)

$$= \max_{||X||_{l_2 \to l_2}=1} tr(\Sigma U^T X V)$$ (Cycle property of trace: $tr(ABC) = tr(BCA) = tr(CAB)$)

$$= tr(\Sigma U^T U V^T V)$$ Maximized by: $X = UV^T$

$$= tr(\Sigma)$$

Orthogonalization via Newton-Schulz

Should we perform SVD every time we update?

1. Compute gradient	$G_t = \nabla_\theta L$
2. Update momentum	$B_t = \mu B_{t-1} + G_t$
3. Orthogonalize	$B_t = U\Sigma V^T \rightarrow O_t = UV^T$
4. Update	$\theta_t = \theta_{t-1} - \eta O_t$

People have found a much faster way to find **UV^T** without performing SVD!

The Newton-Schulz iteration:

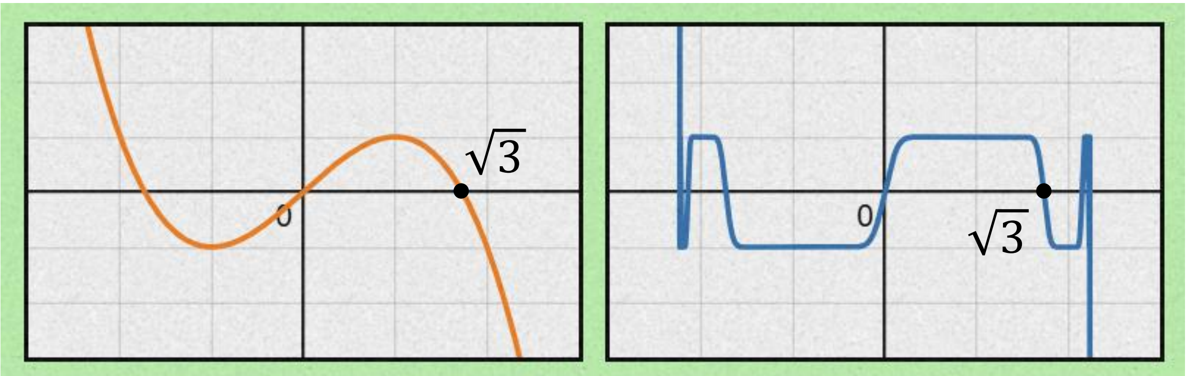
$$X_{t+1} = \frac{2}{3} X_t - \frac{1}{2} X_t X_t^T X_t$$

...that's it. X gets closer to orthogonal after every iter.

Orthogonalization via Newton-Schulz

I can't believe it's that easy!

$$
\begin{aligned}
X_{t+1} &= \frac{2}{3}X_t - \frac{1}{2}X_t X_t^T X_t \\
&= \frac{2}{3}U\Sigma V^T - \frac{1}{2}U\Sigma V^T \cdot V\Sigma^T U^T \cdot U\Sigma V^T \\
&= \frac{2}{3}U\Sigma V^T - \frac{1}{2}U\Sigma^3 V^T \\
&= U \underbrace{\left(\frac{2}{3}\Sigma - \frac{1}{2}\Sigma^3\right)}_{f(\sigma) \text{ on each entry}} V^T
\end{aligned}
$$



$f(\sigma)$ $\quad$ $fffff(\sigma)$

As long as $0 \leq \sigma \leq \sqrt{3}$, every iteration gets σ closer to 1!

Note: Muon uses higher polynomials with better coefficients in practice, but they basically do the same thing but faster.
Note: The $0 \leq \sigma \leq \sqrt{3}$ condition can be satisfied by normalizing the matrix by its Frobenius norm first (which doesn't affect the final output).

So far...

Understand that

muon is a steepest descent under spectral norm

and why it's a good idea.

V. Unfinished Business

Unanswered Questions
(And Where to Find the Answers)

Q1. Why is spectral descent a good idea?

A1. This blog (3min): https://jeremybernste.in/writing/deriving-muon

tl;dr – Because it takes the largest improvement within a safe range.

For the third step in the derivation, we consider choosing a weight update to maximize the linear improvement in loss $\mathcal{L}$ while maintaining a bound on the amount that the outputs can change in response. The rationale is that if the weight update makes the layer outputs change too much, this could de-stabilize the overall network. In symbols, we would like to solve:

$$\min_{\Delta W} \langle \nabla_W \mathcal{L}, \Delta W \rangle \quad \text{subject to} \quad \|\Delta y\|_{\mathrm{RMS}} \leq \eta.$$

change to directly controlling the size of the weight update itself. If the input has size $\|x\|_{\mathrm{RMS}} \leq 1$, we obtain the following problem as a proxy:

$$\min_{\Delta W} \langle \nabla_W \mathcal{L}, \Delta W \rangle \quad \text{subject to} \quad \|\Delta W\|_{\mathrm{RMS}\to\mathrm{RMS}} \leq \eta. \qquad (\dagger)$$

$$\Delta W = -\eta \times \sqrt{\frac{\texttt{fan-out}}{\texttt{fan-in}}} \times UV^\top.$$

Also recommended: Spectral Condition / Appendix J of Tensor Programs V

Unanswered Questions
(And Where to Find the Answers)

Q2. Why should we use L2 or RMS norm?

A2. We don't need to. There is little known about what is the best norm.
(The modular norm paper came out late 2024, sooo...)
In fact, there are words that EMA allows optimizers like Adam or Shampoo to adjust to the 'right' norm for each layer.

🌐 https://x.com/leloykun/status/1847919153589735705

Unanswered Questions

(And Where to Find the Answers)

Q3. What will happen in the future?

A3. Bernstein is expanding on the idea of modular optimization.
He believes that every layer can be designed like lego blocks,
which will make it easier to understand what's going on inside the NNs.

- 🌐 https://jeremybernste.in/writing/deriving-muon Conclusion
- 🌐 https://docs.modula.systems/
- 📗 Scalable Optimization in the Modular Norm., Large et al.
- 📗 Modular Duality in Deep Learning., Bernstein, Newhouse.

Unanswered Questions
(And Where to Find the Answers)

Q4. Has Muon been applied to larger LLMs?

A4. Yes.

📗 Muon is Scalable for LLM Training., Liu et al.

Q5. Has Muon been applied to RL?

A5. Not yet. Our attempt while building V-Simba was unsuccessful.
Could not outperform Adam, but I may not have been cautious enough with the implementation details (e.g., on convolution).

📗 Modular Duality in Deep Learning., Bernstein, Newhouse.

🌐 https://x.com/jxbz/status/1846188906733044029

🌐 https://x.com/tianylin/status/1896542545557262606

"Though this be madness, yet there is method in't."

Hamlet