Lecture 5

- Local linear perspective (Recap)
- Sign SGD
- Gradient Descent
- Shampoo.
- Towards μP and muon.

Why do we care about optimizers?

- Want to train fast. Training time = $$$.
- AdamW + tuned hyperparameters are default.
- Hyperparameter search is difficult for new optimizers.
  - → Initialization rules are nice.
  - → Before this everything was $N(0,1)$
- → But under Adam we see lazy training → i.e. models barely change from their random initialization
  - → Kernel / feature learning perspective → NTK
  - → There is no way random initialization should work
    - → we are not using all the power of these networks ...

$\vec{\theta}$: parameters. $\quad$ $\ell(\vec{x}, \vec{\theta})$ is the loss.

$\vec{x}$: data.

Linearized perspective:

$$\ell(\vec{x}, \vec{\theta}_i + \Delta\vec{\theta}) \simeq \ell(\vec{x}, \vec{\theta}_i) + \left.\frac{\partial \ell}{\partial \theta}\right|_{\vec{\theta} = \theta_i} \Delta\vec{\theta} + \cdots$$

Gradient Descent takes a step.

→ Want this to be small enough that the approximation still holds

→ Want to converge fast

Follow perspective from Bernstein + Newhouse 2024.

↳ Build on many other works

$$\mathcal{L}(\vec{\theta}) = \frac{1}{n} \sum_{j=1}^{n} \ell(\vec{x}_j, \vec{\theta})$$

average over data points.

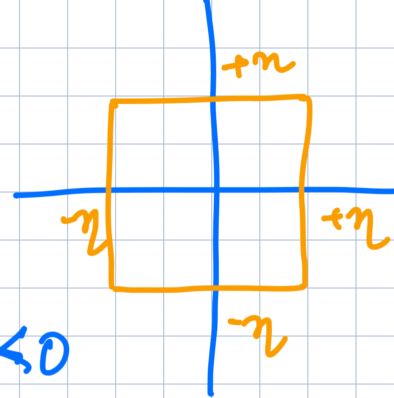
Want:

$$\underset{\|\Delta\vec{\theta}\| \leq \eta}{\operatorname{argmin}} \left(\mathcal{L}(\vec{\theta}) + \langle \nabla_\theta \mathcal{L}(\vec{\theta}), \Delta\vec{\theta} \rangle \right)$$

$$= \underset{\|\Delta\vec{\theta}\| \leq \eta}{\operatorname{argmin}} \langle \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}), \Delta\vec{\theta} \rangle.$$

Choice 1:

$\|\vec{\Delta\theta}\|_\infty \leq \eta \rightarrow$ Linear program



Solution:

$$\Delta\theta[j] = \begin{cases} +\eta & \text{if } \nabla_\theta \mathcal{L}(\vec{\theta})[j] < 0 \\ -\eta & \text{if } \nabla_\theta \mathcal{L}(\theta)[j] > 0 \end{cases}$$

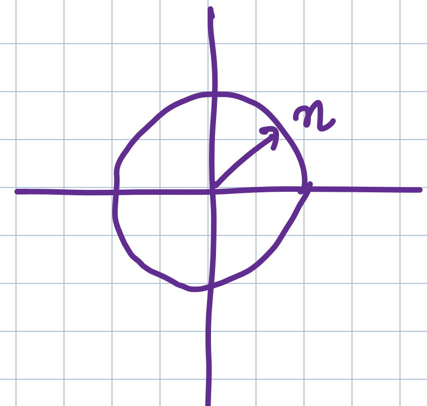
$$\Rightarrow \vec{\Delta\theta} = \underbrace{-\eta \cdot \text{sgn}\left(\nabla_\theta \mathcal{L}(\vec{\theta})\right)}_{\text{sign SGD}}.$$

$$\underset{\|\vec{\Delta\theta}\|_\infty \leq \eta}{\text{argmin}} \langle \nabla_\theta \mathcal{L}(\vec{\theta}), \Delta\vec{\theta} \rangle.$$

$$\underset{\|\vec{y}\|_\infty \leq \eta}{\text{argmin}} \vec{x}^T \vec{y}$$

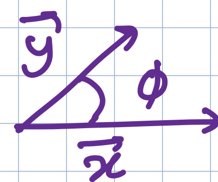
Choice 2:

$\|\vec{\Delta\theta}\|_2 \leq \eta.$



$$\underset{\|\Delta\theta\|_2 \leq \eta}{\text{argmin}} \langle \nabla_{\vec{\theta}} \mathcal{L}(\vec{\theta}), \Delta\vec{\theta} \rangle.$$

$$\vec{x}^T \vec{y} = \|\vec{x}\|_2 \|\vec{y}\|_2 \cos\phi$$



Cauchy-Schwartz : Inner product absolute value maximized when vectors are aligned.

$$\Rightarrow \quad \vec{\Delta\theta} = -\overset{\text{scale by } \eta.}{\eta} \underbrace{\frac{\nabla_\theta \mathcal{L}(\vec{\theta})}{\|\nabla_\theta \mathcal{L}(\vec{\theta})\|_2}}_{\text{unit norm.}}$$

Alternatively, take the regularization / Lagrange multiplier perspective.

$$\underbrace{\underset{\vec{\theta}}{\operatorname{argmin}}}_{\text{unconstrained.}} \cdot \quad \underbrace{\langle \nabla_\theta \mathcal{L}(\vec{\theta}), \Delta\theta \rangle + \lambda \|\Delta\theta\|_2^2}_{g(\Delta\theta).\ \text{convex.}}$$

→ Sweeps out the same family of solutions as constrained version.

$$\nabla g(\Delta\theta) = \nabla_\theta \mathcal{L}(\vec{\theta}) + 2\lambda \vec{\Delta\theta} \quad \longrightarrow \quad \text{Set to zero.}$$

$$\text{Minimizer @ } \quad \vec{\Delta\theta}^* = -\frac{1}{\underbrace{2\lambda}_{\text{stepsize.}}} \underbrace{\nabla_\theta \mathcal{L}(\vec{\theta})}_{\text{gradient descent}}$$

Linearization around matrices?

$\vec{\theta}$: we've been thinking of this as a vector.

But neural network parameters are naturally organized in matrices.

$$\vec{h}_1 = \mathrm{ReLU}[\vec{b}_0 + \boxed{W_0 \vec{x}}]$$
$$\vec{h}_j = \mathrm{ReLU}[\vec{b}_j + W_{j-1} \cdot \vec{h}_{j-1}]$$
$$y = \mathrm{ReLU}[\vec{b}_k + W_k \vec{h}_k]$$

Recall from Xavier initialization ... scaling problems come from the weight matrices.

So what if we think about change ΔW in "weight matrix space" instead.

$$\underset{\|\Delta W\| \leq \eta}{\operatorname{argmin}} \; \langle \underbrace{\nabla_W \mathcal{L}(W)}_{\text{matrix}}, \underbrace{\Delta W}_{\text{matrix.}} \rangle$$

Recall: $\langle A, B \rangle_F = \mathrm{trace}(A^T B)$

Consider: $\|\Delta W\|_2 \leq \eta$ i.e. spectral norm.

Recall: $\|A\|_2 = \sigma_{max}$ largest singular value of A.

$\max_{\|\vec{x}\|_2 = 1} \|A\vec{x}\|_2$ → what is the maximum scaling for a unit vector.

$$\underset{\|\Delta W\|_2 \leq \eta}{\operatorname{argmin}} \langle \nabla_W \mathcal{L}(W), \Delta W \rangle.$$

Let $\nabla_W \mathcal{L}(W) = U \Sigma V^T$.

Then $\Delta W^* = -\eta\, U V^T$.

This gives us the Shampoo Optimizer (Gupta 2017/2018)

→ A variant of Shampoo (Dahl 2023) won AlgoPerf
→ Training alg. competition.

$$\max_{\|B\|_2 \le 1} \operatorname{trace}(A^T B)$$

$$\operatorname{trace}(ABC)$$
$$= \operatorname{trace}(BCA)$$

Let $A = U\Sigma V^T$

$$= \sum_i \sigma_i \vec{u}_i \vec{v}_T^{\,T}$$

$$A = U\Sigma V^T$$

$$\max_{\|B\|_2 \le 1} \operatorname{trace}(A^T B)$$

$$= \max_{\|B\|_2 \le 1} \operatorname{trace}(V \Sigma^T U^T B)$$

$$= \max_{\|B\|_2 \le 1} \operatorname{trace}\left(\sum_i \sigma_i \vec{v}_i \vec{u}_i^{\,T} B\right)$$ Trace is linear.

$$= \max_{\|B\|_2 \le 1} \sum_i \operatorname{trace}(\sigma_i \vec{v}_i \vec{u}_i^{\,T} B)$$

$$= \max_{\|B\|_2 \le 1} \sum_i \operatorname{trace}\left(\sigma_i \underbrace{\vec{u}_i^{\,T} B \vec{v}_i}_{\text{scalar}}\right)$$

$$= \max_{\|B\|_2 \le 1} \sum_i \sigma_i \vec{u}_i^{\,T} \underbrace{B \vec{v}_i}$$

$$\|B\vec{v}_i\|_2 \le \|\vec{v}_i\|_2 \cdot \|B\|_2 \le 1.$$

$$\therefore \|\vec{u}_i^{\,T} B \vec{v}_i\|_2 \le 1.$$ Cauchy-Sch.

$$\le \sum_i \sigma_i$$

$\rightarrow$ To attain bound $B = UV^T$.

$$\operatorname{trace}(V\Sigma^T \cancel{U^T U} V^T)$$
$$= \operatorname{trace}(\Sigma^T)$$

Bernstein + Newhouse 2024

- Choose a norm
- Choose a stepsize.

→ The right norm may depend on geometry / architecture of network.

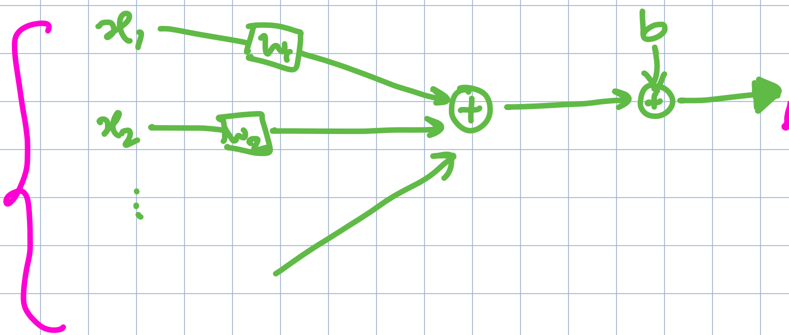
- But we have a recepie
  - → Given a norm, make an optimizer.
  - → examples for $\|\cdot\|_2$, $\|\cdot\|_\infty$, spectral norm.

→ Why do we care about this recepie?

So what might be a good norm for deep learning?

Recap: Xavier initialization.

inputs $N(0,1)$



want: approx $N(0,1)$.

Can we rewrite this in terms of norms?

What norm are we trying to preserve?

Say $\|\vec{x}\|_2 = 1$ Xavier does not imply $\|\vec{h}\|_2 = 1$.

$x_i \sim \frac{1}{\sqrt{d_{in}}}$ $h_i \sim \frac{1}{\sqrt{d_{out}}}$

Consider RMS-norm:

$$\|\vec{x}\|_{RMS} = \frac{1}{\sqrt{d_{in}}} \|\vec{x}\|_2 = \frac{1}{\sqrt{d_{in}}} \sqrt{\sum_i x_i^2} \longrightarrow$$ Average size of each entry 1!

Root mean square.

Hmm... So maybe RMS norm is interesting for DNNs.
Is there a matrix version? (Recall Shampoo...)

Recall : induced Matrix-norm.

$$\|A\|_{\alpha \to \beta} = \max_{\|\vec{x}\|_\alpha = 1} \|A\vec{x}\|_\beta$$

RMS → RMS norm?

$$\|A\|_{RMS \to RMS} = \max_{\|\vec{x}\|_{RMS} = 1} \|A\vec{x}\|_{RMS}$$

A d_1 d_2

$$= \max_{\|\vec{x}\|_2 = \sqrt{d_1}} \frac{1}{\sqrt{d_2}} \cdot \|A\vec{x}\|_2$$

$$= \frac{\sqrt{d_1}}{\sqrt{d_2}} \cdot \boxed{\max_{\|\vec{x}\|_2 = 1} \|A\vec{x}\|_2}$$

Spectral norm!

So can we use this induced RMS→RMS norm in our optimizer recepie?

$$\|\Delta W\|_{RMS^*} \leq \eta \text{ ?}$$

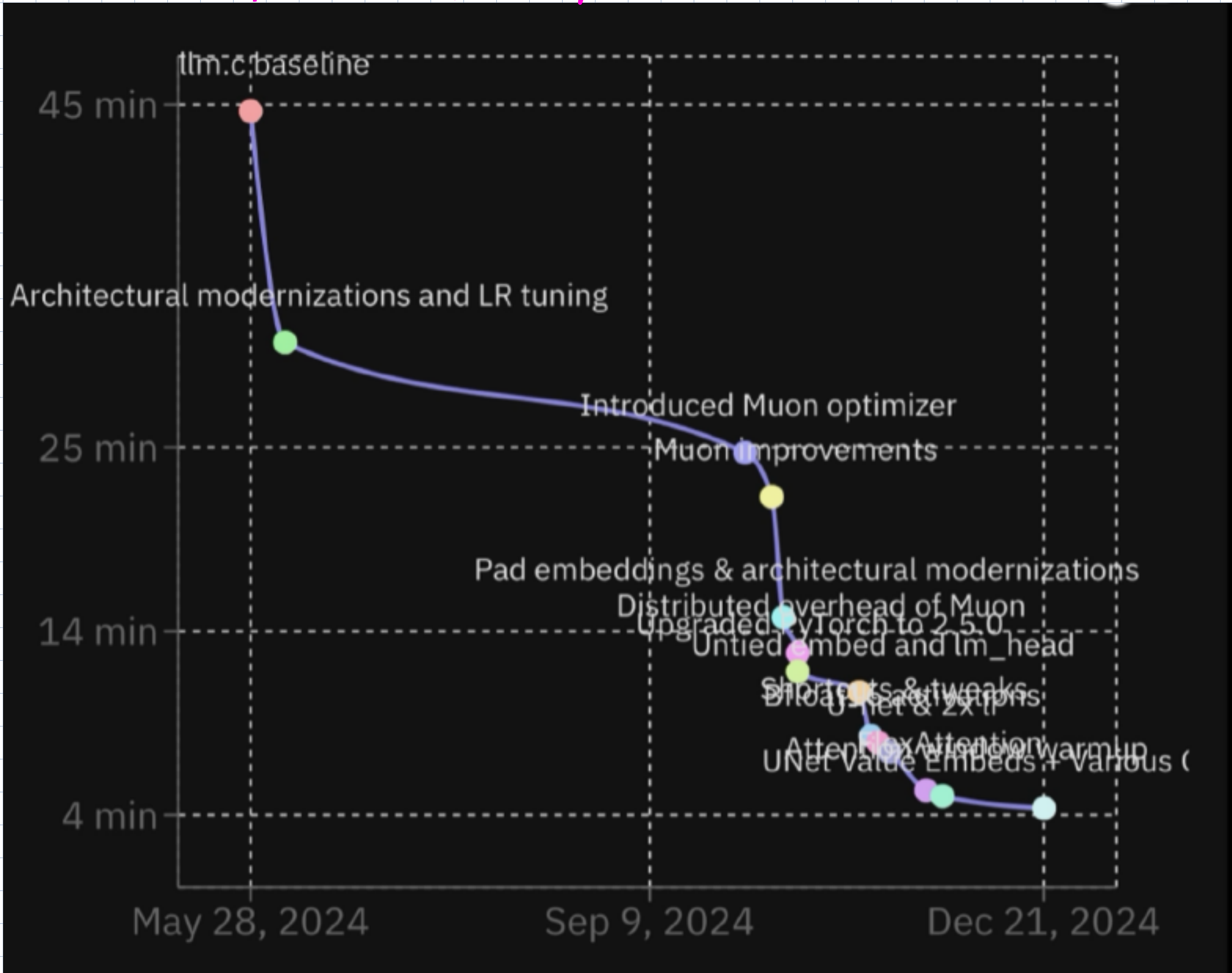
<u>Key observation</u>: If we choose this norm, we can choose only one hyperparameter η across all layers, but the effect will be layer specific learning rates, due to fan-in and fan-out.

$$\|\Delta W\|_{RMS^*} \leq \eta \implies \sqrt{\frac{d_1}{d_2}}\,\|\Delta W\|_2 \leq \eta.$$

$$\implies \|\Delta W\|_2 \leq \eta\sqrt{\frac{d_1}{d_2}}$$

→ Essential idea behind muP → maximal update parameterization.

Nano GPT speedrun.



Muon is Scalable for LLM Training

Technical Report

Jingyuan Liu[1] Jianlin Su[1] Xingcheng Yao[2] Zhejun Jiang[1] Guokun Lai[1] Yulun Du[1]
Yidao Qin[1] Weixin Xu[1] Enzhe Lu[1] Junjie Yan[1] Yanru Chen[1] Huabin Zheng[1]
Yibo Liu[1] Shaowei Liu[1] Bohong Yin[1] Weiran He[1] Han Zhu[1] Yuzhi Wang[1]
Jianzhou Wang[1] Mengnan Dong[1] Zheng Zhang[1] Yongsheng Kang[1] Hao Zhang[1]
Xinran Xu[1] Yutao Zhang[1] Yuxin Wu[1] Xinyu Zhou[1 *] Zhilin Yang[1]

[1] Moonshot AI [2] UCLA

KIMI

22 February 2025