

神经网络Hessian阵的特性 及其在算法分析的应用

Ruoyu Sun



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen



深圳市大数据研究院
Shenzhen Research Institute of Big Data



深圳国际工业与应用数学中心
SHENZHEN INTERNATIONAL CENTER FOR INDUSTRIAL AND APPLIED MATHEMATICS

Overview of This Talk

- Part I: Why do Transformers need Adam?
 - We explain it from **Hessian** perspective
 - Provide a new algorithm Adam-mini
- Part II: On the special structure of Hessian of NNs (Preprint)
 - Empirical observations in Part I: Hessian of NNs exhibit **near-block-diagonal** structure
 - We provide the first theory on **when and why** the special Hessian structure occurs
 - Math tools: Random Matrix Theory

Contents

- Part I: Why Transformers Need Adam and Adam-mini
- Part II: On the Special Structure of Hessian of NNs

Background

- **Adam** is the most popular algorithm in deep learning (DL).
(**210k** citations, by April 2025)

```
optimizer = optim.Adam(net.parameters(), lr=args.lr, betas=(args.beta1, args.beta2), eps=1e-08,|  
                        weight_decay=args.weightdecay, amsgrad=False)
```

- Adam is the most cited optimization algorithm
(2nd most cited is AdaGrad: 15k citations, Adam is 10 x more than AdaGrad)
- Adam just won **ICLR 2025 Test of Time Award**
- Empirical fact: Adam is (almost) **the only choice** for training LLMs
(exception: **Muon** is gaining growing interest very recently)

背景：理论学家对Adam的负面结果居多

2017 年: Berkeley研究者说:
Adam 泛化性能不好

The **marginal** value of **adaptive gradient methods** in machine learning

..., [M Stern](#), [N Srebro](#), [B Recht](#) - Advances in neural ...
[proceedings.neurips.cc](#)

2019 年ICLR outstanding paper:
MIT&Google 研究者: Adam 不收敛

On the **convergence** of adam and beyond

[SJ Reddi](#), [S Kale](#), [S Kumar](#) - arXiv preprint
arXiv:1904.09237, 2019 - [arxiv.org](#)

... ..

但这些理论没有改变 Adam 一枝独秀的现实。

2022年: 我们证明了 Adam 在允许调参的意义下是收敛的!

- **Theorem 1:** Consider the previous setting.

When $\beta_2 \geq 1 - O\left(\frac{1-\beta_1^n}{n^{3.5}}\right)$ and $\beta_1 < \sqrt{\beta_2} < 1$, Adam with stepsize $\eta_k = \frac{1}{\sqrt{k}}$ converges to the neighborhood of stationary points:

$$\min_{k \in [1, T]} \mathbb{E} \|\nabla f(x_k)\|_2^2 = O\left(\frac{\log T}{(1 - \beta_2)^2 \sqrt{T}} + (1 - \beta_2) D_0\right).$$

Shi, Li, Hong, & **Sun**, RMSprop converges with proper hyper-parameter. **ICLR 2021 spotlight**.
Zhang, Chen, Shi, **Sun***, & Luo, Adam can converge without any modification on update rules. **NeurIPS 2022 spotlight**.

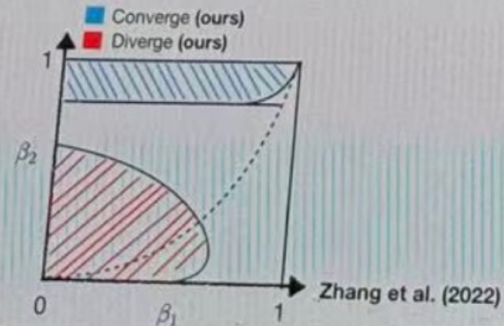


Test of Time Award Speech: Adam Convergence

Convergence - divergence - convergence

E.g.:

- On the convergence of Adam and beyond, Reddi et al. (2018)
 - Worst-case divergence
- A simple convergence proof of Adam and Adagrad, Defossez et al. (2022)
 - Adam is to Adagrad like constant step size SGD is to decaying step size SGD
- Adam can converge without any modification on update rules, Zhang et al. (2022)



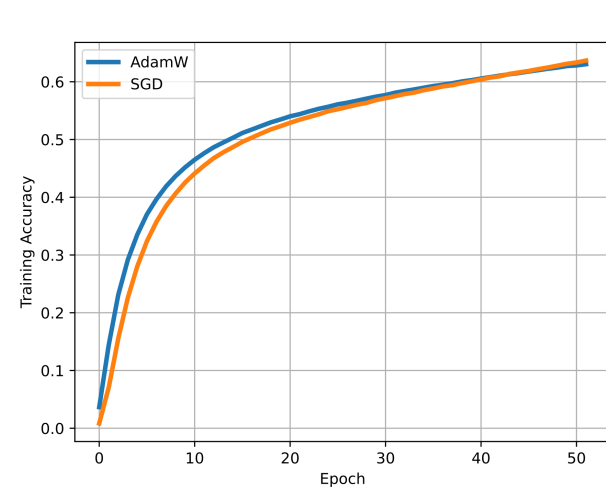
Kingma's ICLR 2025 Test of Time Speech:

--引用我们的工作,
说明 Adam 是收敛的

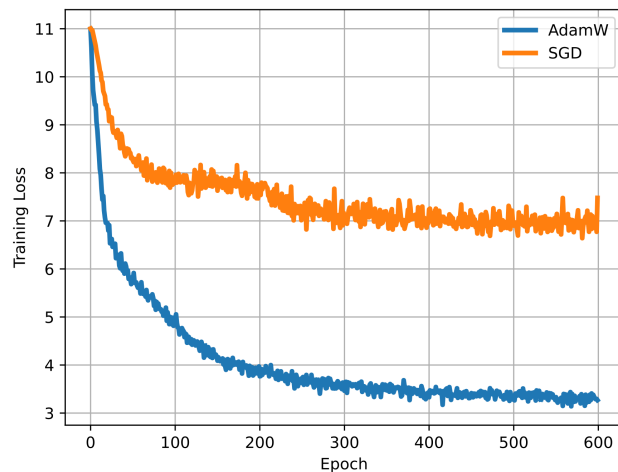
Contents

- Part I-1: Why Transformer Needs Adam

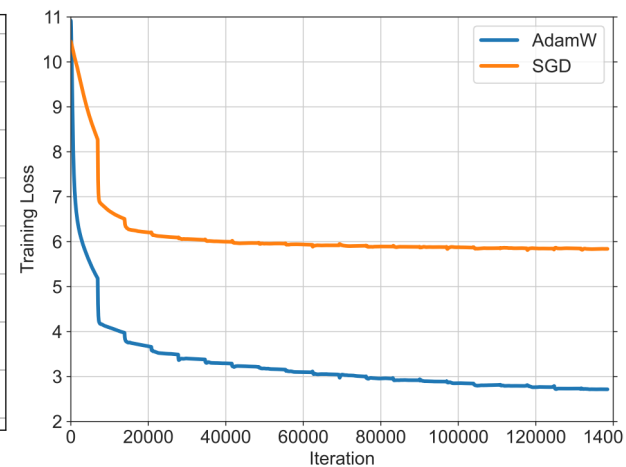
SGD underperforms Adam on Transformers



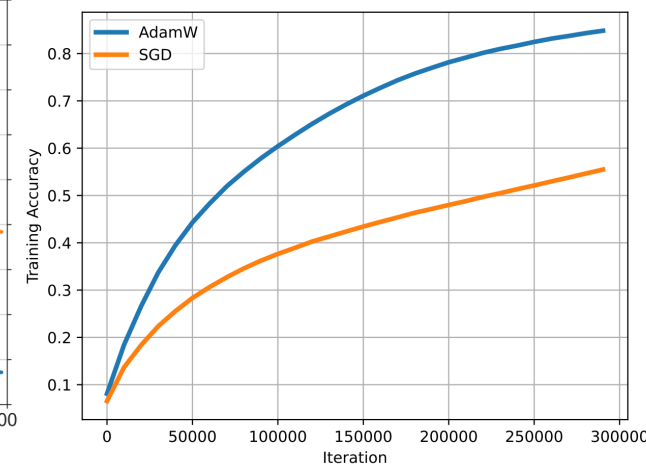
VGG (CNN)



GPT2



BERT



ViT

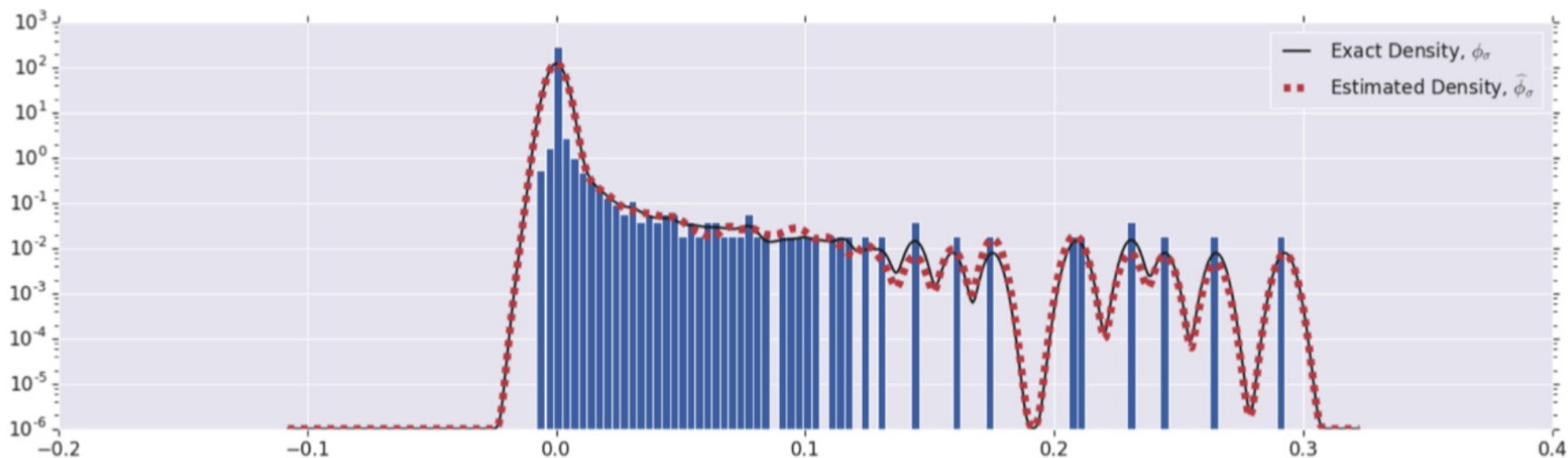
Transformers Need Adam, but why?

What problem structure might hamper SGD?

- **Hessian eigenvalues** largely decides behavior of gradient methods
[Nocedal & Wright'99, Nesterov'13, Goh'17, Sun'2019]
- For instance: ill-conditioning slow down GD
- Can Hessian spectrum explain the gap of SGD and Adam?
- Unfortunately, no (not directly)...

Preliminary: Hessian spectrum

y-axis:
frequency



x-axis: Eigenvalues.

Figure from [Ghorbani et al. 19]

What is spectrum: histogram of eigenvalues

Remark: How to plot Hessian spectrum?

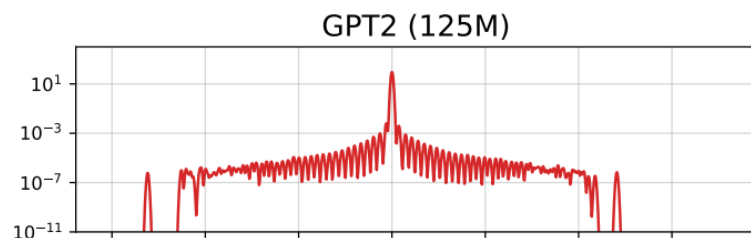
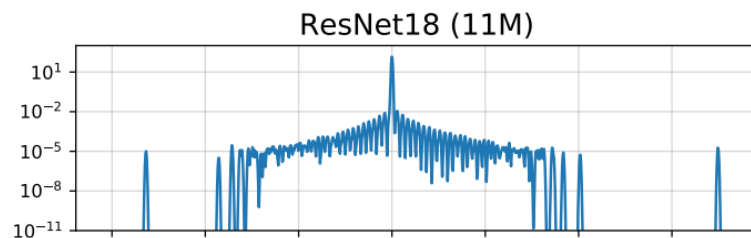
We use **Stochastic Lanczos Quadrature (SLQ)**

[Bai, Fahey, and Golub 1996]

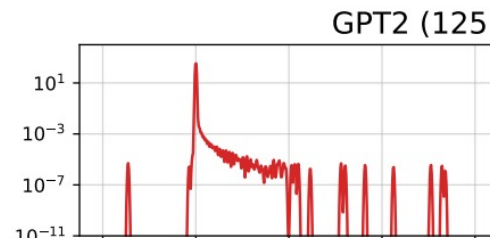
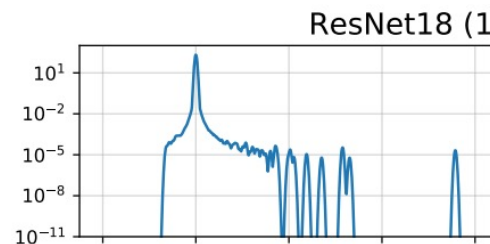
(will take >10 pages to explain, omitted today)

**We will compute the Hessian spectrum
for a wide range of neural networks**

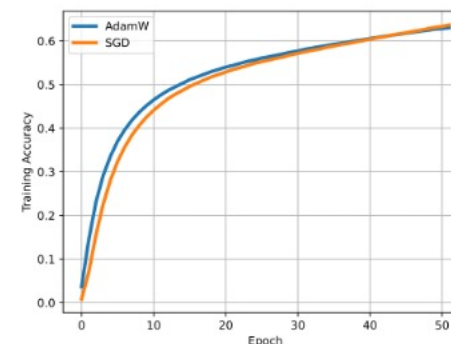
Hessian spectrum cannot explain the gap



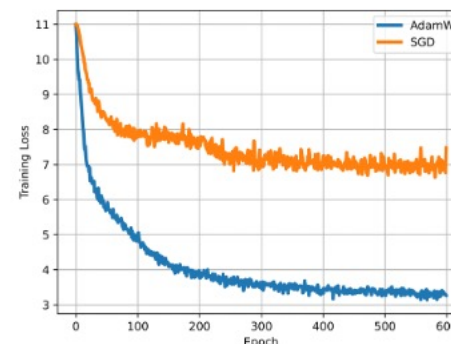
(a) Initialization



(c) 50% step



SGD $\approx$ Adam
on **CNNs**



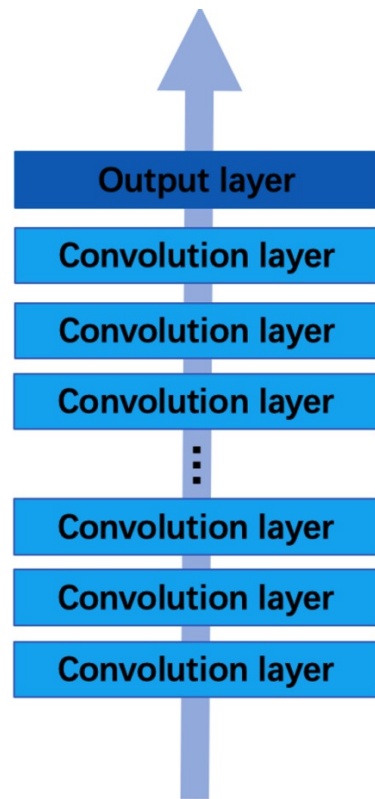
SGD $\ll$ Adam
on **Transformers**

CNN and **Transformers**:
Spectrum looks quite similar!
(see more figures in the paper)

Something must be overlooked...

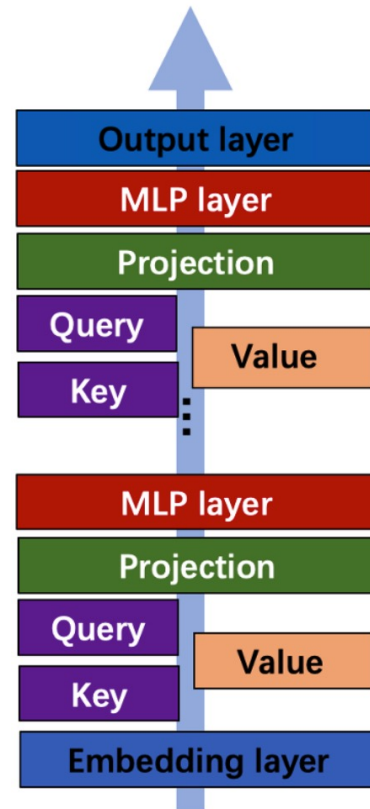
- Full hessian spectrum does not seem informative enough
- What else?
- We find one important features that are overlooked:
The build-up rules of the architecture
 - Transformers are stacked up by different kinds of layers

Build-up rules of architectures



CNNs

Repetitive stack
of similar layers



Transformers

each block follow
different design
(e.g., Q, K, V, MLP)

We hypothesize:

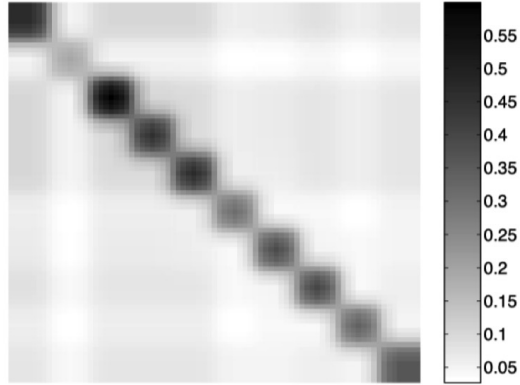
Different designs among parameter blocks



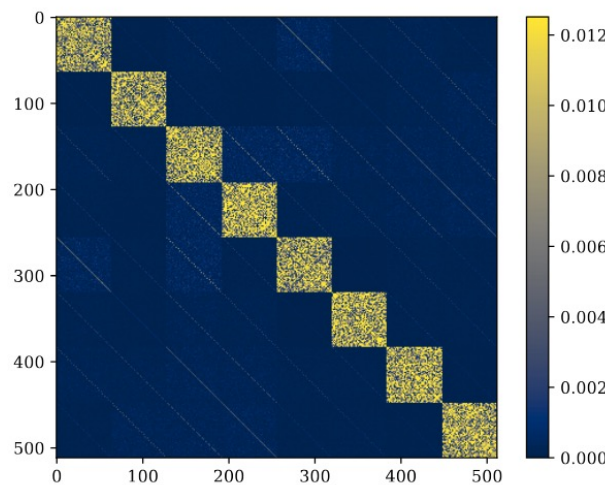
Hessian of these parameter blocks.

This inspires us to investigate the **blockwise Hessian spectra**
(i.e., principle diag blocks in Hessian)

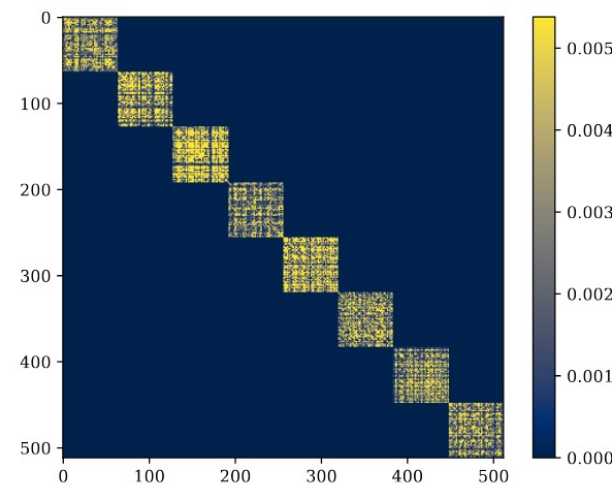
Another reason for studying blockwise Hessian near **Block Diagonal** structure



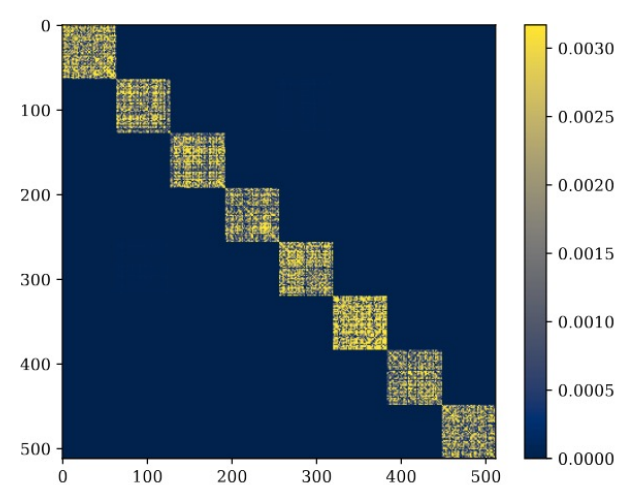
(a) Exact Hessian of a MLP (Collobert, 2004)



(a) Hessian at 5% steps



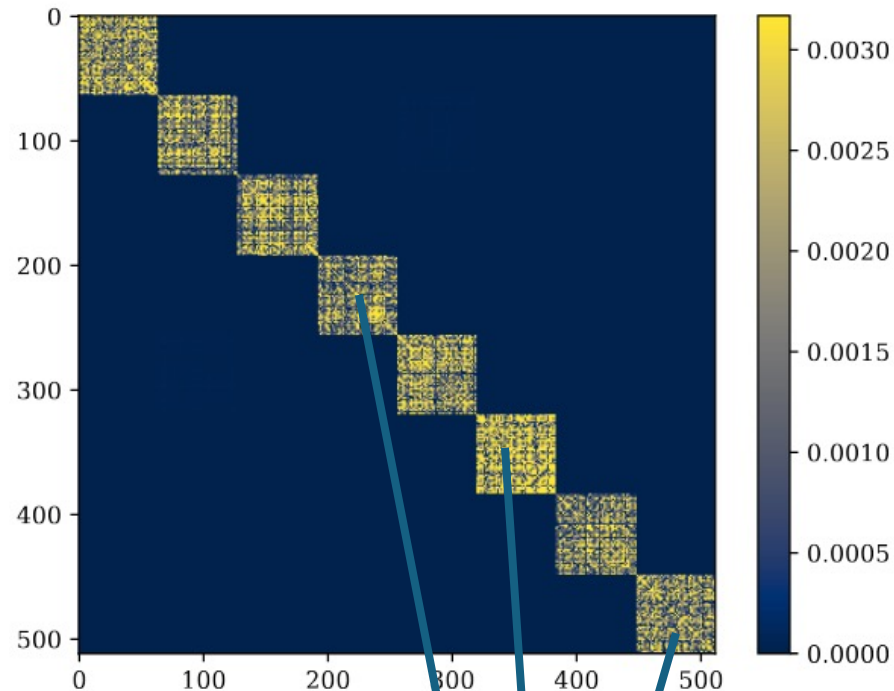
(b) Hessian at 50% steps



(c) Hessian at 100% steps

- Why block-diagonal? No theory so far
- We provide the first theory recently, will show later in **Part III**
- Let us first accept that this structure exists

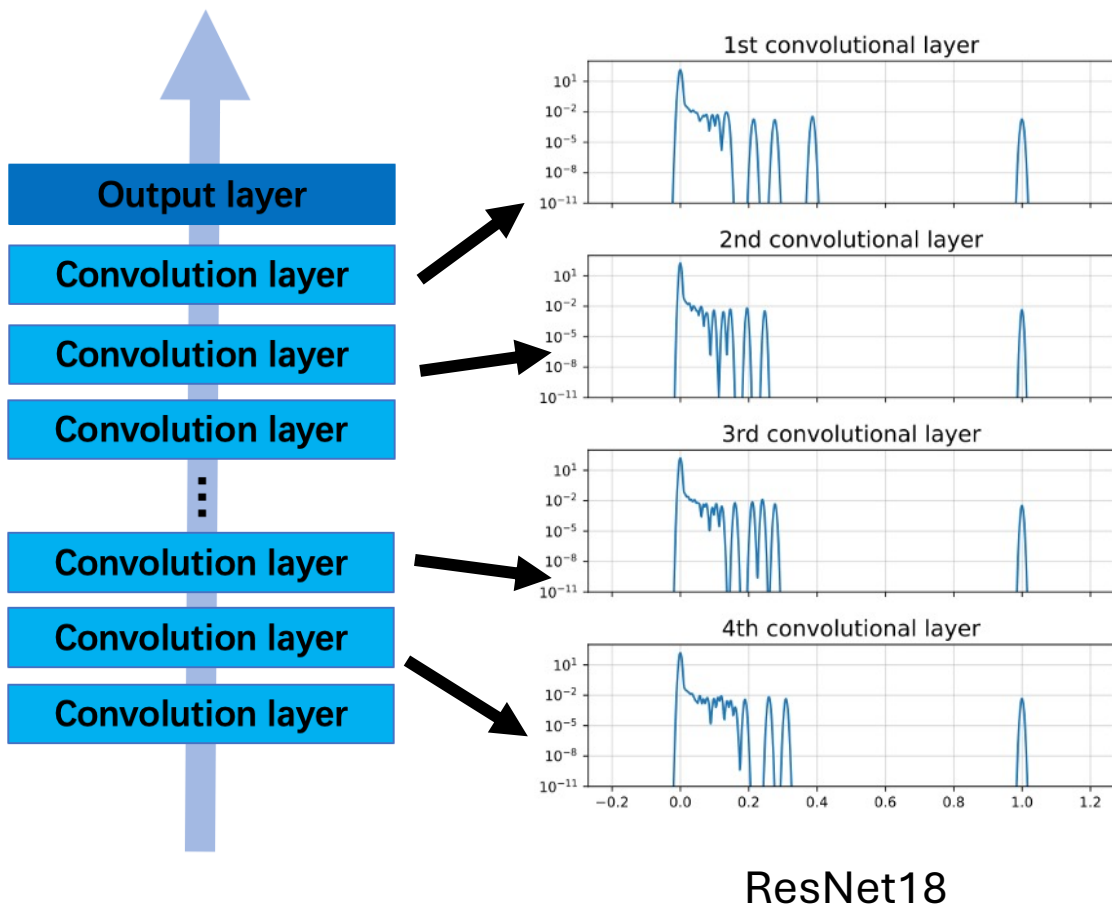
Blockwise Hessian spectrum might matter



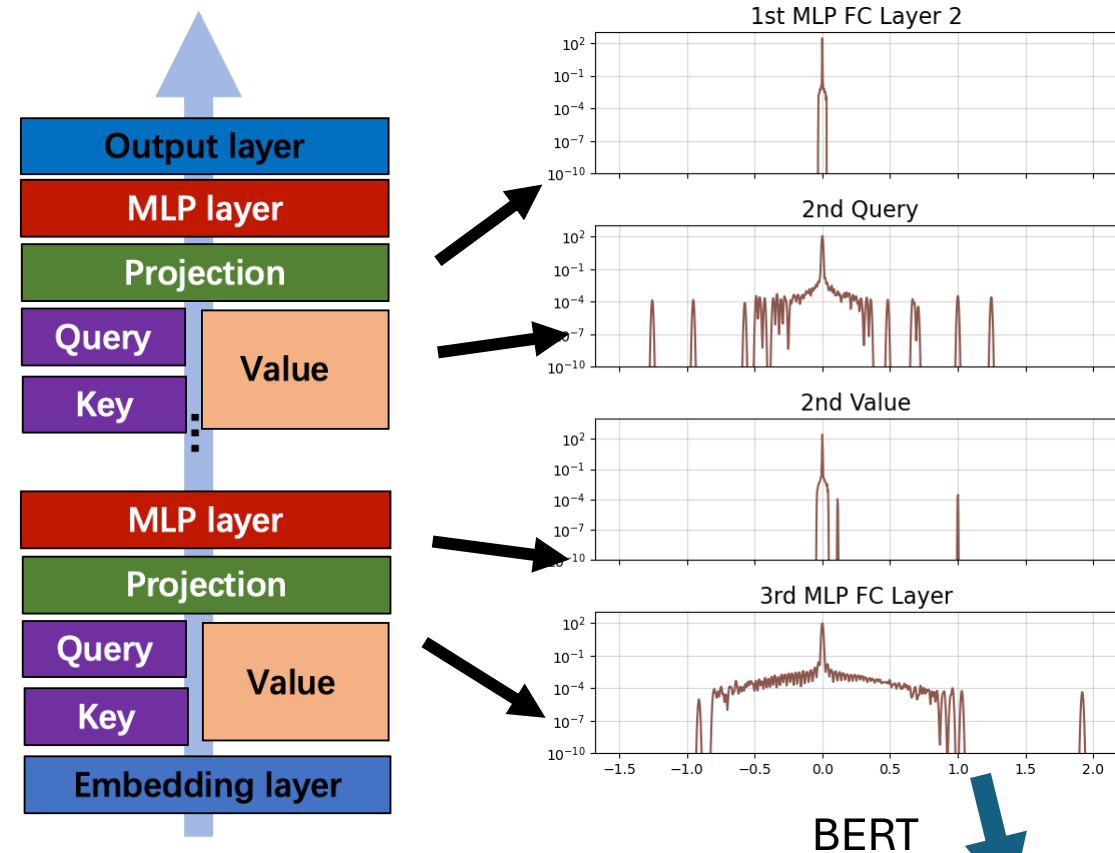
Conjecture: Eigenvalues in **each block (e.g., Q, K, V)** could be important

- What extra info over full spectrum?
- By linear algebra: **location** of eigenvalues

Blockwise Hessian spectrum



CNNs: blockwise spectrum are quite **similar**
We call it ``**homogeneity**”

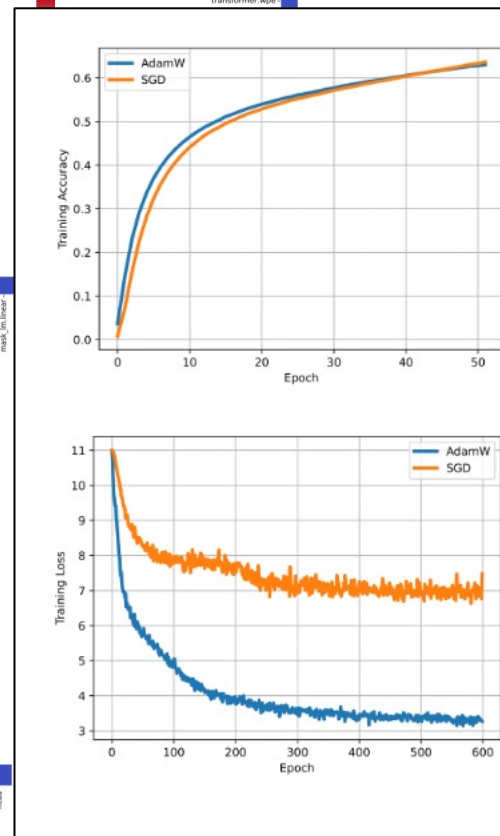
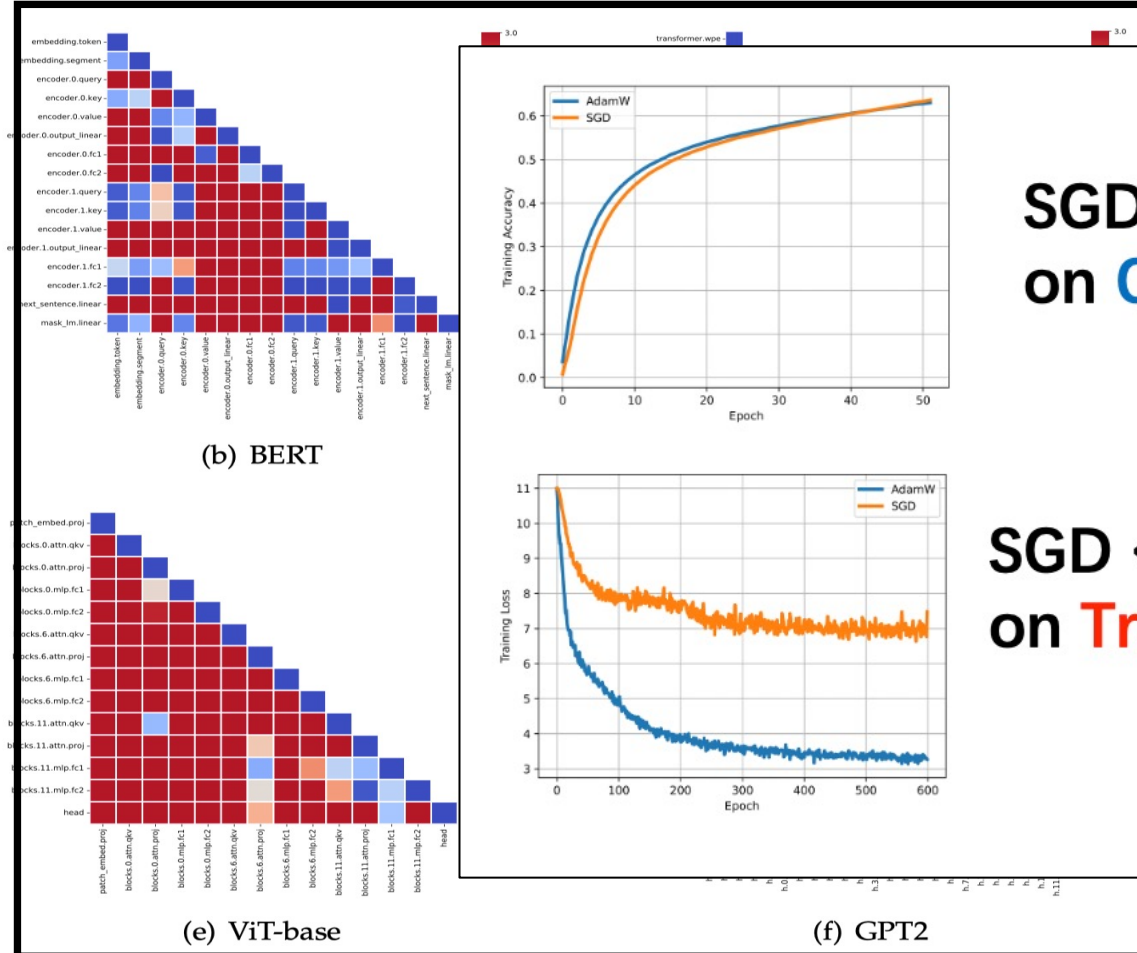
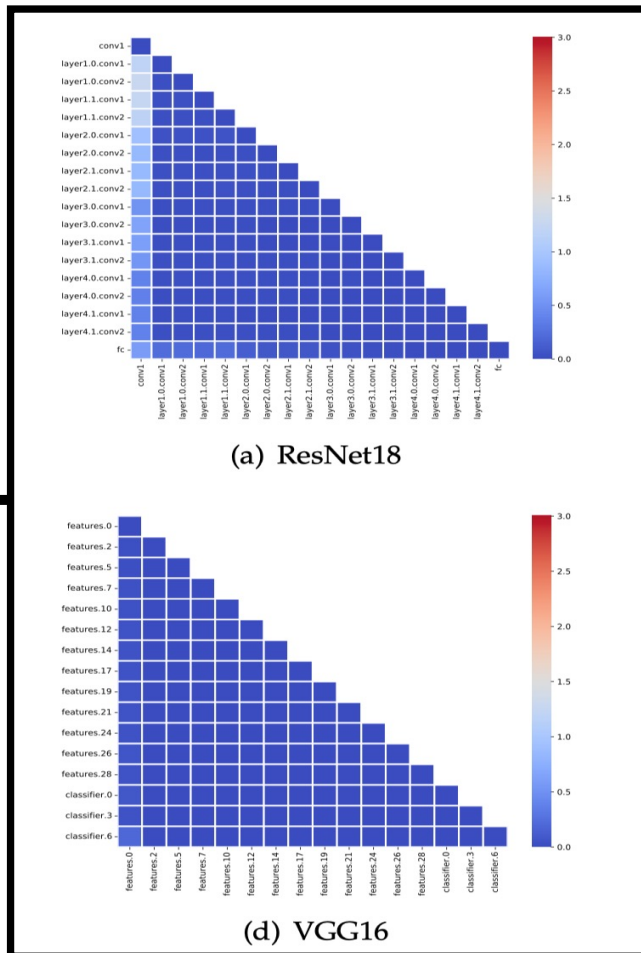


The figure is normalize
(for visualization)
The actual Eigen-range **differs > 200x**

Transformers: blockwise spectrum are largely **different**
We call it ``**heterogeneity**”

JS-distance among blocks

CNNs ←



SGD $\approx$ Adam
on CNNs

SGD $\ll$ Adam
on Transformers

Figure 4: The JS distance among blockwise Hessian spectra for different models at initialization.

Observation: Heterogeneity is widely observed in Transformers, but not on CNNs!

Our Explanation (I): Why Transformer Needs Adam

Observation 1:

Transformer's block Hessian are heterogeneous

Previous part

Observation 2:

If block Hessian are heterogeneous, then SGD worse than Adam.

Next part:

Claim 2A: Heterogeneity causes SGD to underperform Adam

Claim 2B: Homogeneity makes SGD work

Together explains:

Original claim:

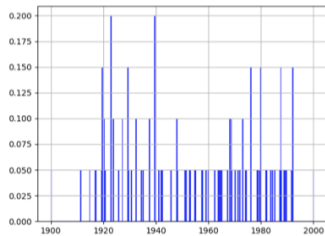
Transformer:
SGD worse than Adam.

Heterogeneity makes SGD worse: Quadratic Prob

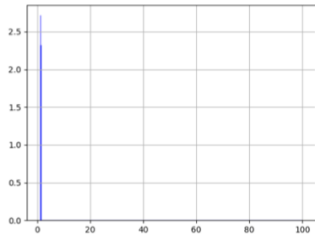
- Case study on quadratic models

$$\min_w \frac{1}{2} w^T H w, H = \text{diag}(H_1, H_2, H_3, H_4), H \text{ is PD}$$

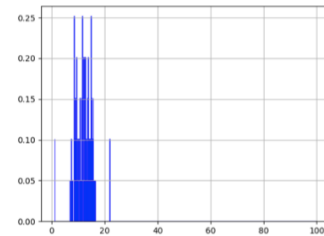
Case 1: H with **Transformer-type spectra**: sampled from GPT2



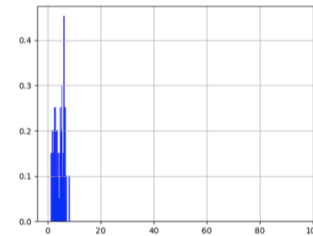
(a) Spectrum of H_1



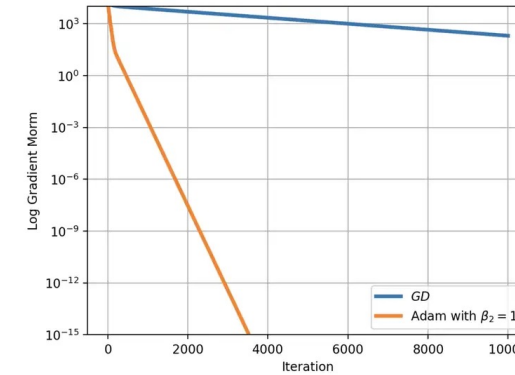
(b) Spectrum of H_2



(c) Spectrum of H_3

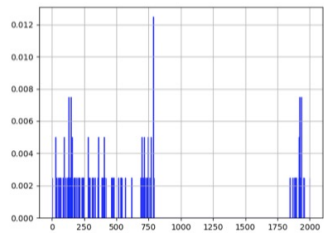


(d) Spectrum of H_4

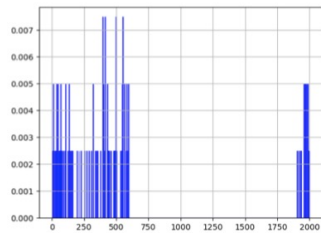


**GD is slower
than Adam**

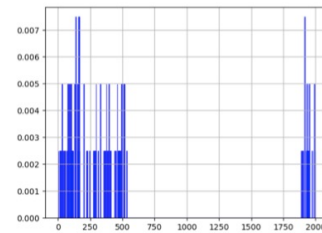
Case 2: H with **CNN-type spectra**: sampled from ResNet18



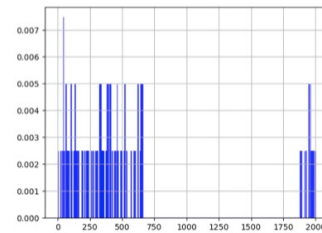
(a) Spectrum of H_1



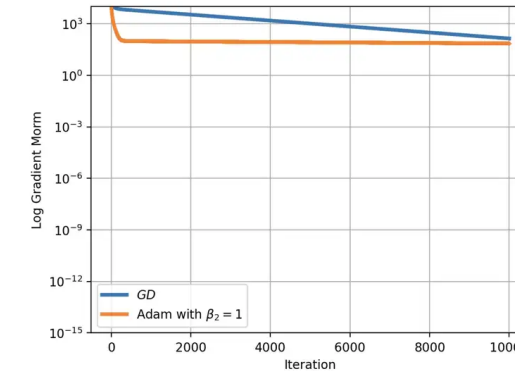
(b) Spectrum of H_2



(c) Spectrum of H_3



(d) Spectrum of H_4



**GD is similar
as Adam**

Remark:

Same condition number for case 1 & 2

but the performance is different, due to **homo & heterogeneity**

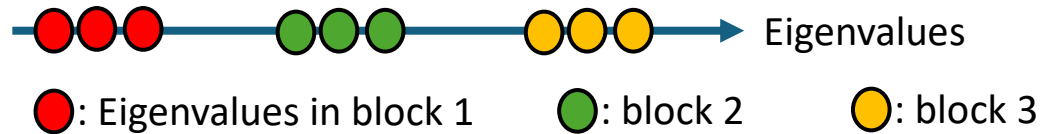
Heterogeneity makes SGD worse: Quadratic Prob

- Case study on quadratic models

$$\min_w \frac{1}{2} w^T H w, H = \text{diag}(H_1, H_2, H_3), H \text{ is PD, } H_l \in \mathbb{R}^{3 \times 3}, l = 1, 2, 3$$

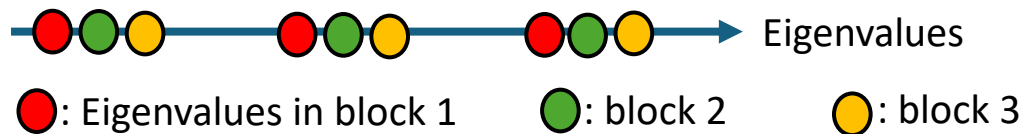
Case 3: H with simplified **heterogeneous** spectra

Eigenvalues of H_l : $\{ 1, 2, 3 \}, \{ 99, 100, 101 \}, \{ 1998, 1999, 2000 \}$



Case 4: H with simplified **homogeneous** spectra

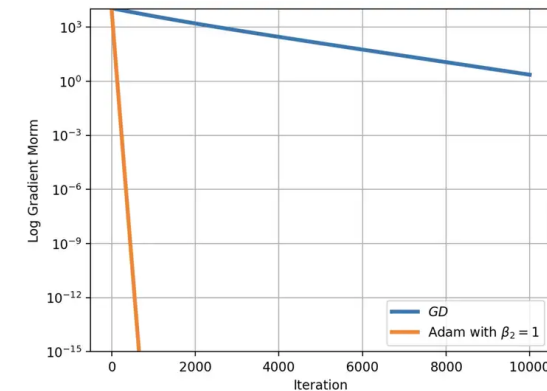
Eigenvalues of H_l : $\{ 1, 99, 1998 \}, \{ 2, 100, 1999 \}, \{ 3, 101, 2000 \}$



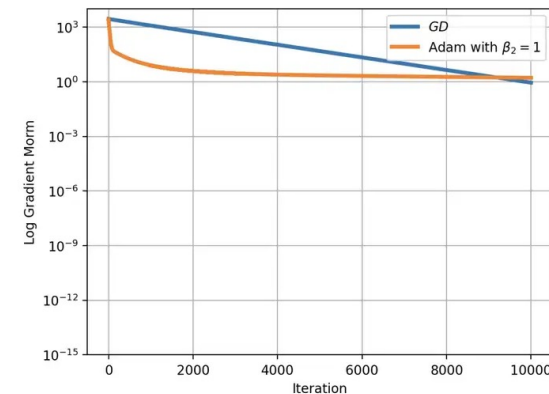
Remark:

All eigenvalues are the SAME for case 3 & 4

but the performance is different, due to **homo & heterogeneity**



GD is slower than Adam



GD is similar as Adam

Theoretical results

A well-known result for GD: Consider $\min_w f(w) = \frac{1}{2} w^T H w$, where H is PD, then there exists a H and w^0 :

$$f(w_{GD}^{t+1}) - f^* \geq \left(1 - \frac{2}{\kappa}\right) (f(w_{GD}^t) - f^*)$$

where κ is the condition number of H

Theorem 1 (Adam): Consider $\min_w f(w) = \frac{1}{2} w^T H w$, where H a block-diagonal PD matrix with L blocks, then:

$$f(w_{Adam}^{t+1}) - f^* \leq \max_{l \in [L]} \left(1 - \frac{1}{\kappa_{Adam,l}}\right) (f(w_{Adam}^t) - f^*)$$

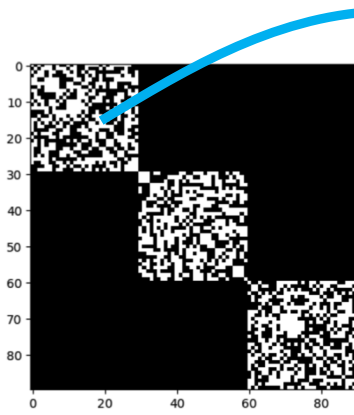
where $\kappa_{Adam,l} = r \kappa_l$, κ_l is the condition number of H_l ; r is a constant related to initialization w^0

Comparing Two Results

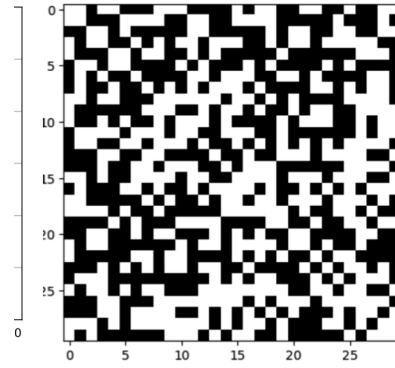
- Compare GD vs Adam:
 - Adam is faster than GD when $r \max_l \kappa_l \leq \kappa$,
- Happens in the heterogeneous quadratic examples 1 & 2
 - Quantity of Adam ~20x smaller, and Adam is also 20x faster
- This provides a partial theoretical explanation why Adam works better than SGD on Heterogeneous case

Effectiveness of D_{Adam} : Dense Hessian

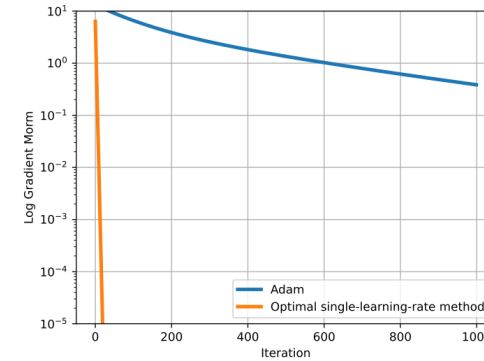
- $\min_x \frac{1}{2} x^T H_1 x$, where H_1 is a random matrix.



(a) Hessian matrix



(c) Dense sub-block



(d) Sub-problem loss

- **Another Case: Dense block.**

We take the 1st sub-block H_1 in (a) and use it for a new problem (c)

- Figure (c): **GD with (a different) optimal lr** outperforms **Adam**, even though Adam uses more lr!
- This means: D_{Adam} is NOT so effective on H_1 !

Our Explanation (II): Why Transformer Needs Adam

Observation 1:

Adam is good for block-Hessian.

Previous Part 1

Observation 2:

Adam is bad for dense Hessian

Previous slide

Together reveals:

Implication:

Adam mainly handles “cross-block” challenge,
NOT “within-block” challenge.

在第二篇文章 Adam-mini 中，
这个解释才补全。

Further Implication:

We may replace Adam’s coordinate-wise lr by block-wise lr

Part 1-2

Related Work on Why Adam better than SGD

- One perspective [J. Zhang et al. 19]

NLP problems exhibit heavy-tailed noise in g  SGD fails

- However, [Chen, Kunstner, Schmidt'21] provides negative evidence:

SGD is worse than Adam on Transformers, even in full-batch case
(with no stochasticity)

**Heavy-tailed noise does not explain the gap between
SGD and Adam on Transformers**

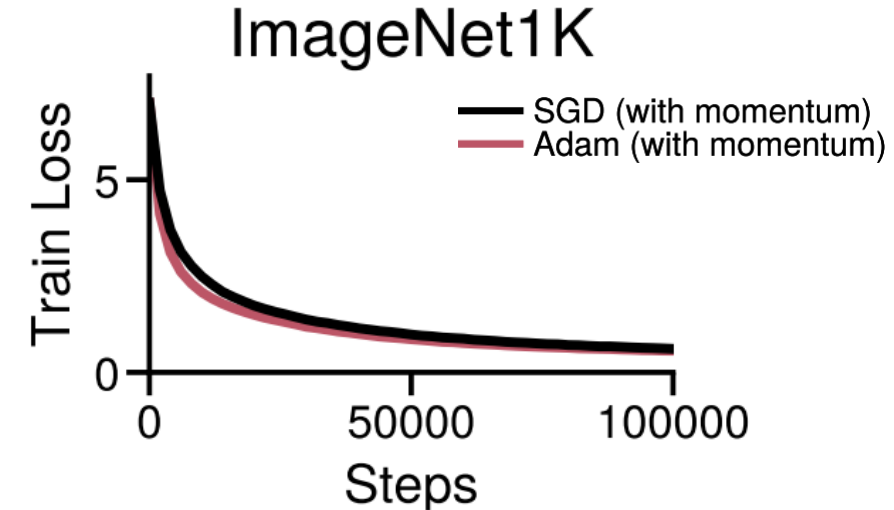
Jacques Chen, Frederik Kunstner, Mark Schmidt
University of British Columbia

- So there may be other reasons...

Related Work on Why Adam better than SGD

- Schmidt et al.'s work: class imbalance causes the gap

	Balanced Data	Imbalanced Data
ResNet18	Adam ~ SGD	Adam > SGD
ViT	Adam ~ SGD	Adam > SGD



➡ Strong advantage of Adam over SGD is largely due to class imbalance.

Our Point: Special Case of Hessian Explanation

Ongoing work: this is a special case of our explanation

	Balanced Data	Imbalanced Data
ResNet18 (Hessian Homo.)	Adam ~ SGD	Adam > SGD
ViT (Hessian Hetero.)	Adam > SGD	Adam > SGD



- 1) Imbalanced data create Hetero. Hessian blocks in last layer (classifier)
- 2) Both architecture and imbalanced data contribute to Hessian heterogeneity, leading to the gap between Adam and SGD.

Contents

- Part I-2: Adam-mini and Block Partition Principle

Another Trick: Use Mean of 2nd Momentum

But how to find these good block-wise lrs without grid-search?

We tried constant block-lr method for Transformer:

- hard to make it work
- maybe it exists, but searching one lr for each block is costly!

Our solution (simple and cheap)

Use **block-mean of 2nd-order momentum** term
to compute block-wise lr

Adam-mini (Framework)

- **Adam-mini:**

- **Step 1:** partition the gradient g into B sub-vectors **according to the dense Hessian sub-block** $g_b, b = [B]$
- **Step 2:** for each g_b , calculate:

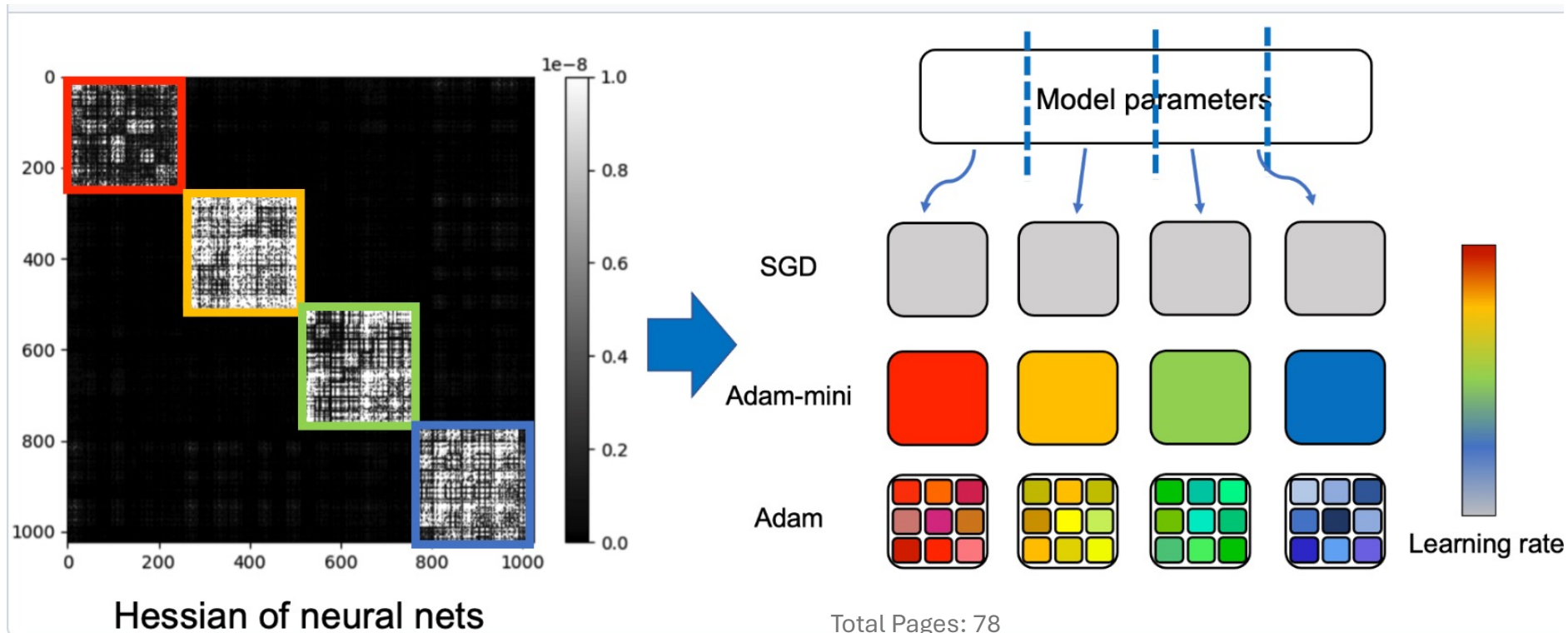
$$v_b = (1 - \beta_2) * \text{mean}(g_b \circ g_b) + \beta_2 * v_b, \quad b = 1, \dots, B$$

- **Step 3:** then use $\frac{\eta}{\sqrt{v_b}}$ as the lr for the parameters associated with g_b

Partition Principle I

Partition Principle I (Basic):

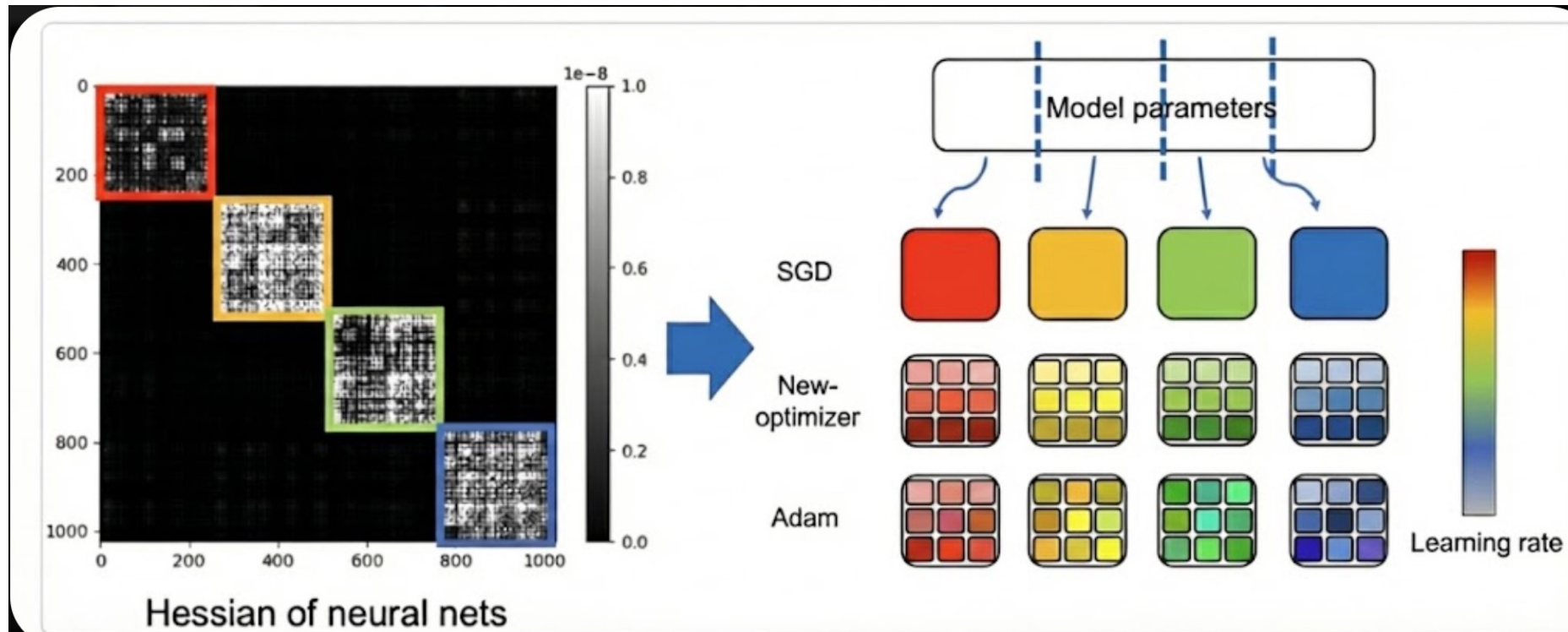
Partition parameters into blocks s.t.
each block is associated with a **dense sub-block** in Hessian.



Partition Principle II: Generalized

Partition Principle II (Generalized):

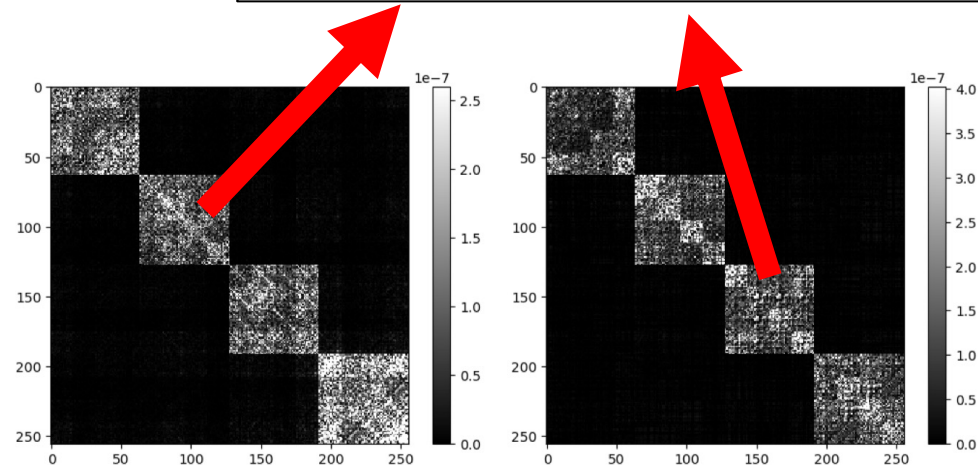
Partition parameters into blocks s.t.
each block is a **sub-block** of a **dense sub-block** in Hessian.



Blocks of Transformer Hessian

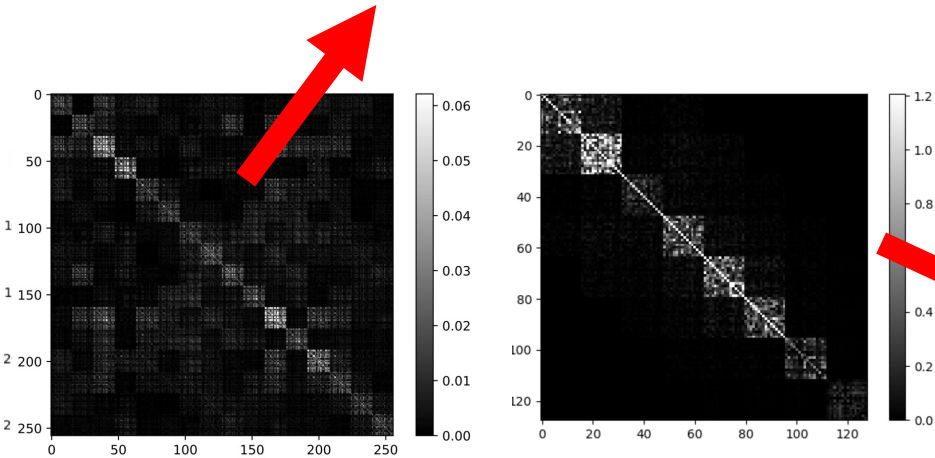
#blocks = **#attention heads**

Less clear (treat as **#output neurons**)



(a) query (4 heads)

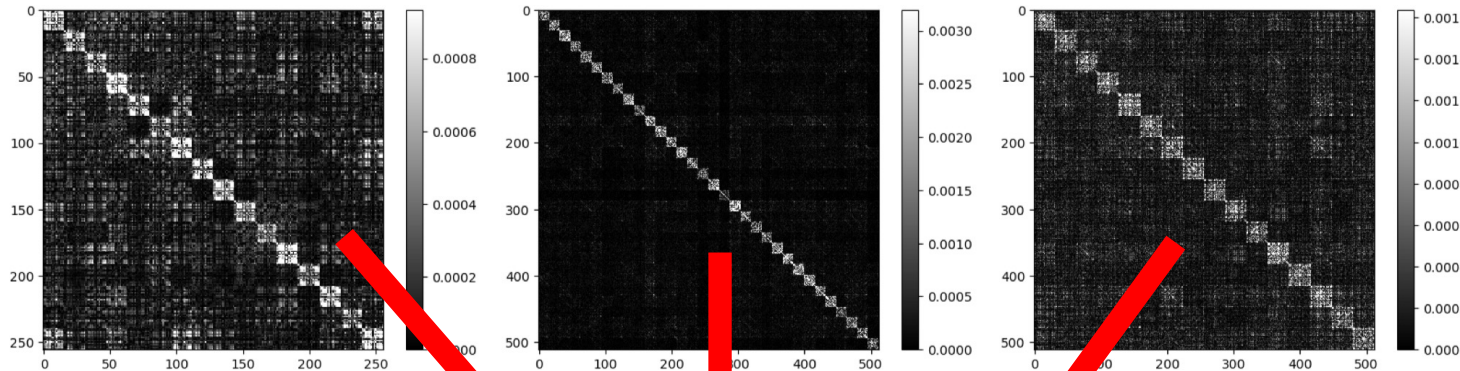
(b) key (4 heads)



(c) value (4 heads)

Embedding layer (8 tokens)

#blocks = **#tokens**



(d) attn.proj (16 neurons)

(e) mlp.fc1 (32 neurons)

(f) mlp.proj (16 neurons)

Output layer (8 tokens)

#blocks = **#output neurons**

Transformer Partition Rules

Partition Principle I:

Partition parameters into blocks s.t. each block is associated with a **dense sub-block** in Hessian.

Partition Rule I for Transformers:

Rule 1: Each head of Q&K layers is a block.

Rule 2: For other layers, each neuron is a block.

Partition Principle II (Generalized):

Partition parameters into blocks s.t.
each block is a **sub-block** of a **dense sub-block** in Hessian.

Partition Rule II for Transformers:

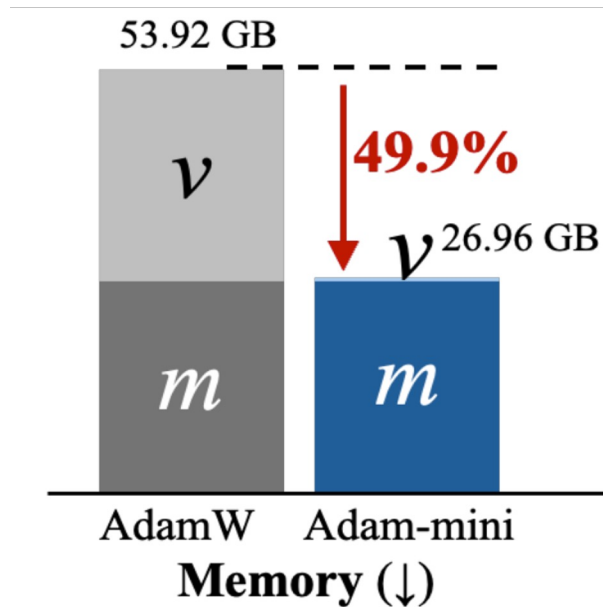
Rule: For each layers, each neuron is a block.

是 Adam-mini 划分方法
的自然推广
(原文中没有写)

Major Differences with Layer-wise Methods

- Some methods, like LARS, LAMB use layer-wise lr.
- Adam-mini is quite different from them in two aspects:
 - **First, principled block-partition:**
 - **Adam-mini:** blocks decided by Hessian spectrum structure
 - **Layer-wise partition:** not working for 1B training
 - **Second, Adam-inherited block-lr.**
 - **Adam-mini:** uses using block-mean of Adam's original lr (by 2nd order momentum)
 - **Other layerwise methods:** not borrowing from Adam

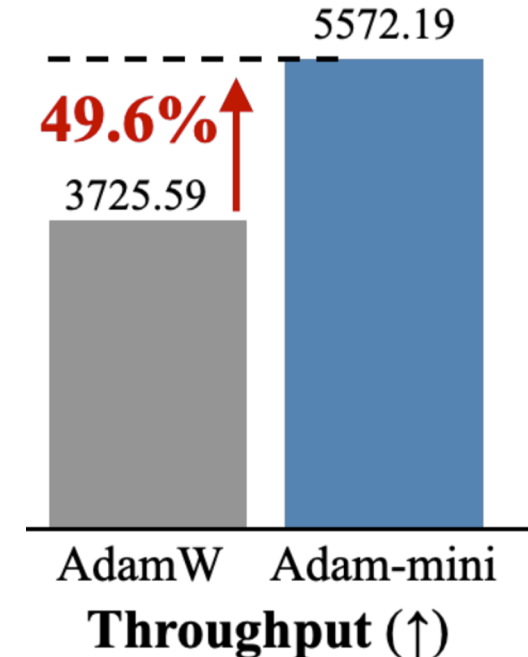
Memory Reduction and Throughput Enhancement



Saves about **50%** memory of Adam

Optimizer states in Llama2-7b

	#m	#v	mem(m)	mem(v)
AdamW	6.74B	6.74B	26.95GB	26.95GB
Adam-mini	6.74B	1.16M	26.95GB	4.65MB



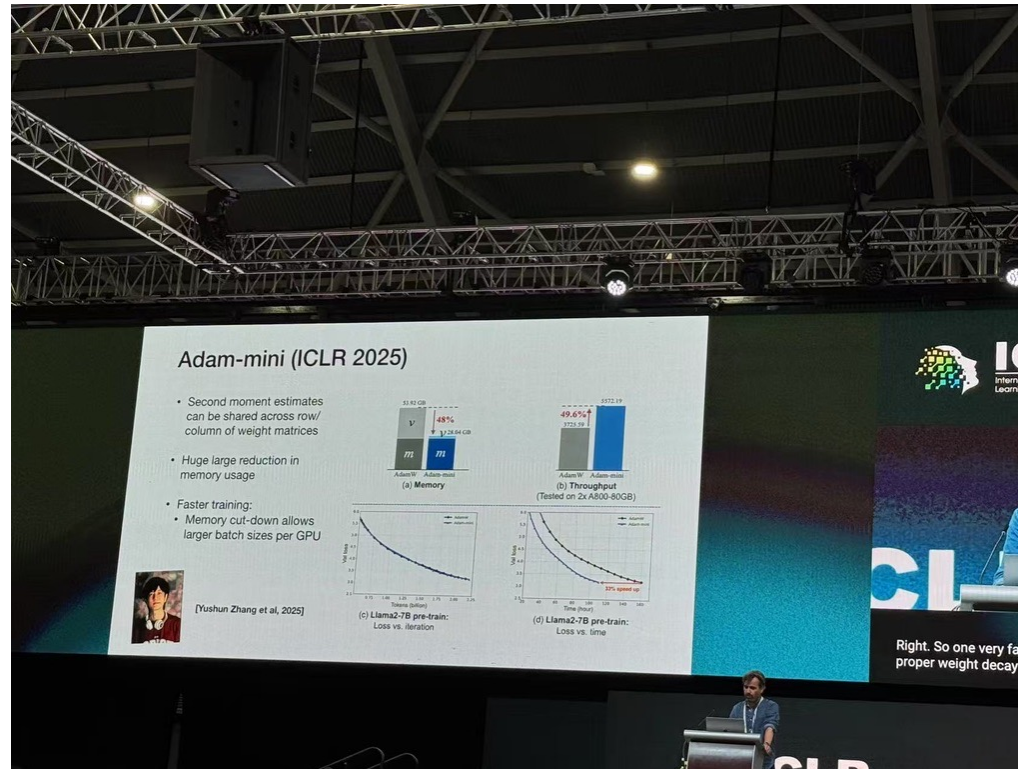
Can increase about **50%** throughput of Adam
(# processed data per second)

Why?

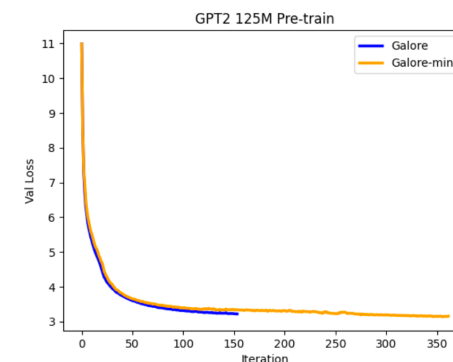
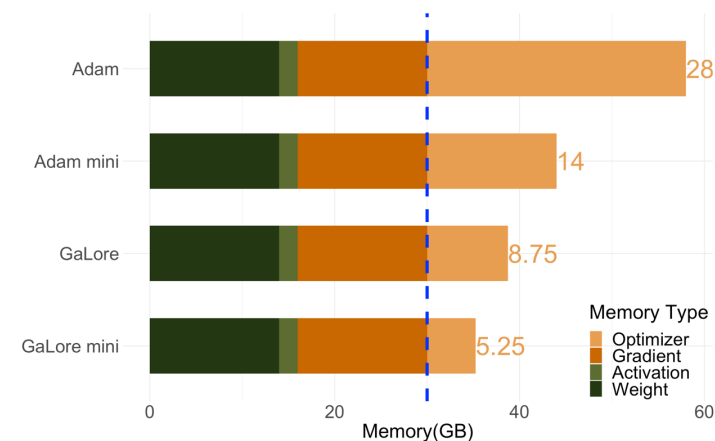
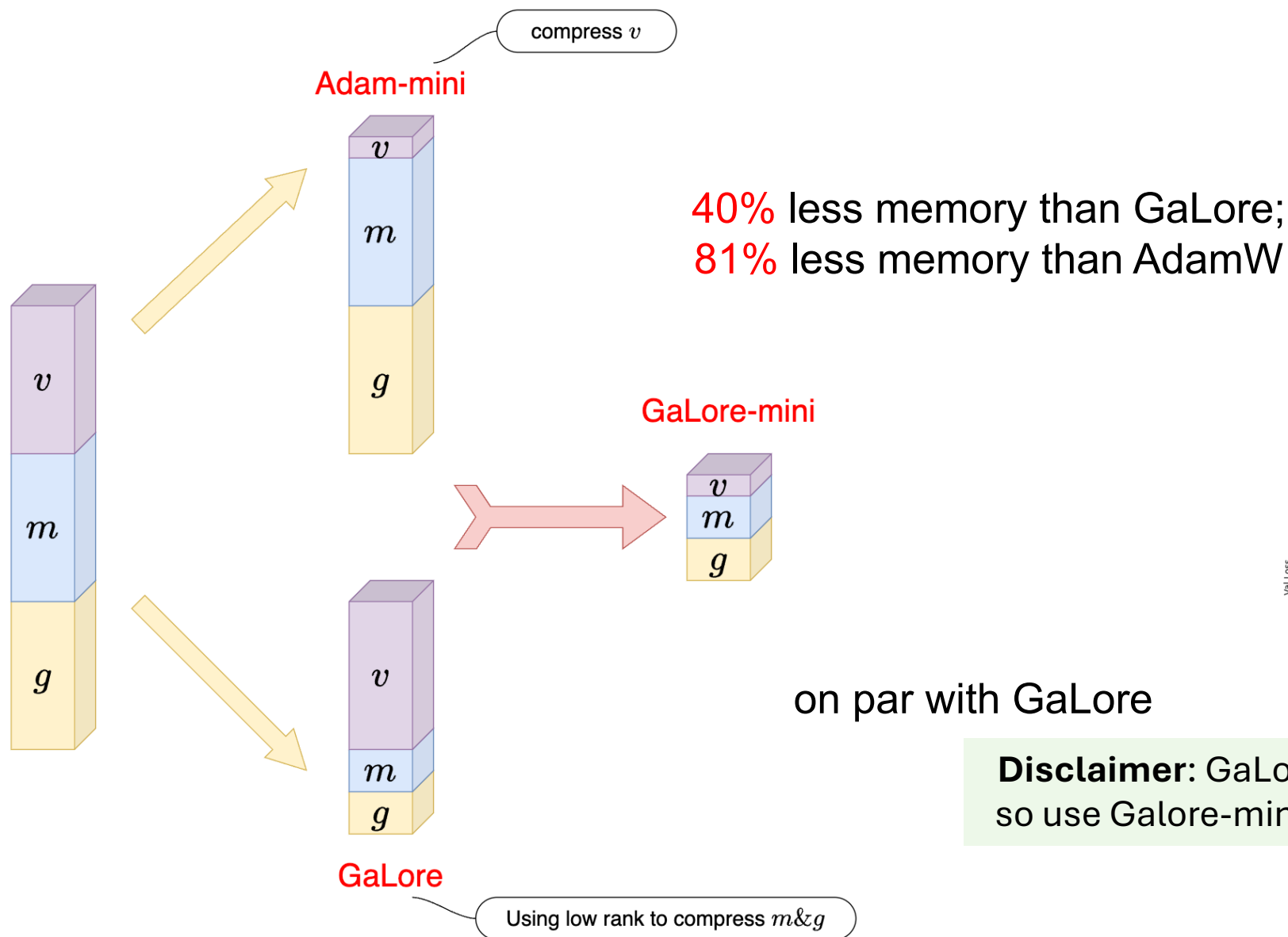
Reduce communication + larger per-GPU batch size

Acknowledgements from Authors of Adam

- Photo shot at **ICLR 2025 Test of Time Speech** by Dr. Durk Kingma and Prof. Jimmy Ba
- “This work allows you to reduce the memory of Adam by a large factor ...
This is, I think, a great result that argued from theory ”



Follow-up Work: GaLore-mini



on par with GaLore

Disclaimer: GaLore's throughput is lower than Adam; so use GaLore-mini if you are OK with lower throughput

Follow-up Work: Q-Adam-mini

Q-Adam-mini: Memory-Efficient 8-bit Quantized Optimizer for Large Language Model Training

Yizhou Han^{*12} Chaohao Yang^{*1} Xingjian Wang¹² Congliang Chen¹²
Ruoyu Sun¹²

量化Adam-mini 中的 m 为 8-bit
进一步降低优化器显存.

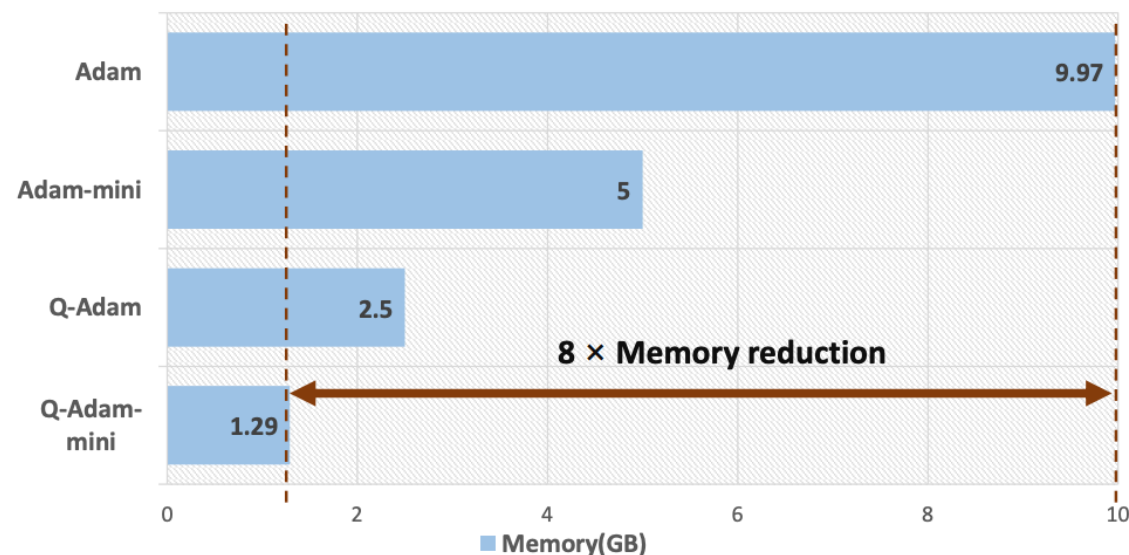
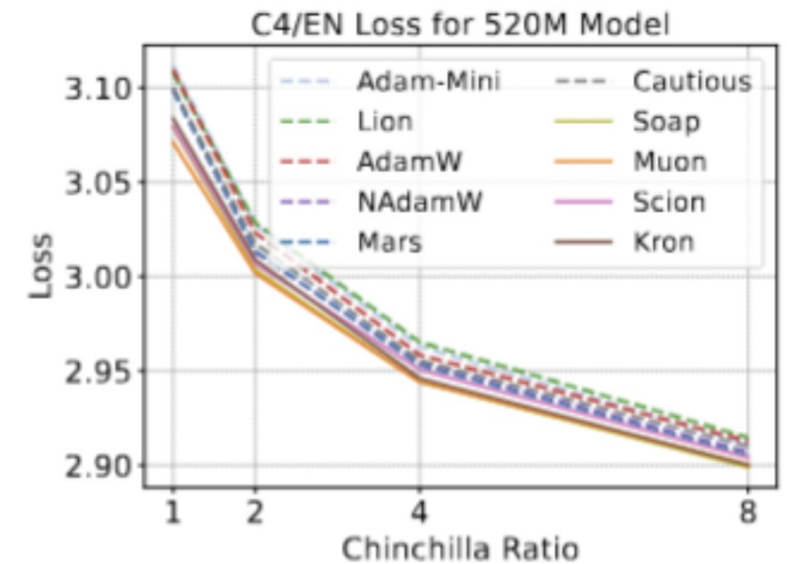
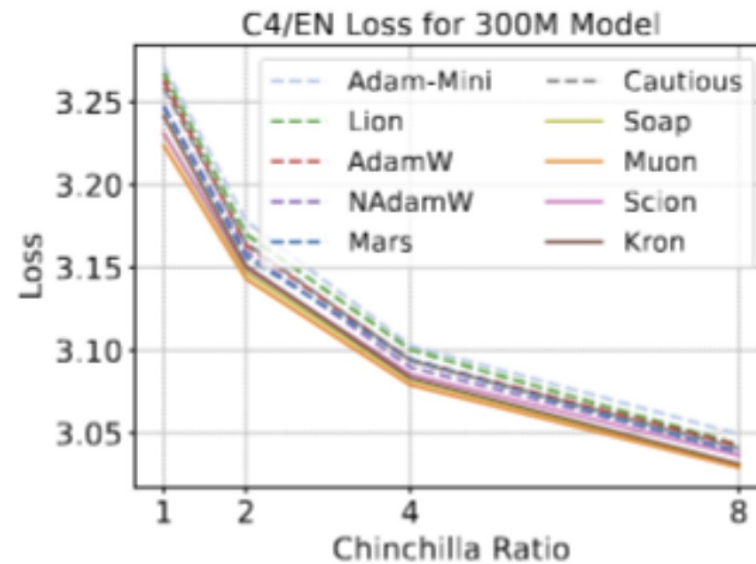
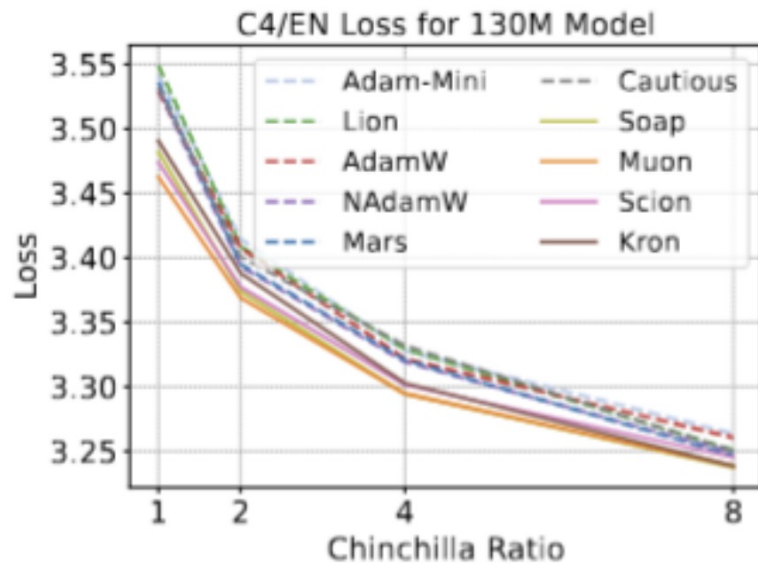


Figure 1. Optimizer states' memory consumption for pre-training Llama-1B

Recent Benchmark by Stanford

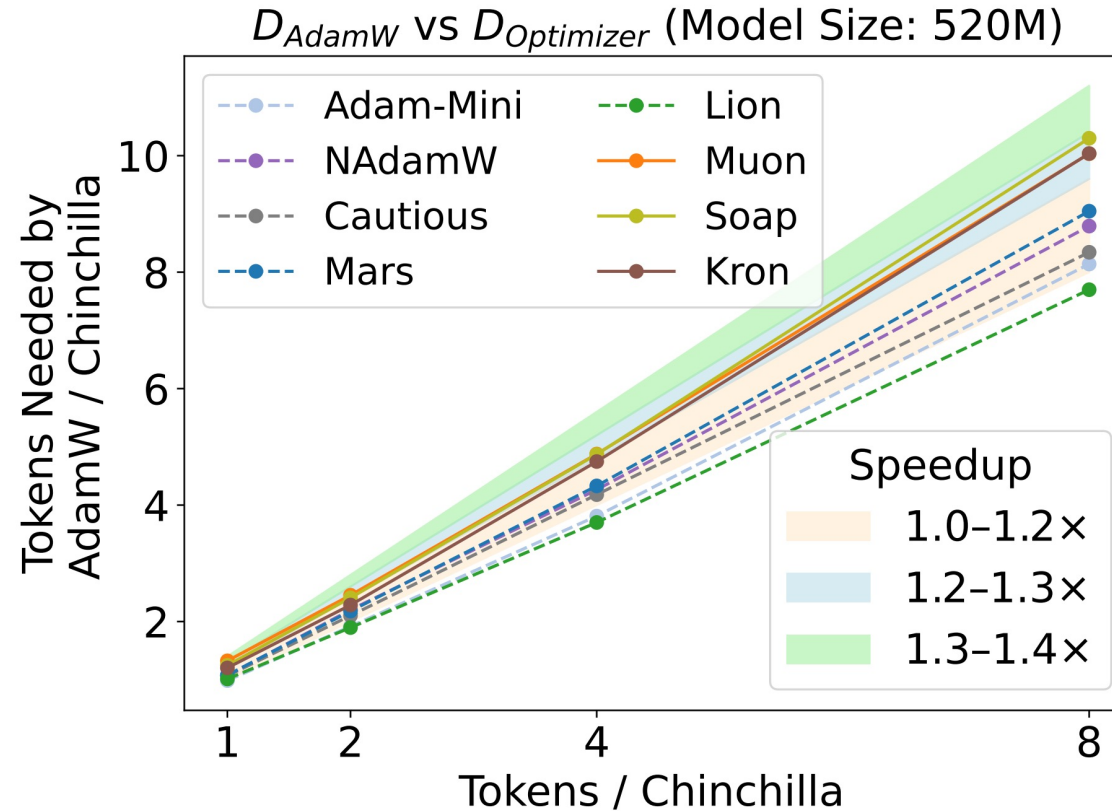
- 130/ 300/ 520M model with 1x / 2x / 4x / 8x Chinchilla token amount
- Full hyperparameter sweep
- Adam-mini performs on par with AdamW



<https://wandb.ai/marin-community/marin/reports/Fantastic-Optimizers-and-Where-to-Find-Them--VmldzoxMjgzMzQ2NQ>

Recent Benchmark by Stanford

- On 520M model, Adam-mini is **faster than AdamW**
(unexpected benefit)



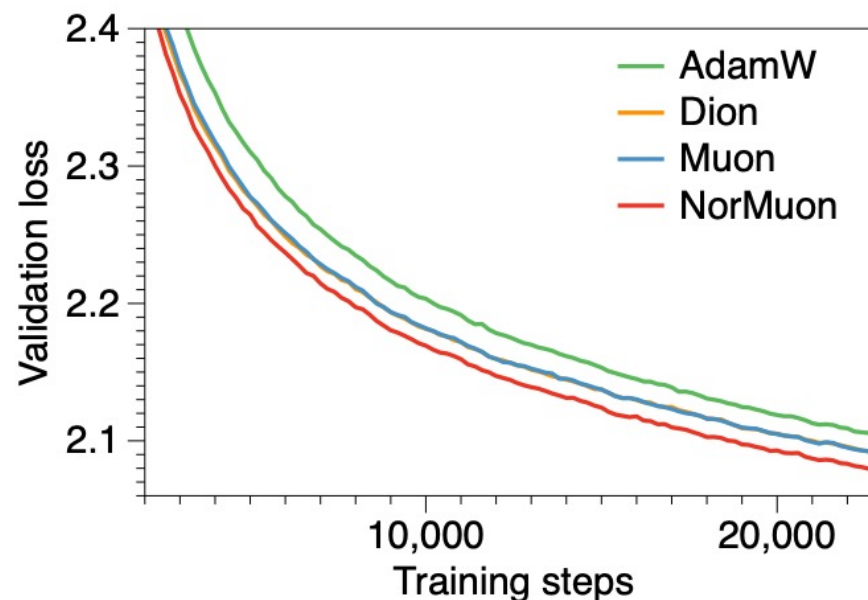
Follow-up Work: NorMuon

NorMuon (by Microsoft and GaTech)

Combine the **block-wise scaling idea of Adam-mini (neuron-wise)** and **muon**. It takes the crown as the leading NanoGPT speedrun optimizer!

Algorithm 1 NorMuon

```
1: Input: Initial weights  $\mathbf{W}_0 \in \mathbb{R}^{m \times n}$ , loss  $L$ , learning rate  $\eta$ , max  
   perturbation parameter  $\varepsilon$ , weight decay  $\lambda$ .  
2: Initialize  $\mathbf{M}_0 \in \mathbb{R}^{m \times n} \leftarrow \mathbf{0}$ ,  $\mathbf{v}_0 \in \mathbb{R}^m \leftarrow \mathbf{0}$   
3: for  $t = 1, 2, \dots$  do  
4:    $\mathbf{G}_t \leftarrow \nabla_{\mathbf{W}} L(\mathbf{W}_t)$   
5:    $\mathbf{M}_t \leftarrow \beta_1 \mathbf{M}_{t-1} + (1 - \beta_1) \mathbf{G}_t$   
6:    $\mathbf{O}_t \leftarrow \text{NS5}(\mathbf{M}_t)$   
7:    $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \text{mean}_{\text{cols}}(\mathbf{O}_t \odot \mathbf{O}_t)$   
8:    $\mathbf{V}_t \leftarrow \text{ExpandRows}(\mathbf{v}_t)$   
9:    $\widehat{\mathbf{O}}_t \leftarrow \mathbf{O}_t \odot (\sqrt{\mathbf{V}_t} + \varepsilon)$   
10:   $\hat{\eta} = 0.2\eta\sqrt{mn}/\|\widehat{\mathbf{O}}_t\|_F$   
11:   $\mathbf{W}_{t+1} \leftarrow \mathbf{W}_t - \eta\lambda\mathbf{W}_t - \hat{\eta}\widehat{\mathbf{O}}_t$   
12: end for
```



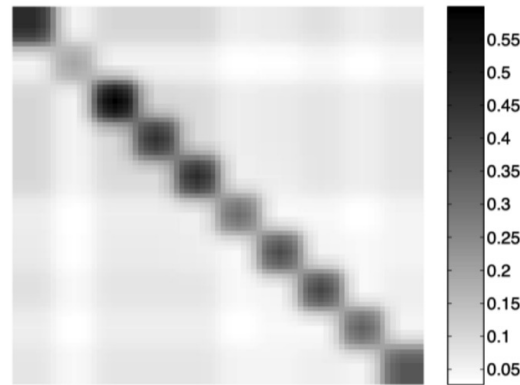
(a) Pretraining results of 1.1B model.

Contents

- Part I: Why Transformers Need Adam
- **Part II: On the Special Structure of Hessian of NNs**

Empirical Observation I

- Hessian of NNs: nearly block-diagonal



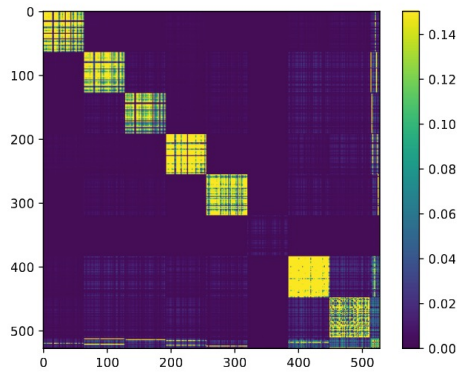
(a) Hessian of an MLP
[18] after 1 step

Hessian of a 1-hidden-layer NN

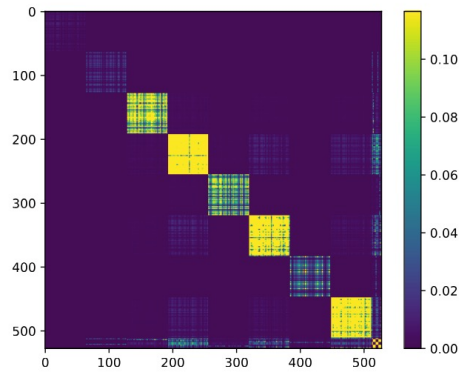
Figure from: Large Scale Machine Learning, Collobert, thesis, 2004

Empirical Observation II

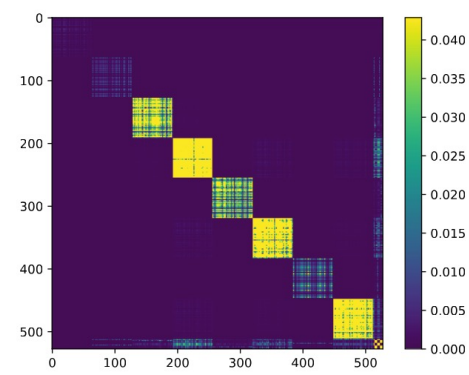
- Hessian of NNs: nearly block-diagonal



(b) Hessian of an MLP at 1% step



(c) Hessian of an MLP at 50% step



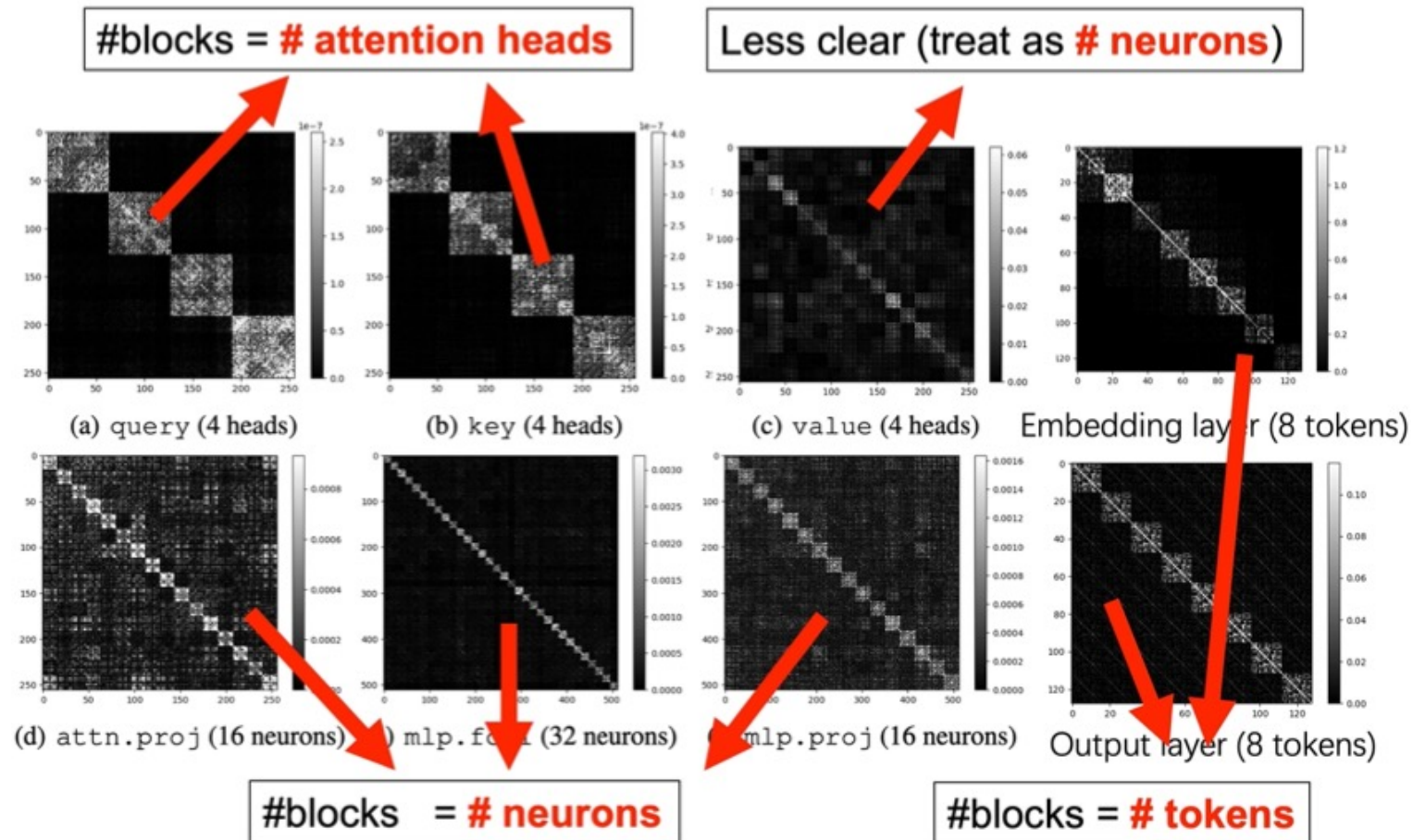
(d) Hessian of an MLP at 100% step

Hessian of 1-hidden-layer NNs

Figure (b,c,d): Why Transformers Need Adam: A Hessian Perspective, Zhang, Chen, Ding, Li, Sun*, Luo, NeurIPS 2024

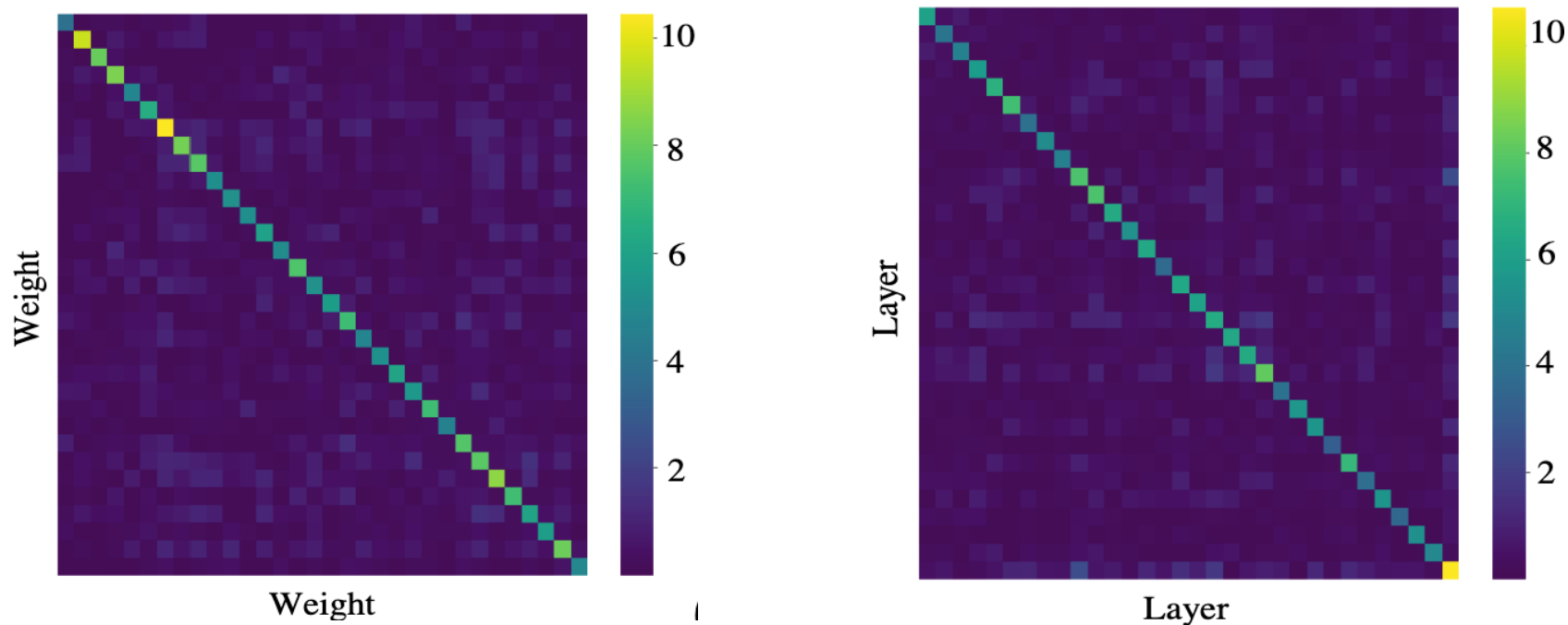
Empirical Observation III

Hessian of Transformers: nearly block-diagonal



Empirical Observation IV

- **Hessian of LLMs:** nearly block-diagonal



Approximated Hessian of 1 layer in Llama-7B & 32 layers in Llama-7B

Figure from: CBQ: Cross-Block Quantization for Large Language Models, Ding, et al., ICLR 2025

Motivation: Why Studying Hessian Structure?

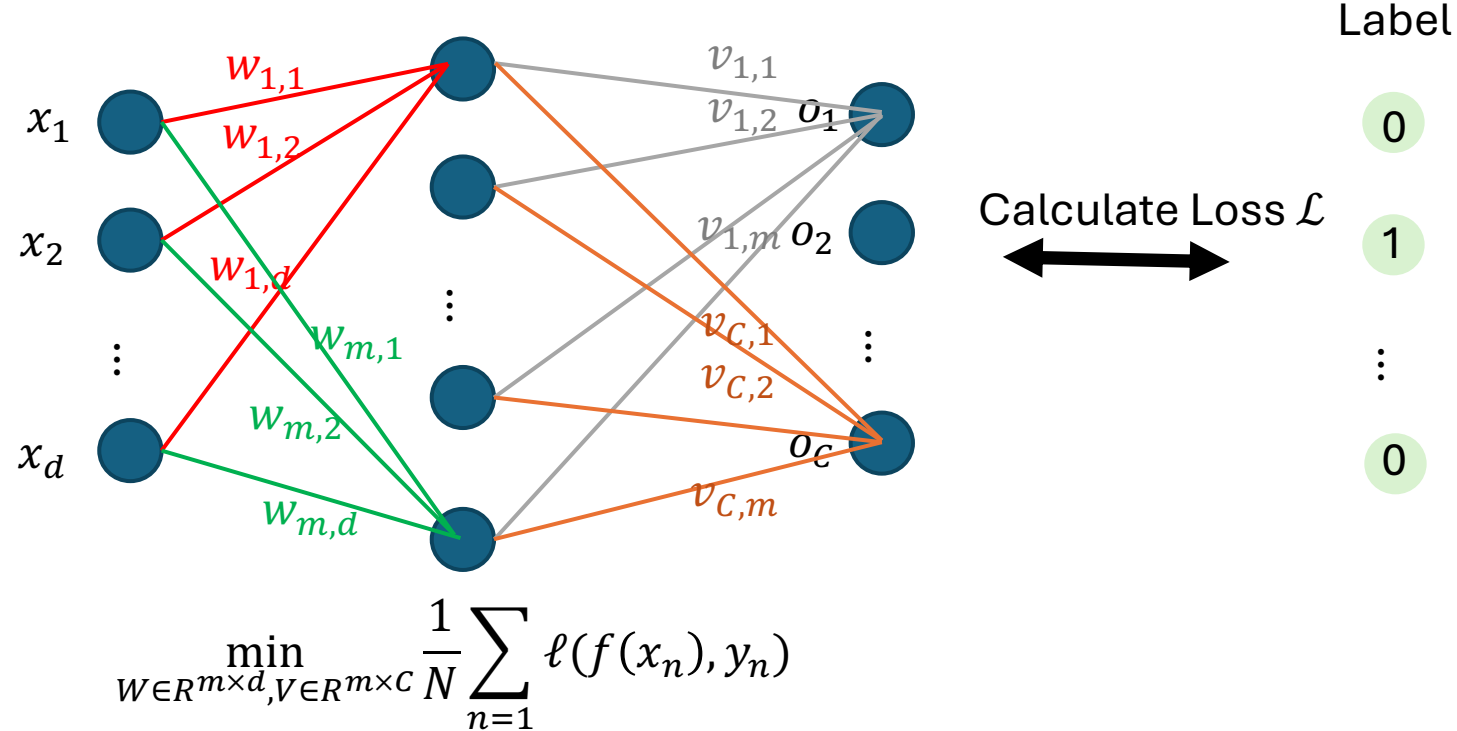
- 1. Hessian structure is crucial for understanding NN training
 - The effectiveness of **Adam** (Zhang et al 24a, Kunstner et al. 24)
 - The effectiveness of general **diagonal-preconditioned methods** (Sun and Ye, 21, Qu et al. 22, Das et al. 24)
 - The effectiveness of recent **block-diagonal-preconditioned methods** (Shampoo, Muon)
- 2. Hessian structure can help design new training methods for NNs
 - Recently, Adam-mini utilizes the block-diag structure to cut down 50% memory in Adam
 - More is coming..
- 3. **Offering a new function class for optimization community**
 - Typical problems do NOT have such structure:
In classical non-linear programming dataset (Lavezzi et al 22), all problems have non-block-diag Hessian
 - Motivate new study into this specific class of problems

We focus on...

- Why do Hessian matrices look like this? Is it trivial?
- What does one block correspond to?
- Does it always hold for arbitrary NNs?
 - If not, is there common factor holds in all above, but we overlooked?
- Any more structure missed in the previous experiments?

Review: What is Hessian Matrix for NNs

Data: $x \in R^d, y \in R^C$ one-hot



$$W = \begin{bmatrix} w_1^T \\ \vdots \\ w_m^T \end{bmatrix} \in R^{m \times d}, V = \begin{bmatrix} v_1^T \\ \vdots \\ v_C^T \end{bmatrix} \in R^{C \times m}$$

$$f(x_n) = \begin{bmatrix} v_1^T \sigma(Wx_n) \\ \vdots \\ v_C^T \sigma(Wx_n) \end{bmatrix} \in R^C$$

$$\min_{W, V} \ell_{\text{MSE}}(W, V) := \frac{1}{N} \sum_{n=1}^N \|V \sigma(Wx) - \mathcal{Y}_n\|_2^2,$$

$$\min_{W, V} \ell_{\text{CE}}(W, V) := -\frac{1}{N} \sum_{n=1}^N \log \left(\frac{\exp(v_{y_n}^\top \sigma(Wx))}{\sum_{k=1}^C \exp(v_k^\top \sigma(Wx))} \right)$$

Review: What is Hessian Matrix for NNs

	d	d	d	m	m	m
d	$H_{w_1 w_1}$	$\cdots$	$H_{w_1 w_m}$	$H_{w_1 v_1}$	$\cdots$	$H_{w_1 v_C}$
d		$\ddots$				
d	$H_{w_m w_1}$	$\cdots$	$H_{w_m w_m}$	$H_{w_m v_1}$	$\cdots$	$H_{w_m v_C}$
m	$H_{v_1 w_1}$	$\cdots$	$H_{v_1 w_m}$	$H_{v_1 v_1}$	$\cdots$	$H_{v_1 v_C}$
m		$\ddots$			$\ddots$	
m	$H_{v_C w_1}$	$\cdots$	$H_{v_C w_m}$	$H_{v_C v_1}$	$\cdots$	$H_{v_C v_C}$

Size of Hessian = $(md + Cm) * (md + Cm)$

$$W = \begin{bmatrix} w_1^T \\ \vdots \\ w_m^T \end{bmatrix} \in R^{m \times d}, V = \begin{bmatrix} v_1^T \\ \vdots \\ v_C^T \end{bmatrix} \in R^{C \times m}$$

$$H_{w_i w_i} = \frac{\partial^2 \mathcal{L}}{\partial w_i \partial w_i^T} \in R^{d \times d}$$

$$H_{w_i w_j} = \frac{\partial^2 \mathcal{L}}{\partial w_i \partial w_j^T} \in R^{d \times d}$$

$$H_{v_i v_i} = \frac{\partial^2 \mathcal{L}}{\partial v_i \partial v_i^T} \in R^{m \times m}$$

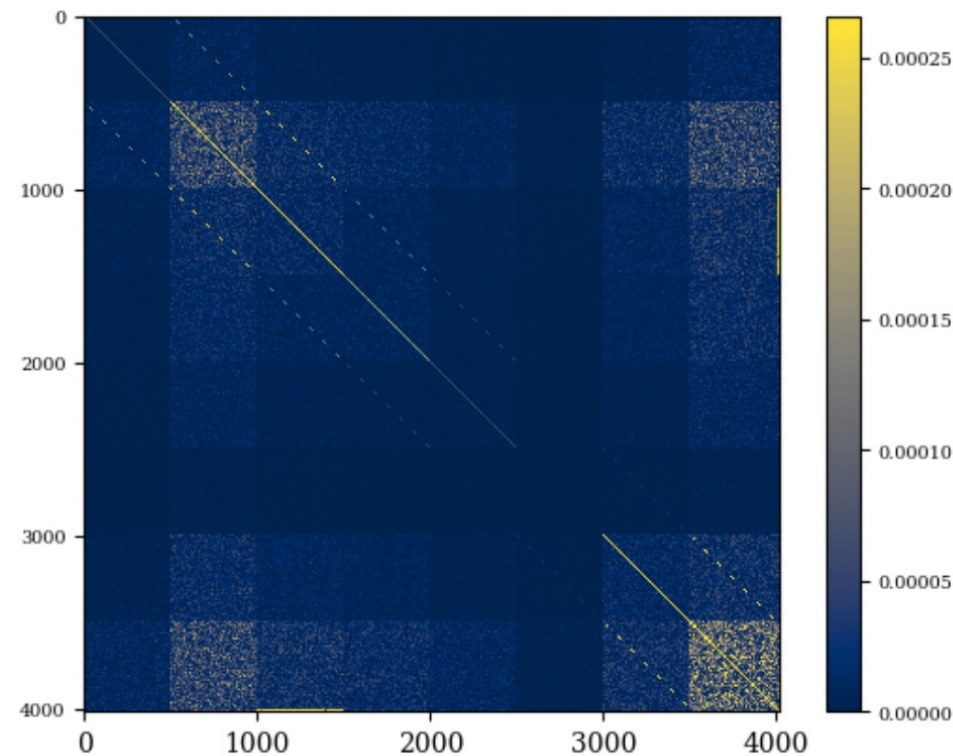
$$H_{v_i v_j} = \frac{\partial^2 \mathcal{L}}{\partial v_i \partial v_j^T} \in R^{m \times m}$$

$$H_{w_i v_i} = \frac{\partial^2 \mathcal{L}}{\partial w_i \partial v_i^T} \in R^{d \times m}$$

$$H_{w_i v_j} = \frac{\partial^2 \mathcal{L}}{\partial w_i \partial v_j^T} \in R^{d \times m}$$

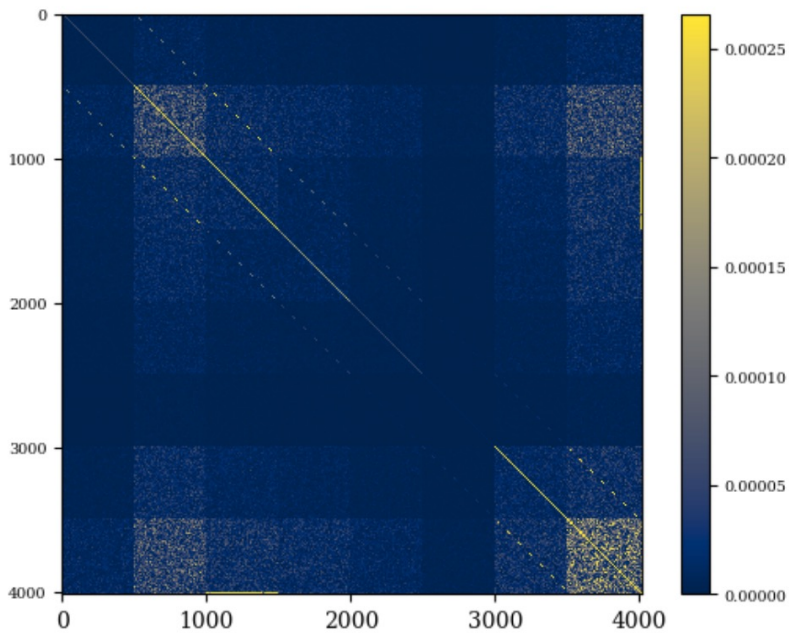
Initial trial: binary classification

- Simple setting: Linear model + CE loss, **binary classification**
- Cannot see special Hessian structures. Why?

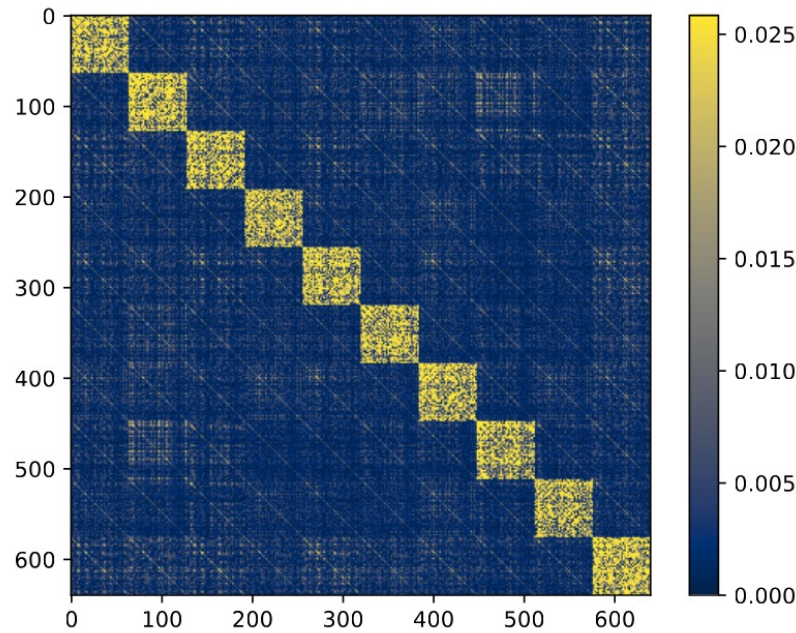


We find a phase transition as # class $C \rightarrow \infty$

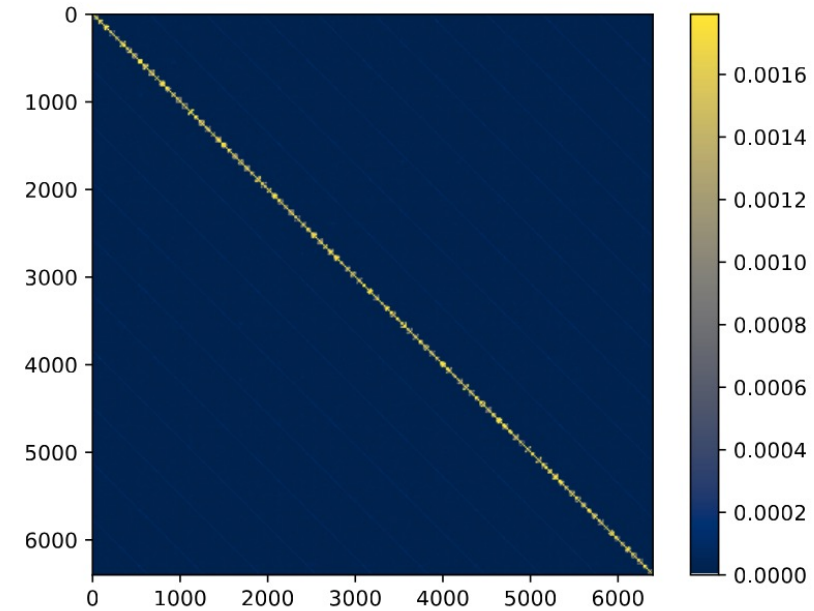
- Simple setting: Linear model + CE loss, #C class classification



C = 2



C = 10



C = 100



It seems that **large #class C** is important

Discussion on Twitter

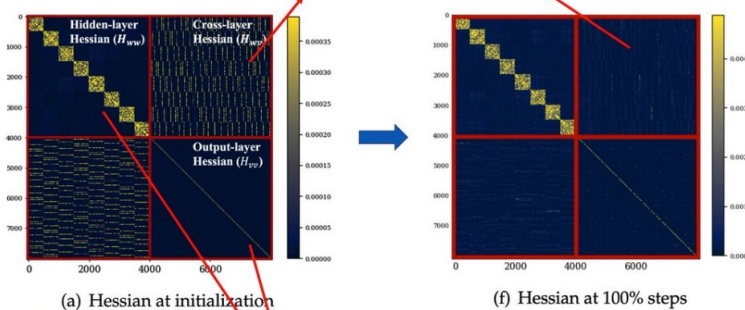
--We announced the paper a few months ago in X
--Unexpectedly high attention

Check out more details in our paper arxiv.org/pdf/2505.02809. Math tools: Random Matrix Theory.

翻译帖子

What does the Hessian of NNs look like?

A “dynamic force” that erases the “block-circulant” structure in H_{WV}



(a) Hessian at initialization

(f) Hessian at 100% steps

A “static force” that maintains the block-diag structure in H_{WV} and H_{VV} .
A primary driver of this structure: **large # classes C!! (sounds familiar to LLMers?)**

See the rigorous proof in our paper: <https://arxiv.org/pdf/2505.02809>. Math tools: Random Matrix Theory

下午10:04 · 2025年5月8日 · 6.3万 查看

查看 帖子 互动量

15

67

392

382



发布你的回复

回复



Norm Matloff

Fascinating results. I generally believe that ML has become far to mathematical, but this one is a major exception.

1

2

812



Yushun Zhang @ericzhang0410 · 5月10日

Thank you professor for the kind words!! 😊

3

646



Jason Lee @jasondeanlee · 5月13日

Is there some simple intuition why C to infnty gives block diag ?

1

1

546



Yushun Zhang @ericzhang0410 · 5月13日

Thank you for the question, Prof. Lee! One intuition is that neurons within the same hidden layer are not directly connected to each other—there are no explicit links between them. The correlation between two hidden neurons arises solely through the output weights stemming from [显示更多](#)

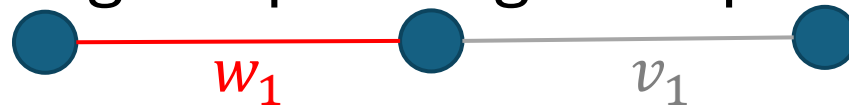
119

Contents

- Part II-1: Hessian structure in Simple Cases
- Part II-2: Dynamic force during training
- Part II-3: Our results

SISO Case

- Let us start from the most simple NN:
- Example 1: Single-input-single-output (SISO):



Input data $x = 1$. No activation, label = 0, MSE loss: $\ell(w_1 v_1) = \frac{1}{2} (w_1 v_1)^2$

Gradient: $\frac{\partial \ell}{\partial w_1} = w_1 v_1^2$ $\frac{\partial \ell}{\partial v_1} = w_1^2 v_1$

Hessian: $\frac{\partial^2 \ell}{\partial w_1 \partial w_1} = v_1^2$ $\frac{\partial^2 \ell}{\partial v_1 \partial w_1} = 2w_1 v_1$

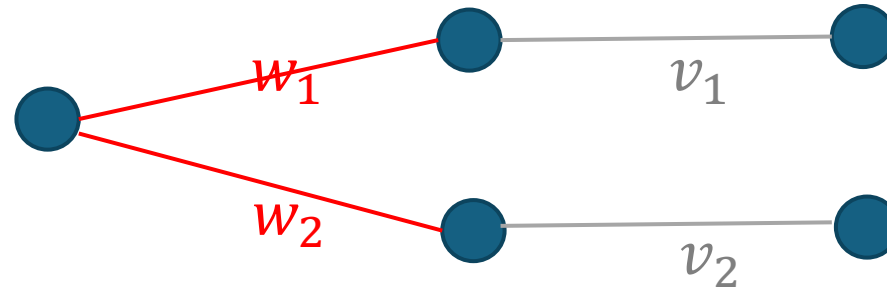
Observation: off-diagonal entries are non-zero
i.e., w_1 and v_1 has "correlations"

$\frac{\partial^2 \ell}{\partial w_1 \partial v_1} = 2w_1 v_1$ $\frac{\partial^2 \ell}{\partial v_1 \partial v_1} = w_1^2$

Why? See from computation graph w_1 and v_1 are **linked together**

SIMO Case (I)

- Example 2-1: Single-input-multi-output (SIMO):
(this is not a standard NN, but is good for understanding)



Input data $x = 1$. No activation, label = 0, MSE loss: $\ell(w_1, w_2, v_1, v_2) = \frac{1}{2}(w_1 v_1)^2 + \frac{1}{2}(w_2 v_2)^2$

Gradient: $\frac{\partial \ell}{\partial w_1} = w_1 v_1^2$

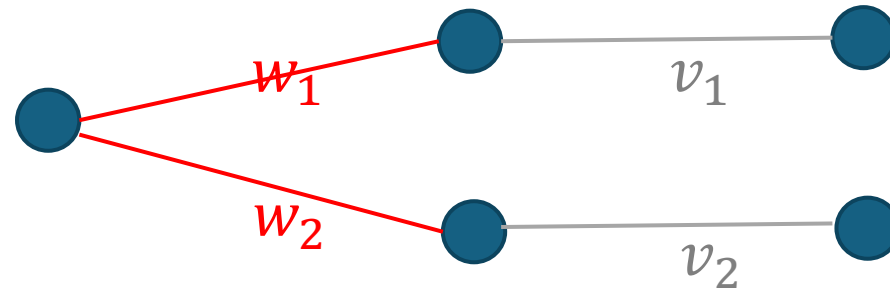
Hessian (1st row): $\frac{\partial^2 \ell}{\partial w_1 \partial w_1} = v_1^2$ $\frac{\partial^2 \ell}{\partial w_1 \partial w_2} = 0$ $\frac{\partial^2 \ell}{\partial w_1 \partial v_1} = 2w_1 v_1$ $\frac{\partial^2 \ell}{\partial w_1 \partial v_2} = 0$

Two zeros in the first-row of Hessian:

1st 0: w_1 and w_2 independent for the loss; 2nd 0: similar

SIMO Case (II)

- Example 2-1: Single-input-multi-output (SIMO):
(this is not a standard NN, but is good for understanding)



Input data $x = 1$. No activation, label = 0, MSE loss: $\ell(w_1, w_2, v_1, v_2) = \frac{1}{2}(w_1 v_1)^2 + \frac{1}{2}(w_2 v_2)^2$

Hessian:

	w_1	w_2	v_1	v_2
w_1		0		0
w_2	0		0	
v_1		0		0
v_2	0		0	

This is a most simple block-circulant-block-diagonal matrix

Observation:

$H_{i,j} \neq 0$ $\leftarrow$ i and j are connected in the graph
(which means: i and j has **multiplicative relation**)

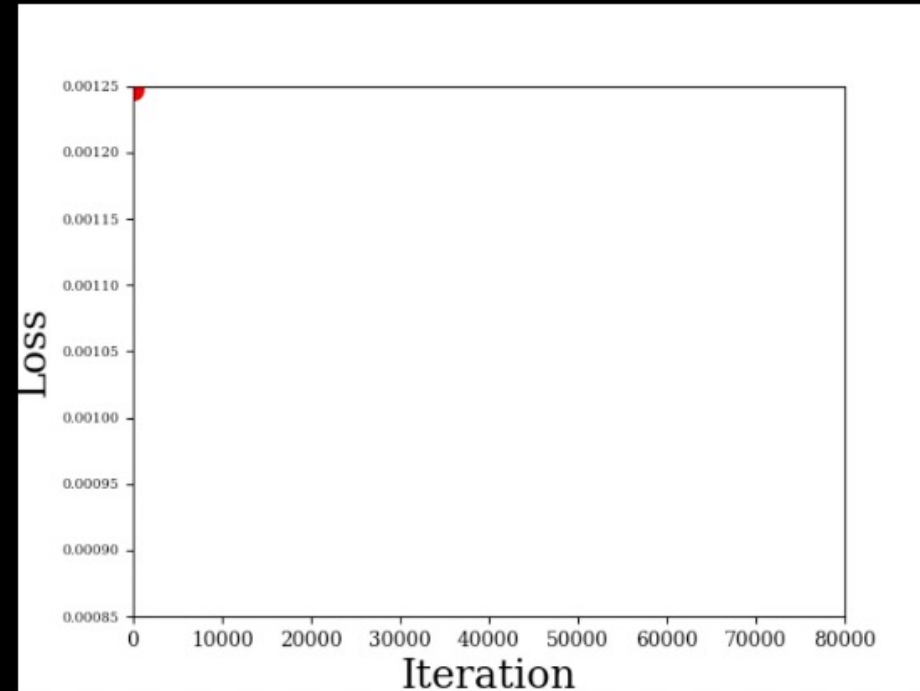
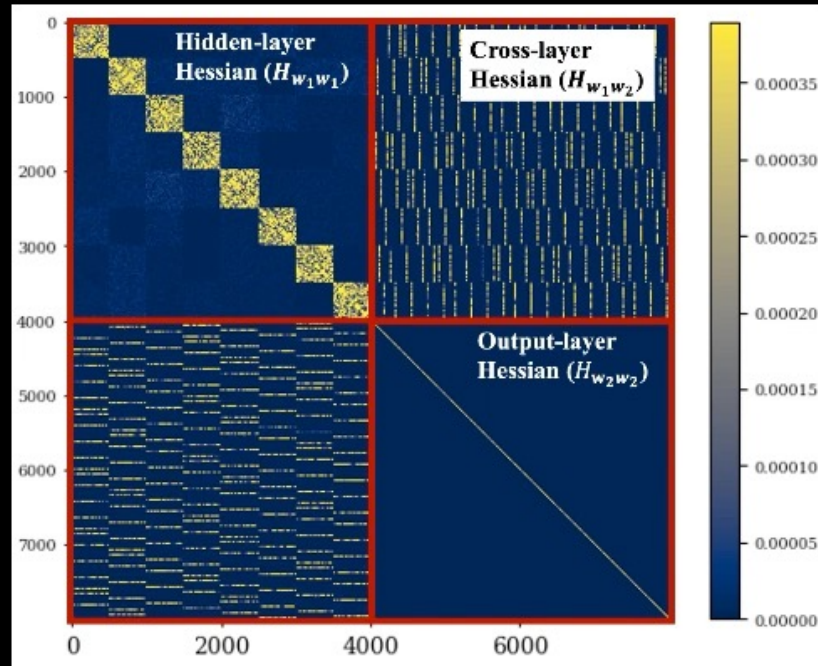
$H_{i,j} = 0$ $\leftarrow$ i and j are not “connected” in the graph
(which means: i and j has **no multiplicative relation**)

Contents

- Part II-1: Native Hessian structure in 2-layer Linear Nets
- **Part II-2: Hessian during training and dynamic Force**
- Part II-3: Our results

Empirical Observations: 1-hidden-layer NN

Hessian of a 2-layer relu NN, input dim = # classes = 500, width = 8, CE loss +Adam, Gaussian data + random label, sample size = 5000

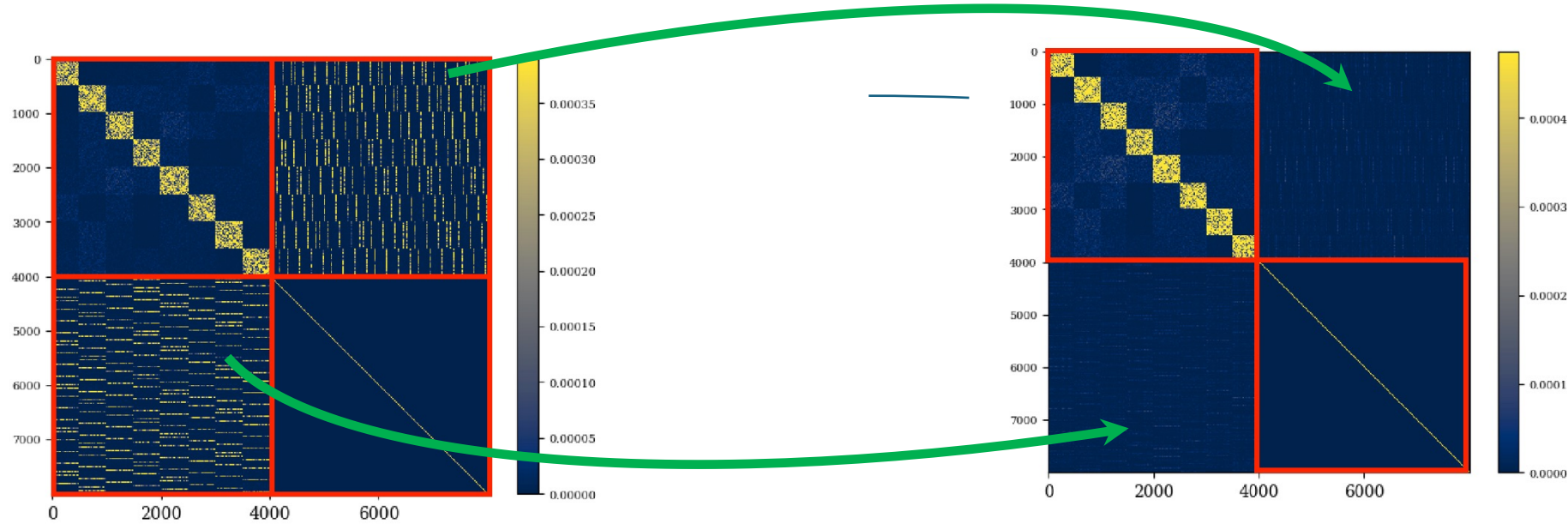


[Click to play the video]

- (i) block-circulant-block-diagonal structure at initialization
- (ii) The block-circulant part vanishes along training
- (iii) The near-block-diagonal pattern maintains along training

🤔 Why?

Two forces that shape Hessian structure



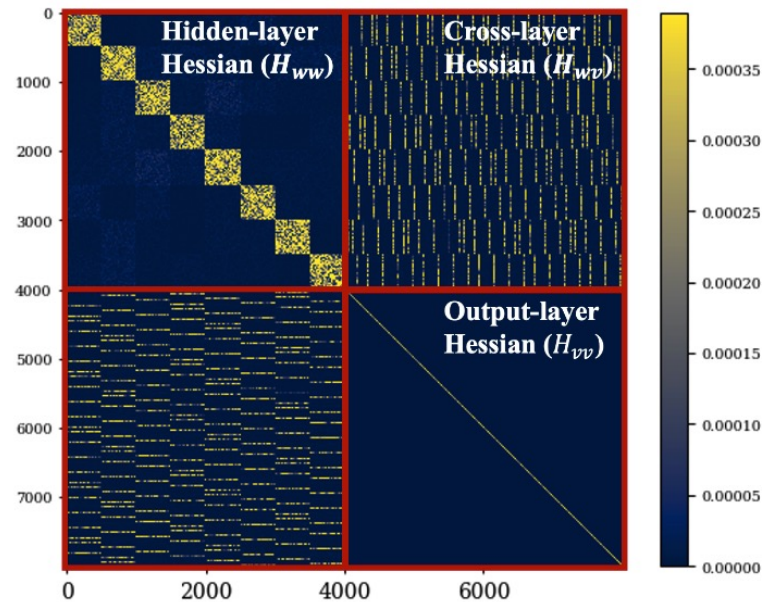
(a) Hessian at initialization

(f) Hessian at 100% steps

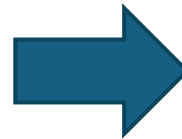
Force I: a **static force** rooted in the architecture design;
Force II: and a **dynamic force** arisen from training.

- We will provide:
 - 1. A simple explanation on the **dynamic force**
 - 2. a rigorous theoretical analysis of **static force** at random initialization

Hessian during training: the “dynamic force”



(a) Hessian at initialization



Training eliminates the off-block part in H_{wv} . Why?

Training eliminates the off-block part in H_{wv} . Why?

Hessian during training: the “dynamic force”

$$\min_{W \in \mathbb{R}^{m \times d}, V \in \mathbb{R}^{m \times C}} \frac{1}{N} \sum_n \ell(f(x_n), y_n) = \min_{W \in \mathbb{R}^{m \times d}, V \in \mathbb{R}^{m \times C}} \frac{1}{N} \sum_n -\log \frac{e^{\sigma(Wx_n)^T v_{y_n}}}{\sum_c e^{\sigma(Wx_n)^T v_c}}$$

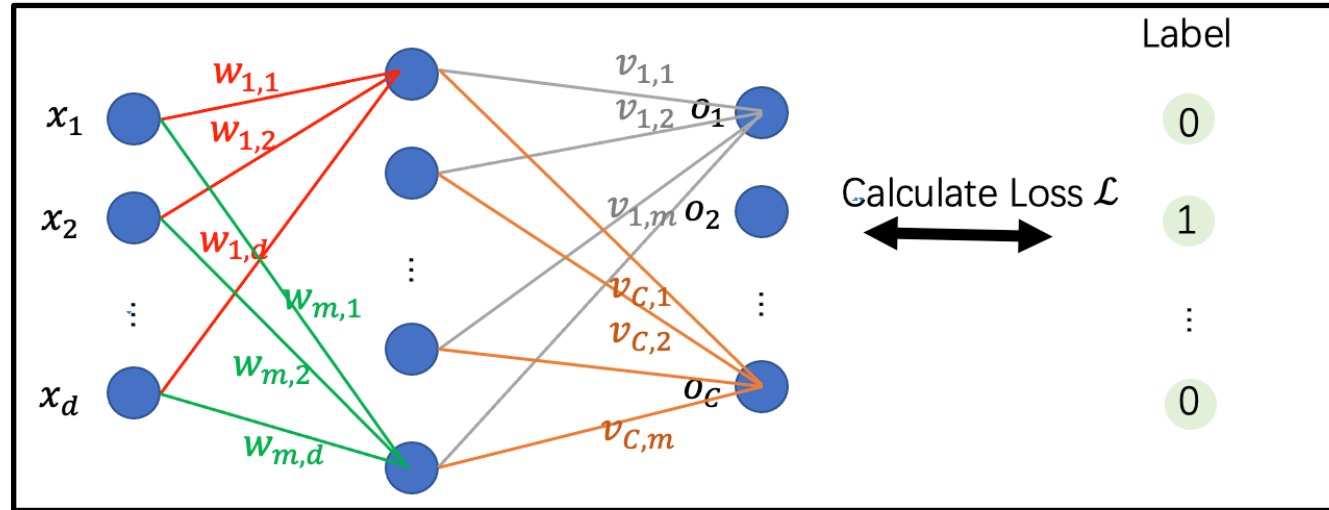
$$\frac{\partial \ell}{\partial w_i} = -\frac{1}{N} \sum_n \sum_c (\delta_{y_n, c} - p_{n, c}) v_{c, i} \mathbb{I}(w_i^T x_n \geq 0) x_n \in \mathbb{R}^d$$

$$H_{w_i v_j} = \frac{\partial^2 \ell}{\partial w_i \partial v_j^T} = \begin{bmatrix} 0 & \cdots & a_{i,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & a_{i,d} & 0 & \cdots & 0 \end{bmatrix} + O\left(\frac{1}{c}\right) \in \mathbb{R}^{d \times m}, \quad \text{only the } i\text{-th column is non-zero (if ignoring the } O\left(\frac{1}{c}\right) \text{ noise)}$$

$$\text{where } a_{i, d'} = -\frac{1}{N} \sum_n \sum_c (\delta_{y_n, c} - p_{n, c}) v_{c, i} \mathbb{I}(w_i^T x_n \geq 0) x_{n, d'}$$

- **Remark:** as training goes on, we have : $p_{n, c} \rightarrow 1$ for $c = y_n$
 $p_{n, c} \rightarrow 0$ for $c \neq y_n$  Therefore, $(\delta_{y_n, c} - p_{n, c}) \rightarrow 0$ along training
- This can explain the “dynamic force”: how the “off-diagonal” part vanishes along training

“Dynamic Force”



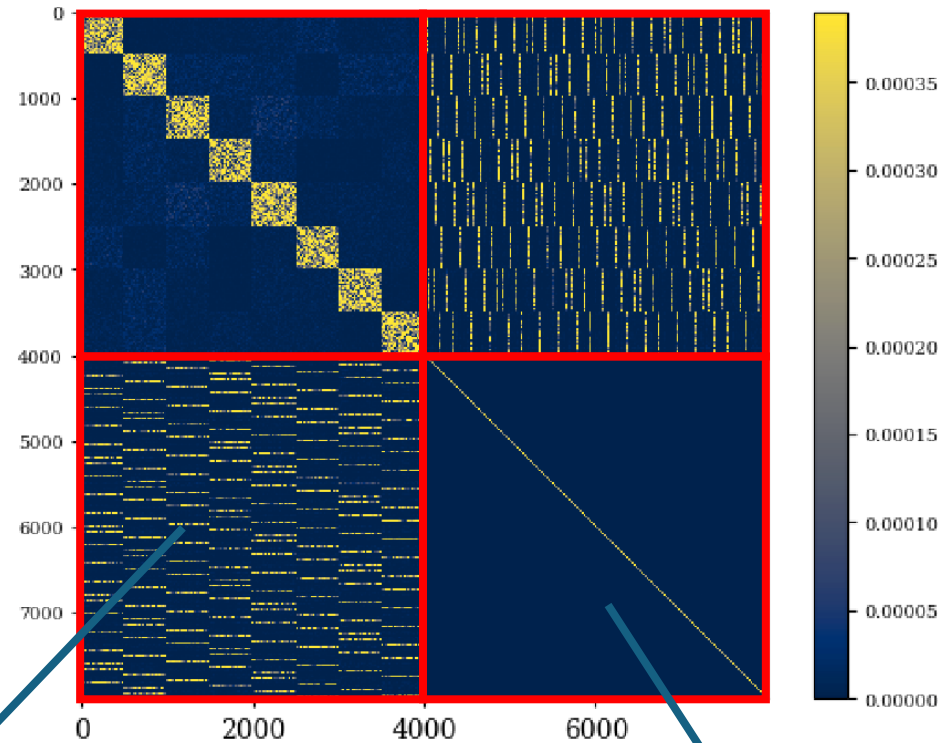
$$H_{w_i v_j} = \frac{\partial^2 \ell}{\partial w_i \partial v_j^T} = \begin{bmatrix} 0 & \cdots & a_{i,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & a_{i,d} & 0 & \cdots & 0 \end{bmatrix} + O\left(\frac{1}{c}\right) \in R^{d \times m}, \text{ where } a_{i,d'} = -\frac{1}{N} \sum_n \sum_c (\delta_{y_n, c} - p_{n,c}) v_{c,i} \mathbb{I}(w_i^T x_n \geq 0) x_{n,d'}$$

Key take-away: $H_{wv} \approx O(\text{optimality gap})$, which are expected to vanish (experiments: vanishes quickly as training begins)

Contents

- Part II-1: Native Hessian structure in 2-layer Linear Nets
- Part II-2: Hessian during training and dynamic Force
- **Part II-3: Our results**

Intuition from statistics: the “static force”



Hessian at initialization with CE loss

Training eliminates
“block-circulant” in H_{wv} : (previous parts)

H_{ww} and H_{vv} seems always “block-diag”:
(More involved, see next slides)

Case 1: linear model + MSE loss

Case 1: linear models with MSE loss. Consider the linear model $f(V; x) = Vx \in \mathbb{R}^C$, where $V = (v_1^\top; \dots; v_C^\top) \in \mathbb{R}^{C \times d}$ is the weight matrix, and $v_i \in \mathbb{R}^d$ is the weight associated with the i -th class (or output neuron). Consider minimizing the MSE loss as follows:

$$\min_V \ell_{\text{MSE}}(V) := \frac{1}{N} \sum_{n=1}^N \|Vx_n - \mathcal{Y}_n\|_2^2, \quad (2)$$

where $\mathcal{Y}_n \in \{0, 1\}^C$ is a C -dimensional one-hot vector with 1 at the index for the class of y_n and 0 elsewhere. The Hessian matrix is, for $i, j \in [C]$:

$$\begin{cases} \frac{\partial^2 \ell_{\text{MSE}}(V)}{\partial v_i \partial v_i^\top} = \frac{1}{N} \sum_{n=1}^N x_n x_n^\top \\ \frac{\partial^2 \ell_{\text{MSE}}(V)}{\partial v_i \partial v_j^\top} = 0_{d \times d}. \end{cases} \quad (3)$$

Hessian in Case 1 is trivially block diagonal
We will not discuss this case in the sequel

Case 2: linear model + CE loss

Case 2: linear models with CE loss. Consider the same linear model as in **Case 1**. We now change the loss function to the CE loss.

$$\min_V \ell_{\text{CE}}(V) := -\frac{1}{N} \sum_{n=1}^N \log \left(\frac{\exp(v_{y_n}^\top x_n)}{\sum_{c=1}^C \exp(v_c^\top x_n)} \right). \quad (4)$$

Define $p_{n,i} := \exp(v_i^\top x_n) / \left(\sum_{c=1}^C \exp(v_c^\top x_n) \right)$. The Hessian matrix is, for $i, j \in [C]$.

$$\begin{cases} \frac{\partial^2 \ell_{\text{CE}}(V)}{\partial v_i \partial v_i^\top} = \frac{1}{N} \sum_{n=1}^N p_{n,i} (1 - p_{n,i}) x_n x_n^\top \\ \frac{\partial^2 \ell_{\text{CE}}(V)}{\partial v_i \partial v_j^\top} = -\frac{1}{N} \sum_{n=1}^N p_{n,i} p_{n,j} x_n x_n^\top. \end{cases} \quad (5)$$

Intuitive understanding: at random initialization, suppose each entry in V follows i.i.d. zero-mean Gaussian distribution, we have $p_{n,i} \approx \frac{1}{C}$ for all $n \in [N], i \in [C]$. As such:

$$\frac{\left\| \frac{\partial^2 \ell_{\text{CE}}(V)}{\partial v_i \partial v_j^\top} \right\|_{\text{F}}}{\left\| \frac{\partial^2 \ell_{\text{CE}}(V)}{\partial v_i \partial v_i^\top} \right\|_{\text{F}}} \approx \frac{\sum_{n=1}^N p_{n,i} p_{n,j}}{\sum_{n=1}^N p_{n,i} (1 - p_{n,i})} \approx \frac{\frac{1}{C^2}}{\frac{1}{C} \left(1 - \frac{1}{C}\right)} = \frac{1}{C - 1}, \quad (6)$$

which pushes the Hessian to become block-diagonal as $C \rightarrow \infty$.

This is why large # class C helps!

Overview of our results

Settings: Consider general **C-class** classification problem: $(x_n, y_n)_{n=1}^N, x_n \in R^d, y_n \in \{1, 2, \dots, C\}$

We prove the following results (informal): when $N, d \rightarrow \infty$ with $\frac{d}{N} = \gamma$, we have

- Case 1 (linear model + MSE loss):
For any C , Hessian is strictly block-diag with **C blocks**
- Case 2 (linear model + CE loss):
Hessian approaches block-diag with **C blocks** with rate $O(1/C)$
- Case 3 (1-hidden-layer-NN with **m neurons** + MSE loss):
 - Hessian of hidden weights approach block-diag with **m blocks** with rate $O(1/\sqrt{C})$
 - Hessian of output weights approach block-diag with **C blocks** with rate $O(1/C)$
- Case 4 (1-hidden-layer-NN with **m neurons** + CE loss):
 - Hessian of hidden weights approach block-diag with **m blocks** with rate $O(1/\sqrt{C})$
 - Hessian of output weights approach block-diag with **C blocks** with rate $O(1/C)$

Main Results

Assumption 1 The entries of the data matrix $X_N = (x_1, \dots, x_N) \in \mathbb{R}^{d \times N}$ are i.i.d. $\mathcal{N}(0, 1)$.

Assumption 2 The model weights in W and V are initialized by LeCun initialization. That is: for the linear model, $V_{i,j} \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \frac{1}{d})$, $i \in [C], j \in [d]$; for 1-hidden-layer network, $W_{i,j} \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \frac{1}{d})$, $i \in [m], j \in [d]$, $V_{i,j} \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \frac{1}{m})$, $i \in [C], j \in [m]$.

Theorem 2 (1-hidden-layer networks.) Consider the Hessian expressions in (8) to (13), and assume Assumptions 1 and 2 hold. Then for any fixed $m \geq 3$, suppose $d, N \rightarrow \infty$, $\frac{d}{N} \rightarrow \gamma \in (0, +\infty)$, it holds that

$$\lim_{d, N \rightarrow \infty} \frac{\mathbf{E} \left[\left\| \frac{\partial^2 \ell_{\text{MSE}}(W, V)}{\partial w_i \partial w_j^\top} \right\|_{\text{F}}^2 \right]}{\mathbf{E} \left[\left\| \frac{\partial^2 \ell_{\text{MSE}}(W, V)}{\partial w_i \partial w_i^\top} \right\|_{\text{F}}^2 \right]}, \quad \lim_{d, N \rightarrow \infty} \frac{\mathbf{E} \left[\left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial v_i \partial v_j^\top} \right\|_{\text{F}}^2 \right]}{\mathbf{E} \left[\left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial v_i \partial v_i^\top} \right\|_{\text{F}}^2 \right]} \quad (28)$$

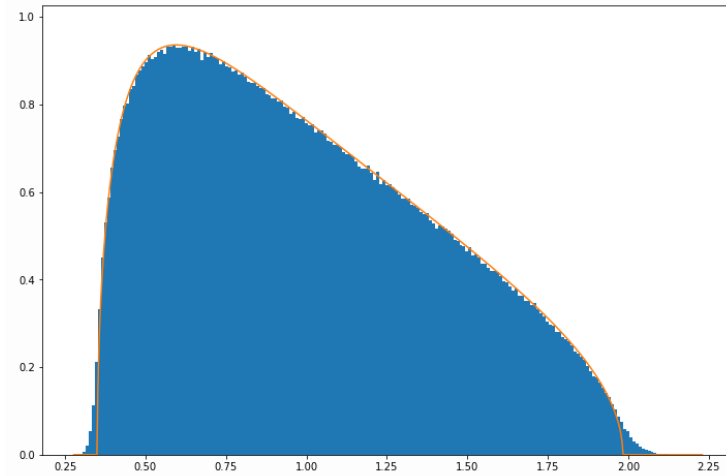
Key message:

the block-diagonal structure arises when

classes $C \rightarrow \infty$

vanish at the rate $\mathcal{O}(1/C)$, $\mathcal{O}(1/C)$, $\mathcal{O}(1/C^2)$, respectively, and the block-diagonal structure also emerges as C increases.

Exmple of GMP law: (Assume X and Λ are independent)



Eigenvalue histogram of $A_n = \frac{1}{n} X \Lambda X^T \in R^{d \times d}$, $\Lambda = I$, $n = 50$, $d = 300$, 1000 samples of A_n

Yellow curve: density of MP distribution with $d/n = 50/300$

But wait... In our $\frac{1}{N} X_N \Lambda_N X_N^T$, X_N, Λ_N are **clearly NOT independent**, so MP theorem cannot be applied

- Dependent matrix product is a difficult topic in RMT
- Fortunately, we observe additional good properties in our $\frac{1}{N} X_N \Lambda_N X_N^T$

Key properties in our matrix

$$\frac{\partial^2 \ell_{\text{CE}}(V)}{\partial v_i \partial v_i^\top} \stackrel{(5)}{=} \frac{1}{N} \sum_{n=1}^N p_{n,i}(1 - p_{n,i}) x_n x_n^\top := \frac{1}{N} X_N \Lambda_N X_N^\top \in \mathbb{R}^{d \times d}, \quad \text{and } p_{n,i} = \frac{\exp(v_i^\top x_n)}{\sum_{c=1} \exp(v_c^\top x_n)}$$

Key observation: Λ_N and X_N are asymptotic independence

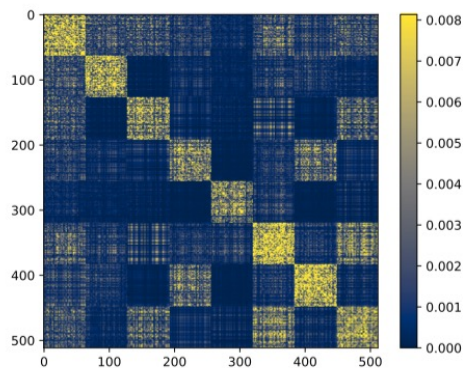
- Recall $v_i \sim \mathcal{N}(0, \frac{1}{d})$ and denote $z = v_i^\top x_n$, then $z|x \sim \mathcal{N}(0, \frac{\|x\|_2^2}{d})$. Further, since $x \sim \mathcal{N}(0, 1)$, we have $\frac{\|x\|_2^2}{d} \sim \mathcal{X}^2(1, \frac{2}{d})$, which concentrates to 1 as $d \rightarrow \infty$.
- As such, z asymptotically follows $\mathcal{N}(0, 1)$ and thus is independent of x . Therefore, Λ_N and X_N are asymptotically independent.

• **Guess:** limiting eigenvalue $X_N \Lambda_N X_N^\top \approx$ those as if X_N, Λ_N are independent

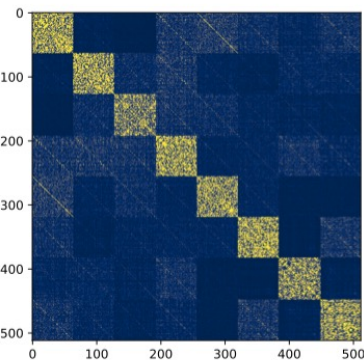
• **Proof technique:**
the Lindeberg principle (originally proposed to prove CLT)

The Effect of Increasing # classes C

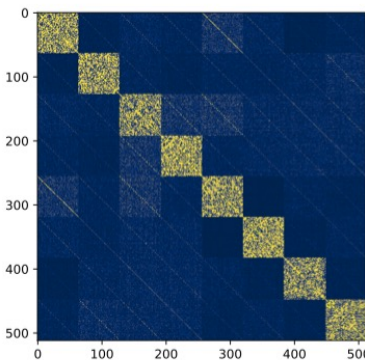
Hessian of
hidden weights



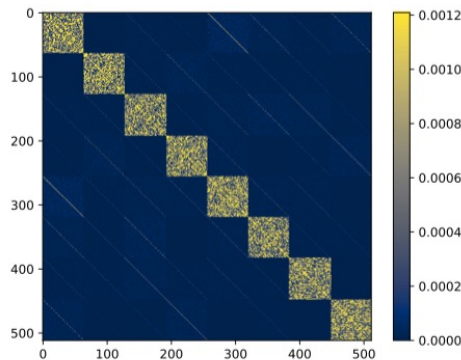
(a) $C = 10$



(b) $C = 50$

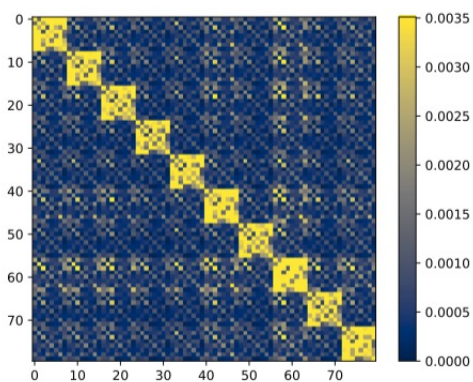


(c) $C = 100$

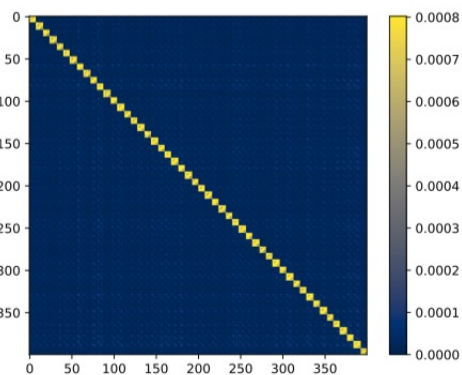


(d) $C = 1000$

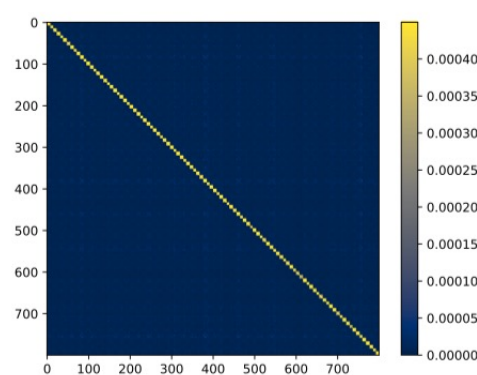
Hessian of
output weights



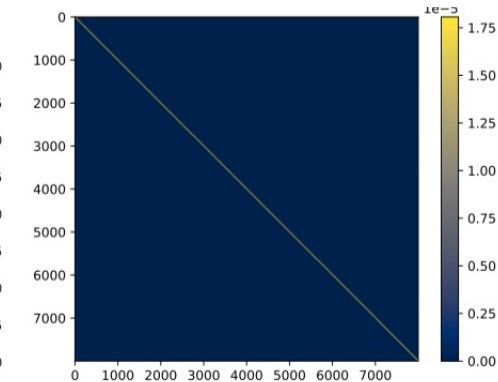
(a) $C = 10$



(b) $C = 50$



(c) $C = 100$

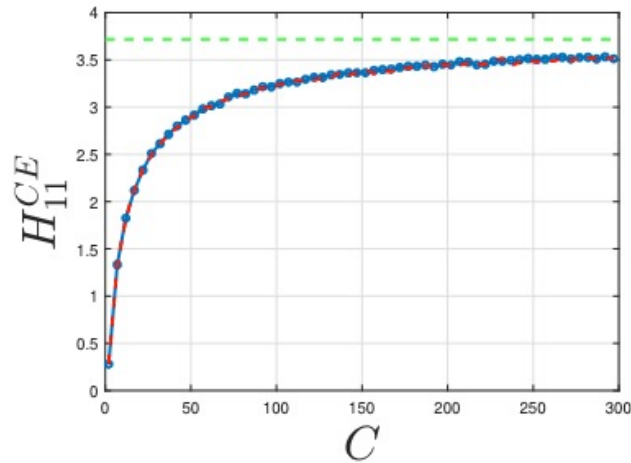


(d) $C = 1000$

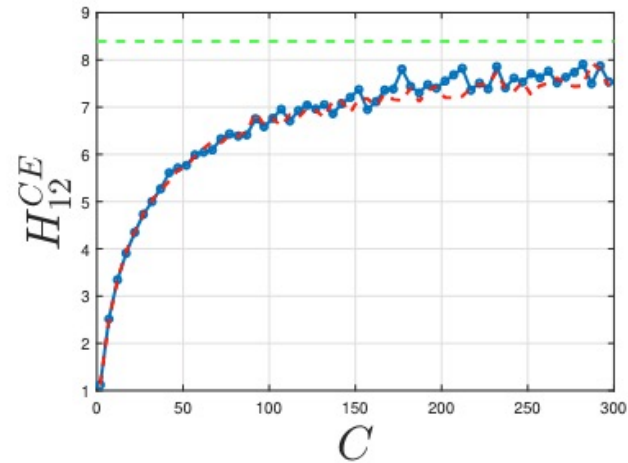
- The Hessian blocks of **1-hidden-laye NN** with 8 hidden neurons + #class C at random init.
- The block-diag structure **becomes clearer as C increases**

The Effect of Increasing C

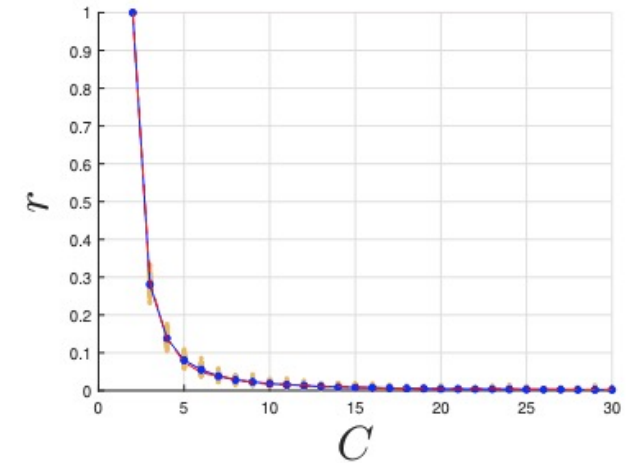
$$\tilde{H}_{11}^{\text{CE}} := \frac{1}{d} \left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial w_1 \partial w_1^\top} \right\|_{\text{F}}^2, \quad \tilde{H}_{12}^{\text{CE}} = \frac{C}{d} \left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial w_1 \partial w_2^\top} \right\|_{\text{F}}^2, \quad \tilde{r} = \frac{\left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial w_1 \partial w_2^\top} \right\|_{\text{F}}^2}{\left\| \frac{\partial^2 \ell_{\text{CE}}(W, V)}{\partial w_2 \partial w_2^\top} \right\|_{\text{F}}^2}$$



(a) C v.s. H_{11}^{CE}



(b) C v.s. H_{12}^{CE}



(c) C v.s. r

These quantities match our **theoretical prediction**

Guess: Hessian for Deep NNs?

Guess: Hessian for Deep NNs?

Evolution of Hessian of 4-layer MLP

初始点:

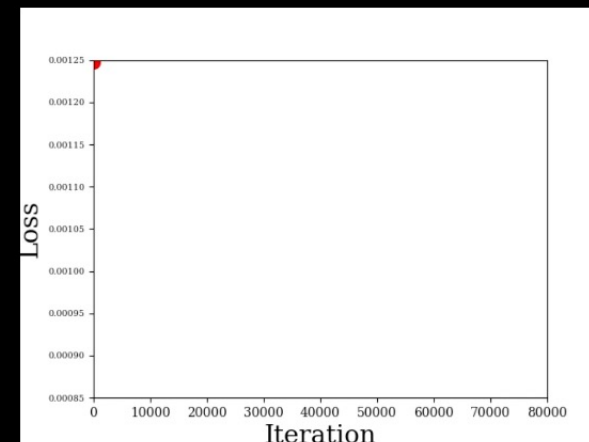
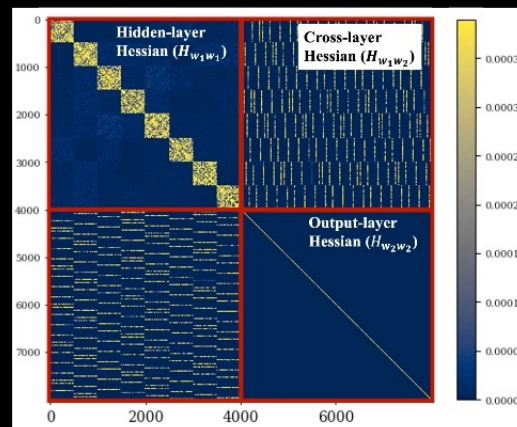
- 1) 跨层Hessian 非零;
- 2) 同层跨神经元的 Hessian ≈ 0 .

训练过程:

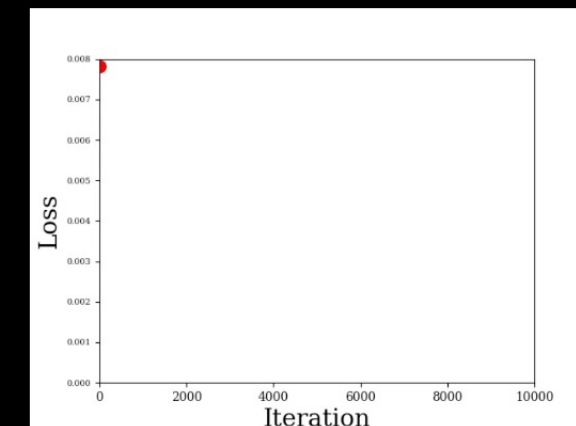
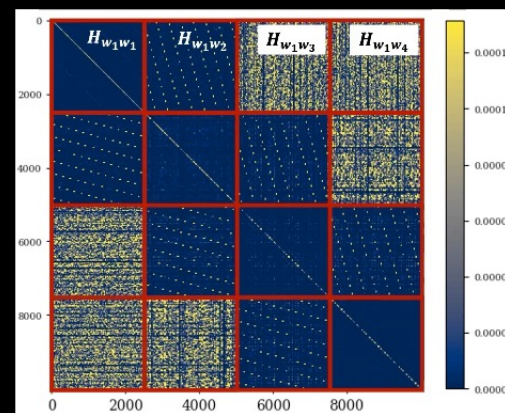
- 1) 跨层 Hessian 迅速收敛到 0;
- 2) 同层跨神经元 Hessian 保持很小。

Theoretical Justification?
Ongoing work.

Hessian of a **2-layer** relu NN, input dim = # classes = 500, width = 8, CE loss + Adam, Gaussian data + random label, sample size = 5000



Hessian of a **4-layer** relu NN, input dim = # classes = width = 50, CE loss + Adam, Gaussian data + random label, sample size = 500



Summary

- Part I: We provide an explanation why Transformer needs Adam, not SGD
 - **Heterogeneity of block-Hessian-spectra** causes SGD to underperform Adam
 - Proposed a new algorithm Adam-mini
- Part II: We prove that why Hessian of NN exhibits near-block-diagonal structure:
 - 1-hidden-layer-net, block-diagonal structure arises when **# classes $C \rightarrow \infty$**
 - **Key challenge:** Need $\|\frac{1}{N}X_N\Lambda_NX_N^T\|_F$, but X_N and Λ_N **are dependent**
 - Math tools: Random Matrix Theory

Mainly based on:

- Zhang, Chen, Ding, Li, Sun*, & Luo; Why Transformers Need Adam: A Hessian Perspective, NeurIPS 2024
- Zhang*, Chen*, Li, Ding, Chen, Kingma, Ye, Luo & Sun*; Adam-mini: Use Fewer Learning Rate To Gain More, ICLR 2025
- Dong*, Zhang* (Alphabetically ordered), Yao*, Sun*; Towards Quantifying the Hessian Structure of Neural Networks, Preprint
- **Thanks to all the collaborators!**

On the market:

Yushun Zhang



Thank You!

Email 邮箱: sunruoyu@cuhk.edu.cn

Website 个人网站: ruoyus.github.io

Interested in LLMs, machine learning,
optimization, communications, etc.

研究兴趣: 大模型、机器学习、优化、通信网络等



our githublink for
[Hessian spectrum calculation](#)



our githublink for Adam-mini