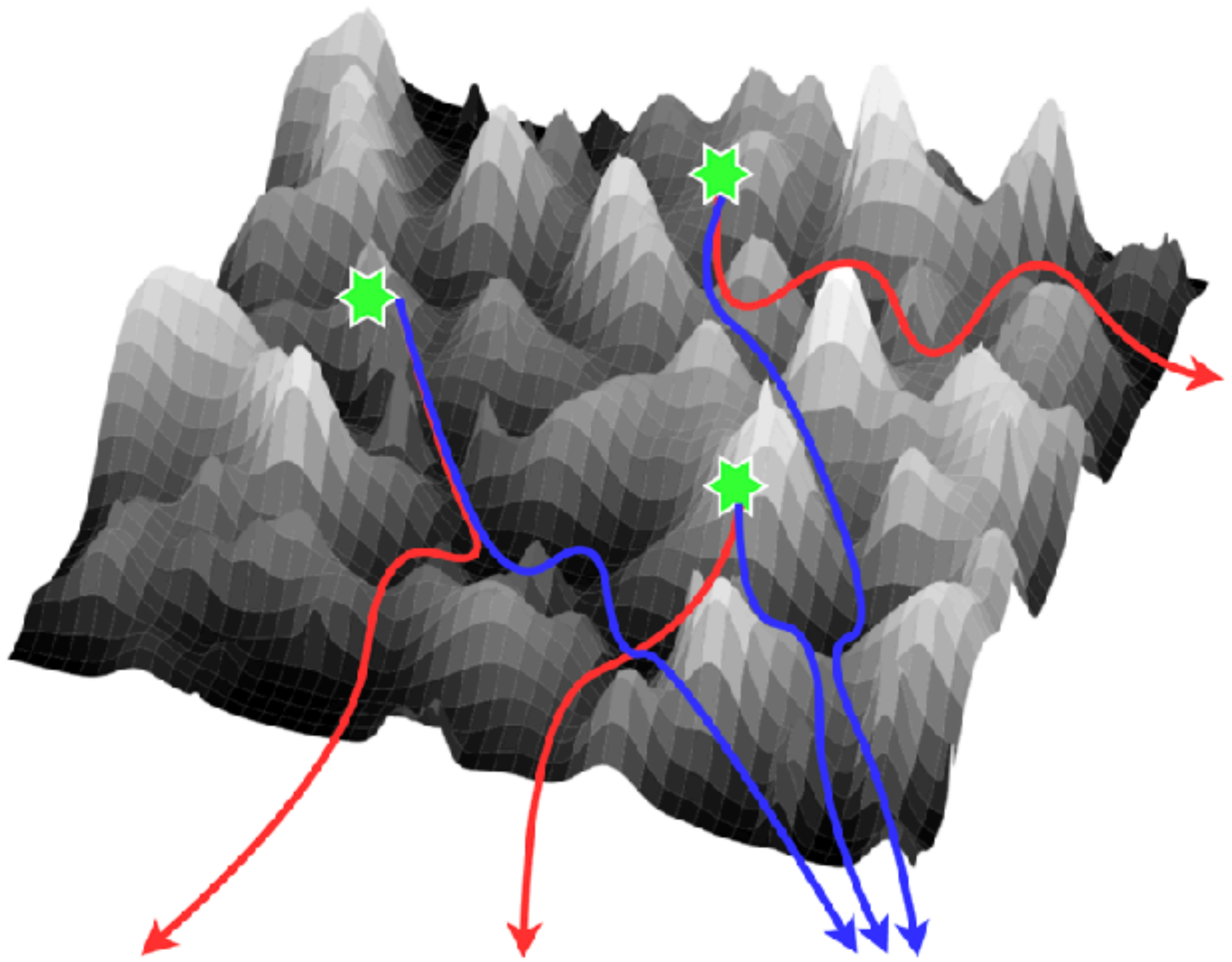
Feature learning from non-Gaussian inputs

Sebastian **Goldt** (SISSA, Trieste)

joint work w/ Lorenzo Bardone and Fabiola Ricci



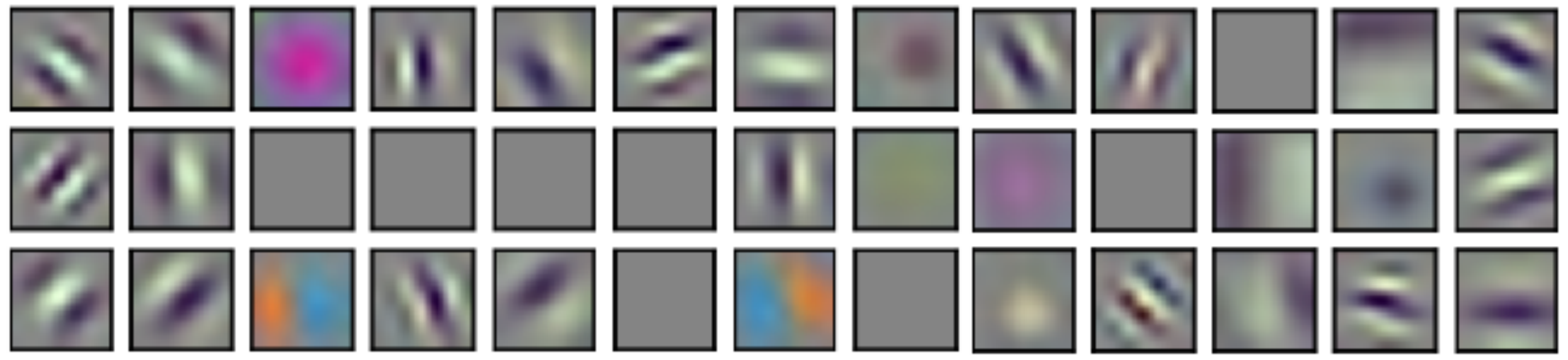
Statistical Physics & machine learning: moving forward — Cargèse, august 2025

What do neural networks learn from their inputs?

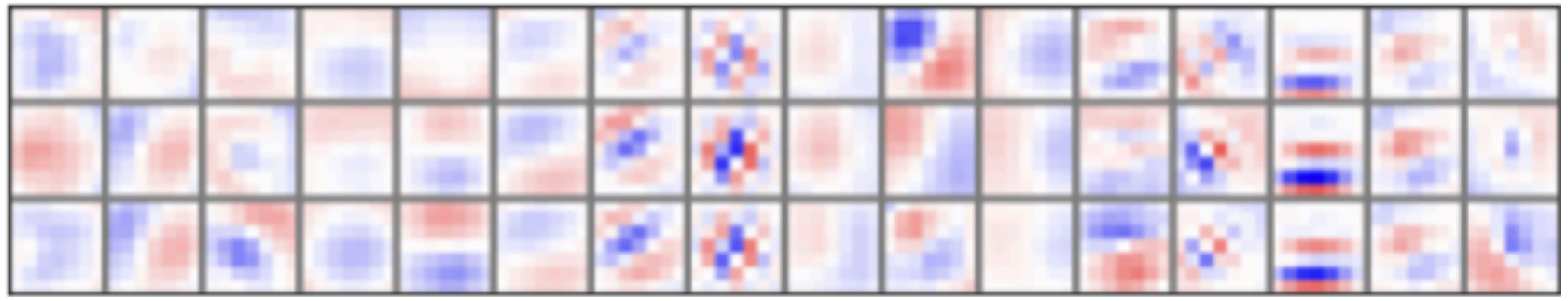
Neural networks learn stereotypical features

First-layer filters learnt from ImageNet resemble Gabor filters across architectures

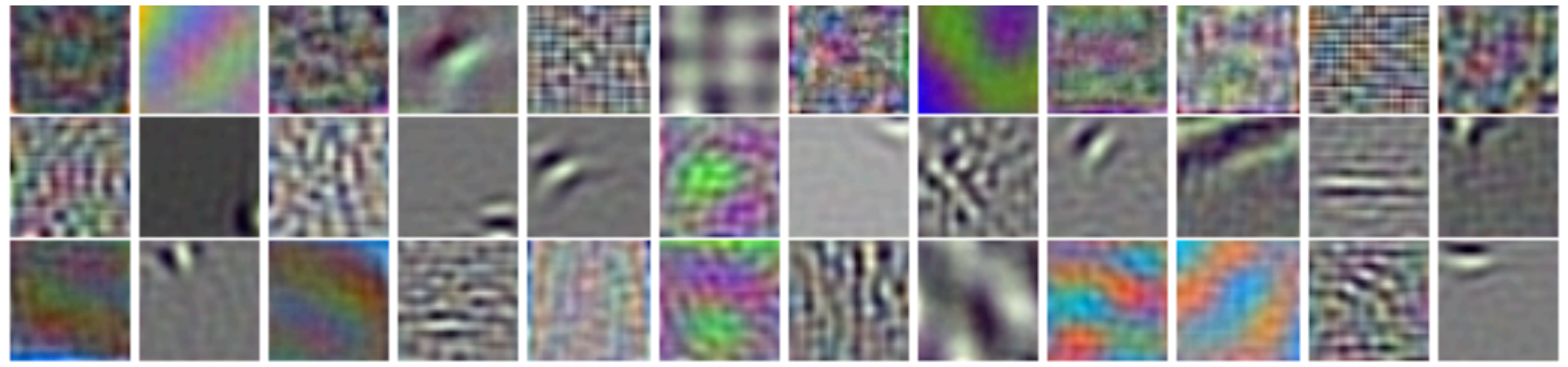
AlexNet Krizhevsky, Sutskever, Hinton (2012)



DenseNet121



VGG-11 Guth & Ménard (2024)



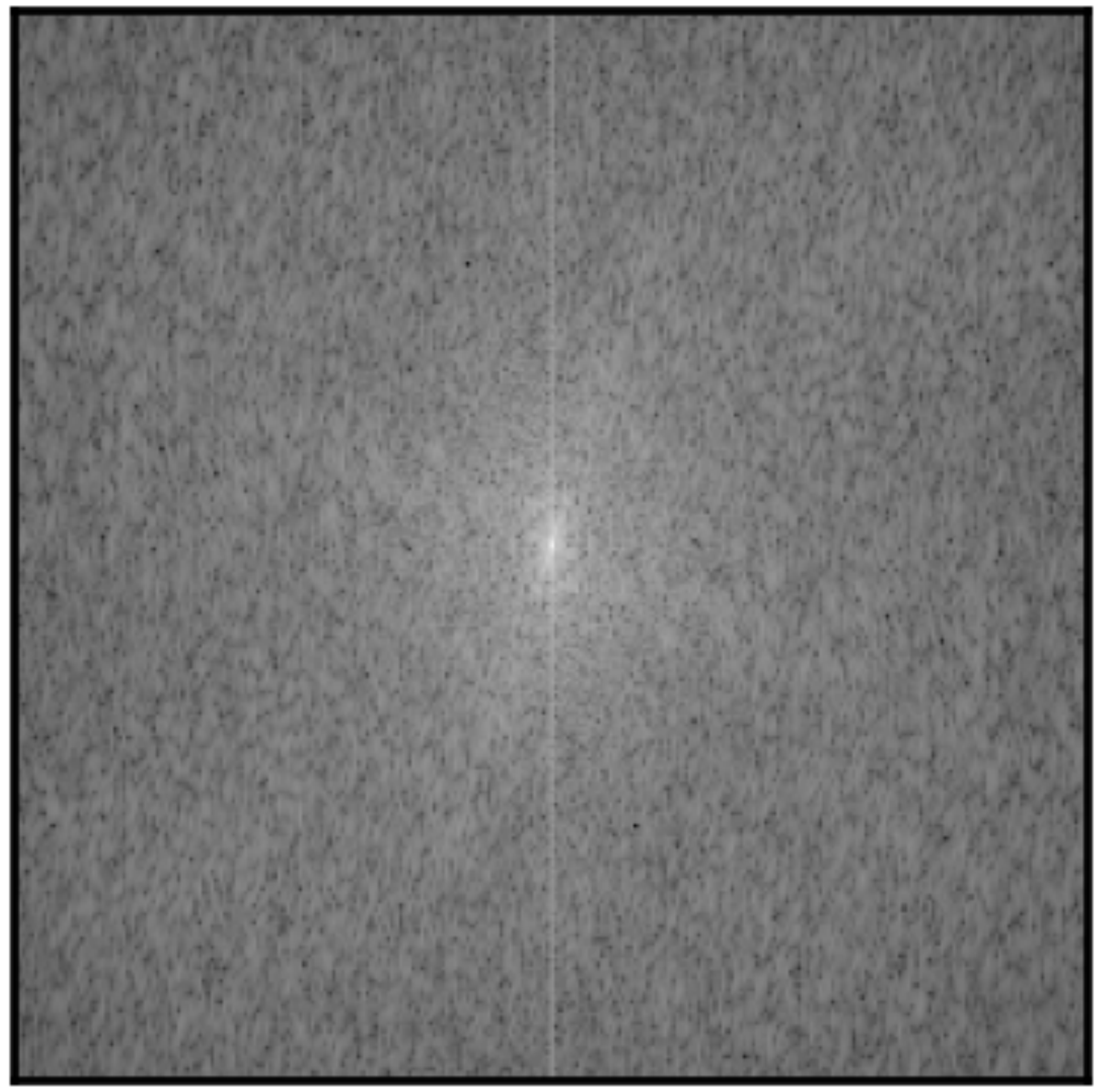
MLP mixer Tolstikhin et al. NeurIPS '21

Convergence of features across architectures —
inputs drive feature learning!

What is in an image?

A Fourier perspective

$X_{tt'}$

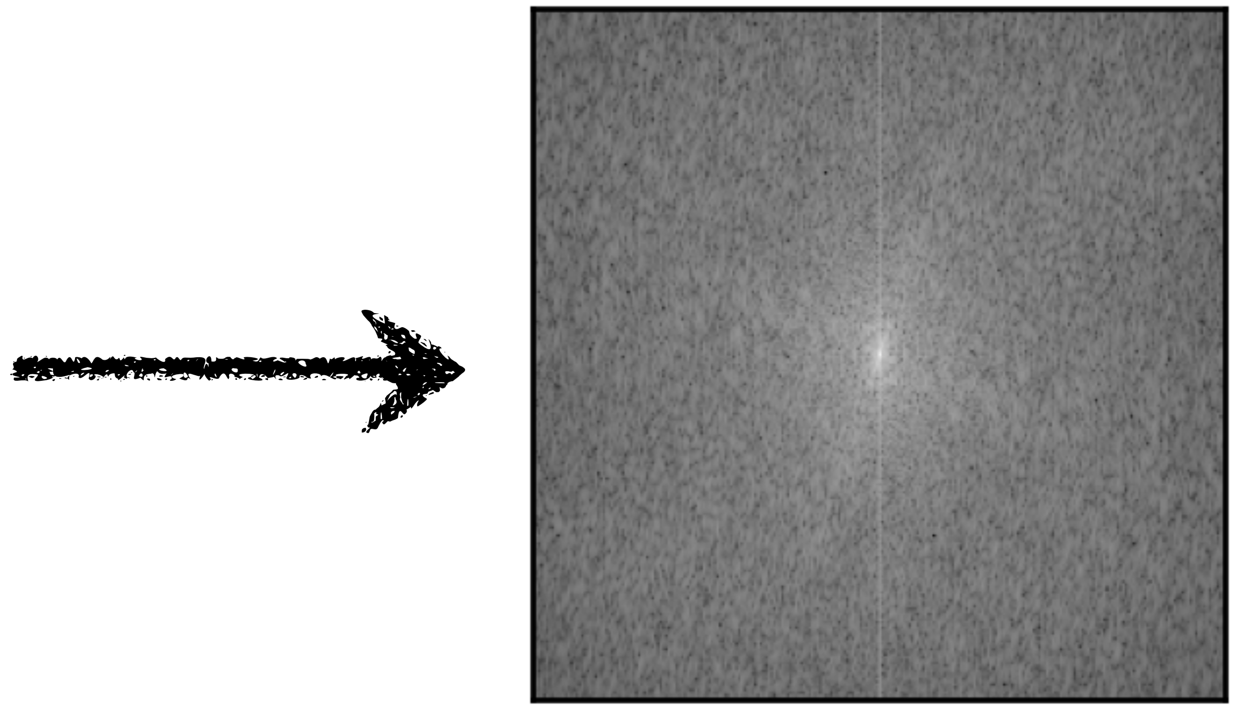


$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$

What is in an image?

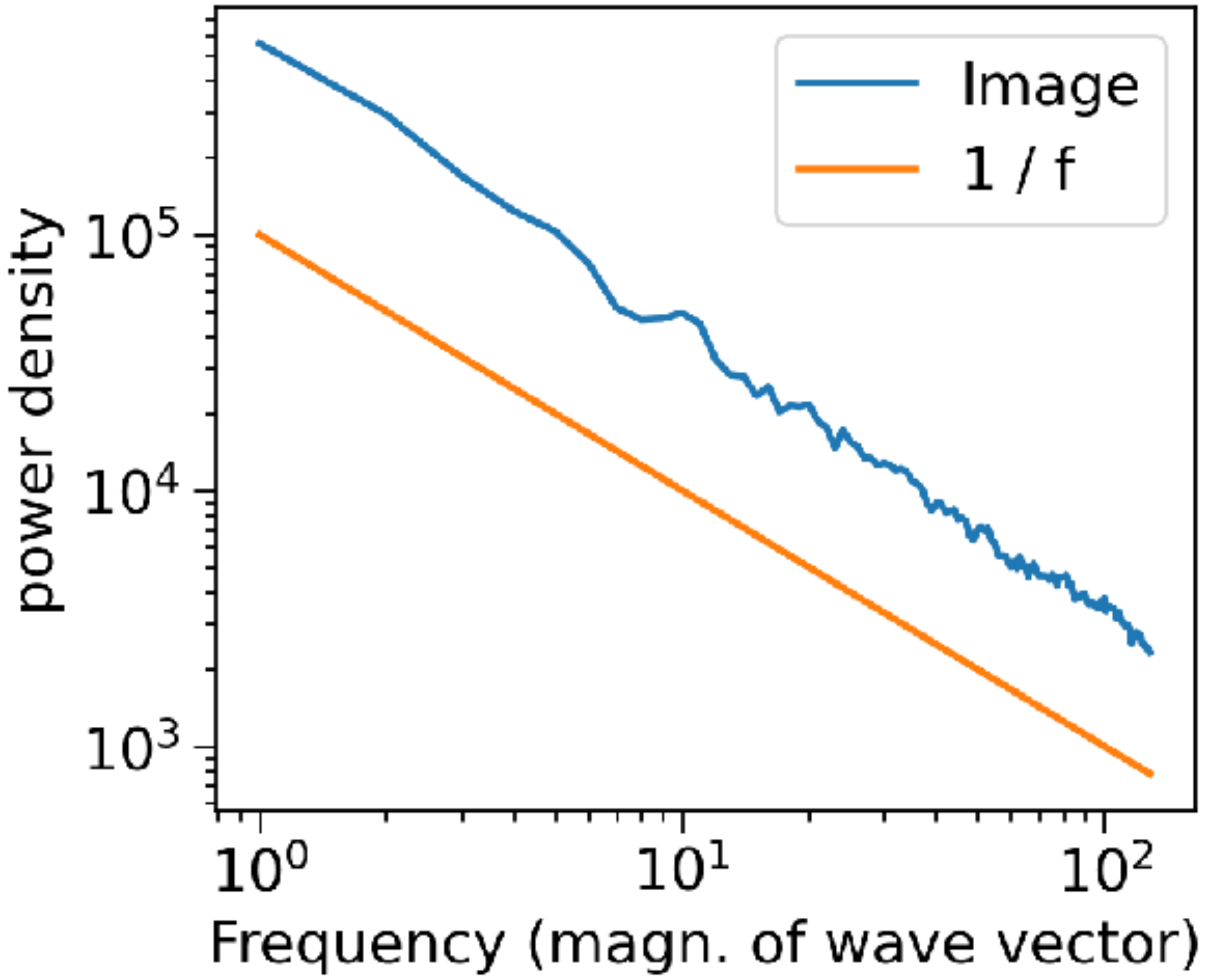
A Fourier perspective

$X_{tt'}$

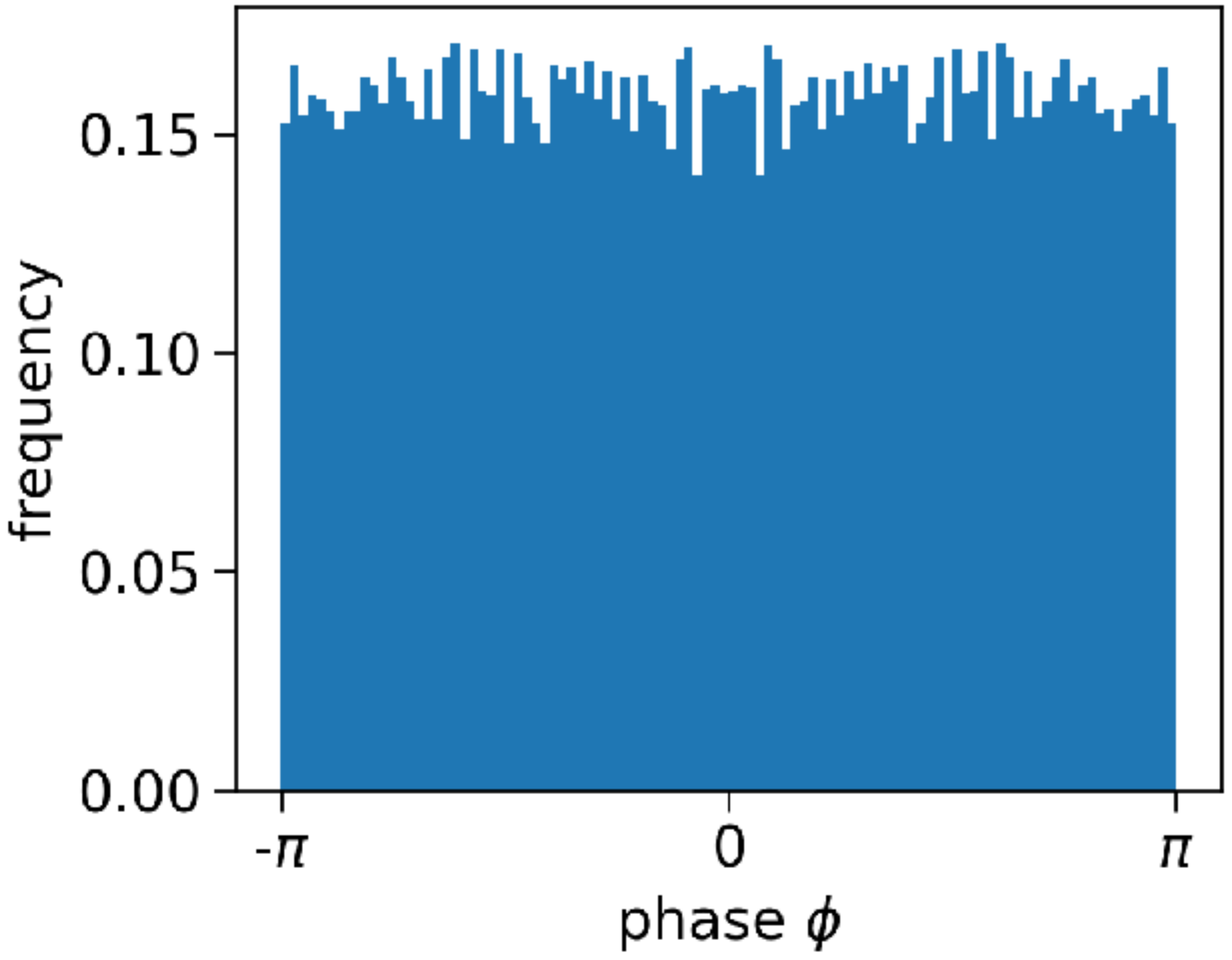


$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$

Amplitudes $A_{kk'}$ encode **pair-wise** correlations



Phases $\phi_{kk'}$ encode **higher-order** correlations



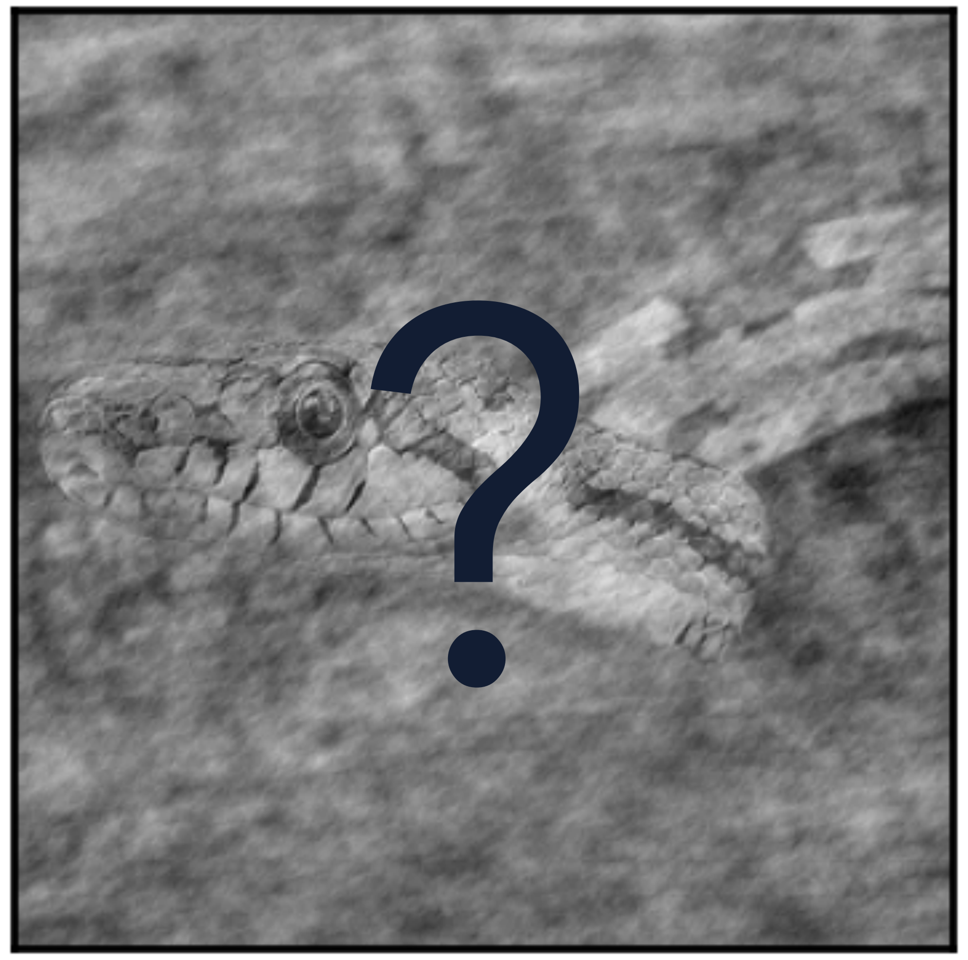
What matters in an image?

Let's do an experiment to find out! (Piotrowski & Campbell '82)

$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$

$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$

$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$



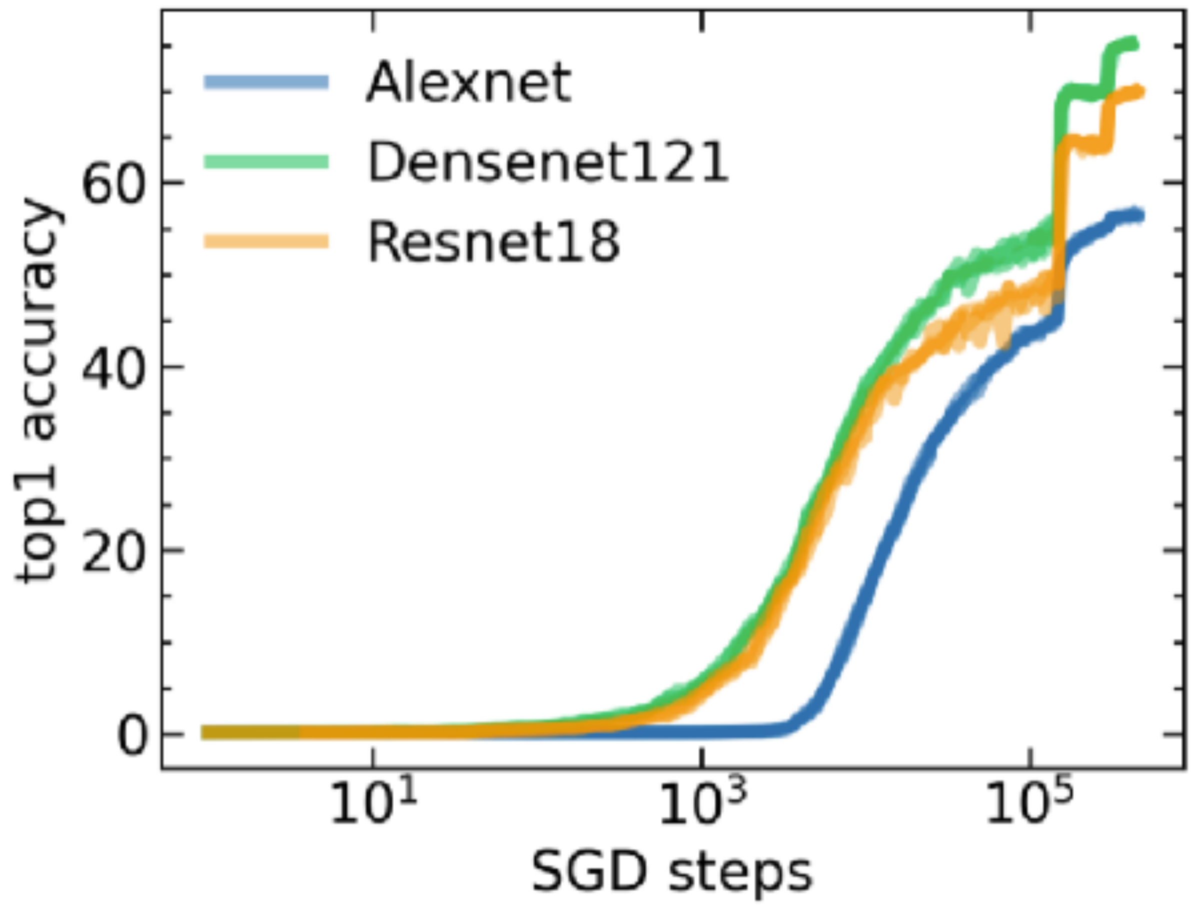
$\tilde{X}_{kk'} = A_{kk'} \exp\left(i\phi_{kk'}\right)$

Higher-order correlations are perceptually more important!

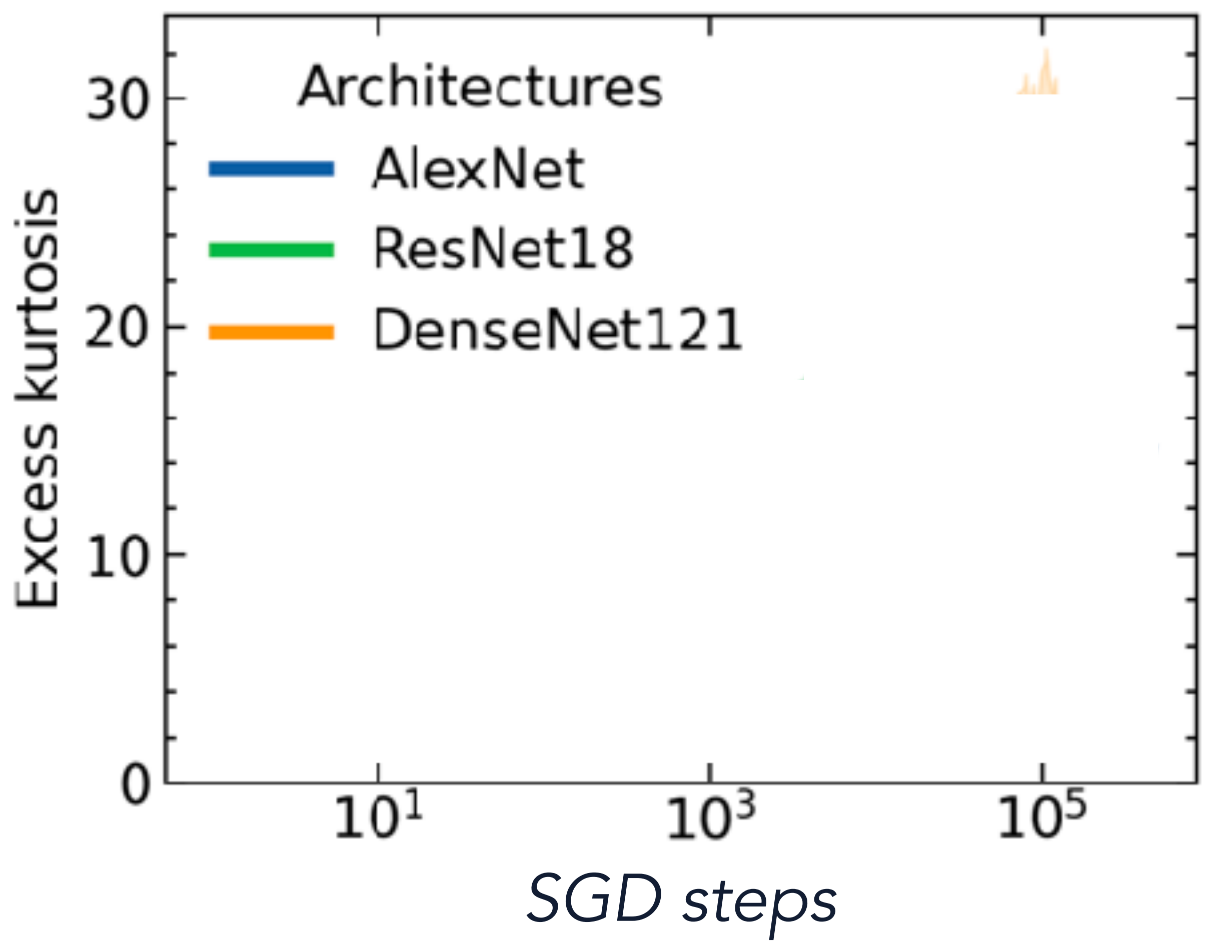
Oppenheim & Lim (1981); Piotrowski & Campbell (1982); Wichmann et al. (2005)

HOCs shape neural representations

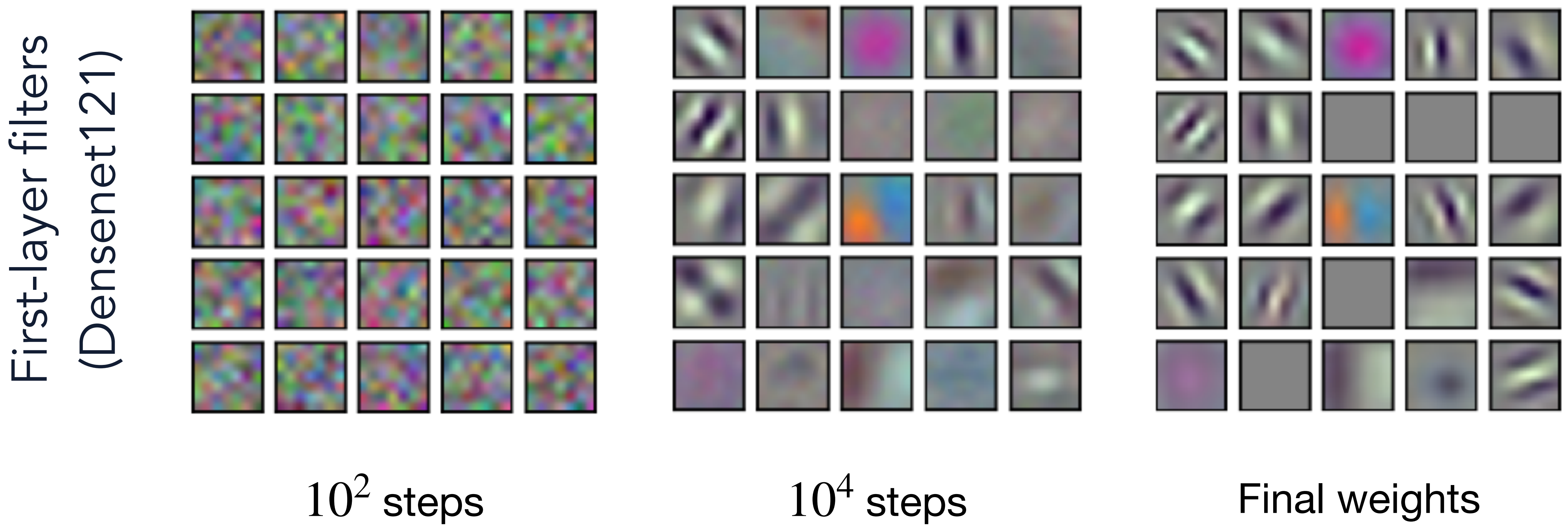
First layer filters relate to strongly non-Gaussian directions



$$\mathrm{kurt}(\lambda) = \mathbb{E}\lambda^4 - 3\mathbb{E}\lambda^2$$



$$\lambda = w^{(1)} \cdot x$$



Neural networks **learn features** from **non-Gaussian** input fluctuations.

How can we **analyse** this?

A simpler model for learning

Finding “interesting” projections of data

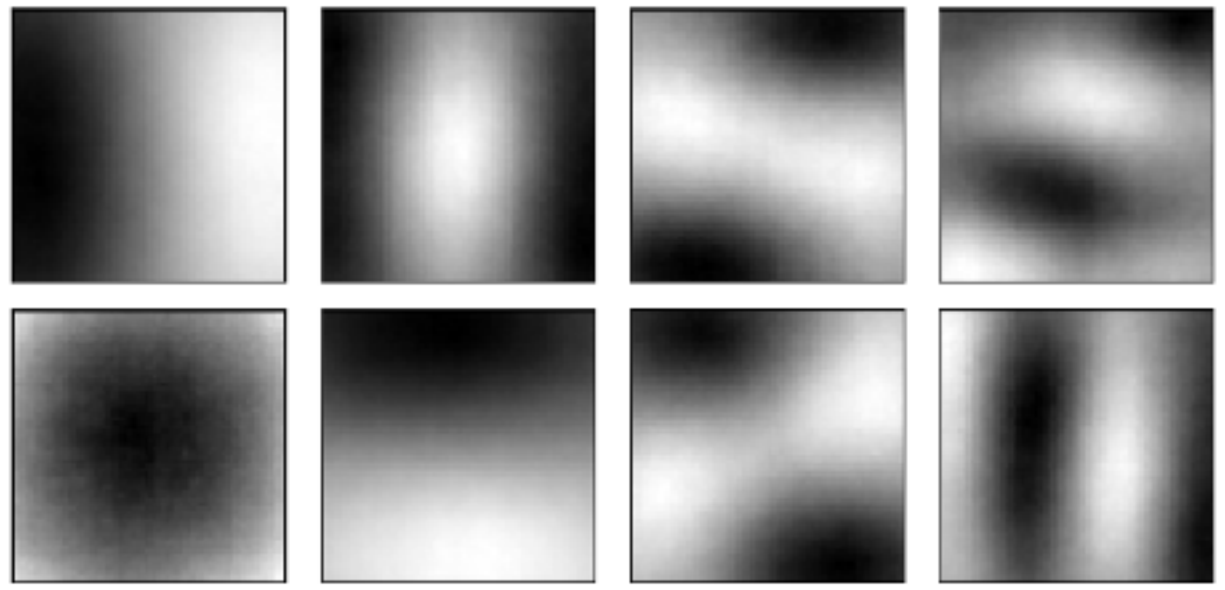
Given a dataset $\mathscr{D} = \{x_1, x_2, \ldots, x_n\}$ of d-dimensional, zero-mean inputs **with identity covariance**

$$w^* := \text{argmax}_{|w|=1} \, \mathbb{E}_{\mathscr{D}} \, G(w \cdot x)$$

Principal components (PCA)

Pearson 1901

$$G(s) = s^2$$

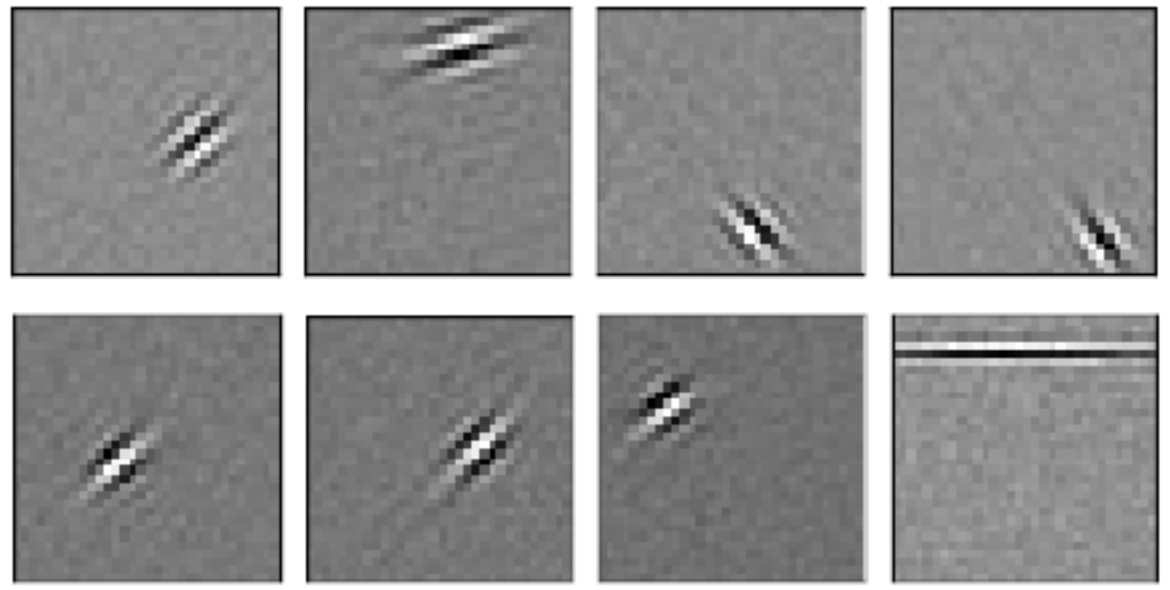


Translation-invariance of images
=> Fourier components

Independent Components (ICA)

Comon ’94; Bell & Sejnowski ’95; Oja & Hyvärinen ’00

$$G(s) = s^4 - 3 \quad e^{-s^2/2}$$



The most **non-Gaussian projections** yield **CNN-like** filters !

Fundamental limits of ICA

A synthetic data model gives fundamental insights

Spiked cumulant model: $x^\mu = \beta g^\mu \mathbf{u} + \mathbf{w}^\mu \qquad g^\mu = \pm 1, \quad \mathbf{w}^\mu \sim \mathcal{N}\left(0, 1 - \beta \mathbf{u}\mathbf{u}^\top\right)$

$\mathbb{E} x_i = 0$

$\mathbb{E} x_i x_j = \delta_{ij}$

PCA is **useless**!

$\mathbb{E} x_i x_j x_k = 0$

$\mathbb{E} x_i x_j x_k x_\ell - \mathbb{E} x_i x_j \mathbb{E} x_k x_\ell [3] \propto (u^{\otimes 4})_{ijk\ell}$

Goal of ICA = finding $\mathbf{u}$!

How to **analyse** this problem?

$$\mathcal{L}(w) := \mathbb{E}_{\mathbb{P}}[G(w \cdot x)] = \mathbb{E}_{\mathbb{P}_0}[G(w \cdot x)\ell(v \cdot x)]$$

Likelihood ratio $\ell(s) := \frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{P}_0}(s)$

- Algorithmic threshold: $n \gtrsim d^2$
  (Auddy & Yuan '24, Annals Apl Prob '24
  Szekely, Bardone, Gerace & SG, NeurIPS '24)
- **How do algorithms actually perform?**

Feature learning from non-Gaussian inputs: the case of Independent Component Analysis in high dimensions

Fabiola Ricci [1] **Lorenzo Bardone** [1] **Sebastian Goldt** [1]

ICML 2025
arXiv:2503.23896

FastICA is slow in high dimensions

The most popular ICA algorithm needs a lot of data

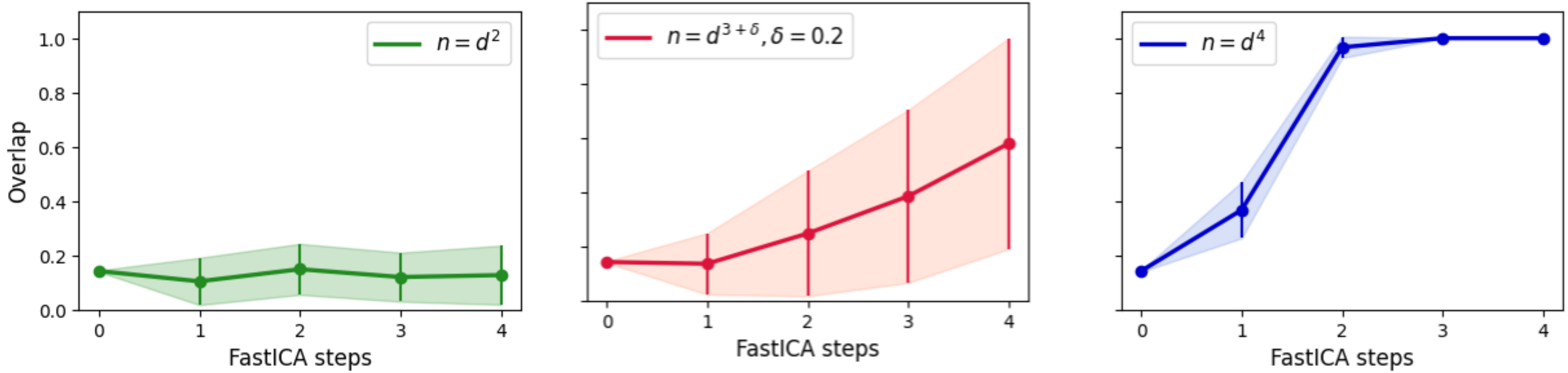
ICA model: $x^\mu = \beta g^\mu \mathbf{u} + \mathbf{w}^\mu \qquad g^\mu = \pm 1, \quad \mathbf{w}^\mu \sim \mathcal{N}\left(0,1 + \beta \mathbf{u}\mathbf{u}^\top\right)$

FastICA Algorithm:

$$\begin{cases} \widetilde{w}_t &= \mathbb{E}_{\mathcal{D}}[x\, G'(w_{t-1} \cdot x)] - \mathbb{E}_{\mathcal{D}}[G''(w_{t-1} \cdot x)] w_{t-1}, \\ w_t &= \widetilde{w}_t / \|\widetilde{w}_t\|. \end{cases}$$

$$G(s) := -\, e^{-s^2/2}$$

$$G(s) := 1/a \log \cosh(as)$$



FastICA is slow in high dimensions

The most popular ICA algorithm needs a lot of data

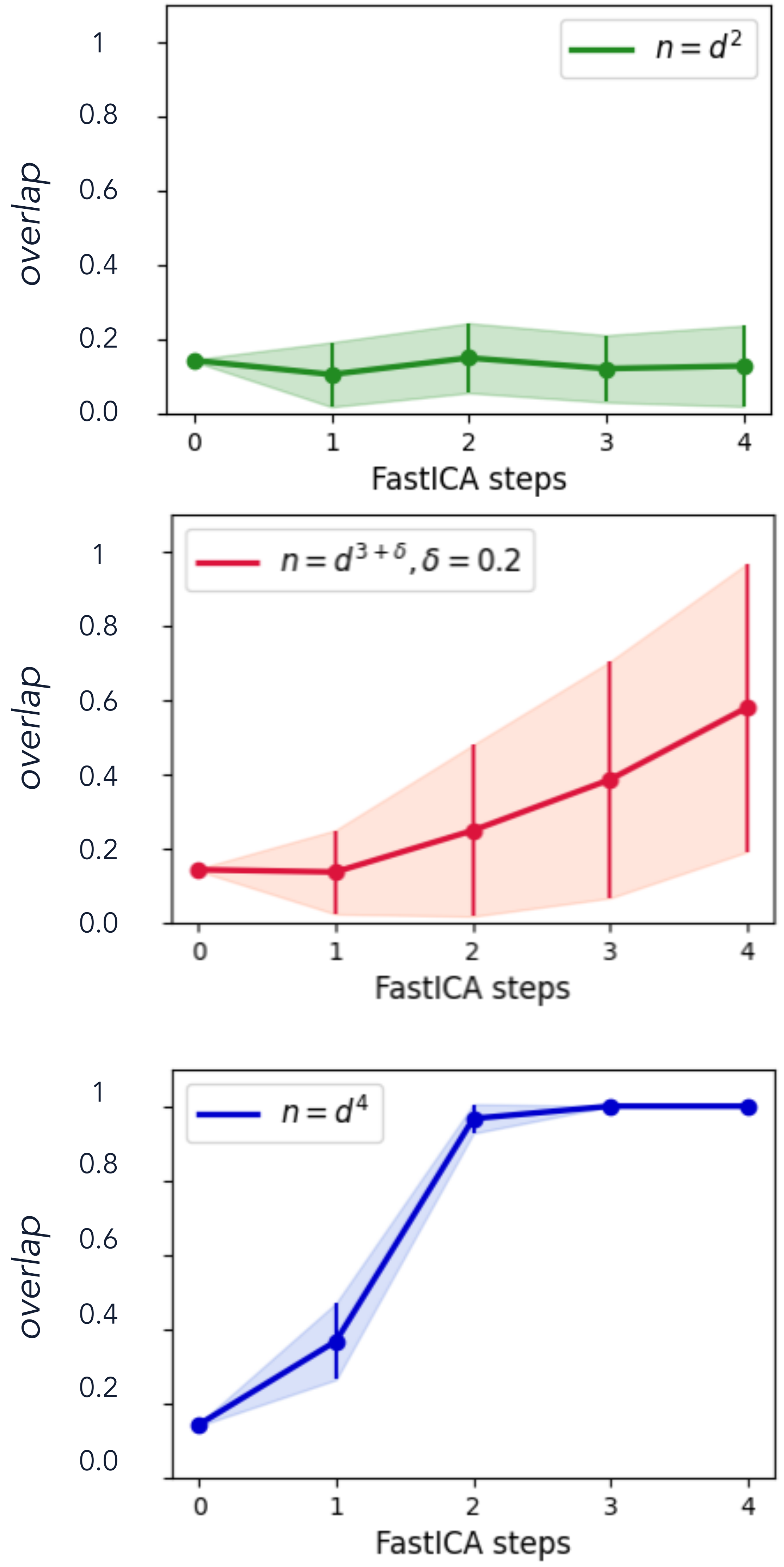
FastICA as a full-batch fixed point iteration:
analyse in the ***giant steps* framework**!
(Ba et al. '22; Damian et al. '24; Dandi et al. '24; Ben Arous et al. '21)

Theorem (informal)**.**
Take $n = d^{\vartheta}$ samples. After one step of FastICA, the overlap α scales as

$$\vartheta \leq 3 \quad \Longrightarrow \quad \alpha^2 = O\left(\frac{1}{d}\right)$$

$$3 < \vartheta < 4 \quad \Longrightarrow \quad \alpha^2 = o(1)$$

$$4 \leq \vartheta \quad \Longrightarrow \quad \alpha^2 = 1 - o(1)$$



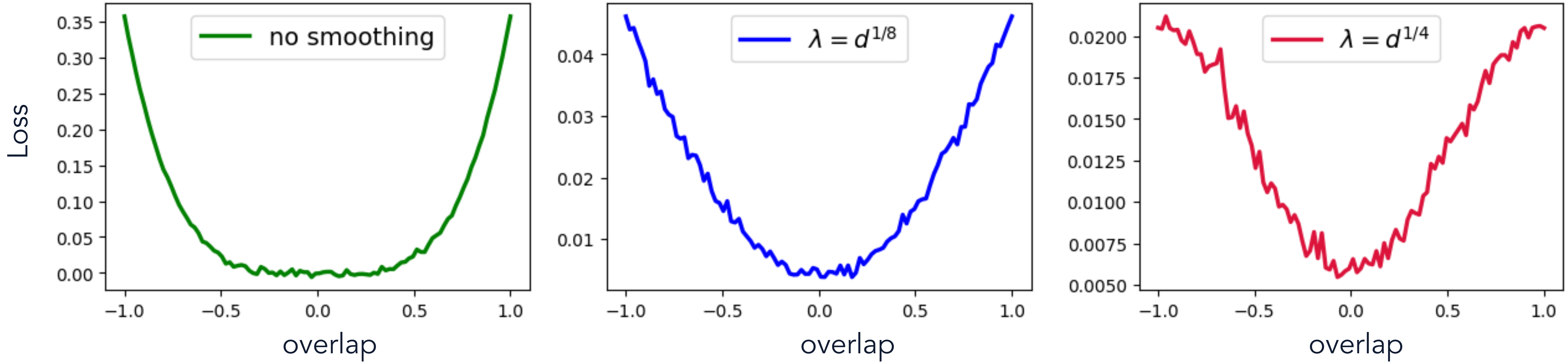
Speeding up ICA with SGD

Smoothing the landscape is the key!

Vanilla SGD (Ben Arous et al. JMLR '21): recovers the spike in $n = \Omega(d^3 \log^2 d)$ steps.

SGD on a **smoothed loss**
Biroli, Cammarota, Ricci-Tersenghi J Phys A '20
Damian *et al.* NeurIPS '23

$$\mathscr{L}_\lambda[G(w \cdot x)] := \mathbb{E}_{z \sim \mu_w} G\left(\frac{w + \lambda z}{\|w + \lambda z\|} \cdot x\right) \qquad \lambda \geq 0$$



Speeding up ICA with SGD

Smoothing the landscape is the key!

Vanilla SGD (Ben Arous et al. JMLR '21): recovers the spike in $n = \Omega(d^3 \log^2 d)$ steps.

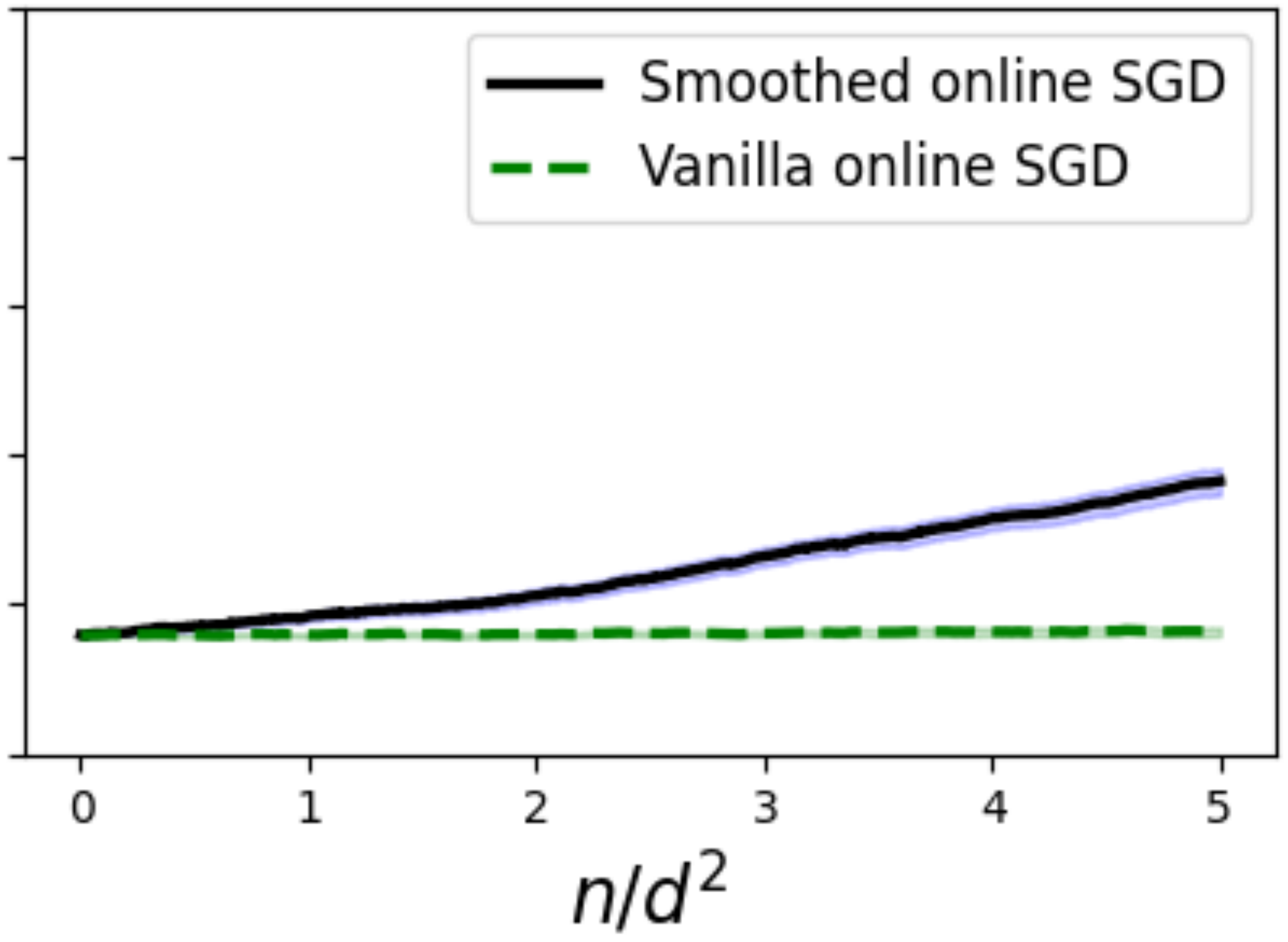
SGD on a **smoothed loss**
Biroli, Cammarota, Ricci-Tersenghi J Phys A '20
Damian *et al.* NeurIPS '23

$$\mathscr{L}_\lambda[G(w \cdot x)] := \mathbb{E}_{z \sim \mu_w} G\left(\frac{w + \lambda z}{\|w + \lambda z\|} \cdot x\right) \qquad \lambda \geq 0$$

Generalised ODE for the overlap:
(accounting for data/contrast fn mismatch)

$$m'(t) = \frac{m(t)}{d^{k_1^* - k_2^*/2}}$$

- Speed-up requires fine-tuning of "activation" function!
- Optimal choice for spiked cumulant is $\mathrm{He}_4(s)$. Matches LDLR bound!
- Trade-off: stability vs. speed!

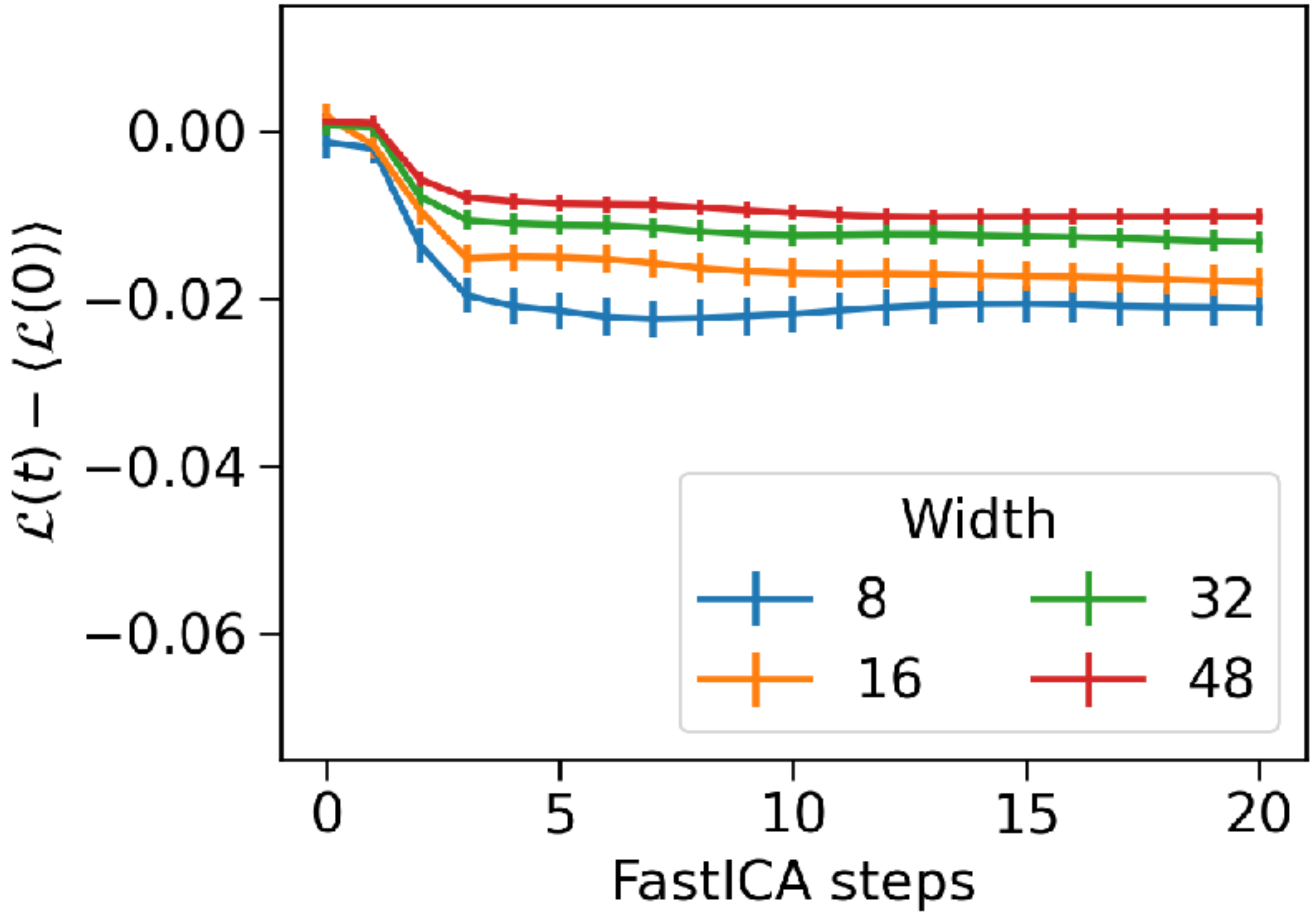


ICA is a **hard** problem in high dimensions.

So what happens on **real images** with deep **neural networks**?

What about real data?

FastICA fails on real images at linear sample complexity



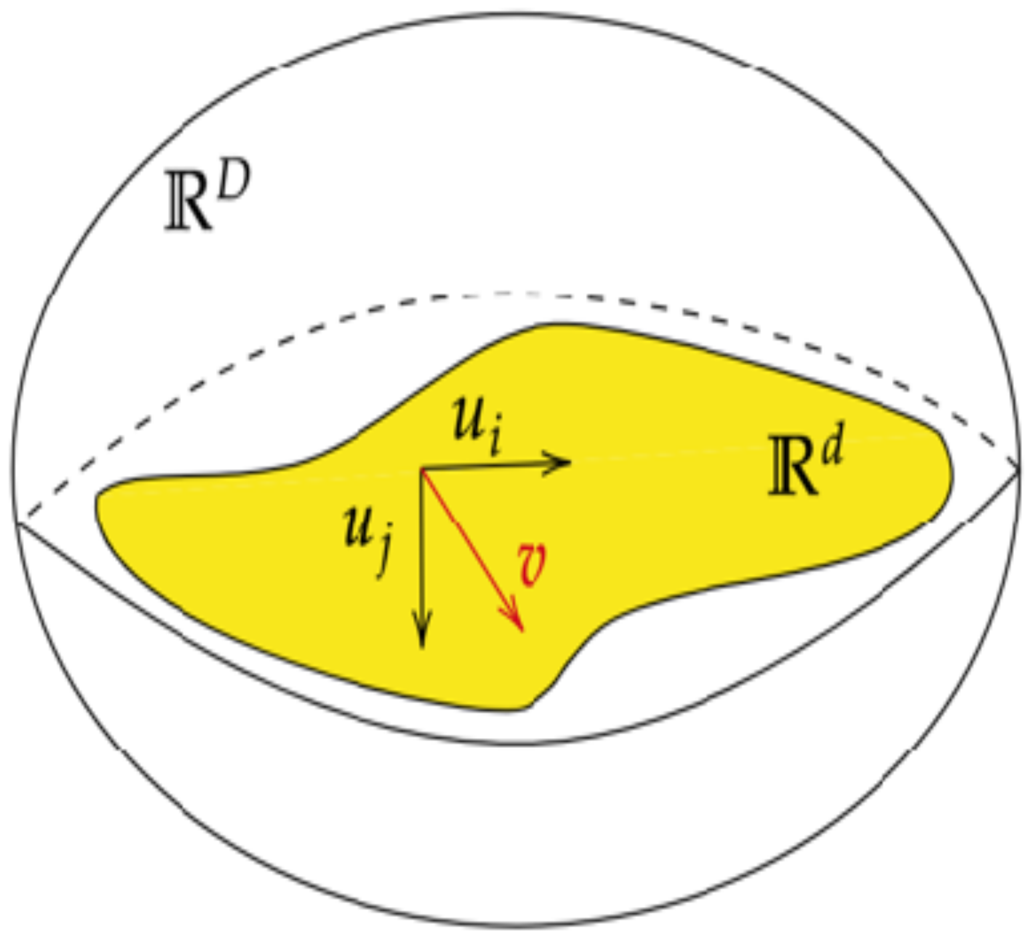
FastICA, logcosh activation,
$n = 2D$, d=D (left) vs. d=32 (right)

Reduce and conquer

Reduce the dimension, conquer with ICA

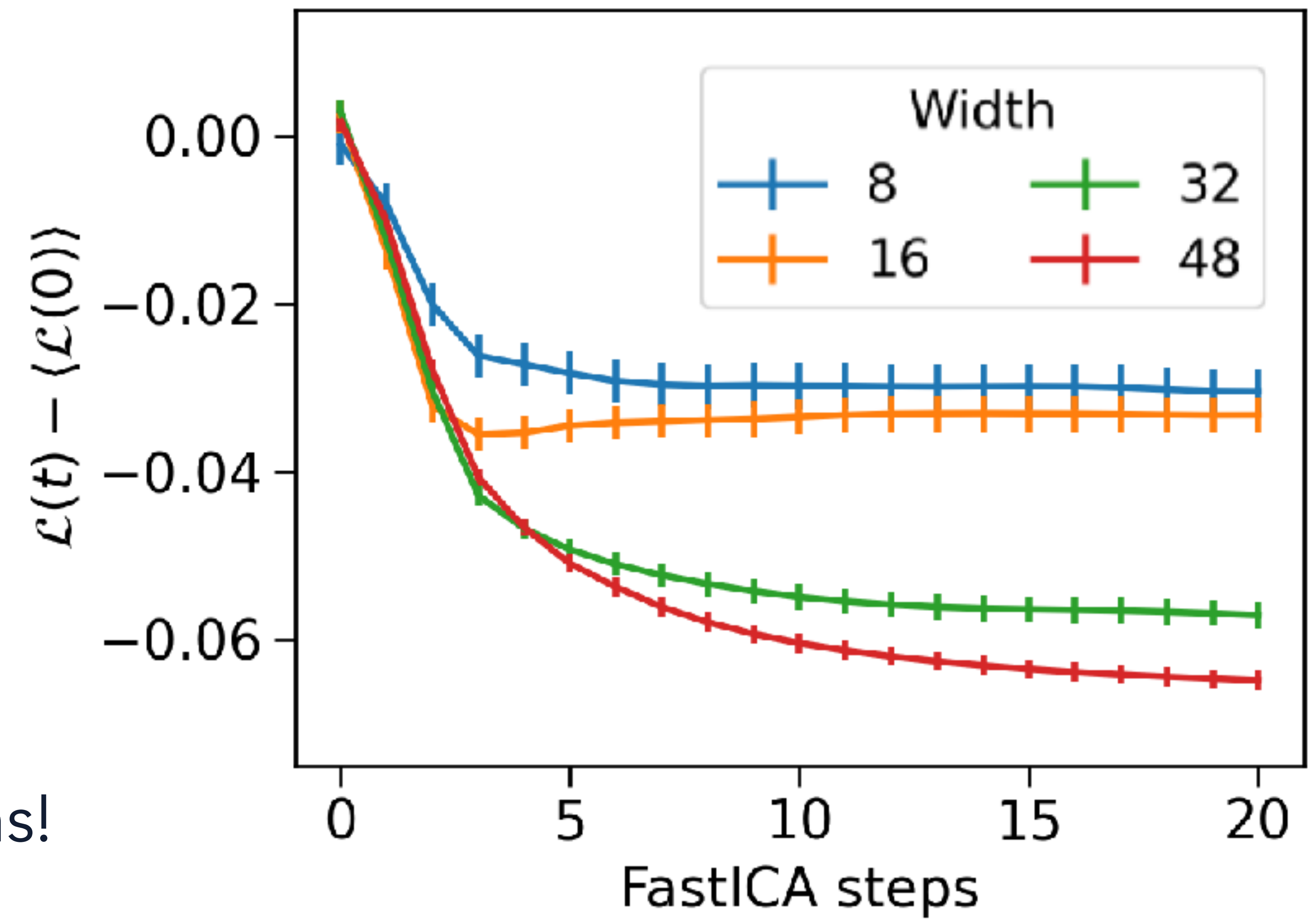
Project inputs to principal subspace

Hyvärinen '99



Run **FastICA** in **low dimensions**

Loss in **high** dimensions!



- Success reveals something about the **structure of the images**

- Linear Sample complexity can be proven in "**subspace model**"

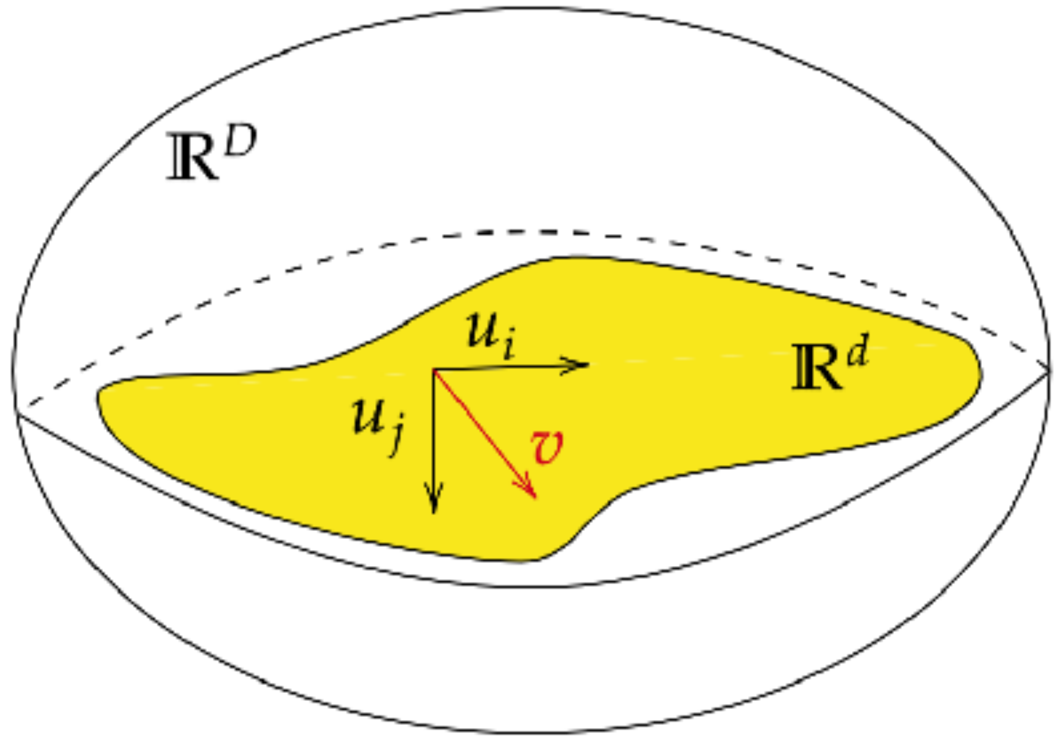
What about real data?

A mixed matrix-tensor model

Subspace model (rank-1 in Bardone & SG, ICML '24)

$$\mathbf{x}^\mu = \sum_r \beta_1 g_r^\mu \mathbf{u}_r + \beta_2 h^\mu \mathbf{v} + \mathbf{w}^\mu$$

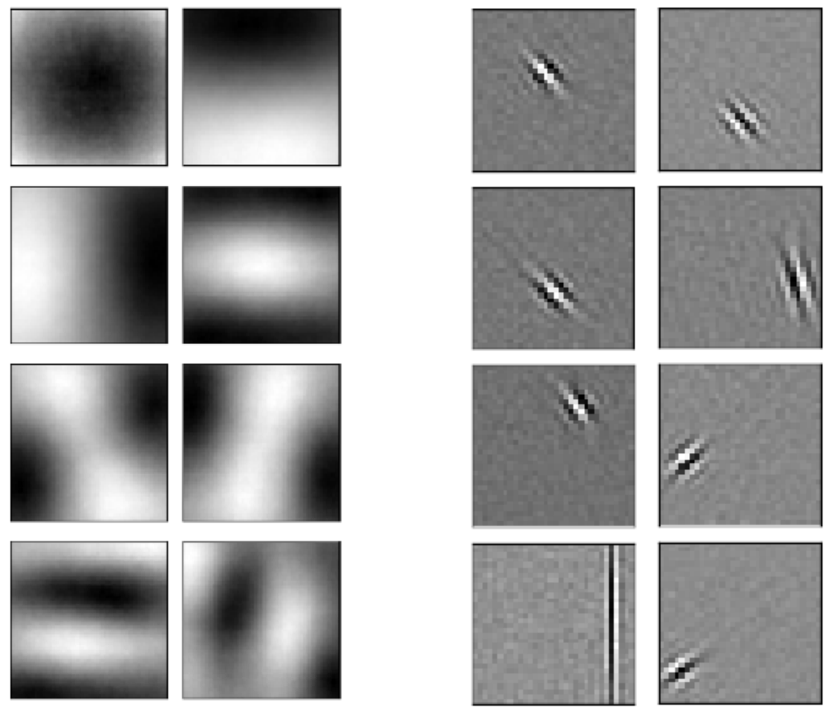
$$g_r^\mu \sim \mathcal{N}(0,1), \quad h^\mu = \pm 1$$
$$\mathbf{w}^\mu \sim \mathcal{N}\left(0, 1 - \beta_2 \mathbf{v}\mathbf{v}^\top\right)$$



Prove recovery by analysing GD in finite-dimensional sub-space spanned by PCs

Mixed matrix-tensor models

- Richard & Montanari (NeurIPS '14). observe $X = \beta v^{\otimes p} + Z$ and $y = \beta x + z$, $\beta > 0$
- Sarao Mannelli et al. ('19a, '19b, '20) observe $M \propto vv^\top + Z_M$ and $T \propto v^{\otimes p} + Z_T$
  - Asymmetric case: Tabanelli et al. arXiv:2506.02664



Ricci, Bardone, SG — in prep.

What about (shallow) neural networks?

Learning distributions of increasing complexity

A. Ingrosso

vs.

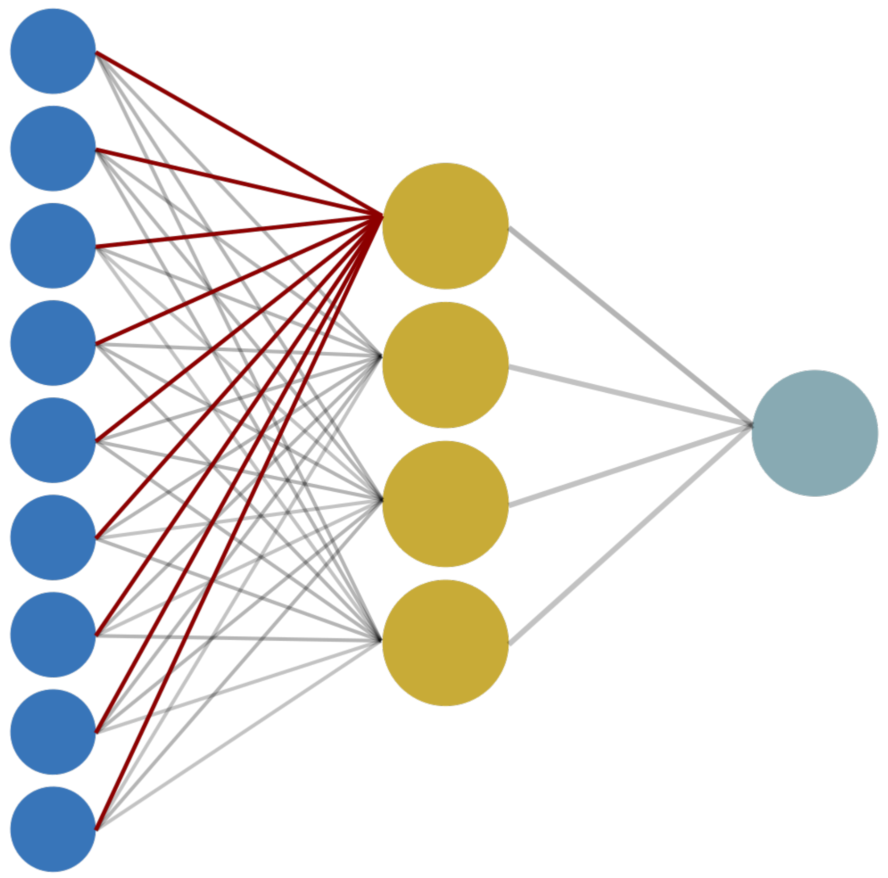
Translation-invariance:

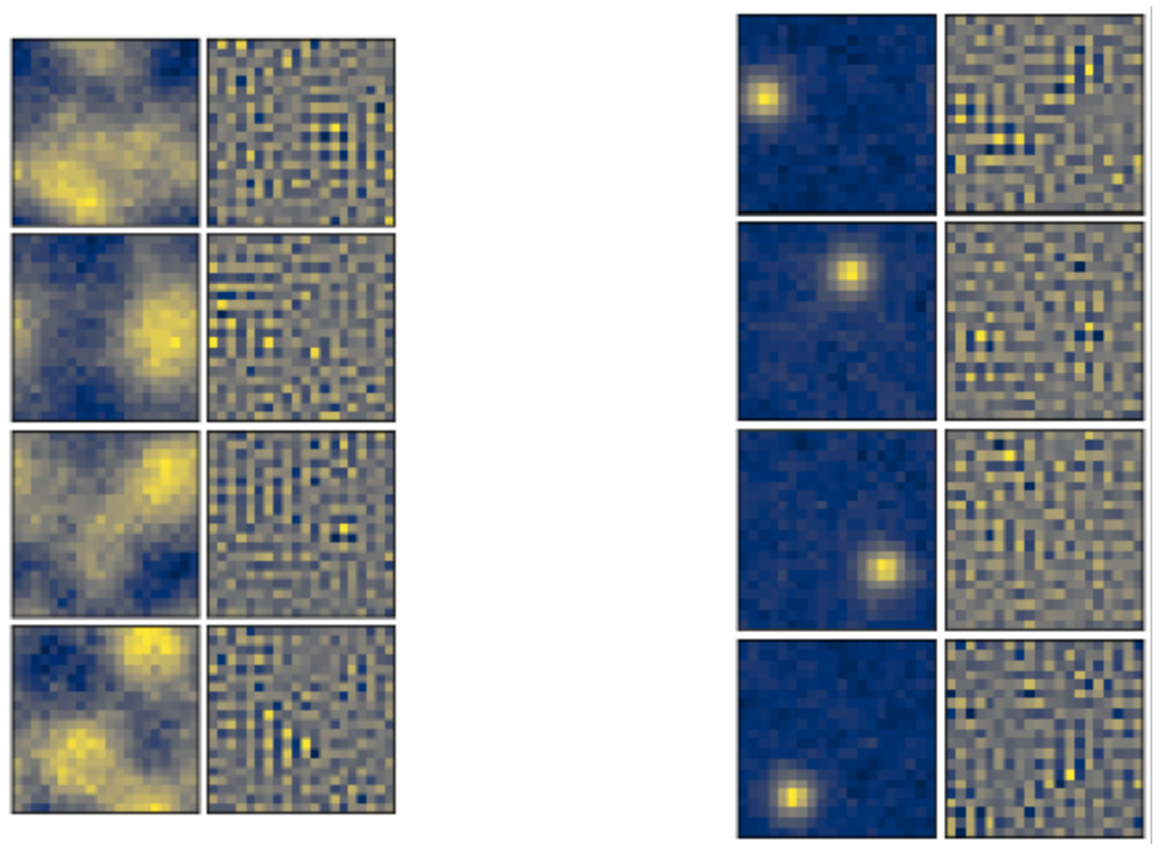
$$\mathbb{E} z_k^\pm z_l^\pm = \exp\left(-|k-l|/\xi^\pm\right)$$

Sharp edges:

(from saturating non-linearity)

$$x_j^\pm \propto \mathrm{erf}\left(g z_j^\pm\right)$$





$t \approx 10^1$ $\quad$ $t \approx 10^4$

- **Early** in training: neurons ≈ Fourier modes, doing **PCA**
- **Later** in training, neurons become localised, doing ≈ **ICA**

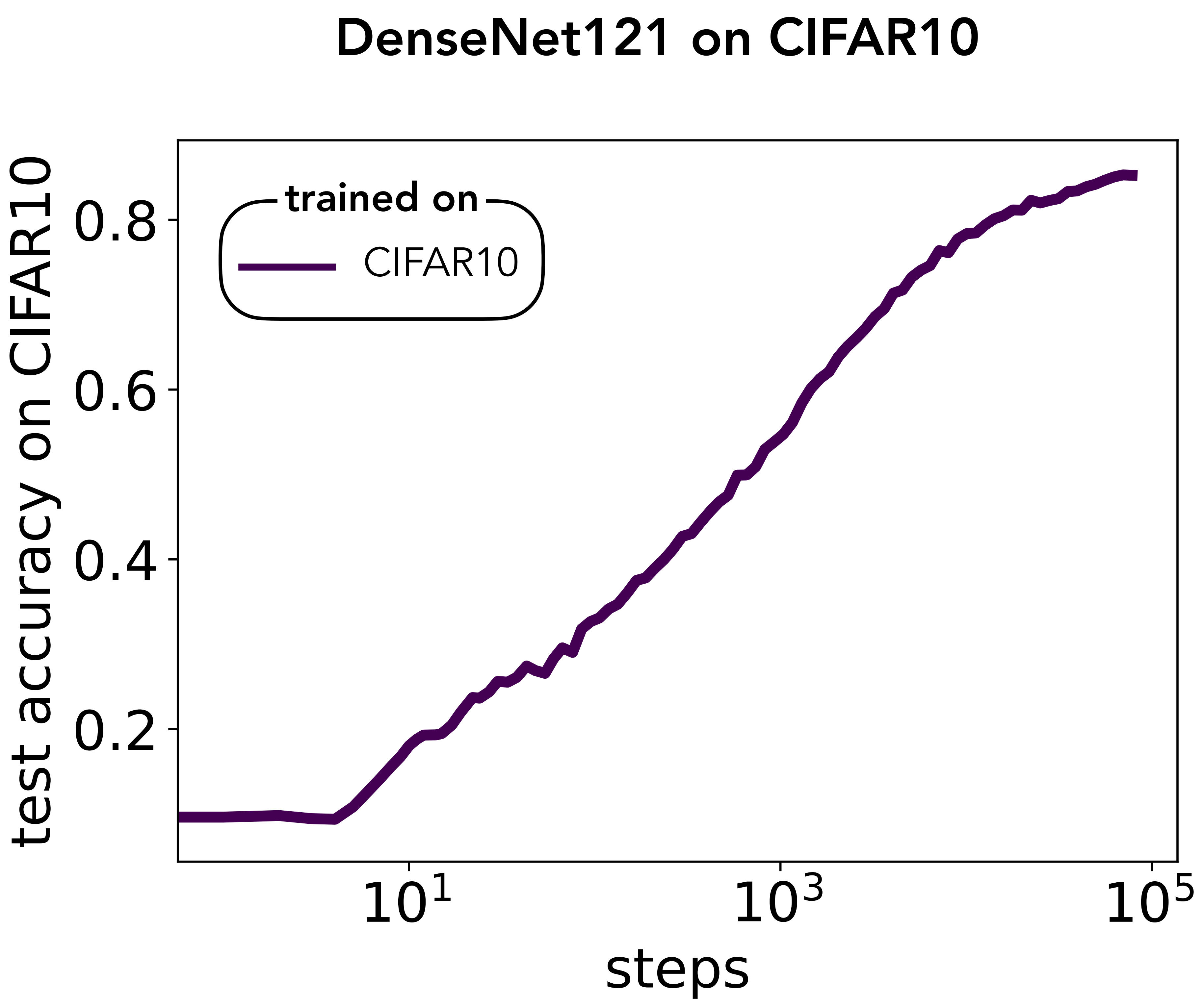
Sequential learning !

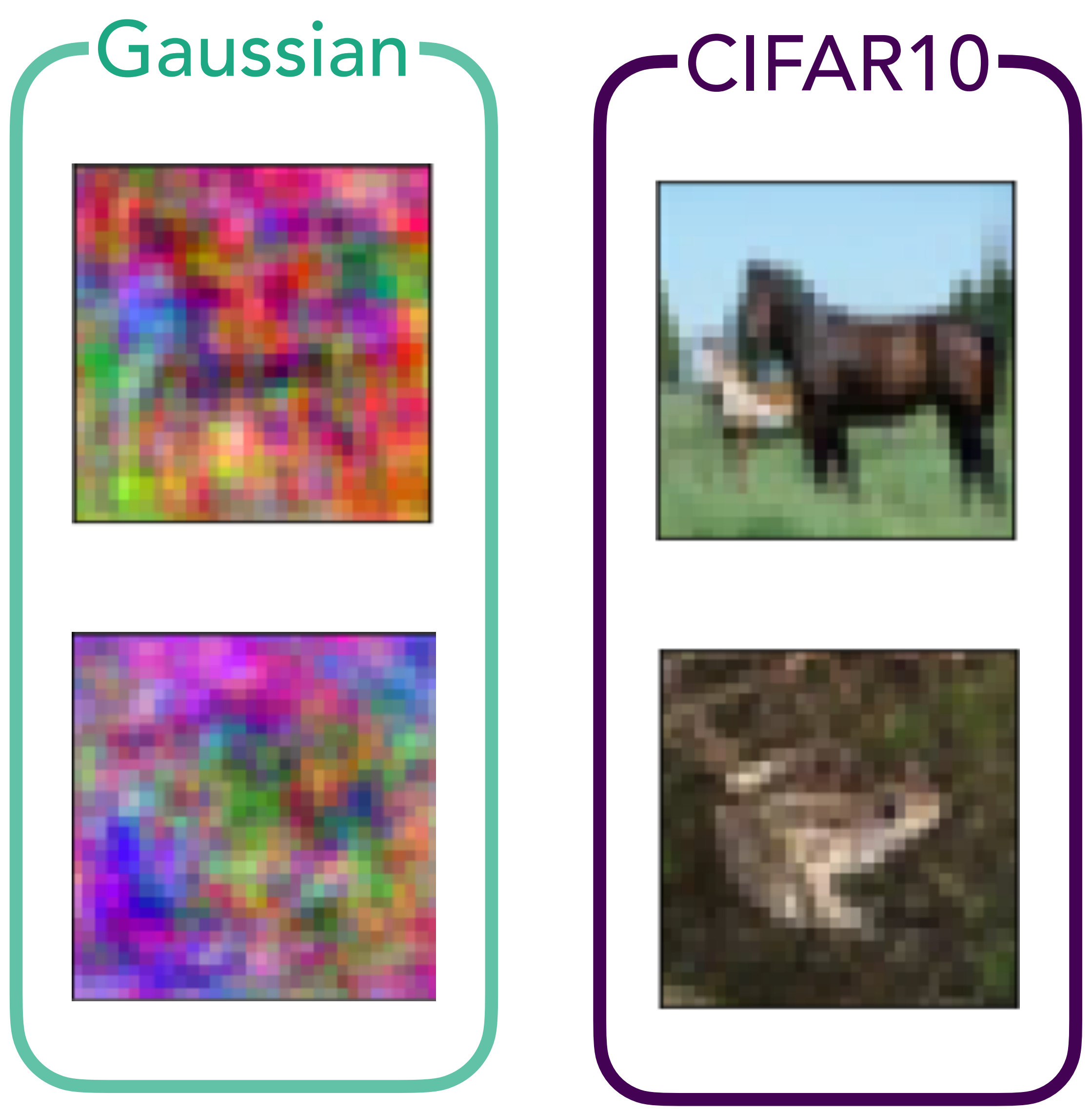
What about deep networks?

Learning distributions of increasing complexity

What about deep networks?

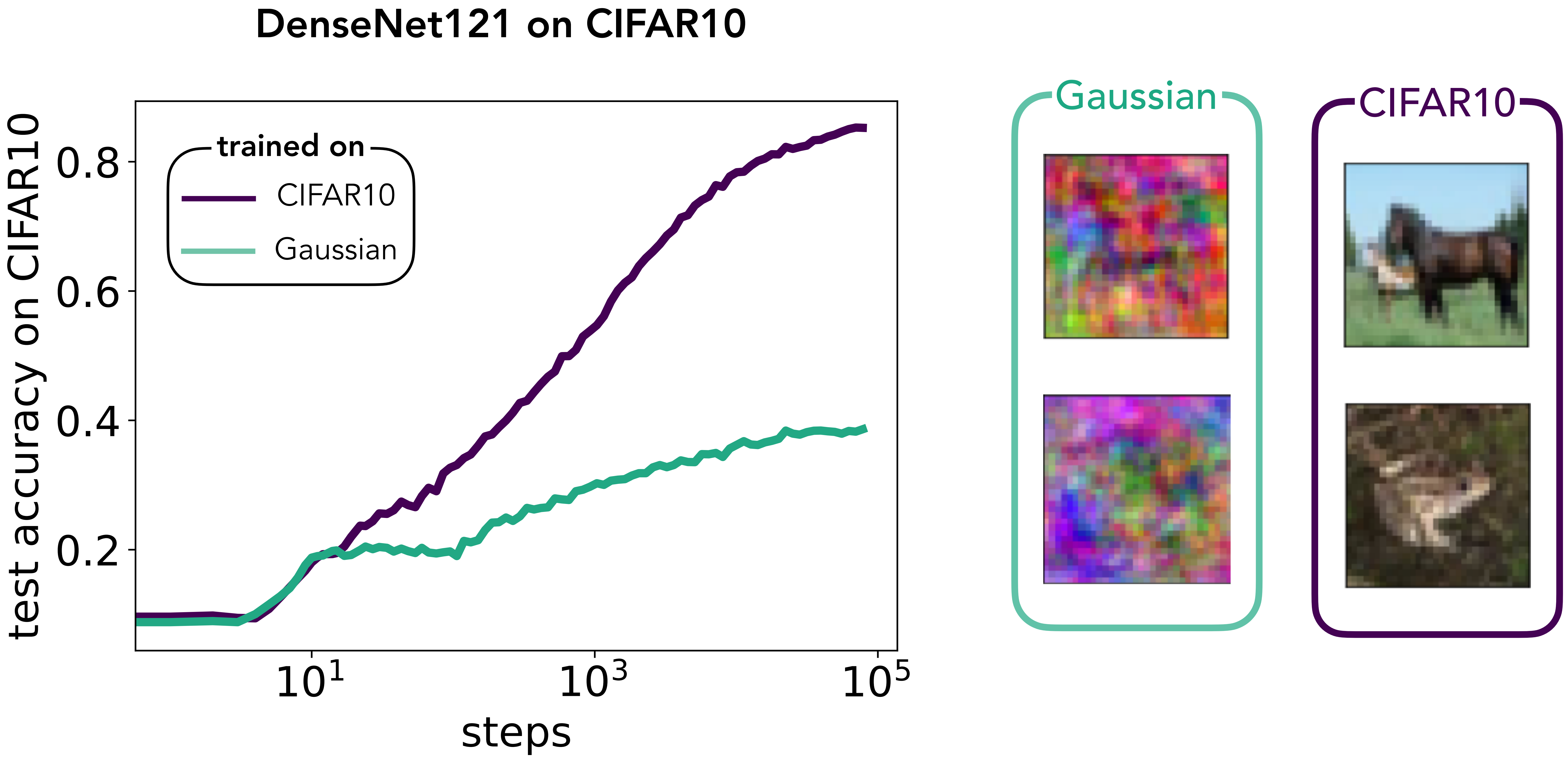
Learning distributions of increasing complexity





What about deep networks?

Learning distributions of increasing complexity

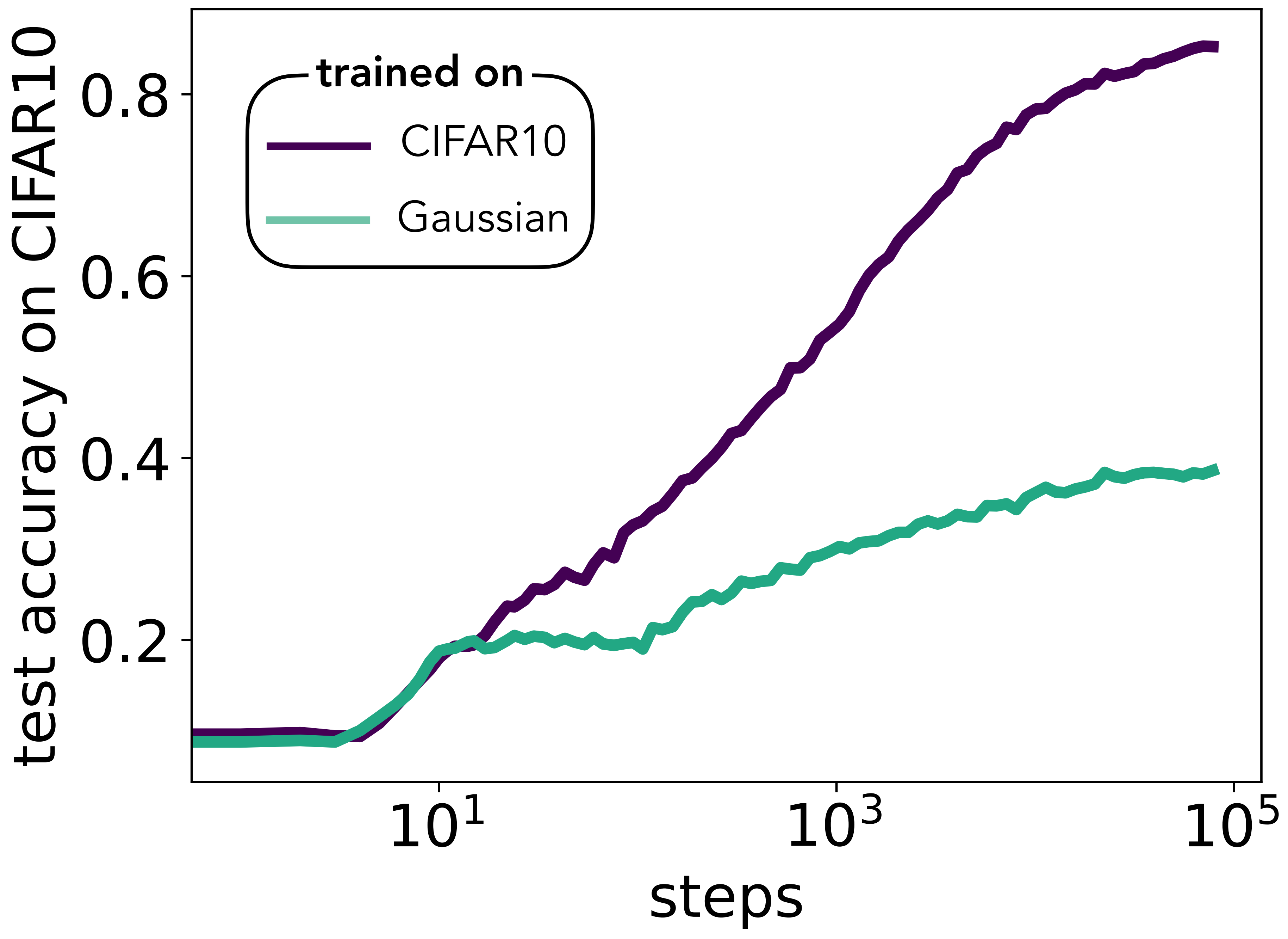


Refinetti, Ingrosso, SG — ICML 2023

What about deep networks?

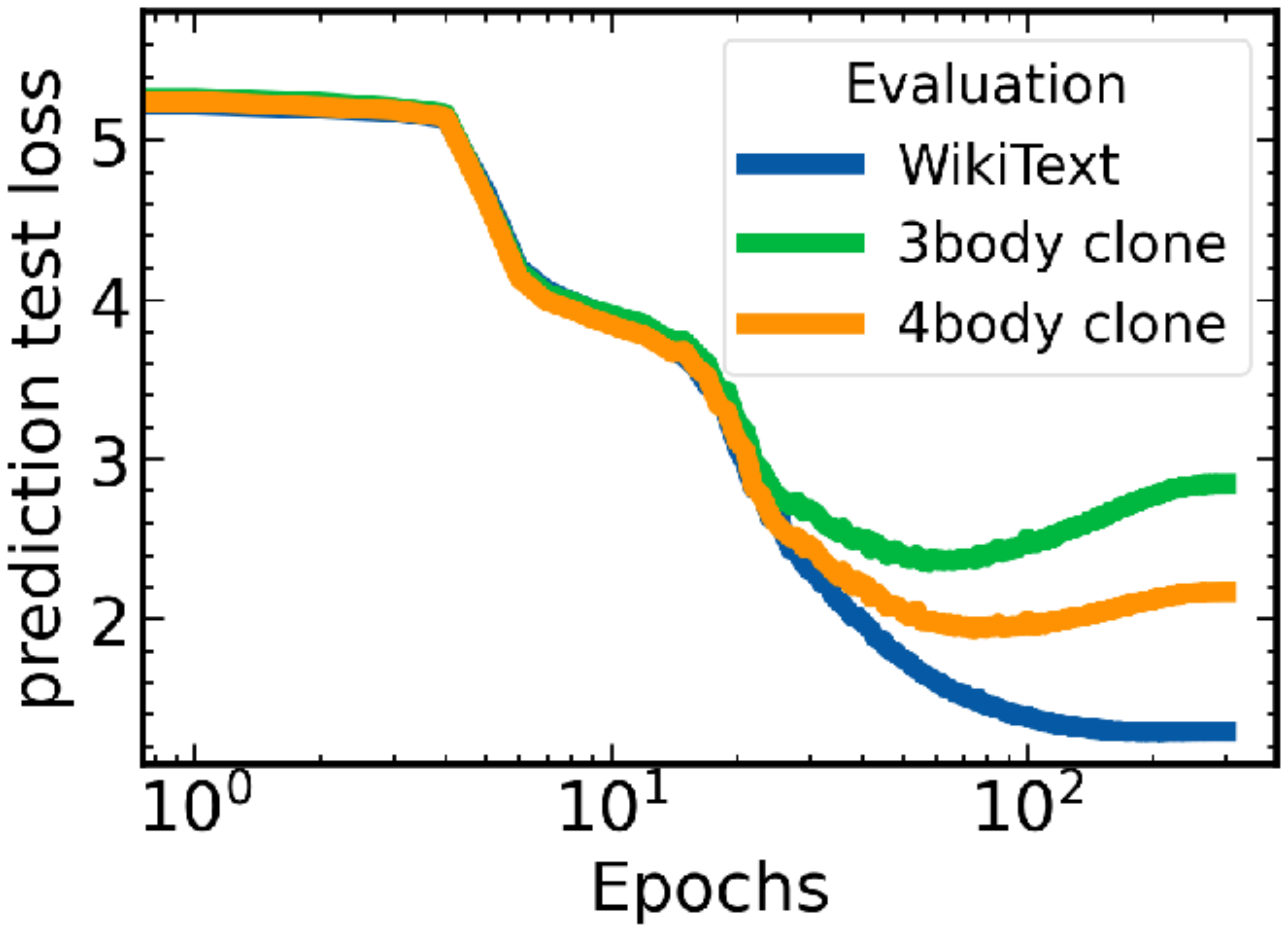
Learning distributions of increasing complexity

DenseNet121 on CIFAR10



Refinetti, Ingrosso & SG — ICML '23

Vanilla Transformer on WikiText101

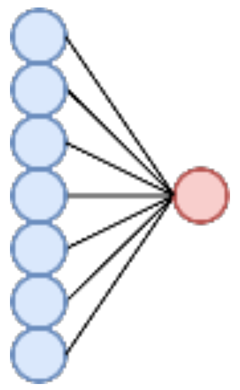


Rende, Gerace, Laio, SG
NeurIPS '24

Connected subspaces accelerate neural networks

Rigorous analysis for a spherical perceptron

Spherical perceptron



$\hat{y}(x) = \sigma(w \cdot x)$

$$\begin{cases} w_0 \sim \text{Unif}\left(\mathbb{S}^{d-1}\right) \\ \tilde{w}_t = w_{t-1} - \frac{\delta}{d} \nabla_{\text{sph}}\left(\mathscr{L}(w, (x_t, y_t))\right) \\ w_t = \frac{\tilde{w}_t}{||\tilde{w}_t||} . \end{cases}$$

Correlation loss

$$\mathscr{L}(w, (x, y)) = 1 - y f(w, x) .$$

Mixed cumulant model: $\mathbf{x}^\mu = \mathbf{w}^\mu$ vs. $\mathbf{x}^\mu = \beta_1 g^\mu \mathbf{u} + S\left(\beta_2 h^\mu \mathbf{v} + \mathbf{w}^\mu\right)$

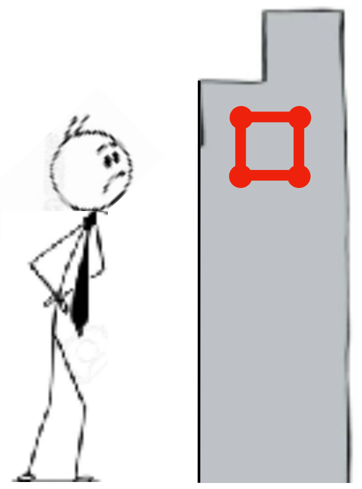
Two overlaps: $m_u = u \cdot w$ $m_v = v \cdot w$

Disconnnected subspaces:

Ben Arous et al. '21

$$m_v'(t) \approx 4 c_{04} m_v^3$$

$n_v \gg d^3$



Correlated latents (=connected subspaces)

$$m_v'(t) \approx c_{11} m_u + 4 c_{04} m_v^3$$

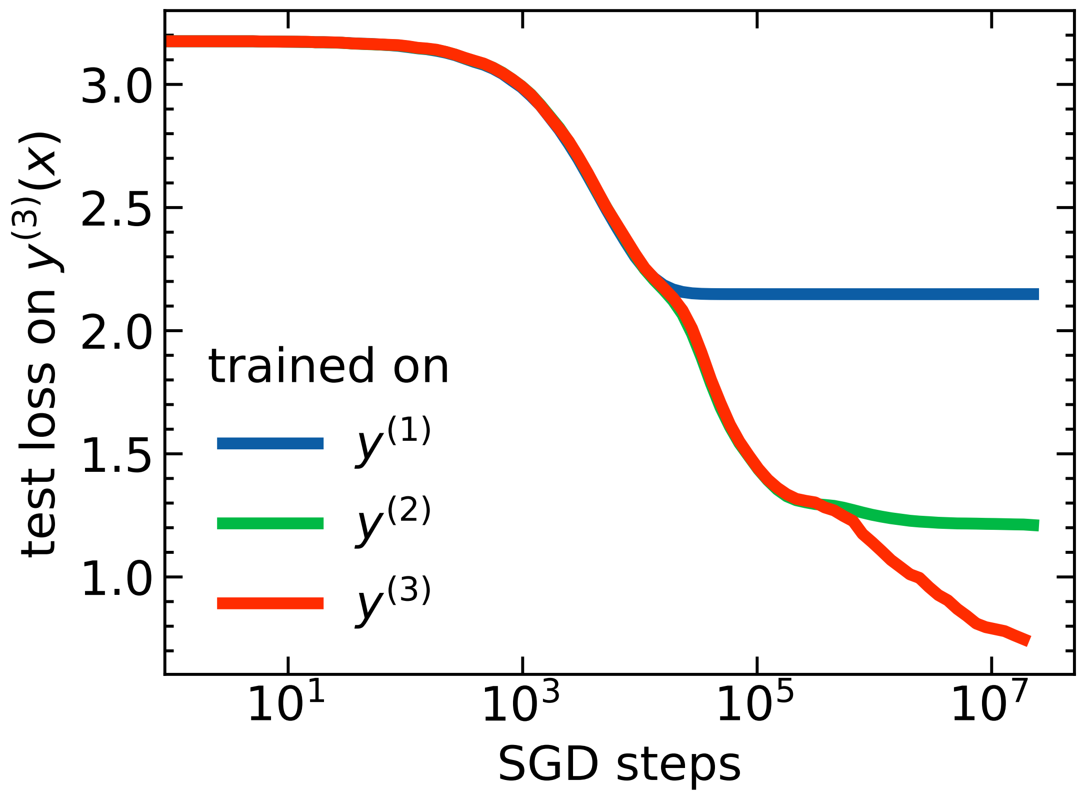
$n_v \gg d$

Relation to teacher-student models

Staircases, staircases everywhere!

Teacher-student model: $x \sim \mathcal{N}(0, 1_d)$

$$y^*(x) = h_1(m \cdot x) + h_2(u \cdot x) + h_4(v \cdot x)$$



Abbé ’21, ’22, ’23; Jacot et al. ’21; Boursier et al. ’22; Dandi et al. ’23; Damian et al. ’23; Bietti et al. ’23; Mousavi-Hosseini et al. ’24

Key difference between spiked cumulants and teacher-student:

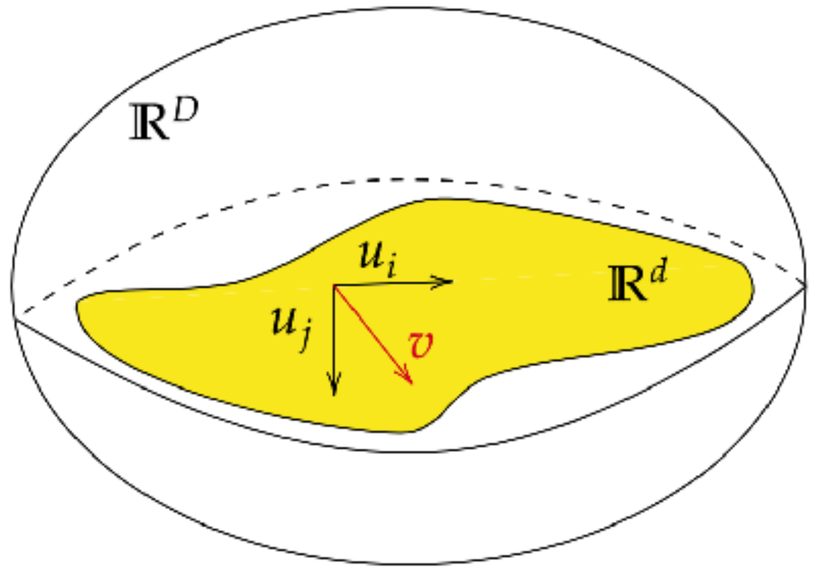
- **Generative exponent** [Damian et al. ’24] of any **polynomial** is at most 2, so $y^*(x)$ can be learnt at linear sample complexity (e.g. by repeating batches [Dandi et al. ’24])
- The generative exponent of the **spiked cumulant model** is at least four, since for binary labels, there is no transform T such that $\mathbb{E}\left[h_{1/2/3}(x) \,|\, T(y)\right] \neq 0$.

Concluding perspectives

How do neural networks learn from their data, efficiently?

- Neural networks learn features from **higher-order correlations.**

- ICA as a model system reveals the crucial role of **sequential learning** to access HOCs.



- We find similar behaviour in **deep CNNs**.

- Key challenge: towards more realistic models of **unsupervised learning**?!

Acknowledgements

Lorenzo **Bardone**
(SISSA)

Fabiola **Ricci**
(SISSA)

We're hiring!

datascience.sissa.it

@sebastiangoldt

Finanziato
dall'Unione europea
NextGenerationEU

erc
European Research Council
Established by the European Commission

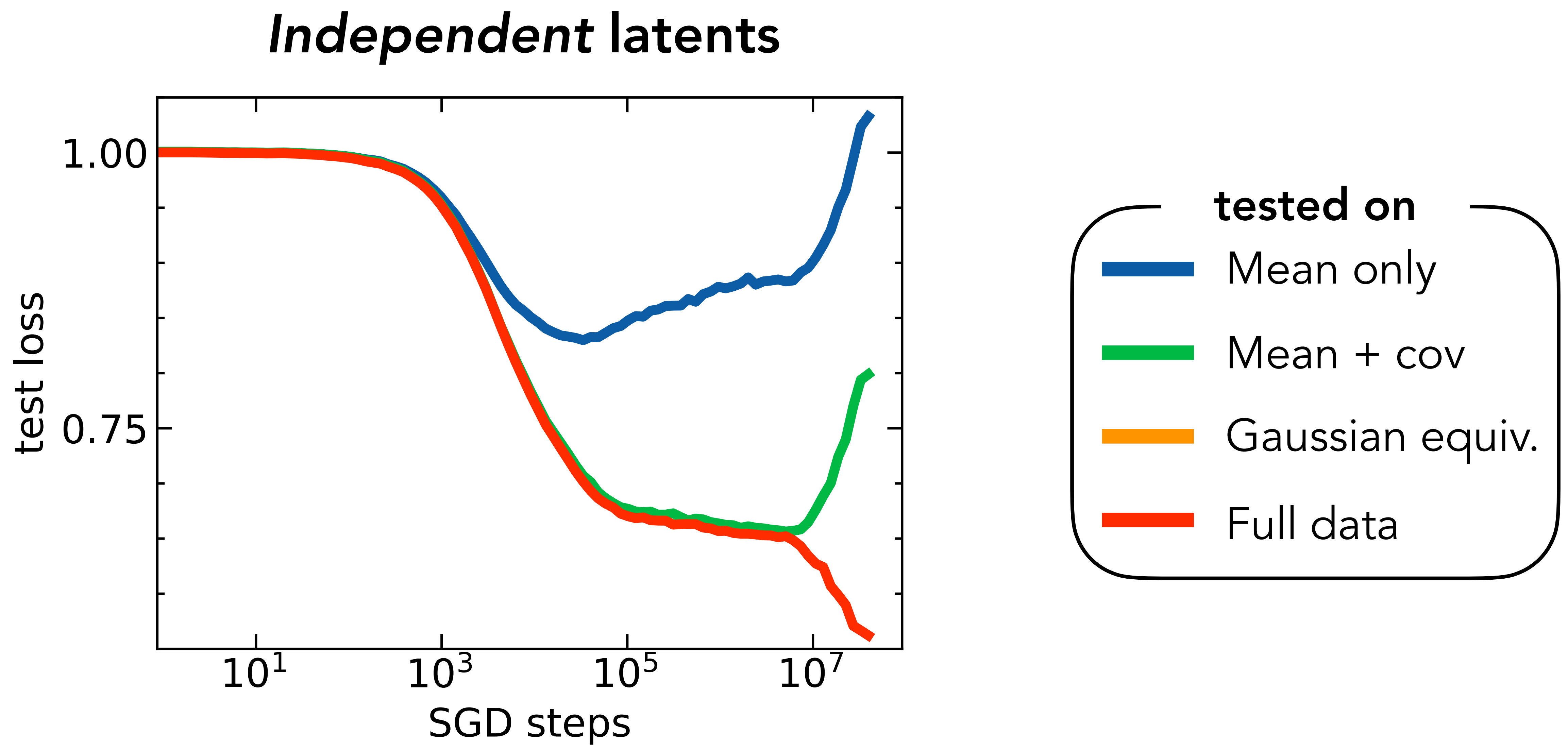
Connected subspaces accelerate neural networks

Two-layer neural networks exploit correlations between subspaces

Classification task: $\mathbf{x}^\mu = \mathbf{w}^\mu$ vs. $\mathbf{x}^\mu = \beta_0 \mathbf{m} + \beta_1 g^\mu \mathbf{u} + \beta_2 h^\mu \mathbf{v} + \mathbf{w}^\mu$

Three spikes: $\mathbf{m}, \mathbf{u}, \mathbf{v}$

Two latent variables: $g^\mu \sim \mathcal{N}(0,1)$, $h^\mu = \pm 1$



Bardone, SG — ICML 2024

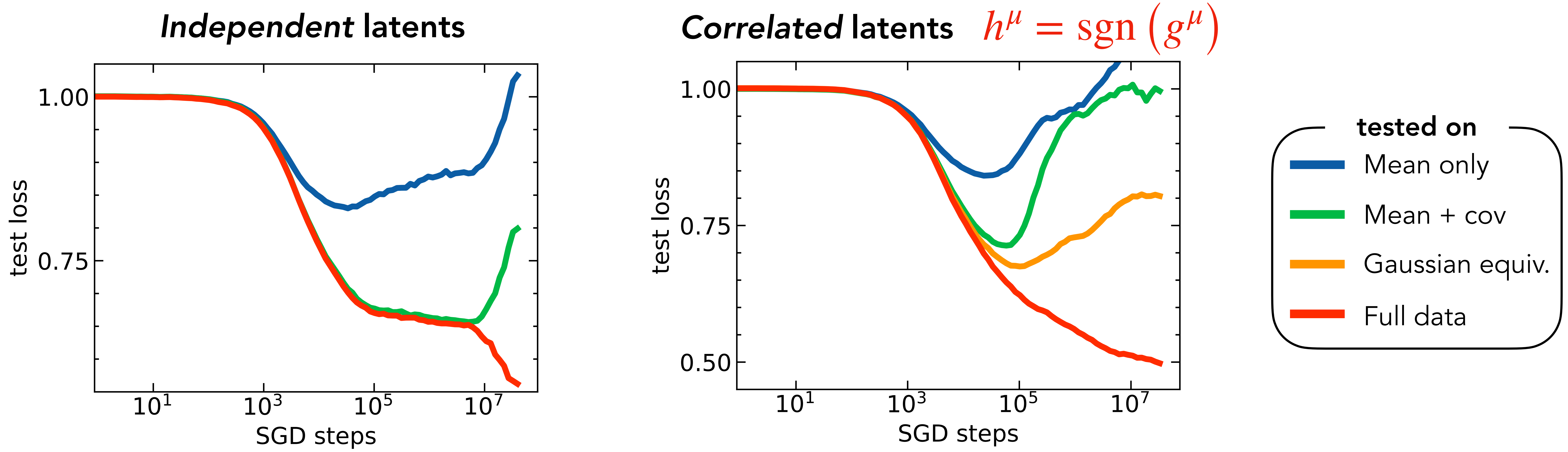
Connected subspaces accelerate neural networks

Two-layer neural networks exploit correlations between subspaces

Classification task: $\mathbf{x}^\mu = \mathbf{w}^\mu$ vs. $\mathbf{x}^\mu = \beta_0\mathbf{m} + \beta_1 g^\mu \mathbf{u} + \beta_2 h^\mu \mathbf{v} + \mathbf{w}^\mu$

Three spikes: $\mathbf{m}, \mathbf{u}, \mathbf{v}$

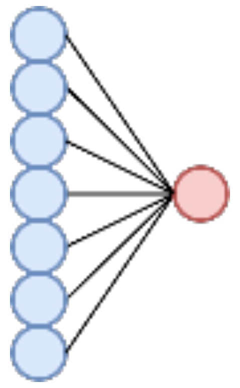
Two latent variables: $g^\mu \sim \mathcal{N}(0,1)$, $h^\mu = \pm 1$



Connected subspaces accelerate neural networks

Rigorous analysis for a spherical perceptron

Spherical perceptron



$\hat{y}(x) = \sigma(w \cdot x)$

$$\begin{cases} w_0 \sim \text{Unif}\left(\mathbb{S}^{d-1}\right) \\ \tilde{w}_t = w_{t-1} - \frac{\delta}{d} \nabla_{\text{sph}} \left(\mathcal{L}(w, (x_t, y_t))\right) \\ w_t = \frac{\tilde{w}_t}{||\tilde{w}_t||} \,. \end{cases}$$

Correlation loss

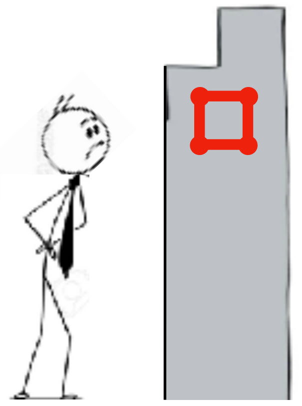
$$\mathcal{L}(w, (x, y)) = 1 - yf(w, x) \,.$$

Mixed cumulant model: $\mathbf{x}^\mu = \mathbf{w}^\mu$ vs. $\mathbf{x}^\mu = \beta_1 g^\mu \mathbf{u} + S\left(\beta_2 h^\mu \mathbf{v} + \mathbf{w}^\mu\right)$

Two overlaps: $\alpha_u = u \cdot w, \quad \alpha_v = v \cdot w$

Cumulant spike only: $g^\mu = 0; \quad h^\mu = \pm 1 \quad \Rightarrow \quad n_v \gg d^3$

Ben Arous et al. '21



Correlated latents: $g^\mu \sim \mathcal{N}(0,1); \quad h^\mu = \text{sgn}\left(g^\mu\right)$

$$\begin{cases} \dot{\alpha}_u(t) = 2c_{20}\alpha_u + c_{11}\alpha_v + O(\eta^2) \\ \dot{\alpha}_v(t) = c_{11}\alpha_u + 4c_{04}\alpha_v^3 - 2c_{20}\alpha_u^2\alpha_v + O(\eta^4) \end{cases}$$

For correlated latents $\mathbb{E}\lambda\nu > 0$, the coefficient $c_{11} > 0$ and $n_v \gg d$

Bardone, SG — ICML 2024