Parton Distributions from NTK analysis

Amedeo Chiefa

In Collaboration with L. Del Debbio and R. Kenway

Higgs Centre of Theoretical Physics
The University of Edinburgh

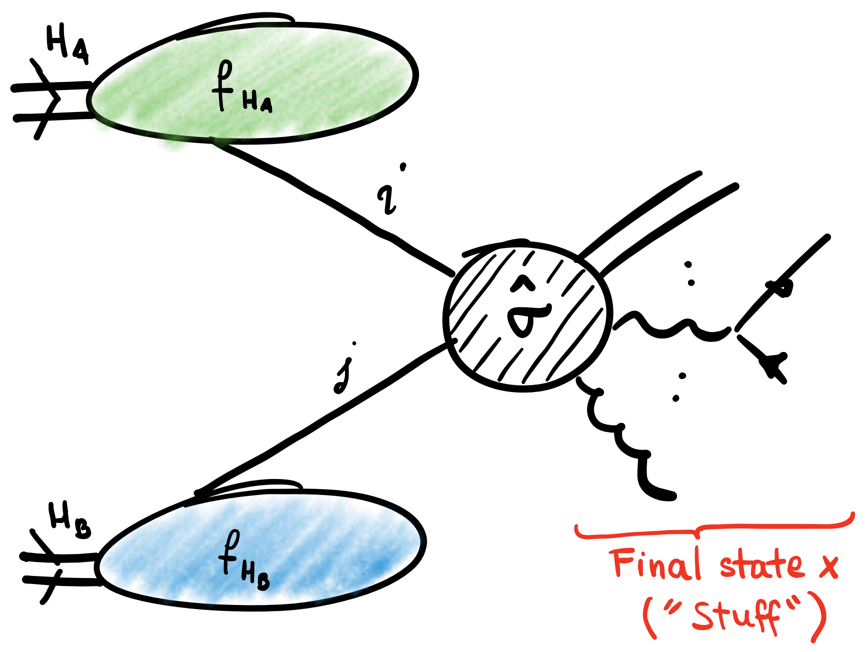
NPTwins 2025, Università di Messina

THE UNIVERSITY of EDINBURGH

The Backstory

Phenomenology and High-Precision Physics at LHC

Proton-proton collisions at particle colliders are predicted using the **factorisation theorem**



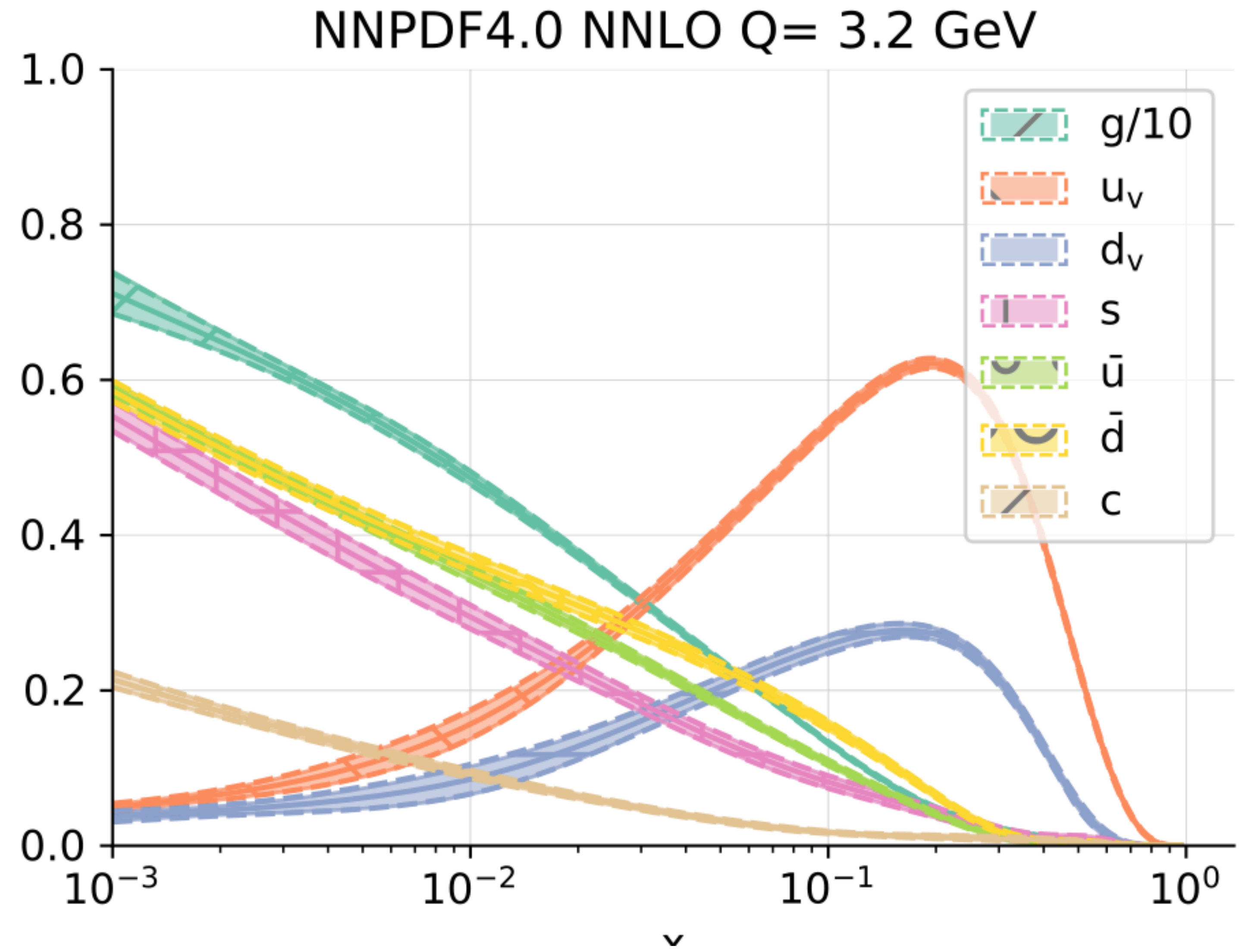
$$\sigma = \sum_{i,j} \int_0^1 dx_A \int_0^1 dx_B \; f_{i/H_A}(x_B) \; f_{j/H_B}(x_B) \; \hat{\sigma}_{ij}(x_A, x_B)$$

PDFs are essential for LHC predictions

They can be viewed as the p.d.f. of a parton entering a collision with a momentum fraction x

They cannot be determined in perturbative QCD

Their shape must be inferred from data



Current landscape of PDF determination

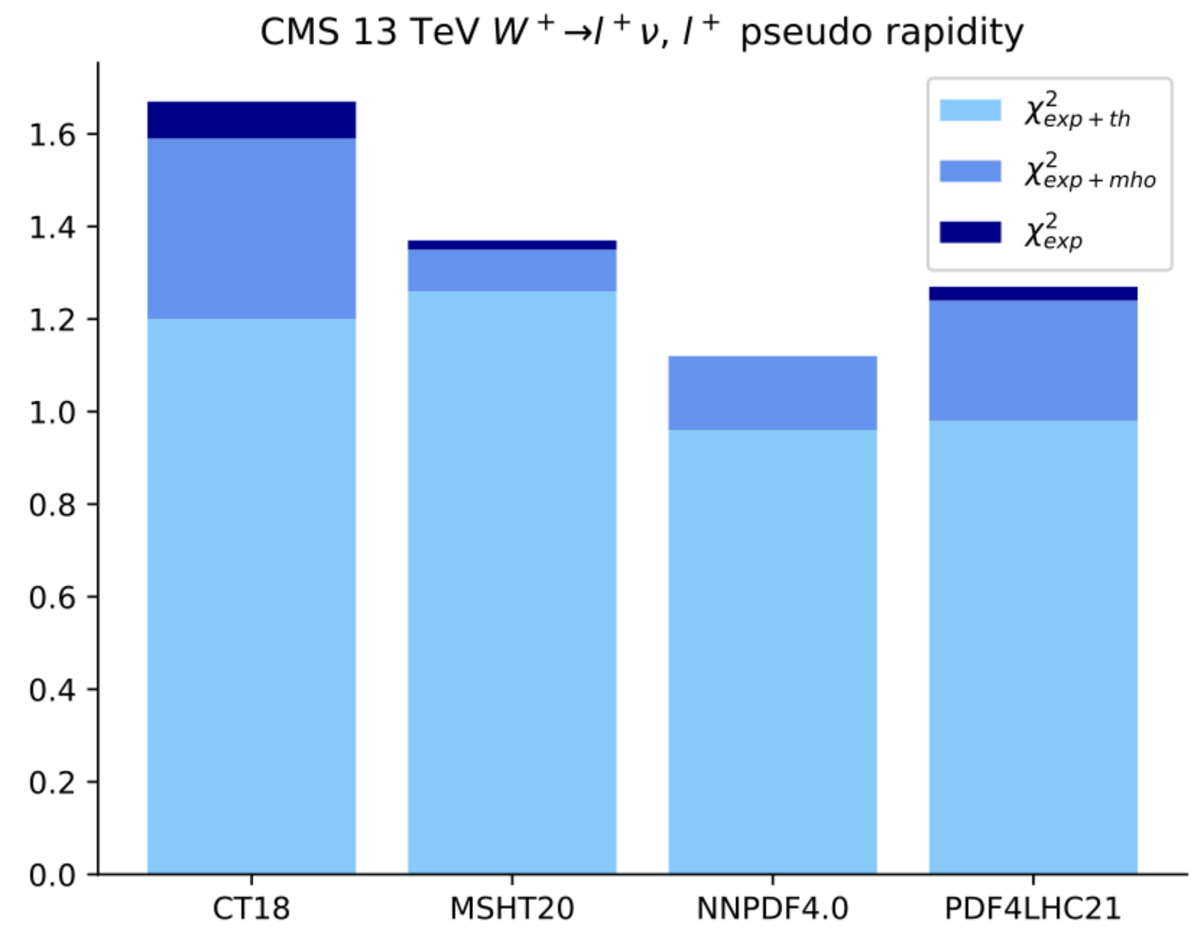
Many collaborations aim at extracting PDFs from data, performing *uniformly* well

Caution is needed though — **PDF uncertainty** is often dominant in the total error budget

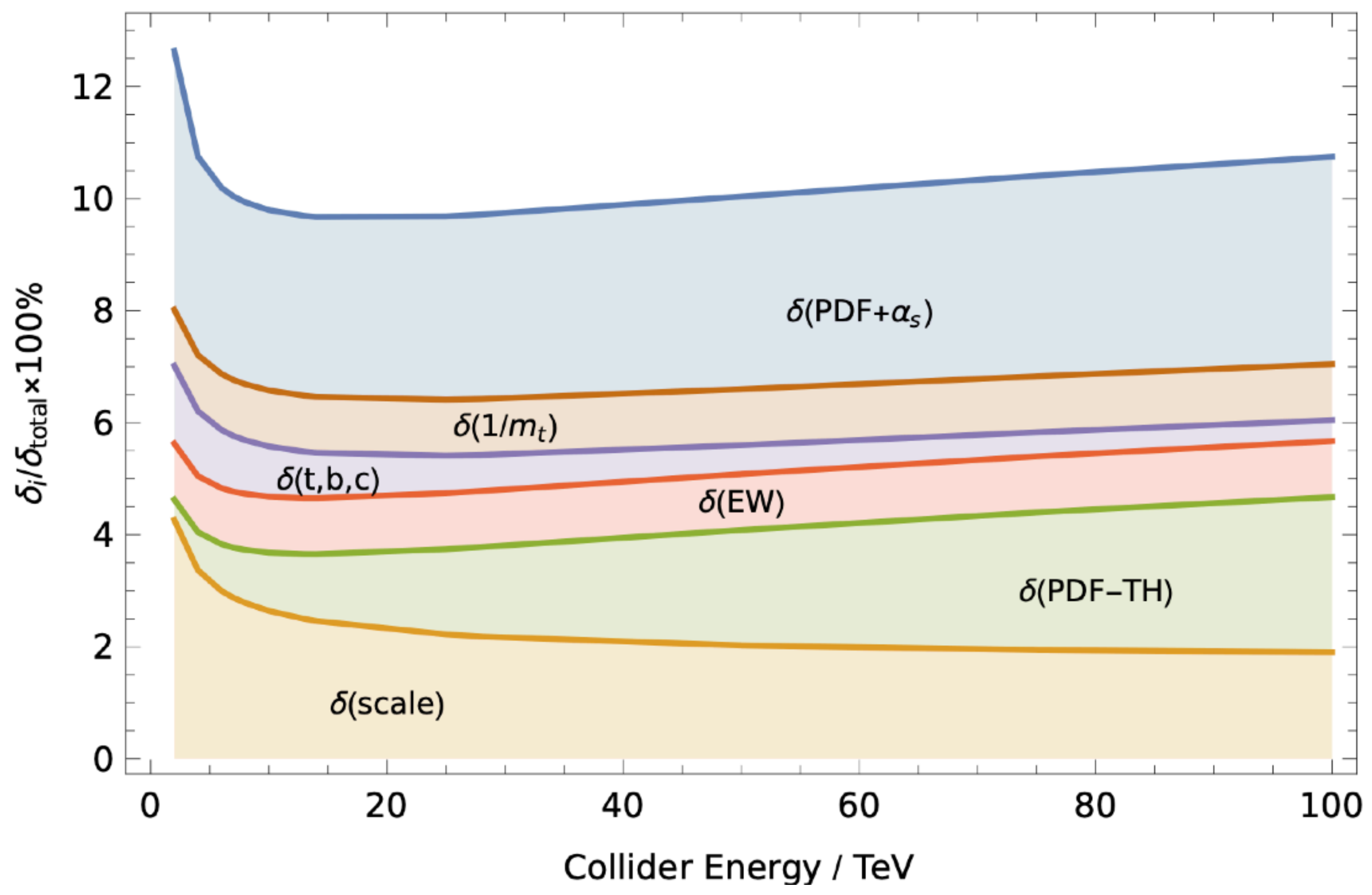
Remember: The goal is to provide reliable uncertainties!

Differences between various methodologies — can we understand their origins?

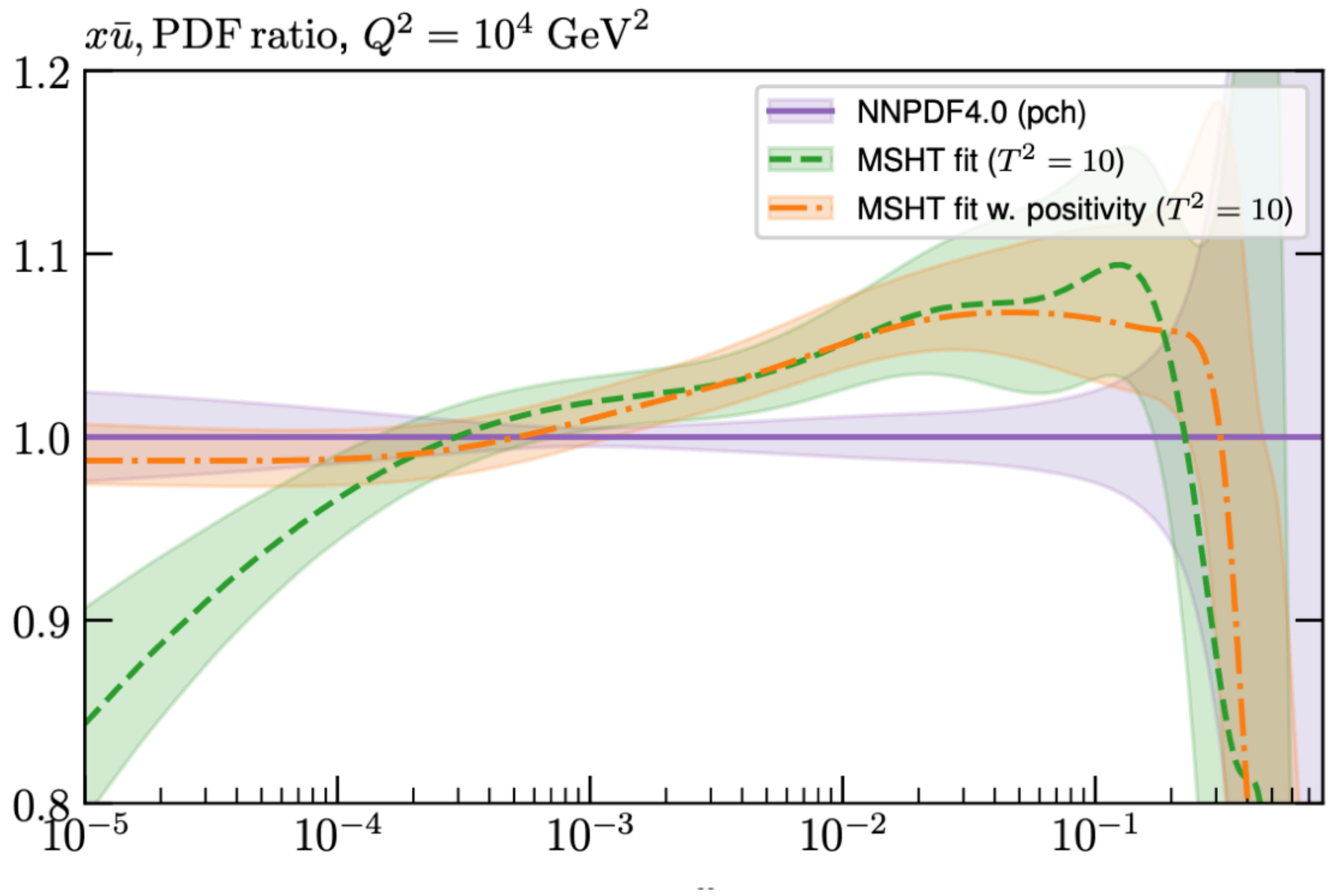
Again, we don't want to provide the best PDF set ever existed, but rather spell out our **assumptions** and understand their **implications**



Chiefa et al. JHEP 07 (2025) 067



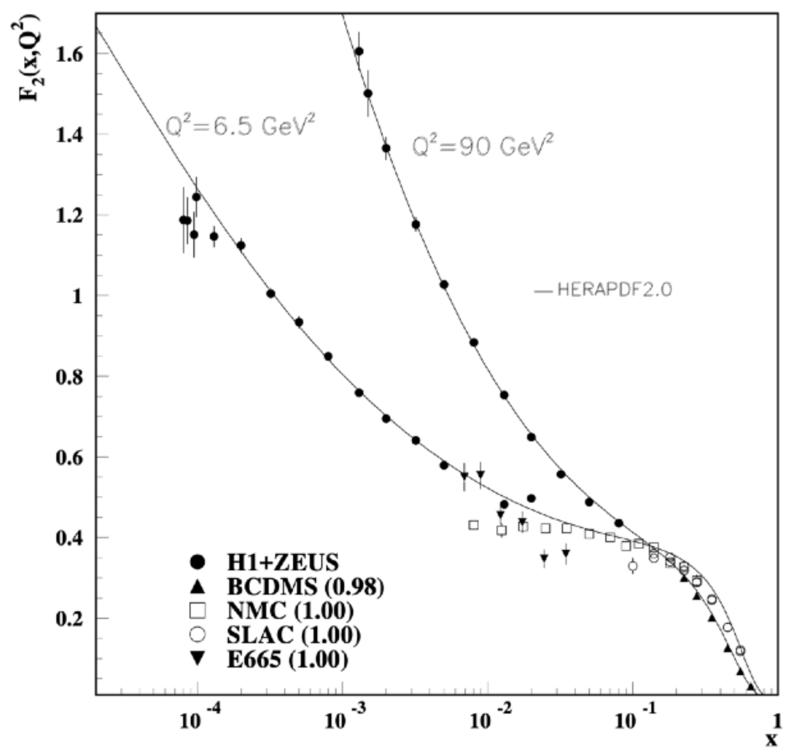
Dulat et al. CPC 233 (2018) 243



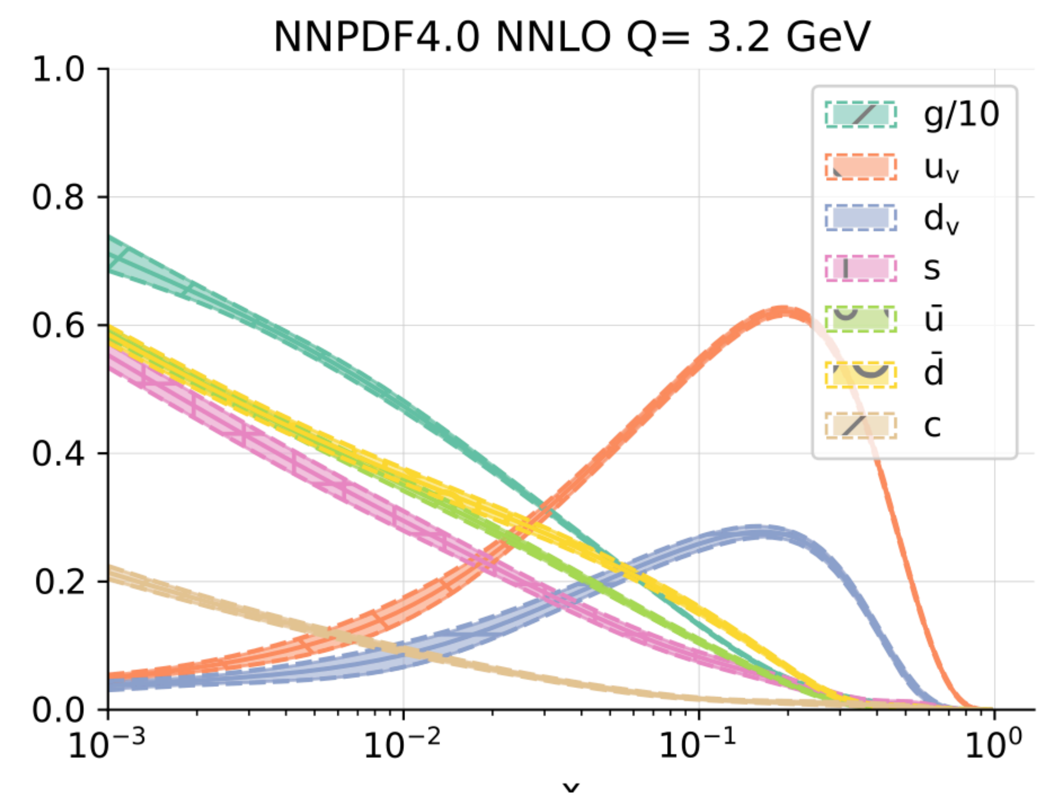
Harland-Lang et al. EPJC 85 (2025) 316

The inverse problem of PDF determination

Consider the simple case of DIS data with one PDF flavour (this work)



$$y_I = \int dx \; C(x)_I f(x)$$



This is a classic example of an **ill-defined inverse problem** to find f

Uncertainty quantification on the PDF requires to estimate $P(f \mid D)$

Determine $P(f \mid D)$ in the space of continuous functions $f : [0,1] \to \mathbb{R}$ from a finite set of data D

Parametric regression: NNPDF

The problem is discretised on a grid in $x \in \mathbb{R}^{N_{\rm grid}}$

$$y_I = \int dx \; C_I(x)\, f(x) \;\Rightarrow\; \boldsymbol{y} = \sum_{\alpha}^{N_{\rm grid}} (FK)\, \boldsymbol{f}, \qquad \boldsymbol{y} \in \mathbb{R}^{N_{\rm dat}},\; \boldsymbol{f} \in \mathbb{R}^{N_{\rm grid}},\; (FK) \in \mathbb{R}^{N_{\rm dat} \times N_{\rm grid}}$$

Neural Network parameterisation $f(x, \boldsymbol{\theta})$ with prior distribution for the parameters $P(\boldsymbol{\theta})$

$$P(\boldsymbol{\theta}\,|\,D) \longrightarrow P(f\,|\,D)$$

Probability distributions represented in terms of Monte Carlo *replicas* $\{f^{(k)}\}$

$$E\left[O\right] = \int P(f\,|\,D) O[f] \approx \frac{1}{N_{\rm rep}} \sum_{k=1}^{N_{\rm rep}} \mathcal{O}\left[\, f^{(k)} \right] \qquad \mathrm{Var}\left[O\right] \approx \frac{1}{N_{\rm rep}} \sum_{k} \Big(O\left[f^{(k)}\right] - E\left[O\right] \Big)^2$$

Train each replica using MAP with Monte Carlo sampling of experimental uncertainties

$$\{f^{(k)}\,|\,D^{(k)}\} \longrightarrow P(f\,|\,D)$$

Shedding light on the dark corners: the training process

The NNPDF methodology has been rigorously **validated** through closure tests, future and generalisation tests

HOWEVER

Lacking a one-to-one dictionary map between our methodology and others

Some aspects of our methodology remain not yet understood…

<u>In this work</u>

Opening the black box: the learning process

Taking first steps towards explainable training dynamics

Can we decode how and what the NN learns during training?

The training process

Training as Gradient Flow

We consider a **quadratic loss** function (χ^2) and the continuous version of Gradient Descent

$$\frac{d}{dt}\boldsymbol{\theta}_t = -\nabla_{\boldsymbol{\theta}}\mathscr{L}_t \qquad\qquad \mathscr{L}_t = \frac{1}{2}\left(\boldsymbol{Y} - (FK)\,\boldsymbol{f}_t\right)^T C_Y^{-1}\left(\boldsymbol{Y} - (FK)\,\boldsymbol{f}_t\right)$$

Express training from **parameter-space** to **functional-space**

$$\frac{d}{dt}\boldsymbol{f}_t = \left(\nabla_{\boldsymbol{\theta}}\boldsymbol{f}_t\right)\frac{d\boldsymbol{\theta}_t}{dt} = \Theta_t (FK)^T C_Y^{-1}\left(\boldsymbol{Y} - (FK)\,\boldsymbol{f}_t\right) = -\Theta_t\left(M\,\boldsymbol{f}_t + \boldsymbol{b}\right)$$

$$M = (FK)^T C_Y^{-1}(FK)\,, \quad \boldsymbol{b} = (FK)^T C_Y^{-1}\boldsymbol{y}$$

where we have defined the **Neural Tangent Kernel** (NTK)

Jacot et al. [arXiv:1806.07572]

$$\Theta_t = (\nabla_{\boldsymbol{\theta}}\boldsymbol{f}_t)(\nabla_{\boldsymbol{\theta}}\boldsymbol{f}_t)^T \in \mathbb{R}^{N_{\mathrm{grid}}\times N_{\mathrm{grid}}}$$

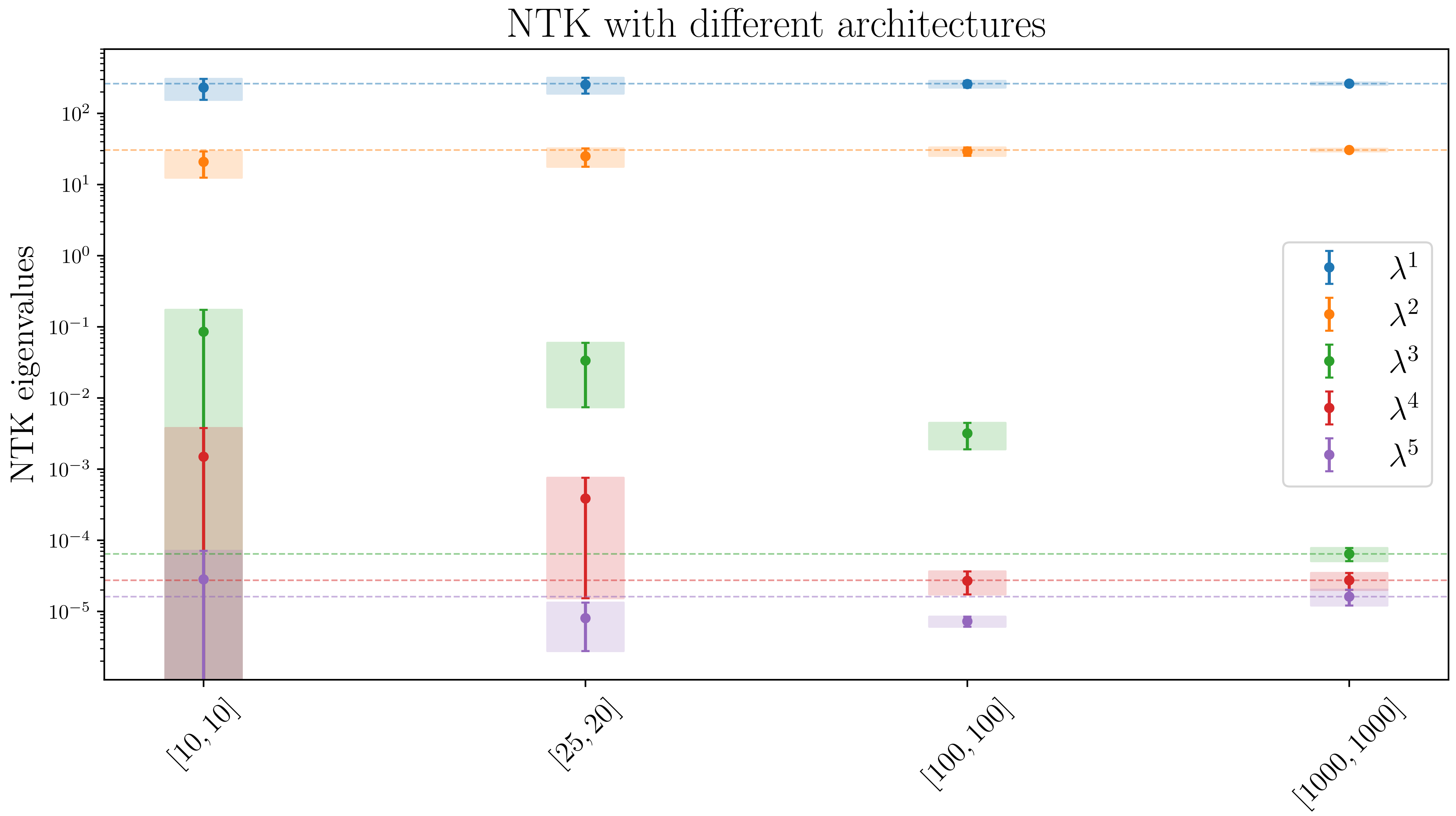
What does the NTK tell us?

The NTK spectrum **at initialisation** is heavily **hierarchical** — few eigenvalues are non-zero

<u>Will this picture change during training?</u>

Initialise $\{f^{(k)}\}$ and generate synthetic data using a known underlying law

Train each replica and monitor the evolution of the NTK



Model: MPL [1,25, 20,1] with tanh

Level 0: $\boldsymbol{y}_{L0}^{(k)} = (FK)\boldsymbol{f}^{(\mathrm{in})}$

Level 1: $\boldsymbol{y}_{L1}^{(k)} = \boldsymbol{y}_{L0}^{(k)} + \eta, \quad \eta \sim \mathcal{N}$

Level 2: $\boldsymbol{y}_{L2}^{(k)} = \boldsymbol{y}_{L1}^{(k)} + \xi^{(k)}, \quad \xi^{(k)} \sim \mathcal{N}$

NTK during training

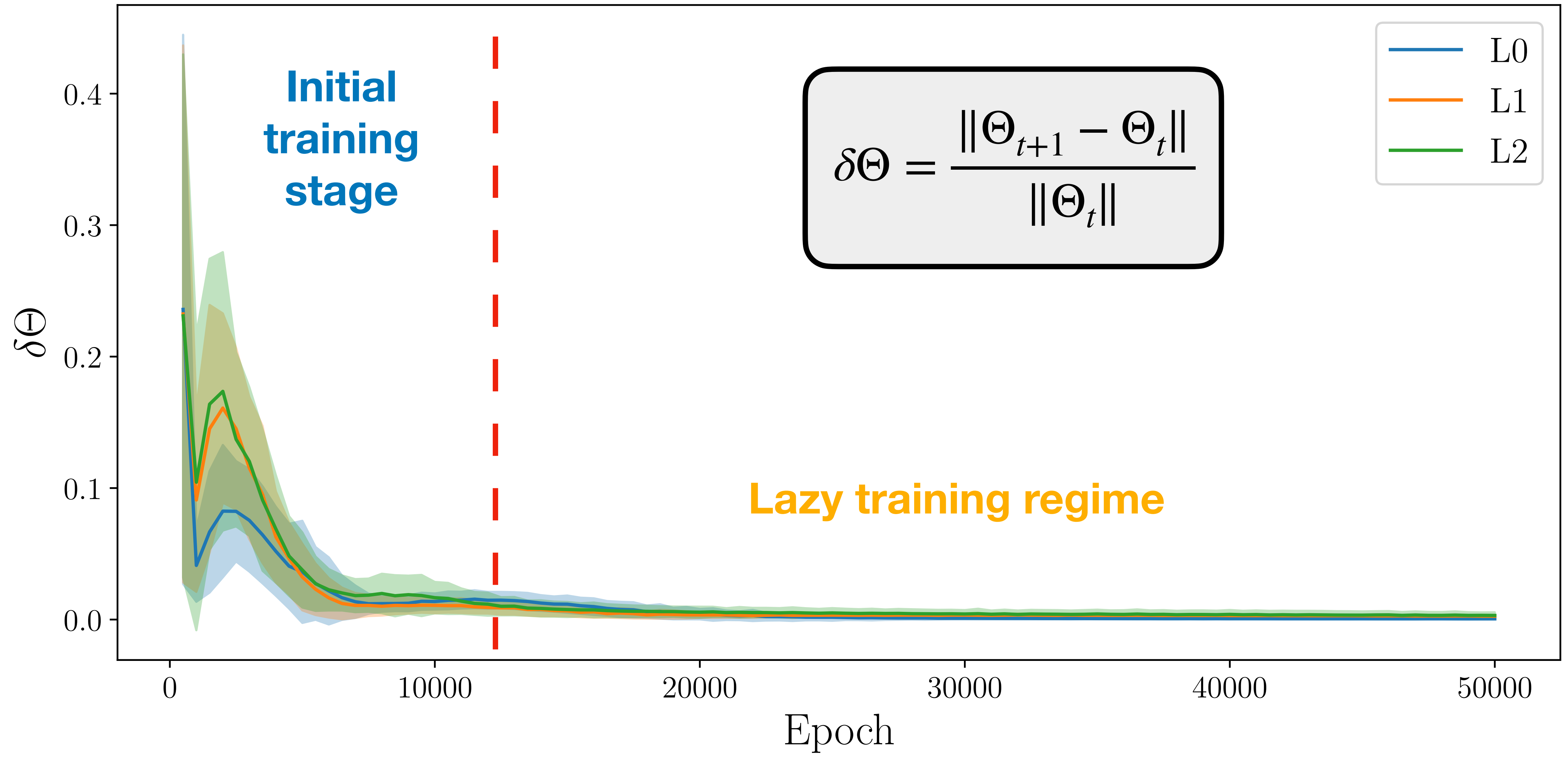
The NTK changes during training — we are not in the infinite-width regime

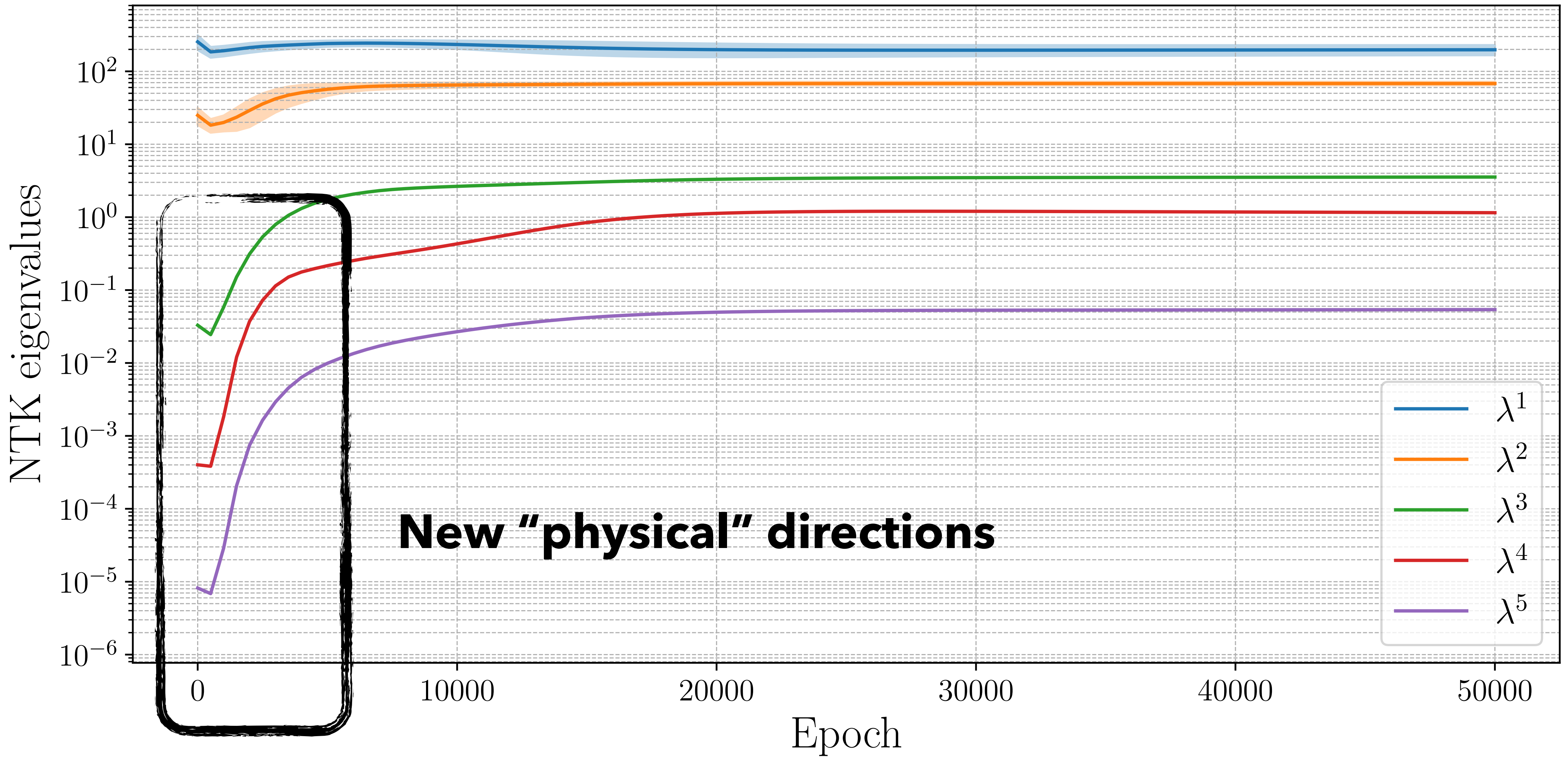
Two distinct training phases — **rich** and **lazy** regime

The **hierarchy** of the spectrum is **preserved** even during training…

…although new **directions** are discovered!

Arch. [1,25,20,1]

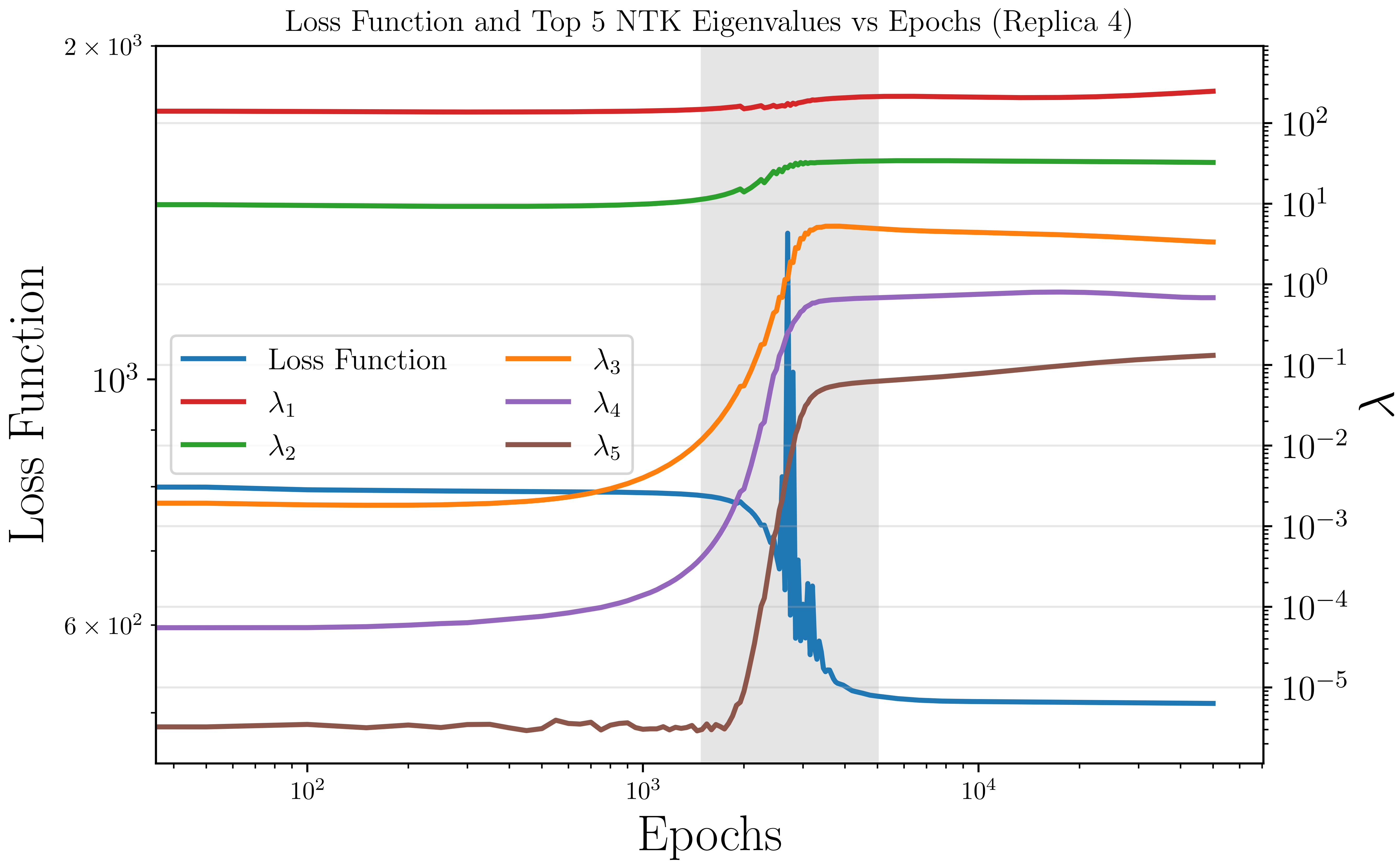




Loss Function and NTK

The NN is adapting its internal representation with the NTK eigenvalues

This allows the process to minimise the loss function, at the cost of local instability

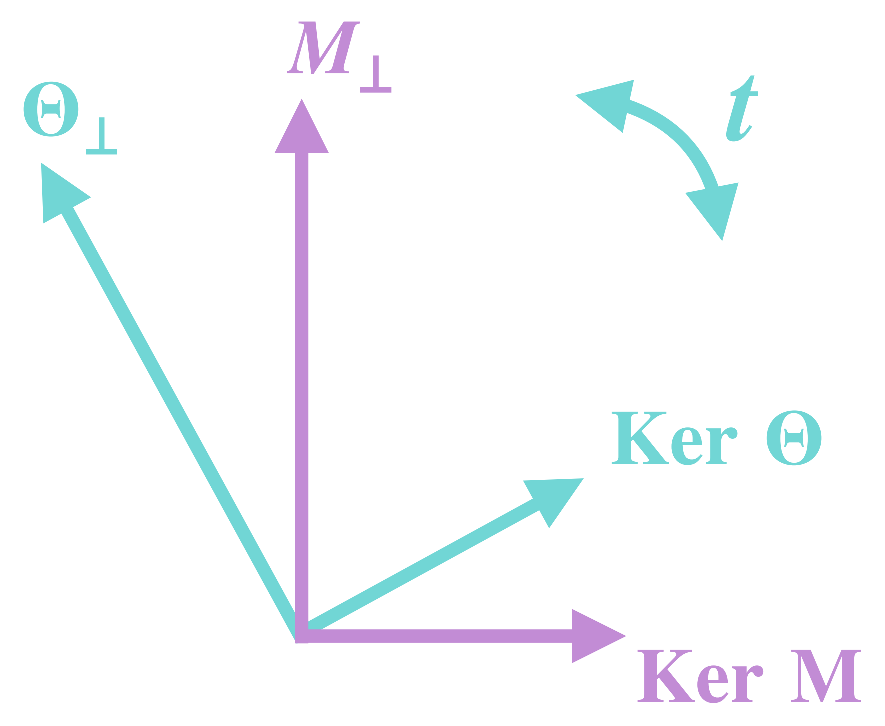


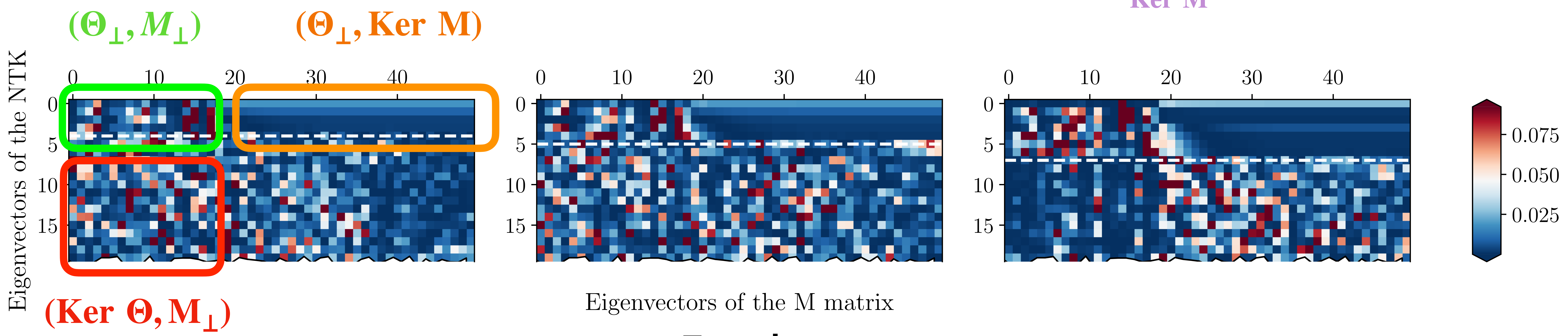
Alignment of the NTK with FK tables

Interplay between the spectrum of the NTK and the FK tables (time-independent)

Only components in both orthogonal spaces are "learned" by the network

$$\frac{d}{dt}f_t = -\Theta_t\left(M f_t + b\right)$$





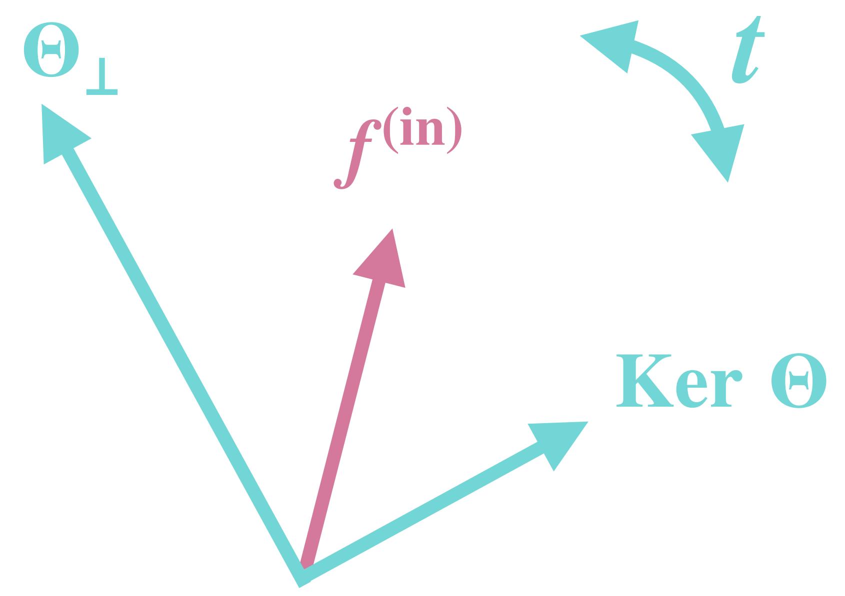
Epochs

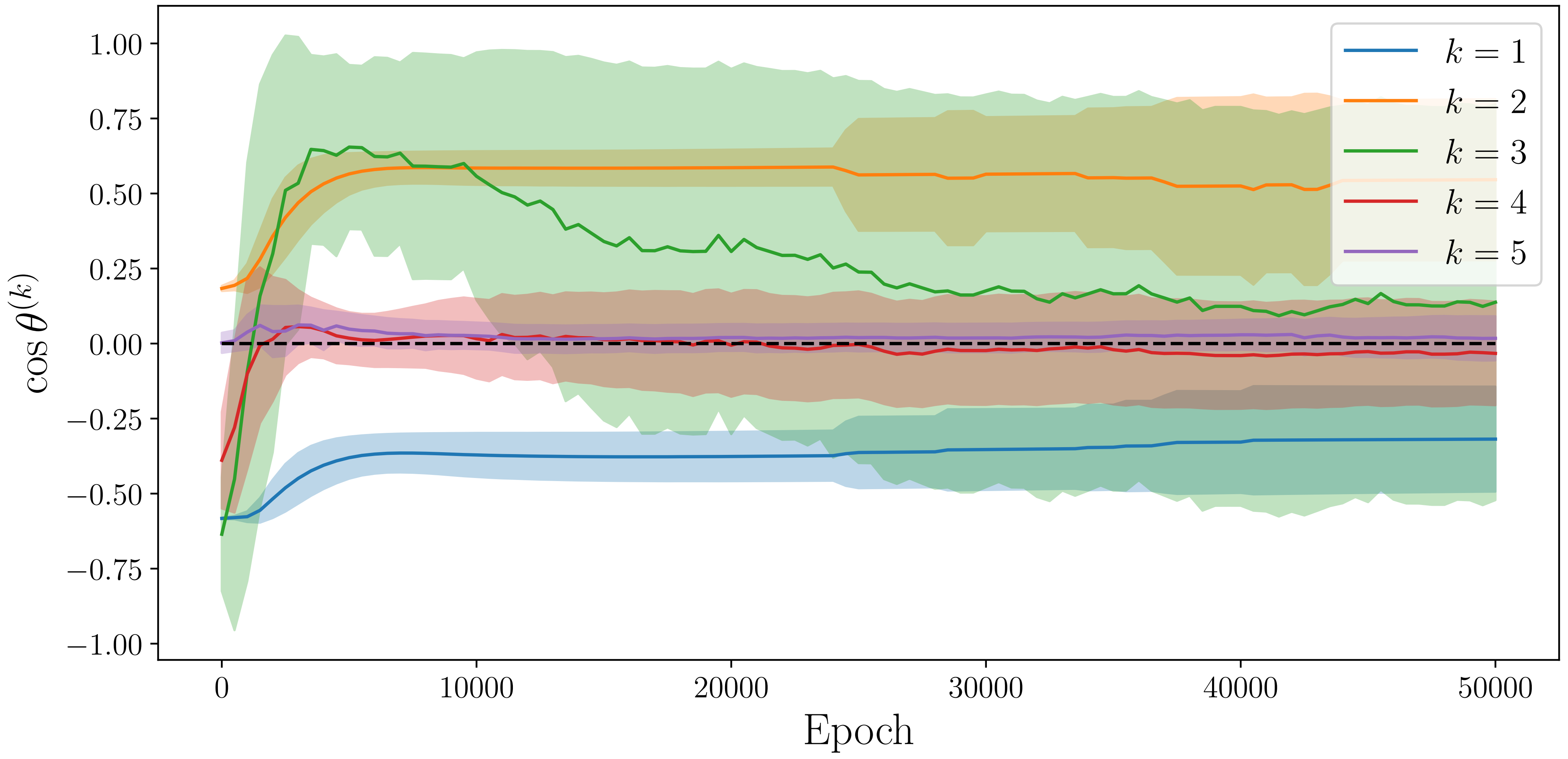
Alignment of the NTK with the input function

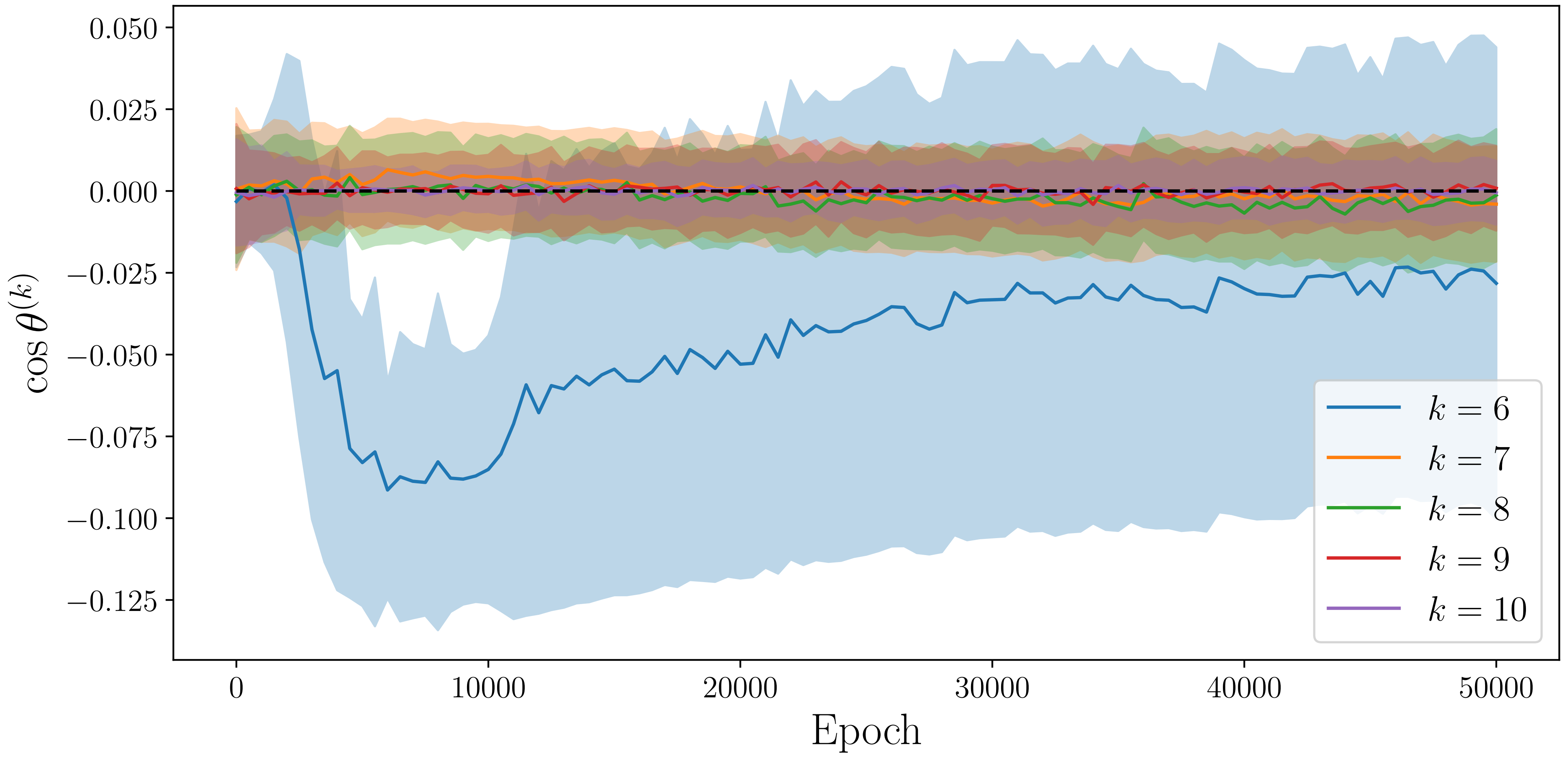
A similar study can be done with the input function

How do the **modes** of the NTK align with the input function?

$$\frac{d}{dt}\boldsymbol{f}_t = -\boxed{\Theta_t}\left(M\boldsymbol{f}_t + \boldsymbol{b}\right)$$





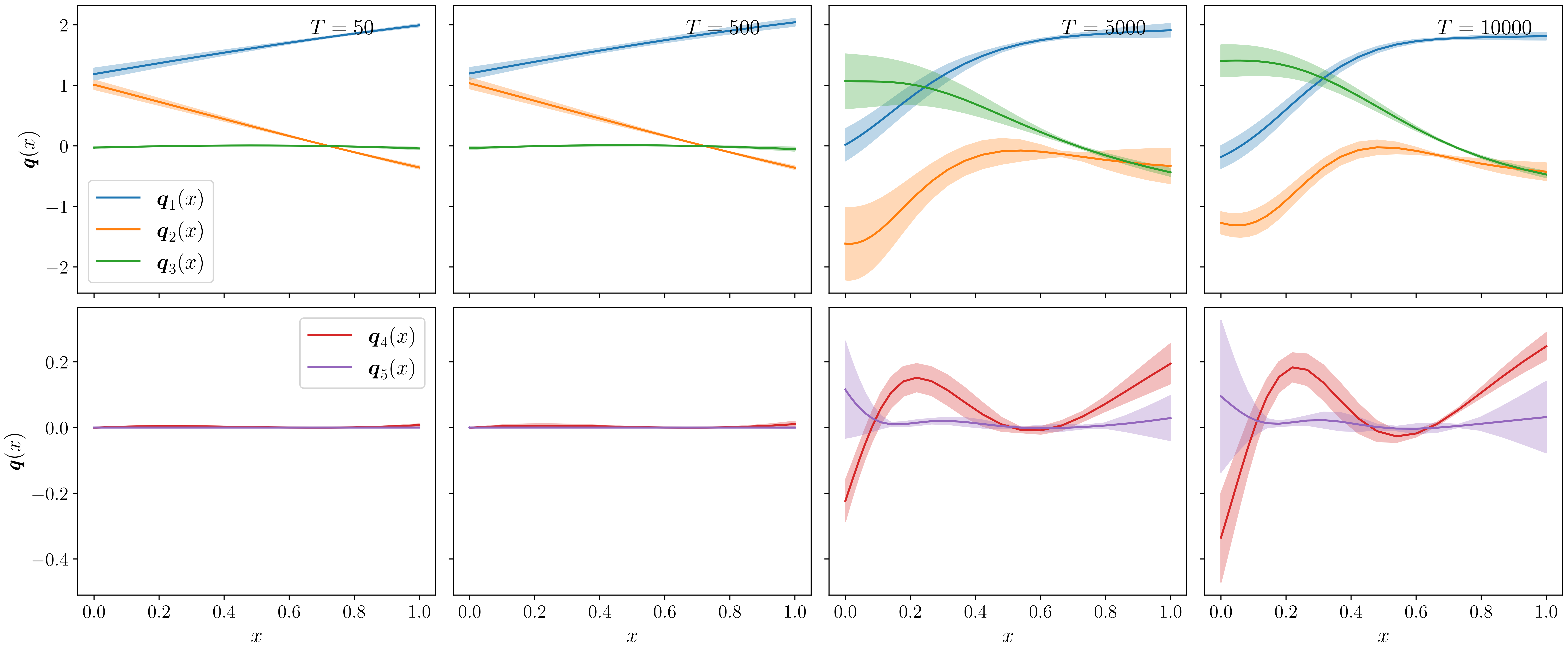


Increasing complexity of the representation

The output of the NN is a superposition of the modes dictated by the NTK

As training proceeds, the modes gain in complexity

When the NTK enters the lazy regime, the modes stop changing…



The lazy-training regime

Lazy gradient flow

Can we solve the equation in the lazy regime where $\Theta_t \approx \Theta_{T_{\mathrm{ref}}}$ for $t > T_{\mathrm{ref}}$?

In this regime, the NTK provides a natural basis for projecting the equation

$$\frac{d}{dt}\boldsymbol{f_t} = -\,\Theta_t\left(M\,\boldsymbol{f}_t + \boldsymbol{b}\right) \qquad \Theta \sim \begin{pmatrix} \Lambda_\perp & 0 \\ 0 & 0 \end{pmatrix} \quad \begin{array}{l} \Theta_\perp \\ \mathrm{Ker}\Theta \end{array}$$

We immediately see that the components in $\mathrm{Ker}\Theta$ do **not** evolve

$$\frac{d}{dt}\boldsymbol{f}_t^{\mathrm{Ker}\Theta} = 0 \quad \Rightarrow \quad \boldsymbol{f}_t^{\mathrm{Ker}\Theta} = \boldsymbol{f}_{\mathrm{init}}^{\mathrm{Ker}\Theta}$$

There is an irreducible **noise** dictated by the functional initial condition

To what extent does this contribution affect the final result?

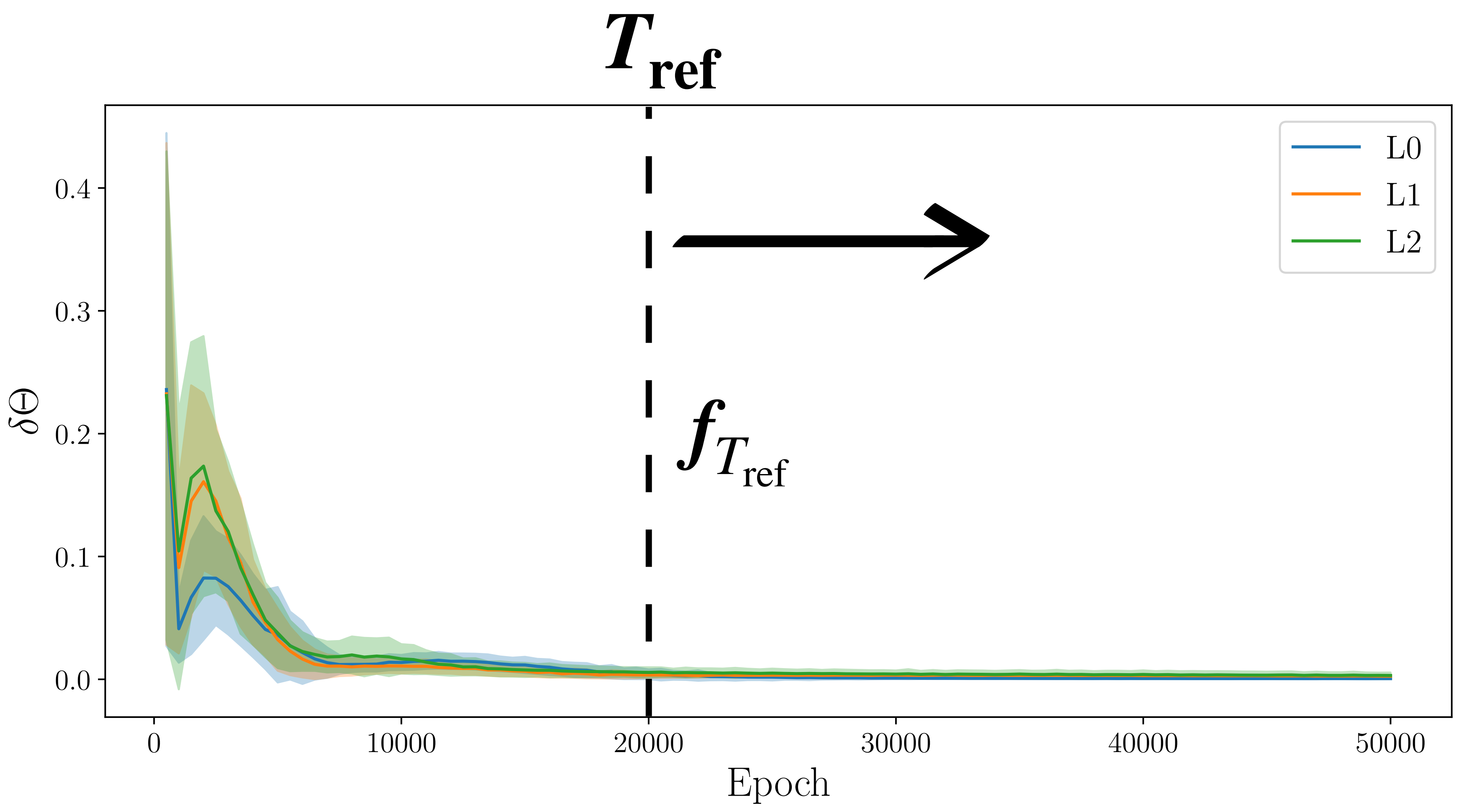
Solution of the gradient flow

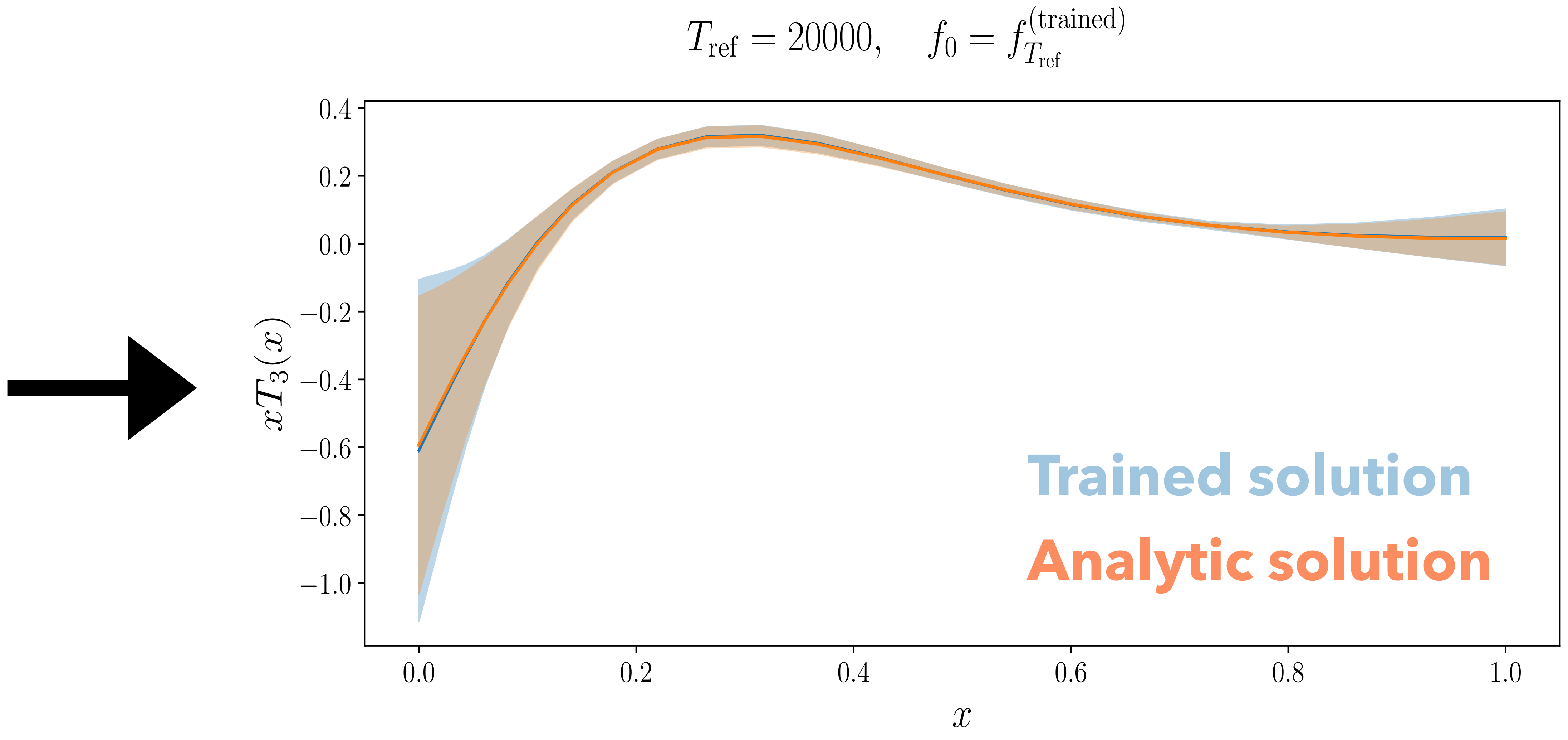
Solving the flow in the orthogonal space $\Theta_\perp$ and collecting all terms yields

$$\boldsymbol{f}_t = U(t)\, \boldsymbol{f}_{\text{init}} + V(t)\, \boldsymbol{y}$$

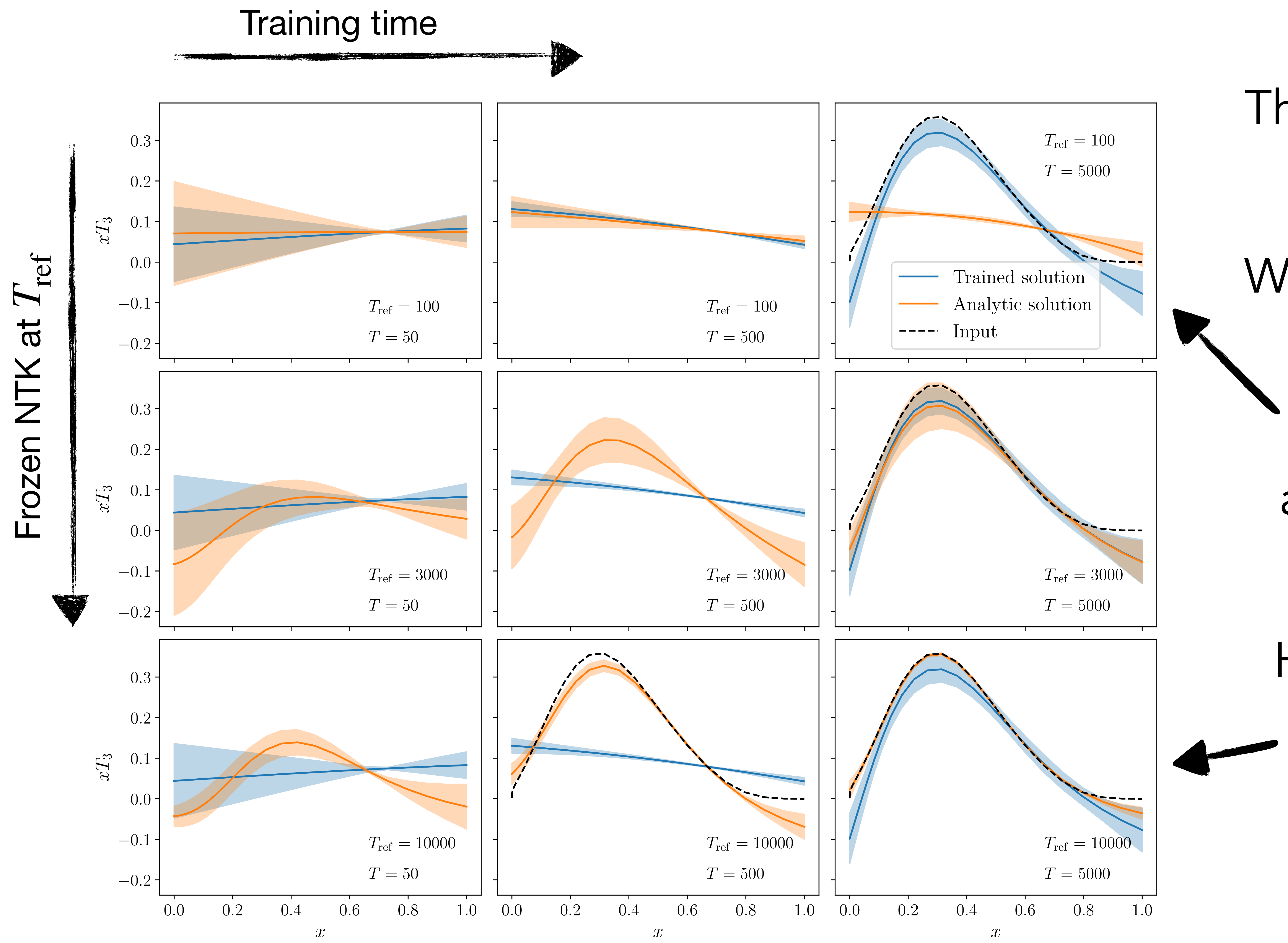
The solution factorises into two terms representing **data** and **model** dependence

Setting $f_{\text{init}} = f_{T_{\text{ref}}}$ we can reconstruct the numerical integration at $t > T_{\text{ref}}$





The onset of the lazy regime



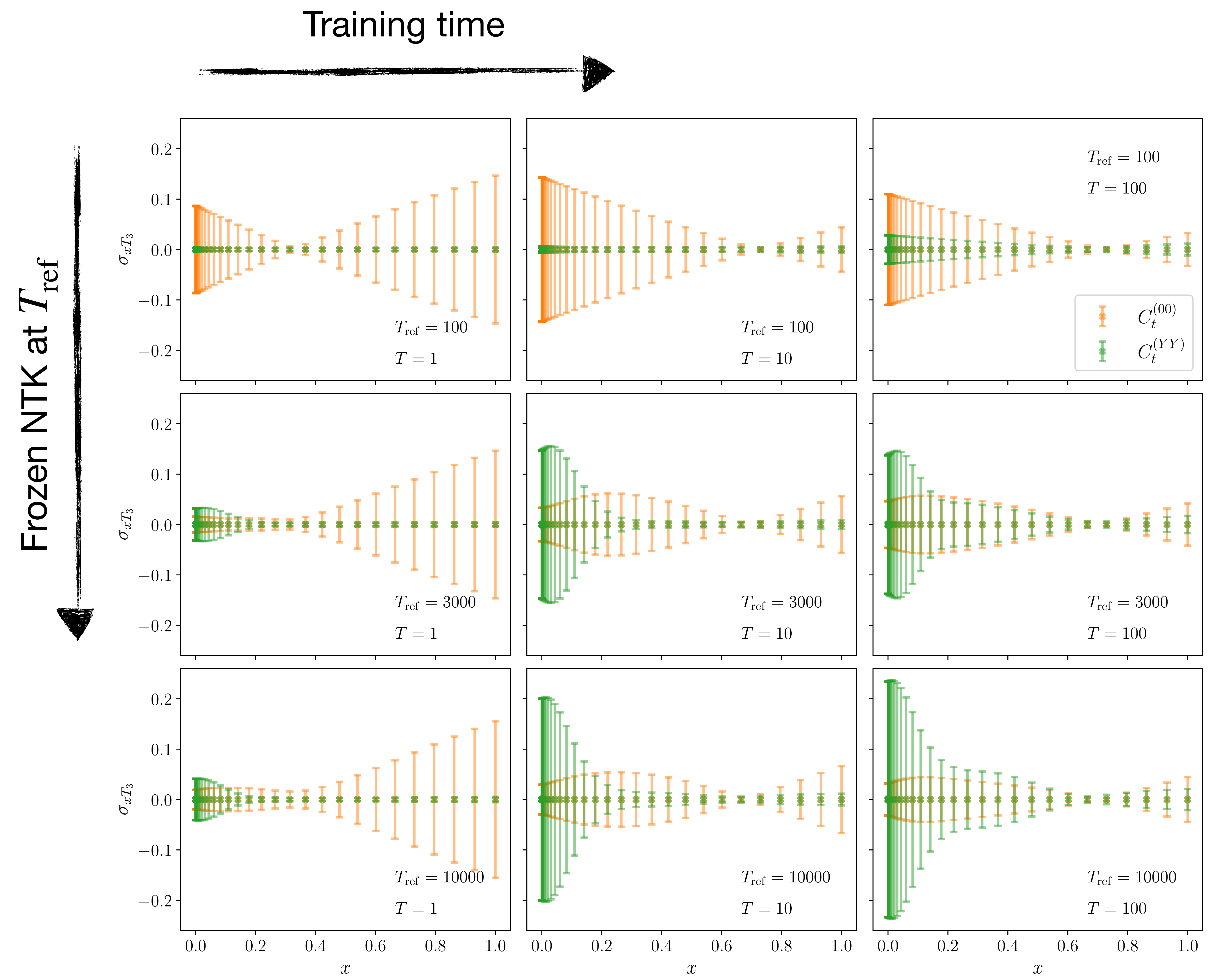
The analytic solution can be used with a different initial condition $f_{\text{init}} = f_0$

We can study the onset of lazy regime by probing different frozen NTKs

If the NTK is too premature, the analytic solution cannot provide an accurate answer

However, a properly **informed** NTK speeds up the analytic solution compared to the numerical one

Interplay between data and model error



We can compute the covariance of the analytic solution

$$\mathrm{Cov}\left[f_t, f_t^T\right] = \boxed{C_t^{(00)}} + C^{(0Y)} + \boxed{C_t^{(YY)}}$$

The covariance breaks down into **model** and **data** error plus a cross-correlation term

The interplay between these two depends on the NTK and on the training time

Concluding – what did we learn?

<u>Laying the groundwork to understand methodological differences</u>

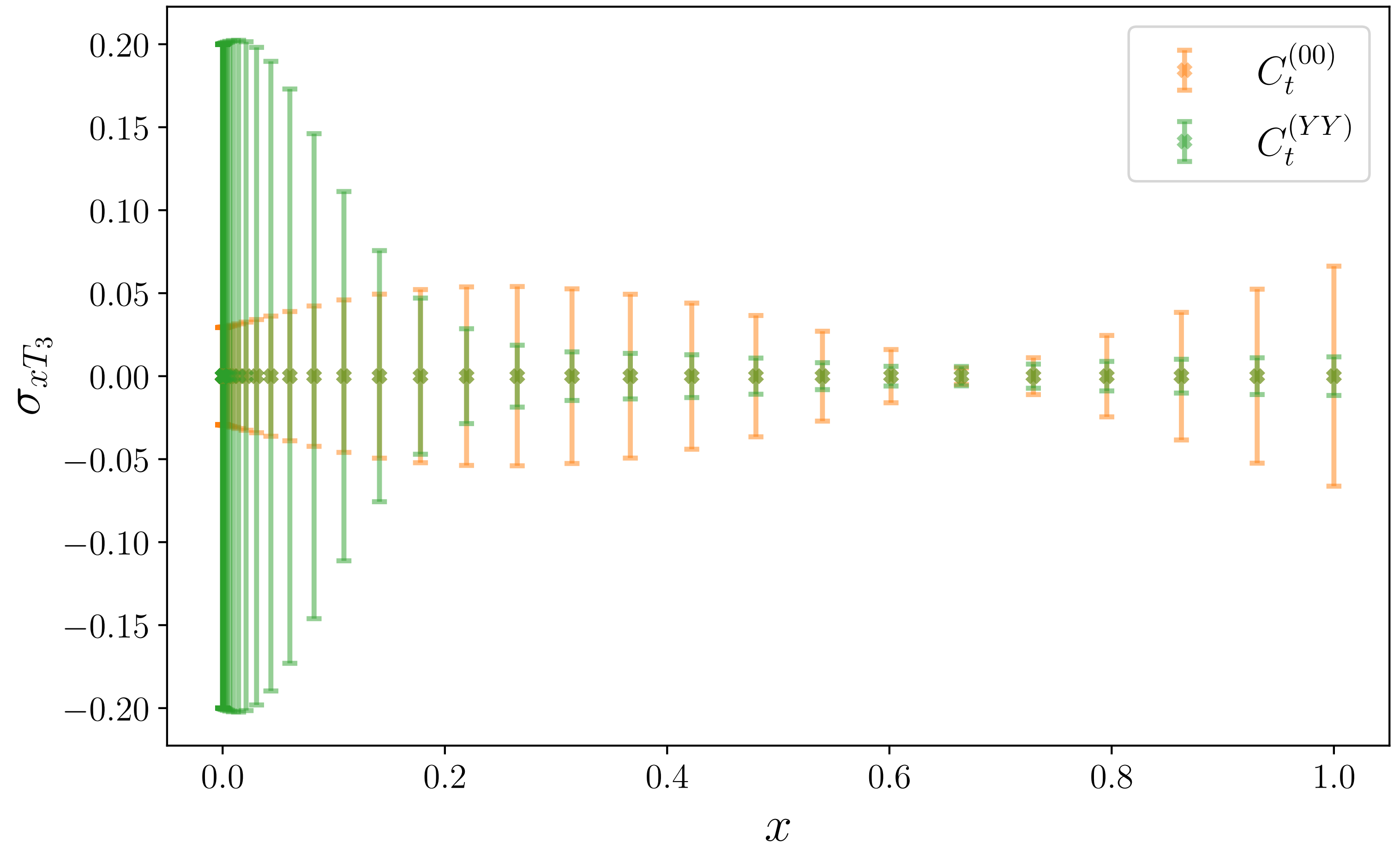
The NTK is a valuable tool to unravel the training process

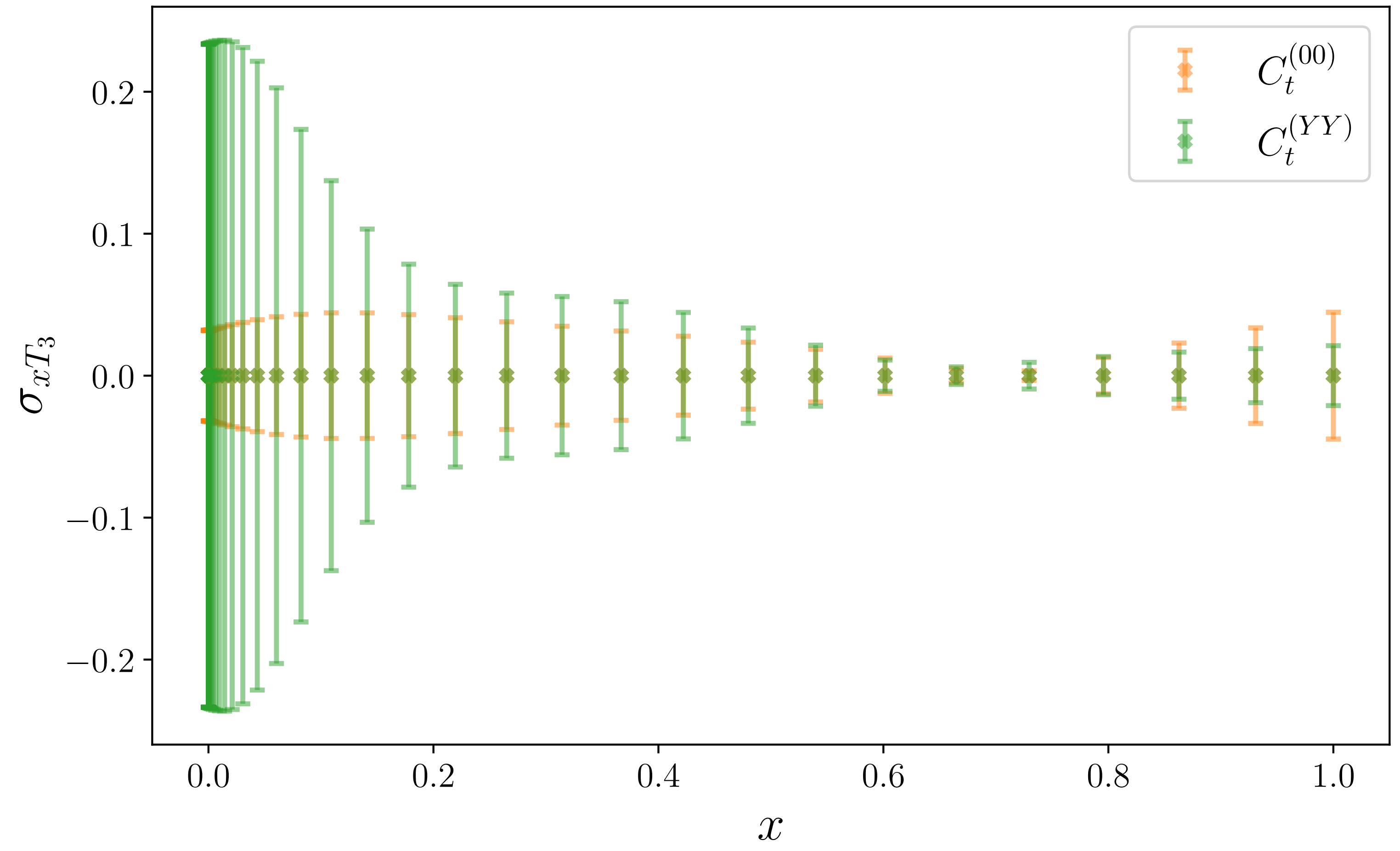
Starting from a suitable training time, we have analytic control of the training process

The road ahead: from proof-of-concept to full complexity

These results hold in a simplified framework – extending to NNPDF remains open

What insights can we learn from applying these tools to other methodologies?





Concluding – what did we learn?

<u>Laying the groundwork to understand methodological differences</u>

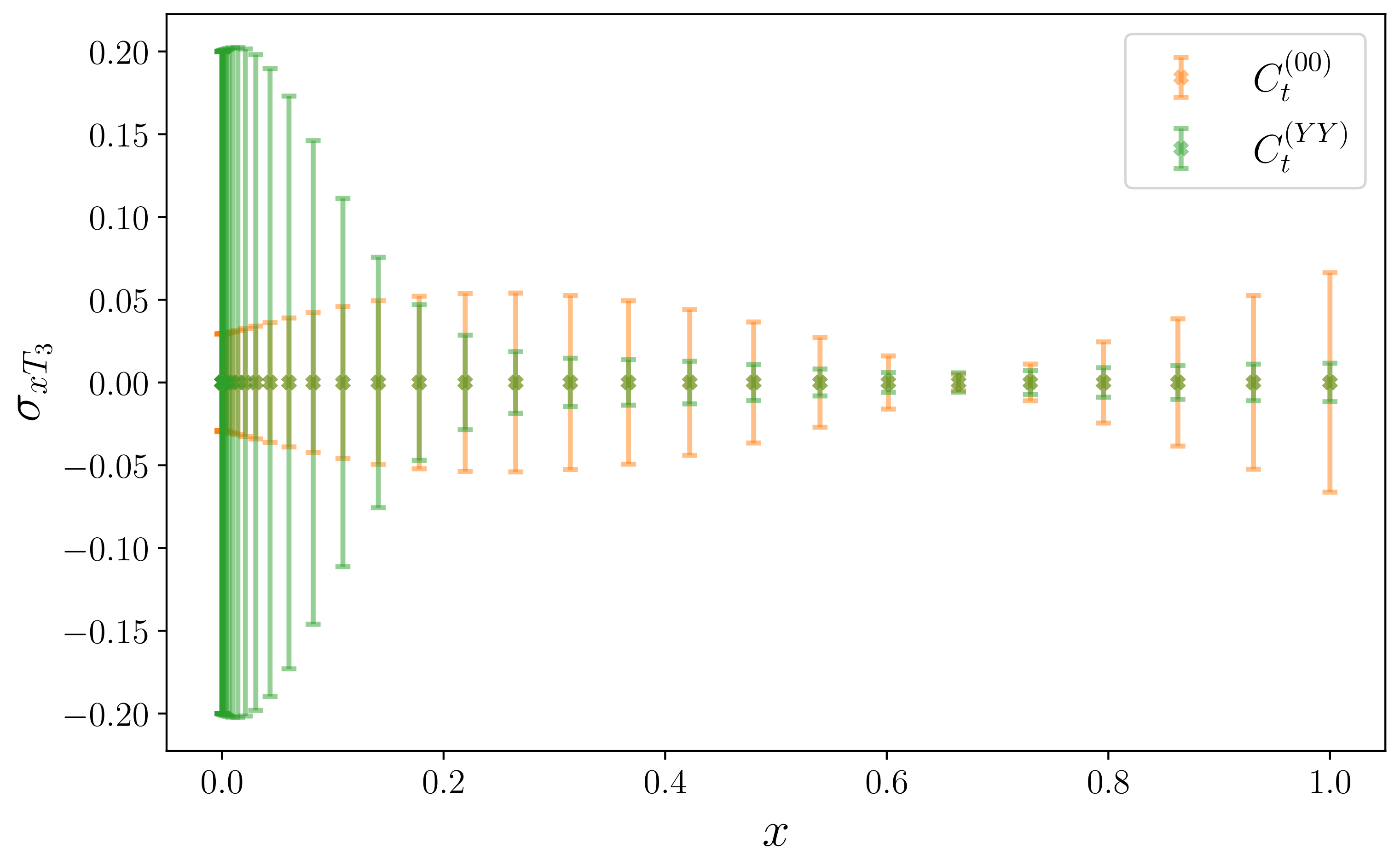
The NTK is a valuable tool to unravel the training process

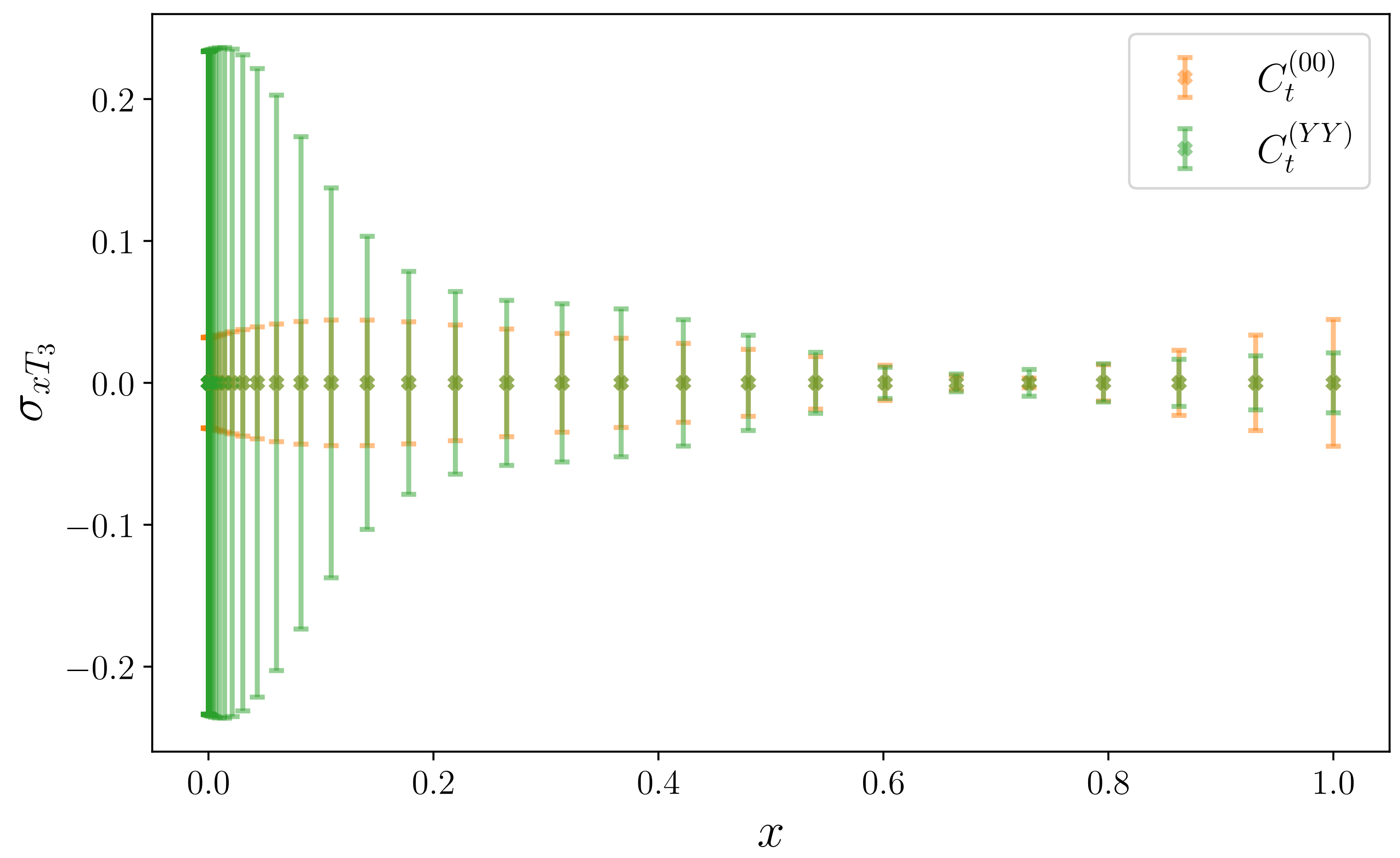
Starting from a suitable training time, we have analytic control of the training process

The road ahead: from proof-of-concept to full complexity

These results hold in a simplified framework – extending to NNPDF remains open

What insights can we learn from applying these tools to other methodologies?



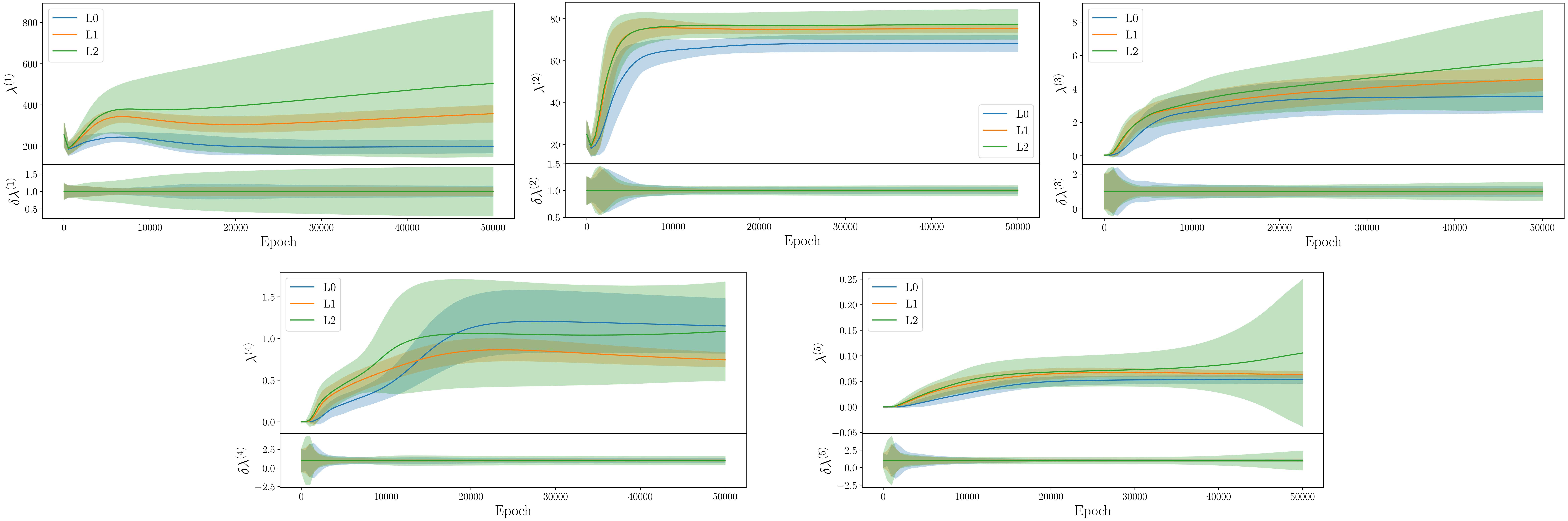


Thank you!

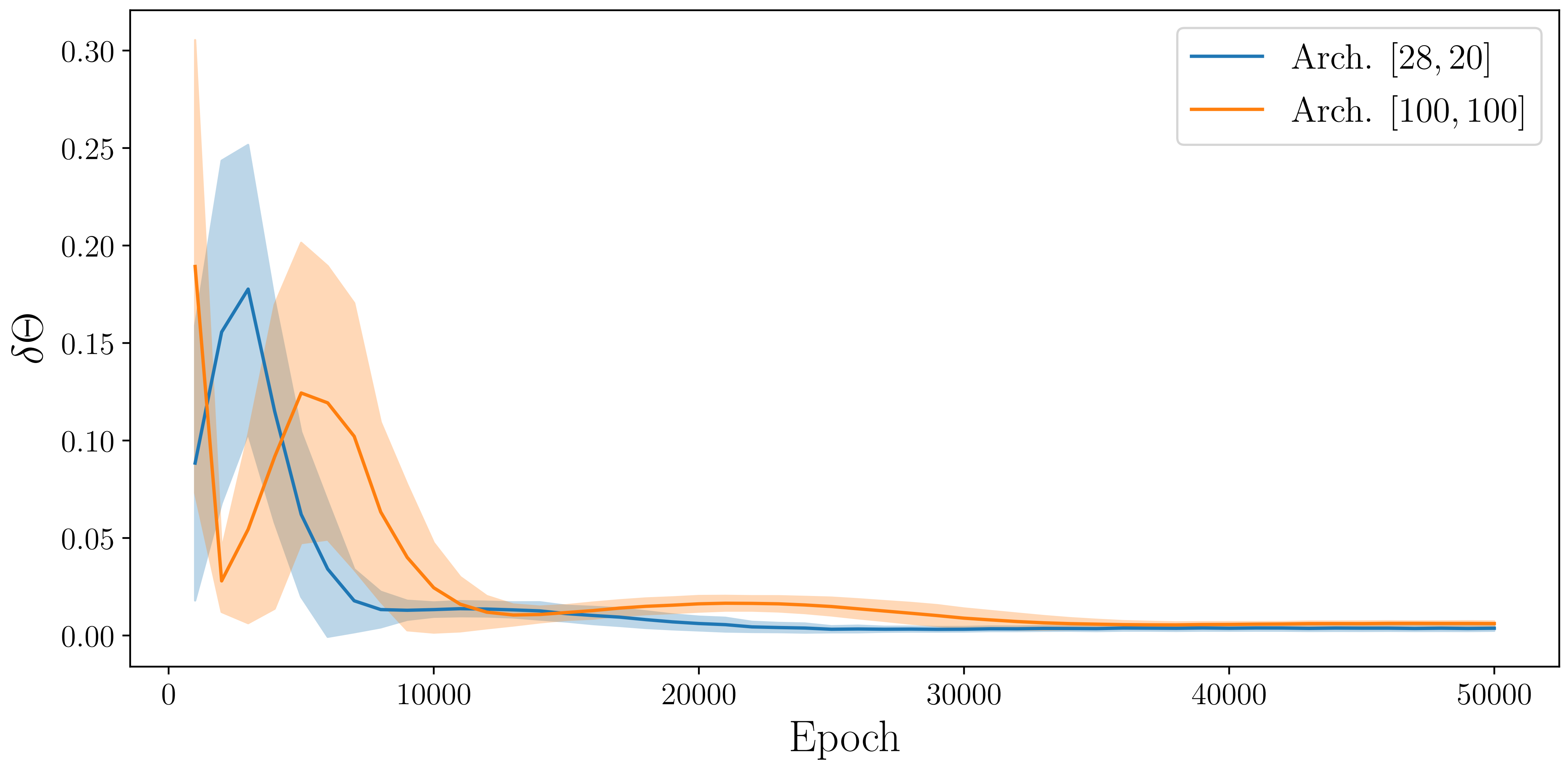
Backup slides

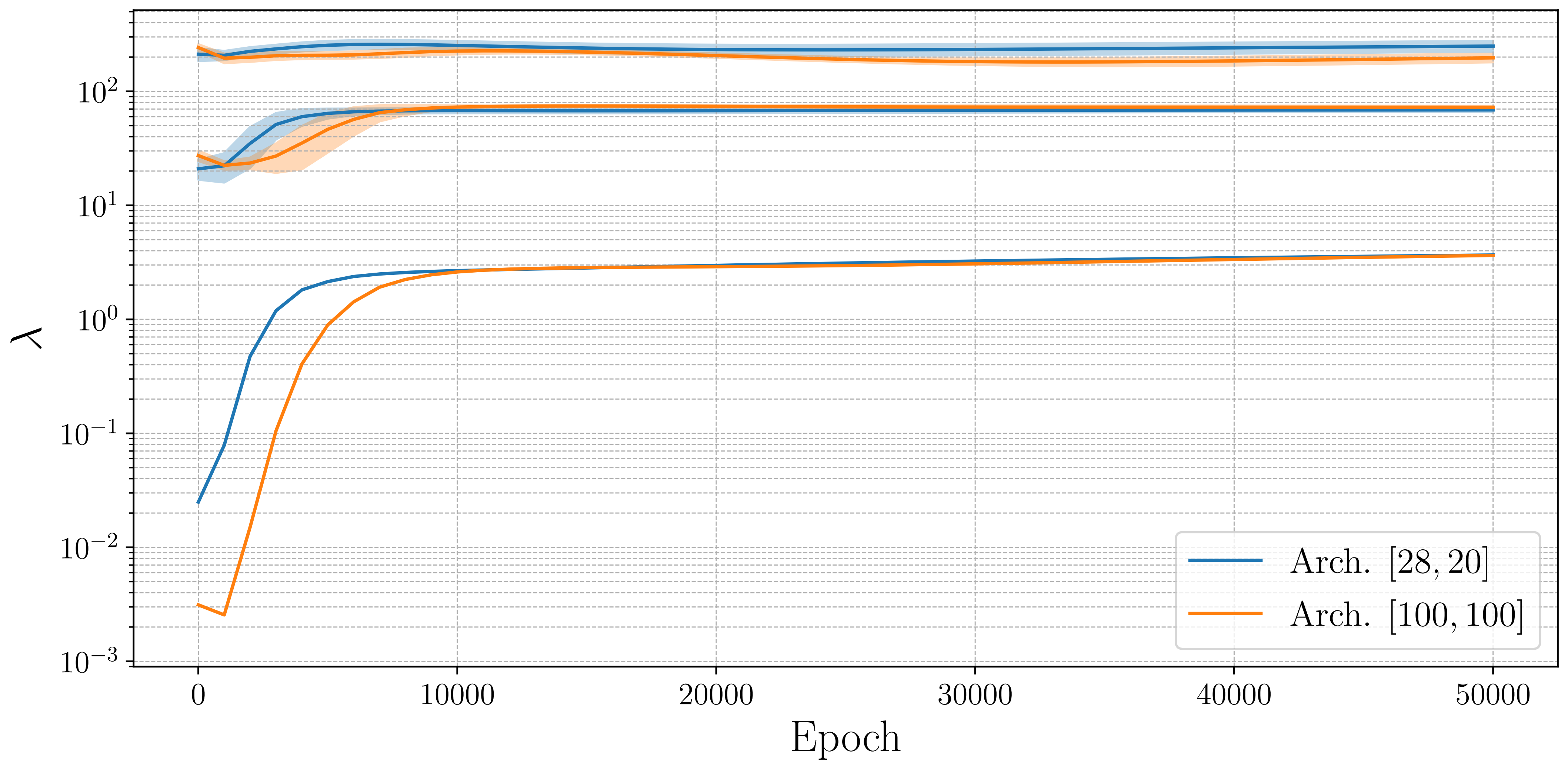
Spectrum of the NTK

The eigenvalues of the NTK evolve during training, but preserve the hierarchy



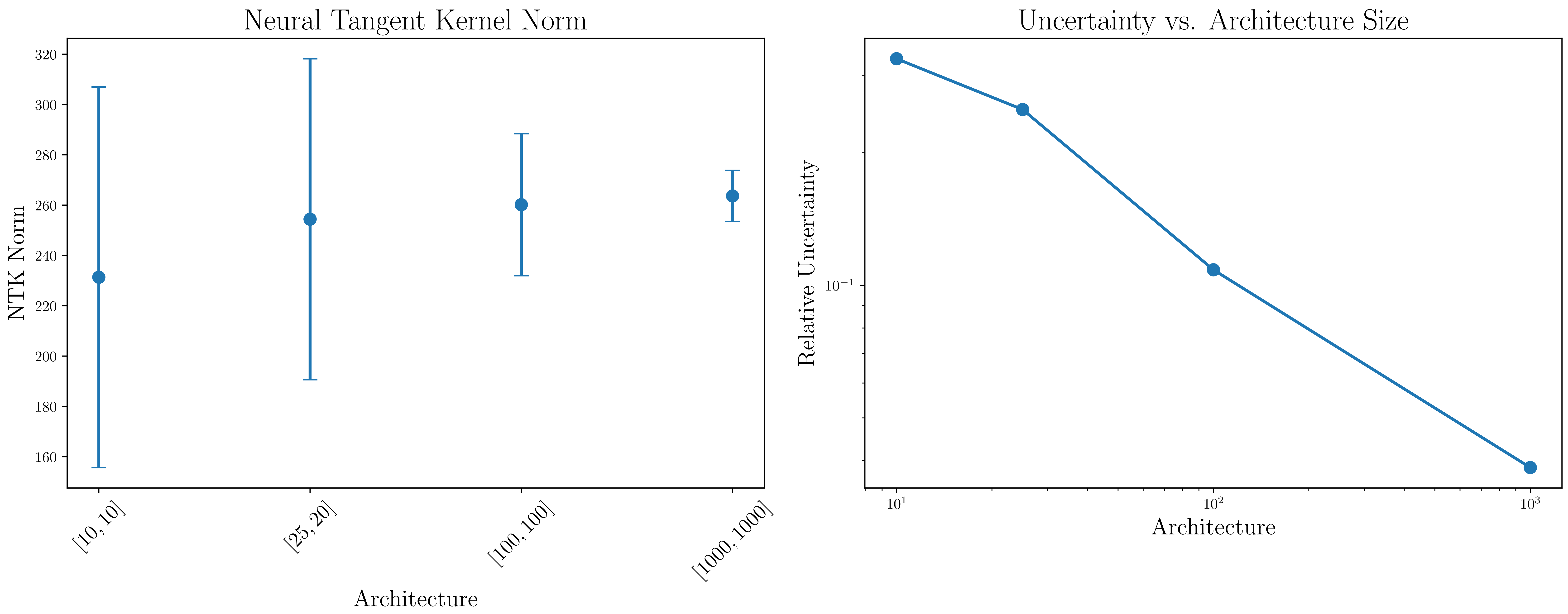
Dependence on the architecture





- Changing the size of the architecture has little impact
- The onset of lazy training is slower for larger networks
  - Larger networks cover a **wider** functional space
  - The identification of the physical feature takes more epochs

NTK at initialisation for NNs: 1/n scaling



- NTK fluctuations are proportional to the **inverse** of the **width** (see *e.g.*[arXiv:1806.07572] [arXiv:2106.10165])
- The NTK **at initialisation** remains constant over the ensemble of replicas