Notes on Neural Tangent Kernels

Luigi Del Debbio and Amedeo Chiefa

Higgs Centre for Theoretical Physics, School of Physics and Astronomy,
Peter Guthrie Tait Road, Edinburgh EH9 3 FD, United Kingdom.

May 8, 2025

Abstract

Summary of results for NTKs. These notes follow the notation introduced in Ref. [1].

1 Definitions

1.1 Neural Networks

We consider the case of neural networks (NNs) with $L+1$ layers, labelled by the index $\ell = 0, \dots, L$. The number of neurons in layer ℓ is denoted by n_ℓ. The $\ell = 0$ and $\ell = L$ layers are the input and output layers respectively, so that a NN parametrizes an element of

$$\mathcal{F} = \{ f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L} \} \, . \tag{1}$$

The space of inputs in this case is $\mathcal{X} = \mathbb{R}^{n_0}$, while the space of outputs is $\mathcal{Y} = \mathbb{R}^{n_L}$. The number of parameters used in the parametrization is the total number of weights and biases,

$$P = \sum_{\ell=0}^{L-1} (n_\ell + 1) n_{\ell+1} \, . \tag{2}$$

The weights connecting the layer ℓ to the layer $\ell+1$ are denoted $W^{(\ell+1)}$ and the biases in layer $\ell+1$ are denoted $b^{(\ell+1)}$. With this convention

$$W^{(\ell+1)} \in \mathbb{R}^{n_{\ell+1} \times n_\ell} \, , \quad b^{(\ell+1)} \in \mathbb{R}^{\ell+1} \, . \tag{3}$$

When referring to the whole set of parameters, we use the symbol

$$\theta = \left\{ W^{(\ell)}, b^{(\ell)};\ \ell = 0, \dots, L-1 \right\} \, , \tag{4}$$

while individual parameters are denoted θ_μ, with $\mu = 1, \dots, P$.

The input space is $\mathcal{X} = \mathbb{R}^{n_0}$ and we consider a probability distribution p^{in} over $\mathcal{X}$, which induces a scalar product in $\mathcal{F}$,

$$\langle f, g \rangle_{\text{in}} = \mathbb{E}_{x \sim p^{\text{in}}} \left[f(x)^T g(x) \right] \tag{5}$$

$$= \int dx \, p^{\text{in}}(x) f_i(x) g_i(x) \, . \tag{6}$$

For a discrete set of datapoints $\mathcal{D}$,

$$p^{\text{in}}(x) = \frac{1}{|\mathcal{D}|} \sum_{\alpha \in \mathcal{D}} \delta(x - x_\alpha) \, , \tag{7}$$

where the index α labels the elements of the dataset. For a given set of parameters, the function parametrized by the NN, which we denote f_θ, is given by the preactivation function of the output layer L,

$$f(x; \theta) = \phi^{(L)}(x) \, . \tag{8}$$

The preactivations $\phi^{(\ell)}$ and the activations $\rho^{(\ell)}$ are defined recursively as

$$\rho_i^{(0)}(x) = x_i \, , \tag{9}$$

$$\phi_i^{(\ell+1)}(x) = \sum_{j=1}^{n_\ell} W_{ij}^{(\ell+1)} \rho_j^{(\ell)}(x) + b_i^{(\ell+1)} \, , \tag{10}$$

$$\rho_i^{(\ell)}(x) = \rho\big(\phi_i^{\ell}(x)\big) \, , \tag{11}$$

where ρ is the activation function. The function

$$F^{(L)} : \quad \mathbb{R}^P \to \mathcal{F} \tag{12}$$

$$\theta \mapsto F^{(L)}(\theta) = f_\theta \tag{13}$$

is called the NN realization function. We also introduce the (dual) space of linear functionals acting on $\mathcal{F}$,

$$\mathcal{F}^* = \{ \mu : \mathcal{F} \to \mathbb{R} \, ; \; \mu = \langle d, \cdot \rangle_{\text{in}}, \; d \in \mathcal{F} \} \, , \tag{14}$$

and a loss function

$$\mathcal{L} : \mathcal{F} \times \mathcal{Z} \to \mathbb{R} \, . \tag{15}$$

Here $\mathcal{Z}$ denotes the space of datapoints, which we can identify for now with $\mathcal{X} \times \mathcal{Y}$.

1.2 Kernel Gradient

A kernel K is defined as a symmetric function

$$K : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_L \times n_L} , \tag{16}$$

such that $K(x, x') = K(x', x)^T$. Using the kernel, a bilinear map is defined on $\mathcal{F}$

$$\langle f, g \rangle_K = \mathbb{E}_{x,x' \sim p^{\text{in}}} \left[f(x)^T K(x, x') g(x') \right] . \tag{17}$$

The kernel is positive definite if

$$||f||_{p^{\text{in}}} > 0 \quad \Longrightarrow \quad ||f||_K > 0 . \tag{18}$$

Given a kernel K, we define a map

$$\Phi_K : \mathcal{F}^* \to \mathcal{F} \tag{19}$$

$$\mu \mapsto f_\mu , \tag{20}$$

such that

$$f_{\mu,i}(x) = \langle d, K_{i\cdot}(x, \cdot) \rangle_{p^{\text{in}}} \tag{21}$$

$$= \sum_{j=1}^{n_L} \int dx' \, p^{\text{in}}(x') \, d_j(x') K_{ij}(x, x') , \tag{22}$$

where $d \in \mathcal{F}$, and $\mu = \langle d, \cdot \rangle_{p^{\text{in}}}$.

The cost function for a given dataset depends on the values of the function f at the datapoints x_α, so that we can write

$$C_{\mathcal{D}}(f) = \widehat{C}(\mathbf{f}) , \tag{23}$$

where $\mathbf{f}$ is the finite-dimensional vector made of the values $f_\alpha = f(x_\alpha)$ for $\alpha = 1, \ldots, |\mathcal{D}|$. The variation of the cost function at a point $f_0 \in \mathcal{F}$ is

$$\delta C_{\mathcal{D}}\big|_{f_0} = C_{\mathcal{D}}(f_0 + \delta f) - C_{\mathcal{D}}(f_0) \tag{24}$$

$$= \sum_{\alpha \in \mathcal{D}} \left. \frac{\partial \widehat{C}}{\partial f_\alpha} \right|_{f_0} \delta f_\alpha \tag{25}$$

$$= \partial_f^{\text{in}} C\big|_{f_0} \, \delta f \tag{26}$$

$$= \langle \, d|_{f_0} \, , \delta f \rangle_{p^{\text{in}}} , \tag{27}$$

where $\delta f \in \mathcal{F}$, $\delta f_\alpha = \delta f(x_\alpha)$ and $\partial_f^{\text{in}} C\big|_{f_0} \in \mathcal{F}^*$.

Using the kernel K, the *kernel gradient* $\nabla_K C|_{f_0}$ is defined as $\Phi_K\left(\partial_f^{\text{in}} C\big|_{f_0}\right)$, *i.e.*

$$\nabla_K C|_{f_0}(x) = \frac{1}{|\mathcal{D}|}\sum_{\alpha\in\mathcal{D}} K(x,x_\alpha)\left.\frac{\partial \widehat{C}}{\partial f_\alpha}\right|_{f_0} . \tag{28}$$

A function $f(t)$ follows the kernel gradient descent if

$$\partial_t f(t) = -\nabla_K C|_{f(t)} . \tag{29}$$

During the kernel descent, the cost function variation is given by

$$\partial_t\, C|_{f(t)} = -\langle d|_{f(t)}\,, \nabla_K C|_{f(t)}\rangle_{p^{\text{in}}} = -||\, d|_{f(t)}\,||_K^2 . \tag{30}$$

More generally, for a function $\widehat{O}$ that depends on the values of f on the points in the dataset,

$$\partial_t\, \widehat{O}\Big|_{f(t)} = -\left\langle \frac{\partial \widehat{O}}{\partial f}, \frac{\partial \widehat{C}}{\partial f}\right\rangle_K . \tag{31}$$

2 Gradient Descent for NNPDF

We work in a basis where the data covariance C_Y is diagonal. Such a choice has two main advantages:

1. The data points in this basis are statistically independent, so that when the data is split, e.g. between training and validation sets, the points in each subset are not correlated.

2. Working in a basis where the data covariance is diagonal, requires to compute the eigenvectors and eigenvalues of C_Y, thereby automatically checking for a potentially large condition number.

Note that, in case we need to apply kinematic cuts, these can be applied before rotating to the diagonal basis. The diagonal basis is then defined by determining the eigensystem of the reduced covariance matrix, i.e. the matrix obtained by removing the lines and columns that correspond to data points that do not pass the kinematic cuts.

The loss function is defined to be the Mean Square Error (MSE),

$$\mathcal{L} = \frac{1}{2}(y-T)^T C_Y^{-1}(y-T) , \tag{32}$$

$$= \frac{1}{2}||y-T||^2_{C_Y} , \tag{33}$$

where

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_I \\ \vdots \\ y_{N_{\rm dat}} \end{pmatrix}, \quad \text{and} \quad T = \begin{pmatrix} T_1 \\ \vdots \\ T_I \\ \vdots \\ T_{N_{\rm dat}} \end{pmatrix} \tag{34}$$

are vectors containing the data points y_I and their respective theoretical predictions T_I, where $I = 1, \ldots, N_{\rm dat}$. The loss function can be written using explicit indices,

$$\mathcal{L} = \frac{1}{2} \sum_I \frac{1}{\sigma_I^2} \left(y_I - T_I\right)^2 . \tag{35}$$

However, here and in what follows, we try to minimize the usage of explicit indices.

Denoting by t the training time, and by θ_μ, with $\mu = 1, \ldots, P$, the parameters used to specify the functional form of f, gradient descent is defined by the flow equation,

$$\frac{d}{dt}\theta_{\mu,t} = -\sum_\nu \lambda_{\mu\nu} \nabla_\nu \mathcal{L}_t \,, \tag{36}$$

where we added a suffix $\mathcal{L}_t$ to emphasise that the loss function is evaluated using the function f_t at training time t. We introduced the tensor $\lambda_{\mu\nu}$ to take into account the case of a non-trivial metric in the space of parameters. We will use $\lambda_{\mu\nu} = \lambda_\mu \delta_{\mu\nu}$, unless explicitly stated. The gradient on the RHS of Eq. 36 above is

$$\nabla_\mu \mathcal{L}_t = -\left(\nabla_\mu f_t\right)^T \left(\frac{\partial T}{\partial f}\right)_t^T C_Y^{-1} \epsilon_t \,, \tag{37}$$

where we introduced $\epsilon_t = y - T_t$, to denote the difference between the data and the theoretical prediction obtained from f_t. The quantities that depend on the training time have an explicit suffix t. Using the chain rule yields

$$\frac{d}{dt} f_t = \sum_\mu \left(\nabla_\mu f_t\right) \frac{d}{dt}\theta_{\mu,t} \tag{38}$$

$$= \left[\sum_{\mu,\nu} \lambda_{\mu\nu} \left(\nabla_\mu f_t\right) \left(\nabla_\nu f_t\right)^T\right] \left(\frac{\partial T}{\partial f}\right)_t^T C_Y^{-1} \epsilon_t \,. \tag{39}$$

Introducing the Neural Tangent Kernel (NTK),

$$\Theta_t = \sum_{\mu,\nu} \lambda_{\mu\nu} \left(\nabla_\mu f_t\right) \left(\nabla_\nu f_t\right)^T , \tag{40}$$

or equivalently using explicit indices for once,

$$\Theta_{\delta_1\delta_2,t} = \sum_{\mu\nu} \lambda_{\mu\nu} \nabla_\mu f_{\delta_1,t} \, \nabla_\nu f_{\delta_2,t} \, . \tag{41}$$

Note that the NTK is a tensor in the space of the grid points on which the function f is evaluated, *i.e.* $f_\delta = f(x_\delta)$.

The flow equation becomes

$$\frac{d}{dt} f_t = \Theta_t \left(\frac{\partial T}{\partial f} \right)_t^T C_Y^{-1} \epsilon_t \, . \tag{42}$$

This flow equation is completely general, for any parametrization and any dependence of the theoretical predictions on the function f.

Let us now focus on the case where the theoretical prediction is linear in f,

$$T_I = (\mathrm{FK})_{I\alpha} f_\alpha \, , \tag{43}$$

so that

$$\left(\frac{\partial T_I}{\partial f_\alpha} \right) = (\mathrm{FK})_{I\alpha} \, , \tag{44}$$

is actually independent of t. Note in passing that (FK) is not a square matrix, and hence not invertible.

The loss function is a quadratic form in f,

$$2\mathcal{L} = f^T M f - f^T (\mathrm{FK})^T C_Y^{-1} y - y^T C_Y^{-1} (\mathrm{FK}) f + y^T C_Y^{-1} y \, . \tag{45}$$

We can restrict f to the subspace orthogonal to ker(FK); the components in ker(FK) do not enter the loss function and therefore are unconstrained. In this subspace, the matrix

$$M = (\mathrm{FK})^T C_Y^{-1} (\mathrm{FK}) \tag{46}$$

is symmetric and positive definite, and hence invertible. Rewriting

$$f = \chi + M^{-1} (\mathrm{FK})^T C_Y^{-1} y \, , \tag{47}$$

yields

$$2\mathcal{L} = \chi^T M \chi + y^T C_Y^{-1} \left[1 - (\mathrm{FK}) M^{-1} (\mathrm{FK})^T C_Y^{-1} \right] y \, . \tag{48}$$

Therefore the absolute minimum is obtained for

$$f^* = M^{-1} (\mathrm{FK})^T C_Y^{-1} y \quad \Longrightarrow \quad \mathcal{L}^* = \frac{1}{2} y^T C_Y^{-1} \left[1 - (\mathrm{FK}) M^{-1} (\mathrm{FK})^T C_Y^{-1} \right] y \, . \tag{49}$$

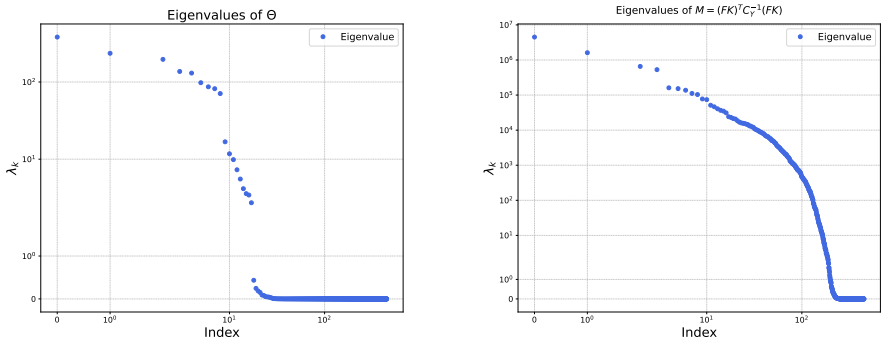

Figure 1: Eigenvalues of the NTK (left) and of the matrix M (right) in logarithmic scale.

Note that the minimum of the loss function is *not* zero. However, in a level-0 closure test, $y = (\mathrm{FK})f^{\mathrm{in}}$, and then $f^* = f^{\mathrm{in}}$ and $\mathcal{L}^* = 0$. The absolute minimum of the loss yields the correct input function and a vanishing loss function. This is something we always stress in NNPDF. If we push the training for long enough in L0, we can get a vanishing loss. The flow equation for linear data becomes

$$\frac{d}{dt} f_t = \Theta_t \, (\mathrm{FK})^T C_Y^{-1} \epsilon_t \, . \tag{50}$$

Evolution of f_t. In order to solve for the evolution of f_t, we start by finding the eigenvectors and eigenvalues of Θ,

$$\Theta v_i = \lambda_i v_i \, . \tag{51}$$

The eigenvalues of the NTK, together with those of the matrix M, are displayed in Fig. 1. We rewrite the flow equation, Eq. 50, as

$$\frac{d}{dt} f_t = b - \Theta M f_t \, , \tag{52}$$

where

$$b = \Theta (\mathrm{FK})^T C_Y^{-1} y \, , \tag{53}$$

while the matrix M has been introduced above. Using the eigendecomposition $M = RDR^T$, where R is an orthogonal matrix, we introduce

$$\tilde{f}_t = D^{1/2} R^T f_t \, , \quad \tilde{b} = D^{1/2} R^T b \, , \tag{54}$$

so that

$$\frac{d}{dt}\tilde{f}_t = \tilde{b} - D^{1/2}R^T\Theta R D^{1/2}\tilde{f}_t \tag{55}$$

$$= \tilde{b} - \tilde{H}\tilde{f}_t\,. \tag{56}$$

The eigenvalues of the matrix $\tilde{H}$ are shown in Fig. 2.

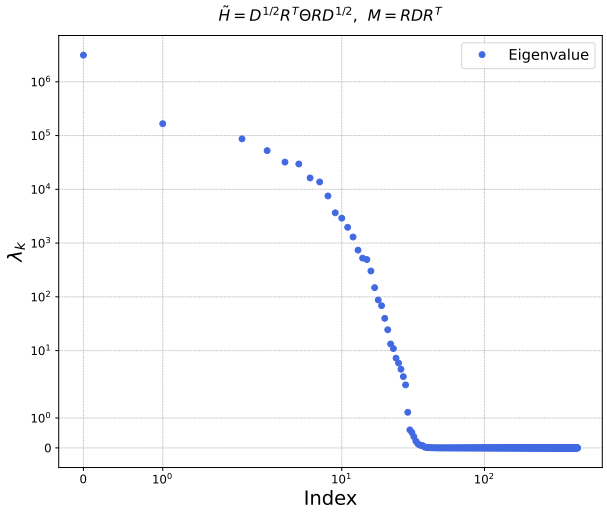

Figure 2: Eigenvalues of $\tilde{H}$ in logarithmic scale.

The solution, assuming that $\tilde{H}$ is independent of the flow time, is

$$\tilde{f}_t = e^{-\tilde{H}t}\tilde{f}_0 + \left(1 - e^{-\tilde{H}t}\right)\tilde{H}^{-1}\tilde{b}\,. \tag{57}$$

Taking the limit $t \to \infty$, we find

$$f_\infty = M^{-1}(\mathrm{FK})^T C_Y^{-1} y = f^*\,, \tag{58}$$

and thus

$$\epsilon_\infty = \left[1 - (\mathrm{FK})M^{-1}(\mathrm{FK})^T C_Y^{-1}\right] y\,. \tag{59}$$

A few lines of algebra yield

$$\frac{1}{2}||\epsilon_\infty||^2_{C_Y} = \mathcal{L}^*\,, \tag{60}$$

in agreement with Eq. 49. As before, if we consider a level-0 test,

$$f_\infty = M^{-1}(\mathrm{FK})^T C_Y^{-1}(\mathrm{FK}) f^{\mathrm{in}} = f^{\mathrm{in}}\,. \tag{61}$$

Evolution for ϵ_t. We can rewrite Eq. 50 as

$$\begin{aligned}\frac{d}{dt}\epsilon_t &= -(\mathrm{FK})\frac{d}{dt}f_t \\ &= -(\mathrm{FK})\Theta(\mathrm{FK})^T C_Y^{-1}\epsilon_t \\ &= -(\mathrm{FK})\Theta(\mathrm{FK})^T R_Y D_Y R_Y^T \epsilon_t \,, \end{aligned} \tag{62}$$

where we decompose $C_Y^{-1} = R_Y D_Y R_Y^T$. Following the derivation for f_t, we define

$$\tilde{\epsilon}_t = D_Y^{1/2} R_Y^T \epsilon_t \,, \tag{63}$$

so that

$$\frac{d}{dt}\tilde{\epsilon}_t = -\tilde{H}_\epsilon \tilde{\epsilon}_t \,, \tag{64}$$

where

$$\tilde{H}_\epsilon = D_Y^{1/2} R_Y^T (\mathrm{FK})\Theta(\mathrm{FK})^T R_Y D_Y^{1/2} \tag{65}$$

is a symmetric, positive definite, matrix, and therefore

$$\epsilon_t = R_Y D_Y^{-1/2} e^{-\tilde{H}_\epsilon t} D_Y^{1/2} R_Y^T \epsilon_0 \,. \tag{66}$$

Note that

$$\begin{aligned}\tilde{H}_\epsilon \tilde{\epsilon}_\infty &= D_Y^{1/2} R_Y^T (\mathrm{FK})\Theta \underbrace{(\mathrm{FK})^T C_Y^{-1}\left[1 - (\mathrm{FK})M^{-1}(\mathrm{FK})^T C_Y^{-1}\right]}_{(*)} y \,, \\ (*) &= (\mathrm{FK})^T C_Y^{-1} - \underbrace{(\mathrm{FK})^T C_Y^{-1}(\mathrm{FK})M^{-1}}_{=1!}(\mathrm{FK})^T C_Y^{-1} = 0 \,, \end{aligned} \tag{67}$$

and therefore the evolution operator for $\tilde{\epsilon}$, $\tilde{H}_\epsilon$, has a zero mode, which is not driven to zero by the gradient flow. The zero mode is exactly ϵ_∞, *i.e.* the residual error at the end of the gradient flow. All other components of ϵ_t vanish exponentially with t. This result is consistent with the result above for the evolution of f_t. Now we need to check them numerically!

Consistency of the two evolutions. As a cross check of our calculation, we verify that the evolution of ϵ_t is consistent with the evolution of f_t. Starting from Eq. 63,

$$\tilde{\epsilon}_t = D_Y^{1/2} R_Y^T \left(y - (\mathrm{FK})RD^{-1/2}\tilde{f}_t\right) \,. \tag{68}$$

We compute separately the time-independent and the time-dependent contributions to $\tilde{\epsilon}_t$ in the RHS of Eq. 68. Starting from the time-independent terms, we have

$$\begin{aligned} & y - (\mathrm{FK}) R D^{-1/2} \tilde{H}^{-1} \tilde{b} = \\ & \quad = y - (\mathrm{FK}) \underbrace{R D^{-1/2} \left[D^{-1/2} R^T \right]}_{=M^{-1}} \Theta^{-1} \underbrace{R D^{-1/2}}_{=1} \left[D^{1/2} R^T \right] \Theta (\mathrm{FK})^T C_Y^{-1} y] \\ & \quad = \epsilon_\infty \,, \end{aligned} \tag{69}$$

and therefore

$$D_Y^{1/2} R_Y^T \left(y - (\mathrm{FK}) R D^{-1/2} \tilde{H}^{-1} \tilde{b} \right) = \tilde{\epsilon}_\infty \,. \tag{70}$$

The time-independent component of $\tilde{\epsilon}_t$ yields $\tilde{\epsilon}_\infty$, as expected. The time-dependent terms in $(\mathrm{FK}) R D^{-1/2} \tilde{f}_t$ are

$$(\mathrm{FK}) R D^{-1/2} e^{-\tilde{H} t} \left(\tilde{H}^{-1} \tilde{b} - \tilde{f}_0 \right) . \tag{71}$$

Focussing on the first term above,

$$\begin{aligned} & (\mathrm{FK}) R D^{-1/2} e^{-\tilde{H} t} \tilde{H}^{-1} \tilde{b} = \\ & \quad = (\mathrm{FK}) R D^{-1/2} \tilde{H}^{-1} \tilde{b} + \sum_{k>0} (\mathrm{FK}) R D^{-1/2} \frac{(-t)^k}{k!} \left(D^{1/2} R^T \Theta R D^{1/2} \right)^{k-1} D^{1/2} R^T b \\ & \quad = (\mathrm{FK}) R D^{-1/2} \tilde{H}^{-1} \tilde{b} + \sum_{k>0} \frac{(-t)^k}{k!} \underbrace{(\mathrm{FK}) \left(\Theta M \right)^{k-1}}_{(*)} b \,. \end{aligned} \tag{72}$$

Note that

$$(*) = (\mathrm{FK}) \underbrace{\left[\Theta (\mathrm{FK})^T C_Y^{-1} (\mathrm{FK}) \right] \left[\Theta (\mathrm{FK})^T C_Y^{-1} (\mathrm{FK}) \right] \ldots}_{k-1 \text{ factors}} \tag{73}$$

$$= \underbrace{\left[(\mathrm{FK}) \Theta (\mathrm{FK})^T C_Y^{-1} \right] \left[(\mathrm{FK}) \Theta (\mathrm{FK})^T C_Y^{-1} \right] \ldots}_{k-1 \text{ factors}} (\mathrm{FK}) \tag{74}$$

$$= \left[(\mathrm{FK}) \Theta (\mathrm{FK})^T C_Y^{-1} \right]^{k-1} (\mathrm{FK}) \,, \tag{75}$$

and therefore

$$(\mathrm{FK}) R D^{-1/2} e^{-\tilde{H} t} \tilde{H}^{-1} \tilde{b} = \sum_k \frac{(-t)^k}{k!} \left[(\mathrm{FK}) \Theta (\mathrm{FK})^T C_Y^{-1} \right]^k y - \epsilon_\infty \,. \tag{76}$$

Hence the contribution to $\tilde{\epsilon}_t$ from this term

$$\begin{aligned}\sum_k \frac{(-t)^k}{k!} D_Y^{1/2} R_Y^T \left[(\mathrm{FK})\Theta(\mathrm{FK})^T R_Y D_Y^{1/2} D_Y^{1/2} R_Y^T\right]^k y = \\ = \sum_k \frac{(-t)^k}{k!} \left[D_Y^{1/2} R_Y^T (\mathrm{FK})\Theta(\mathrm{FK})^T R_Y D_Y^{1/2}\right]^k D_Y^{1/2} R_Y^T y \\ = e^{-\tilde{H}_\epsilon t} \left(\tilde{y} - \tilde{\epsilon}_\infty\right) . \end{aligned} \tag{77}$$

The second term in Eq. 71 reads

$$(\mathrm{FK}) R D^{-1/2} e^{-\tilde{H}t} \tilde{f}_0 = (\mathrm{FK}) R D^{-1/2} e^{-\tilde{H}t} D^{1/2} R^T f_0 \tag{78}$$

$$= \sum_k \frac{(-t)^k}{k!} (\mathrm{FK}) \left(\Theta M\right)^k f_0 \tag{79}$$

$$= \sum_k \frac{(-t)^k}{k!} \left[(\mathrm{FK})\Theta(\mathrm{FK})^T C_Y^{-1}\right]^k (\mathrm{FK}) f_0 \,, \tag{80}$$

and hence the contribution to $\tilde{\epsilon}_t$

$$D_Y^{1/2} R_Y^T R_Y D_Y^{-1/2} e^{-\tilde{H}_\epsilon t} D_Y^{1/2} R_Y^T (\mathrm{FK}) f_0 = e^{-\tilde{H}_\epsilon t} D_Y^{1/2} R_Y^T \left((\mathrm{FK}) f_0\right) . \tag{81}$$

Combining Eqs. 77 and 81 yields

$$e^{-\tilde{H}_\epsilon t} D_Y^{1/2} R_Y^T \left(y - (\mathrm{FK}) f_0\right) - \tilde{\epsilon}_\infty = e^{-\tilde{H}_\epsilon t} \tilde{\epsilon}_0 - \tilde{\epsilon}_\infty \,. \tag{82}$$

And finally, adding Eqs. 69 and 82, we obtain

$$\tilde{\epsilon}_t = e^{-\tilde{H}_\epsilon t} \tilde{\epsilon}_0 \,, \tag{83}$$

as expected.

2.1 Evolution reloaded

A basis in the N_{grid}-dimensional space of vectors $f_t \in \mathbb{R}^{N_{\mathrm{grid}}}$, $\left\{ z^{(k)}; k = 1, \ldots, N_{\mathrm{grid}} \right\}$, can be constructed from the eigenvectors of Θ,

$$\Theta z^{(k)} = \lambda^{(k)} z^{(k)} \,. \tag{84}$$

Since Θ is symmetric and positive semi-definite, we know that $\lambda^{(k)} \geq 0$ and that the eigenvectors are a complete set of orthonormal vectors. It is useful to introduce the following notation:

$$k \in \mathcal{I}_{\Theta\perp} \,, \text{ if } \lambda^{(k)} \neq 0 \,, \quad k \in \mathcal{I}_{\Theta\parallel} \,, \text{ if } \lambda^{(k)} = 0 \,. \tag{85}$$

The dimension of the kernel of Θ is $N_{\Theta\|} = |\mathcal{I}_{\Theta\|}|$, the dimension of its orthogonal complement is $N_{\Theta\perp} = |\mathcal{I}_{\Theta\perp}|$, and $N_{\Theta\|} + N_{\Theta\perp} = N_{\rm grid}$.

Projecting the flow equation onto the basis vectors $z^{(k)}$ yields

$$\frac{d}{dt} f_{t,k} = 0\,, \quad \text{for } k \in \mathcal{I}_{\Theta\|}\,, \tag{86}$$

$$\frac{d}{dt} f_{t,k} = -\lambda^{(k)} \left(z^{(k)}, M f_t\right) + \lambda^{(k)} \left(z^{(k)}, (\mathrm{FK})^T C_Y^{-1} y\right)\,, \quad \text{for } k \in \mathcal{I}_{\Theta\perp}\,, \tag{87}$$

where $f_{t,k} = \left(z^{(k)}, f_t\right)$. Using the completeness of the basis,

$$\left(z^{(k)}, M f_t\right) = \sum_{k'} \left(z^{(k)}, M z^{(k')}\right) f_{t,k'} \tag{88}$$

$$= \sum_{k' \in \mathcal{I}_{\Theta\perp}} \left(z^{(k)}, M z^{(k')}\right) f_{t,k'} + \sum_{k' \in \mathcal{I}_{\Theta\|}} \left(z^{(k)}, M z^{(k')}\right) f_{0,k'}\,. \tag{89}$$

Finally, we can rewrite Eq. (87) as

$$\frac{d}{dt} f_{t,k} = -\sum_{k' \in \mathcal{I}_{\Theta\perp}} H^{\perp}_{kk'} f_{t,k'} + B_k\,, \tag{90}$$

where

$$H^{\perp}_{kk'} = \lambda^{(k)} \left(z^{(k)}, M z^{(k')}\right)\,, \tag{91}$$

is a $N_{\Theta\perp} \times N_{\Theta\perp}$ matrix, and

$$B_k = -\sum_{k' \in \mathcal{I}_{\Theta\|}} \lambda^{(k)} \left(z^{(k)}, M z^{(k')}\right) f_{0,k'} + \lambda^{(k)} \left(z^{(k)}, (\mathrm{FK})^T C_Y^{-1} y\right)\,, \tag{92}$$

is a $N_{\Theta\perp}$-dimensional vector, with $k \in \mathcal{I}_{\Theta\perp}$.

The other basis, $\left\{v^{(m)}; m = 1, \ldots, N_{\rm grid}\right\}$, is made of eigenvectors of M,

$$M v^{(m)} = \sigma^{(m)} v^{(m)}\,. \tag{93}$$

Similarly to the convention introduced above, the eigenvectors in the kernel of M are denoted by indices in $\mathcal{I}_{M\|}$, while the eigenvectors in the orthgonal complement are labelled by indices in $\mathcal{I}_{M\perp}$.

2.2 Evolution of f_t revisited.

We start again from the eigenvectors and eigenvalues of Θ

$$\Theta \boldsymbol{e}^{(k)} = \lambda_k \boldsymbol{e}^{(k)}\,, \tag{94}$$

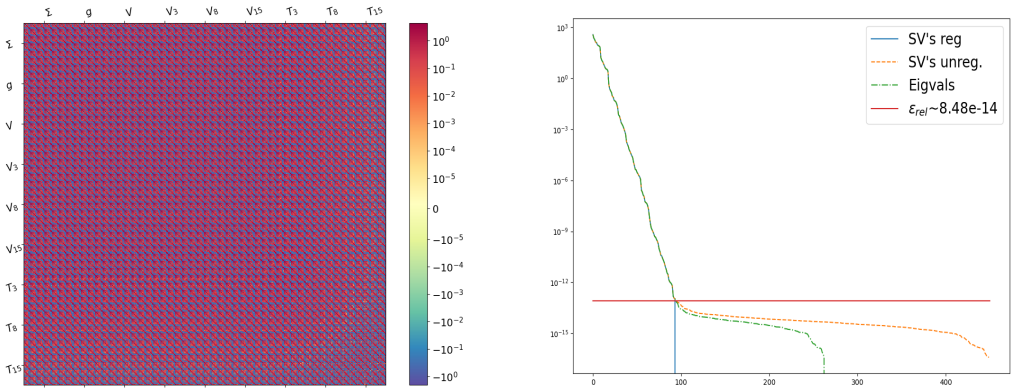

Figure 3: (Left) Representation of the NTK matrix in logarithmic scale. (Right) Decomposition of the NTK matrix in terms of the eigenvalues and of the singular values of the matrix M, and the subsequent regularisation (see main text).

to which we apply the same analysis carried out in Appendix B. In this section, we use a feed-forward neural network with architecture $[1, 28, 20, 9]$. The activation function used in both the two deep layers is `tanh`, while the output function is linear. The weights, rescaled by the square root of the size of the previous layer, are initialised using a normal distribution. The biases are initialised to zero. The input data are the points of the grid in x-space taken from the FK tables.

We start by looking at the NTK matrix, displayed in the left panel of Fig. 3. Upon inspecting the figure, we see that the NTK is uniform across flavours and bins in x. This is indeed expected, as the NTK does not bring any physical information and it does not favour any particular flavour to start with. In other words, the left plot in Fig. 3 shows the amount of information per flavour that we can ever reach given the network architecture and the choice of the grid in x.

The right panel of Fig. 3 shows the decomposition of the NTK matrix in terms of the singular values and in terms of the eigenvalues. The relative machine precision for double precision floating numbers ($\epsilon \sim 2.2 \times 10^{-16}$) is also shown. We see that the singular values and eigenvalues start to diverge when machine precision is reached. This is a symptom of noise in the decomposition, and values below this threshold are set to zero. Above threshold, the singular values and eigenvalues are consistent and they show the usual hierarchy of the NTK extensively discussed in the literature.

The physical information is brought in by the FK tables, which combine with NTK in the flow equation, reported below for convenience

$$\frac{d}{dt}\boldsymbol{f}_t = \boldsymbol{b} - \Theta M \boldsymbol{f}_t \,. \tag{95}$$

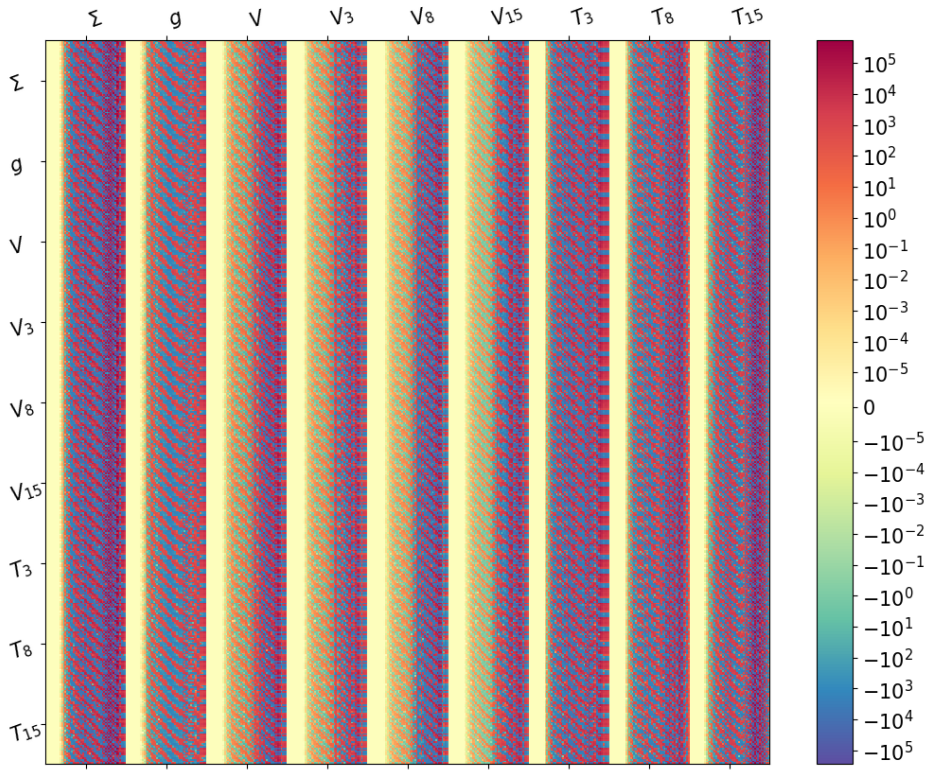

Figure 4:

In a standard ML application of neural networks, the evolution kernel in eq. 95 would be the NTK solely. On the other hand, in the context of PDF fitting, we are interested in solving the inverse problem that arise from the factorization theorem, and that is encoded in the matrix M. This introduces additional complications, as the inverse problem is generally ill-posed and the matrix M is not well-defined (see Appendix B). Moreover, the product taking place in the second term on the rhs of the equation above mixes the NTK and the FK spaces, making it harder to disentangle the information brought in by the NTK and the FK tables separately.

The resulting evolution kernel is shown in Fig. 4. Compared to the left panel of Fig. (3), the kernel preserves the same underlying structure, yet modified by the action of the FK tables. We also see that the overall architecture (network and FK tables) is not sensitive to the low-x region, while the mid- and large-x region is more sensitive.

Unfortunately, the evolution kernel is singular and cannot be inverted as it is, preventing any attempt to integrate the flow equation eq. (95). In order to proceed, we need to decompose the kernel in its various components and discard the zero modes. This is the subject of the next section.

Projection of the flow equation

The evolution kernel cannot be inverted directly, due to the presence of many zero modes. We then resort to working in a subspace of the kernel, where it can be inverted. In doing so, we will also try to disentangle the effects of the NTK and of the FK tables.

We start by projecting the flow equation in the eigenspace of the NTK, so that we have

$$\frac{d}{dt}f'_{k,t} = b'_k - \lambda_k(\boldsymbol{e}^{(k)}, M\boldsymbol{f}_t). \tag{96}$$

Here the prime denotes the k-th component of the vector in the eigenspace of the NTK, and $(\boldsymbol{e}^{(k)}, Mf_t)$ is the ordinary scalar product. Eq. (96) can also be recast into matrix form. Writing $\Theta = U_\Theta \Sigma_\Theta U_\Theta^T$, then we have

$$\frac{d}{dt}\boldsymbol{f}'_t = \boldsymbol{b}' - \Sigma_\Theta U_\Theta^T M U_\Theta \boldsymbol{f}' = \boldsymbol{b}' - \Sigma_\Theta M' \boldsymbol{f}'. \tag{97}$$

The equation above requires two observations. The first one is that some eigenvalues of the NTK are actually zero, so that the second term on the rhs of eq. (96) is zero. Furthermore, for the first term on the rhs we have

$$b'_k = (\boldsymbol{e}^{(k)}, \boldsymbol{b}) = (\boldsymbol{e}^{(k)}, \Theta(FK)^T C_Y^{-1} \boldsymbol{Y}) = 0 \quad \forall \boldsymbol{e}^{(k)} \in Ker(\Theta). \tag{98}$$

Thus, for zero eigenvalues of the NTK, we have

$$\frac{d}{dt}f'_{k,t} = 0 \quad \Rightarrow f'_{k,t} = f'_{k,0}, \tag{99}$$

abd this result holds regardless of the FK tables.

The second observation involves the matrix M. In particular, we can decompose M as follows

$$M = LL^T = (RD^{1/2})(D^{1/2}R^T), \tag{100}$$

where D is the diagonal matrix of the eigenvalues (or singular values) of M and R is the matrix of the eigenvectors. Using the completness relation of the vectors $\boldsymbol{e}^{(k)}$, we can write

$$M'_{k_1 k_2} = (e^{(k_1)}, Le^{(k)})(e^{(k)}, L^T e^{(k_2)}) = L'_{k_1 k}(L^T)'_{k k_2} = L'_{k_1 k} L'_{k_2 k}, \tag{101}$$

or, in matrix form,

$$M' = U_\Theta^T M U_\Theta = U_\Theta^T L U_\Theta U_\Theta^T L^T U_\Theta = L'(L')^T, \tag{102}$$

where in the second equality we have used the completeness relation of the eigenvectors.

A close-up inspection of L' reveals that

$$\begin{aligned} L'_{k_1 k_2} &\equiv \left(\boldsymbol{e}^{(k)}, RD^{1/2}\boldsymbol{e}^{(k)}\right) \\ &= e^{(k_1)}_{i_1} R_{i_1 i_2} \mu_{i_2} e^{(k_2)}_{i_2} \\ &= e^{(k_1)}_{i_1} v^{(i_2)}_{i_1} \mu_{i_2} e^{(k_2)}_{i_2} \\ &= \mu_{i_2} e^{(k_2)}_{i_2} (\boldsymbol{e}^{(k_1)}, \boldsymbol{v}^{(i_2)}), \end{aligned} \tag{103}$$

which means that the intersection between the NTK and the FK tables takes place in the scalar product of the last equality above.

By means of eq. (101), we can rewrite the flow equation as follows

$$\frac{d}{dt} f'_{k_1,t} = b'_{k_1} - \lambda_{k_1} L'_{k_1 k_2} L'_{k_3 k_2} f'_{k_3,t}\,. \tag{104}$$

Finally, in order to obtain a symmetric kernel, we can multiply both sides by $(L')^T$ so that we have

$$\frac{d}{dt} \tilde{f}'_{k_1,t} = \tilde{b}'_{k_1} - \tilde{K}_{k_1 k_2} \tilde{f}'_{k_2,t} \tag{105}$$

where $\tilde{f}'_{k_1,t}$ and $\tilde{b}'_{k_1}$ are the components rotated by the action of of L', and $\tilde{K}_{k_1 k_2}$ is the symmetric kernel defined as

$$\tilde{K} = (L')^T \Sigma_\Theta L'. \tag{106}$$

Removing zero modes of the NTK

In obtaining Eq. (105), we have kept all the zero modes of the NTK. However, we can filter them out and write an expression that contains zero modes of the FK tables solely. To do so, we restart from eq. (97), and we recall that the rotations applied to the vectors are

$$\boldsymbol{b}' = U_\Theta^T \boldsymbol{b} \quad \text{and} \quad \boldsymbol{f}'_t = U_\Theta^T \boldsymbol{f}_t\,. \tag{107}$$

In order to separate that zero modes, we can split the vector $\boldsymbol{f}'_t$ into a part that is orthogonal to the kernel of the NTK, and another which lives in the kernel and hence is constant

$$\frac{d}{dt}\begin{pmatrix} \boldsymbol{f}'_{\perp,t} \\ \boldsymbol{f}'_{\parallel,t} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}'_\perp \\ \boldsymbol{b}'_\parallel \end{pmatrix} - \begin{pmatrix} \Sigma_\Theta & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} M'_{\perp\perp} & M'_{\perp\parallel} \\ M'_{\parallel\perp} & M'_{\parallel\parallel} \end{pmatrix} \begin{pmatrix} \boldsymbol{f}'_{\perp,t} \\ \boldsymbol{f}'_{\parallel,t} \end{pmatrix} \tag{108}$$

where the subscripts $\perp$ and $\parallel$ denote the components orthogonal and parallel to the kernel of the NTK, respectively. The equation above unravels as follows

$$\frac{d}{dt}\boldsymbol{f}'_{\perp,t} = \boldsymbol{b}'_\perp - \Sigma_\Theta \left(M'_{\perp\perp}\boldsymbol{f}'_{\perp,t} + M'_{\perp\parallel}\boldsymbol{f}'_{\parallel,t}\right), \tag{109}$$

$$\frac{d}{dt}\boldsymbol{f}'_{\parallel,t} = \boldsymbol{0}, \tag{110}$$

where the second equation is a consequence of the fact that the kernel of the NTK is a zero mode for the learning flow. Note that the parallel component $\boldsymbol{f}'_{\parallel,t}$ appears in the flow equation of $\boldsymbol{f}'_{\perp,t}$, but it does not evolve in time. Hence, it can be reabsorbed in the constant factor $\boldsymbol{b}'_{\perp}$, giving

$$\frac{d}{dt}\boldsymbol{f}'_{\perp,t} = \boldsymbol{B}'_{\perp} - \Sigma_{\Theta} M'_{\perp\perp} \boldsymbol{f}'_{\perp,t}, \tag{111}$$

where

$$\boldsymbol{B}'_{\perp} = \boldsymbol{b}'_{\perp} - \Sigma_{\Theta} M'_{\perp\parallel} \boldsymbol{f}'_{\parallel,t}. \tag{112}$$

Eq. (111) restricts the evolution of the vector $\boldsymbol{f}'_t$ to the components orthogonal to the kernel of the NTK. The only flat directions left are those encoded in the FK tables.

Flat directions of the FK tables

Following the same reasoning that led to eq. (105), we can rewrite eq. (111) as follows

$$\frac{d}{dt}\tilde{\boldsymbol{f}}'_{\perp,t} = \tilde{\boldsymbol{B}}'_{\perp} - \tilde{K}_{\perp} \tilde{\boldsymbol{f}}'_{\perp,t}, \tag{113}$$

where

$$M'_{\perp\perp} = L'_{\perp}(L'_{\perp})^T, \quad \tilde{\boldsymbol{f}}'_{\perp,t} = (L'_{\perp})^T \boldsymbol{f}'_{\perp,t}, \quad \text{and} \quad \tilde{K}_{\perp} = (L'_{\perp})^T \Sigma_{\Theta} L'_{\perp}. \tag{114}$$

The entries of the symmetric kernel $\tilde{K}_{\perp}$ and its decomposition are shown in the left and right panel of Fig. ... As previously done for then NTK, we can rewrite this kernel as follows

$$\tilde{K}_{\perp} = U_K \Sigma_K U_K^T, \tag{115}$$

and rotate equation eq. (113) to the eigenbasis of the kernel

$$\frac{d}{dt}\hat{\boldsymbol{f}}_{\perp} = \hat{\boldsymbol{B}}_{\perp} - \Sigma_K \hat{\boldsymbol{f}}_{\perp,t}, \tag{116}$$

where

$$\hat{\boldsymbol{f}}_{\perp} = U_K^T \tilde{\boldsymbol{f}}'_{\perp}, \quad \text{and} \quad \hat{\boldsymbol{B}}_{\perp} = U_K^T \tilde{\boldsymbol{B}}'_{\perp}. \tag{117}$$

As displayed in Fig. ..., the kernel contains many flat directions due to the FK tables. We can then decompose the vectors as follows

$$\frac{d}{dt}\begin{pmatrix} \hat{\boldsymbol{f}}_{\perp\perp,t} \\ \hat{\boldsymbol{f}}_{\perp\parallel,t} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{B}}_{\perp\perp} \\ \hat{\boldsymbol{B}}_{\perp\parallel} \end{pmatrix} - \begin{pmatrix} \Sigma_K & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{f}}_{\perp\perp,t} \\ \hat{\boldsymbol{f}}_{\perp\parallel,t} \end{pmatrix} \tag{118}$$

yielding the two independent equations

$$\frac{d}{dt}\hat{\boldsymbol{f}}_{\perp\perp,t} = \hat{\boldsymbol{B}}_{\perp\perp} - \Sigma_K \hat{\boldsymbol{f}}_{\perp\perp,t} \tag{119}$$

$$\frac{d}{dt}\hat{\boldsymbol{f}}_{\perp\parallel,t} = 0. \tag{120}$$

Once inverted, the first equation provides the evolution of the components of the vector orthogonal to the null space of both NTK and FK tables. To retrieve the original vector in flavour basis, we need to apply to the solution for $\hat{\boldsymbol{f}}_{\perp\perp,t}$ the inverse rotations applied so far. This gives

$$\boldsymbol{f}_t = [U_\Theta]_\perp \, L'_\perp U_K \hat{\boldsymbol{f}}_{\perp\perp,t}, \tag{121}$$

where $[U_\Theta]_\perp$ is the subspace of the eigenvectors of the NTK that is orthogonal to its kernel.

Properties of the symmetric kernel

The kernel $\tilde{K}$ ...

Obsolete Material – to be deleted, later

We then project the flow equation for f_t, Eq. 50, in this basis

$$\frac{d}{dt} f_{i,t} = b_i - \lambda_i M_{ij} f_{j,t} \,, \tag{122}$$

where

$$f_{i,t} = (v_i, f_t) \,, \tag{123}$$
$$b_i = (v_i, \Theta(\mathrm{FK})^T C_Y^{-1} y) \tag{124}$$
$$M_{ij} = (v_i, (\mathrm{FK})^T C_Y^{-1} (\mathrm{FK}) v_j) \,. \tag{125}$$

Note that M is symmetric and positive definite. We can write $M = RD^{1/2}D^{1/2}R^T$ and repeat the derivation we did for the ϵ.

To be continued...

At order $O(1)$ in an expansion in $1/n$, where n is the width of the NN, Θ is constant during training, and the flow equation can be integrated analytically,

$$f_t = M^{-1} (\mathrm{FK})^T \left[\left(1 - e^{-\hat{\Theta}_Y t} \right) y + e^{-\hat{\Theta}_Y t} (\mathrm{FK}) f_0 \right] . \tag{126}$$

where

$$\Theta_Y = (\mathrm{FK})^T \Theta \, (\mathrm{FK}) \,, \tag{127}$$
$$\hat{\Theta}_Y = \Theta_Y C_Y^{-1} \,, \tag{128}$$
$$M = (\mathrm{FK})^T (\mathrm{FK}) \,. \tag{129}$$

Note that, if (FK) were invertible, then for $t \to \infty$

$$(\mathrm{FK}) f_\infty = (\mathrm{FK}) M^{-1} (\mathrm{FK})^T y = y \,, \tag{130}$$

and the theoretical predictions go exactly through the points. However this is not true in the real life scenario where (FK) is *not* invertible. It is therefore interesting to compute the matrix (in data space)

$$(\mathrm{FK}) M^{-1} (\mathrm{FK})^T \,, \tag{131}$$

in order to understand how close the training can get to the training points.

For points f^* that do not enter in the computation of the theory prediction, the flow equation is

$$\frac{d}{dt} f_t^* = \Theta_t^* \, (\mathrm{FK})^T C_Y^{-1} \epsilon_t \,, \tag{132}$$

where

$$\Theta_t^* = \sum_{\mu,\nu} \lambda_{\mu\nu} \left(\nabla_\mu f_t^*\right) \left(\nabla_\nu f_t\right)^T \,. \tag{133}$$

The solution to this flow equation is

$$f_t^* = \Theta^* \, \Theta^{-1} \, f_t \,. \tag{134}$$

Need to check the constant term so that the boundary condition at $t = 0$ is correct.

References

[1] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *CoRR*, abs/1806.07572, 2018.

A Null space of FK

The FK tables can be regarded as a linear map from the space of PDFs to the space of data:

$$(\mathrm{FK}) : \mathbb{R}^{\mathrm{PDF}} \to \mathbb{R}^{N_{\mathrm{dat}}} . \tag{135}$$

Note that the matrix corresponding to this linear map is not square, and hence $(\mathrm{FK}) \neq (\mathrm{FK})^T$. We can define the null space of the FK tables

$$\ker(\mathrm{FK}) \equiv K_{(\mathrm{FK})} = \left\{ f \in \mathbb{R}^{\mathrm{PDF}} \ : \ (\mathrm{FK})\, f = 0 \right\} , \tag{136}$$

together with its orthogonal space

$$R_{(\mathrm{FK})} \equiv K^{\perp}_{(\mathrm{FK})} = \left\{ f \in \mathbb{R}^{\mathrm{PDF}} \ : \ f \cdot f_K = 0, \ \ \forall f_K \in K_{(\mathrm{FK})} \right\} . \tag{137}$$

Note that

$$(\mathrm{FK}) f = 0 \Longrightarrow Mf = 0 . \tag{138}$$

The converse is also true,

$$\begin{aligned} Mf = 0 &\Longrightarrow (f, Mf) = 0 && (139)\\ &\Longrightarrow ((\mathrm{FK})f, C_Y^{-1}(\mathrm{FK})f) = 0 && (140)\\ &\Longrightarrow (\mathrm{FK})f = 0 , && (141) \end{aligned}$$

where the last inequality follows from the fact that $C_Y > 0$.

For each of these two subspaces we can define a basis

$$\mathcal{B}_K = \left\{ \boldsymbol{u}_K^{(i)} , \ \ i = 1, \ldots, \dim K_{(\mathrm{FK})} \right\} , \tag{142}$$

$$\mathcal{B}_\perp = \left\{ \boldsymbol{u}_\perp^{(i)} , \ \ i = 1, \ldots, \dim R_{(\mathrm{FK})} \right\} . \tag{143}$$

Remember that $K_{(\mathrm{FK})} \bigoplus R_{(\mathrm{FK})} = \mathbb{R}^{\mathrm{PDF}}$ and thus the basis $\mathcal{B} = \mathcal{B}_K \bigoplus \mathcal{B}_\perp$ is a basis for $\mathbb{R}^{\mathrm{PDF}}$. Henceforth, when decomposing a vector in $\mathbb{R}^{\mathrm{PDF}}$, I will use the ordering $\{\mathcal{B}_K, \mathcal{B}_\perp\}$. Hence, given $\boldsymbol{f} \in \mathbb{R}^{\mathrm{PDF}}$, we can write

$$\boldsymbol{f} = \boldsymbol{f}_K + \boldsymbol{f}_\perp = \begin{pmatrix} f_K \\ f_\perp \end{pmatrix} . \tag{144}$$

Finally, note that (FK) is not symmetric (not even square). Thus, the right null space is not the same as the left null space, in particular

$$(\mathrm{FK})\boldsymbol{u}_K = 0 \ \not\Rightarrow \ \boldsymbol{u}_K^T(\mathrm{FK}) = 0 . \tag{145}$$

With the two bases in eqs. (143), we can decompose the (FK) as follows

$$(\mathrm{FK}) = \begin{pmatrix} (\mathrm{FK})_{KK} & (\mathrm{FK})_{K\perp} \\ (\mathrm{FK})_{\perp K} & (\mathrm{FK})_{\perp\perp} \end{pmatrix} , \tag{146}$$

where

$$(\mathrm{FK})_{B_1 B_2} = \sum_{i=1}^{\dim \mathcal{B}_1} \sum_{j=1}^{\dim \mathcal{B}_2} \langle u_{\mathcal{B}_1}^{(i)}| \, (\mathrm{FK}) \, |u_{\mathcal{B}_2}^{(j)}\rangle \qquad \mathcal{B}_1, \mathcal{B}_2 = K_{(\mathrm{FK})}, R_{(\mathrm{FK})} \, . \tag{147}$$

By definition of the kernel, we also have

$$(\mathrm{FK})_{KK} = (\mathrm{FK})_{\perp K} = 0 \, , \tag{148}$$

while $(\mathrm{FK})_{K\perp}$ would be zero only if (FK) was diagonal. Thus, in the basis $\mathcal{B}$, the FK tables can be expressed as

$$(\mathrm{FK}) = \begin{pmatrix} 0 & (\mathrm{FK})_{K\perp} \\ 0 & (\mathrm{FK})_{\perp\perp} \end{pmatrix} . \tag{149}$$

The product with a vector $\boldsymbol{f} \in \mathbb{R}^{\mathrm{PDF}}$ becomes

$$(\mathrm{FK})\boldsymbol{f} = \begin{pmatrix} 0 & (\mathrm{FK})_{K\perp} \\ 0 & (\mathrm{FK})_{\perp\perp} \end{pmatrix} \begin{pmatrix} f_K \\ f_\perp \end{pmatrix} = \Big((\mathrm{FK})_{K\perp} + (\mathrm{FK})_{\perp\perp} \Big) f_\perp \, . \tag{150}$$

The matrix M, eq. (46), can also be decomposed as in Eq. (147). However, now $M = M^T$ and thus only one component is non-zero:

$$M = \begin{pmatrix} 0 & 0 \\ 0 & M_{\perp\perp} \end{pmatrix} . \tag{151}$$

We can decompose the other elements in eq. (52) in a similar vein:

$$b = \begin{pmatrix} b_K \\ b_\perp \end{pmatrix} , \quad \Theta = \begin{pmatrix} \Theta_{KK} & \Theta_{K\perp} \\ \Theta_{\perp K} & \Theta_{\perp\perp} \end{pmatrix} , \tag{152}$$

where $\Theta \in \mathbb{R}^{\mathrm{PDF}\times\mathrm{PDF}}$ and can thus be projected in $\mathcal{B}$. Owing to this decomposition, we can now split eq. (52) into the two components of f_t, so that we are left with two coupled linear differential equations:

$$\begin{cases} \frac{d}{dt} f_\perp = b_\perp - \Theta_{\perp\perp} M_{\perp\perp} f_\perp \, , \\ \frac{d}{dt} f_K = b_K - \Theta_{K\perp} M_{\perp\perp} f_\perp \, , \end{cases} \tag{153}$$

where the dependence on the training time t is left implicit. From eq. (153) we see that f_K does *not* evolve linearly, but depends on the evolution of $f_\perp$. Moreover, the differential equation for $f_\perp$ is diagonal, and can be solved as shown in the main text, namely diagonalising $M_{\perp\perp}$. Note that, in the orthogonal subspace, $M_{\perp\perp}$ is positive-definite as we have projected out the null eigenvalues.

B Numerical validation of eq. (67)

In eq. (67) we showed that the solution for ε_∞ in the limit of infinite training length corresponds to the zero mode of the evolution matrix in data space $\tilde{H}_\varepsilon$. Here we provide the numerical argument that supports this result.

We start from the matrix M which, roughly speaking, is the contraction of two FK tables. Since FK tables contain many zeros, the matrix M will be sparse. The matrix M is displayed element-wise in Fig. 5. Each block identifies a flavour, and the length of the side of the block is equal to the size of the grid. The matrix M is decomposed using the singular

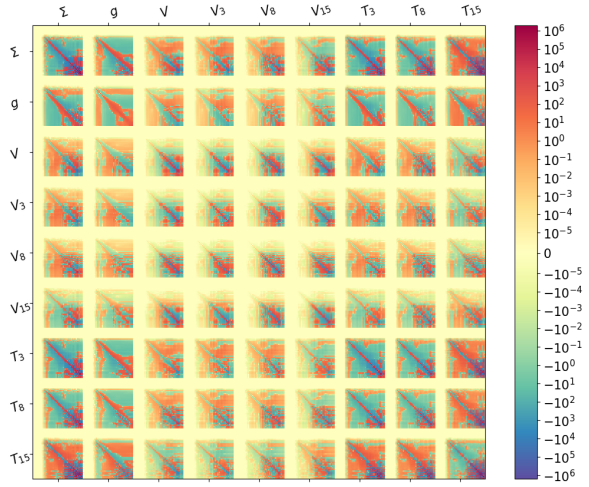

Figure 5: Matrix M in logarithmic scale.

value decomposition (SVD) method

$$M = U\Sigma V^T\,, \tag{154}$$

where U and V are orthogonal matrices, and Σ is a diagonal matrix containing the singular values sorted in decreasing order. Recall that the matrix M is also symmetric by construction, and this guarantees the existence of the solution of the associated eigensystem. Singular values and the absolute eigenvalues are displayed in Fig. 6. Here we also show the relative machine precision error, defined as the product of the largest singular value with the machine precision for double precision floating numbers ($\epsilon \sim 2.2 \times 10^{-16}$). Since the largest singular value is of order 10^6, the relative machine precision error is $\epsilon_{\rm rel} \sim 1.2 \times 10^{-9}$. The relevance of this number will be clear in the following observation.

As it can be seen from the figure, the singular values and the eigenvalues overlap as long as the relative numerical precision is not reached. Once this numerical threshold is exceeded, the two sets of values start to diverge. This is a consequence of the fact that $\epsilon_{\rm rel}$

defines the smallest double floating number that machine precision allows us to distinguish. Everything below this value is noise and should be discarded. This prompts us to apply an *ad hoc* regularisation to Σ, whereby the values below $\epsilon_{\rm rel}$ are set to zero. Note that we have

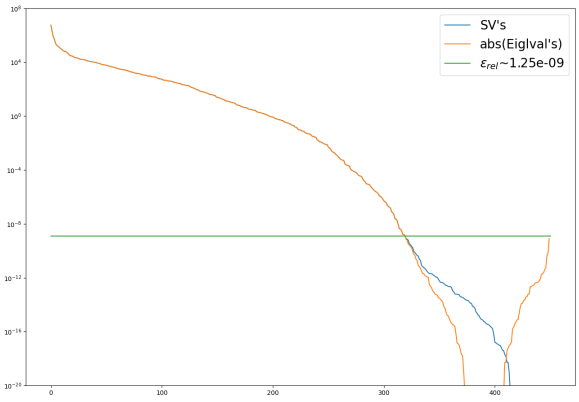

Figure 6: Singular values and (absolute) eigenvalues of the matrix M plotted in logarithmic scale. The green horizontal line marks the relative machine precision error, as explained in the main text.

also checked that the singular vectors of U and V are the same if they are associated to the singular values above the numerical threshold. In other words, in the space orthogonal to $\mathrm{Ker}(M)$, the SVD matches the eigendecomposition.

In principle, the singular vectors of the non-zero values span the space orthogonal to $\mathrm{Ker}(M)$. In this subspace, the matrix M is diagonal and the entries are exactly the non-zero singular values. However, if we compute the condition number of M in this subspace, we find that it is remarkably large, $\kappa(M) \sim 10^{15}$. In other words, even in this subspace the matrix M is ill-defined and cannot be inverted. The solution to this problem is to further reduce the dimensionality of the orthogonal space, at the cost of introducing some arbitrariness into the problem. Specifically, the final solution will depend on the number of small singular values that we discard. For example, the solution at infinity, eq. (58), will be influenced by this cut-off, as shown in Fig. 7. In this figure, eq. (58) is first computed in the truncated orthogonal space; then the result is transformed back to the original flavour basis using the matrix U restricted to the vectors that span the (reduced) orthogonal space.

As a final check, we can verify that eq. (67) is numerically satisfied. As we have just mentioned, even ϵ_∞ will depend on the severity of the cut-off. We then expect different results as we modify the number of singular values that we discard. For instance, using the orthogonal subspace without any cut, eq. (67) evaluates to

$$\left|\tilde{H}_\varepsilon \varepsilon_\infty\right| \approx 0.28\,. \tag{155}$$

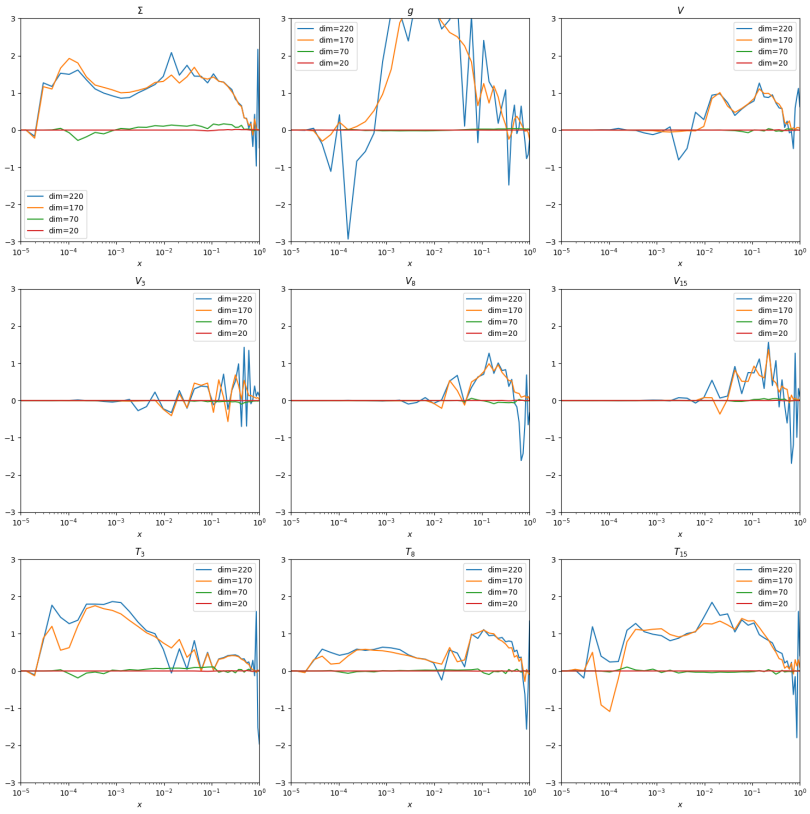

Figure 7:

On the other hand, if we reduce the size of the subspace to 170 (from 320), we get

$$\left|\tilde{H}_{\varepsilon}\varepsilon_{\infty}\right| \approx \ 1.33 \times 10^{-5}\,. \tag{156}$$