Parton Distributions from Neural Networks: Analytical Results

Amedeo Chiefa, Luigi Del Debbio, and Richard Kenway

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

May 26, 2025

Abstract

The determination of Parton Distribution Functions (PDFs) from experimental data is a central ingredient in the theoretical prediction of experimental results at hadronic colliders. As the LHC is now producing high-precision results, a robust determination of PDFs and their associated errors have become increasingly important. The NNPDF Colalaboration has pioneered the use of Machine Learning for the determination of PDFs. In this paper, we develop a theoretical formalism to analyse the training of Neural Networks in the context of PDF determinations.

1 Introduction

A first paragraph on precision physics at the LHC and the need for robust determinations of PDFs.

The extraction of PDFs from experimental data is a classic example of an inverse problem, namely the reconstruction of a function $f(x)$ from a finite set of data points Y_I, where the index $I = 1, \dots, N_{\mathrm{dat}}$. [1] In particular, for this study, we will focus on DIS data, which depend linearly on the function $f(x)$. The theoretical prediction for the data point Y_I is denoted

$$T_I[f] = \int dx\, C_{Ii}(x) f_i(x)\,, \tag{1}$$

where $C_{Ii}(x)$ is a coefficient function, i labels the parton flavor, and $f_i(x)$ is the PDF (or set of PDFs) that we want to determine.

Trying to determine a function f in an infinite dimensional space of solutions with a finite set of data leads to an ill-defined problem, whose solution will depend on assumptions made.

[1] When omitting the data index I, we will always assume $Y \in \mathbb{R}^{N_{\mathrm{dat}}}$.

In particular, the choice of a parametrization for f leads to a bias in the space of solutions that can be obtained. Together with the fit methodology, the parametrization also determines the propagation of the error on the data to the error on the fitted solution. Understanding the bias and the variance of the fitted PDF is therefore a major challenge for precision physics.

Following the ideas highlighted in Refs. [1, 2], we find it useful to introduce a Bayesian framework for this analysis. The function f is promoted to a stochastic process; for any grid of points x_α, $\alpha = 1, \dots, N_{\mathrm{grid}}$, the vector $f_\alpha = f(x_\alpha)$ is a vector of N_{grid} stochastic variables, for which we introduce a prior distribution $p(f)$. [2]

Any fitting procedure is interpreted as a recipe that yields the posterior distribution $\tilde{p}(f)$.

In this study, probability distributions are represented by ensembles of i.i.d. replicas. So, for instance, the prior distribution $p(f)$ is described by an ensemble

$$\left\{ f^{(k)} \in \mathbb{R}^{N_{\mathrm{grid}}}; k = 1, \dots, N_{\mathrm{rep}} \right\} , \tag{2}$$

drawn from the distribution p, so that

$$\mathbb{E}_p[O(f)] = \frac{1}{N_{\mathrm{rep}}} \sum_{k=1}^{N_{\mathrm{rep}}} O(f^{(k)}) , \tag{3}$$

for any observable O that is built from the PDFs.

The prior distribution $p(f)$ is defined by initializing a set of replicas using a Glorot-Normal initializer. The result of this initialization is discussed below in Sec. 2. For each replica, a new set of data $Y^{(k)}$ is generated from an N_{dat} dimensional Gaussian distribution centred at the experimental central value Y, with the covariance given by the experimental covariance matrix C_Y,

$$Y^{(k)} \sim \mathcal{N}\left(Y, C_Y\right) . \tag{4}$$

Each replica is $f^{(k)}$ trained on its corresponding data set $Y^{(k)}$. We denote the replicas at training time t, $f_t^{(k)} \in \mathbb{R}^{N}_{\mathrm{grid}}$. Stopping the training at time $\bar{t}$, the posterior probability distribution is represented by the set of replicas $\left\{ f_{\bar{t}}^{(k)} \right\}$, so that

$$\mathbb{E}_{\tilde{p}}[O(f)] = \frac{1}{N_{\mathrm{rep}}} \sum_{k=1}^{N_{\mathrm{rep}}} O\left(f_{\bar{t}}^{(k)} \right) . \tag{5}$$

All knowledge about the solution of the inverse problem, f, is encoded in the posterior $\tilde{p}$ and is expressed as expectation values of observables O using Eq. (5).

[2] Following the same convention used for the data, when omitting the grid index α, we will always refer to a vector $f \in \mathbb{R}^{N_{\mathrm{grid}}}$.

2 Neural Networks at Initialization

As detailed in Ref. [3], the NNs used for the NNPDF fit have a 2-25-20-8 architecture, a tanh activation function and are initialized using a Glorot normal distribution [4]. The preactivation function of a neuron is denoted as $\phi^{(\ell)}_{i,\alpha} = \phi^{(\ell)}_i(x_\alpha)$, where ℓ denotes the layer of the neuron, i identifies the neuron within the layer [3], and x_α is a point in the interval $[0,1]$. A grid of $N_{\rm grid} = 50$ points is used to compute observables in the NNPDF formalism and we will focus on those values of x_α here. For completeness, we list the values of x_α in Tab. 1.

α	x_α	α	x_α	α	x_α	α	x_α	α	x_α
1	2.00×10^{-7}	11	1.29×10^{-5}	21	8.31×10^{-4}	31	0.0434	41	0.422
2	3.03×10^{-7}	12	1.96×10^{-5}	22	1.26×10^{-3}	32	0.0605	42	0.480
3	4.60×10^{-7}	13	2.97×10^{-5}	23	1.90×10^{-3}	33	0.0823	43	0.540
4	6.98×10^{-7}	14	4.51×10^{-5}	24	2.87×10^{-3}	34	0.109	44	0.601
5	1.06×10^{-6}	15	6.84×10^{-5}	25	4.33×10^{-3}	35	0.141	45	0.665
6	1.61×10^{-6}	16	1.04×10^{-4}	26	6.50×10^{-3}	36	0.178	46	0.730
7	2.44×10^{-6}	17	1.57×10^{-4}	27	9.70×10^{-3}	37	0.220	47	0.796
8	3.70×10^{-6}	18	2.39×10^{-4}	28	0.0144	38	0.265	48	0.863
9	5.61×10^{-6}	19	3.62×10^{-4}	29	0.0211	39	0.314	49	0.931
10	8.52×10^{-6}	20	5.49×10^{-4}	30	0.0305	40	0.367	50	1.00

Table 1: Values of x_α used in the NNPDF grids for the computation of observables. The points are equally spaced on a logarithmic scale for $\alpha = 1, \dots, XXX$, and linearly spacing for $\alpha > XXX$. **Amedeo: Maybe we need to rethink the layout of this table...**

The output of the neuron identified by the pair (ℓ, i) is $\rho^{(\ell)}_{i,\alpha} = \tanh\left(\phi^{(\ell)}_{i,\alpha}\right)$. The parameters of the NN are the weights $w^{(\ell)ij}$ and the biases $b^{(\ell)}_i$, which are collectively denoted as θ_μ, where $\mu = 1, \dots, P$ and the total number of parameters is

$$P = \sum_{\ell=1}^{L} \left(n_\ell n_{\ell-1} + n_\ell\right) \, . \tag{6}$$

The PDFs in the so-called evolution basis are parametrized by the preactivation functions of the output layer L, $f_i(x_\alpha) = A_i \phi^{(L)}_{i,\alpha}$, where $i = 1, \dots, 8$ labels the flavors. [4] The input layer is identified by $\ell = 0$ and the activation function for that specific layer is the identity, so that

$$\rho^{(0)}_{i,\alpha} = \phi^{(0)}_{i,\alpha} = x_{i,\alpha} = \begin{cases} x_\alpha \, , & \text{for } i = 1 \, ; \\ \log\left(x_\alpha\right) \, , & \text{for } i = 2 \, . \end{cases} \tag{7}$$

[3] We will refer to i as the *neuron* index.

[4] For simplicity, we ignore the preprocessing function $x^{-\alpha_i}(1-x)^{\beta_i}$ that is currently used in the NNPDF fits. While the preprocessing may be useful in speeding the training it does not affect the current discussion.

In the following we refer to the preactivation functions as *fields*.

The Glorot normal initialiser draws each weight and bias of the NN from independent Gaussian distributions, denoted p_w and p_b respectively, centred at zero and with variances rescaled by the number of nodes in adjacent layers,

$$\frac{C_w^{(\ell)}}{\sqrt{n_{\ell-1}+n_\ell}}\,, \quad \frac{C_b^{(\ell)}}{\sqrt{n_{\ell-1}+n_\ell}}\,. \tag{8}$$

The probability distribution of the NN parameters induces a probability distribution for the preactivations,

$$p\left(\phi^{(\ell)}\right) = \int \mathcal{D}w\, p_w(w)\, \mathcal{D}b\, p_b(b) \prod_{i,\alpha} \delta\left(\phi_{i\alpha}^{(\ell)} - \sum_j w_{ij}^{(\ell)} \rho\left(\phi_{j\alpha}^{(\ell-1)}\right) - b_i^{(\ell)}\right)\,. \tag{9}$$

Note that, here and in what follows, $p(\phi^{(\ell)})$ denotes the joint probability for all the components of $\phi^{(\ell)}$,

$$p\left(\phi^{(\ell)}\right) = p\left(\phi_{1,\alpha_1}^{(\ell)}, \phi_{2,\alpha_1}^{(\ell)}, \dots, \phi^{(\ell)} - n_\ell, \alpha_1, \dots, \phi_{n_\ell, N_{\mathrm{grid}}}^{(\ell)}\right)\,. \tag{10}$$

This duality between parameter-space and function-space provides a powerful framework to study the behaviour of an ensemble of NNs, and in particular the symmetry properties of the distribution $p(\phi^{(\ell)})$, see *e.g.* [5]. Working in parameter space, *i.e.* computing the expectation values of correlators of fields as integrals over the NN parameter, one can readily show that

$$\mathbb{E}\left[R_{i_1j_1}\phi_{j_1\alpha_1}^{(n_\ell)} \dots R_{i_nj_n}\phi_{j_n\alpha_n}^{(n_\ell)}\right] = \mathbb{E}\left[\phi_{i_1\alpha_1}^{(n_\ell)} \dots \phi_{i_n\alpha_n}^{(n_\ell)}\right]\,, \tag{11}$$

where R is an orthogonal matrix in $\mathrm{SO}(n_\ell)$. Eq.(11) implies that the probability distribution in Eq. (9) is also invariant under rotations, and therefore it can only be a function of $\mathrm{SO}(n_\ell)$ invariants. Therefore

$$p\left(\phi^{(n_\ell)}\right) = \frac{1}{Z^{(\ell)}} \exp\left(-S\left[\phi_{\alpha_1}^{(\ell)} \cdot \phi_{\alpha_2}^{(\ell)}\right]\right)\,, \tag{12}$$

where

$$\phi_{\alpha_1}^{(\ell)} \cdot \phi_{\alpha_2}^{(\ell)} = \sum_{i=1}^{n_\ell} \phi_{i\alpha_1}^{(\ell)} \phi_{i\alpha_2}^{(\ell)}\,. \tag{13}$$

The action can be expanded in powers of the invariant bilinear,

$$S\left[\phi_{\alpha_1}^{(\ell)} \cdot \phi_{\alpha_2}^{(\ell)}\right] = \frac{1}{2}\gamma_{\alpha_1\alpha_2}^{(\ell)} \phi_{\alpha_1}^{(\ell)} \cdot \phi_{\alpha_2}^{(\ell)} + \frac{1}{8n_{\ell-1}} \gamma_{\alpha_1\alpha_2,\alpha_3\alpha_4}^{(\ell)} \phi_{\alpha_1}^{(\ell)} \cdot \phi_{\alpha_2}^{(\ell)}\, \phi_{\alpha_3}^{(\ell)} \cdot \phi_{\alpha_4}^{(\ell)} + O(1/n_{\ell-1}^2)\,, \tag{14}$$

so that the probability distribution is fully determined by the couplings $\gamma^{(\ell)}$. In Eq. (14), we have factored out inverse powers of n_ℓ for each coupling. With this convention, and with the scaling of the parameters variances in Eq. (8), the couplings in the action are all $O(1)$ in the limit where $n_\ell \to \infty$. As a consequence, the probability distribution at initialization is a multidimensional Gaussian at leading order in $1/n_\ell$, with quartic corrections that are $O(1/n_\ell)$, while higher powers of the invariant bilinear are suppressed by higher powers of the width of the layer. This power counting defines an effective field theory, where deviations from Gaussianity can be computed in perturbation theory to any given order in $1/n_\ell$, see *e.g.* Ref. [6] for a detailed presentation of these ideas. While the actual calculations become rapidly cumbersome, the conceptual framework is straightforward.

At leading order, the second and fourth cumulant are respectively

$$\langle \phi^{(\ell)}_{i_1,\alpha_1} \phi^{(\ell)}_{i_2,\alpha_2} \rangle = \delta_{i_1 i_2} K^{(\ell)}_{\alpha_1\alpha_2} + O(1/n_{\ell-1})\,, \tag{15}$$

$$\langle \phi^{(\ell)}_{i_1,\alpha_1} \phi^{(\ell)}_{i_2,\alpha_2} \phi^{(\ell)}_{i_3,\alpha_3} \phi^{(\ell)}_{i_4,\alpha_4} \rangle_c = O(1/n_{\ell-1})\,, \tag{16}$$

where

$$K^{(\ell)}_{\alpha_1\alpha_2} = \left(\gamma^{(\ell)}\right)^{-1}_{\alpha_1\alpha_2}\,. \tag{17}$$

The "evolution" of the couplings as we go deep in the NN, *i.e.* the dependence of the couplings on ℓ, is governed by Renormalization Group (RG) equations, which preserve the power counting in powers of $1/n_\ell$. At leading order,

$$K^{(\ell+1)}_{\alpha_1\alpha_2} = C_b^{(\ell+1)} + C_w^{(\ell+1)} \frac{1}{n_\ell} \langle \vec{\rho}^{(\ell)}_{\alpha_1} \cdot \vec{\rho}^{(\ell)}_{\alpha_2} \rangle \Big|_{O(1)} \tag{18}$$

$$= C_b^{(\ell+1)} + C_w^{(\ell+1)} \frac{1}{n_\ell} \langle \vec{\rho}^{(\ell)}_{\alpha_1} \cdot \vec{\rho}^{(\ell)}_{\alpha_2} \rangle_{K^{(\ell)}}\,, \tag{19}$$

where

$$\frac{1}{n_\ell} \langle \vec{\rho}^{(\ell)}_{\alpha_1} \cdot \vec{\rho}^{(\ell)}_{\alpha_2} \rangle_{K^{(\ell)}} = \int \prod_\alpha d\phi_\alpha \, \frac{e^{-\frac{1}{2}\left(K^{(\ell)}\right)^{-1}_{\beta_1\beta_2} \phi_{\beta_1}\phi_{\beta_2}}}{\left|2\pi K^{(\ell)}\right|^{1/2}} \rho(\phi_{\alpha_1}) \rho(\phi_{\alpha_2})\,.$$

Eq. (19) can be solved for the NNPDF architecture leading to the covariance matrices for the output of the NNs displayed in Figs. 1 and 2.

As a consequence of the symmetry of the probability distribution, the mean value of the fields at initialization needs to vanish, while their variance at each point x_α is given by the diagonal matrix elements of $K^{(\ell)}$. The central value and the variance of the parametrized singlet (Σ) and gluon (g) at initialization are shown in Fig. 3 for an ensemble of $N_{\text{rep}} = 100$. The central value is computed as discussed above in Eq. (3),

$$\bar{f}_{i\alpha} = \bar{f}_i(x_\alpha) = \frac{1}{N_{\text{rep}}} \sum_{k=1}^{N_{\text{rep}}} f_i^{(k)}(x_\alpha)\,, \tag{20}$$

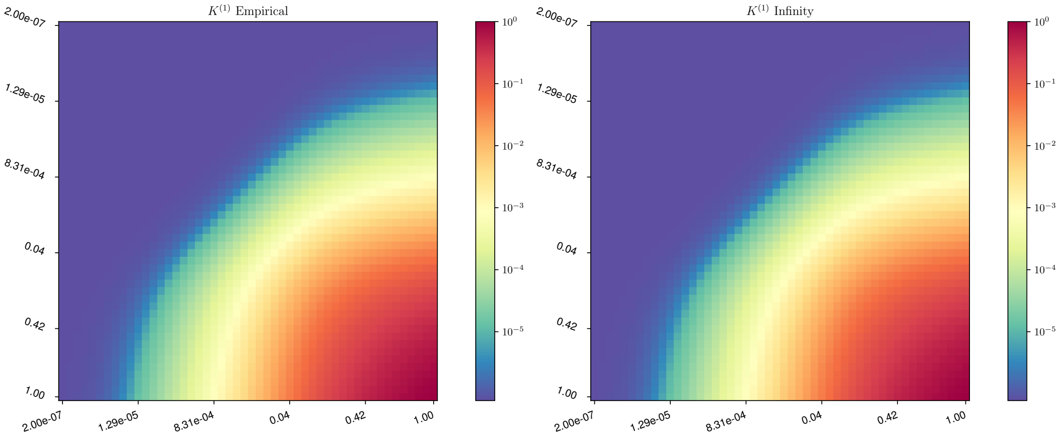

Figure 1: The empirical (left) and analytical (right) covariance matrices $K^{(1)}$ of the first layer of the NNPDF architecture. The covariance in the left panel is computed "bootstrapping" over an ensemble of 100 replicas, initialised using the Glorot normal distribution. The covariance in the right panel is obtained by solving Eq. (19) numerically.

and the variance $\sigma^2_{i\alpha}$ is computed using the same formula with

$$O(f) = \frac{N_{\mathrm{rep}}}{N_{\mathrm{rep}}-1} \left(f_i(x_\alpha) - \bar{f}_i(x_\alpha)\right)^2 \, . \tag{21}$$

Dependence on the architecture. Having analytical expressions for the variance at initialization allows us to investigate the impact of the NN architecture on the prior that is imposed on the PDFs. Iterating Eq. (19) yields the covariance at initialization for various depths. Do we get something interesting? worth mentioning?

We should also look at the recursion relations with other activations. Again, check whether we get something interesting...

3 Training

3.1 Gradient Flow

The parameter-space/function-space duality described in Sec. 2 yields intersting insight on the training process. Our main concern in this paper is understanding the dynamics driving the training process, therefore we work in a simplified setting where we consider standard gradient descent and data that depend linearly on the unknwon PDFs, as shown in Eq. (1). The generalization to other minimizers and non-linear data is left to future investigations,

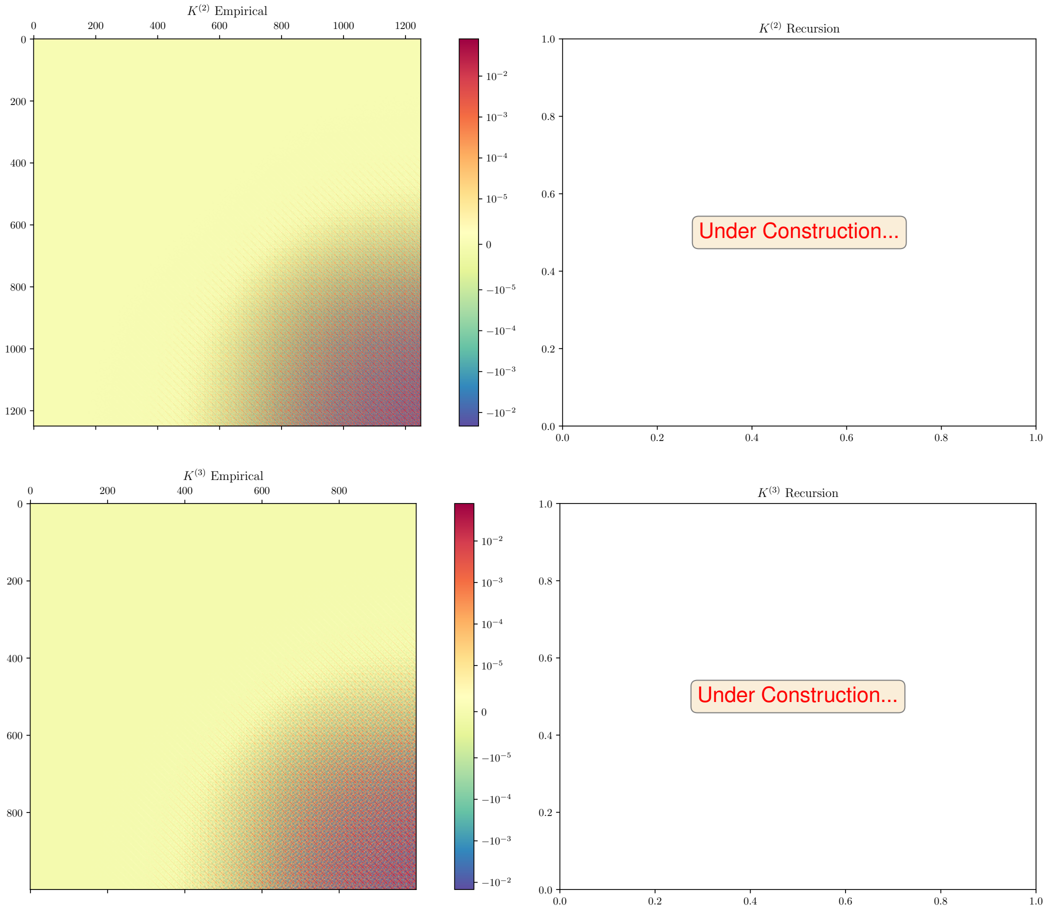

Figure 2: Same as Fig. 1, but for the second (top) and third (bottom) layers of the NNPDF architecture. **Amedeo: Here the four indices of the covariance $K_{i_1 i_2, \alpha_1 \alpha_2}$ are flattened into two indices for the sake of graphical representation. Maybe we should group the labels into groups of N_{grid} ticks on the axes.**

but is expected to yield qualitatively similar results. The results in this subsection apply to any generic parametrization of the unknown function, they are *not* specific to the case of using NN for the parametrization.

Gradient descent is described as a continuous flow of the parameters θ in training time t along the negative gradient of the loss function $\mathcal{L}$. The parameters and the fields during training are labelled by adding an index t, so that *e.g.* $\theta_{t,\mu}$ identifies the parameter θ_μ at

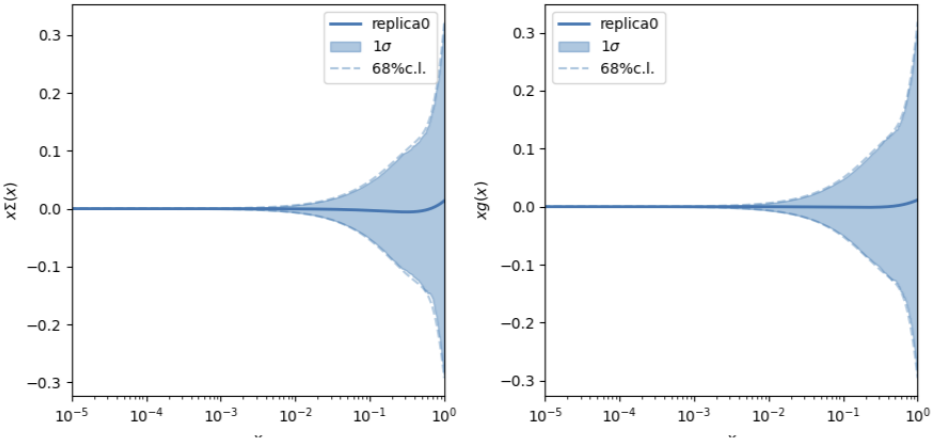

Figure 3: Parametrized singlet (left) and gluon (right) at initialization. The line labeled as Replica0 is the average of the functions over the ensemble of replicas. The blue 1σ band shows the variance of the ensemble of replicas, while the dotted line is the 68% CL. The agreement between these two quantities is a confirmation of the Gaussianity of the distribution of replicas.

time t. The gradient flow is given by

$$\frac{d}{dt}\theta_{t,\mu} = -\nabla_\mu \mathcal{L}_t \,. \tag{22}$$

We focus here on quadratic loss functions that are obtained as the negative logarithm of Gaussian data distributions around their theoretical predictions,

$$\mathcal{L}_t = \frac{1}{2}\left(Y - T[f_t]\right)^T C_Y^{-1}\left(Y - T[f_t]\right) \,, \tag{23}$$

where C_Y is the covariance of the data, which includes statistical and systematic errors given by the experiments and also any theoretical error, like *e.g.* missing higher orders in the theoretical predictions. Indices that are summed over are suppressed to improve the clarity of the equations. Note that the loss function at training time t is computed using the theoretical prediction $T[f_t]$, *i.e.* the result of Eq. (1) computed using the fields at training time t. For a quadratic loss, the gradient is

$$\nabla_\mu \mathcal{L}_t = -\left(\nabla_\mu f_t\right)^T \left(\frac{\partial T}{\partial f}\right)_t C_Y^{-1} \epsilon_t \,, \tag{24}$$

where, writing explicitly the data index,

$$\epsilon_{t,I} = Y_I - T_I[f_t] \,, \quad I = 1, \ldots, N_{\mathrm{dat}} \,. \tag{25}$$

For the specific case of a quadratic loss function, the gradient is proportional to ϵ_t, which is the difference between the theoretical prediction and the data at training time t. If at some point during the training the theoretical predictions reproduce all the data, the training process ends. A further simplification is obtained in the case of data that depend linearly on the unknown function f. In the specific case of NNPDF fits, the integrals in Eq. (1) are approximated by a Riemann sum over the grid of x points,

$$T_I[f] \approx \sum_{\alpha=1}^{N_{\rm grid}} (\mathrm{FK})_{I i\alpha} f_{i\alpha} \,, \tag{26}$$

and hence

$$\left(\frac{\partial T_I}{\partial f_{i\alpha}}\right)_t = (\mathrm{FK})_{I i\alpha} \,, \tag{27}$$

and is independent of t. The flow of parameters θ translates into a flow for the fields,

$$\frac{d}{dt} f_{t,i_1\alpha_1} = (\nabla_\mu f_{t,i_1\alpha_1}) \frac{d}{dt}\theta_\mu = \Theta_{t,i_1\alpha_1 i_2\alpha_2} (\mathrm{FK})^T_{i_2\alpha_2 I} \left(C_Y^{-1}\right)_{IJ} \epsilon_{t,J} \,, \tag{28}$$

where

$$\Theta_{t,i_1\alpha_1 i_2\alpha_2} = \sum_\mu \nabla_\mu f_{t,i_1\alpha_1} \nabla_\mu f_{t,i_2\alpha_2} \,. \tag{29}$$

For clarity, we often omit indices and write

$$\left(\frac{\partial T}{\partial f}\right)_t = (\mathrm{FK}) \,, \tag{30}$$

$$\Theta_t = (\nabla_\mu f_t)^T \, (\nabla_\mu f_t) \,, \tag{31}$$

$$\frac{d}{dt} f_t = \Theta_t (\mathrm{FK})^T C_Y^{-1} \epsilon_t \,. \tag{32}$$

Note that these equations do not refer to a specific parametrization and remain valid when some explicit functional form is chosen to parametrize the PDFs, as *e.g.* in Refs. [7, 8].

3.2 Lazy Training for the Flow Equation

The large-n_ℓ effective theory discussed in Sect. 2 also predicts that the NTK remains constant along training, up to corrections that are $O(1/n_\ell)$, see Ref. [9] and references therein for a derivation of this result. This regime is sometimes referred to as *lazy kernel training*, and allows an analytical solution of the flow equation.

We start by rewriting Eq. (32) as

$$\frac{d}{dt} f_t = -\Theta M f_t + b \,, \tag{33}$$

where

$$M = (\mathrm{FK})^T C_Y^{-1} (\mathrm{FK}) \,, \quad b = \Theta (\mathrm{FK})^T C_Y^{-1} Y \,. \tag{34}$$

The eigenvectors of Θ,

$$\Theta z^{(k)} = \lambda^{(k)} z^{(k)} \,, \tag{35}$$

provide a basis for expanding Eq. (33). It is necessary at this stage to distinguish the components of f_t that are in the kernel of Θ from the ones that are in the orthogonal complement, hence we introduce the notation

$$f^{\parallel}_{t,k} = \left(z^{(k)}, f_t\right) \,, \quad \text{if } \lambda_{(k)} = 0 \,, \tag{36}$$

$$f^{\perp}_{t,k} = \frac{1}{\sqrt{\lambda^{(k)}}} \left(z^{(k)}, f_t\right) \,, \quad \text{if } \lambda^{(k)} \neq 0 \,. \tag{37}$$

One can readily see that the components in the kernel of Θ, ker Θ, do not evolve during the flow,

$$\frac{d}{dt} f^{\parallel}_{t,k} = 0 \quad \Longrightarrow \quad f^{\parallel}_{t,k} = f^{\parallel}_{0,k} \,. \tag{38}$$

The flow equation for the orthogonal components can be written as

$$\frac{d}{dt} f^{\perp}_{t,k} = -H^{\perp}_{kk'} f^{\perp}_{t,k'} + B^{\perp}_{k} \,, \tag{39}$$

where the indices on quantities that have a $\perp$ suffix only span the space orthogonal to the kernel of Θ, while the indices on quantities that have a $\parallel$ suffix span the kernel. In Eq. (39), we introduced

$$H^{\perp}_{kk'} = \sqrt{\lambda^{(k)}} \left(z^{(k)}, M z^{(k')}\right) \sqrt{\lambda^{(k')}} \,, \tag{40}$$

$$B^{\perp}_{k} = -\sqrt{\lambda^{(k)}} \left[\left(z^{(k)}, M z^{(k')}\right) f^{\parallel}_{0,k'} - \left(z^{(k)}, (\mathrm{FK})^T C_Y^{-1} Y\right) \right] \,. \tag{41}$$

We refer to $H^{\perp}$ as the flow (or training) Hamiltonian; we see explicitly in the definition above that the flow dynamics is determined by a combination of the architecture of the NN, encoded in the NTK, and the data, on which M depends. More specifically, the matrix elements of M can be written as

$$\left(z^{(k)}, M z^{(k')}\right) = T^{(k)T} C_Y^{-1} T^{(k')} \,, \tag{42}$$

where $T^{(k)} = T[z^{(k)}]$ is the vector of theory predictions for the data obtained using $z^{(k)}$ as the input PDF. Denoting by $d^\perp$ the dimension of the subspace orthogonal to ker Θ, $H^\perp$ is a $d^\perp \times d^\perp$ symmetric matrix, whose eigenvalues and eigenvectors satisfy

$$H^\perp_{kk'} w^{(i)}_{k'} = h^{(i)} w^{(i)}_k \,. \tag{43}$$

The solution to Eq. (39) can be written as the sum of the solution of the homogeneous equation, $\hat{f}^\perp_{t,k}$, and a particular solution of the full equation. The solution of the homogeneous equation is

$$\hat{f}^\perp_{t,k} = \sum_{i=1}^{d^\perp} C_i e^{-h^{(i)}t} w^{(i)}_k \,, \tag{44}$$

where

$$C_i = \sum_{k=1}^{d_\perp} w^{(i)}_k f^\perp_{0,k} \,, \tag{45}$$

guarantees that the initial condition $\hat{f}^\perp_{t,k} = f^\perp_{0,k}$ is satisfied. Similarly, if we define

$$B_i = \sum_{k=1}^{d_\perp} w^{(i)}_k B^\perp_k \,, \tag{46}$$

then

$$\check{f}^\perp_{t,k} = {\sum_i}' \frac{1}{h^{(i)}} B_i \left(1 - e^{-h^{(i)}t}\right) w^{(i)}_k \,, \tag{47}$$

where the sum only involves the non-zero modes of $H^\perp$, is a particular solution of the inhomogeneous equation, which satisfies the boundary condition $\check{f}_{0,k} = 0$. Finally, the solution of the flow equation in the subspace orthogonal to ker Θ is

$$f^\perp_{t,k} = \hat{f}^\perp_{t,k} + \check{f}^\perp_{t,k} \,. \tag{48}$$

Collecting all terms yields a simple (and useful!) expression,

$$f_{t,\alpha} = U(t)_{\alpha\alpha'} f_{0,\alpha'} + V(t)_{\alpha I} Y_I \,. \tag{49}$$

The two evolution operators $U(t)$ and $V(t)$ have lengthy, yet explicit, expressions, which we summarise here or move to an appendix:

$$U(t)_{\alpha\alpha'} = \hat{U}^\perp(t)_{\alpha\alpha'} + \check{U}^\perp(t)_{\alpha\alpha'} + U^\parallel(t)_{\alpha\alpha'} \,, \tag{50}$$

where

$$\hat{U}^{\perp}(t)_{\alpha\alpha'} = \sum_i Z_\alpha^{(i)} e^{-h^{(i)}t} Z_{\alpha'}^{(i)}\,, \tag{51}$$

$$Z_\alpha^{(i)} = \sum_{k\in\perp} \sqrt{\lambda^{(k)}} z_\alpha^{(k)} w_k^{(i)}\,, \tag{52}$$

and

$$\check{U}^{\perp}(t)_{\alpha\alpha'} = \sum_i Z_\alpha^{(i)} \frac{1}{h^{(i)}} \left(1 - e^{-h^{(i)}t}\right) \tilde{Z}_{\alpha'}^{(i)}\,, \tag{53}$$

$$\tilde{Z}_\alpha^{(i)} = \sum_{k'\in\perp} \sum_{k''\in\parallel} w_{k'}^{(i)} \left(-\sqrt{\lambda^{(k')}}\right) T^{(k')T} C_Y^{-1} T^{(k'')} z_\alpha^{(k'')}\,, \tag{54}$$

and

$$U^{\parallel}_{\alpha\alpha'} = \sum_{k\in\parallel} z_\alpha^{(k)} z_{\alpha'}^{(k)}\,, \tag{55}$$

and

$$V(t)_{\alpha I} = \sum_i Z_\alpha^{(i)} \frac{1}{h^{(i)}} \left(1 - e^{-h^{(i)}t}\right) \tilde{T}_I^{(i)}\,, \tag{56}$$

and

$$\tilde{T}_I^{(i)} = \sum_{k'\in\perp} w_{k'}^{(i)} \left(\sqrt{\lambda^{(k')}}\right) T_J^{(k')} \left(C_Y^{-1}\right)_{JI}\,. \tag{57}$$

3.3 Behaviour of the solution

The solution in Eq. (49) is the main result of this section. It shows that the training process can be described as a linear transformation of the initial fields $f_{0,\alpha}$, which are the preactivations of the output layer at initialization, and a linear transformation of the data Y_I. The two transformations depend on the flow time t and are given by the evolution operators $U(t)$ and $V(t)$. Eq. (49) yields informations on both the central value and the variance of the trained fields.

The central values of the trained fields is obtained by taking the expectation value of Eq. (49) over the initial fields, which are Gaussian distributed at initialization,

$$\bar{f}_{t,\alpha} = \mathbb{E}\left[f_{t,\alpha}\right] = \mathbb{E}\left[U(t)_{\alpha\alpha'} f_{0,\alpha'}\right] + \mathbb{E}\left[V(t)_{\alpha I}\right] Y_I\,. \tag{58}$$

Taking the limit $t \to \infty$, we can check that the central value of the trained fields does indeed minimize the loss function, as expected. In this limit, the evolution operators $U(t)$ and $V(t)$ become

$$U(\infty)_{\alpha\alpha'} = \lim_{t\to\infty} U(t)_{\alpha\alpha'} = \sum_i Z_\alpha^{(i)} Z_{\alpha'}^{(i)} \,, \tag{59}$$

$$V(\infty)_{\alpha I} = \lim_{t\to\infty} V(t)_{\alpha I} = \sum_i Z_\alpha^{(i)} \tilde{T}_I^{(i)} \,. \tag{60}$$

Eq. (58) explicitly shows the contribution of each data point to the central value of the trained fields at each value of x_α. Note that the first term on the right-hand side, $\mathbb{E}\left[U(t)_{\alpha\alpha'} f_{0,\alpha'}\right]$ is entirely determined by the correlation between $U(t)_{\alpha\alpha'}$ and $f_{0,\alpha'}$; in the absence of correlations the expectation value would factorise and therefore vanish, since the expectation value of the initial fields vanishes.

4 Behaviour of the NTK

variation of the NTK

eigenvalues at initialization, eigenvalues at t after the NTK has stabilized, criteria to define the kernel

try a few variations of the architecture in order to get a feeling

eigenvalues and eigenvectors of $H^{\perp}$, dependence on the number of datapoints, on the choice of experiments

5 Training and Closure

level-0, level-1, level-2

analytical expressions should simplify

distance between the numerical minimization and the analytical formula as a function of training time t

6 First Training Results

results with real data

References

[1] Luigi Del Debbio, Tommaso Giani, and Michael Wilson. Bayesian approach to inverse problems: an application to NNPDF closure testing. *Eur. Phys. J. C*, 82(4):330, 2022.

[2] Alessandro Candido, Luigi Del Debbio, Tommaso Giani, and Giacomo Petrillo. Bayesian inference with Gaussian processes for the determination of parton distribution functions. *Eur. Phys. J. C*, 84(7):716, 2024.

[3] Richard D. Ball et al. The path to proton structure at 1% accuracy. *Eur. Phys. J. C*, 82(5):428, 2022.

[4] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[5] Anindita Maiti, Keegan Stoner, and James Halverson. Symmetry-via-Duality: Invariant Neural Network Densities from Parameter-Space Correlators. 6 2021.

[6] Daniel A. Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory.* Cambridge University Press, 5 2022.

[7] S. Bailey, T. Cridge, L. A. Harland-Lang, A. D. Martin, and R. S. Thorne. Parton distributions from LHC, HERA, Tevatron and fixed target data: MSHT20 PDFs. *Eur. Phys. J. C*, 81(4):341, 2021.

[8] Tie-Jiun Hou et al. New CTEQ global analysis of quantum chromodynamics with high-precision data from the LHC. *Phys. Rev. D*, 103(1):014013, 2021.

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