

PARTON DISTRIBUTION FUNCTIONS

solving the probabilistic inverse problem - or “How to learn a function”

Alessandro Candido & NNPDF

September, 2022

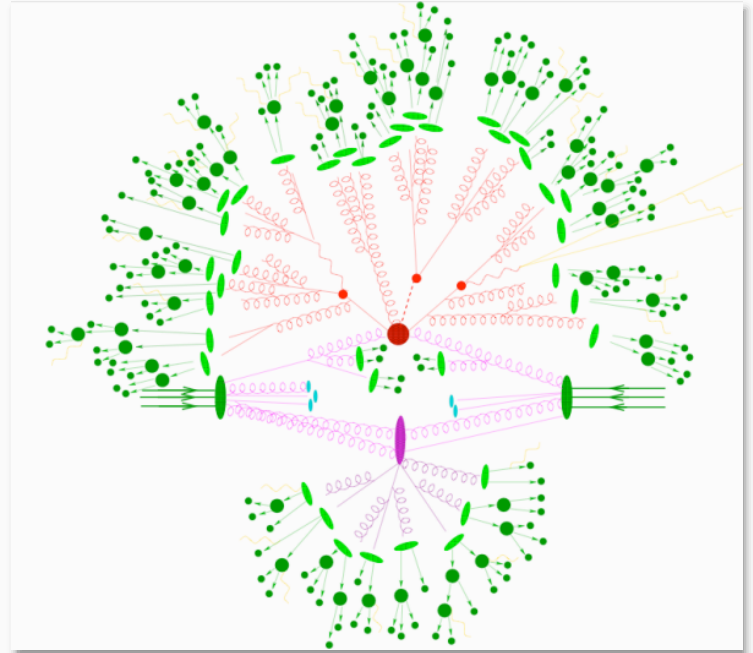
THE PHYSICS

In order to describe a *collision* a lot of ingredients are involved:

- a **hard scattering** matrix element (partonic level)
- **parton shower**, on the colored products
- color reconnection and **hadronization**

This is what is going on in a generic collider, but an *hadronic collider* needs one more ingredient:

- *partonic description* of the **hadronic initial state**



Since QCD is non-perturbative at low energies, there is something special about hadrons: we are almost *guessing in the wild*^a.

$$\begin{aligned}
 |\langle h | S | p \rangle|^2 &\approx \underbrace{\langle h | \mathcal{O}_j | h \rangle}_{\text{FF}} \underbrace{|\langle j | S | i \rangle|^2}_{\text{hard ME}} \underbrace{\langle p | \mathcal{O}_i | p \rangle}_{\text{PDF}}
 \end{aligned}$$

Figure 1: Scattering Matrix element, \mathcal{S} . Fragmentation Functions only relevant for non-outgoing inclusive processes.

The operator \mathcal{O}_i is essentially a number operator for parton i , with a Wilson line (time-like path) insertion for gauge covariance.

^aActually, this is a very very poor statement: we have constraints about theory symmetries, and the bound state net content. But what we **do not know** from theory is the **detailed structure**, and we need it to *predict experiments*.

Fortunately, there is also something very **special** about our **theory**, and the breakdown we described: they **factorize** (up to controlled sub-leading terms).

Then we are left with our unknown non-perturbative physics, but nicely packaged in some **descriptive functions**, the **Parton Distribution Functions** (PDFs).

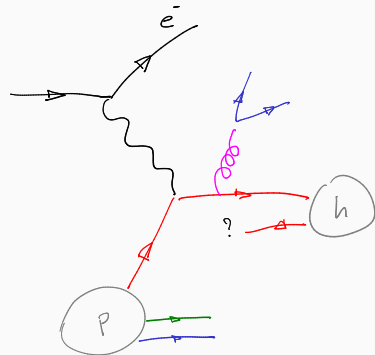
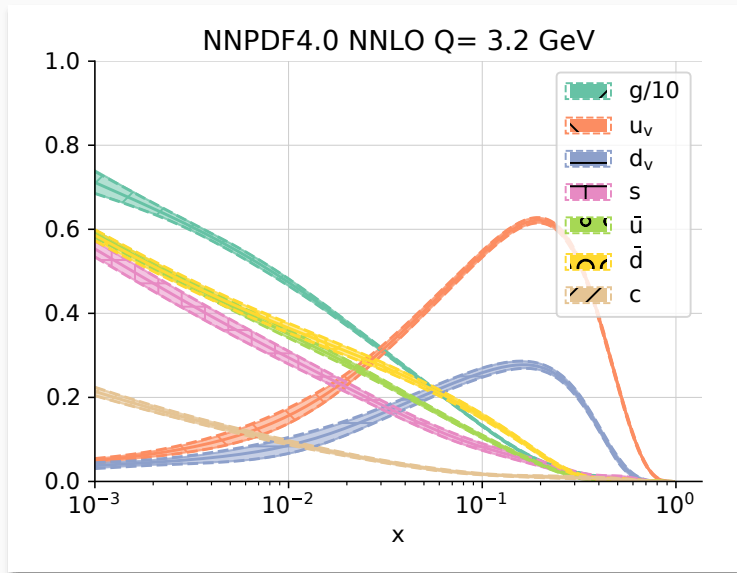


Figure 2: Scattering process with initial and final state hadrons.

PDFs then are just a set of **unknown functions**:

$$f_i : [0, 1] \rightarrow \mathbb{R}$$

There is one for each flavor i , and they obey some *theoretical constraints* (e.g. number and momentum sum rules, constraining the integral).



INVERSE PROBLEM

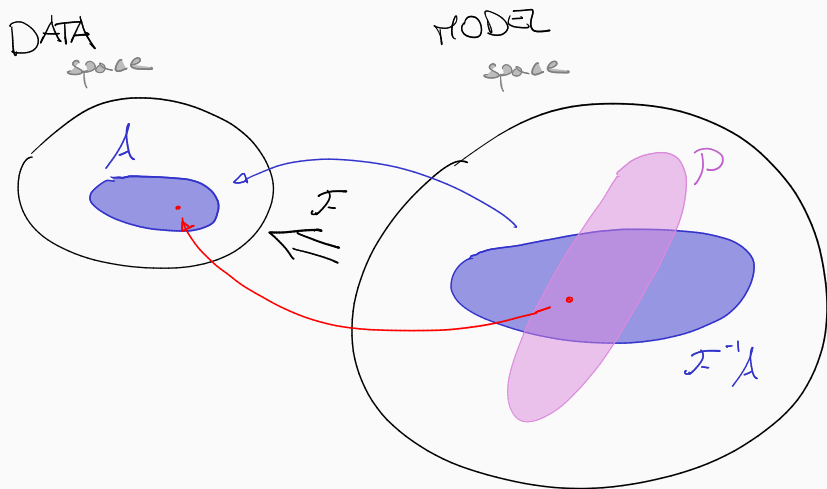


Figure 3: The map \mathcal{F} transform the model into data. \mathcal{P} is the parametrization space.

No direct observation of samples of unknown objects.

The observed objects are obtained from the unknown ones through a **singular transformation** (non-bijective)^a.

Our **inference** will then follow the opposite direction, that is why it is called the **inverse problem**.

^aTrivial, because function space is infinite-dimensional, while observed objects are finite (samples, and then even more "kinds").

So, **which is the map** from PDF (a.k.a. *the object to be determined*) to the experimental data?

Physics, i.e. **perturbative QCD**:

$$\sigma(x, Q^2) = \hat{\sigma}_{ij} \otimes f_i \otimes f_j = \int dz_1, dz_2 \hat{\sigma}(z_1, z_2, Q^2) f_i\left(\frac{x}{z_1}, Q^2\right) f_j\left(\frac{x}{z_2}, Q^2\right)$$

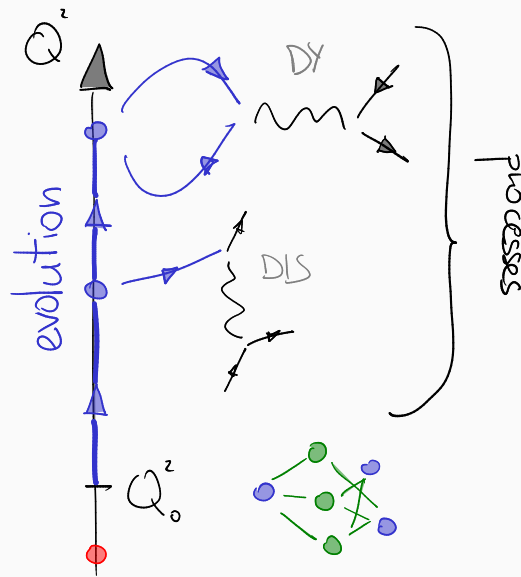
DGLAP EVOLUTION integro-differential equation, determining PDFs at all scales from a border condition at a given scale Q_0^2

HARD CROSS SECTION short distance physics, computed diagrammatically (possibly supplemented with resummation)

Since we can compute everything else^a, we are left with the determination of the **border condition** only:

$$f_i(x, Q_0^2)$$

^aFor inclusive final states.



Evolution and scattering, both computed in pQCD

EKO

Evolution Kernel Operators

DGLAP evolution framework



DIS predictions module

PineAPPL

General process, PDF-independent, grid storage

Since we want to go through minimization, it is crucial to have a **fast map** from *model space* to *data space*:

$$f_0(x) = f(x, Q_0^2) \rightarrow \sigma_a(x, Q^2) = \hat{\sigma}_a(Q^2) \otimes f(Q^2) \otimes f(Q^2)$$

We call such maps **Fast Kernel (FK) tables**:

$$\begin{cases} F_a = FK_{a,i\alpha} f_{0,i\alpha} \\ \sigma_b = FK_{b,j\beta k\gamma} f_{0,j\beta} f_{0,k\gamma} \end{cases}$$

since data are either *linear* (one initial proton) or *quadratic* (two protons collision) in the PDF.

Plus *composite observables* (e.g. ratios).

For people that are familiar with perturbation theory and PDF nomenclature:

when a PDF is told to be LO, NLO, NNLO, or N³LO, this is the *underlying theory order* (the **FkTable**).

A function $f: \mathbb{R} \rightarrow \mathbb{R}$ (or suitable intervals) lives in an infinite-dimensional space.

This has a simple consequence:

Under-determination

Fitting an **unknown function** on a finite number of data is always an **under-determined** problem.

How to choose a solution, when **many** are available and **equivalent**?

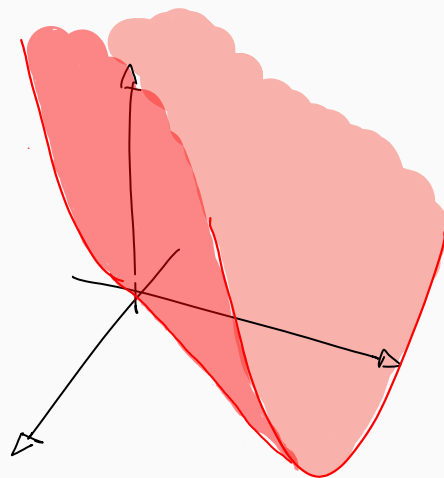


Figure 4: Possible χ^2 profile in 2D parameter space.

There are two main ways to attack the problem.

1. One consists in *reducing the number of parameters*, by **slicing** a suitable **hyperplane**.^a

This is what we do in PDFs when choosing a **fixed parametrization**: we decide which parameters to fit, and take a single given value for everything else.

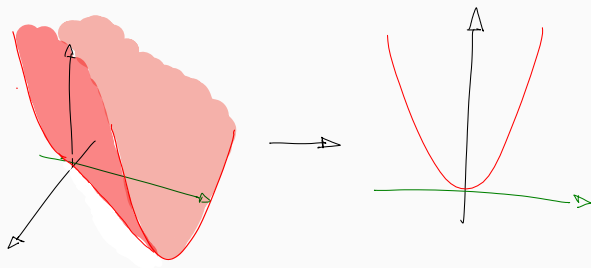


Figure 5: χ^2 profile in sliced parameter space.

2. The second approach removes the zero-direction by adding **regularization**.

This is what the **Neural Network** (and its training algorithm) is doing under the hood.

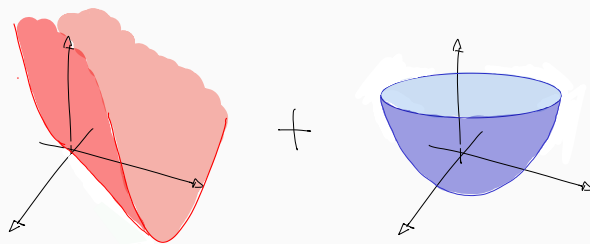
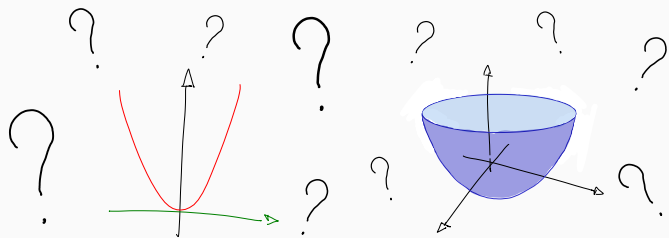


Figure 6: χ^2 profile plus regularization function.

^aNot containing zero-directions.

Then which one should we choose?



There is **not** an **absolute best answer**, because both procedures might be **arbitrary**.

In need for guiding principle to make a good choice, proposal:

Choices should encode physics assumptions

and as little arbitrariness as possible^a.

So what about PDFs?

CUTS cutting the space (fixed parametrization) is no bad, **if you know how to do it**: you need **precise theoretical insight** on the **PDF shape**

REGULARIZATION also this need as much motivation as the former, but it makes *to shift the focus* from the exact shape to more abstract **features**

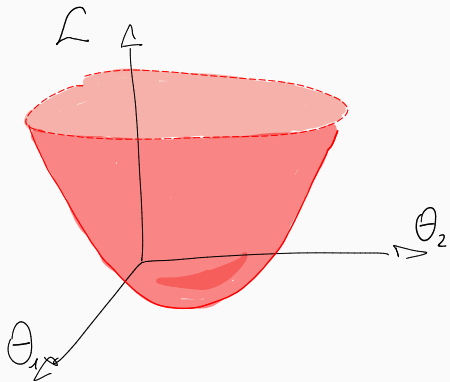
^aThough nothing is actually bias-free, so better to know and declare your bias, rather than hiding it.

NNPDF approach → use a **Neural Network** as the **representation** for the unknown function.

Notice that usually a NN is used to perform a task. Here the NN is actually representing the unknown object itself.

I.e., usually, evaluate on your input once, to obtain the candidate output. Here, many evaluation of the same network on an array of inputs needed to perform the same task.

UNCERTAINTIES = DISTRIBUTION



Usually the Hessian is used together with the χ^2 as loss.

The Hessian approach is possibly the simplest: essentially, relies on a **quadratic** approximation of the **loss function near the minimum** (*best fit*) \leftrightarrow **Gaussian** distribution **in parameter space**.

So, given a restricted set of parameters, the determination of the multi-Gaussian relies on:

- EIGENVECTORS** as eigenvectors of the loss itself
- VARIANCES** derived from inverse loss curvature

So *the distribution* resembles:

$$P(\theta|\mathcal{D}) = \exp\left(-\chi^2(\theta; \mathcal{D})\right)$$

NNPDF Approach: Monte Carlo sample (empirical distribution)

Naïve idea: **uncertainty comes from data**, let's propagate the data distribution back through the *regularized inverse function*:

$$f_i(x) = \arg \min_f (\mathcal{L}(D_i)) = \mathcal{NN}(x | D_i)$$

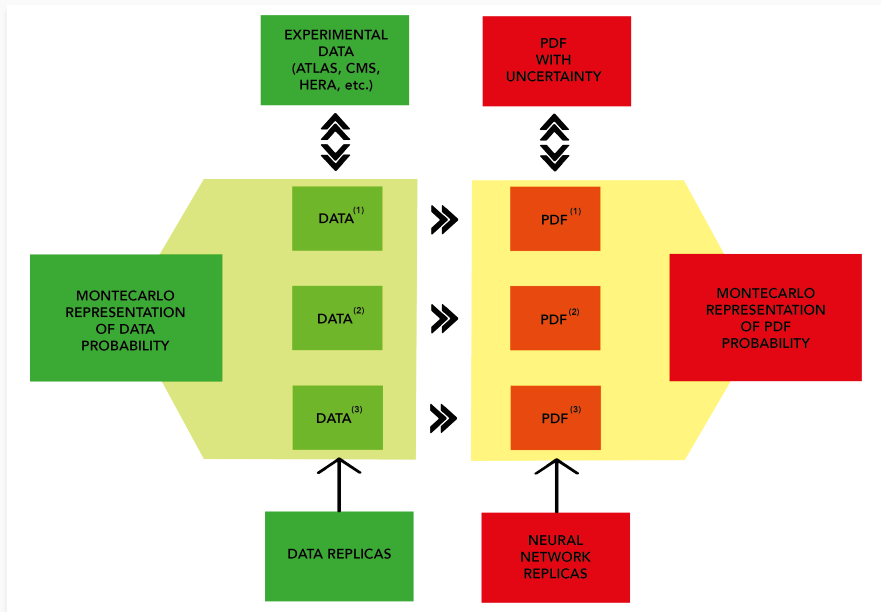
Loss contains the χ^2 , but χ^2 is not the end of the story.

Very simple (possibly expensive^a) usage: in order to obtain the **distribution of predictions**, apply your theoretical calculation to the element of the sample:

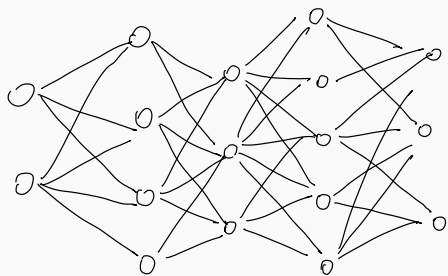
$$PredS = \{\mathcal{F}(f) \ \forall f \in PostS\}$$

^aTo address this compressed sets are distributed along the main release. There is also a program to obtain specialized minimal sets for dedicated studies:

<https://github.com/scarrazza/smpdf>



What is then doing the neural network (NN)? Why is it working so well?



Essentially, the NN is encoding an **interpolation hypothesis** in the architecture **and** training algorithm^a.

^aTraining-validation split, even though interpretation is far from clear.

While both the approaches are limiting *model complexity*, the **Neural Network** is free enough to follow strong **data trends**.

Physical Wiggles

A criticism we collected has been about NN *opaque assumptions* might **prevent** to follow **small physical fluctuations**, but it is actually the other way round:^a

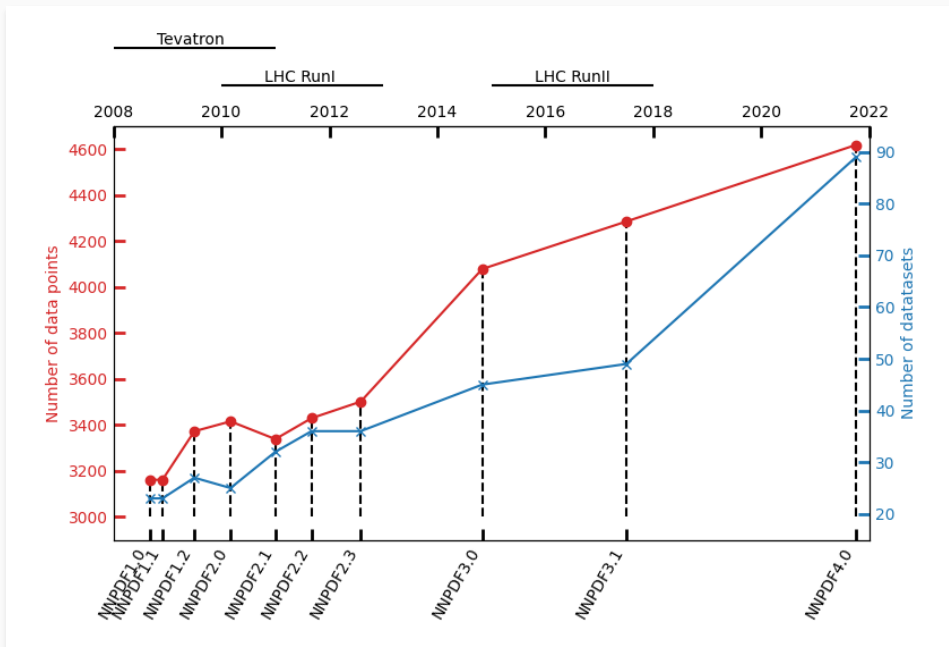
FIXED PARAM. in this case, you can only find oscillations you are allowing for, so you need to know in advance **each and every wiggle** to discriminate physical ones from noise

NN all wiggles are dewighted, but physical wiggles should be resolved more and more by data, so "*enough precision*" in the **data** will naturally **overwhelm the theoretical (learning) bias**

^aIn principle, yet to be proven in practice.

TRUST THE RESULTS

As physicists, we **trust in data** (physics \sim experimental science).



Variety of datasets, to cover different features of the *underlying object*.

What happens when you **do not trust** NN?

Common criticism is:

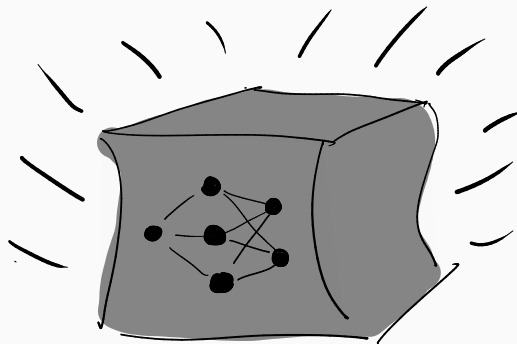
*the NN is a **black box***

Well... **no**.

We know a lot of things about a NN: the *structure*, the *task* is performing, and we can gather insights *inspecting* it and the *training process*.

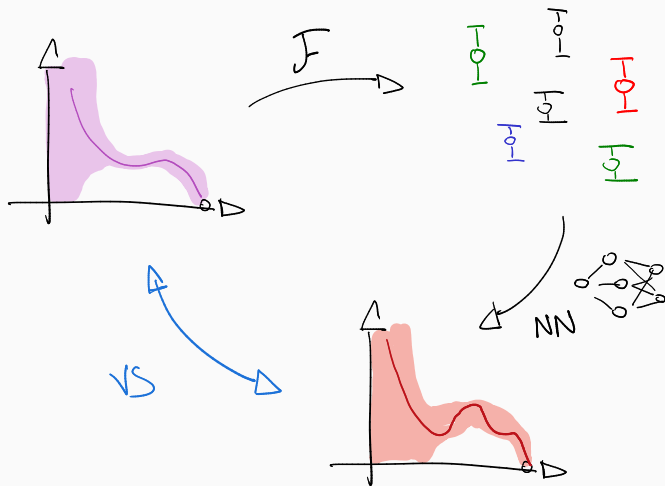
But skeptics are not wrong: what we are missing is a **full analytical insight** on the process. A **proof** of solving the specific task.

We can claim the algorithm to “work” (having a suitable definition for it...). But, even when we are right, we are often lacking the specific **why**.



If ~~analytically~~ there is no obvious way, let's go **empirically**.

Controls data region.



LEVEL 0 generate data from “true” central value only: $\chi^2_{expected} \sim 0$
 (essentially: can we find that central value? Is it forbidden?)

LEVEL 1 generate data from the “true” distribution $\chi^2_{expected} \sim 1$
 a “usual” distribution determination

LEVEL 2 Add pseudo-data generation, $\chi^2_{expected} \sim 2$
 intuitively because it “fluctuates twice”

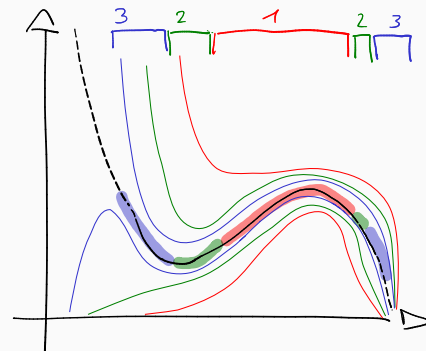
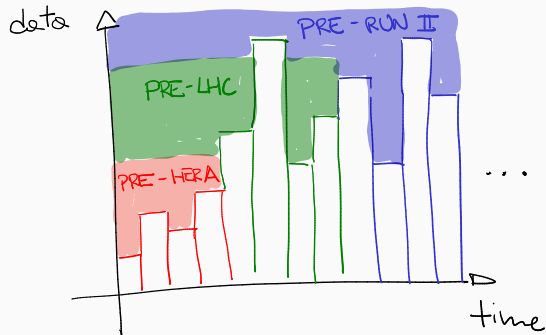
Too large errors and too small are both wrong. The test is not only for the central value (but full posterior).

(unless being able to prove to consistently over-estimate, without being able to compute it to subtract)

Experimental data are far from perfect, they might be affected by several inconsistencies.

→ model *inconsistencies* into the data generation process, a.k.a. “closure tests with inconsistent data”, WIP

Controls extrapolation region: what about data you have not seen yet?



I can not travel to the future, but I know **history!**

I can **divide** my dataset **chronologically** (in a possibly meaningful way), and fit all the incremental sets. Then:

*Compare **older extrapolation** region with **newer analogue**, where it becomes **data region***

It is not like a full blind analysis, realistically my prior knows about “the future”, but it is a further consistency check.

RESULTS

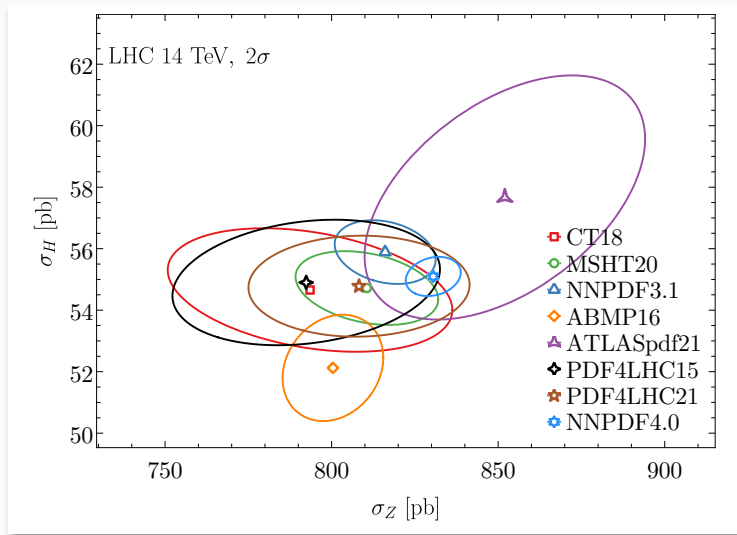
Here, the result of the “*Physical Wiggles*” issue is evident: in order to nicely fit result with a rather **restrictive parametrization**, you need to inflate your errors (*tolerance procedure*).

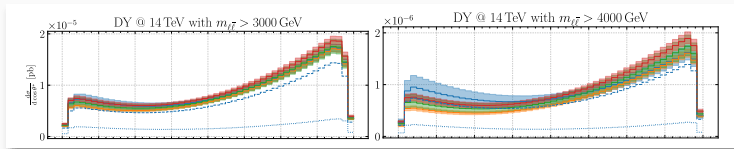
However, keep in mind that **NNPDF4.0** has a *wider datasets* than the other releases.

Implementing data has a non-trivial retrieval and interpretation cost (lack of standard language, especially for systematics), plus theory implementation (the **FkTable**).

But **NNPDF3.1** has already smaller uncertainties than more recent releases, e.g. **CT18** and **MSHT18**, despite a slightly smaller^a dataset. This is the **methodology impact**.

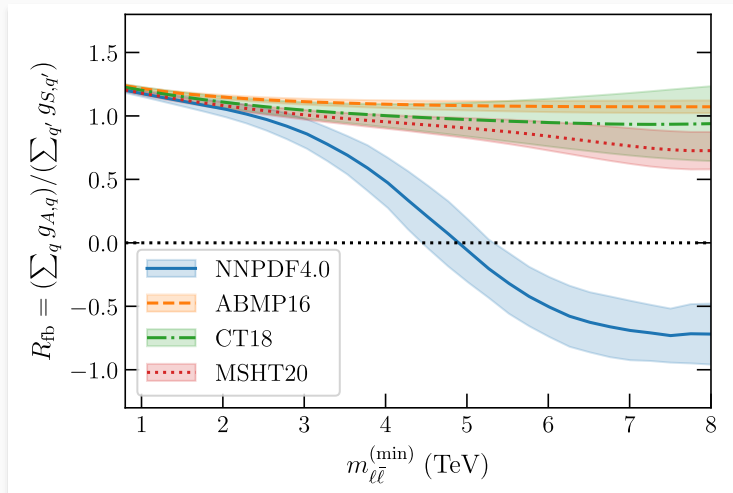
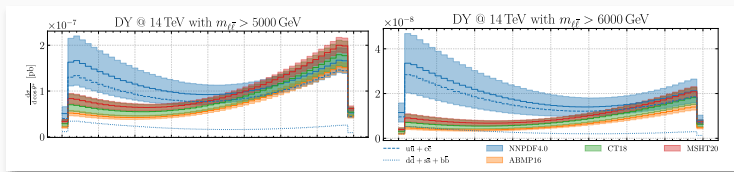
^aDisputable.





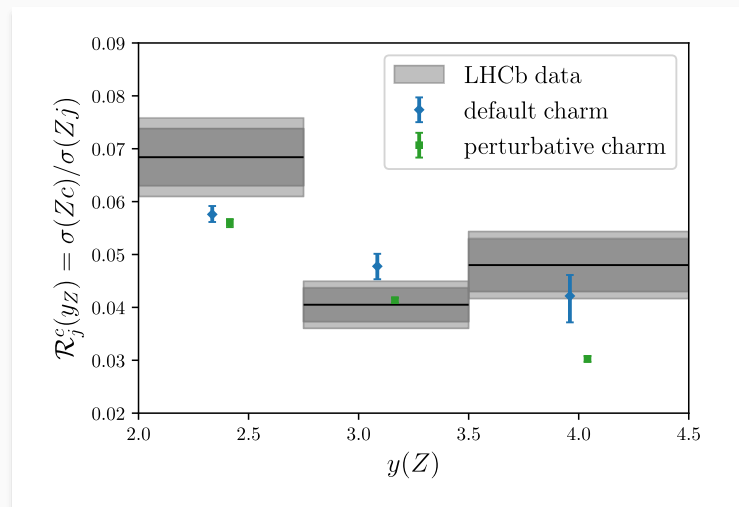
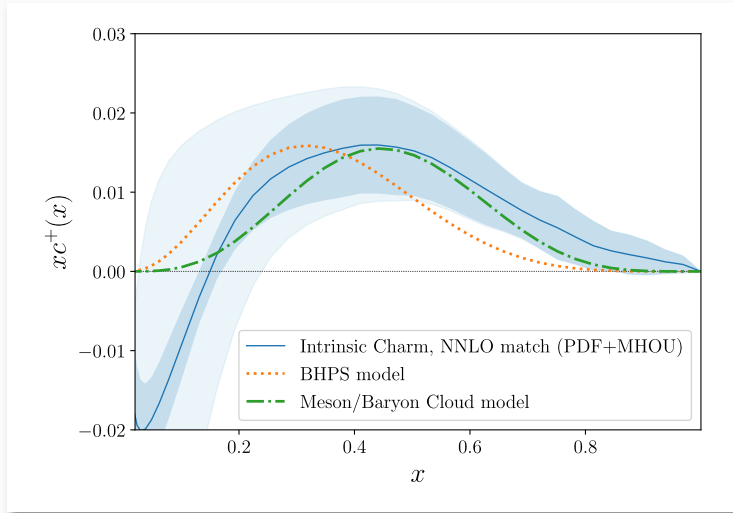
Rather biased result **encoded** in the **hypothesis**: the large invariant mass extrapolation region maps to the large- x behavior of the PDF, that is **little constrained by data**. But a fixed parametrization propagate the behavior of the data region.

While **NNPDF4.0** has very **small uncertainties** in the **data region**, it is still flexible enough to **inflate in extrapolation**, encoding the **lack of experimental knowledge**.



3σ of an intrinsic (non-perturbative) **charm** component of the proton.

 Nature 608 (2022) 7923, 483-487



CONCLUSIONS

Title question: “How to learn a function?”

1. Reduce to a **finite problem** through soft but **sensible assumptions** (e.g. interpolation)
2. **Minimize suitable loss** function in data space, including **extra constraints** (usually theoretical knowledge)
3. Apply an “*error propagation*” technique (**posterior determination/sampling**)
4. **Test** over and over your methodology (empirical robustness proof, i.e. consistency check)

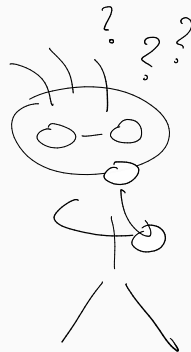
But: is a Neural Network really **the best tool** for this kind of tasks?

Next Episode (clue above ↑: *posterior sampling*)

In this way, it is possible to learn a lot!

But we actually have little to poor insight on “*why is my inference working?*”. And thus “*Is it actually working?*”

Good scientists are **skeptic** of their own **results**. But, we still need to **act to the best of our options**, while preserving **caution**.



Collaboration and references

For all info about the collaboration, cf:

<https://nnpdf.mi.infn.it>

including talks, papers and code.



A few more items at:

<https://n3pdf.mi.infn.it>



Public Code

Introduction paper:  Eur.Phys.J.C 81 (2021) 10, 958

Code and docs:

 <https://github.com/NNPDF/nnpdf>

<https://docs.nnpdf.science>



THANKS FOR YOUR ATTENTION!

NNPDF "framework" is much **more** than a Neural Network only:

- nice representation of a generic distribution (**replicas**)
- many **tests** in place
- **fast theory** framework
- **lots of data** implemented
 - both theory calculation and
 - commondata implementation with uncertainties (including correlated systematics)
- and more...

We do **not** want to **loose any** of these.

Another limitation* is that the current methodology need to make a **fit to one sample at a time**. The distribution is only the result of many fits.

What if we could fit the distribution all at once?

But sometimes we **lack insight**: we have tests for this, but they are not always a handy tool to answer all questions.

Hopscotch

What are our criticism to CT replicas?


Mostly we blame the method, but little criticism on the result.

Some **physical assumptions** have been checked:

- χ^2 vs $\chi_{t_0}^2$
- integrability and sum rules
- positivity

But it might be hidden in the NN **regularization**.

Attempt to reinterpret analytically the **NNPDF** approach.

 *Eur.Phys.J.C* 82 (2022) 4, 330, L. Del Debbio, T. Giani, M. Wilson, “Bayesian approach to inverse problems: an application to NNPDF closure testing”

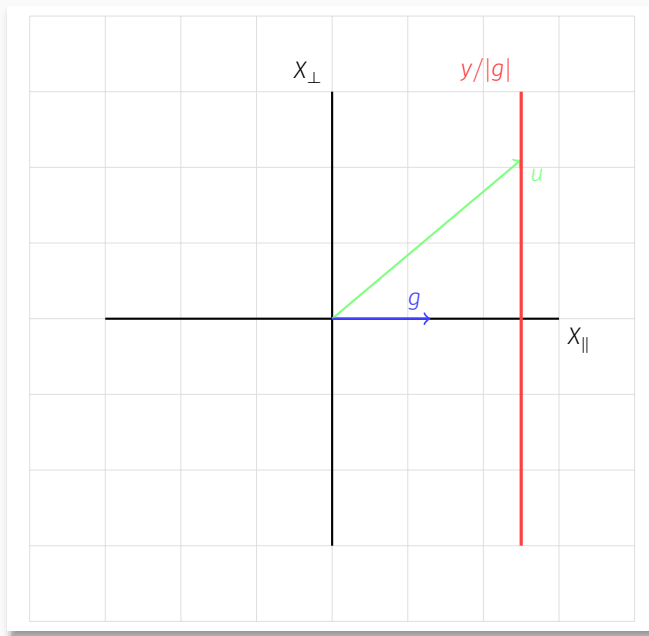
In the simplistic case of:

- **Gaussian** prior and likelihood *not so unrealistic*
- **linear** map to data space *not our problem*
- no further **regularization** on the minimizer (the NN) **definitely not** our problem

it is possible to prove that the **NNPDF** approach boils down to the equivalent **Bayesian posterior sampling**.



Analogous picture to the underdetermined loss landscape, without loss axis. u is the data point, g the theoretical map $\mathcal{F}(u) = g^T u$. The direction X_{\perp} is unconstrained.



It has a hint for the direction to move for a more explicit approach.

Typical examples of ML are:

- image and speech recognition
- generative tasks,
- style transfer
- and so on...

All these problems have in common:

very **high dimensional** objects, with **poor analytical/algorithmic insight** on its structure.

Working out an explicit and effective representation for them would be difficult.

Not the case for PDFs: **math language description** and **clear analytic properties** at hand (sum rules, power-like behavior, and so on...).

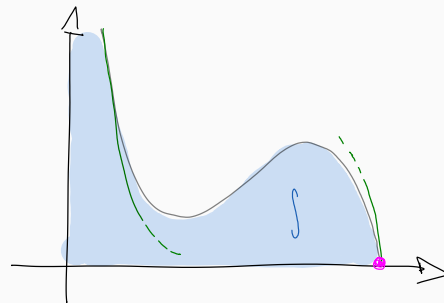


Figure 7: PDF analytical features.

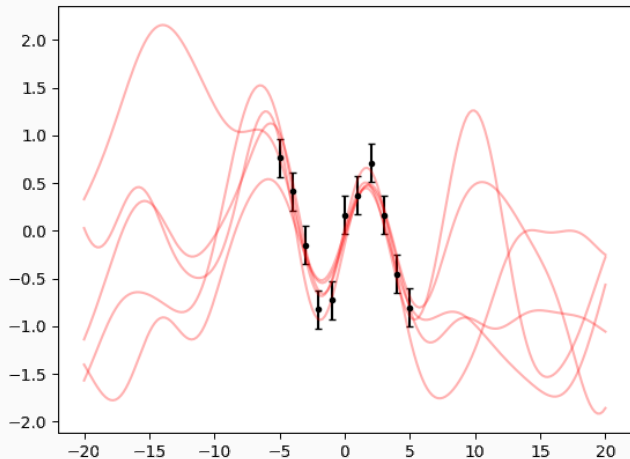
Then, we can follow a rather simple analytical approach:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Bayes formula exposes **regularization** explicitly in the prior, so the assumptions are analytically transparent.

But then the question: *which prior?*

A: a Gaussian process



Essentially a *multi-Gaussian* with a **metric-driven kernel**, with the motivation that is simple, and sufficiently flexible.

Basic ideas:

PARAMETRIZATION exactly our delivery: we use PDF values at grid points (we would not expose more degree of freedom anyhow, so no need to use them)

TRANSFORMATIONS data are not in the PDF space, but we can use linear and non-linear (quadratic) transformations

SUM RULES the Gaussian process allow us to impose them analytically (in practice, it is easier to impose them as *zero-error data*, but it is only a technicality)

- the important thing is that we can **constrain integral and derivatives** as much as the process, thanks to the *metrical kernel*

INTEGRABILITY integrability and extrapolation behavior it is implemented as constraints on hyper-parameters

POSITIVITY we can implement as constraint on the process

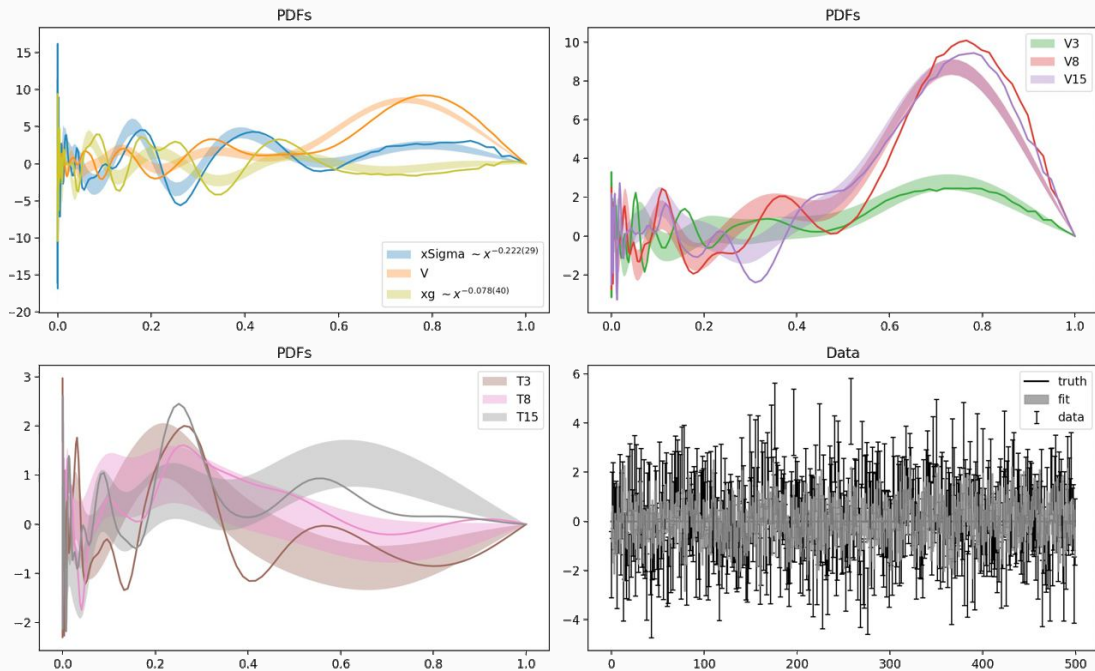


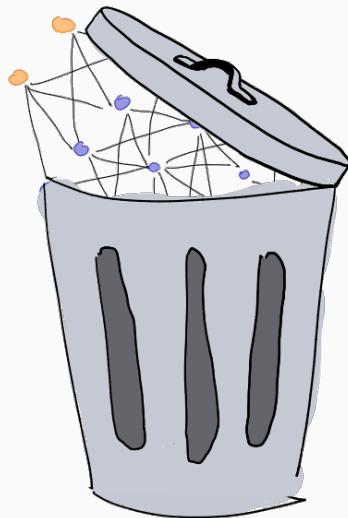
Figure 8: Even with completely random data and theory, we get a valence like structure (valence peak, wrong place, wrong height) by virtue of **sum rules**. Still, they are too much unconstrained to get correct values.

Can we *retire* the Neural Network?

The basic, and most uninformative, answer is “*maybe*”.

In practice, the Neural Network is doing *quite a good job* at giving a simple enough *recipe to determine our kernel*. → At the price of insight loss.

It has trade-offs, but it is doing a rather good job, as proven by tests. It would be silly to completely forget about it, since it is providing another access path to, *in principle*, equivalent results.



You can get the full distribution, including **hyper-parameters**:

$$P(A|B; \lambda) = \frac{P(B|A)P(A; \lambda)}{P(B)}$$

And, *if you wish*, you can **hyper-optimize** to get a single value:

$$\begin{cases} \lambda^* = \max_{\lambda \in \Lambda} P(A|B; \lambda) \\ P(A|B) = \max_{\lambda \in \Lambda} P(A|B; \lambda) = P(A|B; \lambda^*) \end{cases}$$

This is just the **Maximum A Posteriori (MAP)** estimate on hyper-parameters, but the full distribution contains more information.

But at that point you might want to choose a **hyper-prior**, that will affect also the MAP estimate of hyper-parameters (you decide where to stop...but essentially is all *part of your prior*).

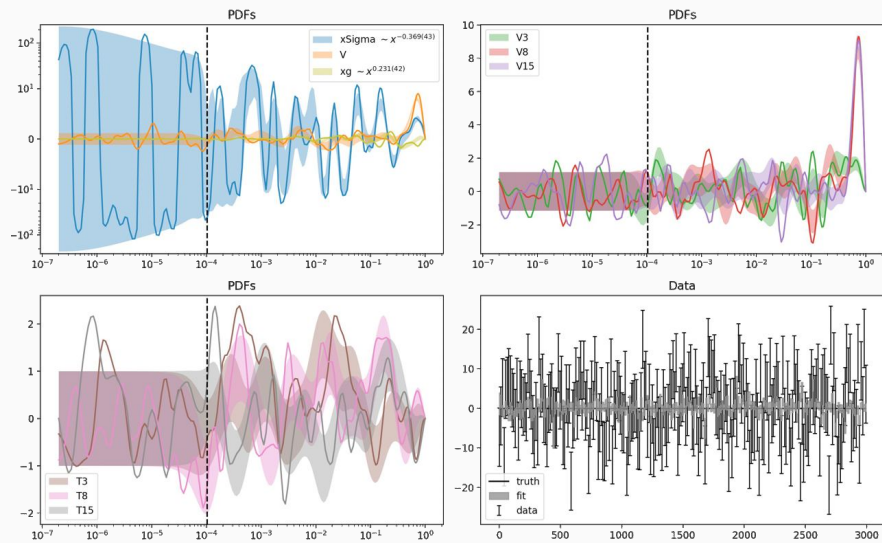


Figure 9: New fit candidate, with less random data (still pretty random).

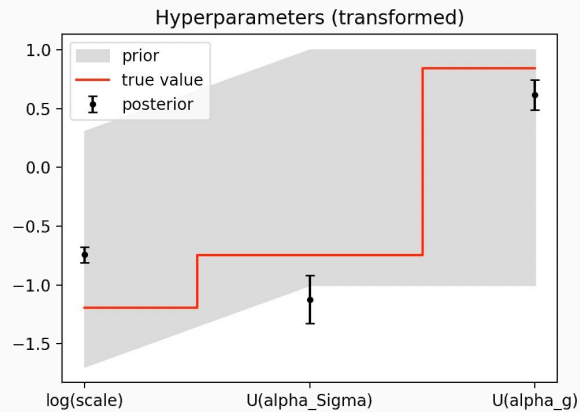


Figure 10: Hyperparameters fit: notice that while exponents are working, we are still missing the correlation length.