NNPDF4.0: Towards a new generation of PDFs using ML

Roy Stegeman

University of Milan and INFN Milan

ML4Jets 2021, 6 July 2021

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 740006.

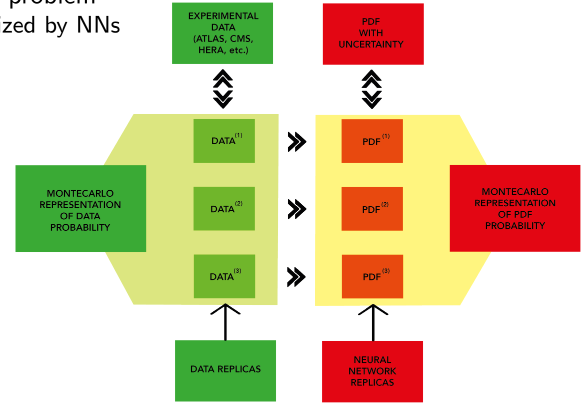
PDFs as an ML problem: the NNPDF approach

Why use machine learning for PDF determination?

- Unknown functional form which needs to be inferred from data
- Well defined input and output

⇒ Supervised learning problem

- PDFs parametrized by NNs



PDF challenges

Key points of the technology used in **NNPDF3.1**:

- Genetic algorithm for optimization
- Implemented in in-house c++ code
- Manual tuning of fit parameters

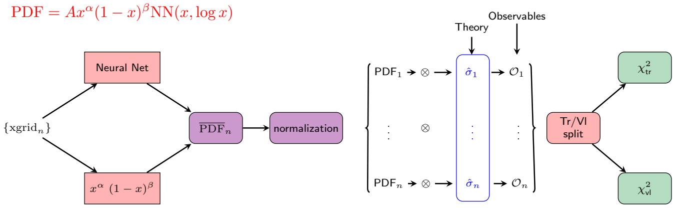
Challenges:

- Can we increase the fit speed?
  - Faster fits $\Rightarrow$ Speed-up of research
- Can we learn the methodology?
  - Systematically determine the best model hyperparameters for our data and theory

$\Rightarrow$ Use technologies from the deep learning community

NNPDF4.0 model

For more information see EPJ C79 (2019) 676



Main changes:

- `Python` codebase
  - Easier and faster development
- Freedom to use external libraries (default: `TensorFlow`)
- Modularity $\Rightarrow$ ability to vary all aspects of the methodology

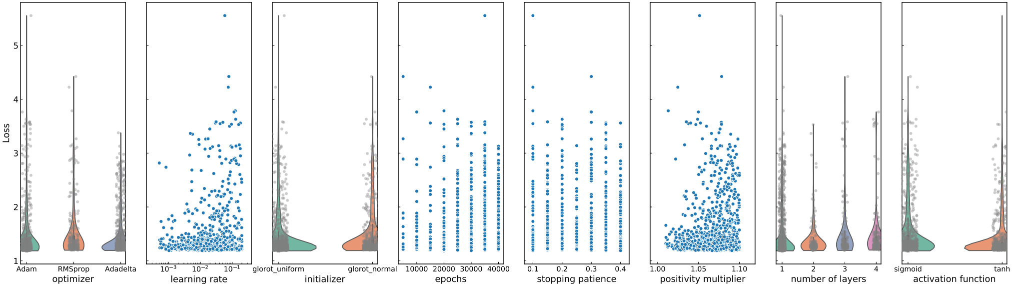
Performance benefit - time per replica

	NNPDF3.1	NNPDF4.0 (CPU)	NNPDF4.0 (GPU)
Fit timing per replica	15.2 h	38 min	6.6 min
Speed up factor	1	24	140
RAM use	1.5 GB	6.1 GB	NA

$\Rightarrow$ More fits in less time

Finding the best methodology: hyperoptimization

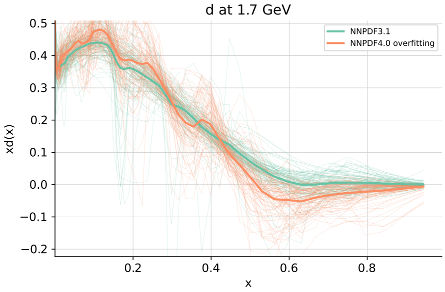
Scan over thousands of hyperparameter combinations and select the best one



- **Optimize** figure of merit: validation χ^2

Overfitting

Using the validation set χ^2 as figure of merit leads to overfitting:



- NNPDF3.1: wiggles are a **finite size effect** that vanishes as N_{rep} grows
- NNPDF4.0: genuine **overfitting** with $\chi^2_{\text{train}} \ll \chi^2_{\text{val}}$

What happened?

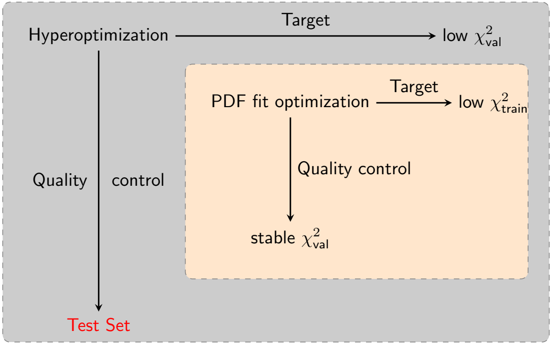
Correlations between training and validation data

$$
\begin{tikzcd}
\text{PDF fit optimization} \arrow[r, "\text{Target}"] \arrow[d, "\text{Quality control}"] & \text{low } \chi^2_{\text{train}} \\
\text{stable } \chi^2_{\text{val}} &
\end{tikzcd}
$$

$\Rightarrow$ Define a proper quality control criterion

Removing overfitting: the test set

Define an uncorrelated test set to test generalization power on unseen data



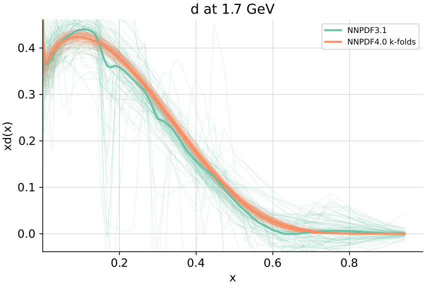
How to choose the test set?

Removing overfitting: k-fold cross-validation

We avoid choosing a test set

The basic idea of **k-fold cross-validation**:

1. Divide the data into k representative subsets
2. Fit $k-1$ sets and use k-th as test set
   $\Rightarrow$ k values of χ^2_{test}
3. Optimize the average χ^2_{test} of the k test sets



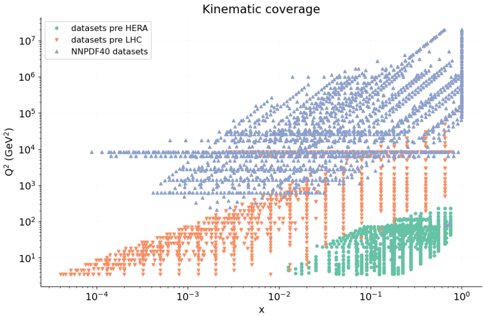
- No overfitting
- Compared to NNPDF3.1:
  - Increased stability
  - Reduced uncertainties

Trusting uncertainties outside the data region

- The improved methodology and extended dataset result in a reduction of the PDF uncertainties
- ‘Closure test’ to validate uncertainty in the data region: arxiv:1410.8849
- Can we trust the uncertainties in the extrapolation region?

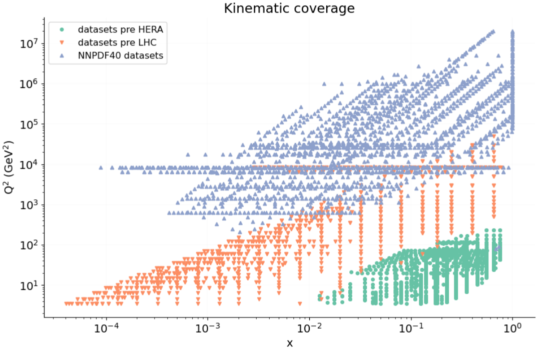
Idea:

1. Take a historic dataset
   e.g. pre-HERA or pre-LHC
2. Perform fit
3. Compare predictions to “future” data



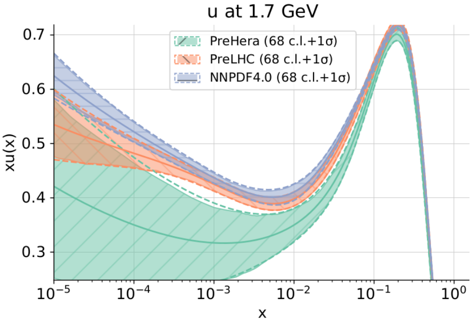
Future tests

For more information see arxiv:2103.08606



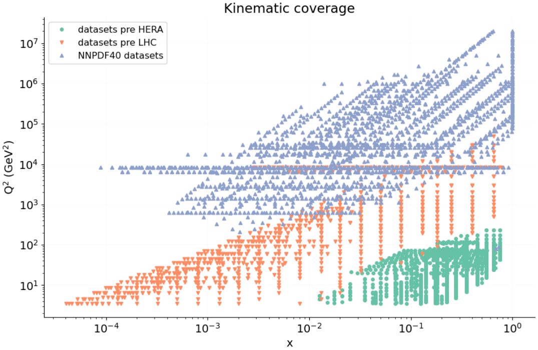
χ^2/N (only exp. covmat)

(dataset)	NNPDF4.0	pre-LHC	pre-Hera
pre-HERA	1.09	1.01	0.90
pre-LHC	1.21	1.20	**23.1**
NNPDF4.0	1.29	**3.30**	**23.1**



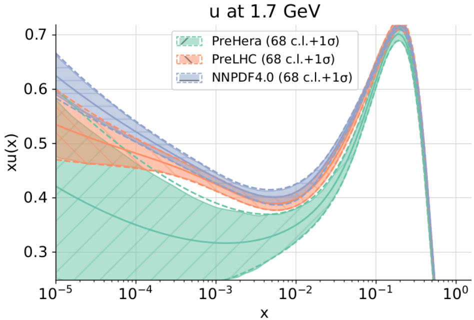
Future tests

For more information see arxiv:2103.08606



χ^2/N (exp. and PDF covmat)

(dataset)	NNPDF4.0	pre-LHC	pre-Hera
pre-HERA			0.86
pre-LHC		1.17	**1.22**
NNPDF4.0	1.12	**1.30**	**1.38**



The total uncertainty increases, and accommodates for difference between predictions and new data.

Open problems

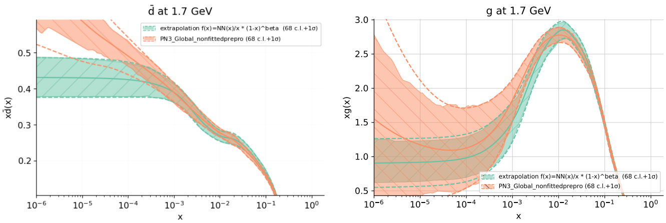
Preprocessing

In future test, extrapolation based on **preprocessing**:

$\mathrm{PDF} = x^{\alpha}(1-x)^{\beta}\mathrm{NN}(x, \log x)$

α, β randomly varied with uniform distribution

If preprocessing is removed, we observe saturation at small-x:



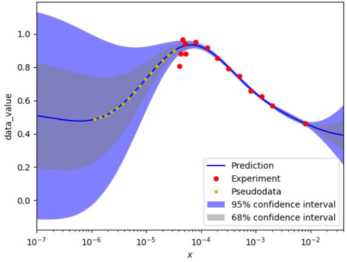
- Modify input scaling
- Model the extrapolation behaviour

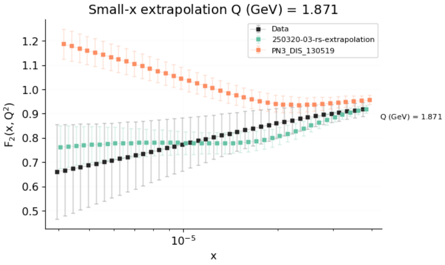
The extrapolation region

Idea:

1. Use Gaussian Process to model DIS observables
2. Propagate a Gaussian prior into the extrapolation region
3. Generate Gaussian pseudodata and include in in a fit

- No preprocessing needed
- x, $\log x$ replaced by a single scaled input





Summary

- Faster and more stable results
- Possibility to learn the methodology
- Faithful reduction of uncertainties in the extrapolation region

- NNPDF code will be made publicly available with documentation

Summary

- Faster and more stable results
- Possibility to learn the methodology
- Faithful reduction of uncertainties in the extrapolation region

- NNPDF code will be made publicly available with documentation

Thank you!

Backup

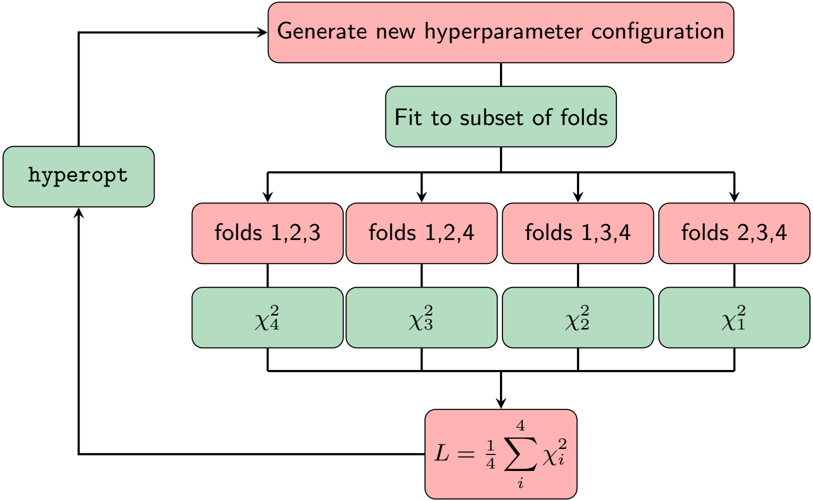
The χ^2 loss function

The fitting strategy is based on the minimization of χ^2:

$$\chi^2 = \frac{1}{N} \sum_i (\mathcal{O}^i - \mathcal{D}^i) \sigma_{ij}^{-1} (\mathcal{P}^i - \mathcal{D}^i), \tag{1}$$

N: number of datapoints,
$\mathcal{D}^i$: experimental data point,
$\mathcal{O}^i$: theoreical prediction,
σ_{ij}: covariance matrix.

K-folding


Generate new hyperparameter configuration
Fit to subset of folds
hyperopt
folds 1,2,3
folds 1,2,4
folds 1,3,4
folds 2,3,4
χ_4^2
χ_3^2
χ_2^2
χ_1^2
$L = \frac{1}{4}\sum_i^4 \chi_i^2$