Towards a new generation of PDFs with deep learning models

Juan M Cruz-Martinez

in collaboration with: S. Carrazza

hep-ph/1907.05075

N3PDF
Machine Learning • PDFs • QCD

QCD@LHC 2019, Buffalo

European Research Council
Established by the European Commission

UNIVERSITAS STUDIORUM MEDIOLANENSIS

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

Outline

NNPDF & N3PDF

N3PDF Machine Learning • PDFs • QCD

NNPDF

n3pdf.mi.infn.it

nnpdf.mi.infn.it

More of NNPDF & N3PDF in QCD@LHC:

- Zahari Kassabov: Recent developments in PDFs
  [Today in the morning]
- Christopher Schwan: PDF-independent Electroweak and Photon-induced Theoretical Predictions
  [Today after the coffee break]

NNPDF

Quick recap:

NNPDF is a big collaboration spanning several universities and research centers.

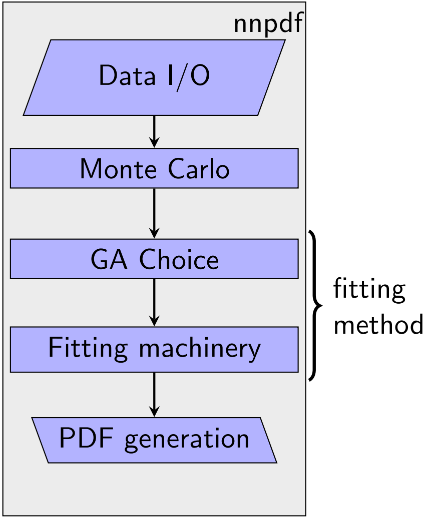
There are two main characteristics that define NNPDF fits:

→ Monte Carlo method applied to PDF determination

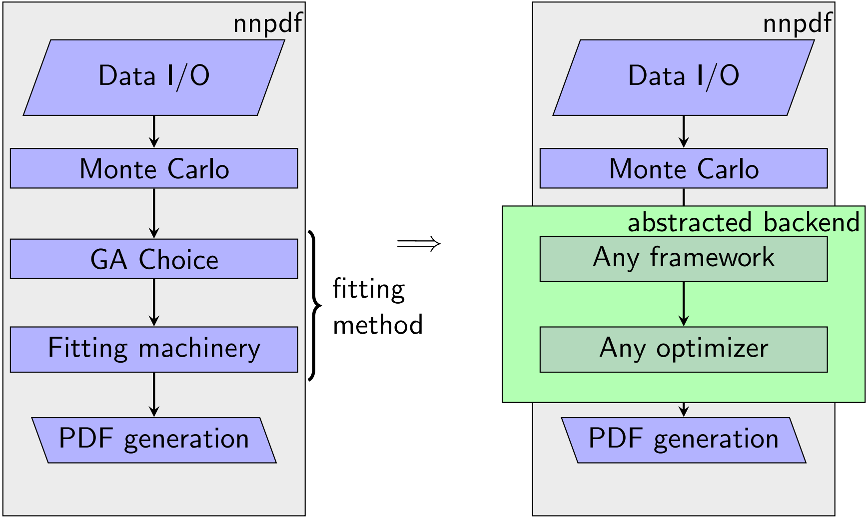
→ Each PDF flavour corresponds to a different Neural Network: i.e., no guesses made on their functional form. Optimization with genetic algorithms.

Read more: hep-ph/1410.8849

The goal: towards new methodologies



The goal: towards new methodologies



Motivation: methodological gains

✓ Gains on speed and efficiency:
- Less CPU hours for a fit
- More studies available

Motivation: methodological gains

✓ Gains on speed and efficiency:
  - Less CPU hours for a fit
  - More studies available

✓ Rationalization of development
  - Easier development
  - OOP: full freedom and flexibility

Motivation: methodological gains

✓ Gains on speed and efficiency:
  - Less CPU hours for a fit
  - More studies available

✓ Rationalization of development
  - Easier development
  - OOP: full freedom and flexibility

✓ Usage of new technologies
  - Use of alternative devices: GPU / CPU / FPGAs
  - Implement latest development in ML libraries
  - Gradient Descent algorithms

Design choices: language and framework

```
Language: python
      > version: 3.7
```

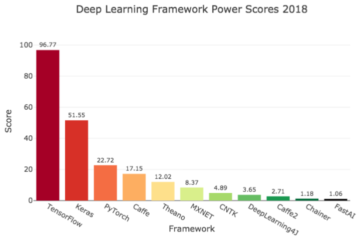
- ✓ Widely used and easy to learn
- ✓ Supported by most ML libraries:
  (Tensorfow, Pytorch, CNTK, ...)

Design choices: language and framework

`Language: python`
`> version: 3.7`

- ✓ Widely used and easy to learn
- ✓ Supported by most ML libraries: (Tensorfow, Pytorch, CNTK, . . .)

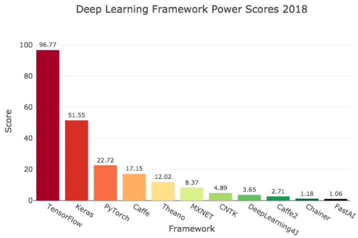
`Framework: Keras`
`> backend: Tensorflow`



Design choices: language and framework

`Language: python`
`> version: 3.7`

- ✓ Widely used and easy to learn
- ✓ Supported by most ML libraries: (Tensorfow, Pytorch, CNTK, ...)



`Framework: Keras`
`> backend: Tensorflow`

- ✓ High level of abstraction
- ✓ Powerful features of Tensorflow
- ✓ Trivially change between (supported) libraries
- ✓ CPU parallelization
- ✓ GPU parallelization
- ✓ FPGA support

Plot Source

Note: the metric in the figure includes GitHub activity and ArXiv articles: Research and development.

Ingredients

Experimental data and theoretical predictions: NNPDF 3.1

Neural Network defition

Algorithm selection

Gradient descent: deterministic optimization

Object oriented approach: everything is (as) independent (as possible) from everything else!

Ingredients

Experimental data and theoretical predictions: NNPDF 3.1

Neural Network defition

- A loss function
- A definition for the PDF model (dense, dropout, . . .)

Algorithm selection

Gradient descent: deterministic optimization

Object oriented approach: everything is (as) independent (as possible) from everything else!

Ingredients

Experimental data and theoretical predictions: NNPDF 3.1

Neural Network defition

- A loss function
- A definition for the PDF model (dense, dropout, ...)

Algorithm selection

- Stopping algorithm
- Gradient descent algorithm (RMSprop, Adadelta, ...)

Gradient descent: deterministic optimization

Object oriented approach: everything is (as) independent (as possible) from everything else!

The Loss Function

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

$$\chi^2 = \frac{1}{N} \sum \left(\mathcal{O}^i - \mathcal{D}^i\right) \sigma_{ij}^{-1} \left(\mathcal{O}^j - \mathcal{D}^j\right)$$

N: number of data points

$\mathcal{O}^i$: theoretical prediction

$\mathcal{D}^i$: experimental data point

σ_{ij}: covariance matrix

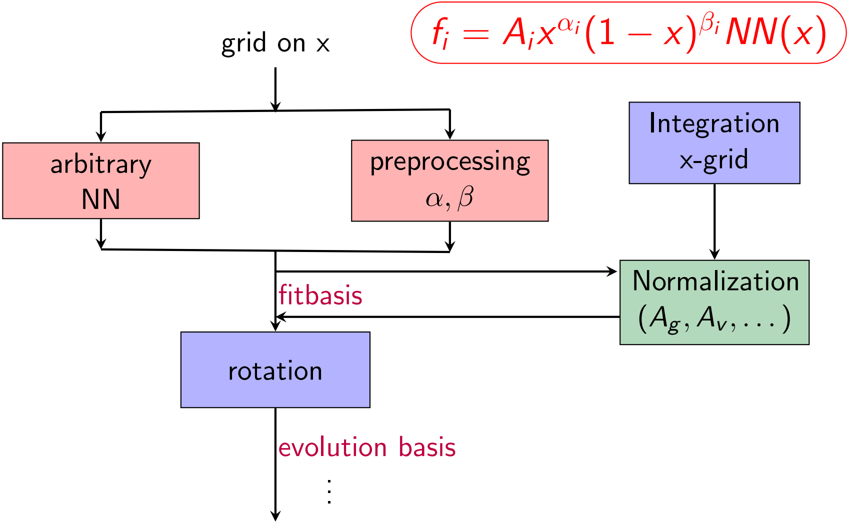
Unknown functions

$$\boxed{\mathcal{O}} = \boxed{pdf \otimes} \boxed{(pdf \otimes)} \boxed{\hat{\sigma}}$$

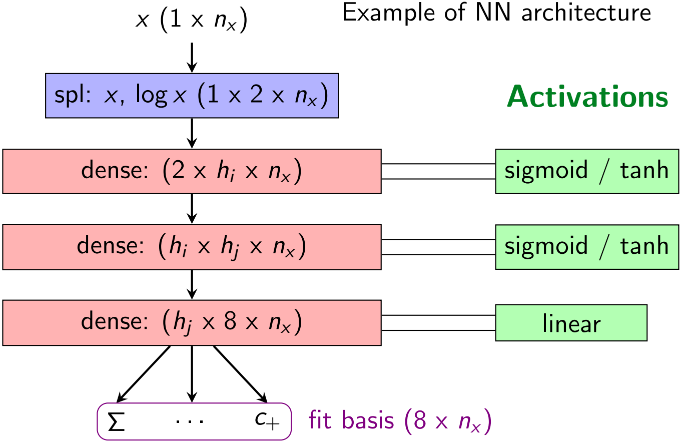
independent layers

Note: The partonic cross section σ correspond to APFELgrid tables as described in hep-ph/1605.02070

The PDF model

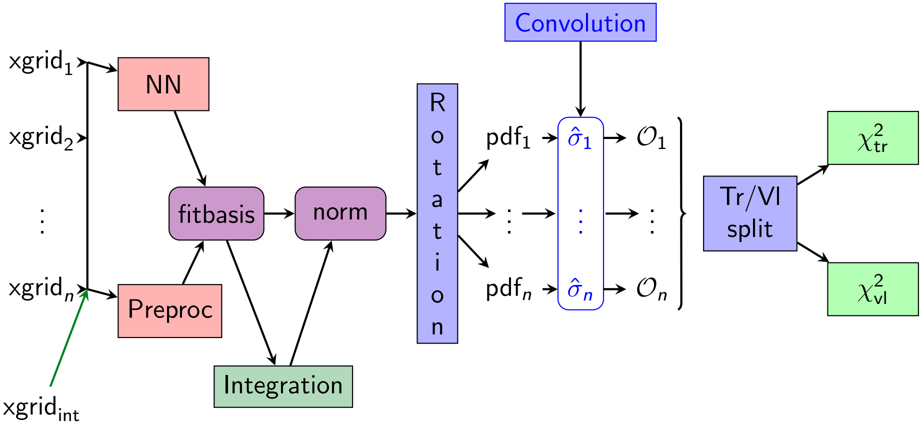


The Neural Network: $NN(x)$



Swapping and testing different network architectures is a matter of seconds: we can systematically scan and find the best model.

The full model



Scan over hyperparameters: fitting the methodology

The main goal of NNPDF was to reduce the bias introduced in the PDF fits by the choice of the functional form of the PDFs, but...

- ✗ Neural Networks are defined by set of parameters

  ...any choice can introduce a human bias

In order to overcome these issues we implement the hyperopt library.

The next step: fitting the whole methodology

Scan over hyperparameters: fitting the methodology

The main goal of NNPDF was to reduce the bias introduced in the PDF fits by the choice of the functional form of the PDFs, but...

- ✗ Neural Networks are defined by set of parameters

  ...any choice can introduce a human bias

In order to overcome these issues we implement the hyperopt library.

- ✓ Scan over thousands of hyperparameter combinations

The next step: fitting the whole methodology

Scan over hyperparameters: fitting the methodology

The main goal of NNPDF was to reduce the bias introduced in the PDF fits by the choice of the functional form of the PDFs, but. . .

✗ Neural Networks are defined by set of parameters

. . . any choice can introduce a human bias

In order to overcome these issues we implement the hyperopt library.

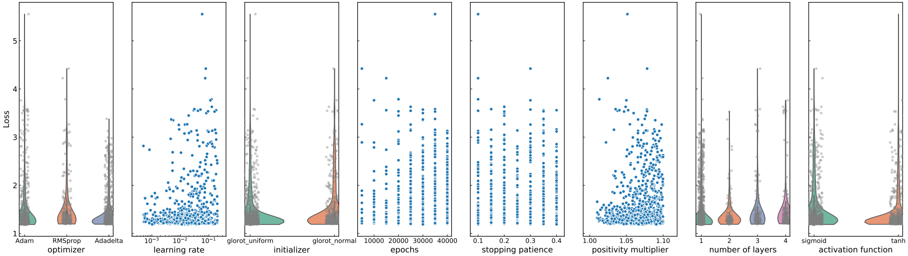
✓ Scan over thousands of hyperparameter combinations

✓ Define a reward function to grade the model

The next step: fitting the whole methodology

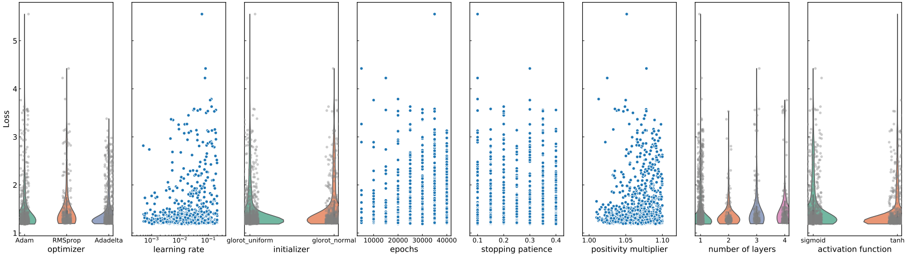
Hyperparameter scan

Each blue dot corresponds to a different set of hyparameters.



Hyperparameter scan

Each blue dot corresponds to a different set of hyparameters.

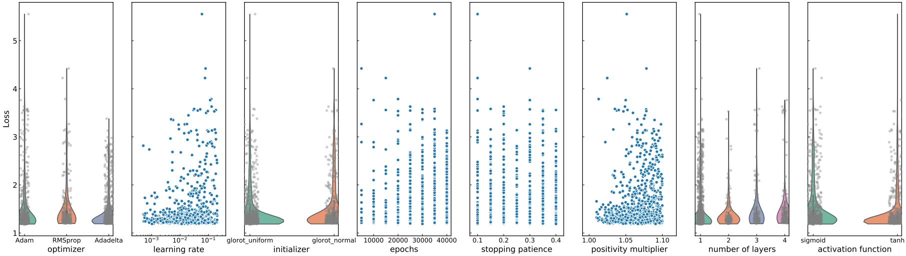


✓ Optimizer

✓ Learning Rate

Hyperparameter scan

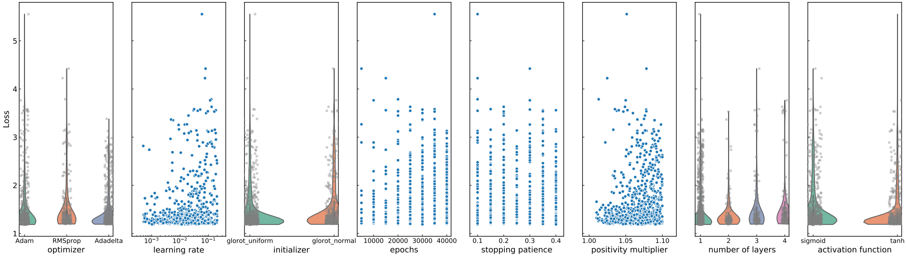
Each blue dot corresponds to a different set of hyparameters.



- ✓ Optimizer
- ✓ Initializer
- ✓ Learning Rate
- ✓ Epochs

Hyperparameter scan

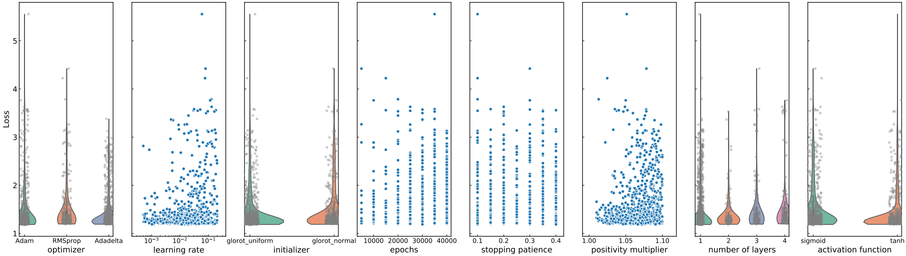
Each blue dot corresponds to a different set of hyparameters.



- ✓ Optimizer
- ✓ Initializer
- ✓ Stopping Patience
- ✓ Learning Rate
- ✓ Epochs
- ✓ Positivity Multiplier

Hyperparameter scan

Each blue dot corresponds to a different set of hyparameters.



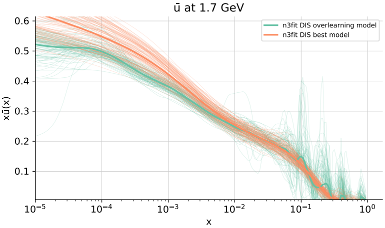
- ✓ Optimizer
- ✓ Initializer
- ✓ Stopping Patience
- ✓ Number of Layers
- ✓ Learning Rate
- ✓ Epochs
- ✓ Positivity Multiplier
- ✓ Activation Function

Warning: overfitting!

With great power comes great responsability.

An unsupervised parameter scan is dangerous: it can find that overfitting is preferrable.

- ✗ It did minimise the validation!
- ✗ Hyperopt is able to trick cross-validation when choosing the model.

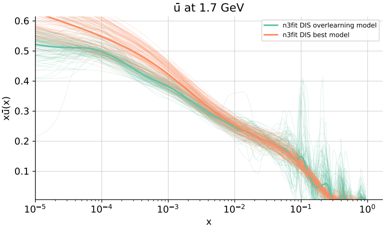


Warning: overfitting!

With great power comes great responsability.

An unsupervised parameter scan is dangerous: it can find that overfitting is preferrable.

- ✗ It did minimise the validation!
- ✗ Hyperopt is able to trick cross-validation when choosing the model.



Solution:

- ✓ Create a test-set:

  Take a few experiments out of the hyperparameter scan and use them to probe the generalization power of the network

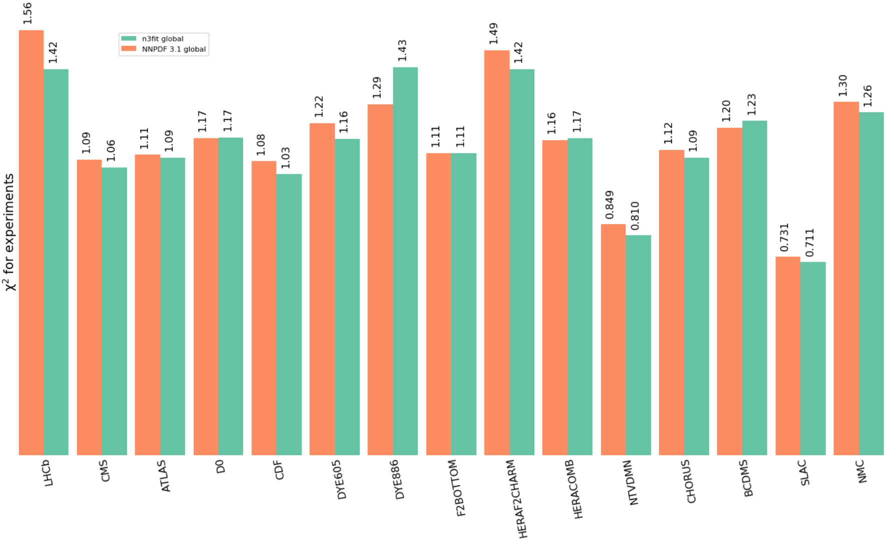
Results report

As an example we show a comparison using as a baseline a global NNPDF 3.1 NNLO fit against a model created by the hyperoptimization procedure. In green we show our implementation, called here `n3fit`, in orange the old methodology (NNPDF 3.1)

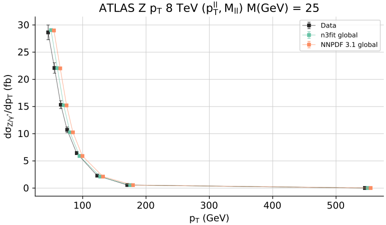
χ^2	`n3fit`	NNPDF 3.1
Experimental χ^2	1.149	1.158
Average time per replica	90 minutes	30 hours
Average mem. use per replica	14 Gb	5 Gb

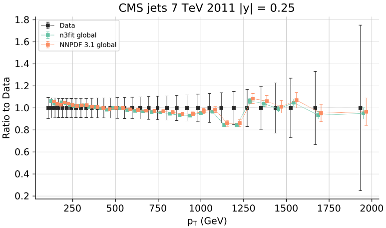
Memory consuption can be reduced by replacing some of Tensorflow operators (most notably, convolution). Currently undergoing!

Per-experiment results



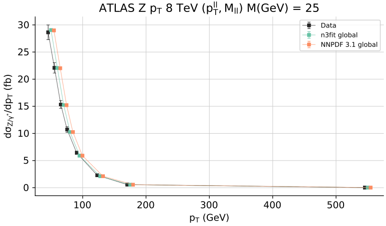
Comparison to data

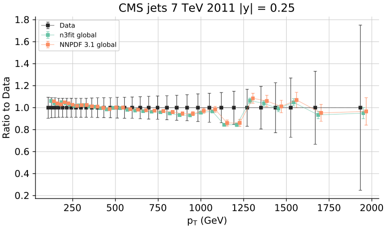




$\rightarrow$ Results compatible with NNPDF 3.1

Comparison to data

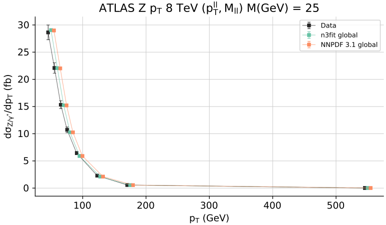


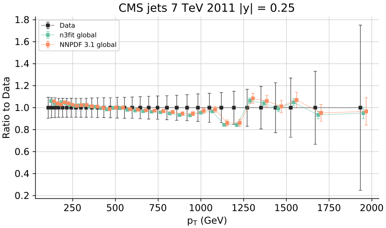


$\rightarrow$ Results compatible with NNPDF 3.1

$\rightarrow$ Not only a similar χ^2-goodness but also similar per-point results

Comparison to data



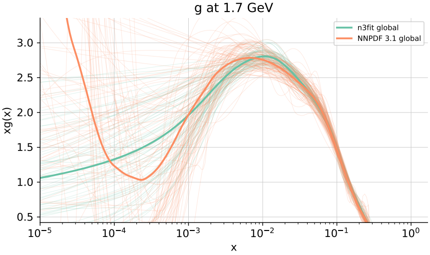


→ Results compatible with NNPDF 3.1

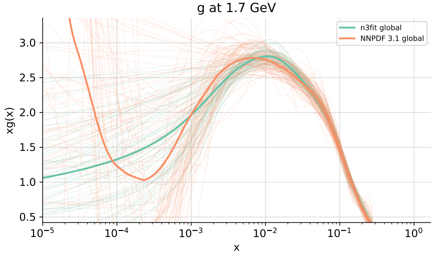
→ Not only a similar χ^2-goodness but also similar per-point results

✓ The new methodology is compatible with the previous one!

Stability

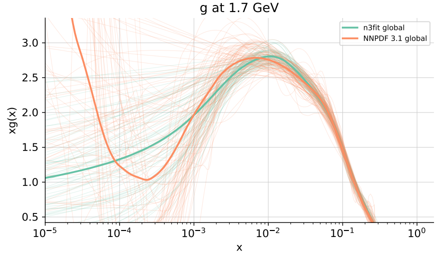

g at 1.7 GeV
n3fit global
NNPDF 3.1 global
xg(x)
x
3.0
2.5
2.0
1.5
1.0
0.5
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}

Stability



✓ Better stability replica-by-replica

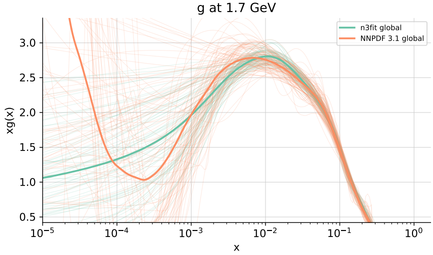
Stability

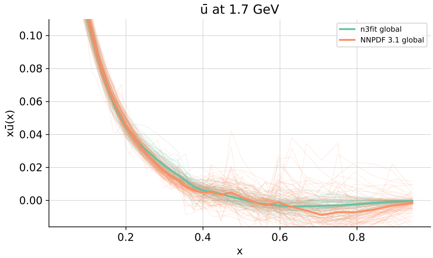


- ✓ Better stability replica-by-replica
- ✓ More replicas satisfiy post-fit requirements

Which translates to

Stability



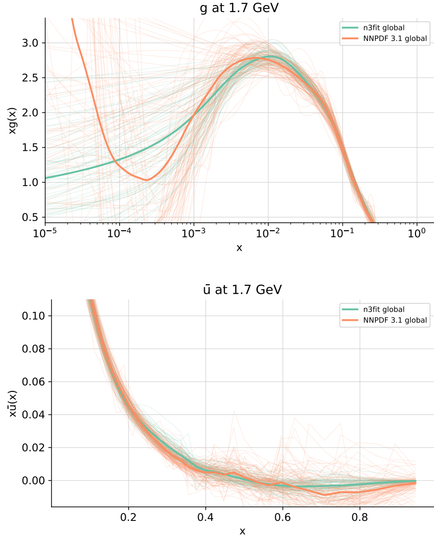


- ✓ Better stability replica-by-replica
- ✓ More replicas satisfiy post-fit requirements

Which translates to

- ✓ Even smaller computing times!

Stability



✓ Better stability replica-by-replica

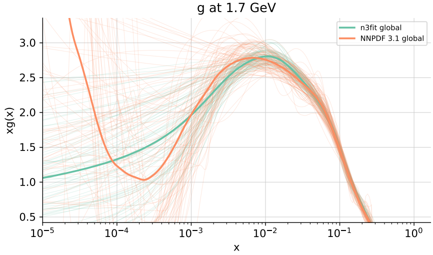
✓ More replicas satisfiy post-fit requirements

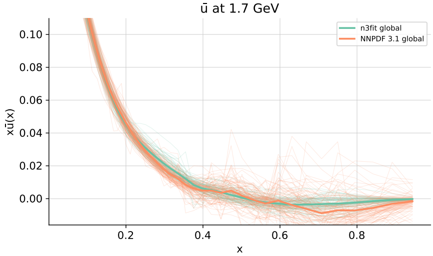
Which translates to

✓ Even smaller computing times!

✓ More complete statistical analysis at the same cost

Stability





✓ Better stability replica-by-replica

✓ More replicas satisfiy post-fit requirements

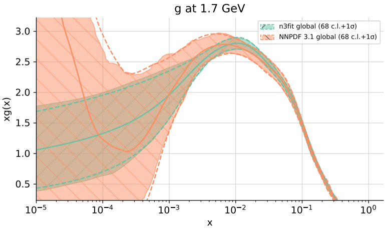
Which translates to

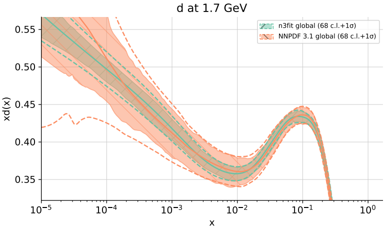
✓ Even smaller computing times!

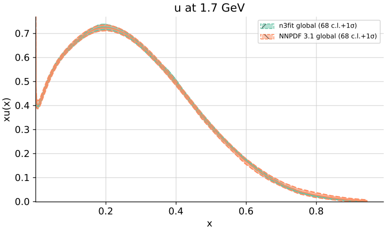
✓ More complete statistical analysis at the same cost

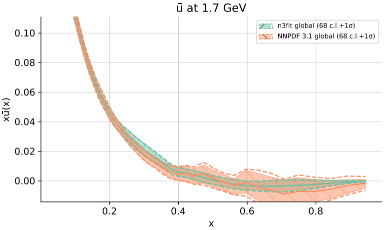
✓✓ More accurate PDF determination!

PDF shapes


g at 1.7 GeV
n3fit global (68 c.l.+1σ)
NNPDF 3.1 global (68 c.l.+1σ)
xg(x)
x


d at 1.7 GeV
n3fit global (68 c.l.+1σ)
NNPDF 3.1 global (68 c.l.+1σ)
xd(x)
x


u at 1.7 GeV
n3fit global (68 c.l.+1σ)
NNPDF 3.1 global (68 c.l.+1σ)
xu(x)
x


ū at 1.7 GeV
n3fit global (68 c.l.+1σ)
NNPDF 3.1 global (68 c.l.+1σ)
xū(x)
x

Summary

- ✓ We have achieved a very powerful, flexible and fast machinery for PDF fitting.

- ✓ Faster run times: iterate over different choices of models or parameters
- ✓ The framework allows full customization *by design*
- ✓ PDFs are more accurate than before and compatible with previous results: methodological error reduced!

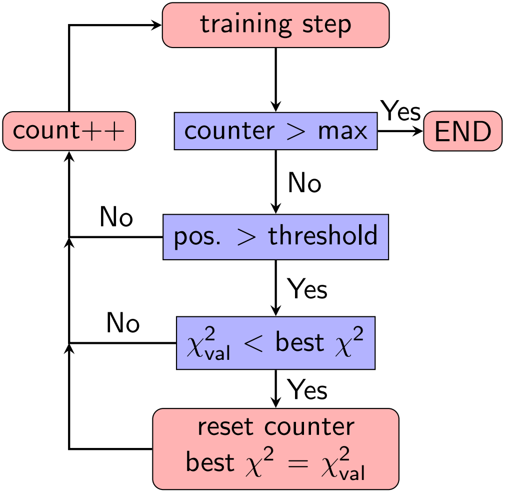
Lovely bonus: we can automatically pull enhancements build on top of the libraries we are already using!

The end

Thanks!

Stopping

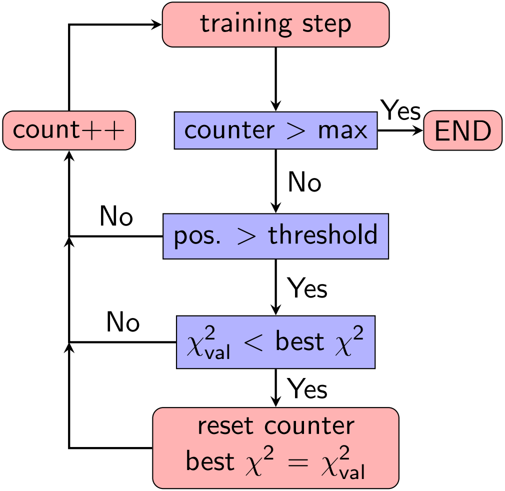
`Stopping method:` **Look-back method where positivity passes**



Stopping

`Stopping method:` **Look-back method where positivity passes**

Early stopping: reduce overfitting

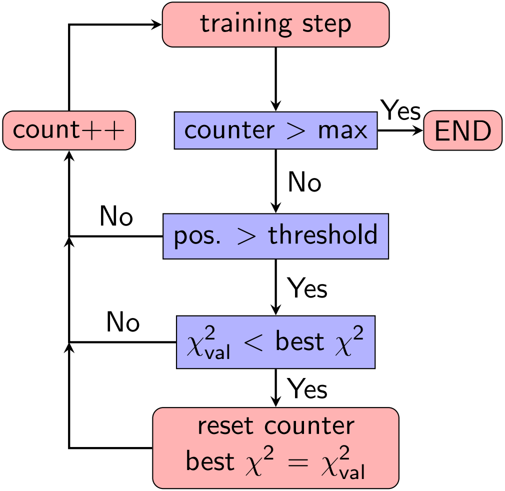


Stopping

`Stopping method:` **Look-back method where positivity passes**

Early stopping: reduce overfitting

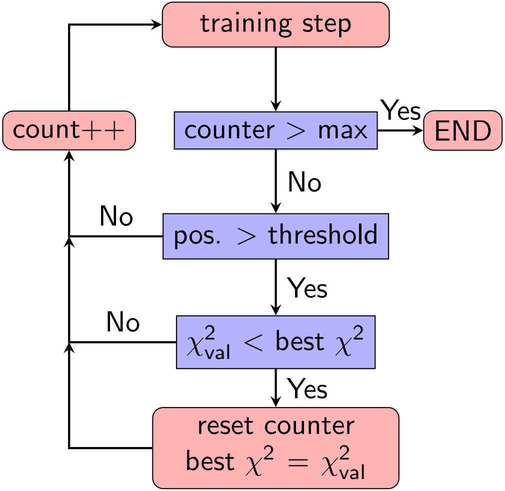
✓ Minimize χ^2 of validation sets



Stopping

`Stopping method:` **Look-back method where positivity passes**

Early stopping: reduce overfitting

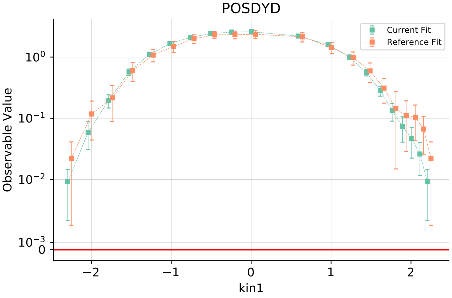


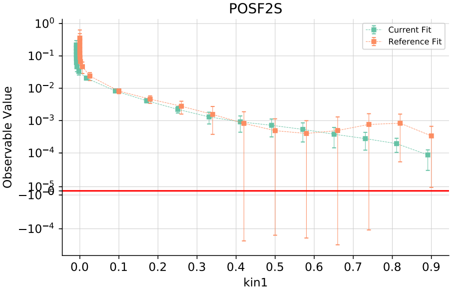
✓ Minimize χ^2 of validation sets

✓ Enforces positivity constraints

Positivity constrained

Once all these considerations are applied, we obtain no replicas of negative positivity.

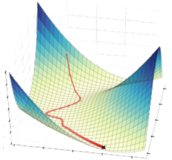




Design choices: the optimisation algorithm

```
Family:  Gradient Descend
      > algorithm:  RMSprop
```

Plot Source



The family of Gradient Descent algorithms are based on the minimization of the loss function by computing its gradient and updating all weights in the opposite direction.

Our goal is to achieve a methodology general enough so the algorithm is not fixed.

We could even swap the GD algorithm for a Genetic Algorithm!