Methodological improvements in PDF determination

Juan M Cruz-Martinez

Machine Learning • PDFs • QCD

PDF4LHC & James Stirling Memorial, Durham (2019)

European Research Council
Established by the European Commission

Outline

1. A new methodology, codename `n3fit`
   - N3PDF & NNPDF
   - Motivation: speed & flexibility $\rightarrow$ more physics
2. New code: `n3fit`
   - In detail
   - Hyperoptimization: fitting the methodology
   - New methodology, new fits
3. Accelerating the fit
   - Handcrafting operations
   - Hardware acceleration

NNPDF & N3PDF

Machine Learning • PDFs • QCD

n3pdf.mi.infn.it

nnpdf.mi.infn.it

The N3PDF group develops and tests new strategies which, if successful, will be used by the NNPDF collaboration.

- ✓ New methodologies
- ✓ New hardware
- ✓ State-of-the-art tools
- ✓ Experimental techniques

The N3PDF project has received funding from the EU's Horizon 2020 research and innovation programme under grant agreement No 740006.

NNPDF & N3PDF

n3pdf.mi.infn.it nnpdf.mi.infn.it

The first main result of the N3PDF group is the new fitting code: `n3fit` which will be used for the forthcoming NNPDF4.0 PDF set.

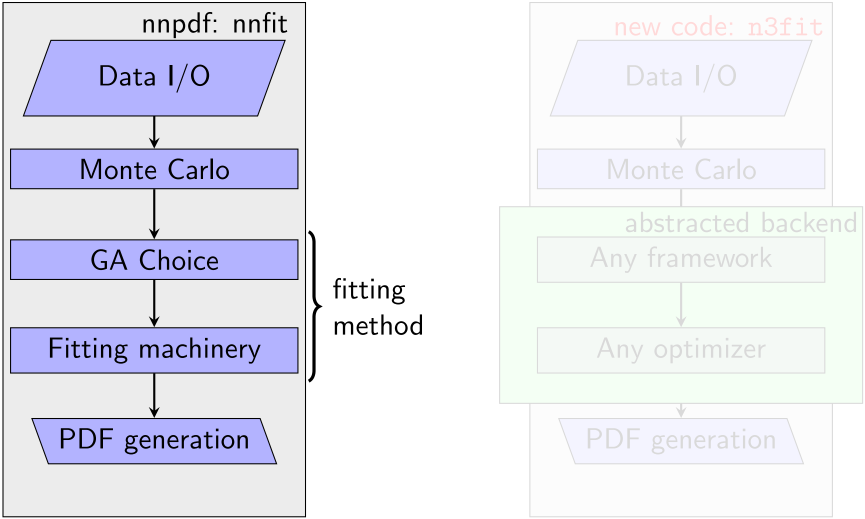
$\longrightarrow$ A publication detailing `n3fit` can be found at

hep-ph/1907.05075 (S. Carrazza, **JCM**).

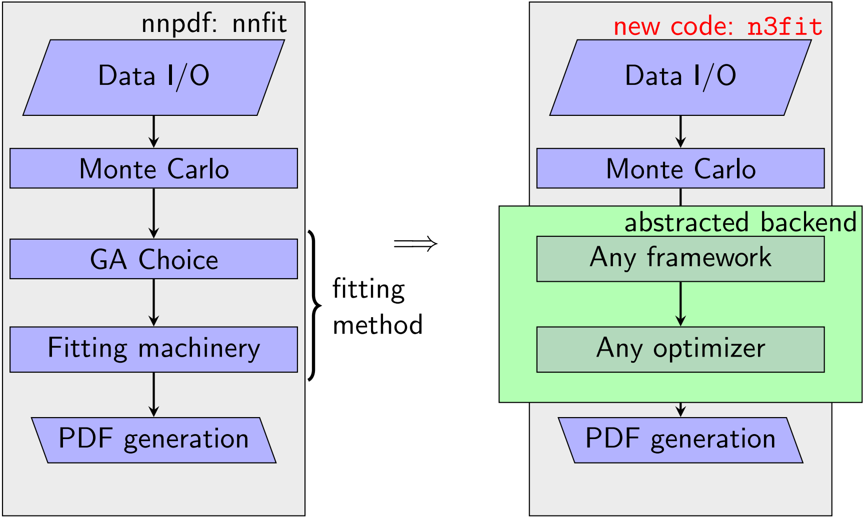
$\longrightarrow$ A publication detailing progress in hardware acceleration of PDF fitting is in preparation

(S. Carrazza, **JCM**, J. Urtasun-Elizari, E.Villa)

The goal: towards new methodologies

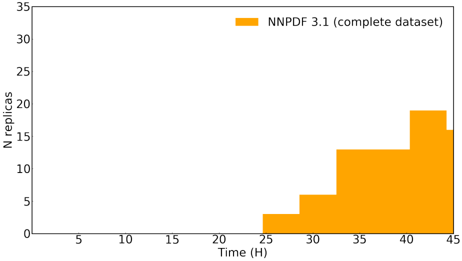


The goal: towards new methodologies



Motivation: more studies available

✓ Rationalization of development
- Easier and faster development
- Object orientation: full freedom and flexibility



✓ Gains on speed and efficiency:
- Less CPU hours for a fit
- Usage of new technologies
  - ✓ New hardware
  - ✓ New libraries

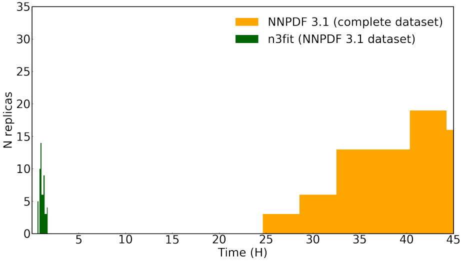
✓ Consequences
- **Speed-up of research**: faster to develop, test, run
- **More studies available**

→ Example: **fitting the methodology** (hyperparameter scan)

Motivation: more studies available

✓ Rationalization of development
  - Easier and faster development
  - Object orientation:
    full freedom and flexibility

✓ Gains on speed and efficiency:
  - Less CPU hours for a fit
  - Usage of new technologies
    ✓ New hardware
    ✓ New libraries



✓ Consequences
  - **Speed-up of research**: faster to develop, test, run
  - **More studies available**
  → Example: **fitting the methodology** (hyperparameter scan)

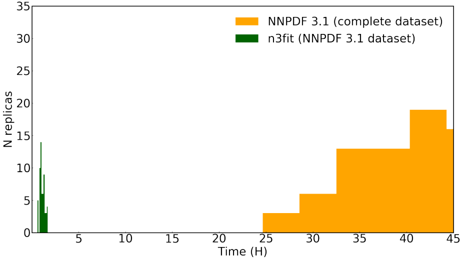
Motivation: more studies available

✓ Rationalization of development

- Easier and faster development
- Object orientation:
  full freedom and flexibility

✓ Gains on speed and efficiency:

- Less CPU hours for a fit
- Usage of new technologies
  - ✓ New hardware
  - ✓ New libraries



✓ Consequences

- **Speed-up of research**: faster to develop, test, run
- **More studies available**

→ Example: **fitting the methodology** (hyperparameter scan)

Some differences with respect to the old methodology

NNPDF 3.1 code

- → Genetic Algorithm optimizer
- → One network per flavour
- → Sum rules imposed outside of optimization
- → Preprocessing fixed per each of the replicas
- → C++ monolithic codebase
- → Fit parameters manually chosen (i.e., manual optimization of hyperparameter)
- → In-house ML framework

`n3fit` code

- → Gradient Descent optimization
- → One network for all flavours
- → Sum rules imposed during optimization
- → Preprocessing fitted within replicas
- → Python object oriented codebase
- → Fit parameters chosen automatically (hyperparameter scan)
- → Complete freedom for choosing the ML library: i.e., tensorflow.

Some differences with respect to the old methodology

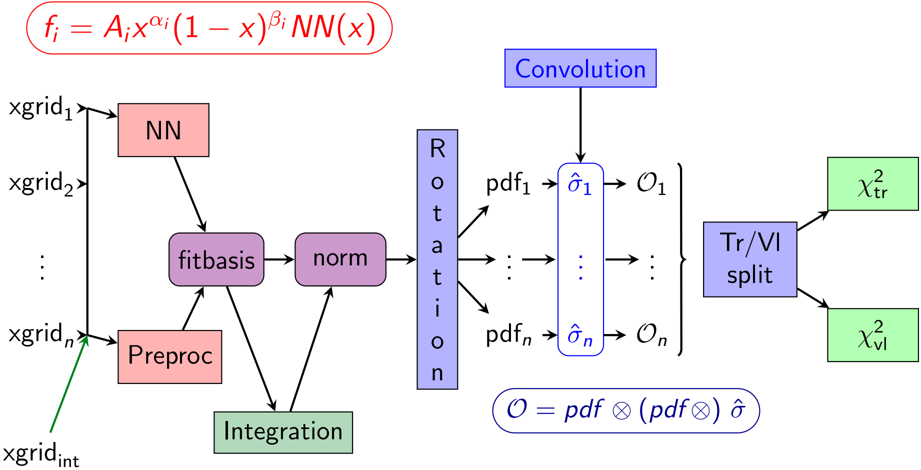
NNPDF 3.1 code

- → **Genetic Algorithm optimizer**
- → One network per flavour
- → Sum rules imposed outside of optimization
- → Preprocessing fixed per each of the replicas
- → C++ monolithic codebase
- → **Fit parameters manually chosen (i.e., manual optimization of hyperparameter)**
- → In-house ML framework

`n3fit` code

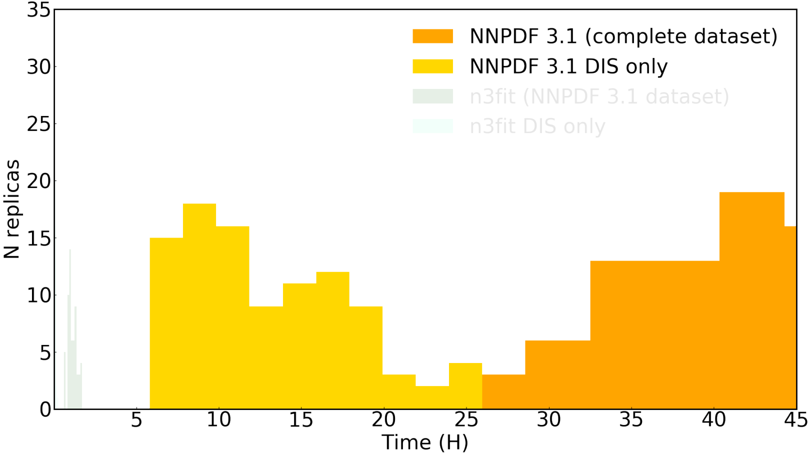
- → **Gradient Descent optimization**
- → One network for all flavours
- → Sum rules imposed during optimization
- → Preprocessing fitted within replicas
- → Python object oriented codebase
- → **Fit parameters chosen automatically (hyperparameter scan)**
- → Complete freedom for choosing the ML library: i.e., tensorflow.

The full model



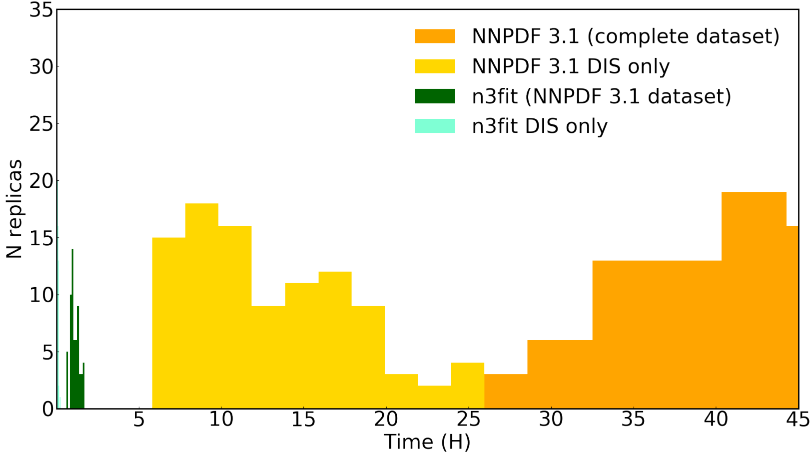
Practical example: time distribution of replica fitting

An important outcome of the development of `n3fit` is that the performance of the fits has been greatly increased, enabling new studies.

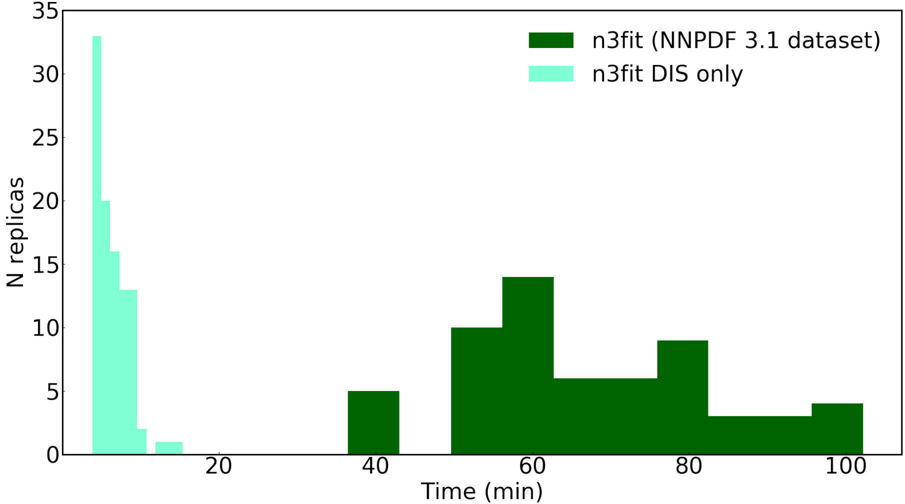


Practical example: time distribution of replica fitting

An important outcome of the development of `n3fit` is that the performance of the fits has been greatly increased, enabling new studies.

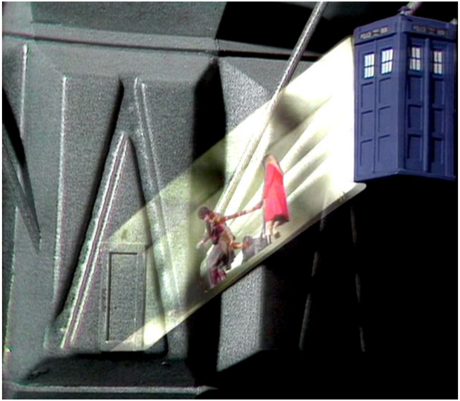


From hours to minutes



The art of the hyperparameter selection

Just as technology has changed the way movies are done, one of studies that the new code enables, is the automatic and systematic **hyperparameter scan** which is rendered possible by the advances in technology and the new code's speed.



1978

2014

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…

⟶ NN are defined by set of parameters

⟶ Humans are good at recognising patterns

✗ selecting the right parameters is a slow process and success is not guaranteed

To overcome these issues we implement a hyperparameter scan: let the computer decide automatically

✓ Scan over thousands of hyperparameter combinations

✓ Define a reward function to grade the model

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...

⟶ NN are defined by set of parameters

⟶ Humans are good at recognising patterns

✗ selecting the right parameters is a slow process and success is not guaranteed

To overcome these issues we implement a hyperparameter scan: let the computer decide automatically

✓ Scan over thousands of hyperparameter combinations

✓ Define a reward function to grade the model

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...

- ⟶ NN are defined by set of parameters
- ⟶ Humans are good at recognising patterns
- ✗ selecting the right parameters is a slow process and success is not guaranteed

To overcome these issues we implement a hyperparameter scan: let the computer decide automatically

- ✓ Scan over thousands of hyperparameter combinations
- ✓ Define a reward function to grade the model

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...

- ⟶ NN are defined by set of parameters
- ⟶ Humans are good at recognising patterns
- ✗ selecting the right parameters is a slow process and success is not guaranteed

To overcome these issues we implement a hyperparameter scan: let the computer decide automatically

- ✓ Scan over thousands of hyperparameter combinations
- ✓ Define a reward function to grade the model

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...

- $\longrightarrow$ NN are defined by set of parameters
- $\longrightarrow$ Humans are good at recognising patterns
- ✗ selecting the right parameters is a slow process and success is not guaranteed

To overcome these issues we implement a hyperparameter scan: let the computer decide automatically

- ✓ Scan over thousands of hyperparameter combinations
- ✓ Define a reward function to grade the model

The next step: fitting the whole methodology

Hyperoptimization: The reward function

The human component is not completely detached it is necessary to define a reward function by choosing the characteristics we find desirable in a fit:

- Goodness of the fit.
- Smoothness of the result.
- Time it takes to complete the full fit.
- Generalization power to future exp data.

Selecting a good reward function (although can be highly non-trivial) offers several advantages:

- ✓ Several fits can present similar goodness but differ in other features.
- ✓ Reduced bias on parameter selection.
- ✓ Automatic scan of the whole space of parameters.

Hyperoptimization: The reward function

The human component is not completely detached it is necessary to define a reward function by choosing the characteristics we find desirable in a fit:

- Goodness of the fit.
- Smoothness of the result.
- Time it takes to complete the full fit.
- Generalization power to future exp data.

Selecting a good reward function (although can be highly non-trivial) offers several advantages:

- ✓ Several fits can present similar goodness but differ in other features.
- ✓ Reduced bias on parameter selection.
- ✓ Automatic scan of the whole space of parameters.

Hyperoptimization: The reward function

The human component is not completely detached it is necessary to define a reward function by choosing the characteristics we find desirable in a fit:

- Goodness of the fit.
- Smoothness of the result.
- Time it takes to complete the full fit.
- Generalization power to future exp data.

Selecting a good reward function (although can be highly non-trivial) offers several advantages:

✓ Several fits can present similar goodness but differ in other features.

✓ Reduced bias on parameter selection.

✓ Automatic scan of the whole space of parameters.

Hyperoptimization: The reward function

The human component is not completely detached it is necessary to define a reward function by choosing the characteristics we find desirable in a fit:

- **Goodness of the fit.**
- Smoothness of the result.
- Time it takes to complete the full fit.
- **Generalization power to future exp data.**

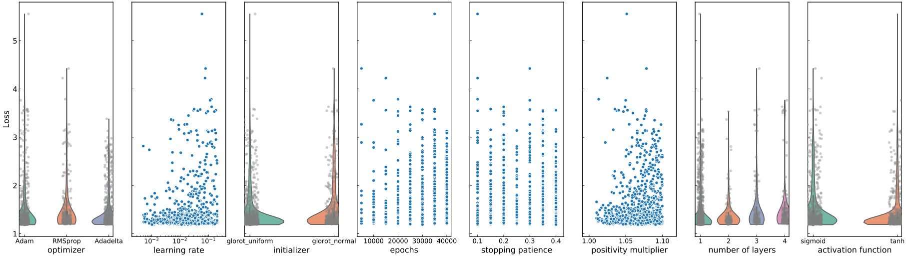
Example of function to hyperoptimize:

$$\text{Loss} = \frac{1}{2}\left(\chi^2_{\text{fit}} + \chi^2_{\text{generalization set}}\right)$$

Where "generalization set" corresponds to experimental data that did not enter the fit.

Hyperparameter scan

Each blue dot corresponds to a fit of a different set of hyperparameters:



Thousands of fits for the hyperoptimization algorithm to choose:

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

Comparison between new and old methodologies

`n3fit` is fully implemented now and produces results which are compatible with previous releases of NNPDF at a lesser cost.

As a proof of concept we present a fit done with `n3fit` after a run of the automated hyperoptimization

	`n3fit`	NNPDF 3.1
χ^2	1.149	1.158
Avg time	70 minutes	35 hours
Memory	16 Gb	5 Gb
Good replicas	95%	70%

- Same dataset selection
- Same positivity constraints
- ✓ Very different methodologies
- ✓ Very similar fit goodness
- ✓ Orders of magnitude faster

Note: Good replicas refer to those which produce a good fit in the allotted number of iterations.

Comparison between new and old methodologies

`n3fit` is fully implemented now and produces results which are compatible with previous releases of NNPDF at a lesser cost.

As a proof of concept we present a fit done with `n3fit` after a run of the automated hyperoptimization

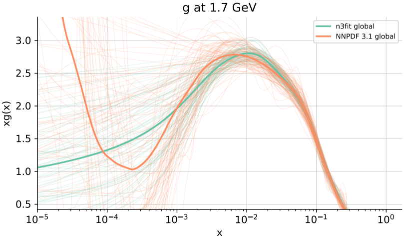
	`n3fit`	NNPDF 3.1
χ^2	1.149	1.158
Avg time	70 minutes	35 hours
Memory	16 Gb	5 Gb
Good replicas	95%	70%

- Same dataset selection
- Same positivity constraints
- ✓ Very different methodologies
- ✓ Very similar fit goodness
- ✓ Orders of magnitude faster

Note: Good replicas refer to those which produce a good fit in the allotted number of iterations.

Result comparison: replica-by-replica

Comparison, with the same selection of data, of the old and new codes.



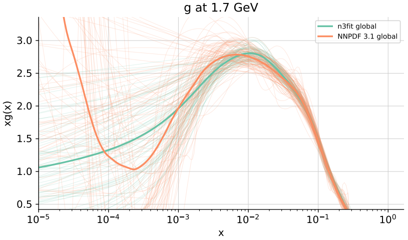
- ✓ Smoother results in the data region
- ✓ More replicas satisfy post-fit requirements

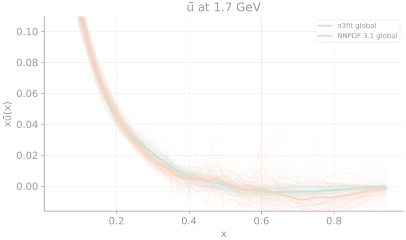
Which translates to

- ✓ Even smaller computing times!
- ✓ Many more studies can be performed at the same cost
- ✓✓ Leading to a more accurate PDF determination

Result comparison: replica-by-replica

Comparison, with the same selection of data, of the old and new codes.





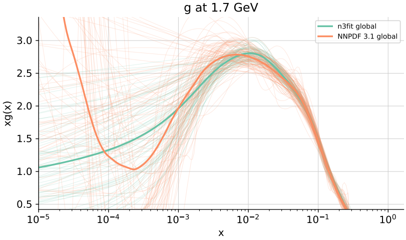
- ✓ Smoother results in the data region
- ✓ More replicas satisfy post-fit requirements

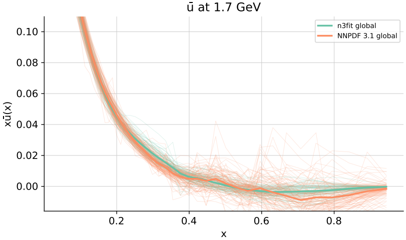
Which translates to

- ✓ Even smaller computing times!
- ✓ Many more studies can be performed at the same cost
- ✓✓ Leading to a more accurate PDF determination

Result comparison: replica-by-replica

Comparison, with the same selection of data, of the old and new codes.





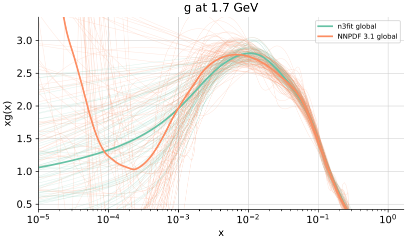
- ✓ Smoother results in the data region
- ✓ More replicas satisfy post-fit requirements

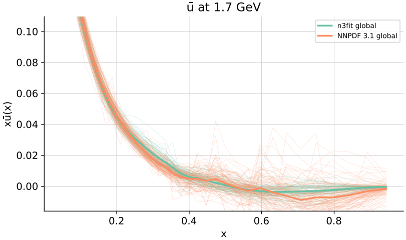
Which translates to

- ✓ Even smaller computing times!
- ✓ Many more studies can be performed at the same cost
- ✓✓ Leading to a more accurate PDF determination

Result comparison: replica-by-replica

Comparison, with the same selection of data, of the old and new codes.





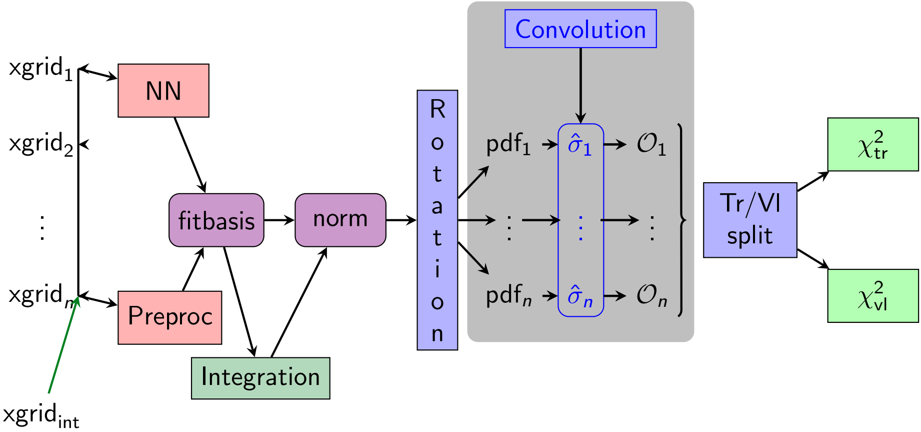
- ✓ Smoother results in the data region
- ✓ More replicas satisfy post-fit requirements

Which translates to

- ✓ Even smaller computing times!
- ✓ Many more studies can be performed at the same cost
- ✓✓ Leading to a more accurate PDF determination

Customizing the operators

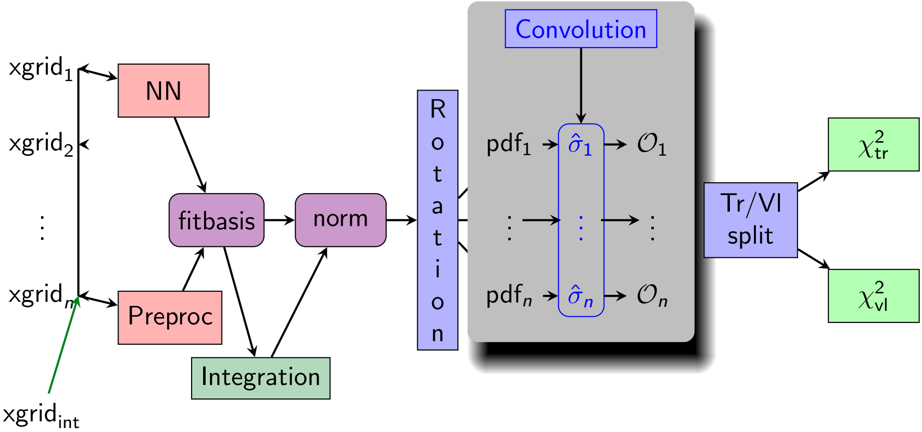
Tensorflow is very clever, but we have more information:



Customizing the operators

Tensorflow is very clever, but we have more information:

$\longrightarrow$ It is possible to hand-craft our own operators



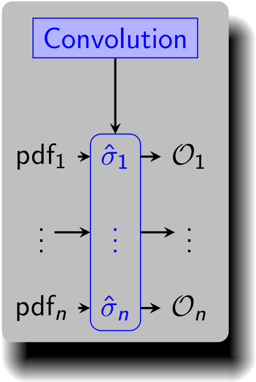
Customizing the operators

Tensorflow is very clever, but we have more information:

$\longrightarrow$ It is possible to hand-craft our own operators

	TensorFlow	Our own
Memory Total	18.4 Gb	12.5 Gb
Memory Fit	16.3 Gb	10.4 Gb

Timings are similar between the hand-crafted and the default TF convolution



As the memory is reduced we can "fit" more and more replicas in one single run: time reduction is a function of the memory.

Hardware accelerating the fits

The problem of fitting many replicas is the perfect candidate for GPU parallelization

$\longrightarrow$ Not massively CPU intensive

$\longrightarrow$ Same operations are repeated for all replicas

Example operation, contraction of rank-2 tensors: $z^N_M = x^N_\alpha y^\alpha_M$.

N	M	α	CPU AVX	TF (CPU)	TF (GPU)	OpenCL (GPU)
8	10^3	10^5	0.48	0.44	0.552	1.10
8	10^4	10^5	4.86	4.13	4.68	3.41
$8 \cdot 10^3$	10^4	10^4	48.8	1.89	1.24	15.8

Comparison on the time-cost (in seconds) per operation

CPU in table corresponds to intel i9-9980XE

GPU in table corresponds to nvidia Titan V

Summary

- ✓ **Towards NNPDF 4.0:** NNPDF machinery for PDF fitting is now more powerful, flexible and faster.

- ✓ Faster run times: iterate over different choices of models or parameters.
- ✓ The framework allows full customization *by design*.

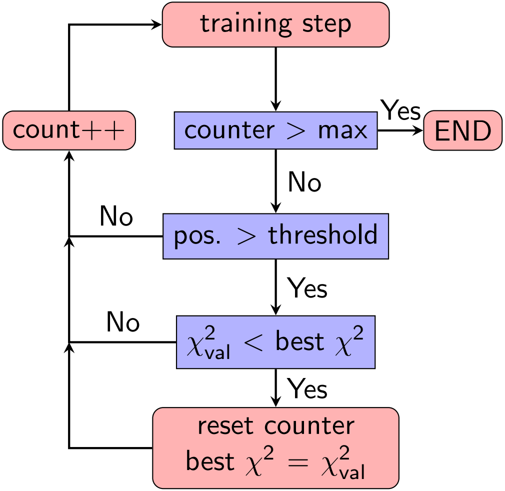
$\longrightarrow$ **The cost of doing new studies is reduced, both the development/implementation and the raw computational cost.**

Future: can we also fit using FPGAs?

Thanks!

Stopping

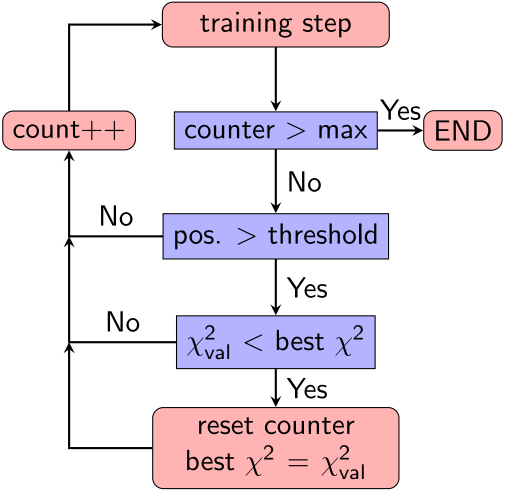
`Stopping method:` **Look-back method where positivity passes**



Stopping

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

Early stopping: reduce overfitting

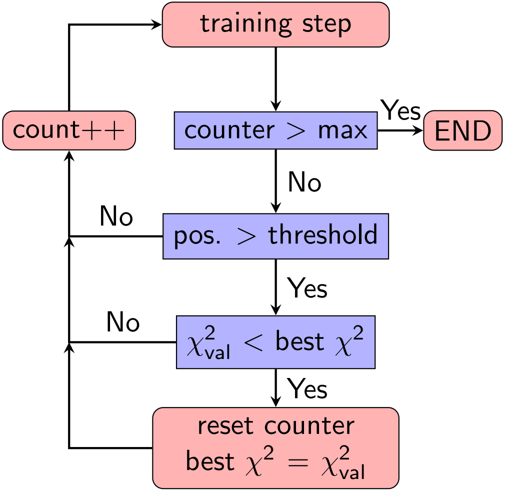


Stopping

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

Early stopping: reduce overfitting

✓ Minimize χ^2 of validation sets



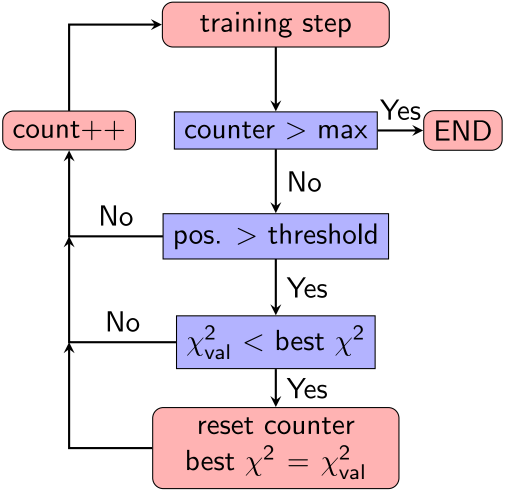
Stopping

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

Early stopping: reduce overfitting

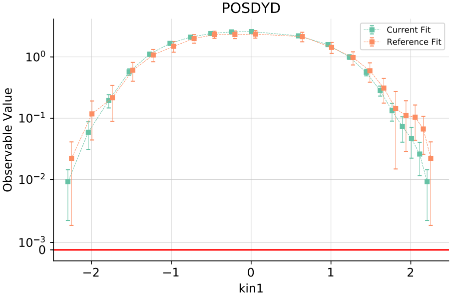
✓ Minimize χ^2 of validation sets

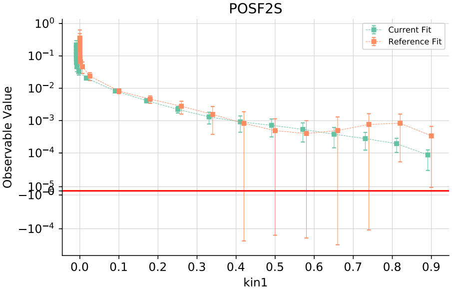
✓ Enforces positivity constraints



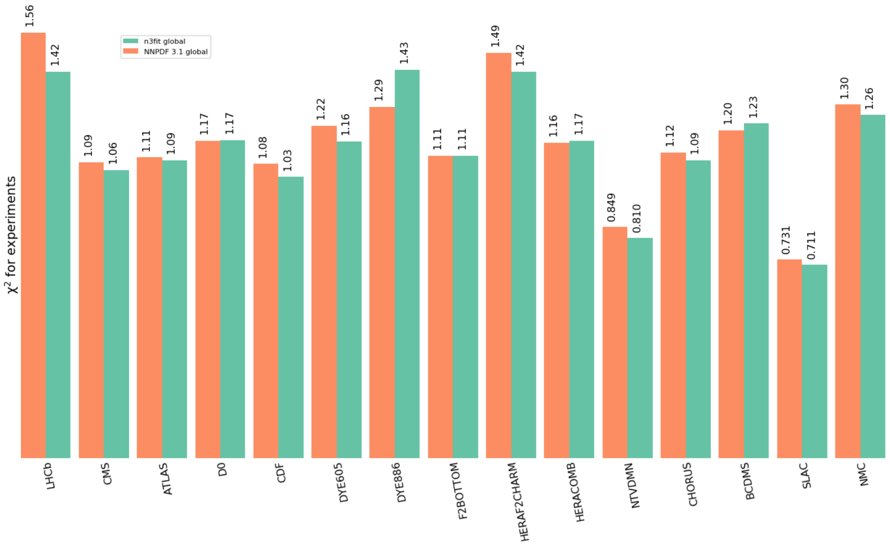
Positivity constrained

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

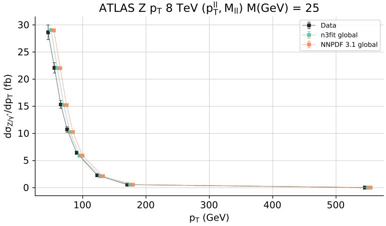


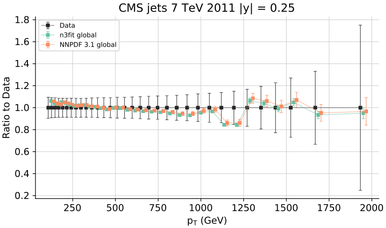


Per-experiment results



Comparison to data





→ Results compatible with NNPDF 3.1

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

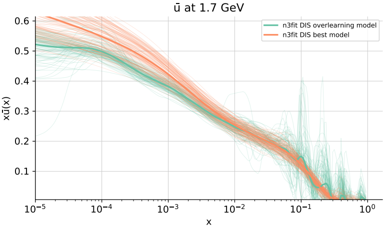
✓ The new methodology is compatible with the previous one!

Warning: overfitting!

With great power comes great responsability.

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

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

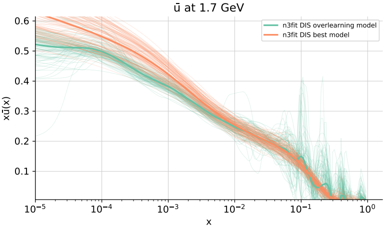


Warning: overfitting!

With great power comes great responsability.

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

- ✗ 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

The test set

The creation of a properly defined test set is quite a convoluted task. For [hep-ph/1907.05075] we have restricted ourself to the following two items:

- Redundant datasets: we select processes with more than one dataset of experimental data..
- Smaller kinetic range: of the redundant datasets we select the one that covers a smaller kinematic range (in practice, we take out the one whose x_{min} is bigger).

Finally the hyperoptimization itself is performed on a combination of the validation loss of each fit and the χ^2 of the fit to the testing set. Furthermore the fits are tested for stability in order to remove potentially