Learning parton densities with neural networks: The NNPDF methodology

IML Machine Learning Working Group, CERN

Zahari Kassabov

June 16th 2017

University of Turin, University of Milan

The problem

- We want to study the Standard Model and eventually find deviations.

The problem

- We want to study the Standard Model and eventually find deviations.
- We need to compare theoretical predictions to experimental data.

The problem

- We want to study the Standard Model and eventually find deviations.
- We need to compare theoretical predictions to experimental data.
- Theory predictions at a *pp* collider:

$$\sigma_X(s, M_X^2) = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)$$

The problem

- We want to study the Standard Model and eventually find deviations.
- We need to compare theoretical predictions to experimental data.
- Theory predictions at a *pp* collider:

$$\sigma_X(s, M_X^2) = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)$$

- $\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)$ Cross sections for *partons a, b* interacting at $\hat{s} = x_1 x_2 M_X^2$ to produce the final state X at characteristic scale M_X. Can be calculated in perturbation theory.

The problem

- We want to study the Standard Model and eventually find deviations.
- We need to compare theoretical predictions to experimental data.
- Theory predictions at a *pp* collider:

$$\sigma_X(s, M_X^2) = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \hat{\sigma}_{a,b \to X}(x_1 x_2 s, M_X^2)$$

- $\hat{\sigma}_{a,b \to X}(x_1 x_2 s, M_X^2)$ Cross sections for *partons* a, b interacting at $\hat{s} = x_1 x_2 M_X^2$ to produce the final state X at characteristic scale M_X. Can be calculated in perturbation theory.
- $f_i(x, M_X^2)$ PDF of parton i carrying a fraction of momentum x at scale M_X. Needs to be *learned* from data.

The solution

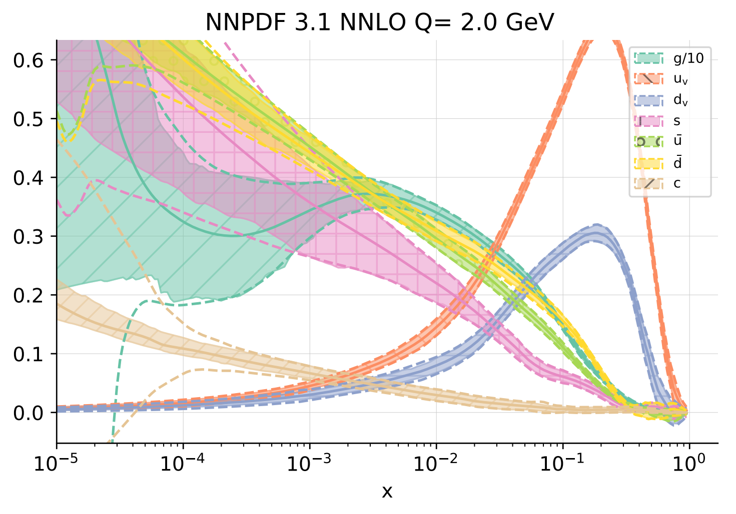


Figure 1: NNPDF Collaboration, arxiv:1706.00428

This talk

I will describe the NNPDF methodology for determining PDFs

- As an application of Machine Learning: How it compares to other problems.
- Possible ways to improve it.

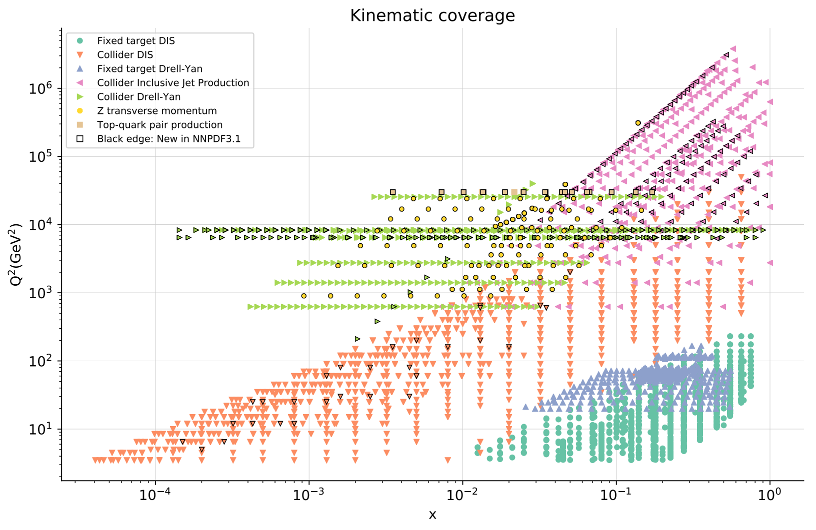
Uncertainties

- We not only want to determine the PDFs, but also provide a sensible estimate of the uncertainty.
- Sources of uncertainty:
    - Uncertainties in input experimental data (part of the input: We have the covariance matrix).
    - Degenerate minima (+inefficiencies in the minimization).
    - Theoretical uncertainties (value of α_s, fixed order calculations, etc)

Uncertainties

- We not only want to determine the PDFs, but also provide a sensible estimate of the uncertainty.
- Sources of uncertainty:
  - Uncertainties in input experimental data (part of the input: We have the covariance matrix).
  - Degenerate minima (+inefficiencies in the minimization).
  - Theoretical uncertainties (value of α_s, fixed order calculations, etc)
- Not a well researched topic in ML.

Experimental inputs (an oversimplification)



Experimental Inputs

$$\underbrace{\sigma_X(s, M_X^2)}_{Y} = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \underbrace{\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)}_{X}$$

- Constraints come in the form of *convolutions*:

$$\cancel{f(X) \longrightarrow Y}$$

Experimental Inputs

$$\underbrace{\sigma_X(s, M_X^2)}_{Y} = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \underbrace{\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)}_{X}$$

- Constraints come in the form of *convolutions*:

$$f \otimes X \longrightarrow Y$$

Experimental Inputs

$$\underbrace{\sigma_X(s, M_X^2)}_{Y} = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \underbrace{\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)}_{X}$$

- Constraints come in the form of *convolutions*:

$$f \otimes X \longrightarrow Y$$

- 4285 data points points. Not a *big data* problem.

Experimental Inputs

$$\underbrace{\sigma_X(s, M_X^2)}_{Y} = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \underbrace{\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)}_{X}$$

- Constraints come in the form of *convolutions*:

$$f \otimes X \longrightarrow Y$$

- 4285 data points points. Not a *big data* problem.
- 7 physical processes from 14 experiments over ~30 years: Need to deal with inconsistencies.

Experimental Inputs

$$\underbrace{\sigma_X(s, M_X^2)}_{Y} = \sum_{a,b} \int_{x_{min}}^{1} dx_1 dx_2 f_a(x_1, M_X^2) f_b(x_2, M_X^2) \underbrace{\hat{\sigma}_{a,b\to X}(x_1 x_2 s, M_X^2)}_{X}$$

- Constraints come in the form of *convolutions*:

$$f \otimes X \longrightarrow Y$$

- 4285 data points points. Not a *big data* problem.
- 7 physical processes from 14 experiments over ~30 years: Need to deal with inconsistencies.
- Few data constrains on high and low x: Need to deal with extrapolation.

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

$$f_a(x_1, M_X^2) f_b(x_2, M_X^2) \hat{\sigma}_{a,b\rightarrow X}(x_1 x_2 s, M_X^2)$$

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

$$f_j(x_\alpha, Q_0^2) f_k(x_\delta, Q_0^2) \Gamma(M_X^2, Q_0)_{aj\beta\alpha} \Gamma(M_X^2, Q_0)_{kj\beta\delta} \hat{\sigma}_{a,b\to X}(x_\alpha x_\beta s, M_X^2)$$

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

$$f_j(x_\alpha, Q_0) f_k(x_\delta, {Q_0}^2)\left(\Gamma(M_X^2, Q_0)_{aj\beta\alpha}\Gamma(M_X^2, Q_0)_{kj\beta\delta}\hat{\sigma}_{a,b\to X}(x_\alpha x_\beta s, M_X^2)\right)$$

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

$$f_j(x_\alpha, Q_0) f_k(x_\delta, {Q_0}^2) \left(\Gamma(M_X^2, Q_0)_{aj\beta\alpha} \Gamma(M_X^2, Q_0)_{kj\beta\delta} \hat{\sigma}_{a,b\to X}(x_\alpha x_\beta s, M_X^2) \right)$$

- Can compute the DGLAP operator and apply to the partonic cross section. APFEL, [Bertone et al, arxiv:1310.1394].

Scale dependence: The FastKernel method

Given a set PDFs at some scale Q_0, the scale dependence is given by the solution of the renormalization group equation (DGLAP):

$$f_i(x_\beta, Q^2) = \Gamma(Q, Q_0)_{ij\beta\alpha} f_j(x_\alpha, {Q_0}^2)$$

$$f_j(x_\alpha, Q_0) f_k(x_\delta, {Q_0}^2)(\Gamma(M_X^2, Q_0)_{aj\beta\alpha}\Gamma(M_X^2, Q_0)_{kj\beta\delta}\hat{\sigma}_{a,b\to X}(x_\alpha x_\beta s, M_X^2))$$

- Can compute the DGLAP operator and apply to the partonic cross section. APFEL, [Bertone et al, arxiv:1310.1394].
- Can store the result and perform much faster convolutions. APFELgrid [Bertone er al, arxiv:1605.02070].
- Only need to deal with the x dependence at some initial scale

$$f(x, Q^2) \longrightarrow f(x, Q_0^2) \coloneqq f(x)$$

Constraints on PDFs

- Sum rules:
    - $\sum_i^{\text{partons}} \int_0^1 x f_i(x) dx = 1$
    - $\int_0^1 (u(x) - \bar{u}(x)) dx = 2$
    - $\int_0^1 (d(x) - \bar{d}(x)) dx = 1$
    - $\int_0^1 (q(x) - \bar{q}(x)) dx = 0, \;\; q = s, b, t$
- Continuity:
    - $f(x) \xrightarrow{x \to 1} 0$
- ...and that's it!

Outstanding features of the problem

To recapitulate, compared to a typical ML problem:

- We require a statistically sound uncertainty estimate.
- The problem is *regression* but the available data has a complex dependence on the PDFs.
- There are some physical constraints.

Parametrizations for PDFs

- Early models:

$$f(x) = Cx^{\alpha}(1 - x)^{\beta}$$

Parametrizations for PDFs

- Early models:

$$f(x) = Cx^{\alpha}(1 - x)^{\beta}$$

  - Parameters can be chosen to satisfy the constraints.

Parametrizations for PDFs

- Early models:

$$f(x) = Cx^{\alpha}(1-x)^{\beta}$$

  - Parameters can be chosen to satisfy the constraints.
  - Can a simple model provide a reliable uncertainty? What is the "modelization" uncertainty? Is it possible to make any claims if the data doesn't fit?

Parametrizations for PDFs

- Early models:

$$f(x) = Cx^{\alpha}(1-x)^{\beta}$$

  - Parameters can be chosen to satisfy the constraints.
  - Can a simple model provide a reliable uncertainty? What is the "modelization" uncertainty? Is it possible to make any claims if the data doesn't fit?
- The NNPDF approach:

Parametrizations for PDFs

- Early models:

$$f(x) = Cx^{\alpha}(1-x)^{\beta}$$

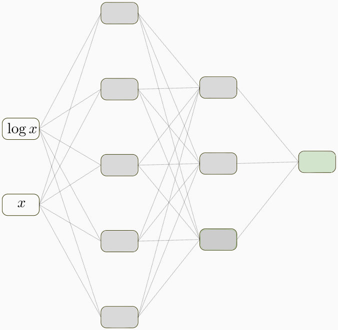
  - Parameters can be chosen to satisfy the constraints.
  - Can a simple model provide a reliable uncertainty? What is the "modelization" uncertainty? Is it possible to make any claims if the data doesn't fit?

- The NNPDF approach:
  - Since we don't have constraints, we should have a very general parametrization:

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

NN(x)

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



- Fully connected.
- Two sigmoid hidden layers.
- One linear layer.
- $\times 8$ PDF flavour combinations = 296 network parameters.

Error function

- We minimize the error function

$$\chi^2 = \sum_{ij} (D_i - O_i)\Sigma_{i,j}^{-1}(D_j - O_j)$$

  - D_i experimental measurement for point i
  - O_i prediction for point $i(=f \otimes \hat{\sigma})$.
  - $\mathbf{\Sigma}_{ij}$ covariance between points i and j (corrected for normalization uncertainties [Ball et al 2009]).
  - There is an additional penalty term for positivity observables.

Propagating experimental unccertainties

Perform N_{rep} O(1000), fits, sampling *pseudodata replicas*:

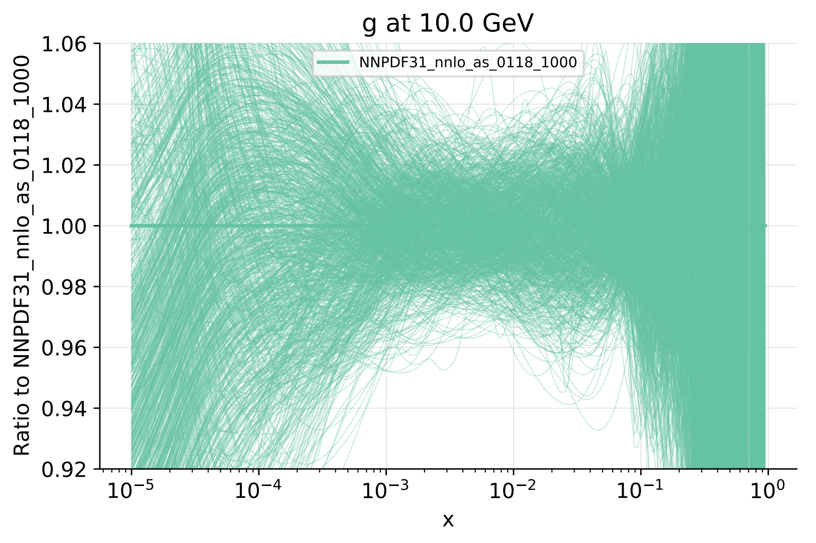
$$D_i^{(r)} \longrightarrow D_i^{(r)} + \text{chol}(\Sigma)_{i,j}\xi_j$$
$$\xi_j \sim \mathcal{N}(0,1)$$
$$i,j = 1..N_{\text{dat}}$$
$$r = 1..N_{\text{rep}}$$

Obtain N_{rep} *PDF replicas*. All statistics of the PDFs (and functions thereof) can be computed from the ensemble of PDF replicas.

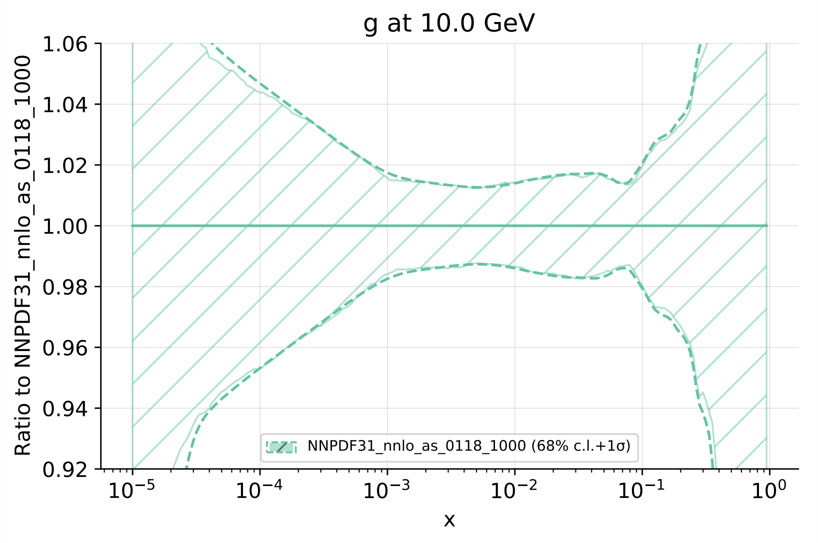
No assumptions at all about the Gaussianty of the errors. We also provide:

- Compressed Monte Carlo sets [Carrazza, et al, arxiv:1504.06469].
- Compressed [Carrazza et al, arxiv:1505.06736] and supercompressed [Carrazza et al, arxiv:1602.00005] Hessian sets.

Simple example



Simple example



Fitting NNPDFs

(Discussed in detail in [Ball et al, arxiv:1410.8849])

The current approach is a genetic algorithm. At each iteration, select a node with $P = 5\%$

$$w \to w + \frac{\eta r_\delta}{N_{\text{ite}}^{r_{\text{ite}}}}$$

$$\begin{aligned} \eta &= 15 \\ r_\delta &\sim U(-1, 1) \\ r_{\text{ite}} &\sim U(1, 0) \end{aligned}$$

At each iteration, generate 80 mutants, and select best mutant.

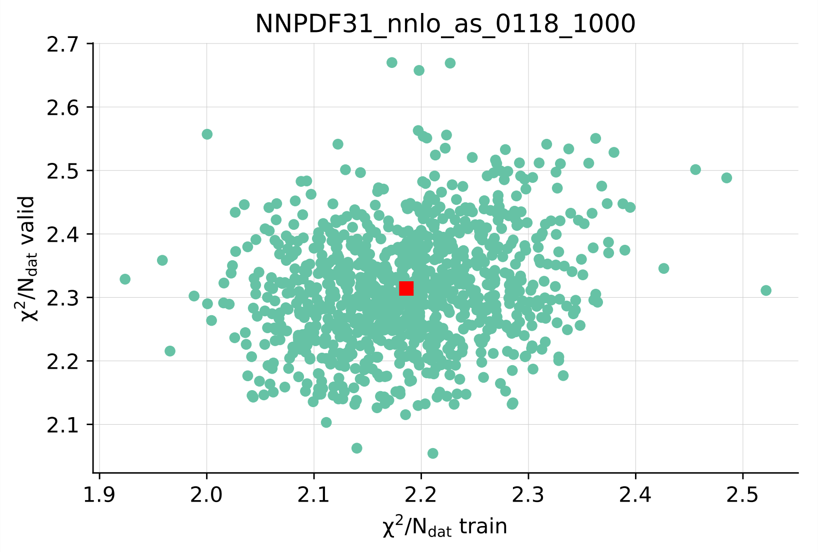
Tradeoffs of the GA

- Advantages:
  - Simple to implement and understand.
  - Good dealing with complex analytic behaviour.
  - Doesn't require evaluating the gradient.
- Disadvantages:
  - May not be close to a global minimum.
  - Requires many function evaluations (i.e. convolutions).
  - Needs tuning (discussed in later slides).

Stopping

- We split the data in a training and validation set.
- Roughly 50%, different for each replica.
- We run the GA on the training set for a fixed number of iterations $O(30000)$.
- We select the minimum of the validation set as the parameters from the replica.

Training-validation distribution



Preprocessing exponents

- We had $f(x) = \mathrm{CNN}(x) x^{\alpha} (1-x)^{\beta}$.
- α and β chosen at random with ranges set from the results of a previous fit.
- We iterate the distribution of *effective exponents* doesn't change:

$$\alpha_{\mathrm{eff}} = \left.\frac{\log f(x)}{\log x}\right|_{x\to 0}$$

$$\beta_{\mathrm{eff}} = \left.\frac{\log f(x)}{\log(1-x)}\right|_{x\to 1}$$

Closure tests

- We want to assess the validity of our procedure.
- We want to tune the parameters of the GA, architecture of the neural net, etc.
- Closure tests
    - Assume that the underlying PDF is known.
    - Generate data, fluctuating around the prediction of the true PDF.
    - Perform a fit and compare with assumed PDF.
    - Check that the results are consistent.

Levels of closure tests

Level 0 Fit predictions of the underlying PDF without fluctuations.

- Expect $\chi^2/N_{\text{dat}} = 0$.
- Tests the adequacy of the architecture and the GA.

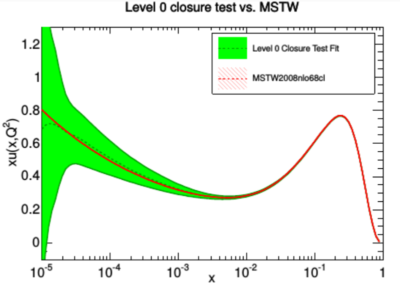
Level 1 Fit fluctuations using the experiment covariance matrix.

- Expect $\chi^2/N_{\text{dat}} = 1$.
- Test stability of central values.

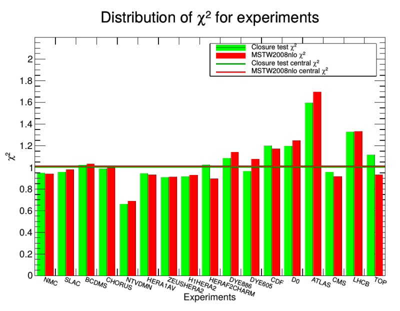
Level 2 Generate pseudodata replicas on top of the replicas.

- Expect $\chi^2/N_{\text{dat}} = 2$ for data replicas.
- Expect $\chi^2/N_{\text{dat}} = 1$ for data central values.
- Validate the full procedure.

Closure test results (using MSTW2008 as Truth)



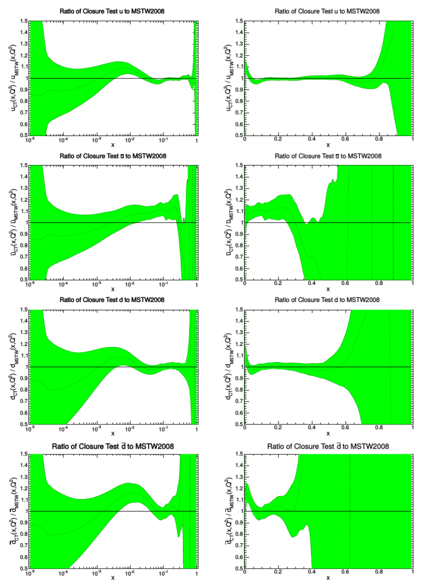
(a) Level 0 fit



(b) Level 2 results

- Level 0 fit perfectly consistent: We reproduce the truth value as much as possible with the available data.
- At Level 2 (equivalent to our methodology) we reproduce exactly the fluctuated χ^2s.

Closure tests vs Truth



If MSTW2008 was the truth, we would reproduce it within uncertainties!

Summary: NNPDF methodology

- Parametrize the NNPDFs with neural networks.
- Propagate uncertainties by fitting to *pseudodata replicas*.
- Fit using a Genetic algorithm
- Use cross validation to avoid overfitting.
- Closure tests to validate the methodology.

Thank you!

Backup: Code

- In house C++ with increasing supporting code in Python.
- Core loop (convolution) written in assembly.
- Memory layout optimized.
- APFEL used for evolution.