

Theory Predictions in PDF Fitting

Felix Hekhorn

CFNS 2022, Stony Brook, August 2022

Acknowledgement: This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement number 740006.

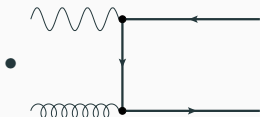


1. Motivation
2. PineAPPL [JHEP12.108]
3. EKO [2202.02338]
4. Evidence for Intrinsic Charm in the Proton [in print]
5. yadism [in preparation]
6. Outlook

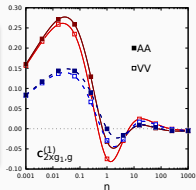
1. Motivation

Including New Computation

- Computing new observables is expensive both in runtime (days/weeks) and development time (month/years)
- E.g. NLO heavy quark production in polarized DIS [PRD98.014018] [1910.01536] [PRD104.016033]

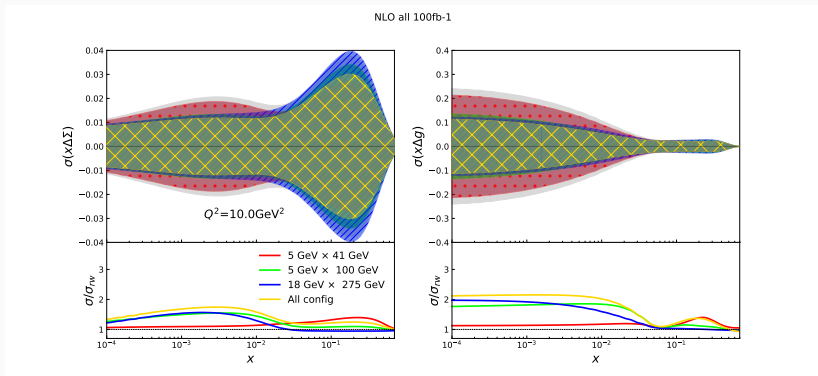


```
File Edit Shell View Help
m001 f1|gofft--MC_00k_f2_fm0k_m0_c01_qp_c01_qp_c0k_f1_c0k_v1_c0k
fm_00k_wp |
[...]
```



- How to measure the actual impact on PDFs?

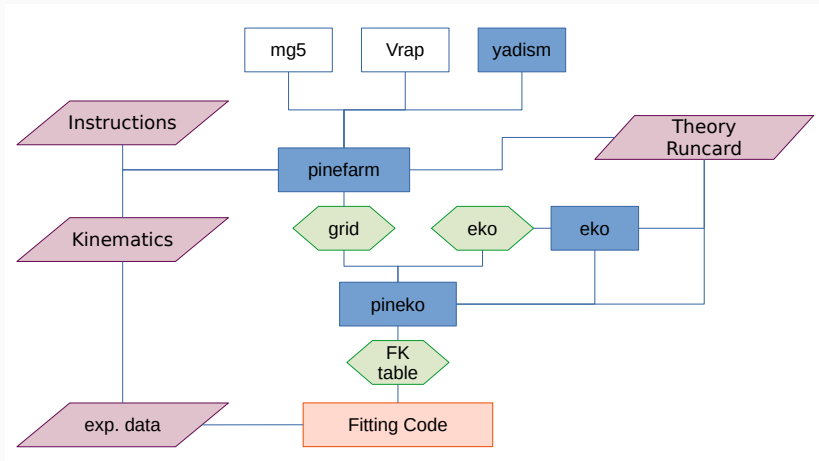
Reweighting is possible [PRD104.114039] - and even needed for the EIC



but a new PDF fit would be better!

New Theory Prediction Pipeline

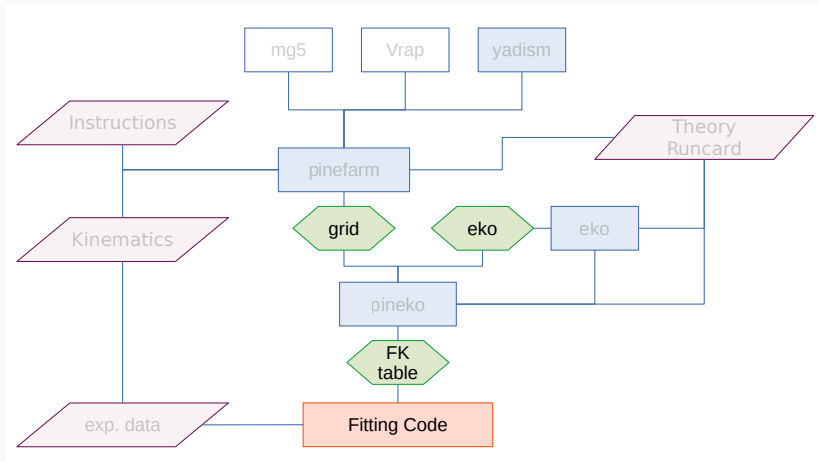
Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL

New Theory Prediction Pipeline

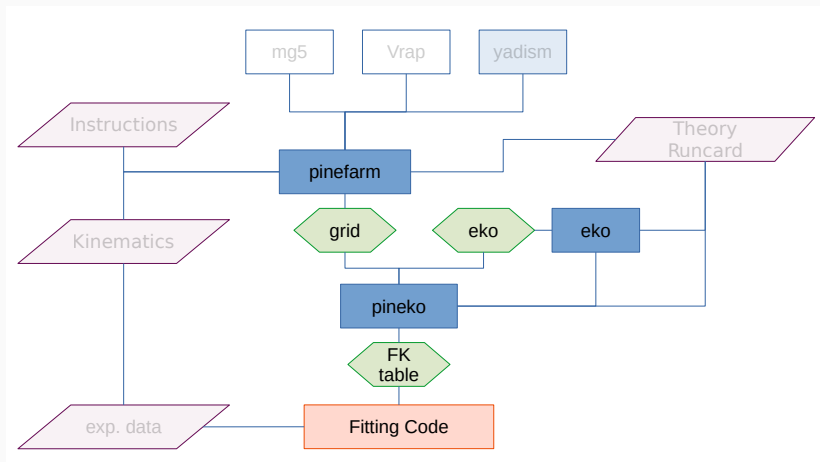
Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL

New Theory Prediction Pipeline

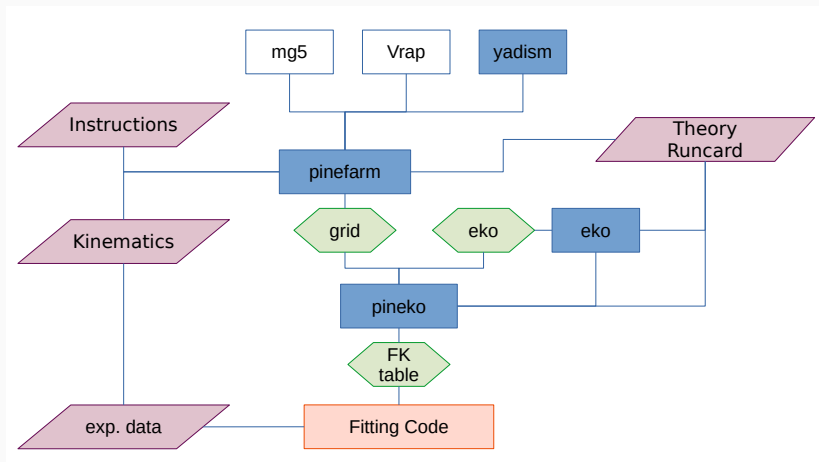
Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL

New Theory Prediction Pipeline

Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL

2. PineAPPL [JHEP12.108]

PineAPPL is a fast interpolation grid library that

- extends to arbitrary orders in QCD and EW coupling
- provides a very good CLI
- provides several interfaces: Rust, Python, C/C++, Fortran
- can convert APPLgrid [EPJC66.503] and FastNLO [DIS12.217]

Check the documentation: <https://n3pdf.github.io/pineappl/>

The Command Line Interface

The CLI (`pineappl`) serves for the everyday life questions:

- `convolute` - get the predictions for any PDF set including uncertainties
- `channels` - split the predictions into luminosity channels
- `orders` - split the predictions into perturbative orders
- `info` - access meta data
- `plot` - generate a (customizable) python plot script

The Command Line Interface

```
$ pineappl convolute CMS_DY_14TEV_MLL_5000_COSTH.pineappl.lz4 \  
NNPDF40_nnlo_as_01180
```

```
b          costh      dsig/dcosth  scale uncertainty  
          []          [pb]          [%]
```

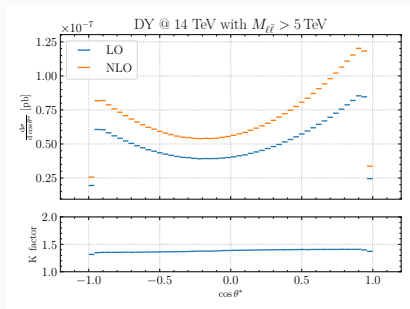
```
-----+-----+-----+-----+-----
```

0	-1	-0.96	5.0382145e-8	-4.73	4.32
1	-0.96	-0.92	1.6366674e-7	-4.98	4.62
2	-0.92	-0.88	1.6611145e-7	-5.06	4.70
3	-0.88	-0.84	1.5983761e-7	-5.08	4.74
4	-0.84	-0.8	1.5374426e-7	-5.09	4.75
5	-0.8	-0.76	1.4800320e-7	-5.10	4.76
6	-0.76	-0.72	1.4238050e-7	-5.10	4.76
7	-0.72	-0.68	1.3708378e-7	-5.10	4.76
8	-0.68	-0.64	1.3191722e-7	-5.12	4.78
9	-0.64	-0.6	1.2720559e-7	-5.13	4.79

```
[...]
```

The Python Interface

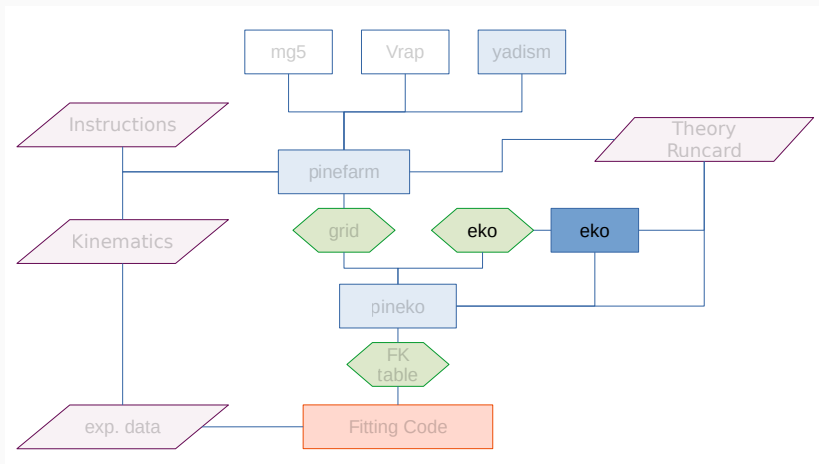
```
1 import lhpdf
2 import pineappl
3 # load PDF
4 pdf = lhpdf.mkPDF("
      NNPDF40_nnlo_as_01180",0)
5 # load grid
6 grid = pineappl.grid.Grid.read(
      "
      CMS_DY_14TEV_MLL_5000_COSTH
      .pineappl.lz4")
7 # convolute
8 print(grid.convolute_with_one(
      (2212, pdf.xfxQ2, pdf.
      alphasQ2))
9 # prints the same list of
      numbers
```



3. EKO [2202.02338]

New Theory Prediction Pipeline

Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL



DGLAP:

$$\mu_F^2 \frac{d\mathbf{f}}{d\mu_F^2}(\mu_F^2) = \mathbf{P}(a_s(\mu_R^2), \mu_F^2) \otimes \mathbf{f}(\mu_F^2)$$

as operator equation for the evolution kernel operator (EKO) \mathbf{E} :

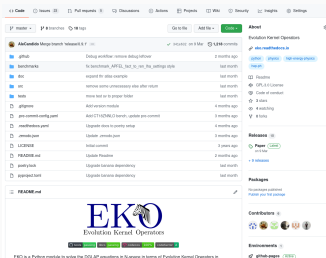
$$\mu_F^2 \frac{d}{d\mu_F^2} \mathbf{E}(\mu_F^2 \leftarrow \mu_{F,0}^2) = \mathbf{P}(a_s(\mu_R^2), \mu_F^2) \otimes \mathbf{E}(\mu_F^2 \leftarrow \mu_{F,0}^2)$$

with

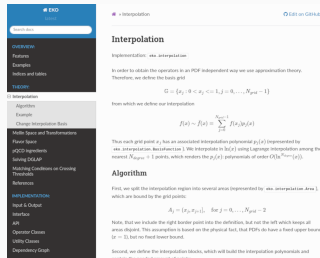
$$\mathbf{f}(\mu_F^2) = \mathbf{E}(\mu_F^2 \leftarrow \mu_{F,0}^2) \otimes \mathbf{f}(\mu_{F,0}^2)$$

- independent of boundary condition \rightarrow PDF fitting
- Mellin (N -) space solution, but momentum (x -) space delivery via piecewise Lagrange-interpolation
- Intrinsic heavy quark distributions \rightarrow see IC
- Backward VFNS evolution (i.e. across thresholds and with intrinsic) \rightarrow see IC

EKO Project Management



The screenshot shows the GitHub repository page for 'eko' by 'N3PDF'. The repository is described as 'Evolution Kernel Operators'. It lists several files and folders, including 'gitlab', 'LICENSE', 'README.md', 'src', 'tests', 'programs', 'pre-compiled-wheel', 'readthedocs.yml', 'conda.yml', 'CMakeLists.txt', 'README-std', 'pybind11', and 'pybind11-std'. The sidebar on the right shows repository statistics, including 'Evolution Kernel Operators' and 'Contributors'. The EKO logo is displayed at the bottom of the repository page.



The screenshot shows the documentation page for 'Interpolation' in the EKO project. The page is titled 'Interpolation' and includes a sub-section 'Implementation: via approximation'. It explains the process of obtaining operators in an PDF independent way using approximation theory. It defines the basis and provides the formula for the interpolation polynomial $p_j(x)$. The page also includes a section for the 'Algorithm' and a note about the grid points x_j .

Interpolation

Implementation: via approximation

In order to obtain the operators in an PDF independent way we use approximation theory. Therefore, we define the basis and

$$\mathbb{G} = \{x_j \mid 0 < x_j < 1, j = 0, \dots, N_{grid} - 1\}$$

from which we define our interpolation

$$f(x) = f(x_j) = \sum_{j=0}^{N_{grid}-1} f(x_j) \phi_j(x)$$

Thus each grid point x_j has an associated interpolation polynomial $p_j(x)$ (represented by `via_interpolation_basis` function 1. We interpolate in $\mathbb{R}^d(x)$ using Lagrange interpolation among the nearest $N_{grid} = 1$ points, which renders the $p_j(x)$ (polynomials of order $O(N_{grid}^{d-1})$).

Algorithm

First, we split the interpolation region into several areas (represented by `via_interpolation_area`), which are located by the grid points:

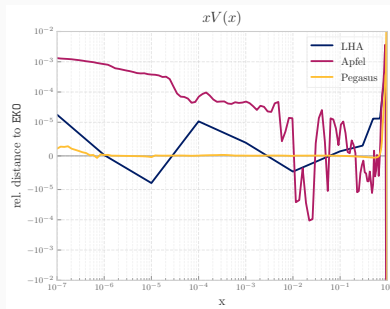
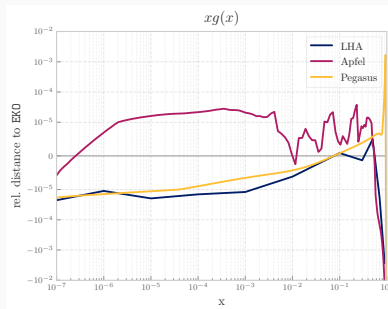
$$A_j = [x_j, x_{j+1}], \quad \text{for } j = 0, \dots, N_{grid} - 2$$

Note, that we include the right border point into the definition, but not the left which keeps all areas disjoint. This assumption is based on the physical fact, that PDFs do have a fixed upper bound ($\Omega = \Omega$) but no fixed lower bound.

Second, we define the interpolation blocks, which will build the interpolation polynomials and contain the central amount of weights.

- Fully open source: <https://github.com/N3PDF/eko>
- Written in Python
- Fully documented: <https://eko.readthedocs.io/>

LHA benchmark [0204316][0511119]:

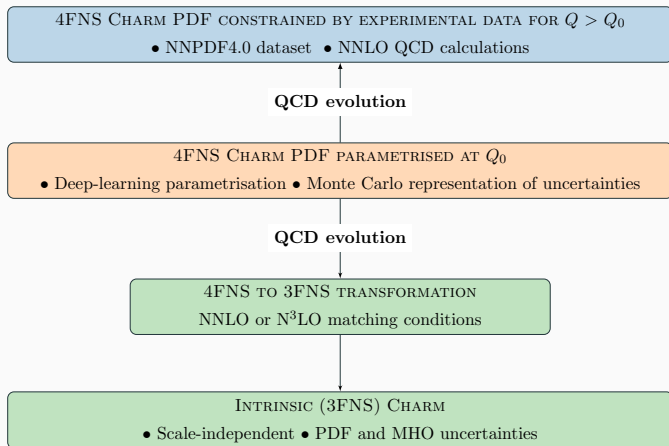


⇒ EKO is working!

4. Evidence for Intrinsic Charm in the Proton `[in print]`

Intrinsic Charm: Strategy

based on NNPDF4.0 [EPJC82.428]



Matching Conditions and Backward Evolution

For (forward) evolution across a matching scale μ_h^2 :

$$\tilde{\mathbf{f}}^{(n_f+1)}(\mu_{F,1}^2) = \tilde{\mathbf{E}}^{(n_f+1)}(\mu_{F,1}^2 \leftarrow \mu_h^2) \mathbf{R}^{(n_f)} \tilde{\mathbf{A}}^{(n_f)}(\mu_h^2) \tilde{\mathbf{E}}^{(n_f)}(\mu_h^2 \leftarrow \mu_{F,0}^2) \\ \times \tilde{\mathbf{f}}^{(n_f)}(\mu_{F,0}^2)$$

with $\mathbf{R}^{(n_f)}$ a flavor rotation matrix and $\tilde{\mathbf{A}}^{(n_f)}(\mu_h^2)$ the operator matrix elements (partially known up to N³LO)

Matching Conditions and Backward Evolution

For (forward) evolution across a matching scale μ_h^2 :

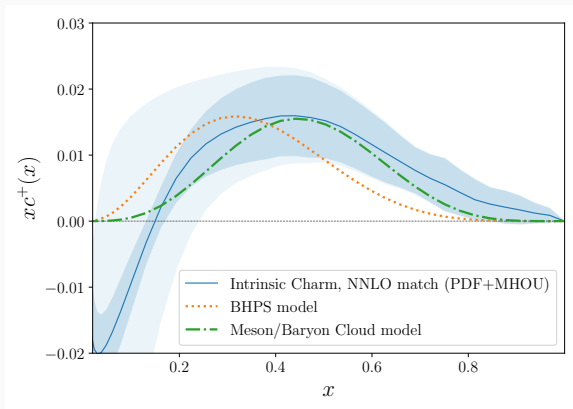
$$\tilde{\mathbf{f}}^{(n_f+1)}(\mu_{F,1}^2) = \tilde{\mathbf{E}}^{(n_f+1)}(\mu_{F,1}^2 \leftarrow \mu_h^2) \mathbf{R}^{(n_f)} \tilde{\mathbf{A}}^{(n_f)}(\mu_h^2) \tilde{\mathbf{E}}^{(n_f)}(\mu_h^2 \leftarrow \mu_{F,0}^2) \times \tilde{\mathbf{f}}^{(n_f)}(\mu_{F,0}^2)$$

with $\mathbf{R}^{(n_f)}$ a flavor rotation matrix and $\tilde{\mathbf{A}}^{(n_f)}(\mu_h^2)$ the operator matrix elements (partially known up to N³LO)

for backward evolution:

- invert $\tilde{\mathbf{E}}^{(n_f)}$: simple
- invert $\mathbf{R}^{(n_f)}$: simple
- invert $\tilde{\mathbf{A}}^{(n_f)}$: expanded or exact

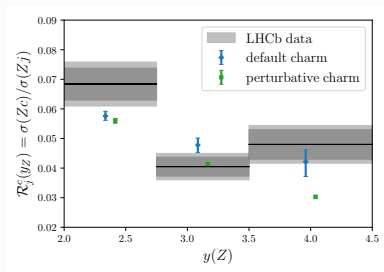
Intrinsic Charm: PDF plot



[BHPS] or [Meson/Baryon Cloud Model]

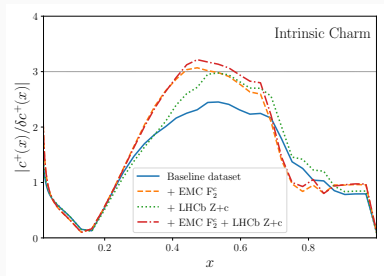
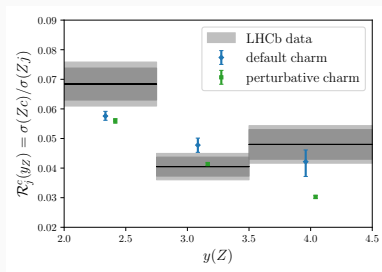
- in **3FNS** a valence-like peak is present
- for $x \leq 0.2$ the perturbative uncertainties are quite large
- the carried momentum fraction is within **1%**

Intrinsic Charm: LHCb and Significance



- match better recent **LHCb** $Z+c$ measurement [[PRL128.082001](#)]

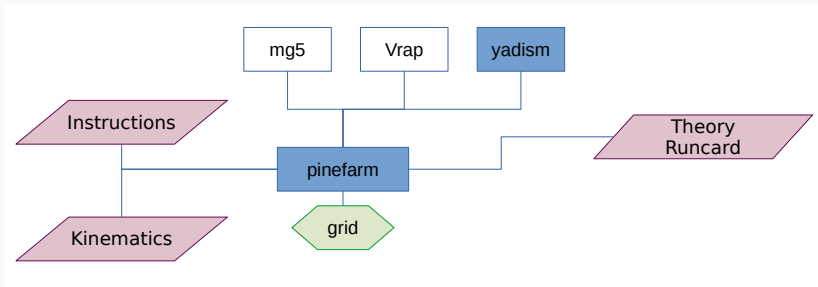
Intrinsic Charm: LHCb and Significance



- match better recent **LHCb** Z+c measurement [[PRL128.082001](#)]
- we find a **3 σ** evidence of intrinsic charm
- result is **stable** with mass variation, dataset variation

5. yadism [in preparation]

Interface to Other Programs: pinefarm



```
.....
WELCOME to MADGRAPH5_aMC@NLO
.....
      .
     . .
    . . .
   . . . .
  . . . . .
 . . . . .
. . . . .
.....

The MadGraph5_aMC@NLO Development Team - Find us at
https://server06.fynu.ucl.ac.be/projects/madgraph
and
http://amcatnlo.cern.ch

Code download from:
https://launchpad.net/madgraph5

Please refer to: MadGraph5_aMC@NLO paper
J. Alwall et al.
arXiv:1405.0301, JHEP 1407 (2014) 079
.....
```

[JHEP07.079]



[in preparation]

Vrap [PRD69.094008]

more MC interfaces
coming soon



- DIS coefficient function database
- independent of boundary condition \rightarrow PDF fitting
- separate features: TMC, FNS
- constant benchmark against APFEL

same improvement in terms of project management as EKO!

Coefficient Functions

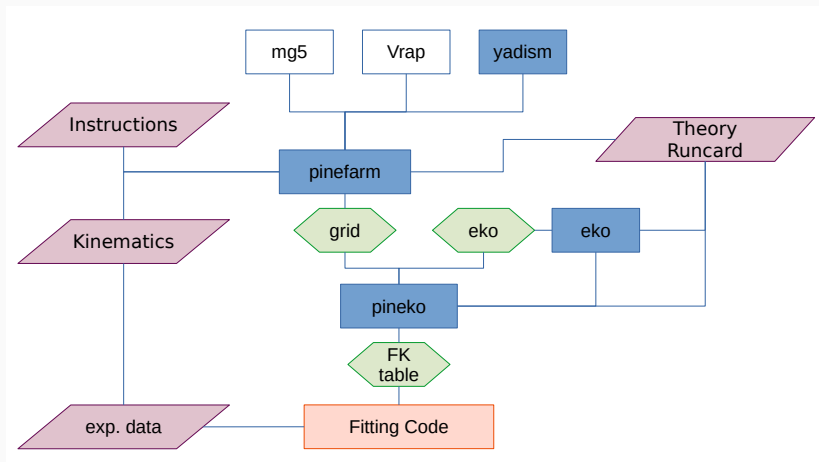
- implemented coefficient functions:

	light	heavy	intrinsic
NC	$O(a_s^2)$ [VVM05,MVV05,MV00]	$O(a_s^2)$ [Hek19]	$O(a_s)$ [KS98]
CC	$O(a_s^2)$ [MRV08,MVV09]	$O(a_s)$ [GKR96]	$O(a_s)$ [in prep.]

- implemented flavor number schemes: FFNS, ZM-VFNS, FONLL

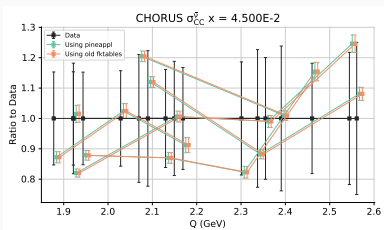
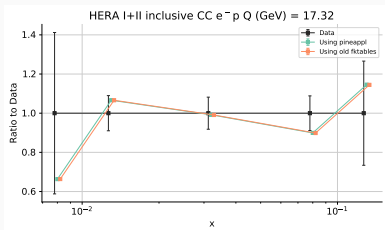
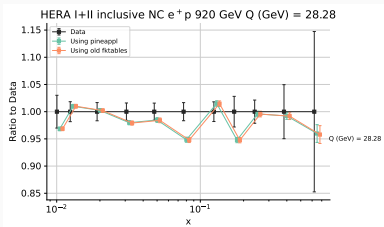
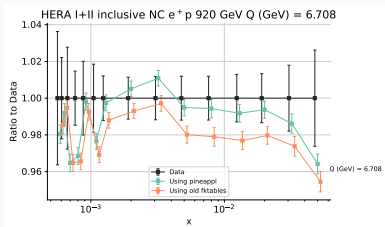
New Theory Prediction Pipeline

Produce FastKernel (FK) tables!



The workhorse in the background: PineAPPL

Comparison yadism against APFEL



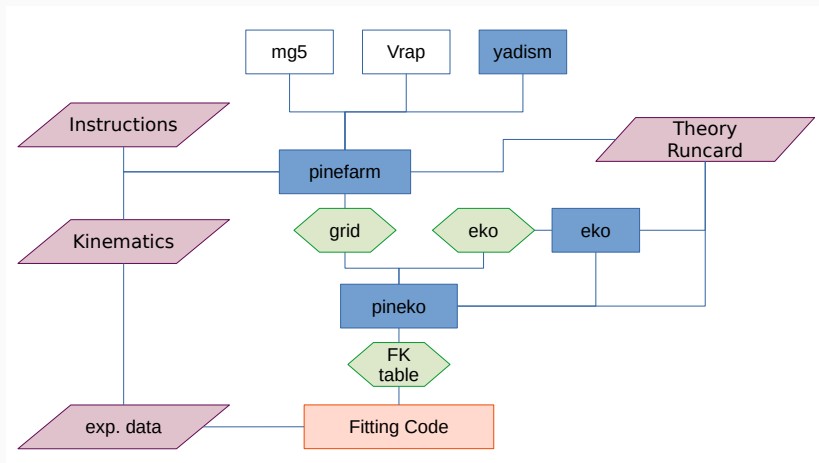
green, “pineappl” = yadism vs. orange, “old” = APFEL

6. Outlook

- extend to N3LO
- include MHO
- include QED corrections
- add polarized setup
- extend to fragmentation function
- ...

New Theory Prediction Pipeline

Produce FastKernel (FK) tables!

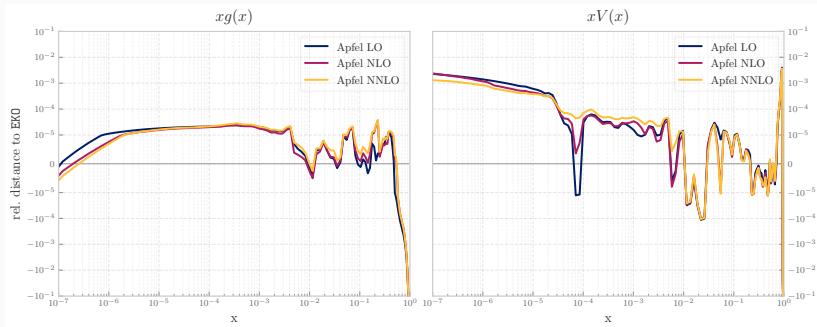


The workhorse in the background: PineAPPL

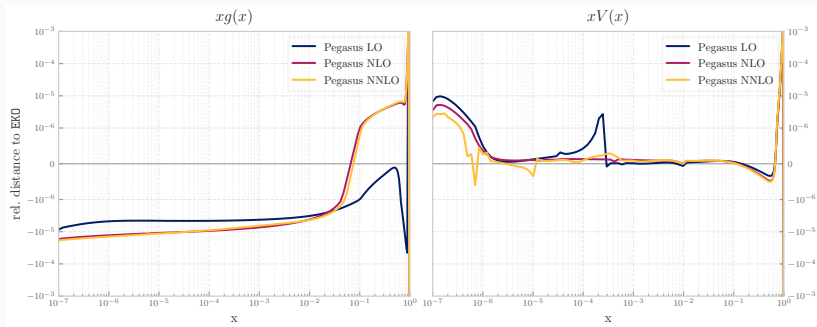
Thank you!

7. Backup slides

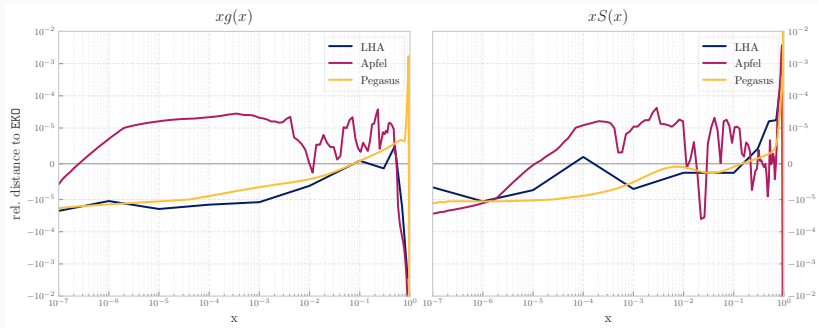
EKO APFEL benchmark



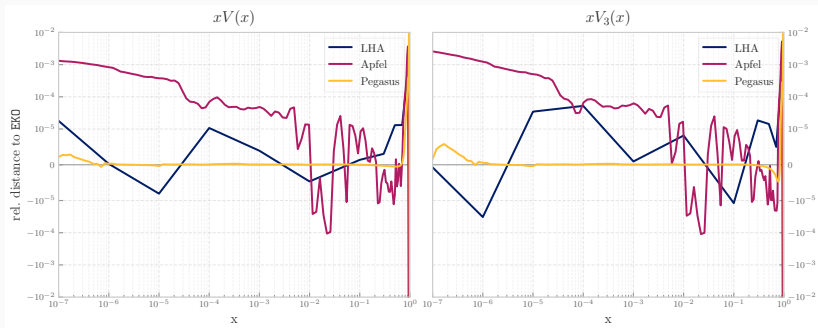
EKO PEGASUS benchmark



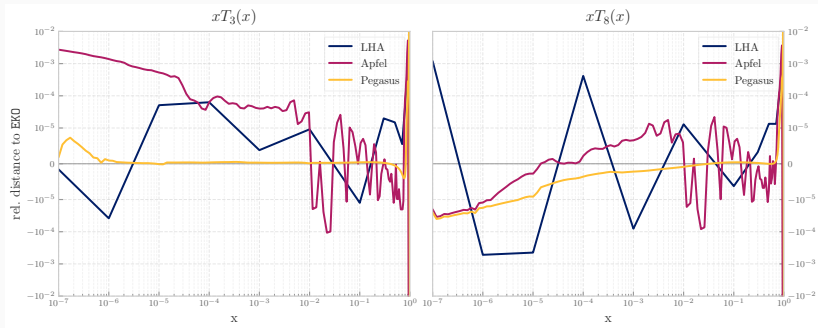
EKO LHA benchmark: g and Σ



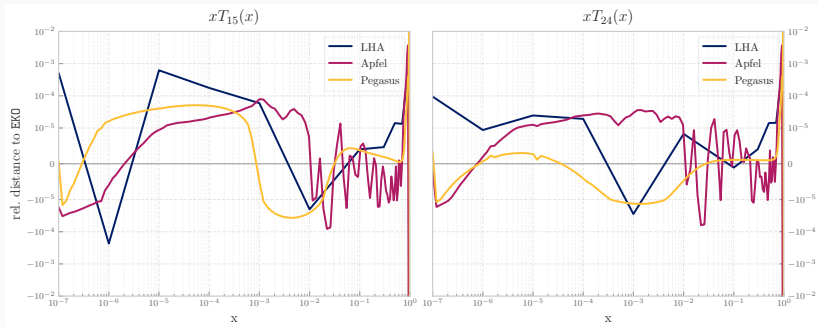
EKO LHA benchmark: V and V_3



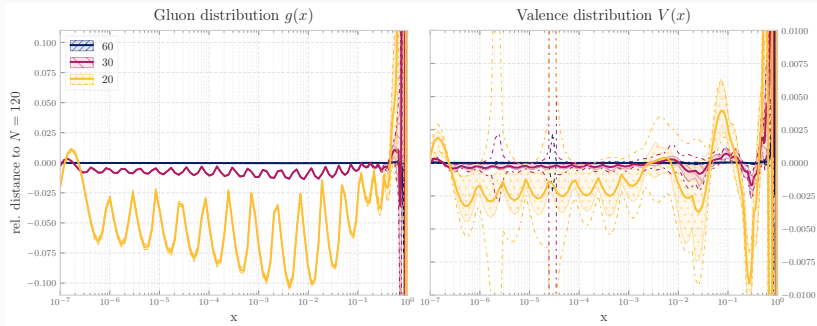
EKO LHA benchmark: T_3 and T_8



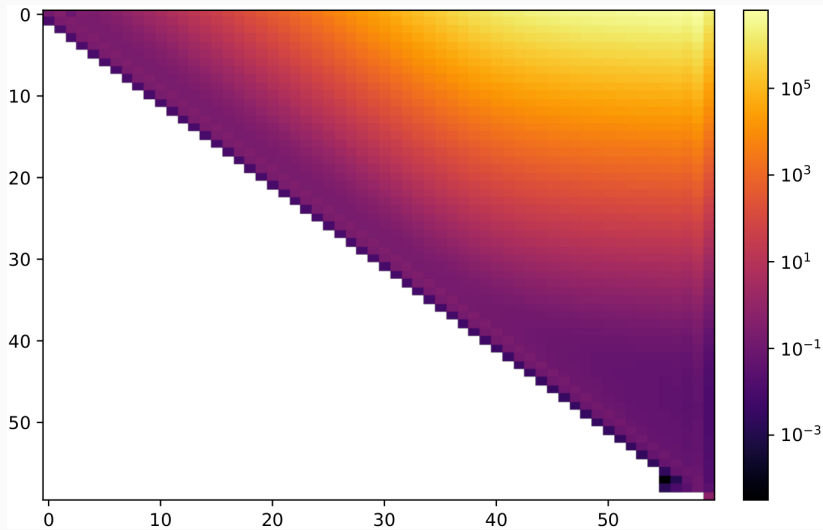
EKO LHA benchmark: T_{15} and T_{24}



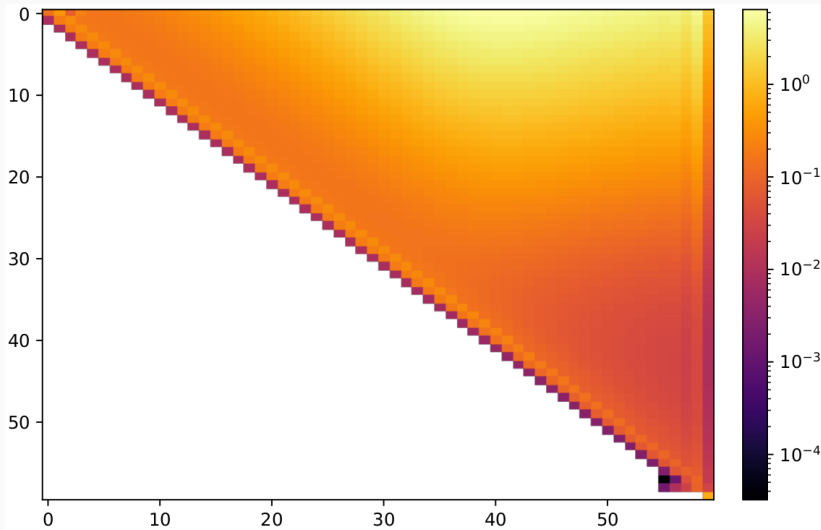
EKO Interpolation Error



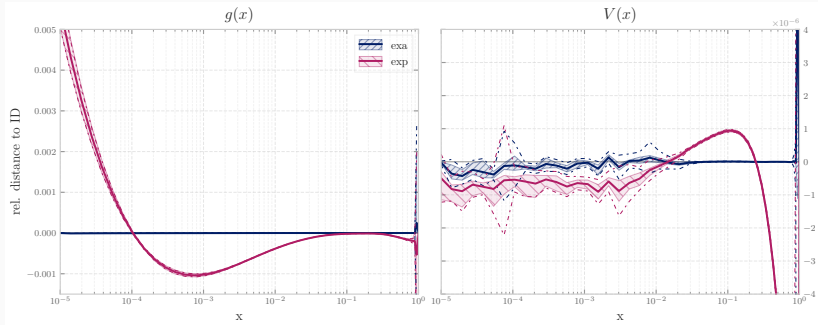
EKO Snapshot $S \leftarrow S$



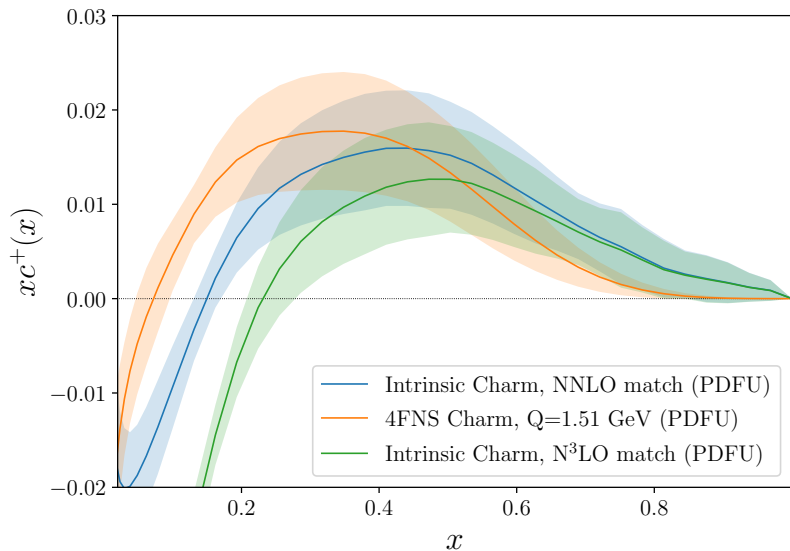
EKO Snapshot $V \leftarrow V$



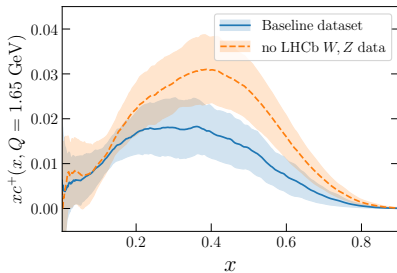
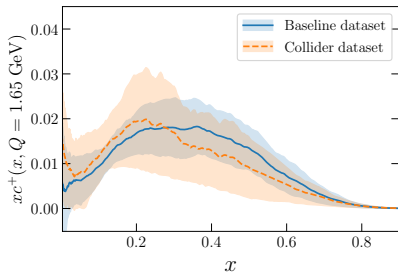
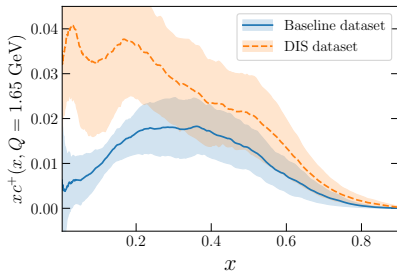
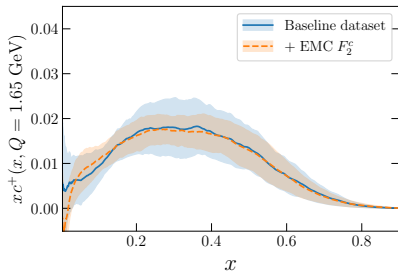
EKO Backward Evolution



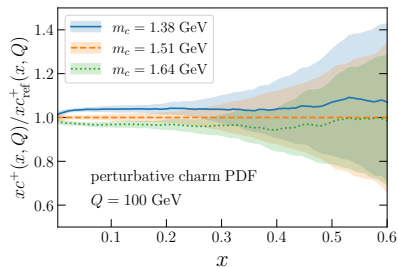
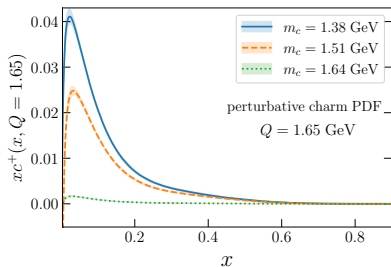
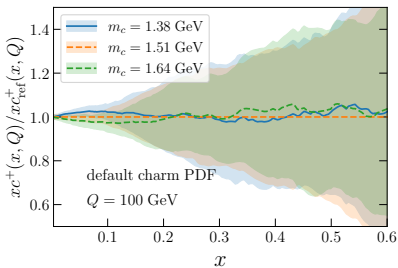
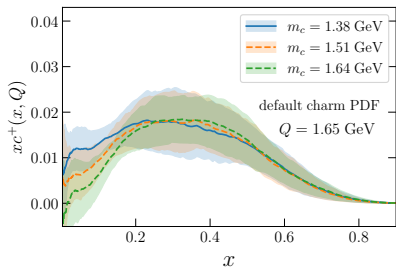
IC - uncertainties splitted



IC - dataset variation

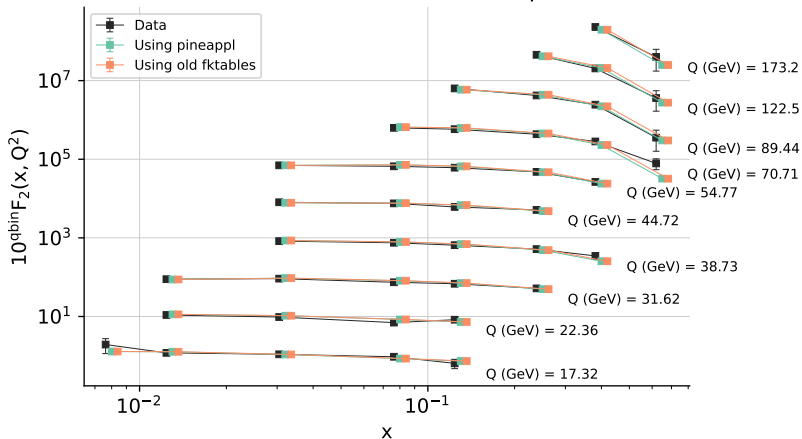


IC - mass variation



Comparison yadism against APFEL

HERA I+II inclusive CC e^-p



Comparison yadism against APFEL

