Theory predictions in PDF fitting

Andrea Barontini[1], Alessandro Candido[1], Juan M. Cruz-Martinez[2], Felix Hekhorn[1], Christopher Schwan[3]

[1] TIF Lab, Dipartimento di Fisica, Università degli Studi di Milano and INFN Sezione di Milano, Via Celoria 16, 20133, Milano, Italy
[2] CERN, Theoretical Physics Department, CH-1211 Geneva 23, Switzerland
[3] Universität Würzburg, Institut für Theoretische Physik und Astrophysik, 97074 Würzburg, Germany

Abstract. Continuously comparing theory predictions to experimental data is a common task in analysis of particle physics such as fitting parton distribution functions (PDFs). However, typically, both the computation of scattering amplitudes and the evolution of candidate PDFs from the fitting scale to the process scale are non-trivial, computationally intensive tasks. We develop a new stack of software tools that aim to facilitate the theory predictions by computing FastKernel (FK) tables that reduce the theory computation to a linear algebra operation. Specifically, I present PineAPPL, our workhorse for grid operations, EKO, a new DGLAP solver, and yadism, a new DIS library. Alongside, I review several projects that become available with the new tools.

1. Introduction

Parton distribution functions [1] (PDFs) describe the dynamics of the elementary partons, such as quarks and gluon, inside hadrons, such as the proton. PDFs are defined in factorization theorems in high-energy scattering and they encode the non-perturbative physics reigning inside hadrons. The extraction of PDFs relies on three pillars: first, the precise measurements of observables at experiments, such as the LHC, second, the reliable theoretical predictions of the associated observable, and, third, combining the former two inside a fitting framework. While either of these three steps poses several computational challenges, we discuss here only the efficient computation of the theory predictions.

The computation of the various high-energy observables that enter a PDF fit are pursued by different groups applying a range of strategies using several dedicated programs tailored to the specific case at hand. Yet, in a PDF fit one wants to include *consistently* all predictions available, which can be up to 4500 data points across almost 100 different datasets (as is the case, e.g., in [2]).

To achieve this goal we develop a framework of software tools, dubbed `pineline` [3], that provides an easily accessible way to include theory predictions in QCD fitting applications such as PDF fits. We put an explicit emphasis on the scalability of the procedure to ensure future measurements from new or existing experiments [4, 5, 6] can be included seamlessly. We also strive to track all runcards and meta data in the generated objects to ensure, we are able to reproduce the existing results. Finally, we stress that all participating programs are developed Open Source to facilitate the interaction with users and developers (following the efforts of the NNPDF fitting code [7]).

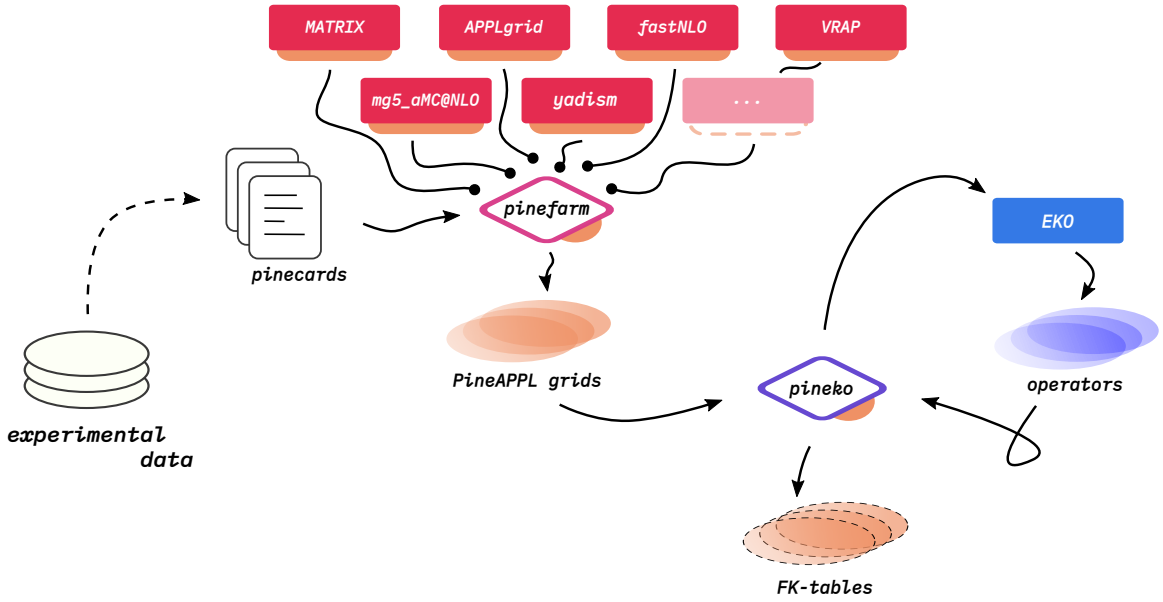

Figure 1. Flow diagram showing the overall pipeline architecture and deliverables in the case of parameter fits. Arrows in the picture indicate the flow of information (together with the execution order) and the orange insets on other elements indicate an interface to PineAPPL. The programs `pinefarm` and `pineko` act as interfaces between other programs and the deliverable objects, represented by ovals. These objects can be PineAPPL grids (orange) or Evolution Kernel Operators (blue).

2. Technical Overview of the `pineline`

We refrain here from repeating all technical details from the previous publication [3], but give in the following just a brief overview of the framework and its participating programs.

The core part of the framework is to use PDF independent interpolation grids, as provided by `PineAPPL` [8, 9] to be able to reuse predictions for any candidate PDF. We then extend the idea to the concept of FastKernel (FK) tables [10] where we now include also the perturbative evolution of PDFs inside the interpolation grid. FK tables can then be used efficiently inside a PDF fit as the usually complicated convolution between candidate PDF and object is now replaced by a simple linear algebra operation. To accomplish the various steps we develop the following programs:

- `pinefarm` acts as an interface to existing Monte Carlo generators to produce `.pineappl` grids containing the partonic information
- `yadism` provides structure functions in deep-inelastic scattering [11] (DIS) (and is interfaced to `pinefarm`)
- `eko` provides the solution to the (perturbative) evolution equations in terms of evolution kernel operators [12, 13]
- `pineko` combines partonic grids and evolution kernel operator into an FK table

The flow of programs is summarized in Fig. 1.

3. Benchmarking

The major focus of the `pineline` is to join several dedicated programs into a consistent framework for theory predictions. However, it is relying on those programs to provide the

necessary physical ingredients as the framework itself is (almost) physics agnostic and can indeed not only be applied to PDF fitting, but also the determination of other factorized objects, such as fragmentation functions [14], or beyond standard model (BSM) searches [15].

Still, it is convenient to develop some new programs which are tailored for the specific needs and concepts of the `pineline`. Specifically, we develop `EKO` [13, 12], a new DGLAP solver [16, 17, 18], and `yadism` [11], a new provider of deep-inelastic structure functions. Either of these tasks is an already solved problem for which several implementation exist [19, 20], yet none of them provide the exact deliverable that we require here.

In order to benchmark `EKO` and `yadism` to former implementations we develop `banana` [21]. `banana` provides a convenient benchmarking framework that turned out very useful during the development of both codes. It features a full-fledged database system (based on `SQLite`) to map input runcards to the respective outputs of both programs, the one to be benchmarked and the reference implementation. This allows a seamless comparison between the ongoing development and the reference implementation, such as investigating the difference in certain regions of the parameter space (which often can be mapped back onto a certain region in physics space). Since the reference implementation is fixed we implement a caching algorithm which ensures a fast iteration in the development process. Keeping a history of the runs allows us also to easily compare our program in two different version which again simplifies the development significantly.

In order to facilitate the access to the database we provide an interface built on top of `IPython` [22], dubbed `navigator`, that allows to easily query and retrieve records from the database. Moreover, being embedded into a live Python interpreter makes the `navigator` flexible and powerful.

We also add a few simple operations to the `navigator` such as the difference between program outputs. This was very helpful e.g. in the implementation of the FONLL prescription [23] for DIS structure function, which can be boiled down to the line

$$F^{\text{FONLL}} = F^{(n_l+1)} + F^{(n_l)} - F^{(n_l,0)} \tag{1}$$

where F refers to any DIS structure function and the various expressions on the right hand side refer to specific physical prescriptions. Each of these prescriptions are well-defined on their own, but already non-trivial, composite objects. Thus, having the possibility to benchmark them one at a time and subtracting either from the combination is very beneficial.

While `banana` provides the necessary framework that abstracts most tasks away, `EKO` and `yadism` have to implement each a specialization of the framework, dubbed `ekomark` and `yadmark` respectively, as indeed they compute different, but related objects. These specialized programs are developed in unison together with the main programs. `ekomark` currently provides interfaces to the LHA benchmark tables [24, 25], `PEGASUS` [26], `APFEL` [19], and to any LHAPDF [27] set. `yadmark` currently provides interfaces to `APFEL` [19], `QCDNUM` [20], and `xspace_bench` [23]. In Fig. 2 we show an application of the `ekomark` benchmarks using the `banana` framework.

4. Conclusion and outlook

With the `pineline` we aim to provide an simple framework to generate theory predictions from a common input and to deliver a unified output. This can be beneficial for the fitting of parton distribution functions [1] (PDF) or any factorized function such as fragmentation functions [14]. We split the necessary tasks to produce FastKernel tables [10] into separate programs, namely `pinefarm`, `EKO`, and `pineko`, each focusing only on the specific part. This allows to extend the framework to more involved physics cases such as the determination of the photon PDF [28, 29, 30], the inclusion of missing higher order uncertainties into PDF fits [31], or the extension to next-to-next-to-next-to-leading order (N3LO) perturbation theory [32, 33].

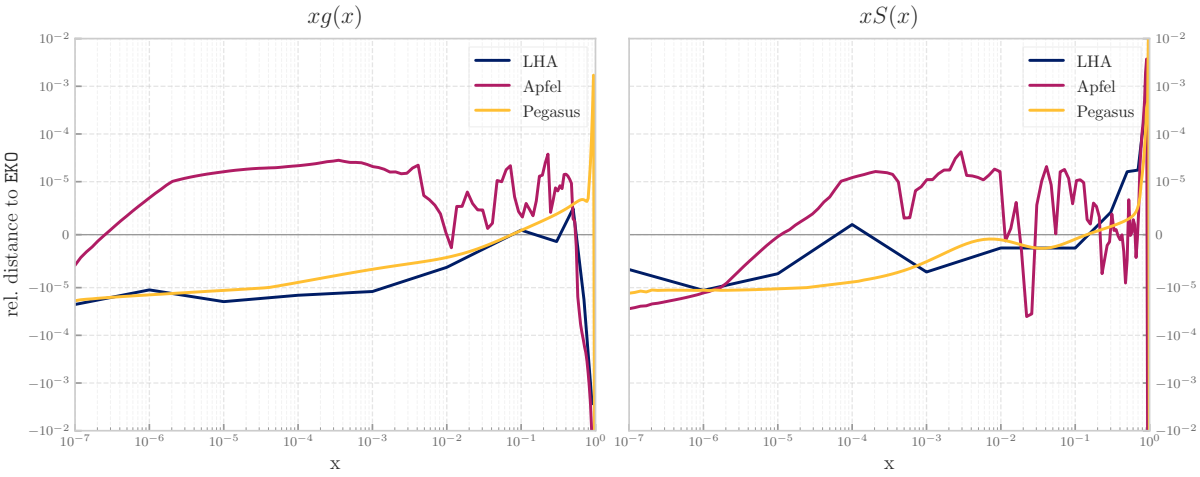

Figure 2. Part of [13, Fig. 1] comparing different implementation of the evolution equations to the LHA benchmark tables [24, 25] using `banana`. On the left (right) panel the gluon distribution $g(x)$ (singlet distribution $S(x)$) is plotted as a function of the momentum fraction x.

While developing the `pineline`, we also develop two participating programs: `EKO` and `yadism`. Either one is reimplementing algorithms and ingredients from already known calculations. It is thus imperative to ensure the new code reproduce previous implementations in a benchmarking process. To achieve this task we also develop `banana` which provides a sophisticated benchmarking framework tailored for the case at hand which allows a reliable and repeated execution of benchmarks.

Acknowledgments

A.C. and F.H. are supported by the European Research Council under the European Union's Horizon 2020 research and innovation Programme (grant agreement number 740006). C.S. is supported by the German Research Foundation (DFG) under reference number DE 623/6-2.

References

[1] Amoroso S *et al.* 2022 *Acta Phys. Polon. B* **53** A1 (*Preprint* 2203.13923)
[2] Ball R D *et al.* (NNPDF) 2022 *Eur. Phys. J. C* **82** 428 (*Preprint* 2109.02653)
[3] Barontini A, Candido A, Cruz-Martinez J M, Hekhorn F and Schwan C 2023 (*Preprint* 2302.12124)
[4] Gao J, Harland-Lang L and Rojo J 2018 *Phys. Rept.* **742** 1–121 (*Preprint* 1709.04922)
[5] Accardi A *et al.* 2016 *Eur. Phys. J. A* **52** 268 (*Preprint* 1212.1701)
[6] Anderle D P *et al.* 2021 *Front. Phys. (Beijing)* **16** 64701 (*Preprint* 2102.09222)
[7] Ball R D *et al.* (NNPDF) 2021 *Eur. Phys. J. C* **81** 958 (*Preprint* 2109.02671)
[8] Carrazza S, Nocera E R, Schwan C and Zaro M 2020 *JHEP* **12** 108 (*Preprint* 2008.12789)
[9] Schwan C, Candido A, Hekhorn F and Carrazza S 2023 Nnpdf/pineappl: v0.5.9 URL https://doi.org/10.5281/zenodo.7499507
[10] Ball R D *et al.* (NNPDF) 2015 *JHEP* **04** 040 (*Preprint* 1410.8849)
[11] Candido A, Hekhorn F and Magni G 2022 N3pdf/yadism: Fonll-b URL https://doi.org/10.5281/zenodo.6285149
[12] Candido A, Hekhorn F and Magni G 2022 N3pdf/eko: Paper URL https://doi.org/10.5281/zenodo.6340153
[13] Candido A, Hekhorn F and Magni G 2022 *Eur. Phys. J. C* **82** 976 (*Preprint* 2202.02338)
[14] Abdul Khalek R, Bertone V, Khoudli A and Nocera E R 2022 *Phys. Lett. B* **834** 137456 (*Preprint* 2204.10331)
[15] Ball R D, Candido A, Forte S, Hekhorn F, Nocera E R, Rojo J and Schwan C 2022 *Eur. Phys. J. C* **82** 1160 (*Preprint* 2209.08115)

[16] Dokshitzer Y L 1977 *Sov. Phys. JETP* **46** 641–653 [Zh. Eksp. Teor. Fiz.73,1216(1977)]
[17] Gribov V N and Lipatov L N 1972 *Sov. J. Nucl. Phys.* **15** 438–450 [Yad. Fiz.15,781(1972)]
[18] Altarelli G and Parisi G 1977 *Nucl. Phys.* **B126** 298–318
[19] Bertone V, Carrazza S and Rojo J 2014 *Comput. Phys. Commun.* **185** 1647–1668 (*Preprint* 1310.1394)
[20] Botje M 2011 *Comput.Phys.Commun.* **182** 490–532 (*Preprint* 1005.1481)
[21] Barontini A, Candido A, Hekhorn F and Magni G 2023 N3pdf/banana: polarized toy URL https://doi.org/10.5281/zenodo.7636142
[22] Pérez F and Granger B E 2007 *Computing in Science and Engineering* **9** 21–29 ISSN 1521-9615 URL https://ipython.org
[23] Forte S, Laenen E, Nason P and Rojo J 2010 *Nucl. Phys. B* **834** 116–162 (*Preprint* 1001.2312)
[24] Giele W *et al.* 2002 The QCD / SM working group: Summary report *2nd Les Houches Workshop on Physics at TeV Colliders* pp 275–426 (*Preprint* hep-ph/0204316)
[25] Dittmar M *et al.* 2005 (*Preprint* hep-ph/0511119)
[26] Vogt A 2005 *Comput. Phys. Commun.* **170** 65–92 (*Preprint* hep-ph/0408244)
[27] Buckley A, Ferrando J, Lloyd S, Nordström K, Page B, Rüfenacht M, Schönherr M and Watt G 2015 *Eur. Phys. J. C* **75** 132 (*Preprint* 1412.7420)
[28] Xie K, Hobbs T J, Hou T J, Schmidt C, Yan M and Yuan C P (CTEQ-TEA) 2022 *Phys. Rev. D* **105** 054006 (*Preprint* 2106.10299)
[29] Cridge T, Harland-Lang L A, Martin A D and Thorne R S 2022 *Eur. Phys. J. C* **82** 90 (*Preprint* 2111.05357)
[30] Bertone V, Carrazza S, Hartland N P and Rojo J (NNPDF) 2018 *SciPost Phys.* **5** 008 (*Preprint* 1712.07053)
[31] Abdul Khalek R *et al.* (NNPDF) 2019 *Eur. Phys. J. C* **79** 931 (*Preprint* 1906.10698)
[32] McGowan J, Cridge T, Harland-Lang L A and Thorne R S 2023 *Eur. Phys. J. C* **83** 185 (*Preprint* 2207.04739)
[33] Caola F, Chen W, Duhr C, Liu X, Mistlberger B, Petriello F, Vita G and Weinzierl S 2022 The Path forward to N^3LO *2022 Snowmass Summer Study* (*Preprint* 2203.06730)