



ChiliPDF: Chebyshev interpolation for parton distributions

Markus Diehl¹ , Riccardo Nagar^{2,a} , Frank J. Tackmann¹

¹ Deutsches Elektronen-Synchrotron DESY, Notkestr. 85, 22607 Hamburg, Germany

² Università degli Studi di Milano-Bicocca, INFN Sezione di Milano-Bicocca, Piazza della Scienza 3, 20126 Milan, Italy

Received: 10 February 2022 / Accepted: 13 March 2022 / Published online: 25 March 2022
© The Author(s) 2022

Abstract Parton distribution functions (PDFs) are an essential ingredient for theoretical predictions at colliders. Since their exact form is unknown, their handling and delivery for practical applications relies on approximate numerical methods. We discuss the implementation of PDFs based on a global interpolation in terms of Chebyshev polynomials. We demonstrate that this allows for significantly higher numerical accuracy at lower computational cost compared with local interpolation methods such as splines. Whilst the numerical inaccuracy of currently used local methods can become a nontrivial limitation in high-precision applications, in our approach it is negligible for practical purposes. This holds in particular for differentiation and for Mellin convolution with kernels that have end point singularities. We illustrate our approach for these and other important numerical operations, including DGLAP evolution, and find that they are performed accurately and fast. Our results are implemented in the C++ library CHILIPDF.

1 Introduction

Theoretical predictions at hadron colliders require parton distribution functions (PDFs), which describe the partonic content of the colliding hadrons. PDFs are nonperturbative objects and their exact form is unknown, such that their handling and delivery in practical applications requires approximate numerical methods. Currently available tools use local interpolation over a finite grid of function values, such as splines. For example, the LHAPDF library [1], which has de facto become the standard interface with which PDFs are provided to the community, implements a cubic spline interpolation.

In this paper, we demonstrate a different approach for the numerical representation of PDFs, which is based on a global, high-order interpolation using Chebyshev poly-

nomials, and which allows for significantly higher numerical accuracy at lower computational cost than local interpolation methods. We have implemented this approach in the C++ library CHILIPDF,¹ which is used for most of the numerical demonstrations in the following. We think, however, that the methods presented in our paper are of interest beyond their implementation in a specific software package.

There is a long history of using families of polynomials for handling PDFs or related quantities. Without any claim to completeness, let us mention a few examples. To express PDFs in terms of their Mellin moments, Bernstein polynomials were proposed in Ref. [2] and Jacobi polynomials in Ref. [3]. An explicit solution of the evolution equations was obtained in Refs. [4, 5] by expanding PDFs and splitting functions in Laguerre polynomials. More references and discussion can be found in Refs. [6–8]. To compute analytically the complex Mellin moments of parton luminosities, Refs. [9, 10] used an expansion in Chebyshev polynomials. The latter also appear in the parameterization of initial conditions in the PDF fits of Refs. [11–14]. We find that all these methods differ quite significantly from each other and from the method to be presented in this work.

Let us emphasize that we are not concerned here with the question of how best to parameterize input PDFs that are fitted to data, nor with the associated systematic error or bias due to the choice of parameterization. For our purposes, we consider the fitted input PDFs to be known “exactly”. Our method could be used to address this parameterization issue as well, but we leave this for future exploration.

What we are concerned with is the numerical implementation and handling of PDFs in practical applications, for instance when using PDFs evolved to some scale to obtain quantitative predictions from analytic cross section formulae or with Monte-Carlo event generators. The interpolation of PDFs comes with an inherent inaccuracy that is of purely numerical origin, similar to numerical integration errors.

^a e-mail: riccardo.nagar@unimib.it (corresponding author)

¹ [Chebyshev Interpolation Library for PDFs](#).

Such inaccuracies should at the very least be small compared to uncertainties due to physics approximations such as the perturbative expansion. But ideally, one would like such inaccuracies to be negligible or at least small enough to be of no practical concern. We will show that this goal can indeed be achieved with Chebyshev interpolation.

Several observations lead us to believe that the performance of available local interpolation methods is becoming insufficient. Generically, the relative accuracy of the interpolation provided by LHAPDF is expected (and found to be) in the ballpark of 10^{-3} to 10^{-4} . For high-precision predictions this is already close – perhaps uncomfortably close – to the desired percent-level theoretical precision. Moreover, PDFs typically appear in the innermost layer of the numerical evaluation, whose output is then further processed. For example, they enter at the maximally differential level, which is subject to subsequent numerical integrations. Each step comes with some loss of numerical precision, which means the innermost elements should have a higher numerical accuracy than what is desired for the final result. If an integration routine becomes sensitive to numerical PDF inaccuracies, its convergence and hence the computational cost can suffer greatly, because the integrator can get distracted (or even stuck) trying to integrate numerical noise. We have explicitly observed this effect for quadrature-based integrators, which for low-dimensional integrations are far superior to Monte-Carlo integrators, but whose high accuracy and fast convergence strongly depends on the smoothness of the integrand. Another issue is that with increasing perturbative order, the convolution kernels in cross section formulae become more and more steep or strongly localized. This requires a higher numerical accuracy of the PDF, because its convolution with such kernels is sensitive not just to its value but also to its derivatives or its detailed shape. In Ref. [15], it was indeed observed that the limited numerical accuracy of PDFs provided by LHAPDF can cause instabilities in the final result.

Many applications require an accurate and fast execution of nontrivial operations on PDFs, such as taking Mellin moments, computing Mellin convolutions with various kernels, taking derivatives, and so forth. A prominent example is DGLAP evolution, which has been extensively studied in the literature, see e.g. Refs. [16–20], and for which there is a variety of codes that solve the evolution equations either via Mellin moments [21–23] or directly in x space [24–28]. The latter require in particular the repeated Mellin convolution of PDFs with splitting functions, in addition to interpolation in x .

Another important application is the computation of beam functions and similar quantities, which typically appear in resummation formulae. In multiscale problems, beam functions can depend on additional dynamic variables [29–37], in which case their evaluation necessitates fast on-the-fly evaluation of Mellin convolutions and in some cases their sub-

sequent DGLAP evolution. A further area is the calculation of subleading power corrections, where the first and second derivatives of PDFs with respect to x are explicitly required [38–44]. With cubic splines, the first and second derivative respectively correspond to quadratic and linear interpolation and hence suffer from a poor accuracy. Numerical instabilities in derivatives computed from spline interpolants were for instance reported in Ref. [45].

Last but not least, for double parton distributions [46] or other multi-dimensional distribution functions, local interpolation methods become increasingly cumbersome due to the large number of variables and even larger number of distributions. By contrast, our global interpolation approach allows for a controllable numerical accuracy with reasonable memory and runtime requirements. This will be discussed in future work.

Clearly, there is always a trade-off between the numerical accuracy of a method and its computational cost, and the accuracy of different methods should be compared at similar computational cost (or vice versa). In our case, the primary cost indicator is the number of points on the interpolation grid, which controls both the number of CPU operations and the memory footprint.² As a result of its low polynomial order, local spline interpolation has an accuracy that scales rather poorly with the grid size, so that even a moderate increase in precision requires a substantial increase in computational effort. This can be (partially) offset by improving the performance of the implementation itself, see for example Refs. [28, 47]. Such improvements can be made for any given method, but they do not change the accuracy scaling of the method itself.

Global interpolation approximates a function over its domain by a single, high-order polynomial. On an equispaced grid, this leads to large oscillations near the edges of the interpolation interval, such that the interpolant never converges and may in fact diverge exponentially. This is well known as Runge's phenomenon [48]. It is often misinterpreted as a problem of polynomial interpolation in general, which may be one reason why local interpolation methods such as splines tend to be preferred. What is perhaps not sufficiently appreciated [49] is the fact that Runge's phenomenon is caused by the use of an *equidistant* grid and can be completely avoided by using a non-equidistant grid that clusters the grid points toward the edges of the interval. An example for this is Chebyshev interpolation, which leads to a well-convergent approximation, i.e., one that can be made arbitrarily precise by increasing the number of interpolation points. Moreover, approximation with Chebyshev polynomi-

² Of course, the inherent complexity of a method matters as well. For instance, more complex and thereby more accurate splines tend to have a higher computational cost per grid point. However, these are secondary effects we will not focus on.

als on a finite interval is closely related to and thus as reliable as the Fourier series approximation for periodic functions. We find that Chebyshev interpolation for PDFs has an excellent accuracy scaling. As we will show, it easily outperforms local spline interpolation by many orders of magnitude in accuracy for the same or even lower number of grid points.

In the next section, we provide some mathematical background on Chebyshev interpolation as we require it. In Sect. 3, we present the global Chebyshev interpolation of PDFs used in CHILIPDF and compare its accuracy with local spline interpolation. We also discuss methods for estimating the numerical accuracy, as well as the basic operations of taking derivatives and integrals of PDFs. In Sects. 4 and 5, we describe the implementation and accurate evaluation of Mellin convolutions and of DGLAP evolution with CHILIPDF. We conclude in Sect. 6. In an appendix, we discuss the performance of different Runge–Kutta algorithms for solving the evolution equations.

2 Chebyshev interpolation

In this section, we give a brief account of Chebyshev interpolation, i.e. the interpolation of a function that is discretized on a grid of Chebyshev points. This includes the topics of differentiation, integration, and of estimating the interpolation accuracy. A wealth of further information and mathematical background can be found in Ref. [50].³

Throughout this section, we consider functions of a variable t restricted to the interval $[-1, 1]$. The relation between t and the momentum fraction x of a PDF is specified in Sect. 3.

2.1 Chebyshev polynomials

The Chebyshev polynomials of the first and second kind, $T_k(t)$ and $U_k(t)$, are defined by

$$T_k(\cos \theta) = \cos k\theta, \quad U_k(\cos \theta) = \frac{\sin(k+1)\theta}{\sin \theta} \quad (1)$$

for integer $k \geq 0$. They are related by differentiation as $dT_k(t)/dt = kU_{k-1}(t)$, and they are bounded by $|T_k(t)| \leq 1$ and $|U_k(t)| \leq k+1$. The relations

$$\begin{aligned} V_k(-t) &= (-1)^k V_k(t), & V_0(t) &= 1, \\ V_{k+2}(t) &= 2t V_{k+1}(t) - V_k(t) \end{aligned} \quad (2)$$

hold for both $V = T$ and $V = U$. They show that $T_k(t)$ and $U_k(t)$ are indeed polynomials, which may not be immediately

obvious from Eq. (1). Both families of polynomials form an orthogonal set, i.e. for $k, m \geq 0$ they satisfy

$$\begin{aligned} \int_{-1}^1 \frac{dt}{\sqrt{1-t^2}} T_k(t) T_m(t) &= \frac{\alpha_k \pi}{2} \delta_{km}, \\ \int_{-1}^1 dt \sqrt{1-t^2} U_k(t) U_m(t) &= \frac{\pi}{2} \delta_{km}, \end{aligned} \quad (3)$$

where $\alpha_0 = 2$ and $\alpha_k = 1$ otherwise.

For given N , the *Chebyshev points* are given by

$$t_j = \cos \theta_j, \quad \theta_j = \frac{j\pi}{N} \quad \text{with } j = 0, \dots, N. \quad (4)$$

They form a descending series from $t_0 = 1$ to $t_N = -1$ and satisfy the symmetry property $t_{N-j} = -t_j$. The polynomial $T_N(t)$ assumes its maxima $+1$ and minima -1 at the Chebyshev points. We call the set of Chebyshev points a *Chebyshev grid*. Using Eq. (1) and expressing sines and cosines as complex exponentials, one readily derives the discrete orthogonality relations

$$\begin{aligned} \sum_{j=0}^N \beta_j T_k(t_j) T_m(t_j) &= \frac{N}{2\beta_k} \delta_{km} \\ &\quad \text{for } k, m = 0, \dots, N, \\ \sum_{j=1}^{N-1} (1-t_j^2) U_{k-1}(t_j) U_{m-1}(t_j) &= \frac{N}{2} \delta_{km} \\ &\quad \text{for } k, m = 1, \dots, N-1, \end{aligned} \quad (5)$$

where $\beta_0 = \beta_N = 1/2$ and $\beta_j = 1$ otherwise.

Notice that the density of Chebyshev points increases from the center toward the end points of the interval $[-1, 1]$. This feature is crucial to avoid Runge's phenomenon for equispaced interpolation grids, as discussed in the introduction.

2.2 Chebyshev interpolation and Chebyshev series

We now consider the approximation of a function $f(t)$ by a finite sum of Chebyshev polynomials $T_k(t)$. Using $T_k(t_j) = T_j(t_k)$ and the first relation in Eq. (5), one finds that the sum

$$p_N(t) = \sum_{k=0}^N \beta_k c_k T_k(t) \quad (6)$$

with the interpolation coefficients

$$c_k = \frac{2}{N} \sum_{j=0}^N \beta_j f(t_j) T_k(t_j) \quad (7)$$

satisfies

³ The first chapters of this book are available on <https://people.maths.ox.ac.uk/trefethen/ATAP>.

$$p_N(t_j) = f(t_j) \quad \text{for } j = 0, \dots, N. \quad (8)$$

In other words, $p_N(t)$ is the unique polynomial of order N that equals the function $f(t)$ at the $N + 1$ Chebyshev points t_0, \dots, t_N . We therefore call $p_N(t)$ the *Chebyshev interpolant*.

A sufficiently smooth function $f(t)$ can be expanded in the *Chebyshev series*

$$f(t) = \lim_{N \rightarrow \infty} f_N(t), \quad f_N(t) = \sum_{k=0}^N a_k T_k(t), \quad (9)$$

whose series coefficients

$$a_k = \frac{2}{\alpha_k \pi} \int_{-1}^1 \frac{dt}{\sqrt{1-t^2}} f(t) T_k(t) \quad (10)$$

readily follow from the first orthogonality relation in Eq. (3). Substituting $t = \cos \theta$ and using Eq. (1), we recognize in Eqs. (9) and (10) the Fourier cosine series for the function $F(\theta) = f(\cos \theta)$, which is periodic and even in θ . The Chebyshev series on the interval $[-1, 1]$ is thus nothing but the Fourier series of a periodic function in disguise, with the same excellent convergence properties for $N \rightarrow \infty$. The Chebyshev series $f_N(t)$ is not immediately useful in practice, because its coefficients can only be obtained by explicitly carrying out the integral in Eq. (10). The Chebyshev interpolant $p_N(t)$, however, can be computed very efficiently (see below) and only requires evaluating $f(t_j)$ at the $N + 1$ Chebyshev points. The key property of interpolating in the Chebyshev points is that the resulting interpolation coefficients c_k approach the series coefficients a_k in the limit $N \rightarrow \infty$. Their precise relation can be found in [50, chapter 4]. It is therefore guaranteed that $p_N(t)$ approaches $f_N(t)$ and thus $f(t)$ for $N \rightarrow \infty$.

2.3 Interpolation accuracy

How accurately $p_N(t)$ approximates the function $f(t)$ depends on the smoothness of f and its derivatives. We give here a convergence statement that is useful for the interpolation of PDFs. For typical parameterizations, input-scale PDFs are analytic functions of the momentum fraction x for $0 < x < 1$, but nonanalytic at $x = 1$, where they behave like $(1-x)^\beta$ with noninteger β . We anticipate that this corresponds to a behavior like $(1+t)^\beta$ at $t = -1$ when we map an x interval onto a Chebyshev grid in t .

Suppose that on the interval $[-1, 1]$ the function f and its derivatives up to $f^{(\nu-1)}$ with $\nu \geq 1$ are Lipschitz continuous,⁴ and that the derivative $f^{(\nu)}$ is of bounded vari-

ation V . We recall that a function $F(t)$ is Lipschitz continuous on $[-1, 1]$ if there exists a constant C such that $|F(s) - F(t)| \leq C|s - t|$ for all $s, t \in [-1, 1]$. A differentiable function $F(t)$ is of bounded variation V on $[-1, 1]$ if the integral

$$V = \int_{-1}^1 dt |F'(t)| = \sum_{k=0}^K |F(t_{k+1}) - F(t_k)| \quad (11)$$

is finite, where in the second step we have split the interval $[-1, 1]$ into K subintervals with boundaries t_0, \dots, t_{K+1} such that on each subinterval $F'(t)$ has a definite sign. Under these conditions, one has

$$|f(t) - p_N(t)| \leq \frac{4V}{\pi \nu (N - \nu)^\nu} \quad (12)$$

for all $N > \nu$ and all $t \in [-1, 1]$. We note that the Chebyshev series has a similar convergence property, which is obtained by replacing $p_N(t)$ with $f_N(t)$ and $4V$ with $2V$ in Eq. (12).

For the example of a function $f(t) = (1+t)^{n+\delta}$ with integer $n \geq 1$ and $0 < \delta < 1$, the above statement holds with $\nu = n$. The function and its derivatives up to $f^{(n-1)}$ are Lipschitz continuous on $[-1, 1]$. The n th derivative is proportional to $(1+t)^\delta$ and not Lipschitz continuous but of bounded variation. The $(n+1)$ st derivative is proportional to $(1+t)^{-1+\delta}$ and thus not of bounded variation, because it diverges at $t = -1$.

It is important to note that Eq. (12) provides a bound on the maximal *absolute* interpolation error anywhere in the interval. Often the absolute interpolation error will be much smaller over most of the interval. In the vicinity of a point where $f(t)$ goes to zero, the *relative* error can still remain large, and the convergence $p_N(t)/f(t) \rightarrow 1$ for $N \rightarrow \infty$ is in general not uniform over the full interval. We will indeed see this for the interpolation of PDFs close to zero crossings or to the end point $x = 1$.

2.4 Barycentric formula

A simple and efficient way to compute the Chebyshev interpolant is given by the *barycentric formula*

$$p_N(t) = \sum_{j=0}^N f(t_j) b_j(t), \quad (13)$$

⁴Footnote 4 continued
of Lipschitz continuity, which we find sufficient for our purpose and easier to state.

where t_j denotes again the Chebyshev points, and the barycentric basis functions are given by

$$b_j(t) = \frac{\beta_j (-1)^j}{t - t_j} \bigg/ \sum_{i=0}^N \frac{\beta_i (-1)^i}{t - t_i}. \quad (14)$$

$b_j(t)$ is a polynomial of order N , although this is not evident from Eq. (14). The number of operations for evaluating the barycentric formula scales like N . The formula is found to be numerically stable in the interpolation interval. We note that it is *not* stable for extrapolating the function $f(t)$ outside this interval [50, chapter 5].

The representation given by Eqs. (13) and (14) is a special case of the barycentric formula for the polynomial $L(u)$ of order N that interpolates a function $f(u)$ given on a set of $N + 1$ distinct points u_0, \dots, u_N :

$$L_N(u) = \sum_{j=0}^N f(u_j) \ell_j(u) \quad (15)$$

with basis functions

$$\ell_j(u) = \frac{\lambda_j}{u - u_j} \bigg/ \sum_{i=0}^N \frac{\lambda_i}{u - u_i}, \quad (\lambda_i)^{-1} = \prod_{j \neq i} (u_j - u_i). \quad (16)$$

L_N is called the *Lagrange polynomial* for the pairs of values $\{u_j, f(u_j)\}$. We will again use Eq. (15) below. The simple form of Eq. (14) comes from the fact that the Chebyshev points yield the very simple weights $\lambda_i = \beta_i (-1)^i 2^{n-1}/n$.

2.5 Differentiation

Given the Chebyshev interpolant $p_N(t)$ for a function $f(t)$, one can approximate the derivative $f'(t) = df(t)/dt$ by the derivative $p'_N(t)$. Note that in general f' is not equal to p'_N at the Chebyshev points. Obviously, one cannot compute the exact values of $f'(t_j)$ from the function values $f(t_j)$ on the grid.

The derivative $p'_N(t)$ is a polynomial of degree $N - 1$ and thus also a polynomial of degree N (with vanishing coefficient of t^N). It is therefore identical to its own Chebyshev interpolant of order N , so we can compute it on the full interval $[-1, 1]$ by the barycentric formula

$$p'_N(t) = \sum_{j=0}^N p'_N(t_j) b_j(t). \quad (17)$$

To obtain the values of $p'_N(t_j)$, we take the derivative of Eq. (6) using $T'_k = k U_{k-1}$. The resulting discrete sums are

easily evaluated using Eq. (1) and expressing the sine function in terms of complex exponentials. We then obtain the relation

$$p'_N(t_j) = \sum_{k=0}^N D_{jk} f(t_k) \quad (18)$$

with $D_{00} = -D_{NN} = (2N^2 + 1)/6$ and

$$D_{jj} = -\frac{\cos \theta_j}{2 \sin^2 \theta_j} \quad \text{for } j \neq 0, N, \\ D_{jk} = \frac{\beta_k (-1)^{j+k}}{\beta_j (t_j - t_k)} \quad \text{for } j \neq k. \quad (19)$$

Note that the matrix multiplication (18) maps a vector $f(t_k) = \text{const}$ onto the zero vector, as it must be because the derivative of a constant function is zero.

Higher derivatives of the Chebyshev interpolant $p_N(t)$ can be computed by repeated multiplication of $f(t_k)$ with the differentiation matrix D_{jk} and subsequent application of the barycentric formula. Since each derivative reduces the degree of the interpolating polynomial by 1, the accuracy of approximating $f^{(n)}(t)$ by $p_N^{(n)}(t)$ gradually degrades with increasing n . This loss of accuracy can be accounted for by choosing a sufficiently large order N to start with.

2.6 Integration

Using Eq. (6) together with

$$\int_{-1}^1 dt T_k(t) = \begin{cases} 2/(1 - k^2) & \text{for even } k, \\ 0 & \text{for odd } k, \end{cases} \quad (20)$$

one readily obtains the integration rule

$$\int_{-1}^1 dt f(t) \approx \sum_{\substack{k=0 \\ \text{even}}}^N \frac{2\beta_k c_k}{1 - k^2} = \sum_{j=0}^N w_j f(t_j) \quad (21)$$

with weights

$$w_j = \frac{4\beta_j}{N} \sum_{\substack{k=0 \\ \text{even}}}^N \beta_k \frac{\cos(k\theta_j)}{1 - k^2}. \quad (22)$$

This is known as *Clenshaw–Curtis quadrature*. A detailed discussion of its accuracy (and comparison with Gauss quadrature) can be found in Ref. [50, chapter 19] and Ref. [51].

2.7 Interpolation without end points

The interpolant (6) is a sum over Chebyshev polynomials T_k . Another interpolant can be obtained from the polynomials U_k , namely

$$q_{N-2}(t) = \sum_{k=1}^{N-1} d_k U_{k-1}(t) \quad (23)$$

with coefficients

$$d_k = \frac{2}{N} \sum_{j=1}^{N-1} f(t_j) (1 - t_j^2) U_{k-1}(t_j). \quad (24)$$

Using the second relation in Eq. (5) together with the equality $\sin \theta_j U_{k-1}(t_j) = \sin \theta_k U_{j-1}(t_k)$, one finds that

$$q_{N-2}(t_j) = f(t_j) \quad \text{for } j = 1, \dots, N-1. \quad (25)$$

This means that $q_{N-2}(t)$ interpolates $f(t)$ on the same Chebyshev points as $p_N(t)$, with the exception of the interval end points. Correspondingly, q_{N-2} has polynomial degree $N-2$ rather than N , as is evident from Eq. (23). We thus have an alternative approximation for $f(t)$, which can be computed from the same discretized function values as those needed for computing $p_N(t)$. For sufficiently large N , one may expect that q_{N-2} approximates $f(t)$ only slightly less well than $p_N(t)$, given that its polynomial degree is only smaller by two units. We may hence use $|q_{N-2}(t) - p_N(t)|$ as a conservative estimate for the interpolation error $|f(t) - p_N(t)|$.⁵

To evaluate $q_{N-2}(t)$, we can again use a barycentric formula, namely

$$q_{N-2}(t) = \sum_{j=1}^{N-1} f(t_j) \tilde{b}_j(t). \quad (26)$$

The corresponding basis functions

$$\tilde{b}_j(t) = \frac{(-1)^j \sin^2 \theta_j}{t - t_j} \bigg/ \sum_{i=1}^{N-1} \frac{(-1)^i \sin^2 \theta_i}{t - t_i} \quad (27)$$

can easily be obtained from Eq. (14) by comparing the general expressions (15) and (16) for the sets of points t_j with $j = 0, \dots, N$ or $j = 1, \dots, N-1$. We note that using the barycentric formula (26) in the full interval $[-1, 1] = [t_N, t_0]$ involves an extrapolation from the interval $[t_{N-1}, t_1]$ on

⁵ To be precise, since $p_N(t)$ is more accurate than $q_{N-2}(t)$, the difference $|q_{N-2}(t) - p_N(t)|$ is actually an estimate of the accuracy of $q_{N-2}(t)$ and thus a conservative estimate for the accuracy of $p_N(t)$.

which $q_{N-2}(t)$ interpolates $f(t)$. For sufficiently large N , the extrapolation is however very modest, because $t_{N-1} - t_N = t_0 - t_1 \approx \pi^2/(2N^2)$.

Formulae for differentiation and integration of $f(t)$ that use the approximant q_{N-2} are easily derived in analogy to the formulae that use p_N . For differentiation, one obtains

$$q'_{N-2}(t) = \sum_{j=1}^{N-1} q'_{N-2}(t_j) \tilde{b}_j(t) \quad (28)$$

and

$$q'_{N-2}(t_j) = \sum_{k=1}^{N-1} \tilde{D}_{jk} f(t_k) \quad (29)$$

with

$$\begin{aligned} \tilde{D}_{jj} &= \frac{3 \cos \theta_j}{2 \sin^2 \theta_j}, \\ \tilde{D}_{jk} &= \frac{\sin^2 \theta_k}{\sin^2 \theta_j} \frac{(-1)^{j+k}}{t_j - t_k} \quad \text{for } j \neq k. \end{aligned} \quad (30)$$

For integration, one uses

$$\int_{-1}^1 dt U_{j-1}(t) = \begin{cases} 2/j & \text{for odd } j, \\ 0 & \text{for even } j, \end{cases} \quad (31)$$

to obtain an open integration rule

$$\int_{-1}^1 dt f(t) \approx \sum_{\substack{k=1 \\ \text{odd}}}^{N-1} \frac{2d_k}{k} = \sum_{j=1}^{N-1} \tilde{w}_j f(t_j) \quad (32)$$

with weights

$$\tilde{w}_j = \frac{4 \sin \theta_j}{N} \sum_{\substack{k=1 \\ \text{odd}}}^{N-1} \frac{\sin(k \theta_j)}{k}. \quad (33)$$

This is well known as *Fejér's second rule*, see e.g. Ref. [52].⁶

Using Eq. (32) to estimate the error of Clenshaw–Curtis integration (21) is similar to *Gauss–Kronrod* quadrature [53, 54], where for a grid with $2N+1$ points one has an integration rule of order $3N+1$ and a Gauss rule of order $2N-1$, where the latter uses a subset of the points and is used to estimate the integration uncertainty.⁷ The difference in polynomial

⁶ This paper is also available at <http://www.sam.math.ethz.ch/~waldvogel/Papers/fejér.html>.

⁷ An integration rule is of order p if polynomials up to order p are integrated exactly. For odd N , the order of the Gauss–Kronrod rule is increased from $3N+1$ to $3N+2$, because odd polynomials are correctly integrated to zero for symmetry reasons [54]. Likewise, the order of the

orders between the two rules is hence larger in this case than for the pair of Clenshaw–Curtis and Fejér rules, so that one may expect the error estimate in the latter case to be closer to the actual error. We shall come back to this in Sect. 3.4.

Given the Chebyshev grid (4) with $N+1$ points for even N , one might also think of estimating the integration or interpolation accuracy by using the subgrid $t_0, t_2, \dots, t_{N-2}, t_N$, which has $N/2 + 1$ points and is again a Chebyshev grid. For the grids and functions we will consider in this work, this would however give a gross overestimate of the actual error, because the interpolant with $N/2 + 1$ points is significantly worse than the one with $N + 1$ points. By contrast, using interpolation without the end points of the original $N + 1$ point grid, we obtain error estimates that are rather reliable, as will be shown in Sect. 3.4.

3 Chebyshev interpolation of PDFs

3.1 Interpolation strategy

To interpolate a parton distribution function $f(x)$ in the momentum fraction x , we interpolate the function $\tilde{f}(x) = xf(x)$ in the variable $u = \ln x$. For a given minimum momentum fraction x_0 , the interval $[x_0, 1]$ is thus mapped onto the interval $[u_0, 0]$. For reasons discussed below, we usually split the x interval into a few subintervals. On each subinterval, we perform a linear transformation from $u = \ln x$ to $t \in [-1, 1]$ and introduce a Chebyshev grid in t , which is used to interpolate the function as described in Sect. 2. To specify the full grid, which is a conjunction of k subgrids, we use the notation

$$[x_0, x_1, \dots, 1]_{(n_1, n_2, \dots, n_k)}, \quad (34)$$

where the x_i are the subinterval boundaries and $n_i = N_i + 1$ is the number of Chebyshev points for subgrid i . We will refer to this as an (n_1, n_2, \dots, n_k) -point grid. Note that adjacent subgrids share their end points, so that the total number of grid points is $n_{\text{pts}} = \sum_i n_i - (k - 1)$.

To be specific, let us consider one subgrid with $N + 1$ points u_0, \dots, u_N , which is mapped by a linear transform onto the Chebyshev grid t_0, \dots, t_N given by Eq. (4). The corresponding grid points in x are $x_i = e^{u_i}$. We can then interpolate the PDF using the barycentric formula

$$\tilde{f}(x) \approx \sum_{j=0}^N \tilde{f}_j b_j(\ln x) \quad \text{for } x_0 \leq x \leq x_N, \quad (35)$$

where

Footnote 7 continued
quadrature rules (21) and (32) for even N is increased from N to $N + 1$ and from $N - 2$ to $N - 1$, respectively.

$$\tilde{f}_j = x_j f(x_j), \quad b_j(u) = \frac{\beta_j (-1)^j}{u - u_j} \bigg/ \sum_{i=0}^N \frac{\beta_i (-1)^i}{u - u_i}. \quad (36)$$

Here we have used that the form of the barycentric basis functions (14) remains unchanged under a linear transform of the interpolation variable. Formulae analogous to Eq. (35) can be used to interpolate functions derived from PDFs, such as their derivatives (see Sect. 3.3) or Mellin convolutions of PDFs with an integral kernel (see Sect. 4).

One reason to use subgrids concerns error propagation. As seen in Eq. (35), the interpolation of \tilde{f} at a certain value x involves the values of \tilde{f} on *all* grid points in the interpolation interval, although the weight of points close to x is higher than the weight of points far away. In an x region where \tilde{f} is much smaller than its maximum in the interval, numerical errors from regions of large \tilde{f} can strongly affect the interpolation accuracy. This is not much of an issue in our tests below, where \tilde{f} is computed from an analytic expression, but it does become important when interpolating a PDF that has been evolved to a higher scale and is thus affected by numerical errors from the solution of the DGLAP equations.

So on one hand, the accuracy depends on the behavior of the interpolated function on each subgrid. On the other hand, using multiple subgrids for a fixed total number n_{pts} of points decreases the polynomial degree of the interpolant on each subgrid, and the accuracy quickly degrades when the polynomial degree becomes too small. Hence, for given n_{pts} and x_0 , there is a certain range for the number of subgrids that give the best performance for typical PDFs. We find that the optimum is to take 2 or 3 subgrids for the values of n_{pts} and x_0 used in the following.

To study the accuracy of Chebyshev interpolation for typical PDFs, we consider a number of representative test functions, covering a broad range of shapes and analytic forms,

$$\begin{aligned} x f_1(x) &= 0.0703 x^{-0.415 (1+4.44x) (1+0.0373 \ln x)} (1-x)^{7.75}, \\ x f_2(x) &= 17.217 x^{-0.33293} (1-x)^{5.3687} \\ &\quad \times [1 - 1.664 T_1(y(x)) + 0.99169 T_2(y(x)) \\ &\quad - 0.42245 T_3(y(x)) + 0.10176 T_4(y(x))], \\ x f_3(x) &= 4.34 x^{-0.015} (1-x)^{9.11} \\ &\quad - 1.048 x^{-0.167} (1-x)^{25.0}, \\ x f_4(x) &= 7.4 x^{0.92} (1-x)^{4.6} \\ &\quad \times (1 - 2.8 \sqrt{x} + 4.5 x - 2.0 x^2), \end{aligned} \quad (37)$$

where $y(x) = 1 - 2\sqrt{x}$ and T_k denotes the Chebyshev polynomials defined in Eq. (1). These functions correspond to PDFs at the input scales of several common PDF sets. Specifically, we have $f_1 = \bar{u}$ at NNLO for ABMP16 [55], $f_2 = g$ at LO for MMHT2014 [14], $f_3 = g$ at NLO for HERAPDF2.0 [56], and $f_4 = d_v$ at NLO for JR14 [57]. We have studied

several other functions, including $xf(x) \propto x^{-0.7}(1-x)^{9.2}$, which decreases very steeply and behaves like a typical gluon density at high scale. The results shown in the following are representative of this more extended set of functions.

Throughout this work, the relative numerical accuracy of interpolation for a given quantity is obtained as

$$\text{relative accuracy} = |\text{interpolated result}/\text{exact result} - 1|, \quad (38)$$

where the exact result is evaluated using the analytic form of the functions in Eq. (37). In most plots, we also show the exact result itself as a thick black line, which is solid (dashed) when the result is positive (negative). The relative accuracy for other numerical operations is obtained in full analogy to Eq. (38).

3.2 Interpolation accuracy and comparison with splines

We now compare Chebyshev interpolation with local interpolation. As a prominent example for a method widely used in high-energy physics calculations, we take the interpolation provided by the LHAPDF library [1], which has become a standard interface for accessing parton densities. LHAPDF offers linear and cubic splines in either x or $\ln x$. We use cubic splines in $\ln x$, which is the default in LHAPDF and gives the most accurate results among these options.⁸ For brevity, we refer to these as “L-splines” in the following. Since spline interpolants come in a wide variety, we also consider the cubic splines used by the `Interpolation` command of Mathematica (versions 11 and 12) and refer to them as “M-splines”. For M-splines, both the first and second derivative of the interpolant are continuous, whilst for L-splines only the first derivative is continuous but the second is not.

The interpolation grids used in LHAPDF depend on the PDF set and cover a wide range in the minimum momentum fraction x_0 and in the total number of grid points n_{pts} , as shown in Table 1 for several common PDF sets. The resulting average density of points per decade in x is $\rho = -n_{\text{pts}}/\log_{10}(x_0)$ and essentially determines the numerical accuracy of the spline interpolation. For the sake of comparison, we use the grids with the smallest and the largest density among common PDF sets, which happen to be the MMHT2014 grid ($n_{\text{pts}} = 64$ with $\rho = 10.7$) and the HERAPDF2.0 grid ($n_{\text{pts}} = 199$ with $\rho = 33.1$).

In Fig. 1 we compare the spline interpolation on the low-density grid (MMHT2014 grid with $n_{\text{pts}} = 64$) with Chebyshev interpolation on a (32, 32)-point grid, which has nearly

Table 1 LHAPDF grid parameters for several common PDFs ordered by increasing grid density $\rho = -n_{\text{pts}}/\log_{10}(x_0)$

PDF set	x_0	n_{pts}	ρ
MMHT2014 [14]	10^{-6}	64	10.7
ABMP16 [55]	10^{-7}	99	14.1
NNPDF3.1 [58]	10^{-9}	150	16.7
CT18 [59]	0.926×10^{-9}	161	17.8
JR14 [57]	10^{-9}	190	21.1
MSHT20 [60]	10^{-6}	127	21.2
NNPDF4.0 [61]	10^{-9}	196	21.8
CT14 [62]	10^{-9}	240	26.7
HERAPDF2.0 [56]	0.99×10^{-6}	199	33.1

the same total number of points ($n_{\text{pts}} = 63$). The M-splines turn out to be more accurate than the L-splines, but this comes at the expense of them being more complex to construct. The Chebyshev interpolation achieves a significantly higher accuracy by several orders of magnitude than either of the splines. This reflects that, contrary to splines, Chebyshev interpolation uses polynomials of a high degree. In Fig. 2, we compare splines on the high-density grid (HERAPDF2.0 grid with $n_{\text{pts}} = 199$) with Chebyshev interpolation on a (40, 32)-point grid with a total of $n_{\text{pts}} = 71$. Here, even with less than half the number of points, the Chebyshev interpolation achieves several orders of magnitude higher accuracy. This also highlights that the interpolation accuracy scales much better with the number of points for Chebyshev interpolation than for splines.

In Fig. 3 we compare the accuracy of interpolation on a single Chebyshev grid with the accuracy obtained with the two subgrids used in Fig. 2. We see that for the same total number of points, interpolation on two subgrids is more accurate, as anticipated above.

We observe in Figs. 1, 2 and 3 that the relative accuracy varies with x for Chebyshev interpolation somewhat more than it does for splines. In fact, the *absolute* accuracy of Chebyshev interpolation varies much less with x , as can be seen by comparing Fig. 3 with Fig. 4. The opposite holds for splines, where the relative accuracy shows less variation than the absolute one. This reflects that Chebyshev interpolation is “global” over the full interpolation interval, whilst splines are quite “local” (although the continuity conditions for neighboring splines lead to some correlation over larger distances in x).

As is seen in Figs. 1, 2 and 3, the relative accuracy degrades in the limit $x \rightarrow 1$ for both splines and Chebyshev interpolation. This is not surprising, because in this limit the PDFs in Eq. (37) approach zero. Moreover, they behave like $(1-x)^\beta$ with noninteger β and are hence nonanalytic at $x = 1$. This behavior cannot be accurately reproduced by interpolating

⁸ Technically, we generate LHAPDF data files for the functions in Eq. (37) and then run the LHAPDF interpolation routines with the option `logcubic`. The LHAPDF data files are generated with double precision to avoid any artificial loss of numerical accuracy due to the intermediate storage step.

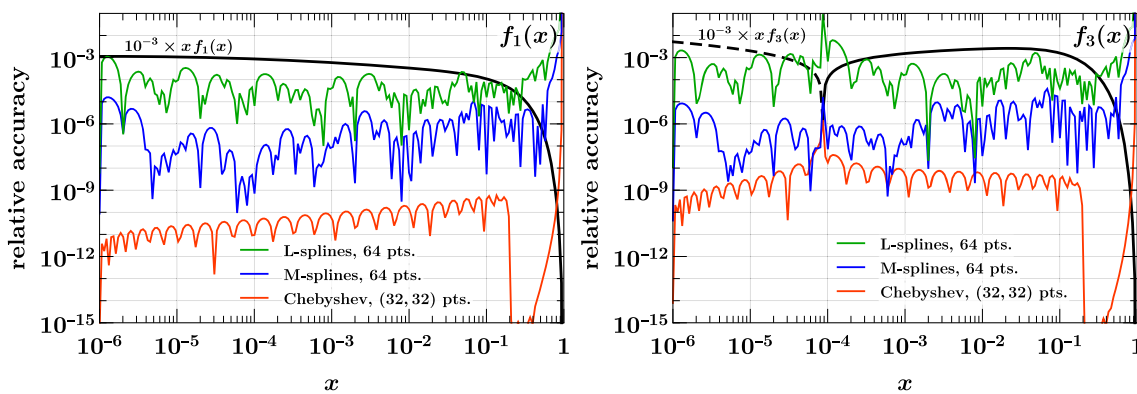


Fig. 1 Relative interpolation accuracy (38) for the sample PDFs $f_1(x)$ and $f_3(x)$ in Eq. (37). The spline interpolants (green and blue) use the low-density grid with $n_{\text{pts}} = 64$. The Chebyshev interpolation (red) uses

the grid $[10^{-6}, 0.2, 1]_{(32,32)}$, which has $n_{\text{pts}} = 63$. Here and in similar plots, the exact result that is being interpolated is shown in black (and dashed where the result is negative)

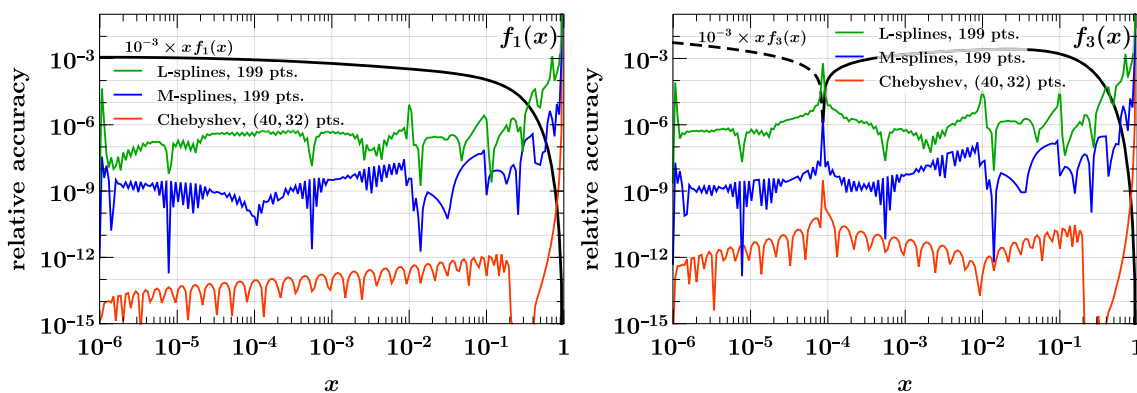


Fig. 2 As Fig. 1, but for denser grids. The spline interpolants (green and blue) use the high-density grid with $n_{\text{pts}} = 199$. The Chebyshev interpolation (red) uses the grid $[10^{-6}, 0.2, 1]_{(40,32)}$, which has $n_{\text{pts}} = 71$

polynomials in the vicinity of $x = 1$, whatever their degree. We emphasize that this problem concerns the *relative* interpolation accuracy. As seen in Fig. 4, the *absolute* error of interpolation does remain small for x up to 1, as is expected from Eq. (12). This is sufficient for many practical purposes, including the evaluation of convolution integrals that appear in cross sections or evolution equations. If high relative accuracy is required at large x , one needs to use subgrids with a sufficient number of points tailored to the region of interest.

3.3 Differentiation and integration

We now turn to Chebyshev interpolation for derivatives of the function $\tilde{f}(x) = xf(x)$. We use the barycentric formula (17) and its analog for the second derivative, multiplying of course with the Jacobian for the variable transformation from t to x . For comparison, we also take the numerical derivative $(\tilde{f}_{\text{sp}}(x+h) - \tilde{f}_{\text{sp}}(x-h))/2h$ of the L-spline interpolant $\tilde{f}_{\text{sp}}(x)$. We use a variable step size $h = 10^{-4}x$, having verified that the result remains stable when decreasing h even

further. The second derivative of \tilde{f}_{sp} is evaluated in an analogous way.

In Fig. 5 we consider the quantities

$$\begin{aligned} x^2 f'(x) &= x \tilde{f}'(x) - \tilde{f}(x), \\ x^3 f''(x) &= x^2 \tilde{f}''(x) - 2x \tilde{f}'(x) + 2\tilde{f}(x) \end{aligned} \quad (39)$$

and the accuracy of evaluating them using the two methods just described. We see that with Chebyshev interpolants, a significantly higher accuracy is obtained. This is not surprising, since the polynomials approximating the derivatives are of a high degree in that case, whereas with cubic splines, one locally has a quadratic polynomial for the first derivative and a linear approximation for the second derivative. For the low-density grid with $n_{\text{pts}} = 64$, the L-splines in fact give errors around 10% for the first and 100% for the second derivative in parts of the x range. We note that on each Chebyshev grid, the absolute accuracy of the derivatives (not shown here) has a milder variation in x than the relative one, following the pattern we saw in Fig. 4 for $xf(x)$ itself.

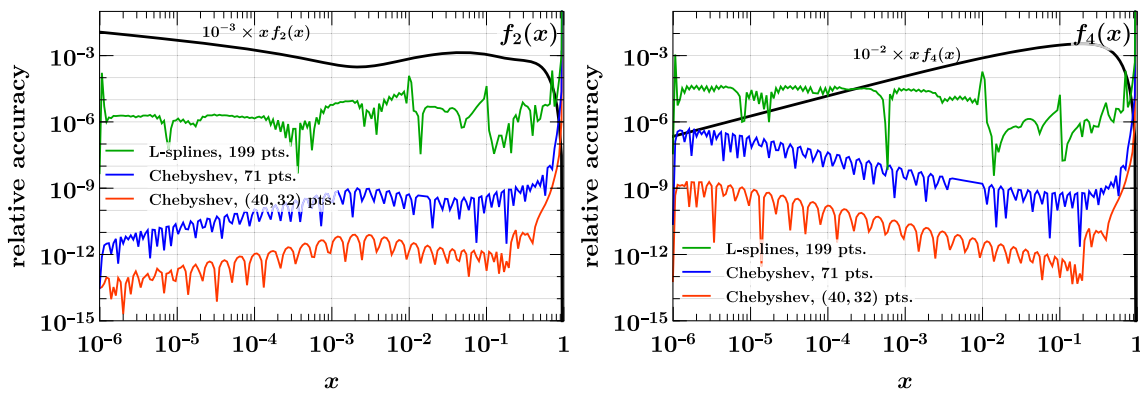


Fig. 3 Relative interpolation accuracy for the sample PDFs $f_2(x)$ and $f_4(x)$ in Eq. (37). The L-splines (green) use the high-density grid with $n_{\text{pts}} = 199$. The Chebyshev interpolations use grids with $n_{\text{pts}} = 71$, either with a single subgrid (blue) or with two subgrids $[10^{-6}, 0.2, 1]_{(40,32)}$ (red)

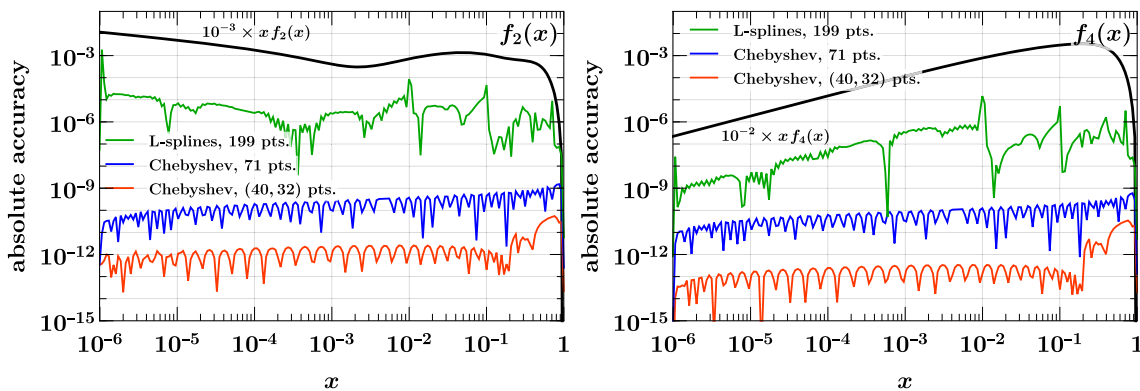


Fig. 4 As Fig. 3, but showing the absolute instead of relative interpolation accuracy for $xf(x)$, i.e. the absolute difference between the interpolated and exact results

To explore the accuracy of numerical integration, we consider the truncated Mellin moments

$$M_i(j) = \int_{x_0}^1 dx x^{j-1} f_i(x), \quad (40)$$

where the lower integration limit is the lowest point x_0 of the grid. In Fig. 6, we show the dependence of the relative accuracy on the total number of grid points when using 1 to 4 subgrids of equal size. The advantage of using subgrids is clearly seen, especially for high moment index j , which emphasizes the large x region of the integrand. We find that taking 2 or 3 subgrids gives best results for a wide range of j . The disadvantage of using interpolants with lower polynomial degree becomes the dominant limiting factor with 4 or more subgrids.

3.4 Estimating the numerical accuracy

Let us now take a closer look at methods to estimate the numerical accuracy of interpolation or integration with Chebyshev polynomials. An obvious option is to re-compute the quantity of interest with an increased number of points.

However, the examples in Fig. 6 show that the accuracy is not a strictly decreasing function of n_{pts} , and we see fluctuations with local minima and maxima as n_{pts} varies by about 10 units. For a reliable estimate, one should therefore take a sufficiently large increase in n_{pts} . Increasing n_{pts} in several steps provides an additional way of ensuring that the estimate is trustworthy.

The procedure just described can give sound accuracy estimates and is what we will adopt for assessing the accuracy of DGLAP evolution in Sect. 5.3. However, since the result must be evaluated multiple times on increasingly dense grids, this procedure can be computationally expensive, especially when the cost scales more than linearly with the number of grid points. This is indeed the case of DGLAP evolution, where the evaluation of convolution integrals involves $n_{\text{pts}} \times n_{\text{pts}}$ matrices.

An alternative with much less computational overhead is to estimate the accuracy from the difference between Chebyshev interpolation and interpolation on the same grid without the end points, as described in Sect. 2. This does not require any additional function evaluations. We compare this estimate with the actual interpolation accuracy in Fig. 7. In the subinterval for large x , the estimate is very close to the true

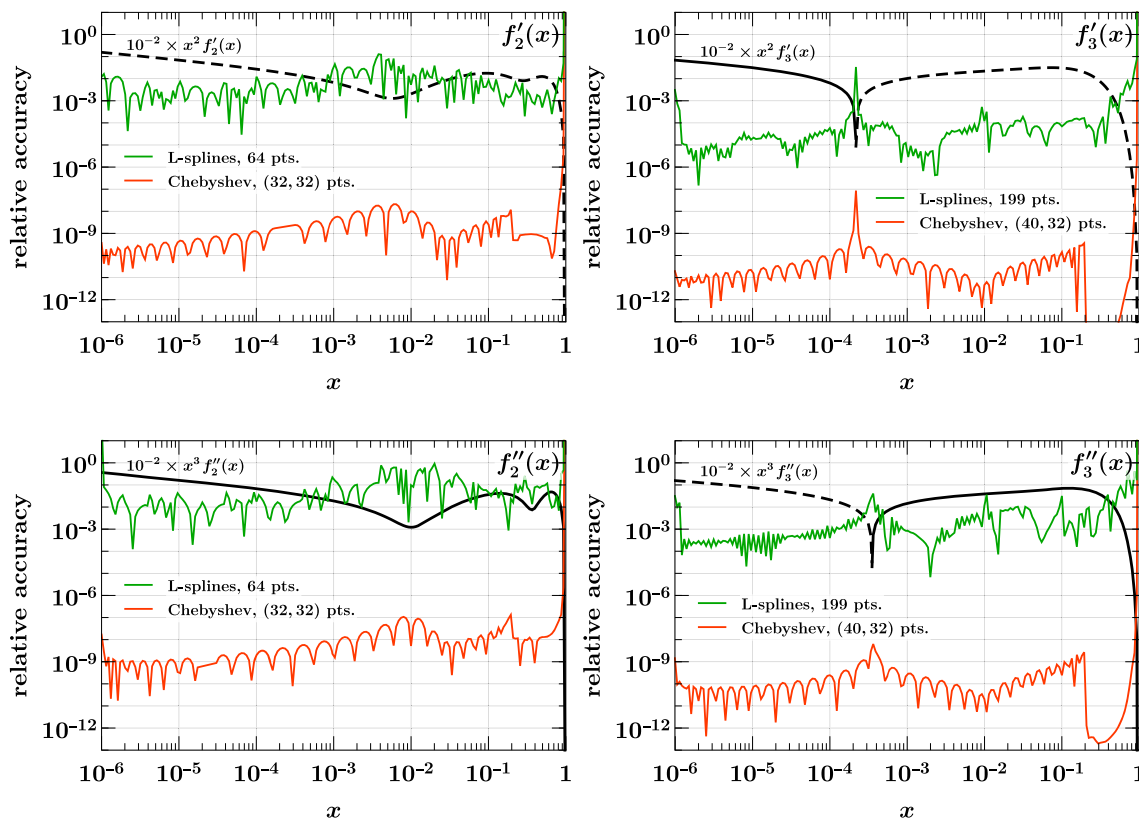


Fig. 5 Relative accuracy of the first derivative (top) and second derivative (bottom) for Chebyshev interpolation (red) and numerical differentiation of L-splines (green). The grids used on the left have lower n_{pts} as in Fig. 1, and those on the right have higher n_{pts} as in Fig. 2

error, whilst in the subinterval for small to intermediate x it somewhat overestimates the actual numerical error. The amount of overestimation is highest close to the interval limits, which is not surprising, because this is where the two interpolation methods differ most. The corresponding comparison for differentiation is shown in Fig. 8 and follows the same pattern. We also note that the quality of the estimate for other sample PDFs is as good as or even better than in the examples shown here.

The analog for integration of the previous method is to estimate the accuracy of Clenshaw–Curtis integration from the difference to the Fejér quadrature rule.⁹ In Fig. 9, we compare this estimate with the actual numerical error for the Mellin moment (40). We find that it tends to overestimate the actual error by two to three orders of magnitude, which is more than what we saw for interpolation. For comparison, we also show the relative accuracy obtained with the widely used Gauss–Kronrod rule, using the same two subintervals and the same number of grid points per subinterval as for Clenshaw–Curtis. As is commonly done, the accuracy estimate for Gauss–Kronrod is obtained from the difference between the nominal Gauss–Kronrod rule and the

lower-order Gauss rule on a subset of the grid points. We see that the Gauss–Kronrod rule has the highest accuracy. In practical applications, the true result is however not known, and an integrator is only as good as its accuracy estimation. Here, the accuracy estimate for Gauss–Kronrod is much worse than our estimate for Clenshaw–Curtis.¹⁰ As discussed at the end of Sect. 2, this is not unexpected given the comparison of polynomials orders, but it is interesting to see that the quantitative effect can be as pronounced as in our examples. We also note that the quantitative differences between the integration methods vary significantly with the shape of the integrands, as is evident from the two panels in Fig. 9.

Overall, as far as numerical accuracy estimates are concerned, using interpolation, differentiation, or integration without the end points of a Chebyshev grid can be considered as an inexpensive estimate of reasonable quality. It is not an alternative to the high-quality estimate one can achieve by a stepwise increase in the number of grid points. Its biggest advantage is thus to provide a fast and reliable indicator whether the accuracy of a result is sufficient or calls for a dedicated, more expensive investigation.

⁹ We always refer to what is known in the literature as Fejér’s *second* rule, given in Eq. (32).

¹⁰ Expert readers will know that it is in fact not uncommon for Gauss–Kronrod integrators to have overly conservative accuracy estimates.

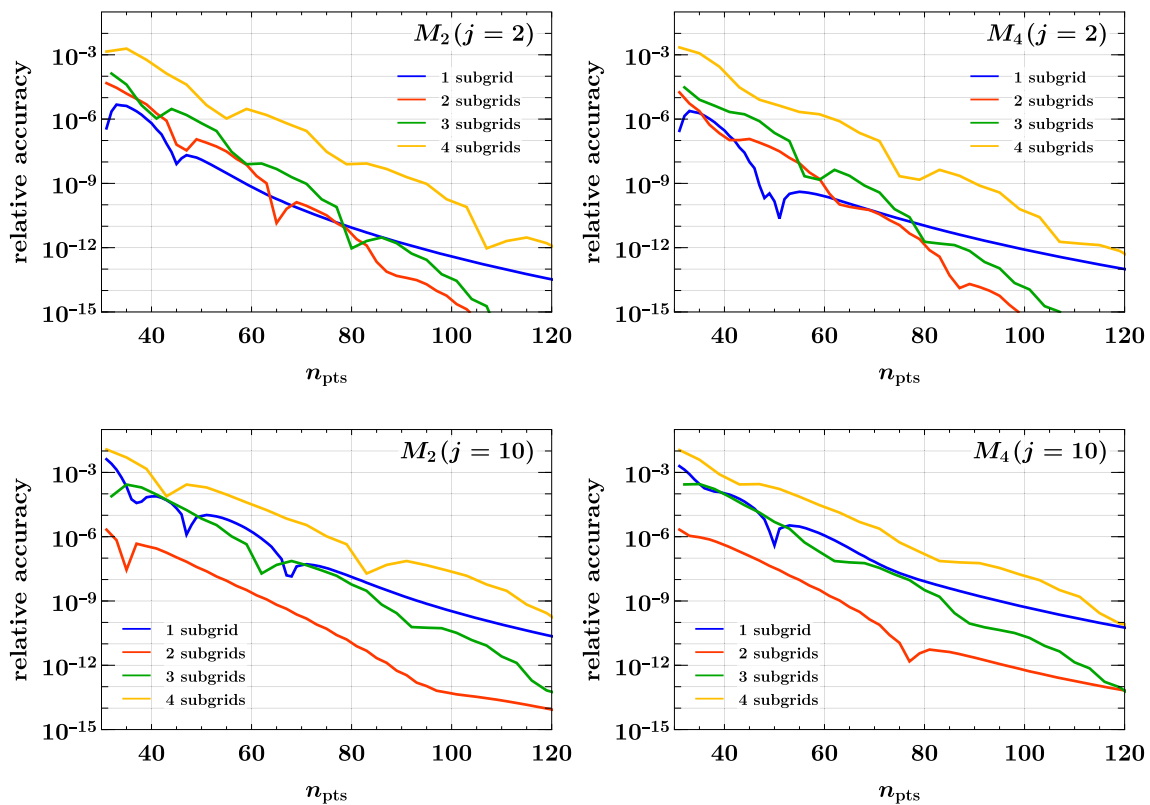


Fig. 6 Relative accuracy of the truncated Mellin moments (40) with $x_0 = 10^{-9}$ and with $j = 2$ (top) or $j = 10$ (bottom), for $f_2(x)$ (left) or $f_4(x)$ (right). We use Chebyshev grids with $k = 1$ to 4 subgrids

and n_{pts}/k points per subgrid, namely $[x_0, 1]$ (blue), $[x_0, 0.2, 1]$ (red), $[x_0, 10^{-3}, 0.5, 1]$ (green), and $[x_0, 10^{-6}, 10^{-3}, 0.5, 1]$ (yellow). The results for different n_{pts} are connected by lines to guide the eye

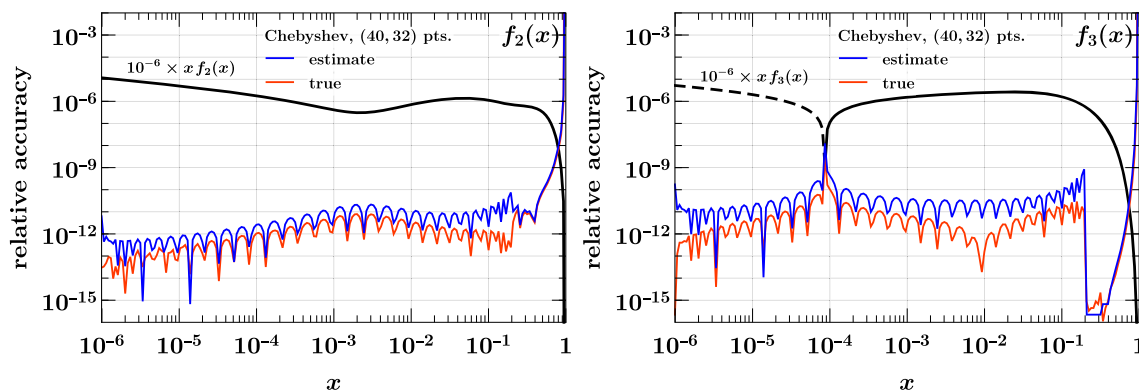


Fig. 7 Comparison of the true relative interpolation accuracy (red) and its estimate (blue) obtained from the difference between interpolation on the Chebyshev grid with and without its end points. The (40, 32)-point grid used is the same as in Fig. 2

4 Mellin convolution

In this section, we consider the Mellin convolution of a PDF with an integral kernel, such as a DGLAP splitting function, a beam-function matching kernel, or a hard-scattering coefficient. In terms of the scaled PDF $\tilde{f}(x) = xf(x)$, we wish to compute

$$(K \otimes \tilde{f})(x) = \int_x^1 \frac{dz}{z} K(z) \tilde{f}\left(\frac{x}{z}\right). \quad (41)$$

Here, $K(z)$ is the scaled kernel, given e.g. by $K(z) = zP(z)$ for a DGLAP splitting function $P(z)$.¹¹

¹¹ We recall that $h_1 = h_2 \otimes h_3$ implies $\tilde{h}_1 = \tilde{h}_2 \otimes \tilde{h}_3$ with $\tilde{h}_i(z) = zh_i(z)$.

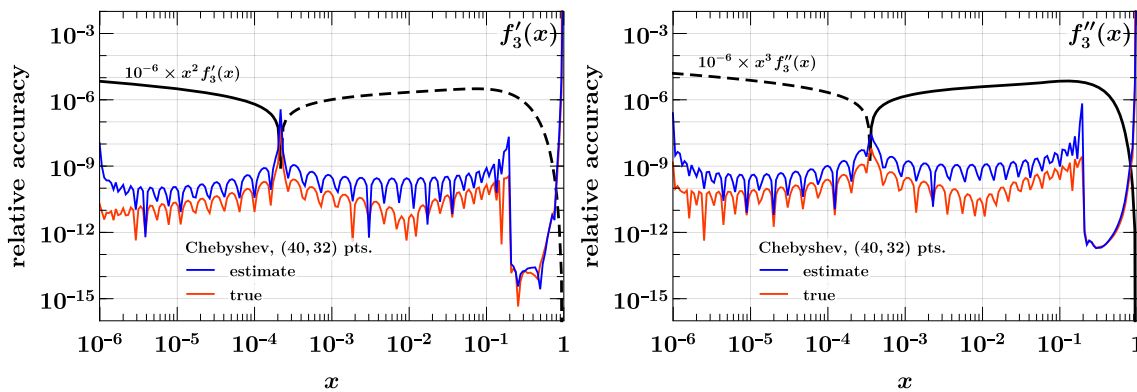


Fig. 8 Comparison of true (red) and estimated (blue) interpolation accuracy as in Fig. 7, but for the first and second derivative of the sample PDF $f_3(x)$

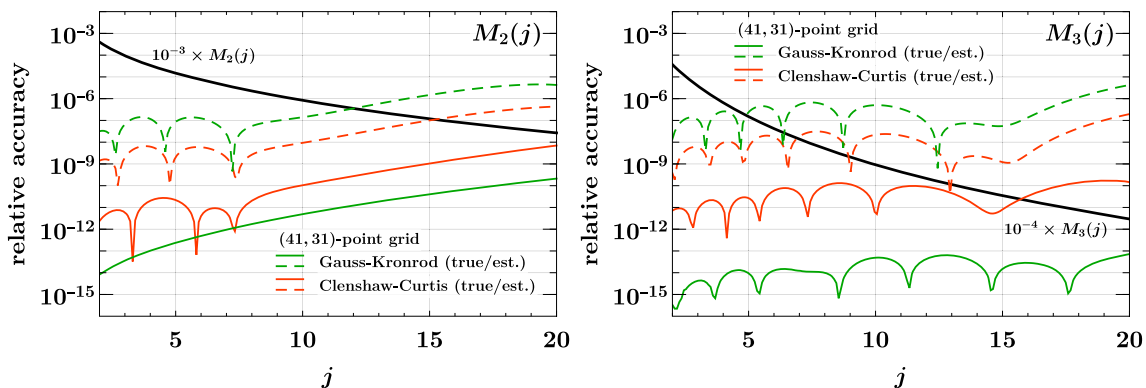


Fig. 9 Comparison of the relative integration accuracy (solid) and its estimate (dashed) for Clenshaw–Curtis (red) and Gauss–Kronrod (green) quadrature. The considered integrals are the truncated Mellin moments $M_i(j)$ of the test functions $f_i(x)$ with $i = 2, 3$. For both methods we use two subgrids $[10^{-9}, 0.2, 1]_{(41, 31)}$ with the appropriate

Let us see how Eq. (41) can be evaluated in a discretized setting. For simplicity, we first consider a single Chebyshev grid for the range $x_0 \leq x \leq 1$ as described in Sect. 3.1. The convolution $(K \otimes \tilde{f})(x)$ is a function of x over the same domain as $\tilde{f}(x)$. Hence, we can interpolate it in complete analogy to $\tilde{f}(x)$ itself. For this purpose, we only need to evaluate Eq. (41) at the grid points x_i ,

$$\begin{aligned} (K \otimes \tilde{f})(x_i) &= \int_{x_i}^1 \frac{dz}{z} K(z) \tilde{f}\left(\frac{x_i}{z}\right) \\ &\approx \int_{x_i}^1 \frac{dz}{z} K(z) \sum_{j=0}^N \tilde{f}_j b_j \left(\ln \frac{x_i}{z}\right), \end{aligned} \quad (42)$$

where in the last step we used the barycentric formula (35) to interpolate $\tilde{f}(x_i/z)$ under the integral. Introducing the *kernel matrix*

$$K_{ij} = \int_{x_i}^1 \frac{dz}{z} K(z) b_j \left(\ln \frac{x_i}{z}\right), \quad (43)$$

Chebyshev or Gauss–Kronrod points for the integration variable $\ln x$. In the accuracy estimates, the results for a given integration rule are first added for the two subintervals, and then the absolute difference between the involved higher-order and lower-order rules is taken

we can compute the function values of $K \otimes \tilde{f}$ on the grid by a simple matrix multiplication,

$$(K \otimes \tilde{f})(x_i) \approx \sum_{j=0}^N K_{ij} \tilde{f}_j. \quad (44)$$

Importantly, the matrix K_{ij} only depends on the given kernel and grid but not on \tilde{f} . It can thus be pre-computed once using standard numerical integration routines.

In general, $K(z)$ is a distribution rather than an ordinary function. Restricting ourselves to plus and delta distributions, we write

$$K(z) = K_{\text{sing}}(z) + K_{\text{reg}}(z) + \delta(1-z) K_{\delta}, \quad (45)$$

where $K_{\text{sing}}(z)$ is the singular part of the kernel and contains plus distributions. The function $K_{\text{reg}}(z)$ contains at most an integrable singularity at $z = 1$, for instance powers of $\ln(1-z)$. The separation between K_{sing} and K_{reg} is not unique and may be adjusted as is convenient. The convolution (41) can now be written as

$$(K \otimes \tilde{f})(x) = \int_x^1 \frac{dz}{z} K_{\text{sing}}(z) \left[\tilde{f}\left(\frac{x}{z}\right) - \tilde{f}(x) \right] + \int_x^1 \frac{dz}{z} K_{\text{reg}}(z) \tilde{f}\left(\frac{x}{z}\right) + K_d(x) \tilde{f}(x), \quad (46)$$

where

$$K_d(x) = K_\delta + \int_x^1 \frac{dz}{z} K_{\text{sing}}(z) \quad (47)$$

can be evaluated analytically if the separation between K_{sing} and K_{reg} is chosen appropriately. Due to the explicit subtraction of $\tilde{f}(x)$ in the first integral in Eq. (46), the plus prescription in K_{sing} can now be omitted, because

$$\int_x^1 dz [g(z)]_+ h(z) = \int_x^1 dz g(z) h(z) \quad \text{for } h(1) = 0. \quad (48)$$

From the definition (47) one readily finds that $K_d(x)$ diverges like $\ln^{n+1}(1-x)$ for $x \rightarrow 1$ if $K(z)$ contains a term $\mathcal{L}_n(1-z)$, where we write

$$\mathcal{L}_n(y) = \left[\frac{\ln^n(y)}{y} \right]_+ \quad (49)$$

for the logarithmic plus distribution of degree n . However, a PDF vanishes much faster for $x \rightarrow 1$ than $K_d(x)$ diverges, such that the product $K_d(x) \tilde{f}(x)$ tends to zero in that limit.

Applying Eq. (46) to the evaluation of the K_{ij} matrix in Eq. (43), we have

$$K_{ij} = \int_{u_i}^0 dv K_{\text{sing}}(e^v) [b_j(u_i - v) - \delta_{ij}] + \int_{u_i}^0 dv K_{\text{reg}}(e^v) b_j(u_i - v) + \delta_{ij} K_d(x_i), \quad (50)$$

where u_i and u_j are grid points in u , and where we have changed the integration variable to $v = \ln z$. The plus prescription in K_{sing} can again be omitted, because the term in square brackets vanishes at least linearly in v for $v \rightarrow 0$. Hence, all integrals can be evaluated numerically, for which we use an adaptive Gauss–Kronrod routine.

Given that $x_N = 1$, the matrix element K_{NN} is ill defined if K contains a plus distribution, because $K_d(x)$ diverges for $x \rightarrow 1$ in that case. This does not present any problem: we already noticed that $K_d(x) \tilde{f}(x) \rightarrow 0$ for $x \rightarrow 1$, which translates to $K_{NN} \tilde{f}_N = 0$ in the discretized version. This term may hence be omitted in the matrix multiplication (44), along with all other terms $K_{iN} \tilde{f}_N$.

It is not difficult to generalize the preceding discussion to the case of several subgrids in u , each of which is mapped

onto a Chebyshev grid. One then has a distinct set of barycentric basis functions for each subgrid. If u_j and $u_i - v$ are not on the same subgrid, then the basis function $b_j(u_i - v)$ in Eq. (50) must be set to zero. As a consequence, the matrix K_{ij} has blocks in which all elements are zero. Let us, however, note that K_{ij} always has nonzero elements below the diagonal $i = j$ and is *not* an upper triangular matrix.

We now study the numerical accuracy of this method relative to the exact result, which we obtain by the direct numerical evaluation of the convolution integral (46). For comparison, we also show the accuracy of the result obtained by approximating $\tilde{f}(x)$ in Eq. (46) with its L-splines interpolant and performing the convolution integral numerically. All explicit numerical integrals in this study are performed at sufficiently high precision to not influence the results.

In Fig. 10, we show the results of this exercise for the convolution of our sample PDFs $f_i(x)$ with kernels that have different singular behavior at the end point $z = 1$. The kernel in the top row is the leading-order DGLAP splitting function P_{gg} , which contains a term $\mathcal{L}_0(1-z)$. The kernel in the second row is $\ln^4(1-z)$, which is the most singular term in the three-loop splitting functions P_{gq} and P_{qg} [63,64]. In the bottom row, we take $\mathcal{L}_5(1-z)$ as kernel, which appears in the $N^3\text{LO}$ corrections to the rapidity spectrum of inclusive Higgs production in pp collisions [15]. In all cases, the accuracy of our Chebyshev based method is several orders of magnitude higher than the one obtained when interpolating PDFs with L-splines and then performing the convolution integral.

The numerical inaccuracies for the convolution with $\mathcal{L}_5(1-z)$ using L-splines are in the percent range over a wide range of x and much higher than those for the other kernels we looked at. This is in line with the results of Ref. [15], which finds that the convolution of $\mathcal{L}_5(1-z)$ with PDFs from the LHAPDF library can lead to considerable numerical instabilities. The PDFs used in that work are those of the NNPDF3.0 analysis [65], and for the sake of comparison we use the corresponding grid for spline interpolation in the plots on the right of Fig. 10.

An alternative method to evaluate the convolution is to use the Chebyshev interpolant of \tilde{f} and perform the convolution integral itself numerically, analogous to what we did with L-splines above. This avoids the additional interpolation of $(K \otimes \tilde{f})$ that happens in the above kernel matrix method. The numerical accuracy of this alternative method differs slightly from the accuracy of the kernel matrix method, but it is of the same order of magnitude. Hence, the additional interpolation of $(K \otimes \tilde{f})$ does not introduce a significant penalty in accuracy beyond that of interpolating \tilde{f} itself. This makes the kernel matrix method far more attractive, as it avoids the evaluation of a numerical integral for each desired \tilde{f} and x .

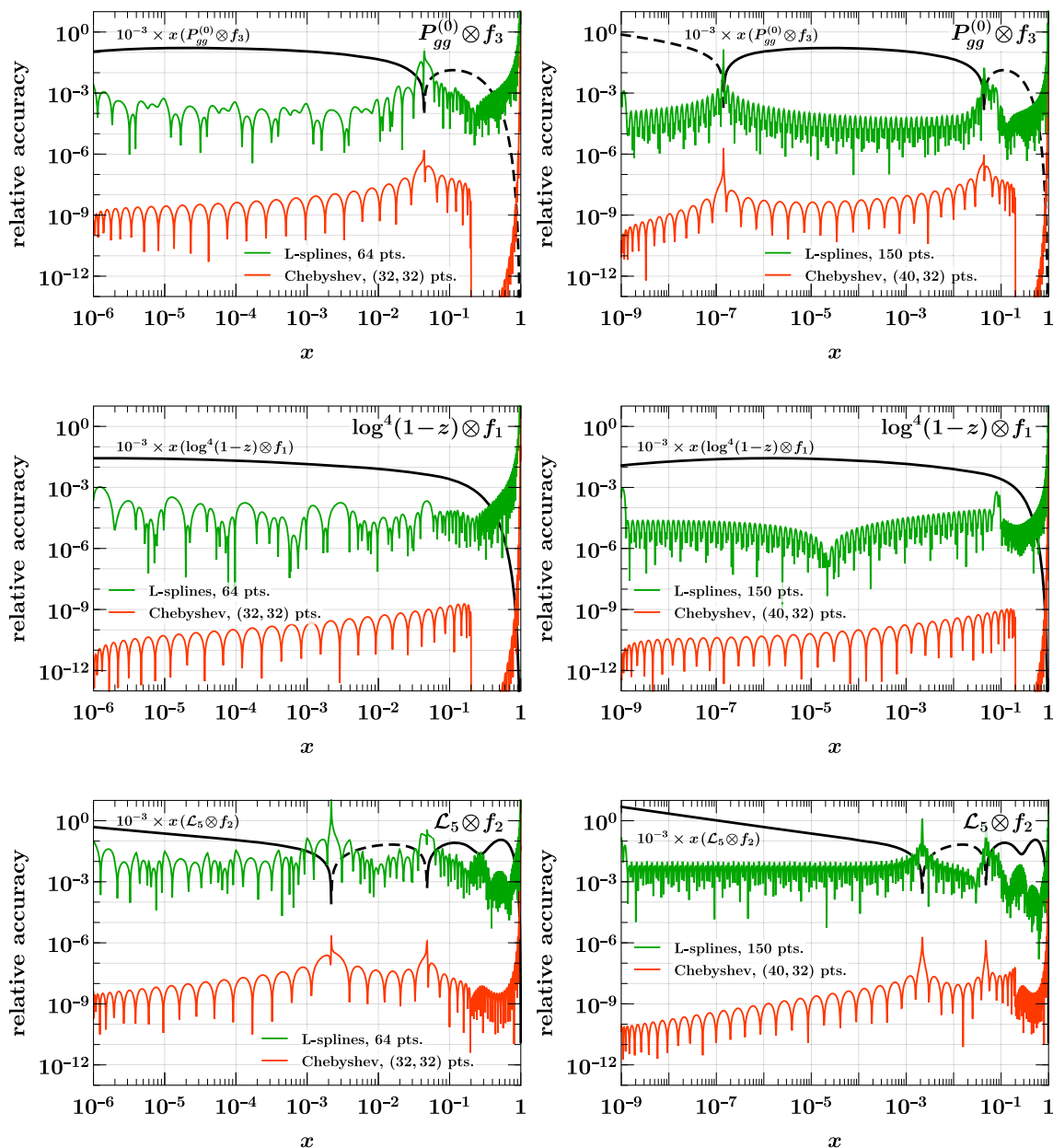


Fig. 10 Relative accuracy of Mellin convolutions. The red curves correspond to Chebyshev interpolation for both the PDF and the convolution result, as described in the text. The green curves are obtained by performing the numerical convolution integral for the PDF interpolated with L-splines. Details about the convolution kernels are given in the

text. On the left, we use the same grids as in Fig. 1, with $n_{\text{pts}} = 63$ for Chebyshev interpolation and $n_{\text{pts}} = 64$ for L-splines. On the right, we use the Chebyshev grid $[10^{-9}, 0.2, 1]_{(40,32)}$ with $n_{\text{pts}} = 71$ and for L-splines a grid with $n_{\text{pts}} = 150$ (corresponding to the NNPDF3.1 grid in Table 1)

5 DGLAP evolution

5.1 Numerical solution of DGLAP equations

In this section, we present our approach to the numerical solution of the DGLAP evolution equations [66–68]. Up to order α_s^{n+1} , they read

$$\frac{df(x, \mu)}{d \ln \mu^2} = \sum_{m=0}^n \left[\frac{\alpha_s(\mu)}{4\pi} \right]^{m+1} (P^{(m)} \otimes f(\mu))(x), \quad (51)$$

where $P^{(m)}$ is the splitting function at order m . For notational simplicity, we suppress the indices for the parton type and the associated sum on the right-hand side. To solve Eq. (51), we

discretize the scaled PDFs $\tilde{f}(x) = xf(x)$ on a Chebyshev grid and evaluate the Mellin convolutions on the right-hand side as discussed in Sect. 4. The integro-differential equation (51) then turns into a coupled system of ordinary differential equations (ODEs),

$$\frac{d\tilde{f}_i(\mu)}{d\ln\mu^2} = \sum_{j=0}^N \sum_{m=0}^n \left[\frac{\alpha_s(\mu)}{4\pi} \right]^{m+1} \tilde{P}_{ij}^{(m)} \tilde{f}_j(\mu), \quad (52)$$

where $\tilde{f}_i(\mu) = \tilde{f}(x_i, \mu)$ and $\tilde{P}_{ij}^{(m)}$ is the kernel matrix for $K(z) \equiv zP^{(m)}(z)$ as defined in Eq. (43). This system of ODEs can be solved numerically using the standard Runge–Kutta algorithm, which is described in more detail in Appendix A. This formulation as a linear system of ODEs by discretization in x is common to many approaches that solve the DGLAP equations in x space [17–19, 25–27]. By using the Chebyshev grid for the discretization in x , the resulting evolved PDF can then be Chebyshev interpolated.

The Runge–Kutta algorithm uses a discretization of the evolution variable t (not to be confused with the argument of Chebyshev polynomials in Sect. 2). It is advantageous if the function multiplying \tilde{f} on the right-hand side of the evolution equation depends only weakly on t , because this tends to give a uniform numerical accuracy of evolution with a fixed step size in t . We therefore evolve in the variable

$$t = -\ln\alpha_s(\mu) \quad (53)$$

instead of $\ln\mu^2$. For the running coupling at order n , we write

$$\frac{d\alpha_s^{-1}}{d\ln\mu^2} = -\frac{\beta(\alpha_s)}{\alpha_s^2} = \sum_{m=0}^n \frac{\beta_m}{4\pi} \left(\frac{\alpha_s}{4\pi} \right)^m \quad (54)$$

and use a Runge–Kutta routine (see the end of Appendix A) to solve for α_s^{-1} as a function of $\ln\mu^2$. At leading order, the DGLAP equation (51) then becomes

$$\frac{df(t)}{dt} = \frac{1}{\beta_0} P^{(0)} \otimes f(t). \quad (55)$$

with no explicit t dependence on the right-hand side. Using $\alpha_s = e^{-t}$, we have at NLO

$$\frac{df(t)}{dt} = \frac{1}{\beta_0 + (\beta_1/4\pi)e^{-t}} \left[P^{(0)} + \frac{P^{(1)}}{4\pi} e^{-t} \right] \otimes f(t), \quad (56)$$

where an explicit t dependence appears in the two-loop terms with β_1 and $P^{(1)}$. The corresponding equations at NNLO and higher are easily written down.

The pattern of mixing between quarks, antiquarks, and gluons under DGLAP evolution is well known. Its structure at NNLO is given in Refs. [63, 69] and remains the same at

higher orders. To reduce mixing to a minimum, we work in the basis formed by

$$g, \quad \Sigma^\pm = \sum_i q_i^\pm, \quad u^\pm - d^\pm, \quad d^\pm - s^\pm, \\ s^\pm - c^\pm, \quad c^\pm - b^\pm, \quad b^\pm - t^\pm, \quad (57)$$

where $q^\pm = q \pm \bar{q}$. Contrary to the often-used flavor non-singlet combinations $u + d - 2s$, $u + d + s - 3c$, etc., the differences between consecutive flavors in Eq. (57) are less prone to a loss of numerical accuracy due to rounding effects in regions where the strange and heavy-quark distributions are much smaller than their counterparts for u and d quarks. Note that at NNLO, the combination Σ^- has a different evolution kernel than the flavor differences in the second line of Eq. (57). This is due to graphs that have a t channel cut involving only gluons.

We implement the unpolarized splitting functions at LO and NLO in their exact analytic forms. For the unpolarized NNLO splitting functions, we use the approximate expressions given in Refs. [63, 69], which are constructed from a functional basis containing only the distributions $\mathcal{L}_0(1-z)$, $\delta(1-z)$ and polynomials in z , $1/z$, $\ln z$, and $\ln(1-z)$. With these parameterizations, the numerical evaluation of the kernel matrices in all channels takes about as long for $P^{(2)}$ as it does for $P^{(1)}$.¹²

Both α_s and the PDFs depend on the number n_F of quark flavors that are treated as light and included in the $\overline{\text{MS}}$ renormalization of these quantities. For the conversion between α_s for different n_F , we use the matching conditions in Ref. [70] at the appropriate order. For the transition between PDFs with different n_F , we implement the matching kernels given in Ref. [71], which go up to order α_s^2 . We have verified that they agree with the independent calculation in Ref. [72]. The Mellin convolutions of matching kernels with PDFs are also evaluated as described in Sect. 4.

5.2 Validation

To validate our DGLAP evolution algorithm, we compare our results with the benchmark tables in section 1.33 of Ref. [73] and section 4.4 of Ref. [74]. These tables contain PDFs evolved to the scale $\mu = 100$ GeV from initial conditions given in analytic form at $\mu_0 = \sqrt{2}$ GeV. Evolution is performed at LO, NLO and NNLO, either at fixed $n_F = 4$ or with a variable number of flavors $n_F = 3 \dots 5$ and flavor transitions at scales μ_c and μ_b . The default choice is $\mu_c = m_c = \sqrt{2}$ GeV and $\mu_b = m_b = 4.5$ GeV. We com-

¹² Using the approximate, parameterized NNLO splitting functions is a matter of convenience and for compatibility with the benchmark results below. It is also possible to use the exact expressions, since the kernel matrices are evaluated only once.

pare with the LO and NLO results in Ref. [73] and with the NNLO results in Ref. [74]. The latter are obtained with the same parameterized NNLO splitting functions that we use. Furthermore, the parameterization in eq. (3.5) of Ref. [22] is used for the two-loop matching kernel for the transition from a gluon to a heavy quark. We also use this parameterization for the sake of the present comparison, noting that visible differences with the benchmark results appear when we use the exact analytic form for this kernel instead.

The benchmark tables were obtained using two programs, HOPPET [25] and QCD-PEGASUS [22], both run with high-precision settings. The former solves the evolution equations in x space, whilst the latter uses the Mellin moment technique. The tables give results for the evolved PDFs at 11 values of x between 10^{-7} and 0.9. The results are given with five significant digits, with the exception of several sea-quark combinations at $x = 0.9$, which are very small and are given to only four significant digits.

As specified in Ref. [73], evolution with HOPPET was performed by using seventh-order polynomials for the interpolation on multiple x grids spanning the interval $[10^{-8}, 1]$ with a total of $n_{\text{pts}} = 1,170$. A uniform grid in $\ln \mu^2$ was used, with 220 points between $\mu = \sqrt{2}$ GeV and 1000 GeV. The results were verified to be stable within a 10^{-5} relative error for $x < 0.9$ by comparing them with the results of the same program with half the number of points on both the x and the μ grids. Furthermore, they were compared with the results obtained with QCD-PEGASUS.

For the comparison with our approach, we use a Chebyshev grid with 3 subgrids $[10^{-8}, 10^{-3}, 0.5, 1]_{(24,24,24)}$ which has $n_{\text{pts}} = 70$. The DGLAP equations are solved using a Runge–Kutta algorithm with the DOPRI8 method (see Appendix A) and a maximum step size in t of $h = 0.1$. This amounts to about 12 Runge–Kutta steps from the starting scale to $\mu = 100$ GeV. Our results for the benchmark numbers (rounded as stated above) remain the same if we take 40 instead of 24 points for each subgrid in x . They also remain unchanged if we use the same Runge–Kutta method with maximum step size $h = 0.3$ or $h = 0.02$.

We compare our results with the numbers reported in tables 2, 3, 4 of Ref. [73] and in tables 14 and 15 of Ref. [74], which cover unpolarized evolution both with fixed $n_F = 4$ and with variable $n_F = 3 \dots 5$. Whilst the tables also give results for evolution with different scales μ_r in α_s and μ_f in the PDFs, we always set $\mu_r = \mu_f$. We agree with all benchmark numbers,¹³ except for those given in our Tables 2 and 3. In all cases where we differ, the differences can be attributed to the benchmark results and fall into two categories:

1. The benchmark tables contain a number of entries, marked by an asterisk, for which the results of the two used codes differ in the sixth digit and give different numbers when rounded to five digits. The number with the smaller modulus is then given in the tables. In several cases, our result agrees with number with the larger modulus and in this sense agree with the benchmark results.
2. We differ from the benchmark numbers in five more cases. For the two cases at NNLO, the benchmark results have typographical errors in the exponent. In the other three cases, we differ by one unit in the fifth digit. We contacted the authors of the benchmark tables, who confirmed that indeed their respective codes agree with our numbers [75].

5.3 Numerical accuracy

We now study the numerical accuracy of our method in some detail. We use $x_0 = 10^{-7}$ and 3 subgrids. The evolution equations are solved with the DOPRI8 method (see Appendix A). To assess the accuracy, we compare three settings with different numbers of grid points and different Runge–Kutta steps h :

1. $[10^{-7}, 10^{-2}, 0.5, 1]_{(24,24,24)}$ ($n_{\text{pts}} = 70$) and $h = 0.1$,
2. $[10^{-7}, 10^{-2}, 0.5, 1]_{(40,40,40)}$ ($n_{\text{pts}} = 118$) and $h = 0.1$,
3. $[10^{-7}, 10^{-2}, 0.5, 1]_{(40,40,40)}$ ($n_{\text{pts}} = 118$) and $h = 0.004$.

To estimate the numerical error due to the discretization in x , we take the difference between settings 1 and 2, whereas the error due to the Runge–Kutta algorithm is estimated from the difference between settings 2 and 3. For the combined error, we take the difference between settings 1 and 3. Note that these estimates correspond to the accuracy of setting 1. We use the same initial conditions at $\mu_0 = \sqrt{2}$ GeV as in the benchmark comparison described in the previous subsection. Starting at $n_F = 3$, heavy flavors are added at $m_c = \sqrt{2}$ GeV, $m_b = 4.5$ GeV, and $m_t = 175$ GeV. In the remainder of this section, we always evolve and match at NNLO.

The combined discretization and Runge–Kutta accuracy for evolution to $\mu = 100$ GeV and $\mu = 10$ TeV is shown in Figs. 11 and 12, respectively, both for the individual parton flavors g, q, \bar{q} , and for the valence combinations $q^- = q - \bar{q}$. The relative accuracy is better than 10^{-7} up to $x \leq 0.8$, and much better than that for smaller x . The same holds when we evolve to $\mu = 1.01 m_c$ or $\mu = 1.01 m_b$, where the charm or bottom distributions are very small. The relative accuracy of u_v and d_v increases towards small x . This reflects the strong decrease of these distributions in the small- x limit, as we already observed and explained in Sect. 3. The combinations $s^-, c^-, b^-,$ and t^- show less variation at small x , and so does their relative accuracy.

¹³ Note that the value of $\alpha_s(100 \text{ GeV})$, evolved at LO with $n_F = 4$, is mistyped in table 2 of Ref. [73] and corrected in table 16 of Ref. [74].

Table 2 Results for evolution with $n_F = 4$ for which we differ from the numbers in the benchmark tables in Ref. [73]. The notation for PDF combinations is $L_{\pm} = \bar{d} \pm \bar{u}$ and $q_+ = q + \bar{q}$ for $q = s, c$. The number format a^b is shorthand for $a \times 10^b$. As discussed in the text, all differ-

ences can be attributed to the benchmark results, where entries with an asterisk correspond to category 1 and the entry in blue corresponds to category 2. We note that in the table headers of Ref. [73] it should read $2xL_+$ instead of xL_+

order	x	xu_v	xd_v	xL_-	$2xL_+$	xs_+	xc_+	g
LO	0.5					7.3137 ⁻⁴		
	0.9						4.8894 _* ⁻⁹	
NLO	10 ⁻⁷					6.6914 _* ⁺¹		1.1484 _* ⁺³
	10 ⁻⁴							9.2873 _* ⁺¹
	10 ⁻³				6.1649 _* ⁺⁰			
	10 ⁻²					8.4221 _* ⁻¹		

Table 3 As Table 2, but for evolution and flavor matching from $n_F = 3 \dots 5$. The corresponding benchmark tables are in Ref. [73] for LO and NLO and in Ref. [74] for NNLO

order	x	xu_v	xd_v	xL_-	$2xL_+$	xs_+	xc_+	xb_+	g
LO	10 ⁻⁷							4.6071 ⁺¹	
NLO	10 ⁻⁶								5.2290 _* ⁺²
	0.7	2.0102 ⁻²							
NNLO	10 ⁻⁷				1.0699 ⁻⁴				9.9694 ⁺²

In Fig. 13 we show examples for the separate errors due to discretization and the Runge–Kutta algorithm. With our settings, the overall numerical accuracy is entirely determined by the discretization in x , whilst the inaccuracy due to the Runge–Kutta algorithm can be neglected. The Runge–Kutta accuracy for u_v is in fact determined by the machine precision except for very large x , as signaled by the noisy behavior of the error curve.

It is often found that backward evolution, i.e. evolution from a higher to a lower scale, is numerically unstable.¹⁴ The structure of the DGLAP equations is such that the relative uncertainties (physical or numerical) of PDFs become larger when one evolves from a scale μ_1 to a lower scale μ_0 . This property is shared by other renormalization group equations in QCD, including the one for $\alpha_s(\mu)$. To which extent it leads to numerically unreliable results is, however, a separate question.

To study backward evolution within our method, we perform the following exercise. We start with the input PDFs of the benchmark comparison, evolve them with $n_F = 4$ from $\sqrt{2}$ GeV to $m_b/2 = 2.25$ GeV and match to $n_F = 5$ at that

point.¹⁵ The result is taken as initial condition for $n_F = 5$ evolution from $\mu_0 = 2.25$ GeV to $\mu_1 = 1$ TeV (step 1). The result of step 1 is evolved down to μ_0 (step 2), and the result of step 2 is again evolved up to μ_1 (step 3). We thus have two quantities that are sensitive to the accuracy of backward evolution:

- the difference between the output of step 2 and the input to step 1, which corresponds to the evolution path $\mu_0 \rightarrow \mu_1 \rightarrow \mu_0$,
- the difference between the output of step 3 and the input to step 2, corresponding to the evolution path $\mu_1 \rightarrow \mu_0 \rightarrow \mu_1$.

The relative accuracy for the two evolution paths is shown in Fig. 14 for the DOPRI8 method with maximum step size $h = 0.1$. In both cases, we find very high accuracy, in some cases near machine precision as signaled by the noisy behavior of the curves. Relative errors tend to be larger for the evolution path $\mu_0 \rightarrow \mu_1 \rightarrow \mu_0$, but they are all below 10^{-8} for $x \leq 0.9$

¹⁴ See, however, Ref. [76] for an early study that concluded the contrary.

¹⁵ Note that by matching at $\mu = m_b/2$, we obtain nonzero values for the b and \bar{b} distributions.

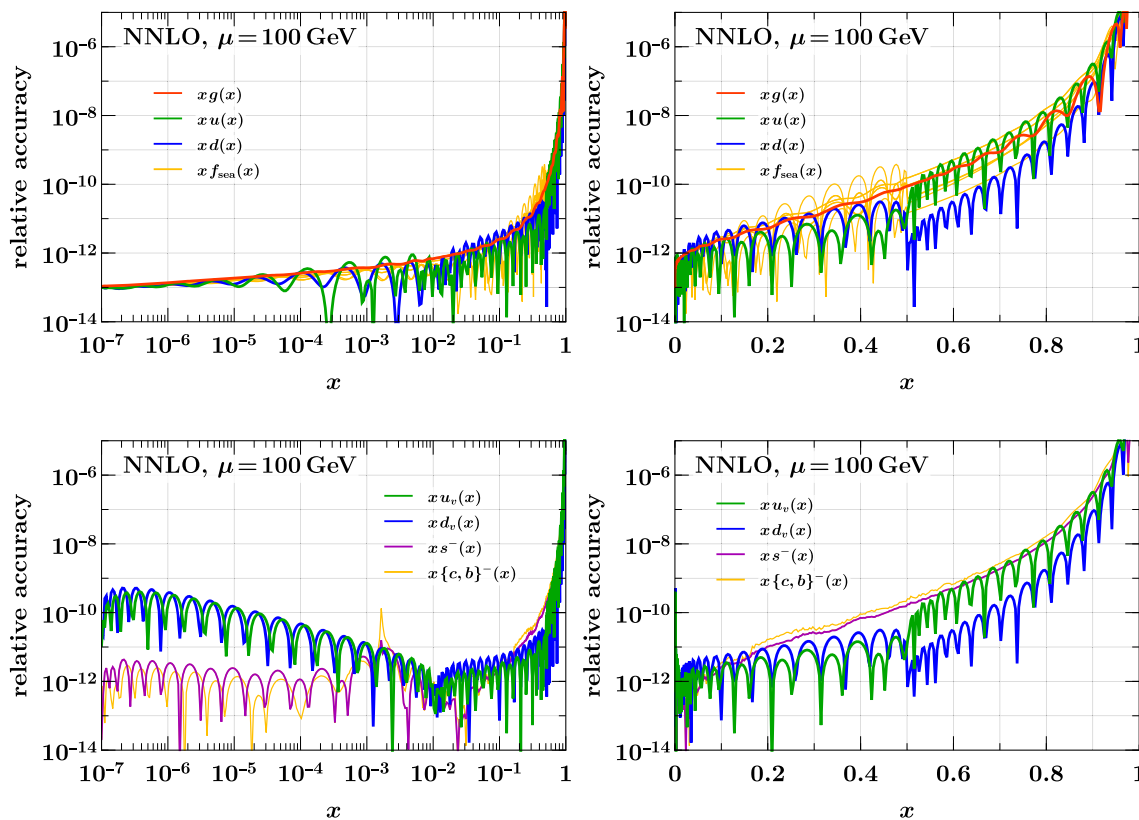


Fig. 11 Relative numerical accuracy of NNLO DGLAP evolution and flavor matching from $\mu_0 = \sqrt{2}$ GeV and $n_F = 3$ to $\mu = 100$ GeV and $n_F = 5$. We use the PDFs defined in Refs. [73,74]. The top

row shows quark, antiquark, and gluon distributions (with $f_{\text{sea}} \in \{\bar{u}, \bar{d}, s, \bar{s}, c, \bar{c}, b, \bar{b}\}$), and the bottom row shows the differences $q - \bar{q}$

(except in the vicinity of zero crossings). Note that a high accuracy is also obtained for the small combinations s^- , c^- , and b^- , which are induced by evolution at order α_s^3 .

We have also repeated our exercise with the widely-used RK4 method and $h = 0.03$, which requires approximately the same number of function calls as DOPRI8 with $h = 0.1$ (see the appendix for more detail). This yields relative errors that are orders of magnitude larger for both evolution paths, but still below 10^{-5} for $x \leq 0.9$ and away from zero crossings. This shows the significant benefit of using Runge–Kutta methods with high order for DGLAP evolution. Nevertheless, even with the standard RK4 method, we find no indication for numerical instabilities of backward evolution in our setup.

Finally, we note that a Runge–Kutta method with less than the 13 stages of DOPRI8 can be useful if one needs to perform evolution in many small steps, for instance when computing jet cross sections with $\mu \sim p_T$ for a dense grid in the transverse jet momentum p_T . Setting $h \sim 0.03$, we find the DOPRI6 method with its 8 stages to be well suited for such situations.

6 Conclusions

We have shown that a global interpolation using high-order Chebyshev polynomials allows for an efficient and highly accurate numerical representation of PDFs. Compared with local interpolation methods on equispaced grids, such as splines, our method can reach much higher numerical accuracies whilst keeping the computational cost at a comparable or reduced level.

Not only interpolation, but also differentiation and integration of PDFs are numerically accurate and computationally simple in our approach. The same holds for the Mellin convolution of a PDF with an integral kernel, which can be implemented as a simple multiplication with a pre-computed matrix. In particular, high accuracy is retained if the kernel has a strong singularity at the integration end point, such as $[\ln^5(1-z)/(1-z)]_+$ or $\ln^4(1-z)$. Even for such demanding applications, it is possible to achieve a numerical accuracy below 10^{-6} with only about 60 to 70 grid points. Hence, with our method, numerical inaccuracies become safely negligible for practical physics calculations involving PDFs. If desired, the inaccuracy caused by the interpolation can be estimated with modest additional computational effort by using interpo-

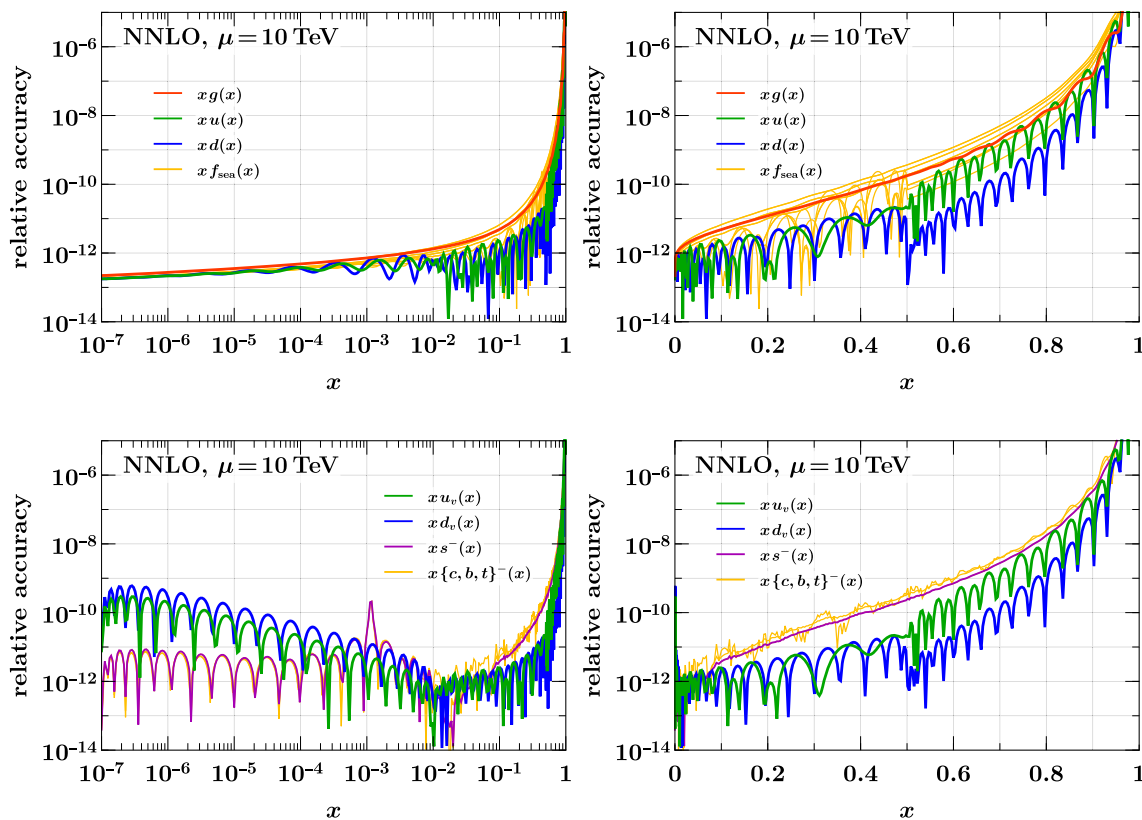


Fig. 12 As Fig. 11, but for evolution and flavor matching up to $\mu = 10$ TeV and $n_F = 6$

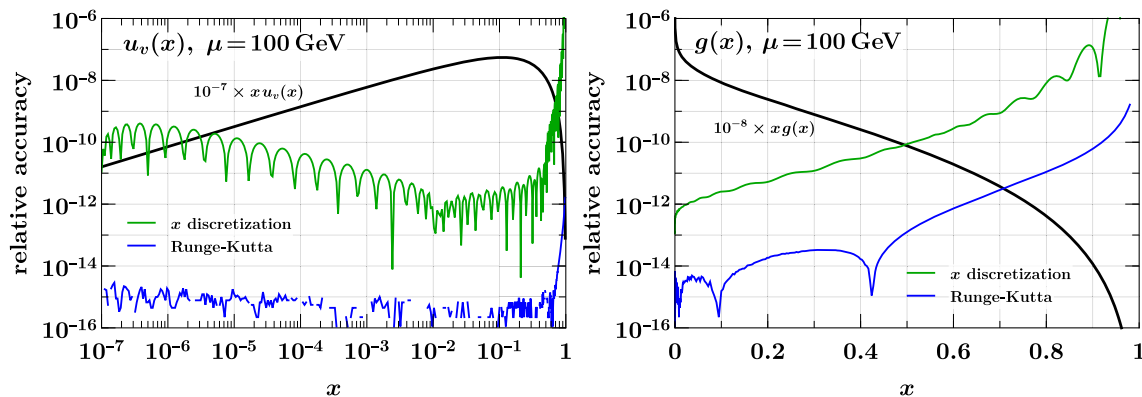


Fig. 13 Relative accuracy of NNLO DGLAP evolution and flavor matching, distinguishing the contributions due to x -space discretization (green) and the Runge–Kutta algorithm (blue). The plot on the left shows $xu_v(x)$ and the plot on the right $xg(x)$. Notice the different scales in x

lation on the same grid without the end points. This yields an estimate that is appropriately conservative, but not as overly conservative as the estimate we obtain with Gauss–Kronrod quadrature in the case of integration.

By combining our Chebyshev-based interpolation with a Runge–Kutta method of high order, we obtain a very accurate implementation of DGLAP evolution. Using 70 grid points in x and evolving from $\mu = 1.41$ GeV to $\mu = 10$ TeV, we find relative errors below 10^{-7} for x between 10^{-7} and 0.8. We successfully tested our implementation against the bench-

mark evolution tables in Refs. [73, 74]. Performing backward evolution with our method, we find no indications for any numerical instabilities.

Let us briefly point out differences between our implementation of Chebyshev interpolation and the use of Chebyshev polynomials in parameterizations of PDFs. Perhaps most striking is that the order of the highest polynomial used is much larger in our case. There are several reasons for this. First, our approach aims at keeping uncertainties due to interpolation small compared to physics uncertainties. By con-

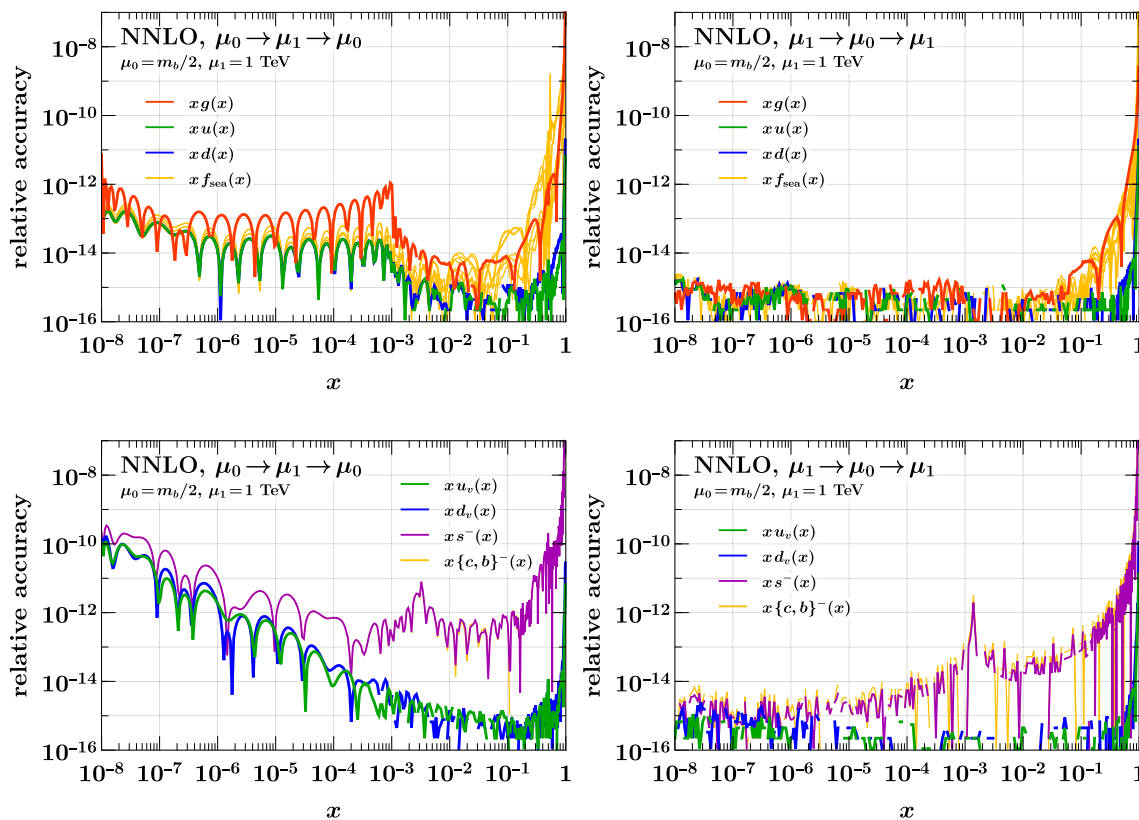


Fig. 14 Relative accuracy of NNLO DGLAP evolution at fixed $n_F = 5$ from $\mu_0 \rightarrow \mu_1 \rightarrow \mu_0$ (left) and from $\mu_1 \rightarrow \mu_0 \rightarrow \mu_1$ (right). The scales are $\mu_0 = 2.25$ GeV and $\mu_1 = 1$ TeV, and the DOPRI8 method

with maximum step size $h = 0.1$ is used to solve the evolution equations. The same grid in x is used as for the benchmark comparison in Sect. 5.2

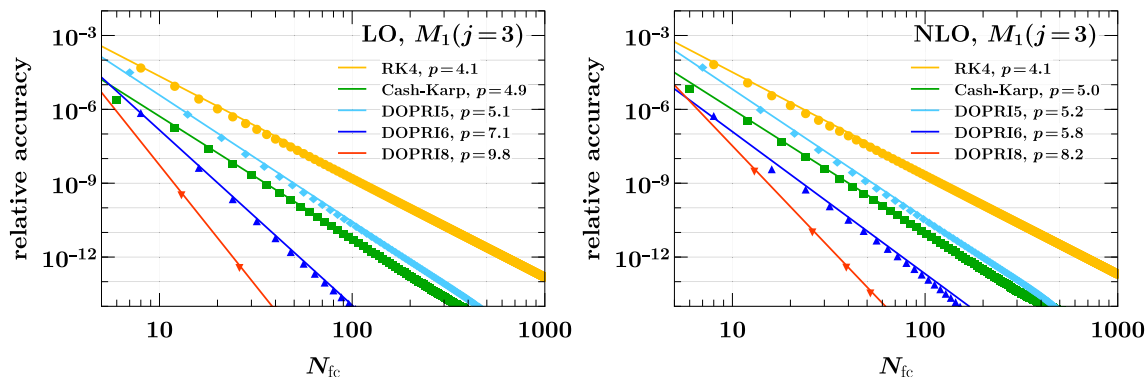


Fig. 15 Relative accuracy of different Runge–Kutta methods as a function of the total number $N_{fc} = s N_{steps}$ of calls to the function $F(t, y)$ in Eq. (58). The accuracy is evaluated for the truncated Mellin moment

$M(j)$ of the sample PDF $f_1(x)$, evolved from $t_0 = 0.7$ to $t_f = 1.7$. The lines connecting the points correspond to a fitted power law $(N_{fc})^{-p}$ with p reported in the legend

trast, as argued in [13], a PDF parameterization need not be much more accurate (relative to the unknown true form) than the uncertainty resulting from fitting the PDFs to data. Second, as we have seen, polynomials up to order 20 or 40 can be handled with ease in our method, whereas PDF determinations naturally tend to limit the number of parameters to be fitted to data. Third, the detailed functional forms used in the PDF parameterizations Refs. [11–14] at a fixed scale μ differ from each other and from the form (35) we use for

interpolation at any scale μ . It may be interesting to further investigate the impact of these differences on the number of polynomials required for a satisfactory description of PDFs.

Our approach is implemented in the C++ library CHILIPDF, which is still under development and which we plan to eventually make public. To give a sense of its current performance, we note that the evaluation of the interpolated PDF takes less than a microsecond. With the settings in Sect. 5.2, NNLO DGLAP evolution from $\mu = 1.41$ GeV to $\mu = 100$ GeV at

$n_F = 4$ takes on the order of 10 to 20 milliseconds.¹⁶ If significantly faster access to evolved PDFs is needed, it may be preferable to pre-compute the μ dependence and use interpolation in both x and μ . This can easily be done using the techniques we have described.

To conclude, let us mention additional features that are already available in CHILI PDF but not discussed here: (i) polarized evolution up to NNLO for longitudinal and up to NLO for transverse and linear polarizations, with flavor matching up to NLO for all cases, (ii) combined QCD and QED evolution of PDFs with kernels up to $\mathcal{O}(\alpha_s \alpha)$ and $\mathcal{O}(\alpha^2)$, with QED flavor matching up to $\mathcal{O}(\alpha)$, and (iii) evolution and flavor matching of double parton distributions $F(x_1, x_2, \mathbf{y}; \mu_1, \mu_2)$. For the latter, accuracy goals are typically lower than for computations involving PDFs, but the possibility to work with a low number of grid points is crucial for keeping memory requirements manageable. This will be described in a future paper.

Acknowledgements We thank Florian Fabry, Mees van Kampen, and Peter Plöchl for their contributions to CHILI PDF. It is a pleasure to thank Arnd Behring, Johannes Blümlein, Johannes Michel, Sven Moch, and Lorenzo Zoppi for valuable discussions and input. Special thanks are due to Gavin Salam and Andreas Vogt for clarifying the discrepancies between our results and the tables in Refs. [73, 74]. This work was in part funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Research Unit FOR 2926, grant number 409651613. The work of RN is supported by the ERC Starting Grant REINVENT-714788. RN also acknowledges the CINECA award under the ISCRA initiative for the availability of the performance computing resources needed for this work.

Data Availability Statement This manuscript has no associated data or the data will not be deposited. [Authors' comment: There is no associated data for this paper.]

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>. Funded by SCOAP³.

A Runge–Kutta methods

In this appendix, we take a closer look at the Runge–Kutta (RK) algorithm, which we use to solve the discretized DGLAP equation (52), as well as the renormalization group equation (54) for the running coupling. We pay special attention to different Runge–Kutta methods and their accuracy.

The RK algorithm solves the initial value problem

$$dy(t)/dt = F(t, y) \quad \text{with } y(t_0) = y_0. \quad (58)$$

It approximates the solution $y(t)$ by the set of values $y_i \approx y(t_i)$ at equispaced points t_i . The distance between two consecutive points is called the step size $h = t_{i+1} - t_i$, and the algorithm computes y_{i+1} from the approximate solution y_i at the previous point. The solution at a given t_f is obtained by dividing the interval $[t_0, t_f]$ into N steps, such that $t_f = t_N$. When using the algorithm in our work, we fix a maximum step size h_{\max} . For any given t_0 and t_f , the number of steps is then taken as the smallest integer N that satisfies $|t_f - t_0|/N \leq h_{\max}$. For brevity, h_{\max} is denoted by h in Sect. 5.

In so-called explicit RK methods, the algorithm for a step is of the form

$$y_{i+1} = y_i + h \sum_{j=1}^s b_j k_j \quad (59)$$

with

$$\begin{aligned} k_1 &= F(t_i, y_i), \\ k_2 &= F(t_i + h c_2, y_i + h a_{21} k_1), \\ k_3 &= F(t_i + h c_3, y_i + h [a_{31} k_1 + a_{32} k_2]), \\ &\vdots \\ k_s &= F(t_i + h c_s, y_i + h [a_{s1} k_1 + \dots + a_{s,s-1} k_{s-1}]). \end{aligned} \quad (60)$$

In standard nomenclature, one calls s the number of stages, a_{ij} the RK matrix, b_i the weights and c_i the nodes. Different RK methods are characterized by different values of these parameters.

A method is of order p if a single step has a numerical accuracy of $\mathcal{O}(h^{p+1})$. For a given set of RK parameters, it is easy to determine p from Eqs. (59) and (60) by a Taylor expansion in h . Since for given t_0 and t_f the number of required steps scales like $1/h$, the cumulated numerical error of N steps in the solution of the initial value problem (58) is

$$y_N = y(t_f) + \mathcal{O}(h^p). \quad (61)$$

A broad range of RK methods with different orders and numbers of stages is known [77]. Widely used is the RK4 method, also called the “classic RK method” or simply “the

¹⁶ These timings are on a recent desktop or laptop computer with an Intel® Core™ i5 or i7 processor. We expect that with a dedicated performance tuning, the code can still be made faster.

Table 4 Order and number of stages of the different Runge–Kutta methods investigated in this work. The entries in parentheses are explained in the text

Method	RK4	Cash-Karp	DOPRI5	DOPRI6	DOPRI8
Order (p)	4	5	5	6(7)	8(10?)
Stages (s)	4	6	7	8	13

RK method”, which has four stages and order $p = 4$. We also investigate several methods with $p > 4$, one by Cash and Karp [78] and three by Dormand and Prince [79,80]. According to their order, we denote the latter by DOPRI5, DOPRI6, and DOPRI8. The main parameters of these methods are given in Table 4. As noted in Ref. [80], the order of the DOPRI6 method is 7 rather than 6 if the function $F(t, y)$ in Eq. (58) has no explicit t dependence. The numerical study described below suggests that the DOPRI8 method is of order 10 if $F(t, y)$ is t independent. We do not attempt to provide a general proof for this conjecture and hence put a question mark next to the corresponding entry in our table.

To quantify the performance of the different methods for DGLAP evolution, we evolve the sample PDF $f_1(x)$ given in Eq. (37) with the DGLAP kernel for the flavor non-singlet combination $u^+ - d^+$, either at LO or at NLO. As can be seen in Eqs. (55) and (56), LO evolution corresponds to a t independent function $F(t, y)$, whereas NLO evolution implies an explicit t dependence in $F(t, y)$. We evolve from $t_0 = 0.7$ to $t_f = 1.7$. According to Eq. (53), this respectively gives $\alpha_s \approx 0.5$ and $\alpha_s \approx 0.18$, so that the t dependence of $F(t, y)$ for NLO evolution is quite strong at the lower end of the evolution interval.

To monitor the accuracy of evolution, we consider either the evolved PDF at a given x or the truncated Mellin moment (40) with a given moment index j . The results are similar in both cases. In Fig. 15, we show the accuracy for the moment with $j = 3$ as a function of $N_{fc} = s N_{\text{steps}}$, where s is the number of stages of the RK method and N_{steps} the number of RK steps between t_0 to t_f . Hence, N_{fc} is the total number of calls to the function $F(t, y)$ during evolution.¹⁷ The accuracy is determined from the difference between the solution at the selected N_{fc} and the solution at a value of N_{fc} in the region where the result does not significantly change any more.

For each RK method, we can identify a region in which the accuracy follows a power law $(N_{fc})^{-p}$. We determine the corresponding value p by a fit and find it in reasonable agreement with the order of the method. This indicates that in the corresponding region of N_{fc} the error estimate in Eq. (61) is indeed applicable.

¹⁷ Note that $F(t, y)$ is vector valued in the discretized DGLAP equation (52). Its evaluation accounts for the bulk of the computational cost of evolution.

Our investigation shows the significant advantage of RK methods with high order p for solving the DGLAP equations. For the example shown in the figure, the DOPRI8 method yields a relative accuracy below 10^{-8} with a single step (corresponding to 13 function calls), and the limits of machine precision are reached with 50 to 100 function calls.

A similar conclusion can be drawn for the computation of $\alpha_s(\mu)$ from Eq. (54). To illustrate this, we evolve $\alpha_s(M_Z) = 0.118$ down to $\mu_0 = 2.25$ GeV with $n_F = 5$. We use the DOPRI8 method and impose a maximum step size $h_{\text{max}} = 0.2$. The evolved value $\alpha_s(\mu_0)$ is then inserted into the exact analytic expression of $\mu(\alpha_s)$ at the appropriate perturbative order. Both at NLO and at NNLO, we find that the resulting value of μ agrees with μ_0 at the level of 2×10^{-15} .

References

1. A. Buckley, J. Ferrando, S. Lloyd, K. Nordström, B. Page, M. Rüfenacht et al., LHAPDF6: parton density access in the LHC precision era. *Eur. Phys. J. C* **75**, 132 (2015). <https://doi.org/10.1140/epjc/s10052-015-3318-8> arXiv:1412.7420
2. F.J. Yndurain, Reconstruction of the deep inelastic structure functions from their moments. *Phys. Lett. B* **74**, 68 (1978). [https://doi.org/10.1016/0370-2693\(78\)90062-X](https://doi.org/10.1016/0370-2693(78)90062-X)
3. G. Parisi, N. Surlas, A simple parametrization of the Q^2 dependence of the quark distributions in QCD. *Nucl. Phys. B* **151**, 421 (1979). [https://doi.org/10.1016/0550-3213\(79\)90448-6](https://doi.org/10.1016/0550-3213(79)90448-6)
4. W. Furmanski, R. Petronzio, A method of analyzing the scaling violation of inclusive spectra in hard processes. *Nucl. Phys. B* **195**, 237 (1982). [https://doi.org/10.1016/0550-3213\(82\)90398-4](https://doi.org/10.1016/0550-3213(82)90398-4)
5. R. Kobayashi, M. Konuma, S. Kumano, FORTRAN program for a numerical solution of the nonsinglet Altarelli–Parisi equation. *Comput. Phys. Commun.* **86**, 264 (1995). [https://doi.org/10.1016/0010-4655\(94\)00159-Y](https://doi.org/10.1016/0010-4655(94)00159-Y) arXiv:hep-ph/9409289
6. J. Chyla, J. Rames, On methods of analyzing scaling violation in deep inelastic scattering. *Z. Phys. C* **31**, 151 (1986). <https://doi.org/10.1007/BF01559606>
7. J. Blümlein, M. Klein, G. Ingelman, R. Rückl, Testing QCD scaling violations in the HERA energy range. *Z. Phys. C* **45**, 501 (1990). <https://doi.org/10.1007/BF01549682>
8. V.G. Krivokhizhin, S.P. Kurlovich, R. Lednicky, S. Nemecek, V.V. Sanadze, I.A. Savin et al., Next-to-leading order QCD analysis of structure functions with the help of Jacobi polynomials. *Z. Phys. C* **48**, 347 (1990). <https://doi.org/10.1007/BF01554485>
9. M. Bonvini, S. Forte, G. Ridolfi, Soft gluon resummation of Drell–Yan rapidity distributions: theory and phenomenology. *Nucl. Phys. B* **847**, 93 (2011). <https://doi.org/10.1016/j.nuclphysb.2011.01.023> arXiv:1009.5691
10. M. Bonvini, S. Marzani, Resummed Higgs cross section at $N^3\text{LL}$. *JHEP* **09**, 007 (2014). [https://doi.org/10.1007/JHEP09\(2014\)007](https://doi.org/10.1007/JHEP09(2014)007) arXiv:1405.3654
11. J. Pumplin, Parametrization dependence and $\Delta\chi^2$ in parton distribution fitting. *Phys. Rev. D* **82**, 114020 (2010). <https://doi.org/10.1103/PhysRevD.82.114020> arXiv:0909.5176
12. A. Glazov, S. Moch, V. Radescu, Parton distribution uncertainties using smoothness prior. *Phys. Lett. B* **695**, 238 (2011). <https://doi.org/10.1016/j.physletb.2010.11.025> arXiv:1009.6170
13. A.D. Martin, A.J.T.M. Mathijssen, W.J. Stirling, R.S. Thorne, B.J.A. Watt, G. Watt, Extended parameterisations for MSTW PDFs and their effect on lepton charge asymmetry from W decays.

- Eur. Phys. J. C **73**, 2318 (2013). <https://doi.org/10.1140/epjc/s10052-013-2318-9> arXiv:1211.1215
14. L.A. Harland-Lang, A.D. Martin, P. Motylinski, R.S. Thorne, Parton distributions in the LHC era: MMHT 2014 PDFs. Eur. Phys. J. C **75**, 204 (2015). <https://doi.org/10.1140/epjc/s10052-015-3397-6> arXiv:1412.3989
 15. F. Dulat, B. Mistlberger, A. Pelloni, Differential Higgs production at N³LO beyond threshold. JHEP **01**, 145 (2018). [https://doi.org/10.1007/JHEP01\(2018\)145](https://doi.org/10.1007/JHEP01(2018)145) arXiv:1710.03016
 16. M. Miyama, S. Kumano, Numerical solution of Q^2 evolution equations in a brute force method. Comput. Phys. Commun. **94**, 185 (1996). [https://doi.org/10.1016/0010-4655\(96\)00013-6](https://doi.org/10.1016/0010-4655(96)00013-6) arXiv:hep-ph/9508246
 17. P.G. Ratcliffe, A matrix approach to numerical solution of the DGLAP evolution equations. Phys. Rev. D **63**, 116004 (2001). <https://doi.org/10.1103/PhysRevD.63.116004> arXiv:hep-ph/0012376
 18. C. Pascaud, F. Zomer, A fast and precise method to solve the Altarelli–Parisi equations in x space. arXiv:hep-ph/0104013
 19. M. Dasgupta, G. Salam, Resummation of the jet broadening in DIS. Eur. Phys. J. C **24**, 213 (2002). <https://doi.org/10.1007/s100520200915> arXiv:hep-ph/0110213
 20. NNPDF Collaboration, L. Del Debbio, S. Forte, J.I. Latorre, A. Piccione, J. Rojo, Neural network determination of parton distributions: the Nonsinglet case. JHEP **03**, 039 (2007). <https://doi.org/10.1088/1126-6708/2007/03/039> arXiv:hep-ph/0701127
 21. S. Weinzierl, Fast evolution of parton distributions. Comput. Phys. Commun. **148**, 314 (2002). [https://doi.org/10.1016/S0010-4655\(02\)00584-2](https://doi.org/10.1016/S0010-4655(02)00584-2) arXiv:hep-ph/0203112
 22. A. Vogt, Efficient evolution of unpolarized and polarized parton distributions with QCD-PEGASUS. Comput. Phys. Commun. **170**, 65 (2005). <https://doi.org/10.1016/j.cpc.2005.03.103> arXiv:hep-ph/0408244
 23. A. Candido, F. Hekhorn, G. Magni, EKO: evolution kernel operators. arXiv:2202.02338
 24. A. Cafarella, C. Coriano, M. Guzzi, Precision studies of the NNLO DGLAP evolution at the LHC with CANDIA. Comput. Phys. Commun. **179**, 665 (2008). <https://doi.org/10.1016/j.cpc.2008.06.004> arXiv:0803.0462
 25. G.P. Salam, J. Rojo, A Higher Order Perturbative Parton Evolution Toolkit (HOPPET). Comput. Phys. Commun. **180**, 120 (2009). <https://doi.org/10.1016/j.cpc.2008.08.010> arXiv:0804.3755
 26. M. Botje, QCDNUM: fast QCD evolution and convolution. Comput. Phys. Commun. **182**, 490 (2011). <https://doi.org/10.1016/j.cpc.2010.10.020> arXiv:1005.1481
 27. V. Bertone, S. Carrazza, J. Rojo, APFEL: a PDF evolution library with QED corrections. Comput. Phys. Commun. **185**, 1647 (2014). <https://doi.org/10.1016/j.cpc.2014.03.007> arXiv:1310.1394
 28. V. Bertone, APFEL++: a new PDF evolution library in C++. PoS DIS2017, 201 (2018). <https://doi.org/10.22323/1.297.0201> arXiv:1708.00911
 29. M. Procura, W.J. Waalewijn, L. Zeune, Resummation of double-differential cross sections and fully-unintegrated parton distribution functions. JHEP **02**, 117 (2015). [https://doi.org/10.1007/JHEP02\(2015\)117](https://doi.org/10.1007/JHEP02(2015)117) arXiv:1410.6483
 30. G. Lustermans, J.K.L. Michel, F.J. Tackmann, W.J. Waalewijn, Joint two-dimensional resummation in q_T and 0-jettiness at NNLL. JHEP **03**, 124 (2019). [https://doi.org/10.1007/JHEP03\(2019\)124](https://doi.org/10.1007/JHEP03(2019)124) arXiv:1901.03331
 31. G. Lustermans, J.K.L. Michel, F.J. Tackmann, Generalized threshold factorization with full collinear dynamics. arXiv:1908.00985
 32. J.R. Gaunt, M. Stahlhofen, The fully-differential quark beam function at NNLO. JHEP **12**, 146 (2014). [https://doi.org/10.1007/JHEP12\(2014\)146](https://doi.org/10.1007/JHEP12(2014)146) arXiv:1409.8281
 33. J.R. Gaunt, M. Stahlhofen, The fully-differential gluon beam function at NNLO. JHEP **07**, 234 (2020). [https://doi.org/10.1007/JHEP07\(2020\)234](https://doi.org/10.1007/JHEP07(2020)234) arXiv:2004.11915
 34. A. Hornig, D. Kang, Y. Makris, T. Mehen, Transverse vetoes with rapidity cutoff in SCET. JHEP **12**, 043 (2017). [https://doi.org/10.1007/JHEP12\(2017\)043](https://doi.org/10.1007/JHEP12(2017)043) arXiv:1708.08467
 35. J.K.L. Michel, P. Pietrulewicz, F.J. Tackmann, Jet veto resummation with jet rapidity cuts. JHEP **04**, 142 (2019). [https://doi.org/10.1007/JHEP04\(2019\)142](https://doi.org/10.1007/JHEP04(2019)142) arXiv:1810.12911
 36. M. Bonvini, A.S. Papanastasiou, F.J. Tackmann, Resummation and matching of b-quark mass effects in $b\bar{b}H$ production. JHEP **11**, 196 (2015). [https://doi.org/10.1007/JHEP11\(2015\)196](https://doi.org/10.1007/JHEP11(2015)196) arXiv:1508.03288
 37. P. Pietrulewicz, D. Samitz, A. Spiering, F.J. Tackmann, Factorization and resummation for massive quark effects in exclusive Drell–Yan. JHEP **08**, 114 (2017). [https://doi.org/10.1007/JHEP08\(2017\)114](https://doi.org/10.1007/JHEP08(2017)114) arXiv:1703.09702
 38. I. Moul, L. Rothen, I.W. Stewart, F.J. Tackmann, H.X. Zhu, Subleading power corrections for N-jettiness subtractions. Phys. Rev. D **95**, 074023 (2017). <https://doi.org/10.1103/PhysRevD.95.074023> arXiv:1612.00450
 39. I. Moul, L. Rothen, I.W. Stewart, F.J. Tackmann, H.X. Zhu, N-jettiness subtractions for $gg \rightarrow H$ at subleading power. Phys. Rev. D **97**, 014013 (2018). <https://doi.org/10.1103/PhysRevD.97.014013> arXiv:1710.03227
 40. M.A. Ebert, I. Moul, I.W. Stewart, F.J. Tackmann, G. Vita, H.X. Zhu, Power corrections for N-jettiness subtractions at $\mathcal{O}(\alpha_s)$. JHEP **12**, 084 (2018). [https://doi.org/10.1007/JHEP12\(2018\)084](https://doi.org/10.1007/JHEP12(2018)084) arXiv:1807.10764
 41. R. Boughezal, X. Liu, F. Petriello, Power corrections in the N-jettiness subtraction scheme. JHEP **03**, 160 (2017). [https://doi.org/10.1007/JHEP03\(2017\)160](https://doi.org/10.1007/JHEP03(2017)160) arXiv:1612.02911
 42. R. Boughezal, A. Isgro, F. Petriello, Next-to-leading-logarithmic power corrections for N-jettiness subtraction in color-singlet production. Phys. Rev. D **97**, 076006 (2018). <https://doi.org/10.1103/PhysRevD.97.076006> arXiv:1802.00456
 43. A. Bhattacharya, I. Moul, I.W. Stewart, G. Vita, Helicity methods for high multiplicity subleading soft and collinear limits. JHEP **05**, 192 (2019). [https://doi.org/10.1007/JHEP05\(2019\)192](https://doi.org/10.1007/JHEP05(2019)192) arXiv:1812.06950
 44. M.A. Ebert, I. Moul, I.W. Stewart, F.J. Tackmann, G. Vita, H.X. Zhu, Subleading power rapidity divergences and power corrections for q_T . JHEP **04**, 123 (2019). [https://doi.org/10.1007/JHEP04\(2019\)123](https://doi.org/10.1007/JHEP04(2019)123) arXiv:1812.08189
 45. R.D. Ball, E.R. Nocera, J. Rojo, The asymptotic behaviour of parton distributions at small and large x . Eur. Phys. J. C **76**, 383 (2016). <https://doi.org/10.1140/epjc/s10052-016-4240-4> arXiv:1604.00024
 46. M. Diehl, J.R. Gaunt, Double parton scattering theory overview. Adv. Ser. Direct. High Energy Phys. **29**, 7 (2018). https://doi.org/10.1142/9789813227767_0002 arXiv:1710.04408
 47. S. Carrazza, J.M. Cruz-Martinez, M. Rossi, PDFFlow: parton distribution functions on GPU. Comput. Phys. Commun. **264**, 107995 (2021). <https://doi.org/10.1016/j.cpc.2021.107995> arXiv:2009.06635
 48. C. Runge, Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten. Z. Math. Phys. **46**, 224 (1901)
 49. L.N. Trefethen, Six myths of polynomial interpolation and quadrature. <https://people.maths.ox.ac.uk/trefethen/mythspaper.pdf> (2011)
 50. L.N. Trefethen, *Approximation Theory and Approximation Practice* (Society for Industrial and Applied Mathematics, 2012)
 51. L.N. Trefethen, Is Gauss quadrature better than Clenshaw–Curtis? SIAM Rev. **50**, 67 (2008). <https://doi.org/10.1137/060659831>

52. J. Waldvogel, Fast construction of the Fejér and Clenshaw–Curtis quadrature rules. *Bit Numer. Math.* **46**, 195 (2006). <https://doi.org/10.1007/s10543-006-0045-4>
53. Gauss–Kronrod quadrature formula, in *Encyclopedia of Mathematics*. https://www.encyclopediaofmath.org/index.php/Gauss-Kronrod_quadrature_formula
54. T.N.L. Patterson, The optimum addition of points to quadrature formulae. *Math. Comput.* **22**, 847 (1968). <https://doi.org/10.1090/S0025-5718-68-99866-9>
55. S. Alekhin, J. Blümlein, S. Moch, R. Placakyte, Parton distribution functions, α_s , and heavy-quark masses for LHC Run II. *Phys. Rev. D* **96**, 014011 (2017). <https://doi.org/10.1103/PhysRevD.96.014011> arXiv:1701.05838
56. H1, ZEUS Collaboration, H. Abramowicz et al., Combination of measurements of inclusive deep inelastic $e^\pm p$ scattering cross sections and QCD analysis of HERA data. *Eur. Phys. J. C* **75**, 580 (2015). <https://doi.org/10.1140/epjc/s10052-015-3710-4>. arXiv:1506.06042
57. P. Jimenez-Delgado, E. Reya, Delineating parton distributions and the strong coupling. *Phys. Rev. D* **89**, 074049 (2014). <https://doi.org/10.1103/PhysRevD.89.074049> arXiv:1403.1852
58. NNPDF Collaboration, R.D. Ball et al., Parton distributions from high-precision collider data. *Eur. Phys. J. C* **77**, 663 (2017). <https://doi.org/10.1140/epjc/s10052-017-5199-5>. arXiv:1706.00428
59. T.-J. Hou et al., New CTEQ global analysis of quantum chromodynamics with high-precision data from the LHC. *Phys. Rev. D* **103**, 014013 (2021). <https://doi.org/10.1103/PhysRevD.103.014013> arXiv:1912.10053
60. S. Bailey, T. Cridge, L.A. Harland-Lang, A.D. Martin, R.S. Thorne, Parton distributions from LHC, HERA, Tevatron and fixed target data: MSHT20 PDFs. *Eur. Phys. J. C* **81**, 341 (2021). <https://doi.org/10.1140/epjc/s10052-021-09057-0> arXiv:2012.04684
61. R.D. Ball et al., The path to proton structure at one-percent accuracy. arXiv:2109.02653
62. S. Dulat, T.-J. Hou, J. Gao, M. Guzzi, J. Huston, P. Nadolsky et al., New parton distribution functions from a global analysis of quantum chromodynamics. *Phys. Rev. D* **93**, 033006 (2016). <https://doi.org/10.1103/PhysRevD.93.033006> arXiv:1506.07443
63. A. Vogt, S. Moch, J.A.M. Vermaseren, The three-loop splitting functions in QCD: the singlet case. *Nucl. Phys. B* **691**, 129 (2004). <https://doi.org/10.1016/j.nuclphysb.2004.04.024> arXiv:hep-ph/0404111
64. J. Ablinger, A. Behring, J. Blümlein, A. De Freitas, A. von Manteuffel, C. Schneider, The three-loop splitting functions $P_{qg}^{(2)}$ and $P_{gg}^{(2,N_F)}$. *Nucl. Phys. B* **922**, 1 (2017). <https://doi.org/10.1016/j.nuclphysb.2017.06.004> arXiv:1705.01508
65. NNPDF Collaboration, R.D. Ball et al., Parton distributions for the LHC Run II. *JHEP* **04**, 040 (2015). [https://doi.org/10.1007/JHEP04\(2015\)040](https://doi.org/10.1007/JHEP04(2015)040). arXiv:1410.8849
66. Y.L. Dokshitzer, Calculation of the structure functions for deep inelastic scattering and e^+e^- annihilation by perturbation theory in quantum chromodynamics. *Sov. Phys. JETP* **46**, 641 (1977)
67. V.N. Gribov, L.N. Lipatov, Deep inelastic $e p$ scattering in perturbation theory. *Sov. J. Nucl. Phys.* **15**, 438 (1972)
68. G. Altarelli, G. Parisi, Asymptotic freedom in parton language. *Nucl. Phys. B* **126**, 298 (1977). [https://doi.org/10.1016/0550-3213\(77\)90384-4](https://doi.org/10.1016/0550-3213(77)90384-4)
69. S. Moch, J.A.M. Vermaseren, A. Vogt, The three loop splitting functions in QCD: the nonsinglet case. *Nucl. Phys. B* **688**, 101 (2004). <https://doi.org/10.1016/j.nuclphysb.2004.03.030> arXiv:hep-ph/0403192
70. K. Chetyrkin, B.A. Kniehl, M. Steinhauser, Strong coupling constant with flavor thresholds at four loops in the $\overline{\text{MS}}$ scheme. *Phys. Rev. Lett.* **79**, 2184 (1997). <https://doi.org/10.1103/PhysRevLett.79.2184> arXiv:hep-ph/9706430
71. M. Buza, Y. Matiounine, J. Smith, W. van Neerven, Charm electroproduction viewed in the variable flavor number scheme versus fixed order perturbation theory. *Eur. Phys. J. C* **1**, 301 (1998). <https://doi.org/10.1007/BF01245820> arXiv:hep-ph/9612398
72. A. Behring, I. Bierenbaum, J. Blümlein, A. De Freitas, S. Klein, F. Wißbrock, The logarithmic contributions to the $O(\alpha_s^3)$ asymptotic massive Wilson coefficients and operator matrix elements in deeply inelastic scattering. *Eur. Phys. J. C* **74**, 3033 (2014). <https://doi.org/10.1140/epjc/s10052-014-3033-x> arXiv:1403.6356
73. W. Giele et al., The QCD/SM working group: summary report, in *Physics at TeV colliders. Proceedings, Euro Summer School, Les Houches, France, May 21–June 1, 2001*, pp. 275–426 (2002). arXiv:hep-ph/0204316
74. M. Dittmar et al., Working Group I: Parton Distributions: Summary Report for the HERA LHC Workshop Proceedings. arXiv:hep-ph/0511119
75. G.P. Salam, A. Vogt, private communication
76. E. Ruiz Arriola, NLO evolution for large scale distances, positivity constraints and the low-energy model of the nucleon. *Nucl. Phys. A* **641**, 461 (1998). [https://doi.org/10.1016/S0375-9474\(98\)00489-8](https://doi.org/10.1016/S0375-9474(98)00489-8)
77. E. Hairer, S.P. Nørsett, G. Wanner, *Solving Ordinary Differential Equations I: Nonstiff Problems* (Springer, New York, 1993)
78. J.R. Cash, A.H. Karp, A variable order Runge–Kutta method for initial value problems with rapidly varying right-hand sides. *ACM Trans. Math. Softw.* **16**, 201 (1990). <https://doi.org/10.1145/79505.79507>
79. J. Dormand, P. Prince, A family of embedded Runge–Kutta formulae. *J. Comput. Appl. Math.* **6**, 19 (1980). [https://doi.org/10.1016/0771-050X\(80\)90013-3](https://doi.org/10.1016/0771-050X(80)90013-3)
80. P. Prince, J. Dormand, High order embedded Runge–Kutta formulae. *J. Comput. Appl. Math.* **7**, 67 (1981). [https://doi.org/10.1016/0771-050X\(81\)90010-3](https://doi.org/10.1016/0771-050X(81)90010-3)