October 22, 2024

Readme for LePDF v.1.1

Francesco Garosi,[a] **David Marzocca,**[b] **Sokratis Trifinopoulos**[c]

[a] *Max-Planck-Institute für Physik, Boltzmannstraße 8, 85748 Garching, Germany*
[b] *INFN, Sezione di Trieste, SISSA, Via Bonomea 265, 34136, Trieste, Italy*
[c] *Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA*

Contents

1 Introduction **1**

2 Using LePDFs **2**
2.1 Structure of the PDFs sets 2
2.2 Use in Mathematica 3
2.3 Use in MonteCarlo generators 4

3 DGLAP evolution code **5**
3.1 The Runge-Kutta algorithm 6
3.2 Mass thresholds 8
3.3 File list 8
3.4 Structure of "Main.cpp" 10
3.5 The choice of the parameters 11

1 Introduction

LePDFs [1] is a set of Parton Distribution Functions describing the emission of collinear radiation off an elementary lepton, once it is factorised from the hard scattering process. The output files and source code can be downloaded from

https://github.com/DavidMarzocca/LePDF/releases .

We computed, from first principles, PDFs for both electron and muon beams (as well as their antiparticles) by solving numerically the corresponding DGLAP equations from the lepton mass m_ℓ up to factorization scales $Q_{\max}$ well above the electroweak (EW) scales. This consists of two phases: below the electroweak (EW) scale only QED and QCD interactions contribute, while above we include the complete structure of Standard Model interactions, which is needed in case of multi-TeV lepton colliders, such as the muon colliders currently

being proposed. This introduces many novel effects. In particular, PDFs of fermions and gauge bosons become polarized due to the chiral nature of EW interactions.

In this document we report useful information to understand the structure of the source code and of the LePDF output files, as well as some examples of usage. We refer to our paper [1] for the discussion of the physics associated with LePDFs, here we just focus on the implementation part, recalling our numerical strategy and explaining in details how the code works and how the results are saved.

2 Using LePDFs

2.1 Structure of the PDFs sets

We discretize the x-space from a minimum value x_0 up to 1 using a grid of N_x points, $x_j = \{x_0, x_1, \ldots, x_{N_x-1} \equiv 1\}$, with spacing $\delta x_j = x_j - x_{j-1}$.[1] For a better description of the sharp peak of valence PDFs near $x = 1$ we choose a spacing that is denser near $x = 1$ and sparser at small values, in practice we set

$$x_j = 10^{-6((N_x-1-j)/(N_x-1))^{2.5}}, \qquad j = 1, \ldots, N_x \tag{1}$$

to get values from $x_0 = 10^{-6}$ to $x_{N_x-1} = 1$. After studying uncertainties due to discretization, by comparing results obtained with different N_x, we choose $N_x = 1001$ since the relative difference with finer grids would be at most of order of a few %, which is below the expected uncertainty from higher orders. This discretization allows us to obtain a set of ODEs, where the integrals are computed using the rectangles method.[2]

We discretize the factorization scale variable $t = \log(Q^2/m_\ell^2)$ with spacing dt_0 below the EW scale $t_0 = \log(m_W^2/m_\ell^2)$. The value of dt_0 is then fixed by the number N_{t_0} of grid points needed to reach the electroweak scale[3]:

$$dt_0 = \frac{t_0}{N_{t_0} - 1}.$$

We choose $N_{t_0} = 351$ for the electron and $N_{t_0} = 201$ for the muon, in order to have a similar dt_0 in both cases. In the second phase of the evolution we use N_t grid points with spacing dt, starting from t_0. In our implementation we set $N_t = 150$ for the electron and $N_t = 200$ for the muon, with $dt = dt_0$.

The results are then exported in a format inspired by the LHAPDF6 one used for proton PDFs. The structure of the output files is reported below.

- The first three lines just specify the format.

[1] Notice that here N_x is just the length of the grid, while in our paper we used it to count the number of bins. We also set $\delta x_0 = x_0$.

[2] Due to the use of the rectangles method, we note that special care should be taken when interpolating the LePDFs near the region of $x = 1$, where the muon PDF changes very steeply. In this case we recommend using zeroth order interpolation for consistency.

[3] Here $t = 0$ and $t = t_0$ are included in the counting. This means that the EW scale is reached with $N_{t_0} - 1$ steps of the algorithm.

- In the fourth and fifth line are reported respectively the grids in x and in Q. In order to publish lighter files, the grids we report are a subset of the ones used in the code for the derivation: we save 1 every 6 points for the Q-grid and 1 every 5 points for the x-grid, except for the last 100, that are all reported. We report the x-grid with more significant digits than the other outputs in order to properly reconstruct the grid spacings in the region where the points are denser.

- The next three lines report the particles' list as in Table 1: name, PDG ID (for the Z/γ and h/Z_L interference we join the PDG ID of the two states) and the additional label specifying the helicity (it is understood that for fermions or vectors it will be $\pm 1/2$ or ± 1, respectively). This last line, and the inclusion of the mixed Z/γ and h/Z_L PDFs represent the main differences with the standard LHAPDF6 format.

- In all the remaining lines we report the quantities $xf(x, Q)$: each column corresponds to a particle, following the order of the previous lines. We start at $x = x_0$ increasing Q at each row and repeating for each x, so that the data have the form reported in Table 2.

2.2 Use in Mathematica

LePDFs can be directly used in Mathematica for analytical studies. In the notebook "LePDF_examples.nb", that you can find together with the code and the outputs, we show some examples. At the beginning of the notebook one can select the valence lepton (electron or muon), the flavour scheme (`FS = 5` or `6`), and the order used to interpolate the x grid (`xGridInterpolationOrder`). LePDF files are then loaded and interpolated in x and t (linear interpolation is used for t) to get $f_A(x, Q(t))$, which in the example correspond to the function μ`PDFpart[part,Q][x]`. LePDFs for antileptons can be either loaded from the files in the same way as done for the lepton, or can be derived by applying CP to the PDF of the lepton (this is preferred, since reduces the memory usage). Some of considerations are in order:

- Due to the sharp growth of the valence lepton PDF around $x = 1$, to reconstruct it properly it is important to use many grid points in that region. This is why we used a denser grid around $x = 1$ and we reported all the last 100 points in the outputs.

- For the same reason, performing integrals involving the valence PDF with `NIntegrate` may give wrong results for valence PDFs. We then recommend to use the rectangles method with x-grid to compute such integrals, or to use zeroth order interpolation for x for consistency(achieved by setting `xGridInterpolationOrder = 0`). In Table 3 you can find an example of this issue in the calculation of momentum fraction of the various partons at $Q = 3$ TeV: while for other partons the two methods are compatible, `NIntegrate` does not work for the valence lepton.

For completeness, we report here the new version of Table 2 of our paper, which was incorrect due to a typo in the Mathematica code. The calculation is done using the rectangles method for the reasons above.

e_L	eL	11	-
e_R	eR	11	+
ν_e	nue	12	-
μ_L	muL	13	-
μ_R	muR	13	+
ν_μ	numu	14	-
τ_L	taL	15	-
τ_R	taR	15	+
ν_τ	nuta	16	-
$\bar{e}_L$	eLb	-11	+
$\bar{e}_R$	eRb	-11	-
$\bar{\nu}_e$	nueb	-12	+
$\bar{\mu}_L$	muLb	-13	+
$\bar{\mu}_R$	muRb	-13	-
$\bar{\nu}_\mu$	numub	-14	+
$\bar{\tau}_L$	taLb	-15	+
$\bar{\tau}_R$	taRb	-15	-
$\bar{\nu}_\tau$	nutab	-16	+

d_L	dL	1	-
d_R	dR	1	+
u_L	uL	2	-
u_R	uR	2	+
s_L	sL	3	-
s_R	sR	3	+
c_L	cL	4	-
c_R	cR	4	+
b_L	bL	5	-
b_R	bR	5	+
t_L	tL	6	-
t_R	tR	6	+
$\bar{d}_L$	dLb	-1	+
$\bar{d}_R$	dRb	-1	-
$\bar{u}_L$	uLb	-2	+
$\bar{u}_R$	uRb	-2	-
$\bar{s}_L$	sLb	-3	+
$\bar{s}_R$	sRb	-3	-
$\bar{c}_L$	cLb	-4	+
$\bar{c}_R$	cRb	-4	-
$\bar{b}_L$	bLb	-5	+
$\bar{b}_R$	bRb	-5	-
$\bar{t}_L$	tLb	-6	+
$\bar{t}_R$	tRb	-6	-

g_+	gp	21	+
g_-	gm	21	-
γ_+	gap	22	+
γ_-	gam	22	-
Z_+	Zp	23	+
Z_-	Zm	23	-
Z_L	ZL	23	0
Z/γ_+	Zgap	2223	+
Z/γ_-	Zgam	2223	-
W_+^+	Wpp	24	+
W_-^+	Wpm	24	-
W_L^+	WpL	24	0
W_+^-	Wmp	-24	+
W_-^-	Wmm	-24	-
W_L^-	WmL	-24	0
h	h	25	0
h/Z_L	hZL	2523	0

Table 1. Names, PDG ID and polarisations of the particles. In particular, the second, third and fourth columns of the tables correspond to the sixth, seventh and eighth lines of the output file.

2.3 Use in MonteCarlo generators

At the moment there is no implementation of LePDF into MonteCarlo generators (such as MadGraph5_aMC), due to the added difficulty of having polarised PDFs. Work on such an implementation is in progress.

$$
\begin{array}{ccc}
x_0 f_{e_L}(x_0, Q_0) & \dots & x_0 f_{h/Z_L}(x_0, Q_0) \\
x_0 f_{e_L}(x_0, Q_1) & \dots & x_0 f_{h/Z_L}(x_0, Q_1) \\
\vdots & \vdots & \vdots \\
x_0 f_{e_L}(x_0, Q_{N_Q-1}) & \dots & x_0 f_{h/Z_L}(x_0, Q_{N_Q-1}) \\
x_1 f_{e_L}(x_1, Q_0) & \dots & x_1 f_{h/Z_L}(x_1, Q_0) \\
x_1 f_{e_L}(x_1, Q_1) & \dots & x_1 f_{h/Z_L}(x_1, Q_1) \\
\vdots & \vdots & \vdots \\
x_1 f_{e_L}(x_1, Q_{N_Q-1}) & \dots & x_1 f_{h/Z_L}(x_1, Q_{N_Q-1}) \\
\vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots \\
x_{N_x-1} f_{e_L}(x_{N_x-1}, Q_0) & \dots & x_{N_x-1} f_{h/Z_L}(x_{N_x-1}, Q_0) \\
x_{N_x-1} f_{e_L}(x_{N_x-1}, Q_1) & \dots & x_{N_x-1} f_{h/Z_L}(x_{N_x-1}, Q_1) \\
\vdots & \vdots & \vdots \\
x_{N_x-1} f_{e_L}(x_{N_x-1}, Q_{N_Q-1}) & \dots & x_{N_x-1} f_{h/Z_L}(x_{N_x-1}, Q_{N_Q-1})
\end{array}
$$

Table 2. Structure of our PDF data in the LHAPDF6 format. The labels for x and Q are referred to the reduced grids reported in the fourth and fifth lines of the output files.

Parton	`NIntegrate`	Rectangles
μ_L	36.22	48.03
μ_R	34.03	45.54
ν_μ	1.72	1.75
ν_ℓ	0.0076	0.0075
ℓ	0.074	0.073
q	0.154	0.152
γ	3.01	3.00
W^-	1.287	1.281
W^+	0.098	0.096
Z	0.418	0.416
g	0.0192	0.0187
h	0.0032	0.0031
Total	77.04	100.37

Table 3. Fraction of the momentum carried by each parton at $Q = 3$ TeV. computed using NIntegrate and the rectangles method. Here the muon is the valence lepton.

3 DGLAP evolution code

Parton	$Q = 3$ TeV	$Q = 10$ TeV	$Q = 30$ TeV
μ_L	48.0	47.8	47.3
μ_R	45.5	43.1	40.6
ν_μ	1.75	3.58	5.89
$\bar{\nu}_\ell$	0.00201	0.00371	0.00579
ℓ	0.0164	0.0222	0.0282
q	0.125	0.180	0.240
γ	3.00	3.22	3.39
W_T^-	01.16	1.50	1.78
W_T^+	0.0926	0.196	0.333
Z_T	0.383	0.537	0.691
g	0.0187	0.0267	0.0359

Table 4. Fraction of the momentum carried by each parton at $Q = 3, 10, 30$ TeV.

3.1 The Runge-Kutta algorithm

The DGLAP differential equations, schematically $y'(t) = F(t, y)$, can be solved numerically with a 4^{th}-order Runge-Kutta algorithm as follows:

$$\begin{aligned} k_1(t) &= dtF(t, y(t)) \\ k_2(t) &= dtF(t + \frac{dt}{2}, y(t) + \frac{k_1(t)}{2}) \\ k_3(t) &= dtF(t + \frac{dt}{2}, y(t) + \frac{k_2(t)}{2}) \\ k_4(t) &= dtF(t + dt, y(t) + k_3(t)) \\ y(t + dt) &= y(t) + \frac{k_1(t)}{6} + \frac{k_2(t)}{3} + \frac{k_3(t)}{3} + \frac{k_4(t)}{6} + \mathcal{O}(dt^5) \ . \end{aligned} \tag{2}$$

This means that starting with the initial condition $y(t_0)$ we can obtain the solution at any t through the proper number of steps in dt. Our goal is to solve the EW DGLAP equations

$$\frac{df_B(x,t)}{dt} = P_B^v(t)\, f_B(x,t) + \sum_{A,C} \frac{\alpha_{ABC}}{2\pi} \widetilde{P}_{BA}^C(t) \otimes f_A + \frac{v^2}{16\pi^2 m_0^2 e^t} \sum_{A,C} \widetilde{U}_{BA}^C(t) \otimes f_A(t) \ , \tag{3}$$

where the indices A, B, C label the N partons we consider. After the discretization of the x-space, we obtain a set of $N \times N_x$ ordinary differential equations for the variables $f_{Bj}(t) \equiv f_B(x_j, t)$. The discretized equations have the following form:

$$\begin{aligned} &\frac{df_{Bj}(t)}{dt} = P_B^v(t) f_{Bj}(t) \\ &+ (\log(1 - x_j) - X_j) \sum_{A,C} \left[\frac{\alpha_{ABC}(t)}{2\pi} \widetilde{D}_{BA}^C(1) + \frac{v^2}{16\pi^2 m_0^2 e^t} \widetilde{V}_{BA}^C(1) \right] f_{Aj}(t) \\ &+ \sum_{k=j+1}^{N_x - 1} \frac{\delta x_k}{x_k} \sum_{A,C} \left[\frac{\alpha_{ABC}(t)}{2\pi} \widetilde{P}_{BA}^C \left(\frac{x_j}{x_k} \right) + \frac{v^2}{16\pi^2 m_0^2 e^t} \widetilde{U}_{BA}^C(t) \left(\frac{x_j}{x_k} \right) \right] f_{Ak}(t) \ , \end{aligned} \tag{4}$$

where D and V are the splitting functions without the $1-x$ at the denominator:

$$\widetilde{P}^C_{BA}(x) = \frac{\widetilde{D}^C_{BA}(x)}{(1-x)_+} \,, \qquad \widetilde{U}^C_{BA}(x) = \frac{\widetilde{V}^C_{BA}(x)}{(1-x)_+} \,, \tag{5}$$

and X_j is given by

$$X_j \equiv x_j \sum_{k=j+1}^{N_x-1} \frac{\delta x_k}{x_k^2} \frac{1}{1-\frac{x_j}{x_k}} \,. \tag{6}$$

The first two lines, which we call j-terms, come from the virtual corrections and from the definition of the plus distribution:

$$\int_x^1 dz \frac{f(z)}{(1-z)_+} = \int_x^1 dz \frac{f(z)-f(1)}{1-z} - f(1)\int_0^x \frac{dz}{1-z} = \int_x^1 dz \frac{f(z)-f(1)}{1-z} + f(1)\log(1-x) \,. \tag{7}$$

In particular, X_j comes from the discretization of the $-f(1)$ part. The last line of Eq. (4), which we call k-terms, contains instead the contribution of all the higher grid points. The number of equations is then reduced using momentum conservation:

$$\sum_A \int_0^1 dx\, x f_A(x,t) = 1 \quad \Longrightarrow \quad \sum_A \sum_{j=0}^{N_x-1} \delta x_j x_j f_{Aj}(t) = 1 \quad \forall t, \tag{8}$$

Since the initial conditions on PDFs are given by

$$f_\mu(0,x_j) = \delta(1-x_j) = \frac{1}{\delta x_{N_x-1}} \delta_{jN_x-1} \,, \qquad f_{i\neq\mu}(0,x) = 0 \,, \tag{9}$$

only $f_{\mu j}$ will be nonzero for $j = N_x - 1$ (i.e. for $x = 1$) throughout the evolution. Then we fix

$$f_{iN_x-1}(t) = \begin{cases} \frac{1}{2}\frac{L(t)}{\delta x_{N_x-1}} & i = \mu_{L,R} \\ 0 & i \neq \mu \end{cases} \,, \tag{10}$$

reducing by N the number of variables and solving the remaining equations. The factor $L(t)$ is computed using Eq. (8)

$$L(t) = 1 - \sum_A \sum_{j=0}^{N_x-2} \delta x_j x_j f_{Aj}(t) \,. \tag{11}$$

The Runge-Kutta coefficients are then computed for all the $N \times (N_x - 1)$ variables $f_{B,j}$ using Eq. (2), with the right hand side of Eq. (4) as the function F. This is done with three different functions (see Section 3.3 for details): **Rj** and **Rjuc** calculate the j terms for massless and ultra-collinear contributions respectively and **RK** computes the k terms. The coefficients are collected in four $(N_t - 1) \times N_x N$ matrices, each row corresponding to a given step of the algorithm. For each row, the first N_x entries correspond to the first parton and so on until the last one.

3.2 Mass thresholds

Along the evolution of the factorization scale Q from the muon mass up to TeV energies, several energy thresholds are encountered. They represent the crossing of particle masses or dynamical scales and in each case we match the PDFs before and after the threshold.

We start at the muon mass scale by considering only QED interactions and massless fermions e, μ, u, d, s (before version 1.2 we also considered τ and c as massless). The first threshold encountered is the QCD scale $Q_{\rm QCD}$. We implement this threshold by activating QCD interactions and including gluons for $Q > Q_{\rm QCD} = 0.7$GeV, the effect of varying this non-perturbative scale has been studied in [1]. The charm, tau, and bottom fermions are then added at the scales corresponding to their mass, the effects of including these thresholds are discussed in F. Garosi's PhD thesis [2]. Given the relations due to C, P symmetries and the fact that we neglect masses for the lightest SM fermions (e, μ, u, d, s), the total count of independent PDFs below the EW scale is 8.

At the EW scale we introduce each heavy state (W, Z, Higgs, and top) at the corresponding mass threshold and include the EW interactions and top Yukawa coupling (in the 5-flavour scheme, instead, we do not introduce the top and top Yukawa). Since EW interactions are chiral we also describe independently the PDF for each helicity state. The total count of independent PDFs rises to 54 [2]. In fact, at this stage only a few PDFs are related, for the PDFs of a muon we have:

$$f_{\bar{e}_L} = f_{\bar{\mu}_L} \ , \quad f_{\bar{e}_R} = f_{\bar{\mu}_R} \ , \quad f_{\bar{\nu}_e} = f_{\bar{\nu}_\mu} \ , \quad f_{d_R} = f_{s_R} \ , \quad f_{\bar{d}_R} = f_{\bar{s}_R} \ . \tag{12}$$

3.3 File list

The code for the computation of LePDFs consists of the following files:

- **Main.cpp**: it is the file to be run, containing the calculations.
- **Matrix_alloc.cpp**: to allocate and free $n \times m$ matrices. When allocated, all the entries of the matrix are set to zero.
- **Couplings_RG.cpp**: it contains the one-loop running coupling of α_γ (for the QED+QCD phase), α_1 and α_2 (for the SM phase). α_s instead is obtained through a linear interpolation starting from a matrix with the scale t in the first column and the corresponding value of the coupling in the second one. The file ends with functions calculating the running of other couplings starting from α_1, α_2 and α_s.
- **Cuts.cpp**: it contains the function to impose the cut $z_{\rm MAX}$ [4] and the definition of the $\theta-$function, which is used to insert the mass thresholds.
- **EVA.cpp**: here we write the formulas for the PDFs of W^- and Z in Effective Vector Approximation, which are used at $t = t_0$ to match between the two phases.

[4] In the code we use the DGLAP equations with the splitting functions computed in x/z and the PDFs in z, while is the opposite in the paper. In this way the upper bound $z_{\rm MAX}$ becomes a lower bound, which is the one implemented in the function "cut".

- **Prop_corr.cpp**: contains the mass corrections to the splitting functions coming from the propagators, including the one for the mixed PDFs.
- **Massless_splitting.cpp** and **Uc_splitting.cpp**: they collect all the splitting functions.
- **QED_QCD.cpp**: it contains the functions to implement the Runge-Kutta algorithm for the first phase of the evolution. Here you can find the formulas for the coefficients of the algorithm, the implementation of the step and the imposition of momentum conservation.
- **Massless_DGLAP.cpp**: it contains the recurring massless splittings contributing to the DGLAP equations.
- **Massless_eqs.cpp**: it contains the expressions for the k-terms massless contributions to the evolution coefficients for the SM degrees of freedom.
- **Uc_eqs.cpp**: same as the one above, but for ultra-collinear contributions.
- **RK_sum.cpp**: here is reported the function `RK`, which is used to update the matrices of the evolution coefficients adding both the massless and ultra-collinear k contributions (see Section 3.4 for its usage).
- **Massless_virt.cpp** and **Uc_virtual.cpp**: they contain the contributions to the virtual coefficients P_B^v from massless and ultra-collinear terms respectively.
- **RK_virtual.cpp** and **RK_virtual_uc.cpp**: as in "RK-sum.cpp", here are reported the functions to update the matrices with the evolution coefficients adding the j-terms, massless and ultra-collinear respectively. The former contains also the imposition of momentum conservation.
- **Writing.cpp**: it contains the functions used to write the results in LHAPDF format, both for the particle and the antiparticle.

3.4 Structure of "Main.cpp"

"Main.cpp" is the main file to be run. In the following you can find in details its structure.

- In order to be used by all the files in the code, we declare as global variables all the constants we need: charges, group factors, masses, energy scales, beta function coefficients, initial conditions for running couplings and discretization constants such as the number of points in t and x. The constants depending on the choice of the valence lepton and of the flavour scheme are set through functions.

- We allocate the matrices containing the PDFs, for both phases, with the rows corresponding to the energy scale t and the columns to the point of the x-grid. In particular there are 7 $N_t^0 \times N_x$ matrices for the first phase and 42 $N_t \times N_x$ for the second one: in the former case the first row contains the initial conditions at $t = 0$ and in the last one are reported the PDFs at $t = t_0$. We then use the last row to generate the initial conditions for the second phase, reported in the first row of the other 42 matrices. We then allocate the matrices for the evolution coefficients.

- The output files are opened with name "LePDF_ℓ_nFS_0000.dat" (ℓ = e, mu is the valence lepton and nFS = 5, 6 is the flavour scheme). The code compiles the name automatically depending on the choice of the valence lepton and of the flavour scheme.

- α_s is imported from the file "alphas_NNLO.dat" and interpolated.

- The grid in x is generated through Eq. (1)[5] and the grid spacings are computed.

- We perform the first phase of the evolution, for QED+QCD, with three for-cycles: the first one spans the scale t through $i = 0, \ldots, N_{t_0} - 1$ and for each i we compute the $(j + nN_x)^{\text{th}}$ entries of the matrices ($j = 0, \ldots, N_x - 2$ and $n = 0, \ldots, 6$) with the functions `RKj0`, and `RK0` (no ultra-collinear contributions in this phase). The latter, giving the k terms, is summed over k with the last cycle (for $k > j$). The PDFs at the scale $i + 1$ are then computed summing the coefficients according to Eq. (2). We do not consider $j = N_x - 1$, since at $x = 1$ we use momentum conservation, which is imposed through the function $Lt0$.

- After the first phase we match the 8 PDFs of QED+QCD to the full set of 54 variables of the second one, including the EVA, and we repeat the three for-cycles to perform the Runge-Kutta.

- The PDFs are saved in LHAPDF6 format. The code generates two files, one with the PDFs for the lepton (electron or muon) and one with the PDFs for the antiparticle, obtained through CP conjugation. Notice that the output is different for the two flavour schemes, since in the `5FS` the top is absent and we do not report it.

[5] We use a formula to generate the grid, but it can also be imported from a file.

3.5 The choice of the parameters

Most of the parameters used in the code are fixed and must be left unchanged. This is a list of what can be modified.

- Choice of the valence lepton, through the integer variable `lepton`: 0 corresponds to the electron, any other value to the muon. The code is exactly the same, with a different value of the valence mass m_0 (set automatically by the function "leptonmass"), since the equations are the same. We use the same notation for both cases, labelling the valence lepton as the muon: then, when we insert the results in the output files, we do it accordingly to what the valence lepton actually is.[6]
- Grid in x: any grid can be used by changing the formula which generates it (function `xgrid`) or loading it from a file. The number of points `Nx` should be modified accordingly.
- Evolution steps: the number of steps can be changed modifying the variables `NtO` and `Nt`. This is done through the functions `NtOchoice` and `Ntchoice`.
- Choice of the flavour scheme, through the integer variable `FS`: 5 means 5FS (without the top quark), while any other integer corresponds to the 6FS. Again the code is the same and the 5FS is obtained setting the top threshold `tt` to a value above the maximum t reached during the evolution: this is done by the function `ttchoice`.
- α_s: the strong coupling is loaded from file, which can be changed if desired. In this case, modify the variables `Na` and `dta`, corresponding to the number of points in the file and their spacing in t. One could also use an analytic formula, but in this case the code should be modified accordingly.
- QCD interactions become active from the scale `tQCD`. In our implementation it corresponds to 0.7 GeV, but it can be changed.
- Finally, one can change what is to be written in the output files: one can print the results only for a subset of values of (x,t). By default, we save the PDFs every 6 values of t and every 5 values of x, with the exception of the last 100, which are all saved. You can change these parameters through the variables `itSteps`, `ixStep` and `NixLast` respectively. It is also possible to modify the value of t at which one starts writing to file, by changing the value in GeV inside the variable `twrite`. By default we start from 10 GeV, safely above the non-perturbative regime of QCD.

References

[1] F. Garosi, D. Marzocca and S. Trifinopoulos, *LePDF: Standard Model PDFs for high-energy lepton colliders*, *JHEP* **09** (2023) 107, [2303.16964].

[2] F. Garosi, *Present and future tools for testing the Standard Model and beyond*, Ph.D. thesis, SISSA, Trieste, 2024 [https://hdl.handle.net/20.500.11767/140970].

[6] For instance, `fml` is written in the column corresponding to e_L if lepton = 0 and in the one corresponding to μ_L otherwise.