Contents

1 QCD Evolution Implementation

The QCD DGLAP equation looks like this:

$$\mu^2 \frac{\partial q_i(x,\mu^2)}{\partial \mu^2} = \int_x^1 \frac{dy}{y} P_{ij}\left(\frac{x}{y}, \alpha_s(\mu^2)\right) q_j(y,\mu^2) \tag{1}$$

Now let us make the following definitions $t = \ln(\mu^2)$, $\tilde{q}(x,t) = xq(x,\mu^2)$, $\tilde{P}_{ij}(x,t) = xP_{ij}(x,\alpha_s(\mu^2))$ so that eq (1) becomes:

$$\frac{\partial \tilde{q}_i(x,t)}{\partial t} = \int_x^1 \frac{dy}{y} \tilde{P}_{ij}\left(\frac{x}{y}, t\right) \tilde{q}_j(y,t) \tag{2}$$

2 Interpolation

In order to numerically solve the above equation, we need to write PDFs as interpolated functions over an x grid. In particular we want to have something like this:

$$\tilde{q}(y,t) = \sum_{\alpha=0}^{N_x} w_\alpha^{(k)}(y)\tilde{q}(x_\alpha,t)\,, \tag{3}$$

where $w_\alpha^{(k)}$ are the interpolation functions of degree k we are looking for.

Using the Lagrange formula, we find that:

$$w_\alpha^{(k)}(x) = \sum_{j=0, j\le\alpha}^{k} \theta(x - x_{\alpha-j})\theta(x_{\alpha-j+1} - x) \prod_{\delta=0,\delta\neq j}^{k} \left[\frac{x - x_{\alpha-j+\delta}}{x_\alpha - x_{\alpha-j+\delta}}\right]. \tag{4}$$

This automatically means that:

$$w_\alpha^{(k)}(x) \neq 0 \quad \text{for} \quad x_{\alpha-k} < x < x_{\alpha+1}. \tag{5}$$

This way we have that eq. (2) becomes:

$$\frac{\partial \tilde{q}_i(x,t)}{\partial t} = \sum_\alpha \int_x^1 \frac{dy}{y} \tilde{P}_{ij}\left(\frac{x}{y}, t\right) w_\alpha^{(k)}(y) \tilde{q}_j(x_\alpha, t) \tag{6}$$

more in particular, if x is one of the grid points x_β, simplifying a bit the notation we have that:

$$\frac{\partial \tilde{q}_i(x_\beta,t)}{\partial t} = \sum_\alpha \underbrace{\left[\int_{x_\beta}^1 \frac{dy}{y} \tilde{P}_{ij}\left(\frac{x_\beta}{y}, t\right) w_\alpha^{(k)}(y)\right]}_{\Pi_{ij,\beta\alpha}(t)} \tilde{q}_j(x_\alpha, t)\,. \tag{7}$$

Given eq. (5), it follows the condition:

$$\Pi_{ij,\beta\alpha}(t) \neq 0 \quad \text{for} \quad \beta \leq \alpha. \tag{8}$$

In addition, the integral in eq. (7) which gives $\Pi_{ij,\beta\alpha}$ can be optimized again using eq. (5) and it can be written as:

$$\Pi_{ij,\beta\alpha}(t) = \int_a^b \frac{dy}{y} \tilde{P}_{ij}\left(\frac{x_\beta}{y}, t\right) w_\alpha^{(k)}(y) \tag{9}$$

where:

$$a = \max(x_\beta, x_{\alpha-k}) \quad \text{and} \quad b = \min(1, x_{\alpha+1})\,. \tag{10}$$

However, performing a change of variables, $\Pi_{ij,\beta\alpha}$ can also be written as:

$$\Pi_{ij,\beta\alpha}(t) = \int_c^d \frac{dy}{y} \tilde{P}_{ij}(y,t) w_\alpha\left(\frac{x_\beta}{y}\right) \tag{11}$$

where this time:

$$c = \max(x_\beta, x_\beta/x_{\alpha+1}) \quad \text{and} \quad d = \min(1, x_\beta/x_{\alpha-k})\,. \tag{12}$$

Verifying that eqs. (9) and (11) give the same numerical result provides a cross-check of the correctness of the procedure.

Now, if rather than eq. (4), one uses a logarithmic interpolation of the form:

$$w_\alpha^{(k)}(x) = \sum_{j=0, j\leq\alpha}^{k} \theta(x - x_{\alpha-j})\theta(x_{\alpha-j+1} - x) \prod_{\delta=0,\delta\neq j}^{k} \left[\frac{\ln(x) - \ln(x_{\alpha-j+\delta})}{\ln(x_\alpha) - \ln(x_{\alpha-j+\delta})}\right] \tag{13}$$

over a logarithmically distributed grid, i.e. such that $\ln(x_\beta) - \ln(x_\alpha) = (\beta - \alpha)\delta x$, where the step δx is a constant, one has that:

$$w_\alpha^{(k)}(x) = \sum_{j=0, j\leq\alpha}^{k} \theta(x - x_{\alpha-j})\theta(x_{\alpha-j+1} - x) \prod_{\delta=0,\delta\neq j}^{k} \left[\frac{1}{\delta x} \ln\left(\frac{x}{x_\alpha}\right) \frac{1}{j-\delta} + 1\right] \tag{14}$$

that in general means that $w_\alpha^{(k)}(x) \equiv w_\alpha^{(k)}[\ln(x) - \ln(x_\alpha)]$. Therefore in eq. (11) we have that:

$$\begin{aligned} w_\alpha^{(k)}\left(\frac{x_\beta}{y}\right) &\equiv w_\alpha^{(k)}[\ln(x_\beta) - \ln(x_\alpha) - \ln(y)] \\ &= w_\alpha^{(k)}[(\beta - \alpha)\delta x - \ln(y)] \end{aligned} \tag{15}$$

which means that $w_\alpha^{(k)}(x_\beta/y)$ only depends on the difference $(\beta-\alpha)$ with the consequence that also $\Pi_{ij,\beta\alpha}$ only depends on $(\beta-\alpha)$. Now, one can use this information with eq. (8) to represent $\Pi_{ij,\beta\alpha}(t)$ as a matrix, where β is the row index and α the column index. In this way $\Pi_{ij,\beta\alpha}(t)$ would look like this:

$$\Pi_{ij}(t) = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{N_x} \\ 0 & a_0 & a_1 & \cdots & a_{N_x-1} \\ 0 & 0 & a_0 & \cdots & a_{N_x-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_0 \end{pmatrix} \tag{16}$$

therefore, if one knows the first raw of the matrix above, i.e. $\Pi_{ij,0\alpha}(t)$, it is possible to reconstruct the whole matrix. Of course, this feature must be numerically verified but it makes the computation of the evolution operators much faster because it reduces the number of integrals to be computed by a factor N_x.

3 Splitting Functions Treatment

The general form DGLAP splitting functions is the following:

$$\tilde{P}_{ij}(x,t) = xP_{ij}^R(x,t) + \frac{xP_{ij}^S(t)}{(1-x)_+} + P_{ij}^L(t)x\delta(1-x) \tag{17}$$

where $P_{ij}^R(x,t)$ is the regular term that can be integrated without any problem over any range, $P_{ij}^S(x,t)$ is instead the function that multiplies the singular term which is regularized by means of the plus prescription whose definition, referring to eq. (11), is:

$$\int_c^d dy \frac{f(y)}{(1-y)_+} = \int_c^d dy \frac{f(y)-f(1)\theta(d-1)}{1-y} - f(1)\theta(d-1)\int_0^c \frac{dy}{1-y} =$$

$$\int_c^d dy \frac{f(y)-f(1)\theta(d-1)}{1-y} + f(1)\ln(1-c)\theta(d-1)\,. \tag{18}$$

Finally $P_{ij}^L(t)$ is the coefficient of the local term, i.e. the term proportional to $\delta(1-x)$. Each of these terms has a perturbative expansion that at $\mathrm{N^kLO}$ looks like this:

$$P_{ij}^J(x,t) = \sum_{n=0}^{k} a_s^{n+1}(t) P_{ij}^{J,(n)}(x) \qquad \text{with} \qquad J = R,S,L \tag{19}$$

Therefore one has that:

$$\Pi_{ij,\beta\alpha}(t) =$$

$$\sum_{n=0}^{k} a_s^{n+1}(t) \Bigg\{ \int_c^d dy \left[P_{ij}^{R,(n)}(y) w_\alpha\left(\frac{x_\beta}{y}\right) + \frac{P_{ij}^{S,(n)}}{1-y}\left(w_\alpha\left(\frac{x_\beta}{y}\right) - w_\alpha^{(k)}(x_\beta)\theta(d-1)\right)\right]$$

$$+ \left[P_{ij}^{S,(n)}\ln(1-c)\theta(d-1) + P_{ij}^{L,(n)}\right] w_\alpha^{(k)}(x_\beta)\Bigg\}\,. \tag{20}$$

Moreover it is easy to see that $w_\alpha^{(k)}(x_\beta) = \delta_{\beta\alpha}$, so that:

$$\Pi_{ij,\beta\alpha}(t) =$$

$$\sum_{n=0}^{k} a_s^{n+1}(t)\Bigg\{ \int_c^d dy \left[P_{ij}^{R,(n)}(y) w_\alpha\left(\frac{x_\beta}{y}\right) + \frac{P_{ij}^{S,(n)}}{1-y}\left(w_\alpha\left(\frac{x_\beta}{y}\right) - \delta_{\beta\alpha}\theta(d-1)\right)\right] + \left[P_{ij}^{S,(n)}\ln(1-c)\theta(d-1) + P_{ij}^{L,(n)}\right]\delta_{\beta\alpha}\Bigg\}. \tag{21}$$

Calling:

$$\Pi_{ij,\beta\alpha}^{(n)} =$$

$$\int_c^d dy \left[P_{ij}^{R,(n)}(y) w_\alpha\left(\frac{x_\beta}{y}\right) + \frac{P_{ij}^{S,(n)}}{1-y}\left(w_\alpha\left(\frac{x_\beta}{y}\right) - \delta_{\beta\alpha}\theta(d-1)\right)\right] + \left[P_{ij}^{S,(n)}\ln(1-c)\theta(d-1) + P_{ij}^{L,(n)}\right]\delta_{\beta\alpha}\,, \tag{22}$$

we have that:

$$\Pi_{ij,\beta\alpha}(t) = \sum_{n=0}^{k} a_s^{n+1}(t)\Pi_{ij,\beta\alpha}^{(n)}\,, \tag{23}$$

and the integrals $\Pi_{ij,\beta\alpha}^{(n)}$ do not depend on the energy therefore, once the grid (and the number of active flavours) has been fixed, they can be evaluate once and for all at the beginning and used for the evolution to any scale.

It is not very easy to see that eq. (22) respects the symmetry described in eq. (16). To show this, we distinguish two case: 1) $d < 1$ and 2) $d = 1$. In the case 1), due to the presence of the Heaviside's functions $\theta(d-1)$, eq. (22) reduces to:

$$\Pi_{ij,\beta\alpha}^{(n)} = \int_c^d dy \left[P_{ij}^{R,(n)}(y) + \frac{P_{ij}^{S,(n)}}{1-y}\right] w_\alpha\left(\frac{x_\beta}{y}\right) + P_{ij}^{L,(n)}\delta_{\beta\alpha}\,, \tag{24}$$

which evidently obeys eq. (16). In the case 2), instead, we have:

$$\Pi_{ij,\beta\alpha}^{(n)} = \int_c^1 dy \left[P_{ij}^{R,(n)}(y) w_\alpha\left(\frac{x_\beta}{y}\right) + \frac{P_{ij}^{S,(n)}}{1-y}\left(w_\alpha\left(\frac{x_\beta}{y}\right) - \delta_{\beta\alpha}\right)\right] + \left[P_{ij}^{S,(n)}\ln(1-c) + P_{ij}^{L,(n)}\right]\delta_{\beta\alpha}\,, \tag{25}$$

and apparently, if $\alpha = \beta$, the term $\ln(1-c)$ seems to break the symmetry. However, this is not the case. In fact, from eq. (12), we know that in this particular case:

$$c = \max(x_\beta, x_\beta/x_{\beta+1}) = \frac{x_\beta}{x_{\beta+1}} \tag{26}$$

because $x_{\beta+1} < 1$. In addition, on a logarithmically distributed grid, $x_{\beta+1} = x_\beta \exp(\delta x)$, where δx is the constant step. Therefore, it turns out that:

$$\ln(1-c) = \ln\left(1 - \frac{x_\beta}{x_{\beta+1}}\right) = \ln[1 - \exp(-\delta x)]\,, \tag{27}$$

that is a constant which does not depend on the indices α and β and therefore does not break the symmetry given in eq. (16).

4 Solution of the DGLAP Equation

As a consequence of the DGLAP equation form, we can assume that $\tilde{q}_i(x_\beta, t) \equiv q_{i,\beta}(t)$ evolves between the energies t and t_0 according to the following (discretized) evolution equation:

$$\tilde{q}_{i,\beta}(t) = \sum_{\gamma,k} M_{ik,\beta\gamma}(t, t_0)\tilde{q}_{k,\gamma}(t_0) \tag{28}$$

with the boundary condition $M_{ik,\beta\gamma}(t_0, t_0) = \delta_{ik}\delta_{\beta\gamma}$. It follows that eq. (7) takes the form:

$$\begin{cases} \dfrac{\partial M_{ij,\alpha\beta}(t, t_0)}{\partial t} = \displaystyle\sum_{\gamma,k} \Pi_{ik,\alpha\gamma}(t) M_{kj,\gamma\beta}(t, t_0) \\ \\ M_{ij,\alpha\beta}(t_0, t_0) = \delta_{ij}\delta_{\alpha\beta} \end{cases} \tag{29}$$

which is a first order linear differential equation in the quantity $M_{ij,\alpha\beta}(t, t_0)$ that, as we actually do, can be numerically solved using the fourth order Adaptive Step-size Control Runge-Kutta algorithm. Using the arguments we discussed above, we do not need to compute all the entries of $\Pi_{ik,\gamma\alpha}(t)$. In addition, as we have already shown, the perturbative contributions to $\Pi_{ik,\gamma\alpha}(t)$ can be precomputed before solving the differential equation in eq. (29).

4.1 The Non Singlet

The non-singlet case is the easiest one because the differential equations in eq. (29) decouple in the flavour pair (i, j), and we can write them as:

$$\begin{cases} \dfrac{\partial \mathcal{M}^{(i)}_{\alpha\beta}(t, t_0)}{\partial t} = \displaystyle\sum_{\gamma=0}^{N_x} \mathcal{P}^{(i)}_{\alpha\gamma}(t) \mathcal{M}^{(i)}_{\gamma\beta}(t, t_0) \\ \\ \mathcal{M}^{(i)}_{\alpha\beta}(t_0, t_0) = \delta_{\alpha\beta} \end{cases} \qquad \text{with } i = +, -, V\,. \tag{30}$$

As a further simplification, we can use the fact that at LO $+$, $-$ and V all the evolution operators are equal, therefore solving only one of them is enough. at NLO instead only $-$ and V are equal while at NNLO they are all different.

Now, given the symmetries carried by $\Pi_{ij,\alpha\beta}$, we can write:

$$\mathcal{P}^{(i)}_{\alpha\gamma} = \mathcal{P}^{(i)}_{0(\gamma-\alpha)}\theta(\gamma - \alpha)\,, \tag{31}$$

so that eq. (30) becomes:

$$\begin{cases} \dfrac{\partial \mathcal{M}^{(i)}_{\alpha\beta}(t, t_0)}{\partial t} = \displaystyle\sum_{\delta=0}^{N_x-\alpha} \mathcal{P}^{(i)}_{0\delta}(t) \mathcal{M}^{(i)}_{(\alpha+\delta)\beta}(t, t_0) \\ \\ \mathcal{M}^{(i)}_{\alpha\beta}(t_0, t_0) = \delta_{\alpha\beta} \end{cases} \qquad \text{with } i = +, -, V\,. \tag{32}$$

4.2 The Singlet

The singlet sector is totally analogous to the non-singlet one, the only difference is that there is one additional summation over the flavours. In practice we have that:

$$\begin{cases} \dfrac{\partial \mathcal{M}^{\rm SG}_{ij,\alpha\beta}(t,t_0)}{\partial t} = \displaystyle\sum_k \sum_{\delta=0}^{N_x-\alpha} \mathcal{P}^{\rm SG}_{ik,0\delta}(t)\mathcal{M}^{\rm SG}_{kj,(\alpha+\delta)\beta}(t,t_0) & \text{with } i,j,k = q,g\,. \\ \mathcal{M}^{\rm SG}_{ij,\alpha\beta}(t_0,t_0) = \delta_{ij}\delta_{\alpha\beta} \end{cases} \tag{33}$$

5 Alternative Solutions of the DGLAP Equation

In this section we show how it is possible to solve the DGLAP equation in an alternative way with respect to that shown in the previous sections exploiting the RGE of the running coupling α_s. This will lead to a different equation that admits two preturbatively equivalent solutions: the first, that we will refer to as "exact" solution and that reproduces the solution seen above, and the second, the so-called "expanded" solution, that reproduces the solution usually adopted in the N-space code (like NNPDF).

The starting point is the RGE:

$$\mu^2 \frac{\partial a_s}{\partial \mu^2} = \frac{\partial a_s}{\partial t} = \beta(a_s)\,, \tag{34}$$

where:

$$a_s \equiv \frac{\alpha_s}{4\pi} \tag{35}$$

and:

$$\beta(a_s) = -a_s^2 \sum_{n=0}^{N} a_s^n \beta_n\,. \tag{36}$$

where N represents the desired preturbative order. Using eq. (34) and eq. (23), we can rewrite eq. (29) as:

$$\begin{cases} \dfrac{\partial M_{ij,\alpha\beta}(t,t_0)}{\partial a_s} = -\dfrac{1}{a_s} \displaystyle\sum_{\gamma,k} \left[\frac{\displaystyle\sum_{n=0}^{N} a_s^n \Pi^{(n)}_{ik,\alpha\gamma}}{\displaystyle\sum_{n=0}^{N} a_s^n \beta_n} \right] M_{kj,\gamma\beta}(t,t_0) \\ M_{ij,\alpha\beta}(t_0,t_0) = \delta_{ij}\delta_{\alpha\beta} \end{cases} \tag{37}$$

Now there are two possibile way to solve eq. (37): either we solve directly it numerically as it is or we first expand the term in the square brackets as a series of a_s keeping only the terms up to order a_s^N and then we solve the equation. It is obvious that the first way to solve eq. (37) must be numerical equal to the solution of eq. (29). The second way instead is not numerically

equal but is perturbatively equivalent. This second solution is referred to as N-space solution as it is usually used in the N-space approach because it permits to solve analytically the DGLAP equation.

To expand the term in the square brackets, we notice that up to NNLO ($N = 2$) we have that:

$$\frac{1}{\sum_{n=0}^{2} a_s^n \beta_n} = \frac{1}{\beta_0}\left[1 - \frac{\beta_1}{\beta_0} a_s + \left(\frac{\beta_1^2}{\beta_0^2} - \frac{\beta_2}{\beta_0}\right) a_s^2\right] + \mathcal{O}(a_s^3)\,, \tag{38}$$

so that:

$$\begin{aligned}\frac{\sum_{n=0}^{2} a_s^n \Pi_{ik,\alpha\gamma}^{(n)}}{\sum_{n=0}^{2} a_s^n \beta_n} &= \frac{1}{\beta_0}\left\{\Pi_{ik,\alpha\gamma}^{(0)} + a_s\left[\Pi_{ik,\alpha\gamma}^{(1)} - b_1 \Pi_{ik,\alpha\gamma}^{(0)}\right]\right. \\ &+ \left. a_s^2\left[\Pi_{ik,\alpha\gamma}^{(2)} - b_1 \Pi_{ik,\alpha\gamma}^{(1)} + \left(b_1^2 - b_2\right)\Pi_{ik,\alpha\gamma}^{(0)}\right]\right\} + \mathcal{O}(a_s^3)\end{aligned} \tag{39}$$

where we have defined:

$$b_n \equiv \frac{\beta_n}{\beta_0}\,. \tag{40}$$

Finally, defining:

$$\begin{aligned}\widetilde{\Pi}_{ik,\alpha\gamma}^{(0)} &\equiv \Pi_{ik,\alpha\gamma}^{(0)}\,, \\ \widetilde{\Pi}_{ik,\alpha\gamma}^{(1)} &\equiv \Pi_{ik,\alpha\gamma}^{(1)} - b_1 \Pi_{ik,\alpha\gamma}^{(0)}\,, \\ \widetilde{\Pi}_{ik,\alpha\gamma}^{(2)} &\equiv \Pi_{ik,\alpha\gamma}^{(2)} - b_1 \Pi_{ik,\alpha\gamma}^{(1)} + \left(b_1^2 - b_2\right)\Pi_{ik,\alpha\gamma}^{(0)}\,,\end{aligned} \tag{41}$$

we can write eq. (37) up to NNLO as:

$$\begin{cases} \dfrac{\partial M_{ij,\alpha\beta}(t,t_0)}{\partial a_s} = -\dfrac{1}{a_s \beta_0} \displaystyle\sum_{\gamma,k}\left[\sum_{n=0}^{2} a_s^n \widetilde{\Pi}_{ik,\alpha\gamma}^{(n)}\right] M_{kj,\gamma\beta}(t,t_0) \\ \\ M_{ij,\alpha\beta}(t_0,t_0) = \delta_{ij}\delta_{\alpha\beta} \end{cases}\,. \tag{42}$$

Solving eq. (42) provides the so-called "expanded" solution.

6 Small-x Resummation using `Hell`

The implemenatation of the small-x resummation in `APFEL` is done by interfacing it to the code `Hell` by Marco Bonvini. `Hell` provides, amongst other things, the small-x resummed singlet splitting functions (times x) up to NLL to be matched to the unresummed LO, NLO and NNLO splitting functions. In practice, the user can specify the logarithmic accuracy (LL or NLL) and, for each logarithmic accuracy, the perturbative accuracy to which the rusummed splitting functions are to be matched.

With the inclusion of the small-x resummation, the splitting functions P_{ij} in eq. (1) should be interpreted as:

$$P_{ij}(x,\alpha_s) = \underbrace{P_{ij}^{\rm FO}(x,\alpha_s)}_{\text{Already present in APFEL}} + \underbrace{P_{ij}^{\rm Res-FO}(x,\alpha_s)}_{\text{Provided by Hell}} \tag{43}$$

where "FO" stands for Fixed Order and "Res" for Resummed. The essential difference between $P_{ij}^{\rm FO}$ and $P_{ij}^{\rm Res-FO}$ stems from the fact that the former admits the usual pertubative expansion:

$$P_{ij}^{\rm FO}(x,\alpha_s) = \sum_{n=0}^{N} a_s^{n+1} P_{ij}^{(n)}(x)\,, \tag{44}$$

while the latter, by definition, does not. This feature of $P_{ij}^{\rm Res-FO}$ forbids to precompute the perturbative coefficients in the r.h.s. of the DGLAP equation before solving it. In principle then, one should recompute the integrals of the splitting functions on the x-space interpolation grid at every step of the algorithm that numerically solves the discretized DGLAP equation. This is clearly very unefficient and would enormously inflate the computation time. In order to keep the computation time under control, we use an interpolation grid also in α_s. In practice, we precompute the integrals of the resummed splitting function over the x-space interpolation grid for several values of α_s (logarithmically?) distributed over a reasonable range, so that the values of the same integrals for any value of α_s needed during the numerical solution of the DGLAP equation would be obtained by interpolation.

Using the notation of eq. (22), we have that the integral of the resummed splitting functions on the x-space grid would take the form:

$$\Pi_{ij,\beta\alpha}^{\rm Res}(\alpha_s) = \int_c^d dy P_{ij}^{\rm Res-FO}(y,\alpha_s) w_\alpha\left(\frac{x_\beta}{y}\right)\,, \tag{45}$$

so that eq. (29) would become:

$$\begin{cases} \dfrac{\partial M_{ij,\alpha\beta}(t,t_0)}{\partial t} = \displaystyle\sum_{\gamma,k}\left[\Pi_{ik,\alpha\gamma}(t) + \Pi_{ij,\beta\alpha}^{\rm Res}(\alpha_s(t))\right] M_{kj,\gamma\beta}(t,t_0) \\ \\ M_{ij,\alpha\beta}(t_0,t_0) = \delta_{ij}\delta_{\alpha\beta} \end{cases}\,. \tag{46}$$

As we mentioned above, $\Pi_{ij,\beta\alpha}^{\rm Res}$ cannot be expandend as a truncated series of α_s and thus we cannot precompute the perturbative coefficients making the numerical solution of the DGLAP equation efficient. A possible way out is to precompute $\Pi_{ij,\beta\alpha}^{\rm Res}$ over a grid in α_s, say $\alpha_s^{(\tau)}$ with $\tau = 0,\ldots,m$, so that, after the initialization step, we have the set of integrals:

$$\Pi_{ij,\beta\alpha,\tau}^{\rm Res} = \int_c^d dy P_{ij}^{\rm Res-FO}(y,\alpha_s^{(\tau)}) w_\alpha\left(\frac{x_\beta}{y}\right) \quad \tau = 0,\ldots,m\,, \tag{47}$$

and for obtaining the value of $\Pi_{ij,\beta\alpha}^{\rm Res}$ for a generic value of α_s we use the linear interpolation. Supposing that $\alpha_s^{(\tau)} \le \alpha_s < \alpha_s^{(\tau+1)}$, we have that:

$$\Pi_{ij,\beta\alpha}^{\rm Res}(\alpha_s) = \left(\frac{\alpha_s^{(\tau+1)} - \alpha_s}{\alpha_s^{(\tau+1)} - \alpha_s^{(\tau)}}\right)\Pi_{ij,\beta\alpha,\tau}^{\rm Res} + \left(\frac{\alpha_s - \alpha_s^{(\tau)}}{\alpha_s^{(\tau+1)} - \alpha_s^{(\tau)}}\right)\Pi_{ij,\beta\alpha,\tau+1}^{\rm Res} \tag{48}$$

There is a further complication that we need to deal with that is the number of active flavours. In fact, any integral must be computed with the correct number n_f of active flavours. Assuming to be working in the VFNS (the FFNS is instead trivial), what we need then is a grid in α_s that has a node for each crossing point, that is there must be one value of the index τ such that $\alpha_s^{(\tau)} = \alpha_s(m_h)$ where m_h is the mass of any heavy flavour and such that for $\alpha_s < \alpha_s^{(\tau)}$ there are, say, n_f active flavours, while for $\alpha_s \geq \alpha_s^{(\tau)}$ there are $n_f + 1$ active flavours. This in practice means that the grid in α_s has as many fixed points as potentially active flavours. Unfortunately this is not enough because, when considering the NNLO evolution in the VFNS, the evolution of α_s, as well as that of PDFs, has a discontinuity in correspondence of the heavy quark thresholds([1]). To overcome this problem we may assign to the point of the grid corresponding to the heavy threshold m_h two value, *i.e.* the values $\alpha_s^{(\tau)} = \alpha_s(m_h - \varepsilon)$ and $\alpha_s^{(\tau+1)} = \alpha_s(m_h)$. This trick, in conjuction with the linear interpolation, does to job without any further assumption.

We will now try to derive the form of the "expanded" solution in the presence of small-x resummation. In order to do so, we need to recognise that the resummed spliiting functions in eq. (43) admit an expansion in α_s for fixed

A A Remark on the Interpolation Functions

Just for the record, it is useful to derive the expression for the interpolation functions given in eq. (4) and show how this is not the only possible choice.

Suppose we want to perform an interpolation of degree k of the test function g in the point x. As is well known, we will need a subset of $k+1$ consecutive points on the total interpolation grids $\{x_\alpha, \ldots, x_{\alpha+k}\}$. However, the relative position between the point x and the subset of points used for the interpolation is arbitrary. In principle, it is not even required that x is somewhere between x_α and $x_{\alpha+k}$. However, in this case one would talk about extrapolation rather than interpolation and this is clearly not a convenient option because it would lead to a substantial deteriotation in the accuracy with which $g(x)$ is determined. As a consequence, it is convenient to choose the subset of points in such a way that $x_\alpha < x \leq x_{\alpha+k}$. However, the ambiguity remains because there are k possible choices of the subset of points accordind to which $x_\alpha < x \leq x_{\alpha+1}$, or $x_{\alpha+1} < x \leq x_{\alpha+2}$, and so on.

In particular, to derive eq. (4) we have assumed that:

$$x_\alpha < x \leq x_{\alpha+1}\,. \tag{49}$$

Let's see how eq. (4) comes out. Using the standard Lagrange interpolation procedure, we can approximate the function g in x as:

$$g(x) = \sum_{i=0}^{k} \ell_i^{(k)}(x) g(x_{\alpha+i}) \tag{50}$$

[1] A discontinuity appears also at NLO if factorization and renormalization scales are not equal.

where $\ell_i^{(k)}$ is the i-th Lagrange polynomial of degree k which can be written as:

$$\ell_i^{(k)}(x) = \prod_{m=0, m\neq i}^{k} \frac{x - x_{\alpha+m}}{x_{\alpha+i} - x_{\alpha+m}} \,. \tag{51}$$

However, as we said, we impose that eq. (50) applies only for the assumption in eq. (49) is fulfilled. We can then generalize it by writing:

$$g(x) = \theta(x - x_\alpha)\theta(x_{\alpha+1} - x) \sum_{i=0}^{k} g(x_{\alpha+i}) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\alpha+m}}{x_{\alpha+i} - x_{\alpha+m}} \,. \tag{52}$$

Now, if we want to relax the restriction in eq. (49), we just have to sum over all nodes of the global interpolation grid, that is:

$$g(x) = \sum_{\alpha=0}^{N_x} \theta(x - x_\alpha)\theta(x_{\alpha+1} - x) \sum_{i=0}^{k} g(x_{\alpha+i}) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\alpha+m}}{x_{\alpha+i} - x_{\alpha+m}} \,. \tag{53}$$

Defining $\beta = \alpha + i$, we can rewrite the equation above as:

$$g(x) = \sum_{\beta=0}^{N_x} g(x_\beta) \sum_{i=0, i\leq\beta}^{k} \theta(x - x_{\beta-i})\theta(x_{\beta-i+1} - x) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\beta-i+m}}{x_\beta - x_{\beta-i+m}} \,, \tag{54}$$

where the additional condition $i \leq \beta$ comes from the condition $\alpha \geq 0$. Eq. (54) is clearly equivalent to eq. (4), assuming that:

$$w_\beta^{(k)}(x) = \sum_{i=0, i\leq\beta}^{k} \theta(x - x_{\beta-i})\theta(x_{\beta-i+1} - x) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\beta-i+m}}{x_\beta - x_{\beta-i+m}} \,. \tag{55}$$

Now, instead of starting from the assumption in eq. (49), we start from the more general condition:

$$x_{\alpha+t} < x \leq x_{\alpha+t+1} \quad \text{with} \quad t = 0, \ldots, k-1 \,, \tag{56}$$

we have that the interpolation formula would look like this:

$$g(x) = \sum_{\alpha=0}^{N_x} \theta(x - x_{\alpha+t})\theta(x_{\alpha+t+1} - x) \sum_{i=0}^{k} g(x_{\alpha+i}) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\alpha+m}}{x_{\alpha+i} - x_{\alpha+m}} \,, \tag{57}$$

that can be rearranged as:

$$g(x) = \sum_{\beta=0}^{N_x} g(x_\beta) \sum_{i=0, i\leq\beta}^{k} \theta(x - x_{\beta-i+t})\theta(x_{\beta-i+t+1} - x) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\beta-i+m}}{x_\beta - x_{\beta-i+m}} \,. \tag{58}$$

Therefore the "generalized" interpolation functions are:

$$w_{\beta,t}^{(k)}(x) = \sum_{i=0, i\leq\beta}^{k} \theta(x - x_{\beta-i+t})\theta(x_{\beta-i+t+1} - x) \prod_{m=0, m\neq i}^{k} \frac{x - x_{\beta-i+m}}{x_\beta - x_{\beta-i+m}} \,, \tag{59}$$

and they assume that:

$$x_{\alpha+t} < x \leq x_{\alpha+t+1} \quad \text{with} \quad t = 0, \ldots, k-1 \,. \tag{60}$$