Preprint typeset in JHEP style - HYPER VERSION

Fast Computations of Jet Cross Sections in Hadron-Induced Processes

Early Draft from April 7, 2005

Thomas Kluge, DESY, E-mail: thomas.kluge@desy.de

Klaus Rabbertz, Univ. Karlsruhe, E-mail: klaus.rabbertz@cern.ch

Markus Wobisch, Fermilab, E-mail: wobisch@fnal.gov

Abstract: Standard methods to compute cross sections in hadron-induced collisions at higher orders in perturbative QCD are time-consuming. We present a method which allows very fast computations of cross sections which can be used in fits of parton density functions and/or the strong coupling constant. A full implementation of the method is made for the computation of jet cross sections at next-to-leading order in Deep-Inelastic Scattering and in Hadron Collisions for Tevatron and LHC energies. Computer code is provided to compute NLO predictions for jets in DIS at HERA, for jets in Run I and Run II at the Tevatron and at the LHC.

Contents

1. Introduction

Theoretical predictions for observables in hadron-induced processes are usually depending on the parton density functions of the hadron(s) involved.... For many observables the calculation of the cross sections requires a Monte Carlo integration over the phase space. The computation of predictions beyond the lowest order is often very time-consuming. In many practical cases, a repeated calculation of the same observable is needed, e.g. to compute the PDF, or $alpha_s$ dependence, or to use the observable in an iterative fitting procedure to determine the PDFs.

Currently k-factor method is used: Determine k-factor (the ratio of the NLO over LO result) for one PDF – during fit: compute only LO predictions, multiplied by k-factor. Problem: k-factor also depends on the PDFs (although only weakly: different x coverage, different decomposition of partonic subprocesses); plus: even the LO calculation is relatively time consuming. (cite Tung - priv comm.: Tevtron Jets and Drell-Yan data are currently the most time-consuming observables in the CTEQ PDF fits.

Here: method for computations within a second!

2. Parton Density Functions

2.1 Linear Combinations of PDFs

The parton density functions (PDFs) of a hadron are defined in a given factorization scheme. Usually the $\overline{\text{MS}}$-scheme is used or, sometimes, the DIS-scheme. In this factorization scheme they depend on two variables: the hadron momentum fraction x, carried by the parton and the factorization scale μ_f at which initial-state singularities are factorized. In general one has to consider thirteen different PDFs (the gluon density, six quark and six anti-quark densities). For processes with two initial state hadrons this corresponds to 169 different combinations. In most applications the number of independent PDFs can, however, be reduced. All jet cross sections discussed in this note can be expressed using four linear combinations of PDFs: $G, Q, \bar{Q}, C$ plus two 2-parton correlation functions S, A which are defined as

$$G(x,\mu_f) = g(x,\mu_f)\,, \tag{2.1}$$

$$Q(x,\mu_f) = \sum_i q_i(x,\mu_f)\,, \tag{2.2}$$

$$\bar{Q}(x,\mu_f) = \sum_i \bar{q}_i(x,\mu_f)\,, \tag{2.3}$$

$$C(x,\mu_f) = \sum_i e_i^2 \left(q_i(x,\mu_f) + \bar{q}_i(x,\mu_f)\right)\,, \tag{2.4}$$

$$S(x_1,x_2,\mu_f) = \sum_i \left(q_i(x_1,\mu_f)\, q_i(x_2,\mu_f) + \bar{q}_i(x_1,\mu_f)\, \bar{q}_i(x_2,\mu_f)\right)\,, \tag{2.5}$$

$$A(x_1,x_2,\mu_f) = \sum_i \left(q_i(x_1,\mu_f)\, \bar{q}_i(x_2,\mu_f) + \bar{q}_i(x_1,\mu_f)\, q_i(x_2,\mu_f)\right)\,. \tag{2.6}$$

The quark (anti-quark) density of flavor i is labeled $q_i(x)$ ($\bar{q}_i(x)$) and e_i denotes the electrical charge of the quarks of flavor i. The sums run over all quark flavors i considered in the calculation ($i = 1, ..., n_f$). $G(x)$ is the gluon density. For simplicity, the dependence of the PDFs on the factorization scale μ_f is not noted in the following.

2.2 Approximations of PDFs

The procedure discussed later in this note requires to express a given PDF $f(x)$ by a linear combination of eigenfunctions $E^{(i)}(x)$. We introduce a set of discrete x-values labeled $x^{(i)}$ ($i = 0, 1, 2, \cdots, n$) with $x^{(n)} < x^{(n-1)} < x^{(n-2)} < \cdots < x^{(0)} = 1$. Around each $x^{(i)}$ we define an eigenfunction $E^{(i)}(x)$ such that $E^{(i)}(x^{(i)}) = 1$, $E^{(i)}(x^{(j)}) = 0$ for $i \neq j$ and $\sum_i E^{(i)}(x) = 1$ for all x. The PDF $f(x)$ can now be approximated by a linear combination of the eigenfunctions $E^{(i)}(x)$ where the coefficients are given by the PDF values $f(x^{(i)})$ at the discrete points $x^{(i)}$

$$f(x) = \sum_i f(x^{(i)})\, E^{(i)}(x)\,. \tag{2.7}$$

With $E^{(i,j)}(x_1, x_2) \equiv E^{(i)}(x_1)E^{(j)}(x_2)$ the product $f(x_1, x_2) \equiv f_1(x_1)\, f_2(x_2)$ of two PDFs can be written as

$$f(x_1, x_2) = \sum_{i,j} f(x_1^{(i)}, x_2^{(j)})\; E^{(i,j)}(x_1, x_2)\,. \tag{2.8}$$

The precision of the approximation depends on the choice of the x_i and the eigenfunctions $E^{(i)}(x)$. This is discussed in the Appendix.

3. Hadron-Hadron Collisions

3.1 PDFs and Partonic Subprocesses

The scattering of two hadrons is described by seven different subprocesses (in all cases $i \neq j$)

$$\begin{array}{rcrcl}
gg \to \text{jets} & & & \propto & H_1(x_1, x_2) \\
qg \to \text{jets} & \text{plus} & \bar{q}g \to \text{jets} & \propto & H_2(x_1, x_2) \\
gq \to \text{jets} & \text{plus} & g\bar{q} \to \text{jets} & \propto & H_3(x_1, x_2) \\
q_i q_j \to \text{jets} & \text{plus} & \bar{q}_i \bar{q}_j \to \text{jets} & \propto & H_4(x_1, x_2) \\
q_i q_i \to \text{jets} & \text{plus} & \bar{q}_i \bar{q}_i \to \text{jets} & \propto & H_5(x_1, x_2) \\
q_i \bar{q}_i \to \text{jets} & \text{plus} & \bar{q}_i q_i \to \text{jets} & \propto & H_6(x_1, x_2) \\
q_i \bar{q}_j \to \text{jets} & \text{plus} & \bar{q}_i q_j \to \text{jets} & \propto & H_7(x_1, x_2)
\end{array}$$

Their contributions are directly proportional to the functions H_k ($k = 1, ..., 7$) which are defined as

$$H_1(x_1, x_2) = G(x_1)\, G(x_2)\,, \tag{3.1}$$

$$H_2(x_1, x_2) = \big(Q(x_1) + \bar{Q}(x_1)\big)\; G(x_2)\,, \tag{3.2}$$

$$H_3(x_1, x_2) = G(x_1)\; \big(Q(x_2) + \bar{Q}(x_2)\big)\;, \tag{3.3}$$

$$H_4(x_1, x_2) = Q(x_1)Q(x_2) + \bar{Q}(x_1)\bar{Q}(x_2) - S(x_1, x_2)\,, \tag{3.4}$$

$$H_5(x_1, x_2) = S(x_1, x_2)\,, \tag{3.5}$$
$$H_6(x_1, x_2) = A(x_1, x_2)\,, \tag{3.6}$$
$$H_7(x_1, x_2) = Q(x_1)Q(x_2) + \bar{Q}(x_1)\bar{Q}(x_2) - A(x_1, x_2)\,. \tag{3.7}$$

These functions exhibit the symmetries $H_n(x_1, x_2) = H_n(x_2, x_1)$ for $n = 1, 4, 5, 6, 7$ and $H_2(x_1, x_2) = H_3(x_2, x_1)$. Please note that in collisions of a hadron and an anti-hadron the PDFs of the anti-hadron are expressed by the PDFs of the hadron, where the quarks and the anti-quarks are swapped. As a consequence one needs to swap the functions $H_4 \leftrightarrow H_7$ and $H_5 \leftrightarrow H_6$ in the respective equations.

3.2 Jet Cross Sections in Hadron-Hadron Collisions

The jet cross section in hadron-hadron collisions is given as a perturbative expansion in α_s. The contribution from each order n is a convolution of the perturbative coefficients and the parton density functions of the hadrons, summed over the seven subprocesses and multiplied with α_s^n

$$\sigma_{\mathrm{hh}} = \sum_n \alpha_s^n(\mu_r) \sum_{k=1}^{7} c_{k,n}(\mu_r, \mu_f) \otimes H_k(x_1, x_2, \mu_f)\,. \tag{3.8}$$

Using eq. (2.8) we can replace the functions H_k by linear combinations of eigenfunctions $F^{(i,j)}$ and obtain

$$\sigma_{\mathrm{hh}} = \sum_n \alpha_s^n(\mu_r) \sum_{k=1}^{7} c_{k,n}(\mu_r, \mu_f) \otimes \left(\sum_{i,j} H_k(x_1^{(i)}, x_2^{(j)})\, F^{(i,j)}(x_1, x_2) \right)\,. \tag{3.9}$$

or

$$\sigma_{\mathrm{hh}} = \sum_n \alpha_s^n(\mu_r) \sum_{k=1}^{7} \sum_{i,j} H_k(x_1^{(i)}, x_2^{(j)}) \left(c_{k,n}(\mu_r, \mu_f) \otimes F^{(i,j)}(x_1, x_2) \right)\,. \tag{3.10}$$

We define

$$\tilde{\sigma}_{k,n}^{(i,j)} \equiv c_{k,n}(\mu_r, \mu_f) \otimes F^{(i,j)}(x_1, x_2)\,. \tag{3.11}$$

Please note that while this convolution is independent of the PDFs and α_s, the $\tilde{\sigma}_{k,n}^{(i,j)}$ contain all information on the observable (the perturbative coefficients, the jet definition, and the phase space restrictions). The cross section can now be written as a simple product

$$\sigma_{\mathrm{hh}} = \sum_{i,j,k,n} \alpha_s^n(\mu_r)\, H_k(x_1^{(i)}, x_2^{(j)})\, \tilde{\sigma}_{k,n}^{(i,j)}\,. \tag{3.12}$$

While the time consuming computation of the $\tilde{\sigma}_{k,n}^{(i,j)}$ needs to be done only once, the cross section can quickly be reevaluated for different PDFs and α_s values, as e.g. required in the determination of PDF uncertainties or in global fits of PDFs.

4. Lepton-Hadron Scattering

4.1 PDFs and Partonic Subprocesses

In Lepton-Hadron Scattering there are two interaction modes for the photon: the "direct" process in which the photon interacts as a point-like object with a parton from the hadron and the "resolved" process in which the photon interacts via it's hadronic structure. The latter becomes relevant at lower photon virtualities ($Q^2 \lesssim 10\,\mathrm{GeV}^2$) and, of course, in photoproduction where the photon is real. In the resolved process the photon is described by it's parton density functions and the description is analog to hadron-hadron scattering, as described before. The only difference in collisions of a resolved photon with a hadrons is that one has to deal with different PDFs on both sides. Therefore one can not apply the same symmetries as in in the collisions of two identical hadrons (or hadron and anti-hadron). The direct process is described by three partonic subprocesses

$$\begin{aligned} \gamma\, G \to \text{jets} \quad &\propto \quad D_1(x) \quad , \\ \gamma\, C \to \text{jets} \quad &\propto \quad D_2(x) \quad , \\ \gamma\, \left(Q + \bar{Q}\right) \to \text{jets} \quad &\propto \quad D_3(x) \quad . \end{aligned}$$

The contributions of the different subprocesses are directly proportional to the functions D_k $(k = 1, 2, 3)$ which are defined as

$$D_1(x) = G(x)\,, \tag{4.1}$$

$$D_2(x) = C(x)\,, \tag{4.2}$$

$$D_3(x) = Q(x) + \bar{Q}(x)\,. \tag{4.3}$$

4.2 Jet Cross Sections in Deep-Inelastic Lepton-Hadron Scattering

The jet cross section is given as a perturbative expansion in α_s. The contribution from each order n is a convolution of the perturbative coefficients and the parton density functions of the hadron, multiplied with α_s^n

$$\sigma_{\mathrm{DIS}} = \sum_n \alpha_s^n(\mu_r) \sum_{k=1}^{3} c_{k,n}(\mu_r, \mu_f) \otimes D_k(x, \mu_f)\,. \tag{4.4}$$

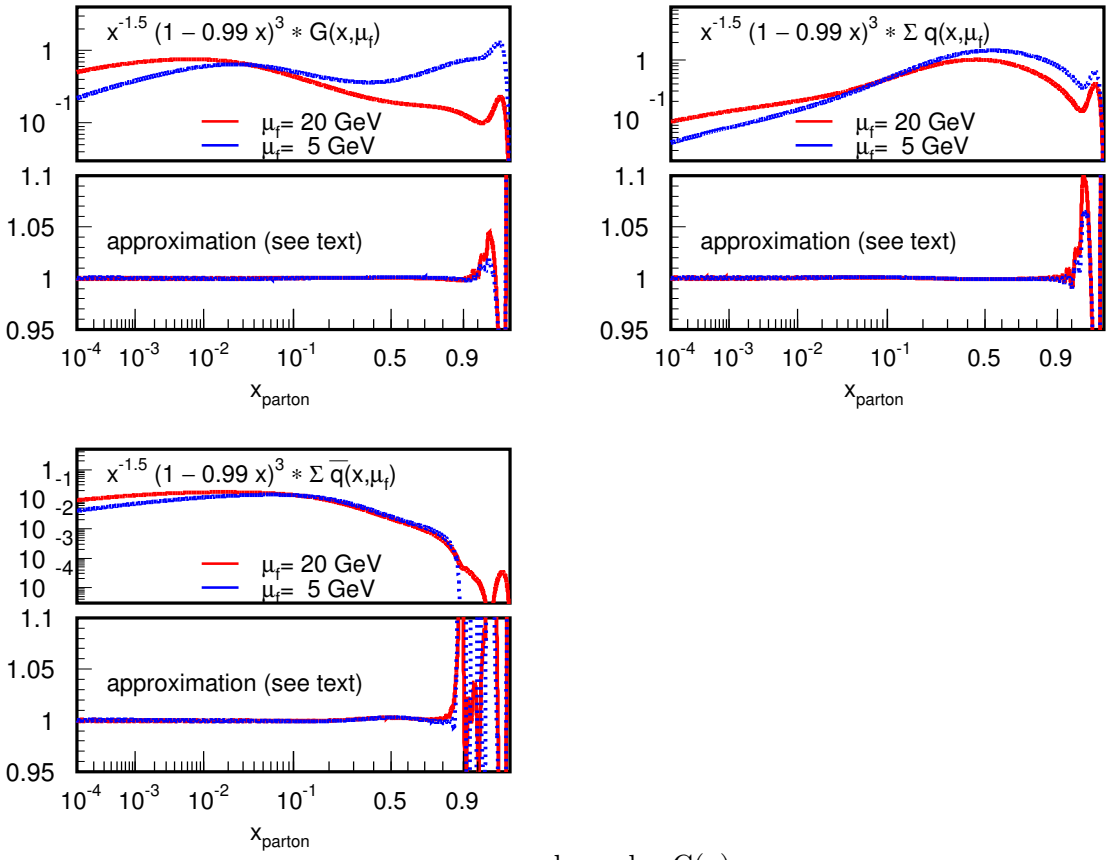
This describes only the contribution from direct photon exchange. The contribution from resolved photons is very similar to the hadron-hadron case, except that different PDFs are used for the photon and the hadron. (... can easily be extended – not described here ...)

In eq. (2.7) ...

A. Eigenfunctions for the PDFs

using triangular eigenfunctions — the $x^{(i)}$ are chosen such that the values $g(x^{(i)}$ are equidistant. Possible choices $g(x) = \log_{10}(1/x)$ (for not too large x), or $g(x) = \sqrt{\log_{10}(1/x)}$ (which expands the x range at large x drastically)).

The choice of triangular eigenfunctions corresponds to a linear approximation of the PDFs between adjacent $x^{(i)}$. Such a locally linear approximation can be significantly



show also C(x)

Figure 1: The three different pdfs (gluon, quarks and anti-quarks), taken from the CTEQ6.1M parameterization and normalized by the function $x^{1.5}(1-0.99x)^{-3}$ (top). The bottom plots show the precision of the linear approximation between adjacent bins (as described in the text).

improved by reweighting the PDFs with a function which makes them much more flat. We found empirically that this can be done at all factoriation scales studied (from .. to .. GeV). by reweighting with $w(x) = x^{-1.5}(1-0.99x)^3$. This function is therefore multiplied into the eigenfunctions (and later the PDFs are divided by $w(x)$).

In Fig. 1 we display the quality of the approximation ...

B. Examples for $\sqrt{s} = 1.96\,\text{TeV}$ and $14\,\text{TeV}$

C. The Implementation

... of the inclusive jet cross section ...

C.1 The x Bins

The accessible x range for any observable extends from $x = 1$ down to a minimum value $x_{\min}$ which is specific for each observable. To reduce the number of eigenfunctions needed to cover the accessible x range it is important to make a good (but safe!) estimate of the smallest accessible x-value in each bin of the observable. (see later).

C.2 Structure of the Tables

The output tables are designed such that all information can be extracted automatically (binning of the observable, x-binning, ...). The table consist of five parts: A header which is universal for all obervables, a list of numbers, describing technical details how the table is organized, a list of all bin boundaries (observable specific), a list of the lower limits for the x-ranges, and the list of all coefficients, convoluted with the eigenfunctions. The following example is for the inclusive jet cross section with a binning in p_T and rapidity. A restriction of the format is that it can only deal with a continuos binning in rapidity and in p_T.

```
******************************************************************************
*******                fastNLO  -  Result Table
******************************************************************************
ireaction       ! int: reaction       1 ep, 2 pp, 3ppbar
Ecms            ! dbl: center-of-mass energy in GeV
iproc           ! int: process type   1 incl jets, 2 dijets, ...
ialgo           ! int: jet algo 1 kT, 2 midpoint cone, 3 rsep cone, 4 search cone
JetResol1       ! dbl: jet resolution paramter (kT distance D / R_cone)
JetResol2       ! dbl: 2nd parameter for jet algo (e.g. Rsep, search cone)
Oalphas         ! dbl: order in alpha_s  (??absolute - or 1: LO, 2: NLO??)
1234567890      ! --- End-of-block --- now come some technical details
nevt            ! int: No of events used to create the table
nxbin           ! int: No of eigenfunctions that cover the x-ranges
ixscheme        ! int: No of scheme for EF x-binning  1 log 1/x  2 sqrt(log 1/x)
ipdfwgt         ! int: No of scheme for PDF weighting  0 no weigthing
1234567890           ! --- End-of-block --- now come the pT,y bin boundaries
Nrapidity            ! int: No of rapidity intervals
Rap0                 ! dbl: lower boundary of 1st rapidity bin
...                  !
Rap[Nrapidity]       ! dbl: upper boundary of last rapidity bin
Npt1                 ! int: No of pT bins in first rapidity interval
...                  !
Npt[Nrapidity]       ! int: No of pT bins in last rapidity interval
Pt-1-0               ! dbl: lower boundary of 1st pT bin in 1st rapidity bin
...                  !
Pt-1-[Npt1]          ! dbl: upper boundary of last pT bin in 1st rapidity bin
...                  !
...                  !
Pt-[Nrapidity]-0     ! dbl: lower boundary of 1st pT bin in last rapidity bin
...                  !
Pt-[Nrapidity]-[Npt?] ! dbl: upper boundary of last pT bin in last rapidity bin
1234567890           ! --- End-of-block --- now come the x_min values for all bins
xmin-1-1             ! dbl:  xmin in 1st rapidity bin / 1st pT bin
xmin-1-2             ! dbl:  xmin in 1st rapidity bin / 2nd pT bin
...                  !
xmin-1-[Npt1]        ! dbl:  xmin in 1st rapidity bin / last pT bin
xmin-2-1             ! dbl:  xmin in 2nd rapidity bin / 1st pT bin
...                  !
xmin-[Nrap]-[Npt?]   ! dbl:  xmin in last rapidity bin / last pT bin
1234567890           ! --- End-of-block --- now come the renorm. scales
murscale             ! dbl: overall scale factor to adjust the renorm. scale
```

```
murval-1-1            ! dbl: Array for all mu_r values - for all (y,pT) bins
...                   !             (same structure as xmin-Array)
murval-[Nrap]-[Npt?]  ! dbl:
1234567890            ! --- End-of-block --- now come the factoriz. scales
mufscale              ! dbl: overall scale factor to adjust the fact. scale
mufval-1-1            ! dbl: Array for all mu_f values - for all (y,pT) bins
...                   !           (same structure as mu_r and xmin Arrays)
mufval-[Nrap]-[Npt?]  ! dbl:
1234567890            ! --- End-of-block --- now come the sigma_tilde
...                   ! dbl: each rapidity range
...                   !        each pT bin
...                   !          seven subprocesses:
...                   !            each (of five) subprocesses:
...                   !               half-grid with (n**2+n)/2  x-bins
...                   !            remaining two subprocesses:
...                   !               can be combined to one full n**2 grid
...                   !                       (or better: two half grids)
1234567890            ! --- End of Table -----------------------------------
```

C.3 Some Formulae ....

C.3.1 Lowest Accessible x Value

A trivial, but not very efficient, lower limit on the accessible x is given by

$$x_{\text{limit}}^{\text{trivial}}(p_{T\,\text{min}}, |y|_{\text{max}}) \;=\; \frac{4p_{T\,\text{min}}^2}{\sqrt{s}}\,. \tag{C.1}$$

where $p_{T\,\text{min}}$ and $|y|_{\text{max}}$ are the respective lower and upper limits for a given $(p_T, |y|)$ bin. While this approximation is already very good at high p_T and at large $|y|$, in the region of low p_T and low $|y|$ it is too low by more than a factor of ten. For a better approximation of the smallest accessible x we define the factor c as

$$c = \left(2 \,+\, \frac{1}{\cosh(|y|_{\text{max}})} \,\left(\frac{\sqrt{s}}{2p_{T\,\text{min}}} - 1\right)\right) \,\cdot\, 0.22\,. \tag{C.2}$$

A very efficient approximation of the lowest accessible x is then given by

$$x_{\text{limit}}(p_{T\,\text{min}}, |y_{\text{max}}|) \;=\; x_{\text{limit}}^{\text{trivial}}(p_{T\,\text{min}}, |y|_{\text{max}}) \,\cdot\, \max\,(1.0, c)\,. \tag{C.3}$$

For $\sqrt{s} = 1960\,\text{GeV}$ and $p_{T\,\text{min}} = 50\,\text{GeV}$ the lowest value is $x_{\text{limit}} = 0.0026$.

C.3.2 Largest Accessible p_T Value

In a given rapidity range with lower limit $|y|_{\text{min}}$ the largest kinematically accessible p_T is given by

$$p_{T,\,\text{max}}(|y|_{\text{min}}) = \frac{\sqrt{s}}{1 + \cosh(|y|_{\text{min}})}\,. \tag{C.4}$$

In the central rapidity region this is an efficient estimate of the upper limit. However, in the forward region (at $|y| = 4.0$) this kinematic limit is above the observed limit by a factor of two.

for sqrt(s)=1960GeV:

y_min	kinematic pT limit	observed pT limit	factor
0.0	980	980	1
0.4	940	890	1.06
0.8	830	720	1.15
1.2	690	540	1.28
1.6	540	380	1.42
2.0	410	260	1.57
2.4	290	180	1.61
2.8	210	120	1.75
3.2	140	80	1.75
3.6	100	60	1.67

C.3.3 Linear Binning of the p_T and $|y|$ Phase Space

The jet phase space in p_T and $|y|$ is divided into $N_{\text{rap, max}}$ bins in rapidity $|y|$. Each $|y|$ bin with number $N_{\text{rap}}(|y|)$ is divided into $N_{p_T,\,\text{max}}(N_{\text{rap}}(|y|))$ bins. The p_T bin within a given

$|y|$ bin $N_{\rm rap} = i$ is $N_{p_T}^{(i)}$. In the end we want to address each $(|y|, p_T)$ on a linear scale. The bin number $N_{\rm bin,\,linear}$ on the linear scale is given by

$$N_{\rm bin,\,linear} \;=\; \sum_{i=1}^{N_{\rm rap}-1} N_{p_T,\,\rm max}^{(i)} \;+\; N_{p_T}^{(i)}\,. \tag{C.5}$$

C.3.4 Format of the x_1, x_2 Tables

Of the seven subprocesses, 1, 4, 5, 6, 7 are symmetric in (x_1, x_2), so they can be stored in terms od $x_{\rm max}$ and $x_{\rm min}$. Subprocesses 2, 3 are not symmetric, but their coefficients C_2, C_3 are related by $C_2(x_1, x_2) = C_3(x_2, x_1)$. So we can redefine these subprocesses into $D_2(x_{\rm max}, x_{\rm min})$ and $D_3(x_{\rm min}, x_{\rm max})$. If $x_1 \geq x_2$ we set $D_2(x_1, x_2) = C_2(x_1, x_2)$ and $D_3(x_1, x_2) = C_3(x_1, x_2)$. In the case $x_1 < x_2$ we set $D_2(x_2, x_1) = C_3(x_1, x_2)$ and $D_3(x_2, x_1) = C_2(x_1, x_2)$ (later, when the tables are multiplied with the PDFs we have to remember that in the re-defined third subprocess the indicees are swapped).

To store the coefficients we need $(n^2 + n)/2$ elements for the table where n is the number of bins to cover the x-range for each $(|y|, p_T)$-bin. This is multiplied by seven subprocesses (for $n = 30$ these are 3255 elements for each bin of the observable — ... and $n = 30$ could be too small.... in my thesis I used 25 x bins per decade in $\log_{10}(1/x)$, which was very good approximation up to $x = 0.5$).

The total number of table elements is then

$$N_{\rm tot} = N_{\rm bin,\,linear,\,max} \cdot 7 \cdot \frac{n^2 + n}{2}. \tag{C.6}$$

For the DØ Run I measurement of the inclusive jet cross section in 90 bins, this would be 292,950 table elements. If we wanted to provide a fully flexible table for the Run II measurements in $|y|$ bins of 0.1 from $0 < |y| < 2.5$ and p_T bins of 10 GeV from $50 < p_T < 700$ GeV (varying with rapidity) this can easily be 3,255,000 elements.

C.3.5 The x-Binning

The coefficients are stored on a two-dimensional x grid at discrete values $x^{(i)}$. Between the value $x^{(i)}$ and $x^{(i-1)}$ we are using a linear interpolation to approximate the PDFs. The precision of the approximation will, of course, depend on the choice of the $x^{(i)}$. The binning is done in equidistant intervals in the function $f(x)$. For $f(x)$ we propose two scenarios: Either $f_1(x) = \log_{10}(1/x)$ or $f_2(x) = \sqrt{\log_{10}(1/x)}$. The former is a natural choice for most observables which are only sensitive to smaller values of x. Above $x = 0.5$ the PDFs exhibit a strong curvature and a linear approximation becomes worse. In these cases we propose to use $f_2(x)$ which allows for a finer binning of the large-x range (the present figures in this note are using $f_2(x)$). One could also think of using different functions for the two different x-values in an event. **(looking at a few events, I would actually propose this for the inclusive jet cross section – on the other hand this would not allow to use a 'half grid' anymore. Question: is it worth to almost double the grid size to get a better approximation? Probably "yes". The alternative would be, to use the poorer approximation and increase the number of x bins. But already a small**

increase of the bin number by a factor of 1.4 would also increase the grid by a factor of two. Conclusion: either use f_2 for both x values and use half grids – or combine $f_1(x_{\min})$ and $f_2(x_{\max})$ and use a full grid.)

We assume that all PDFs go to zero for $x = 1$, so we don't need to store the coefficients at $x^{(0)}$. (please note that for this reason, the numbers for the coefficients in the C++ code are reduced by one!)

For every jet four (x_1, x_2) bins receive contributions: $(x_1^{(i)}, x_2^{(j)})$, $(x_1^{(i+1)}, x_2^{(j)})$, $(x_1^{(i)}, x_2^{(j+1)})$, and $(x_1^{(i+1)}, x_2^{(j+1)})$. We need to compute i, j, as well as the distances between x_1 and $x^{(i)}$ and x_2 and $x^{(j)}$ to obtain the respective contributions for the four bins. The values i and j are obtained as

$$i = \mathrm{Int}\left(n_{\mathrm{tot}} \cdot \frac{f(x_{\max})}{f(x_{\mathrm{limit}})}\right) \qquad \text{and} \qquad j = \mathrm{Int}\left(n_{\mathrm{tot}} \cdot \frac{f(x_{\min})}{f(x_{\mathrm{limit}})}\right), \tag{C.7}$$

with $(0 \le i, j < n_{\mathrm{tot}})$ where n_{tot} is the total number of x bins used. **Note that in the C++ code we subtract 1 fom i, j** The distances between adjacent bins are given by $\Delta = f(x_{\mathrm{limit}})/n_{\mathrm{tot}}$. So we compute

$$\delta_1 = \frac{f(x^{(i)}) - f(x_1)}{\Delta} \qquad \text{and} \qquad \delta_2 = \frac{f(x^{(j)}) - f(x_2)}{\Delta} \qquad (0 < \delta_{1,2} \le 1), \tag{C.8}$$

where the value of $f(x^{(i)})$ is given by $\frac{i}{n_{\mathrm{tot}}} f(x_{\mathrm{limit}})$. The four bins receive the following values

$$(x_1^{(i)}, x_2^{(j)}) \ : (1-\delta_1)\,(1-\delta_2) \cdot C\,, \tag{C.9}$$

$$(x_1^{(i+1)}, x_2^{(j)}) \ : \delta_1\,(1-\delta_2) \cdot C\,, \tag{C.10}$$

$$(x_1^{(i)}, x_2^{(j+1)}) \ : (1-\delta_1)\,\delta_2 \cdot C\,, \tag{C.11}$$

$$(x_1^{(i+1)}, x_2^{(j+1)}) \ : \delta_1\,\delta_2 \cdot C\,, \tag{C.12}$$

where C is the perturbative coefficient for the specific subprocess. (To be checked by the reader: prove that the sum of all four contributions is equal to one, i.e. that nothing is getting lost).

D. PDF Correction Aposteriori

By assuming that the PDFs are locally linear in $f(x)$ we introduce an error. By increasing the number of bins n_{tot}, this error can be made arbitrarily small. However, since the tablesize is proportional to n_{tot}^2 there is a practical limit to this. One can also improve the approximation later, by computing correction terms based on the comparison of the PDFs inbetween the discrete values $x^{(i)}$ with the locally linear approximation. One could e.g. integrate the difference between the approximation and the true PDF, weighted by one minus the relative distance to the bin-center

$$c_{(i)} \ = \int_{x^{(i-1)}}^{x^{(i+1)}} dx\ (\mathrm{PDF}(x) - \mathrm{approximation}(x)) \cdot \left(1 - \frac{|f(x) - f(x^{(i)})|}{\Delta}\right). \tag{D.1}$$

This needs to be done in both dimensions (x_1, x_2). In principle one could also integrate in two-dimensional (x_1, x_2) space but it is not too likely that the improvement would justfy the effort (to be tested ....). (note: don't integrate beyond last bin which is non-zero)

E. CPU Usage

To test if the performance depends on the size (and maybe the organization) of the coefficient array we study the CPU time to process 1M events for different numbers of x bins ($nxbin$) for 275 analysis bins in $(|y|, p_T)$ (five rapidity regions with 65, 60, 55, 50 and 45 p_T bins). This test is done in LO:

```
Nevts  time     scenario (all jobs: LO)
1M   00:00:17   nxbins=2
1M   00:00:18   nxbins=30  (program size in memory:  15MB)
1M   00:00:19   nxbins=60  (program size in memory:  54MB)
1M   00:00:19   nxbins=100 (program size in memory: 149MB)
```

The CPU time does not depend on the size of the array, so further optimization is not needed (at least as long as the the number of analysis and/or x bins is not increased and the memory size of the computer is exceeded). For the given scenario we can easily use 100 x bins and still obtain a reasonable program size. Please note that for the above comparisons the internal α_s and PDF calculations in NLOJET++ had been disabled. Now we investigate the CPU resources used by the internal NLOJET++ histogram package and by the internal α_s and PDF calculations.

```
Nevts   time     scenario (all: LO jobs)
 1M   00:00:20  as above: nxbins=30
 1M   00:00:54   + enable alpha_s+PDF calculations  (factor 2.7)
 1M   00:01:47   + book and fill 2 histograms with total of 70 bins (f=5.3)
 1M   00:04:22   + book and fill 10 histograms with total of 259 bins (f=13)
 1M   00:22:20   + book and fill 40 histograms with total of 957 bins (f=67)
                 (times don't change if histo-filling is disabled)
```

In LO there are huge improvements in the CPU time usage when using our method, as compared to the default usage of NLOJET++. Now we compare the CPU usage in NLO calculations:

```
Nevts  time     scenario (all: NLO jobs)
100k  00:00:24  nxbins=30
100k  00:00:58   + include NLOJET histos with  70 bins   (f=2.4)
100k  00:01:39   + include NLOJET histos with 259 bins   (f=4.1)
100k  00:03:29   + include NLOJET histos with 957 bins   (f=8.7)
```

The results show that the computation of the NLO coefficients in NLOJET++ increases the total CPU time by a factor of twelve. Due to the overall increase of CPU usage the relative contribution from the internal histogramming becomes smaller. This reduces the advantage of our method, although it is still significantky fast (between factors of four and eight for a larger numbers of bins).

However, the overall efficiency can still be improved, by computing the cross sections for multiple renormalization and factorization scales in a single job. The coefficient table is therefore increased by a factor of five, to compute separately the up and down variations for both scales in a single job, together with the default choice.

```
Nevts  time     scenario (all: NLO jobs)
100k  00:00:38  nxbins= 30 - 5 scales               (memory: 68MB)
100k  00:00:38  nxbins= 60 - 5 scales               (memory: 267M)
       ....     nxbins=100 - 5 scales -> swapping almost hangs the PC
```

By computing all five scales in a single job, one gains a factor of 3.1, as compared to running five independent jobs (and, of course, statistical fluctuations in the ratios are reduced). However, one needs to be careful that one does not reach the memory limit of the computer, at which memory swapping becomes an issue.

If we add more NLOJET++ histograms to compute five scale choices in the same job we obtain the following CPU times

```
 Nevts  time     scenario (all: NLO jobs)
 100k  00:00:38  nxbins=30 - 5 scales               (memory: 68MB)
 100k  00:03:32   + include NLOJET histos with  5*70 bins   (f=5.6)
 100k  00:06:36   + include NLOJET histos with 5*259 bins   (f=10)
 100k  00:18:30   + include NLOJET histos with 5*957 bins   (f=29)
```

It is visble that the jobs with the NLOJET++ histograms do not benefit significantly from computing five scales in parallel, as compared to running individual jobs. The overall improvement of our method is therefore becoming even better (between factors of five and thirty in our examples).

F. Next Steps ...

Work on user Routine:

```
- define flexible pT Array (use in mu_r,mu_f, etc.)
- introduce switch to switch on/off 4 add. weight arrays for scale studies
- Can we put ‘‘initfunc’’ and ‘‘inputfunc’’ in separate file?
  These two routines contain all the switches needed - would be nice if separated.
- Finalize Definition of Table (done - only need details on numbering of x-bins)
  - test ‘‘swap’’ problem (comment weight filling - compare CPU time
  - write full table from NLOJET++
```

Work on Fortran stand-alone code:

```
  - write Fortran Code to add tables (or better C++ code which can read binary format?)
  - write Fortan code to read table and output results in text file
    as PAW vectors (including PAW code to define Histograms)
  - default: read 6 tables: LO +NLO +4 scale variations
    (scale variations can be switched off)
  - include switch for PDF correction (on/off)  (-> test)
```

When everything is set up - make test jobs:

```
*** (1) technical studies: is the implementation correct?
        (relation i<->bin numbering, subprocesses 2,3, etc...)
101 - leave PDFs and alpha_s in NLOJET code included in weights:
102 - run job with many nxbins - four (y,pT) bins
         (choose 2 low pT, 2 high pT; 2 central, 2 forward bins)
103 - run job with many nxbins, over large y,pT range (many bins)
      compare direct output and sum of coefficients
      (should be exactly identical - including stat. fluctuations)
      do this for both LO (higher precision) and NLO (check fluctuations)
```

```
 -> plot ratios of histograms table/NLOJET

*** (2) check the quality of the PDF approximation
   - remove the PDFs in NLOJET
       (but keep alpha_s, to be sure we are using the same formula)
   - check that we get the same result from the table as in NLOJET
201  - first use large nxbin & large y,pT phase space
       do high precision LO jobs, so we can really judge the physics
       plot ratios Table/NLOJET(with error)
202  - plot the x-distributions (can be done from the table!)
       -> important to get a feeling for x1,x2 phase space
203  - reduce nxbin to see how the approximation gets worse
204  - try PDF correction to see if it works & improves things
205  - find optimum: (nxbin + PDF correction) -> be conservative!!

*** (3) do the real thing (for incl. jets)
301  - set up jobs for Run I (question: can we store the CDF jets
       parallel to the D0 jets? -> overlapping bins - need to assign
       two grid places for the same jet - but: very efficient!)
       -> need to use Run I algo: ET scheme - ET,eta, Rsep
       -> this will be used by CTEQ,MRST,Alehkin,H1,ZEUS,...
302  - jobs for the LHC (for workshop - find good binning!)
303  - jobs for Run II (start with current D0 binning)
   -> in all cases: one LO job plus multiple NLO jobs for different scales

*** (4) think about different observables: dijet mass, Chi-Angle,
DeltaPhi, three-jet variables, ....
```

Decide: at which level publish paper??
Important:
Include also some "new" physics messages – especially for LHC!
Easy-to-do stuff: extensive (y, p_T) dependent k-factor and scale studies – search best scale; x-sensitivity for different subprocesses: $x_{\min}, x_{\max}$ plots – and for qg subprocesses: $x_{\text{gluon}}, x_{\text{quark}}$ – can all be very easily extracted from tables!