The DISENT Program

Abstract

This is a collection of notes about the DISENT program, which implements the Catani-Seymour algorithm for next-to-leading order corrections to $(1+1)$- and $(2+1)$-jet event observables in DIS. It is not intended to describe the algorithm itself, which is written up elsewhere. It will hopefully evolve into proper program documentation at some stage though.

Overview

The program is written somewhat like an event generator — a main routine administers the generation of parton momenta, calls the matrix element routines to calculate their weight, and finally calls a user routine to analyse them, with the momenta stored in something vaguely resembling an event record.

The program itself does not do any analysis of the events. This is left entirely to the user routine. However, it does come with a set of demonstration routines showing the kind of analysis one might like to do.

In addition, the program does not contain an α_s calculation, or any parametrizations of parton distribution functions. If one is only calculating perturbative coefficient functions these are not even necessary. However, for physical cross sections they are, so the demonstration routines show how it can be easily interfaced to PDFLIB.

The most important point to note about the algorithm is that the singularities of the $(m+1)$-parton real matrix element are cancelled by subtracting several $(m(m-1)^2/2)$ $([m-1]+1)$-parton counterterms. These are weighted such that when the $(m+1)$-parton term diverges, so do one or more $([m-1]+1)$-parton terms, so that their difference stays finite. Therefore an 'event' consists of many separate sets of momenta with coupled weights, and it is important that the user routine takes this into account. If one is not interested in the errors on the calculated quantities this is trivial — events can be directly histogrammed. However, if error bars are needed it is important that all the sets of momenta are entered into the histogram in one go at the end of the event, so that large positive and negative weights have been cancelled already.

Phase-space generation

Although we said that the remaining cross-section is finite, it is not actually. It is guaranteed to have finite integral, but can (and does in general) contain integrable singularities, most importantly square-root singularities. These are notorious for spoiling the convergence of Monte

Carlo integrals (their variance is infinite). They must therefore be removed by suitable Jacobian factors. In DISENT this is done by using a fairly sophisticated multi-channel system.

The phase-space generation looks at first sight something like a parton shower — first a parton-model ($(1+1)$-parton) event is generated, then, using this as a starting point, a $(2+1)$-parton event is generated using dipole kinematics, conserving the x and Q^2 of the $(1+1)$-parton event. Finally a $(3+1)$-parton event is generated in the same way, using two of the three partons chosen at random. However, unlike most parton showers, this can be described as a proper transformation of the full $(3+1)$-parton phase-space such that the Jacobian factor can be easily calculated. Different choices of which parton emits and which absorbs the recoil correspond to the different channels of the multi-channel integration. In principle, the convergence might be improved by assigning different weights to different channels. Algorithms even exist to optimize this automatically, but in practice we find that the convergence is perfectly satisfactory simply assigning the same weight to all channels. As usual in multi-channel Monte Carlo, the Jacobian factor is the sum of the factors for all channels, not just the one that was used to generate the event.

The square-root singularities are removed by ensuring that for each variable x in which the full matrix element is singular as $1/x$ (i.e. two-parton invariant mass-squareds and parton energies), x is generated according to

$$\frac{dx}{x^{1-1/n}},$$

where n is an adjustable parameter. Clearly $n = 1$ gives flat phase-space, $n = 2$ removes square-root singularities, while even larger n pays even more attention to the almost-singular regions. Obviously the appropriate Jacobian factors are applied so that the final result does not depend on the value of n, only the size of the errors does. There are actually two adjustable parameters, `NPOW(1)` and `NPOW(2)`, defaulting to 2 and 4 respectively, which control the $(1+1) \to (2+1)$ and $(2+1) \to (3+1)$ parton steps respectively. For most studies the default values are sufficient, but if for example one was studying the next-to-leading order corrections to a $(2+1)$-jet quantity in the extreme $(1+1)$-jet-like region, increasing `NPOW(1)` would improve the convergence. For historical reasons (certain constants used to be tabulated rather than being calculated from scratch) `NPOW(1)` and `NPOW(2)` are integer parameters.

The initial parton-model event is chosen according to phase space limits on x, Q^2 and y. Any of them can be fixed at a given value, or limited to a given interval (obviously only two of the three can be exactly fixed). It is not (yet) possible to choose limits that are arbitrary functions of x, Q^2 and y such as HERA-frame angles or energies (although of course these can be implemented as cuts on the generated events in the user routine). Any of the variables that are not exactly fixed are generated uniformly in dx/x, dQ^2/Q^2 and dy/y.

Matrix elements

The matrix elements used are those of the Leiden group (somewhat massaged by us for more convenient numerical evaluation). At present they only include photon exchange. The electromagnetic coupling constant does not run, but is fixed at $\alpha = 1/137$. There is no attempt to keep track of the flavour of outgoing partons, since all partons are considered equal in infrared-safe jet quantities.

Cross section normalization

The Monte Carlo weights are normalized differently, according to the lepton cuts specified by the user. If none of the variables are fixed, the weights are normalized to the total cross section within the selected region, σ, in nanobarns. If one of the variables are fixed, then the normalization is to the differential cross section in that variable, $\frac{d\sigma}{dx}$ (nb), $\frac{d\sigma}{dQ^2}$ (nb/GeV^2) or $\frac{d\sigma}{dy}$ (nb), integrated over the selected regions of the other variables. If two of the variables are fixed, then the normalization is to the double-differential cross sections, $\frac{d\sigma}{dx\,dQ^2}$ (nb/GeV^2), $\frac{d\sigma}{dy\,dQ^2}$ (nb/GeV^2) or $\frac{d\sigma}{dx\,dy}$ (nb).

The main routine

The main routine is called DISENT. It takes five arguments,

```
SUBROUTINE DISENT(NEV,S,NFL,USER,CUTS)
INTEGER NEV,NFL
DOUBLE PRECISION S
EXTERNAL USER,CUTS
```

`NEV` is the number of events to generate,
`S` is the total lepton-hadron centre-of-mass energy^2,
`NFL` is the number of flavours produced by gluon splitting,
`USER` is the name of a user-supplied analysis routine, and
`CUTS` is the name of a user-supplied routine giving the cuts on the parton-model event.

The program starts by setting various parameters. Most of these are pretty standard and would never need changing, but at some stage perhaps an improved interface in which the default parameters can be modified should probably be provided. The internal parameters are described below.

The user routine

The user must supply a routine to analyze events

```
SUBROUTINE USER(N,NA,ITYPE,P,S,WEIGHT)
INTEGER N,NA,ITYPE
DOUBLE PRECISION P(4,7),S,WEIGHT(-6:6)
```

`N` is the number of partons in the 'event record',
`NA` is the order of α_s (i.e. 0, 1 or 2),
`ITYPE` is the type of contribution (shouldn't be needed for physical observables but provided anyway): 0=tree level, 1=subtraction counterterm, 2='finite virtual' term, 3='finite colinear' term,
`P` is the primitive 'event record': `P`(1–4, i) is the momentum of the ith entry. Entry 1 is the incoming parton. Entries 2–`N` are the outgoing partons. Entry 5 is the virtual photon four-momentum. Entries 6 and 7 are the incoming and outgoing lepton.
`WEIGHT(i)` is the weight of events initiated by partons of flavour i: $1 \ldots 6$ for d,u,s,... t, 0 for gluons and $-1 \ldots -6$ for $\bar{\mathrm{d}}, \ldots, \bar{\mathrm{t}}$.

The weights are normalized such that their *total* value is the final cross-section specified above. Thus they do not need to be renormalized at the end of the run. At each order in α_s, a factor of $\left(\frac{\alpha_s}{2\pi}\right)^{\mathtt{NA}}$ is extracted. Furthermore, the normalization is such that to form physical cross sections one must multiply the given weights by $\eta f(\eta)$, where $f(\eta)$ is the parton distribution function.

The whole of the DISENT program is Lorentz invariant. The momenta can be specified in any frame in routine `GENTWO`, which generates the original $(1+1)$-parton event, and the choice will carry through the whole procedure. However, the default frame is (of course!) the Breit frame, so it should be safe to assume that in the user routine. For die-hard dogmatists it is possible to write any event shape in a Lorentz-invariant way though, so it should be possible to avoid any assumption about the frame used.

In some event types (the 'finite colinear' term, and subtraction terms from initial-state dipoles) one of the partons can be exactly colinear with the incoming parton. These have no effect on colinear-safe jet observables. They could be removed, at the expense of violating momentum conservation (the incoming momentum has to be the sum of the outgoing ones *and the colinear parton*). To save keeping track of whether a particular subtraction term came from an initial- or final-state dipole, the final-state ones have a zero-momentum parton inserted at the same position in the 'event record'.

==========

Add and anotate the demonstration user routine!

==========

The common blocks

There are two common blocks. They are mainly intended to be for internal use, but in fact one contains two variables that are needed by the user routine, so perhaps it should be reorganized into a different block, or passed to the user routine as a parameter...

COLFAC holds all the constants used by the program,

```
INTEGER SCHEME,NF
DOUBLE PRECISION CF,CA,TR,PI,PISQ,HF,CUTOFF,EQ(-6:6),SCALE
COMMON  /COLFAC/ CF,CA,TR,PI,PISQ,HF,CUTOFF,EQ,SCALE,SCHEME,NF
```

All values are set at the beginning of DISENT. At present there is no possibility to change them without recompiling. The first few should be reasonably obvious, (with TR = HF = $\frac{1}{2}$).

CUTOFF is an infrared cut on the phase-space. If any pair of partons has $|m_{ij}^2| <$ CUTOFFQ^2 the whole event is thrown away. Strictly speaking, the algorithm does not need an infrared cutoff — all integrals are finite, so it can be set to zero. However, for very small m_{ij} one ends up taking the difference between two very large numbers and the numerical inaccuracy of the machine becomes important. The default value CUTOFF $= 10^{-8}$ works just fine. In fact it can probably be safely reduced somewhat, but I haven't made a detailed study as yet...

EQ(i) holds the charge of parton type i, in units of the proton charge. NF is just a copy of the user input NFL.

SCHEME indicates the factorization scheme used. At present there are only two: zero gives $\overline{\text{MS}}$, and non-zero gives DIS. SCALE indicates the factorization scale used, as a multiple of Q^2 (i.e. we set $\mu_F^2 =$ CUTOFFQ^2). It is not at present possible to set it on an event-by-event basis, as would be needed to implement scales not proportional to Q^2. These are the two variables that must be known by the user routine, in order to be able to evaluate the parton distribution functions.

SAMPLE holds the constants associated with importance sampling.

```
INTEGER NPOW(2)
DOUBLE PRECISION XPOW(2)
COMMON  /SAMPLE/ XPOW,NPOW
```

NPOW was described above, and XPOWi is simply shorthand for $1 - 1/$NPOWi.

There is also a common block SUBCOM in some routines, which is included for the sake of compatability with EVENT2, the e^+e^- program, and is not used by DISENT.

Known bugs, problems and future development

DISENT is in a more primitive state than EVENT2 was when it was released. Therefore it can be expected to develop more. This is to some extent also a reflection of the fact that there are many more types of calculation one can chose to make in DIS than in e^+e^-.

An important bug was found in version 0.0. One of the dipole contributions was actually subtracted twice. Therefore the amount by which the program was wrong is formally divergent. In practice, this means it is of order log(`CUTOFF`). Numerically, we find that this log has a very small coefficient, so the actual effect on physical observables is small, of order 1% in most regions.

The only known bug is the fact that the cross section normalizations listed above are not actually true: the program actually only implements limits on y and Q^2 internally — if limits on x are given, they are translated into limits on the others. This means that the normalization gets an additional factor, because what should be $\frac{d\sigma}{dx}$ is actually $\frac{d\sigma}{dy}\Big|_x = \frac{x}{y}\frac{d\sigma}{dx}$. This should be fixed soon...

There are a whole host of minor problems that will be fixed over the coming months. Several constants that users might want to set are actually hard-coded in DISENT. The implementation of lepton cuts should be improved to enable arbitrary cuts to be made. The routine that implements the dipole splitting functions for the subtraction term is actually rather badly written (it is just a long list of special cases) and will in future be rewritten to make the process-independence manifest. Also, at present no attempt has been made to integrate DISENT with EVENT2, but large chunks are very similar, so in future they should both be set up to make use of a library of common routines.

As for future development, the main development on the horizon is the inclusion of arbitrary currents, including Z exchange, charged currents and neutrino-induced events.

Modification Log

0.0 7/10/96 First prerelease version put on web page.

0.1 22/10/97 Important bug fixed. Several other minor improvements.