Preprint typeset in JHEP style - PAPER VERSION

August 1, 2012
POWHEG BOX 1.0

The POWHEG BOX user manual: Single-top t-channel in the 4-flavour scheme

Emanuele Re

Institute for Particle Physics Phenomenology, Department of Physics
University of Durham, Durham, DH1 3LE, UK
E-mail: emanuele.re@durham.ac.uk

Abstract: This note documents the use of the package POWHEG BOX for the single-top t-channel production process computed in the 4-flavour scheme. Results can be easily interfaced to shower Monte Carlo programs, in such a way that both NLO and shower accuracy are maintained. Please read carefully the manual before using this code.

Keywords: POWHEG, Shower Monte Carlo, NLO.

Contents

1. Introduction

The POWHEG BOX program is a framework for implementing NLO calculations in Shower Monte Carlo programs according to the POWHEG method. An explanation of the method and a discussion of how the code is organized can be found in refs. [1, 2, 3]. The code is distributed according to the "MCNET GUIDELINES for Event Generator Authors and Users" and can be found at the web page

http://powhegbox.mib.infn.it.

In the following we will focus on the implementation of single-top t-channel production in the 4-flavour scheme, whose source files can be found in the POWHEG-BOX/ST_tch_4f subdirectory.

This program is an implementation of the NLO cross section calculated in [4] in the POWHEG formalism of refs. [1, 2]. Tree level Born and real matrix-elements were obtained from scratch, whereas the one-loop contribution was taken from the MCFM package.

A detailed description of the implementation, together with a comparison with the results for the same process as implemented with the MC@NLO method, can be found in ref. [5]. In the current version, spin correlations of the top-quark decay products are not included, which means that angular correlation between the production and the decay stage are not present.

2. Installation

Before describing the installation procedure, we want to recall that a NLO+PS simulation of the single-top t-channel process, in the "5-flavour" scheme, is already present in the POWHEG BOX package (POWHEG-BOX/ST_tch directory, ref. [6]). However, the process of single-top t-channel production can be described also in a different scheme, namely the

"4-flavour" scheme. Although the 4- and 5-flavour prescriptions are formally equivalent if all the perturbative expansion were taken into account, they can show differences when the perturbative series is truncated, as unavoidably happens in a real computation. Both schemes have advantages and disadvantages, depending on the observables studied. The purpose of this implementation is to allow a simulation in the 4-flavour scheme too, in order to allow for an assessment of these differences with NLO+PS accuracy. More commments and a more detailed explanation of differences and similarities between the 2 approaches can be found in ref. [5], and in the references therein.

Please notice that the PDF choice is **not** arbitrary for this implementation: a 4-flavour PDF set (as for example `MSTW2008nlo68cl_nf4`) must be used for consistency of the calculation. To this aim, the installation of LHAPDF [7] is strongly suggested. Notice also that, in the current version, the program doesn't check explicitly that such a set is actually selected in the input file.[1]

In order to run the `POWHEG BOX` program, we recommend the reader to start from the `POWHEG BOX` user manual, which contains all the information and settings that are common between all subprocesses. In this note we focus on the settings and parameters specific to the single-top t-channel 4-flavour implementation.

3. Generation of events and showering

The executable is built with the following commands

```
$ cd POWHEG-BOX/ST_tch_4f
$ make pwhg_main
```

In the `testrun` folder, there are examples of input files. For example, you can start a run doing

```
$ cd testrun
$ ../pwhg_main
```

and then insert the prefix of an input file when prompted. The input file read in this case will be `<prefix>-powheg.input` and at the end a file named `<prefix>-events.lhe` will contain events, in the Les Houches format. To shower them with `PYTHIA` do

```
$ cd POWHEG-BOX/ST_tch_4f
$ make main-PYTHIA-lhef
$ cd testrun
$ ../main-PYTHIA-lhef
```

[1] It is possible to add in the code some compensating terms, to make the computation consistent with a 5-flavour PDF set. By default these terms are off. An explicit switch to turn them on from the input card could be made available upon request, if needed.

Similar commands will run the HERWIG shower.

It is possible to speed up the generation of events by splitting this stage in several short runs, producing several .lhe files. The full description of this procedure can be found in POWHEG-BOX/Docs/Manyseeds.pdf, and a summary of it, suited for this implementation, is reported in the following.
To parallelize the event generation, the code has first to be run setting in the input card

```
numevts 0
```

When this run is finished all the grids for subsequent runs will be ready, but no event file will be present yet. At this point, the user should uncomment the option manyseeds in the input card:

```
manyseeds 1
```

and decide the number of events for each short run, setting numevts as usual. In order to start the event generation, a file named <prefix>-seeds.dat has to be created, with the following format:

```
randomseed1
randomseed2
randomseed3
..
..
```

where each line has to be the integer that will be used to initialize the random numbers generator. At this point, by running the code with the usual command pwhg_main, the user will be asked not only the prefix of the input file, but also an integer. This integer corresponds to the line in <prefix>-seeds.dat that the program will read to initialize the random numbers generator. The program will use all the grids generated, and the event file produced will contain in the name also the integer typed, so there is no risk of overwriting files.

4. Process specific input parameters

```
facscfact 1 !  factorization scale factor:  mufact=muref*facscfact
renscfact 1 !  renormalization scale factor:  muren=muref*renscfact
```

Factorization and renormalization scale factors appearing here have to do with the computation of the inclusive cross section (i.e. the $\bar{B}$ function [1, 2, 3]), and can be varied by a factor of order 1 to study scale dependence. A physically motivated choice for this process

is of the order of the transverse mass of the b-quark [5]. The default scale choice is therefore

$$\mu = 4\sqrt{(m_b^2 + p_{T,b}^2)}\,,$$

although other scale choices are possible. The experienced user can change this setting modifying the `set_fac_ren_scales` routine.

It follows a description of parameters which are relevant for this production process:

- When generating events with this implementation, negative weights can appear. As usual the fraction of negative weights in `POWHEG` can be largely reduced increasing `foldcsi`, `foldy`, `foldphi`. Allowed values are 1, 2, 5, 10, 25, 50. The speed of the program is inversely proportional to the product of these numbers, so that a reasonable compromise should be found. When using the folding parameters provided in the examples, the fraction of negative-weighted generated events is quite small (few %), even at LHC energies. However, it is recommended to keep these events in the generated sample, by keeping the `withnegweights` flag enabled, as in the example input files.
- Enabling the `withdamp` option can slightly improve the speed of event-generation. Notice that, for this process, no special damping function is needed (cfr. the Higgs via gluon-fusion implementaion and the large $\bar{B}/B$ factor). The only effect here is to improve the efficency of event generation for events where the ratio R/B in the `POWHEG` Sudakov exponent becomes too large with respect to its corresponding collinear or soft approximation [3].

The other relevant parameters are:

```
ttype 1              !  1 for t, -1 for tbar
topmass 175.0        !  top mass
bmass 4.75           !  b mass
wmass 80.398         !  w mass
sthw2 0.2226459      !  (sin(theta_W))**2
alphaem_inv 132.3384 !  1/alphaem

charmthr 1.5
bottomthr 4.75

CKM_Vud 0.9740         !  CKM matrix entries ...
CKM_Vus 0.2225
CKM_Vub 0.000001
CKM_Vcd 0.2225
CKM_Vcs 0.9740
CKM_Vcb 0.000001
CKM_Vtd 0.000001
```

```
CKM_Vts 0.000001
CKM_Vtb 1.0
```

The value of `ttype` is mandatory and it is used to decide if top or antitop quarks will be produced. The meaning of all the other parameters is self-explanatory. We remind that it is not allowed to set any entry of the CKM matrix exactly equal to zero.

References

[1] P. Nason, "A new method for combining NLO QCD with shower Monte Carlo algorithms," JHEP **0411** (2004) 040 [arXiv:hep-ph/0409146].

[2] S. Frixione, P. Nason and C. Oleari, "Matching NLO QCD computations with Parton Shower simulations: the POWHEG method," JHEP **0711** (2007) 070 [arXiv:0709.2092 [hep-ph]].

[3] S. Alioli, P. Nason, C. Oleari and E. Re, "A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX," JHEP **1006**, 043 (2010) [arXiv:1002.2581 [hep-ph]].

[4] J. M. Campbell, R. Frederix, F. Maltoni and F. Tramontano, "Next-to-Leading-Order Predictions for t-Channel Single-Top Production at Hadron Colliders," Phys. Rev. Lett. **102**, 182003 (2009) [arXiv:0903.0005 [hep-ph]].

[5] R. Frederix, E. Re and P. Torrielli, "Single-top t-channel hadroproduction in the four-flavour scheme with POWHEG and aMC@NLO," arXiv:1207.5391 [hep-ph].

[6] S. Alioli, P. Nason, C. Oleari and E. Re, "NLO single-top production matched with shower in POWHEG: s- and t-channel contributions," JHEP **0909**, 111 (2009) [arXiv:0907.4076 [hep-ph]].

[7] M. R. Whalley, D. Bourilkov and R. C. Group, "The Les Houches accord PDFs (LHAPDF) and LHAGLUE," [arXiv:hep-ph/0508110].