Preprint typeset in JHEP style - PAPER VERSION

July 13, 2012
POWHEG BOX 1.0

The POWHEG BOX user manual: $W^{\pm}$ production

Simone Alioli

LBNL & UC Berkeley
1 Cyclotron Road, MS 50A 5104
94720 CA Berkeley, USA
E-mail: salioli@lbl.gov

Paolo Nason

INFN, Sezione di Milano-Bicocca, Piazza della Scienza 3, 20126 Milan, Italy
E-mail: Paolo.Nason@mib.infn.it

Carlo Oleari

Università di Milano-Bicocca and INFN, Sezione di Milano-Bicocca
Piazza della Scienza 3, 20126 Milan, Italy
E-mail: Carlo.Oleari@mib.infn.it

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 $W^{\pm}$ production processes. Results can be easily interfaced to shower Monte Carlo programs, in such a way that both NLO and shower accuracy are maintained.

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.

This program is an implementation of the Drell-Yan NLO cross sections $pp \to W \to \ell\nu$. A detailed description of the implementation can be found in ref. [4]. Please cite the paper when you use the program. The implementation included in the POWHEG-BOX/W subdirectory package is based on the subtraction scheme by Frixione, Kunszt and Signer rather than on the scheme by Catani and Seymour discussed in the paper.

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 W implementation.

2. Generation of events

Build the executable

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

Then do (for example)

```
$ cd testrun-wp-lhc
```

```
$ ../pwhg_main
```
At the end of the run, the file `pwgevents.lhe` will contain 100000 events for $W^+ \to e^+\nu_e$ in the Les Houches format. In order to shower them with `PYTHIA` do

```
$ cd POWHEG-BOX/W
$ make main-PYTHIA-lhef
$ cd testrun-wp-lhc
$ ../main-PYTHIA-lhef
```

3. Process specific input parameters

For $W^\pm$ production, it is required to activate the `withdamp` option, to enable the Born-zero damping factor. Therefore, the line

```
withdamp 1 !  (default 0, do not use) use Born-zero damping factor
```

should be added (or uncommented) in the corresponding input files.

Other mandatory parameters are those needed to select the final state leptonic species coming from the vector-boson:

```
idvecbos 24     !  PDG code for vector boson to be produced
                !  ( W+:  24 W-:  -24 )
vdecaymode 1    !  code for selected W decay
                !  (1:  electronic; 2:  muonic; 3:  tauonic)
```

Together with the mandatory parameters, the `POWHEG BOX` input facility allows for an easy setting of EW and run parameters, by explicitly adding the relevant lines to the input card. In case one of the following entries is not present in the input card the reported default value is assumed. In any case, these parameters are printed in the output of the program, so their values can be easily tracked down.

```
Wmass 80.398 !  W mass in GeV
Wwidth 2.141 !  W width in GeV
Zmass 91.1876 !  Z mass in GeV
Zwidth 2.4952 !  Z width in GeV
sthw2 0.22264 !  sin**2 theta_w
alphaem 0.0072973 !  em coupling
masswindow_low 10 !  M_W > Wmass - masswindow_low * Wwidth
masswindow_high 10 !  M_W < Wmass + masswindow_high * Wwidth
```

Alternatively, one may directly define the integration range for the W virtuality by using

```
mass_low 200 !  M_W > mass_low
mass_high 10 !  M_W < mass_high
```

If these tokens are defined, they overwite what specified by `masswindow_low` and `masswindow_high`. This may be particularly useful if one wants to restrict the integration region away from the W peak. In this last case, however, no BW importance sampling is adopted.

```
runningscale 0 !  choice for ren and fac scales in Bbar integration
                                  0:  fixed scale M_W
                                  1:  running scale inv mass W

running_width 1 !  (default 0) (0:  fixed width,
                                 1:  running width, (mw*gw) -> (gw*s/mw) )

CKM_Vud 0.9740 !  Entries of CKM mixing matrix
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
```

4. Generation of a sample with W^+ and W^- events

In case the user is interested in the generation of a sample where both W^+ and W^- events appear, a script and a dedicated executable have been included. The script is named `merge_wp_wm.sh` and can be found in the directory `W/testrun-merge`. It can be run in any subfolder of `W` however. Three inputs are mandatory: the first two are the prefixes of the input files used to generate W^+ and W^- events. The third input has to be an integer and correspond to the total number of events that the final *merged* sample will contain. The script has to be run twice, using a positive integer value at the first call and its opposite afterwards. Therefore, for example, to produce a sample of 10000 events, starting from the input files `wp-powheg.input` and `wm-powheg.input`, the invocation lines should be as follows:

```
$ sh merge_wp_wm.sh wp wm 10000
```

and then

```
$ sh merge_wp_wm.sh wp wm -10000
```

Few remarks are needed:

- it is responsability of the user to check that the 2 input files are equal. The `idvecbos` and `vdecaymode` tokens have to be different, obviously.
- the two values of `numevts` are not really used: the program re-calculate the needed values as a function of the W^+ and W^- cross sections and of the total number of events to be generated.
- the final event file is always named `wp_wm_sample-events.lhe`. In the header section it also contains a copy of the two input files used to generate it, for cross-checking purposes

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," [arXiv:1002.2581 [hep-ph]].

[4] S. Alioli, P. Nason, C. Oleari and E. Re, "NLO vector-boson production matched with shower in POWHEG," JHEP **0807**, 060 (2008) [arXiv:0805.4802 [hep-ph]].