Preprint typeset in JHEP style - PAPER VERSION

May 10, 2015
POWHEG-BOX-V2

The $W^{\pm}$ EW NLO & QCD production in the POWHEG BOX-V2 user manual

Luca Barzè
PH-TH Department, CERN, CH-1211 Geneva 23, Switzerland
E-mail: Luca.Barze@cern.ch

Guido Montagna
Dipartimento di Fisica Nucleare e Teorica, Università di Pavia and INFN, Sezione di Pavia, Via A. Bassi 6, 27100 Pavia, Italy
E-mail: Guido.Montagna@pv.infn.it

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

Oreste Nicrosini
INFN, Sezione di Pavia, Via A. Bassi 6, 27100 Pavia, Italy
E-mail: Oreste.Nicrosini@pv.infn.it

Fulvio Piccinini
INFN, Sezione di Pavia, Via A. Bassi 6, 27100 Pavia, Italy
E-mail: Fulvio.Piccinini@pv.infn.it

Abstract: This note documents the use of the package POWHEG-BOX-V2 for $W^{\pm}$ production processes including QCD and ElectroWeak NLO corrections. 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, Electroweak.

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 $pp \to W \to \ell\nu$ including QCD and ElectroWeak (EW) radiative corrections. A detailed description of the implementation can be found in Ref. [4]. 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-V2/W_ew-BMNNP
$ make pwhg_main
```

Then do (for example)

```
$ cd runtest-lhc-8TeV
$ ../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 `PYTHIA6` do

```
$ cd POWHEG-BOX-V2/W_ew-BMNNP
$ make main-PYTHIA-lhef
$ cd runtest-lhc-8TeV
$ ../main-PYTHIA-lhef
```

If you prefer to shower the event with `PYTHIA8` do

```
$ cd POWHEG-BOX-V2/W_ew-BMNNP
$ make main-PYTHIA8-lhef
$ cd runtest-lhc-8TeV
$ ../main-PYTHIA-lhef
```

3. Process specific input parameters

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 11 !  code for selected W decay
              !  (11(-11):  electronic; 13(-13):  muonic; 15(-15):  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. If 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
alphaem 0.00729735254  !  em coupling alpha(0)
gmu     1.16637d-5     !  Fermi constant in GeV^-2
Hmass   120.           !  Higgs mass in GeV
Tmass   172.9          !  Top mass in GeV
Bmass   4.6            !  B quark mass in GeV
Cmass   1.2            !  C quark mass in GeV
Smass   0.15           !  S quark mass in GeV
Umass   0.06983        !  U quark mass in GeV
Dmass   0.06984        !  D quark mass in GeV
Elmass  0.005109989    !  Electron mass in GeV
Mumass  0.105658369    !  Mu mass in GeV
Taumass 1.77699        !  Tau mass in GeV
```

The following parameter limits from below the virtuality of the W boson:

```
mass_low 1          !  W virtuality > mass_low in GeV
```

If absent, it is set to 1 GeV. In order to avoid edge effects, the lower limit `mass_low` should be more inclusive w.r.t. cuts applied at the analysis level. Notice that, if photons are generated, the W virtuality is not necessarily the mass of the charged lepton neutrino system.

```
runningscale 0      !  choice for ren and fac scales in Bbar integration
                                  0:  fixed scale M_W
                                  1:  running scale lνγ) inv mass
                                      γ included with QED FSR
scheme 1!  choice for EW NLO scheme calculation
                                  0:  Alpha(0)
                                  1:  G_μ
CKM_Vud 0.975 !  Entries of CKM mixing matrix
CKM_Vus 0.222
CKM_Vub 1d-10
CKM_Vcd 0.222
CKM_Vcs 0.975
CKM_Vcb 1d-10
CKM_Vtd 1d-10
CKM_Vts 1d-10
CKM_Vtb 1.0
```

The EW radiative corrections can be calculated according to two different schemes: the $\alpha(0)$ scheme, where the input parameters are $\alpha(0)$, M_W and M_Z; the G_μ scheme, where the input parameters are G_μ, M_W and M_Z. The latter one (default in the code) is preferred because it minimizes the EW corrections and the uncertainties due to the light quark masses.
The EW corrections can be switched off by setting

```
no_ew 1 !  default 0
```

and the strong corrections can be switched off by setting

```
no_strong 1 !  default 0
```

This last option is just to check EW corrections at the NLO level (i.e., the Les Houches events do not have much meaning).

The program can be interfaced to both `PYTHIA6` and `PYTHIA8`, by doing

```
make main-PYTHIA-lhef
```

for `PYTHIA6` and

```
make main-PYTHIA8-lhef.
```

for `PYTHIA8`.

In the case of PYTHIA6 one can also optionally switch off photon radiation from PYTHIA6 and use PHOTOS [5] instead. This is done by setting:
`use_photos 1 !  default 0`
in the `powheg.input` file.
The PHOTOS source code is included in the POWHEG BOX.

For photon final state radiation a comment is in order. According to the POWHEG method, the radiation by the shower has to be generated from a starting scale given by the hardest p_{T} tried at the matrix element level (the variable `scalup` written in the event file `pwgevents.lhe`). This is true also in the case that both QCD and QED radiation are present, as detailed in Ref. [4] . Both PYTHIA6 and PYTHIA8 do not use this starting scale for the generation of QED final state radiation from the W. Hence, in order to avoid double counting of QED radiation, a veto algorithm is necessary. The same problem is present also using PHOTOS. This algorithm is provided automatically in the files `main-PYTHIA-lhef.f` and `scalupveto.f`. The same algorithm is implemented for PYTHIA8. In this case, however, the user can optionally adopt the internal algorithm of PYTHIA8, which is switched on by setting:
`py8veto 1 !  default 0`
in the `powheg.input` file.

A general issue is the matching between the NLO calculation and the (QCD and QED) higher order corrections given by the parton shower: due to the different definitions of $p_{\perp}$ in POWHEG and PYTHIA8/PYTHIA6, some double counting or dead zone can arise. In PYTHIA8 the default is to generate all QCD/QED shower emissions up to the kinematical limit and then veto emissions harder than the POWHEG emission, according to the POWHEG $p_{\perp}$ definition. This is done, as default, by means of the provided `class PowhegHooks`. With the provided PYTHIA8 interface the user can optionally choose an alternative scheme, where the shower starting scale is fixed to `scalup` and no veto is performed. This choice can be activated by setting
`veto1 1 !  default 0`
in the `powheg.input` file.

In this case, if the QED higher order radiation is handled by PYTHIA8, also the QED starting scale is set to `scalup` through the `class MyUserHooks`.

Additional flags available for the PYTHIA8 interface are the following (they can be set in the `powheg.input` file):
`noQEDq 1 !  default 0`, which allows to switch off QED radiation from quarks;
`pytune xx !  default 5`, which allows to change the tune;
`nohad 1 !  default 0`, which allows to swith off the hadronization.

As a last comment, in the provided interface to `PYTHIA8` the decay of hadronic resonances which can proceed radiatively has been suppressed. In order to let the resonances decay, the user should open the file `pythia8F77.cc` and comment the relevant lines in `pythia_init`.

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] L. Barzè, G. Montagna, P. Nason, O. Nicrosini and F. Piccinini, “Implementation of electroweak corrections in the POWHEG BOX: single W production,” [arXiv:1202.0465 [hep-ph]]

[5] P. Golonka and Z. Was, Eur. Phys. J. C **45** (2006) 97 [hep-ph/0506026].