Preprint typeset in JHEP style - PAPER VERSION

March 21, 2017

POWHEG-BOX-V2

The $W^{\pm}$ EW NLO & QCD production in the POWHEG BOX-V2 user manual: svn v3375

Mauro Chiesa

INFN, Sezione di Pavia, Via A. Bassi 6, 27100 Pavia, Italy
E-mail: `Mauro.Chiesa@physik.uni-wuerzburg.de`

Homero Martinez

Dipartimento di Fisica, Università di Pavia and INFN, Sezione di Pavia, Via A. Bassi 6, 27100 Pavia, Italy
E-mail: `Homero.Martinez@pv.infn.it`

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]. Major improvements w.r.t. svn version 3370 are described in Ref. [5]. 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

To build the main POWHEG executable, do:

```
$ cd POWHEG-BOX-V2/W_ew-BMNNP
$ make pwhg_main
```

Then do (for example)

```
$ cd runtest-lhc-14Tev-wp
$ ../pwhg_main
```

At the end of the run, the file `pwgevents.lhe` will contain 1000 events for $W^+ \to \mu^+ \nu_\nu$ in the Les Houches format (`.lhe` file).

We provide four executables, called "interfaces", which process the events in the file `pwgevents.lhe` and give them as inputs to parton shower programs, performing the required vetoes and setting the required flags in a manner consistent with the physical accuracy of the input events. PYTHIA is used to perform the QCD shower. The QED shower can be done using PYTHIA or the independent program PHOTOS [6]. The four interfaces are:

- `main-PYTHIA81-lhef` : In this interface, PYTHIA8 (versions 8.1xx) is used to perform the QCD shower, while the QED shower can be done using PHOTOS++ (C++ version of PHOTOS), or PYTHIA8. It is designed to work with versions of PYTHIA8 up to 8.186. It has been tested with PYTHIA version 8.185 and PHOTOS++ version 3.56. The output of the shower can be analyzed looking at the histograms in the output `.top` file. Optionally, the events after the shower can be saved in another `.lhe` file (see section 3).

  Even if you do not use PHOTOS++, it has to be linked. The minimal procedure is to use the version 3.56 included in the W_ew-BMNNP package, doing:

  ```
  $ cd PHOTOS
  $./configure --without-hepmc
  $ make
  ```

  The flag PHOTOSCC_LOCATION must be set in the `Makefile` to the path of installation of PHOTOS++. This path can be the folder W_ew-BMNNP/PHOTOS if the user wants to use the included version of PHOTOS++, otherwise it should be set to the folder of an external installation.

  PYTHIA8 has to be downloaded and compiled by the user. The script `pythia8-config` should configure automatically the path to the PYTHIA8 installation in the `Makefile`, if this is not the case, the user must set manually the PYTHIA8LOCATION flag.

  Once PHOTOS++ and PYTHIA8 are compiled and the flags are set properly in the `Makefile`, the interface can be compiled, doing:

  ```
  $ cd POWHEG-BOX-V2/W_ew-BMNNP
  $ make main-PYTHIA81-lhef
  ```

  Before running, the path to PHOTOS++ libraries needs to be added to the list of dynamically linked libraries, and a variable pointing to the path of PYTHIA8 particle data (`.xml`) files) needs to be set. In order to do this, the script `setlibrarypaths.sh` must be edited to point to the correct paths, and then executed in the current shell, doing:

```
$ source setlibrarypaths.sh
```

To run the interface, do:

```
$ cd runtest-lhc-14TeV-wp
$ ../main-PYTHIA81-lhef
```

- `main-PYTHIA82-lhef` : Similar interface to `main-PYTHIA81-lhef`, but designed to work with newer versions of `PYTHIA` (versions 8.2xx). It has been tested with version 8.223. The compilation / linking procedure is similar to the one described in the previous item.
- `main-PYTHIA6-lhef` : Interface to `PYTHIA6` and `PHOTOS` (both codes written in Fortran). The source codes of `PYTHIA6` and `PHOTOS` are included in the `POWHEG BOX` and in the `W_ew-BMNNP` folder, respectively, so no flag or linking to external libraries need to be done. To compile and run, simply do:

```
$ cd POWHEG-BOX-V2/W_ew-BMNNP
$ make main-PYTHIA6-lhef
$ cd runtest-lhc-14TeV
$ ../main-PYTHIA6-lhef
```

- `main-PHOTOS-lhef` : Interface to `PHOTOS++`, it processes the `POWHEG` generated events, calls the QED final state shower implemented by `PHOTOS++` and generates a new event file in LHE format, that can be then interfaced to a QCD shower program, where QED radiation must be switched off to avoid double counting. The compilation and linking requires the `PHOTOS` related steps described in item (1).

The customization of the histograms and cuts provided in the output `.top` file (for interfaces `main-PYTHIA81-lhef`, `main-PYTHIA82-lhef` and `main-PYTHIA6-lhef`), can be done modifying the code in `pwhg_analysis.f` (`analysis` subroutine). In addition, C++ based analysis can be done accessing the events from the files `pythia81F77.cc`, `pythia82F77.cc` (function `pythia_next_`) or in the file `photosCCF.cc` (function `photos_process_`), for the interfaces `main-PYTHIA81-lhef`, `main-PYTHIA82-lhef` and `main-PHOTOS-lhef`, respectively.

3. Process specific input parameters

All the parameters and flags are set in the input card file `powheg.input`. The 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)
```

In addition to the mandatory parameters, the `POWHEG BOX` input 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 lν(γ) 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).

3.1 Flags used by the shower interfaces

The flags starting with `SI_` configure the behavior of the shower interfaces. The following flags are used by the interfaces `main-PYTHIA81-lhef` and `main-PYTHIA82-lhef`:

`SI_inputfile`: Configure the input file for the shower interfaces (default: `./pwgevents.lhe`).

`SI_maxshowerevents`: Number of events to read from the input file (default: all events).

`SI_pythiamatching`, explained below.

`SI_pytune`: Set the `PYTHIA` tune used in the QCD shower.

`SI_dopythiaqed`: Turn ON / OFF `PYTHIA` QED shower (default OFF).

`SI_use_photos`: Turn ON / OFF the final state QED shower by `PHOTOS` (default OFF). If it is ON, it automatically sets off the final state QED radiation from `PYTHIA8`.

`SI_kt2minqed`: Set value of photos low energy cut off (default is 10^{-6}).

`SI_usepy8veto`, explained below.

SI_nohad: Allows to switch OFF the hadronization (the hadronization is ON by default).

SI_savelhe: Turn ON / OFF the production of an output LHE file (default OFF).

SI_savehistos: Turn ON / OFF the production of histograms in .top file (default OFF).

SI_no_tworadevents, explained below.

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. Traditionally, this scale is written in the variable scalup in the tt LHE event file. However, for the DY process, when both QCD and EW NLO corrections are present (Ref. [4]), it is necessary to keep track in the POWHEG events of two scales, one for initial state radiation (scalup-isr) and one for final state radiation (scalup-fsr). These two scales are used as starting points for the IS and FS showers, respectively. PYTHIA8 and PHOTOS++ do not use scalup-fsr 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. This veto is activated unless the flag no_ew is activated in powheg.input.

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, some double counting or dead zone can arise. By default (flag SI_pythiamatching equal to 2), the interface generates all QCD/QED shower emissions up to the kinematical limit and then veto emissions harder than the POWHEG scales (scalup-isr and scalup-fsr), computed according to the POWHEG $p_\perp$ definition. The user can optionally choose an alternative scheme, where the shower starting scales are fixed and no veto's are performed. This choice can be activated by setting the flag SI_pythiamatching equal to 1.

When FS QED radiation is implement by PHOTOS++, a Fortran-based function is used to veto QED emissions and perform the correct matching (implemented in the file main-PYTHIA-8.f). If FS QED radiation is done by PYTHIA8, the matching can be done using Fortran-based algorithms coded in the files main-PYTHIA-8.f and scalupveto.f (flag SI_usepy8veto equal to 0), or alternatively, use the matching algorithms provided by PYTHIA8 (flag SI_usepy8veto equal to 1). Those algorithms use the UserHooks functions, and their behavior depend on the value of the flag SI_pythiamatching.

In the case the interface is used to read events where the two scales scalup-isr and scalup-fsr are not present, and scalup is to be used as starting scale for the showers, the flag SI_no_tworadevents should be set to 1.

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 pythia81F77.cc (or pythia82F77.cc) and comment the relevant lines in the function pythia_init.

The interface main-PYTHIA6-lhef can be customized using the following flags: SI_inputfile, SI_maxshowerevents, SI_pytune, SI_use_photos, SI_kt2minqed, SI_savehistos, SI_no_tworadevents, read also from the powheg.input file. The other flags have no effect. Notice that in this case, the QED radiation from PYTHIA is ON by default.

The interface `main-PHOTOS-lhef` uses only the relevant flags, namely: `SI_inputfile`, `SI_maxshowerevents`, `SI_kt2minqed`.

For further customization of the settings used by the shower interfaces, beyond the flags available in `powheg.input`, the user can modify the following source code files:

- Interface `main-PYTHIA81-lhef`: Settings in file `pythia81F77.cc`.
- Interface `main-PYTHIA82-lhef`: Settings in file `pythia82F77.cc`.
- Interface `main-PYTHIA6-lhef`: Settings in file `setup-PYTHIA6-lhef.f`
- Interface `main-PHOTOS-lhef`: Settings in file `photosCCF.cc`.

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 responsibility 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] C.M. Carloni Calame, M. Chiesa, H. Martinez, G. Montagna, O. Nicrosini, F. Piccinini, and A. Vicini, [arXiv:1612.02841[hep-ph]].

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