Preprint typeset in JHEP style - PAPER VERSION

July 13, 2012
POWHEG BOX 1.0

The POWHEG BOX user manual: Z/γ^* production

Simone Alioli

LBNL & UC Berkeley
1 Cyclotron Road, MS50A 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 Z/γ^* 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 $pp \to Z/\gamma^* \to \ell^+\ell^-$. 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/Z 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 Z/γ^* implementation.

2. Generation of events

Build the executable

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

Then do (for example)

```
$ cd testrun-z-lhc
$ ../pwhg_main
```

At the end of the run, the file pwgevents.lhe will contain 100000 events for $Z/\gamma^* \to e^+\nu_e$

in the Les Houches format. In order to shower them with PYTHIA do

```
$ cd POWHEG-BOX/Z
$ make main-PYTHIA-lhef
$ cd testrun-z-lhc
$ ../main-PYTHIA-lhef
```

3. Process specific input parameters

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

```
vdecaymode 1    !  code for selected Z decay
                !  (1:  e+ e-; 2:  mu+ mu-; 3:  tau+ tau- ...)
```

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.

```
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
Wmass 80.398 !  W mass in GeV
Wwidth 2.141 !  W width in GeV

masswindow_low 10 !  M_Z > Zmass - masswindow_low * Zwidth
masswindow_high 10 !  M_Z < Zmass + masswindow_high * Zwidth
```

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

```
mass_low 160 !  M_Z > mass_low
mass_high 30 !  M_Z < 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 Z 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_Z
                                1:  running scale inv mass Z
running_width 1 !  (default 0) (0:  fixed width,
                               1:  running width, (mz*gz) -> (gz*s/mz) )
```

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]].