TUTORIAL: Polarised cross sections for vector boson production with Sherpa

Mareen Hoppe, Erik Bachmann, Frank Siegert

Institute for Nuclear and Particle Physics, TUD Dresden University of Technology, D–01062 Dresden, Germany

Welcome to the Sherpa polarisation tutorial! In this tutorial you will learn how to use Sherpa to produce fully-realistic polarised events for processes with intermediate vector bosons. This includes the incorporation of higher order QCD effects up to nLO QCD matched to parton shower and/or via multi-jet merging. We will use fully-leptonic inclusive W^+Z production as our testbed. In order to analyse our simulation results, we will use Rivet, i.e. this tutorial also contains a short introduction into the usage of Rivet with a given analysis.

1 Sherpa and Sherpa's polarisation framework

Sherpa is one of the three fully-realistic, general-purpose event generators on the market. It can perform all event generation steps of a typical collision event, i.e. including hard scattering, parton showering, hadronisation, hadron decays, QED radiation and the underlying event. Sherpa's polarisation framework is part of the hard scattering process simulation and can be combined with all the event generation steps that follow afterwards. If you are interested in any details about the other parts of Sherpa, we can refer you to the Sherpa manual (`https://sherpa-team.gitlab.io/sherpa/v3.0.0/index.html`) and paper (`https://arxiv.org/abs/1905.09127`).

Sherpa's polarisation framework is based on an unpolarised simulation in an extended narrow width approximation where spin correlations are preserved and off-shell effects are taken into account by a mass smearing of the on-shell vector bosons according to the Breit-Wigner distribution. Amplitudes in narrow-width approximation can be factorized into the production of intermediate resonances (here: massive vector bosons) and their corresponding decays. In Sherpa, those two parts are handled separately. For the spin correlation algorithm, amplitude tensors for the vector boson production process and decay matrices for the decay part are calculated. Their multiplication, after a transformation into the desired spin basis, allows for a calculation of polarisation fractions on top of the unpolarised simulation. Multiplied with the event cross section, resulting polarised cross sections are output as additional event weights. With this methodology, Sherpa can provide all polarisation combinations in one simulation run including the interference between different polarisations. In addition, different polarisation definitions can be investigated in a single simulation run. Results for all possible polarisation combinations including transverse polarisation components and a summed interference are output by default.

The polarisation weights are named according to the scheme given below. The ordering of the particles follows Sherpa's internal particle ordering; for W and Z bosons, the Z boson comes first:

```
PolWeight_ReferenceSystem.particle1.polarization1_particle2.polarization2...
```

In this tutorial, we simulate polarised cross sections in the laboratory and in the WZ center of mass frame, which are abbreviated by `Lab` and `COM`. Hence, e.g. the $W_0^+Z_0$ contribution in the laboratory frame is labeled

PolWeight_Lab.Z.0_W+.0. For W^+Z production, typically $W_0^+Z_0$, $W_0^+Z_T$, $W_T^+Z_0$ and $W_T^+Z_T$ as well as the summed interference are investigated. The summed interference corresponding to the transverse polarisation components is labeled as `PolWeight_ReferenceSytem.coint`[1].

2 Installation

In this tutorial, we will use the SHERPA 3.0.0 release together with OPENLOOPS, LHAPDF and RIVET. In order to spare you the installation time, we provide two different ways to get SHERPA for this tutorial. If you have access to lxplus, you can directly use the SHERPA installation from `/cvmfs`. Alternatively, you can use a Docker image we have built for this tutorial. The required PDF set and OPENLOOPS library are already installed in both cases.

2.1 lxplus

In order to setup the SHERPA environment on lxplus and to make SHERPA available, the following script has to be sourced:

```
source /cvmfs/sft.cern.ch/lcg/releases/LCG_106_ATLAS_5/MCGenerators/sherpa/\
3.0.0/x86_64-el9-gcc13-opt/sherpaenv-genser.sh
```

LHAPDF and OPENLOOPS become available by:

```
export LHAPDF_DATA_PATH=\
/cvmfs/sft.cern.ch/lcg/external/lhapdfsets/current/:$(lhapdf-config --datadir):

export OL_PREFIX=/cvmfs/sft.cern.ch/lcg/releases/LCG_106_ATLAS_5/MCGenerators/\
openloops/2.1.2/x86_64-el9-gcc13-opt
```

2.2 Docker

To setup the Docker Engine on your machine, follow the Docker Engine installation instructions given at `https://docs.docker.com/engine/install`. On Linux, also follow the post-installation steps detailed at `https://docs.docker.com/engine/install/linux-postinstall/#manage-docker-as-a-non-root-user` to be able to run Docker as non-root.
With Docker set up, you can pull the Docker image we have prepared for this tutorial:

```
docker pull ebachman/sherpa-tutorial
```

This will download and store the Docker image in `/var/lib/docker` on your Linux or Mac system.
To run the Docker container interactively, change to a directory of your liking and execute:

```
docker run -it --rm -v $PWD:/host ebachman/sherpa-tutorial
```

This give you an interactive, terminal container session, and links `/host` inside the container to your current working directory on the host machine for easier file transfer in and out of the container. Keep in mind that once the session is terminated, Docker will delete the container (as instructed via the `--rm` option), therefore you should move all files you would like to keep to `/host` first. The Docker image will persist on your system until deleted manually (e.g. by executing `docker system prune`), so only the above `docker run [...]` command is needed to start the Docker container the next time.
Inside the container, you can work with SHERPA and RIVET exactly as if they were natively installed on your machine.

[1]In case you are more interested in the separate left and right polarisation contributions, `PolWeight_ReferenceSytem.int` marks the right interference contribution.

2.3 Basic SHERPA usage

Both on lxplus and in the Docker container, SHERPA can be executed simply by typing `Sherpa`. With `-e <number>` you can specify how many events you would like to get. For the processes we'll look at today, we have compiled a list of time estimates for a reasonable number of events for both lxplus as well as Docker on an average laptop in Appendix 4.3.
If you use SHERPA's built-in matrix element generator AMEGIC, the run will terminate after the process initialisation step and you need to compile the AMEGIC libraries by following the instructions on the screen. After that, you can execute the above command a second time to start generating events.
SHERPA also allows for MPI usage to accelerate integration and event generation. For this tutorial, this is currently only working within the docker container. If you would like to use MPI (which would be strongly recommend to reduce waiting time), a dedicated process initialisation step is needed beforehand by calling

```
Sherpa -I
```

(and compiling the AMEGIC libraries if prompted). After that, you can start your MPI event generation by calling

```
mpirun -n <cores> Sherpa -e <number>
```

Finally, we need to tell SHERPA what we actually want it to do, which is done using a run card file. In the next section, you will learn what the basic structure of SHERPA run cards looks like. Note that, by default, SHERPA will look for a "Sherpa.yaml" file in the current working directory. For different names or directories, use the `-f` command line option to specify the path to the run card.
In addition, it is recommended to run each individual simulation in a separate directory since otherwise SHERPA will overwrite previous results.

2.4 Some information about RIVET

We will use RIVET 4, a widespread tool to analyse HepMC events, to analyse our simulated events. This is steered by an analysis file written in C++. It calculates event properties, implements kinematic cuts using projections, and book, fill and output histograms in the YODA format. Each event is passed to the analysis and operated on by the projections. The result of the projections operating on the event determines whether the event is accepted and what is plotted.
Since this is not a dedicated RIVET tutorial, we provide an analysis named "MC_WZ_NLOQCD" for the purpose of this tutorial. It attempts to mimic the phase space of the recent polarised WZ production ATLAS analysis (`https://arxiv.org/abs/2211.09435`). When running on lxplus, you need to download the analysis files from `https://cernbox.cern.ch/s/NLbz8VUhmwvDPEs`; in the Docker container they are already provided at `/home/sherpa/Rivet`.
The analysis needs to be compiled before you can use it by executing:

```
rivet-build MC_WZ_NLOQCD.cc
```

In order to make it available for RIVET, you have to export the path where the analysis is saved:

```
export RIVET_ANALYSIS_PATH=<analysis_path>
```

More details on what an analysis consists of, how you can build your own analysis, and a lot of inspiration through a huge variety of standard analyses can be found here: `https://gitlab.com/hepcedar/rivet/-/blob/release-4-0-x/README.md?ref_type=heads` and `https://rivet.hepforge.org/analyses.html`.

Plotting instructions

RIVET also comes with its own Python plotting scripts which take YODA files as input and generate histogram plots out of it. We will also use this plotting tool within this tutorial. Generally, you can produce plots by executing

```
rivet-mkhtml <Analysis_File>.yoda
```

However, for polarised simulations this would only plot the unpolarised cross sections. Polarised cross sections, generated as additional event weights in SHERPA, are understood as variations weights by RIVET and can all be plotted by adding `--show-variations` to the plotting command.
Besides that, you can also plot only some of the polarisation weights by declaring them as the new central value. This can also be done multiple times in order to plot several polarisation weights in one plot:

```
rivet-mkhtml <Analysis_File>.yoda:"DefaultWeight=PolWeight_COM.Z.0_W+.0" \
<Analysis_File>.yoda:"DefaultWeight=PolWeight_COM.Z.T_W+.T"
```

This command would plot $\mathrm{W}^+_0\mathrm{Z}_0$ as well as $\mathrm{W}^+_\mathrm{T}\mathrm{Z}_\mathrm{T}$. An overview over all simulated event weights and their corresponding names is given by the results table at the end of SHERPA's log output.
It is also possible to hand over further plotting instructions during the call of the plotting script, most notably for this tutorial the title of the different results. It supports LaTeX:

```
rivet-mkhtml <Analysis_File>.yoda:"DefaultWeight=PolWeight_COM.Z.0_W+.0:\
Title=$\mathrm{W}^+_0\mathrm{Z}_0$" \
<Analysis_File>.yoda:"DefaultWeight=PolWeight_COM.Z.T_W+.T:\
Title=$\mathrm{W}^+_\mathrm{T}\mathrm{Z}_\mathrm{T}$"
```

Together with the RIVET analysis, we provide a simple plot instruction file (`MC_WZ_NLOQCD.plot`) which will be read by the plotting routine if you have exported the analysis path as described above. If you would like to improve the appearance of specific histograms, you can add additional plotting commands to this plotting file and rerun the plotting command. In order to change properties of all plots (like line colors, line styles...), you can add more plotting instructions to the plotting commands itself in the same manner as done for the titles. You can find the available options in the dedicated sections of the RIVET manual: `https://gitlab.com/hepcedar/rivet/-/blob/release-4-0-x/doc/tutorials/plotting.md` and `https://gitlab.com/hepcedar/rivet/-/blob/release-4-0-x/doc/tutorials/makeplots.md`.
All resulting histograms can be viewed by opening the generated `index.html` file in your favourite browser.

3 First SHERPA run - off-shell calculation at leading order

SHERPA simulations are steered by one run card written in the YAML format, typically called `Sherpa.yaml`. For a comprehensive list of possible settings and steering options you can consult the manual `https://sherpa-team.gitlab.io/sherpa/master/index.html`. Since SHERPA's main building objective are fully-realistic simulations including all event generation steps of a typical collision event, all event generation modes are switched on and configured per default. Hence, only a few settings need to be set explicitly in the run card in order to start a simple simulation run. They consist of:

Beam setup We would like to use proton beams with the LHC run 3 center of mass energy of 13.6 TeV.

```
BEAMS: 2212
BEAM_ENERGIES: 6800
```

Factorisation scale will be set to $\mu_f = 0.5 * m_Z * m_W$ with m_Z, m_W being the pole masses of the vector bosons. Note that SHERPA expects the square of the desired scale choice.

```
MEPS:
  CORE_SCALE: VAR{0.25*sqr(80.379+91.1876)}
```

Processes and Order For defining the process(es) that should be simulated, particles are identified by their PDG codes, see `https://pdg.lbl.gov/2007/reviews/montecarlorpp.pdf`. In addition, SHERPA also allows to define particle containers and is shipped with some predefined ones. The container 93 is one of those predefined ones and contains all massless quarks and the gluon. It is also called the "jet" container. Each process definition in the `PROCESSES` section starts with a "-". Below the process definition, simulation settings important for the specific process are specified. For this simple case, this just consists of the QCD and EW order in powers of α.

```
PROCESSES:
- 93 93 -> -11 12 -13 13:
    Order: {QCD: 0, EW: 4}
```

Selection requirements The `SELECTORS` section allows for defining the phase space in which the process(es) should be simulated. For simplicity, we will only specify the selection criterion necessary to render the process finite by avoiding the divergence due to on-shell intermediate photons when $m_{ll} \to 0$. Note, that selection criteria in simulations with parton shower enabled should always be chosen significantly looser than in the final desired phase space since selection criteria are only applied during the matrix element calculation before the parton shower is turned on.

```
SELECTORS:
- [Mass, 13, -13, 61, 121]
```

In addition to those fundamental settings, we also adjust the way how a real electroweak coupling is constructed in SHERPA's default EW_scheme (Gmu) when the complex mass scheme is used, in order to be more comparable to later polarised calculations which will not be done in the complex mass scheme.

```
GMU_CMS_AQED_CONVENTION: 4
```

In order to accelerate the event generation, since we do not have any cluster available for this tutorial, we disable fragmentation and multiple interaction simulation for now:

```
FRAGMENTATION: None
MI_HANDLER: None
```

and generate weighted events:

```
EVENT_GENERATION_MODE: Weighted
```

Now, our first simulation is basically ready to go. The only missing piece we still need to specify is how we expect SHERPA to write out its results. This is either possible in one of the supported event formats (see the manual for details) or by interfacing RIVET to directly receive histograms instead of writing out large amounts of event data. We will go with the latter option for now:

```
ANALYSIS: Rivet
RIVET:
  --analyses:
     - MC_WZ_NLOQCD
```

Now we are ready to generate our first SHERPA events as described in Section 2.3. Can you still wait to see the first results of your SHERPA simulation? ;) Then move on to Section 4... If not, follow the plotting instructions in Section 2.4.

4 Simulation of polarised events

4.1 Starting point: LO+PS

The definition of polarisation for intermediate particles requires the use of an on-shell approximation. SHERPA uses the narrow-width approximation (NWA). In order to do calculations in the NWA, we need to adjust our previous run card.
In the NWA, the propagator of intermediate heavy resonances, in our case the vector bosons, is essentially replaced by a delta function. This allows us to factorise the process into production and decay of the intermediate particles. This separation is also found in the run card. The `PROCESSES` section now only contains the vector boson production, while their decay is now steered by a separate section. Do you know how you need to adjust the `PROCESSES` part from the previous section? If not, you can check out the next section for help.
The decay section is called `HARD_DECAYS`. In this section, one is able to specify the channels in which the vector bosons are allowed to decay. In our case, this means:

```
HARD_DECAYS:
  Enabled: true
  Channels:
    24,12,-11: {Status: 2}
    23,13,-13: {Status: 2}
```

Note, that, if not explicitly disabled, SHERPA will also take spin correlations between production and decay via the spin correlation algorithm (`https://arxiv.org/pdf/hep-ph/0110108`) and off-shell effects via a mass smearing according to a Breit-Wigner distribution into account.
In addition to the separation of production and decay, some further settings need to be added to the run card for theoretical or technical reasons:

- Widths of the vector bosons need to be set to zero since they now are considered as final state particles.

  ```
  PARTICLE_DATA:
    24: {Width: 0}
    23: {Width: 0}
  ```

- The complex mass scheme needs to be disabled:

  ```
  WIDTH_SCHEME: Fixed
  ```

  With that, `GMU_CMS_AQED_CONVENTION: 4` is not necessary anymore.

Now that we have configured SHERPA's on-shell configuration, we can finally turn on the polarisation calculation. This can be done by adding the following subsection to the `HARD_PROCESS` section:

```
 Pol_Cross_Section:
     Enabled: true
     Reference_System: [Lab, COM]
```

The `Reference_System` setting specifies two different polarisation definitions. By default, all polarisation calculations are done in the helicity basis where the spin axis is defined with respect to the particle's momentum. The reference system determines the frame from which the momentum is taken to define the spin axis.
In addition, we need to set the gauge of the spinors in COMIX properly in order to receive the usual representation of the polarisation vectors in the helicity basis as you can see e.g. in `https://arxiv.org/abs/2310.14803`, eq. (2.4):

```
COMIX_DEFAULT_GAUGE: 0
```

The selection requirement from the previous section is not necessary anymore as we now have vector bosons in the `PROCESSES` final state on which the selection criteria are applied.

Analyse tasks

Let's plot our first results. You can see, that we can get the cross sections of all polarisation combinations in different polarisation definitions in one simulation run. :)

- Do you see any differences regarding different polarisation definitions? Which reference system would you prefer if you would be mostly interested in the longitudinal polarisation?
- Can you see an additional contribution which is not one of the pure polarisation states? Is it large? Do you know where it is coming from?
- Check the differences between the first calculation and the unpolarised result from the second calculation. Can you imagine which effects could cause differences here? Are they small in our case? If yes, than let's move on ... :)

4.2 Let's do it more accurate: nLO QCD + PS

After a (hopefully) successful LO simulation we now aim for incorporating NLO QCD corrections into our simulation. In order to add them, the `PROCESSES` section needs to be extended to:

```
PROCESSES:
- 93 93 -> 24 23:
    Order: {QCD: 0, EW: 2}
    NLO_Mode: MC@NLO
    NLO_Order: {QCD: 1, EW: 0}
    ME_Generator: Amegic
    RS_ME_Generator: Comix
    Loop_Generator: OpenLoops
```

i.e. for NLO calculations, also the desired NLO order and the NLO_Mode needs to be specified, where Fixed_order and MC@NLO are available for the latter. Since we are aiming for fully-realistic events, MC@NLO is the mode to use. Furthermore, it needs to be specified which generator should be used for the generation of Born (`ME_GENERATOR`), virtual and real+subtraction (RS) terms.

Analyse tasks

- How large is the influence due to the newly included NLO corrections?
- Are there differences regarding different polarisations or polarisation definitions?

4.3 Bonus: Multi-leg merged simulations

In particle-level simulations there are two different ways to incorporate higher-order corrections into your calculations. The first possibility we explored in the previous section where "exact" (beside the approximation necessary to calculate polarisation fractions in Sherpa) fixed-order calculations are matched to parton shower via the MC@NLO algorithm. The other possibility consists of computing the process of interest with different jet multiplicities in the final state at LO (or NLO) and merge them together by a dedicated merging algorithm in order to avoid double counting. Those merging algorithms require the specification of a merging scale - if the energy of the generated jets is below this threshold, they are handled by the parton shower, above by the hard matrix element. We will start with polarised LO merging with one additional jet and merging scale at 20 GeV i.e. the `PROCESSES` section in the LO run card needs to be replaced by:

```
PROCESSES:
- 93 93 -> 24 23 93{1}:
    Order: {QCD: 0, EW: 2}
    CKKW: 20
```

Analyse tasks

- Do you face any differences between nLO QCD and the multi-leg merged calculations? Do you know where these differences come from?
- How does he predictions change in dependency of the merging scale? Can you find out what would be a good choice?
- Do you get additional contributions if you take more jets into account or switch to nLO?

Note that each adjustment of the merging scale or the number of jets / accuracy in general requires a new integration, hence consider to always start in a fresh directory.

Appendix: Event generation times

setup	lxplus single core		docker, four cores	
	No. events	time	No. events	time
off-shell LO	200000	6 min.	400000	4 min.
LO polarised	100000	10 min.	100000	5 min.
nLO polarised	100000	6+12 min.	100000	2+6 min.
LO+1j merged	100000	3+20 min	100000	7 min.

Table 4.1: Event generation times; if two times are given, the first one corresponds to the time needed for process integration and the second one denotes the actual event generation time; if only one time is given, the time spent on integration is negligible compared to the event generation time.