Sherpa manual

(Sherpa version 1.1.2)

Tanju Gleisberg[1], Stefan Höche[2], Frank Krauss[2],
Marek Schönherr[3], Steffen Schumann[4], Frank Siegert[2], Jan Winter[5]

[1]Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309, USA
[2]IPPP Durham, Durham University, Durham DH1 3LE, United Kingdom
[3]Institute of Nuclear and Particle Physics, TU Dresden, D-01062 Dresden, Germany
[4]The University of Edinburgh, Edinburgh EH9 3JZ, Scotland
[5]Fermi National Accelerator Laboratory, Batavia, IL 60510, USA

email: info@sherpa-mc.de

March 25, 2009

Abstract

Sherpa [1] is a Monte Carlo event generator for the *Simulation of High-Energy Reactions of PArticles* in lepton–lepton, lepton–photon, photon–photon and hadron–hadron collisions. This document provides information, which helps users understand and apply Sherpa for their physics studies. The framework is briefly introduced, instructions are given for the installation and running of the whole package, and the procedure of process initialization is described. The various options and parameters specifying the program are compiled, and their meaning is explained. Application examples are listed illustrating Sherpa's capabilities for the production and evolution of hard interactions at high-energy colliders.

Contents

0 Introduction

SHERPA [1] is a Monte Carlo event generator that provides complete hadronic final states in simulations of high-energy particle collisions. The produced events are qualified for inputting them into detector simulations used by the various experiments. In the older versions, series 1.0.x, some physics aspects such as fragmentation and hadron decays have been treated by interfacing other codes, notably Fortran versions of PYTHIA [2,3]. By migrating from these older, proof-of-concept versions to the first physics version 1.1, SHERPA does not rely anymore on such interfaces. It can thus be considered as a truly independent, full-fledged Monte Carlo generator. [1] The entire code has been written in C++, like its competitors HERWIG++ [4] and PYTHIA8 [5].
SHERPA simulations can be achieved for the following types of collisions:

- for lepton–lepton collisions, e.g. as explored by the CERN LEP experiments for colliding electrons and positrons.
- for lepton–photon collisions,
- for photon–photon collisions with both photons either resolved or unresolved, and,
- in particular, for hadronic interactions as studied at the Fermilab Tevatron or CERN LHC, where protons and antiprotons or only protons are brought to collisions, respectively.

The list of physics processes that come with SHERPA covers particle production at tree level in the Standard Model and in models beyond the Standard Model: The complete set of Feynman rules for its Minimal Supersymmetric extension according to [6,7] has been implemented, including general mixing matrices for inter-generational squark and slepton mixing. Among other interaction models the ADD model of Large Extra Dimensions has been made available, too [8]. Furthermore, anomalous gauge couplings [9], a model with an extended Higgs sector [10], and a version of the Two-Higgs Doublet Model are available. The SHERPA program owes this versatility to the inbuilt matrix-element and phase-space generators AMEGIC++ and PHASIC++ [11], which automatically calculate and integrate tree-level amplitudes for the implemented models. This feature enables SHERPA to be used as a cross-section integrator and parton-level event generator as well. This has also been extensively tested, see e.g. [12,13].

As a second key feature of SHERPA the program provides an implementation of the merging approach of [14–16], known by now as CKKW merging. This finally yields improved descriptions of multijet production processes, which copiously appear at the Tevatron and the LHC, see for instance [17–19]. This approach has been compared in great detail in [20] with other approaches, such as the MLM merging prescription [21] as implemented in ALPGEN [22], MADEVENT [23,24], or HELAC [25,26] and the CKKW-L prescription [27,28] of ARIADNE [29].
It should be stressed that some physics cases are still outside the applicability of SHERPA. These include, among others,

- DIS-like configurations. SHERPA can calculate cross sections for DIS and similar processes; however, the currently implemented default parton shower of SHERPA, APACIC++ [30,31], cannot deal with only one space-like evolution to be done. In fact, this kind of virtuality-ordered shower [32,33] is probably not very well-suited for the description of DIS. The authors anticipate, however, that with the full inclusion of two new shower approaches into the SHERPA framework [34,35], this shortcoming will be cured.
- Soft and diffractive hadron collisions. SHERPA up to now has no means to simulate reggeon exchange processes etc.. Therefore, soft events are not accounted for. This is also true for pile-up events, which still are out of SHERPA's current scope.

[1] Up to version 1.0.11, the hadronization and most of the hadron decays were handled by interfacing to the corresponding PYTHIA routines; in the new versions these interface have been kept for backward compatibility and can still be used as an option.

This manual contains all information necessary to get started with SHERPA as quickly as possible. By reading it, users should be enabled to setup the program according to their needs for studying various physics aspects. Therefore, all switches plus options that have been provided are listed. It is explained how to use them, how SHERPA can be run in different modes and how the results and output of SHERPA can be interpreted. For external code that can be linked, corresponding references are given and users are encouraged to cite them accordingly.
On the other hand, the physics of SHERPA and its underlying structure and coding principles are not detailed in this manual. For this, readers are encouraged to refer to original work of the authors. Also, whenever justified, SHERPA users are kindly asked to cite SHERPA's original publication [1]. Moreover the authors strongly recommend the study of the manuals and/or many excellent publications on different aspects of event generation and physics at collider experiments of the other event generator authors.
This manual is organized as follows: in Sec. 1 the modular structure intrinsic to SHERPA is introduced. Sec. 2 contains information about and instructions for the installation of the package. This is followed by the description in Sec. 3 of the steps that are needed to run SHERPA in the different modes and generate events. All parameters and options are discussed in Sec. 4. Subsequently, comments are given concerning the implementation of the CKKW merging of matrix elements and parton showers, see Sec. 5. SHERPA events can be stored in different formats. This and how to interprete them is detailed in the next section, Sec. 6. Finally, in Sec. 7, some examples of SHERPA applications are presented.

Two last comments are added before the discussion is continued.
It should be stressed that the construction of a Monte Carlo program requires a number of implicit assumptions, approximations and simplifications of complicated situations. Potential bugs and other shortcomings of the authors may also be included. The results of event generators, independent of their quality, should therefore always be verified and cross-checked with results obtained by the programs of other authors.
The authors welcome suggestions, bug reports, etc. and endeavour to answer any question as quickly as possible. Furthermore they try to provide user support as far as possible. In order to improve SHERPA, users are kindly ask to send their comments to the author's common mail account,

`info@sherpa-mc.de` .

Please do not hesitate to do so. For bug fixings, updates, etc., please refer to the homepage,

`http://www.sherpa-mc.de` .

1 Basic structure of SHERPA

The construction of the SHERPA program has been pursued in a modular way. It fully reflects the paradigm of Monte Carlo event generation of factorizing the simulation into well defined phases. Accordingly, each module encapsulates a different aspect of event generation for high-energy particle reactions. It resides within its own namespace and is located in its own subdirectory of the same name. The main module called `SHERPA` steers the interplay of all modules – or phases – and the actual generation of the events. Modules labelled by "++" were developed and published (or will be published) separately, correspondingly self-contained documentations exist. [2] Altogether, the following modules are currently distributed with the SHERPA framework:

- `ATOOLS`

  This is the toolbox for all other modules. Since the SHERPA framework does not rely on CLHEP etc., the `ATOOLS` contain classes with mathematical tools like vectors and matrices, organization

[2] Modules that have no documentation (yet) are AMISIC++, AHADIC++, HADRONS++ and PHOTONS++; the documentation for version 1.0 of AMEGIC++ [11] dates back to 2001 and is quite outdated by now. Documentation for some of these modules may become available in future.

tools such as read-in or write-out devices, and physics tools like particle data or classes for the event record.

- `HELICITIES`

  In this module some general methods for the evaluation of helicity amplitudes have been accumulated. They are used in AMEGIC++, the `EXTRA_XS` module, and the new matrix-element generator COMIX that will be made available soon. This module also contains helicity amplitudes for some generic matrix elements, that are, e.g., used by HADRONS++.

- `BEAM`

  This module manages the treatment of the initial beam spectra for different colliders. Currently, three options have been made available for the beams.

  (a) They can either be monochromatic, and therefore require no extra treatment.

  (b) Recently, photon emission in the Equivalent Photon Approximation (EPA) for protons has been added. [3]

  (c) For the case of an electron collider, laser backscattering off the electrons is supported. This leads to photonic initial states. [4]

- `PDF`

  The `PDF` module provides access to various parton density functions (PDFs) for the proton [37–39] and the photon [40]. In addition, it hosts an interface to the LHAPDF package [41], which makes a full wealth of proton PDFs available. An (analytical) electron structure function is supplied in the `PDF` module as well.

- `MODEL`

  This module is responsible for setting up the physics model for a simulation run. It comprises the initialization of particle properties, basic physics parameters (coupling constants, mixing angles, etc.) and the set of available interaction vertices (Feynman rules). By now, there exist explicit implementations of the Standard Model (SM), its Minimal Supersymmetric extension (MSSM) [13], the ADD model of large extra dimensions [8], and a comprehensive set of operators parametrizing anomalous triple and quartic electroweak gauge boson couplings [9].
  The required model inputs either are directly read from SHERPA's parameter file or input through a SLHA-conform file [42] for the case of the MSSM.

- `EXTRA_XS`

  In this module a (limited) collection of analytic expressions for simple $2 \to 2$ processes within the SM are provided together with classes embedding them into the SHERPA framework. This also includes methods used for the definition of the parton-shower evolution, such as colour connections and the hard scale of the process. The classes for phase-space integration are not included here, they are located in the module PHASIC++, since they are needed by AMEGIC++ as well.

- AMEGIC++

  AMEGIC++ [11] is SHERPA's matrix-element generator, which employs the method of helicity amplitudes [43,44]. Currently, it is SHERPA's preferred tool to provide matrix-element configurations. It works as a generator, which generates generators:

[3] The authors would like to thank T. Pierzchala for his help in implementing and testing the corresponding code.

[4] It should be stressed, however, that the parametrization [36], which has been implemented among others, is valid for the proposed TESLA photon collider only. This is because various assumptions concerning the laser parameters and especially the initial energy of the electrons have been made.

During the initialization run the matrix elements for a given set of processes within the SM, the MSSM or the ADD model, as well as their specific phase-space mappings are created by AMEGIC++ and stored in library files. Before the first production run, these libraries have to be linked to the program. They are used to calculate cross sections and to generate single weighted or unweighted events accordingly. A directory may be specified to store these cross sections, the status of the phase-space integrators, and the maximal weights of the processes. This information may be used to accelerate the potentially lengthy calculation phase in further event generation runs.
AMEGIC++ has been tested for multi-particle production in the Standard Model [12]. Its MSSM implementation has been tested in [13].

- PHASIC++

  Here all classes dealing with the Monte Carlo phase-space integration are located. For the evaluation of the initial-state (laser backscattering, initial-state radiation) and final-state integrals, the adaptive multi-channel method of [45, 46] is used by default together with a Vegas optimization [47] of the single channels. In addition, final-state integration accomplished by Rambo [48] and Sarge [49] is supported.

- APACIC++

  APACIC++ [30,31] contains classes for the simulation of both the initial- and the final-state parton shower. [5] The shower evolution is organized through an ordering in the parton virtual mass. Coherence effects are accounted for by explicit ordering of the opening angles in subsequent branchings. [6] All features neede for a consistent merging with matrix elements [14,15] are included, however the main part of the merging procedure is implemented in the `SHERPA` module itself.

- AMISIC++

  AMISIC++ contains classes for the simulation of multiple parton interactions according to [58]. In SHERPA the treatment of multiple interactions has been extended by allowing for the simultaneous evolution of an independent parton shower in each of the subsequent (semi-)hard collisions. This shower evolution is done by APACIC++. The beam–beam remnants are organized such that partons which are adjacent in colour space are also adjacent in momentum space. The corresponding classes for beam remnant handling reside in the `SHERPA` module.

- AHADIC++

  AHADIC++ is SHERPA's hadronization package, for translating the partons (quarks and gluons) into primordial hadrons, to be further decayed in HADRONS++. The algorithm bases on the cluster fragmentation ideas presented in [54, 59–61] and implemented in the HERWIG family of event generators. It should be noted, though, that the SHERPA version, based on [62], indeed differs from the original versions, see Sec. 4.7.2.

- HADRONS++

  HADRONS++ is the module for simulating hadron and τ-lepton decays. The resulting decay products respect full spin correlations (if desired). Several matrix elements and form-factor models have been implemented, such as the Kühn-Santamaría model, form-factor parametrizations from Resonance Chiral Theory for the τ and form factors from heavy quark effective theory or light cone sum rules for hadron decays. For further details, see Sec. 4.8.

[5] From version 1.0.9. final-state QED radiation off quarks and leptons can optionally be included.
[6] This is similar to the way PYTHIA [2, 50] handles the parton shower. HERWIG [51, 52] in contrast ensures coherence by taking angles as ordering parameter [53–55], a similar algorithm [56] is also employed in HERWIG++ [4, 57].

- Photons++

  The Photons++ module holds routines to add QED radiation to hadron and τ-lepton decays. This has been achieved by an implementation of the YFS algorithm [63]; in contrast to the implementation in [64], the structure of Photons++ is such that the formalism can be extended to scattering processes and to a systematic improvement to higher orders of perturbation theory [65]. The application of Photons++ therefore accounts for corrections that usually are added by the application of Photos [66] to the final state.

- `ANALYSIS`

  This package provides all the ingredients necessary to (internally) perform simple analyses of events generated with Sherpa. During generation events can be analyzed at different stages of their evolution, namely at the level of the hard matrix element, after parton-shower evolution and after full hadronization. Apart from numerous observables, a comprehensive list of selectors, jet algorithms and triggers is available to select final states that pass certain criteria.

- `SHERPA`

  Finally, `SHERPA` is the steering module that initializes, controls and evaluates the different phases during the entire process of event generation. Furthermore, all routines for the combination of parton showers and matrix elements, which are independent of the specific parton shower are found in this module. For details concerning the implementation of these routines, cf. [16] and Sec. 5. In addition, this sub-package provides an interface to the Lund String Fragmentation of Pythia 6.214 including its hadron decay routines.

- In addition to the modules of Sherpa, some scripts for alleviated installation of the package and easier handling are distributed. They are located in the subdirectory `TOOLS`. In almost all cases, it will only be necessary to run `makeinstall` and/or `makedist`.

The actual executable of the Sherpa generator can be found in the subdirectory `<prefix>/bin/` or `<prefix>/SHERPA/Run/` and is called `Sherpa`. To run the program, input files have to be provided in the current working directory or elsewhere by specifying the corresponding path. All output files are then written to this directory as well.

2 Installation procedure

Sherpa is distributed as a tarred and gzipped file named `Sherpa-<version>.tar.gz`, and can be unpacked in the current working directory with

```
tar -zxvf Sherpa-<version>.tar.gz .
```

All modules, briefly introduced in Sec. 1, have their own subdirectories in Sherpa's top-level directory. To install the framework, the `makeinstall` script should be used, which is located in the subdirectory `TOOLS`. To guarantee successful installation, the following tools should have been made available on the system: `make`, `autoconf`, `automake` and `libtool`. Furthermore, a C++ and FORTRAN compiler must have been provided. The authors recommend to use the following command line

```
./TOOLS/makeinstall -c
```

in order to run the installation. While processing the installation a log-file called `sherpa_install.log` is created, which records the entire installation procedure. This is useful in cases where the automatic compilation process fails. Then, users are invited to send the log-file as a bug report to the authors. The `makeinstall` script provides a number of combinable options listed and described in Tab. 1, including a help option accessible through the flag `-h`.

-c	configure before compiling (subset of '-t')
-t	rebuild 'Makefile.in's' and 'configure's'; configure before compiling
-f	display the full information on the setup
-s	create a simple install script to be customized by the user
--clean	execute 'make clean' in every module before compiling
--no-abort	ignore exit status of 'make'
--copt	define option for 'configure'
--mopt	define option for 'make' (default is '-j2')
--cxx	define CXX flag for 'make'
--f	define Fortran flag for 'make'
--force-tar	force extraction of tar files (enables '-e')
--clean-up	execute 'make clean' in every module and exit
--rm-libs	remove shared libraries; does not remove '.lo' files
--make-only	do not install
-h	display this help and exit

Table 1: Options of the makeinstall script.

If Sherpa has to be moved to a different directory after the installation, one has to set the following environment variables for each run:

- SHERPA_INCLUDE_PATH=$newprefix/include/SHERPA-MC
- SHERPA_SHARE_PATH=$newprefix/share/SHERPA-MC
- SHERPA_LIBRARY_PATH=$newprefix/lib/SHERPA-MC
- LD_LIBRARY_PATH=$SHERPA_LIBRARY_PATH:$LD_LIBRARY_PATH

SHERPA can be interfaced with various external packages, e.g. HepMC, for event output, or LHAPDF, for PDFs. For this to work, the user has to pass the appropriate commands to the configure step. Using the makeinstall script, this is achieved as shown by the example below:

```
./TOOLS/makeinstall --copt --enable-hepmc2=/path/to/hepmc2
                    --copt --enable-lhapdf=/path/to/lhapdf .
```

If you want to use the built-in interface to Lund fragmentation and hadron decays, you have to compile with Pythia support by specifying the the --enable-pythia option without any argument. The SHERPA package has successfully been compiled, installed and tested on SuSE, RedHat / Scientific Linux and Debian / Ubuntu Linux systems using the GNU C++ compiler versions 3.2, 3.3, 3.4, 4.0, 4.1 and 4.2 as well as on Mac OS X 10 using the GNU C++ compiler version 4.0. In any case the GNU FORTRAN compiler g77 or gfortran has been employed. Note that GCC version 2.96 is **not** supported.
For users, who modified parts of SHERPA according to their own needs or who added own modules and who would like to distribute the code, the makedist script is available. Its options can be listed by running makedist -h.

3 Running SHERPA

The Sherpa executable resides in the directory <prefix>/bin/ where <prefix> denotes the path to the SHERPA installation directory. The way a particular simulation will be accomplished is defined by several parameters, which can all be listed in a common file (or data card).[7] This steering file is

[7]Parameters can be alternatively specified on the command line; more details are given in Sec. 4.

called Run.dat and four example setups (i.e. Run.dat files) are distributed with the current version of Sherpa. They can be found in the subdirectories LEP91, EGamma, Tevatron1800 and LHC (all of which reside in the ./SHERPA/Run/ directory) for hadron production in e^+e^- and $e\gamma$ collisions, for inclusive W boson and Drell–Yan lepton-pair production, respectively. The very first step in running Sherpa is therefore to adjust all parameters to the needs of the desired simulation. The details for properly doing this are given in Sec. 4. In this section, the focus is on main issues for a successful operation of Sherpa. This is illustrated by discussing and referring to the parameter settings that come with Tevatron1800.

3.1 Process selection and initialization

Central to any Monte Carlo simulation is the choice of the hard processes that initiate the events. These hard processes are described by matrix elements. In Sherpa only a few $2 \to 2$ processes have been hard-coded and are available in EXTRA_XS. The more usual way is to employ Sherpa's internal tree-level matrix-element generator Amegic++. This, however, requires Sherpa to run twice: first, in the initialization run the matrix elements are generated and next, in the generation run the cross sections are calculated and/or events are generated.
The selection of the processes (or the process) happens in the (processes) part of the steering file, where particles or groups of particles are defined by means of the PDG numbering scheme. To get, e.g., the Tevatron1800 setup working, the initialization run has to be started by changing into the <prefix>/SHERPA/Run/Tevatron1800/ directory and executing

```
<prefix>/bin/Sherpa
```

for the first time. [8] For good book-keeping, it is highly recommended to reserve different subdirectories for different simulations as it is demonstrated with the four example setups. During the initialization run the Feynman diagrams for the processes are being constructed and translated into helicity amplitudes. Furthermore suitable phase-space mappings are being produced. The amplitudes and the integration channels are stored in library files that still need to be compiled and linked to the Sherpa main program. The library files are placed in a directory called Process, which sits in the Tevatron1800 directory together with the steering file. The initialization run apparently stops with an error message, which is nothing but the request to carry out the compilation and linking procedure for the generated matrix-element libraries: the makelibs script, provided for this purpose and created in the Tevatron1800 directory, has to be executed according to

```
./makelibs .
```

Afterwards Sherpa can be restarted using the same command line as before. In this (generation) run the cross sections of the hard processes are evaluated, [9] and, subsequently events are generated if EVENTS was specified either at the command line or added to the Run.dat file in the (run) section. The generated events are not stored into a file by default; for details on how to store the events see Sec. 6.2. Note that the computational effort to go through this procedure of generating, compiling and integrating the matrix elements of the hard processes depends on the complexity of the parton-level final states. For low multiplicities ($2 \to 2, 3, 4$), of course, it can be followed instantly.
Usually more than one generation run is wanted. As long as the parameters that affect the matrix-element integration are not changed, it is advantageous to store the cross sections obtained during the generation run for later use. This saves CPU time especially for large final-state multiplicities of the matrix elements. To store the integration results, a <result> directory has to be created in Tevatron1800. Then utilizing an extended command line reading

[8] The user may also run from an arbitrary directory, e.g. from <prefix>/SHERPA/Run/ employing <prefix>/bin/Sherpa PATH=Tevatron1800/. In the example, the keyword PATH is specified relative to the current working directory. It may also be specified by an absolute path. If it is not specified at all or it is omitted, the current working directory is understood.

[9] This also means that the integration over phase space is being optimized to arrive at an efficient event generation.

```
<prefix>/bin/Sherpa RESULT_DIRECTORY=<result>/
```

a generation run can be started and the results of the integration will be stored in <result>. The next time this command line is used, SHERPA will look for the integration results in <result> and read them in. Of course, if corresponding parameters do change, the cross sections have to be re-evaluated for a valid new generation run. The new results have to be stored in a new directory or the <result> directory may be re-used once it has been emptied. Basically, most of the parameters listed in the (model), (me) and (selector) part of Run.dat determine the calculation of cross sections. Standard examples are changing the magnitude of couplings, renormalization or factorization scales, changing the PDF or centre-of-mass energy, or, applying different cuts at the parton level. If being unsure whether a re-integration is required, a simple test is to remove the RESULT_DIRECTORY option from the run command and check whether the new integration numbers (statistically) comply with the stored ones. One more remark (or maybe warning) concerning the validity of the process libraries is in order here: it is absolutely mandatory to generate new library files, whenever the physics model is altered, i.e. particles are added or removed and hence new or existing diagrams may or may not anymore contribute to the same final states. Also, when particle masses are switched on or off new library files must be generated (however, masses may be changed between non-zero values keeping the same process libraries). Old library files cannot account for such changes, since once generated their functional structure is fixed. The best thing is to create a new and separate setup directory. Otherwise the Process and Result directories have to be erased:

`rm -rf Process/` and `rm -rf Result/` .

In either case one has to start over with the whole initialization procedure to prepare for the generation of events again.

3.2 Hadron-level event generation with SHERPA: *W* production at Tevatron Run I energies

The setup (or the Run.dat file) provided in ./SHERPA/Run/Tevatron1800/ can be considered as a standard example to illustrate the generation of fully hadronized events in SHERPA. Such events will include effects from parton showering, hadronization into primary hadrons and their subsequent decays into stable hadrons. Moreover, the example chosen here nicely demonstrates how SHERPA is used in the context of merging matrix elements and parton showers [14, 15].[10] In addition to the aforementioned corrections, this simulation of inclusive W production (with the W decaying into $e^-\bar{\nu}_e$) will then include higher-order jet corrections at the tree level. As a result the transverse-momentum distribution of the W boson as measured by the DØ and CDF collaborations at Tevatron Run I can be well described, see also [17–19].
Before event generation, the initialization procedure as described in Sec. 3.1 has to be completed. The matrix-element processes included in the setup are the following:

$$\begin{aligned} p\bar{p} &\to \text{parton parton} \to e^-\bar{\nu}_e\,, \\ p\bar{p} &\to \text{parton parton} \to e^-\bar{\nu}_e + \text{parton}\,. \end{aligned}$$

In the (processes) part of the steering file this translates into

```
Process : 93 93 -> 11 -12 93{1}
Order electroweak : 2
End process
```

[10]For more information on the combination of matrix elements and parton showers as implemented in SHERPA, cf. Sec. 5.

with explanations given in Sec. 5.3. The physics model for these processes is the Standard Model, which is specified in the `(model)` part of `Run.dat` via `MODEL=SM`. Fixing the order of electroweak couplings to 2, matrix elements of all partonic subprocesses for W production without any and with one extra QCD parton emission will be generated by AMEGIC++ during the first run. Note that in the `(me)` section of the data card, `ME_SIGNAL_GENERATOR=Amegic` has been chosen. After the compilation and linking of all newly created process libraries, the $2 \to 2$ and $2 \to 3$ hadronic cross sections are calculated. Various other parameters define the magnitude of these cross sections. Proton–antiproton collisions are considered at beam energies of 900 GeV; under part `(beam)` of the `Run.dat` file, one therefore has `BEAM_1=2212`, `BEAM_2=-2212` and `BEAM_ENERGY_1,2=900.0`. The fact that PDFs have to be used for this calculation and the chosen type of PDF can be found in the part of `Run.dat` named `(isr)`; e.g. here `PDF_SET=cteq6l` is employed. Model parameters and couplings can be set in the `Run.dat` section `(model)`, and the way couplings are treated can be defined under the `(me)` category. Here one also finds the master switch for operating the simulation in the CKKW merging mode: `SUDAKOV_-WEIGHT=1`. This enforces that the scheme to set renormalization and factorization scales is `SCALE_-SCHEME=CKKW`. The matrix elements still have to be regularized to obtain meaningful cross sections. This is achieved by specifying "`JetFinder sqr(20/E_CMS) 1.`" in the `(selector)` part of `Run.dat`, which is the appropriate way of regularizing the matrix elements in the scope of CKKW merging. Always in such cases one has to make sure that

- the Sudakov weights have been enabled (`SUDAKOV_WEIGHT=1`),
- a jet finder with a reasonable $Q_{\rm cut}$ has been specified (here $Q_{\rm cut} = 20$ GeV) and
- both showers, initial- **and** final-state shower, have been turned on.

The last item refers to the `(shower)` part of `Run.dat`, namely `FSR_SHOWER=1` and `ISR_SHOWER=1`. After the integration has been completed and true event generation is being accomplished, in the CKKW mode, the parton-level final states are being reweighted and undergoing vetoed parton showering. This is ensured with both types of showers being enabled (see last item from the above list). To eventually obtain fully hadronized events, the entire soft-physics treatment has to be made available by adjusting the `(fragmentation)` section of the steering file. This is ensured by `FRAGMENTATION=Ahadic`, which will run SHERPA's cluster hadronization, and `DECAYMODEL=Hadrons`, which will run SHERPA's hadron decays. Additionally corrections owing to photon emissions are taken into account.

3.3 Calculation of tree-level cross sections and parton-level event generation with SHERPA

SHERPA has its own tree-level matrix-element generator called AMEGIC++. Furthermore, with the module PHASIC++, sophisticated and robust tools for phase-space integration are provided. Therefore SHERPA obviously can be used as a cross-section integrator. Because of the way Monte Carlo integration is accomplished, this immediately allows for parton-level event generation. Taking the `Tevatron1800` setup, users have to modify just a few settings in `Run.dat` and would arrive at a parton-level generation for the process $gd \to e^- \bar{\nu}_e u$, to name an example. When, for instance, the options "`EVENTS=0 OUTPUT=2`" are added to the command line, a pure cross-section integration for that process would be obtained with the results plus integration errors written to the screen.
For the example, the `(processes)` section alters to

```
Process : 21 1 -> 11 -12 2
Order electroweak : 2
End process
```

and under the assumption to start afresh, the initialization procedure has to be followed as before. For sure, now `SUDAKOV_WEIGHT=0` has to be fixed; the scale scheme `CKKW` would still be working perfectly, however, a more reasonable setting comes with the self-explanatory `SCALE_SCHEME=S_HAT`. The regularization of the infrared divergences can be taken care of, as before, by the jet finder. Picking the same collider environment as in the previous example there are only two more changes before the `Run.dat` file is ready for the calculation of the hadronic cross section of the process $gd \to e^- \bar{\nu}_e u$ at Tevatron Run I and subsequent parton-level event generation with SHERPA. These changes read `FSR_SHOWER=0` and `ISR_SHOWER=0`, to switch off parton showering, and, `FRAGMENTATION=Off`, to do so for the hadronization effects.

3.4 SHERPA's total inclusive cross section

To determine the total cross section, in particular in the context of running CKKW merging with SHERPA, the internal analysis package provided within SHERPA can be employed. A file `Analysis.dat` needs to be created containing the lines

```
BEGIN_ANALYSIS {
  LEVEL Hadron;
  PATH_PIECE Norm/;
  Statistics FinalState;
} END_ANALYSIS;
```

and added to the directory where `Run.dat` resides. Doing SHERPA internal analyses has to be enabled using `ANALYSIS=1`, either in the `(run)` section of `Run.dat` or on the command line.[11] After program termination, a file `Statistics_Observable_FinalState` can be found in the `Norm` subdirectory of the analysis-result path that was specified for the run by `ANALYSIS_OUTPUT`. As an example a file for $p\bar{p} \to e^+e^-$ is listed below:

```
# Process Name              # evts          <w>         <sigma>        <Delta>
2_2__b__bb__e-__e+               1      0.36933        0.399068        0.92548
2_2__c__cb__e-__e+               2     0.566878        0.595396       0.952103
2_2__d__db__e-__e+             271       58.455         62.8626       0.929885
2_2__s__sb__e-__e+              28      4.78567         5.14493       0.930172
2_2__u__ub__e-__e+             443      97.4499         104.727       0.930514
...

# Total XS      : 177.015 +- ( 3.64851 = 2.06113 % )
```

The first column contains the process name, the second the number of generated events for that process. The third column gives the mean weight for the process, which is composed of matrix-element and Sudakov weight. The respective averages of the latter two are given in the last two columns. Finally, the total cross section is listed at the end of the file. Note that the Monte Carlo error quoted for the total cross section is determined during event generation. It, therefore, differs substantially from the error quoted during the integration step.

4 Parameters in SHERPA and their specification

In the following all parameters of SHERPA are described. Note that some of them are not listed in the example data cards that are distributed with SHERPA. The examples can be found in the subdirectories `LEP91`, `EGamma`, `Tevatron1800` and `LHC`, all of which reside in the directory `<prefix>/SHERPA/Run/`.

4.1 Input structure

There are two general ways of modifying the parameters of SHERPA:

1. Through the data file(s):
   SHERPA's parameters are grouped according to the different aspects of event generation, e.g.

[11] Note that none of the predefined setups comes with a `(run)` section, if needed by the user, it should just be added.

the beam parameters in the group `(beam)` and the fragmentation parameters in the group `(fragmentation)`. Each of these groups is described in detail in the rest of this manual.
Each group of parameters can be found in a section of SHERPA's parameter file called `Run.dat`, for example,

```
(beam){
  BEAM_ENERGY_1 = 7000.
  ...
}(beam)
```

or, for backwards compatibility, in a separate file with the appropriate name, e.g. for `(beam)`, it translates to `Beam.dat`. If such a section or file does not exist in the setup directory,[12] a SHERPA-wide fallback mechanism is employed, searching the file in various locations in the following order:

- `$SHERPA_DAT_PATH/<PATH>/` ,
- `$SHERPA_DAT_PATH/` ,
- `$prefix/share/SHERPA-MC/<PATH>/` ,
- `$prefix/share/SHERPA-MC/` .

2. Through command line input:
All parameters can be overwritten on the command line, i.e. command-line input has the highest priority. The syntax is

```
<prefix>/bin/Sherpa KEYWORD1=value1 KEYWORD2=value2 ...
```

and to change, e.g., the default number of events, the corresponding command line is

```
<prefix>/bin/Sherpa EVENTS=10000  .
```

All over SHERPA, particles are defined by the particle code proposed by the PDG. These codes and the particle properties will be listed during each run with `OUTPUT=2` for the elementary particles and `OUTPUT=4` for the hadrons. In both cases, antiparticles are characterized by a minus sign in front of their code, e.g. a μ^- has code `13`, while a μ^+ has `-13`.
There are a few extra features for an easier handling of the parameter file(s), namely:

- Tag replacing: Specifying a tag on the command line using the syntax `<Tag>:=<Value>` will replace every occurrence of `<Tag>` during read-in. For example, one could run

```
<prefix>/bin/Sherpa YCUT:=20
```

  and use this tag in the `(me)` and `(selector)` sections like

```
(me){
  RESULT_DIRECTORY = Result_YCUT/
}(me)
(selector){
  JetFinder   sqr(20/E_CMS) 1.
}(selector)
```

- Almost every parameter has a reasonable default value in the code, so users preparing their own setup file may simply leave out some parameters.

[12] The setup directory can be set by adding the `PATH=<PATH>` option to the command line; as the default the current working directory is taken.

0	errors	all error messages, most general run information
1	events	event output
2	info	more detailed information about what is going on during the run
4	tracking	useful information in order to track problems a user might encounter, i.e. to approximately identify the place/cause of problems
16	logfile	write a log-file containing information about the runtime environment and the setup

Table 2: The output levels in SHERPA.

4.2 General switches

General switches for the steering of the generator can be defined in the (run) section. [13] The following parameters are available:

1. The number of events is set through EVENTS; if this is nowhere specified, no events are generated.

2. The output level can be specified through OUTPUT. If not specified the output level is 2. Different levels can be overlayed by simply adding the level number. For their meaning, please cf. Tab. 2.

3. SHERPA uses a random-number generator as described in [67]. The two independent integer-valued seeds are specified by the option "RANDOM_SEED=A B". The seeds A and B may range from 0 to 31328 and from 0 to 30081, respectively. If RANDOM_SEED is not specified at all or only by one integer number, the old random-number generator (SHERPA 1.0.6 and older) will be used.

4. The internal analysis can be switched on or off by setting the ANALYSIS switch to 1 or 0, respectively. An ANALYSIS_OUTPUT main directory can be specified by setting this keyword meaningfully. The new directory will be created w.r.t. the working directory. For detailed information, cf. Sec. 4.10.

5. Writing a log-file can be initiated through a LOG_FILE setting, however it is only created when using the output mode "logfile" (cf. Tab. 2). It then contains all information about the setup. In case that a SHERPA run crashes unexpectedly, the user may want to report the bug to the authors:

   - First Sherpa's output should be checked for hints on the abort of the program. If present, they can be used to further investigate, e.g. by scanning the manual.
   - If this does not help, the user may contact the SHERPA team, providing all information on the setup. That may also include
     - if it exists, the status recovery directory "Status__<date of crash>" possibly produced just before the program abort.
     - the command line that was used to run Sherpa.

   The authors would like to encourage the user to simplify the setup leading to the crash such that it gets as close as possible to one of the default setups. In that circumstance, the authors may be able to help more quickly.

6. Some more options, which are not shown in the example files, can be specified in the Run.dat file. Some of them are explained below, some only listed and explained elsewhere.

   (a) Options for event output/storage are described in Sec. 6.

[13] Note that none of the example setups comes with a (run) section, if needed by the user, it should simply be added.

(b) A run time limitation can be given in user CPU seconds through `TIMEOUT`. This option is of some relevance when running SHERPA on a batch system. Since in many cases jobs are just terminated, this allows to interrupt a run, to store all relevant information and to restart it without any loss. This is particularly interesting when carrying out long integrations. Contrary, setting the `TIMEOUT` variable to `-1`, which is the default setting, translates into having no run time limitation at all.

(c) The algorithm used to transfer spin-correlation information from AMEGIC++ to HADRONS++ is switched off (`=0`) by default. It can be switched on via `SPIN_CORRELATIONS=1`. Process libraries have to be re-created in this case.

(d) The targeted numerical accuracy can be specified through `NUM_ACCURACY`, e.g. for comparing two numbers. This might have to be reduced if gauge tests fail for numerical reasons.

4.3 Incoming beams and parton density functions

The setup of the colliding beams and the specification of parton density functions is covered by the `(beam)` and `(isr)` sections of the steering file, respectively. The mandatory settings to be made inside `(beam)` include:

1. The initial beam particles specified through `BEAM_`{1,2}, and given by their PDG particle number. For (anti)protons and (positrons) electrons, e.g., these are given by $(-)2212$ or $(-)11$, respectively. The code for photons is 22.

2. The energies of both incoming beams, defined through `BEAM_ENERGY_`{1,2}, given in units of GeV.

Apart from these basic settings there are further options to specify the polarizations of the incoming beams or to specify a certain beam spectrum in case photon collisions initiated by leptonic or hadronic beams are to be considered. The following switches are available:

1. For elementary beam particles, e.g. leptons or photons, their polarization degree in terms of the helicity can be specified by `BEAM_POL_`{1,2}. The valid parameter range is $-1\ldots 1$.

2. The initial beam particles may produce beams of another particle species (well before considering ISR) that is accounted for by specifying a corresponding beam spectrum. The related keyword is `BEAM_SPECTRUM_`{1,2}, and there exist the following options:

   - `Monochromatic`: the beam energy is unaltered and the beam particles remain unchanged. That is the default and corresponds to ordinary hadron–hadron or lepton-lepton collisions.
   - `EPA`: this enables the equivalent photon approximation for colliding protons. The resulting beam particles are photons that follow a dipole form factor parametrization according to [68].
   - `Simple_Compton`: this corresponds to a simple light backscattering off the initial lepton beam and produces initial-state photons with a corresponding energy spectrum.
   - `Laser_Backscattering`: this can be used to describe the backscattering of a laser beam off initial leptons. The energy distribution of the emerging photon beams is modelled by the CompAZ parametrization [36]. Note, this parametrization is valid only for the proposed TESLA photon collider, as various assumptions about the laser parameters and the initial lepton beam energy have been made.
   - `Spectrum_Reader`: a user defined spectrum is used to describe the energy spectrum of the assumed new beam particles. The name of the corresponding spectrum file needs to be given through the keyword `SPECTRUM_FILE_`{1,2}.

3. Considering hadronic collisions the following parameters associated to the hadron remnant treatment can be adjusted: the mean value of the intrinsic $k_\perp$ distribution and its Gaussian variance, `K_PERP_MEAN_`{1,2} and `K_PERP_SIGMA_`{1,2}, respectively. Both values have to be given in units of GeV.

option	comment
0	default, the full spectrum
1	the pure Compton component of the spectrum
2	the pure two photon component of the spectrum
3	the pure re-scattering component of the spectrum

Table 3: Different options available for the variable LASER_MODE and their associated choices for the used components of the entire photon spectrum produced via laser backscattering.

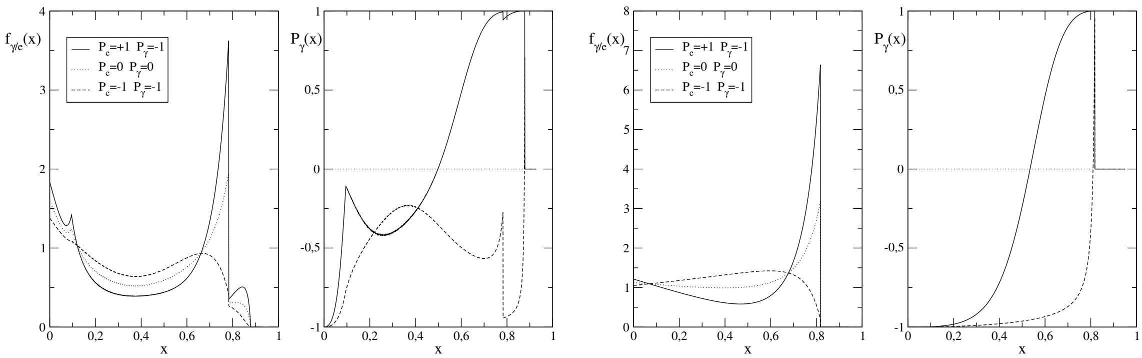


Figure 1: Laser backscattering spectra and degree of circular polarization for backscattered photons according to the CompAZ parametrization (left) and according to the simple Compton spectrum (right). Different combinations of electron and laser photon polarization are shown. The energy of the electron beam and the laser are $E_e = 250$ GeV and $\omega_\gamma = 1.17 \cdot 10^{-9}$ GeV, respectively ($x_e = 3.61$).

4. When considering a non-trivial beam spectrum, e.g. Laser_Backscattering, the minimal and maximal fraction of the total energy s carried by the new beam particles can be constrained by setting BEAM_SMIN and BEAM_SMAX. Possibly, the latter can be subject to a further reduction due to the form of the chosen spectrum.

5. There is a list of additional parameters for specifying the laser light in the backscattering setup. The energy and circular polarization of the used lasers is given by E_LASER_{1,2} and P_LASER_{1,2}, respectively. For the latter, the range of allowed values is again -1 to $+1$. The keyword LASER_MODE denotes the different components of the backscattered laser light brought to collision later on. The default value is 0 (all components). The full set of options is summarized in Tab. 3. An example of the spectra and polarization degrees for photons generated in laser backscattering via Laser_Backscattering and Simple_Compton are shown in Fig. 1. By setting the switch LASER_NONLINEARITY either 1 or 0 non-linear laser effects can be taken into account or not. Note that this option is valid for the full laser backscattering spectrum only.

Having specified the properties of the incoming beams, these beam particles may exhibit a certain substructure, parametrized through parton density functions that account for initial-state radiation. The settings associated to the PDFs reside in the (isr) section of the main steering file. The available switches include:

1. The PDG code of the incoming particles, specified by BUNCH_{1,2}. Per default these are taken to be identical to the parameters BEAM_{1,2}, assuming the default beam spectrum is Monochromatic. In case the Simple_Compton or Laser_Backscattering spectra are enabled the bunch particles would have to be set to 22, the PDG code of the photon.

2. The ISR_{1,2} switch defines if a substructure of the bunch particle is to be resolved. The corresponding settings are On and Off. For incoming protons ISR must be enabled, which is

PDF_GRID_PATH	PDF_SET
CTEQ6Grid	cteq6m, cteq6d, cteq6l, cteq6l1
MRST99Grid	MRST99
MRST01Grid	MRST01LO

Table 4: The proton PDFs available within SHERPA and their associated settings of the corresponding parameters PDF_GRID_PATH and PDF_SET. By choosing one of the PDFs out of the available PDF_SET options the value of the PDF_GRID_PATH has to be set accordingly.

ensured by default. For colliding photons this parameter defines if the resolved or unresolved component of the cross section is to be considered. For leptons this switch steers the inclusion of a structure function, corresponding to the emission of undetected photons "before" the actual lepton–lepton interaction.

3. The minimal and maximal fraction of the total energy s carried by the particles after the ISR treatment can be constrained by the parameters ISR_SMIN and ISR_SMAX. The default values are 0 and 1, respectively.

4. Depending on the species of the bunch particle the structure function to be considered needs to be specified further. The available options for incoming (anti)protons, leptons, and photons are listed here:

   - For protons, SHERPA provides access to a variety of internal PDFs and in addition the LHAPDF package [41].
     - There exist explicit interfaces to the CTEQ6 PDF sets [37] and the MRST99 [38] and MRST01LO structure functions [39]. The required grid files are part of the standard SHERPA distribution. The relevant parameters for specifying a certain PDF and pointing to the corresponding grid file are PDF_SET and PDF_GRID_PATH, respectively. The available parameter combinations are listed in Tab. 4. Apart from this, an explicit set number (error PDFs) can be selected by setting PDF_SET_VERSION correspondingly. The default is 1, the central set.
     - If SHERPA has been compiled with LHAPDF enabled, see Sec. 2, all the PDFs provided by the package can be accessed. To do so, the variable PDF_SET has to be set to the corresponding set name, e.g. cteq6l.LHpdf or MRST2001lo.LHgrid. A specific member of the different PDF fits can be picked via PDF_SET_VERSION. The native wrapper functions of LHAPDF are utilised by Sherpa, so that you can use the LHAPDF environment variables (e.g. $LHAPATH to specify the path to your PDF sets).

     The default proton PDF used by SHERPA is the central set of cteq6l from the internal interface.
   - For leptons, the ISR handling is set up by specifying the analytic structure function through the switches ISR_E_ORDER and ISR_E_SCHEME. The former (default value is 1) fixes the perturbative order in the fine structure constant α, whereas the latter (default value is 2) determines the way of respecting non-leading terms in the structure function. Possible options for the last switch are 0 ("mixed" choice), 1 ("eta" choice), or 2 ("beta" choice).
   - For incoming photons, there exists an internal interface to the GRV photon structure function [40] that is automatically chosen if the bunch particle is a photon and the ISR_{i} switch equals On. Note, at present SHERPA provides no access to the photon structure functions of the LHAPDF package.

4.4 Matrix-element generation

4.4.1 General settings

The (me) section of the steering file covers the general settings for matrix-element generation. The basic specifications include:

1. ME_SIGNAL_GENERATOR to select the module used for matrix-element generation. The choices are Amegic to use this build-in ME generator or Internal for a library of simple $2 \to 2$ cross sections made available through the EXTRA_XS package.

2. Via the switch EVENT_GENERATION_MODE SHERPA provides the option to generate either Weighted or Unweighted events.

Additionally, different scale choices relevant at different stages of process integration and event generation are made available.

1. Gauge couplings in the matrix element can be set through COUPLING_SCHEME. It should be stressed that this setting occurs on the level of vertices. It is permanent and independent of what is suggested by the keyword "Running". In particular, the COUPLING_SCHEME=Fixed refers to using the ALPHAS(default) and 1/ALPHAQED(default) values given in the (model) section of Run.dat or Model.dat for the respective coupling. In contrast, when using the COUPLING_SCHEME=Running the couplings are evaluated at the c.m. scale s of the incoming beams starting the evolution at ALPHAS(MZ) and 1/ALPHAQED(0).

2. For the Yukawa couplings, the couplings in the matrix element can be set through YUKAWA_-MASSES. The options "Running" and "Fixed" correspond to the c.m. energy and the pole mass of the considered particle, respectively. Another keyword of interest is the YUKAWA_MASSES_FACTOR, which works as a prefactor for the couplings.

3. During the calculation of the matrix elements, the renormalization scale for the strong coupling constant and the factorization scale can be modified. The parameter SCALE_SCHEME selects the definition of both. Its value is a string composed from one of the scale tags listed in Tab. 5 and eventually some of the scale modifiers listed in Tab. 6. A corresponding example is

   ```
   SCALE_SCHEME=G_MEAN_PT2+STRONG+DOUBLE
   ```

   Note that the scale scheme tag is read as one single string and therefore no spaces are allowed between the single tags and operators. The parameter KFACTOR_SCHEME switches on (=1) or off (=0) the influence of the renormalization scale on α_s, provided that COUPLING_SCHEME is set to "Running".

4. A fixed, user-defined scale squared can be defined through the keyword FIXED_SCALE.

5. The variable SUDAKOV_WEIGHT is the master switch for preparing the matrix elements in order to combine them with the parton shower, cf. Sec. 5.

4.4.2 Physics models: particle properties, input parameters and interaction vertices

The definition of the assumed physical framework for a generator run is handled by the module MODEL. This part of the code is responsible for defining the (elementary) particle content of the underlying theory, the model parameters determining the interaction strengths, and the interaction vertices or Feynman rules as such. This sets all the necessary inputs for calculating hard matrix elements or emission probabilities in the parton-shower simulation. The parameters and switches related to this task are set in the (model) section of the steering file.

Tag	Scale Definition
CKKW	CKKW weighting scheme
FIX_SCALE	a fixed scale, its value given by the additional keyword FIXED_SCALE
S_HAT	center of mass energy squared of hard process
G_MEAN_PT2	geometric mean of p_T^2; for QCD $2 \to 2$ processes, $2stu/(s^2+t^2+u^2)$
A_MEAN_PT2	arithmetic mean of p_T^2
MIN_PT2	minimum p_T^2
SUM_PT2	sum of p_T^2

Table 5: Available scale schemes.

Tag	Option
STRONG	only strongly interacting particles are considered
MASSIVE	p_T^2 is replaced with m_T^2
HALVE	the scale is halved
DOUBLE	the scale is doubled

Table 6: Available scale modifiers.

At present SHERPA provides implementations of the complete Standard Model of particle physics and some of its speculative extensions. This includes the Two Higgs Doublet Model (THDM), the Minimal Supersymmetric Standard Model (MSSM), the ADD model of large extra dimensions, and a comprehensive list of non-standard triple and quartic electroweak gauge boson couplings (AGC). Although SHERPA's matrix-element generator AMEGIC++ is producing tree-level amplitudes only, loop-induced effective couplings of gluons and photons to the SM Higgs boson (EHC) have been included. The modularity of the code allows for a very simple and well contained implementation of alternative physics scenarios.
The concrete physics model to be used is set through the variable `MODEL`, whose value has to be chosen from the list of available options: `SM`, `SM+EHC`, `SM+AGC`, `SM+4thGen`, `THDM`, `MSSM`, or `ADD`. Accordingly, SHERPA initializes the full particle content of the respective theory, thereby fixing the particles quantum numbers and Monte Carlo codes. For particle numbering SHERPA follows the PDG convention [69].[14]
During initialization default values for particle masses and widths are used that can, however, be adopted by setting corresponding parameters. To modify certain particle properties the following options are available:

1. `MASS[PDG] = <value in GeV>` sets mass of particle with Monte Carlo code `PDG`. Note, that automatically particles and anti-particles are considered.

2. The switch `MASSIVE[PDG]` can be used to decide whether the finite mass of particle `PDG` is to be considered in matrix-element calculations or not.

3. `WIDTH[PDG] = <value in GeV>` width assignment for particle `PDG`.

4. When setting the switch `ACTIVE[PDG] = 0` the particle with code `PDG` is excluded from the model.

5. Through the `STABLE[PDG]` switch a particle can be set either stable or unstable. At present this will affect only the simulation of τ-lepton decays, treated with by the HADRONS++ package, see Sec. 4.8. Decays of other particles have to be specified in the process setup, cf. 4.4.3 and are not dealt with automatically, independent of the setting of `STABLE[PDG]`.

[14]Note, there is one slight variation. In SHERPA the $\chi_i^\pm$ particles in supersymmetric theories have the reversed charge convention for particles and anti-particles.

A complete list of all the particles of the chosen model, including the basic parameters is listed at the beginning of each generator run. This list also contains a set of containers that collect particles with similar properties, namely

- lepton (carrying number 90),
- neutrino (carrying number 91),
- fermion (carrying number 92),
- jet (carrying number 93),
- quark (carrying number 94).

These containers hold all massless particles and anti-particles of the denoted type and allow for a more efficient definition of initial and final states to be considered. The jet container consists of the gluon and all massless quarks (as set by `MASS[..]=0.0` or `MASSIVE[..]=0`).
Having selected the physics model, and with the model's particles being initialized, the input parameters of the model have to be provided. The available variables depend on the concrete model and will be listed in the following for each available model implementation. Note, this list of required model parameters can be printed on the screen, when running `Sherpa` with the command line option `SHOW_MODEL_SYNTAX=1`.

The Standard Model parameters: the SM inputs for the electroweak sector can be given in four different schemes, that correspond to different choices of which SM physics parameters are considered fixed and which are derived from the given quantities. The input schemes are selected through the `EW_SCHEME` parameter, whose default is `0`. The following options are provided:

- Case `0`: all EW parameters are explicitly given. Here M_W, M_Z and M_H are taken as inputs, and the parameters `1/ALPHAQED(0)`, `SIN2THETAW`, `VEV` and `LAMBDA` have to be specified as well. While `1/ALPHAQED(0)` corresponds to the fine structure constant at zero momentum transfer, the parameters `SIN2THETAW`, `VEV`, and `LAMBDA` thereby specify the weak mixing angle, the Higgs field vacuum expectation value, and the Higgs quartic coupling respectively.
- Case `1`: all EW parameters are calculated out of M_W, M_Z, M_H and `1/ALPHAQED(0)`.
- Case `2`: all EW parameters are calculated out of `1/ALPHAQED(0)`, `SIN2THETAW`, `VEV` and M_H.
- Case `3`: this choice corresponds to the G_μ-scheme. The EW parameters are calculated out of the weak gauge boson masses M_W, M_Z, the Higgs boson mass M_H and the Fermi constant `GF`.

To account for quark mixing the CKM matrix elements have to be assigned. For this purpose the Wolfenstein parametrization [70] is employed. The order of expansion in the λ parameter is defined through `CKMORDER`, with default `0` corresponding to a unit matrix. The parameter convention for higher expansion terms reads:

- `CKMORDER = 1`, the `CABIBBO` parameter has to be set, it parametrizes λ and has the default value 0.2272.
- `CKMORDER = 2`, in addition the value of `A` has to be set, its default is 0.818.
- `CKMORDER = 3`, the order λ^3 expansion, `ETA` and `RHO` have to be specified. Their default values are 0.349 and 0.227, respectively.

The remaining parameter to fully specify the Standard Model is the strong coupling constant at the Z-pole, given through `ALPHAS(MZ)`. Its default value is 0.1188. For the two fine structure constants there is the option to provide fixed values that can be used in calculations of matrix elements in case running of the couplings is disabled. The two keywords are `1/ALPHAQED(default)` and `ALPHAS(default)`. When using a running strong coupling, the order of the perturbative expansion used can be set through `ORDER_ALPHAS`, where the default `0` corresponds to one-loop running and `1,2,3` to 2,3,4-loops, respectively.

The Minimal Supersymmetric Standard Model inputs: to input MSSM spectra to SHERPA, files that conform to the SUSY-Les-Houches-Accord [42] are used. The actual SLHA file name has to be specified by `SLHA_INPUT` and has to reside in the current run directory, i.e. `PATH`. From this file the full low-scale MSSM spectrum is read, including sparticle masses, mixing angles etc. In addition information provided on the total particles widths is read from the input file. Note that the setting of masses and widths through the SLHA input is superior to setting through `MASS[PDG]` and `WIDTH[PDG]`.

Parameters of the ADD model of large extra dimensions: the variable `N_ED` specifies the number of extra dimensions. The value of the Newtonian constant can be specified in units of $\hbar c(GeV/c^2)^{-2}$ using the keyword `G_NEWTON`. The size of the string scale M_S can be defined by the parameter `M_S`. Setting the value of `KK_CONVENTION` allows to change between three widely used conventions for the definition of M_S and the way of summing internal Kaluza-Klein propagators. The switch `M_CUT` one restricts the c.m. energy of the hard process to be below this specified scale.
For details of the implementation, the reader is referred to [8].

Parameters for Anomalous Gauge Couplings: SHERPA includes a number of effective Lagrangians describing anomalous gauge interactions:

- ***WWV* interactions:**
  The general coupling between two charged vector bosons and a neutral one is given by [9]

$$\begin{aligned}\mathcal{L}_{WWV}/g_{WWV} &= ig_1^V(W^\dagger_{\mu\nu}W^\mu V^\nu - W^\dagger_\mu V_\nu W^{\mu\nu}) + i\kappa_V W^\dagger_\mu V_\nu W^{\mu\nu} \\ &+\frac{i\lambda_V}{m_W^2}W^\dagger_{\lambda\mu}W^\mu_\nu V^{\nu\lambda} - g_4^V W^\dagger_\mu W_\nu(\partial^\mu V^\nu + \partial^\nu V^\mu) \\ &+g_5^V\epsilon^{\mu\nu\rho\sigma}(W^\dagger_\mu \overleftrightarrow{\partial}_\rho W_\nu)V_\sigma + \frac{i\tilde{\kappa}_V}{2}\epsilon^{\mu\nu\rho\sigma}W^\dagger_\mu W_\nu V_{\rho\sigma} \\ &+\frac{i\tilde{\lambda}_V}{2m_W^2}\epsilon^{\mu\nu\rho\sigma}W^\dagger_{\mu\lambda}W^\lambda_\nu V_{\rho\sigma}\ , \qquad (1)\end{aligned}$$

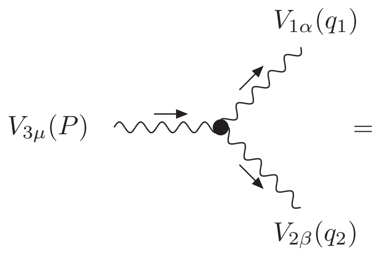
  where V^μ stands for either the photon or the Z field, W^μ is the W^- field, $W_{\mu\nu} = \partial_\mu W_\nu - \partial_\nu W_\mu$, $V_{\mu\nu} = \partial_\mu V_\nu - \partial_\nu V_\mu$, $\tilde{V}_{\mu\nu} = \frac{1}{2}\epsilon_{\mu\nu\rho\sigma}V^{\rho\sigma}$ and $(A\overleftrightarrow{\partial_\mu}B) = A(\partial_\mu B) - (\partial_\mu A)B$. The overall coupling constants are

$$g_{WW\gamma} = -e\ , \qquad (2)$$
$$g_{WWZ} = -e\cot\theta_W\ . \qquad (3)$$

  The parameters for the individual coupling terms can be specified through the switches `G1_GAMMA`, `KAPPA_GAMMA`, `LAMBDA_GAMMA`, `G4_GAMMA`, `G5_GAMMA`, `KAPPAT_GAMMA`, `LAMBDAT_GAMMA`, `G1_Z`, `KAPPA_Z`, `LAMBDA_Z`, `G4_Z`, `G5_Z`, `KAPPAT_Z` and `LAMBDAT_Z`. As a default the Standard Model limit is used (`G1_GAMMA/Z=KAPPA_GAMMA/Z=1`, all other `=0`).

- **Quadrupole interactions:**
  The following SU(2) custodial symmetry conserving interactions are implemented [71]

$$\mathcal{L}_4 = \alpha_4 e^4\left(\frac{1}{2}W^\dagger_\mu W^{\dagger\mu}W_\nu W^\nu + \frac{1}{2}(W^\dagger_\mu W^\mu)^2 + \frac{1}{c_W^2}W^\dagger_\mu Z^\mu W_\nu Z^\nu + \frac{1}{4c_W^4}(Z^\mu Z^\mu)^2\right)\ , \qquad (4)$$
$$\mathcal{L}_5 = \alpha_5\left((W^\dagger_\mu W^\mu)^2 + \frac{1}{c_W^2}W^\dagger_\mu Z^\mu W_\nu Z^\nu + \frac{1}{4c_W^4}(Z^\mu Z^\mu)^2\right)\ . \qquad (5)$$



$$= ie\Gamma^{\alpha\beta\mu}_{V_1 V_2 V_3}(q_1, q_2, P)$$

Figure 2: Feynman rule for the anomalous triple neutral gauge boson vertex.

Here, the parameters can be specified through `ALPHA_4_G_4`(=$\alpha_4 e^4$) and `ALPHA_5`.

- **Anomalous interactions between photons and Z bosons:**
  The general anomalous coupling of two on-shell neutral gauge bosons (V_1 and V_2) to an off-shell boson (V_3) has been given in ref. [9]. The corresponding Feynman rule is displayed in Fig. 2, where the vertex functions are given by ($V = Z/\gamma$)

$$\begin{aligned}
\Gamma^{\alpha\beta\mu}_{ZZV}(q_1, q_2, P) &= \frac{i\left(P^2 - m_V^2\right)}{m_Z^2}\left[f_4^V\left(P^\alpha g^{\mu\beta} + P^\beta g^{\mu\alpha}\right) + f_5^V \epsilon^{\mu\alpha\beta\rho}(q_1 - q_2)_\rho\right], &(6)\\
\Gamma^{\alpha\beta\mu}_{Z\gamma V}(q_1, q_2, P) &= \frac{i\left(P^2 - m_V^2\right)}{m_Z^2}\left\{h_1^V\left(q_2^\mu g^{\alpha\beta} - q_2^\alpha g^{\mu\beta}\right) + \frac{h_2^V}{m_Z^2}P^\alpha\left[(Pq_2)g^{\mu\beta} - q_2^\mu P^\beta\right]\right.\\
&\quad \left. + h_3^V \epsilon^{\mu\alpha\beta\rho} q_{2\rho} + \frac{h_4^V}{m_Z^2}P^\alpha \epsilon^{\mu\beta\rho\sigma}P_\rho q_{2\sigma}\right\}. &(7)
\end{aligned}$$

  The coupling parameters are specified through `F4_GAMMA`, `F5_GAMMA`, `H1_GAMMA`, `H2_GAMMA`, `H3_-GAMMA`, `H4_GAMMA`, `F4_Z`, `F5_Z`, `H1_Z`, `H2_Z`, `H3_Z` and `H4_Z`, all equal zero by default.
  It should be noted that the most general anomalous coupling between three off-shell neutral gauge bosons allows for more coupling terms [72] which are not implemented in the current version. Outside the on-shell limit of two of the vector bosons a symmetrized version of the above vertex is used (e.g. $\Gamma_{ZZZ} = \Gamma_{Z_1 Z_2 Z_3} + \Gamma_{Z_2 Z_3 Z_1} + \Gamma_{Z_3 Z_1 Z_2}$).

Due to the effective nature of the anomalous couplings unitarity might be violated for coupling parameters other than the SM values. For very large momentum transfers, such as probed at the LHC, this will lead to unphysical results. As discussed in Ref. [73] this can be avoided introducing form factors to be applied on the deviation of coupling parameters from their Standard Model values,

$$a(\hat{s}) = \frac{a_0}{(1 + \hat{s}/\Lambda^2)^n},$$

where $\hat{s}$ is the collision energy of the partonic process, Λ represents a scale at which new physics become important and an exponent, typically to be chosen as $n = 2$. The corresponding switches are `UNITARIZATION_SCALE` and `UNITARIZATION_N`. By default the form factor is switched off setting $n = 0$.

Two Higgs Doublet Model inputs: the THDM is incorporated as a subset of the MSSM Lagrangian. It is defined as the extension of the SM by a second $SU(2)$ doublet of Higgs fields. Besides the particle content of the SM it contains interactions of five physical Higgs bosons; a light and a heavy scalar, a pseudo-scalar and two charged ones $H^\pm$. Besides the SM inputs the model is defined through the masses and widths of the Higgs particles, `MASS[PDG]` and `WIDTH[PDG]`, where `PDG = 25,35,36,37` for h^0, H^0, A^0 and $H^\pm$, respectively. The inputs are complete, when `TAN(BETA)`, the ratio of the two Higgs vacuum expectation values, and `ALPHA`, the Higgs mixing angle, are specified.

Effective Higgs Couplings: the EHC describes the effective coupling of gluons to Higgs bosons via a top-quark loop. The gauge invariant interaction is given by

$$\mathcal{L}_{ggH}^{\rm eff} \quad = \quad g_{ggH}\frac{\alpha_S}{2\pi v}G_{\mu\nu}^{a}G_{a}^{\mu\nu}H \; ,$$

where $G_{\mu\nu}^a = \partial_\mu A_\nu^a - \partial_\nu A_\mu^a - gf^{abc}A_\mu^b A_\nu^c$ is the gluon field strength tensor. The effective coupling g_{ggH} can be calculated either for a finite top-quark mass or in the limit $m_t \to \infty$ using the switch `FINITE_TOP_MASS`={1,0}. The default is the latter choice.

Fouth Generation: the 4thGen adds a fourth family of quarks and leptons to the Standard Model. Their masses and widths are defined via the `MASS[PDG]` and `WIDTH[PDG]` switches, where `PDG = 7,8,17,18` for d_4, u_4, ℓ_4 and ν_4. A general mixing is implemented for both leptons and quarks, parametrised through three additional mixing angles and two additional phases, as described in [74]: `A_14`, `A_24`, `A_34`, `PHI_2` and `PHI_3` for quarks, `THETA_L14`, `THETA_L24`, `THETA_L34`, `PHI_L2` and `PHI_L3` for leptons. Both 4×4 mixing matrices expand upon their 3×3 Standard Model counter parts: the CKM matrix for quarks and the unit matrix for leptons. Both mixing matrices can be printed on screen with `OUTPUT_MIXING = 1`.
Per default, all particles are set unstable and have to be decayed into Standard Model particles within the matrix element or set stable via `STABLE[PDG] = 1`.

4.4.3 Process setup

In the section `(processes)` the process(es) to be evaluated by the matrix-element generator and process-specific settings are specified. These processes also serve as seeds for the event generation.

1. In setting up the processes it is necessary that the requested initial states are consistent with the settings in the `(isr)` section, otherwise SHERPA will terminate the run immediately. Therefore, elementary bunches like electrons have to be the initial state of the hard processes as well. For the case of proton bunches, the initial states can be either quarks or gluons.

2. To characterize the initial and final states of the processes, the PDG particle codes are used. Besides the elementary particles, the predefined particle containers can be used, see Sec. 4.4.2.

3. The syntax of calling a specific process is:

```
Process : 93 93 -> -11 12 93
Order electroweak : 2
Order strong : 1
End process
```

   Here all processes of the kind "hadron hadron $\to$ $e^+\nu_e$ jet" are initialized, where contributions originating from the production of heavy quarks have been left out.
   The parameters "`Order electroweak`" and "`Order strong`" restrict the number of electroweak or strong interactions occurring in single Feynman diagrams to exactly the numbers given. If these parameters are not specified all electroweak and strong contributions are taken into account.

4. Processes that differ only in the multiplicity of jets (or any other particle) can be specified using the following short notation:

```
Process : 93 93 -> -11 12 93{3}
Order electroweak : 2
End process
```

The number in curly brackets following a particle code specifies it's maximal multiplicity. The given example would generate all "hadron hadron $\to$ $e^+\nu_e$" processes with up to 3 extra jets in the final state and exactly 2 electroweak interactions.

5. Processes can be also specified by defining a production and subsequent decay processes. In this case SHERPA would generate matrix elements which feature full spin correlation to the final-state particles. The syntax is:

```
Process : 93 93 -> 6[a] -6[b]
Decay : 6[a] -> 5 24[c]
Decay : -6[b] -> -5 -24[d]
Decay : 24[c] -> -13 14
Decay : -24[d] -> 1 -2
Order electroweak : 0
Order strong : 2
End process
```

In the example a hadron–hadron initial state produces a top pair which then enters into a decay chain. The full process is equivalent to "hadron hadron $\to$ $b\bar{b}\mu\bar{\nu}_\mu d\bar{u}$". However in this setup only amplitudes that exactly contain the given intermediate particles are taken into account.
The qualifiers in square brackets must be given to allow for a unique assignment of decaying particles. Any character can be used, but it has to be unique for the specific process. Particle container are only allowed for the initial or final state of the full process, not for particles that are subject to further decay. The given electroweak and strong order only relates to the core process (hadron hadron $\to t\bar{t}$).
This treatment is applicable for processes which are dominated by the resonance of a given intermediate particle. The resulting decay products respect full spin correlation to their mothers and are distributed according their widths. Since the number of amplitudes to be evaluated is usually reduced drastically compared to the full process, the computational effort is significantly smaller and thus much higher final-state multiplicities can be considered.
Note that the treatment is an approximation to the full matrix element and only as good as the narrow width assumption for the intermediate particle. For some processes this might lead to gauge violations up to this order. The gauge violation displayed during the initialization of the process, however, can be much larger. This is no cause for concern and due to the fact that then the matrix element is evaluated only at a single phase-space point, in general far away from the mass-shells of the intermediate particles.
The short notation for different jet multiplicities using curly brackets can be applied here as well. If specified, all possible combinations of production and decay processes will be generated.

6. Replacing the keyword `Decay` by `DecayOS` within the decay chain treatment will cause intermediate particles to be generated exactly on their mass-shells. This treatment will neglect finite width effects of the decaying particle, but spin correlations are still treated properly. This option avoids the possible gauge dependence of the results described in item 5. and simplifies the phase space integration, leading to higher efficiencies. `Decay` and `DecayOS` can by combined in an arbitrary way within one process setup.

7. Additionally, scale choices and prefactors can be defined on a process-by-process base different from the global settings. The keywords look similar to the ones used in Sec. 4.4.1 and correspondingly have the same meaning:

   - "`Scale scheme :` ",
   - "`KFactor scheme :` " and
   - "`Fixed scale :` ".

8. To enable the multi-cut treatment described in [16] the user may set a process dependent value for the jet resolution cut y_{cut} via

   - "YCUT : ".

   Alternatively an individual selector data file can be specified:

   - "Selector file : ".

9. In case unweighted events are to be generated, the unweighting efficiency may be increased through setting the keyword

   - "Max_Epsilon : ".

   The given value specifies the fraction ϵ, for which weights are allowed to be above a "ϵ-dependent maximum weight" $\omega^{\epsilon}_{\text{max}}$. Meaningful values for Max_Epsilon are in the range of 10^{-3} to 10^{-4}. In any case it should be carefully checked that the considered distributions are not altered. To appropriately use this feature, it has to be ensured that the optimization of the phase-space integration has been completed by switching on the FINISH_OPTIMIZATION option, see Sec. 4.4.4. Corresponding weight histograms, which are used to determine $\omega^{\epsilon}_{\text{max}}$ are stored – if it has been provided by the user before – in the Results directory, see subdirectory WD_<process_id>. The first row of the histogram contains the decade logarithm of the weight, the second row the number of events in the corresponding bin.

10. The specification of an

    - "Enhance_Factor : "

    allows the enhancement of specific processes. E.g. a factor of 10 causes the process to produce 10 times more events than its fraction of the total number of events given by the corresponding cross section. This factor is stored in the output of the affected event. To regard for the enhancement a weight of 1/Enhance_Factor has to be applied in the analysis.

11. The keyword

    - "Integration_Error : "

    allows to specify the relative integration error on a process-by-process base. If stated it has priority over the ERROR setting inside the (integration) section and disables the FINISH_OPTIMIZATION option for the process.

12. If the tag

    - "Print_Graphs : "

    is specified, a *.tex file containing figures of all Feynman diagrams will be generated and stored with the process library files.

4.4.4 Integration

The (integration) section of the steering file allows specifications for the phase-space generation and optimization. The possible settings are:

1. With the variable ERROR the maximal relative integration error (statistical error of the MC integration) is determined to which a total cross section is evaluated. The parameter is also important for the generation of unweighted events, since before the actual event generation can start the phase space has to be integrated as accurately as requested by the variable ERROR. Additionally the performance of the MC integrator is optimized at this stage. However, the appropriate choice

option	comment
0	Rambo
2	HAAG
4 . . . 6	Multichannel

Table 7: Available integration algorithms within SHERPA and the corresponding values of the INTEGRATOR variable, which can be set in the (integration) section.

of the integration accuracy is clearly process and purpose dependent. For processes with many final-state particles, the usage of a very small integration error can cause very time-consuming calculations. In most cases an adequate choice seems to be of the order of 0.01 (1%).

2. The switch FINISH_OPTIMIZATION (if not declared its default value is set to Off) steers whether the phase-space integration is performed at least until the integrator is fully optimized or not. In the case of unweighted events this ensures the best unweighting efficiency, however, might become very time-consuming for processes with many particles in the final state. Typically a statistical accuracy of 0.002 . . . 0.005 is achieved after the full optimization.

3. With the variable INTEGRATOR the integration algorithm responsible for the final-state phase-space integration is selected; for the various available choices see Tab. 7.
   The default is the pure Rambo integrator, however, the authors recommend to employ one of the options 4 . . . 6, namely the multichannel integrators. The three given multichannel methods differ in the way the set of single channels is generated during the initialization run.

   - Option 4: building up the channels is achieved through respecting the peak structure given by the propagators. The algorithm works recursively starting from the initial state.
   - Option 5: this is an extension of option 4. In the case of competing peaks (e.g. a Higgs boson decaying into W^+W^-, which further decay), additional channels are produced to account for all kinematical configurations where one of the propagating particles is produced on its mass shell.
   - Option 6: in contrast to option 4 the algorithm now starts from the final state. The extra channels described in option 5 are produced as well. This has been optimized for configurations similar to the example given in Sec. 3.2.

   Note that the choice 4 . . . 6 only affects the channel generation. Once a channel set has been produced, say by using option 4, switching to option 5 or 6 will not be sufficient to obtain the new channel set. For this, the old channel set has to be removed in the Process directory. However, after a completed Vegas optimization (see the VEGAS switch description below) the performance of the three channel sets is quite similar in most cases.
   Considering processes with real graviton production in the framework of the ADD model automatically appropriate channels will be created when choosing one of the options 4 . . . 6, such that they correctly account for the required sum over the tower of external real gravitons.
   For pure (massless) QCD processes the most efficient choice is option 2, which employs the integrator HAAG [75], also refined by a VEGAS grid.

4. The VEGAS switch assigns whether an additional Vegas optimization is performed for the channels of the multichannel integrator. The authors strongly recommend to use this option, since the integration efficiency improves significantly – typically by a factor of 2 . . . 10. So, the default value is of course On.

In general, the best choice for the integration algorithm might depend on the considered process and the applied phase-space cuts.

Keyword/Variable	Particle 1	Particle 2	Minimal value	Maximal value
Energy	22		10.	10000.
ET	22		10.	10000.
PT	22		10.	10000.
Rapidity	22		-2.5	2.5
PseudoRapidity	22		-2.5	2.5
Mass	1	-2	75.	85.
Angle	11	22	-0.995	0.995
DeltaEta	93	93	3.2	10.
DeltaPhi	93	93	0.	1.5
DeltaR	90	93	0.2	10.
BeamAngle	11	0	-0.98	0.98
JetFinder			0.01	1.
ConeFinder			0.4	

Table 8: All possible types of cuts exemplified.

4.4.5 Phase-space cuts

SHERPA provides a number of predefined selectors on the matrix-element level to define kinematical cuts on external particles. These can be found and specified in the `(selector)` section of `Run.dat`.

1. Any selector is included through its keyword, followed by the particle(s) it acts on, and by the minimal and maximal value of the variable to cut on.

2. Particles are referred to by their PDG code. Note, the particle or anti-particle properties are explicitly considered, so specifying a selector for electrons requires to quote code `11` while for positrons `-11` has to be given. Apart from the single particle codes, flavour containers can be used as well, see Sec. 4.4.2.

3. Available predefined selectors, which can be used within SHERPA, are listed together with possible applications in Tab. 8. Note that in general n-particle selectors require the specification of n particles ($n = 1, 2, \ldots$).

4. The thresholds for all energy and mass selectors have to be given in GeV. Note that the `DeltaPhi` boundaries correspond to radian, while the angle selectors are defined in terms of the cosine of the angle between two particles. For the case of the `BeamAngle` selector, the notation is specific: here, with the second particle specifier, one can choose that the angle is taken between the particle given by the first specifier and the first or second incoming (beam) particle by using the option `0` or `1`, respectively.

5. For more information about selectors and how to apply them the reader is referred to the online documentation:

   http://projects.hepforge.org/sherpa/dokuwiki/documentation/matrix_elements/selectors

4.5 Parton showers

The keywords steering the parton showering, to be specified in the `(shower)` section, are:

1. `SHOWER_GENERATOR`, which specifies the shower module, responsible for jet evolution. Currently, the only valid value is `Apacic`.

Setting	Meaning
0	no radiation off resonances
1	radiation in the production process only
2	radiation in the decay process only
3	radiation in both production and decay

Table 9: Possible definitions of the SHOWER_MODE parameter.

2. FSR_SHOWER and ISR_SHOWER, which are the main switches to steer the shower generation, i.e. to turn on (=1) or off (=0) the final- and initial-state shower, respectively. Note that it is not necessary to turn on the showers to calculate cross sections with ISR; for this purpose, it is sufficient to turn on ISR_1 and/or ISR_2 in the (isr) section of the steering file. Contrary, in case of event generation for QCD processes at hadron colliders, it should be stressed that it is essential to have turned on both showers in order to obtain physically meaningful results.

3. The parton showers in the initial and final state terminate when no more parton can be emitted with a transverse momentum squared larger than the values given by IS_PT2MIN and FS_PT2MIN. The default choices of the cut-offs are

$$p_\perp^2(\mathrm{IS}) \;=\; 4\,\mathrm{GeV}^2 \qquad\text{and}\qquad p_\perp^2(\mathrm{FS}) \;=\; 1\,\mathrm{GeV}^2\,.$$

Since version 1.0.9, SHERPA provides the option to include QED final-state radiation off charged leptons and quarks. This option can be disabled through FS_QED_SCHEME=0. The cut-off for emissions off quarks is equal to the final-state QCD shower cut-off. For emissions off leptons a separate cut-off is specified via FS_PT2MIN_QED. The current default is 0.0025 GeV^2.
Since version 1.1 SHERPA also provides the simulation of QCD radiation off intermediate resonances in the narrow width approximation. This feature is enabled by default and can be disabled, if desired. The corresponding parameter is SHOWER_MODE, which is required to take one of the values listed in Tab. 9. A proper matching of different jet multiplicities both in the production and the decay processes is obtained in the SHERPA framework by applying the CKKW prescription separately for production and decays. For this to work, the production process of resonances has to be equipped with corresponding decays in the (processes) part of the paramter file, see Sec. 4.4.3 and also Sec. 7.2.

4.6 Multiple Interactions

So far multiple interactions can **only** be generated for proton–proton collisions. Related parameters, which are set in the (mi) section, are the following:

1. The variable MI_HANDLER specifies the multiple interaction handler. If multiple interactions shall be switched on, the setting is Amisic.

2. The processes, which are to be generated in multiple interactions are given by CREATE_GRID followed by the specification of the process employing a similar nomenclature as in the (processes) section of Run.dat. The tag MI_SCALE_SCHEME sets the scale scheme, while MI_K_FACTOR_SCHEME sets the K factor scheme to be used for these processes, cf. the description of the (me) group of parameters, see Sec. 4.4.1. PS_ERROR gives the error with which the phase-space integrator calculates the cross section for the processes.

3. If REGULATE_XS is turned on, trivial regularization of the cross section is applied. XS_REGULATOR gives the type of the regulator to use. Currently there is only one option, QCD_Trivial, which leads to the multiplication of the differential cross section by $p_\perp^4/(p_\perp^2+p_{\perp 0}^2)$ with $p_{\perp 0}$ given by XS_REGULATION.

4. SCALE_MIN sets the minimum scale $Q_{\rm min}$ for which events are to be generated. This scale may be rescaled with the use of an exponent α, which can be set through RESCALE_EXPONENT, and a reference energy scale $E_{\rm ref}$ made available through REFERENCE_SCALE. The rule that determines the cutoff scale is then $Q_{\rm min} \cdot (E_{\rm cms}/E_{\rm ref})^{\alpha}$.

5. The tag PROFILE_FUNCTION is used to specify the name of the profile function for the hadron profile in impact parameter space. The available options are Exponential, Gaussian and Double_-Gaussian. The parameters of the double Gaussian are set through PROFILE_PARAMETERS.

4.7 Fragmentation

There are, broadly speaking, two options of how SHERPA handles the transition of quarks and gluons into primordial hadrons (hadronization) and their decay:

- In previous versions the only way of dealing with this was to invoke corresponding routines of PYTHIA, invoking the Lund model of string fragmentation and PYTHIA's native hadron decay routines. Although now the "Lund"-option is disfavoured, it will still be kept in SHERPA for backward compatibility and for some consistency checks.

- From version 1.1 on, the default choice will be to rely entirely on two internal modules, AHADIC++ and HADRONS++ for hadronization and hadron decays.

All of the following specifications can be done in the (fragmentation) section of the steering file. The keyword FRAGMENTATION (default value Ahadic), steers whether the fragmentation is switched on and performed by the internal cluster fragmentation module AHADIC++ [76], by an interface to the corresponding routines of the Lund string hadronization of PYTHIA [50]), indicated by Lund), or whether the fragmentation is completely switched (Off). The treatment of hadron and τ decays is specified by DECAYMODEL. Its allowed values are either the default choice Hadrons, which renders the HADRONS++ module responsible for performing the decays, or as alternative, the interface to Pythia can be invoked by setting this parameter to Lund. For the former option, the reader is referred to Sec. 4.8 for a more detailed discussion.
Please note that it is absolutely not advisable to use one fragmentation model with another model for the hadron decays, since there is quite an intimate relation between two - for instance, AHADIC++ knows and allows for many more primordial hadrons than PYTHIA does, which would obviously lead to problems, if they were created in AHADIC++ and left to PYTHIA for decays.

4.7.1 Using the PYTHIA routines

The PYTHIA routines have been made available through an interface to the corresponding Fortran code, with the option to steer PYTHIA parameters through SHERPA. The Lund parameters can be collected in a file set by the parameter LUND_FILE (default is Lund.dat), if "Lund" is chosen as the method of choice for both fragmentation and hadron decays. Driven in this option, SHERPA is more or less ignorant about hadron decays, with the exception of τ-decays, which have been supplemented early in the development of the SHERPA event generator.
The coarse features of the string breakup in the Lund model are characterized by three parameters, Lund-a through PARJ(41) (data file value: 0.431), Lund-b through PARJ(42) (data file value: 0.878), and Lund-σ through PARJ(21) (data file value: 0.336). If the data-file parameters are not set, SHERPA employs the default values in PYTHIA.
More Pythia parameters can be added to the Lund.dat file using the PYTHIA nomenclature in the common-blocks, e.g. MSTJ(11) for changing the fragmentation function for heavy quarks. Please note that particle decays cannot directly be disabled there. Instead, the stable flag for the corresponding hadron needs to be used.

Parameter	Description	Default
`STRANGE_FRACTION`	$\mathcal{P}_s$	0.37
`BARYON_FRACTION`	$\mathcal{P}_d$	0.2
`P_{QQ_1}/P_{QQ_0}`	$\mathcal{P}_1$	0.25
`P_{QS}/P_{QQ}`	$\mathcal{P}_{qs}$	0.40
`P_{SS}/P_{QQ}`	$\mathcal{P}_{ss}$	0.05

Table 10: The global flavour weights employed in gluon splitting and cluster decays in AHADIC++.

4.7.2 Physics and parameters of SHERPA's cluster hadronization

In AHADIC++ a version of the cluster hadronization model [54, 60] built on ideas of [62] has been implemented. There, the hadronization of quarks and gluons proceeds in three steps:

1. First of all, the four-momenta of all partons are boosted and scaled such that all of them now reside on current-mass shells, with $m_u = m_d = 0.3$ GeV, $m_s = 0.5$ GeV, $m_c = 1.8$ GeV and $m_b = 5.2$ GeV. In addition, there are further states, diquarks, which are for the description of baryons. Their masses are: Typically given by the sum of the two quarks' current masses, which constitute them, i.e. $m_{qq'} = m_q + m_{q'}$. In the following, quarks and diquarks, will summarily be called "flavours".
   In contrast to the original version of the cluster model [54], also implemented in HERWIG++, the gluon does not acquire any mass. This is possible, since the forced decay of the gluon into flavour states is performed in a dipole-inspired framework, where the colour partner accounts for the recoil of the massless gluon decaying into the massive flavours.
   In these non-perturbative gluon decays into flavours, the dipole-inspired transverse momentum $p_\perp$ is bounded from above by some value $p_\perp^{\max}$, which is accessible through the keyword `PT_MAX`; its default value is 1 GeV; typically it should be of the order of the parton-shower cut-off scale. In addition, for all decays, a non-perturbative α_s is encoded in AHADIC++, with different choices:

   - `constant`:
     In this choice, a further parameter, $p^2_{\perp,0}$ regulates the form of α_s. The strong coupling constant remains perturbative at scales above $p^2_{\perp,0}$ and it is frozen to the perturbative value $\alpha_s(p^2_{\perp,0})$ for scales below. This parameter can be set by the user through `PT^2_0`, with default value of $p^2_{\perp,0} = 1$ GeV^2.
   - `GDH_inspired`:
     In addition, recently, the option of a non-perturbative strong coupling constant as measured through the GDH sum rule has been added. There, the form of [77] with all parameters has been implemented. It is accessible by setting `PT^2_0` to 0.

   For the production of the flavours, the available phase space leads to kinematic enhancements of lighter flavour pairs in dependence on the mass of the dipole (i.e. gluon and spectator pair). It is supplemented by constant weight factors, $\mathcal{P}_q$ also used in cluster decays to hadrons. They are given in Tab. 10.
   At the end of this step, colour-neutral clusters, consisting of two flavours have emerged. In contrast to the original version of the cluster model, at this stage, already clusters with quantum numbers of baryons have been produced, in rare cases even clusters consisting of two diquarks appear.

2. All clusters are checked whether they fall into one or both of two regimes, namely

   - the regime where mass of the cluster is of the order of the mass of the heaviest hadron with a flavour wave component identical to the flavour content of the cluster, i.e. where $m_c < m_h + \delta_h$,

Parameter	Multiplet content	Default
MULTI_WEIGHT_L0R0_PSEUDOSCALARS	$\pi^{0,\pm}$, η, $K^{\pm}$, K_S, K_L, η'	1.0
MULTI_WEIGHT_L0R0_VECTORS	$\rho(770)$, $\omega(782)$, $K^*(892)$, $\phi(1020)$	0.55
MULTI_WEIGHT_L0R0_TENSORS2	$a_2(1320)$, $f_2(1270)$, $K_2^*(1430)$, $f_2'(1525)$	0.12
MULTI_WEIGHT_L1R0_SCALARS	$a_0(980)$, $f_0(980)$, $K_0^*(1430)$, $f_0(1370)$	0.1
MULTI_WEIGHT_L1R0_AXIALVECTORS	$b_1(1235)$, $h_1(1170)$, $K_1(1270)$, $h_1(1380)$	0.05
MULTI_WEIGHT_L1R0_TENSORS2	$\pi_2(1670)$, $\eta_2(1645)$, $K_2(1770)$, $\eta_2(1870)$	0.01
MULTI_WEIGHT_L2R0_VECTORS	$a_1(1260)$, $f_1(1285)$, $K_1(1400)$, $f_1(1420)$	0.05
MULTI_WEIGHT_L0R0_N_1/2	p, n, $\Sigma^{0,\pm}$, Λ, $\Xi^{0,-}$	1.
MULTI_WEIGHT_L0R0_N*_1/2	$N(1440)$, $\Sigma(1660)$, $\Lambda(1600)$, $\Xi(1690)$	0.05
MULTI_WEIGHT_L0R0_DELTA_3/2	$\Delta(1232)$, $\Sigma(1385)$, $\Xi(1530)$, Ω^-	0.15

Table 11: The $SU(3)_F$-hadron content of different multiplets. For simplicity, charge assignments have been suppressed where possible.

- or the regime where the mass of the cluster is lighter than the summed mass of the two heaviest hadrons it can decay into by adding a flavour-antiflavour pair to its flavour content, i.e., where $m_c < m_h + m_{h'} + \delta_{hh'}$.

In order to steer this classification further, two offsets δ_h and $\delta_{hh'}$ have been introduced, which are accessible through the keywords TRANSITION_OFFSET and DECAY_OFFSET, respectively, with default values -1.0 and 0.2.
If a cluster is outside both regimes, it will decay further into two clusters. This proceeds by the flavour-antiflavour pair firstly emitting a gluon with transverse momentum choices as above, before then the heavier of the two colour dipoles, made of one of the two flavours and the gluon decays further, invoking the routines of step 1.

3. If a cluster falls into one or both transition regimes, probabilities for the corresponding allowed transitions into one or two clusters are calculated. For the transitions, it is given by

$$\mathcal{P}_{M_c\to m_h} = \frac{\kappa}{2M_c}\,\frac{(M_c\Gamma_h)^2}{(M_c^2-m_h^2)^2+(m_h\Gamma_h)^2}\;\times\;|\Psi_{q\bar{q}}(h)|^2\,, \tag{8}$$

where m_h and Γ_h are the mass and width of the hadron in question, and where $\Psi_{q\bar{q}}(h)$ denotes the corresponding flavour wavefunction components of the hadron (e.g. $\Psi_{u\bar{u}}(\pi^0) = 1/\sqrt{2}$), while for decays it reads

$$\begin{aligned}\mathcal{P}_{M_c\to m_1 m_2} \;&=\; \frac{\sqrt{[M_c^2-(m_1+m_2)^2][M_c^2-(m_1-m_2)^2]}}{16\pi M_c^3}\left[\frac{4m_1m_2}{M_c^2}\right]^\alpha\\ &\quad\times|\Psi_{q\bar{q}'}(h_1)|^2|\Psi_{q'\bar{q}}(h_2)|^2\cdot\mathcal{P}_{q'}\,.\end{aligned} \tag{9}$$

Here, the parameter κ (to be set by C->H_TRANSITION_FACTOR, default value 1) can be used to change the relative rate of transitions and decays. In the decay weights α steers the preference of the decays for heavier hadrons; it can be set by C->HH_DECAY_EXPONENT (default value: 8). In addition, $\mathcal{P}_q$ is the probability to pop flavour q in cluster decays. In each channel, the weights are further modified by including weights for the various hadron multiplets, see Tab. 11. In addition, there is a singlet suppression factor SINGLET_SUPPRESSION to reduce transition probabilities to singlet-octet mixed states in the meson sector.
Once a decay into two hadrons is chosen, the decay kinematics have to be fixed. This is achieved by giving the hadrons a transverse momentum w.r.t. the original flavour axis distributed according to

$$p_\perp \;=\; |\vec{p}|\sin\theta \qquad \text{where} \qquad \sin\theta \;=\; \#^\eta \tag{10}$$

where the parameter η is given by `C->HH_DECAY_THETA_EXPONENT` and where # is an uniformly chosen random number between 0 and 1.

The flavour production probabilities can be steered by the parameters in table Tab. 10. The baseline are u and d quarks, coming with relative strength 1. Strange quarks have a weight

$$\mathcal{W}_s \;=\; \mathcal{P}_s\,, \tag{11}$$

given by `STRANGE_FRACTION`. For the diquarks, things are more complicated. Their relative weights are given by

$$\begin{aligned}
\mathcal{W}_{(ud)_0} &= \mathcal{P}_d && (12)\\
\mathcal{W}_{(ud)_1} = \mathcal{W}_{(uu)_1} = \mathcal{W}_{(dd)_1} &= \mathcal{P}_d\mathcal{P}_1 && (13)\\
\mathcal{W}_{(us)_0} = \mathcal{W}_{(ds)_0} &= \mathcal{P}_d\mathcal{P}_{qs} && (14)\\
\mathcal{W}_{(us)_1} = \mathcal{W}_{(ds)_1} &= \mathcal{P}_d\mathcal{P}_{qs}\mathcal{P}_1 && (15)\\
\mathcal{W}_{(ss)_1} &= \mathcal{P}_d\mathcal{P}_{ss}\mathcal{P}_1\,. && (16)
\end{aligned}$$

4.8 Hadron decays

HADRONS++ is the module within the SHERPA framework which is responsible for treating hadron and τ decays. It contains decay tables with branching ratios for ≈ 2500 decay channels, of which many have their kinematics modelled according to a matrix element with corresponding form factors. Especially decays of the τ lepton and heavy mesons have form factor models similar to dedicated codes like Tauola [78] and EvtGen [79].
Some general switches which relate to hadron decays can be adjusted in the `(fragmentation)` section:

- `DECAYPATH`
  The path to the parameter files for the hadron and τ decays (default: `Decaydata/`). It is important to note that the path has to be given relative to the current working directory. If it doesn't exist, the default Decaydata directory (`$prefix/share/SHERPA-MC/Decaydata`) will be used.

- Hadron properties like mass, width, stable/unstable and active can be set in the `(fragmentation)` section in full analogy to the settings for fundamental particles in the `(model)` section (cf. Sec. 4.4.2).

- `MASS_SMEARING = [0,1,2]` (default: 1)
  Determines whether particles entering the hadron decay event phase should be put off-shell according to their mass distribution. It is taken care that no decay mode is suppressed by a potentially too low mass. While HADRONS++ determines this dynamically from the chosen decay channel, for `Pythia` as hadron decay handler its `w-cut` parameter is employed. Choosing option 2 instead of 1 will only set unstable (decayed) particles off-shell, but leave stable particles on-shell.

- `MAX_PROPER_LIFETIME = [mm]`
  Parameter for maximum proper lifetime $c\tau$ (in mm) up to which particles are considered unstable. If specified, this will make long-living particles stable, even if they are set unstable by default or by the user.

Many aspects of the above mentioned "Decaydata" can be adjusted. There exist three levels of data files, which are explained in the following sections. As with all other setup files, the user can either employ the default "Decaydata" in `$prefix/share/SHERPA-MC/Decaydata`, or overwrite it (also selectively) by creating the appropriate files in the directory specified by `DECAYPATH`.

4.8.1 HadronDecays.dat

`HadronDecays.dat` consists of a table of particles that are to be decayed by Hadrons++. Note: Even if decay tables exist for the other particles, only those particles decay that are set unstable, either by default, or in the model/fragmentation settings. It has the following structure, where each line adds one decaying particle:

`<kf-code>`	`->`	`<subdirectory>/`	`<filename>.dat`
↓ decaying particle		↓ path to decay table	↓ decay table file
default names:		`<particle>/`	`Decays.dat`

It is possible to specify different decay tables for the particle (positive kf-code) and anti-particle (negative kf-code). If only one is specified, it will be used for both particle and anti-particle.
If more than one decay table is specified for the same kf-code, these tables will be used in the specified sequence during one event. The first matching particle appearing in the event is decayed according to the first table, and so on until the last table is reached, which will be used for the remaining particles of this kf-code.
Additionally, this file may contain the keyword `CREATE_BOOKLET` on a separate line, which will cause Hadrons++ to write a LaTeX document containing all decay tables.

4.8.2 Decay table files

The decay table contains information about outgoing particles for each channel, its branching ratio and eventually the name of the file that stores parameters for a specific channel. If the latter is not specified Hadrons++ will produce it and modify the decay table file accordingly.
Additionally to the branching ratio, one may specify the error associated with it, and its source. Every hadron is supposed to have its own decay table in its own subdirectory. The structure of a decay table is

`{kf1,kf2,kf3,...}`	`|`	`BR(ΔBR)[Origin]`	`|`	`<filename>.dat`
↓ outgoing particles		↓ branching ratio		↓ decay channel file

It should be stressed here that the branching ratio which is explicitly given for any individual channel in this file is **always used** regardless of any matrix-element value.

4.8.3 Decay channel files

A decay channel file contains various information about that specific decay channel. There are different sections, some of which are optional:

- ```
  <Options>
      AlwaysIntegrate = 0
      CPAsymmetryC = 0.0
      CPAsymmetryS = 0.0
  </Options>
  ```

  - `AlwaysIntegrate = [0,1]` For each decay channel, one needs an integration result for unweighting the kinematics (see below). This result is stored in the decay channel file, such that the integration is not needed for each run. The AlwaysIntegrate option allows to bypass the stored integration result, and do the integration nonetheless (same effect as deleting the integration result).
  ```

  - `CPAsymmetryC/CPAsymmetryS` If one wants to include time dependent CP asymmetries through interference between mixing and decay one can set the coefficients of the cos and sin terms respectively. HADRONS++ will then respect these asymmetries between particle and anti-particle in the choice of decay channels.

-
```
<Phasespace>
  1.0 MyIntegrator1
  0.5 MyIntegrator2
</Phasespace>
```
  Specifies the phase-space mappings and their weight.

-
```
<ME>
  1.0 0.0 my_matrix_element[X,X,X,X,X,...]
  1.0 0.0 my_current1[X,X,...] my_current2[X,X,X,...]
</ME>
```
  Specifies the matrix elements or currents used for the kinematics, their respective weights, and the order in which the particles (momenta) enter them. For more details, the reader is referred to [80].

-
```
<my_matrix_element[X,X,X,X,X,...]>
  parameter1 = value1
  parameter2 = value2
  ...
</my_matrix_element[X,X,X,X,X,...]>
```
  Each matrix element or current may have an additional section where one can specify needed parameters, e.g. which form factor model to choose. Each parameter has to be specified on a new line as shown above. Available parameters are listed in [80]. Parameters not specified get a default value, which might not make sense in specific decay channels. One may also specify often needed parameters in `HadronConstants.dat`, but they will get overwritten by channel specific parameters, should these exist.

-
```
<Result>
  3.554e-11 6.956e-14 1.388e-09;
</Result>
```
  These last three lines have quite an important meaning. If they are missing, HADRONS++ integrates this channel during the initialization and adds the result lines. If this section exists though, and `AlwaysIntegrate` is off (the default value, see above) then HADRONS++ reads in the maximum for the kinematics unweighting.

  Consequently, if some parameters are changed (also masses of incoming and outgoing particles) the maximum might change such that a new integration is needed in order to obtain correct kinematical distributions. There are two ways to enforce the integration: either by deleting the last three lines or by setting `AlwaysIntegrate` to 1. When a channel is re-integrated, HADRONS++ copies the old decay channel file into `.<filename>.dat.old`.

4.8.4 HadronConstants.dat

`HadronConstants.dat` may contain some globally needed parameters (e.g. for neutral meson mixing, see [80]) and also fall-back values for all matrix-element parameters which one specifies in decay channel files. Here, the `Interference_X = 1` switch would enable rate asymmetries due to CP violation in the interference between mixing and decay (cf. Sec.), and setting `Mixing_X = 1` enables explicit mixing in the event record according to the time evolution of the flavour states. By default, all mixing effects are turned off.

- Mixing parameters ($x = \frac{\Delta m}{\Gamma}$, $y = \frac{\Delta\Gamma}{2\Gamma}$) with some example values

```
x_K = 0.946
y_K = -0.9965
qoverp2_K = 1.0
Interference_K = 0
Mixing_K = 0

x_D = 0.0
y_D = 0.0
qoverp2_D = 1.0
Interference_D = 0
Mixing_D = 0

x_B = 0.776
y_B = 0.0
qoverp2_B = 1.0
Interference_B = 1
Mixing_B = 0

x_Bs = 30.0
y_Bs = 0.155
qoverp2_Bs = 1.0
Interference_Bs = 0
Mixing_Bs = 0
```

4.8.5 Further remarks

Spin correlations: the spin correlation algorithm is implemented, also for decays of τ's produced in the signal process. It can be switched on through the keyword `SPIN_CORRELATIONS` in the `(run)` section, see Sec. 4.2 for more details.

Adding new channels: if new channels are added to HADRONS++ (choosing isotropic decay kinematics) a new decay table must be defined and the corresponding hadron must be added to `HadronDecays.dat`. The decay table merely needs to consist of the outgoing particles and branching ratios, i.e. the last column (the one with the decay channel file name) can safely be dropped. By running SHERPA it will automatically produce the decay channel files and write their names in the decay table.

Some details on τ decays: τ decays are treated within the HADRONS++ framework, even though the τ is not a hadron. As for many hadron decays, the hadronic τ decays have form factor models implemented, for details the reader is referred to [80].

4.9 Soft photon radiation

The PHOTONS++ module in the SHERPA framework handles higher order QED corrections to hadron decays. It is based on the Yennie-Frautschi-Suura (YFS) formalism that resums and exponentiates all contributions in the limit of vanishing photon energy and explicitly cancels all infrared divergences caused by real and virtual emissions. The formalism further allows for the inclusion of hard real and virtual emission corrections in a perturbative series in α_{QED}. For details see [65].
PHOTONS++ is steered by three switches, all of which have to be located `(fragmentation)` section. Their meaning is the following:

1. The keyword `YFS_MODE={0,1,2}` determines the mode of operation of PHOTONS++. `YFS_MODE=0` switches PHOTONS++ off. Consequently, no hadron decay will be corrected for soft or hard photon emission. `YFS_MODE=1` sets the mode to "soft only", meaning soft emissions will be treated correctly to all orders but no hard emission corrections will be included. With `YFS_MODE=2` these hard emission corrections will also be included up to $\mathcal{O}(\alpha_{\rm QED})$. This is the default setting.

2. The switch `YFS_USE_ME={0,1}` tells PHOTONS++ how to correct hard emissions to $\mathcal{O}(\alpha_{\rm QED})$. If `YFS_USE_ME=0`, then PHOTONS++ will use collinearly approximated real emission matrix elements. Virtual emission matrix elements of $\mathcal{O}(\alpha_{\rm QED})$ are ignored. If, however, `YFS_USE_ME=1`, then exact real and/or virtual emission matrix elements are used wherever possible. These are presently available for $V \to FF$, $V \to SS$, $S \to FF$, $S \to SS$ type decays and leptonic τ decays. For all other decay types general collinearly approximated matrix elements are used. In both approaches all hadrons are treated as point-like objects. The default setting is `YFS_USE_ME=1`. This switch is only effective if `YFS_MODE=2`.

3. `YFS_IR_CUTOFF` sets the infrared cut-off dividing the real emission in two regions, one containing the infrared divergence, the other the "hard" emissions. This cut-off is currently applied in the rest frame of the multipole of the respective decay. It also serves as a minimum photon energy in this frame for explicit photon generation for the event record. In contrast, all photons below with energy less than this cut-off will be assumed to have negligible impact on the final-state momentum distributions. The default is `YFS_IR_CUTOFF=1E-3` (GeV). Of course, this switch is only effective if PHOTONS++ is switched on, i.e. `YFS_MODE={1,2}`.

4.10 Event analysis

SHERPA has a built-in framework, which is capable of carrying out simple analyses of the generated events. It is possible to restrict the analysis on single event phases: the level of the hard matrix element, the parton-shower evolution level, the hadronization level. On the other hand one can do some or all of these subanalyses in one single run.
The main switch to enable this analysis framework is `ANALYSIS`, which can be set in the file `Run.dat` or from the command line. `ANALYSIS=1` and `ANALYSIS=0` correspond to the analysis framework being switched on and off, respectively. All analysis output will be stored in the directory specified through `ANALYSIS_OUTPUT`, that will be created on run time in the current working directory.

The actual results of the analysis are histograms, which are written to disk as text files. These files have the following form:

```
10 102 -0.221849 1.77815 0.004  0  100000
-0.221849  0
-0.201849  0.001
      .        .
      .        .
      .        .
-0.181849  0.0015
-0.161849  0.001
-0.141849  0.001
```

where the first line gives (in order of appearance) the type of the histogram, the number of bins, the lower and the upper range of the histogram, the underflow and the overflow weight and the number of entries. The type of the histogram is `10` in case the histogram is binned logarithmically; if it is binned linearly the histogram type is `0`. Each of all the other lines gives the lower edge of the corresponding bin and its entry. If error storage is enabled an additional item specifies the Monte Carlo error of the entry.

By default, the analysis setup will be read from the file `Analysis.dat`. There are example files for the `LEP91` and `Tevatron1800` setup of the SHERPA distribution. It is possible to specify another input file name by setting the variable `ANALYSIS_DATA_FILE` in `Run.dat` with this name.
The syntax of the setup file is as follows

1. Each analysis specification must be enclosed between the statements `BEGIN_ANALYSIS` and `END_ANALYSIS`. An event phase for the analysis must be specified through the keyword `LEVEL`. The possible phases are `ME`, `Shower` and `Hadron`. To have these or some of these analyses in one run, the corresponding values are simply concatenated by a space.

2. An output subdirectory for the analysis results may be given via `PATH_PIECE <directory>`. If a directory with the corresponding name does not exist yet, SHERPA will create it, if there are sufficient permissions on the selected file system.

3. The next step is to specify cuts to be applied on the final-state particle list. If any of these cuts is not passed the event enters the analysis as a nil event. There are several possibilities to restrict the final state. To obtain a complete list of selectors including the syntax for their setup SHERPA provides the command line option `SHOW_ANALYSIS_SYNTAX=1`. Running SHERPA with this command will stop the program after the printout of the syntax for the analysis setup. The interested user is kindly referred to inform about the various selectors by scanning the generated output. Each list entry containing the tag `Sel` in its name is in fact a selector. A special selector is `Trigger`.

4. Finally the observables have to be specified. For the list of observables available in SHERPA, the user is kindly asked to inform about them using the `SHOW_ANALYSIS_SYNTAX=1` option described above. To get used to the syntax specifying the analysis setup, the analysis example file for the `Tevatron1800` setup of the SHERPA distribution may serve as a helpful guide.

5. At the end of the event generation run, there will be created a histogram output file for each observable. It resides in the directory with the name of the corresponding analysis phase within the subdirectory given by `PATH_PIECE`, which itself resides in the directory given by `ANALYSIS_OUTPUT`. The filename depends on the observable itself. For the one- and two-particle observables, it is a concatenation of the observable name and the name of the particle (or the names of both particles in the case of a two-particle observable) for which the analysis has been carried out. This is preceded by the default filename extension `dat`. In any case the filename is suggestive for the data contained in the file.

6. Within the analysis setup file `C`-style comments such as `// <comment>` are supported.

It is possible to specify multiple analyses in the setup file. These may, e.g., contain different restrictions on the final state. In general it is however necessary to specify different output directories for different analyses, in order to avoid similar histograms being overwritten.

5 The SHERPA method of merging matrix elements and parton showers

5.1 Improved multijet final-state predictions

For a large fraction of LHC final states, the application of reconstruction algorithms will lead to the identification of several hard jets. A major task is to distinguish whether such events are signals for new physics or just manifestations of SM physics. Related calculations therefore need to describe as accurately as possible both the full matrix element for the underlying hard processes as well as the subsequent evolution and conversion of the hard partons into jets of hadrons. Several scales therefore determine the thorough development of the event. This makes it difficult to unambiguously disentangle the components, which belong to the hard process from those of the hard-parton evolution. Given an n jet event of well separated partons, its jet structure is retained when emitting a further collinear or soft parton only. An additional hard, large-angle emission however gives rise to an extra jet changing the n to an $n+1$ final state. The merging scheme has to define, on an event-by-event basis, which possibility has to be followed. Its primary goals are therefore to avoid double counting by preventing events to appear twice, i.e. once for each possibility, as well as dead regions by generating each configuration only once and using the appropriate path.

Different schemes have been proposed and exist. The MLM scheme, has been developed using a geometric analysis of the unconstrained radiation pattern in terms of cone jets to generate the inclusive samples [21, 81]. The CKKW scheme or CKKW merging algorithm has been introduced first for electron–positron annihilation into jets in [14]. Its extension to hadronic processes has been discussed in [15] and the approach has been validated for several cases, for e^+e^- collisions into hadrons [16, 30], for the production of single vector bosons at the Fermilab Tevatron [17] and the CERN LHC [18] and for W^+W^- production at hadron colliders, see [19]. A reformulation of CKKW to a merging procedure in conjunction with a dipole shower (CKKW-L) has been presented in [27]. In a number of works, all these different algorithms have been implemented in different variations on different levels of sophistication in conjunction with various matrix-element generators or already in full-fledged event generators. Again, common to all schemes is that sequences of tree-level multileg matrix elements with increasing final-state multiplicity are merged with parton showers to yield a fully inclusive sample with no double counting and correct at leading logarithmic accuracy.

Given this introduction, the facts about the SHERPA merging algorithm can be summarized as follows: SHERPA provides a fully general (i.e. largely process-independent) implementation [16] of the CKKW merging approach [14, 15]. Supplemented by a multiple-interactions description, which respects the jet-production scales of the primary, leading, process, this implementation guarantees an improved treatment of multijet production leading to more realistic predictions of jet observables. This is one of the key features of SHERPA. The idea underlying CKKW is to divide the phase space of partonic emissions according to a k_T measure [82–84] into a regime of jet production, described by appropriate matrix elements, and a regime of jet evolution, described by parton showering. The separation is defined by the merging scale, denoted by $Q_{\rm cut}$. The matrix elements are reweighted by α_s coupling factors and terms that arise from analytic Sudakov form factors. The acceptance/rejection of jet configurations is realized according to this reweighting. Then, each hard parton of the reweighted matrix-element final state undergoes vetoed parton showering, i.e. starting from the scale, where this parton appeared first, any new emission that would give rise to an extra jet is vetoed. In this way, the "unphysical" dependence on the, in principle, "arbitrary" separation scale $Q_{\rm cut}$ regularizing the matrix elements is almost eliminated and the accuracy of the parton shower is preserved.[15] Any unavoidable residual dependence can be used to tune less singular terms to obtain optimal agreement with data.

5.2 The algorithm implemented in SHERPA

In SHERPA the CKKW merging of matrix elements and parton showers is accomplished as follows:

1. All cross sections σ_k for processes with $k = 0, 1, \ldots, N$ extra partons are calculated with the constraint that the matrix-element final states pass the jet criteria. They are determined by a k_T measure and the minimal distance is set by the actual merging scale $Q_{\rm cut}$. Beyond that, $Q_{\rm cut}$ also acts as a regulator setting the factorization (PDF) as well as the renormalization (α_s) scales of the matrix-element calculations. The k_T measure used for the jet identification in electron–positron collisions can be written as

$$y_{ij} \; = \; \frac{2\min\{E_i^2, E_j^2\}}{S}(1-\cos\theta_{ij})\,, \tag{17}$$

and quantifies the k_T distance between the final-state particles i and j. The jet cuts are satisfied

[15] The cancellation of the dependence on the separation scale $Q_{\rm cut}$ has been proven analytically in the e^+e^- case up to next-to-leading logarithmic accuracy [14].

if $y_{ij} > y_{\rm cut} = Q_{\rm cut}/\sqrt{S}$. For hadron–hadron collisions, the k_T scheme is employed, which defines two final-state particles to belong to two different jets, if their relative transverse momentum squared

$$Q_{ij}^2 \;=\; 2\min\left\{p_T^{(i)},\, p_T^{(j)}\right\}^2 \frac{\left[\cosh(\eta^{(i)}-\eta^{(j)})-\cos(\phi^{(i)}-\phi^{(j)})\right]}{D^2} \tag{18}$$

is larger than the critical value $Q_{\rm cut}^2$. In addition, the transverse momentum of each jet has to be larger than the merging scale $Q_{\rm cut}$. The magnitude D, which is of order 1, is a parameter of the jet algorithm, see [85].

2. Processes of fixed parton multiplicity are chosen with probability $\sigma_k/\sum_{l=0}^{N}\sigma_l$. The event's hard process is picked from the list of partonic processes having the desired multiplicity and according to their particular cross-section contributions. All particle momenta are distributed respecting the correlations encoded in the matrix elements. Merged samples therefore fully include lepton-jet and jet–jet correlations correlations up to N extra jets.

3. The parton configuration of the matrix element has to be analyzed to eventually accomplish the reweighting. The partons are clustered backwards according to the same k_T jet clustering algorithm used for the regularization of the final-state phase space of the matrix elements (cf. first item). The clustering is guided by only physically allowed parton combinations and automatically yields the nodal k_T values Q_{ij} of each parton emission. It is stopped after a $2 \to 2$ configuration (a core process) has been identified. In fact, this "cluster-backwards" procedure constructs a limit of leading logarithmic accuracy of the full radiation pattern, i.e. determines a possible parton-shower history. The sequence of clusterings can thus be taken as a pseudo shower configuration, off which the event's evolution will be properly continued by parton showering.

4. The reweighting proceeds according to the reconstructed shower history. For the strong-coupling weight, the identified nodal k_T values are taken as scales in the strong-coupling constants and replace the predefined choice of the initial generation. The Sudakov weight attached to the matrix elements accounts for having no further radiation resolvable at $Q_{\rm cut}$. The NLL Sudakov form factors employed, cf. [82], are defined by

$$\Delta_q(Q,Q_0) \;=\; \exp\left\{-\int\limits_{Q_0}^{Q} dq\,\Gamma_q(Q,q)\right\}\,, \tag{19}$$

$$\Delta_g(Q,Q_0) \;=\; \exp\left\{-\int\limits_{Q_0}^{Q} dq\,[\Gamma_g(Q,q)+\Gamma_f(q)]\right\}\,, \tag{20}$$

where $\Gamma_{q,g,f}$ are the integrated splitting functions for $q \to qg$, $g \to gg$ and $g \to q\bar{q}$ $(f\bar{f})$, given by

$$\Gamma_q(Q,q) \;=\; \frac{2C_F}{\pi}\frac{\alpha_s(q)}{q}\left(\ln\frac{Q}{q}-\frac{3}{4}\right)\,, \tag{21}$$

$$\Gamma_g(Q,q) \;=\; \frac{2C_A}{\pi}\frac{\alpha_s(q)}{q}\left(\ln\frac{Q}{q}-\frac{11}{12}\right)\,, \tag{22}$$

$$\Gamma_f(q) \;=\; \frac{N_f}{3\pi}\frac{\alpha_s(q)}{q}\,, \tag{23}$$

respectively. $\Gamma(Q,q)$ is cut off at zero, such that $\Delta_{q,g}(Q,Q_0)$ retains its probability interpretation for having no emission resolvable at scale Q_0 during the evolution from Q to Q_0. Hence, factors of that kind are used to reweight in accordance to the appearance of external parton lines. The ratio of two Sudakov form factors $\Delta(Q,Q_0)/\Delta(q,Q_0)$ accounts for the probability of having no emission resolvable at Q_0 during the evolution from Q to q. It thus is employed for the reweighting of internal parton lines. In both cases the lower limit is taken to be $Q_0 = Q_{\rm cut}$ or $Q_0 = D\, Q_{\rm cut}$ for partons that are clustered to a beam or to another final-state parton, respectively.

5. After the reweighting of the matrix-element configuration, the parton shower is invoked starting from the pseudo history constructed before. Each parton continues its evolution at the scale where it was produced. For the virtuality-ordered parton shower of APACIC++, this scale is given by the invariant mass of the mother parton belonging to the QCD splitting, through which the considered parton has been initially formed. Recall that this QCD splitting is identified by the "cluster-backwards" procedure. In cases where the considered leg originates from the core process, the (hard) scale needs to be determined. For example, all four partons resulting from a clustering that terminated in a pure QCD $2 \to 2$ process will commence their evolution at the corresponding hard QCD scale.

6. In all circumstances parton-shower radiation is subject to the condition that no extra jet is produced. Any emission that turns out to be harder than the separation cut $Q_{\rm cut}$ is vetoed. The exception to this veto – called highest-multiplicity treatment – is for matrix-element configurations with the maximal number N of extra partons. These cases require the parton shower to cover the phase space for more jets than those produced by the matrix elements. To obtain an inclusive N-jet prediction, the veto therefore is on parton emissions at scales harder than the softest clustering scale, $Q_{\rm softest} \geq Q_{\rm cut}$. Of course, correlations including the $N+1$st jet are only approximately taken into account.

5.3 Generation of CKKW merging samples

The generation of inclusive event samples, i.e. the combination of matrix elements for different parton multiplicities with parton showers and hadronization, is completely automatized within SHERPA.[16] To obtain consistent results, certain parameters related to the matrix-element calculation and the parton showers have to be set accordingly. In the following the basic parameter settings for generating "merged" samples are summarized. Potential pitfalls are pointed out.

1. Process setup

   The starting point is the definition of a basic core (lowest-order) process with respect to which the impact of additional QCD radiation shall be studied. As an illustrative example, consider Drell–Yan lepton-pair production in pp collisions. The lowest-order process reads $pp \to l\bar{l}$, mediated through Z/γ^* exchange. Additional QCD radiation will then manifest itself through additional QCD partons in the final state, i.e. $pp \to l\bar{l}+n$ jets with $n = 1, \ldots, N$. To initialize the calculation of all the different matrix elements (for $pp \to l\bar{l} + 0, 1, \ldots, N$ QCD partons) in a single generator run, besides selecting the basic core process, the maximal number N of additional final-state QCD partons has to be specified in the `(processes)` section of the steering file. For the above example, assuming $N = 3$, this reads:

```
Process : 93 93 -> 90 90 93{3}
Order electroweak : 2
End process
```

   N is given in the curly brackets belonging to the `93`, the code for QCD partons. Note, that it is mandatory to fix the order of electroweak couplings to the corresponding order of the basic core process, here $pp \to l\bar{l}$ or `93 93 -> 90 90`, as only QCD corrections to this process can be considered and further electroweak corrections are not treated by SHERPA's CKKW implementation.

2. Setting the merging scale

[16] Note that the current implementation does not yet support the merging for processes involving supersymmetric particles.

The most important parameter to be specified when generating CKKW merged samples with SHERPA is the actual value of the jet resolution that separates the subsamples of different parton multiplicities, the merging scale. As explained in Sec. 5.2 within SHERPA jet separation is understood in terms of the k_T algorithm. Therefor in the (selector) part of the steering file the keyword JetFinder must occur. Associated with it, the resolution cut has to be given and the D parameter needs to be specified. A valid setting reads

```
JetFinder sqr(20/E_CMS) 1.
```

The first argument yields the dimensionless resolution parameter $y_{\rm cut}$, $y_{\rm cut} = Q^2_{\rm cut}/S$, here corresponding to a merging scale of $Q_{\rm cut} = 20$ GeV. Alternatively, instead of using sqr(20/E_CMS), its floating point value can be explicitly given. The second parameter corresponds to D in Eq. (18) and is ignored for leptonic initial states. In order to largely rely on matrix-element initiated jets in a subsequent external analysis, the internal D should be chosen less than or equal to the D parameter or – in case of a cone jet algorithm – the R parameter employed by the external analysis. As mentioned before, all extra QCD parton radiation is regularized by satisfying the jet-finder. However, divergences of the basic core process, such as vanishing invariant masses of lepton pairs, need to be regularized by imposing additional cuts, cf. Sec. 4.4.5.

3. Matrix-element reweighting and parton showering

   There are only two switches to enable CKKW reweighting of the matrix elements. It is mandatory to select the SCALE_SCHEME option CKKW, cf. Sec. 4.4.1, and to set SUDAKOV_WEIGHT=1.[17] Furthermore, it always should be ensured that the parton showers are switched on.

Further remarks

Although the merging of different multiplicity matrix-element samples with parton showers attached is fully automatized in SHERPA, some care has to be taken to ensure physical meaningful results. Some of the most prominent mistakes are listed here:

- When explicitly specifying the subprocesses instead of using the curly-bracket notation, it should always be ensured that the full sequence $n = 0, \ldots, N$ of processes with extra partons is listed. Otherwise there will be "holes" in phase space that are neither filled by matrix elements nor parton showering.

- The jet-finder criteria define which phase-space regions are populated by matrix elements and which ones by parton showers. In order to guarantee functioning of SHERPA's CKKW implementation, no further phase-space cuts on the additional QCD parton emissions need/should be applied while generating the samples.

- In Sec. 3 it has been pointed out that the cross-section results of the matrix-element integration can be stored and hence re-used in later runs. The same applies to the cross sections σ_k (cf. the first item in Sec. 5.2) of the various matrix-element multiplicities. For the merging algorithm to work correctly, the re-evaluation of these cross sections is however mandatory and stored integration results of former runs will not be valid whenever related parameters are changed. This includes all parameters given in the (selector) part of the steering file, such as the merging scale $Q_{\rm cut}$ or the D parameter. Furthermore, changing the number of matrix-element partons, the centre-of-mass energy, the choice of the PDF or the running of α_s requires to integrate and possibly store afresh.

- The total inclusive cross section of a SHERPA CKKW prediction does not correspond to the sum of the σ_k. Because of the matrix-element reweighting, all these fixed-multiplicity cross sections are modified (scaled down) by the CKKW procedure. The sum of the modified values determines SHERPA's total inclusive cross section, which can be obtained by following the instructions given in Sec. 3.4.

[17] Note that having enabled the Sudakov reweighting the parameters KFACTOR_SCHEME and COUPLING_SCHEME are automatically set to 1 and Running_alpha_S, respectively.

- The CKKW merging implemented in SHERPA is not yet enabled for supersymmetric processes.

Finally, a few more useful comments related to SHERPA's CKKW merging are stated below:

- Factors can be given to vary the default choices for the renormalization and factorization scales used in SHERPA simulations. The corresponding keywords are `RENORMALIZATION_SCALE_FACTOR` and `FACTORIZATION_SCALE_FACTOR`, and they are multiplied to the default μ_R^2 and μ_F^2 scales used for α_s and PDF evaluations, respectively. By varying them in a certain range around 1 (like $[1/4, 4]$), an estimate for the uncertainty of SHERPA CKKW predictions can be obtained conveniently. Note that the application of the factors requires to re-integrate the matrix elements.
- The generation of merged event samples for the production and decay of heavy particles in association with jets has a few particularities, which are best explained by means of an example. The reader is therefore referred to Sec. 7.2 showing the application to top-pair production and subsequent decays.

6 The format of SHERPA events

6.1 SHERPA's event record

SHERPA's event record is based on a list of linked `Blob`s. These reflect some very general structure that transforms particles into other particles. Consequently they have a number $n_{\rm in}$ and a number $n_{\rm out}$ of incoming and outgoing particles, respectively.[18] Some care is taken that four-momentum conservation is locally fulfilled at each of the `Blob`s. Apart from the `Particle`s attached to them, `Blob`s are located in space time, giving rise to production and decay positions of their particles. They are further specified by an identity number, a status flag, and a type – coded as an `enum` structure – such as `Signal_Process`, `ME_PS_Interface`, `IS_Shower` or `FS_Shower`.

Physical particles in SHERPA are implemented in their own class. As mentioned before, apart from those entering the event as beams and those leaving it as final-state particles, particles have a production and a decay blob, providing them with information of where they have been produced and where they decay. This structure also incorporates in a natural way mother-daughter relations (which is therefore obsolete to encode in a different way). When the parton shower is switched on, partons, i.e. quarks and gluons, also carry two colour indices in a kind of $\mathbf{3}\otimes\bar{\mathbf{3}}$ representation, but with N_c going to infinity.[19] Of course, particles carry a four momentum, given in units of GeV and an id-number. Finally, particles also have a status flag, basically `1` or `2` for particles, which are still active, or particles, which are decayed or fragmented, respectively.

To exemplify the event in more detail, consider an example `Blob` in the on-screen output of an event, which reads:

```
Blob [0]( 19, Hadron Decay      , 1 -> 3 @ (0.1196,-0.00101,0.0001581,-0.0001778)
Incoming particles :
[D] 2 B+                    26 (  16 ->   19) [( 3.9963e+01, 1.3754e+01, 3.7148e+01,-1.9295e-01), p^2= 2.7867e+01, m= 5.2789e+00] (  0,  0)
Outgoing particles :
[D] 2 D*(2007)              28 (  19 ->   21) [( 2.3236e+01, 7.1989e+00, 2.1991e+01,-6.8277e-01), p^2= 4.0271e+00, m= 2.0068e+00] (  0,  0)
[D] 2 Db                    29 (  19 ->   20) [( 1.4811e+01, 5.6595e+00, 1.3557e+01, 2.3178e-01), p^2= 3.4767e+00, m= 1.8646e+00] (  0,  0)
[D] 1 K+                    30 (  19 ->     ) [( 1.9161e+00, 8.9547e-01, 1.5998e+00, 2.5804e-01), p^2= 2.4372e-01, m= 4.9368e-01] (  0,  0)
```

The first line is for the blob, its status ([0]), its number (19) and type (`Hadron Decay`), the number of incoming and outgoing particles ($1 \to 3$), and its four position in the notation $(t, \vec{x})$. Then the particles are listed with its particle info code, status, type,[20] its number, information about the production and decay blob,[21] its four momentum in the notation $(E, \vec{p})$, the momentum-square and mass, and finally its colour charges.[22] The colour flow is represented in the $N_c \to \infty$ limit by two numbers standing for the $\mathbf{3}\otimes\bar{\mathbf{3}}$ components.

[18] In terms of programming, the `Blob`s own pointers to the respective `Particle`s and these `Blob`s are responsible of erasing them after each event.

[19] By the way, this is one of the assumptions implicit in one way or the other in all event generators.

[20] In most cases, when the particle is electrically neutral, the charge subscript is missing.

[21] For instance, the incoming B^+ is produced in `Blob 16` and decays in `Blob 19`.

[22] Obviously, hadrons, photons, etc. have no colour charges.

6.2 Event output formats

SHERPA provides the possibility to output events – either to file or to screen – in its native and two other output formats: The `HepEVT` common block structure or the `HepMC` format [86]. The authors of SHERPA assume that the user is sufficiently acquainted with these formats when selecting them.
There are two ways to specify the event record. First of all, the switch `EVENT_MODE` can be set to either `Sherpa` (default), `HepMC`, or `HepEvt`, resulting in the corresponding structure to be filled internally (and printed to screen, if the output-level is set accordingly).
Second, if the events are to be written to file, the keywords `SHERPA_OUTPUT`, `HEPMC2_*_OUTPUT` and `HEPEVT_OUTPUT` are available, see below. With these keywords the filename's root can be specified, i.e. `HEPEVT_OUTPUT=filename` will create files named `filename.#.hepevt`, where the hash mark stands for an increasing number.
By default, each file contains 1000 events, but the number can be set by specifying the keyword `FILE_SIZE`. Use `EVT_FILE_PATH` to change the directory, where the files will be stored. To steer the precision of all `double` numbers written to file, the keyword `OUTPUT_PRECISION` is available, with a default value of 12.
To write events directly to gzipped files instead of plain text, the option `--enable-gzip` has to be specified during configure (cf. Sec. 2).
Note that multiple output formats can be used in the same run:

- `HEPEVT_OUTPUT`: Full events in the HepEvt format.
- `HEPMC2_GENEVENT_OUTPUT`: HepMC output using the `HepMC::IO_GenEvent` class. This assumes that the user enabled the link to the `HepMC` package version 2.x. This is possible through a corresponding `configure` flag, see Sec. 2.
- `HEPMC2_ASCII_OUTPUT`: HepMC output using the `HepMC::IO_Ascii` class (deprecated). This assumes that the user enabled the link to the `HepMC` package version 2.x. This is possible through a corresponding `configure` flag, see Sec. 2.
- `HEPMC2_OUTPUT`: HepMC output using the `HepMC::GenEvent::print` method (deprecated). This assumes that the user enabled the link to the `HepMC` package version 2.x. This is possible through a corresponding `configure` flag, see Sec. 2.
- `SHERPA_OUTPUT`: Full events in the Sherpa format.
- `D0_HEPEVT_OUTPUT`: Specifically for `D0` analysis needs, there is this modified HepEvt output format available. It will output events in the HepEvt format, with one extra line added for each event, containing information about the signal process:

  `enhance-factor` Q^2 x_1 x_2 `parton1 parton2`

It is of course also possible to link a user program to SHERPA, so that the events can be read and/or analyzed "on the flight". Two configurations are supported. Users can of course access SHERPA's internal `Blob_List`, which contains all necessary information about an event. Or, they can access the `HepMC::GenEvent`, if `EVENT_MODE` is set to `HepMC`.

7 Application examples

Before running SHERPA maybe for a different collider with a different process in a different physics model, the revision of SHERPA's parameter file `Run.dat` and various modifications to it are absolutely necessary. Information about available options and detailed instructions are provided in Sec. 4. However, in this section, examples for Monte Carlo simulations will be collected that should help the users get familiar with setting up their own parameter files for running SHERPA for various tasks.

7.1 Default setup: hadron production in $e\gamma$ collisions

The distributed version of SHERPA comes with a default setup for the simulation of an $e\gamma$ collision. The corresponding data card, `Run.dat`, is located in the `<sherpa_path>/SHERPA/Run/EGamma/` directory. The following type of process is considered within the Standard Model (`MODEL=SM`) in this example:

$$e^-e^- \to e^-\gamma \quad \to \quad e^-\gamma \to e^- d\bar{d} \quad \to \quad e^- + \text{hadrons}\,.$$

In the first step the photon is produced through laser backscattering off one of the incoming electrons. Accordingly, in the `(beam)` section of the data card, `BEAM_1,2=11`, `BEAM_SPECTRUM_1=Laser_Backscattering` and `BEAM_SPECTRUM_2=Monochromatic` have been specified. In the second step the other electron experiences some initial-state radiation according to a structure function, before the electron and photon produce a $d\bar{d}$ pair through some kind of peripheral scattering. In the data card's `(isr)` section one therefore finds

```
BUNCH_1 = 22
ISR_1   = Off
BUNCH_2 = 11
ISR_2   = On
```

and the hard process in terms of the `(processes)` syntax reads

```
Process : 22 11 -> 11 1 -1
End process
```

and, of course, there is no merging of matrix elements of various multiplicities here, hence, `SUDAKOV_WEIGHT=0` is set in `(me)` of `Run.dat`. As described in Sec. 3, by performing `./makelibs` in the `EGamma` setup directory the generated process libraries of the first run will be compiled and stored as shared libraries. Restarting the program gets the user to the process integration. In calculating the cross section for this interaction and for the generation of fully hadronized events, some cuts are imposed on the initial and final state, namely

$$\begin{array}{lclclclclcl} E_\perp(e^-) & > & 1\text{ GeV}, & & E_\perp(d) & > & 1\text{ GeV}, & & E_\perp(\bar{d}) & > & 1\text{ GeV}, \\ \cos\theta_{e^-} & \in & [-0.9,\,0.9], & & \cos\theta_d & \in & [-0.9,\,0.9], & & \cos\theta_{\bar{d}} & \in & [-0.9,\,0.9]\,. \end{array}$$

The angles are defined w.r.t. the positive beam axis. Supplementary the Durham jet-finding algorithm is applied employing the resolution cut

$$y_{\text{cut}} \;=\; 0.004\,.$$

These requirements are all implemented in the `(selector)` part of `Run.dat` and read:

```
ET         11       1.   10000.
ET          1       1.   10000.
ET         -1       1.   10000.
BeamAngle  11  0   -0.9       0.9
BeamAngle   1  0   -0.9       0.9
BeamAngle  -1  0   -0.9       0.9
JetFinder   4.e-3  1.
```

where the upper bound for the `ET` entries is just a code requisite. Apart from cuts, the treatment of couplings and scales is important for the determination of the cross section. The `(me)` section (cf. Sec. 4.4.1) specifies that fixed couplings (α and α_s) are considered, i.e. `COUPLING_SCHEME=Fixed`, and the hard scale is set by the partonic centre-of-mass energy, i.e. `SCALE_SCHEME=S_HAT`. Furthermore there is no need to switch on the `KFACTOR_SCHEME`. QCD parton showering can be added only in the final state, so, in the `(shower)` section `ISR_SHOWER=0`. For a full event handling, the resulting final-state partons after the parton shower are converted into hadrons, which are further decayed into stable hadrons under inclusion of photon-emission effects. Thus, the `(fragmentation)` part of `EGamma`'s steering file is the same as that one discussed in Sec. 3.2.

7.2 Top-pair production in association with jets

The semi-leptonic channel that is considered here serves as an example for the more general task of simulating inclusive $t\bar{t}$ production plus their decays. It reads

$$p\,p(\bar{p}) \;\to\; t\,\bar{t} + X \;\to\; b\,e^{+}\nu_e\;\;\bar{b}jj + X\,,$$

and is particularly chosen to discuss additional parameters that are relevant for handling in a simulation the production and decay of heavy objects together with QCD bremsstrahlung. Effects owing to the likely appearance of an extra hard jet shall be included by using SHERPA's capability to merge matrix elements of different order in QCD consistently with the parton showers. [23] Hence,

$$\begin{aligned}&\text{parton parton} \;\to\; t\,\bar{t} \;\to\; b\,e^{+}\nu_e\;\;\bar{b}\,\text{parton parton}\,,\\ &\text{parton parton} \;\to\; t\,\bar{t} \;+\; \text{parton} \;\to\; b\,e^{+}\nu_e\;\;\bar{b}\,\text{parton parton} \;+\; \text{parton}\end{aligned}$$

are the matrix elements taken into account. The `(processes)` part of the parameter file eventually has the following structure:

```
Process : 93 93 -> 6[a] -6[b] 93{1}
Decay :  6[a] ->  5 -11 12
Decay : -6[b] -> 93  93 -5
Order electroweak : 0
Max_Epsilon : 1e-4
End process
```

whereas only those diagrams are kept by AMEGIC++ that in fact feature the top propagators. The treatment therefore neglects certain (interference) contributions, however, preserves the full spin correlations. The statement of having no electroweak couplings merely applies to the $t\bar{t}$ production, i.e. only the strong-interaction processes contribute. Each b quark is treated as a massive particle in the matrix elements, if one adds the assignment `MASSIVE[5]=1` to the `(model)` part. This moreover defines "parton" to be understood as light (massless) quark or gluon. Concerning the cross-section determination and the merging algorithm, jet finders can now be applied with different $y_{\rm cut}$ scales in both production and decays. The idea is that production and decays factorize in the narrow-width approximation. In such cases the merging procedure can be applied separately and independently for the production and the decay processes. The jet-finder syntax therefore has been extended to accept terms `[y_cut(i)]` per decay. [24] A valid `(selector)` paragraph for `Run.dat` thus reads:

```
JetFinder   sqr(40/E_CMS)[sqr(20/E_CMS)][sqr(20/E_CMS)]  1.
"m" -11,12,5   150.0,200.0
"m"  93,93,-5  150.0,200.0
```

where all real numbers are understood in GeV. There are two more comments. The first is about increasing the integration performance: the "`Max_Epsilon : 1e-4`" line and the last two lines of the `(selector)` paragraph are respectively given to allow 0.01% of the weights to be larger than the maximum weight and neglect events where the tops are rather offshell. The second comment concerns parton showering: before the tops decay weakly, soft and collinear gluons may arise from the produced tops as well as when they enter the decays; so, to account for these effects (in addition to the radiating b quarks of the decays) the parton shower has to be run with `SHOWER_MODE=3` being enabled in the `(shower)` part of the data card.

[23] Please cf. Sec. 5 for more details.

[24] If the `[..]` terms are not specified then all $y_{\rm cut}^{(i)}$ are equal to the $y_{\rm cut}$ of the production, i.e. the simple jet-finder syntax works as well.

Acknowledgments

We, the authors of Sherpa would like to thank our users, who helped us improve this package. We are very grateful that many people encouraged us to continue with this project. Special thanks go to all users, who actively supported the development by giving detailed comments and suggestions, and by providing bug reports and fixes to this version and to all previous releases. We are indebted to all supporters, who promoted the usage of Sherpa for carrying out various physics analyses.

References

[1] T. Gleisberg et al., (2008), 0811.4622.

[2] T. Sjöstrand, Comput. Phys. Commun. 82 (1994) 74.

[3] T. Sjöstrand, S. Mrenna and P. Skands, JHEP 05 (2006) 026, hep-ph/0603175.

[4] M. Bahr et al., (2008), 0803.0883.

[5] T. Sjöstrand, S. Mrenna and P. Skands, Comput. Phys. Commun. 178 (2008) 852, 0710.3820.

[6] J. Rosiek, Phys. Rev. D41 (1990) 3464.

[7] J. Rosiek, (1995), hep-ph/9511250.

[8] T. Gleisberg et al., JHEP 0309 (2003) 001, arXiv:hep-ph/0306182.

[9] K. Hagiwara et al., Nucl. Phys. B282 (1987) 253.

[10] A. Dedes et al.

[11] F. Krauss, R. Kuhn and G. Soff, JHEP 0202 (2002) 044, arXiv:hep-ph/0109036.

[12] T. Gleisberg et al., Eur. Phys. J. C 34 (2004) 173, arXiv:hep-ph/0311273.

[13] K. Hagiwara et al., arXiv:hep-ph/0512260.

[14] S. Catani et al., JHEP 0111 (2001) 063, arXiv:hep-ph/0109231.

[15] F. Krauss, JHEP 0208 (2002) 015, arXiv:hep-ph/0205283.

[16] A. Schälicke and F. Krauss, JHEP 0507 (2005) 018, arXiv:hep-ph/0503281.

[17] F. Krauss et al., Phys. Rev. D 70 (2004) 114009, arXiv:hep-ph/0409106.

[18] F. Krauss et al., Phys. Rev. D 72 (2005) 054017, arXiv:hep-ph/0503280.

[19] T. Gleisberg et al., Phys. Rev. D 72 (2005) 034028, arXiv:hep-ph/0504032.

[20] J. Alwall et al., Eur. Phys. J. C53 (2008) 473, 0706.2569.

[21] M.L. Mangano, M. Moretti and R. Pittau, Nucl. Phys. B632 (2002) 343, hep-ph/0108069.

[22] M.L. Mangano et al., JHEP 07 (2003) 001, hep-ph/0206293.

[23] T. Stelzer and W.F. Long, Comput. Phys. Commun. 81 (1994) 357, hep-ph/9401258.

[24] F. Maltoni and T. Stelzer, JHEP 02 (2003) 027, hep-ph/0208156.

[25] A. Kanaki and C.G. Papadopoulos, Comput. Phys. Commun. 132 (2000) 306, hep-ph/0002082.

[26] C.G. Papadopoulos and M. Worek, Eur. Phys. J. C50 (2007) 843, hep-ph/0512150.

[27] L. Lönnblad, JHEP 05 (2002) 046, hep-ph/0112284.

[28] N. Lavesson and L. Lönnblad, JHEP 07 (2005) 054, hep-ph/0503293.

[29] L. Lönnblad, Comput. Phys. Commun. 71 (1992) 15.

[30] F. Krauss, A. Schälicke and G. Soff, (2005), arXiv:hep-ph/0503087.

[31] R. Kuhn et al., Comput. Phys. Commun. 134 (2001) 223, arXiv:hep-ph/0004270.

[32] M. Bengtsson and T. Sjöstrand, Nucl. Phys. B289 (1987) 810.

[33] M. Bengtsson, T. Sjöstrand and M. van Zijl, Z. Phys. C32 (1986) 67.

[34] S. Schumann and F. Krauss, JHEP 03 (2008) 038, 0709.1027.

[35] J. Winter and F. Krauss, JHEP 07 (2008) 040, 0712.3913.

[36] A.F. Zarnecki, Acta Phys. Polon. B 34 (2003) 2741, arXiv:hep-ex/0207021.

[37] J. Pumplin et al., JHEP 0207 (2002) 012, arXiv:hep-ph/0201195.

[38] A.D. Martin et al., Eur. Phys. J. C 14 (2000) 133, arXiv:hep-ph/9907231.

[39] A.D. Martin et al., Eur. Phys. J. C 23 (2002) 73, arXiv:hep-ph/0110215.

[40] M. Gluck, E. Reya and A. Vogt, Phys. Rev. D 46 (1992) 1973.

[41] M.R. Whalley, D. Bourilkov and R.C. Group, (2005), arXiv:hep-ph/0508110, for download consult `http://hepforge.cedar.ac.uk/lhapdf/`.

[42] P. Skands et al., JHEP 0407 (2004) 036, arXiv:hep-ph/0311123.

[43] R. Kleiss and W.J. Stirling, Nucl. Phys. B 262 (1985) 235.

[44] A. Ballestrero, E. Maina and S. Moretti, Nucl. Phys. B 415 (1994) 265, arXiv:hep-ph/9212246.

[45] R. Kleiss and R. Pittau, Comput. Phys. Commun. 83 (1994) 141, arXiv:hep-ph/9405257.

[46] F.A. Berends, R. Pittau and R. Kleiss, Nucl. Phys. B 424 (1994) 308, arXiv:hep-ph/9404313.

[47] G.P. Lepage, (1980), CLNS-80/447.

[48] R. Kleiss, W.J. Stirling and S.D. Ellis, Comput. Phys. Commun. 40 (1986) 359.

[49] P.D. Draggiotis, A. van Hameren and R. Kleiss, Phys. Lett. B 483 (2000) 124, arXiv:hep-ph/0004047.

[50] T. Sjöstrand et al., (2003), arXiv:hep-ph/0308153.

[51] G. Corcella et al., JHEP 0101 (2001) 010, arXiv:hep-ph/0011363.

[52] G. Corcella et al., (2002), arXiv:hep-ph/0210213.

[53] G. Marchesini and B.R. Webber, Nucl. Phys. B238 (1984) 1.

[54] B.R. Webber, Nucl. Phys. B238 (1984) 492.

[55] G. Marchesini and B.R. Webber, Nucl. Phys. B310 (1988) 461.

[56] S. Gieseke, P. Stephens and B. Webber, JHEP 12 (2003) 045, hep-ph/0310083.

[57] M. Bahr et al., (2008), 0804.3053.

[58] T. Sjöstrand and M. van Zijl, Phys. Rev. D 36 (1987) 2019.

[59] T.D. Gottschalk, Nucl. Phys. B214 (1983) 201.

[60] T.D. Gottschalk, Nucl. Phys. B239 (1984) 349.

[61] T.D. Gottschalk and D.A. Morris, Nucl. Phys. B288 (1987) 729.

[62] J. Winter, F. Krauss and G. Soff, Eur. Phys. J. C 36 (2004) 381, arXiv:hep-ph/0311085.

[63] D.R. Yennie, S.C. Frautschi and H. Suura, Ann. Phys. 13 (1961) 379.

[64] K. Hamilton and P. Richardson, JHEP 07 (2006) 010, hep-ph/0603034.

[65] M. Schönherr and F. Krauss, JHEP 12 (2008) 018, 0810.5071.

[66] E. Barberio and Z. Was, Comput. Phys. Commun. 79 (1994) 291.

[67] G. Marsaglia, A. Zaman and W. Zang, (1987), Florida State University Report FSU-SCRI-87-50.

[68] V.M. Budnev et al., Phys. Rept. 15 (1974) 181.

[69] Particle Data Group, W.M. Yao et al., J. Phys. G 33 (2006) 1.

[70] L. Wolfenstein, Phys. Rev. Lett. 51 (1983) 1945.

[71] F. Gangemi et al., (1999), hep-ph/0001065.

[72] G.J. Gounaris, J. Layssac and F.M. Renard, Phys. Rev. D62 (2000) 073012, hep-ph/0005269.

[73] U. Baur and D. Zeppenfeld, Nucl. Phys. B308 (1988) 127.

[74] W.S. Hou, A. Soni and H. Steger, Phys. Lett. B192 (1987) 441.

[75] A. van Hameren and C.G. Papadopoulos, Eur. Phys. J. C25 (2002) 563, hep-ph/0204055.

[76] J. Winter and F. Krauss, Publication in preparation (2008).

[77] A. Deur et al., (2008), 0803.4119.

[78] S. Jadach et al., Comput. Phys. Commun. 76 (1993) 361.

[79] D.J. Lange, Nucl. Instrum. Meth. A462 (2001) 152.

[80] F. Krauss, T. Laubrich and F. Siegert, Publication in preparation (2008).

[81] M.L. Mangano et al., JHEP 01 (2007) 013, hep-ph/0611129.

[82] S. Catani et al., Phys. Lett. B 269 (1991) 432.

[83] S. Catani, Y.L. Dokshitser and B.R. Webber, Nucl. Phys. B 406 (1993) 187.

[84] S. Catani, Y.L. Dokshitser and B.R. Webber, Phys. Lett. B 285 (1992) 291.

[85] G.C. Blazey et al., (2005), arXiv:hep-ex/0005012.

[86] M. Dobbs and J.B. Hansen, Comput. Phys. Commun. 134 (2001) 41.