Eur. Phys. J. C (2008) 58: 639–707
DOI 10.1140/epjc/s10052-008-0798-9

THE EUROPEAN PHYSICAL JOURNAL C

Special Article - Tools for Experiment and Theory

Herwig++ physics and manual

Manuel Bähr[1], Stefan Gieseke[1], Martyn A. Gigg[2], David Grellscheid[2], Keith Hamilton[3], Oluseyi Latunde-Dada[4], Simon Plätzer[1], Peter Richardson[2,5,a], Michael H. Seymour[5,6], Alexander Sherstnev[4], Bryan R. Webber[4]

[1]Institut für Theoretische Physik, Universität Karlsruhe, Karlsruhe, Germany
[2]IPPP, Department of Physics, Durham University, Durham, UK
[3]Centre for Particle Physics and Phenomenology, Université Catholique de Louvain, Louvain-la-Neuve, Belgium
[4]Cavendish Laboratory, University of Cambridge, Cambridge, UK
[5]Physics Department, CERN, Geneva, Switzerland
[6]School of Physics and Astronomy, University of Manchester, Manchester, UK

Received: 1 September 2008 / Published online: 20 November 2008

Abstract In this paper we describe Herwig++ version 2.2, a general-purpose Monte Carlo event generator for the simulation of hard lepton-lepton and hadron-hadron collisions. A number of important hard scattering processes are available, together with an interface via the Les Houches Accord to specialized matrix element generators for additional processes. The simulation of Beyond the Standard Model (BSM) physics includes a range of models and allows new models to be added by encoding the Feynman rules of the model. The parton-shower approach is used to simulate initial- and final-state QCD radiation, including colour coherence effects, with special emphasis on the correct description of radiation from heavy particles. The underlying event is simulated using an eikonal multiple parton-parton scattering model. The formation of hadrons from the quarks and gluons produced in the parton shower is described using the cluster hadronization model. Hadron decays are simulated using matrix elements, where possible including spin correlations and off-shell effects.

PACS 12.38.Cy · 13.87.Ce · 13.87.fh

Contents

[a] e-mail: herwig@projects.hepforge.org

1 Introduction

Herwig++ is a general-purpose event generator for the simulation of high-energy lepton-lepton and hadron-hadron collisions with special emphasis on the accurate simulation of QCD radiation. It builds upon the heritage of the HERWIG program [1–6], while providing a much more flexible structure for further development. It already includes several features more advanced than the last FORTRAN version. Herwig++ provides a full simulation of high energy collisions with the following special features:

- Initial- and final-state QCD jet evolution taking account of soft gluon interference via angular ordering;
- A detailed treatment of the suppression of QCD radiation from massive particles, the *dead-cone* effect [7];
- The simulation of BSM physics including correlations between the production and decay of the BSM particles together with the ability to add new models by simply encoding the Feynman rules;
- An eikonal model for multiple partonic scatterings to describe the underlying event [8];
- A cluster model of the hadronization of jets based on non-perturbative gluon splitting;
- A sophisticated model of hadron and tau decays using matrix elements to give the momenta of the decay products for many modes and including a detailed treatment of off-shell effects and spin correlations.

Some of these features were already present in the first version of Herwig++ [9]. However, there have been many improvements to both the physics and structure of the simulation following this first release, most notably the extension to hadron-hadron collisions. Given the significant differences between the current version of the program, 2.2, and that described in [9] we will describe all of the features of the program in this paper.

A number of other generators are also being (re-)written for the LHC era. The PYTHIA event generator is currently being rewritten as PYTHIA8 [10]. The rewrite of ARIADNE [11] is in progress as well. Like Herwig++, this is built on the platform of ThePEG [12], which we describe below. SHERPA [13] is a completely new event generator project.

It is useful to start by recalling the main features of a generic hard, high-momentum transfer, process in the way it is simulated by Herwig++. The processes involved can be divided into a number of stages corresponding to increasing time and distance scales:

1. *Elementary hard subprocess.* In the hard process the incoming particles interact to produce the primary outgoing fundamental particles. This interaction can involve either the incoming fundamental particles in lepton collisions or partons extracted from a hadron in hadron-initiated processes. In general this is computed at leading order in perturbation theory, although work is ongoing to include higher-order corrections [14, 15]. The energy scale of the hard process, together with the colour flow between the particles, sets the initial conditions for the production of QCD radiation in the initial- and final-state parton showers.
2. *Initial- and final-state parton showers.* The coloured particles in the event are perturbatively evolved from the hard scale of the collision to the infrared cutoff. This occurs for both the particles produced in the collision, the *final-state shower*, and the initial partons involved in the collision for processes with incoming hadrons, the *initial-state shower*. The coherence of the emission of soft gluons in the parton showers from the particles in the hard collision is controlled by the colour flow of the hard collision. Inside the parton shower, it is simulated by the angular ordering of successive emissions. The choice of evolution variable together with the use of quasi-collinear splitting functions allows us to evolve down to zero transverse momentum for the emission, giving an improved simulation of the dead-cone effect for radiation from massive particles [7].
3. *Decay of heavy objects.* Massive fundamental particles such as the top quark, electroweak gauge bosons, Higgs bosons, and particles in many models of physics beyond the Standard Model, decay on time-scales that are either shorter than, or comparable to that of the QCD parton shower. Depending on the nature of the particles and whether or not strongly interacting particles are produced in the decay, these particles may also initiate parton showers both before and after their decay. One of the

major features of the Herwig++ shower algorithm is the treatment of radiation from such heavy objects in both their production and decay. Spin correlations between the production and decay of such particles are also correctly treated.

4. *Multiple scattering.* For large centre-of-mass energies the parton densities are probed in a kinematic regime where the probability of having multiple partonic scatterings in the same hadronic collision becomes significant. For these energies, multiple scattering is the dominant component of the underlying event that accompanies the main hard scattering. These additional scatterings take place in the perturbative regime, above the infrared cutoff, and therefore give rise to additional parton showers. We use an eikonal multiple scattering model [8], which is based on the same physics as the FORTRAN JIMMY package [16], together with some minor improvements.
5. *Hadronization.* After the parton showers have evolved all partons involved in hard scatterings, additional scatters and partonic decays down to low scales, the final state typically consists of coloured partons that are close in momentum space to partons with which they share a colour index, their colour 'partner' (in the large N_c limit this assignment is unique). Herwig++ uses the cluster hadronization model [2] to project these colour–anticolour pairs onto singlet states called clusters, which decay to hadrons and hadron resonances. The original model of Ref. [2], which described this decay as pure phase space has been progressively refined as described in Sect. 7. Clusters that are too massive or too light for decay directly to hadrons to provide a good description are treated differently, again described in Sect. 7.
6. *Hadron decays.* The hadron decays in Herwig++ are simulated using a matrix element description of the distributions of the decay products, together with spin correlations between the different decays, wherever possible. The treatment of spin correlations is fully integrated with that used in perturbative production and decay processes so that correlations between the production and decay of particles like the tau lepton, which can be produced perturbatively but decays hadronically, can be treated consistently.

The program and additional documentation are available from http://projects.hepforge.org/herwig. This manual concentrates on the physics included in the Herwig++ simulation, which has been the subject of a number of publications [9, 14, 15, 17–23]. Additional documentation of the code, together with examples of how to use the program and further information is available from our website and wiki. We provide a bug-tracker, which should be used to report any problems with the program or to request user support.

Herwig++ is distributed under the GNU General Public License (GPL) version 2. This ensures that the source code will be available to users, grants them the freedom to use and modify the program and sets out the conditions under which it can be redistributed. However, it was developed as part of an academic research project and is the result of many years of work by the authors, which raises various issues that are not covered by the legal framework of the GPL. It is therefore distributed together with a set of guidelines,[1] agreed by the MCnet collaboration, which set out various expectations that we have of responsible users. In particular, concerning citation of relevant physics publications, they state that the main software reference as designated by the program authors (*i.e.* this manual for Herwig++ versions 2.1 onwards) should always be cited, as well as the original literature on which the program is based to the extent that it is of relevance for a study, applying the same threshold criteria as for other literature. To help users in this, Herwig++ produces a LaTeX file that lists the primary physics citation(s) for each module that has been active during a given run. The authors are always happy to help users determine which citations are relevant to a particular study.

The remainder of this manual is set out as follows. The next section contains a brief technical description which should be sufficient to understand the details of the program included in the discussion of the physics simulation. More detailed technical documentation can be obtained from the website above, including Doxygen descriptions of all classes.

The rest of the manual then discusses the physics of each stage of the simulation process in detail, describing the physics models used in the simulation, together with the main parameters of the models and the structure of the code. Finally, we give a summary and our plans for future improvements. Appendices give some more technical information, a series of examples of the program in use, and a brief description of the process by which the default parameters were tuned to data.

2 Technical details

While this manual is primarily a description of the physics models used in Herwig++, by its nature we cannot wholly avoid discussing the technical details of the program. We need to discuss some aspects of the program's structure and the mechanism for changing physics model parameters, so that users can adjust parameters, change the hard process they are simulating, or make any of the other modifications that are necessary to make the program useful to an individual user. In this section we will give a basic overview of the

[1] These guidelines are contained in the `GUIDELINES` file distributed with the release and are also available from http://www.montecarlonet.org/index.php?p=Publications/Guidelines.

structure of the program, which is designed to supplement the Doxygen documentation of the source code available at http://projects.hepforge.org/herwig/doxygen. Herwig++ is based on ThePEG [12]—the Toolkit for High Energy Physics Event Generation, a framework for implementing Monte Carlo event generators. ThePEG provides all parts of the event generator infrastructure that do not depend on the physics models used as a collection of modular building blocks. The specific physics models of Herwig++ are implemented on top of these.

Each part of Herwig++ is implemented as a C++ class that contains the implementation of the Herwig++ physics models, inheriting from an abstract base class in ThePEG. This allows the implementations of different physics models to live side-by-side and be easily exchanged.

The central concept in ThePEG is the Repository, which holds building blocks in the form of C++ objects that can be combined to construct an EventGenerator object, which in turn will be responsible for all steps of event generation. Within the Repository, one can create objects, set up references between them, and change all parameter values. The Repository object needs to be populated with references to all required objects for the physics models used at run time. The objects can then be persistently stored, or combined to produce an EventGenerator. The default Repository layout for Herwig++ is shown in Table 1. The composition of the Repository is controlled through a simple configuration language, described in Appendix A. This set of commands allows the user to configure the generator at run time. Through this mechanism, selection of different physics models or different model parameters is possible without recompilation of the code.

The EventGenerator object is responsible for the run[2] as a whole. It holds the infrastructure objects that are needed for the run, like the generation of random numbers, the particle properties stored as ParticleData objects, and handles any exceptions.

The actual generation of each event is the responsibility of the EventHandler. It manages the generation of the hard scattering process[3] and the subsequent evolution of the event through five StepHandler objects, each of which is responsible for generating one main part of the event:

1. The SubProcessHandler is responsible for generating the hard sub-process as described in Sect. 3. This handler is skipped if the hard process is read in from a Les Houches Accord event file.
2. The CascadeHandler generates the parton shower from the hard process.
3. The MultipleInteractionHandler produces additional hard scatters when using a multiple parton-parton scattering model to simulate the underlying event in hadron-hadron collisions. In practice, given the close relationship between the parton shower and the additional hard scatters in Herwig++, the multiple scattering model is implemented as part of the Herwig++ implementation of the CascadeHandler, the ShowerHandler.
4. The HadronizationHandler is responsible for the formation of hadrons out of the quarks and gluons left after the parton shower.
5. The DecayHandler is responsible for decaying both the unstable hadrons produced by the HadronizationHandler, and any unstable fundamental particles that may have been produced in either the hard process or parton shower.

The StepHandler base classes in ThePEG do not implement any physics models themselves. This must be done by inheriting classes, which provide an implementation of a specific model. The Herwig++ ShowerHandler for example, inherits from CascadeHandler and implements the Herwig++ parton shower model by overriding the virtual cascade() member function.

In addition to the five main handlers, the EventHandler allows for pre- and post-handlers to be called before and after each step. This allows for additional processing of the event where required: in Herwig++ BSM physics or top quark production, the HwDecayHandler is used as a pre-handler for the ShowerHandler to ensure that all the unstable fundamental particles have decayed before the parton shower occurs.

The implementation of a physics model as a StepHandler generally does not put all the code needed for the simulation in one class, but makes use of an, often large, number of helper classes.

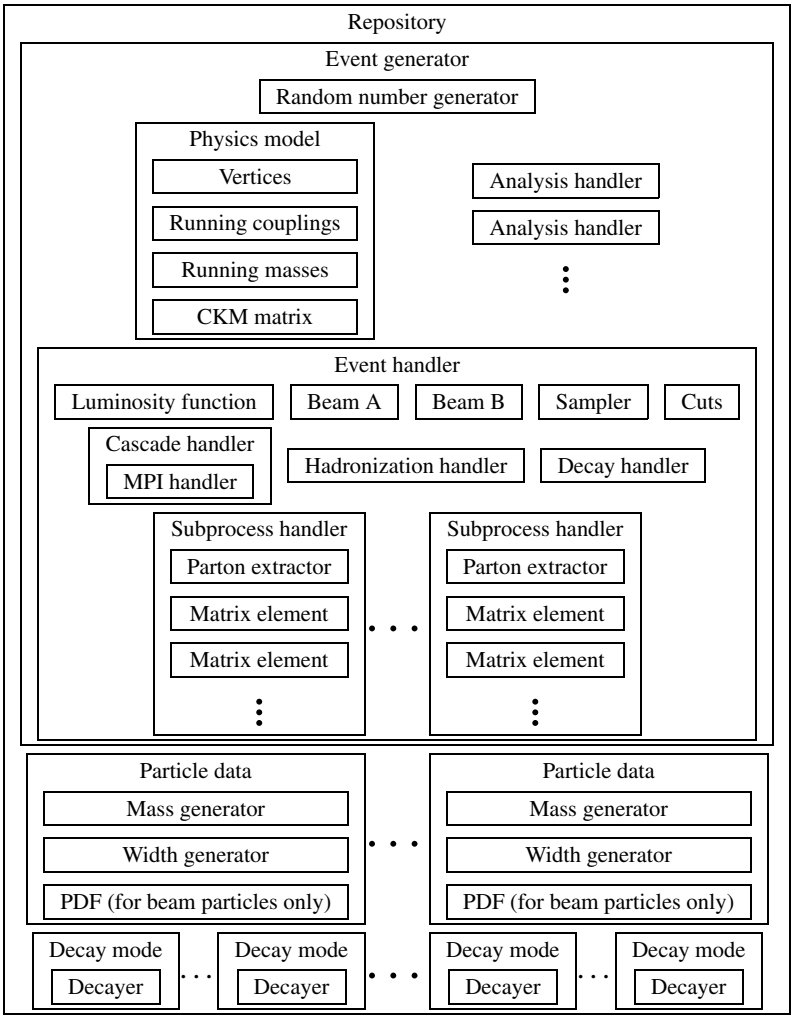
This brief description only discusses the classes responsible for generating the core parts of the event. Other classes and concepts are discussed in more detail in the Doxygen documentation.

The mechanisms for exploring and changing the values of switches and parameters are also described in Appendix A. It is worth mentioning that 'default' values of switches and parameters can appear in one of two places: the repository entries in the default `.in` files; or the class constructors and at present there is no built-in mechanism to ensure that they are consistent. When both exist, the former takes precedence. The values described as 'default' in this manual are those that appear in the default `.in` files. A further confusion appears, because the value described as default in the Doxygen documentation is not guaranteed to be the same as either of the others. A mechanism to ensure that all three default values are the same will be introduced in a future version, but until then, users are reminded that the default `.in` files remain the primary source of parameter values.

[2]The generation of a series of events.

[3]The generation of the hard process by the EventHandler and its inheriting classes is discussed in more detail in Sect. 3.3.

Table 1 Overview of the default Repository layout for Herwig++. Each box represents a reference to an independent C++ object held in the repository, which can be swapped out for a different implementation



3 Matrix elements

In Herwig++ the library of matrix elements for QCD and electroweak processes is relatively small, certainly with respect to the large range of processes available in its FORTRAN predecessor [5, 6]. Indeed, the library of Standard Model processes is largely intended to provide a core of important processes with which to test the program. Whereas, at the time of the development of the original FORTRAN program, matrix elements needed to be calculated and implemented by hand, nowadays there are a number of programs that automate these calculations, for a wide range of processes with high multiplicity final states. It has therefore been our intention that, in general, users should study most processes of interest via our interface to these programs.

Nevertheless, there are still some cases for which it is useful to have Herwig++ handle all stages of the event generation process. This is particularly true for processes in which spin correlations between the production and decay stages are significant *e.g.* those involving top quarks or tau leptons. Such correlation effects are hard to treat correctly if different programs handle different steps of the simulation process.

In order to facilitate the process of adding new matrix elements, where needed, and to enable us to generate the spin correlation effects [24–27], we have based all matrix element calculations on the helicity libraries of ThePEG. As well as providing a native library of Standard Model processes and an interface to parton-level generators, Herwig++ also includes matrix elements for hard $2 \to 2$ collisions and $1 \to 2$ decays, arising in various models of new physics (see Sect. 5).

3.1 Matrix elements for specific processes

For e^+e^- colliders only three hard processes are included:

– Quark-antiquark and dilepton pair production, via interfering photon and Z^0 bosons. The associated matrix elements are implemented in the MEee2gZ2qq and MEee2gZ2ll classes respectively. No approximation is

made regarding the masses of the particles.[4] These processes are essential for us to validate the program using QCD analyses of LEP data and, similarly, to check the implementation of spin correlations in τ decays.
- The Bjorken process, Z^0h^0 production, which is implemented in the MEee2ZH class. This process is included as it is very similar to the production of Z^0h^0 and $W^{\pm}h^0$ in hadron-hadron collisions and uses the same base class for most of the calculation.

A much wider range of matrix elements is included in the standalone code for the simulation of events in hadron colliders:

- Difermion production via s-channel electroweak gauge bosons. The matrix elements for the production of fermion-antifermion pairs through $W^{\pm}$ bosons, or interfering photons and Z^0 bosons, are implemented in the MEqq2W2ff and MEqq2gZ2ff classes respectively. Only s-channel electroweak gauge boson diagrams are included for the hadronic modes.
- The production of a Z^0 or $W^{\pm}$ boson in association with a hard jet is simulated using the MEPP2ZJet or MEPP2WJet class respectively. The decay products of the bosons are included in the $2 \to 3$ matrix element and the option of including the photon for Z^0 production is supported.
- The $2 \to 2$ QCD scattering processes are implemented in the MEQCD2to2 class. Currently all the particles are treated as massless in these processes.
- The matrix element for the production of a heavy quark-antiquark pair (top or bottom quark pairs), is coded in the MEPP2QQ class. No approximations are made regarding the masses of the outgoing $q\bar{q}$ pair.
- The MEPP2GammaGamma class implements the matrix element for the production of prompt photon pairs. In addition to the tree-level $q\bar{q} \to \gamma\gamma$ process the loop-mediated $gg \to \gamma\gamma$ process is included.
- Direct photon production in association with a jet is simulated using the MEPP2GammaJet class. As with the QCD $2 \to 2$ process all of the particles are treated as massless in these processes.
- The production of an s-channel Higgs boson via both $gg \to h^0$ and $q\bar{q} \to h^0$ is simulated using the MEPP2Higgs class.
- The production of a Higgs boson in association with the Z^0 or $W^{\pm}$ bosons is simulated using the MEPP2ZH or MEPP2WH class respectively.
- The production of the Higgs boson in association with a hard jet is simulated using the MEPP2HiggsJet class.

[4] t-channel photon and Z^0 boson exchange are not included.

In addition to the processes described above, which are intended for realistic physics studies, we have a small number of classes that are primarily intended to test certain aspects of the Herwig++ code. These are: the MEee2VectorMeson class for the simulation of an s-channel vector meson, which is used to test the hadron decays by producing the $\Upsilon(4S)$; the MEee2Z class, which produces an s-channel Z^0 boson in e^+e^- collisions to test the spin correlation effects in Z^0 decays; and the MEee2Higgs2SM class, which simulates s-channel Higgs boson production in e^+e^- collisions to test spin correlations in the decays of tau leptons produced in the decay of scalar and pseudoscalar Higgs bosons.

In addition we have one matrix element class, MEQCD2to2Fast, that uses hard-coded formulae for the QCD $2 \to 2$ scattering matrix elements rather than the helicity libraries of ThePEG. This class is significantly faster than the default MEQCD2to2 class, although it does not implement spin correlations. It is intended to be used in the generation of the multiple parton-parton scatterings for the underlying event where the spin correlations are not important but due to the number of additional scatterings that must be generated the speed of the calculation can significantly affect the run time of the event generator.

3.2 Les Houches interface

There are a number of matrix element generators available that can generate parton-level events using either the original Les Houches Accord [28] or the subsequent extension [29], which specified a file format for the transfer of the information between the matrix element generator and a general-purpose event generator, such as Herwig++, rather than the original FORTRAN COMMON block.

In addition to the internal mechanism for the generation of hard processes, ThePEG provides a general LesHouchesEventHandler class, which generates the hard process using the Les Houches Accord. In principle a run-time interface could be used to directly transfer the information between the matrix element generator and Herwig++, however we expect that the majority of such interfaces will be via data files containing the event information using the format specified in Ref. [29].

We expect that this approach will be used for the majority of hard processes in Herwig++.

3.3 Code structure

In ThePEG the generation of the hard process is the responsibility of the EventHandler. The base EventHandler class only provides the abstract interfaces for the generation of the hard process with the actual generation of the kinematics being the responsibility of inheriting classes. There are two such classes provided in ThePEG: the Standard

EventHandler, which implements the internal mechanism of ThePEG for the generation of the hard process; and the LesHouchesEventHandler, which allows events to be read from data files.

3.3.1 StandardEventHandler

The StandardEventHandler uses a SubProcessHandler to generate the kinematics of the particles involved in the hard process. In turn the SubProcessHandler makes use of a number of MEBase objects to calculate the matrix element and generate the kinematics for specific processes. The specific matrix elements used in a given run of the EventGenerator can be specified using the **MatrixElements** interface of the SubProcessHandler. The MEBase object is responsible for:

- defining the particles that interact in a given process, by specifying a number of DiagramBase objects; one DiagramBase is specified per flavour combination.
- returning the differential partonic cross section

$$\frac{\mathrm{d}\sigma}{\mathrm{d}r_1..\mathrm{d}r_n}, \tag{1}$$

  when supplied with the partonic centre-of-mass energy of the collision and n random numbers between 0 and 1. Each MEBase class specifies how many random numbers it requires to calculate the partonic cross section and kinematics for the processes it implements. For example a $2 \to 2$ process typically needs two[5] random numbers, one each for the polar and azimuthal angles.
- creating a HardVertex object describing the interaction that occurred, including the spin-unaveraged matrix element to allow spin correlation effects to be generated.

One MEBase object is generally used to describe one physical process with different partonic flavours. The selection of flavours within each subprocess is carried out internally by the EventHandler. The resulting cross sections can be output with varying levels of detail, controlled by the **StatLevel** switch; by default they are only broken down by MEBase objects. The SubProcessHandler then uses a SamplerBase object to perform the unweighting of the cross section and generate events with unit weight. In practice for $2 \to 2$ cross section the generation of the kinematics and other technical steps is handled by the ME2to2Base class. In addition the actual calculation of the matrix element can be easily implemented using the Helicity classes of ThePEG. All of the matrix elements in Herwig++ inherit[6] from ME2to2Base and make extensive use of the Helicity library of ThePEG.

[5] In practice as the matrix elements do not depend on the azimuthal angle we often only use one random number for the polar angle and generate the second random number locally.

[6] The only exception is the MEQCD2to2Fast class, which is 'hand written' for speed.

In general the main switch for the generation of the hard process is the **MatrixElements** interface, which allows the MEBase objects to be specified and hence determines which hard scattering processes are generated. In addition, each class inheriting from MEBase in Herwig++ has a number of parameters that control the incoming, outgoing and intermediate particles in a specific process. These are controlled by Interfaces in the specific matrix element classes. A number of different partonic subprocesses can be handled at the same time by simply specifying several MEBase objects.

3.3.2 LesHouchesEventHandler

The LesHouchesEventHandler class inherits from the EventHandler class of ThePEG. The class has a list of LesHouchesReader objects that are normally connected to files with event data produced by an external matrix element generator program, although it could in principle include a direct run-time link to the matrix element generator or read events 'on the fly' from the output of a matrix element generator connected to a pipe.

When an event is requested by LesHouchesEventHandler, one of the readers is chosen according to the cross section of the process for which events are supplied by that reader. An event is read in and subsequently handled in the same way as for an internally generated process. The use of the LesHouchesEventHandler class is described in Appendix B.8.

3.3.3 Kinematic cuts

For cuts on the hard process we use Cuts objects from ThePEG. All cuts applied to the generation of the hard process can be specified via its Interfaces. There are many types of cuts that can be applied.

Cuts applied to the overall hard process, such as a minimum or maximum invariant mass $\widehat{M}$ of the process, can be specified directly as a parameter of the Cuts class. The minimum value of the invariant mass for the hard process is set using the **MhatMin** parameter. Many more cuts can be specified by using the Interfaces of the Cuts class. Among those that are used in Herwig++ are cuts on the momentum fractions $x_{1,2}$ of the incoming partons and the hard process scale. The default set of cuts we apply in hadronic collisions is $\widehat{M} > 20\,\mathrm{GeV}$ (**MhatMin**), $x_{1,2} > 10^{-5}$ (**X1Min**, **X2Min**) and $Q > 1\,\mathrm{GeV}$ (**ScaleMin**).

In addition to these general cuts it is possible to specify cuts that are only applied to particular particles, particle pairs or resonant intermediate particles. In order to do so, one has to specify a number of OneCutBase, TwoCutBase or MultiCutBase objects in the Cuts object that is applied.

Whenever we use OneCutBase cuts we use SimpleKTCut objects. These require that a Matcher object is set up for the

particles to which the cut is applied. The Matcher classes used in Herwig++ all inherit from MatcherType. In addition to the Matcher classes provided by ThePEG, Herwig++ provides additional matchers for top quarks, TopMatcher, and photons, PhotonMatcher. This can be either a single particle, for example the top quark, or a group of particles, like the leptons. Then, for example, the minimum transverse momentum of that particle $k_{\perp,\min}$ can be specified as **MinKT**. In addition we use minimum and maximum values of pseudorapidity via **MinEta** and **MaxEta**. By default we use $k_{\perp,\min} > 20\,\text{GeV}$ for partons and $|\eta| < 3$ for photons from the hard scattering process.

An example of a MultiCutBase class is the V2LeptonsCut class. We use it to limit the invariant mass of lepton pairs. It is given similarly to the general cut as **MinM**. We use the rather loose cut $20\,\text{GeV} < M < 1.4\,\text{TeV}$ by default. Another useful parameter of this class is the specification of the lepton families (**Families**) or the charge combination (**CComb**) of the lepton pair the cut is applied to.

As the cuts are applied to all the particles produced in the collision, for $W^{\pm}/Z^0$ production in association with either a jet or a Higgs boson the cuts are also applied to the decay products of the boson. This can lead to inefficiencies in the generation of the hard process and a suppression of the hadronic boson decays with the default cuts on the quarks.

4 Perturbative decays and spin correlations

In Herwig++ the decays of the fundamental particles and the unstable hadrons are handled in the same way in order that correlation effects for particles such as the tau lepton, which is produced perturbatively but decays non-perturbatively, are correctly treated. Eventually it is intended that the unstable fundamental particles will be decayed during the parton-shower stage of the event, however currently in order that the correlation effects are correctly generated all the perturbative particle decays are performed before the generation of the parton shower by using the HwDecayHandler as one of the **PreCascadeHandlers** in the EventHandler responsible for generating the event. The Decayer classes used in Herwig++ to perform the decays of the fundamental Standard Model particles make use of the Helicity classes of ThePEG to calculate the helicity amplitudes for the decay matrix elements. The code structure for the Decayer classes used in Herwig++ and the HwDecayHandler implement the algorithm of Refs. [24–27] to correctly include the spin correlations.

In the next subsection we describe the spin correlation algorithm of [24–27] using the example of top production and decay. This is followed by a description of the modelling of the decay of the fundamental particles of the Standard Model, the production and decays of particles in models of physics Beyond the Standard Model is discussed in Sect. 5. We then describe the simulation of QED radiation in particle decays. Finally we briefly discuss the structure of the code for the decays of fundamental particles.

4.1 Spin correlations

When calculating the matrix element for a given hard process or decay one must take into account the effect of spin correlations, as they will affect the distributions of particles in the final state. In particular these correlations are important in the production and decay of the top quark, for the production of tau leptons in Higgs decays and in models of BSM physics where one can have two models that possess a very similar particle spectrum but with particles that have different spins.

An algorithm for correctly incorporating these correlations into a Monte Carlo is demonstrated in Refs. [24–27]. Rather than discuss the algorithm in full detail here we will describe it by considering the example of the process $e^+e^- \to t\bar{t}$ where the top quark subsequently decays, via a W^+ boson, to a b quark and a pair of light fermions.

Initially, the outgoing momenta of the $t\,\bar{t}$ pair are generated according to the usual cross-section integral

$$\frac{(2\pi)^4}{2s}\int \frac{d^3p_t}{(2\pi)^3 2E_t}\frac{d^3p_{\bar{t}}}{(2\pi)^3 2E_{\bar{t}}}\mathcal{M}^{e^+e^-\to t\bar{t}}_{\lambda_t\lambda_{\bar{t}}}\mathcal{M}^{*e^+e^-\to t\bar{t}}_{\lambda_t\lambda_{\bar{t}}}, \quad (2)$$

where $\mathcal{M}^{e^+e^-\to t\bar{t}}_{\lambda_t\lambda_{\bar{t}}}$ is the matrix element for the initial hard process and $\lambda_{t,\bar{t}}$ are the helicities of the t and $\bar{t}$ respectively. One of the outgoing particles is then picked at random, say the top, and a spin density matrix calculated

$$\rho^t_{\lambda_t\lambda_t'} = \frac{1}{N}\mathcal{M}^{e^+e^-\to t\bar{t}}_{\lambda_t\lambda_{\bar{t}}}\mathcal{M}^{*e^+e^-\to t\bar{t}}_{\lambda_t'\lambda_{\bar{t}}}, \quad (3)$$

with N defined such that $\text{Tr}\,\rho = 1$.

The top is decayed and the momenta of the decay products distributed according to

$$\frac{(2\pi)^4}{2m_t}\int \frac{d^3p_b}{(2\pi)^3 2E_b}\frac{d^3p_{W^+}}{(2\pi)^3 2E_{W^+}}\rho^t_{\lambda_t\lambda_t'}\mathcal{M}^{t\to bW^+}_{\lambda_t\lambda_{W^+}}\mathcal{M}^{*t\to bW^+}_{\lambda_t'\lambda_{W^+}}, \quad (4)$$

where the inclusion of the spin density matrix ensures the correct correlation between the top decay products and the beam.

A spin density matrix is calculated for the W^+ only, because the b is stable

$$\rho^{W^+}_{\lambda_{W^+}\lambda_{W^+}'} = \frac{1}{N}\rho^t_{\lambda_t\lambda_t'}\mathcal{M}^{t\to bW^+}_{\lambda_t\lambda_{W^+}}\mathcal{M}^{*t\to bW^+}_{\lambda_t'\lambda_{W^+}}, \quad (5)$$

and the W^+ decayed in the same manner as the top. Here the inclusion of the spin density matrix ensures the correct

correlations between the W^+ decay products, the beam and the bottom quark.

The decay products of the W^+ are stable fermions so the decay chain terminates here and a decay matrix for the W^+

$$D^{W^+}_{\lambda_{W^+}\lambda'_{W^+}} = \frac{1}{N}\mathcal{M}^{t\to bW^+}_{\lambda_t\lambda_{W^+}}\mathcal{M}^{*t\to bW^+}_{\lambda_t\lambda'_{W^+}}, \tag{6}$$

is calculated. Moving back up the chain a decay matrix for the top quark is calculated using the decay matrix of the W^+,

$$D^{t}_{\lambda_t\lambda'_t} = \frac{1}{N}\mathcal{M}^{t\to bW^+}_{\lambda_t\lambda_{W^+}}\mathcal{M}^{*t\to bW^+}_{\lambda'_t\lambda'_{W^+}}D^{W^+}_{\lambda_{W^+}\lambda'_{W^+}}. \tag{7}$$

Since the top came from the hard scattering process we must now deal with the $\bar{t}$ in a similar manner but instead of using $\delta_{\lambda_t\lambda'_t}$ when calculating the initial spin density matrix, the decay matrix of the top is used and the $\bar{t}$ decay is generated accordingly. The density matrices pass information from one decay chain to the associated chain thereby preserving the correct correlations.

The production and decay of the top, using the spin correlation algorithm, is demonstrated in Figs. 1–3. The hard scattering process and subsequent decays were generated using the general matrix elements described in Sect. 5 rather than the default ones. The results from the full matrix element calculation are also included to show that the algorithm has been correctly implemented. The separate plots illustrate the different stages of the algorithm at work. Figure 1 gives the angle between the beam and the outgoing lepton. The results from the simulation agree well with the full matrix element calculation, which demonstrates the consistency of the algorithm for the decay of the $\bar{t}$.

Figure 2 gives the angle between the top quark and the produced lepton. This shows the same agreement as the previous figure and demonstrates the correct implementation of the spin density matrix for the $\bar{t}$ decay. Finally, Fig. 3 gives the results for the angle between the final-state lepton/antilepton pair showing the correct implementation of the decay matrix that encodes the information about the $\bar{t}$ decay. Distributions for various processes within the Minimal Supersymmetric Standard Model and for tau production in Higgs decay are shown in Refs. [21, 22].

The same algorithm is used regardless of how the particles are produced, in order to consistently implement the spin correlations in all stages of the event generation process.

4.2 Standard model decays

There are a small number of decays of fundamental Standard Model particles currently implemented. These are implemented as Decayer classes for top quark, $W^\pm$ and Z^0, and Higgs boson decays. The following classes are available:

- the SMTopDecayer implements the three-body decay of the top quark to the bottom quark and a Standard Model fermion-antifermion pair, via an intermediate W^+ boson;
- the SMWZDecayer class implements the decay of the $W^\pm$ and Z^0 bosons to a Standard Model fermion-antifermion pair;
- the SMWZGammaDecayer class implements the decay of the $W^\pm$ and Z^0 bosons to a Standard Model fermion-antifermion pair with the radiation of a photon, it is only intended for comparison with the radiation of photons using the YFS formalism [30] by the SOPHTY DecayRadiationGenerator in particle decays [19];
- the SMHiggsFermionsDecayer class implements the decay of the Higgs boson to a Standard Model fermion-antifermion pair, *i.e.* $h^0 \to f\bar{f}$;
- the SMHiggsWWDecayer implements the decay of the Higgs boson to $W^\pm$ or Z^0 bosons, *i.e.* $h^0 \to W^+W^-$, Z^0Z^0, including the decay of the gauge bosons;
- the SMHiggsGGHiggsPPDecayer implements the decay of the Higgs boson to a pair of either gluons or photons.

In general, external top quarks and $W^\pm$ and Z^0 bosons are produced on mass-shell. In cases where we wish to include off-shell effects for the electroweak gauge bosons they are included as intermediate particles, for example in top quark and Higgs boson decays. In the future we will improve this to use the same treatment of off-shell effects we use in hadron decays, see Sect. 9. This approach is already implemented for the Higgs boson as in the FORTRAN HERWIG program together with the more sophisticated approaches described in Ref. [31].

4.3 QED radiation

The simulation of QED radiation using the approach of Ref. [20] has been included for both particle decays and unstable s-channel resonances produced in the hard process. This approach is based on the YFS formalism [30], which takes into account large double- and single- soft photon logarithms to all orders. In addition, the leading collinear logarithms are included to $\mathcal{O}(\alpha)$ by using the dipole splitting functions. By default the production of QED radiation is switched off for both decays and hard processes. It may be included by using the QEDRadiationHandler in the EventHandler as one of the **PostSubProcessHandlers** for the hard process or using the **PhotonGenerator** interface of the relevant Decayer inheriting from the DecayIntegrator class for the decays.

4.4 Code structure

The code structure for particle decays in Herwig++ is described in more detail in Sect. 9.6 for the hadronic decays. All of the Decayer classes for fundamental particles inherit

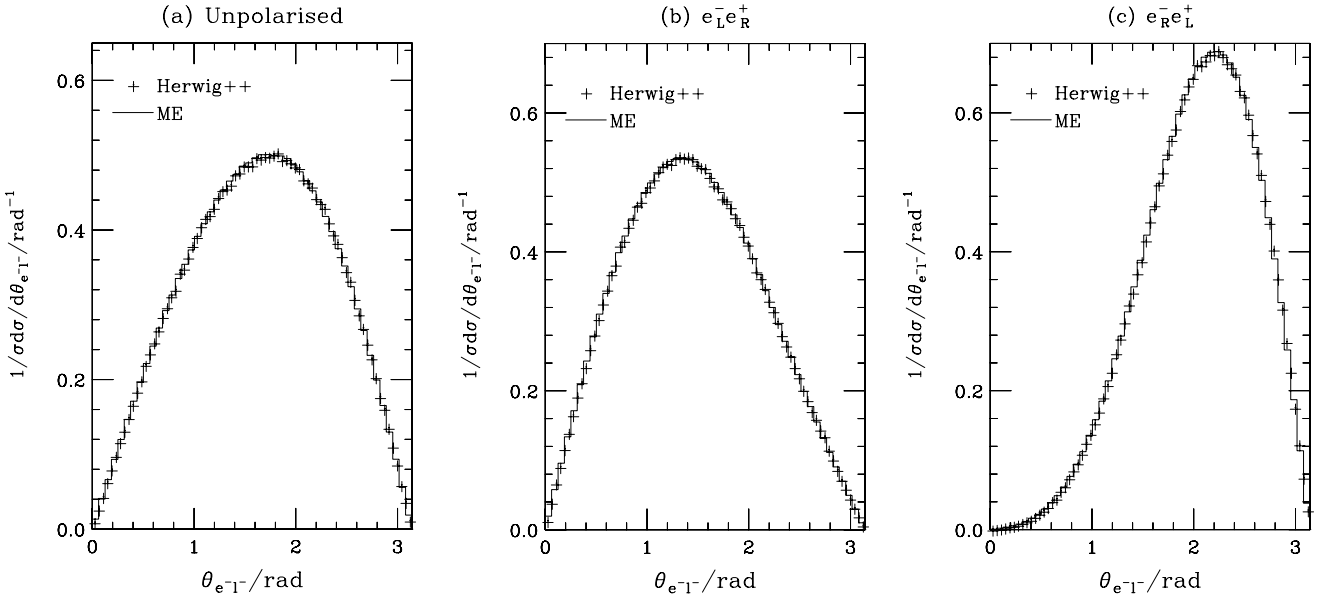


Fig. 1 Angle between the beam and the outgoing lepton in $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$ in the lab frame for a centre-of-mass energy of 500 GeV with (**a**) unpolarized incoming beams, (**b**) negatively polarized electrons and positively polarized positrons and (**c**) positively polarized electrons and negatively polarized electrons. The data points show the results of the simulation as production and decay including spin correlations, while the histograms use the full matrix element for $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$

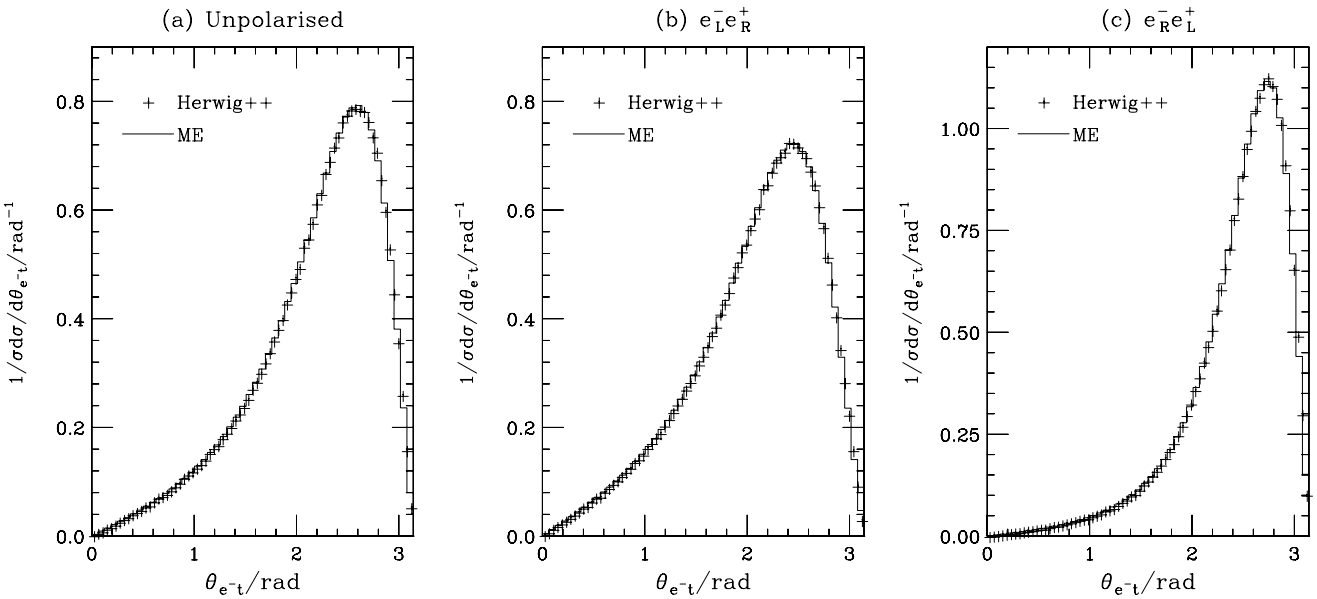


Fig. 2 Angle between the lepton and the top quark in $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$ in the lab frame for a centre-of-mass energy of 500 GeV with (**a**) unpolarized incoming beams, (**b**) negatively polarized electrons and positively polarized positrons and (**c**) positively polarized electrons and negatively polarized electrons. The data points show the results of the simulation as production and decay including spin correlations, while the histograms use the full matrix element for $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$

from the DecayIntegrator class in order to use the multi-channel phase-space integration it provides.

The SMHiggsMassGenerator implements the generation of the mass of off-shell Higgs bosons using the running width implemented in the SMHiggsWidthGenerator class. These classes inherit from the GenericMassGenerator and GenericWidthGenerator classes of Herwig++ in order to have access to the full infrastructure for the simulation of off-shell particles described in Sect. 9.

The structure of the code for the simulation of QED radiation in particle decays is designed to be general, so that other approaches can be implemented. The generation of the radiation is handled by a class inheriting from the abstract DecayRadiationGenerator class. Currently only the YFS ap-

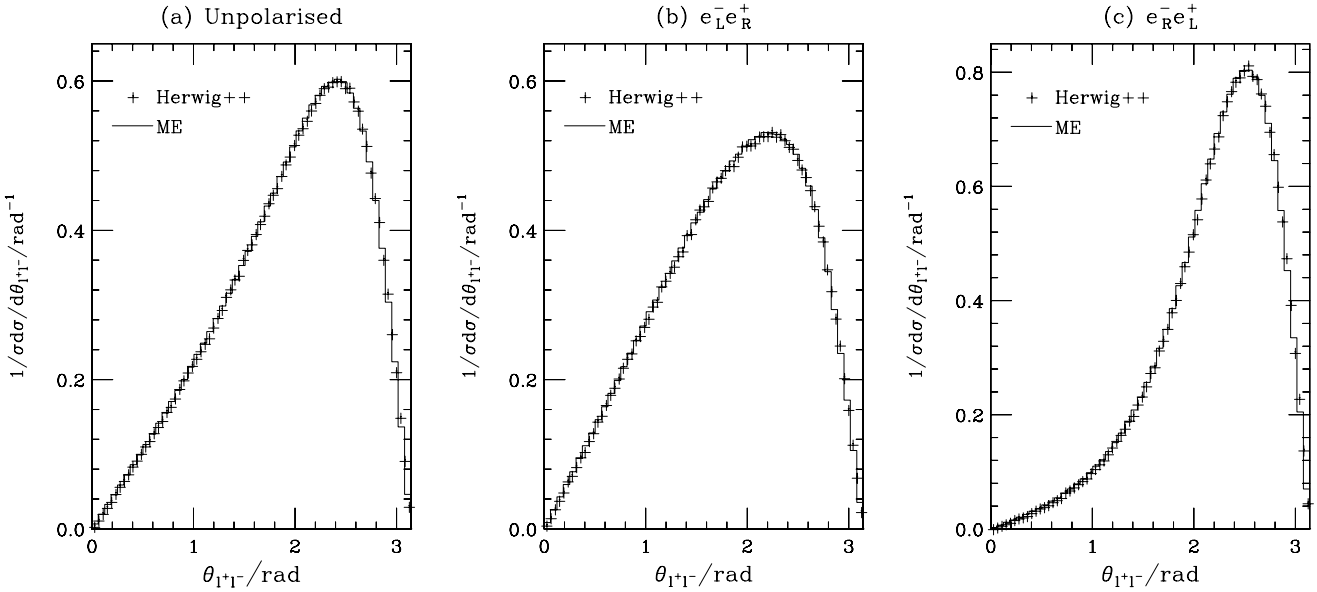


Fig. 3 Angle between the outgoing lepton and anti-lepton in $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$ in the lab frame for a centre-of-mass energy of 500 GeV with (**a**) unpolarized incoming beams, (**b**) negatively polarized electrons and positively polarized positrons and (**c**) positively polarized electrons and negatively polarized electrons. The data points show the results of the simulation as production and decay including spin correlations, while the histograms use the full matrix element for $e^+e^- \to t\bar{t} \to b\bar{b}l^+\nu_l l^-\bar{\nu}_l$

proach, as described in Ref. [20], is implemented in the SOPHTY class, which uses the helper FFDipole and IFDipole classes for radiation from final-final and initial-final dipoles, respectively. In addition the QEDRadiationHandler is included to allow the DecayRadiationGenerator to be used to generate radiation in the decay of particles generated as s-channel resonances in the hard process.

5 Physics beyond the standard model

No one knows what kind of physics will be encountered in the LHC era and it is likely that a variety of new physics models will need to be considered in determining its exact nature. This eventuality has been accounted for in the design of the Herwig++ program, by the inclusion of a general framework for the implementation of new physics models. Using this framework, new models can be realized quickly and efficiently. This method is described in full in Ref. [21] and will be reviewed here.

In describing the features needed to simulate Beyond the Standard Model (BSM) processes, we need only concern ourselves with the hard collisions, either producing known particles through modified couplings or the exchange of new particles, or producing new particles in the final state, and with decays of the new particles. All other steps of event generation are handled in the same way as for Standard Model processes.[7] Both of these steps involve calculating an amplitude, which in turn relies on knowledge of the Feynman rules within the model being used. In Herwig++ the Feynman rules are implemented as a series of Vertex classes, which inherit from the generic classes of ThePEG. These Vertex classes are based on the HELAS formalism [32], with each class able to evaluate the vertex as a complex number or, given different information, an off-shell wavefunction that can be used as input for another calculation. Each Feynman diagram contributing to a given process is evaluated in terms of these vertex building blocks and the sum of the resulting contributions is squared to give the matrix element.

In this section we start by briefly describing the generation of the hard processes and decays in models of new physics, this is followed by a description of models currently implemented in Herwig++, including the Standard Model, and the structure of the code.

5.1 Hard process

Section 3 gave details on the default matrix elements available for generating Standard Model processes in Herwig++. These classes are based on specific particle interactions whereas the classes used for BSM models are based on the external spin structure of a $2 \to 2$ scattering process. To generate a specific process the user specifies the desired states

[7] Other features do emerge in certain models, for example the hadronization of new long-lived coloured particles, which is not yet fully implemented in Herwig++, but for the majority of new physics models under active study this is the case.

that are to participate in the hard interaction, using the configuration files, and the code then generates the relevant diagrams and a MatrixElement object for each process.[8]

The generic matrix elements use a *colour flow* decomposition to calculate the value of $|\overline{\mathcal{M}}|^2$. This method cuts down on the amount of colour algebra necessary in the evaluation of QCD processes by rewriting the colour structures of certain diagrams in terms of others in the same process. As an example, consider the process $q_a\,\bar{q}_b \to \tilde{g}^c\,\tilde{g}^d$, which has diagrams with amplitudes given by

$$t^d_{bi}t^c_{ia}\mathcal{M}_t, \tag{8a}$$

$$t^c_{bi}t^d_{ia}\mathcal{M}_u, \tag{8b}$$

$$if^{cdi}t^i_{ba}\mathcal{M}_s, \tag{8c}$$

where $\mathcal{M}_{\{t,u,s\}}$ is the colour-stripped amplitude for each diagram type. Using the colour matrix identities, (8c) can be rewritten as $[t^c, t^d]_{ba}\mathcal{M}_s$ and is then a combination of the other two colour structures. By defining a colour flow f_i as a combination of colour-stripped amplitudes possessing the same colour structure, in this case $f_1 = \mathcal{M}_t - \mathcal{M}_s$ and $f_2 = \mathcal{M}_u + \mathcal{M}_s$, we can cut down the number of colour factors that need to be evaluated. The full matrix element squared, summed over final-state spins and colours and averaged over initial spins and colours, is obtained by adding up products of colour flows and the appropriate colour factor. For any process $a\,b \to c\,d$ this can be written as

$$|\overline{\mathcal{M}}|^2 = Z\frac{1}{S_a}\frac{1}{S_b}\frac{1}{C_a}\frac{1}{C_b}\sum_{\lambda} C_{ij}\,f_i^{\lambda}\,f_j^{*\lambda}, \tag{9}$$

where C_{ij} is a matrix containing the squared colour factors, f_i^{λ} denotes the ith colour flow for the set of helicities λ, Z is an identical particle factor, $S_{a,b}$ is the number of polarization states for each incoming particle and $C_{a,b}$ is the number of colour states for each incoming particle.

To carry out the parton showering and hadronization stages of the simulation we must assign a colour to each particle participating in each hard collision. This information is needed in determining the initial conditions for the parton shower (Sect. 6.3), and how clusters are formed in the hadronization model (Sect. 7). To this end, each fundamental coloured particle is associated to a ColourLine object. For the particles involved in the hard interactions, the colour assignments are made by selecting a colour flow from a list contained in the corresponding MatrixElement class as follows. Once a momentum configuration for the primary hard scattering has been generated, each colour flow is assigned a weight according to how much it contributes to the total value of the matrix element (neglecting the interference between them, which is typically suppressed by $1/N_c^2$ and also by dynamical effects). One of these colour flows is then probabilistically chosen on the basis of this weight distribution.

[8] It is only necessary to specify a single outgoing particle as the code will produce all processes with this particle in the final state.

5.2 Decays

To be able to decay the BSM states, the possible decay modes must first be known. If a supersymmetric model is required one can use a spectrum generator to produce not only the required spectrum, in accordance with the SUSY Les Houches Accord [33], but also a decay table. Herwig++ is designed to be able to read this information and set up the appropriate decay modes for later use. Other models do not have such programs and therefore the list of possible $1 \to 2$ decays is generated automatically.

When generating the possible decays automatically we also need to be able to calculate the partial width of a given mode so that the branching fraction and total width can be calculated. For a general two-body decay, the matrix element only depends on the mass-square values of each particle so the phase-space factor can be integrated separately and the partial width is given by

$$\Gamma(a \to b, c) = \frac{|\overline{\mathcal{M}}|^2 p_{cm}}{8\pi m_a^2}, \tag{10}$$

where $|\overline{\mathcal{M}}|^2$ is the matrix element squared summed over final-state colours and spins and averaged over initial-state colours and spins and p_{cm} is the centre-of-mass momentum

$$p_{cm} = \frac{1}{2m_a}[(m_a^2-(m_b+m_c)^2)(m_a^2-(m_b-m_c)^2)]^{1/2}. \tag{11}$$

The total width of the parent is then simply the sum of the partial widths.

To compute the momenta of the decay products we need to be able to calculate the matrix element for a selected decay mode. When each mode is created it is assigned a Decayer object that is capable of calculating the value of $|\mathcal{M}|^2$ for that process. It is done in a similar manner to the hard matrix element calculations, *i.e.* using the helicity libraries of ThePEG.

In decays involving coloured particles that have more than one possible colour flow, the colour is treated in exactly the same way as described in Sect. 5.1 for hard processes.

5.3 Model descriptions

This section will give a description of the models that are included in Herwig++. In general in Herwig++ the implementation of a physics model consists of a main class, which inherits from the StandardModel class and implements the

calculation of any parameters required by the model or, for a SUSY model, reads them from an input SUSY Les Houches file. In addition, there are various classes that inherit from the general Vertex classes of ThePEG, which implement the Feynman rules of the model. There may also be some classes implementing other features of the model, for example the running couplings in the specific model.

5.3.1 Standard model

The implementation of the Standard Model in Herwig++ inherits from the StandardModelBase class of ThePEG. ThePEG includes classes to implement the running strong and electromagnetic couplings, together with the CKM matrix.

In Herwig++ we include our own implementations of the running electromagnetic coupling, in the AlphaEM class, and the running strong coupling in the O2AlphaS class. By default we use the implementations of the running couplings from ThePEG and the Herwig++ implementations are only provided to allow us to make exact comparisons with the FORTRAN HERWIG program.

In order to perform helicity amplitude calculations we need access to the full CKM matrix. However the CKMBase class of ThePEG only provides the squares of the matrix elements. The StandardCKM class therefore provides access to the matrix elements as well and it is used in all our helicity amplitude calculations.

We have also included a structure for the implementation of running mass calculations. The RunningMassBase class provides a base class and the two-loop QCD running mass is implemented in the RunningMass class.

The Standard Model input parameters in Herwig++ do not form a minimal set in that it is possible to independently set the value of the weak mixing angle in such a way that the tree-level relationship between the $W^{\pm}$ and Z^0 boson masses is not satisfied. The electroweak parameters we use are:

- the value of the electromagnetic coupling at zero momentum transfer, [**EW/AlphaEM=137.04**];
- the value of $\sin^2\theta_W$, [**EW/Sin2ThetaW=0.232**];
- the masses of the $W^{\pm}$, $M_W = 80.403\,\text{GeV}$, and Z^0, $M_Z = 91.1876\,\text{GeV}$, bosons, which are taken from their ParticleData objects;
- the mixing angles, θ_{12} [**theta_12=0.2262**], θ_{13} [**theta_13=0.0037**] and θ_{23} [**theta_23=0.0413**], and phase, δ [**delta=1.05**], of the CKM matrix.

In addition, many of the Standard Model couplings to the Z^0 boson can be changed to simulate non-Standard Model effects if desired.

5.3.2 Minimal supersymmetric standard model

The Minimal Supersymmetric Standard Model (MSSM) is the most studied supersymmetric model and as such it should be included in any generator attempting to simulate BSM physics. As its name suggests it contains the smallest number of additional fields required for the theory to be consistent. The additional particle content over that of the Standard Model is listed in Table 2.

The additional particles must have masses and couplings to be of any use in an event simulation. For supersymmetric models various programs are available that, given some set of input parameters, produce a spectrum containing all of the other parameters necessary to be able to calculate physical quantities within the model. As stated in the previous section the output from such a generator must comply with the SUSY Les Houches Accord (SLHA) [33] for it to be used with Herwig++.

While reading the information from an SLHA file is straightforward, there is a minor complication when dealing with particle masses that have a mixing matrix associated with them. For example, consider the neutralinos, which are an admixture of the bino $\tilde{b}$, third wino $\tilde{w}_3$ and 2 higgsinos $\tilde{h}_1$ and $\tilde{h}_2$. The physical eigenstates $\tilde{\chi}_i^0$ are given by

$$\tilde{\chi}_i^0 = N_{ij}\tilde{\psi}_j^0, \tag{12}$$

where N_{ij} is the neutralino mixing matrix in the $\tilde{\psi}^0 = (-i\tilde{b}, -i\tilde{w}, \tilde{h}_1, \tilde{h}_2)^T$ basis. The diagonalized mass term for the gauginos is then $N^*\mathcal{M}_{\tilde{\psi}^0}N^{\dagger}$, which in general can produce complex mass values. To keep the mass values real the phase is instead absorbed into the definition of the corresponding field thereby yielding a strictly real mass and mixing matrix. There is however a price to be paid for this—while the masses are kept real they can become negative. For an event generator a negative mass for a physical particle does not make sense so we instead choose a complex-valued mixing matrix along with real and non-negative masses. If a negative mass is encountered while reading a Les Houches file, the physical mass is taken as the absolute value and the

Table 2 The additional particle content of the MSSM contained in Herwig++. The particle's PDG codes are the standard ones given by the Particle Data Group [34]

Spin	Particles
0	$\tilde{d}_L, \tilde{u}_L, \tilde{s}_L, \tilde{c}_L, \tilde{b}_1, \tilde{t}_1$
	$\tilde{e}_L, \tilde{\nu}_{eL}, \tilde{\mu}_L, \tilde{\nu}_{\mu L}, \tilde{\tau}_1, \tilde{\nu}_{\tau L}$
	$\tilde{d}_R, \tilde{u}_R, \tilde{s}_R, \tilde{c}_R, \tilde{b}_2, \tilde{t}_2$
	$\tilde{e}_R, \tilde{\mu}_R, \tilde{\tau}_2$
	H^0, A^0, H^+
1/2	$\tilde{g}, \tilde{\chi}_1^0, \tilde{\chi}_2^0, \tilde{\chi}_3^0, \tilde{\chi}_4^0, \tilde{\chi}_1^+, \tilde{\chi}_2^+$

appropriate row of the mixing matrix is multiplied by a factor of i. This approach is used in order to facilitate the implementation of extended supersymmetric models in the future.

5.3.3 Randall-Sundrum model

The first models proposed with extra dimensions were of the Randall-Sundrum (RS) [35] type where a tensor particle, namely the graviton, is included and is allowed to propagate in the extra dimensions. All other matter, however, is restricted to our usual 4D brane and as a result all of the SM couplings are left unchanged. The only extra couplings required are those of the graviton to ordinary matter, which depend on a single parameter Λ_π.

5.3.4 Minimal universal extra dimensions model

We also include a model based on the idea of universal extra dimensions where all fields are allowed to propagate in the bulk. Following similar lines to supersymmetry, the model included in Herwig++ is of a minimal type and has a single compact extra dimension of radius R [36].

Compactifying the extra dimension and allowing all fields to propagate in it leads to a rich new structure within the theory. Analogous to the particle-in-a-box scenario, one obtains an infinite number of excitations of the fields all characterized by a quantity called the KK-number. This is most easily demonstrated by showing how a scalar field Φ would decompose after compactification

$$\Phi(x^\mu, y) = \frac{1}{\sqrt{\pi R}}\left[\Phi_0(x^\mu) + \sqrt{2}\sum_{n=1}^{\infty}\Phi_n(x^\mu)\cos\left(\frac{ny}{R}\right)\right], \tag{13}$$

where x^μ are the 4D coordinates, y the position in the 5th dimension and n is the KK-number of the mode with $n = 0$ identified as the SM mode. In general, in some compactification schemes, it is possible to have KK-number-violating interactions but in the Minimal Universal Extra Dimensions (MUED) framework in Herwig++ we include only those interactions that conserve KK-parity $P = (-1)^n$ and also limit ourselves to $n = 1$.

Table 3 shows the MUED particle content contained in Herwig++ along with their particle ID codes as these have not been standardized by the Particle Data Group [34]. Unlike the MSSM there are no external programs available that calculate the mass spectrum so this must be done internally by the UEDBase class, which inherits from the StandardModel class and implements the UED model. At tree level the mass of any level-n particle is simply given by $(m_0^2 + (n/R)^2)^{1/2}$, where m_0 is the mass of the SM particle, and $1/R$ is generally much larger than the SM mass so the spectrum is highly degenerate and no decays can occur. This situation changes once radiative corrections are taken into account and a spectrum that can be phenomenologically similar to the MSSM arises. The full set of radiative corrections, as derived in Ref. [37], is incorporated in the UEDBase class to give a realistic spectrum.

Table 3 The MUED particle spectrum contained in Herwig++ along with their ID codes. $^\bullet$ denotes a doublet under SU(2) and $^\circ$ a singlet. As with the standard PDG codes an antiparticle is given by the negative of the number in the table

Spin	Particle	ID code	Spin	Particle	ID code
0	h_1^0	5100025	1	g_1^*	5100021
	A_1^0	5100036		γ_1^*	5100022
	H_1^+	5100037		Z_1^{0*}	5100023
				W_1^{+*}	5100024
1/2	$d_1^\bullet$	5100001	1/2	d_1°	6100001
	$u_1^\bullet$	5100002		u_1°	6100002
	$s_1^\bullet$	5100003		s_1°	6100003
	$c_1^\bullet$	5100004		c_1°	6100004
	$b_1^\bullet$	5100005		b_1°	6100005
	$t_1^\bullet$	5100006		t_1°	6100006
	$e_1^{-\bullet}$	5100011		$e_1^{-\circ}$	6100011
	$\nu_{e1}^\bullet$	5100012			
	$\mu_1^{-\bullet}$	5100013		$\mu_1^{-\circ}$	6100013
	$\nu_{\mu 1}^\bullet$	5100014			
	$\tau_1^{-\bullet}$	5100015		$\tau_1^{-\circ}$	6100015
	$\nu_{\tau 1}^\bullet$	5100016			

5.4 Code structure

The ModelGenerator class is responsible for setting up the new MatrixElement objects, which inherit from the GeneralHardME class, and DecayMode objects for a new physics model. Helper classes aid in the creation of these objects, they are:

HardProcessConstructor this class is responsible for creating the diagrams for the requested processes and constructing the appropriate GeneralHardME object(s);

ResonantProcessConstructor this class is of a similar design to the HardProcessConstructor but it only constructs the resonant diagrams for a process;

DecayConstructor the DecayConstructor stores a collection of objects that inherit from the NBodyDecayConstructor class. Each of these is responsible for constructing all of the decay modes for the $1 \to N$ decays. Currently only the TwoBodyDecayConstructor class, for two-body decays, and the WeakCurrentDecayConstructor class, for weak decays using

the weak currents from Sect. 9.3.1 for decays where two particles are almost mass degenerate, are implemented.

The matrix element classes all inherit from the GeneralHardME class and implement the matrix element for a particular spin configuration. The classes inheriting from the GeneralHardME class and the spin structures they implement are given in Table 4.

All of the on-shell decayer classes inherit from the GeneralTwoBodyDecayer class and each is responsible for calculating the value of the matrix element for that particular set of spins. There is also a GeneralTwoBodyCurrentDecayer class for decay modes created with the WeakCurrentDecayConstructor class. The Decayer classes implemented in Herwig++ and the types of decay they implement are given in Table 5.

The use of BSM physics models is described in Appendix B.6 where examples of using all the models included with the release are given.

The specification of the particles involved in the hard process is achieved through the **Incoming** and **Outgoing** interfaces of the HardProcessConstructor. Both interfaces are lists of ParticleData objects. The switch **IncludeEW** can be set to **No** to include only the strong coupling diagrams.

In order to pass spin correlations through the decay stage, DecayIntegrator objects must be created. This is achieved by populating a list held in the ModelGenerator class, which can be accessed through the **DecayParticles** interface. The particles in this list will have spin correlation information passed along when their decays are generated. If a decay table is read in for a SUSY model then the **CreateDecayModes** interface should be set to **No** so that only the decay modes listed in the externally generated decay table are created.[9] For all other models the possible decay modes are also created from the particles in the **DecayParticles** list.

In addition to the code that handles the calculation of the matrix elements for the decays and scattering cross sections each model requires a number of classes to implement the model.

The Standard Model is implemented in the StandardModel class, which inherits from the StandardModelBase class of ThePEG and implements access to the helicity Vertex classes and some additional couplings, such as the running mass, used by Herwig++. The Vertex classes that implement the Standard Model interactions are given in Table 6.

The structure of the implementation of the MSSM in Herwig++ is designed to allow extended SUSY models to be added in the future. Therefore the SusyBase class, which inherits from the StandardModel class, is designed to read in the SLHA files specifying the SUSY spectrum. The details of the MSSM are implemented in the MSSM class, which inherits from the SusyBase class. The Vertex classes for the MSSM are given in Table 7. A spectrum file in SLHA format must be supplied, as described in Appendix B.6.1, or the MSSM model cannot be used.

The UED model is implemented in the UEDBase class, which inherits from the StandardModel class and implements the calculation of the parameters of the model. The Vertex classes for the UED model are given in Table 8.

There are three parameters that can be set to control the UED model: the inverse of the radius of compactification R^{-1}; the cutoff scale Λ; and the mass of the Higgs boson at the boundary of the compactified dimension $\overline{m}_h$. These are controlled through the interfaces:

Table 4 The general hard process matrix elements, based on spin structures, implemented in Herwig++

Class Name	Hard Process
MEff2ff	Fermion fermion to fermion fermion
MEff2ss	Fermion fermion to scalar scalar
MEff2vs	Fermion fermion to vector scalar
MEff2vv	Fermion fermion to vector vector
MEfv2fs	Fermion vector to fermion scalar
MEfv2vf	Fermion vector to vector fermion
MEvv2ff	Vector vector to fermion fermion
MEvv2ss	Vector vector to scalar scalar
MEvv2vv	Vector vector to vector vector

Table 5 The general decays based on spin structures implemented in Herwig++

Class Name	Decay
FFSDecayer	Fermion to fermion scalar decay
FFVDecayer	Fermion to fermion vector decay
FFVCurrentDecayer	Fermion to fermion vector decay with the vector off-shell and decaying via a weak current from Sect. 9.3.1
SFFDecayer	Scalar to fermion fermion decay
SSSDecayer	Scalar to two scalar decay
SSVDecayer	Scalar to scalar vector decay
SVVDecayer	Scalar to two vector decay
SVVLoopDecayer	Scalar to two vector decay via a loop
VFFDecayer	Vector to two fermion decay
VSSDecayer	Vector to two scalar decay
VVVDecayer	Vector to two vector decay
TFFDecayer	Tensor to two fermion decay
TSSDecayer	Tensor to two scalar decay
TVVDecayer	Tensor to two vector decay

[9] If a decay table is being used with a SUSY model then the DecayParticles list must still be populated so that the decays will have spin correlation information included.

InverseRadius the value of R^{-1}, the default value is 500 GeV;

LambdaR the dimensionless number ΛR, the default value is 20;

HiggsBoundaryMass the value of the Higgs mass at the boundary, the default value is 0 GeV.

The RSModel class inherits from the StandardModel class and implements the calculations needed for the Randall-Sundrum model. We have only implemented the vertices that are phenomenologically relevant and therefore some four-point vertices that are not important for resonance graviton production are not included. The Vertex classes implemented for the Randall-Sundrum model are given in Table 9.

Two parameters can be controlled in the Randall-Sundrum model; the cutoff Λ_π and the mass of the graviton. The default mass of the graviton is 500 GeV and this can be changed via the **NominalMass** interface of its ParticleData object. The cutoff is set via the **Lambda_pi** interface of the RSModel object and has a default value of 10 TeV.

The full list of interfaces for all the classes is provided in the Doxygen documentation.

Table 6 Herwig++ Vertex classes for the Standard Model

Class	Interaction
SMFFGVertex	Gluon with the SM fermions
SMFFPVertex	Photon with the SM fermions
SMFFWVertex	$W^\pm$ boson with the SM fermions
SMFFZVertex	Z^0 boson with the SM fermions
SMFFHVertex	Higgs boson with the SM fermions
SMGGGVertex	Triple gluon vertex
SMGGGGVertex	Four gluon vertex
SMWWWVertex	Triple electroweak gauge boson vertex
SMWWWWVertex	Four electroweak gauge boson vertex
SMWWHVertex	Higgs boson and weak gauge bosons
SMHGGVertex	Higgs boson coupling to two gluons via quark loops
SMHPPVertex	Higgs boson coupling to two photons via fermion and boson loops

6 Parton showers

A major success of the original HERWIG program was its treatment of soft gluon interference effects, in particular the phenomenon of *colour coherence*, via the angular ordering of emissions in the parton shower [1, 38–46]. Herwig++ simulates parton showers using the *coherent branching algorithm* of [17], which generalizes that used in the original HERWIG program [1–3]. The new algorithm retains angular ordering as a central feature and improves on its predecessor in a number of ways, the most notable of these being:

– a covariant formulation of the showering algorithm, which is invariant under boosts along the jet axis;
– the treatment of heavy quark fragmentation through the use of mass-dependent splitting functions [47] and kinematics, providing a complete description of the so-called *dead-cone* region.

Table 7 Herwig++ Vertex classes for the MSSM

Class	Interaction
SSNFSVertex	Neutralino with a SM fermion and a sfermion
SSCFSVertex	Chargino with a SM fermion and a sfermion
SSGFSVertex	Gluino with a quark and squark
SSNNZVertex	A pair of neutralinos with a Z^0 boson
SSCCZVertex	A pair of charginos with a Z^0 boson
SSCNWVertex	Chargino with a neutralino and a $W^\pm$ boson
SSGSGSGVertex	SM gluon with a pair of gluinos
SSGSSVertex	SM gluon with a pair of squarks
SSWSSVertex	SM gauge boson with a pair of sfermions
SSFFHVertex	A pair of SM fermions with a Higgs boson
SSWHHVertex	SM electroweak gauge bosons with a pair of Higgs bosons
SSWWHVertex	A pair of gauge bosons with a Higgs boson
SSGOGOHVertex	A pair of gauginos with a Higgs boson
SSHSFSFVertex	A Higgs boson with a pair of sfermions
SSHHHVertex	Triple Higgs boson self coupling
SSHGGVertex	A Higgs boson with a pair of gluons via quark and squark loops
SSGGSQSQVertex	A pair of gluons with a pair of squarks

Table 8 Herwig++ Vertex classes for the UED model

Class	Interaction
UEDF1F1P0Vertex	SM photon with a pair of KK-1 fermions
UEDF1F1W0Vertex	SM $W^{\pm}$ boson with a pair of KK-1 fermions
UEDF1F1Z0Vertex	SM Z^0 boson with a pair of KK-1 fermions
UEDF1F1G0Vertex	SM gluon with a pair of KK-1 fermions
UEDF1F0W1Vertex	KK-1 fermion with an EW KK-1 boson and a SM fermion
UEDF1F0G1Vertex	KK-1 fermion with a KK-1 gluon and a SM fermion
UEDF1F0H1Vertex	KK-1 fermion with a KK-1 Higgs boson and a SM fermion
UEDP0H1H1Vertex	SM photon with a pair of KK-1 charged Higgs boson
UEDW0W1W1Vertex	A pair of KK-1 gauge bosons with a SM $W^{\pm}$ or Z^0 boson
UEDG1G1G0Vertex	A pair of KK-1 gluons with a SM gluon
UEDG0G0G1G1Vertex	A pair of SM gluons with a pair of KK-1 gluons
UEDW0A1H1Vertex	SM $W^{\pm}$ boson with a KK-1 charged Higgs boson and a KK-1 pseudoscalar Higgs boson
UEDZ0H1H1Vertex	SM Z^0 boson with a pair of KK-1 charged Higgs boson
UEDZ0A1h1Vertex	SM Z^0 boson with a KK-1 pseudoscalar Higgs boson and a KK-1 scalar Higgs boson

Table 9 Herwig++ Vertex classes for the Randall-Sundrum model

Class	Interaction
RSModelFFGRVertex	Graviton to SM fermions
RSModelSSGRVertex	Graviton to the Higgs boson
RSModelFFVGRVertex	Graviton to two SM fermions and a gauge boson
RSModelVVGRVertex	Graviton to two gauge bosons
RSModelVVVGRVertex	Graviton to three gauge bosons

In this section we give a full description of the parton shower model and its implementation in the program. We begin by introducing the fundamental kinematics and dynamics underlying the shower algorithm. This is followed by descriptions of the initial conditions and the Monte Carlo algorithms used to generate the showers. Toward the end of the section we discuss how some next-to-leading log corrections can be included by a redefinition of the running coupling constant and process-specific matrix element corrections. The section concludes with details of the C++ code structure.

6.1 Shower kinematics

Each colour-charged leg of the hard sub-process is considered to be a *shower progenitor*. We associate a set of basis vectors to each progenitor, in terms of which we can express the momentum (q_i) of each particle in the resulting shower as

$$q_i = \alpha_i p + \beta_i n + q_{\perp i}. \tag{14}$$

This is the well known *Sudakov basis*. The vector p is equal to the momentum of the shower progenitor generated by the prior simulation of the hard scattering process, *i.e.* $p^2 = m^2$, where m is the on-shell mass of the progenitor. The *reference vector* n is a light-like vector that satisfies $n \cdot p > m^2$. In practice n is chosen anticollinear to p in the frame where the shower is generated, maximizing $n \cdot p$. Since we almost always generate the shower in the rest frame of the progenitor and an object with which it shares a colour line, n is therefore collinear with this *colour partner* object. The $q_{\perp i}$ vector gives the remaining components of the momentum, transverse to p and n.

In fullness, our basis vectors satisfy the following relations:

$$\begin{aligned} &q_{\perp i} \cdot p = 0, \qquad p^2 = m^2, \qquad q_{\perp i}^2 = -\mathbf{q}_{\perp i}^2, \\ &q_{\perp i} \cdot n = 0, \qquad n^2 = 0, \qquad n \cdot p > m^2, \end{aligned} \tag{15}$$

where $\mathbf{q}_{\perp i}$ is the spatial component of $q_{\perp i}$ in the frame where the shower is generated ($\mathbf{q}_{\perp i}^2 \geq 0$). Given these definitions, calculating q_i^2, one finds that β_i may be conveniently expressed in terms of the mass and transverse momentum of particle i as

$$\beta_i = \frac{q_i^2 - \alpha_i^2 m^2 - q_{\perp i}^2}{2\alpha_i n \cdot p}. \tag{16}$$

The shower algorithm does not generate the momenta or Sudakov parameters directly. In practice what is generated first is a set, each element of which consists of three *shower variables*, which fully parameterize each parton branching. One of these variables parameterizes the scale of each branching, the so-called *evolution scale*, which we shall discuss in more detail below. Typically this evolution scale

starts at a high value, characteristic of the process that produces the progenitors, and continually reduces as the shower develops, via the radiation of particles. When the evolution scale has reduced to the point where there is insufficient phase space to produce any more branchings, the resulting partons are considered to be on-shell, and the reconstruction of the momenta from the shower variables may begin in full. We now define these shower variables.

The first shower variable we introduce is the *light-cone momentum fraction* z. Given a branching, $\widetilde{ij} \to i + j$,[10] this parameterizes how the momentum component of the parent parton, $\widetilde{ij}$, in the direction of the shower progenitor, is divided between its two daughter partons, i and j. We define z as

$$z = \frac{\alpha_i}{\alpha_{\widetilde{ij}}} = \frac{n \cdot q_i}{n \cdot q_{\widetilde{ij}}}. \tag{17}$$

For particles in the final state we use a forward evolution algorithm where the parton shower consists of a sequence of branchings $\widetilde{ij} \to i + j$, ordered in the evolution scale. For incoming particles we use a backward evolution algorithm where we start at the large evolution scale of the scattering process and evolve the incoming particles backwards toward the incoming hadron to give the mother $\widetilde{ij}$ and the sister parton j, again with a decreasing evolution scale. We use the definition of z in (17) both for forward and backward parton shower algorithms.

The second variable used to parameterize a branching is the azimuthal angle, ϕ, of the relative transverse momentum of each branching $p_\perp$, measured with respect to the p direction. The relative transverse momentum $p_\perp$ is *defined* to be

$$p_\perp = q_{\perp i} - z q_{\perp \widetilde{ij}}. \tag{18}$$

As with the definition of z, this definition of the relative transverse momentum is the same for both forward and backward parton-shower evolution algorithms.

The last, and most important, of the shower variables defining a branching is the evolution scale. Parton shower algorithms may be formulated as an evolution in the virtualities of the branching partons, or as an evolution in the transverse momentum of the branching products. However, a careful treatment of colour coherence effects [1, 38–46] reveals that branchings involving soft gluons should be ordered in the angle between the branching products.

The key finding in these studies is that, when soft gluon emissions are considered, branchings that are not angular ordered do not give any leading logarithmic contributions. This is a dynamical effect whereby radiation from the emitting partons, with smaller angular separations, interferes destructively in these non-ordered regions. Some intuitive understanding of the effect may be gained by considering that a soft gluon, emitted at a large angle from a jet-like configuration of partons, does not have sufficient transverse resolving power to probe the internal jet structure. As a result, it is only sensitive to the *coherent sum* of the collinear singular contributions associated with the constituents, resulting in a contribution equivalent to that from the original progenitor parton. Destructive interference in the non-ordered region effectively decreases the available phase space for each branching, from the virtuality-ordered region to the angular-ordered region.

It may be shown that the contributions that angular ordering misses are purely soft and suppressed by at least one power of N_C^2, where $N_C = 3$, the number of colours in QCD. Formally then, omitting such contributions amounts to neglecting terms of next-to-leading-log accuracy that are *also* strongly colour suppressed. We stress however, that whereas angular ordering leads to an *omission* of these suppressed higher order terms, other forms of ordering must prove that they do not overestimate leading-log contributions.

For the forward evolution of partons with time-like virtualities, the variable used to achieve such ordering, $\tilde{q}^2$, is defined according to

$$z(1-z)\tilde{q}^2 = -m^2_{\widetilde{ij}} + \frac{m_i^2}{z} + \frac{m_j^2}{1-z} - \frac{p_\perp^2}{z(1-z)}, \tag{19}$$

where m_i is the on-shell mass of particle i *etc.* This definition is arrived at by generalizing the FORTRAN HERWIG angular evolution variable, $\tilde{q}^2 = q^2_{\widetilde{ij}}/(z(1-z))$, to include the effects of the mass of the emitting parton. This may be seen by writing $q_{\widetilde{ij}} = q_i + q_j$, and calculating $q^2_{\widetilde{ij}}(z, p_\perp^2, q_i^2, q_j^2)$, which shows

$$\tilde{q}^2 = \left.\frac{q^2_{\widetilde{ij}} - m^2_{\widetilde{ij}}}{z(1-z)}\right|_{q_i^2 = m_i^2,\ q_j^2 = m_j^2}. \tag{20}$$

For showers involving the evolution of partons with space-like virtualities, the evolution variable is instead defined by

$$(1-z)\tilde{q}^2 = -z m^2_{\widetilde{ij}} + m_i^2 + \frac{z m_j^2}{1-z} - \frac{p_\perp^2}{1-z}. \tag{21}$$

Once again this definition of the evolution variable is a generalization of the analogous FORTRAN HERWIG angular evolution variable used for initial-state radiation: $\tilde{q}^2 = q_i^2/(1-z)$. Using momentum conservation, $q_{\widetilde{ij}} = q_i + q_j$, we may calculate $q_i^2(z, p_\perp^2, q^2_{\widetilde{ij}}, q_j^2)$, whence one finds

$$\tilde{q}^2 = \left.\frac{m_i^2 - q_i^2}{1-z}\right|_{q^2_{\widetilde{ij}} = m^2_{\widetilde{ij}},\ q_j^2 = m_j^2}. \tag{22}$$

[10] We reserve the tilde notation $\widetilde{ij}$ exclusively to denote the parent parton, which decays into daughters i and j.

To see how these variables relate to the angle between the branching products, consider that the parton shower is generated in the frame where the light-like basis vector n is anticollinear to the progenitor. For forward evolving partons with small time-like virtualities, expanding z and $q^2_{\tilde{i}\tilde{j}}$ in component form, one finds

$$\tilde{q}^2 = \frac{2E^2_{\tilde{i}\tilde{j}}(1-\cos\theta_{ij})(1+\cos\theta_{\tilde{i}\tilde{j}})^2}{(1+\cos\theta_i)(1+\cos\theta_j)}, \tag{23}$$

where θ_i and θ_j are the angles between the daughter particles i, j and the progenitor, $\theta_{\tilde{i}\tilde{j}}$ is the angle between the parent and the progenitor, and θ_{ij} is the angle between the two daughters. $E_{\tilde{i}\tilde{j}}$ denotes the energy of the parent. This expression for the time-like evolution variable in terms of angles is more complicated than the analogous FORTRAN HERWIG formula: $\tilde{q}^2 = 2E^2_{\tilde{i}\tilde{j}}(1-\cos\theta_{ij})$. This is due to the fact that in FORTRAN HERWIG z was defined to be the energy fraction $E_i/E_{\tilde{i}\tilde{j}}$, instead of the light-cone momentum fraction as given in (17). Nevertheless, for small angles we find that the Herwig++ and FORTRAN HERWIG evolution variables are both given by

$$\tilde{q} = E_{\tilde{i}\tilde{j}}\theta_{ij}(1-\mathcal{O}(\theta_x^2)). \tag{24}$$

When a branching occurs, the daughter partons i and j, with momentum fractions z and $1-z$, have their starting evolution scales set to $z\tilde{q}$ and $(1-z)\tilde{q}$ respectively, where $z\tilde{q} \approx E_i\theta_{ij}$ and $(1-z)\tilde{q} \approx E_j\theta_{ij}$. In this way the maximum opening angle of any subsequent branching is θ_{ij}, thereby implementing angular ordering.

For initial-state showers the same QCD coherence argument applies, so in evolving backwards, away from the hard process, the angle between the mother of the branching and its final-state daughter parton must decrease. Writing the space-like evolution variable (see (21)) in terms of angles, neglecting parton virtualities, one finds the same form as for the time-like variable in (24). This means that once a branching has occurred in the course of the backward evolution, the mother of the branching evolves backward from scale $\tilde{q}$, and the daughter evolves forward from scale $(1-z)\tilde{q}$, as in the time-like case.

As stated above, when the evolution in terms of the shower variables has run its course, *i.e.* there is no more phase space available for further emissions, the external particles are taken as being on-shell and the reconstruction in terms of the physical momenta can start. First, all of the α coefficients in the Sudakov decomposition of each momentum are calculated. This is done by first setting α equal to one for final-state progenitors and to the associated PDF light-cone momentum fraction x, generated in the preceding simulation of the hard process, for initial-state progenitors. Using the defining z relation (17), together with the momentum conservation relation $\alpha_{\tilde{i}\tilde{j}} = \alpha_i + \alpha_j$, one can iteratively calculate all α values, starting from the hard process and working outward to the external legs.

For final-state showers the $q_\perp$ components of each momentum may be simultaneously calculated. Final-state showering cannot change the direction of the progenitor since the transverse momentum must be conserved at each branching, hence the $q_\perp$ component of the progenitor is zero. The $q_\perp$ components of the branching products are iteratively computed by adding the relative transverse momentum,

$$p_\perp = (|\mathbf{p}_\perp|\cos\phi, |\mathbf{p}_\perp|\sin\phi, 0; 0), \tag{25}$$

to z times the transverse momentum of the mother, $q_{\perp\tilde{i}\tilde{j}}$, to give $q_{\perp i}$ according to (18); $q_{\perp j} = q_{\perp\tilde{i}\tilde{j}} - q_{\perp i}$ immediately follows by momentum conservation. The magnitude of the relative transverse momentum $|\mathbf{p}_\perp| = \sqrt{-p_\perp^2}$ is calculated in terms of the evolution variables z and $\tilde{q}^2$ using (19).

The only remaining Sudakov parameters to be determined are the β values. These can be obtained once the evolution in terms of the shower variables is complete, by using the fact that the external partons are on-shell, in order to compute their β coefficients from (16). The coefficients of their parent momenta may then be computed using momentum conservation: $\beta_{\tilde{i}\tilde{j}} = \beta_i + \beta_j$. The latter step may be iterated until the progenitor is reached, yielding all β coefficients.

The reconstruction of the initial-state parton showers is slightly different but it follows essentially the same reasoning. Our aim here has been to simply sketch how the reconstruction occurs. More detailed presentations of these procedures will be given later in Sects. 6.4, 6.5 and 6.6.

6.2 Shower dynamics

With the kinematics defined, we now consider the dynamics governing the parton branchings. Each parton branching is approximated by the *quasi-collinear limit* [47], in which the transverse momentum squared, $\mathbf{p}_\perp^2$, and the mass squared of the particles involved are small (compared to $p\cdot n$) but $\mathbf{p}_\perp^2/m^2$ is not necessarily small. In this limit the probability of the branching $\tilde{i}\tilde{j} \to i+j$ can be written as

$$\mathrm{d}\mathcal{P}_{\tilde{i}\tilde{j}\to ij} = \frac{\alpha_S}{2\pi}\frac{\mathrm{d}\tilde{q}^2}{\tilde{q}^2}\,\mathrm{d}z\,P_{\tilde{i}\tilde{j}\to ij}(z,\tilde{q}), \tag{26}$$

where $P_{\tilde{i}\tilde{j}\to ij}(z,\tilde{q})$ are the quasi-collinear splitting functions derived in [47]. In terms of our light-cone momentum fraction and (time-like) evolution variable the quasi-collinear splitting functions are

$$P_{q\to qg} = \frac{C_F}{1-z}\left[1+z^2-\frac{2m_q^2}{z\tilde{q}^2}\right], \tag{27a}$$

$$P_{g\to gg} = C_A\left[\frac{z}{1-z} + \frac{1-z}{z} + z(1-z)\right], \tag{27b}$$

$$P_{g\to q\bar{q}} = T_R\left[1 - 2z(1-z) + \frac{2m_q^2}{z(1-z)\tilde{q}^2}\right], \tag{27c}$$

$$P_{\tilde{g}\to\tilde{g}g} = \frac{C_A}{1-z}\left[1 + z^2 - \frac{2m_{\tilde{g}}^2}{z\tilde{q}^2}\right], \tag{27d}$$

$$P_{\tilde{q}\to\tilde{q}g} = \frac{2C_F}{1-z}\left[z - \frac{m_{\tilde{q}}}{z\tilde{q}^2}\right], \tag{27e}$$

for QCD and singular SUSY QCD branchings.[11] These splitting functions give a correct physical description of the dead-cone region $\mathbf{p}_\perp \lesssim m$, where the collinear singular limit of the matrix element is screened by the mass m of the emitting parton.

The soft limit of the splitting functions is also important. The splitting functions with soft singularities $P_{q\to qg}$, $P_{\tilde{q}\to\tilde{q}g}$, $P_{g\to gg}$, and $P_{\tilde{g}\to\tilde{g}g}$, in which the emitted particle j is a gluon, all behave as

$$\lim_{z\to 1} P_{\widetilde{ij}\to ij} = \frac{2C_{\widetilde{ij}}}{1-z}\left(1 - \frac{m_i^2}{\tilde{q}^2}\right), \tag{28}$$

in the soft $z \to 1$ limit, where $C_{\widetilde{ij}}$ equals C_F for $P_{q\to qg}$ and $P_{\tilde{q}\to\tilde{q}g}$, $\frac{1}{2}C_A$[12] for $P_{g\to gg}$, and C_A for $P_{\tilde{g}\to\tilde{g}g}$. In using these splitting functions to simulate the emission of a gluon from a time-like mother parton $\widetilde{ij}$, associated to a general n parton configuration with matrix element $\mathcal{M}_n$, one is effectively approximating the matrix element for the process with the additional gluon, $\mathcal{M}_{n+1}$, by

$$|\mathcal{M}_{n+1}|^2 = \frac{8\pi\alpha_S}{q_{\widetilde{ij}}^2 - m_{\widetilde{ij}}^2} P_{\widetilde{ij}\to ij}\,|\mathcal{M}_n|^2. \tag{29}$$

Using the definitions of our shower variables, (17), and making the soft emission approximations $q_{\widetilde{ij}} \approx q_i \approx p$, $q_i^2 \approx m_i^2 = m_{\widetilde{ij}}^2$ in (28), (29) we find [20]

$$\lim_{z\to 1} \frac{8\pi\alpha_S}{q_{\widetilde{ij}}^2 - m_{\widetilde{ij}}^2} P_{\widetilde{ij}\to ij} = -4\pi\alpha_S C_{\widetilde{ij}}\left(\frac{n}{n\cdot q_j} - \frac{p}{p\cdot q_j}\right)^2. \tag{30}$$

Recalling that we choose our Sudakov basis vector n to point in the direction of the colour partner of the gluon emitter $(\widetilde{ij}/i)$, (30) is then just the usual soft eikonal dipole function describing soft gluon radiation by a colour dipole [48], at least for the majority of cases where the colour partner is massless or nearly massless. In practice, the majority of processes we intend to simulate involve massless or light partons, or partons that are light enough that n reproduces the colour partner momentum to high accuracy.[13]

For the case that the underlying process with matrix element $\mathcal{M}_n$ is comprised of a single colour dipole (as is the case for a number of important processes), the parton shower approximation to the matrix element $\mathcal{M}_{n+1}$, (29), then becomes exact in the soft limit as well as, and independently of, the collinear limit. This leads to a better description of soft wide angle radiation, at least for the first emission, which is of course the widest angle emission in the angular ordered parton shower. Should the underlying hard process consist of a quark anti-quark pair, this exponentiation of the full eikonal current, (30), hidden in the splitting functions, combined with a careful treatment of the running coupling (Sect. 6.7), will resum all leading and next-to-leading logarithmic corrections [49–52]. In the event that there is more than one colour dipole in the underlying process, the situation is more complicated due to the ambiguity in choosing the colour partner of the gluon, and the presence of non-planar colour topologies.

In general, the emission probability for the radiation of gluons is infinite in the soft $z \to 1$ and collinear $\tilde{q} \to 0$ limits. Physically these divergences would be canceled by virtual corrections, which we do not explicitly calculate but rather include through unitarity. We impose a physical cutoff on the gluon and light quark virtualities and call radiation above this limit resolvable. The cutoff ensures that the contribution from resolvable radiation is finite. Equally the uncalculated virtual corrections ensure that the contribution of the virtual and unresolvable emission below the cutoff is also finite. Imposing unitarity,

$$\mathcal{P}(\text{resolved}) + \mathcal{P}(\text{unresolved}) = 1, \tag{31}$$

gives the probability of no branching in an infinitesimal increment of the evolution variable $\mathrm{d}\tilde{q}$ as

$$1 - \sum_{i,j} \mathrm{d}\mathcal{P}_{\widetilde{ij}\to ij}, \tag{32}$$

where the sum runs over all possible branchings of the particle $\widetilde{ij}$. The probability that a parton does not branch between

[11] The $P_{g\to gg}$ splitting presented here is for final-state branching where the outgoing gluons are not identified and therefore it lacks a factor of two due to the identical particle symmetry factor. For initial-state branching one of the gluons is identified as being space-like and one as time-like and therefore an additional factor of 2 is required.

[12] Note that for $g \to gg$, there is also a soft singularity at $z \to 0$ with the same strength, so that the *total* emission strength for soft gluons from particles of all types in a given representation is the same: C_F in the fundamental representation and C_A in the adjoint.

[13] Even when the colour partner has a large mass, as in $e^+e^- \to t\bar{t}$, the fact that each shower evolves into the forward hemisphere, in the opposite direction to the colour partner, means that the difference between (30) and the exact dipole function is rather small in practice.

two scales is given by the product of the probabilities that it did not branch in any of the small increments $\mathrm{d}\tilde{q}$ between the two scales. Hence, in the limit $\mathrm{d}\tilde{q} \to 0$ the probability of no branching exponentiates, giving the *Sudakov form factor*

$$\Delta(\tilde{q}, \tilde{q}_h) = \prod_{i,j} \Delta_{\widetilde{ij}\to ij}(\tilde{q}, \tilde{q}_h) \tag{33}$$

which is the probability of evolving between the scale $\tilde{q}_h$ and $\tilde{q}$ without resolvable emission. The no-emission probability for a given type of radiation is

$$\Delta_{\widetilde{ij}\to ij}(\tilde{q}, \tilde{q}_h) = \exp\left\{-\int_{\tilde{q}}^{\tilde{q}_h} \frac{\mathrm{d}\tilde{q}'^2}{\tilde{q}'^2} \int \mathrm{d}z \frac{\alpha_S(z, \tilde{q}')}{2\pi} \times P_{\widetilde{ij}\to ij}(z, \tilde{q}')\Theta(\mathbf{p}_\perp^2 > 0)\right\}. \tag{34}$$

The allowed phase space for each branching is obtained by requiring that the relative transverse momentum is real, or $\mathbf{p}_\perp^2 > 0$. For a general time-like branching $\widetilde{ij} \to i + j$ this gives

$$z^2(1-z)^2\tilde{q}^2 - (1-z)m_i^2 - zm_j^2 + z(1-z)m_{\widetilde{ij}}^2 > 0, \tag{35}$$

from (19).

In practice rather than using the physical masses for the light quarks and gluon we impose a cutoff to ensure that the emission probability is finite. We use a cutoff, Q_g, for the gluon mass, and we take the masses of the other partons to be $\mu = \max(m, Q_g)$, *i.e.* Q_g is the lowest mass allowed for any particle.

There are two important special cases.

1. $q \to qg$, the radiation of a gluon from a quark, or indeed any massive particle. In this case (35) simplifies to

$$z^2(1-z)^2\tilde{q}^2 > (1-z)^2\mu^2 + zQ_g^2, \tag{36}$$

which gives a complicated boundary in the $(\tilde{q}, z)$ plane. However as

$$(1-z)^2\mu^2 + zQ_g^2 > (1-z)^2\mu^2, z^2Q_g^2 \tag{37}$$

the phase space lies inside the region

$$\frac{\mu}{\tilde{q}} < z < 1 - \frac{Q_g}{\tilde{q}} \tag{38}$$

and approaches these limits for large values of $\tilde{q}$. In this case the relative transverse momentum of the branching can be determined from the evolution scale as

$$\mathbf{p}_\perp = \sqrt{(1-z)^2(z^2\tilde{q}^2 - \mu^2) - zQ_g^2}. \tag{39}$$

2. $g \to gg$ and $g \to q\bar{q}$, or the branching of a gluon into any pair of particles with the same mass. In this case the limits on z are

$$z_- < z < z_+, \quad z_\pm = \frac{1}{2}\left(1 \pm \sqrt{1 - \frac{4\mu}{\tilde{q}}}\right) \text{ and } \tilde{q} > 4\mu. \tag{40}$$

Therefore analogously to (38) the phase space lies within the range

$$\frac{\mu}{\tilde{q}} < z < 1 - \frac{\mu}{\tilde{q}}. \tag{41}$$

In this case the relative transverse momentum of the branching can be determined from the evolution scale as

$$\mathbf{p}_\perp = \sqrt{z^2(1-z)^2\tilde{q}^2 - \mu^2}. \tag{42}$$

These two special cases are sufficient for all the branchings currently included in the simulation, although the general case of three unequal masses for the particles in the branching is supported.

The cutoff parameter, Q_g, is the minimum virtuality of the gluon. However, if we consider the phase space that is available to the parton shower we would expect a natural threshold of order $m + Q_g$ for gluon emission from a quark of mass m. In practice for the radiation of a gluon from a quark, (39) gives a threshold that behaves as $Q_{\mathrm{thr}} \simeq 1.15(m_q + 2Q_g)$. This means that the phase-space limit is well above our expectation, particularly for heavy quarks.

There is no reason why Q_g should be the same for all quark flavours. Therefore, we have chosen to parameterize the threshold for different flavours as

$$Q_g = \max\left(\frac{\delta - am_q}{b}, c\right), \tag{43}$$

where a [**aParameter=0.3**] and b [**bParameter=2.3**] are parameters chosen to give a threshold $Q_{\mathrm{thr}} = \beta m_q + \delta$, with $\beta = 0.85$, in order to slightly reduce the threshold distance for heavier quarks. As a result, the threshold for radiation from heavy quarks is closer to its physical limit. The parameter δ is tuned to data as [**cutoffKinScale=2.8 GeV**] and, only relevant for partons heavier than the bottom quark, the parameter c is chosen to prevent the cutoff becoming too small, [**cParameter=0.3 GeV**].

The formalism discussed above allows us, if given a starting scale $\tilde{q}_h$, to evolve a parton down in scale and generate the next branching of this particle at a lower scale. The no-emission probability encoded in the Sudakov form factor is used to generate $(\tilde{q}, z)$ for this branching. This procedure can then be iterated to generate subsequent branchings of the particles produced until no further emission occurs above the cutoff.

6.3 Initial conditions

Before we can simulate possible radiation from a hard process we need to know the initial conditions, *i.e.* the scale $\tilde{q}_h$ from which to start the evolution. The initial conditions for the parton shower are determined by the colour flow in the hard process [3]. For each particle involved in the hard process a colour partner is chosen. In the case of particles in the fundamental representation of the SU(3) gauge group this choice is unique, at least in processes where baryon number is conserved. In the case of a gluon a uniform random choice is made between the two possible partners. In processes involving baryon number violation a uniform random choice is made between all the potential colour partners [53, 54]. The direction of this colour partner determines the maximum angle for emission of QCD radiation from a particle in the angular-ordered parton shower.

Following the choice of the colour partner the maximum scale for radiation from the particle must be calculated, as must the choice of the p and n reference vectors defined in (14). We always take the choice of p along the direction of the radiating particle but the choice of n is related to the direction of the colour partner.

6.3.1 Final-final colour connection

The easiest case to consider is the colour connection between two final-state particles, b and c. Working in their centre-of-mass frame, we may write their momenta as

$$p_b = \frac{1}{2}Q(\mathbf{0}, \lambda; 1+b-c), \qquad p_c = \frac{1}{2}Q(\mathbf{0}, -\lambda; 1-b+c), \tag{44}$$

where $Q^2 = (p_b + p_c)^2$, $b = m_b^2/Q^2$, $c = m_c^2/Q^2$ and

$$\lambda = \lambda(1, b, c) = \sqrt{1 + b^2 + c^2 - 2b - 2c - 2bc} \tag{45}$$

is the Callan function.

In order that the soft region of phase space is fully covered, the initial evolution scales for b and c ($\tilde{q}_{hb}$, $\tilde{q}_{hc}$) are related by

$$(\tilde{\kappa}_b - b)(\tilde{\kappa}_c - c) = \frac{1}{4}(1 - b - c + \lambda)^2, \tag{46}$$

where $\tilde{\kappa}_b = \tilde{q}_{hb}^2/Q^2$, $\tilde{\kappa}_c = \tilde{q}_{hc}^2/Q^2$ [17]. By varying the starting scales of the individual particles we can control how much radiation is generated from each of them, in order to assess the uncertainties. In practice we currently allow four choices controlled by the **FinalFinalConditions** switch:

Symmetric The most symmetric choice of the initial conditions, giving equal amounts of radiation from both partons is given by

$$\tilde{\kappa}_b = \frac{1}{2}(1 + b - c + \lambda), \qquad \tilde{\kappa}_c = \frac{1}{2}(1 - b + c + \lambda). \tag{47}$$

This is our default choice [**FinalFinalConditions=Symmetric**].

Coloured The largest emission scale that is possible for radiation from one of the particles is given by

$$\tilde{\kappa}_b = 4(1 - 2\sqrt{b} - b + c). \tag{48}$$

The [**FinalFinalConditions=Coloured**] choice of initial conditions maximizes the initial evolution scale for the shower of the coloured particle. Naturally, this therefore minimizes the phase space volume available for the first emission from the anti-coloured parton.

AntiColoured This choice of initial conditions, [**FinalFinalConditions=AntiColoured**] is the converse of the [**FinalFinalConditions=Coloured**] choice.

Random Selecting the option [**FinalFinalConditions=Random**], the program randomly sets the initial evolution scales according to the **Coloured** or **AntiColoured** options, for each final-state pair of colour partners, for each event.

As stated in Sect. 6.1 the p basis vector (see (14)) is given by the momentum of the progenitor as it was generated in the initial simulation of the hard process. The light-like basis vector n is chosen to be collinear with the colour partner in the rest frame of the coloured connected pair, *i.e.* in simulating radiation from b, n is defined to be

$$n = \frac{1}{2}Q(\mathbf{0}, -\lambda; \lambda). \tag{49}$$

To simulate parton showering from c, we simply reverse the spatial components of n in (49).

6.3.2 Initial-initial colour connection

Here again we opt to work in the rest frame of the colour partners, so that the momenta of the particles are

$$p_b = \frac{1}{2}Q(\mathbf{0}, 1; 1), \qquad p_c = \frac{1}{2}Q(\mathbf{0}, -1; 1), \tag{50}$$

where Q is the partonic centre-of-mass energy of the collision.

In this case the requirement that the soft region of phase space is smoothly covered is simply

$$\tilde{\kappa}_b \tilde{\kappa}_c = 1. \tag{51}$$

Contrary to the case of the final-final colour connection, there is no upper bound on the values of $\tilde{\kappa}_b$ or $\tilde{\kappa}_c$, *i.e.* there is no choice that maximizes the phase space available to one parton relative to the other (at least none that might reasonably be expected to give sensible results). Currently only the most symmetric choice is implemented, *i.e.* $\tilde{\kappa}_b = \tilde{\kappa}_c = 1$.

In this case, as we assume that the incoming particles are massless, we can simply take the p reference vector to be the momentum of the beam particle from which the emitting parton was extracted and the n reference vector to be the momentum of the beam particle from which its colour partner was extracted. The fact that p is parallel to the momentum of the emitting parton makes it easier to reconstruct the momenta of the shower particles in terms of the fraction of the beam momentum they carry.

Finally, defining the p and n vectors as being equal to the beam momenta rather than the actual parton momenta does not affect our earlier assertions relating to the soft limit of the splitting functions, since (30) is clearly invariant under overall rescalings of the dipole momenta n and p.

6.3.3 Initial-final colour connection in the hard process

Consider the initial-final-state colour connection in the context of a process $a + b \to c$, where a is a colour-singlet system and b and c are colour connected, *e.g.* deep inelastic scattering. As in the last two cases we work in the rest frame of the colour dipole, in this case the Breit frame, where we may write

$$p_b = \frac{1}{2} Q(\mathbf{0}, 1 + c; 1 + c), \qquad p_c = \frac{1}{2} Q(\mathbf{0}, -1 + c; 1 + c), \tag{52}$$

with $Q^2 = -p_a^2$.

To achieve a smooth matching of the phase space for the first emission from parton b's shower with that of parton c's shower, at wide angles, requires the initial evolution scales $(\tilde{q}_{hb}, \tilde{q}_{hc})$ to obey

$$\tilde{\kappa}_b(\tilde{\kappa}_c - c) = (1 + c)^2. \tag{53}$$

In practice, we opt to assign more-or-less the same phase space volume to each shower, *i.e.* we use the most symmetric choice: $\tilde{\kappa}_b = 1 + c$, $\tilde{\kappa}_c = 1 + 2c$. Of course, a larger or smaller combination that satisfies (53) is also allowed.

For emission from the final-state particle, the p vector is taken to be the momentum of the radiating particle and the n reference vector is set equal to the momentum of the beam particle from which the initial-state colour partner was extracted. For emission from the initial-state particle the p vector is defined to be the momentum of the beam particle from which the radiating parton was extracted and

$$n = \frac{1}{2} Q(\mathbf{0}, -1 - c; 1 + c), \tag{54}$$

in the Breit frame. As discussed at the end of the description of the initial-initial colour connection, the normalization of n and/or p, does not affect the eikonal dipole limit of the splitting functions (30).

6.3.4 Initial-final colour connection in decays

The Herwig++ shower differs from other approaches in including initial-state radiation from a decaying coloured particle, as well as final-state radiation from the coloured decay products. This is required in order to ensure that the full soft region of phase space is filled by radiation from the parton shower [17, 20].

Consider the decay $b \to ac$, where b and c are colour partners and a is a colour singlet system, in the rest frame of the decaying particle. In this frame the momentum of b and its colour partner c are

$$p_b = m_b(\mathbf{0}, 0; 1), \qquad p_c = \frac{1}{2} m_b(\mathbf{0}, \lambda; 1 - a + c), \tag{55}$$

where $c = m_c^2/m_b^2$ and hence $\lambda = \lambda(1, a, c)$ where $a = m_a^2/m_b^2$.

In this case the requirement that the full soft region of phase space is filled by radiation from the parton shower gives

$$(\tilde{\kappa}_b - 1)(\tilde{\kappa}_c - c) = \frac{1}{4}(1 - a + c + \lambda)^2. \tag{56}$$

While there is no limit on the value of $\tilde{\kappa}_b$ as with the final-final colour connection the maximum value of $\tilde{\kappa}_c$ is

$$\tilde{\kappa}_c = 4(1 + a - 2\sqrt{c} - c). \tag{57}$$

We support three choices for the values of the scales controlled by the switch **InitialFinalDecayConditions**.

Symmetric The most symmetric choice of initial conditions is

$$\tilde{\kappa}_b = \frac{1}{2}(3 - a + c + \lambda), \qquad \tilde{\kappa}_c = \frac{1}{2}(1 - a + 3c + \lambda), \tag{58}$$

which is the default choice [**InitialFinalDecayConditions**=**Symmetric**].

Maximal The maximal choice corresponds to generating the maximal amount of radiation from the final-state particle, *i.e.* κ_c is given by (57). This corresponds to [**InitialFinalDecayConditions**=**Maximal**].

Smooth In this case the initial conditions are chosen in order to guarantee that, in addition to covering the full soft region, the radiation pattern smoothly changes between the region filled by radiation from b and c. In this case

$$\tilde{\kappa}_b = \frac{2\lambda}{\lambda - (1-\sqrt{c})^2 + a}, \tag{59}$$

with $\tilde{\kappa}_c$ obtained from (56). This option is obtained by setting [**InitialFinalDecayConditions=Smooth**]. In, for example, top decays, this choice leads to more radiation from the decaying particle and less from its colour partner than either of the other options.[14]

For radiation from the decaying particle, p is chosen to be the momentum of the decaying particle and

$$n = \frac{1}{2} m_b(\mathbf{0}, 1; 1), \tag{60}$$

in its rest frame, *i.e.* n is aligned with the colour partner.

In the case of radiation from the final-state particle, p is set equal to its momentum, as generated in the hard decay process, however, there is no obvious choice of n related to the colour partner, since we are working in its rest frame. We therefore choose n such that it is in the opposite direction to the radiating particle in this frame, *i.e.*

$$n = \frac{1}{2}(\mathbf{0}, -\lambda; \lambda). \tag{61}$$

A more rigorous approach to this problem was carried out in [20], using a more generalized splitting function, derived assuming a massive gauge vector n. This feature is not implemented in the standard released code, since any related deficiency in the shower is completely avoided by using the associated matrix element correction (Sect. 6.8).

6.4 Final-state radiation

6.4.1 Evolution

The parton shower algorithm generates the radiation from each progenitor independently, *modulo* the prior determination of the initial evolution scale and the n and p basis vectors. Consider then, the evolution of a given final-state progenitor, downward from its initial evolution scale $\tilde{q}_h$. Given that $\Delta(\tilde{q}, \tilde{q}_h)$ gives the *probability* that this parton evolves from scale $\tilde{q}_h$ to $\tilde{q}$ without any resolvable branchings, we may generate the scale of this first branching ($\tilde{q}$) by solving

$$\Delta(\tilde{q}, \tilde{q}_h) = \mathcal{R}, \tag{62}$$

where $\mathcal{R}$ is a random number uniformly distributed between 0 and 1.

In the FORTRAN HERWIG program this equation was solved by a brute force numerical calculation, using an interpolation table for $\Delta(\tilde{q}, \tilde{q}_h)$. In Herwig++ an alternative approach is used, which determines the scale of the branchings without the need for any explicit integration of the Sudakov form factor [55]. The method involves generating each branching according to a crude Sudakov form factor, based on an *overestimated* branching probability (see (26)), simple enough that (62) can be solved analytically. Each of these crudely determined branchings is subject to a vetoing procedure based on a series of weights relating to the true form factor. In this way the overestimated, crude emission rate and emission distribution is reduced to the exact distribution.

The first ingredient we need in order to implement the algorithm is therefore a crude approximation to the Sudakov form factor (see (35)), for each type of branching of a parent parton $\widetilde{ij}$, $\widetilde{ij} \to i + j$. We write these as

$$\Delta^{\text{over}}_{\widetilde{ij}\to ij}(\tilde{q}, \tilde{q}_h) = \exp\left\{ -\int_{\tilde{q}}^{\tilde{q}_h} \mathrm{d}\mathcal{P}^{\text{over,res.}}_{\widetilde{ij}\to ij} \right\}, \tag{63}$$

where

$$\mathrm{d}\mathcal{P}^{\text{over,res.}}_{\widetilde{ij}\to ij} = \frac{\mathrm{d}\tilde{q}^2}{\tilde{q}^2} \int_{z_-^{\text{over}}}^{z_+^{\text{over}}} \mathrm{d}z\, \frac{\alpha_S^{\text{over}}}{2\pi}\, P^{\text{over}}_{\widetilde{ij}\to ij}(z), \tag{64}$$

is the overestimated probability that a resolvable branching occurs in the interval $[\tilde{q}^2, \tilde{q}^2 + \mathrm{d}\tilde{q}^2]$. Overestimates of the splitting functions and the coupling constant are denoted $P^{\text{over}}_{\widetilde{ij}\to ij}(z) \geq P_{\widetilde{ij}\to ij}(z, \tilde{q})$ and $\alpha_S^{\text{over}} \geq \alpha_S(z, \tilde{q})$, while the limits $z_\pm^{\text{over}}$ also denote overestimates of the true z integration region[15] for all $\tilde{q}$. To solve (62) analytically we also require that $P^{\text{over}}_{\widetilde{ij}\to ij}(z)$ should be analytically integrable and, in order to generate z values, this integral should be an invertible function of z.

Using this simplified Sudakov form factor we may analytically solve $\Delta^{\text{over}}_{\widetilde{ij}\to ij}(\tilde{q}, \tilde{q}_h) = \mathcal{R}$ for $\tilde{q}$ as

$$\tilde{q}^2 = \tilde{q}_h^2\, \mathcal{R}^{\frac{1}{r}}, \tag{65}$$

where

$$r = \frac{\mathrm{d}\mathcal{P}^{\text{over,res.}}_{\widetilde{ij}\to ij}}{\mathrm{d}\ln\tilde{q}^2}, \tag{66}$$

[14] In the extreme limit $c \to 0$, *e.g.* if in top decays the bottom quark is considered massless relative to the top, $\tilde{\kappa}_b \to \infty$ and $\tilde{\kappa}_c \to 0$, meaning that emission only comes from the decaying top quark and none at all from the massless bottom quark. This is because in the limit of a massless bottom quark radiation from the top quark gives the correct dipole distribution in the soft limit.

[15] The overestimates of these limits were given in (38), (41).

which can be thought of as the number of emissions per unit of the shower formation 'time' $\ln\tilde{q}^2$ (for the crude distribution this is a constant). It is clear from (65) how increasing this rate r causes the first branching to be generated 'sooner', closer to $\tilde{q}_h$. When a value is generated for the evolution scale of a branching, an associated z value is then generated according to

$$z = I^{-1}[I(z_-^{\text{over}}) + \mathcal{R}'(I(z_+^{\text{over}}) - I(z_-^{\text{over}}))], \tag{67}$$

where $I(z)$ is the primitive integral of $P^{\text{over}}_{\tilde{i}j\to ij}(z)$ over z, I^{-1} is the inverse of I and $\mathcal{R}'$ is a uniformly distributed random number in the interval [0, 1].

We now reject these values of $\tilde{q}$ and z if:

- the value of z lies outside the true phase-space limits, *i.e.* if $\mathbf{p}_\perp^2 < 0$;
- $\frac{\alpha_S(z,\tilde{q})}{\alpha_S^{\text{over}}} < \mathcal{R}_1$;
- $\frac{P_{\tilde{i}j\to ij}(z,\tilde{q})}{P^{\text{over}}_{\tilde{i}j\to ij}(z)} < \mathcal{R}_2$,

where $\mathcal{R}_{1,2}$ are random numbers uniformly distributed between 0 and 1.

If we reject the value of $\tilde{q}$ we repeat the whole procedure with $\tilde{q}_h = \tilde{q}$ until either we accept a value of $\tilde{q}$, or the value drops below the minimum value allowed due to the phase-space cutoffs, in which case there is no radiation from the particle. As shown in [55] this procedure, called the veto algorithm, exponentiates the rejection factors and generates the values of $\tilde{q}$ and z according to (62) for one type of branching.

This procedure is repeated to give a value of the evolution scale for each possible type of branching, and the branching with the largest value of $\tilde{q}$ is selected, which then generates both the type of branching, its scale, and the momentum fraction according to (62), as required.

The relative transverse momentum for the branching $p_\perp$ (see (18)) is then calculated, using (39) or (42) depending on the type of branching. Currently the azimuthal angle of $p_\perp$ is randomly generated between 0 and 2π about the direction of the progenitor (the Sudakov basis vector p), although in future this will change when we include spin correlations in the parton shower as described in [24–27].

The requirement of angular ordering, as discussed in Sect. 6.1, defines the initial scales for the daughter particles, $\tilde{q}_{hi}$ and $\tilde{q}_{hj}$, produced in each branching, $\tilde{ij} \to i + j$, to be

$$\tilde{q}_{hi} = z\tilde{q}, \qquad \tilde{q}_{hj} = (1-z)\tilde{q}, \tag{68}$$

where $\tilde{q}$ and z, are the evolution scale and light-cone momentum fraction of the branching. By imposing these upper bounds on the evolution scale of the emitted partons, subsequent branchings will have a nesting of the angular separation of the resulting daughters, where each one is smaller than the one preceding it.

All of the steps above, required to generate the shower variables associated with this initial branching, may then be repeated for the daughter partons, and their daughter partons, should they also branch. All showering terminates when the evolution scale ($\tilde{q}$) for each final-state parton falls below its minimum value, when there is no phase space for any more resolvable emissions. The resulting partons, at the end of each shower, are deemed to be on constituent mass-shell, as defined in Sect. 7, at which point the perturbative parton shower evolution is no longer sensible, since hadronization effects dominate at these scales.

6.4.2 *Kinematic reconstruction*

At this point we have a set of partons produced in the parton shower from each of the progenitor partons, the scales $\tilde{q}$ at which they are produced, the momentum fractions z and azimuthal angles ϕ of the branchings. Mapping these kinematic variables into physical momenta is what we call *kinematic reconstruction*. We will now describe this procedure for showers generated by final-state progenitors. First, the kinematics of the individual showers are reconstructed by putting the external masses on their constituent mass-shell[16] and working back through the shower, as described in Sect. 6.1.

The shower evolution causes all progenitor partons, J, produced in the hard process to gain a virtual mass, *i.e.* the progenitor partons, which initiated the jets, are no longer on mass shell, $q_J^2 \neq m_J^2$. We want to preserve the total energy of the system in the centre-of-mass frame of the hard collision. If the momenta of the progenitor partons before the shower evolution were $p_J = (\mathbf{p}_J; \sqrt{\mathbf{p}_J^2 + m_J^2})$ in this frame, then

$$\sum_{J=1}^{n} \sqrt{\mathbf{p}_J^2 + m_J^2} = \sqrt{s}, \tag{69}$$

while the sum of the spatial momenta is zero. As the jet parents have momenta $q_J = (\mathbf{q}_J; \sqrt{\mathbf{q}_J^2 + q_J^2})$ after the parton showering, we need to restore momentum conservation in a way that disturbs the internal structure of the jet as little as possible. The easiest way to achieve this is by boosting each jet along its axis so that their momenta are rescaled by a common factor k determined from

$$\sum_{J=1}^{n} \sqrt{k^2\mathbf{p}_J^2 + q_J^2} = \sqrt{s}, \tag{70}$$

[16] The Herwig++ shower allows these masses to be set to zero so that an alternative hadronization model, rather than the cluster model, can be used.

which can be solved analytically for two jets or numerically for higher multiplicities. For every jet a Lorentz boost is applied such that

$$q_J = (\mathbf{q}_J; \sqrt{\mathbf{q}_J^2 + q_J^2}) \xrightarrow{\text{boost}} q'_J = (k\mathbf{p}_J; \sqrt{k^2\mathbf{p}_J^2 + q_J^2}). \quad (71)$$

Applying these boosts to each of the jets, in the centre-of-mass frame of the collision, ensures global energy-momentum conservation. Typically the rescaling parameters k are close to unity, hence the resulting boosts and rotations are small.

6.5 Initial-state radiation

6.5.1 Evolution

As stated in Sect. 6.1, in generating the initial-state radiation we use a backward evolution algorithm, starting with the space-like daughter parton that initiates the hard scattering process, i, and evolving it backward to give its space-like parent, $\widetilde{ij}$, and time-like sister parton j. This evolution algorithm therefore proceeds from the high scale of the hard process to the low scale of the external hadrons. Such a procedure is greatly more efficient than the alternative forward evolution algorithm, which would start from the incoming beam partons and evolve them to the scale of the hard collision. This is because the forward evolution cannot be constrained to end on the x and Q^2 values associated to the hard process, which in turn makes it impossible to perform importance sampling of any significant resonant contributions.

While forward evolution would dynamically generate the parton distribution functions (PDFs), backward evolution uses the measured PDFs to guide the evolution. As with the final-state shower, the initial conditions for the initial-state shower are determined by the colour partners of the incoming particles (Sect. 6.3.2).

The angular-evolution variable $\tilde{q}^2$ for space-like showers was defined in (21). We shall work exclusively with light initial-state partons so we take $m_{\widetilde{ij}} = m_i = 0$, and $m_j = \mu$ if j is a quark and $m_j = Q_g$ if j is a gluon, to regulate the infrared divergent regions, hence (21) simplifies to

$$\tilde{q}^2 = \frac{zm_j^2 - p_\perp^2}{(1-z)^2}, \quad (72)$$

where $p_\perp^2 = -\mathbf{p}_\perp^2$ (see (18), (25)).

From the requirement that $\mathbf{p}_\perp^2 \geq 0$, (72) implies an upper limit on z,

$$z \leq z_+ = 1 + \frac{Q_g^2}{2\tilde{q}^2} - \sqrt{\left(1 + \frac{Q_g^2}{2\tilde{q}^2}\right)^2 - 1}. \quad (73)$$

In addition, if the light-cone momentum fraction of parton i is x, we must have $z \geq x$ to prevent the initial-state branching simulation evolving backward into a parent with $x > 1$.

In this case the Sudakov form factor for backward evolution is [3, 56]

$$\Delta(x, \tilde{q}, \tilde{q}_h) = \prod_{\widetilde{ij},j} \Delta_{\widetilde{ij}\to ij}(x, \tilde{q}, \tilde{q}_h), \quad (74)$$

where the Sudakov form factor for the backward evolution of a given parton i is

$$\Delta_{\widetilde{ij}\to ij}(x, \tilde{q}, \tilde{q}_h) = \exp\left\{ -\int_{\tilde{q}}^{\tilde{q}_h} \frac{\mathrm{d}\tilde{q}'^2}{\tilde{q}'^2} \int_x^{z_+} \mathrm{d}z \, \frac{\alpha_S(z, \tilde{q}')}{2\pi} \, P_{\widetilde{ij}\to ij}(z, \tilde{q}') \times \frac{\frac{x}{z} f_{\widetilde{ij}}(\frac{x}{z}, \tilde{q}')}{x f_i(x, \tilde{q}')} \Theta(\mathbf{p}_\perp^2 > 0) \right\}, \quad (75)$$

and the product runs over all possible branchings $\widetilde{ij} \to i + j$ capable of producing a parton of type i. This is similar to the form factor used for final-state radiation, (35), with the addition of the PDF factor, which guides the backward evolution.

The backward evolution can be performed using the veto algorithm in the same way as the forward evolution. We need to solve

$$\Delta(x, \tilde{q}, \tilde{q}_h) = \mathcal{R}, \quad (76)$$

to give the scale of the branching. We start by considering the backward evolution of i via a particular type of branching, $\widetilde{ij} \to i + j$. We can obtain an overestimate of the integrand in the Sudakov form factor

$$\Delta^{\text{over}}_{\widetilde{ij}\to ij}(x, \tilde{q}, \tilde{q}_h) = \exp\left\{ -\int_{\tilde{q}}^{\tilde{q}_h} \frac{\mathrm{d}\tilde{q}'^2}{\tilde{q}'^2} \int_x^{z_+^{\text{over}}} \mathrm{d}z \, \frac{\alpha_S^{\text{over}}}{2\pi} \times P^{\text{over}}_{\widetilde{ij}\to ij}(z) \text{PDF}^{\text{over}}(z) \right\}, \quad (77)$$

where $P^{\text{over}}_{\widetilde{ij}\to ij}(z)$, α_S^{over} and the overestimate of the limits must have the same properties as for final-state branching. In addition

$$\text{PDF}^{\text{over}}(z) \geq \frac{\frac{x}{z} f_{\widetilde{ij}}(\frac{x}{z}, \tilde{q})}{x f_i(x, \tilde{q})} \quad \forall z, \ \tilde{q}, \ x. \quad (78)$$

In this case the product $P^{\text{over}}_{\widetilde{ij}\to ij}(z)\text{PDF}^{\text{over}}(z)$ must be integrable and the integral invertible. If we define

$$r = \frac{\alpha_S^{\text{over}}}{2\pi} \int_x^{z_+^{\text{over}}} \mathrm{d}z \, P^{\text{over}}_{\widetilde{ij}\to ij}(z)\text{PDF}^{\text{over}}(z), \quad (79)$$

then we can solve (76) using this overestimated Sudakov giving

$$\tilde{q}^2 = \tilde{q}_h^2 \, \mathcal{R}^{\frac{1}{r}}. \quad (80)$$

The value of z can then be generated according to

$$z = I^{-1}[I(x) + \mathcal{R}'(I(z_+^{\text{over}}) - I(x))], \tag{81}$$

where $I(z) = \int \mathrm{d}z\, P^{\text{over}}_{\widetilde{ij}\to ij}(z)\text{PDF}^{\text{over}}(z)$, I^{-1} is the inverse of I and $\mathcal{R}'$ is a random number uniformly distributed between 0 and 1.

We now reject these values of $\tilde{q}$ and z if:

- the value of z lies outside the true phase-space limits, *i.e.* if $\mathbf{p}_\perp^2 < 0$;
- $\frac{\alpha_S(z,\tilde{q})}{\alpha_S^{\text{over}}} < \mathcal{R}_1$;
- $\frac{P_{\widetilde{ij}\to ij}(z,\tilde{q})}{P^{\text{over}}_{\widetilde{ij}\to ij}(z)} < \mathcal{R}_2$;
- $\frac{\frac{x}{z} f_a(\frac{x}{z},\tilde{q}')}{x f_b(x,\tilde{q}')} < \mathcal{R}_3$;

where $\mathcal{R}_{1,2,3}$ are random numbers uniformly distributed between 0 and 1.

As with the final-state branching algorithm, if a set of values of $\tilde{q}$ and z, generated according to the approximate form factor in (78) is rejected, a further set is then generated by repeating the process with $\tilde{q}_h = \tilde{q}$ in (78). This procedure is repeated until either a generated set of branching variables passes all four vetoes, or the generated value of $\tilde{q}$ falls below the minimum allowed value, in which case the showering of the particle in question ceases. To determine the species of partons involved, a trial value of $\tilde{q}$ is generated for each possible type of branching and the largest selected. By applying the four vetoing criteria to each emission generated by the approximate, overestimated, Sudakov form factor, the accepted values of $\tilde{q}$ and z are distributed according to the exact Sudakov form factor, (76) [55].

When a branching is generated, the relative transverse momentum $p_\perp$ (see (18), (25)) is calculated according to (72). At present the azimuthal angle associated to each $p_\perp$ is randomly generated between 0 and 2π, although in future this will change when we include spin correlations in the parton shower as described in [24–27]. In the case of backward evolution the angular ordering requirement is satisfied by simply continuing the backward evolution downward in $\tilde{q}$, starting from the value generated in the previous generated branching.

As stated above, when the evolution scale has reduced to the point where there is no more phase space for further resolvable branchings, the backward evolution ends. The incoming particle produced in the last backward branching, assumed to be on-shell (massless), has no transverse momentum, since this is measured with respect to the beam axis.[17] This final parton also has a light-cone momentum fraction $x/\prod_i z_i$, with respect to the incoming hadron's momentum, where x is the light-cone momentum fraction generated in the initial simulation of the hard process, and the product is comprised of all z values generated in the backward evolution.

Before any momentum reconstruction can begin, we must simulate the effects of final-state showers from each time-like daughter parton j, generated from the backward evolution of each space-like parton i, in branchings $\widetilde{ij} \to i + j$. As discussed in Sect. 6.1, for such a branching occurring at scale $\tilde{q}$ with light-cone momentum fraction z, angular ordering is achieved by evolving j down from an initial scale $\tilde{q}_h = (1-z)\tilde{q}$. This initial condition ensures that for each parton j, the angular separation of any of j's subsequent branching products is less than the angle between j and j's sister i.

This algorithm is all that is needed to generate the values of the scales, momentum fractions and azimuthal angles, for the evolution of both the incoming particles and the time-like particles emitted in their backward evolution. These values are sufficient for us to determine the momenta of all of the particles in the associated showers, to perform the kinematic reconstruction.

6.5.2 *Kinematic reconstruction*

The kinematic reconstruction begins by finding the last initial-state particle produced in the backward evolution of each of the beam particles. This parton's momentum is calculated as described in the previous section. The momentum of the final-state time-like jet that it radiates is then reconstructed in the same way as for the final-state shower. Knowing the momenta of the former light-like parent parton and the latter final-state, time-like, daughter parton, the momentum of the initial-state, space-like, daughter, follows by momentum conservation. This process is iterated for each initial-state branching, eventually giving the momentum of the space-like progenitor parton, colliding in the hard process.

The reconstructed momentum of the colliding parton incident from the $+z$ direction is denoted $q_\oplus$, and that of the colliding parton incident from the $-z$ direction is denoted $q_\ominus$.

Although the showering of initial-state partons with final-state colour partners and initial reconstruction is performed as described above, using the colour partner's direction as the basis vector n, at present the final momentum reshuffling to ensure energy and momentum conservation is performed as if the colour partner were the other incoming parton.

As discussed in Sect. 6.3.2 the hadronic beam momenta, $p_\oplus$ and $p_\ominus$, then define the Sudakov basis for the initial-state shower algorithms, in terms of which we have

$$q_\oplus = \alpha_\oplus\, p_\oplus + \beta_\oplus\, p_\oplus + q_{\perp\oplus}. \tag{82}$$

[17] Herwig++ supports the option of including a non-perturbative intrinsic transverse momentum for the partons inside the incoming hadron, as described in Appendices B.7 and C, which can give the initial incoming parton a transverse momentum.

The Sudakov coefficients may be calculated using the fact that $p_\oplus$ and $p_\ominus$ are light-like and orthogonal to the $q_\perp$ component:

$$\alpha_\oplus = 2p_\oplus \cdot q_\oplus/s, \qquad \beta_\oplus = 2p_\oplus \cdot q_\oplus/s, \tag{83}$$

where $s = 2p_\oplus \cdot p_\ominus$, the hadronic centre-of-mass energy squared. The $q_\perp$ components follow by subtracting $\alpha_\oplus p_\oplus + \beta_\oplus p_\oplus$ from the reconstructed momentum $q_\oplus$.

Through the emission of initial-state radiation the colliding partons acquire both space-like virtualities and transverse momenta, of which they had neither in the initial simulation of the hard process. Consequently, whereas momentum conservation in the prior simulation of the hard process implies that the total initial- and final-state momentum are equal to $p_{\text{cms}} = x_\oplus p_\oplus + x_\ominus p_\ominus$, we now have a momentum imbalance between the two: $q_\oplus + q_\ominus \neq x_\oplus p_\oplus + x_\ominus p_\ominus$.

In order to return to a momentum conserving state we choose to rescale the energies and longitudinal momenta of the colliding initial-state partons, in a way that preserves the invariant mass and rapidity of the centre-of-mass system. The transverse momentum of the emitted radiation can only be absorbed by the final-state system. When the rescaling factors have been determined, we can then calculate a Lorentz boost that produces the same effect. This boost can then be applied to all elements of the initial-state shower, including the final-state jets they emit.

The energies and longitudinal momenta of the colliding partons are rescaled by two factors, $k_\oplus$ and $k_\ominus$, giving *shuffled momenta* $q'_\oplus$ and $q'_\ominus$, according to

$$q'_\oplus = \alpha_\oplus k_\oplus p_\oplus + \frac{\beta_\oplus}{k_\oplus} p_\oplus + q_{\perp\oplus}. \tag{84}$$

In simulating the hard process the momentum of the partonic centre-of-mass system was given by

$$p_{\text{cms}} = x_\oplus p_\oplus + x_\ominus p_\ominus \tag{85}$$

and in terms of the shuffled momenta it is

$$q'_{\text{cms}} = \left(\alpha_\oplus k_\oplus + \frac{\beta_\ominus}{k_\ominus}\right) p_\oplus + \left(\alpha_\ominus k_\ominus + \frac{\beta_\oplus}{k_\oplus}\right) p_\ominus + q_{\perp\oplus} + q_{\perp\ominus}. \tag{86}$$

Imposing that the centre-of-mass energy generated in the simulation of the hard process is preserved, $q'^2_{\text{cms}} = p^2_{\text{cms}}$, the Sudakov decompositions of (85), (86), imply that the rescalings $k_\oplus$ and $k_\ominus$ obey the relation

$$\begin{aligned}&\alpha_\oplus\alpha_\ominus s\, k^2_{\oplus\ominus} + \beta_\oplus\beta_\ominus s \\ &\quad + \big((\alpha_\oplus\beta_\oplus + \alpha_\ominus\beta_\ominus - x_\oplus x_\ominus)s \\ &\quad + (q_{\perp\oplus} + q_{\perp\ominus})^2\big)k_{\oplus\ominus} = 0,\end{aligned} \tag{87}$$

where $k_{\oplus\ominus} = k_\oplus k_\ominus$. The further imposition that the rapidity of the partonic centre-of-mass is preserved requires that the ratio of the $p_\oplus$ coefficient to the $p_\ominus$ Sudakov coefficient in q'_{cms} should equal that in p_{cms}. This implies a second constraint on $k_\oplus$ and $k_\ominus$

$$k^2_\oplus = k_{\oplus\ominus} \frac{x_\oplus}{x_\ominus} \frac{\beta_\oplus + \alpha_\ominus k_{\oplus\ominus}}{\alpha_\oplus k_{\oplus\ominus} + \beta_\ominus}. \tag{88}$$

The two relations in (85), (86) fully determine the $k_\oplus$ and $k_\ominus$ rescaling factors. Having solved these equations for $k_\oplus$ and $k_\ominus$ we go on to determine a longitudinal boost for each initial-state jet such that

$$q_\oplus \xrightarrow{\text{boost}} q'_\oplus. \tag{89}$$

This boost may then be applied to all elements of the initial-state shower including any final-state partons/jets that they emit.

The procedure outlined above is sufficient for the production of colour-singlet systems, such as electroweak gauge bosons in the Drell-Yan process. However, for processes where both the initial- and final-state particles can radiate, a more complicated procedure is needed. In [17] a procedure for the reconstruction of the kinematics based on the colour structure of the hard process was suggested.

In Herwig++ we have opted to use a simpler procedure. In the current approach, first the initial conditions for the shower of both the initial- and final-state particles are chosen (Sect. 6.3). Following this, the evolution of the incoming and outgoing particles is performed as described in Sects. 6.4.1 and 6.5.1. The initial-state jets are then reconstructed as discussed above. The final-state jets are reconstructed in the partonic centre-of-mass frame of the original hard scattering process as described in Sect. 6.4.2. This is effectively the same as reconstructing them in the q'_{cms} rest frame, since the kinematic reconstruction for initial-state radiation, described here, preserves the invariant mass of the hard process. In the end, the jets originating from the final-state particles in the hard process are boosted back to the lab frame, where they have a total momentum q'_{cms}. This procedure is simpler than that suggested in [17]; it represents a general approach to ensuring global energy and momentum conservation in all processes, whereas the methods in [17] are more process-specific, by being sensitive to the details of the underlying colour flow in the hard process.

6.5.3 *Forced splitting*

After the perturbative shower evolution has terminated, the cluster hadronization model may necessitate some additional *forced splitting* of the initial-state parton that results. In hadronic collisions we require the external initial-state partons, which give rise to the first hard interaction, to be valence quarks (antiquarks), colour triplet states. This allows

us to treat each proton (antiproton) remnant as a diquark (antidiquark) which will be in a colour antitriplet/triplet state, in order to keep the incoming hadron colour neutral. Modelling the dissociation in this way allows for a simple, minimal, hadronization of the remnant in the cluster hadronization model.

Usually, the perturbative evolution, which is guided by the PDFs, will terminate on a valence quark, since the evolution works towards large x and small Q^2. In the cases where this has not happened, we force the resulting initial-state parton to undergo one or two additional splittings. The generation of these additional forced splittings is largely based on the same principles as that of the perturbative splittings.

In the perturbative evolution the scale of the PDFs is frozen at a value Q_s for values $Q < Q_s$. The default value of Q_s is chosen to be small, close to the non-perturbative region but still above typical values for the parton shower cutoff [(**PDFFreezingScale=2.5*GeV**]). This freezing scale leaves a little phase space for the (non-perturbative) forced splittings. The forced splittings are generated in much the same vein as the perturbative splittings. The evolution starts at Q_s and the next branching scale is distributed according to $\mathrm{d}Q/Q$, with a lower limit determined by the available phase space. The z values are determined from the splitting functions in the same way as in the perturbative evolution. The splittings are reweighted by ratios of PDFs as in the perturbative evolution. There is only one slight difference, the evolution of the PDFs themselves with Q is frozen below Q_s. Nevertheless, this reweighting gives the right flavour content of the initial hadron. E.g. in the case of a proton we produce twice as many u quarks as d quarks. To force the evolution to end up on a valence quark, we only allow one or two flavours in the evolution:

1. If the initial parton is a seaquark (q) or –antiquark ($\bar{q}$), it is forced to evolve into a gluon, emitting a $\bar{q}$ or q, respectively.
2. If the initial parton is a gluon, from either the perturbative evolution or the forced splitting of a seaquark, it is forced to evolve into a valence quark, emitting the matching antiquark.

In the initial-state showering of additional hard scatters we force the backward evolution of the colliding partons to terminate on a gluon. We therefore only need the first kind of forced splitting in this case. This gluon is assumed to be relatively soft and branches off from the remnant diquark. Again, this allows us to uniquely match up the final-state partons to the cluster hadronization model. We should note that the emitted partons from these forced splittings, as well as the remnant diquarks, will show up in the event record as decay products of a fictious remnant particle, in order to distinguish them from those which originate from the perturbative evolution. Additional details about the colour structure and the event record can be found in [8].

6.6 Radiation in particle decays

In general the hard processes simulated by Herwig++ consist of $2 \to n$ scatterings. These are generated by first using the relevant matrix elements to produce an initial configuration, and then initiating parton showers from the external legs. After this showering phase the final-state consists of a set of partons with constituent masses. For processes involving the production and decay of unstable particles, including decay chains, rather than attempting to calculate high multiplicity matrix elements, the simulation is simplified by appealing to the *narrow width approximation*, *i.e.* treating the production and decay processes according to separate matrix elements, assuming no interference between the two. Unstable coloured particles are therefore produced in hard processes and the decays of other unstable particles, and showered like any other final-state coloured particle. In this case the showering process does not assign a constituent mass to the final state of the shower, but rather preserves whatever mass was assigned at the production stage.

For very high mass coloured particles, *e.g.* the top quark, the phase space available for the decay can be so large that the decay occurs before any hadronization can take place. Consequently, as well as undergoing time-like showers ($q^2 > m^2$) in their production phase, these partons will also undergo a further *space-like* showering ($q^2 < m^2$) of QCD radiation prior to their decay. In addition, due to colour conservation, the decay products themselves will also give rise to time-like showers.

Since, in the narrow width approximation, the matrix element factorizes into one for the production process and another for the decay process, we may regard these as two independent hard processes, and this is the sense in which we simulate the associated parton showers. Given this picture it is immediately clear that the time-like parton showers, from coloured decay products, have an *identical* evolution to those used to simulate final-state radiation in the production process. Only the initial conditions for the shower evolution are different, although their selection is, nevertheless, still based on examining the colour flow in the underlying hard decay process (see Sect. 6.3.4).

Conversely, the initial-state space-like shower created by a decaying particle is quite different to that of an initial-state particle from the production process (Sect. 6.5). In particular, it involves no PDFs, since the heavy parton originates from a hard scattering as opposed to a hadron. Furthermore, in the hard process it was necessary to evolve the initial-state partons backwards from the hard scattering to the incident hadrons, to efficiently sample any resonant structure in the underlying matrix elements. On the contrary, in decay processes, degrading the invariant mass of the decaying particle, via the emission of radiation, does not affect the efficiency with which any resonant structures in the decay

matrix element are sampled. Hence, it is natural for the evolution of space-like decay showers to start with the unstable particle from the production process, and evolve it forward, towards its decay.

6.6.1 Evolution

As in our discussion of the other showering algorithms, the description here uses the Sudakov decomposition of the momenta given in (14). In space-like decay showers, the decaying particle $\widetilde{ij}$ undergoes branchings $\widetilde{ij} \to i + j$, where j is a final-state time-like parton and i is the same decaying particle with an increased space-like virtuality: $q_i^2 < q_{\widetilde{ij}}^2 \leq m_{\widetilde{ij}}^2$. In this process the original particle acquires a space-like virtuality,

$$q_i^2 = z q_{\widetilde{ij}}^2 + \frac{p_\perp^2 - z q_j^2}{1-z}, \tag{90}$$

where $z = \alpha_i/\alpha_{\widetilde{ij}}$, $\mathbf{p}_\perp^2 = -p_\perp^2 \geq 0$, and $p_\perp = q_{\perp i} - z q_{\perp \widetilde{ij}}$. Since, in the decay shower, $m_i = m_{\widetilde{ij}}$, the space-like evolution variable in (21) simplifies to

$$\tilde{q}^2 = m_i^2 + \frac{z m_j^2 - p_\perp^2}{(1-z)^2}. \tag{91}$$

Unlike the previous discussions of final- and initial-state showers, here, by evolving forward toward the decay process, the evolution variable is *increasing*. The requirement that the relative transverse momentum of the branching is real, $\mathbf{p}_\perp^2 \geq 0$, imposes an upper limit, z_+, on z where

$$z_+ = 1 + \frac{m_j^2}{2(\tilde{q}^2 - m_i^2)}\left(1 - \sqrt{1 + 4(\tilde{q}^2 - m_i^2)/m_j^2}\right). \tag{92}$$

For the space-like decay shower we have the further constraint that the parton showering cannot degrade the invariant mass of the decaying object below the threshold required for the decay process, which imposes a lower limit on z.

Since no PDF is involved in this forward parton-shower evolution algorithm, the Sudakov form factor has exactly the same form as that used for final-state radiation in (33), (35). Consequently the forward evolution can be performed using the veto algorithm in almost exactly the same way as was done for the final-state showers (Sect. 6.4.1). The main difference is in the implementation of the angular ordering bounds for subsequent branchings. For final-state radiation involving branchings $\widetilde{ij} \to i + j$, where i has a light-cone momentum fraction z, we evolved i and j *downward* from $\tilde{q}_{hi} = z\tilde{q}$ and $\tilde{q}_{hj} = (1-z)\tilde{q}$ respectively, where $\tilde{q}$ was the scale of the $\widetilde{ij}$ branching. Since the decay shower is really a forward-evolving initial-state shower, we evolve i *upward* from $\tilde{q}_{hi} = \tilde{q}$ and j *downward* from $\tilde{q}_{hj} = (1-z)\tilde{q}$. This procedure is iterated until the scale $\tilde{q}$ approaches the minimum compatible with the threshold for the underlying decay process.

6.6.2 Kinematic reconstruction

In the approach of [17], for the simulation of QCD radiation in particle decays, the recoil due to the radiation emitted from the decaying particle is absorbed by its final-state colour partner. The reconstruction described in [20], valid for the decay of a coloured particle to a colour connected final-state particle and a colour-singlet system, was designed to preserve the mass of the colour-singlet system. In the case of top decay this amounts to preserving the mass of the W boson, and the momentum of the decaying particle. More complicated colour structures, involving more coloured particles in the final-state, *e.g.* gluino decays, require a generalization of this momentum reconstruction procedure.

Consider the decay of a coloured particle with momentum p, to $n+1$ particles. We denote the momentum of the colour partner of the decaying particle $\bar{p}$, and the momenta of the remaining primary decay products are denoted $p_{i=1,n}$. Prior to simulating the effects of QCD radiation,

$$p = \bar{p} + \sum_{i=1}^{n} p_i. \tag{93}$$

After simulating parton-shower radiation in the decay, the original momenta of the decay products must be shifted and rescaled to accommodate the additional *initial-state* radiation. We require the sum of the new momenta of the colour partner, $\bar{q}$, the other primary decay products, q_i, *and* the radiation emitted prior to the decay, q_{ISR}, to equal that of the decaying particle:

$$p = \bar{q} + q_{ISR} + \sum_{i=1}^{n} q_i. \tag{94}$$

To achieve this momentum balance we rescale the three-momenta of all p_i by a common factor k_1, and the three-momentum of the colour partner $\bar{p}$ by a separate factor k_2. The component of the momentum of the emitted radiation transverse to the colour partner is absorbed by the colour partner. In the rest frame of the decaying particle these rescalings and shiftings look as follows:

$$p = (\mathbf{0}; m); \tag{95a}$$

$$q_i = (k_1 \mathbf{p}_i; \sqrt{k_1^2 |\mathbf{p}_i|^2 + p_i^2}); \tag{95b}$$

$$\bar{q} = (k_2 \bar{\mathbf{p}} - \mathbf{q}_{\perp ISR}; \sqrt{k_2^2 |\bar{\mathbf{p}}|^2 + |\mathbf{q}_{\perp ISR}|^2 + \bar{p}^2}), \tag{95c}$$

where m is the mass of the decaying particle and $\mathbf{q}_{\perp ISR}$ is the component of the three-momentum of the initial-state radiation perpendicular to $\bar{p}$.

The rescaling factors $k_{1,2}$ allow for the remaining conservation of energy and of momentum in the longitudinal direction. Three-momentum conservation in the longitudinal, $\bar{\mathbf{p}}$,

direction requires that

$$k_2\bar{\mathbf{p}} + k_1 \sum_{i=1}^{n} \mathbf{p}_i + \mathbf{q}_{\parallel ISR} = 0. \tag{96}$$

The momentum of the initial-state radiation perpendicular to the direction of the colour partner, $\mathbf{q}_{\perp ISR}$, can be projected out, leaving the parallel component $\mathbf{q}_{\parallel ISR}$, by taking the dot product with the spatial component of the n basis vector (aligned with $\bar{p}$). Doing so gives

$$k_1 = k_2 + \frac{\mathbf{q}_{ISR} \cdot \mathbf{n}}{\bar{\mathbf{p}} \cdot \mathbf{n}}. \tag{97}$$

Finally, from the conservation of energy we have

$$\sum_{i=1}^{n} \sqrt{k_1^2 |\mathbf{p}_i|^2 + p_i^2} + \sqrt{k_2^2 |\bar{\mathbf{p}}|^2 + |\mathbf{q}_{\perp ISR}|^2 + \bar{p}^2} + E_{ISR} = m, \tag{98}$$

where E_{ISR} is the energy of the initial-state radiation. This system of equations (96), (97), (98) for the rescaling factors can be solved analytically for two-body decays, or numerically, using the Newton-Raphson method, for higher multiplicities.

6.7 The running coupling constant α_S

The running coupling constant enters every dynamical aspect of the parton shower, so a thorough treatment of it is mandatory for all parton shower simulations.

6.7.1 The argument of α_S

As was noted in Sect. 6.2, our definition of the momentum fraction z is consistent with that used in the derivation of the quasi-collinear splitting functions, hence n does not just define a basis vector in the Sudakov decomposition but it also specifies the choice of light-cone (axial) gauge.

Axial gauges have many special properties, most notable of these is that they are ghost-free. Another, related, interesting feature of the light-cone gauge is that, similar to QED, where the Ward identities guarantee the equality of the electron field and vertex renormalization constants, in light-cone gauge QCD, the Ward identities reveal that the 3-gluon vertex renormalization constant Z_{A^3}, is equal to that of the transverse components gluon field $Z_A^{1/2}$ [57]. This simplifies the usual relation between the bare coupling $g_S^{(0)}$ and renormalized coupling constant g_S from $g_S^{(0)} = Z_{A^3} Z_A^{-3/2} g_S$, to $g_S^{(0)} = Z_A^{-1/2} g_S$, *i.e.* in the light-cone gauge, the running of the QCD coupling constant is due to the gluon self-energy corrections alone. It is therefore no surprise that explicit, dimensionally regulated, one-loop calculations of the gluon self-energy in this gauge possess an ultraviolet divergence proportional to the usual QCD beta function [57, 58].

In calculating higher order corrections to the splitting functions one must consider self-energy corrections to the emitted gluons and their associated counter-terms. The self-energy corrections are equal to zero because the gluons are on-shell and so the associated loop integrals have no scale, which means they vanish in dimensional regularization. This vanishing is essentially a complete cancellation of the ultraviolet and infrared parts of the integrals. Therefore including the counter-terms cancels explicitly the ultraviolet divergent parts of the loop integrals leaving behind infrared divergent parts, which must have the same pole structure as the ultraviolet parts *i.e.* they must also be proportional to the beta function. The residual virtual infrared divergence is canceled by the associated real emission corrections, in this case the two graphs where the emitted gluon splits either to two on-shell gluons or to a quark-antiquark pair.

As usual, this cancellation of infrared poles generates an associated logarithm, with the same coefficient as the pole (the beta function), of the phase space boundary divided by μ (the renormalization scale) [39, 59]. The phase space boundary is equal to the maximum possible virtuality of the daughter gluon, the branchings of which comprise the real emission corrections. For a time-like splitting, $\widetilde{ij} \to i + j$ where $\widetilde{ij}$ is a quark, i is a daughter quark and j is the daughter gluon, to which we consider real and virtual corrections, a quick calculation in the Sudakov basis (14) shows

$$q_j^2 \leq (1-z) q_{\widetilde{ij}}^2. \tag{99}$$

The net effect of these real and virtual corrections is therefore to simply correct the leading order $q \to qg$ splitting function by a multiplicative factor

$$1 - \beta_0 \alpha_S(\mu^2) \ln((1-z) q_{\widetilde{ij}}^2/\mu^2) + \mathcal{O}(\alpha_S), \tag{100}$$

where the omitted $\mathcal{O}(\alpha_S)$ terms are non-logarithmic, non-kinematic, constant terms, β_0 is the QCD beta function, and μ^2 is the renormalization scale.

Two important points follow directly from this analysis. Firstly, for soft configurations, $z \to 1$, the effect of these loop contributions can produce large, numerically significant, logarithms. Secondly, plainly, by choosing the renormalization scale to be $(1-z) q_{\widetilde{ij}}^2$, instead of the more obvious $q_{\widetilde{ij}}^2$, the corrections vanish, or rather, more correctly, they are absorbed in the coupling constant.

For $g \to gg$ splittings the same arguments hold but in this case it is apparent that as well as large logarithms of $1-z$, large logarithms of z are also possible from soft emission in the $z \to 0$ region. We may simultaneously include both types of correction by using $z(1-z) q_{\widetilde{ij}}^2$ as the argument of the running coupling, which we implement in practice as

$$\alpha_S(z^2 (1-z)^2 \tilde{q}^2). \tag{101}$$

From the point of view of the leading-log approximation, the choice of scale is technically a higher order consideration, nevertheless, these effects turn out to be highly phenomenologically significant, particularly their effect on multiplicity distributions and cluster mass spectra [59, 60].

6.7.2 The Monte Carlo scheme for α_S

We reiterate that by choosing the scale of the running coupling as advocated in Sect. 6.7.1 (see (99), (101)) we have

$$\lim_{z\to 1} \alpha_S((1-z)q_{\tilde{ij}}^2) P_{q\to qg}^{[1]}(z) = \alpha_S \frac{2C_F}{1-z}(1-\alpha_S\beta_0 \ln(1-z)) + \mathcal{O}(\alpha_S^3), \quad (102)$$

where we have momentarily abbreviated $\alpha_S(q_{\tilde{ij}}^2)$ by α_S, and used a superscript [1] to denote that $P_{q\to qg}^{[1]}$ is the *one-loop* (*i.e.* leading order) $q \to qg$ splitting function. This is almost, but not exactly equal to the soft $z \to 1$ singular limit of the *two-loop* $q \to qg$ splitting function $P_{q\to qg}^{[2]}$ with α_S evaluated at $q_{\tilde{ij}}^2$,

$$\lim_{z\to 1} \alpha_S(q_{\tilde{ij}}^2) P_{q\to qg}^{[2]}(z) = \alpha_S \frac{2C_F}{1-z}\left(1-\alpha_S\beta_0 \ln(1-z) + \frac{\alpha_S}{2\pi}K_g\right) + \mathcal{O}(\alpha_S^3), \quad (103)$$

where[18]

$$K_g = C_A\left(\frac{67}{18} - \frac{\pi^2}{6}\right) - T_R n_f \frac{10}{9}. \quad (104)$$

On integrating over the phase space of the two-loop splitting function the K_g term gives rise to terms $\sim\alpha_S^2 \ln^2 q_{\tilde{ij}}^2$, *i.e.* it gives next-to-leading log soft-collinear contributions to the Sudakov exponent $\sim\alpha_S^n \ln^n q_{\tilde{ij}}^2$ (as opposed to leading-log contributions $\sim\alpha_S^n \ln^{n+1} q_{\tilde{ij}}^2$). In a similar way to that in which the higher order $\beta_0\alpha_S \ln(1-z)$ term was included, we may exploit the fact that the $z \to 1$ dependence of the K_g term in $P_{q\to qg}^{[2]}(z)$ is equal to that of $P_{q\to qg}^{[1]}(z)$, to incorporate it in the running coupling as well.

This is done by swapping the usual $\Lambda_{\overline{\rm MS}}$ QCD scale, from which the coupling runs, for $\Lambda_{\rm MC}$ [49],

$$\Lambda_{\rm MC} = \Lambda_{\overline{\rm MS}} \exp(K_g/4\pi\beta_0), \quad (105)$$

where MC denotes the so-called *Monte Carlo scheme*. Expanding $\alpha_S P_{q\to qg}^{[1]}(z)$ again, as in (103), but with α_S evaluated at $(1-z)q_{\tilde{ij}}^2$ in the MC scheme, reproduces exactly the two-loop result in (104). With this prescription the Sudakov form factor generally includes all leading and next-to-leading log contributions, except for those due to soft wide angle gluon emissions, however, for the case that the underlying hard process comprises of a single colour dipole, these are also included (see Sect. 6.2 and [50, 51]).

6.7.3 Options for the treatment of α_S in parton showers

Although we have made strong physical arguments restricting the argument of the coupling constant and suggesting a more physical renormalization scheme, there is still some degree of freedom in how precisely α_S is calculated. In what follows below we enumerate the options associated with these in the program.

InputOption This option selects the way in which initial conditions for running the coupling constant are determined. The default setting [**InputOption=AlphaMZ**] uses the experimentally determined value of α_S at the Z^0 resonance to calculate a value of $\Lambda_{\rm QCD}$ from which to run the coupling constant. This experimental input can be reset from the default value[19] of 0.127 using the **AlphaMZ** interface. Alternatively one may select an option [**InputOption=LambdaQCD**], which uses the input or default value of $\Lambda_{\overline{\rm MS}}$ to calculate the coupling. The default value used for $\Lambda_{\overline{\rm MS}}$ is 0.208 GeV, which may be reset using the interface **LambdaQCD**.

LambdaOption This option determines whether the value of $\Lambda_{\rm QCD}$, calculated from $\alpha_S(m_{Z^0})$ or input according to **InputOption**, is given in the MC (Monte Carlo) scheme of Ref. [49], as described in Sect. 6.7.2 [**LambdaOption=Same**], the default, or the $\overline{\rm MS}$ scheme [**LambdaOption=Convert**].

NumberOfLoops This parameter specifies the loop order of the beta function used to calculate the running of α_S. The default setting uses the three-loop beta function.

ThresholdOption This option selects whether to use the current [**ThresholdOption=Current**] or constituent [**ThresholdOption=Constituent**] quark masses in determining the flavour thresholds in the evolution of the coupling constant. The default setting uses the ($\overline{\rm MS}$) current quark masses.

Qmin The Qmin parameter represents the scale beneath which non-perturbative effects are considered to render the usual renormalization group running with a beta function determined at some finite loop order, invalid. Below this

[18] In fact the constants K_g are given by the finite remainder of the real emission phase space corrections due to the daughter gluon splitting discussed in the last Sect. 6.7.1 (see *e.g.* (5.28), (C.12), (C.13) of [61]).

[19] The default value is tuned to e^+e^- annihilation data as described in Appendix C and is typical of the values one gets when fitting leading order QCD predictions to data.

scale, which is currently tuned to 0.935 GeV, a number of parameterizations of the scaling of the coupling with energy may be selected according to the **NPAlphaS** option described below.

NPAlphaS The **NPAlphaS** option selects a parameterization of the scaling of the running coupling with energy in what we regard as the non-perturbative region, where the scale at which it is to be evaluated falls below the value set by Qmin. By setting [**NPAlphaS=Zero**] the coupling is simply taken to be zero for scales Q < Qmin. For [**NPAlphaS=Const**] the coupling *freezes out* at Qmin, *i.e.* it assumes the constant value $\tilde{\alpha}_S = \alpha_S(\mathrm{Qmin})$ for all scales below Qmin. This is the default parameterization. It is the same prescription used in early works on resummation by Curci and Greco [62, 63]. The options [**NPAlphaS=Linear**] and [**NPAlphaS=Quadratic**] calculate the running coupling below Qmin according to $\tilde{\alpha}_S \mathrm{Q}/\mathrm{Qmin}$ and $\tilde{\alpha}_S(\mathrm{Q}/\mathrm{Qmin})^2$ respectively. Setting [**NPAlphaS=Exx1**] assumes a quadratically decreasing running of the coupling in the non-perturbative region from the value **AlphaMaxNP** down to $\tilde{\alpha}_S$. Finally, [**NPAlphaS=Exx2**] sets α_S equal to **AlphaMaxNP** for all input scales Q < Qmin, which amounts to a minor variation of the default *freeze-out* option.

6.8 Matrix element corrections

As stated in Sect. 6.2, the effects of unresolvable gluon emissions have been included to all orders through the Sudakov form factor. The master formula and shower algorithms generate further resolvable emissions by approximating the full next-to-leading order real emission matrix element by a product of quasi-collinear splitting functions multiplying the tree level amplitude. Ideally, we wish to include higher-order effects as accurately as possible and do this for certain processes using matrix element corrections. We aim to correct two deficiencies of the shower algorithm: (i) it may not cover the whole phase space, leaving a region of high $p_\perp$ (*i.e.* non-soft non-collinear) emission unpopulated; and (ii) even in the region it does populate, as one extrapolates away from the soft and collinear limits it may not do a perfect job. We call these the *hard* and *soft* matrix element corrections respectively [64].

6.8.1 Soft matrix element corrections

In the parton shower approximation the probability density that the ith resolvable parton is emitted into $[\tilde{q}^2, \tilde{q}^2 + \mathrm{d}\tilde{q}^2]$, $[z, z + \mathrm{d}z]$ is

$$\mathrm{d}\mathcal{P}(z,\tilde{q}^2) = \frac{\alpha_S}{2\pi}\frac{\mathrm{d}\tilde{q}^2}{\tilde{q}^2}\,\mathrm{d}z\,P_{\tilde{ij}\to ij}(z,\tilde{q}^2)\Theta(\mathbf{p}_\perp^2 \geq 0). \tag{106}$$

This approximation works well for the case that the emission lies within the domain of the quasi-collinear limit. On the other hand the exact matrix element calculation gives us that the probability of a resolved emission as

$$\int_{\mathcal{R}} \mathrm{d}\mathcal{P}^{\mathrm{m.e.}} = \int \mathrm{d}\tilde{q}^2\mathrm{d}z\,\frac{1}{\sigma_0}\frac{\mathrm{d}^2\sigma}{\mathrm{d}z\mathrm{d}\tilde{q}^2}\,\Theta(\mathbf{p}_\perp^2 \geq 0), \tag{107}$$

where $\mathrm{d}\sigma$ is the differential cross section for the underlying process with a further parton emission, and $\mathcal{R}$ denotes the region of phase space corresponding to resolved emissions. The KLN and Bloch-Nordsieck theorems imply that all large logarithmic corrections to the cross section must vanish once the full available phase space is integrated over. It follows that the $\mathcal{O}(\alpha_S)$ contribution to the total cross section from an unresolved emission may be written $-\int_{\mathcal{R}} \mathrm{d}P^{\mathrm{m.e.}}$, at the level of large (leading and next-to-leading) logarithms. Proceeding in the same way as our earlier derivations (6.1), we then have that the probability density that the ith resolvable gluon is emitted into $[\tilde{q}^2, \tilde{q}^2 + \mathrm{d}\tilde{q}^2]$, $[z, z + \mathrm{d}z]$ is given by the integrand of

$$\int_{\tilde{q}^2_{\mathrm{min}}}^{\tilde{q}^2_{i-1}} \mathrm{d}\tilde{q}_i^2\mathrm{d}z\,\frac{1}{\sigma_0}\frac{\mathrm{d}^2\sigma}{\mathrm{d}z\mathrm{d}\tilde{q}_i^2}\,\exp\left(-\int_{\tilde{q}^2_{i-1}}^{\tilde{q}_i^2}\mathrm{d}\tilde{q}^2\mathrm{d}z\,\frac{1}{\sigma_0}\frac{\mathrm{d}^2\sigma}{\mathrm{d}z\mathrm{d}\tilde{q}^2}\right). \tag{108}$$

We may generate the distribution in (108) by simply augmenting the veto algorithm that is used to produce (35) with a single additional rejection weight, simply vetoing emissions if a random number $\mathcal{R}_S$ is such that

$$\mathcal{R}_S \geq \left.\frac{\mathrm{d}\mathcal{P}^{\mathrm{m.e.}}}{\mathrm{d}\mathcal{P}}\right|_{z,\tilde{q}^2}. \tag{109}$$

For this to work we require that the parton shower emission probability $\mathrm{d}P$ always overestimates that of the exact matrix element $\mathrm{d}P^{\mathrm{m.e.}}$, if necessary this can be achieved by simply enhancing the emission probability of the parton shower with a constant factor.

This correction is consistently applied to every emission that has the highest $\mathbf{p}_\perp$ *so far* in the shower. This ensures not only that the leading order expansion of the shower distribution agrees with the leading order matrix element, but also that the hardest (*i.e.* furthest from the soft and collinear limits) emission reproduces it. One might be concerned that it is really only proper to apply this correction to the final, largest $\mathbf{p}_\perp$ emission, however, in the context of a coherent parton branching formalism (angular ordering) the earlier wide-angle emission is considered too soft to resolve the subsequent, smaller angle but larger $\mathbf{p}_\perp$ splitting, and is therefore effectively distributed as if the latter emission did not occur. In this way, not only the hardest emission is improved by the correction, but *all* reasonably hard wide-angle emissions. Thus the correct procedure is to correct all those emissions that are the hardest so far, from the distribution in (35) to that in (108) by applying the veto in (109) [64].

Given that each soft matrix element correction amounts to exponentiating the next-to-leading order real emission matrix element divided by the leading order matrix element, provided one selects the option to evaluate the running coupling in the Monte Carlo scheme [49], the Sudakov form factor is in this case formally of next-to-leading log accuracy for corrections to processes comprised of a single colour flow.[20] For processes involving more than one underlying colour the next-to-leading log accuracy of the Sudakov form factor is only correct up to terms $\mathcal{O}(1/N_C^2)$ [50, 51].

6.8.2 Hard matrix element corrections

In addition to correcting the distribution of radiation inside the regions of phase space that are populated by the parton shower, we also wish to correct the distribution of radiation outside, in the high $\mathbf{p}_\perp$, unpopulated, dead region. We wish to distribute the radiation in the dead regions according to the exact tree-level real emission matrix element *i.e.* according to

$$\frac{1}{\sigma_0}\int_{x_{i,\min}}^{x_{i,\max}} \mathrm{d}x_i \int_{x_{j,\min}(x_i)}^{x_{j,\max}(x_i)} \mathrm{d}x_j \, \frac{\mathrm{d}^2\sigma}{\mathrm{d}x_i \mathrm{d}x_j}, \tag{110}$$

where $\mathrm{d}\sigma$ is the differential cross section obtained using the next-to-leading order, real emission matrix element, and (x_i, x_j) are variables parameterizing the phase space associated with the emission of the extra parton.

The algorithm for populating the dead region is basic in principle. Prior to any showering the program checks if a matrix element correction is available for the hard process. If one is available the algorithm then generates a point in the appropriate region of phase space, ideally with some importance sampling of the integrand. The differential cross section associated with this point, as given in (110), is evaluated and multiplied by a phase space volume factor $\mathcal{V}(x_i)$ given by

$$\mathcal{V}(x_i) = (x_{i,\max} - x_{i,\min})(x_{j,\max}(x_i) - x_{j,\min}(x_i)), \tag{111}$$

giving the event weight. The emission is retained if this weight is less than a uniformly distributed random number $\mathcal{R} \in [0, 1]$, and the momenta of the new parton configuration are reconstructed from the generated values of x_i and x_j.

6.8.3 Using Herwig++ matrix element corrections

The current version of Herwig++ contains matrix element corrections for four different hard processes: neutral and charged current Drell-Yan processes, $gg \to h^0$, top quark decays and $e^+e^- \to q\bar{q}$ processes. The associated C++ classes are DrellYanMECorrection, GGtoHMECorrection, TopDecayMECorrection and VectorBosonQQbarMECorrection.

Naturally each of these process-dependent matrix element corrections checks whether it corresponds to the hard process (or, for top quark decays, the decay process). In other words, users need not worry that, if matrix element corrections are globally switched on in the code, the correction for *e.g.* the Drell-Yan processes is applied to the $gg \to h^0$ process they have selected to generate.

All three corrections are loaded in the Repository in the default set-up. The switch **MECorrMode** determines the way in which all matrix elements are used. If [**MECorrMode=0**] is selected no matrix element corrections will be applied at all. The default setting [**MECorrMode=1**], applies *both* the hard and soft matrix element corrections for each one loaded in the Repository (if the associated processes are generated). Options [**MECorrMode=2**] and [**MECorrMode=3**] turn *off* the soft and hard matrix element corrections respectively.

6.9 Code structure

The Herwig++ shower module consists of a large number of classes and is designed to be flexible, in the sense that any DGLAP-type shower evolution based on $1 \to 2$ branchings where momentum conservation is enforced globally after the evolution has been performed can be implemented. The only concrete implementation so far is the improved angular-ordered shower based on [17] and described above.

We will only describe the structure of the code, *i.e.* how the various classes work together to generate the parton shower evolution. Detailed documentation of all the classes can be found in the Doxygen documentation. In a future release, the structure will be slightly changed to allow for more general shower evolution, such as dipole-type showers.

The main class implementing the Herwig++ shower is the ShowerHandler class, which inherits from the CascadeHandler class of ThePEG. It has responsibility for the overall administration of the multiple interactions, as described in Sect. 8, the showering of primary and secondary hard scattering processes, the decay of any unstable fundamental particles[21] and the generation of any radiation produced in their decays. The ShowerHandler uses a number of helper classes to implement various parts of the algorithm together with some data storage classes, which hold information needed to generate the parton shower.

The ShowerHandler proceeds as follows:

[20] For processes involving initial-state radiation, this also requires evaluating the parton densities at a scale of order $\mathbf{p}_\perp$ [50].

[21] Currently most fundamental particle decays are performed before the parton shower is generated, although in future we plan to generate them as part of the parton-shower algorithm.

– The Event object supplied to the ShowerHandler is first analysed and the particles to be showered extracted. These particles are converted from Particle objects, which store particle information in ThePEG, to ShowerParticle objects, which inherit from Particle and include the storage of the additional information, such as the evolution scales and colour partners, needed to generate the parton shower. Each particle in a hard process, be that the primary scattering process or the subsequent decay of a fundamental particle, is assigned to a ShowerProgenitor object containing references to the particle together with additional information required for particles that initiate a parton shower. For each hard process a ShowerTree object is created containing the ShowerProgenitor objects for all the particles in the hard process and the information required to shower that process.
– The ShowerHandler uses the helper Evolver to generate the radiation from each hard scattering or decay process. Once the parton showers have been generated for all the hard processes the ShowerHandler inserts them into the Event object.
– The MPIHandler then generates any secondary hard scatterings required, which are subsequently showered by the Evolver, as described in Sect. 8.
– Finally, after all the scatterings have been showered, the hadronic remnant is decayed to conserve momentum and flavour using the HwRemDecayer class.

The main helper class of the ShowerHandler is the Evolver, which is responsible for generating the parton shower from an individual hard process, stored as a ShowerTree object. The Evolver first finds the colour partners and initial scale for the parton showers from each particle, as described in Sect. 6.3. At this stage, if there is a suitable class inheriting from MECorrectionBase, which implements the matrix element correction for the process as described in Sect. 6.8, the hard matrix element correction is applied. The Evolver is also currently responsible for generating the intrinsic $p_\perp$ of incoming partons in hadronic collisions at this stage.

Given the initial scale, the evolution of the particles proceeds as described in Sects. 6.4–6.6, using the SplittingGenerator class to generate the types and scales of the branchings. In turn the SplittingGenerator uses the SudakovFormFactor to generate the possible evolution scales for each allowed type of branching and then selects the branching with the highest scale, as described in Sect. 6.4. The new ShowerParticles produced in the branching are then evolved until no further branching is possible. When all the particles have been evolved the KinematicsReconstructor reconstructs the momentum of all the particles in the shower (Sects. 6.4–6.6).

The ShowerHandler and Evolver classes are mainly administrative, the actual physics is implemented in the various helper classes. For this reason these helper classes, which are specific to the details of the parton shower algorithm, are contained in the ShowerModel class. It is intended that different DGLAP based parton shower algorithms, for example the original angular-ordered parton shower algorithm used in FORTRAN HERWIG, can be implemented by inheriting from the ShowerModel and specifying the helper classes to be used in that model, which inherit from the KinematicsReconstructor, PartnerFinder, SudakovFormFactor and MECorrectionBase classes. For example, the QTildeModel, which implements the improved angular-ordered shower described above, uses the QTildeReconstructor, QTildeFinder, QTildeSudakov and QTildeMECorrection classes.

In turn many of the helper classes used by the main classes implementing the shower have their own helper classes for various parts of the simulation.

The SplittingGenerator class holds lists of available branchings, providing interface switches to either enable or disable radiation, in the initial or final state, for different interactions. They are used to generate the shower variables associated with each branching using SudakovFormFactor objects. The SplittingGenerator and SudakovFormFactor classes use the following helper classes:

SplittingFunction This is the base class for defining splitting functions used in the shower evolution. This includes the calculation of the splitting function together with the overestimate, integral and inverse integral of it required to implement the veto algorithm as described in Sects. 6.4 and 6.5. The splitting functions implemented in Herwig++ are listed in Sect. 6.2.

ShowerAlpha This is the base class implementing the running couplings used in the shower evolution.

The Evolver uses the ShowerVeto class to provide a general interface to veto emission attempts by the shower. The veto may be applied to either a single emission (resetting the evolution scale for the particle to the attempted branching scale), an attempt to shower a given event, or the overall event generation.

Finally three special exception classes are used inside the shower module, mainly to communicate exceptional events or configurations, rather than signaling a serious error during event generation. The exceptions are handled completely within the shower module. In particular we use VetoShower to communicate vetoing of a complete shower attempt. KinematicsReconstructionVeto is used to signal an exceptional configuration that cannot be handled by the KinematicsReconstructor, resulting in restarting the shower from the original event (similar to a VetoShower exception). ShowerTriesVeto signals that complete showering of a given event failed a predefined number of times. This is handled together with the generation of multiple interactions.

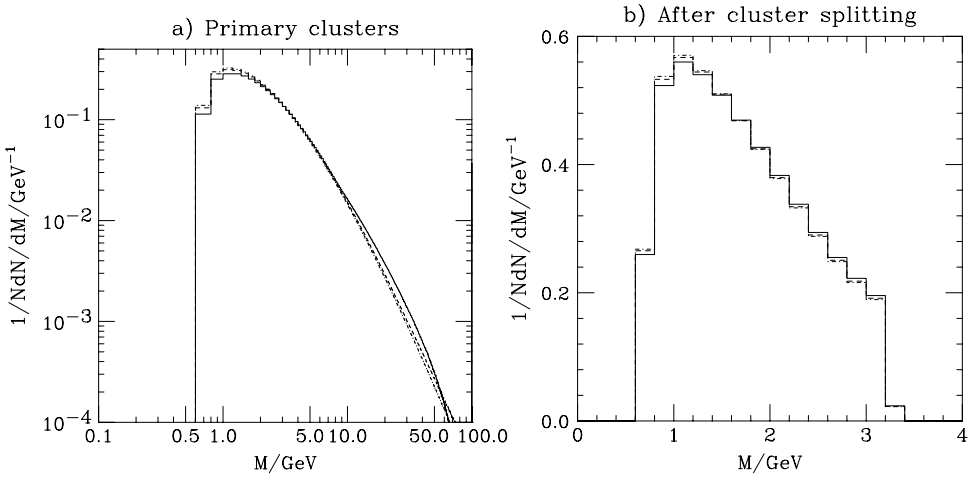


Fig. 4 The mass spectrum of (**a**) the primary clusters and (**b**) the clusters after cluster fission. The *solid*, *dashed* and *dot-dashed lines* show the clusters produced in hadronization of $e^+e^- \to d\bar{d}$ events at a centre-of-mass energy of 100 GeV, 1 TeV and 10 TeV respectively. Only clusters containing light quarks are shown

7 Hadronization

After the parton shower, the quarks and gluons must be formed into the observed hadrons. The colour preconfinement property [60] of the angular-ordered parton shower is used as the basis of the cluster model [2], which is used in Herwig++ to model the hadronization. This model has the properties that it is local in the colour of the partons and independent of both the hard process and centre-of-mass energy of the collision [2, 3].

7.1 Gluon splitting and cluster formation

The first step of the cluster hadronization model is to non-perturbatively split the gluons left at the end of the parton shower into quark-antiquark pairs. Since, at the end of the Herwig++ shower the gluons are given their constituent mass it is essential that this mass is heavier than twice the constituent mass of the lightest quark.[22] The gluon is allowed to decay into any of the accessible quark flavours with probability given by the available phase space for the decay.[23]

The gluon decays isotropically and following this isotropic decay the event only contains colour connected (di)quarks and anti-(di)quarks. The colour singlets formed by these colour connected parton pairs are formed into clusters with the momentum given by the sum of the momenta of the constituent partons. The principle of colour-preconfinement states that the mass distribution of these clusters is independent of the hard scattering process and its centre-of-mass energy. As can be seen in Fig. 4a, the shower algorithm in Herwig++ obeys preconfinement fairly well by 100 GeV and is clearly invariant beyond that.

7.2 Cluster fission

The cluster model is based on the observation that because the cluster mass spectrum is both universal and peaked at low masses, as shown in Fig. 4a, the clusters can be regarded as highly excited hadron resonances and decayed, according to phase space, into the observed hadrons. There is however a small fraction of clusters that are too heavy for this to be a reasonable approach. These heavy clusters are therefore first split into lighter clusters before they decay.

A cluster is split into two clusters if the mass, M, is such that

$$M^{\mathbf{Cl_{pow}}} \geq \mathbf{Cl_{max}}^{\mathbf{Cl_{pow}}} + (m_1 + m_2)^{\mathbf{Cl_{pow}}}, \tag{112}$$

where $\mathbf{Cl_{max}}$ and $\mathbf{Cl_{pow}}$ are parameters of the model, and $m_{1,2}$ are the masses of the constituent partons of the cluster. In practice, in the most recent version of the model, in order to improve the description of the production of bottom and charm hadrons, we include separate values of both $\mathbf{Cl_{max}}$ (**ClMaxLight**, **ClMaxCharm** and **ClMaxBottom**) and $\mathbf{Cl_{pow}}$ (**ClPowLight**, **ClPowCharm**, **ClPowBottom**) for clusters containing light, charm and bottom quarks respectively. The default values of these and other important hadronization parameters are given in Table 10 at the end of this Section.

For clusters that need to be split, a $q\bar{q}$ pair is selected to be popped from the vacuum. Only up, down and strange quarks are chosen with probabilities given by the parameters $\mathbf{Pwt}_i$,[24] where i is the flavour of the quark. Once a pair is

[22] We normally take the constituent masses of the up and down quarks to be equal although they can in principle be different.

[23] The option of gluon decay into diquarks, which was available in FORTRAN HERWIG is no longer supported. Diquarks are therefore present only as remnants of incoming baryons, or from baryon number violating processes (see Sect. 7.4.2).

[24] We use $\mathbf{Pwt}_i$ to denote the probability of selecting a given quark or diquark. This is given by the parameters **PwtDquark**, **PwtUquark**, **PwtSquark**, **PwtCquark** and **PwtBquark** for the quarks and the product of the diquark probability **PwtDiquark**, the probabilities of the quarks forming the diquark, and a symmetry factor for diquarks.

selected the cluster is decayed into two new clusters with one of the original partons in each cluster. Unless one of the partons is a remnant of the incoming beam particle the mass distribution of the new clusters is given by

$$M_1 = m_1 + (M - m_1 - m_q)\mathcal{R}_1^{1/P}, \tag{113a}$$

$$M_2 = m_2 + (M - m_2 - m_q)\mathcal{R}_2^{1/P}, \tag{113b}$$

where m_q is the mass of the parton popped from the vacuum and $M_{1,2}$ are the masses of the clusters formed by the splitting. The distribution of the masses of the clusters is controlled by the parameter P, which is **PSplitLight**, **PSplitCharm** or **PSplitBottom** for clusters containing light, charm or bottom quarks.

In addition to the selection of the mass according to (113) the masses of the daughter clusters are required to be less than that of the parent cluster and greater than the sum of the masses of their constituent partons. The spectrum of the cluster masses after the cluster splitting is shown in Fig. 4b.

For clusters that contain a remnant of the beam particle in hadronic collisions a soft distribution is used for the masses of the clusters produced in the splitting. The **RemnantOption** switch controls whether the soft distribution is used for both daughter clusters [**RemnantOption=0**] or only the daughter cluster containing the remnant [**RemnantOption =1**], the default. The mass of the soft clusters is given by

$$M_i = m_i + m_q + x, \tag{114}$$

where x is distributed between 0 and $M - m_1 - m_2 - 2m_q$ according to

$$\frac{\mathrm{d}P}{\mathrm{d}x^2} = \exp(-bx), \tag{115}$$

where $b = 2/$**SoftClusterFactor**.

7.3 Cluster decays

The final step of the cluster hadronization model is the decay of the cluster into a pair of hadrons. For a cluster of a given flavour $(q_1, \bar{q}_2)$ a quark-antiquark or diquark-antidiquark pair $(q, \bar{q})$ is extracted from the vacuum and a pair of hadrons with flavours $(q_1, \bar{q})$ and $(q, \bar{q}_2)$ formed. The hadrons are selected from all the possible hadrons with the appropriate flavour based on the available phase space, spin and flavour of the hadrons. While the general approach is the same in all cluster models there are some variations. In Herwig++ the original model of Ref. [2] used in FORTRAN HERWIG [5, 6], the approach of Ref. [65], which was designed to solve the problem of isospin violation in the original model if incomplete SU(2) multiplets of hadrons are included, and a new variant that addresses the issue of the low rate of baryon production in the approach of Ref. [65], are implemented.

In all these approaches the weight for the production of the hadrons $a_{(q_1,\bar{q})}$ and $b_{(q,\bar{q}_2)}$ is

$$W(a_{(q_1,\bar{q})}, b_{(q,\bar{q}_2)}) = P_q w_a s_a w_b s_b p^*_{a,b}, \tag{116}$$

where P_q is the weight for the production of the given quark-antiquark or diquark-antidiquark pair, $w_{a,b}$ are the weights for the production of individual hadrons and $s_{a,b}$ are the suppression factors for the hadrons, which allow the production rates of individual meson multiplets, and singlet and decuplet baryons to be adjusted. The momentum of the hadrons in the rest frame of the decaying cluster,

$$p^*_{a,b} = \frac{1}{2M}[(M^2 - (m_a + m_b)^2)(M^2 - (m_a - m_b)^2)]^{\frac{1}{2}}, \tag{117}$$

measures the phase space available for two-body decay. If the masses of the decay products are greater than the mass of the cluster then the momentum is set to zero. The weight for the individual hadron is

$$w_h = w_{\mathrm{mix}}(2J_h + 1), \tag{118}$$

where w_{mix} is the weight for the mixing of the neutral light mesons[22] and J_h is the spin of the hadron.

The different approaches vary in how they implement the selection of the cluster decay products based on this probability.

In the approach of Ref. [2] the probability is generated in a number of pieces. First the flavour of the quark-antiquark, or diquark-antidiquark, pair popped from the vacuum is selected with probability

$$P_q = \frac{\mathbf{Pwt}_q}{\sum_{q'} \mathbf{Pwt}_{q'}}. \tag{119}$$

Both the hadrons produced in the cluster decay are then selected from the available hadrons of the appropriate flavours using the weight

$$P_h = \frac{w_h}{w_{\max(q,\bar{q'})}}, \tag{120}$$

where $w_{\max(q,\bar{q'})}$ is the maximum value of the weight for a given flavour combination.

A weight is calculated for this pair of hadrons

$$W = \frac{s_a s_b p^*_{a,b}}{p^*_{\max}}, \tag{121}$$

[22] $w_{\mathrm{mix}} = 1$ for all other particles.

where $p^*_{a,b}$ is the momentum of the hadrons in the cluster rest frame and $p^*_{\max}$ is the maximum momenta of the decay products for hadrons with the relevant flavour.[23] The hadrons produced are then selected according to this weight.

This procedure approximately gives a probability

$$P(a_{(q_1,\bar{q})}, b_{(q,\bar{q}_2)}|q_1,\bar{q}_2) = P_q \frac{1}{N_{(q_1,\bar{q})}} \frac{1}{N_{(q,\bar{q}_2)}} \frac{w_a}{w_{\max(q_1,\bar{q})}} \frac{w_b}{w_{\max(q,\bar{q}_2)}} \frac{s_a s_b p^*_{a,b}}{p^*_{\max}} \tag{122}$$

of choosing hadrons $a_{(q_1,\bar{q})}$ and $b_{(q,\bar{q}_2)}$. The number of hadrons with flavour $(q_1,\bar{q}_2)$ is $N_{(q_1,\bar{q}_2)}$.

Kupco [65] pointed out one problem with this approach: as new hadrons with a given flavour are added, the production of the existing hadrons with the same flavour is suppressed. In order to rectify this problem he proposed a new approach for choosing the decay products of the cluster. Instead of splitting the probability into separate parts, as in Ref. [2], a single weight was calculated for each combination of decay products

$$W(a_{(q_1,\bar{q})}, b_{(q,\bar{q}_2)}|q_1,\bar{q}_2) = P_q w_a w_b s_a s_b p^*_{a,b}, \tag{123}$$

which gives the probability of selecting the combination

$$P(a_{(q_1,\bar{q})}, b_{(q,\bar{q}_2)}|q_1,\bar{q}_2) = \frac{W(a_{(q_1,\bar{q})}, b_{(q,\bar{q}_2)}|q_1,\bar{q}_2)}{\sum_{c,d,q'} W(c_{(q_1,\bar{q}')}, d_{(q',\bar{q}_2)}|q_1,\bar{q}_2)}. \tag{124}$$

The addition of new hadrons now increases the probability of choosing a particular flavour, however because these new hadrons are usually heavy they will not contribute for the majority of light clusters.

The main problem with this approach is that because many more mesons are included in the simulation than baryons not enough baryons are produced. In order to address this problem in Herwig++, if a cluster mass is sufficiently large that it can decay into the lightest baryon-antibaryon pair the parameter $\mathbf{Pwt}_{qq}$ is used to decide whether to select a mesonic or baryonic decay of the cluster. The probabilities of selecting a mesonic decay or baryonic decay are $\frac{1}{1+\mathbf{Pwt}_{qq}}$ and $\frac{\mathbf{Pwt}_{qq}}{1+\mathbf{Pwt}_{qq}}$. This modification not only increases the number of baryons produced but gives direct control over the rate of baryon production.

Once the decay products of the cluster are selected, the cluster is decayed. In general the cluster decay products are isotropically distributed in the cluster rest frame. However, hadrons that contain a parton produced in the perturbative stage of the event retain the direction of the parton in the cluster rest frame, apart from a possible Gaussian smearing of the direction. This is controlled by the **ClDir** parameter, which by default [**ClDir=true**] retains the parton direction, and the **ClSmr** parameter, which controls the Gaussian smearing through an angle θ_{smear} where

$$\cos\theta_{\text{smear}} = 1 + \mathbf{ClSmr}\log\mathcal{R}. \tag{125}$$

The azimuthal angle relative to the parton direction is distributed uniformly. To provide greater control the parameters **ClDir** (**ClDirLight**, **ClDirCharm** and **ClDirBottom**) and **ClSmr** (**ClSmrLight**, **ClSmrCharm** and **ClSmrBottom**) can be set independently for clusters containing light, charm and bottom quarks.

In practice there is always a small fraction of clusters that are too light to decay into two hadrons. These clusters are therefore decayed to the lightest hadron, with the appropriate flavours, together with a small reshuffling of energy and momentum with the neighbouring clusters to allow the hadron to be given the correct physical mass. The cluster with the smallest space-time distance that can absorb the recoil is used. In addition, for clusters containing a bottom or charm quark in order to improve the behaviour at the threshold the option exists of allowing clusters above the threshold mass, $M_{\text{threshold}}$, for the production of two hadrons to decay into a single hadron such that a single hadron can be formed for masses

$$M < M_{\text{limit}} = (1 + \mathbf{SingleHadronLimit})M_{\text{threshold}}. \tag{126}$$

The probability of such a single-meson cluster decay is assumed to decrease linearly for $M_{\text{threshold}} < M < M_{\text{limit}}$. The parameters **SingleHadronLimitCharm** and **SingleHadronLimitBottom** control the limit on the production of single clusters for charm and bottom clusters respectively. Increasing the limit has the effect of hardening the momentum spectrum of the heavy mesons.

7.3.1 Mixing weights

For neutral mesons that only contain the light (up, down and strange) quarks there is mixing. If we consider the wavefunctions of the neutral mesons, which we write for the $^1S_0$ meson multiplet but the treatment applies to an arbitrary SU(3) flavour multiplet, then

$$\pi^0 = \frac{1}{\sqrt{2}}(d\bar{d} - u\bar{u}), \tag{127a}$$

$$\eta = \psi_8\cos\theta - \psi_1\sin\theta, \tag{127b}$$

$$\eta' = \psi_8\sin\theta + \psi_1\cos\theta, \tag{127c}$$

where θ is the nonet mixing angle and the wavefunctions for the octet and singlet components are

$$\psi_8 = \frac{1}{\sqrt{6}}(u\bar{u} + d\bar{d} - 2s\bar{s}), \tag{128a}$$

[23] That is, the momentum with the lightest possible choices for a and b.

$$\psi_1 = \frac{1}{\sqrt{3}}(u\bar{u} + d\bar{d} + s\bar{s}). \tag{128b}$$

The probabilities of finding a given quark-antiquark inside a particular neutral meson can be calculated, which gives the mixing weights for the neutral light mesons

$$w^{\pi^0}_{u\bar{u}} = w^{\pi^0}_{d\bar{d}} = \frac{1}{2}, \qquad w^{\pi^0}_{s\bar{s}} = 0, \tag{129a}$$

$$w^{\eta}_{u\bar{u}} = w^{\eta}_{d\bar{d}} = \frac{1}{2}\cos^2(\theta+\phi), \qquad w^{\eta}_{s\bar{s}} = \sin^2(\theta+\phi), \tag{129b}$$

$$w^{\eta'}_{u\bar{u}} = w^{\eta'}_{d\bar{d}} = \frac{1}{2}\sin^2(\theta+\phi), \qquad w^{\eta'}_{s\bar{s}} = \cos^2(\theta+\phi), \tag{129c}$$

where $\phi = \tan^{-1}\sqrt{2}$ is the ideal mixing angle.

In the approach of Ref. [2] the factor of $\frac{1}{2}$ in the weights for the $u\bar{u}$ and $d\bar{d}$ components was omitted as this is approximately given by the ratio of the number of charged mesons containing up and down quarks to neutral ones, which is exactly two for ideal mixing where the $s\bar{s}$ mesons do not mix with those containing up and down quarks.

In practice the mixing angles can be adjusted for each meson multiplet that is included in the simulation although with the exception of the lightest pseudoscalar, vector, tensor and spin-3 multiplets the assumption of ideal mixing is used.

7.4 Hadronization in BSM models

In most cases the hadronization of events involving new physics, using the cluster model, proceeds in the same way as for Standard Model events. There are however some classes of new physics model that require special treatment, in particular:

Stable strongly interacting particles if there are strongly interacting particles that are stable on the hadronization timescale, these particles will hadronize before they decay. If the new particles are in the fundamental representation of colour SU(3) then their hadronization proceeds in the same way as for quarks, however if they are in the octet representation the situation is more complicated [66].

Baryon number violation (BNV) there are models of new physics in which the conservation of baryon number is violated. This typically occurs at a vertex that has the colour tensor ϵ^{ijk} leading to three quarks, or antiquarks, that are colour connected to each other after the parton shower and gluon splitting.

The Herwig++ hadronization module is designed so that both stable coloured particles and baryon number violation are correctly treated as described below.

7.4.1 Stable strongly interacting particles

Currently only the hadronization of objects in the fundamental representation of the SU(3) group of the strong force is supported. Provided that the relevant hadrons exist the hadronization of these particles is handled in the same way as for quarks. In the future we will extend this to new particles in the octet representation as described in Ref. [66].

7.4.2 Baryon number violation

The treatment of QCD radiation and hadronization in models that violate baryon number conservation is described in Refs. [53] and [54] and was implemented in the FORTRAN HERWIG program. In events where baryon number is violated there are typically two situations that can arise.

1. The baryon number violating vertices are unconnected, leading to three quarks, or antiquarks, connected to each BNV vertex after the gluon splitting. These (anti)quarks must be formed into a cluster, which decays to give a (anti)baryon and a meson, giving the expected baryon number violation. In the approach of Refs. [53, 54] this is modelled by first combining two of the (anti)quarks into a (anti)diquark, which is in the (anti)-triplet representation of colour SU(3). The (anti)quark and (anti)diquark can then be formed into a colour singlet cluster, which can be handled by the hadronization module in the normal way.
2. Two baryon number violating vertices are colour connected to each other, leading to two quarks connected to one vertex and two antiquarks connected to the second, after gluon splitting. In this case two clusters must be formed by pairing one of the quarks with one of the antiquarks at random and then pairing up the remaining pair.

The handling of these colour flows in both the shower and hadronization is fully supported although there are currently no models that include baryon number violation implemented.

7.5 Code structure

The ClusterHandronizationHandler inherits from the HadronizationHandler of ThePEG and implements the cluster hadronization model. The ClusterHandronizationHandler makes use of a number of helper classes to implement different parts of the model. The helper classes, in the order they are called, are:

PartonSplitter The PartonSplitter performs the non-perturbative splitting of the gluons in quark-antiquark pairs.

ClusterFinder The ClusterFinder is responsible for taking the partons after the gluon splitting and forming them into colour singlet clusters as Cluster particles.

Table 10 Important hadronization parameters. For all parameters other than the light parton constituent masses, the limits given are enforced by the interface. For the light partons, the limits are not enforced but give a sensible range over which the parameters can be varied. For the gluon, the upper limit we give is about the largest value we would consider reasonable, although it is not a hard limit. The up and down masses must be less than half the gluon mass, otherwise the non-perturbative gluon decays are impossible, and the strange mass must be large enough that gluon decays into strange quarks are not possible, to give good agreement with LEP data

Parameter	Default Value	Allowed Range	Description
HadronSelector			
PwtDquark	1.	0–10	Weight for choosing a down quark
PwtUquark	1.	0–10	Weight for choosing a up quark
PwtSquark	0.68	0–10	Weight for choosing a strange quark
PwtDIquark	0.52	0–10	Weight for choosing a diquark
SngWt	0.96	0–10	Weight for singlet baryons
DecWt	0.61	0–10	Weight for decuplet baryons
LightClusterDecayer			
SingleHadronLimitBottom	0.16	0–10	Bottom cluster to 1 hadron param.
SingleHadronLimitCharm	0.0	0–10	Charm cluster to 1 hadron param.
ClusterDecayer			
ClDirLight	1	0/1	Orientation of light cluster decays
ClDirBottom	1	0/1	Orientation of bottom cluster decays
ClDirCharm	1	0/1	Orientation of charm clusters
ClSmrLight	0.78	0–2	Smearing of light cluster decays
ClSmrBottom	0.10	0–2	Smearing of bottom cluster decays
ClSmrCharm	0.26	0–2	Smearing of charm cluster decays
OnShell	0	0/1	Masses of produced hadrons
ClusterFissioner			
ClMaxLight	3.15	0–10	Max. mass for light clusters (GeV)
ClMaxBottom	3.10	0–10	Max. mass for bottom clusters (GeV)
ClMaxCharm	3.00	0–10	Max. mass for bottom clusters (GeV)
ClPowLight	2.00	0–10	Mass exponent for light clusters
ClPowBottom	1.18	0–10	Mass exponent for bottom clusters
ClPowCharm	1.52	0–10	Mass exponent for charm clusters
PSplitLight	1.20	0–10	Splitting param. for light clusters
PSplitBottom	1.00	0–10	Splitting param. for bottom clusters
PSplitCharm	1.18	0–10	splitting param. for charm clusters
RemnantOption	1	0/1	Treatment of remnant clusters
SoftClusterFactor	1	0.1–10	Remnant mass param. (GeV)
ConstituentMasses of light partons (set in their ParticleData objects)			
gluon	0.9	0–1	Gluon constituent mass (GeV)
up	0.325	0–$m_g/2$	Up quark constituent mass (GeV)
down	0.325	0–$m_g/2$	Down quark constituent mass (GeV)
strange	0.5	$m_g/2$–1	Strange quark constituent mass (GeV)

ColourReconnector It is possible that rather than using the leading N_c colour structure of the event there is some rearrangement of the colour connections. The option of implementing such a model in a class inheriting from the ColourReconnector class is available, although the ColourReconnector itself does not implement such a model.

ClusterFissioner The ClusterFissioner class is responsible for splitting large mass clusters into lighter ones as described in Sect. 7.2.

LightClusterDecayer The LightClusterDecayer decays any clusters for which the decay to two hadrons is kinematically impossible into the lightest hadron with the correct flavour together with the reshuffling of momentum with neighbouring clusters, which is required to conserve energy and momentum, as described at the end of Sect. 7.3.

ClusterDecayer The ClusterDecayer decays the remaining clusters into pairs of hadrons as described in Sect. 7.3.

In addition to these classes the ClusterDecayer makes use of a HadronSelector to select the hadrons produced in the cluster decay.[24] In order to support the different options described in Sect. 7.3 the base HadronSelector implements much of the functionality needed to select the hadrons in the cluster model but the chooseHadronPair() method, which is used to select the hadrons, is virtual and must be implemented in inheriting classes that implement specific variants of the cluster model. The FORTRAN HERWIG algorithm is implemented in the Hw64Selector class and the Kupco and Herwig++ methods in theHwppSelector class.

There are a number of switches and parameters that control the hadronization. Here we merely give a summary of the most important ones. All the parameters are described in full in the Doxygen documentation of the relevant classes.

The main choice is which variant of the cluster model to use. This can be controlled by using either the Hw64Selector for the original model of Ref. [2] or theHwppSelector class for the Kupco and Herwig++ variants. The choice of whether to use the Hw64Selector or HwppSelector is controlled by setting the **HadronSelector** interface of the ClusterDecayer and LightClusterDecayer classes. In addition, for the HwppSelector the **Mode** switch controls whether the Kupco [**Mode=0**] orHerwig++ [**Mode=1**], the default, variant is used. The production of specific hadrons by the cluster model can be forbidden via the **Forbidden** interface of the HadronSelector: this option is currently only used to forbid the production of the σ and κ resonances, which are only included in the simulation to model low-mass s-wave $\pi\pi$ and $K\pi$ systems in certain particle decays.

In addition the mixing angles for the various multiplets can be changed in the HadronSelector as can the suppression weights for different SU(3) meson flavour multiplets.

If the option of using the soft underlying event model [67] is used, as described in Sect. 8.3, then the **UnderlyingEventHandler** needs to be set to theUA5Handler, by default this is set to the NULL pointer and the multiple scattering model of the underlying event described in Sect. 8 used.

The other main parameters of the cluster model, and their default values, are given in Table 10.

Finally the **ConstituentMass** of the gluon and, to a lesser extent the light quarks, which can be set in their ParticleData objects, have a major effect on the hadronization since they set the scale for the cluster mass distribution.

8 Underlying event and beam remnants

The default underlying event model of Herwig++ is currently based on the eikonal model discussed in Refs. [16, 68, 69]. Further development is planned, but so far it is intended to provide very similar functionality to FORTRAN HERWIG + JIMMY with some minor improvements. That is, the underlying event is modelled as multiple partonic interactions, where one of them is the process of interest, accompanied by several semi-hard scatterings.

In this section, we briefly discuss the basics of how to calculate the multiplicities of the semi-hard scatterings, before explaining the integration into the full Monte Carlo simulation. For historical reasons, we also briefly mention an alternative underlying event model available in Herwig++: the UA5 model [67], even though this is ruled out by data and not recommended for serious use. Finally we will describe the code structure, which implements these ideas. A more detailed explanation can be found in Ref. [8].

8.1 Model basics

The starting point is the observation that the cross section for QCD jet production may exceed the total pp or $p\bar{p}$ cross section already at an intermediate energy range and eventually violates unitarity. For example, for QCD jet production with a minimum p_T of 2 GeV this already happens at $\sqrt{s} \sim 1$ TeV. This p_T cutoff should however be large enough to ensure that we can calculate the cross section using pQCD. The reason for the rapid increase of the cross section turns out to be the strong rise of the proton structure function at small x, since the x values probed decrease with increasing centre of mass energy. This proliferation of low x partons may lead to a non-negligible probability of having more than one partonic scattering in the same hadronic collision. This is not in contradiction with the definition of the standard parton distribution function as the *inclusive* distribution of a parton in a hadron, with all other partonic interactions summed and integrated out. It does, however, signal the onset of a regime in which the simple interpretation of the pQCD calculation as describing the only partonic scattering must be unitarized by additional scatters.

In principle, predicting the rate of multi-parton scattering processes requires multi-parton distribution functions, about which we have almost no experimental information. However, the fact that the standard parton distribution functions describe the inclusive distribution gives a powerful constraint, which we can use to construct a simple model. The eikonal model used in Refs. [16, 68, 69] derives from the assumption that at fixed impact parameter, b, individual scatterings are independent and that the distribution of partons in hadrons factorizes with respect to the b and x dependence. This implies that the average number of partonic collisions at a given b value is

$$\langle n(b,s)\rangle = A(b)\,\sigma^{\mathrm{inc}}(s;\,p_T^{\mathrm{min}})\,, \tag{130}$$

[24] The LightClusterDecayer also makes use of this class to select the lightest hadron with a given flavour.

where $A(b)$ is the partonic overlap function of the colliding hadrons, with

$$\int \mathrm{d}^2b\, A(b) = 1, \tag{131}$$

and σ^{inc} is the inclusive cross section to produce a pair of partons with $p_T > p_T^{\mathrm{min}}$. We model the impact parameter dependence of partons in a hadron by the electromagnetic form factor, resulting in an overlap function for pp and $p\bar{p}$ collisions of

$$A(b;\mu) = \frac{\mu^2}{96\pi}(\mu b)^3 K_3(\mu b), \tag{132}$$

where μ is the inverse proton radius and $K_3(x)$ is the modified Bessel function of the third kind. We do not fix μ at the value determined from elastic ep scattering, but rather treat it as a free parameter, because the spatial parton distribution is assumed to be similar to the distribution of charge, but not necessarily identical.

The assumption that different scatters are uncorrelated leads to the Poissonian distribution for the number of scatters, n, at fixed impact parameter,

$$\mathcal{P}_n(b,s) = \frac{\langle n(b,s)\rangle^n}{n!}\,\exp(-\langle n(b,s)\rangle)\,. \tag{133}$$

Using (133) the unitarized cross section can now be written as

$$\begin{aligned}\sigma_{\mathrm{inel}}(s) &= \int \mathrm{d}^2b \sum_{k=1}^{\infty} \mathcal{P}_k(b,s) \\ &= \int \mathrm{d}^2b\,[1-\exp(-\langle n(b,s)\rangle)],\end{aligned} \tag{134}$$

which properly takes multiple scatterings into account. The key ingredient for the Monte Carlo implementation is then the probability of having n scatterings given there is at least one, integrated over impact parameter space. This expression reads

$$\mathrm{P}_n(s) = \frac{\int \mathrm{d}^2b\, \mathcal{P}_n(b,s)}{\int \mathrm{d}^2b \sum_{k=1}^{\infty} \mathcal{P}_k(b,s)}. \tag{135}$$

It is worth noting that this distribution, after integration over b, is much broader than Poissonian and has a long tail to high multiplicities.

Equation (135) is used as the basis of the multi-parton scattering generator for events in which the hard process is identical to the one used in the underlying event, *i.e.* QCD $2 \to 2$ scattering. For scatterings of more than one type of hard process, the formulae can be easily generalized, but in fact for the realistic case in which all other cross sections are small compared to the jet cross section, they saturate at a simple form,

$$\mathrm{P}_n(s) = \frac{n}{\sigma^{\mathrm{inc}}} \int \mathrm{d}^2b\, \mathcal{P}_n(b,s), \tag{136}$$

which allows for a more efficient generation of additional scatterings. It is worth noting that the fact that we have 'triggered on' a process with a small cross section leads to a bias in the b distribution and hence a higher multiplicity of additional scatters than in the pure QCD $2 \to 2$ scattering case. A slight further modification to the distribution is needed when the small cross section process is a subset of the large one, for example QCD $2 \to 2$ scattering restricted to the high p_T region.

As described so far, the n scatters are completely independent, which is expected to be a good approximation in the region in which multiple scattering dominates, *i.e.* small momentum fractions. However, some fraction of events come from higher x values and must lead to correlations between the scatters at some level. At the very least, the total momentum and flavour must be conserved: the total x value of all partons extracted from a hadron cannot exceed unity and each valence parton can only be extracted once. In Herwig++ these correlations are included in the simplest possible way, by vetoing any scatters that would take the total extracted energy above unity and by only evolving the first scatter back to a valence parton and all the others back to a gluon.

8.2 Connection to different simulation phases

The model introduced so far is entirely formulated at the parton level. However, an event generator aims for a full description of the event at the level of hadrons. This implies that the implementation of multi-parton scattering must be properly connected to the parton shower and hadronization models, a few details of which we discuss in the following.

8.2.1 Parton showers and hard matrix elements

After generating the hard process and invoking parton showers on its coloured particles, the number of additional scatters is calculated according to (135) or (136) respectively. After the initial-state shower has terminated, the incoming partons are extracted out of the beam particles in the usual way.

The requested additional scatters are then generated using a similar but completely independent infrastructure from the one of the hard process. Dedicated hard matrix elements with hand-coded formulae summed over parton spins are used for greater speed, as mentioned in Sect. 3.1. This also has the advantage that specific cuts, different to those used for the main hard process, can be specified.

For each additional scattering, parton showers evolve the produced particles down to the hadronic scale. The backward evolution of additional scatters is forced to terminate on a gluon. After termination, these gluons are extracted out of the beam particles. If this process leads to a violation of four-momentum conservation, the scattering cannot be established. It is therefore regenerated until the desired multiplicity has been reached. If a requested scattering can never be generated without leading to violation of momentum conservation, the program eventually gives up, reducing the multiplicity of scatters.

8.2.2 *Hadronization*

The underlying event and beam remnant treatment are closely connected because the generation of additional scatters requires the extraction of several partons out of the proton. As described before, all additional partons are extracted from the incoming beam particles. This is different from the procedure that was used in FORTRAN JIMMY, where the successive partons were always extracted from the previous beam remnant, a difference in the structure of the event record that should not lead to significant differences in physical distributions.

The cluster hadronization described in the previous section can only act on (anti)quarks or (anti)diquarks. However, naively extracting several partons from a hadron will not leave a flavour configuration that is amenable to such a description in general. Therefore, the strategy we use, as already mentioned, is to terminate the backward evolution of the hard process on a valence parton of the beam hadron and additional scatterings on gluons, giving a structure that can be easily iterated for an arbitrary number of scatters. This structure is essentially the same as in FORTRAN JIMMY.

8.3 Soft underlying event

While the new multiple interaction model provides a better description of the underlying event and is recommended for all realistic physics studies, Herwig++ still contains the original soft model of the underlying event used before version 2.1.

This model is based on the minimum-bias event generator of the UA5 Collaboration [67], which starts from a parameterization of the $p\bar{p}$ inelastic charged multiplicity distribution as a negative binomial distribution. In Herwig++ the relevant parameters are made available to the user for modification, the default values being the UA5 ones as used in the FORTRAN version of the program. These parameters are given in Table 11.

The first three parameters control the mean charged multiplicity $\bar{n}$ at c.m. energy $\sqrt{s}$ as indicated. The next two specify the parameter k in the negative binomial charged multiplicity distribution,

$$P(n) = \frac{\Gamma(n+k)}{n!\,\Gamma(k)} \frac{(\bar{n}/k)^n}{(1+\bar{n}/k)^{n+k}}.$$

Table 11 Parameters of the soft underlying event event model

Name	Description	Default
N1	a in $\bar{n} = a(s/\mathrm{GeV}^2)^b + c$	9.110
N2	b in $\bar{n} = a(s/\mathrm{GeV}^2)^b + c$	0.115
N3	c in $\bar{n} = a(s/\mathrm{GeV}^2)^b + c$	−9.500
K1	a in $1/k = a\ln(s/\mathrm{GeV}^2) + b$	0.029
K2	b in $1/k = a\ln(s/\mathrm{GeV}^2) + b$	−0.104
M1	a in $(M - m_1 - m_2 - a)e^{-bM}$	0.4 GeV
M2	b in $(M - m_1 - m_2 - a)e^{-bM}$	2.0 GeV^{-1}
P1	p_t slope for d, u	5.2 GeV^{-1}
P2	p_t slope for s, c	3.0 GeV^{-1}
P3	p_t slope for qq	5.2 GeV^{-1}

The parameters **M1** and **M2** describe the distribution of cluster masses M in the soft collision. These soft clusters are generated using a flat rapidity distribution with Gaussian shoulders. The transverse momentum distribution of soft clusters has the form

$$P(p_t) \propto p_t \exp(-b\sqrt{p_t^2 + M^2}),$$

where the slope parameter b depends as indicated on the flavour of the quark or diquark pair created when the cluster was produced. As an option, for underlying events, the value of $\sqrt{s}$ used to choose the multiplicity n may be increased by a factor **EnhanceCM** to allow for an enhanced underlying activity in hard events. The actual charged multiplicity is taken to be n plus the sum of the moduli of the charges of the colliding hadrons or clusters.

8.4 Code structure

In addition to being the main class responsible for the administration of the shower, the ShowerHandler, described in Sect. 6.9, is also responsible for the generation of the additional hard scattering processes. It has a reference to the MPIHandler set in the input files, which is used to actually create the additional scattering processes. It invokes the parton shower on all the available scatters and connects them properly to the incoming beam particles. This includes potential re-extraction of the incoming parton if it is changed as a result of initial-state radiation. It modifies the RemnantParticles that were initially created by the PartonExtractor. A number of classes are used by the ShowerHandler to generate the additional scattering processes.

MPIHandler The MPIHandler administers the calculation of the underlying event activity. It uses MPISampler to sample the phase space of the processes that are connected to it. Using that cross section the probabilities for the individual multiplicities of additional scatters are calculated during initialization. The method MPIHandler::multiplicity() samples a number of extra scatters from that pretabulated probability distribution. The method MPIHandler::generate() creates one subprocess according to the phase space and returns it.

MPISampler The MPISampler performs the phase-space sampling for the additional scatterings. It inherits from SamplerBase and implements the Auto Compensating Divide-and-Conquer phase space generator, ACDCGen.

HwRemDecayer The HwRemDecayer is responsible for decaying the RemnantParticles to something that can be processed by the cluster hadronization, *i.e.* (anti)quarks or (anti)diquarks. This includes the forced splittings to valence quarks and gluons respectively. Also the colour connections between the additional scatters and the remnants are set here.

ForcedSplitting The ForcedSplitting class calculates the kinematics of the forced splittings and creates the corresponding particles.

The most important interfaces to set parameters for the above mentioned classes are described here. An exhaustive description of all interfaces is provided by our Doxygen documentation.

MPIHandler

Cuts: Via a cuts object the minimal p_T of the additional scatters can be set. This is one of the two main parameters of the model. The current default, obtained from a fit to Tevatron data is 3.1 GeV. See Ref. [8] for details.

InvRadius: The inverse beam particle radius squared. The current default is $1.8\,\mathrm{GeV}^2$, obtained from the above mentioned fit.

Algorithm: A switch to enable efficient generation of additional scatters in rare (high-p_T) signal processes. Steers whether to use (135) or (136). The options are:

- 0: Underlying event process and signal process are identical.
- 1: Underlying event process and signal process are of the same type but the signal cross section is small. Here a veto algorithm has to be applied, if an additional scatter is produced with p_T larger than the cutoff for the hard process.
- 2: Underlying event process and signal process are distinct scattering types and the signal cross section is small. This is the default choice.

ShowerHandler

MPI: Switch to turn multiple parton interactions on or off. The default is `Yes`.

HwRemDecayer

ForcedSplitter: A reference to the object that calculates the forced splittings. If this reference is set to `NULL` the forced splittings after the initial-state parton showers are switched off. This can be useful if a different hadronization model should be used. However this model should then take care of the remnants and has to set the colour connections to the additional scatters.

Since it is not a recommended option, we do not go into as much detail, but for completeness, we briefly mention the structure of the UA5 code. The UA5 model is implemented in the UA5Handler class, which is intended to be used as an **UnderlyingEventHandler** in the ClusterHadronizationHandler. The main interfaces of the UA5Handler are the parameters named in Table 11, described in Sect. 8.3.

9 Hadron Decays

Herwig++ uses a sophisticated model of hadronic decays, as described in Refs. [22, 70]. The simulation of decays in Herwig++ is designed to have the following properties:

- a unified treatment of the decays of both the fundamental particles and the unstable hadrons, this is of particular importance for particles like the τ lepton, which, while a fundamental particle, is more akin to the unstable hadrons in the way it decays;
- up-to-date particle properties, *i.e.* masses, widths, lifetimes, decay modes and branching ratios together with a new database to store these properties to make updating the properties easier and the choices made in deriving them clearer;
- full treatment of spin correlation effects using the algorithm of Refs. [24–27] for the decay of all unstable particles, it is important that the same algorithm is used consistently in all stages of the program so that correlations between the different stages can be correctly included;
- a sophisticated treatment of off-shell effects for both the unstable hadrons and fundamental particles;
- a large range of matrix elements for hadron and tau decays including both general matrix elements based on the spin structures of the decays and specific matrix elements for important decay modes;
- the accurate simulation of QED radiation in the particle decays using the Yennie-Frautschi-Suura (YFS) formalism.

In this section we describe the simulation of hadron and tau decays in Herwig++. We start by discussing the physical properties of the hadrons used in the simulation and how they are determined. In ThePEG framework these physical properties are stored using the ParticleData class, which has one instance for each particle used in the simulation. In turn the properties of a given decay mode are stored using the DecayMode class, which contains both the particles involved in the decay and a reference to a Decayer object that can be used to generate the kinematics of the decay products. The DecayHandler class then uses these DecayMode objects to select a decay of a given particle, according to the probabilities given by the branching ratios for the different decay modes, and then generates the kinematics using the relevant Decayer specified by the DecayMode.

Following a brief discussion of the treatment of off-shell effects we therefore discuss the different Decayer classes available in Herwig++ for the decay of tau leptons, strong and electromagnetic meson decays and then weak meson[25] decays. This is followed by a discussion of the code structure.

9.1 Particle properties

The information in the Particle Data Group's (PDG) compilation [34] of experimental data is sufficient in many cases to determine the properties of the hadrons used in Herwig++. However, there are some particles for which the data are incomplete or too inaccurate to be used. Equally, there are a number of particles that are necessary for the simulation but have not been observed, particularly excited bottom and charm hadrons, which should perhaps be regarded as part of the hadronization model affecting the momentum spectrum of lighter states, rather than as physical states. A large number of choices therefore have to be made in constructing the particle data tables used in the event generator based on the data in Ref. [34].

In the past the data were stored as either a text file or the contents of a FORTRAN COMMON block. However, due to the relatively large amount of data that needs to be stored we decided to adopt a database approach based on the MySQL database system. The event generation still uses text files to read in the particle properties but these files are now automatically generated from the database. This provides us with a range of benefits: the data can now be edited using a web interface; additional information describing how the particle properties were determined is stored in the database both improving the long-term maintenance and allowing the user to understand the uncertainties and assumptions involved.

[25] We currently rely on a phase-space distribution for the decay of the baryons and essentially the same decay model as was used in FORTRAN HERWIG. This will be improved in the next major release.

An example of the output from the database for the properties of the ω meson is shown in Fig. 5. This includes the basic properties of the particle together with an explanation of how they were derived. In addition there is a star rating between one and five, which gives an indication of how reliable the properties of the particle and the modelling of individual decay modes are.

In general we used the following philosophy to determine the particle properties used in Herwig++:

- The properties of the light mesons in the lowest lying multiplets were taken from Ref. [34]. In some cases we used either lepton universality or the phase-space factors from our Decayers to average the branching ratios for poorly measured modes.
- Where possible the properties of the excited light mesons were taken from Ref. [34] together with some additional interpretation of the data. Except for the 1^3D_1 multiplet, which is missing a ϕ-like member, the mesons needed to fill the 1^1S_0, 1^3S_1, 1^1P_1, 1^3P_0, 1^3P_1, 1^3P_2, 1^1D_2, 1^3D_1, 1^3D_3, 2^1S_0, 1^1S_0 and 2^3S_1 SU(3) multiplets are included together with the K mesons from the 1^3D_2 multiplet.
- The properties of the $D_{u,d,s}$ mesons were taken from Ref. [34] together with the addition of some high multiplicity modes to ensure that the branching ratios sum to one.
- The branching ratios and properties for $B_{u,d,s}$ mesons were taken from the data tables of EvtGEN [71], which have been extensively tuned to B-factory data. This means that partonic decay modes are used to model many of the inclusive B decay modes.
- The mass of the B_c meson is taken from Ref. [34]. The branching ratios were taken from the theoretical calculations of Ref. [72] together with some partonic modes to ensure that the branching ratios sum to one.
- The properties and decay modes of the charmonium resonances were taken from Ref. [34] where possible together with the use of partonic decays, to ggg, gg or $q\bar{q}$, to model the unobserved inclusive modes. For some of the particles, in particular the h_c and $\eta_c(2S)$, the results of Ref. [73] were used and the $\eta_c(2S)$ branching ratios were taken from the theoretical calculation of Ref. [74].
- The properties and decay modes of the bottomonium resonances were taken from Ref. [34] where possible. In addition we have added a large number of states that are expected to have small widths, *i.e.* the mass is expected to be below the $B\bar{B}$ threshold, using the theoretical calculations of Refs. [75–79] for many of the properties.
- The properties of the excited D and D_s mesons were taken from Ref. [34] including recent results for the D_1' and D_0^* states. The widths of the D_{s1} and D_{s2} mesons were from the theoretical calculations of Ref. [80] and Ref. [81], respectively. For many of the mesons we were forced to assume that the observed modes saturated the

ω *****

The ω is the lightest isospin singlet from the 1^3S_1 multiplet. The modes and properties are taken from Ref. [34] with the lepton modes averaged. The modes with photons that can be produced by QED radiation are included in the mode without the radiation. The ω is allowed to be off-shell by ten times the width. The ω has mass 782.65 MeV and is unstable. The ω has spin 1, charge 0 and is colour neutral. The ω is a meson and is from the 1^3S_1 multiplet. The ω has width 8.49 MeV. The lower limit on the mass of the particle is 84.9 MeV and the upper limit is 84.9 MeV. These are the deviations from the on-shell value. The branching ratios are fixed. The PDG code is 223. The mass generator is omegamass for the ω. The width generator is omegawidth for the ω.

Branching Ratio	Rating	On/ Off	Outgoing Particles	Description	Decayer
0.891174	*****	on	π^+, π^-, π^0	The decay of the ω to three pions. The branching ratio is taken from [34] with a small, order 10^{-4} addition, to ensure the modes sum to one.	Vector3Pion
0.090250	*****	on	π^0, γ	The decay of the ω to a pion and photon. The branching ratio is taken from [34] with the other neutrals mode added here.	VectorVP
0.017000	*****	on	π^+, π^-	The isospin violating decay of the ω to two pions with branching ratio taken from [34].	Vector2Meson
0.000797	****	on	π^0, e^-, e^+	The decay of the ω to a pion and an electron-positron pair. The branching ratio is calculated by averaging the measured electron and muon branching ratios from [34] using the decayer to give the relative rates. This value is within the experimental error from [34].	VectorVPff
0.000490	****	on	η, γ	The decay of ω to η and a pion with branching ratio taken from [34].	VectorVP
0.000078	****	on	π^0, μ^-, μ^+	The decay of the ω to a pion and a $\mu^+\mu^-$ pair. The branching ratio is calculated by averaging the measured electron and muon branching ratios from [34] using the decayer to give the relative rates. This value is within the experimental error from [34].	VectorVPff
0.000072	****	on	e^-, e^+	The decay of the ω to an electron-positron pair. The branching ratio is obtained by averaged the electron and muon channel from [34] including a kinematic factor from the decayer.	Vector2Leptons
0.000072	****	on	μ^-, μ^+	The decay of the ω to a muon-antimuon pair. The branching ratio is obtained by averaged the electron and muon channel from [34] including a kinematic factor from the decayer.	Vector2Leptons
0.000067	***	on	π^0, π^0, γ	The decay of ω to two pions and a photon, with branching ratio taken from [34].	DecayME0

Fig. 5 An example of the particle properties in Herwig++, in this case for the ω meson. The properties of the particle including the mass, width, decay modes and branching ratios are given together with comments on how properties were determined. In the full web version links are included to the descriptions of the objects responsible for generating the kinematics for the various decay modes

total width in order to obtain the branching ratios using the results in Ref. [34].

– The properties of the excited $B_{u,d,s}$ mesons are uncertain. The $B^*_{u,d,s}$ have been observed and there is evidence in Ref. [34] from LEP for further excited states, however it was unclear which states have been observed. There have been recent claims for the observation of the B_1, B_2^* and B_{s2}^* states by CDF and D0 [82, 83] and the B_{s1} by CDF. The situation is still unclear, the masses measured by the two experiments disagree for the B_1, B_2^* states and D0 do not observe the B_{s1} state. We have chosen to use the D0 results for the B system and the CDF results for the B_s

system for the observed states and have taken the properties of the remaining unobserved states from Ref. [81].
– The masses of the excited B_c mesons, which have not been observed, are taken from the lattice results in Ref. [84], which agree with potential model calculations. The widths and branching ratios were taken from the theoretical calculation of Ref. [85].

All the particle properties used in Herwig++ can be accessed via the online interface to our database of particle properties at http://www.ippp.dur.ac.uk/~richardn/particles/.

9.2 Line shapes

In general, if we wish to include the off-shell effects for an outgoing external particle in a hard production or decay process we need to include the following factor in the calculation of the matrix element

$$W_{\text{off}} = \frac{1}{\pi}\int \mathrm{d}m^2 \frac{m\Gamma(m)}{(M^2-m^2)^2+m^2\Gamma^2(m)}, \tag{137}$$

where M is the physical mass of the particle, m is the off-shell mass and $\Gamma(m)$ is the running width evaluated at scale m. In practice other effects can be included to improve this simple formula, for example we include the Flatté lineshape [86] for the $a_0(980)$ and $f_0(980)$ mesons. In Herwig++ we calculate the running width of the particle based on its decay modes. The Decayer responsible for each decay mode specifies the form of the running partial width, $\Gamma_i(m)$, for the decay mode either in a closed analytic form for two-body decays or as a WidthCalculatorBase object, which is capable of calculating the partial width numerically and is used to construct an interpolation table. The running width for a given particle is then the sum of the partial widths

$$\Gamma(m) = \sum_i \Gamma_i(m). \tag{138}$$

This gives both a sophisticated model of the running width based on the decay modes and allows us to use the partial widths to normalize the weights for the phase-space integration of the decays to improve efficiency close to the kinematic threshold for the decay.

In some cases, where the partial width varies significantly over the mass range allowed in the simulation, we can choose to use a variable branching ratio

$$\mathrm{BR}_i(m) = \frac{\Gamma_i(m)}{\Gamma(m)} \tag{139}$$

both to prevent the production of kinematically unavailable modes and to improve the physics of the simulation. The classic examples are the decays of the f_0 and a_0 scalar mesons, which lie close to the $K\bar{K}$ threshold. This means that, depending on their mass, they decay to either $\pi\pi$ or $\eta\pi$ respectively below the threshold or with a significant $K\bar{K}$ branching fraction above the $K\bar{K}$ threshold.

The weight in (137) is automatically included for all the Decayers inheriting from the DecayIntegrator class, which is the case for vast majority of the Herwig++ Decayers. The GenericWidthGenerator calculates the running widths using information from the Herwig++ Decayers inheriting from the DecayIntegrator class. The GenericMassGenerator is responsible for calculating the weight in (137) or generating a mass according to this distribution.

9.3 Tau decays

The simulation of τ lepton decays in Herwig++ is described in detail in Ref. [22], together with a detailed comparison between the results of Herwig++ and TAUOLA [87, 88]. Here we simply describe the basic formalism for the decays of the tau and the different models available for the different decays, together with the structure of the code.

The matrix element for the decay of the τ lepton can be written as

$$\mathcal{M} = \frac{G_F}{\sqrt{2}} L_\mu J^\mu, \quad L_\mu = \bar{u}(p_{\nu_\tau})\gamma_\mu(1-\gamma_5)u(p_\tau), \tag{140}$$

where p_τ is the momentum of the τ and p_{ν_τ} is the momentum of the neutrino produced in the decay. The information on the decay products of the virtual W boson is contained in the hadronic current, J^μ. This factorization allows us to implement the leptonic current L_μ for the decaying tau and the hadronic current separately and then combine them to calculate the τ decay matrix element.

In Herwig++ this factorization is used to have a TauDecayer class, which implements the calculation of the leptonic current for the τ decay and uses a class inheriting from WeakDecayCurrent to calculate the hadronic current for a given decay mode. This factorization allows us to reuse the hadronic currents in other applications, for example in weak meson decay using the naïve factorization approximation or in the decay of the lightest chargino to the lightest neutralino in Anomaly Mediated SUSY Breaking (AMSB) models where there is a small mass difference between the neutralino and chargino.

9.3.1 Hadronic currents

We have implemented a number of hadronic currents, which are mainly used for the simulation of τ decays. These are all based on the WeakDecayCurrent class. In this section we list the available hadronic currents together with a brief description, a more detailed description can be found in either Ref. [22] or the Doxygen documentation.

ScalarMesonCurrent The simplest hadronic current is that for the production of a pseudoscalar meson, *e.g.* the current for the production of $\pi^{\pm}$ in the decay of the tau. The hadronic current can be written as

$$J^{\mu} = f_P p_P^{\mu}, \tag{141}$$

where p_P^{μ} is the momentum of the pseudoscalar meson and f_P is the pseudoscalar meson decay constant.

VectorMesonCurrent The current for the production of a vector meson is given by

$$J^{\mu} = \sqrt{2} g_V \epsilon_V^{*\mu}, \tag{142}$$

where $\epsilon_V^{*\mu}$ is the polarization vector for the outgoing meson and g_V is the decay constant of the vector meson.

LeptonNeutrinoCurrent The current for weak decay to a lepton and the associated anti-neutrino is given by

$$J^{\mu} = \bar{u}(p_{\ell})\gamma^{\mu}(1-\gamma_5)v(p_{\bar{\nu}}), \tag{143}$$

where $p_{\bar{\nu}}$ is the momentum of the anti-neutrino and p_{ℓ} is the momentum of the charged lepton.

TwoMesonRhoKStarCurrent The weak current for production of two mesons via the ρ or K^* resonances has the form

$$J^{\mu} = (p_1 - p_2)_{\nu}\left(g^{\mu\nu} - \frac{q^{\mu}q^{\nu}}{q^2}\right)\frac{1}{\sum_k \alpha_k}\sum_k \alpha_k \mathrm{BW}_k(q^2), \tag{144}$$

where $p_{1,2}$ are the momenta of the outgoing mesons, $q = p_1 + p_2$, $\mathrm{BW}_k(q^2)$ is the Breit-Wigner distribution for the intermediate vector meson k and α_k is the weight for the resonance, which can be complex. The Breit-Wigner terms are summed over the ρ or K^* resonances that can contribute to a given decay mode.

The models of either Kühn and Santamaria [89], which uses a Breit-Wigner distribution with a p-wave running width, or Gounaris and Sakurai [90] are supported for the shape of the Breit-Wigner distribution.

KPiCurrent Unlike the $\pi^+\pi^0$ decay of the tau the $K\pi$ decay mode can occur via either intermediate scalar or vector mesons. We therefore include a model for the current for the $K\pi$ decay mode including the contribution of both vector and scalar resonances based on the model of Ref. [91]. The current is given by

$$J^{\mu} = c_V(p_1 - p_2)_{\nu}\frac{1}{\sum_k \alpha_k}\sum_k \alpha_k \mathrm{BW}_k(q^2)\left(g^{\nu\mu} - \frac{q^{\nu}q^{\mu}}{M_k^2}\right) + c_S q^{\mu}\frac{1}{\sum_k \beta_k}\sum_k \beta_k \mathrm{BW}_k(q^2), \tag{145}$$

where $p_{1,2}$ are the momenta of the outgoing mesons, $q = p_1 + p_2$, $\mathrm{BW}_k(q^2)$ is the Breit-Wigner distribution for the intermediate mesons and α_k is the weight for the resonance. The sum over the resonances is over the vector K^* states in the first, vector, part of the current and the excited scalar K^* resonances in the second, scalar, part of the current. By default the vector part of the current includes the $K^*(892)$ and $K^*(1410)$ states and the scalar part of the current includes the $K_0^*(1430)$ together with the option of including the $\kappa(800)$ to model any low-mass enhancement in the mass of the $K\pi$ system, although additional resonances can be included if necessary.

ThreeMesonCurrentBase In order to simplify the implementation of a number of standard currents for the production of three pseudoscalar mesons we define the current in terms of several form factors. The current is defined to be [87]

$$J^{\mu} = \left(g^{\mu\nu} - \frac{q^{\mu}q^{\nu}}{q^2}\right) \times [F_1(p_2 - p_3)^{\mu} + F_2(p_3 - p_1)^{\mu} + F_3(p_1 - p_2)^{\mu}] + q^{\mu}F_4 + iF_5\epsilon^{\mu\alpha\beta\gamma}p_1^{\alpha}p_2^{\beta}p_3^{\gamma}, \tag{146}$$

where $p_{1,2,3}$ are the momenta of the mesons in the order given below and $F_{1\to5}$ are the form factors. We use this approach for a number of three-meson modes that occur in τ decays: $\pi^-\pi^-\pi^+$; $\pi^0\pi^0\pi^-$; $K^-\pi^-K^+$; $K^0\pi^-\bar{K}^0$; $K^-\pi^0K^0$; $\pi^0\pi^0K^-$; $K^-\pi^-\pi^+$; $\pi^-\bar{K}^0\pi^0$; $\pi^-\pi^0\eta$; $K_S^0\pi^-K_S^0$; $K_L^0\pi^-K_L^0$; $K_S^0\pi^-K_L^0$. The current is implemented in terms of these form factors in a base class so that any model for these currents can be implemented by inheriting from this class and specifying the form factors.

We currently implement three models for these decays, the ThreeMesonDefaultCurrent model of Refs. [87, 89, 92], which treats all the decay modes, the ThreePionCLEOCurrent model of CLEO [93] for the three-pion modes and the KaonThreeMesonCurrent model of Ref. [94] for the kaon modes.

ThreeMesonDefaultCurrent This is the implementation of the model of Refs. [87, 89, 92], which uses the form of Ref. [89] for the a_1 width. The form factors for the different modes are given in Refs. [87, 92].

ThreePionCLEOCurrent This is the implementation of the model of Ref. [93] for the weak current for three pions. This model includes ρ mesons in both the s- and p-wave, the scalar σ resonance, the tensor f_2 resonance and scalar $f_0(1370)$. The form factors for the $\pi^0\pi^0\pi^-$ mode are given in Ref. [93] and the others can be obtained by isospin rotation.

KaonThreeMesonCurrent Like the model of Ref. [92] the model of Ref. [94] is designed to reproduce the correct chiral limit for tau decays to three mesons. However, this model makes a different choice of the resonances to use away from this limit for the decays involving at least one kaon and in the treatment of the K_1 resonances. The form factors for the different modes are given in Ref. [94].

TwoPionPhotonCurrent The branching ratio for the decay $\tau^- \to \omega\pi^-\nu_\tau$ is 1.95% [34]. The majority of this decay is modelled as an intermediate state in the four-pion current described below. However there is an 8.90% [34] branching ratio of the ω into $\pi^0\gamma$, which must also be modelled. We do this using a current for $\pi^\pm\pi^0\gamma$ via an intermediate ω. The hadronic current for this mode, together with the masses, widths and other parameters, are taken from Ref. [87].

FourPionNovosibirskCurrent We use the model of Ref. [95][26] to model the decay of the τ to four pions. The model is based on a fit to e^+e^- data from Novosibirsk.

FivePionCurrent We use the model of Ref. [96], which includes $\rho\omega$ and $\rho\sigma$ intermediate states, via the a_1 meson to model the five-pion decay modes of the τ.

9.4 Strong and electromagnetic hadron decays

The vast majority of the strong and electromagnetic decays in Herwig++ are simulated using a few simple models based on the spin structure of the decay. These simple models are supplemented with a small number of specialized models, usually from experimental fits, for specific decay modes. In this section we describe the different models we use for these decays for the scalar, vector and tensor mesons. All of these are implemented in Decayer classes that inherit from the DecayIntegrator class of Herwig++.

For a number of the decays of bottomonium and charmonium resonances we use inclusive electromagnetic and strong decays to $q\bar{q}$, gg, ggg and $gg\gamma$, which are described in a separate section.

A number of decays are still performed using a phase-space distribution generated using the Hw64Decayer, which implements the same models as were available in the FORTRAN HERWIG program. In addition we use the MAMBO algorithm, [97], implemented in the MamboDecayer class, to generate the momenta of the decay products according to a phase-space distribution for a number of high-multiplicity modes.

9.4.1 Scalar mesons

While the majority of the scalar meson decays are performed using general Decayers based on the spin structures there are a number of models implemented for the rare radiative decays of the light pseudoscalar mesons, three-body decays of the η and η', and the decay $\pi^0 \to e^+e^-e^+e^-$.

EtaPiGammaGammaDecayer We use the Vector-Meson Dominance (VMD)-based model of Ref. [98] for the decays $\eta, \eta' \to \pi^0\gamma\gamma$. In practice we use a running width for the ρ to include the η' decay as well as the η decay and take the parameters from Ref. [98].

EtaPiPiGammaDecayer We use either a VMD type model or a model using either the theoretical or experimental form of the Omnes function[27] taken from Refs. [98, 99] for the decay of the η or η' to $\pi^+\pi^-\gamma$.

EtaPiPiPiDecayer The decay of a pseudoscalar meson, for example the η or η', to two charged and one neutral or three neutral pions, or of the η' to two pions and the η, is performed using a parameterization of the matrix element squared taken from Ref. [100]. The experimental results of Refs. [101] and [102] are used for the $\eta \to \pi^+\pi^-\pi^0$ and $\eta \to \pi^0\pi^0\pi^0$ decays respectively. The theoretical values from Ref. [100] are used for the other decays.

PScalar4FermionsDecayer As the π^0 is so copiously produced it is one of the small number of particles for which we include branching ratios below the level of 10^{-4}. The matrix element for the sub-leading decay $\pi^0 \to e^+e^-e^+e^-$ is taken to be the combination of the standard matrix element for $\pi^0 \to \gamma\gamma$ and the branching of the photons into e^+e^-.

PScalarPScalarVectorDecayer This matrix element is used to simulate the decay of the 2S pseudoscalar mesons to a vector meson and a 1S pseudoscalar meson. It is also used for the decay of some scalar mesons to vector mesons and another scalar meson, which has the same spin structure. The matrix element has the form

$$\mathcal{M} = g\epsilon_2^\mu (p_0 + p_1)_\mu, \tag{147}$$

where ϵ_2 is the polarization vector of the vector meson, p_0 is the momentum of the decaying particle, p_1 is the momentum of the outgoing pseudoscalar meson and g is the coupling for the decay.

[26] It should be noted that there were a number of mistakes in this paper, which were corrected in Ref. [88].

[27] Our default choice is to use the experimental form of the Omnes function.

PScalarVectorFermionsDecayer There are a number of decays of a pseudoscalar meson to either a vector meson or the photon and a lepton-antilepton pair. The classic example is the Dalitz decay of the neutral pion, $\pi^0 \to \gamma e^+ e^-$. We take the propagator of the off-shell photon to be $\frac{1}{m^2_{f\bar{f}}}$, where $m_{f\bar{f}}$ is the mass of the fermion-antifermion pair. The option of including a vector meson dominance form factor is included.

PScalarVectorVectorDecayer In practice the vast majority of the decays of pseudoscalar mesons to two spin-1 particles are of the form $P \to \gamma\gamma$ for which, because the photon is stable, it is not as important to have a good description of the matrix element. There are however some decays, *e.g.* $\eta' \to \omega\gamma$, for which this matrix element is needed.

The matrix element is taken to be

$$\mathcal{M} = g\epsilon^{\mu\nu\alpha\beta} p_{1\mu}\epsilon_{1\nu} p_{2\alpha}\epsilon_{2\beta}, \tag{148}$$

where $p_{1,2}$ and $\epsilon_{1,2}$ are the momenta and polarization vectors of the outgoing vector particles and g is the coupling for the decay.

ScalarMesonTensorScalarDecayer There are a limited number of decays of a (pseudo)scalar meson to a tensor meson and another (pseudo)scalar meson. The matrix element takes the form

$$\mathcal{M} = g\epsilon^{\alpha\beta} p_{0\alpha} p_{2\beta}, \tag{149}$$

where $\epsilon^{\alpha\beta}$ is the polarization tensor of the outgoing tensor meson, p_0 is the momentum of the decaying particle, p_2 is the momentum of the outgoing (pseudo)scalar meson and g is the coupling for the decay.

ScalarScalarScalarDecayer The decay of a scalar meson to two scalar mesons has no spin structure and we assume that the matrix element is simply constant, *i.e.*

$$\mathcal{M} = g. \tag{150}$$

We still include a matrix element for this decay in order to simulate both the off-shell effects in the decay and to give the correct partial width to be used in the running width calculation for the incoming particle.

ScalarVectorVectorDecayer A number of the scalar mesons decay to two vector mesons. The matrix element is taken to have the form

$$\mathcal{M} = g[p_1 \cdot p_2 \epsilon_1 \cdot \epsilon_2 - p_1 \cdot \epsilon_2 p_2 \cdot \epsilon_1], \tag{151}$$

where $\epsilon_{1,2}$ are the polarization vectors of the outgoing vector particles and $p_{1,2}$ are their momenta.

9.4.2 Vector mesons

With the exception of the three-pion decay modes of the ω, ϕ and a_1 mesons, and the two-pion decays of onium resonances, we use general Decayers based on the spin structure for all the strong and electromagnetic vector and pseudovector meson decays.

a1SimpleDecayer This class implements the model of Ref. [89] for the decay of the a_1 meson to three pions and only includes the lightest two ρ meson multiplets in the modelling of the decay.

a1ThreePionCLEODecayer This class implements the model of CLEO [93] for a_1 decay to three pions, which is a fit to CLEO data on $\tau^- \to \pi^0\pi^0\pi^-\nu_\tau$. The model includes the coupling of the a_1 to the ρ, $\rho(1450)$, $f_0(1370)$, $f_2(1270)$ and σ mesons.

a1ThreePionDecayer This class implements a model of a_1 decay to three pions based on the modelling of the a_1 used in the 4π currents for tau decays presented in Ref. [95] and includes the ρ and σ resonances.

OniumToOniumPiPiDecayer The decay of onium resonances to lighter states and a pion pair, $\mathcal{O}' \to \mathcal{O}\pi\pi$, uses the matrix element [103]

$$\begin{aligned}\mathcal{M} = {} & \epsilon' \cdot \epsilon[\mathcal{A}(q^2 - 2m_\pi^2) + \mathcal{B}E_1E_2] \\ & + \mathcal{C}((\epsilon' \cdot q_1)(\epsilon \cdot q_2) + (\epsilon' \cdot q_2)(\epsilon \cdot q_1)),\end{aligned} \tag{152}$$

where ϵ' is the polarization vector of the decaying onium resonance, ϵ is the polarization vector of the outgoing onium resonance, $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ are complex couplings, m_π is the pion mass, $E_{1,2}$ are the pion energies, $q_{1,2}$ are the pion momenta and q is the momentum of the $\pi\pi$ system.

The results of BES [104] are used for $\psi' \to J/\psi$ and CLEO [105] for $\Upsilon(3S)$ and $\Upsilon(2S)$ decays. The remaining parameters are chosen to approximately reproduce the distributions from BaBar [106] and CLEO [107] for $\Upsilon(4S)$ and $\psi(3770)$ decays respectively.

PVectorMesonVectorPScalarDecayer The matrix element for the decay of a pseudovector meson to a spin-1 particle, either a vector meson or a photon, and a pseudoscalar meson is taken to be

$$\mathcal{M} = g\epsilon_\mu[p_V \cdot p_0 \epsilon_V^\mu - p_V^\mu \epsilon_V \cdot p_0], \tag{153}$$

where ϵ_V is the polarization vector of the outgoing vector meson, p_V is the momentum of the outgoing vector meson, ϵ is the polarization vector of the decaying pseudovector and p_0 is the momentum of the decaying particle.

VectorMeson2FermionDecayer Most of the decays of the vector mesons to a fermion-antifermion pair are the decays of the light vector mesons to electron and muon pairs, and of the bottomonium and charmonium resonances to all the charged leptons. In addition we use this matrix element for some baryonic charmonium decays.

The matrix element is taken to have the form

$$\mathcal{M} = g\epsilon_\mu \bar{u}(p_f)\gamma^\mu v(p_{\bar{f}}), \tag{154}$$

where g is the coupling for the decay, p_f is the four-momentum of the outgoing fermion, $p_{\bar{f}}$ is the four-momentum of the outgoing antifermion and ϵ is the polarization vector of the decaying particle.

VectorMeson2MesonDecayer The matrix element for the decay of a vector meson to two scalar (or pseudoscalar) mesons is given by

$$\mathcal{M} = g_{VPP}\epsilon \cdot (p_1 - p_2), \tag{155}$$

where g_{VPP} is a dimensionless coupling, ϵ is the polarization vector of the decaying particle and $p_{1,2}$ are the momenta of the outgoing scalars.

VectorMeson3PionDecayer Both the lowest-lying isospin-zero vector mesons, ω and ϕ, have large branching ratios for the decay into three pions. For these mesons the decay is assumed to be dominated by the production of the lowest lying ρ multiplet. Our default model for the matrix element for this decay is

$$\mathcal{M} = g\epsilon^{\mu\alpha\beta\nu}\epsilon_\mu p_{1\alpha} p_{2\beta} p_{3\nu} \times \left[d + \sum_i f_i[\mathrm{BW}_i(s_{12}) + \mathrm{BW}_i(s_{13}) + \mathrm{BW}_i(s_{23})]\right], \tag{156}$$

where $p_{1,2,3}$ are the momenta of the outgoing particles, $s_{ij} = (p_i + p_j)^2$, g is the overall coupling for the decay, d is a complex coupling for the direct interaction, f_i is the coupling of the ith ρ multiplet and $\mathrm{BW}_i(s)$ is a Breit-Wigner distribution with a p-wave running width. This is an extension of the model used by KLOE [108] to include higher ρ multiplets.

VectorMesonPScalarFermionsDecayer The decay of a vector meson to a pseudoscalar meson and a fermion-antifermion pair is simulated using a matrix element based on that for the $V \to VP$ vertex combined with the branching of the vector, which is in reality always a photon, into a fermion-antifermion pair.

VectorMesonPVectorPScalarDecayer There are a number of decays of both the charmonium resonances and light vector mesons from the higher multiplets to pseudovector mesons. The matrix element for the decay is

$$\mathcal{M} = g[p_A \cdot p_0 \epsilon_A \cdot \epsilon - p_A \cdot \epsilon \epsilon_A \cdot p_0], \tag{157}$$

where ϵ_A is the polarization vector of the outgoing pseudovector meson, p_A is its momentum, ϵ is the polarization vector of the decaying particle and p_0 is its momentum.

VectorMesonVectorPScalarDecayer The decay of a vector meson to another spin-1 particle and a pseudoscalar meson is common in both the radiative decay of the 1S vector mesons and the decay of higher vector multiplets to the 1S vector mesons. The matrix element for the decay is

$$\mathcal{M} = g\epsilon^{\mu\nu\alpha\beta}\epsilon_{0\mu} p_{0\nu} p_{1\alpha}\epsilon_{1\beta}, \tag{158}$$

where p_0 is the momentum of the decaying particle, p_1 is the momentum of the outgoing vector particle, ϵ_0 is the polarization vector of the incoming meson and ϵ_1 is the polarization vector of the outgoing vector particle.

VectorMesonVectorScalarDecayer We include a number of decays of the vector mesons to a scalar meson and either the photon or another vector meson. In practice the vast majority of these decays are radiative decays involving scalar mesons. The remaining decays use the σ meson as a model for four-pion decays of the excited ρ multiplets.

The matrix element for the decay is

$$\mathcal{M} = g\epsilon_\mu [p_V \cdot p_0 \epsilon_V^\mu - p_V^\mu \epsilon_V \cdot p_0], \tag{159}$$

where g is the coupling for the decay, ϵ is the polarization vector of the decaying vector meson, ϵ_V is the polarization vector of the outgoing vector meson, p_0 is the momentum of the decaying particle and p_V is the momentum of the outgoing vector meson.

VectorMesonVectorVectorDecayer There are a small number of decays of excited ρ multiplets to ρ mesons included in the simulation. We model these decays using the matrix element

$$\mathcal{M} = \frac{g}{M_0^2}(p_{0\nu}\epsilon^\alpha - p_{0\alpha}\epsilon^\nu) \times [(p_{1\nu}\epsilon_1^\beta - p_1^\beta \epsilon_{1\nu})(p_{2\alpha}\epsilon_{2\beta} - p_{2\beta}\epsilon_{2\alpha}) - (\nu \leftrightarrow \alpha)], \tag{160}$$

where g is the coupling for the decay, $\epsilon_{1,2}$ are the polarization vectors of the outgoing mesons, $p_{1,2}$ are the momenta of the outgoing mesons, ϵ is the momentum of the decaying particle and p_0 is its momentum.

9.4.3 Tensor mesons

Only a relatively small number of tensor meson states are included in the simulation, compared to the vector and scalar mesons. All their decays are simulated using a small number of matrix elements based on the spin structure of the decays. Many of the multi-body decays of the tensor mesons are simulated using these two-body matrix elements with off-shell vector and scalar mesons.

TensorMeson2PScalarDecayer The matrix element for the decay of a tensor meson to two pseudoscalar (or scalar) mesons is

$$\mathcal{M} = g\epsilon^{\mu\nu} p_{1\mu} p_{2\mu}, \tag{161}$$

where g is the coupling for the decay, $p_{1,2}$ are the momenta of the decay products and $\epsilon^{\mu\nu}$ is the polarization tensor for the decaying meson.

TensorMesonVectorPScalarDecayer There are a number of decays of tensor mesons to a spin-1 particle, either a vector meson or the photon, and a pseudoscalar meson, examples include $a_2 \to \rho\pi$ and $a_2 \to \pi\gamma$. The matrix element is taken to be

$$\mathcal{M} = \epsilon^{\mu\nu} p_{P\mu} \epsilon_{\nu\alpha\beta\gamma} p_V^{\alpha} \epsilon_V^{\beta} p_P^{\gamma}, \tag{162}$$

where g is the coupling for the decay, p_P is the momentum of the pseudoscalar meson, p_V is the momentum of the vector, ϵ_V is the polarization vector of the outgoing vector meson and $\epsilon^{\mu\nu}$ is the polarization tensor for the decaying meson.

TensorMesonVectorVectorDecayer We have based our matrix element for the decay of a tensor meson to two vector mesons on the perturbative graviton decay matrix element [109] in such a way that it vanishes if the polarizations of the outgoing vectors are replaced with their momenta. The matrix element is

$$\mathcal{M} = g\Big[\epsilon_{\mu\nu}\{(\epsilon_{1\alpha} p_1^{\mu} - \epsilon_1^{\mu} p_{1\alpha})(\epsilon_2^{\alpha} p_2^{\nu} - \epsilon_2^{\nu} p_2^{\alpha}) + (\mu \leftrightarrow \nu)\} - \frac{1}{2}\epsilon_{\mu}^{\mu}(\epsilon_{1\alpha} p_{1\beta} - \epsilon_{1\beta} p_{1\alpha})(\epsilon_2^{\alpha} p_2^{\beta} - \epsilon_2^{\beta} p_2^{\alpha})\Big], \tag{163}$$

where g is the coupling for the decay, $\epsilon_{1,2}$ are the polarization vectors for the outgoing vector mesons and $\epsilon^{\mu\nu}$ is the polarization tensor for the decaying meson. In practice this matrix element is mainly used with off-shell vector mesons to model three- and four-body decays of the tensor mesons.

9.4.4 Inclusive strong and electromagnetic decays

For a number of bottomonium and charmonium resonances we make use of partonic decays of the mesons to model the unobserved inclusive modes needed to saturate the branching ratios. These decays are modelled using the QuarkoniumDecayer class, which implements the decay of the onium resonances to $q\bar{q}$ and gg according to a phase-space distribution, and the decay to ggg and $gg\gamma$ according to the Ore-Powell matrix element [110]. The QuarkoniumDecayer class inherits from the PartonicDecayerBase, which uses the cluster model to hadronize the resulting partonic final state with a veto to ensure that there is no double counting with the exclusive modes.

9.5 Weak hadronic decays

There are five classes of weak mesonic decays currently included in the simulation:

1. weak exclusive semi-leptonic decays of bottom and charm mesons;
2. weak exclusive hadronic decays of bottom and charm mesons;
3. weak inclusive decays;
4. weak leptonic decay of pseudoscalar mesons;
5. weak inclusive $b \to s\gamma$ mediated decays.

We adopt a number of different approaches for these decays as described below.

9.5.1 Exclusive semi-leptonic decays

The matrix element for exclusive semi-leptonic decays of heavy mesons, $X \to Y\ell\nu$, can be written as

$$\mathcal{M} = \frac{G_F}{\sqrt{2}} \langle X|(V-A)^{\mu}|Y\rangle \bar{u}(p_{\nu})\gamma^{\mu}(1-\gamma_5)u(p_{\ell}), \tag{164}$$

where p_{ℓ} is the momentum of the outgoing charged lepton, p_{ν} is the momentum of the neutrino and G_F is the Fermi constant. The hadronic current $\langle X|(V-A)^{\mu}|Y\rangle$ can be written as a general Lorentz structure, for a particular type of decay, with a number of unknown form factors.

We have implemented a number of form-factor models based on experimental fits, QCD sum rule calculations and quark models. The form factors for the weak decay of pseudoscalar mesons are implemented using the general Lorentz-invariant form. In each case the momentum of the decaying particle, X, is p_X while the momentum of the decay product, Y, is p_Y. In general the form factors are functions of the momentum transfer q^2 where $q = p_X - p_Y$. The masses of the decaying particle and hadronic decay product are m_X and m_Y respectively.

The scalar-scalar transition matrix element is defined by

$$\begin{aligned}&\langle Y(p_Y)|(V-A)_\mu|X(p_X)\rangle\\&\quad=(p_X+p_Y)_\mu f_+(q^2)\\&\qquad+\left\{\frac{m_X^2-m_Y^2}{q^2}\right\}q_\mu[f_0(q^2)-f_+(q^2)],\end{aligned}\tag{165}$$

where $f_+(q^2)$ and $f_0(q^2)$ are the form factors for the transition. In general the terms proportional to q^μ give rise to contributions proportional to the lepton mass for semi-leptonic decays and therefore only contribute to the production of tau leptons.

The scalar-vector transition matrix element is defined to be

$$\begin{aligned}&\langle Y(p_Y)|(V-A)_\mu|X(p_X)\rangle\\&\quad=-i\epsilon_\mu^*(m_X+m_Y)A_1(q^2)\\&\qquad+i(p_X+p_Y)_\mu\epsilon^*\cdot q\frac{A_2(q^2)}{m_X+m_Y}\\&\qquad+iq_\mu\epsilon^*\cdot q\frac{2m_Y}{q^2}(A_3(q^2)-A_0(q^2))\\&\qquad+\epsilon_{\mu\nu\rho\sigma}\epsilon^{*\nu}p_X^\rho p_Y^\sigma\frac{2V(q^2)}{m_X+m_Y},\end{aligned}\tag{166}$$

where the form factor $A_3(q^2)$ can be defined in terms of A_1 and A_2 using

$$A_3(q^2)=\frac{m_X+m_Y}{2m_Y}A_1(q^2)-\frac{m_X-m_Y}{2m_Y}A_2(q^2)\tag{167}$$

and $A_0(0)=A_3(0)$. The independent form factors are $A_0(q^2)$, $A_1(q^2)$, $A_2(q^2)$ and $V(q^2)$.

The transition matrix element for the scalar-tensor transition is

$$\begin{aligned}&\langle Y(p_Y)|(V-A)_\mu|X(p_x)\rangle\\&\quad=ih(q^2)\epsilon_{\mu\nu\lambda\rho}\epsilon^{*\nu\alpha}p_{Y\alpha}(p_X+p_Y)^\lambda(p_X-p_Y)^\rho\\&\qquad-k(q^2)\epsilon_{\mu\nu}^*p_Y^\nu-b_+(q^2)\epsilon_{\alpha\beta}^*p_X^\alpha p_X^\beta(p_X+p_Y)_\mu\\&\qquad-b_-(q^2)\epsilon_{\alpha\beta}^*p_X^\alpha p_X^\beta(p_X-p_Y)_\mu,\end{aligned}\tag{168}$$

where $h(q^2)$, $k(q^2)$, $b_-(q^2)$ and $b_+(q^2)$ are the Lorentz invariant form factors.

The combination of the form factors and the leptonic current is handled by the SemiLeptonicScalarDecayer class, which combines the form factor and the current to calculate the matrix element and uses the methods available in the DecayIntegrator class, from which it inherits, to generate the momenta of the decay products.

In addition to the form factors for the standard weak current we include the form factors needed for weak radiative decays where available, although these are not currently used in the simulation.

The various form factors that are implemented in Herwig++ are described below. They all inherit from the ScalarFormFactor class and implement the relevant virtual member functions for the calculation of the form factors.

BallZwickyScalarFormFactor This is the implementation of the QCD sum rule calculation of the form factors of Ref. [111] for the decay of a B-meson to a light pseudoscalar meson.

BallZwickyVectorFormFactor This is the implementation of the QCD sum rule calculation of the form factors of Ref. [112] for the decay of a B-meson to a light vector meson.

HQETFormFactor This implements the form factors for the transitions between mesons containing bottom and charm quarks in the heavy quark limit. The parameterization of Ref. [113] for the finite-mass corrections is used together with the experimental results of Refs. [114, 115].

ISGWFormFactor The ISGW form factor model [116] is one of the original quark models for the form factors and is included in the simulation mainly for comparison with the later, ISGW2 [117], update of this model. This set of form factors has the advantage that it includes form factors for most of the transitions required in the simulation. The form factors are taken from Ref. [116] together with the form factors that are suppressed by the lepton mass from Refs. [118, 119].

ISGW2FormFactor The ISGW2 form factors [117] are an update of the original ISGW form factors [116]. As with the original model they are based on a quark model and supply most of the form factors we need for the simulation.

KiselevBcFormFactor This is the implementation of the form factors of Ref. [120] for the weak decays of B_c mesons. This model is used as the default model for weak B_c decays as the branching ratios for the B_c meson used in the simulation are calculated using the same model.

MelikhovFormFactor This is the implementation of the relativistic quark model form factors of Ref. [121] for $B\to\pi,\rho$.

MelikhovStechFormFactor This is the implementation of the model of Ref. [122], which is an update of the model of Ref. [121] including additional form factors.

WSBFormFactor This is the implementation of the form factor model of Ref. [123] for the semi-leptonic form factors. It includes form factors for a number of D, B and D_s

decays. In practice the parameters of the model were taken from Ref. [124], which includes a number of transitions that were not considered in the original paper.

This form factor model is included both to give an alternative for many modes to the ISGW models and for use in the factorization approximation for hadronic charm meson decays.

9.5.2 Exclusive hadronic decays

We include two types of simulation of exclusive weak meson decays. The first is based on the naïve factorization approximation. If we consider, for example, the decay of a charm meson then the effective Lagrangian for the interaction can be written as [124]

$$\mathcal{L}_{\text{eff}} = \frac{G_F}{\sqrt{2}} V_{ud} V_{sc} [a_1 (\bar{u}\gamma_\mu P_L d)(\bar{s}\gamma_\mu P_L c) + a_2 (\bar{s}\gamma_\mu P_L d)(\bar{u}\gamma_\mu P_L c)], \tag{169}$$

where G_F is the Fermi constant, V_{ud} and V_{sc} are the relevant CKM matrix elements and $a_{1,2}$ are scale-dependent coefficients. The remainder of the expression involves the currents for the quark fields. When we consider the transition between mesonic states the matrix element can be written in terms of the form factors, for the $c \to s$ or $c \to u$ transitions, and weak currents for $(\bar{u}\gamma_\mu P_L d)$ or $(\bar{s}\gamma_\mu P_L d)$.

This allows us to simulate weak hadronic decays using the form factors we have already implemented for semi-leptonic meson decays together with the weak currents from tau decays. The combination of the form factor classes, which inherit from ScalarFormFactor, and weak currents, which inherit from WeakDecayCurrent, is handled by the ScalarMesonFactorizedDecayer class for the simulation of weak hadronic decays in the factorization approximation.

In addition to the weak exclusive decays based on the factorization approximation we include a small number of classes for the simulation of $D \to K\pi\pi$ Dalitz decays based on various experimental fits. Currently there are three such models implemented.

DtoKPiPiCLEO This class implements the CLEO fits of Refs. [125] and [126] for the decays $D^0 \to \bar{K}^0\pi^+\pi^-$ and $D^0 \to K^-\pi^+\pi^0$. This is our default simulation of these decays.

DtoKPiPiE691 The DtoKPiPiE691 class implements the model of E691 [127] for the decays $D^0 \to \bar{K}^0\pi^+\pi^-$, $D^0 \to K^-\pi^+\pi^0$ and $D^+ \to K^-\pi^+\pi^-$. This is our default simulation for the $D^+ \to K^-\pi^+\pi^-$ decay.

DtoKPiPiMarkIII This class implements the model of the Mark-III collaboration for the decays $D^0 \to \bar{K}^0\pi^+\pi^-$, $D^0 \to K^-\pi^+\pi^0$, $D^+ \to K^-\pi^+\pi^-$ and $D^+ \to \bar{K}^0\pi^+\pi^0$. This is our default model for the decay mode $D^+ \to \bar{K}^0\pi^+\pi^0$.

9.5.3 Weak inclusive decays

In addition to the exclusive weak decays of the mesons to specific final states we include a number of models of the decay of mesons containing a heavy, *i.e.* bottom or charm, quark based on the partonic decay of the heavy quark. The Herwig++ cluster hadronization model is then applied to the resulting partonic final state to produce hadrons. This approach is primarily used for the bottom mesons where there are insufficient exclusive modes to saturate the branching ratios. All of the classes implementing partonic decay models inherit from the PartonicDecayerBase to use the cluster hadronization model to hadronize the partonic final state.

The HeavyDecayer class implements the weak decays of mesons using either the weak $V - A$ matrix element or flat phase space. The WeakPartonicDecayer includes additional features to simulate decays intended to increase the rate of baryon production and gluonic penguin decays.

In addition the BtoSGammaDecayer for weak penguin-mediated decays, described in Sect. 9.5.5, and the QuarkoniumDecayer class for the decay of bottomonium and charmonium resonances, described in Sect. 9.4.4, also perform partonic decays and inherit from the PartonicDecayerBase class.

9.5.4 Leptonic decays

There are a small number of decays of pseudoscalar mesons to a charged lepton and a neutrino, *e.g.* $\pi \to \ell\nu$ and $D_s \to \ell\nu$. For most of these decays the inclusion of the matrix element is superfluous as the decay products are stable. However the B and D_s mesons can decay in this way to the τ and therefore we include the PScalarLeptonNeutrinoDecayer class to simulate these decays using the matrix element

$$\mathcal{M} = \frac{1}{\sqrt{2}} f_P G_F V_{CKM} m_l \bar{u}(p_\ell)(1 - \gamma_5) v(p_\nu), \tag{170}$$

where f_P is the pseudoscalar decay constant, G_F is the Fermi constant, V_{CKM} is the relevant CKM matrix element, m_ℓ is the mass of the lepton, p_ℓ is the momentum of the charged lepton and p_ν is the momentum of the neutrino.

9.5.5 $b \to s\gamma$

There is a range of decays, both inclusive and exclusive, mediated by the $b \to s\gamma$ transition. We currently only include modelling of the inclusive decay. These decays are simulated by using a partonic decay of the B meson to a photon and a hadronic system, composed of a quark and antiquark, which recoils against the photon. The mass spectrum of the hadronic system is calculated using a theoretical model.

The calculation of the mass spectrum is handled by a class inheriting from the BtoSGammaHadronicMass class.

Different models of the mass spectrum can then be implemented by inheriting from this class. Currently we have only implemented two such models. The first, BtoSGammaFlatEnergy, is solely designed for testing and generates a mass spectrum such that the photon energy distribution is flat. The second model, BtoSGammaKagan, which is the default, implements the theoretical calculation of Ref. [128]. The BtoSGammaDecayer then uses the calculation of the hadronic mass spectrum to simulate the partonic decay as a model of the inclusive mode. As with the Decayers described in Sect. 9.5.3 the BtoSGammaDecayer inherits from the PartonicDecayerBase class to use the cluster model to perform the hadronization of the partonic final state.

9.6 Code structure

The HwDecayHandler class, which inherits from the DecayHandler class of ThePEG, is responsible for handling all particle decays in Herwig++. It uses the DecaySelector from the ParticleData object of the decaying particle to select a DecayMode object corresponding to a specific decay according to the probabilities given by the branching ratios for the different modes. The DecayMode object then specifies a Decayer object that is responsible for generating the kinematics of the decay products for a specific decay.

All of the Decayer classes in Herwig++ inherit from the HwDecayerBase class, which in turn inherits from the Decayer class of ThePEG. In turn, with the exception of the Hw64Decayer and MamboDecayer classes, which implement general phase-space distributions for the decay products, all the Decayer classes in Herwig++ inherit from either the DecayIntegrator or PartonicDecayBase classes.

The DecayIntegrator class provides a sophisticated multichannel phase space integrator to perform the integration over the phase space for the decays. This means that the calculation of the matrix element and specification of the phase-space channels are all that is required to implement a new decay model. The majority of the matrix elements are calculated as helicity amplitudes, which allows the spin-propagation algorithm of Refs. [24–27] to be implemented. The structure of the Herwig++ Decayer classes and HwDecayHandler is designed so that these correlations are automatically included provided the helicity amplitudes for the matrix elements are supplied.

The PartonicDecayBase class provides a structure so that the decay products of a partonic hadron decay can be hadronized using the cluster model, while at the same time ensuring that there is no overlap with the particle's exclusive decay modes. All classes implementing partonic decays in Herwig++ inherit from the PartonicDecayBase class.

Certain Decayer classes also make use of helper classes to implement the decays. The main examples are:

- the WeakDecayCurrent, which provides a base class for the implementation of weak hadronic currents, is used by the TauDecayer, SemiLeptonicScalarDecayer and ScalarMesonFactorizedDecayer classes, which implement tau decays, semi-leptonic meson decays and hadronic weak meson decays using the naïve factorization approximation, respectively;
- the ScalarFormFactor, which provides a base class for the implementation of the scalar form factors and is used by the SemiLeptonicScalarDecayer and ScalarMesonFactorizedDecayer classes, which implement semi-leptonic meson decays and hadronic weak meson decays using the naïve factorization approximation, respectively;
- the BtoSGammaHadronicMass, which provides a model of the hadronic mass spectrum in inclusive $b \to s\gamma$ decays performed by the BtoSGammaDecayer class.

The vast majority of the decay models have a large number of parameters, all of which are accessible via the Interfaces of the classes. A more detailed description of both the physics models used in the code and their parameters can be found in the Doxygen documentation and Refs. [22, 70].

There are a number of classes that are designed to include the off-shell weight given in (137) in the generation of the particle decays. The GenericWidthGenerator is designed to use the information on the partial widths for the different decay modes supplied by the Decayer classes, which inherit from DecayIntegrator, to calculate the running width for a given particle. The GenericMassGenerator class then uses the running width to allow the weight given in (137) to be included when generating the particle decays. The inheriting ScalarMassGenerator class implements the Flatté lineshape [86] for the $a_0(980)$ and $f_0(980)$ mesons.

For decays where the decay products can be off-shell, and three-body decays, integrals over either the masses of the decay products or the three-body phase space must be performed in order to calculate the running partial widths. In order to facilitate the calculation of the partial widths a number of classes inheriting from the WidthCalculatorBase class are implemented to calculate the partial widths for various decays:

- the TwoBodyAllOnCalculator returns the partial width for a two-body decay where both the decay products are on mass-shell;
- the OneOffShellCalculator returns the partial width for a decay where one of the outgoing particles is off mass-shell;
- the TwoOffShellCalculator returns the partial width for a decay where two of the outgoing particles are off mass-shell;
- the ThreeBodyAllOnCalculator returns the partial width for a three-body decay where all the decay products are on mass-shell by performing the two non-trivial integrals over the phase-space variables;

– the ThreeBodyAllOn1IntegralCalculator returns the partial width for a three-body decay where all the decay products are on mass-shell by performing one integral over the phase-space variables, this requires that the second integral has already been performed analytically.

10 Summary

In this manual we have described the physics and structure of Herwig++ version 2.2. More detailed technical documentation can be obtained from the web site http://projects.hepforge.org/herwig as well as a growing number of user guides, example applications, frequently-asked-questions and other useful information. Most of this is obtained by following the "wiki" link at the top of the page. To be able to contribute to the wiki and submit trac tickets, please email the authors, at herwig@projects.hepforge.org To improve the current version of Herwig++ and plan development of future versions, we depend on feedback from users. If you use Herwig++ please register at the address above and post your experience (positive or negative) and code examples you feel other users would benefit from, and open a trac ticket for any bugs or unexpected features you find, as well as any new features or improvements you would like to see. Of course, for any bug report, the more clearly you can illustrate the problem, and the fact that it is a problem with Herwig++ and not an external package it is connected to, the more quickly we are likely to be able to solve it.

Herwig++ has been extended enormously since the last version for which a published manual exists, 1.0. It now provides complete simulation of hadron–hadron collisions with a new coherent branching parton shower algorithm, including quark mass effects, a sophisticated treatment of BSM interactions and new particle production and decay, an eikonal model for multiple partonic scattering, greatly improved secondary decays of hadrons and tau leptons and a set of input parameters that describe e^+e^- annihilation data rather well.

New features planned for the near future include: an improved treatment of baryon decays; spin correlations within the parton shower; 'multiscale' showering of unstable particles; simulation of DIS processes; B mixing; and an improved treatment of gluon splitting to heavy quarks. Of course we are all users of Herwig++ as well as developers and are working on a large number of other new features related to phenemenological studies we are making. The list will continue to grow, according to the physics interest and needs of ourselves and others using it for physics studies.

In many aspects, the physics simulation included in Herwig++ is already superior to that in the FORTRAN HERWIG and our intention is that with the features just listed, the next major version release of Herwig++ will replace HERWIG as *the* recommended product for simulating hadron emission reactions with interfering gluons.

Acknowledgements This work was supported by the Science and Technology Facilities Council, formerly the Particle Physics and Astronomy Research Council, the European Union Marie Curie Research Training Network MCnet under contract MRTN-CT-2006-035606 and the Helmholtz–Alliance "Physics at the Terascale". Manuel Bähr and Simon Plätzer acknowledge support from the Landesgraduiertenförderung Baden-Württemberg. Keith Hamilton acknowledges support from the Belgian Interuniversity Attraction Pole, PAI, P6/11.

Development of Herwig++ would not have been possible without the early work of Alberto Ribon and Phil Stephens or the parallel development of ThePEG and the support provided by Leif Lönnblad. We are indebted to our collaborators Christoph Hackstein, Andrzej Siódmok and Jon Tully for their valuable input and feedback, as well as the users who have helped with testing of early versions, particularly Jeremy Lys. The LCG Generator Services project have provided useful feedback. Fruitful discussions with Andy Buckley are gratefully acknowledged. We have received technical advice and support from the HepForge project who host the Herwig++ development environment and provide a variety of related services. The tuning of Herwig++ to experimental data would not have been possible without the use of GRIDPP computer resources.

Appendix A: Repository commands

The composition of the Repository is controlled through a simple command language, which can be used either interactively after calling `Herwig++ read` without any arguments, or through input files, which can be provided as arguments to the `Herwig++ read` command. The following overview only describes the most important repository commands. Examples of input files using this command language can be found in the

```
HERWIGPATH/share/Herwig++
HERWIGPATH/share/Herwig++/defaults
```

directories. Please note that the repository allows for an internal filesystem-like structure of directories and entries. This does not, however, correspond to any physical files on the operating system.

We first give the commands that affect the overall state of the Repository, followed by commands for navigating the filesystem-like structure, event generation, creating and modifying objects in the Repository, and finally some miscellaneous commands. We conclude with a brief example of using the filesystem-like structure of the Repository to obtain the parameter values used in a run.

Repository state

`save` *file*
Save the current repository state.
`load` *file*
Load a repository. Replaces the current state.

`read` *file*
Read in additional commands from *file*.
`library` *lib*
Load the dynamic shared library *lib* immediately, making all classes in the library available.

Repository tree

All operations in this section affect the repository tree only, not the file system.

`pwd`
Print the current directory path.
`cd` *dir*
Change the current directory to *dir*.
`mkdir` *dir*
Make a directory called *dir*.
`ls` *[dir]*
List the entries in the current directory or in *dir*.
`rmdir` *dir*
Remove an empty directory.
`rrmdir` *dir*
Remove a directory and all its contents recursively.

Event generation

`run` *run-name generator*
Run the *generator* object for the pre-set number of events. Files are saved under the label *run-name*.
`saverun` *run-name generator*
Save a *generator* as a file *run-name.run*, ready to use with `Herwig++ run`.

Classes, objects, interfaces

`create` *classname name [library]*
Create a new object of C++ class *classname* and store it under *name*. Optionally, specify the name of the library file containing the class.
`mv` *old-name name*
Rename a repository object.
`cp` *old-name name*
Copy a repository object. The copy's interfaces will be identical to the original's at the time of copying, but can then be set independently.
`rm` *name*
Remove *name* from the repository.
`get` *interface*
Get the current value of an interface.
`set` *interface value*
Set the value of an interface. This can be either a numerical value, the name of an object in the Repository, or a defined key word for a Switch. `set` can also be used to set the value of a member of an interface vector.
`insert` *vector-interface[index] value*
Insert a value into a vector of interface parameters.
`erase` *vector-interface[index]*
Remove a value from a vector of interface parameters.
`describe` *object[:interface]*
Describes *object* and lists its interfaces, or describes an interface.

Miscellaneous commands

`setup` *object args...*
Passes *args* to *object*'s own setup function.[28]
`decaymode` *tag BR active? decayer*
Register a decay mode where *tag* is a semicolon-delimited description of a decay, using the repository particle names, such as `pi0->gamma,e-,e+;`, *BR* is the mode's branching ratio, *active?* is either 1 or 0, indicating whether this decay mode is active or not, and *decayer* is the object that handles the generation of the kinematics for this decay mode.
`makeanti` *particle1 particle2*
Register *particle1* and *particle2* to be a particle-antiparticle pair.
`defaultparticle` *particle [particle ...]*
Register *particle*s as default particles, only these particles are used with every event generator.

A.1 Example

This is a brief example of using the Repository to extract the values of the default kinematic cuts on particles produced in the hard scattering process. Many more complicated tasks can also be performed.

While we expect that the most common way of using the Repository will be changing the `.in` file for the relevant collider it is sometimes useful to browse the directory-like structure to check the parameters being used.

The filesystem-like structure of the Repository can be explored using

```
Herwig++ read
```

which gives access to a command-line prompt. The current directory will be the last one used in the default Herwig++ Repository, currently `/Herwig/Analysis`. Typing `ls` will give a list of the AnalysisHandler objects that have been created to analyse events generated by Herwig++.

The objects that supply the kinematic cuts are in the directory `/Herwig/Cuts` and can be listed using

```
cd /Herwig/Cuts
ls
```

[28] Used *e.g.* for particle data as `setup` *particle ID PDGname mass width cut ctau charge colour spin stable*.

which will list the following objects

```
EECuts
JetKtCut
LeptonKtCut
MassCut
PhotonKtCut
QCDCuts
TopKtCut
```

The QCDCuts and EECuts objects are the main objects that impose the cuts for hadron-hadron and lepton-lepton events respectively. Repository commands can now be used to get information about the objects and their parameters, for example

```
describe QCDCuts
describe QCDCuts:OneCuts
get QCDCuts:OneCuts
```

will give a brief description of the QCDCuts object and its interfaces, followed by the description of the OneCuts interface and the list of objects used to give the cuts on individual particles, or groups of particles.

The JetKtCut object is used to impose cuts on partons (the quarks other than the top quark, and the gluon). The value of the cut on the transverse momentum of the partons can be accessed and increased from the default value of 20 GeV to 30 GeV using

```
get JetKtCut:MinKT
set JetKtCut:MinKT 30.*GeV
```

A new event generator file with this changed cut could now be written to file using

```
saverun LHCnew LHCGenerator
```

for the LHC.

Appendix B: Examples

This appendix contains a number of examples of using Herwig++. Example input files for Herwig++ are also supplied in the directory

```
HERWIGPATH/share/Herwig++/
```

where HERWIGPATH is the location of the Herwig++ installation. There are examples for e^+e^- collisions at LEP and ILC energies and hadron-hadron collisions at the Tevatron and LHC, as well as examples of using the different BSM models included in Herwig++.

These can all be run with

```
Herwig++ read Collider.in
Herwig++ run -N no_of_events Collider.run
```

where Collider.in is one of the example input files. The first read stage reads the input file and persistently writes the EventGenerator object it creates into the Collider.run file for future use. The second run stage then uses this persistently stored generator to generate no_of_events events.

The default parameters for the generator have already been pre-set using the files contained in the directory

```
HERWIGPATH/share/Herwig++/defaults
```

and used to build the HerwigDefaults.rpo Repository file distributed with the release. Most users will not need to rebuild this file, but may need to look at the default parameters contained in the files used to build it.

More information on running Herwig++ can be found on the wiki and in Appendix A.

The remainder of this appendix is designed to illustrate how these input files can be adapted to simulate the physics scenario of interest to the user by changing the hard processes, cuts, etc. All of the examples, together with the source code, can be obtained from our wiki, where new examples will also be added in the future. Several of the examples assume that hadron-hadron collisions are being generated. If you are simulating lepton-lepton collisions replace LHCGenerator with LEPGenerator.

B.1 Switching parts of the simulation off

In some cases it may be useful to switch off certain stages of the simulation. The most simple way to do that is by assigning NULL pointers to the appropriate StepHandlers of the EventHandler. The following statements have to be added to the Generator.in file used.

```
cd /Herwig/EventHandlers
set LHCHandler:CascadeHandler NULL
set LHCHandler:HadronizationHandler NULL
set LHCHandler:DecayHandler NULL
```

to switch off the parton shower, hadronization and hadronic decays. For e^+e^- collisions the corresponding EventHandler is called LEPHandler. In e^+e^- collisions it is possible, although not recommended, to switch the shower off while still hadronizing the event. This is not possible in hadron collisions because the decay of the hadronic remnant, which must occur before the event can be hadronized, is currently handled by the shower module.

The Shower step can be controlled in more detail: initial-state radiation can be turned off using

```
set /Herwig/Shower/SplittingGenerator:ISR No
```

Final-state radiation can be turned off using

```
set /Herwig/Shower/SplittingGenerator:FSR No
```

Multiple interactions can be turned off using

```
set /Herwig/Shower/ShowerHandler:MPI No
```

By default Herwig++ now uses a multiple scattering model of the underlying event. If you wish to use the old UA5 model, which we do not recommend for realistic physics studies, you should first turn off the multiple scattering model and then enable the UA5 model[29]:

[29] It should be remembered that there is a difference between the name of the class used to create objects in the Repository and the names of

```
set /Herwig/Shower/ShowerHandler:MPI No
cd /Herwig/Hadronization/
set ClusterHadHandler:UnderlyingEventHandler \
UA5
```

B.2 Changing particle properties

In Herwig++ each particle's properties are contained in a ParticleData object. This has a number of interfaces that can be used to change the properties. The files `leptons.in`, `quarks.in`, `bosons.in`, `mesons.in`, `baryons.in` and `diquarks.in`, which can be found in the `HERWIGPATH/share/Herwig++/defaults` directory, set up the default properties of each particle type. The names of the ParticleData objects in the Repository can be found in these input files or by browsing the `/Herwig/Particles` directory in the Repository using `Herwig++ read`.

All properties can be changed in the input file for an event generator. For example to change the mass of the top quark to 170 GeV the following lines should be added

```
set /Herwig/Particles/t:NominalMass 170.*GeV
```

By default, the properties of particles and their antiparticles are forced to be the same so this will change the mass of both the top and antitop.

The neutral pion can be set stable using

```
set /Herwig/Particles/pi0:Stable Stable
```

B.3 Changing some simple cuts

In many cases it will be important to specify particular cuts on the hard process. The default values for all cuts in Herwig++ are given in the file[30] `Cuts.in`. Here we give a number of examples of changing the cuts.

For example, in order to change the minimum $k_\perp$ for a parton produced in the hard process to 30 GeV one should add

```
set /Herwig/Cuts/JetKtCut:MinKT 30.0*GeV
```

The pseudorapidity cut on hard photons can be changed to $|\eta| < 4$ with

```
set /Herwig/Cuts/PhotonKtCut:MinEta -4.
set /Herwig/Cuts/PhotonKtCut:MaxEta 4.
```

and the cut on the minimum invariant mass of the hard process can be increased to 50 GeV with

```
set /Herwig/Cuts/QCDCuts:MHatMin 50.*GeV
```

If one wants to restrict the invariant mass of the final state in lepton pair production, however, one should use the class V2LeptonsCut, our default instance of this is called `MassCut`. In this case one has to specify

```
set /Herwig/Cuts/MassCut:MinM 20.*GeV
```

B.4 Setting up an AnalysisHandler

Creating a new AnalysisHandler requires the following steps:

1. Create skeleton class files. This can be done in emacs by loading a Lisp script that can be found at `THEPEGPATH/lib/ThePEG.el`.
2. Invoking `M-x ThePEG-AnalysisHandler-class-files` queries the user for some input and interactively creates the necessary files for an AnalysisHandler. These are the questions asked:
   (a) **Class name:**
   Use for example `MyName::Foo`. It is useful to use a namespace (replacing `MyName` with your name, of course)
   (b) **Base class name:**
   The right answer is already suggested: `AnalysisHandler`
   (c) **include file for the base class:**
   Also filled out already
   (d) **Will this class be persistent (y or n)**
   If persistent members are needed: `y` otherwise `n`. `n` is appropriate here.
   (e) **Will this class be concrete (y or n)**
   The answer `y` is appropriate unless you're writing an abstract base class.
   This will create the following files:
   `Foo.h, Foo.fh, Foo.icc, Foo.cc`
3. If actions need to be performed as part of the initialization (*e.g.* booking histograms) or termination (*e.g.* writing results to disk), the required class methods can be automatically created by the same Lisp script:
   (a) First the declaration of the methods. Go to `Foo.h` where it says

   ```
   // If needed, insert declarations of
   // virtual function defined in the
   // InterfacedBase class here (using
   // ThePEG-interfaced-decl in Emacs).
   ```

   and in emacs use `M-x ThePEG-interfaced-decl`. This will insert the declaration of the methods needed.
   (b) To insert the implementation of these methods, go to `Foo.icc` where it says

   ```
   // If needed, insert default implement-
   // ations of virtual function defined
   // in the InterfacedBase class here
   // (using ThePEG-interfaced-impl in Emacs).
   ```

the objects, here `ClusterHadHandler` is the name of the ClusterHadronizationHandler object used by default in Herwig++ to perform the hadronization.

[30] This can be found in the directory `HERWIGPATH/share/Herwig++/defaults`

and start `M-x ThePEG-interfaced-impl`.

4. There is one important check left. Every class that can be administered by ThePEG has to specify a static function returning the name of the library that the class is stored in. This has to agree with the library name in the `Makefile`. In our case it is:

```
static string library() { return "Foo.so"; }
```

   By default it is set to the name of the class, *i.e.* `Foo.so` in our case, but may need changing if you are linking several classes into one library.
5. A fully working AnalysisHandler, which currently has no functionality, is now implemented. A `Makefile` to compile it is supplied with the release. Copy it to your working directory

```
cp HERWIGPATH/share/Herwig++/Makefile-UserModules\
Makefile
```

   It will create a shared library object named after the `.cc` filename, *e.g.* `Foo.so`.
6. The class can now be compiled by invoking `make`. This command should terminate successfully.
7. Calling the newly created class requires copying an appropriate `Generator.in` file into your directory from `HERWIGPATH/share/Herwig++` and modifying it with the following statements

```
cd /Herwig/Analysis
create MyName::Foo foo Foo.so
insert \
/Herwig/Generators/LHCGenerator:AnalysisHandlers \
0 foo
```

   which will create an instance of the new class Foo and then insert it at position 0 in the vector of references to AnalysisHandlers. It is always safest to insert the newly created AnalysisHandler as the first entry in the list unless you are sure of how many AnalysisHandlers have already been inserted.

B.5 Usage of ROOT

To write a ROOT [129] based analysis outside the Herwig++ source tree, which is the recommended way of doing it, it is *not* necessary to use the configure flag `-with-root=/path/to/root`. This is only required to enable internal ROOT based analyses, but you must define the environment variable `$ROOTSYS` in any case, as it is required by ROOT, as described in the ROOT manual.

In the following we will show two examples of an AnalysisHandler that will use ROOT output. Please refer to Appendix B.4 for the generic instructions on setting up an analysis. Here, we will only mention specific code snippets, which should be inserted in the appropriate locations.

The short description of what has to be done is:

1. create a new class derived from AnalysisHandler;
2. implement the functionality required;
3. compile a library from it;
4. create a `Generator.in` file where this AnalysisHandler is called and run it.

Points 1 and 3 are universal for every AnalysisHandler and are described in Appendix B.4. However, the corresponding library and include statements for ROOT have to be added: First copy the Makefile

```
cp HERWIGPATH/share/Herwig++/Makefile-UserModules \
Makefile
```

and then add the following lines

```
ROOTCFLAGS := $(shell root-config --cflags)
ROOTGLIBS := $(shell root-config --glibs)
ROOT = $(ROOTCFLAGS) $(ROOTGLIBS)
```

Finally the line containing the compilation command has to be changed to include the content of the `ROOT` variable:

```
%.so : %.cc %.h
$(CXX) -fPIC $(CPPFLAGS) $(INCLUDE) $(ROOT) \
$(CXXFLAGS) -shared $< -o $@
```

A shared library with your code will be created in the directory where you execute `make`. You need to make sure that the ROOT libraries can be found at run-time. On Linux systems you can add paths to the libraries to the environment variable `$LD_LIBRARY_PATH`.

B.5.1 Root histograms

The goal of this example is to write an AnalysisHandler that writes the charged particle multiplicity per event to a histogram and saves it as an encapsulated postscript (`eps`) file. This is only a minimal example of the use of ROOT in the analysis of Herwig++ events. It may for example be more useful to write the histogram to a file, but we leave this to the user as it is beyond the scope of this manual.

First a new AnalysisHandler has to be created, as described in Appendix B.4. After setting up the necessary files, the new functionality can be implemented:

– **Foo.h**
  In the header file, several additional include files have to be specified

```
#include "TH1F.h"
#include "TCanvas.h"
```

  They are the ROOT headers of histograms and a canvas upon which to draw the histogram.

  The histogram should be available as a member of this new class, because information on every event has to be stored in it. A pointer to the histogram as private member variable of the class can be used for that purpose:

```
private:
/**
* A pointer to a Root histogram
*/
TH1F* histo;
```

– **Foo.icc**
The histogram should be booked in

```
inline void Foo::doinitrun()
```

with the following commands:

```
histo = new TH1F("test",
"charged multiplicity",
150, 0, 600);
histo->SetXTitle("N_{ch}");
histo->SetYTitle("events");
```

In

```
inline void Foo::dofinish()
```

the histogram is drawn on a canvas and saved to disk. Finally the pointers are freed:

```
TCanvas *can = new TCanvas("plot", "");
histo->Draw();
can->SaveAs("plot.eps");
delete can;
delete histo;
```

– **Foo.cc**
All that remains is the actual filling of the histogram. This functionality will be added to the method

```
void Foo::analyze(tEventPtr event,
long, int loop, int state){
if ( loop > 0 || state != 0 || !event )
return;
/** create local variable to
store the multiplicity */
int mult(0);
/** get the final-state particles */
tPVector particles=event->getFinalState();
/** loop over all particles */
for (tPVector::const_iterator
pit = particles.begin();
pit != particles.end(); ++pit){
/** Select only the charged particles */
if( ChargedSelector::Check(**pit) )
++mult;
}
histo->Fill(mult);
}
```

The test in the first line is recommended for all simple AnalysisHandlers. The meaning of `loop` and `state` can be obtained from the Doxygen documentation of the AnalysisHandler class.

B.5.2 rtuple with TTree

If you are working with ROOT already, you can store events in an rtuple directly. This example shows how to define an AnalysisHandler that prepares an rtuple with ROOT TTree. It is extracted from a more detailed example, available from the wiki, for analysing four-*b* events at LEP.

– **Foo.h**
First, add the needed ROOT header files to your header file for declaration of all ROOT classes you are going to use. In this case:

```
#include "TTree.h"
#include "TFile.h"
```

Add TTree and TFile objects to the private part of the class:

```
private:
// ROOT Tree
TTree * theTree;
// ROOT File
TFile * theFile;
```

Define all the variables and arrays that will be kept in the ROOT tree:

```
private:
// ROOT tree internal arrays and variables
int Nentry, Nqurk, Nhdrn;
int Kf[16], Kp[16];
double Wgt, Alphas;
double Qscl[4];
double Px[16], Py[16], Pz[16], P0[16];
```

– **Foo.icc**
Methods for TTree booking and the writing of the TFile to disk should be called in `doinitrun()` and `dofinish()` respectively. Add the following lines to `doinitrun()`:

```
LEPbbbbComparison::doinitrun () {
...
// create ROOT Tree
theTree = new TTree ("bbbb",
"myAnalysis root tree",
1);
if (!theTree) {
cerr << "ROOT tree has not been created\n";
return;
}
// create ROOT File
theFile = new TFile (outname,"RECREATE");
if (!theFile) {
cerr << "ROOT file has not been created\n";
return;
}
theTree->SetDirectory (theFile);
// define ROOT Tree branches/leaves
theTree->Branch ("Nentry", &Nentry,
"Nentry/I");
theTree->Branch ("Nqurk", &Nqurk, "Nqurk/I");
...
theTree->Branch ("Pz", Pz, "Pz[Nentry]/D");
theTree->Branch ("P0", P0, "P0[Nentry]/D");
...
}
```

The last parameter in each command `theTree->Branch()` should be equal to “Name/Type” of each

variable, *e.g.* `I` → `int`, `D` → `double`, etc. (Information on other types can be found in the ROOT manual). Final commands should be placed in `LEPbbbbComparison::dofinish()`. So, add the following lines to `dofinish()`:

```
LEPbbbbComparison::dofinish() {
...
theTree->GetCurrentFile();
theTree->Write();
theFile->Close();
cout << "ROOT file has been written on disk"
<< endl;
...
}
```

After that, the class will keep `theTree` in `theFile` and write `theFile` to disk.

– **Foo.cc**
All the TTree variables should be set in `analyze(...)`. As soon as all the variables have the right values for analysing an event, execute the `Fill()` method for `theTree`.

```
void Foo::analyze(tEventPtr event, long,
int loop, int state) {
...
// Fill TTree record
if (2 < bquark.size ()) {
theTree->Fill();
}
...
}
```

B.6 Using BSM models

There are example files installed in `HERWIGPATH/share/Herwig++` that show how to use the implemented BSM physics modules. Each one is labelled `Generator-Model.in`. Also associated with each BSM physics module is a `.model` file that is required to run with a specific module but otherwise does not need to be touched by the user. The easiest method to run a BSM physics module is to copy the `Generator-Model.in` file that is appropriate to the collider and model under study and make the necessary changes there.

B.6.1 MSSM

To generate a process in the MSSM, first decide on the accelerator to use, the LHC for example, and then copy `MSSM.model` and `LHC-MSSM.in` files to the location where Herwig++ will be used. `LHC-MSSM.in` contains the settings that a user can manipulate, the default settings are for squark production at the LHC. To change this to gluino production one should delete the lines

```
insert HPConstructor:Outgoing 0 \
/Herwig/Particles/~u_L
insert HPConstructor:Outgoing 1 \
/Herwig/Particles/~u_Lbar
insert HPConstructor:Outgoing 2 \
/Herwig/Particles/~d_L
insert HPConstructor:Outgoing 3 \
/Herwig/Particles/~d_Lbar
```

and insert the line

```
insert HPConstructor:Outgoing 0 \
/Herwig/Particles/~g
```

A SUSY model requires a spectrum file to set the masses and couplings. This file is produced using a spectrum generator.[31] The name of the file, *e.g.* `spectrum.spc`, is set via the command

```
setup MSSM/Model spectrum.spc
```

If the decay table is in a separate file to the spectrum then a second `setup` line should be used to supply this file name.

The next step is to set up the particles that will require spin correlations included in their decays. This is achieved through the **DecayParticles** interface. In the example of gluino production firstly one should remove the lines

```
insert NewModel:DecayParticles 0 \
/Herwig/Particles/~d_L
insert NewModel:DecayParticles 1 \
/Herwig/Particles/~u_L
insert NewModel:DecayParticles 2 \
/Herwig/Particles/~e_R-
insert NewModel:DecayParticles 3 \
/Herwig/Particles/~mu_R-
insert NewModel:DecayParticles 4 \
/Herwig/Particles/~chi_10
insert NewModel:DecayParticles 5 \
/Herwig/Particles/~chi_20
insert NewModel:DecayParticles 6 \
/Herwig/Particles/~chi_2+
```

and then insert the line

```
insert NewModel:DecayParticles 0 \
/Herwig/Particles/~g
```

This will generate spin correlations in the decay of the gluino but not in the subsequent decays of its children. Assuming these too are required then additional lines containing all of unstable products in the cascade decays are needed.

```
insert NewModel:DecayParticles 0 \
/Herwig/Particles/~g
insert NewModel:DecayParticles 1 \
/Herwig/Particles/~d_L
insert NewModel:DecayParticles 2 \
/Herwig/Particles/~u_L
...
```

The rest of the settings in the file deal with general parameters for the run. Herwig++ can then be run as described at the beginning of this appendix.

B.6.2 MUED

The MUED model works in a similar fashion to the MSSM but with some important differences due to the unavailability of spectrum and decay generators. The mass spectrum is

[31] Some of these are listed at http://home.fnal.gov/~skands/slha/.

calculated by Herwig++ once the main parameters have been set via the interfaces

```
set MUED/Model:InverseRadius 500.*GeV
set MUED/Model:LambdaR 20
```

and optionally

```
set MUED/Model:HiggsBoundaryMass 0.*GeV
```

Similarly to above the file `LHC-MUED.in` should be copied to a new file *i.e.* `mymued.in` and the relevant parameters changed there.

The specification of the hard process is done in the same manner as above using the particle content of the MUED model. As there are no decay table generators for UED the possible perturbative decays are calculated automatically for the particles specified through the **DecayParticles** interface. It is advisable to leave the list as it stands in the file as then all of the necessary decays modes for the parents that are children in cascade decays will be created properly.

Finally, the methods for running the generator are the same as above.

B.6.3 RS Model

Currently there are no matrix elements for the hard scattering that have tensor particles as external particles, they are only included as intermediates. The graviton can therefore only be included as a resonance. There is a special class designed to handle this as described in Appendix 5.

The set up differs only slightly from the MSSM and MUED models. Using the example in `LHC-RS.in`, upon copying this to a new file, the lines

```
insert ResConstructor:Incoming 0 \
/Herwig/Particles/g
insert ResConstructor:Incoming 1 \
/Herwig/Particles/u
insert ResConstructor:Incoming 2 \
/Herwig/Particles/ubar
insert ResConstructor:Incoming 3 \
/Herwig/Particles/d
insert ResConstructor:Incoming 4 \
/Herwig/Particles/dbar

insert ResConstructor:Intermediates 0 \
/Herwig/Particles/Graviton

insert ResConstructor:Outgoing 0 \
/Herwig/Particles/e+
insert ResConstructor:Outgoing 1 \
/Herwig/Particles/W+
insert ResConstructor:Outgoing 2 \
/Herwig/Particles/Z0
insert ResConstructor:Outgoing 3 \
/Herwig/Particles/gamma
```

can be changed to suit the user's needs. The only parameter in this model is the cutoff scale and it is changed through the line

```
set RS/Model:Lambda_pi 10000*GeV
```

Again, running the generator follows the same steps as before.

B.7 Intrinsic p_T

An example of a particular choice for the implementation of the intrinsic p_T can be found in the default file `Shower.in`.

```
set Evolver:IntrinsicPtGaussian 2.2*GeV
```

As discussed in Appendix C, a Gaussian distribution for intrinsic p_T has been implemented. The root mean square intrinsic p_T of the Gaussian distribution required, σ, is set using the `IntrinsicPtGaussian` parameter. The values for the intrinsic p_T are generated according to:

$$\mathrm{d}^2 p_T \frac{1}{\pi\sigma^2} \exp\left[-\left(\frac{p_T}{\sigma}\right)^2\right]. \tag{B.1}$$

The default example above is for a Gaussian distribution with root mean square p_T of 2.2 GeV. In addition to this, there is the option of selecting an inverse quadratic distribution for the intrinsic p_T according to:

$$\mathrm{d}^2 p_T \frac{1}{\pi \ln(1+\frac{p_{T\max}^2}{\gamma^2})} \frac{1}{\gamma^2 + p_T^2}, \tag{B.2}$$

where γ is a constant and $p_{T_{\max}}$ is an upper-bound on the modulus of p_T and makes the distribution normalizable. These parameters can be changed from their default values in `Shower.in`.

```
set Evolver:IntrinsicPtGamma 0*GeV
set Evolver:IntrinsicPtIptmax 0*GeV
```

A mixture of both distributions can also be selected by setting a parameter β in `Shower.in` and is the proportion of the inverse quadratic distribution required and ranges between 0 and 1.

```
set Evolver:IntrinsicPtBeta 0
```

Here the default setting is to generate the intrinsic p_T according to the Gaussian distribution only.

B.8 LesHouchesEventHandler

In order to use partonic events generated by an external matrix element generator, a LesHouchesEventHandler object has to be created in the Repository. This object is supplied with at least one LesHouchesReader object. LesHouchesReader objects supply events in the Les Houches Accord (LHA) format [29] reading a file of events.

Here we give an example of how to use LHA event files. The reading of the events is performed by the MadGraphReader class. This is not, however, limited to reading events generated by MadEvent [130] but can handle arbitrary event files in the Les Houches format.

First, the libraries required must be loaded,

```
library LesHouches.so
library MadGraphReader.so
```

Suppose the event file is called `myEvents.lhe`.[32] We will assume it contains some process of interest at the LHC. First, a *LesHouchesReader object needs to be created and given the name of the file:

```
cd /Herwig/EventHandlers
create ThePEG::MadGraphReader myReader
set myReader:FileName myEvents.lhe
```

In principle, the information needed to generate full events, *i.e.* beam energies, incoming particles and parton distributions, is extracted from the event file, but may also be set explicitly. For these switches, see the interface documentation of the LesHouchesReader and MadGraphReader classes, respectively.

In case files with unweighted events not generated by MadEvent are used, the LesHouchesReader needs to be assigned an event cache to gain information on the event sample. If, for example, events should be cached in a file named `cacheevents.tmp` the following setting should be used:

```
set myReader:CacheFileName cacheevents.tmp
```

The cuts on the hard process cannot, in general, be extracted from event files. If the interface value

```
set myReader:InitCuts 0
```

is assigned, the LesHouchesReader object expects to be given a Cuts object. For example, typical cuts for hadron collisions may be chosen:

```
set myReader:Cuts /Herwig/Cuts/QCDCuts
```

The use of cuts in Herwig++ and examples of changing them are given in Appendices 3.3.3 and B.3, respectively. If no Cuts object is assigned, the Cuts object assigned to the LesHouchesEventHandler is used.

Similar remarks apply to the use of parton distribution functions, which can be set explicitly using

```
set myReader:InitPDFs 0
set myReader:PDFA firstBeamPDF
set myReader:PDFB secondBeamPDF
```

where `firstBeamPDF` and `secondBeamPDF` are PDFBase objects. Here, either the built-in PDF set or LHAPDF may be used, see Appendix B.9.

Next a LesHouchesEventHandler object has to be created. Objects of this class inherit from EventHandler and provide the same interfaces. The setup is therefore similar to the setup of a StandardEventHandler object, which needs to be equipped with showering, hadronization and decay handlers:

```
create ThePEG::LesHouchesEventHandler \
myLesHouchesHandler
set myLesHouchesHandler:CascadeHandler \
/Herwig/Shower/ShowerHandler
set myLesHouchesHandler:HadronizationHandler \
/Herwig/Hadronization/ClusterHadHandler
set myLesHouchesHandler:DecayHandler \
/Herwig/Decays/DecayHandler
set myLesHouchesHandler:PartonExtractor \
/Herwig/Partons/QCDExtractor
```

A Cuts object that is applied to all processes may be set as for every EventHandler. Finally, the LesHouchesReaders from which the event handler should draw events have to be specified:

```
insert myLesHouchesHandler:LesHouchesReaders 0 \
myReader
insert myLesHouchesHandler:LesHouchesReaders 1 \
myOtherReader
...
```

An arbitrary number of readers may be used.

A default or custom EventGenerator object can use the LesHouchesEventHandler object `myLesHouchesHandler` and a run file can be created from this event generator:

```
cd /Herwig/Generators
cp LHCGenerator myLesHouchesGenerator
set myLesHouchesGenerator:EventHandler \
/Herwig/EventHandlers/myLesHouchesHandler
saverun myLesHouches myLesHouchesGenerator
```

The event generator can then be used as described at the beginning of Appendix B.

B.9 Use of LHAPDF

Herwig++ provides a built-in PDF set.[33] Other PDF sets may be used through the LHAPDF [132] interface ofThePEG. This section contains an outline of the use of LHAPDF.

ThePEG has to be configured to use LHAPDF by adding the option

```
--with-LHAPDF=/path/to/LHAPDF/lib
```

to the call of the `configure` script. Note that the full path to the LHAPDF libraries needs to be given. Once Herwig++ is built using ThePEG configured to use LHAPDF, PDF sets can be created easily in the Repository, for example the CTEQ6L set:

```
create ThePEG::LHAPDF myPDFset
set myPDFset:PDFName cteq6l.LHpdf
set myPDFset:RemnantHandler \
/Herwig/Partons/HadronRemnants
set /Herwig/Particles/p+:PDF myPDFset
set /Herwig/Particles/pbar-:PDF myPDFset
```

The custom PDF set `myPDFset` may also be used in a LesHouchesReader object, see Appendix B.8.

[32] The *LesHouchesReader class is also able to read in compressed event files, `.lhe.gz`.

[33] The default PDF set in Herwig++ is the leading-order fit from the MRST'02 family [131].

B.10 Use of a simple saturation model for PDFs

A very simple modification that mimics parton saturation effects can be applied for any PDF by using the SatPDF class. The modification replaces $xf(x)$ below x_0 by

$$xf(x) \to \left(\frac{x}{x_0}\right)^{\text{Exp}} x_0 f(x_0), \quad \forall x < x_0, \tag{B.3}$$

where **X0** and **Exp** are the changeable parameters. After copying an appropriate `Collider.in` to your local directory, adding the following lines *before* any `run` or `saverun` statement will enable the PDF modifications.

```
################################################
# saturation modifications
################################################
cd /Herwig/Partons
create Herwig::SatPDF SaturationMod HwSatPDF.so
set SaturationMod:RemnantHandler HadronRemnants

## Assign the pdf that should be modified:
## use internal pdf
set SaturationMod:PDF MRST
## use lhapdf. This depends on the name you have
## chosen for the LHAPDF set
#set SaturationMod:PDF foo

## may change X0: default is 1E-4
#set SaturationMod:X0 1E-3

## may change Exp: default is 0
#set SaturationMod:Exp 1

## Assign the modified pdf to the beam particles,
## without this step the original pdf's will be used
set /Herwig/Particles/p+:PDF SaturationMod
set /Herwig/Particles/pbar-:PDF SaturationMod
cd /Herwig/Generators
```

Appendix C: Tuning

The default hadronization and shower parameters in Herwig++ have been tuned to a wide range of experimental data, primarily from LEP and B-factory experiments.

The following experimental data were used, with the exception of charm hadron spectra from the B-factory experiments, all are from e^+e^- experiments operating at the Z^0 peak:

- the momentum spectra of charm hadrons, *i.e.* $D^{*\pm,0}$, $D^{\pm,0}$, $D_s^{\pm}$, and Λ_c^+, measured by the Belle collaboration away from the $\Upsilon(4S)$ resonance, [133];
- the momentum spectra of charm hadrons, *i.e.* $D^{*\pm,0}$ and $D^{\pm 0}$, measured by the CLEO collaboration away from the $\Upsilon(4S)$ resonance, [134];
- the weakly decaying B-hadron fragmentation functions measured by the ALEPH [135] and SLD [136] collaborations;
- four-jet angles measured by the ALEPH collaboration [137];
- the momentum spectrum of charged particles, charged pions, charged kaons and protons for all, light, charm and bottom quark events measured by the SLD collaboration [138];
- the momentum spectra for the production of $\pi^{\pm}$ [139], $K^{\pm}$ [139], p [139], Δ^{++} [140], Ξ^{*0} [141], f_2 [142], $f_0(980)$ [142], ϕ [142], K^{*0} [143], K^0 [144], π^0 [145], η [145], η' [145], $\rho^{\pm}$ [145], ω [145], $a_0^{\pm}$ [145], Ξ^- [141], $\Sigma^{*\pm}$ [141], measured by the OPAL collaboration;
- the multiplicity of charged particles measured by the OPAL collaboration [146];
- the momentum spectra for the production of ρ^0 [147] and D^0 [148] measured by the DELPHI collaboration;
- the momentum spectrum of $D^{*\pm}$ mesons measured by the ALEPH collaboration [149];
- the momentum spectrum of Λ^0 baryons [150] and $K^{*\pm}$ mesons [150] measured by the ALEPH collaboration;
- the differential distributions y_{nm} where an event changes from being an n to an m jet event according to the Durham jet algorithm, jet production rates and the average jet multiplicity as a function of the Durham jet measure measured by the OPAL collaboration [151];
- the differential jet rates with respect to the Durham jet measure measured by the DELPHI collaboration [152];
- the thrust, thrust major, thrust minor, sphericity, oblateness, planarity, aplanarity, C and D parameters, hemisphere masses, and jet broadening event shapes measured by the DELPHI collaboration [152];
- the rapidity, and transverse p_T in and out of the event plane with respect to the thrust and sphericity axes measured by the DELPHI collaboration [152];
- the average multiplicities of charged particles, photons, π^0, ρ^0, π^+, ρ^+, η, ω, f_2, K^0, K^{*0}, K_2^{*0}, K^+, $K*^+$, η', ϕ, f_2', D^+, D^{*+}, D^0, D_s^+, J/ψ, n^0, p^+, Δ^{++}, Σ^-, Σ^{*-}, Λ^0, Σ^0, Σ^+, Σ^{*+}, $\Xi-$, Ξ^{*0}, $\Omega-$, Λ_c^+, f_0',f_1, $\psi(2S)$, a_0^+ taken from the PDG [34];
- the fractions of B^0, $B^{\pm}$ and b-baryons from the Heavy Flavour Averaging Group (HFAG) [153].

The following parameters were tuned:

1. the value of α_S at the Z^0 mass, **AlphaMZ**;
2. the cutoff scale in the parton shower **cutoffKinScale**;
3. the **ConstituentMass** of the gluon used in the hadronization model;
4. the scale **Qmin** below which a non-perturbative treatment of α_S is used, the default is to set α_S to a constant below this scale;
5. the maximum mass **ClMaxLight** above which clusters containing light quarks undergo cluster fission, see (112);
6. the exponent **ClPowLight** controlling whether clusters containing light quarks undergo fission, see (112);

7. the exponent **PSplitLight** controlling the masses of the daughter clusters for light quark clusters that undergo fission, see (113);
8. the **ClSmrLight** parameter, which controls the smearing of the direction of hadrons containing perturbatively produced light quarks, see (125);
9. the weight **PwtSquark** for producing a strange quark-antiquark pair in the hadronization;
10. the weight **PwtDIquark** for producing a diquark-antidiquark pair in the hadronization;
11. the relative weight **SngWt** for the production of singlet baryons;
12. the relative weight **DecWt** for the production of decuplet baryons;
13. the maximum mass **ClMaxCharm** above which clusters containing charm quarks undergo cluster fission, see (112);
14. the exponent **ClPowCharm** controlling whether clusters containing charm quarks undergo fission, see (112);
15. the exponent **PSplitCharm** controlling the masses of the daughter clusters for charm quark clusters that undergo fission, see (113);
16. the **ClSmrCharm** parameter, which controls the smearing of the direction of hadrons containing perturbatively produced charm quarks, see (125);
17. the **SingleHadronLimitCharm** parameter, which controls the splitting of charm clusters to a single hadron above the threshold for producing two hadrons, see (126);
18. the maximum mass **ClMaxBottom** above which clusters containing bottom quarks undergo cluster fission, see (112);
19. the exponent **ClPowBottom** controlling whether clusters containing bottom quarks undergo fission, see (112);
20. the exponent **PSplitBottom** controlling the masses of the daughter clusters for bottom quark clusters that undergo fission, see (113);
21. the **ClSmrBottom** parameter, which controls the smearing of the direction of hadrons containing perturbatively produced bottom quarks, see (125);
22. the **SingleHadronLimitBottom** parameter, which controls the splitting of bottom clusters to a single hadron above the threshold for producing two hadrons, see (126).

The tuning was performed in a number of stages:

- 200,000 events were generated at each of 2000 randomly selected parameter points for the first 7 parameters, which are sensitive to general properties of the events;
- for the values of the first 7 parameters that gave the lowest χ^2 from the first scan 200,000 events were generated for randomly selected values of parameters 8–11, which mainly control the multiplicities of different hadron species;
- for the values of the first 11 parameters that gave the lowest χ^2 from the second scan 200,000 events were generated for randomly selected values of parameters 12–21, which mainly control the production of bottom and charm hadrons;
- the parameters were then scanned about the minimum χ^2 point and the parameter that gave the largest reduction in the χ^2 was adjusted to the value that gave the minimum value;
- the scanning of parameters about the minimum was repeated until no significant improvement was found;
- finally some parameters, particularly in the charm and bottom sector, that are not particularly sensitive to the global χ^2 were adjusted to reduce the χ^2 for observables sensitive to them. In practice the parameters 13–16 were adjusted to improve the quality of the fit to charm hadron multiplicities and spectra, the parameters 17–21 were adjusted to improve the quality of the fit to bottom hadron multiplicities and spectra, and the parameters 10–11 were adjusted to improve the quality of the fit to baryon multiplicities and spectra.

In each case 200,000 events were generated at both the Z^0 pole for the LEP observables and below the $\Upsilon(4S)$ resonance for the B-factory observables. The χ^2 value included all the observables but in order to increase the sensitivity to the particle multiplicities the χ^2 for the total particle multiplicities were multiplied by 10 when computing the global χ^2, and the total charged multiplicities at LEP by 100.

The variation of the χ^2 is shown in Fig. C.1 for some of the parameters that are sensitive to the event shapes and production of hadrons containing the light, *i.e.* down, up and strange, quarks. The best fit point has a $\chi^2 = 6.4$, with the increased weights for the hadron multiplicities and, $\chi^2 = 5.4$ if all observables have unit weight. While this may seem too high a value, given the limited nature of the tuning it is not out of line with previous event generator tunings and the χ^2 is about 4 times lower than before the tuning.

In addition to the above, the option of including an intrinsic transverse momentum for partons within a hadron in hadron-hadron collisions has been implemented. It is chosen from the Gaussian distribution shown in Appendix B.7. For Drell Yan Z/W boson production at the Tevatron ($\sqrt{s} = 1.96$ TeV), the best fit tune has an rms transverse momentum of 2.2 GeV [15]. For the CERN ISR experiment ($\sqrt{s} = 62$ GeV) likewise, a best fit rms value of 0.9 GeV was obtained. Assuming a linear dependence of the rms value on $\ln(M/\sqrt{s})$ where M is the invariant mass, the corresponding value estimated for Z/W boson production at the LHC is within the range 3.7–7.7 GeV. It is worth noting that the lower value of 3.7 GeV gives the best agreement with an alternative model [23], which introduces non-perturbative smearing during the perturbative evolution by modifying the implementation of α_S.

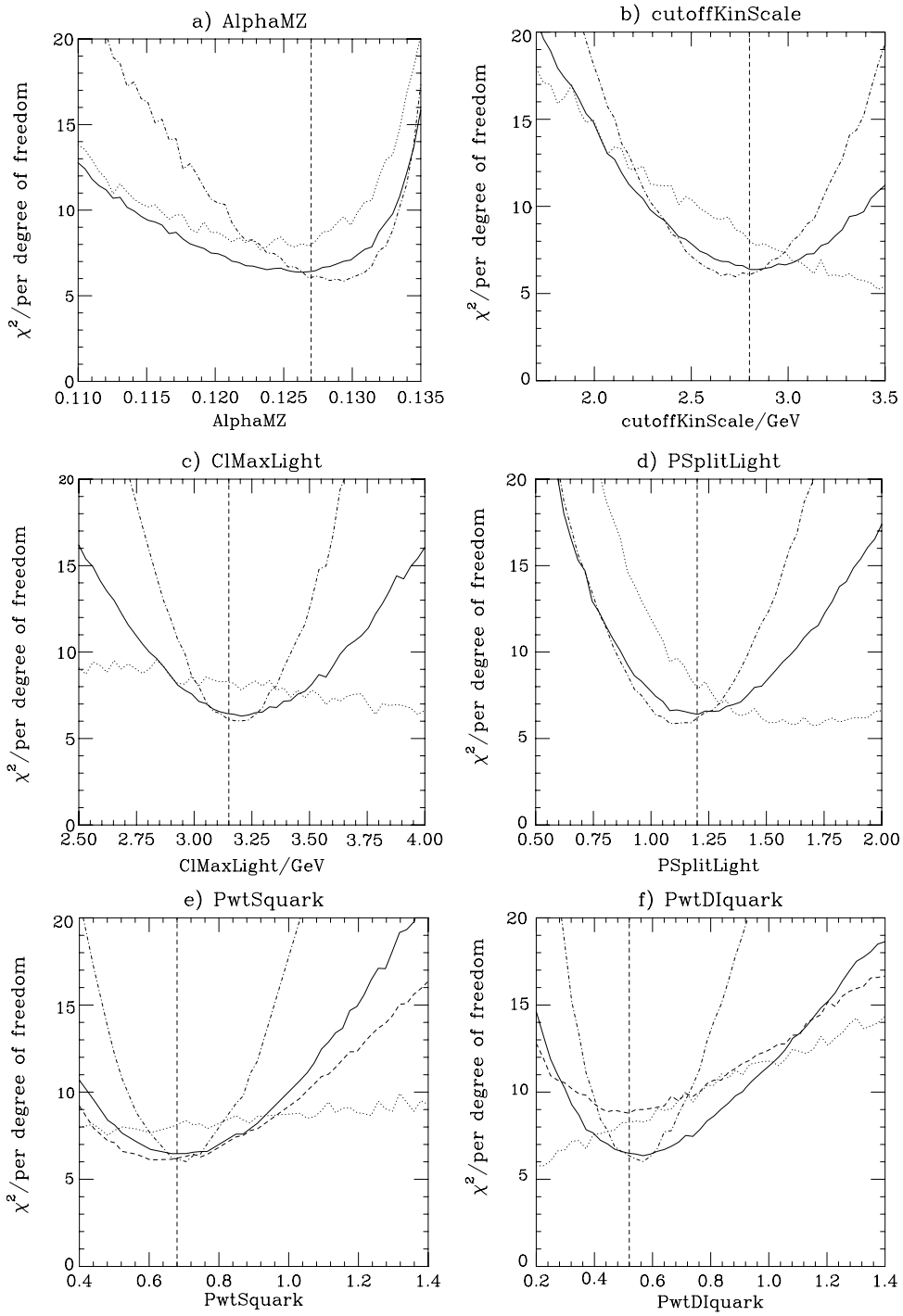


Fig. C.1 Variation of the χ^2 about the minimum points for the (**a**) **AlphaMZ**, (**b**) **cutoffKinScale**, (**c**) **ClMaxLight**, (**d**) **PSplitLight**, (**e**) **PwtSquark**, and (**f**) **PwtDIquark** parameters. The *solid line* shows the total χ^2, the *dot-dashed line* shows the χ^2 for the particle multiplicities and the *dotted line* shows the χ^2 for the event shape observables. In (**e**) the *dashed lines* show the χ^2 for observables sensitive to strange hadron production and in (**f**) the *dashed lines* show the χ^2 for observables sensitive to baryon production. The *vertical dashed lines* show the final values of the parameters, described as default throughout this manual. In each figure, all other parameters are kept at their default values

References

1. G. Marchesini, B.R. Webber, Nucl. Phys. B **38**, 1 (1984)
2. B.R. Webber, Nucl. Phys. B **238**, 492 (1984)
3. G. Marchesini, B.R. Webber, Nucl. Phys. B **310**, 461 (1988)
4. G. Marchesini et al., Comput. Phys. Commun. **67**, 465 (1992)
5. G. Corcella et al., J. High Energy Phys. **01**, 010 (2001), hep-ph/0011363
6. G. Corcella et al. (2002), hep-ph/0210213
7. G. Marchesini, B.R. Webber, Nucl. Phys. B **330**, 261 (1990)
8. M. Bähr, S. Gieseke, M.H. Seymour (2008), arXiv:0803.3633 [hep-ph]
9. S. Gieseke, A. Ribon, M.H. Seymour, P. Stephens, B. Webber, J. High Energy Phys. **02**, 005 (2004), hep-ph/0311208
10. T. Sjöstrand, S. Mrenna, P. Skands (2007), arXiv:0710.3820 [hep-ph]
11. L. Lönnblad, Comput. Phys. Commun. **71**, 15 (1992)
12. L. Lönnblad, Nucl. Instrum. Methods A **559**, 246 (2006)
13. T. Gleisberg et al., J. High Energy Phys. **02**, 056 (2004), hep-ph/0311263

14. O. Latunde-Dada, S. Gieseke, B. Webber, J. High Energy Phys. **02**, 051 (2007), hep-ph/0612281
15. O. Latunde-Dada (2007), arXiv:0708.4390 [hep-ph]
16. J.M. Butterworth, J.R. Forshaw, M.H. Seymour, Z. Phys. C **72**, 637 (1996), hep-ph/9601371
17. S. Gieseke, P. Stephens, B. Webber, J. High Energy Phys. **12**, 045 (2003), hep-ph/0310083
18. S. Gieseke, J. High Energy Phys. **01**, 058 (2005), hep-ph/0412342
19. K. Hamilton, P. Richardson, J. High Energy Phys. **07**, 010 (2006), hep-ph/0603034
20. K. Hamilton, P. Richardson, J. High Energy Phys. **02**, 069 (2007), hep-ph/0612236
21. M. Gigg, P. Richardson, Eur. Phys. J. C **51**, 989 (2007), hep-ph/0703199
22. D. Grellscheid, P. Richardson (2007), arXiv:0710.1951 [hep-ph]
23. S. Gieseke, M.H. Seymour, A. Siódmok (2007), arXiv:0712.1199 [hep-ph]
24. P. Richardson, J. High Energy Phys. **11**, 029 (2001), hep-ph/0110108
25. I.G. Knowles, Nucl. Phys. B **310**, 571 (1988)
26. I.G. Knowles, Comput. Phys. Commun. **58**, 271 (1990)
27. J.C. Collins, Nucl. Phys. B **304**, 794 (1988)
28. E. Boos et al. (2001), hep-ph/0109068
29. J. Alwall et al., Comput. Phys. Commun. **176**, 300 (2007), hep-ph/0609017
30. D.R. Yennie, S.C. Frautschi, H. Suura, Ann. Phys. **13**, 379 (1961)
31. M.H. Seymour, Phys. Lett. B **354**, 409 (1995), hep-ph/9505211
32. H. Murayama, I. Watanabe, K. Hagiwara (1992), KEK-91-11
33. P. Skands et al., J. High Energy Phys. **07**, 036 (2004), hep-ph/0311123
34. W.M. Yao et al. (Particle Data Group), J. Phys. G **33**, 1 (2006)
35. L. Randall, R. Sundrum, Phys. Rev. Lett. **83**, 3370 (1999), hep-ph/9905221
36. D. Hooper, S. Profumo, Phys. Rep. **453**, 29 (2007), hep-ph/0701197
37. H.C. Cheng, K.T. Matchev, M. Schmaltz, Phys. Rev. D **66**, 036005 (2002), hep-ph/0204342
38. A. Bassetto, M. Ciafaloni, G. Marchesini, A.H. Mueller, Nucl. Phys. B **207**, 189 (1982)
39. A. Bassetto, M. Ciafaloni, G. Marchesini, Phys. Rep. **100**, 201 (1983)
40. S. Catani, M. Ciafaloni, Nucl. Phys. B **236**, 61 (1984)
41. M. Ciafaloni, Phys. Lett. B **95**, 113 (1980)
42. M. Ciafaloni, Lectures given at Summer Workshop on High Energy Physics, Trieste, Italy, Aug. 1981
43. Y.L. Dokshitzer, V.A. Khoze, S.I. Troian, Adv. Ser. Direct. High Energy Phys. **5**, 241 (1988)
44. A.H. Mueller, Phys. Lett. B **104**, 161 (1981)
45. B.I. Ermolaev, V.S. Fadin, JETP Lett. **33**, 269 (1981)
46. Y.L. Dokshitzer, V.S. Fadin, V.A. Khoze, Phys. Lett. B **115**, 242 (1982)
47. S. Catani, S. Dittmaier, Z. Trócsányi, Phys. Lett. B **500**, 149 (2001), hep-ph/0011222
48. R.K. Ellis, W.J. Stirling, B.R. Webber, Camb. Monogr. Part. Phys. Nucl. Phys. Cosmol. **8**, 1 (1996)
49. S. Catani, B.R. Webber, G. Marchesini, Nucl. Phys. B **349**, 635 (1991)
50. S. Frixione, P. Nason, C. Oleari, J. High Energy Phys. **11**, 070 (2007), arXiv:0709.2092 [hep-ph]
51. R. Bonciani, S. Catani, M.L. Mangano, P. Nason, Phys. Lett. B **575**, 268 (2003), hep-ph/0307035
52. M. Cacciari, G. Corcella, A.D. Mitov, J. High Energy Phys. **12**, 015 (2002), hep-ph/0209204
53. H.K. Dreiner, P. Richardson, M.H. Seymour, J. High Energy Phys. **04**, 008 (2000), hep-ph/9912407
54. M.J. Gibbs, B.R. Webber, Comput. Phys. Commun. **90**, 369 (1995), hep-ph/9504232
55. T. Sjöstrand, S. Mrenna, P. Skands, J. High Energy Phys. **05**, 026 (2006), hep-ph/0603175
56. T. Sjöstrand, Phys. Lett. B **157**, 321 (1985)
57. A. Bassetto, G. Nardelli, R. Soldati, *Yang–Mills Theories in Algebraic Non-covariant Gauges* (World Scientific, Singapore, 1991), p. 227
58. M. Dalbosco, Phys. Lett. B **180**, 121 (1986)
59. D. Amati, A. Bassetto, M. Ciafaloni, G. Marchesini, G. Veneziano, Nucl. Phys. B **173**, 429 (1980)
60. D. Amati, G. Veneziano, Phys. Lett. B **83**, 87 (1979)
61. S. Catani, M.H. Seymour, Nucl. Phys. B **485**, 291 (1997), hep-ph/9605323
62. G. Curci, M. Greco, Phys. Lett. B **92**, 175 (1980)
63. G. Curci, M. Greco, Phys. Lett. B **102**, 280 (1981)
64. M.H. Seymour, Comput. Phys. Commun. **90**, 95 (1995), hep-ph/9410414
65. A. Kupčo (1999), hep-ph/9906412
66. W. Kilian, T. Plehn, P. Richardson, E. Schmidt, Eur. Phys. J. C **39**, 229 (2005), hep-ph/0408088
67. G.J. Alner et al. (UA5), Nucl. Phys. B **291**, 445 (1987)
68. L. Durand, P. Hong, Phys. Rev. Lett. **58**, 303 (1987)
69. L. Durand, H. Pi, Phys. Rev. D **40**, 1436 (1989)
70. D. Grellscheid, K. Hamilton, P. Richardson, in preparation
71. D.J. Lange, Nucl. Instrum. Methods A **462**, 152 (2001)
72. V.V. Kiselev (2003), hep-ph/0308214
73. T. Skwarnicki, Int. J. Mod. Phys. A **19**, 1030 (2004), hep-ph/0311243
74. E.J. Eichten, K. Lane, C. Quigg, Phys. Rev. Lett. **89**, 162002 (2002), hep-ph/0206018
75. W. Kwong, J.L. Rosner, Phys. Rev. D **38**, 279 (1988)
76. S. Godfrey, J.L. Rosner, Phys. Rev. D **66**, 014012 (2002), hep-ph/0205255
77. E.J. Eichten, C. Quigg, Phys. Rev. D **49**, 5845 (1994), hep-ph/9402210
78. D. Ebert, R.N. Faustov, V.O. Galkin, Phys. Rev. D **67**, 014027 (2003), hep-ph/0210381
79. W. Kwong, P.B. Mackenzie, R. Rosenfeld, J.L. Rosner, Phys. Rev. D **37**, 3210 (1988)
80. W.A. Bardeen, E.J. Eichten, C.T. Hill, Phys. Rev. D **68**, 054024 (2003), hep-ph/0305049
81. M. Di Pierro, E. Eichten, Phys. Rev. D **64**, 114004 (2001), hep-ph/0104208
82. V.M. Abazov et al. (2007), arXiv:0705.3229 [hep-ex]
83. F. Filthaut (D0) (2007), arXiv:0705.0245 [hep-ex]
84. N. Brambilla et al. (2004), hep-ph/0412158
85. S. Godfrey, Phys. Rev. D **70**, 054017 (2004), hep-ph/0406228
86. S.M. Flatté, Phys. Lett. B **63**, 224 (1976)
87. S. Jadach, Z. Wąs, R. Decker, J.H. Kühn, Comput. Phys. Commun. **76**, 361 (1993)
88. P. Golonka et al. (2003), hep-ph/0312240
89. J.H. Kühn, A. Santamaria, Z. Phys. C **48**, 445 (1990)
90. G.J. Gounaris, J.J. Sakurai, Phys. Rev. Lett. **21**, 244 (1968)
91. M. Finkemeier, E. Mirkes, Z. Phys. C **72**, 619 (1996), hep-ph/9601275
92. R. Decker, E. Mirkes, R. Sauer, Z. Wąs, Z. Phys. C **58**, 445 (1993)
93. D.M. Asner et al. (CLEO), Phys. Rev. D **61**, 012002 (2000), hep-ex/9902022
94. M. Finkemeier, E. Mirkes, Z. Phys. C **69**, 243 (1996), hep-ph/9503474
95. A.E. Bondar et al., Comput. Phys. Commun. **146**, 139 (2002), hep-ph/0201149
96. J.H. Kühn, Z. Wąs (2006), hep-ph/0602162
97. R. Kleiss, W.J. Stirling, Nucl. Phys. B **385**, 413 (1992)

98. B.R. Holstein, Phys. Scripta T **99**, 55 (2002), hep-ph/0112150
99. E.P. Venugopal, B.R. Holstein, Phys. Rev. D **57**, 4397 (1998), hep-ph/9710382
100. N. Beisert, B. Borasoy, Nucl. Phys. A **716**, 186 (2003), hep-ph/0301058
101. M. Gormley et al., Phys. Rev. D **2**, 501 (1970)
102. W.B. Tippens et al. (Crystal Ball), Phys. Rev. Lett. **87**, 192001 (2001)
103. L.S. Brown, R.N. Cahn, Phys. Rev. Lett. **35**, 1 (1975)
104. J.Z. Bai et al. (BES), Phys. Rev. D **62**, 032002 (2000), hep-ex/9909038
105. D. Cronin-Hennessy et al. (CLEO) (2007), arXiv:0706.2317 [hep-ex]
106. B. Aubert et al. (BABAR), Phys. Rev. Lett. **96**, 232001 (2006), hep-ex/0604031
107. N.E. Adam et al. (CLEO), Phys. Rev. Lett. **96**, 082004 (2006), hep-ex/0508023
108. A. Aloisio et al. (KLOE), Phys. Lett. B **561**, 55 (2003), hep-ex/0303016
109. T. Han, J.D. Lykken, R.J. Zhang, Phys. Rev. D **59**, 105006 (1999), hep-ph/9811350
110. A. Ore, J.L. Powell, Phys. Rev. **75**, 1696 (1949)
111. P. Ball, R. Zwicky, Phys. Rev. D **71**, 014015 (2005), hep-ph/0406232
112. P. Ball, R. Zwicky, Phys. Rev. D **71**, 014029 (2005), hep-ph/0412079
113. I. Caprini, L. Lellouch, M. Neubert, Nucl. Phys. B **530**, 153 (1998), hep-ph/9712417
114. B. Aubert et al. (BABAR) (2007), arXiv:0705.4008 [hep-ex]
115. A.E. Snyder (2007), hep-ex/0703035
116. N. Isgur, D. Scora, B. Grinstein, M.B. Wise, Phys. Rev. D **39**, 799 (1989)
117. D. Scora, N. Isgur, Phys. Rev. D **52**, 2783 (1995), hep-ph/9503486
118. N. Isgur, M.B. Wise, Phys. Rev. D **43**, 819 (1991)
119. D. Scora, N. Isgur, Phys. Rev. D **40**, 1491 (1989)
120. V.V. Kiselev (2002), hep-ph/0211021
121. D. Melikhov, Phys. Lett. B **380**, 363 (1996), hep-ph/9603340
122. D. Melikhov, B. Stech, Phys. Rev. D **62**, 014006 (2000), hep-ph/0001113
123. M. Wirbel, B. Stech, M. Bauer, Z. Phys. C **29**, 637 (1985)
124. M. Bauer, B. Stech, M. Wirbel, Z. Phys. C **34**, 103 (1987)
125. H. Muramatsu et al. (CLEO), Phys. Rev. Lett. **89**, 251802 (2002), hep-ex/0207067
126. S. Kopp et al. (CLEO), Phys. Rev. D **63**, 092001 (2001), hep-ex/0011065
127. J.C. Anjos et al. (E691), Phys. Rev. D **48**, 56 (1993)
128. A.L. Kagan, M. Neubert, Eur. Phys. J. C **7**, 5 (1999), hep-ph/9805303
129. R. Brun, F. Rademakers, Nucl. Instrum. Meth. A **389**, 81 (1997)
130. F. Maltoni, T. Stelzer, J. High Energy Phys. **02**, 027 (2003), hep-ph/0208156
131. A.D. Martin, R.G. Roberts, W.J. Stirling, R.S. Thorne, Phys. Lett. B **531**, 216 (2002), hep-ph/0201127
132. M.R. Whalley, D. Bourilkov (R.C. Group) (2005), hep-ph/0508110
133. R. Seuster et al. (Belle), Phys. Rev. D **73**, 032002 (2006), hep-ex/0506068
134. M. Artuso et al. (CLEO), Phys. Rev. D **70**, 112001 (2004), hep-ex/0402040
135. A. Heister et al. (ALEPH), Phys. Lett. B **512**, 30 (2001), hep-ex/0106051
136. K. Abe et al. (SLD), Phys. Rev. D **65**, 092006 (2002), hep-ex/0202031
137. A. Heister et al. (ALEPH), Eur. Phys. J. C **27**, 1 (2003)
138. K. Abe et al. (SLD), Phys. Rev. D **59**, 052001 (1999), hep-ex/9805029
139. R. Akers et al. (OPAL), Z. Phys. C **63**, 181 (1994)
140. G. Alexander et al. (OPAL), Phys. Lett. B **358**, 162 (1995)
141. G. Alexander et al. (OPAL), Z. Phys. C **73**, 569 (1997)
142. K. Ackerstaff et al. (OPAL), Eur. Phys. J. C **4**, 19 (1998), hep-ex/9802013
143. K. Ackerstaff et al. (OPAL), Phys. Lett. B **412**, 210 (1997), hep-ex/9708022
144. G. Abbiendi et al. (OPAL), Eur. Phys. J. C **17**, 373 (2000), hep-ex/0007017
145. K. Ackerstaff et al. (OPAL), Eur. Phys. J. C **5**, 411 (1998), hep-ex/9805011
146. P.D. Acton et al. (OPAL), Z. Phys. C **53**, 539 (1992)
147. P. Abreu et al. (DELPHI), Phys. Lett. B **449**, 364 (1999)
148. P. Abreu et al. (DELPHI), Z. Phys. C **59**, 533 (1993)
149. R. Barate et al. (ALEPH), Eur. Phys. J. C **16**, 597 (2000), hep-ex/9909032
150. R. Barate et al. (ALEPH), Phys. Rept. **294**, 1 (1998)
151. P. Pfeifenschneider et al. (JADE), Eur. Phys. J. C **17**, 19 (2000), hep-ex/0001055
152. P. Abreu et al. (DELPHI), Z. Phys. C **73**, 11 (1996)
153. http://www.slac.stanford.edu/xorg/hfag/osc/PDG_2007