MASTER THESIS PHYSICS

A fast simulation model for mass spectrum shapes of particle decays in high-energy physics experiments

Author
Jim Baarslag

Supervisor
dr. ir. C.J.G. Onderwater

Co-assessor
prof. dr. A. Pellegrino

July 2017

University of Groningen
Faculty of Science and Engineering
Van Swinderen Institute for Particle Physics and Gravity

Abstract

We have developed a framework for the Monte Carlo simulation of experimental mass spectra in high-energy physics experiments. Contrary to existing simulation software, our framework has focused primarily on computational speed, rather than on extreme accuracy. A sufficient level of accuracy has been ensured, however, by including those elements that are identified to be the most essential for determining the shapes of these mass spectra, and by excluding all elements of which the effects are deemed only marginal.

Specifically, the resulting simulation model includes decay kinematics and the most important detection effects: acceptance, resolution and bremsstrahlung effects. Moreover, it can generate background shapes for peaking or semi-peaking backgrounds by simulating the effects of particle misidentifications or of the partial reconstruction of events.

In addition, a method for calibrating the free parameters in the simulation model to experimental datasets, based on simulated annealing, is presented. This calibration procedure allows the model to be adapted to any general high-energy collider experiment. Finally, we also discuss an application of the model, where the simulation framework is used for fitting background-dominated experimental mass spectra.

Contents

Chapter 1

Introduction

The Standard Model is one of the greatest successes of modern physics, as it is able to accurately predict and describe practically all phenomena observed at particle physics experiments. For example, some of its accomplishments include the prediction of the heavy quarks, the gluons, the W and Z bosons and the Higgs boson; and numerical predictions for e.g. the anomalous magnetic moment of the electron, and the W and Z masses up to extreme accuracies [1]. In addition, the Standard Model makes very precise predictions on what (elementary or composite) particles can be observed, what their properties are, and how they interact with each other.

However, the Standard Model is also inherently incomplete, and many open questions remain. As theoretical shortcomings, for instance, the Standard Model is unable to describe gravity, and currently lacks a mechanism for incorporating non-zero neutrino masses [2]. In addition, there are also several (mostly cosmological) observational facts that the Standard Model is unable to explain. Most importantly, the Standard Model seems to account for only 5 percent [3] of the total energy in the universe—what constitutes the other 95 percent of the energy budget is still completely unknown—and does not offer an explanation for the observed asymmetry between the abundances of matter and antimatter. Finally, many unsatisfactory aesthetic issues remain that also hint at the existence of a more complete theory: the fact that the model has many (20+) free parameters, i.e. parameters that are all unrelated; that these parameters differ unnaturally many orders of magnitude from each other (this is in particular the case for the particle masses and the gauge couplings); and that these parameters must be fine-tuned to each other to an extreme extent to generate the physics that we observe [1, 4].

To fix these shortcomings, many Standard Model extensions have been (and are being) proposed, including additional particles, interactions, or symmetries (see e.g. Refs. [5–16]). Now, although many of these extensions apply to energy scales (either very low or very high) that cannot be directly probed at particle physics experiments, most of these models do give rise to effects that could be indirectly measurable at e.g. high-energy colliders. For example, such theories may predict CP violation, charged lepton flavour violation, or otherwise affect interaction cross sections, and these effects may be measurable in general high-energy physics experiments [17–24]. Such experiments then allow us to indirectly probe these Standard Model extensions.

However, such predicted deviations from the Standard Model are often of relatively small magnitudes (otherwise we would have found or falsified them already), and therefore require high ex-

perimental accuracy and sensitivity. In the context of collider experiments, on which this thesis focuses, this generally means that experiments search for extremely rare events, or search for extremely small deviations from precisely calculable Standard Model parameters.

In such cases, it is important that systematic uncertainties are sufficiently minimized. This, in turn, requires a thorough (a priori) understanding of both the signal that the new physics theory would give rise to, and of the possible backgrounds that may contaminate the signal. This latter point is particularly important in searches for rare processes, as in such experiments the backgrounds are generally of (at least) the same order of magnitude as the actual signal itself.

For that reason, in many of such high-precision collider experiments, extensively simulating the expected signal and background shapes is a standard part of the analysis. In addition, such simulations can be used for preliminary studies, e.g. for exploring what types of interactions would be most interesting to study further, how sensitive the experiment is to certain processes, or what level of precision is attainable for a given experiment. As such, appropriate simulation tools are invaluable for high-precision experiments in high-energy particle physics.

At present, already a comprehensive set of simulation software is available. For typical LHCb[1] analyses, for example, Pythia [25] is used for simulating the proton-proton collisions; EvtGen [26] is used for simulating the decays of the particles produced in the collisions; Geant [27] is used for simulating the interactions of the produced particles with the detector; and Boole [28] is used for simulating the detector response.

Now, these simulation packages focus on maximizing accuracy, at the expense of computational speed. As a result, these simulations are indeed accurate, but slow: given that simulating a single event requires over a minute of computation [29], a complete Monte Carlo dataset that can be used for analyses may easily take in the order of weeks to generate. Of course, for high-precision measurements, accuracy should have the highest priority, and these simulation tools are indeed adequate. On the other hand, for many other applications, e.g. for exploratory studies, where the objective is to gain quick insights rather than to achieve the highest accuracy, these tools are hardly appropriate. Instead, for such purposes a tool that focuses on computational speed rather than extreme accuracy is desirable.

The objective of our research is therefore to develop a framework for simulating signal shapes (in particular, invariant mass spectra) and backgrounds as observed at high-energy physics experiments, that explicitly focuses on computational speed. In particular, this is done by including only the most essential experimental features in our model, i.e. those features that have the largest effects on the observed signal, and exclude any features that are deemed to have only marginal effects. This way, we can build a model that is indeed fast—it takes seconds instead of weeks to generate a dataset of sufficiently many events—while also ensuring a sufficient level of accuracy.

Specifically, the resulting model consists of the following elements. First, we must generate particle decays according to some channel of interest, which is done by making draws from phase space. Those particles are then subjected to three essential detection effects: resolution effects, acceptance effects and, in the case of electrons and positrons, bremsstrahlung energy loss effects. The resulting four-momenta of the secondary particles (i.e. the decay products) are then used to

[1]This thesis research has been conducted within the LHCb collaboration. As such, for reasons of availability of data and expertise, throughout this thesis many examples are taken from the LHCb experiment. It should however be noted that the findings of this thesis can be generalized to other high-energy physics experiments as well.

reconstruct the invariant mass of the primary particle (i.e. the parent particle), from which we obtain our simulated mass spectrum. A similar procedure can be followed for generating background contributions.

This thesis is structured as follows. In Chapter 2 we discuss the Standard Model phenomenology as studied at high-energy physics experiments. This chapter particularly focuses on the types of particles that exist in the Standard Model, and on how they interact and decay; as such, it provides a comprehensive context in which searches for new physics can be described and motivated. In Chapter 3 we describe the setup of the LHCb experiment, to sketch how measurements and analyses are done in a typical high-precision high-energy collider experiment. Subsequently, our simulation model is discussed: Chapter 4 covers particle decays; Chapter 5 deals with resolution and acceptance effects; Chapter 6 treats the fast simulation of bremsstrahlung energy loss; and Chapter 7 addresses the simulation of backgrounds. After that, in Chapter 8 a method is presented for calibrating the model parameters to real data, which is essential if the simulation model is to be used in the context of a specific experiment. Chapter 9 then treats an application of our model for making fits to data. Finally, Chapter 10 presents the final conclusions.

Chapter 2

Theory and phenomenology[1]

2.1 The Standard Model

The Standard Model (SM) of particle physics is a quantum field theory that describes the behaviour of elementary particles in nature. The currently known elementary particles are tabulated with their most important properties in Figure 2.1, and can be divided into four categories: quarks, leptons, gauge bosons and Goldstone bosons.

Roughly stated, quarks and leptons make up matter, whereas the bosons mediate interactions between matter particles (and among themselves). This section describes these particles and their interactions in more detail. It also describes the need for hadrons, which are the particles commonly studied at particle physics experiments, and it sets the stage for Sections 2.2 and 2.3, in which the properties of hadrons are discussed; in particular, these sections discuss the spectrum of existing hadrons, and their decay modes.

It should be noted in advance that this chapter, and in particular this first section, is rather extensive. Instead of just stating and summarizing experimental observations—which to a large extent would suffice for our purposes—we approach the Standard Model phenomenology from a theoretical perspective. The reason for this is twofold. First, a treatment from first principles is simply more informative than a list of phenomena, and better allows us to determine what assumptions and simplifications (either implicit or explicit) are made in our simulation model. Second, it provides a more complete framework on which specific searches for beyond Standard Model physics can build; for instance, within this framework it becomes relatively straightforward to determine why certain interactions (such as charged lepton flavour violating or CP violating interactions, or the production of exotic states) are suppressed or forbidden by the Standard Model, and how such hypothetical interactions should be simulated.

2.1.1 Elementary matter particles

Quarks and leptons all have spin-$\frac{1}{2}$ and therefore obey Fermi-Dirac statistics. As such, no two quarks or leptons can be in the same quantum state, so that identical quarks or leptons effectively

[1] All information found in this chapter can be found in e.g. Refs. [30–34], unless stated otherwise.

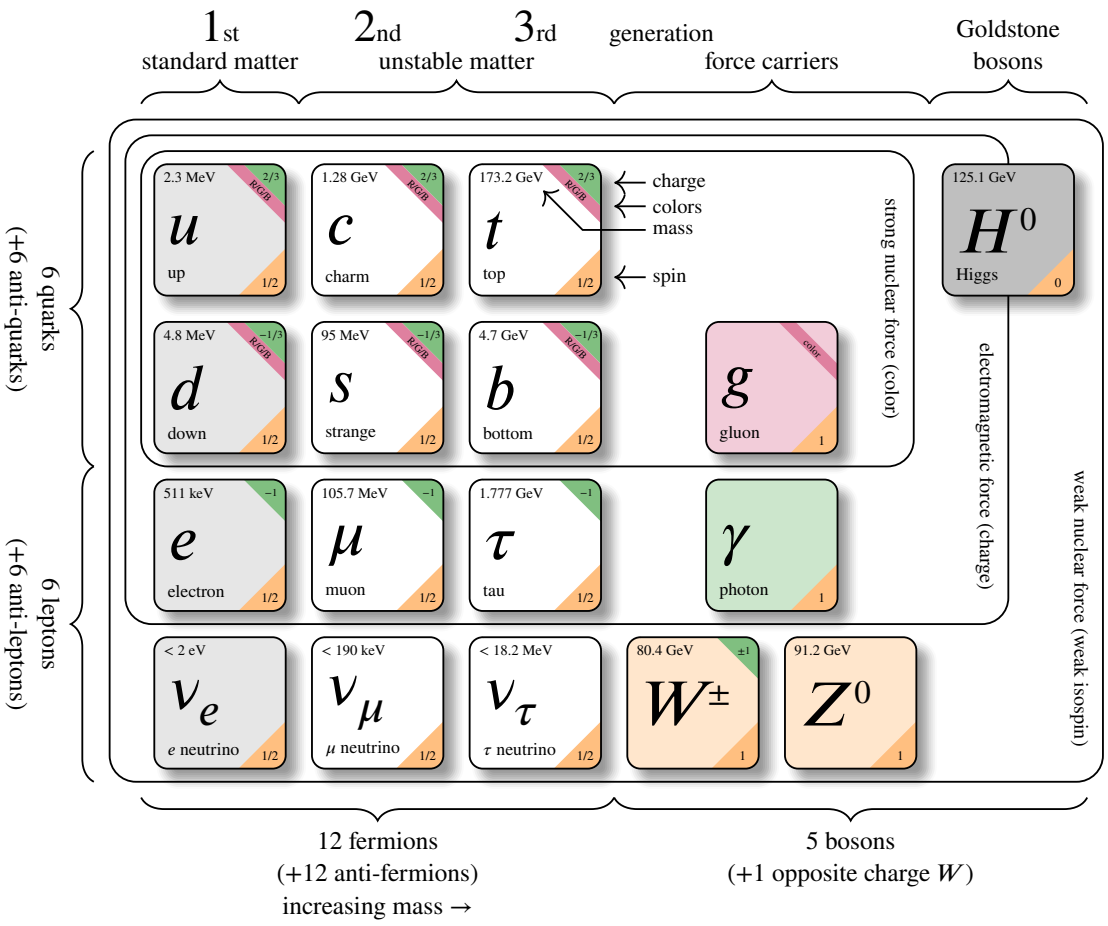

Figure 2.1
The Standard Model of particle physics. Adapted from Ref. [35].

repel each other and are thus forced to take up a finite amount of space. As a result, fermions are able to form matter structures that do not collapse, as would be the case with bosons.

There are three generations of quarks and leptons. The first generation contains the lightest particles, which make up all stable matter; the other generations contain heavier particles, for which it will eventually be energetically favourable to decay to the lighter particles of the first generation. Alternatively stated, only the first generation contains fermions which (are thought to) have an infinite lifetime.

In experimental particle physics, such decays are a primary subject of study, because the process via which a particle decays tells us about the properties of the decaying particle, but also about the properties of the interaction mediating the decay. We now turn to the different categories of elementary matter particles and the interactions that they have, as found from experiments.

Charged leptons

Charged leptons come in three flavours: the electron, muon and tau. For each flavour there is a negatively charged particle and a positively charged antiparticle, so that in total we have six different charged leptons in the Standard Model.

Of the charged leptons, the electron and positron form the only stable flavour. The (anti-)muon has a lifetime of 2.2×10^{-6} s [36], which is extremely long compared with lifetimes of other unstable particles; in the context of experiments, a muon produced at high energy may travel a distance of about $c\tau_\mu = 659$ m, making it perhaps the easiest particle to detect. In comparison, the lifetime of the tau is 2.9×10^{-13} s, so that a high-energy tau travels about $c\tau_\tau = 87$ μm [36].

With each flavour, a flavour quantum number is associated. For example, the electron has electron lepton number $L_e = +1$; the positron has $L_e = -1$; and the other flavours (and all non-lepton particles as well) have $L_e = 0$. In a similar fashion, we have the muon and tau lepton numbers.

In the Standard Model, these lepton numbers are conserved by all interactions. This means that lepton flavour changing interactions such as $\mu \to e\gamma$ are forbidden (initially we have $L_e = 0$ and $L_\mu = +1$, while afterwards we have $L_e = +1$ and $L_\mu = 0$), and that observing such an interaction would with certainty indicate physics beyond the Standard Model. Indeed, it is the existence of such lepton number conservation violating interactions that is a primary research subject of the LHCb experiment (and many other experiments).

Neutrinos

Neutral leptons, neutrinos, come in the same flavours as the charged leptons, and have the same lepton quantum numbers associated with them. That is, the electron-neutrino has the same electron lepton number as the electron (i.e. ν_e has $L_e = +1$ and $\bar{\nu}_e$ has $L_e = -1$); and the same holds for the muon and tau flavours.

Interestingly, so far only neutrinos of left-handed chirality have been observed, and it is unknown whether right-handed neutrinos exist as well. It is however clear that while left-handed neutrinos interact only weakly, right-handed neutrinos interact neither via the electromagnetic, strong nor weak forces.

Initially, the neutrinos were hypothesized to be massless. However, experimental observations indicate that the flavour of a neutrino may change over time; for example, an electron-neutrino may transition into a muon-neutrino via a two-point interaction (i.e. without some mediating particle exchange—see Figure 2.2).[2] This has two implications: at least two of the three neutrino flavours must be massive; and lepton flavour numbers are apparently not conserved for neutral leptons.

[2]In terms of quantum field theory, neutrino oscillations can only occur if the flavour and mass eigenstates are different. This would imply that the state in which a neutrino is produced (i.e. a flavour eigenstate), is not a pure state in the basis in which it propagates (i.e. the basis spanned by mass eigenstates). Consequently, a neutrino is produced in a superposition of mass eigenstates, which propagate with different frequencies, thus giving rise to neutrino flavour oscillations.

Quarks

There are six quark flavours with varying masses and electric charges: the up, charm and top quark have electric charge $+1/3$, and the down, strange and bottom quark have charge $-2/3$. Their antiparticles have opposite charges, as usual.

The masses of the up, down and strange quarks are of the same order of magnitude (order MeV), whereas the charm, bottom and top have masses exceeding many GeV. In fact, the top mass is the heaviest particle in the Standard Model at 173 GeV, and due to its mass it has a lifetime of only 5×10^{-25} s [36]. Because this lifetime is shorter than the time scales of any of the interactions, a produced top quark will generally decay before it can interact with any other particle [37].

Quarks are the only fermions in the Standard Model that interact strongly with each other, implying that they have colour charges. These colour charges are often labelled r (red), g (green), and b (blue); or $\bar{r}$ (anti-red), $\bar{g}$ (anti-green), and $\bar{b}$ (anti-blue) for the antiparticles.

In nature, only colour-neutral combinations of quarks are found, which can be combinations of e.g. $r+g+b=0$ or $r+\bar{r}=0$. This observation has led to the colour confinement hypothesis, which states that quarks are confined to bound states that have no colour charge. For this reason, quarks cannot be investigated directly, and their properties are experimentally derived from the properties of composite particles consisting of multiple quarks and/or anti-quarks.[3] Such particles, referred to as hadrons, are discussed in Section 2.2 below. Confinement is a feature of the strong interaction, and will be treated more in-depth in Section 2.1.2.

2.1.2 Fundamental interactions and gauge bosons

The elementary particles described above are all fermions, and therefore obey the Dirac Lagrangian density[4]

$$\mathcal{L} = \bar{\psi}\left(i\gamma^\mu\partial_\mu - m\right)\psi, \tag{2.2}$$

where each particle is associated with a field $\psi = \psi(x)$, the conjugate field $\bar{\psi} \equiv \psi^\dagger\gamma^0$, γ^μ are the gamma matrices obeying the Clifford algebra $\{\gamma^\mu, \gamma^\nu\} = 2g^{\mu\nu}$ with $g^{\mu\nu}$ the Minkowski metric, and

Figure 2.2
Neutrino oscillation from ν_e to ν_μ. The flavour transition is marked with a cross.

[3] A notable exception is the study of quark-gluon plasmas (QGP), which may exist under certain extreme circumstances (most notably, high pressure and high temperature are required). In QGPs, quarks are deconfined and behave like an ideal gas [38], and are thus of special interest for studying quarks and the strong interaction. QGPs are generally produced in relativistic heavy ion collisions; in particular in Pb+Pb collisions at the LHC [39] and in Au+Au collisions at the RHIC [40].

[4] This is at least assumed to be true for charged leptons and (free) quarks, but for neutrinos this is still unclear: if neutrinos are Majorana particles, they are their own antiparticles, and they obey the Majorana equation

$$-i\gamma^\mu\partial_\mu\psi + m\psi_c = 0, \tag{2.1}$$

where ψ is the fermion field, $\psi_c \equiv i\psi^*$, and m is the Majorana mass. For this text, it will not be relevant whether neutrinos are Dirac or Majorana fermions, and for simplicity they are therefore assumed to be Dirac particles.

m is the particle's Dirac mass.

In Yang-Mills theory, the general recipe for constructing a Lagrangian that describes a field of fermions (ψ) that interact via a field of gauge bosons (A^μ) is the following. One starts by defining a symmetry group G for the fields under which the Lagrangian $\mathcal{L}$ is locally invariant; that is, if $U = U(x)$ is an element of this symmetry group, the field transformation

$$\psi \to \psi' = U\psi \tag{2.3}$$

$$\bar{\psi} \to \bar{\psi}' = \bar{\psi}U^\dagger \tag{2.4}$$

must leave the Lagrangian density invariant. Evidently, this is easily accomplished for the Dirac Lagrangian (2.2) by taking the symmetry group to be unitary ($U^\dagger = U^{-1}$ for all $U \in G$) and replacing any derivatives ∂_μ of the field by covariant derivatives D_μ which are defined such that

$$D_\mu\psi \to D_\mu\psi' = D_\mu(U\psi) = U D_\mu\psi. \tag{2.5}$$

Of course, since U is x-dependent, this promotion is not trivial; rather, it requires the introduction of additional terms. These terms should also depend on x to ensure the correct covariance, and must also transform under the operations of the group. In fact, it can be shown that the correct transformation properties are obtained for any unitary group G of dimension N [i.e. U(N) or SU(N)] with generators T_a by introducing an additional gauge field A^a_μ for each generator, and imposing [32]

$$D_\mu = \partial_\mu - igA^a_\mu T_a \tag{2.6}$$

and

$$A^a_\mu T_a \to A'^a_\mu T_a = U A^a_\mu T_a U^{-1} - \frac{i}{g}(\partial_\mu U)U^{-1}. \tag{2.7}$$

It is exactly these transformations that introduce the gauge fields A^a_μ and the terms $g\bar{\psi}\gamma^\mu A^a_\mu T_a\psi$ into the Lagrangian density that are responsible for the coupling (with strength g) between the fermion fields ψ and the gauge fields.

In addition, the total Lagrangian density should then also contain a kinetic term for the gauge fields. The only form for this term that is in agreement with theoretical requirements and empirical observations is [32]

$$\mathcal{L}_{\text{gauge}} = -\frac{1}{4}F^a_{\mu\nu}F^{a,\mu\nu} \tag{2.8}$$

with

$$F^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + g f^{abc} A^b_\mu A^c_\nu, \tag{2.9}$$

where f^{abc} are the structure constants of G (i.e. $[T_a, T_b] = f_{ab}{}^c T_c$).

The resulting Yang-Mills Lagrangian density then reads

$$\mathcal{L}_{\text{YM}} = \bar{\psi}\left(i\gamma^\mu D_\mu - m\right)\psi - \frac{1}{4}F^a_{\mu\nu}F^{a,\mu\nu}, \tag{2.10}$$

where D_μ and $F^a_{\mu\nu}$ are determined by the structure of the gauge group G.

This is a very general procedure for constructing Lagrangians for the forces in the Standard Model.[5] In particular, in this framework, the electromagnetic force has gauge group U(1), the

[5]Except, of course, for gravity, for which we have no consistent description yet. Since the effects of gravity are (thought to be) negligible at the energy scales considered in this thesis, gravity is not discussed any further.

strong force follows from SU(3) and the weak force is described by SU(2); although still several extensions are required to include all phenomenology, especially for the latter interaction.[6] We will now treat each of these forces, and the particular extensions that they require, in more detail.

Electromagnetic interaction

The electromagnetic (EM) interaction is mediated by a massless spin-1 boson, the photon. All particles that have a non-zero electric charge will interact with each other via the EM force, where particles with opposite charges attract each other, and those with equal charges repel each other. The photon itself does not have an electric charge, so the photon field will not interact with itself.

In terms of field theory, the EM interaction is described by quantum electrodynamics (QED), a Yang-Mills theory with U(1) gauge symmetry. Its Lagrangian density is [31]

$$\mathcal{L}_{\text{QED}} = \bar{\psi}(i\gamma^\mu D_\mu - m)\psi - \frac{1}{4}F_{\mu\nu}F^{\mu\nu} \tag{2.11}$$

with

$$D_\mu = \partial_\mu + ieQA_\mu \tag{2.12}$$

and

$$F_{\mu\nu} = \partial_\mu A_\nu - \partial_\nu A_\mu, \tag{2.13}$$

where $e = \sqrt{4\pi\alpha} = 0.024$ is the (dimensionless) coupling, A^μ is the photon field, and Q is the particle's electric charge.

The properties that we only have one force mediator and that this mediator is chargeless follow from the fact that U(1) has only one generator and is an Abelian group, respectively. The masslessness of the photon is not a priori fixed by the gauge group, but is rather imposed because the photon is found to be massless by observation. Alternatively, these features can be derived from the Lagrangian (2.11) by noting that there is only one field A^μ; the Lagrangian contains no factors of $A^\mu A^\nu$; and there is no kinematic term for the photon field that defines a photon mass.

Figure 2.3 shows the most relevant Feynman diagrams for QED: the photon propagator, and

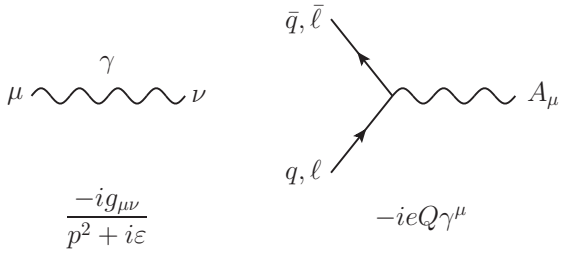

Figure 2.3
The propagator (left) and tree-level interaction vertex (right) of quantum electrodynamics [31], which describes the electromagnetic interaction. At any vertex, conservation of four-momentum is implied.

[6] In fact, the electromagnetic and weak force are more appropriately described in unified form, i.e. in electroweak theory, which is a Yang-Mills theory with gauge group $\text{SU}(2) \otimes \text{U}(1)$. However, for the purpose of this thesis, it is more convenient to consider the weak and electromagnetic forces as separate.

the tree-level interaction vertex. The propagator is dominated by a factor $1/p^2$, showing that the interaction amplitude decreases in transferred momentum. The tree-level diagram follows from the $-eQ\bar{\psi}\gamma^\mu\psi A_\mu$ term in the Lagrangian (2.11), which indicates that the tree-level interaction is a vector interaction (due to the γ^μ term) between the anti-fermion field ($\bar{\psi}$), the fermion field (ψ) and the photon field (A^μ), with coupling strength e.

Since the coupling constant $e < 1$, the amplitude of a diagram is decreasing in the number of interaction vertices, so that higher-order diagrams are increasingly suppressed and, as a result, first-order diagrams are often sufficiently accurate approximations. As an illustration, see Figure 2.4, which depicts the electron scattering process in QED. If electrons scatter, they can exchange some number of photons; in principle, this can be any number of photons. The total cross-section for electron scattering should then be calculated by taking into account the possibility that any N number of photons are exchanged, by adding the contributions from all these possible diagrams. However, each exchanged photon decreases the amplitude of a diagram by a factor of $\alpha = e^2/4\pi \approx 1/137$, so that diagrams with many photon exchanges are heavily suppressed; in fact, this suppression is generally sufficiently strong to justify neglecting all higher-order terms and only keep the diagram in which only a single photon is exchanged. This is an important result, as there is generally no way to directly calculate the sum of all possible diagrams, and thus a relatively accurate (and converging) approximation is essential.

Weak interaction

The weak interaction is the interaction between flavours, and is mediated by three massive spin-1 bosons: the Z^0, the W^- and the W^+. Since all fermions in the Standard Model have flavours, all these fermions interact weakly. For that reason, weak interactions are sometimes categorized as either leptonic, if only leptons participate (e.g. $\mu^- \to e^-\bar{\nu}_e\nu_\mu$); semi-leptonic, if both leptons and quarks participate (e.g. beta decay); or non-leptonic, if only quarks participate (e.g. $\Lambda^0 \to p\pi^-$) [30].

An interesting feature of the weak interaction is that interactions between W bosons and quarks always change the flavour of the quarks involved—such interactions are known as flavour-changing charged currents (FCCC). Of course, electric charge is conserved in such interactions, so that for example an up quark can only be converted into a down, strange or bottom quark, but not to a charm or top quark. Other than that, the weak interaction does not distinguish between flavours within a generation, but differs only between generations. That is, the weak force treats up and down quarks

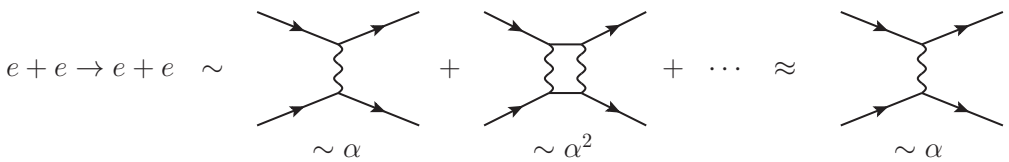

Figure 2.4
The scattering $ee \to ee$ can in principle be take place through the exchange of any number of photons. The diagram is proportional to α raised to the power of the number of photons exchanged. However, higher-order terms can generally be neglected in calculations.

on equal footing, while it has a different coupling to the doublet of strange and charm quarks, for example.

Interactions with Z^0 bosons never change flavours, i.e. flavour-changing neutral currents (FCNC) do not occur. Also, flavour changes are restricted to the quark sector; lepton flavours are not changed by the weak interaction in the Standard Model.

In the field theory describing the weak interaction, quark flavour changes are described by the Cabibbo–Kobayashi–Maskawa (CKM) matrix, which transforms the mass eigenstate doublets[7] d, s and b to the weak interaction doublets d', s' and b',

$$\begin{pmatrix} d' \\ s' \\ b' \end{pmatrix} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix} \begin{pmatrix} d \\ s \\ b \end{pmatrix}. \tag{2.14}$$

It then follows that a vertex that converts a q quark into a q' by W-exchange contributes a factor $V_{qq'}$ to the amplitude. The magnitude of the CKM elements are measured to be [36]

$$\begin{pmatrix} |V_{ud}| & |V_{us}| & |V_{ub}| \\ |V_{cd}| & |V_{cs}| & |V_{cb}| \\ |V_{td}| & |V_{ts}| & |V_{tb}| \end{pmatrix} \approx \begin{pmatrix} 0.974 & 0.225 & 0.004 \\ 0.225 & 0.974 & 0.041 \\ 0.009 & 0.040 & 0.999 \end{pmatrix}, \tag{2.15}$$

from which it can be seen that the elements for within-generation flavour conversions (i.e. $u \leftrightarrow d$, $c \leftrightarrow s$ and $t \leftrightarrow b$) are of order unity whereas those for between-generation conversions are orders of magnitude smaller. For that reason, interactions that have between-generation conversions are generally suppressed. Of course, the same holds for anti-quarks.

Another feature is that the charged weak interaction only applies to particles with left-handed (negative) chirality [or antiparticles with right-handed (positive) chirality]. Chirality is an intrinsic quantum number of a particle that for massless particles coincides with the particle's helicity, which is the projection of the spin onto the particle's momentum direction; thus a particle has right-handed helicity if its momentum and spin are parallel, and left-handed helicity if they are anti-parallel.

For massive particles, chirality and helicity coincide only in the ultrarelativistic limit. More specifically, in the limit that $E \gg m$, if a massive particle has left-handed chirality, its state $|\phi_L\rangle$ is decomposed in the helicity eigenstates $|\phi^-\rangle$ (left-handed helicity) and $|\phi^+\rangle$ (right-handed helicity) as [30]

$$|\phi_L\rangle = \left(1 - \frac{1}{2}\frac{m}{E}\right)|\phi^-\rangle + \frac{1}{2}\frac{m}{E}|\phi^+\rangle. \tag{2.16}$$

Similarly, for a right-handed fermion,

$$|\phi_R\rangle = \frac{1}{2}\frac{m}{E}|\phi^-\rangle + \left(1 - \frac{1}{2}\frac{m}{E}\right)|\phi^+\rangle. \tag{2.17}$$

As a consequence, for high-energy charged-current interactions, we can see that the production of massive right-handed particles is suppressed by a factor $1/\gamma$.

As with QED, all of the aforementioned behaviour can be derived from the theory's Lagrangian. However, because the weak Lagrangian $\mathcal{L}_{\text{weak}}$ is rather complicated, we will not state it in full here but only discuss the relevant features. For a full treatment, we refer to e.g. Refs. [31, 32].

[7] These doublets refer to their quark generation, i.e. d refers to the doublet consisting of the up and down quark, and similar for the doublets s and b.

Most importantly, the term related to the interactions between W bosons and leptons and quarks is [41]

$$-\frac{g}{2\sqrt{2}}\left[\sum_{\ell=e,\mu,\tau}\bar{\nu}_\ell\gamma^\mu(1-\gamma_5)\ell W^+_\mu+\sum_{q=u,c,t}\sum_{q'=d,s,b}\bar{q}\gamma^\mu(1-\gamma_5)V_{qq'}q'W^+_\mu\right]+\text{h.c.} \tag{2.18}$$

where g is the weak coupling constant, and h.c. denotes the Hermitian conjugate (which includes the W^- terms). In particular, the left-handedness of the weak interaction is due to the projection operator $(1-\gamma_5)/2$, and the quark flavour mixing is due to the non-diagonal CKM matrix $V_{qq'}$.

For the Z^0 boson interactions, we have the corresponding term [41]

$$-\frac{g}{2\cos\theta_W}\left[\sum_{\phi=\ell,\nu}\bar{\phi}\gamma^\mu(1-\gamma_5)\phi Z^0_\mu-\sum_q T^3_q\bar{q}\gamma^\mu(1-\gamma_5)qZ^0_\mu\right] \tag{2.19}$$

where the first summation sums over all leptons, i.e. $\phi=\{e,\mu,\tau,\nu_e,\nu_\mu,\nu_\tau\}$, and the second over all quark flavours, i.e. $q=\{u,d,s,c,b,t\}$; $T^3_q=+1/2$ for $q=u,c,t$ and $T^3_q=-1/2$ for $q=b,s,d$ is the weak isospin; and θ_W is the weak mixing angle. Again, we find a left-handed interaction; however, in this term there is no flavour mixing.

In addition, the Lagrangian contains three and four-point interaction terms for the gauge boson fields with themselves and with each other, and interaction terms between the W bosons and the photon. Finally, it contains interaction terms between the W and Z fields and the fermion fields with the Higgs field. These terms are beyond the scope of this text, and the reader is referred to e.g. Refs. [31, 32, 41].

However, it is useful to note that the Higgs field is responsible for the non-zero masses of the $W^\pm$ and Z^0 bosons, which are respectively 80.4 and 91.2 GeV, making them among the heaviest elementary particles in the Standard Model. Because particles often do not have sufficient energy to emit such heavy gauge bosons, the range of the weak interaction is constrained by the time-energy uncertainty relation,

$$\Delta t\leq\frac{\hbar}{2\Delta E}, \tag{2.20}$$

which limits the range to the order $\hbar c/2M_{W,Z}\sim 10^{-18}$ m.

The Feynman diagrams for the gauge boson propagators and the interaction terms in equations (2.18) and (2.19) are shown in Figure 2.5. These diagrams are similar to the diagrams of QED shown in Figure 2.3, except for two qualitative differences. First, the propagator contains a gauge boson mass term in the denominator. Since these masses are relatively large, they significantly decrease the amplitude of weak interaction diagrams. Second, the projection operator $(1-\gamma_5)/2$ suppresses the creation of particles with right-handed helicity (or antiparticles with left-handed helicity) according to equations (2.16) and (2.17). Evidently, a consequence of these both effects is for instance that weak interaction cross-sections are relatively small, and that particles that decay via the weak interaction will have relatively long lifetimes.

Similar to the case of QED, weak interactions generally also have contributions from diagrams with any number of boson exchanges. However, higher-order diagrams are even more strongly suppressed than in the case of QED, because each W exchange decreases the amplitude with a factor $g^2/\sqrt{2}M^2_W$, with $g\approx 0.65$ [36]. Therefore, approximating a weak interaction by its leading-order diagram is often sufficiently accurate.

Strong interaction

The strong interaction is mediated by eight massless spin-1 bosons, the gluons. The gluons couple only to quarks, as these are the only fermions with colour charge. In addition, gluons show self-interactions. Because the coupling constant of the strong force, α_s, is of order unity, these self-interactions give rise to non-trivial effects, such as confinement and asymptotic freedom.[8]

As mentioned before, the strong interaction couples to three different colours (often labelled r, g and b) and three corresponding anti-colours ($\bar{r}$, $\bar{g}$ and $\bar{b}$). Then, quarks have a colour (c), anti-quarks have an anti-colour ($\bar{c}$), and gluons have a combination of both ($c + \bar{c}$). Since total colour charge is always conserved, an quark-gluon interaction always changes the colour of the quark.

The experimental observation that free particles are always colour-neutral (which is any multiple of $r+g+b$) is called colour confinement. This directly implies that strongly interacting particles cannot be observed as free particles, but only as constituents of some colour-neutral bound state.

To determine what these bound states look like, we must consider the field theory description of the strong interaction, quantum chromodynamics (QCD), which is a Yang-Mills theory with SU(3) gauge symmetry. This means that each quark field ψ_q is an SU(3) colour triplet, denoted $\mathbf{3}$. The anti-quark fields are defined by the complex conjugate representation $\bar{\mathbf{3}}$. Now, for a state to be non-strongly interacting, it must be an SU(3) colour singlet, $\mathbf{1}$. Therefore, only bound states exist which combine to colour singlets. Of these, the most important are the mesons, which are bound states of a quark and an anti-quark,

$$\left|q\bar{q}'\right\rangle = \left|q\right\rangle \otimes \left|\bar{q}'\right\rangle \quad \sim \quad \mathbf{3} \otimes \bar{\mathbf{3}} = \mathbf{1} \oplus \mathbf{8} \tag{2.21}$$

and (anti-)baryons, which are bound states of three (anti-)quarks,

$$\left|qq'q''\right\rangle = \left|q\right\rangle \otimes \left|q'\right\rangle \otimes \left|q''\right\rangle \quad \sim \quad \mathbf{3} \otimes \mathbf{3} \otimes \mathbf{3} = \mathbf{1} \oplus \mathbf{8} \oplus \mathbf{8} \oplus \mathbf{10} \tag{2.22}$$

$$\left|\bar{q}\bar{q}'\bar{q}''\right\rangle = \left|\bar{q}\right\rangle \otimes \left|\bar{q}'\right\rangle \otimes \left|\bar{q}''\right\rangle \quad \sim \quad \bar{\mathbf{3}} \otimes \bar{\mathbf{3}} \otimes \bar{\mathbf{3}} = \mathbf{1} \oplus \mathbf{8} \oplus \mathbf{8} \oplus \mathbf{10} \tag{2.23}$$

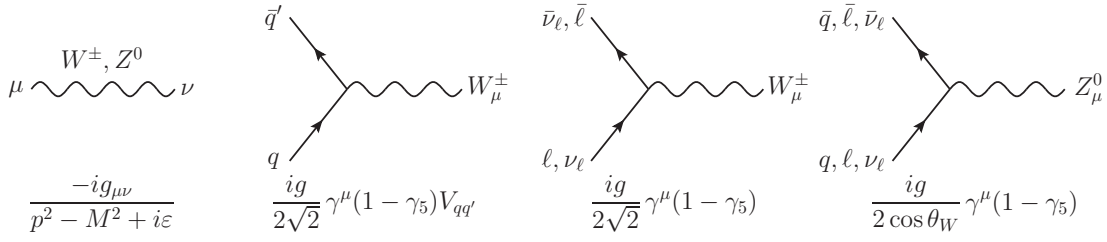

Figure 2.5
The propagator (left) and tree-level interaction vertices between the gauge bosons and the quarks and leptons of the weak interaction [31]. Besides these vertices, there are several interaction vertices between the different gauge bosons, and between the gauge bosons and the Higgs field. At any vertex, conservation of four-momentum is implied.

[8] At least, that is what is commonly believed—as of today, although there is ample experimental evidence, no rigorous proofs exist for these hypotheses.

These hadron states are generally the most important subjects of research at particle colliders, and we return to them in more detail in Section 2.2. Other combinations of quarks may also have singlet representations, for example tetraquarks ($|qq'\bar{q}''\bar{q}'''\rangle$) and pentaquarks ($|qq'q''q'''\bar{q}''''\rangle$ plus antiparticle), which both have been found in the LHCb experiment [42, 43].

A related feature of quantum chromodynamics is asymptotic freedom, which refers to the running of the coupling α_s. While the couplings of the electromagnetic and weak interactions also run, this leads to non-trivial phenomena only for the strong interaction.

More specifically, the strong coupling, which is related to g_s through $\alpha_s = g_s^2/4\pi$, asymptotically approaches zero as the distance scale goes to zero (or, equivalently, as the energy scale goes to infinity). As a consequence, quarks exchange fewer gluons as they approach each other, and can thus behave like free particles as long as they are sufficiently close to each other.

On the other hand, if the distance between two quarks increases, the coupling between them increases as well. As a result, the binding energy of the quarks increases, up to the point that it becomes energetically favourable to create an additional quark-anti-quark pair from the vacuum. This effect is especially important in high-energy collisions, where produced free quarks may have large momenta with respect to each other (i.e. the distance between them rapidly grows), which in that case leads to hadronization of the free quarks and jet production (see Figure 2.6) [30, 44].

The Lagrangian density of quantum chromodynamics [31],

$$\mathcal{L}_{\text{QCD}} = \bar{\psi}_q(i\gamma^\mu D_\mu - m)\psi_q - \frac{1}{4}G^a_{\mu\nu}G^{a,\mu\nu} \tag{2.24}$$

with

$$D_\mu = \partial_\mu - ig_s A^a_\mu \lambda^a, \tag{2.25}$$

and

$$G^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + g_s f^{abc} A^b_\mu A^c_\nu, \tag{2.26}$$

is very similar to the QED Lagrangian (2.11), except that (2.24) contains eight gauge fields (i.e. $a = 1, \ldots, 8$) rather than just one. This is due to the fact that SU(3) has eight generators. In addition, the field strength tensor $G^a_{\mu\nu}$ contains couplings between the different gluon fields through

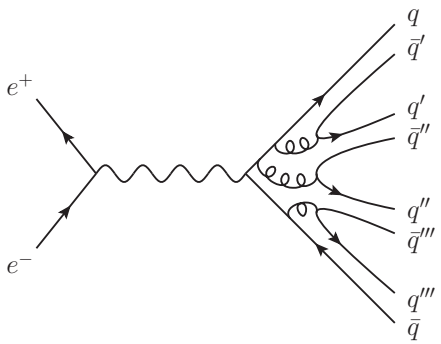

Figure 2.6
The process $ee \to$ hadrons. Initially, the electron-positron pair creates a $q\bar{q}$ pair via a photon. These quarks subsequently move apart and create additional $q\bar{q}$ pairs via gluon emissions. (Anti-)baryons may be produced in a similar way.

the non-zero structure constants f^{abc} of SU(3). Also, the gluon fields couple to the quarks through the non-scalar generators of SU(3), the Gell-Mann matrices λ^a, so that the quark-gluon interaction term becomes $g_s \bar{\psi}_q \gamma^\mu A^a_\mu \lambda^a \psi_q$, with the strong coupling $g_s = \sqrt{4\pi\alpha_s} = 1.22$ [36].

Curiously, another term is allowed in (2.24) that satisfies all general requirements for a Lagrangian density (that it is Lorentz invariant and local, and that it obeys the symmetries of the particular theory [45]), namely the CP violating term [7]

$$\mathcal{L}'_{\text{QCD}} = \mathcal{L}_{\text{QCD}} + \theta \frac{i g_s^2}{32\pi^2} G^a_{\mu\nu} \tilde{G}^{a,\mu\nu} \tag{2.27}$$

with $\tilde{G}^{a,\mu\nu} = \frac{1}{2}\varepsilon^{\mu\nu\rho\sigma} G^a_{\rho\sigma}$ the dual field strength tensor. However, currently no evidence exists that the constant θ is non-zero. Since we would naturally expect any term that is allowed in the Lagrangian to be observable in nature, this is striking. Accurately determining the value of θ is therefore an important field of research, which is also done extensively at particle colliders.

The relevant Feynman vertices that result from the QCD Lagrangian (2.24) are shown in Figure 2.7. Again, the results are similar to those obtained in QED.

Because the coupling α_s is of order unity at the energy scales of particle colliders, we can generally not approximate the amplitude of a process by taking only the diagrams of order α_s and ignoring higher orders. This often makes it difficult to make accurate calculations in QCD. In addition, it should be noted that some phenomena such as confinement cannot be calculated or described at all in our perturbative description of QCD. While this need not be a problem for interpreting experimental results, it may be problematic for making theoretical predictions or for Monte Carlo calculations based on first principles.

2.2 Hadrons

As mentioned before, the hadrons, i.e. bound states of quarks, are perhaps the most important subjects of research in high-energy particle experiments. The primary reason for this is that these are the only particles that interact through all forces in the Standard Model. In addition, many hadrons are relatively easy to produce in colliders.

More specifically, at colliders the decays of hadrons are often studied, as these let us determine the parameters in the Standard Model and allow us to verify (or, hopefully, falsify and extend) the

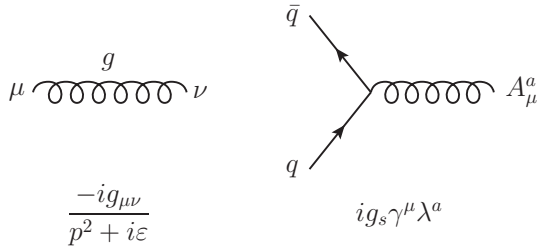

Figure 2.7
The propagator (left) and tree-level interaction vertex (right) of quantum chromodynamics [31], which describes the strong interaction. Besides these vertices, there are interaction vertices between the gluons. At any vertex, conservation of four-momentum and total colour charge is implied.

Standard Model. Some predictions of particular interest that the Standard Model makes are for example decay modes and (partial) decay rates depending on the quark contents of a particle.

In this section, we will discuss the two most important types of hadrons: mesons, consisting of a quark and an anti-quark ($q\bar{q}'$), and baryons, consisting of three quarks or three anti-quarks ($qq'q''$ or $\bar{q}\bar{q}'\bar{q}''$). To that end, we start by discussing some particle properties that are commonly used for classifying hadrons. In the next section, general rules for their decay modes and rates are examined.

2.2.1 Hadron classification

There are several quantum numbers that are commonly used for the classification of hadrons. First of all, the total angular momentum $J = L \oplus S$ of the particle is used, and the intrinsic parity and charge conjugation eigenvalues, respectively P and C. Note that these quantum numbers cannot be derived from the quark content of the particle, but must be experimentally determined (for instance by looking at their decay products).

Second, the similar masses of the up and down quarks imply that their behaviour under the strong force should be very similar. This suggests that they can be treated on equal footing by treating them as an (approximate) SU(2) doublet rather than as two distinct particles.

For that reason, the isospin quantum number I was introduced. Because the u and d quarks form an SU(2) doublet, they must have $I = 1/2$. Then, isospin is defined such that u has eigenvalue $I_3 = +1/2$ and d has $I_3 = -1/2$.[9] The anti-quarks $\bar{u}$ and $\bar{d}$ have opposite values for I_3. All other particles are isospin singlets, and consequently have $I = 0$ and $I_3 = 0$. The total isospin of a multi-quark state then has

$$I_3 = \frac{1}{2}\left[N(u) - N(\bar{u}) - N(d) + N(\bar{d})\right], \tag{2.28}$$

where $N(q)$ denotes the number of q quarks in the state.

It turns out to be convenient to define similar flavour quantum numbers for the other quark flavours:

$$\text{Strangeness:} \quad S = -\left[N(s) - N(\bar{s})\right] \tag{2.29a}$$
$$\text{Charm:} \quad C = +\left[N(c) - N(\bar{c})\right] \tag{2.29b}$$
$$\text{Bottomness:} \quad B' = -\left[N(b) - N(\bar{b})\right] \tag{2.29c}$$
$$\text{Topness:} \quad C = +\left[N(t) - N(\bar{t})\right] \tag{2.29d}$$

In addition, the baryon number is defined as

$$\text{Baryon number:} \quad B = \frac{1}{3}\left[N(q) - N(\bar{q})\right], \tag{2.30}$$

where q can be any quark flavour.

These flavour quantum numbers can subsequently be combined into the hypercharge,

$$Y = B + S + C + B' + T, \tag{2.31}$$

[9] Note that isospin, because it has an SU(2) structure, features the same mathematics as spin.

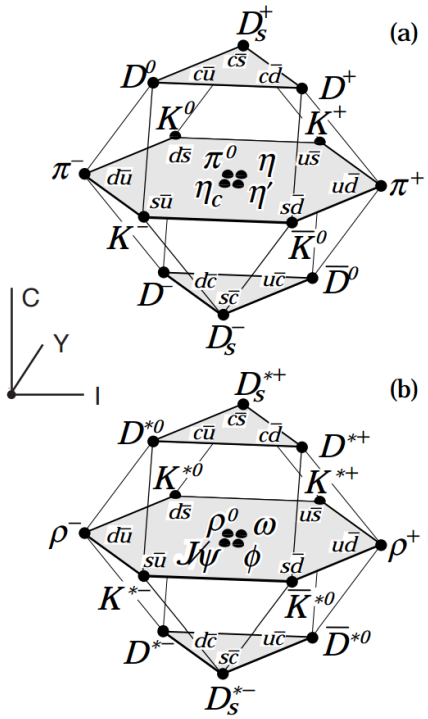

Figure 2.8
The meson 16-plets for (a) $J^P = 0^-$ (pseudoscalar) mesons and (b) $J^P = 1^+$ (vector) mesons including u, d, s, c quarks. From Ref. [36].

which together with the isospin I_3 and the quantum numbers J, P and C (often denoted in the form J^{PC} or just J^P) forms the basis for classifying hadrons.

There are several ways of constructing the spectrum of hadrons. In general, one starts with putting the N lightest quarks in an SU(N) N-tuplet, and then construct combinations through multiplication. This product is then decomposed into representations that have similar properties. For example, for mesons we would calculate and decompose the product $\mathbf{N} \otimes \overline{\mathbf{N}}$.

Then the choice of N remains. As mentioned earlier, there is an approximate SU(2) flavour symmetry under the interchange of the u and d quarks, suggesting $N = 2$. However, since the strange quark also has approximately the same mass, taking $N = 3$ will also lead to a classification of hadrons in which each group has approximately similar properties.

In principle, this method will only predict hadrons consisting of u, d and s quarks. Therefore, to find all possible combinations, one should take $N = 6$, so that all quark flavours are included. While this will not yield a classification in which each group consists of similar particles, there will be some groups for which this is true; in fact, because SU(3) < SU(6), this will give the groups obtained using SU(3) as subgroups.

2.2.2 Mesons

For $N = 4$, i.e. including quarks up to the charm, the mesons with $J^P = 0^-$ (anti-parallel spins) and $J^P = 1^+$ (parallel spins) are shown in Figure 2.8. The structure is obtained from

$$\mathbf{4} \otimes \bar{\mathbf{4}} = \mathbf{1} \oplus \mathbf{15}. \tag{2.32}$$

If the same would be done with $N = 3$, the middle planes (with $C = 0$) would be obtained (except for the $c\bar{c}$ particles η_c and J/ψ). For those planes, the flavour symmetry is still a good approximation, and these particles therefore indeed have approximately equal masses. However, SU(4) flavour symmetry is clearly badly broken by the significantly heavier charm mass, as the particles on the $C = \pm 1$ planes have considerably more mass.

This method is particularly successful in predicting the states at the centres of the middle planes, as some of these states are no straightforward $q\bar{q}$ pairs. Specifically, [46]

$$\pi^0 = \frac{1}{\sqrt{2}} \left(u\bar{u} - d\bar{d} \right) \tag{2.33a}$$

$$\eta = \frac{1}{\sqrt{6}} \left(u\bar{u} + d\bar{d} - 2s\bar{s} \right) \tag{2.33b}$$

$$\eta' = \frac{1}{\sqrt{3}} \left(u\bar{u} + d\bar{d} + s\bar{s} \right) \tag{2.33c}$$

The ρ^0, ω and ϕ mesons have respectively the same quark constitutions but different J^P quantum numbers. η_c and J/ψ are $c\bar{c}$ mesons. η' and ϕ are the only fully symmetric states, and they therefore correspond to the singlet representation in equation (2.32).

It should be noted that the states (2.33) are only approximate, as they follow from an approximate symmetry. This is also the reason that there are no states that are superpositions of charm states, i.e. because the charm quark is distinctly heavier. Contrarily, $u\bar{u}$, $d\bar{d}$ and $s\bar{s}$ do not come as pure states because the symmetry between these states do not allow them to (the symmetry fixes a degree of freedom by which it prevents the existence of these pure states).

Extending this model to the bottom quarks, we similarly obtain the $J^P = 0^-$ mesons B^+ ($u\bar{b}$), B^0 ($d\bar{b}$), B_s^0 ($s\bar{b}$), B_c^0 ($c\bar{b}$), and their antiparticles; and the $J^P = 1^+$ mesons B^{*+}, B^{*0}, B_s^{*0}, B_c^{*0}, which respectively have the same quark content. This has historically been the most important use for this quark model: it successfully predicts the types of particles that exist in nature, and some of their properties (in particular, J^P).

2.2.3 Baryons

A similar structure can be made for baryons, using that [47]

$$\mathbf{4} \otimes \mathbf{4} \otimes \mathbf{4} = \mathbf{20} \oplus \mathbf{20} \oplus \mathbf{20} \oplus \mathbf{4}. \tag{2.34}$$

Of these **20**s, one is fully symmetric, and two are mixed-symmetric. The **4** is anti-symmetric.

The resulting baryons are shown in Figure 2.9. Again, several of these states are non-pure, except in te case of the symmetric 20-plet. For instance, for the mixed-symmetric 20-plet and the anti-symmetric 4-plet we have respectively

$$\Sigma_c^+ = \frac{1}{\sqrt{2}} \left(udc + duc \right) \tag{2.35a}$$

$$\Lambda_c^+ = \frac{1}{\sqrt{2}} \left(udc - duc \right) \tag{2.35b}$$

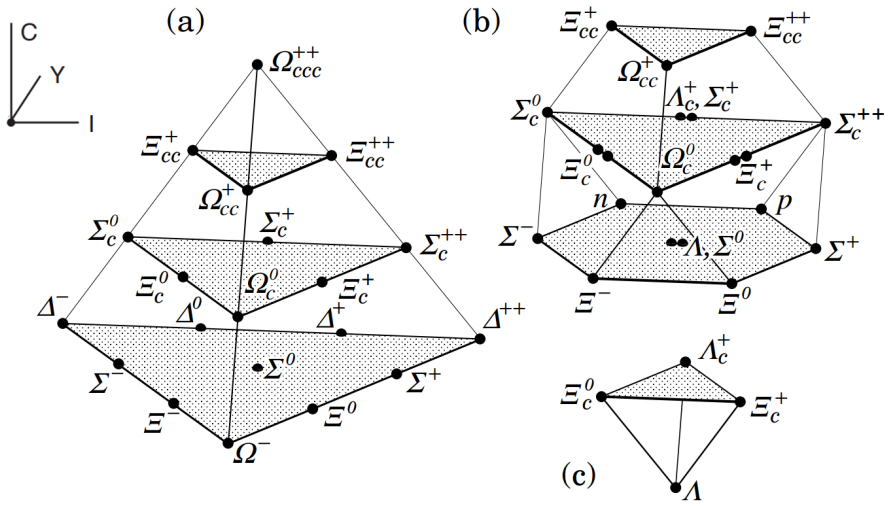

Figure 2.9
The baryon multiplets: (a) the symmetric $J^P = \frac{3}{2}^+$ 20-plet; (b) the mixed-symmetric $J^P = \frac{1}{2}^+$ 20-plets; and (c) the anti-symmetric $J^P = \frac{1}{2}^-$ 4-plet. Adapted from Refs. [36, 47].

Thus, this method is also useful for constructing baryons and predicting some of their properties. For a more detailed treatment, the reader is referred to e.g. Crede and Roberts [47].

2.3 Hadron decays

Now that we now what kind of hadrons exist, we can use the theory from Section 2.1 to determine the decay modes of these particles, and the rates at which they decay. This gives important insights into what decays are worthwhile to investigate or look for at collider experiments.

The partial transition rate (in the centre-of-mass frame) from an initial state $|i\rangle$ of one particle with mass M and four-momentum P to a final state $|f\rangle$ consisting of N particles with four-momenta $\{p_1, \ldots, p_N\}$ is given by Fermi's golden rule [30, 48],[10]

$$\mathrm{d}\Gamma_{i\to f} = \frac{(2\pi)^4}{2M}|\mathcal{M}_{i\to f}|^2 \,\mathrm{d}\Phi_N(P; p_1, \ldots, p_N), \tag{2.36}$$

where $\mathcal{M}_{i\to f}$ is the Feynman amplitude for the interaction, and Φ_N is the Lorentz invariant phase space volume of the final state,

$$\mathrm{d}\Phi_N(P; p_1, \ldots, p_N) = \delta^4\left(P - \sum_{i=1}^{N} p_i\right) \prod_{i=1}^{N} \frac{\mathrm{d}^3\mathbf{p}_i}{(2\pi)^3 2E_i}. \tag{2.37}$$

Evidently, the total decay rate (to any final state), Γ_i, is obtained by integrating over the phase space, and summing over all possible final states. It should also be emphasised that $\mathcal{M}_{i\to f}$ contains a sum over all possible decay channels that can mediate a transition from state $|i\rangle$ to $|f\rangle$.

[10] For a derivation of Fermi's golden rule in quantum mechanics, see e.g. Sakurai and Napolitano [49].

These equations show that any decay is essentially governed by two elements: the delta function, that contains all kinematic constraints; and the Feynman amplitude, which contains all other particle physics. We will now briefly consider the implications of both.

2.3.1 Phase space constraints

The phase space constraint, given by the delta function in equation (2.37), is simply that total four-momentum is conserved. Of course, if this constraint holds in the centre-of-mass frame (in terms of which the constraint is formulated), this also holds in the lab frame.

An implication of (2.37) is that the partial decay rate is proportional to the phase space volume of the final state, which is consistent with the assumptions of statistical mechanics. As a consequence, configurations in which the the total energy is evenly distributed over the individual particles (i.e. in the centre-of-mass frame) are favoured, as such configurations have the highest degeneracies [in terms of (2.37), such configurations maximize the term $1/\prod_i E_i$].

It should be noted here that the rule (2.36) holds only for transitions to non-virtual final states $|f\rangle$; i.e. it does not describe transitions to intermediate states containing gauge bosons. For example, the emission of a virtual photon in $e \to e\gamma^*$ as part of the scattering $ee \to ee$ is not described by the golden rule (2.36).

2.3.2 The Feynman amplitude[11]

The amplitude $\mathcal{M}_{i\to f}$ is proportional to the vertices given by the interaction Lagrangians (2.11), (2.18), (2.19) and (2.24); these vertices are given in Figures 2.3, 2.5 and 2.7, which then also depict all the possible interactions in the Standard Model (with the exception of interactions between gauge bosons, and interactions with the Higgs field).

The dependence of the golden rule (2.36) on the (squared modulus of) the amplitude $\mathcal{M}_{i\to f}$ thus strongly restricts which decays are allowed; that is, many decay modes are forbidden in the Standard Model. In particular, in the following, we focus on the conservation laws for lepton and quark flavours; the conservation laws for parity and resulting implication that some decays are suppressed by helicity constraints; and the general strengths of the different interactions. Because these rules can all be directly derived from Section 2.1, we will only state the implications in the section below.

Lepton flavour conservation

There are no vertices connecting two different charged leptons (i.e. no vertices proportional to $\bar{\ell}\ell'$). Therefore, charged lepton flavours are always conserved in Standard Model interactions. Neutrinos can violate lepton flavour conservation through flavour oscillations as they propagate, although this effect is negligible at the scales of particle colliders.

[11] A complete discussion of the amplitude $\mathcal{M}_{i\to f}$ is beyond the scope of this text, and we will limit our treatment to the relevant factors only.

Quark flavour conservation

Quark flavours are always conserved, except in charged-current weak interactions (i.e. W exchanges); for this latter interaction there is a vertex proportional to $\bar{q}V_{qq'}q'\gamma^\mu W_\mu$. This implies that quark flavour changing interactions are allowed, but suppressed by a factor $V_{qq'}$. These factors are given in equation (2.15), from which it can be seen that same-generation flavour changes are hardly suppressed (as the diagonal elements of $V_{qq'}$ are of order unity), whereas different-generation flavour changes are increasingly suppressed as the generation difference increases (i.e. in particular $t \leftrightarrow d$ and $b \leftrightarrow u$ changes are strongly suppressed).

No such vertices exist for other interactions, implying that quark flavour changing interactions must always take place via exchange of a W boson.

Parity conservation

The parity of a state is related to how it transforms under the operation $\mathbf{x} \to -\mathbf{x}$. In particular, a state $|\psi\rangle$ may remain unchanged under such an operations (so that its parity $P_\psi = +1$); or it may pick up a minus sign (so that $P_\psi = -1$).

Such a state may consist of multiple particles, but it can also consist of a single particle. In the latter case, the parity of the state is given by the intrinsic parity of the particle. In principle, this intrinsic parity cannot be derived a priori; however, from the quark model presented in Section 2.2 we can predict the existence of particles with some J^P, and can thus predict the intrinsic parity of some particle if we know how its place in the quark model. In addition, we know that for bosons, particles and antiparticles have the same intrinsic parity; for fermions, parity is defined such that particles and antiparticles have opposite intrinsic parities [33].

For states of multiple particles, their intrinsic parities multiply. In addition, the orbital angular momentum of the state must be taken into account. Specifically, if the state $|\psi\rangle$ consists of particles ϕ_i with intrinsic parities P_{ϕ_i} and orbital angular momentum l, the resulting parity is [33]

$$P_\psi = (-1)^l \prod_i P_{\phi_i}. \tag{2.38}$$

It can be shown that all interaction vertices proportional to γ^μ conserve parity, while all vertices proportional to $\gamma^\mu(1-\gamma_5)/2$ maximally violate parity. Since the weak interaction is the only interaction which has such vertices (and only such vertices), the weak interaction maximally violates parity, while the other forces conserve parity.

Helicity suppression

Another effect due to the vertex term $(1-\gamma_5)$, and thus specific to the weak interaction, is that only particles of left-handed chirality participate in interactions. For massless particles, chirality and helicity (the orientation of the spin with respect to the momentum) coincide, and consequently only particles with left-handed helicity (and thus a specific spin direction) can be produced.

On the other hand, for massive particles, chirality and helicity are related through equations (2.16)–(2.17). Then, if e.g. conservation of angular momentum requires the helicity of a produced particle to be right-handed, this particle can only be produced through the chirally left-handed

Interaction	Coupling strength	Lifetime [s]
Strong	$\sim 10^{\ 0}$	10^{-22}–10^{-24}
Electromagnetic	$\sim 10^{-2}$	10^{-16}–10^{-21}
Weak	$\sim 10^{-6}$	$10^{-\ 7}$–10^{-13}

Table 2.1
Indication of a particle's lifetime given the force by which it primarily decays. The coupling constants are given at energy scales of 90 GeV (M_Z). Adapted from Refs. [34, 36, 50].

component of its wave function. But this component may be very small if the particle is produced at high energy, and, as a consequence, such a decay is suppressed by a factor $m/2E$.

This suppression is especially relevant if particles are produced at high energies, as is the case at colliders; or if the particles have very small (but non-zero) masses. In fact, for the neutrinos the suppression $m/2E$ is prohibitively small, even at low energies, so that right-handed neutrinos (which can only be produced in weak interactions) have not yet been observed in nature.

Interaction coupling strength

The amplitude $\mathcal{M}_{i\to f}$ is directly proportional to the coupling constant of the interaction which mediates the decay. Generally stated (i.e. ignoring suppressions mentioned before), the coupling strength is the leading factor determining the magnitude of the amplitude. As a direct consequence, the (partial) decay rates for decays mediated by the strong interaction will be considerably higher than those for weak or electromagnetic decays. Moreover, if some decay can take place via either the strong and the weak or electromagnetic force, it is much more likely that it will take place via the strong force than via another force.

Whether a particle decays primarily via the strong, weak or electromagnetic force is therefore a useful way of classifying particles, as the primary decay mode is a strong indicator of the particle's lifetime (as a high decay rate trivially implies a short lifetime). For each interaction type, an indication of the lifetime is given in Table 2.1.

Evidently, hadrons that have decay modes mediated by the strong interaction (i.e. modes that conserve parity, do not require quark flavour changes and are kinematically allowed) will have significantly shorter lifetimes. Hadrons for which such channels are forbidden by some conservation law, on the other hand, will only decay via the weak or electromagnetic interaction, and will therefore have substantially longer lifetimes. At the extreme, all decay channels of a particle are forbidden by some conservation law, and the particle will be stable; in the current model, this is only the case for the proton and the electron.

2.3.3 Resonances

In some cases, an initial state $|i\rangle$ decays to an intermediate state $|r\rangle$ before decaying to the final state $|f\rangle$. For example, the decay

$$\Lambda_c^+ \to p\pi^+ K^- \tag{2.39}$$

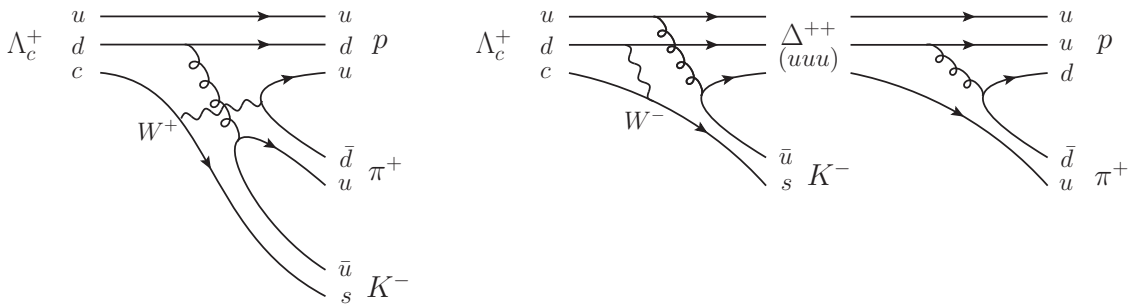

Figure 2.10
Two possible decay modes for $\Lambda_c^+ \to p\pi^+K^-$: a non-resonant decay (left) and a resonant decay via Δ^{++} (right).

may take place via an intermediate state containing $\Delta^{++}K^-$, i.e.

$$\begin{aligned} \Lambda_c^+ &\to \Delta^{++}K^- \\ &\hookrightarrow p\pi^+ \end{aligned} \tag{2.40}$$

However, it may also decay directly (i.e. not via some intermediate state); both decay modes are shown in Figure 2.10.

These channels differ in two respects. First, and perhaps most obvious, the amplitudes for both channels will be different, because the diagrams and the interactions involved differ. In the general case, the total transition rate from some state $|i\rangle$ to some final state $|f\rangle$ is given by the sum of the transition rates via each possible intermediate state $|r\rangle$.

Second, the kinematics differ. In particular, in the case that the decay is via a Δ^{++}, the invariant mass of the $p\pi^+$ pair is constrained to the Δ^{++} mass, while this is not true in the case of a three-body decay without an intermediate Δ^{++}.

As a result, the partial decay rate for the interaction (2.39) will peak at the configuration where the mass of the $p\pi^+$ pair is around the Δ^{++} mass. Interestingly, however, while the Δ^{++} has probably existed as an intermediate state in such a decay, it is generally not observable due to its short lifetime: because it decays via the strong interaction, it can travel only about 10^{-15} m before decaying, which is much too short to detect. For these reasons, i.e. because the Δ^{++} state gives rise to a peak in the partial decay rate at certain energy configurations of the final particles, and because the Δ^{++} itself cannot be observed directly, such states are often referred to as resonances.

Interestingly, because these resonance states are so short-lived, their energies and thus rest masses are not well-defined. As a consequence, there will be a significant variance in the invariant mass of the final particles that were produced from this resonance particle. That is, in our previous example, the variance in the invariant mass of the $p\pi^+$ pair is directly related to the lifetime of the Δ^{++}. Specifically, the probability distribution for the observed mass M of the resonance is

$$\text{pdf}\, M \propto \frac{1}{(M - M_0)^2 - \Gamma^2/4}, \tag{2.41}$$

where M_0 is the characteristic mass of the resonance, and Γ is the total decay rate of the resonance (where $\Gamma = 1/\tau$, if τ is the lifetime).

This distribution can be relatively wide for short-lived particles. For example, for the Δ^{++}, which has a lifetime of 6×10^{-24} s, the width of the mass distribution is 117 MeV (which is about 10 percent of its characteristic mass, which is 1232 MeV) [36]. However, for longer-lived particles, distribution (2.41) also holds; although in many cases the decay rate Γ is so low that (2.41) is in fact sharply peaked, up to the point that its variance goes to zero.

This behaviour will also be relevant for the simulation of particle decays, especially when cascade decays with intermediate resonances are of interest. Since many such resonance particles exist, it is not uncommon that they are produced in decays as intermediate states. We will return to this issue in Chapter 4, of which the subject is to simulate particle decays. But first, the next chapter will describe the LHCb detector and the experiment's data flow, to put the theoretical work from this chapter in an experimental perspective.

Chapter 3

The LHCb experimental setup

3.1 Detector layout[1]

The layout of the LHCb detector is shown in Figure 3.1. It is designed for high-precision study of heavy flavour physics through the analysis of charm and beauty hadrons. Since such hadrons are generally produced at relatively small angles with respect to the beam line, the LHCb detector is not centred about the collision point, but instead is behind the collision point; i.e. it is a single arm spectrometer. This yields a unique acceptance for the pseudorapidity between $1.8 < \eta < 4.9$, compared to general purpose detectors which typically have an acceptance $|\eta| < 2.4$ [52].

Beauty or charm hadrons generally have the following signature [53]: they have a relatively long lifetime, of about a picosecond (so that $c\tau \sim 0.1$ mm); their mass exceeds 1 GeV, and consequently their decay products carry relatively large transverse momenta p_T; and many decay channels of interest produce muons, which are relatively easy to detect.

Figure 3.2 shows the production angle plots for $b\bar{b}$ pairs produced at LHC. Because the LHCb detector has only a single arm, at least half of the events will be outside the acceptance. Nonetheless, the detector will be exposed to large numbers of $b\bar{b}$ pairs since its acceptance is centred about a high-density region of the angular production distribution. The total production cross section for $b\bar{b}$ pairs within the acceptance is $\sigma_{b\bar{b}} = (75.3 \pm 14.1)\ \mu$b; for $c\bar{c}$ pairs, this is $\sigma_{c\bar{c}} = (1419 \pm 134)\ \mu$b [53].

The detection and measurement of events can be divided in four stages. First, because the collider produces more data than the detector and computing farms can process, the detector must make an online selection of events that are potentially interesting. Second, the tracks of detected particles must be reconstructed, to determine from which decay these particles originate and where that decay took place. Third, the momenta of the particles must be measured. And, fourth, the identity of the particles must be determined. These stages, from the point of view of the detector, are the subject of this section; after these stages, the collected data must be processed and analysed, which is the subject of Section 3.2.

[1] This section strongly relies on Ref. [51], unless stated otherwise.

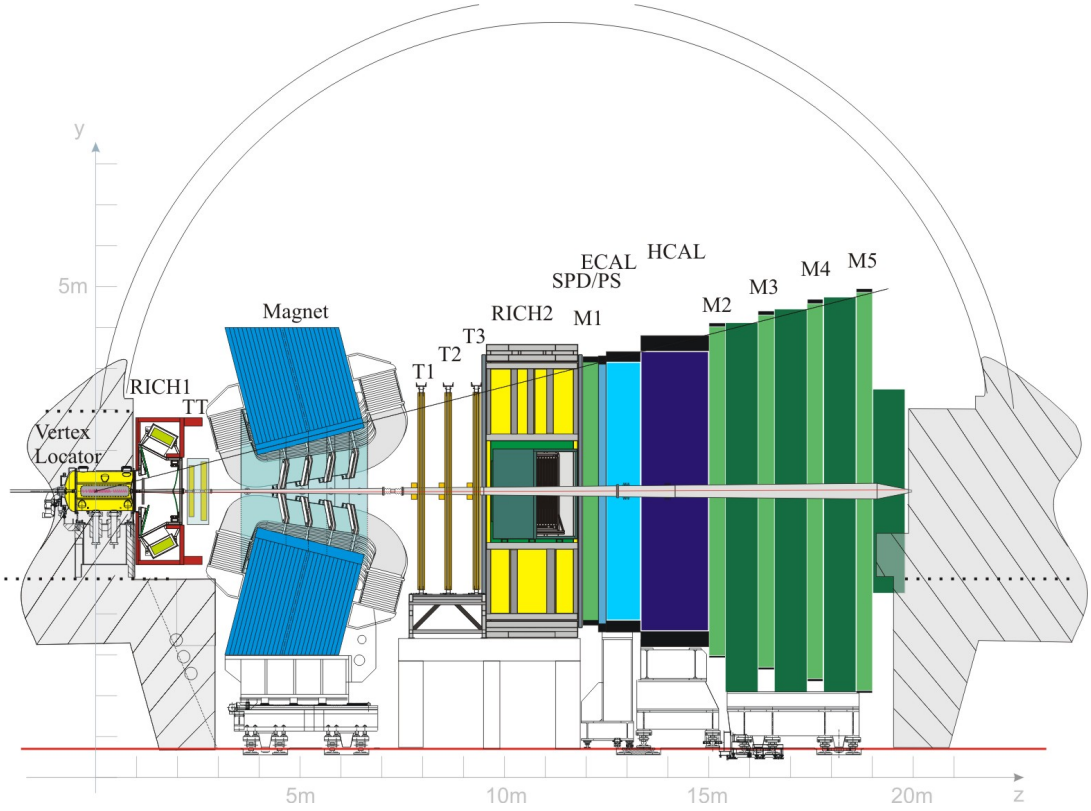

Figure 3.1
The LCHb detector layout [51]. The tracking system consists of the vertex locator, the TT and the T1–3 stations. Particle identification depends on data from RICH1–2, M1–5, SPD/PS, ECAL and HCAL.

3.1.1 Event selection

There are two main reasons for processing only a selection of the events inside the detector: first of all, the 40 million bunch crossings per second that the LHC generates, that generate about 13 million visible events per second [53], is simply far exceeding the detector bandwidth, which is about 1 million event readouts per second [54]; and, in addition, not all events contain physics of interest, and we wish to reject those events.

The event selection process is schematically shown in Figure 3.3. It roughly consists of two stages: a hardware stage (the level 0 trigger), reducing the readout rate to 1 MHz, followed by a software stage (the high level trigger), which reduces this rate further to 12.5 kHz. The resulting events are then saved to disk for later (offline) processing and analysis.

Level 0 trigger

The level 0 (L0) trigger decreases the visible interaction rate from 13 MHz to 1 MHz through a series of crude selection criteria. The L0 decision depends solely on data from the calorimeters

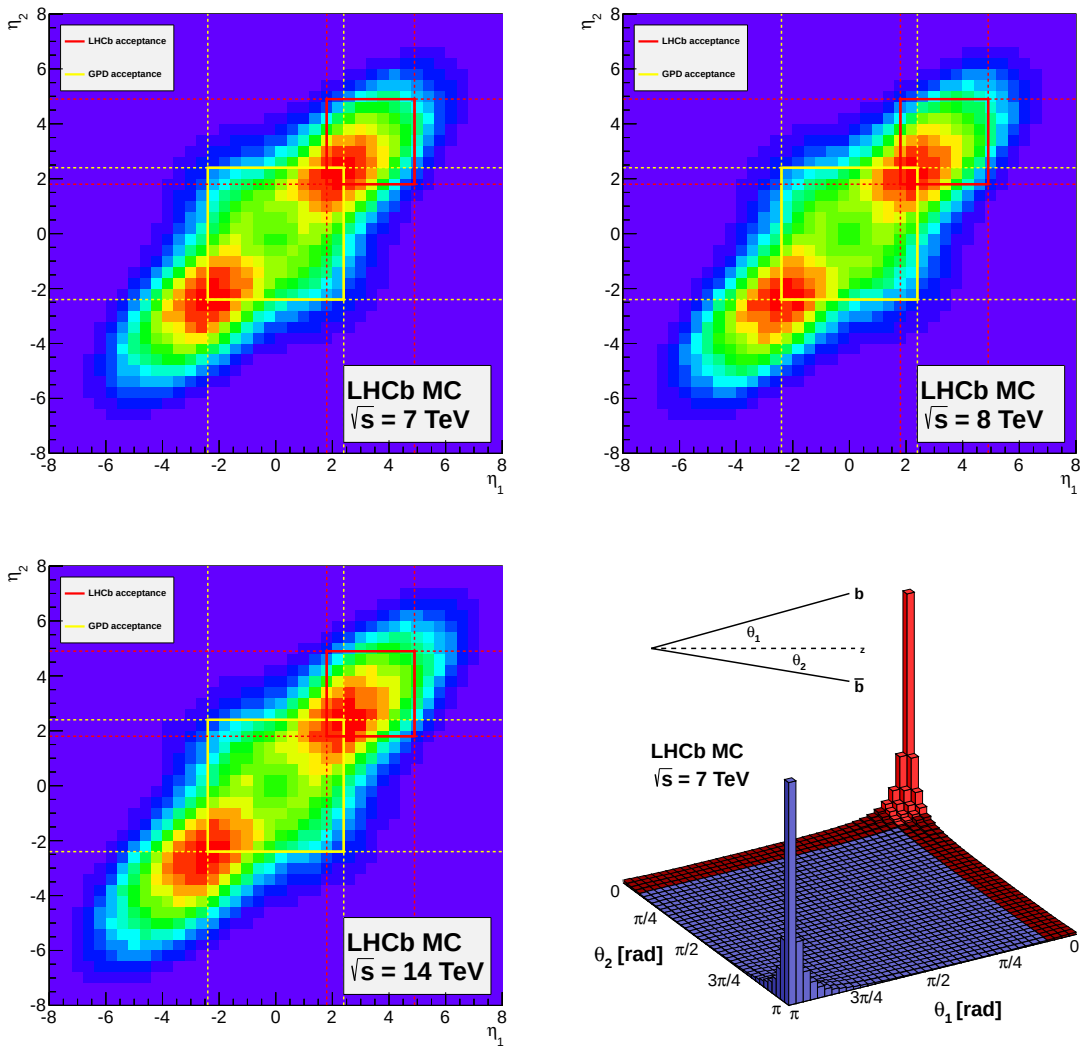

Figure 3.2
Bottom left and top: Angular distributions for the production of b and $\bar{b}$ pairs, corresponding to respectively pseudorapidities η_1 and η_2, for centre of mass energies $\sqrt{s}$ of 7, 8 and 14 TeV at LHC. The LHCb acceptances (red lines) are compared to the acceptances of a typical general purpose detector (yellow lines). Bottom right: Similar, but with geometrical angles in radians, at $\sqrt{s} = 7$ TeV, with the LHCb acceptance shown by the red bands. From Ref. [52].

(ECAL and HCAL in Figure 3.1) and the muon stations (M1–5), which are read out synchronous with the bunch crossings; other detector components cannot be read out at this rate [53]. Events are then selected either if there are high-p_T muons detected by the muon system, or if the calorimeters detect high energy deposits in the transverse direction, as such events are in agreement with the general signature of b and c hadron decays.

In addition, the pile-up system in the vertex locator determines the number of pp collisions in the bunch crossing. Because there is a large probability that combinatoric background events would

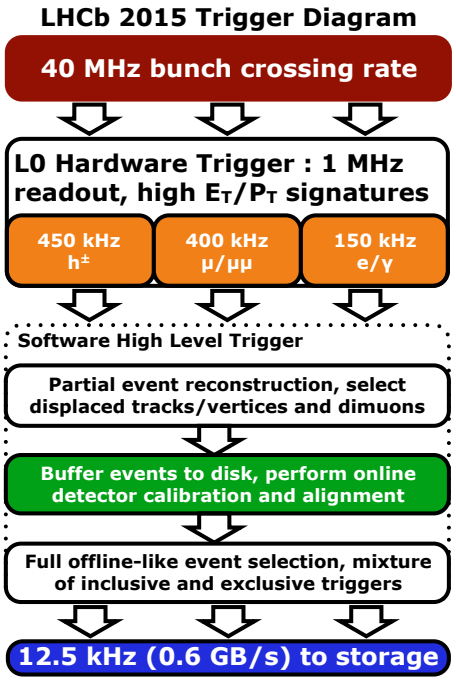

Figure 3.3
The event selection process of the LHCb detector. From Ref. [54].

falsely pass the L0 trigger if there are many *pp* collisions, events that are too busy are rejected by the pile-up system.

High level trigger

The high level trigger (HLT), which is software based and executed on a computing grid, consists of two stages. In the first stage (HLT1), the event is partially reconstructed, and contains the following steps: first, track segments in the vertex locator are reconstructed; then, those tracks that are sufficiently distant from the primary vertex to be consistent with decays of particles with long lifetimes, or those that can be connected to muon hits, are selected and extrapolated to the main tracker; finally, the event is selected if the tracks are consistent with an event with sufficiently high transverse momentum [53]. This procedure reduces the event rate by approximately 90 percent.

In the second stage (HLT2), all tracks that pass some loose requirements are reconstructed. Moreover, combinations of tracks are searched that may be consistent with decays of interest. If such combinations are found, the event is written to disk. The event rate has now been reduced to about 12.5 kHz [53, 54].

3.1.2 Track and vertex reconstruction

Track reconstruction is done using hits in the vertex locator (VELO) and the Tracker Turicensis (TT), which are upstream of the magnet, and the T1–3 trackers, which are downstream of the magnet. If a (charged) particle passes through one of these detector components, they leave hits.

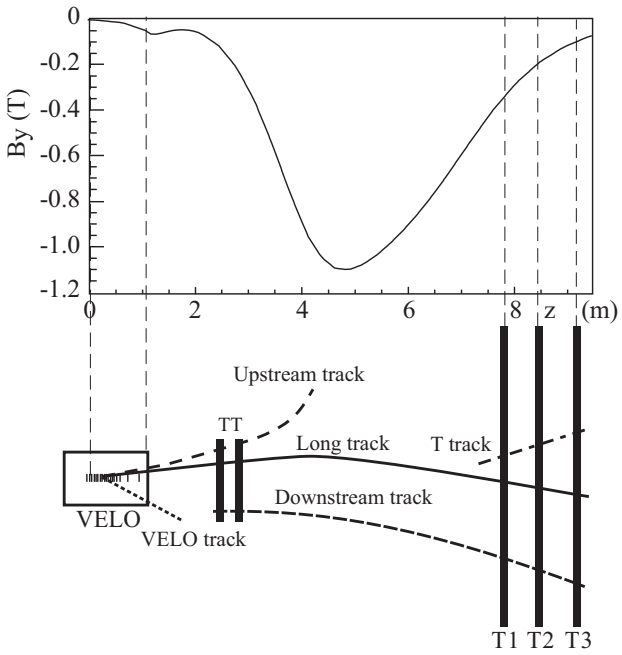

Figure 3.4
The different classes of tracks that are reconstructed in the detector. For reference, the y-component of the magnetic field is shown above. From Ref. [51].

Track reconstruction then is about reconstructing a set of tracks that are consistent with the given collection of hits in some event.

Ideally, tracks can be reconstructed that run from the VELO up to the calorimeters; these are referred to as long tracks. However, it is also possible that there is no track in the VELO because the particles are products of a decay that has taken place past the VELO, which yields a downstream track; or that the track is from a particle produced in a secondary interaction, which give downstream or T tracks; or that the particle is bent out of the acceptance region by the magnet, which gives an upstream track. These tracks and the magnetic field strength are shown in Figure 3.4. While not all tracks are useful for analysis, most can still be used for e.g. calibration.

After tracks have been reconstructed, they are refitted using a Kalman filter, which smoothens the track. This is useful to correct the trajectories for energy loss and scattering effects.

Of special importance is the VELO, which can determine the vertices from which tracks originate with high precision, so that it can identify which particles were produced in the same decay. In addition, it can determine the primary vertex, i.e. the location of the pp collision.

3.1.3 Energy and momentum measurement

The particle momentum is straightforwardly determined using the fitted track, by examining the curvature in the magnetic field, and noting that this curvature is due to the Lorentz force, which is set equal to the centripetal force,

$$Q\,\mathbf{p}\times\mathbf{B} = \frac{|\mathbf{p}|^2}{|\mathbf{r}|^2}\mathbf{r}, \tag{3.1}$$

where $\mathbf{r}$ is the vector from the particle to the focus of the curvature. The solution for the magnitude of the momentum is then just

$$|\mathbf{p}| = Q\,|\mathbf{B}|\,|\mathbf{r}|. \tag{3.2}$$

Evidently, the sign of the charge Q is given by the direction of the curvature; the magnitude of Q can be obtained from the particle identification.[2]

The energy of a particle is subsequently also determined from the particle identity, as the identity fixes the mass m of the particle, so that we simply have

$$E = \sqrt{|\mathbf{p}|^2 + m^2}. \tag{3.3}$$

That is, the particle's momentum and identity are determined by the detector, and its energy is calculated afterwards.

Energy measurement of electrons and positrons

An exception to this procedure is the electron[3] energy. Electrons can lose a relatively large fraction of their energy (by emitting photons) as they traverse the detector due to interactions with the material, and in particular due to bremsstrahlung. As a result, the momentum measured by the curvature in the magnetic field does not correspond to the momentum of the electron at production, which will significantly degrade the mass reconstruction of the signal.

In particular, this will happen if the bremsstrahlung emission takes place before the magnet (upstream); if the photon is emitted downstream of the magnet, the energy lost is already taken into account by momentum reconstructed from the track fit [55]. However, downstream photon emissions are still taken into account, since these photons will end up in the same ECAL cell as the electron, and the ratio between the electron momentum (as measured by curvature) and energy deposited in the ECAL (i.e. the combined energy from the electron and the emitted photon) are a useful measure for particle identification purposes [51].

The solution is to recombine the electron and the bremsstrahlung photons it has emitted. This is done using the electromagnetic calorimeter (ECAL), which can measure the momenta of photons and electrons (per cluster).[4] Specifically, the reconstruction extrapolates the electron tracks from before the magnet to the ECAL, which defines a region of ECAL clusters in which bremsstrahlung photons should end up if emitted upstream (see Figure 3.5); then, if there were any neutral clusters (i.e. clusters triggered by a neutral charge, most likely a photon) with a transverse energy deposit above a certain threshold (75 MeV), these energies are added to the electron [55–58].

While this procedure improves the signal significantly [58], some issues that are difficult to solve remain, introducing additional error sources. For example, if a neutral cluster can be matched

[2] In principle, the magnitude of Q can always be assumed to be 1, since neutral particles do not curve (and are not detected in the trackers), and all known particles with charge larger than 1 have such short lifetimes that they leave no measurable track.

[3] If, in this section, electrons are mentioned, both electrons and/or positrons are implied.

[4] The ECAL and HCAL (hadronic calorimeters) is not generally used for the momenta measurements because measuring the momenta using the curvature in the magnetic field is simply much more accurate. The main use of the calorimeters is therefore not to accurately measure momenta, but rather to (i) give crude momentum estimates for the trigger (the calorimeters can be read out at the rate of the bunch crossings, whereas track fitting requires too much time); and (ii) to provide additional information for the particle identification.

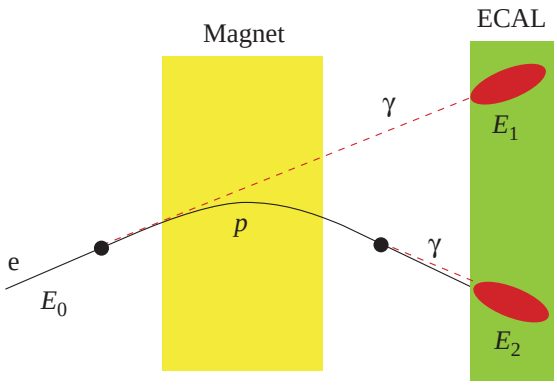

Figure 3.5
Illustration of bremsstrahlung photon recombination. The e track is extrapolated from before the magnet to the electromagnetic calorimeter (ECAL). If there is a neutral cluster with sufficient deposited transverse energy in the neighbourhood of the extrapolation (E_1), then the energy is added to the electron. From Ref. [51].

to multiple different $e^{\pm}$ tracks, the energy is added to one of these tracks at random [57]. In addition, because individual ECAL cells are saturated at transverse energies above 10 GeV, it is not always possible to measure the photon energy; this can degrade the measured electron energies by an amount that is typically of the order of 25 percent [59]. Moreover, the bremsstrahlung photons can convert to e.g. an ee pair before reaching the calorimeter [58]. Finally, it can occur that background photons (i.e. non-bremsstrahlung photons) are matched to electron tracks, which consequently will have an associated momentum that is too high [58].

We return to bremsstrahlung energy loss in Chapter 6, where the theory of bremsstrahlung and its simulation are treated in more detail.

3.1.4 Particle identification

Particle identification for charged particles is based on the combined information from the Ring-Imaging Cherenkov (RICH1–2) detectors, the calorimeters and the muon system. Neutral particles are identified using information from the ECAL. In principle, we only have to be able to distinguish between a handful of particle types that are sufficiently long-lived to reach these detector components; any short-lived particles will decay to these long-lived particles long before. Therefore, for charged particles we need only be able to distinguish between $e^{\pm}$, $\mu^{\pm}$, $\pi^{\pm}$, $K^{\pm}$ and $p^{\pm}$; and for neutral particles between γ and π^0 [51].

Measuring particle properties

Evidently, determining the charge of these particles can hardly go wrong, as it is given by the direction of curvature in the magnetic field.

The velocity of a particle can be determined by the RICH detectors from the Cherenkov radiation it emits. If a particle has a velocity β that is larger than the phase velocity of light ($1/n$, with n the refraction index of the medium), the particle will emit Cherenkov radiation under a fixed angle

θ_c, according to [30]

$$\cos\theta_c = \frac{1}{n\beta}, \tag{3.4}$$

so that a cone of radiation is emitted around the particle's trajectory. This cone can then be detected by the RICH detectors.

In principle, from this cone the velocity can be calculated, which can then be combined with the momentum determined from the track fit, to yields a mass of the particle and thus an identity. However, of course, this cone does not give a perfect signal; therefore, for each (long-lived) particle type, the likelihood that it would generate the observed Cherenkov cone is determined, and the type that has the largest likelihood will be considered to be the identity of the particle by the RICH detectors [51].

Another important type of information is in which calorimeter the particle deposits its energy: e and γ particles deposit their energies mainly in the ECAL; π, K and p in the HCAL; and μ in the muon system (M1–5).

For electrons, also the momentum reconstructed from the track fit is compared to the energy deposition in the ECAL. If these are approximately equal, the particle is likely an electron. In addition, if there is little or no energy deposition in the HCAL along the same trajectory as the deposition in the ECAL, or if the signal in the ECAL is consistent with an electron that has emitted bremsstrahlung, this also increases the likelihood. Finally, the preshower (PS) detector also provides information on whether the particle was an electron.

Photons are also reconstructed using the ECAL. One characteristic of the photon is that it is neutral and consequently leaves no track. Thus, an energy deposit in the ECAL without an associated track is likely to have been a photon. Similarly, a deposit in the HCAL without a track has a high probability of being a π^0.

The signatures of the $\pi^\pm$, $K^\pm$ and $p^\pm$ are then energy deposits in the HCAL, but not in the ECAL or muon system.

Combining the measurements with decision trees

Finally, when the information from all relevant detector components has been collected, each observed particle is assigned a likelihood that it was any of the long-lived particles. This assignment is often done after training a boosted decision tree (BDT) [60], which is an algorithm specifically for classification problems.

The basic idea behind BDTs is the following [62]. Suppose we have an observation x (in our case x contains data from the RICH, the calorimeters, the track fit, etcetera), and we are interested in variable Y (the particle's identity). Then a decision tree T consists of a series of vertices, where at each vertex some feature of x is evaluated (e.g. whether there was any deposit in the ECAL, or whether the track's transverse momentum was within a certain window), which determines which vertex will be evaluated next (see Figure 3.6, which shows an ensemble of decision trees). At the final vertex, the tree returns a probability mass function y for variable Y (in our case, the probability of the particle being a muon, the probability that it is a kaon, etcetera). Mathematically, a decision tree is thus a function that maps a set of input variables to a probability mass function for an output variable.

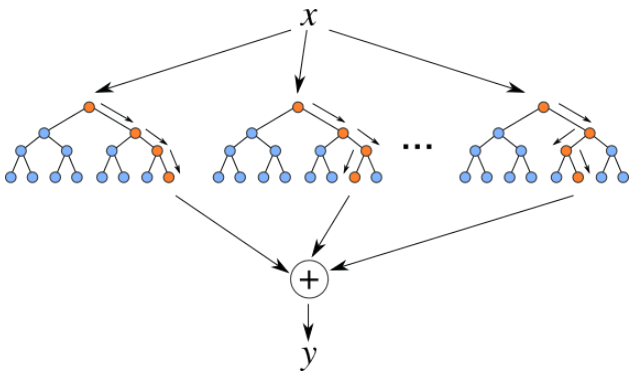

Figure 3.6
Schematic of an ensemble of decision trees. An observation x is the input for each individual tree T_i, which then each return a probability mass distribution y_i for the variable of interest, Y. These distributions y_i are then combined into a single distribution y. For the BDT algorithm, y is a weighted sum of the y_i, where the weights are generated by the boosting procedure. From Ref. [61].

Now, a single tree is likely to give poor predictions for the probability mass function in practice. Instead, predictions can be improved by constructing an ensemble of N different trees T_i, all producing different y_i. Subsequently, if each tree is given a weight w_i, the individual probability mass functions can be combined into a single one, $y = \sum_i w_i y_i$ or $y(x) = \sum_i w_i T_i(x)$. These weights are then determined through boosting, an iterative process that increases the weights of better performing trees versus trees that more often misclassify. The value that is associated with the highest probability in the resulting y is finally assumed to be the real value of Y. That is, in our context, the particle identity that gets assigned the highest probability (or, equivalently, the largest likelihood) is assumed to be the true identity; however, in practice the whole distribution y is saved because it is a measure of how certain it is that the particle indeed has the identity that was assigned. That way, an analysis can be done with different degrees of certainty with respect to the correct identification of particles.

The construction of the trees and the boosting must be done using a training set, where the actual values (the actual particle identities) Y are known. This can be done by selecting events where the identity of a particle is known with certainty, and where the identification is independent of any measurements (e.g. if some decay has a very unique signature, so that it can only be a certain particle). Of course, some care must be taken when constructing the BDT; for example, if the ensemble is made too large, or the trees made too complex, then at some point each observation in the training set will be predicted correctly, while this would obviously not be the case with some new data set. Therefore, training a BDT is a process that must be done with care. For more details, we refer to e.g. Ref. [62]; for uses in analyses, see e.g. Refs. [63, 64].

3.2 Data flow and analysis

Taking raw data is just a small part of the LHCb analysis chain, which is shown in Figure 3.7. To understand the raw data, it must for example be validated against and calibrated to (Monte Carlo) calculations, in order to verify that the data is interpreted correctly, to correct the data for systematic

effects, and to estimate levels of systematic errors.

The data flow consists of roughly five parts: (i) raw data taking by the detector (including the L0 trigger); (ii) Monte Carlo event simulation; (iii) the high level trigger; (iv) preparing the data for analysis; and (v) the analysis itself. Part (i) consists solely of hardware, and parts (ii)–(v) are purely software.

The detector components and the type of data these take, have been discussed in the previous section, where a high-level approach was taken. On a low level, of course, a calorimeter will not give you the energy deposited in a certain cell, but will rather give a series of electronic signals that should somehow be interpreted as the amount of energy in some cell; it is these electronic signals that constitute part (i).

Alternatively, instead of generating data through actual interactions within the detector, such data can also be simulated. For simulating particle physics interactions and detector responses, an extensive package of software is available; these software components, which make up part (ii), will be discussed in Section 3.2.1.

The data output from either part (i) or (ii) is evaluated by the high level trigger [part (iii)], which is discussed above in Section 3.1.1. After this, in part (iv) the selected raw data is used to reconstruct the events (i.e. reconstruct the tracks and vertices, and determine particle identities, as described in the previous section); and these events are combined into data packs (strippings) based on the physics they contain (e.g. all B_s^0 candidate decays are combined into a single pack for further processing and analysis) and structured (in Ntuples, which basically can be seen as the rows in a data table) for analysis. The reconstruction is done by the Brunel software [65], and Stripping and Ntuple making are done in DaVinci [66]; the details of these processes are beyond the scope of this text, and the interested reader is referred to the literature.

Finally, these Ntuples, containing event data regarding a specific physics research question, are analysed. This is often done in the data analysis framework ROOT [67]. ROOT is described in Section 3.2.2.

3.2.1 Event simulation

The general framework for simulating events is Gauss [68], which integrates all the individual steps that make up an event. More specifically, it integrates the generator phase, in which the pp collisions (Pythia) and the decays of the produced particles (EvtGen) are simulated; and the simulation phase, in which the physics processes behind these decay products traversing the detector are simulated (Geant). The latter step also generates simulated hits in the detector.

Afterwards, these simulated hits are forwarded to the phase where the detector response is mimicked (Boole). In this phase e.g. the responses of the tracking system and the L0 trigger are simulated.

Pythia

Pythia [25] is the tool used for generating final states with high multiplicities produced in pp, $p\bar{p}$, e^+e^- or $\mu^+\mu^-$ collisions at high energy. It contains several physics models, based on both earlier empirical research and on theoretical models (e.g. parton and QCD models), which can be used

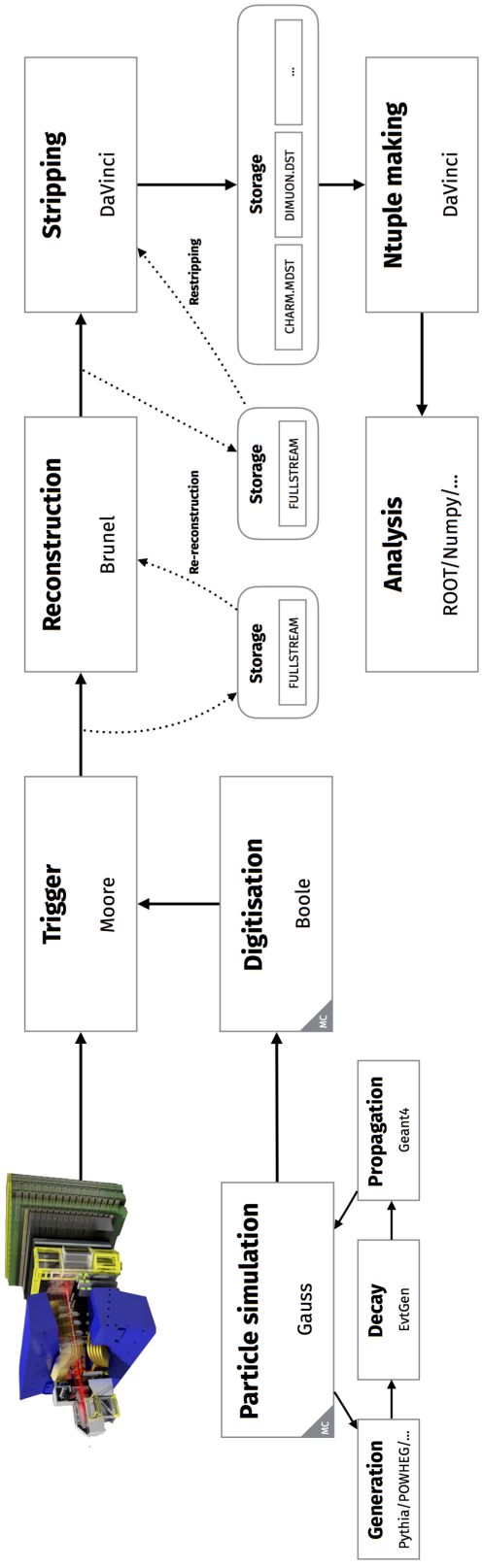

Figure 3.7
The LHCb data flow. From Ref. [29].

to produce the final states. Typically, based on the specific process modelled, Pythia can generate 10–1000 events per second.

EvtGen

EvtGen [26] is a framework for particle decays; in particular for the B mesons, which are of special interest to the LHCb experiment. It can simulate simple two-body decays, but also more complicated decays such as cascade decays containing (multiple) resonances. It also calculates angular distributions for such decays. This can be done using a range of decay models [69]. Its generation rate is comparable to that of Pythia.

Geant

Geant [27, 70] is a toolkit for simulating the interactions of particles with matter. It includes a wide range of processes regarding e.g. scattering and energy loss. Moreover, the simulation includes the detector geometry (e.g. misalignments) and the detector's response to hits. It is thus relatively extensive and complete. The downside is its speed: generating a single event within the LHCb detector using Geant takes time of the order of minutes [29].

Boole

Boole [28] is the final stage of the simulation process, where the hits generated with Geant are digitised. More specifically, the simulated hits are translated to a detector response, including the L0 trigger stage, in a format equivalent to that obtained from real events. It is then ready to be fed to the high level trigger stage, after which it continues the data flow chain. That is, the Boole output receives the same data preparation treatment as real event data, and thus ends up in a similar Ntuple.

3.2.2 ROOT

The most widely used data analysis framework within high-energy physics is ROOT [67]. It is written in C++ and is designed for handling large dataset (in particular, datasets that do not fit into memory), and contains many classes that are related to physics (classes for e.g. particle decays or Lorentz vectors), analysis (classes for e.g. fitting or decision trees) or visualisation (classes for e.g. creating graphs and histograms). Moreover, it offers numerous auxiliary classes for numerical methods (e.g. function minimization) and statistics.

Chapter 4

Simulating particle decays

In this chapter, we turn to the simulation of particle decays. We discuss these decays mainly in a semi-classical relativistic framework; i.e. we focus on the kinematics of particle decays and ignore conservation laws for quantum mechanical quantities such as parity, chirality and angular momentum quantum numbers.

The main complication with particle decays is that they are inherently stochastic. In particular, given a primary particle with mass M and momentum $\mathbf{P}$ that decays into N secondary particles with masses $\{m_1, \dots, m_N\}$ (see Figure 4.1), the momentum vectors $\{\mathbf{p}_1, \dots, \mathbf{p}_N\}$ of the secondary particles are random (subject to the constraint that four-momentum is conserved). However, in general we cannot derive the probability distributions of these momentum vectors $\mathbf{p}_i$ analytically, and for that reason we instead rely on Monte Carlo methods for approximating these distributions.

In the following, we start by briefly reviewing the relevant relativistic kinematics in Section 4.1. The simple case of two-body decays will be treated in Section 4.2, for which the probability density functions for the momentum vectors can be derived analytically, after which Section 4.3 extends the analysis to N-body decays. In subsequent chapters we will discuss to what extent $\mathbf{p}_i$ can actually be observed within a detector, and how this observable is affected by detector resolution (Chapter 5) and energy losses (Chapter 6), and what the effect is of assigning wrong masses m_i (Chapter 7).

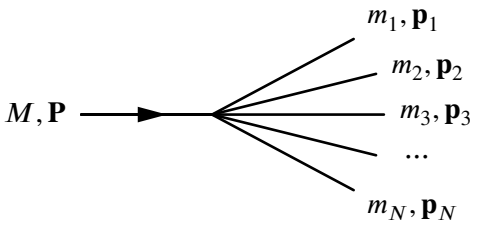

Figure 4.1
A general N-body decay, where a primary particle with mass M decays into N secondary particles with masses $\{m_1, \dots, m_N\}$ via an $(N+1)$-point interaction.

4.1 Physics of particle decays

The physics behind particle decays has been broadly discussed in Chapter 2. Among other things, the particles and interactions in the Standard Model were discussed, and how (and according to what rules) these particles decay. In this section, we will briefly revisit the partial decay rate, [Equations (2.36)–(2.37)]

$$\mathrm{d}\Gamma_{i\to f} = \frac{(2\pi)^4}{2M}|\mathcal{M}_{i\to f}|^2\,\delta^4\left(P - \sum_{j=1}^{N} p_j\right)\prod_{j=1}^{N}\frac{\mathrm{d}^3\mathbf{p}_j}{(2\pi)^3\,2E_j}, \tag{4.1}$$

and discuss its relevance to the simulation of particle decays.

4.1.1 Simplifications regarding the amplitude

As mentioned earlier, the partial decay rate is essentially governed by the Feynman amplitude $\mathcal{M}_{i\to f}$ and the phase space constraints. The physics contained in the amplitude has already been qualitatively treated. While a more detailed or quantitative treatment is beyond our scope, some remarks about the amplitude are in order, and about the assumptions and simplifications that we will make.

Most importantly, $\mathcal{M}$ may depend on the angles the secondary particles make with respect to some particular axis, e.g. with respect to the spin of the primary particle. In general, however, the particle of interest will not be polarized and thus its spin direction will be randomized. We can then average over this random spin to obtain a decay that can be treated as isotropic, i.e. angle-independent.

A possible exception to this approach is if the decaying particle is produced in a weak interaction, which couples only to particles with left-handed chirality. In that case, the particle has its spin aligned (anti-)parallel to its momentum and its decay will not be isotropic. However, since such angular dependences do not show up in experimental mass spectra, we can safely ignore this possibility. For a detailed treatment of the angular distributions of decays, see e.g. Ref. [71].

In addition, it should be emphasised that we are not concerned about the absolute magnitude of the amplitude. In fact, in our simulation we will ignore it completely; rather, it will simply be assumed that the decay defined by the user is physically allowed, and that the amplitude is a constant. That is, taking these simplifications together, we explicitly assume

$$\forall i, f\ :\quad |\mathcal{M}_{i\to f}|^2 = \text{constant} > 0. \tag{4.2}$$

While this is certainly not physically correct, it is a useful assumption for two reasons: first, not concerning ourselves with whether a decay is allowed greatly simplifies the simulation; and, second, and more important, it allows for the simulation of beyond Standard Model decays (e.g. $B^0 \to e\mu$).

4.1.2 Phase space constraints

Regarding phase space, it is only assumed that only total four-momentum must be conserved, as governed by the delta function in equation (4.1). Other than that, no other constraints are taken into account. For instance, all states, both initial and final, are assumed to have zero angular momentum.

4.1.3 Reconstructing the primary particle

Given the masses and momenta of the secondary particles (i.e. the particles produced in a decay), one can simply impose total four-momentum conservation to obtain the mass and momentum of the primary particle. Specifically, the mass of the primary particle, M, is obtained from

$$M^2 = \left(\sum_i P_i^\mu \right)^2 = \left[\sum_{i=1}^{N} E_i \right]^2 - \left| \sum_{i=1}^{N} \mathbf{p}_i \right|^2 , \tag{4.3}$$

with

$$E_i = \sqrt{|\mathbf{p}|^2 + m_i^2}, \tag{4.4}$$

where the summations are over all secondary particles.

As described in Chapter 3, the momenta are determined from the curvature of the particles in the detector's magnetic field. The masses are indirectly obtained using the particle identities, which are also determined by the detector. The code for the reconstruction of the primary particles can be found in Appendix A.7. We will now turn to the simulation of particle decays.

4.2 Two-body decays

Before turning to the general case of the N-body decay, we first examine the simplest type of decay: the two-body decay. An obvious advantage of treating two-body decays first is that the mathematics is still analytically tractable. Therefore it is a more appropriate model for gaining insights in the kinematics of particle decays. Moreover, two-body decays make up a large share of the decays investigated at experiments. N-body decays will subsequently be examined in Section 4.3.

Consider a particle with mass M at rest, decaying to two particles $i = 1, 2$. Four-momentum conservation requires that the rest frame energy E_i^* of the secondary particle i will be fixed at

$$E_i^* = \frac{1}{2M}(M^2 + m_i^2 - m_j^2) \tag{4.5}$$

where j denotes the other particle. If the primary particle had momentum P in the lab frame, the energies of the secondary particles in the lab frame are given by

$$E_i = \gamma(E_i^* + \upsilon p_i^* \cos\theta_i^*), \tag{4.6}$$

where θ_i^* is the polar angle at which the particle is emitted in the rest frame with respect to the boost direction (i.e. the flight direction of the primary particle). Because two-body decays are constrained to a two-dimensional plane, we can fix the problem to the xz-plane and thus set the azimuthal angle $\phi_i^* = \phi_i = 0$.[1] Since momentum conservation requires that the secondary particles are emitted back-to-back in the rest frame, we have the restriction $\theta_2^* = \theta_1^* + \pi$. As the decay is isotropic,

[1] Alternatively, we can define ϕ_i^* to be uniformly distributed, and note that it is defines a direction perpendicular to the boosting direction, so that $\phi_i^* = \phi_i$; and that this angle is the same for both particles as they are produced in the same plane.

the solid angle distribution is constant [i.e. $\mathrm{pdf}(\Omega_i^*) = 1/4\pi$], and since $\mathrm{d}\Omega_i^* = \mathrm{d}\cos\theta_i^*\,\mathrm{d}\phi_i^*$, the distribution of θ_i^* must be such that $\cos\theta_i^*$ is uniformly distributed:

$$\mathrm{pdf}(\cos\theta_i^*) = \begin{cases} 1/2 & \text{for } \cos\theta_i^* \in [-1,1] \\ 0 & \text{else} \end{cases} . \tag{4.7}$$

4.2.1 Generation of two-body decay events

Given a primary particle of given mass M and energy E, and given the masses of the secondary particles, m_1 and m_2, equations (4.5–4.7) are sufficient to simulate two particle decays. Specifically, the following steps generate one decay event:

1. Define a primary particle with mass M, energy[2] E and a flight direction given by the angles (θ, ϕ) (with respect to the z-axis in the lab frame).
2. Define two secondary particles with masses m_1 and m_2, respectively.
3. Calculate the energies E_i^* the secondary particles will have in the rest frame. Given m_i, these also define the (three-)momenta $p_i^* = |\mathbf{p}_i^*|$.
4. Draw an angle θ_1^* from distribution (4.7), and set $\theta_2^* = \theta_1^* + \pi$.
5. Optionally, draw an angle $\phi_1^* = \phi_1$ from a uniform distribution ($\mathrm{pdf}(\phi_i^*) = 1/2\pi$ for $\phi_i^* \in [-\pi, \pi]$), and set $\phi_2^* = \phi_2 = \phi_1^*$.
6. Boost the secondary particles to the lab frame using equation (4.6), along the lab frames z-axis.
7. Calculate the momentum directions θ_i of the secondary particles using

$$\theta_i = \arctan\left(\frac{p_i^* \sin\theta_i^*}{\gamma(p_i^* \cos\theta_i^* + \upsilon E_i^*)} + \theta\right). \tag{4.8}$$

Steps 3–6 can be repeated to generate a set of independent observations for a specific decay. The implementation of this algorithm can be found in Appendix A.3.

4.2.2 Validation: distributions of relevant observables

It is informative to examine the distributions of certain relevant observables, both to verify that the simulation algorithm works, and to study the kinematic dependence of these observables. Below we compare several simulated distributions to the corresponding analytical solutions. As an example, we take the decay $B^\pm \to J/\psi\, K^\pm$ with $\gamma_{B^\pm} = 15$.

[2] One can also define a decay width Γ, so that there is a Breit-Wigner randomness in M.

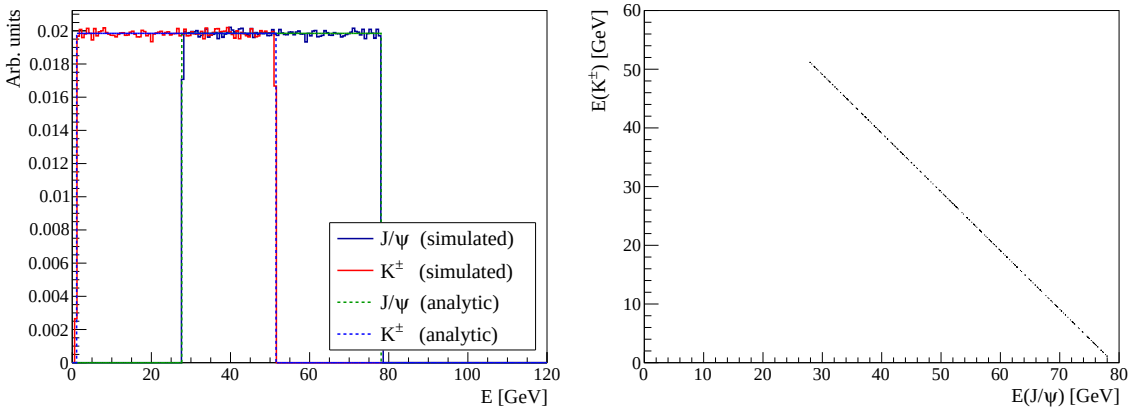

Figure 4.2
Energy distributions for the decay products of $B^{\pm} \to J/\psi K^{\pm}$ decays, generated using the two-body decay algorithm from Section 4.2.1. Left: the (lab frame) energy distributions of both particles, compared to the analytical expression for the distribution (4.12). Right: the relation between the J/ψ and $K^{\pm}$ energies, where each dot represents an observation. It is assumed that $B^{\pm}$ has $\gamma = 15$.

Secondary particle energy and momentum distributions

To obtain an energy distribution of the secondary particles, a change of variables from θ_i^* to E_i must be applied to the angular distribution (4.7). That is, defining $\tilde{\theta}_i^* \equiv \cos\theta_i^*$,

$$\text{pdf}(E) = \text{pdf}(\tilde{\theta}_i^*(E_i)) \left| \frac{\mathrm{d}\tilde{\theta}_i^*(E_i)}{\mathrm{d}E_i} \right|, \tag{4.9}$$

with, from the boost equation (4.6),

$$\tilde{\theta}_i^*(E_i) = \frac{E_i - \gamma E_i^*}{\gamma \upsilon p_i^*} \tag{4.10}$$

and

$$\text{pdf}(\tilde{\theta}_i^*(E_i)) = \frac{1}{2} \qquad \text{for } \tilde{\theta}_i^*(E_i) \in [-1, 1]. \tag{4.11}$$

gives us the energy distribution of the secondary particles:

$$\text{pdf}(E_i) = \frac{1}{2\gamma \upsilon p_i^*} \qquad \text{for } \gamma(E_i^* - \upsilon p_i^*) < E_i < \gamma(E_i^* + \upsilon p_i^*). \tag{4.12}$$

Of course, if E_1 is drawn from this distribution, E_2 is automatically fixed by kinematic constraints. In fact, from (4.10) and using that $\cos\theta_1^* = -\cos\theta_2^*$ (and $\phi_1^* = \phi_2^*$) for back-to-back decays, we find that

$$E_1 + E_2 = \gamma(E_1^* + E_2^*). \tag{4.13}$$

Equation (4.12) is compared to simulations in Figure 4.2, and one can see that there is good correspondence. This confirms the correctness of equations (4.12) and (4.13).

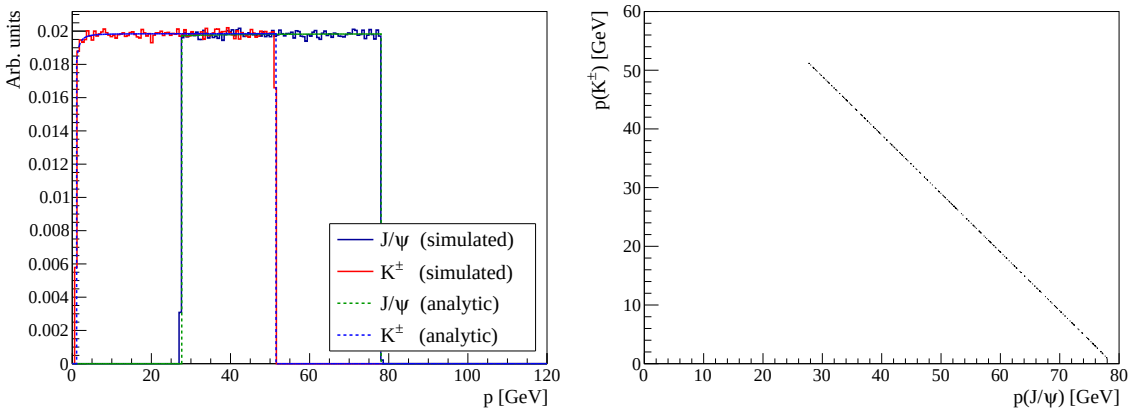

Figure 4.3
Momentum distributions for the decay products of $B^{\pm} \to J/\psi K^{\pm}$ decays, generated using the two-body decay algorithm from Section 4.2.1. Left: the (lab frame) momentum distributions of both particles, compared to the analytical expression for the distribution (4.15). Right: the relation between the J/ψ and $K^{\pm}$ momenta, where each dot represents an observation. It is assumed that $B^{\pm}$ has $\gamma = 15$.

Similarly, with

$$\tilde{\theta}_i^*(p_i) = \frac{\sqrt{p_i^2 + m_i^2} - \gamma E_i^*}{\gamma \upsilon p_i^*}, \tag{4.14}$$

we can calculate the momentum distributions of the particles,

$$\text{pdf}(p_i) = \frac{1}{2\gamma \upsilon p_i^*} \frac{p_i}{\sqrt{p_i^2 + m_i^2}} \tag{4.15}$$

$$\text{for } \sqrt{\gamma^2 (E_i^* - \upsilon p_i^*)^2 - m_i^2} \leq p_i \leq \sqrt{\gamma^2 (E_i^* + \upsilon p_i^*)^2 - m_i^2},$$

which is shown in Figure 4.3. Note that for sufficiently large γ of the primary particle, the distributions for energy and momentum are approximately equal. That is, at high energies, the momentum distribution is also approximately uniform. This is not the case at lower energies, as can be seen in Figure 4.4, which shows the momentum distributions for $\gamma = 2$; these are significantly non-uniform.

Inverse energy and momentum fraction distributions

We can define an inverse energy fraction[3]

$$\xi_i = \frac{\sum_j E_j}{E_i} = \frac{E}{E_i}, \tag{4.16}$$

[3]This definition is adapted from the work of Van Veghel [72], who similarly defines the inverse momentum fraction.

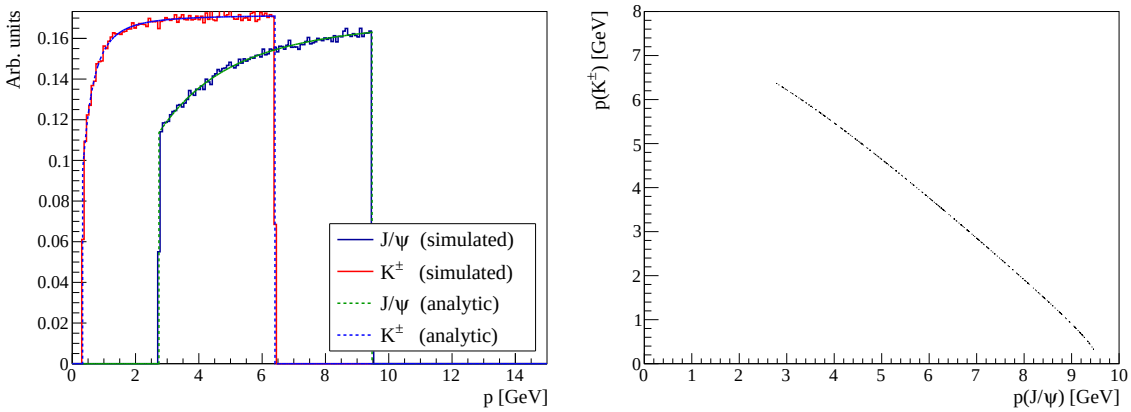

Figure 4.4
Same figure as Figure 4.3, but for $B^{\pm}$ with $\gamma = 2$.

which is a measure of how evenly the energy is distributed among the two particles. By a change of variables, then, given a primary particle with energy E and mass M, we can derive

$$\begin{aligned} \mathrm{pdf}(\xi_i) &= \mathrm{pdf}(E_i(\xi_i)) \left| \frac{dE_i}{d\xi_i} \right| \\ &= \frac{E}{2\gamma v p_i^*} \frac{1}{\xi_i^2} \\ &\text{for } \frac{M}{E_i^* + v p_i^*} \leq \xi_i \leq \frac{M}{E_i^* - v p_i^*}. \end{aligned} \tag{4.17}$$

This equation is compared to simulated distributions for ξ in Figure 4.5. Again, the left panel shows full agreement between the generated events and the analytic solution. In addition, the right panel shows that the phase space density is higher for configurations where both particles have relatively moderate inverse energy fractions (the density of points is highest at the bottom left part of the curve).

4.3 N-body decays

We will now turn to the general case in which N particles are produced in a decay. Again, we emphasize that we make the simplifying assumption that all decays from some state $|i\rangle$ to some $|f\rangle$ (ignoring possible intermediate states) are produced according to

$$\begin{aligned} \mathrm{d}\Gamma_{i\to f} &= \text{constant} \times \mathrm{d}\Phi_N(P, p_1, \ldots, p_N) \\ &= \text{constant} \times \delta^4\left(P - \sum_{j=1}^{N} p_j\right) \prod_{j=1}^{N} \frac{\mathrm{d}^3\mathbf{p}_j}{(2\pi)^3 2E_j} \end{aligned} \tag{4.18}$$

That is, we do not concern ourselves with the absolute magnitude of the amplitude or whether the decay is allowed, and we ignore angular dependences due to e.g. spin or helicity effects.

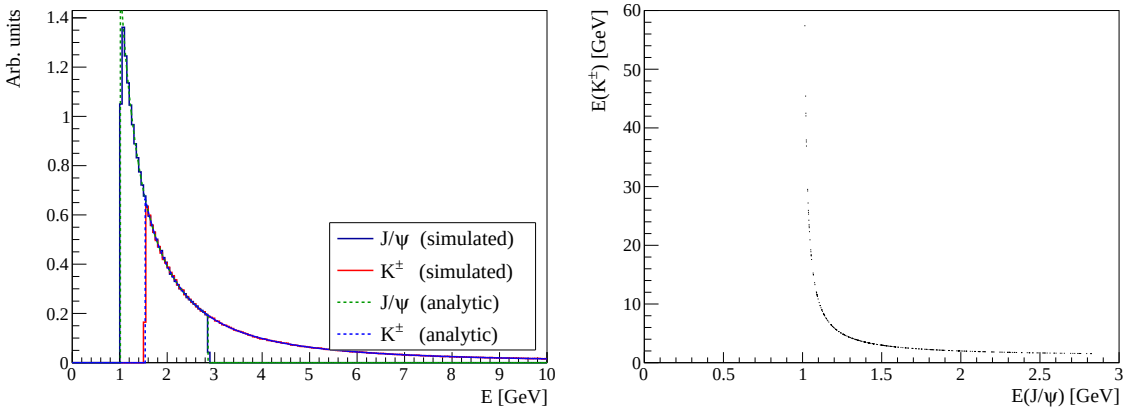

Figure 4.5
Inverse energy fraction distributions for the decay products of $B^{\pm} \to J/\psi K^{\pm}$ decays, generated using the two-body decay algorithm from Section 4.2.1. Left: the (lab frame) inverse energy fraction distributions of both particles, compared to the analytical expression for the distribution (4.17). Right: the relation between the J/ψ and $K^{\pm}$ inverse energy fractions, where each dot represents an observation. It is assumed that $B^{\pm}$ has $\gamma = 15$.

In that case, it is natural to view the partial decay rate (4.18) as a probability distribution for the kinematic configuration $\{(E_1, \mathbf{p}_1), \ldots, (E_N, \mathbf{p}_N)\}$ of the secondary particles produced in the decay (where, of course, the energies E_i are fixed by the momenta $\mathbf{p}_i$ and the predetermined masses m_i). The Monte Carlo simulation of such decays is then a matter of drawing configurations from this distribution.

4.3.1 Generation of N-body decay events[4]

An important realization is that each allowed configuration (where degenerate states count as different configurations) is equally likely, which is a standard assumption from statistical mechanics. Then, the approach that is generally taken, is to make uniform draws from all allowed different configurations (i.e. degenerate states are counted as a single configuration), and give these draws weights proportional to their degeneracies [73]. This approach yields exactly the phase space differential (4.18).

The most straightforward solution would then be to first define a $3N$-dimensional hypercube in $(p_{1x}, p_{1y}, p_{1z}, \ldots, p_{Nx}, p_{Ny}, p_{Nz})$-space, such that it includes all allowed configurations. Then one uniformly draws $3N - 4$ momentum components from a $(3N - 4)$-dimensional subspace of this cube; the last four components are then fixed by four-momentum conservation. If this yields a physically allowed state, then give it a weight proportional to $1/\prod_i E_i$ to obtain a draw from distribution (4.18). However, in many cases this will not yield a kinematically allowed configuration, because the actual boundary of the allowed configurations is not the surface of a cube, but rather a smooth surface within this cube. That is, while the $3N$-dimensional cube is defined such that it includes all allowed configurations, it also contains many non-physical states. As a result, this naive approach quickly becomes prohibitively inefficient.

[4]The algorithms and their derivations presented in this section are taken from James [73].

This efficiency can be somewhat increased by expressing (4.18) in terms of kinetic energies T_i; if we change variables to spherical coordinates and make the identification $T_i = |\mathbf{p}_i|^2/2m_i$, we obtain a distribution of the form

$$\mathrm{d}\Phi_N \propto \delta^4\left(P - \sum_{j=1}^{N} p_j\right) \prod_{j=1}^{N} \frac{|\mathbf{p}_j|}{E_j}\mathrm{d}T_j\,\mathrm{d}\cos\theta_j\,\mathrm{d}\phi_j. \tag{4.19}$$

This solution still requires drawing $3N-4$ numbers (and fixing the other 4), but the space from which draws are made represents the volume of allowed configurations considerably better than the cube discussed above, and therefore increases the efficiency of the algorithm. However, this approach is also insufficiently fast as N becomes larger.

Alternatively, invariant masses can be used to define the configuration space, rather than the momentum vectors or kinetic energies. The main idea is then to recursively draw two invariant masses: one for one of the particles, and one for all other remaining particles. Then, once the former particle has obtained its invariant mass, it is split off from the system, and the step is repeated. Essentially, the method is to treat the decay as a sequence of two-body decays, where in each two-body decay one particle is split off until all particles are split off.

To accomplish this, it can be shown that the phase space $\mathrm{d}\Phi_N$ can be separated into two subsystems containing respectively $N-k$ and k particles by writing

$$\mathrm{d}\Phi_N(P, p_1, \ldots, p_N) = \mathrm{d}\Phi_{N-k+1}(P, P_k, p_{k+1}, \ldots, p_N)\mathrm{d}\Phi_k(P_k, p_1, \ldots, p_k)\mathrm{d}P_k^2, \tag{4.20}$$

where $P = \sum_{i=1}^{N} p_i$ is the total four-momentum of all particles, and $P_k = \sum_{i=1}^{k} p_i$ is the total four-momentum of the k-particle subsystem. But if this is done repeatedly, we have

$$\mathrm{d}\Phi_N(P, p_1, \ldots, p_N) = \mathrm{d}P_{N-1}^2 \cdots \mathrm{d}P_2^2 \prod_{i=1}^{N-1} \mathrm{d}\Phi_2(P_{i+1}, P_i, p_{i+1}). \tag{4.21}$$

That is, the problem is indeed reduced to a sequence of two-body decays. Since the two-body phase space integral $\mathrm{d}\Phi_2$ is known,

$$\mathrm{d}\Phi_2(P_{i+1}, P_i, p_{i+1}) = \frac{2\pi}{|P_{i+1}|}\sqrt{|P_{i+1}| + \left(\frac{P_i^2 - p_{i+1}^2}{|P_{i+1}|}\right)^2 - 2\left(P_i^2 + p_{i+1}^2\right)}, \tag{4.22}$$

it can be plugged in to give, using $\mathrm{d}P_i^2 = 2|P_i|\,\mathrm{d}|P_i|$, and after some rearranging,

$$\mathrm{d}\Phi_N(P, p_1, \ldots, p_N) = \frac{\mathrm{d}|P_2| \cdots \mathrm{d}|P_{N-1}|}{|p_1|} \prod_{i=1}^{N-1} 2|P_i|\,\mathrm{d}\Phi_2(P_{i+1}, P_i, p_{i+1}). \tag{4.23}$$

Note that we have obtained an expression purely in terms of the invariant masses of two-body subsystems. The fact that we can indeed describe an N-body decay as a sequence of two-body decays greatly simplifies the problem, since the kinematics of two-body decays are fixed. Moreover, we automatically generate values for θ_i and ϕ_i by recognizing that the single particle and the

remaining subsystem of particles will decay back-to-back according to an isotropic distribution (in the decaying subsystem's rest frame).

The only question that remains is how to draw values for the invariant masses of the subsystems $(|P_2|, \ldots, |P_{N-1}|)$, as these are not fixed (contrary to the invariant masses of single particles). The main restriction regarding the invariant masses $(|P_2|, \ldots, |P_{N-1}|)$ is that four-momentum conservation requires that for all $|P_j|$, in the primary particle's rest frame,

$$\sum_{i=1}^{j} m_i < |P_j| < M - \sum_{i=j+1}^{N} m_i, \tag{4.24}$$

where M is again the primary particle's mass. That is, the invariant mass of a subsystem must exceed the sum of the masses of the particles it contains, and must be less than the total invariant mass minus the masses of the particles that it does not contain.

Now, if $N-2$ independent uniformly distributed random numbers $\{r_2, \ldots, r_{N-1}\} \in \langle 0, 1\rangle$ are generated,[5] then constraint (4.24) is satisfied if we take[6]

$$|P_j| = r_j \left(M - \sum_{i=1}^{N} m_i \right) + \sum_{i=1}^{j} m_i. \tag{4.25}$$

Then, to get the ordering correct, such that

$$|P_{j-1}| + m_j < |P_j| < |P_{j+1}| - m_{j+i}, \tag{4.26}$$

it is sufficient to order the r_j such that

$$r_2 < r_2 < \cdots < r_{N-1}. \tag{4.27}$$

Conveniently, then, the problem of drawing from the N-body decay distribution (4.18) comes down to drawing a sequence of uniformly distributed numbers and ordering them. The corresponding event weight w is obtained from (4.23),

$$w \propto \frac{1}{m_1} \prod_{i=1}^{N-1} 2|P_i| \, \mathrm{d}\Phi_2(P_{i+1}, P_i, p_{i+1}), \tag{4.28}$$

[5] For $j = 1$ and $j = N$, we consider subsystems of single particles, which have a fixed, i.e. non-random, invariant mass. Therefore no random numbers are drawn for these subsystems.

[6] The left-hand inequality in (4.24) holds if we note that $|P| > \sum_{i=1}^{N} m_i$ and require $r_j > 0$, and the right-hand inequality is shown to hold if we rearrange (4.25) to

$$M - \sum_{i=j+1}^{N} m_i = \frac{1}{r_j} \left[|P_j| - (1 - r_j) \sum_{i=1}^{j} m_i \right] = \frac{1}{r_j} \left[|P_j| - \sum_{i=1}^{j} m_i \right] + \sum_{i=1}^{j} m_i,$$

and confirm that the last term

$$\frac{1}{r_j} \left[|P_j| - \sum_{i=1}^{j} m_i \right] + \sum_{i=1}^{j} m_i > |P_j| \quad \Leftrightarrow \quad \frac{1}{r_j} \left[|P_j| - \sum_{i=1}^{j} m_i \right] > |P_j| - \sum_{i=1}^{j} m_i$$

indeed satisfies (4.24) if $r_j < 1$. [This derivation is given explicitly because Ref. [73] has printed an erroneous definition of (4.24).]

with $\mathrm{d}\Phi_2$ given by (4.22).

To summarize, the algorithm for generating one N-body decay consists of the following steps (given that all primary and secondary particles have been appropriately defined).

1. Draw $N-2$ independent uniformly distributed random numbers $\{r_2, \ldots, r_{N-1}\} \in \langle 0, 1\rangle$, and sort them in ascending order.

2. Calculate invariant masses $\{|P_2|, \ldots, |P_{N-1}|\}$ using these random numbers and equation (4.25). In addition, use $|P_1| = m_1$ and $|P_N| = |P| = M$.

3. For each subsystem of j particles, do the following steps in descending order of j, i.e. $j = N, N-1, \ldots, 1$. Each time, consider the steps in the rest frame of the j-particle subsystem.

   (a) Determine a direction in which the j-th particle is produced, by drawing independent uniformly distributed values for $\cos\theta_j \in [-1, 1]$ and $\phi \in [-\pi, \pi]$ (i.e. the decays are isotropic). The other particles are treated as a subsystem of $j-1$ particles, which is produced in the opposite direction.

   (b) Calculate the three-momentum of the j-th particle by imposing four-momentum conservation,
   $$p_j = P_j - P_{j-1}, \tag{4.29}$$
   where $P_j = (|P_j|, \mathbf{0})$, since we are in the rest frame of the j-particle subsystem, and the direction of the three-momenta in p_j and P_{j-1} are as determined in the previous step.

   (c) Switch to the rest frame of the $(j-1)$-particle subsystem, and redo these steps for that subsystem, until $j = 1$ where the drawn configuration is completely specified.

The implementation of the N-body decay generator class can be found in Appendix A.4.1.[7]

4.3.2 Validation

Again, it is important to verify that distributions of relevant variables are reproduced by our algorithm. This was extensively done for the two-body decay simulation in Section 4.2.2, and the N-body decay simulation should of course produce similar results for two-body decays. It can be seen from Figure 4.6 that this is indeed the case for the energy distributions of the secondary particles; i.e. the simulation reproduces the analytical energy distributions (4.12), and returns a plot which is similar to Figure 4.3. The same holds for the momentum and inverse energy fraction distributions, but these results are not shown here.

As a second validation, it is checked whether the simulation reproduces the correct Dalitz plots for three-body decays. Such diagrams plot the squared invariant mass of one pair of particles versus the squared invariant mass of another pair; e.g. it plots $(p_1^\mu + p_2^\mu)^2$ versus $(p_2^\mu + p_3^\mu)^2$. For such plots, it can be shown that all events must fall in a region similar to that depicted in the left panel of Figure 4.7. Moreover, because for three-body decays it can be derived that [48]

[7] Note that the algorithm presented above is implemented in terms of ROOT's `TGenPhaseSpace` class. The class shown in Appendix A.4.1 is therefore rather trivial, and its main purpose is to provide an appropriate interface for other parts of the simulation program.

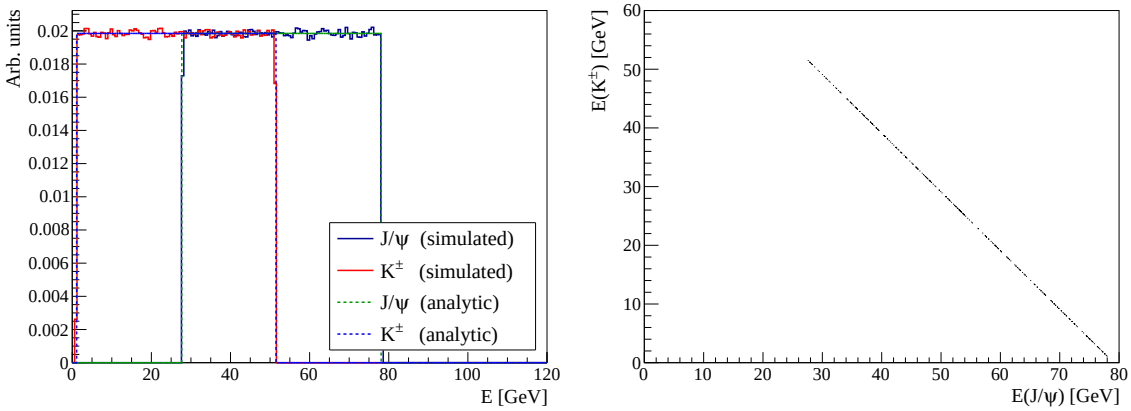

Figure 4.6
Energy distribution for the decay products of $B^{\pm} \to J/\psi K^{\pm}$ decays, generated using the N-body decay algorithm from Section 4.3.1. Left: the (lab frame) energy distributions of both particles, compared to the analytical expression for the distribution (4.12). Right: the relation between the J/ψ and $K^{\pm}$ energies, where each dot represents an observation. It is assumed that $B^{\pm}$ has $\gamma = 15$.

$$\mathrm{d}\Phi_3 \propto \mathrm{d}m_{12}^2\,\mathrm{d}m_{23}^2 \tag{4.30}$$

(or permutations thereof), we see that the events must be distributed uniformly within this allowed region.

The right panel of Figure 4.7 shows the same decay as the left panel, but is produced by our simulation (Section 4.3.1). It correctly reproduces both the kinematically allowed region, and the predicted uniform distribution within that region, which verifies that the used algorithm is produces correct results.

4.4 Cascade decays

In many cases, a particle does not decay directly to some set of N particles, but rather via some intermediate resonance state (see Section 2.3.3). For many decays, such resonance channels even dominate the non-resonance channel in terms of branching fractions [36]. Because, as we have found in Section 2.3.3, such resonances effectively fix the invariant mass of a subset of the decay products [up to the decay width of the resonance, given by the Breit-Wigner distribution (2.41)], these resonances should turn up non-trivially (i.e. non-uniformly) in Dalitz plots.

For example, in terms of Figure 4.7, if an intermediate resonance fixes the mass of of the $\pi^+\bar{K}^0$ pair, we would expect this resonance to produce an area of high density around $m(\pi^+\bar{K}^0)^2 = M_R^2$ (i.e. a vertical line), where M_R is the (Breit-Wigner distributed) mass of the resonance state.

Now, for the decay of some particle A to N secondary particles,

$$A \to 1 + 2 + \cdots + N, \tag{4.31}$$

the total decay rate is given by the sum of the decay rate for the non-resonant channel (i.e. where the primary particle decays directly to the secondary particles, without any intermediate resonance

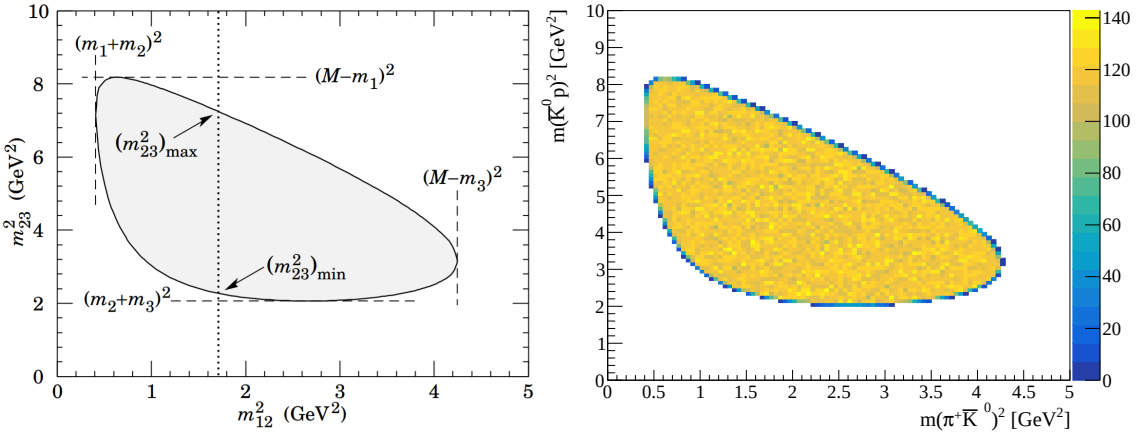

Figure 4.7
Left: Dalitz plot showing the allowed region for three-body decays (in this example, the secondary particles are $\pi^+\bar{K}^0 p$ produced from a 3 GeV particle). Each axis shows the squared invariant mass of a pair of particles, i.e. m_{ij}^2 is the squared invariant mass of particles i and j. From Ref. [36]. Right: the same decay simulated by the algorithm from Section 4.3.1.

states) and the decay rates for each possible resonant channel.[8] As a result, the Dalitz plots produced by each decay channel should also be additive, taking into account that each Dalitz plot should have a weight proportional to the branching ratio of the corresponding decay channel.

It is perhaps easiest to illustrate this with an example. Suppose we wish to study the decay

$$D_s^+ \to K^+K^-\pi^+. \tag{4.32}$$

The dominating channels for this decay are [36]

$$D_s^+ \to \phi\pi^+ \qquad \text{BR} = 2.27\% \tag{4.33a}$$
$$\hookrightarrow K^+K^-$$

$$D_s^+ \to K^+\bar{K}^*(892)^0 \qquad \text{BR} = 2.61\% \tag{4.33b}$$
$$\hookrightarrow K^-\pi^+$$

$$D_s^+ \to f_0(980)\pi^+ \qquad \text{BR} = 1.15\% \tag{4.33c}$$
$$\hookrightarrow K^+K^-$$

where the branching ratios (BR) have been given in terms of the full decay rate of the D_s^+ (i.e. $D_s^+ \to$ anything). For the non-resonant channel, a branching ratio of 8.1×10^{-4} [74] is assumed.

[8] In reality, this summation is not so straightforward, because rather than simply summing the decay rates, one has to sum the amplitudes $\mathcal{M}$ for each possible channel to obtain the total decay rate. That is, if we denote the amplitude for the decay via resonance state R_j as $\mathcal{M}_{i\to R_j\to f}$, and the non-resonant decay as $\mathcal{M}_{i\to f}$, the total decay rate contains the term $\left|\mathcal{M}_{i\to f} + \sum_{R_j}\mathcal{M}_{i\to R_j\to f}\right|^2$. Evidently, this gives rise to cross-terms in the total amplitude, and thus to quantum mechanical interference effects. As a result, the effective decay rates via some channels may be suppressed or boosted by these effects, and these effects are often dependent on kinematics (in particular, there often is angular dependence). Such effects also show up in Dalitz plots, but are beyond the scope of this text; therefore, we will ignore this point and assume these different decay channels to be additive.

In principle, the decays (4.33) are just sequences of N-body decays. For example, mode (4.33a) is simply a two-body decay from D_s^+ to $\phi\pi^+$, after which the produced ϕ decays (also via a two-body process, in this case) to K^+K^-. The only additional difficulty that arises in such decays is that the resonance has a relatively large width, and this must be taken into account. That is, it must be ensured that the mass of the ϕ resonance (and thus of the resulting K^+K^- pair) is Breit-Wigner distributed according to equation (2.41); and that energy and momentum are conserved at each step of the sequential decay.

4.4.1 Generation of cascade decay events

If we implement the sequential decay procedure as described above, we arrive at the following concrete steps. Again, it is assumed that all primary, secondary and intermediate particles have been appropriately defined.

1. For each step of the decay, execute the following procedure. In terms of decay (4.33a), decay step 1 would be the $D_s^+ \to \phi\pi^+$ decay; step 2 would be the $\phi \to K^+K^-$ decay.
   (a) Determine the masses of the particles that are produced in the decay step. For stable particles, this mass is a fixed number; for unstable particles, a mass must be drawn from the Breit-Wigner distribution (2.41).
   (b) Once the masses are defined by the previous step, we can simulate the decay step as a regular N-body decay using the routine given in Section 4.3.1.
2. Recursively proceed until all steps of the decay have been simulated.

The implementation of this routine is given in Appendix A.4.2.

4.4.2 Validation and shortcomings

For the evaluation of our approach, we return to our example decay, $D_s^+ \to K^+K^-\pi^+$, and will compare the Dalitz plot produced by our simulation with experimental observations for this decay from the CLEO Collaboration [74]. Our simulation results and the CLEO results are both shown in Figure 4.8.

It can be seen that the simulated plots agree well with the experimental results. In particular, the shape of the kinematically allowed region is again correctly reproduced, as well as the locations of the resonance bands.

However, an exception is the resonance channel via a $\bar{K}^*(892)^0$, which has an incorrect shape. More specifically, the simulation produces a horizontal resonance band around $m(K^-\pi^+)^2 = m(\bar{K}^*(892)^0)^2$ which is uniform across different values for $m(K^+K^-)^2$. The experimental data, on the other hand, do not seem to contain a significant number of $\bar{K}^*(892)^0$ decays in the region $2.0 < m(K^+K^-)^2 < 2.5$, implying that certain K^+K^- configurations are suppressed.

Such kinematic suppressions generally originate from an angular dependency in the matrix element $\mathcal{M}$. While angular dependencies average out in many cases, this is often not true for cascade decays. In our example, the D_s^+ decays weakly, which produces a left-handed $\bar{K}^*(892)^0$; as a consequence, the $\bar{K}^*(892)^0$ always have the same spin orientation, so that the spin orientations no longer

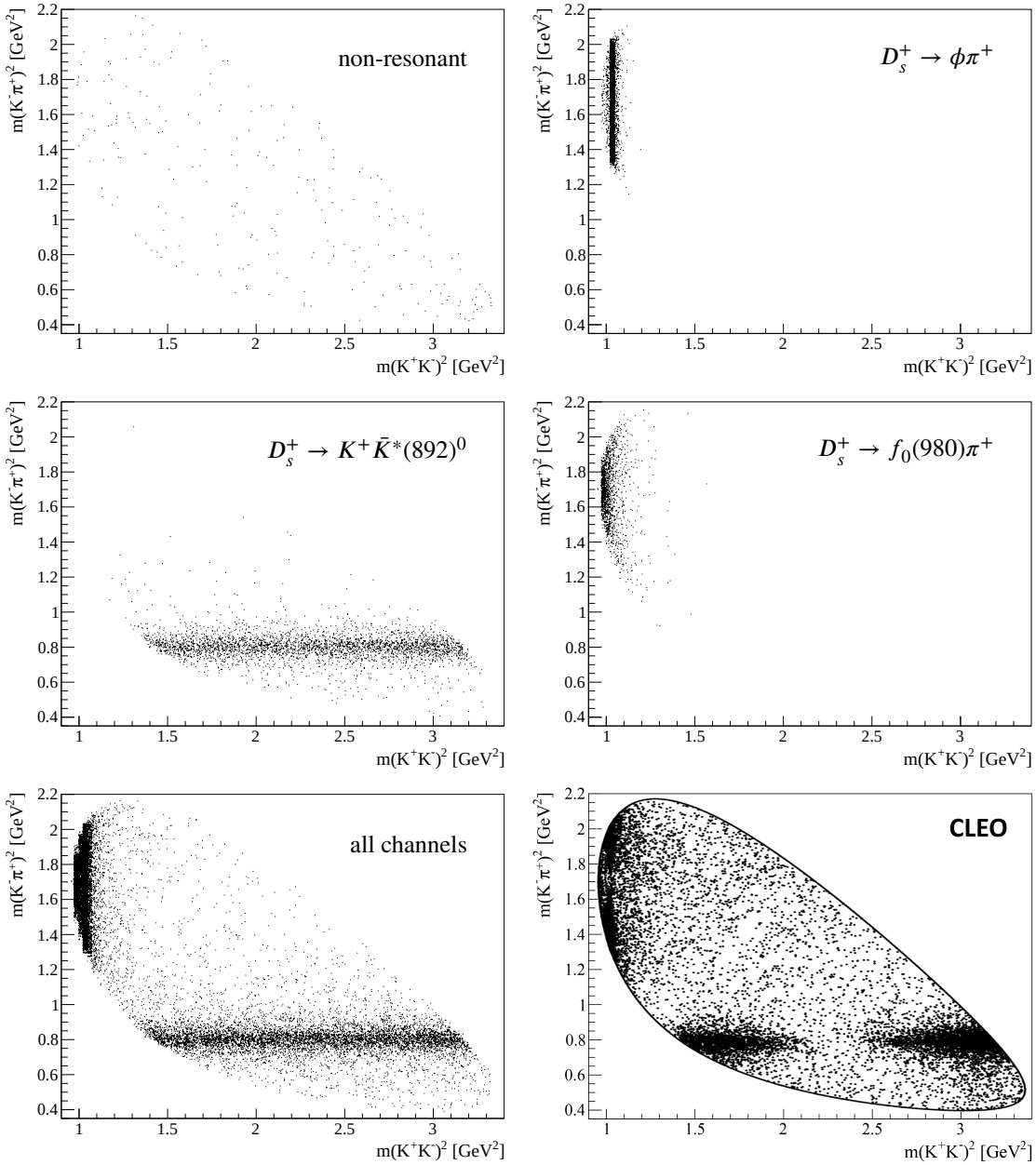

Figure 4.8
Dalitz plots for the decay $D_s^+ \to K^+K^-\pi^+$. The top four panels show the most important contributions to the total Dalitz plot, each simulated using the procedure of Section 4.4.1: the non-resonant decay and the decays (4.33). The bottom left panel shows the total Dalitz plot that is obtained by superimposing the top four panels. The bottom right panel shows the experimental data from the CLEO experiment [74] (note: the figure from Ref. [74] has been adapted to fit the other panels).

average out and we obtain an angle-dependent amplitude $\mathcal{M}$. By assuming that $\mathcal{M}$ = constant, we thus explicitly ignore these and similar effects, and it is important to keep this shortcoming in mind.

Nonetheless, it should also be noted that in many cases, this assumption is still valid. In our example, the plots for the ϕ and $f_0(980)$ resonance channels are still correct, due to the fact that these are spin-0 particles. Then, there is no preferred direction in the system, in which case decays should always be isotropic. Therefore, in many cases our simplification is not problematic, and our simulation will provide valid results. We will see how we can use these event generation routines to simulate experimental mass spectra in the following chapters.

Chapter 5

Simulating resolution and acceptance

Now that we have an appropriate procedure for simulating decays, we can turn to the simulation of particle detection. In particular, to obtain useful simulations, detector effects have to be taken into account, while keeping the model relatively simple. Therefore, only the most important effects will be considered: uncertainty effects, detector acceptance and bremsstrahlung. Minor issues such as misalignment or detection asymmetries are ignored, since these would greatly complicate the model while offering only marginal improvements. Finally, in the following chapters, the simplification is made that the detector only measures the momentum[1] p and direction (θ, ϕ) of the particle, and that the particle identity that gives that gives the particle mass m is set by the user.

5.1 Measurement error

It is generally assumed that the error in momentum is normally distributed with momentum-dependent standard deviation $\sigma_p(p)$, which is often taken of the form [51, 75]

$$\frac{\sigma_p(p)}{p} = a + bp, \tag{5.1}$$

with a and b constants. For LHCb, these values are typically around [75]

$$a = 3 \times 10^{-3} \quad \text{and} \quad b = 2 \times 10^{-5}\ \text{GeV}^{-1}. \tag{5.2}$$

To obtain an energy distribution which includes such uncertainty effects, the energy distribution has to be convoluted with the error distribution. Assuming normally distributed errors and $\sigma_p(E) \approx \sigma_p(p)$ (high-energy particles), the distribution of the observed particle energy, denoted $\tilde{E}_i$, becomes

$$\text{pdf}(\tilde{E}_i) = \frac{1}{\sqrt{2\pi}} \int \text{d}E_i \, \frac{\text{pdf}(E_i)}{\sigma_p(E_i)} \exp\left[-\frac{1}{2}\left(\frac{E_i - \tilde{E}_i}{\sigma_p(E_i)}\right)^2\right]. \tag{5.3}$$

The integration limits are the same limits as those on the probability distribution function of E_i.

The effect of the momentum-dependent error on the mass spectrum is shown in Figure 5.1,

[1] At this point, it is convenient to switch notation: rather than denoting the magnitude of the three-momentum as $|\mathbf{p}|$, it is simply denoted p.

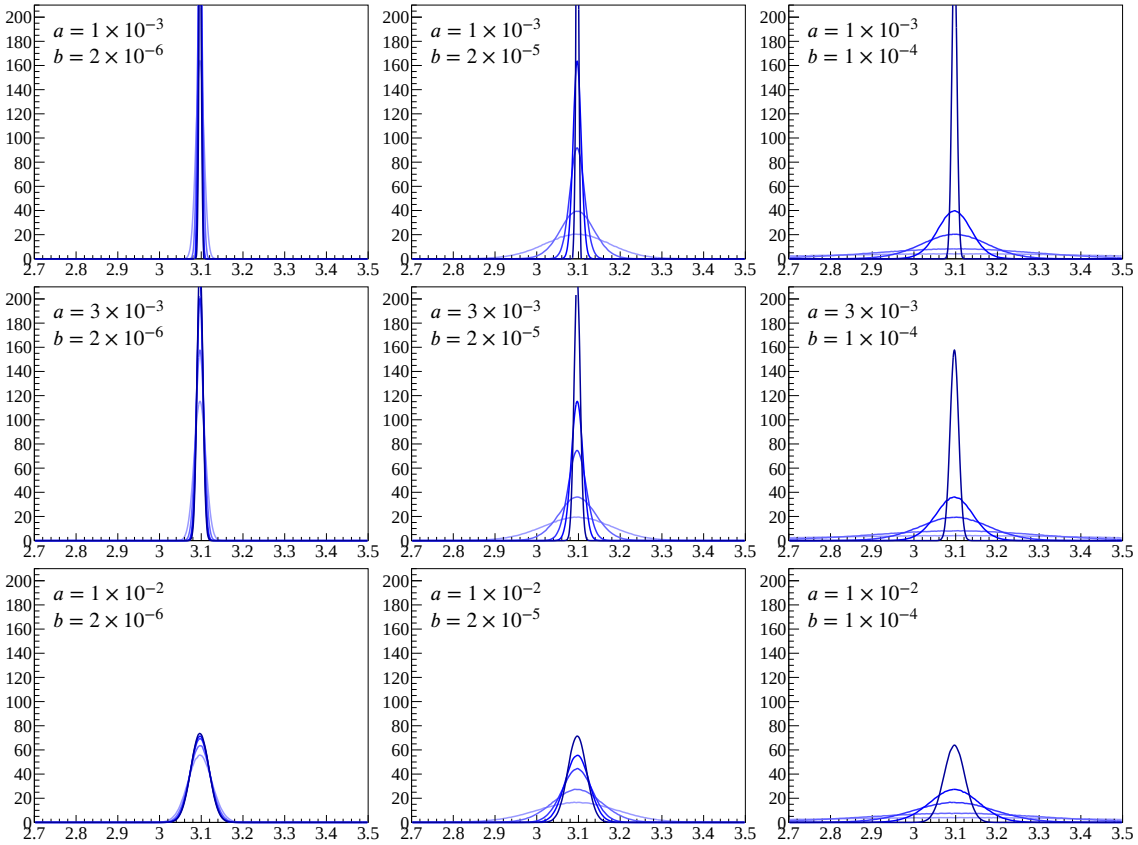

Figure 5.1
Mass spectra for $J/\psi \to \mu\mu$, assuming Gaussian momentum resolution with momentum-dependent error (5.1) with different values of parameters a and b (the latter is in units of GeV^{-1}). Each panel shows the spectrum for different J/ψ momenta: from narrow to wide (or from dark to light blue), the curves are for $\gamma_{J/\psi} = 10, 100, 200, 500, 1000$. The x-axis shows $m(\mu\mu)$; the y-axis shows the relative frequency in arbitrary units.

which shows mass spectra for $J/\psi \to \mu\mu$ decays for different values of a and b, and for different $\gamma_{J/\psi}$. As expected, the peaks indeed become wider as a and b increase. However, it can also be seen that for low values of b, the shape of the peak depends much less on the energy of the primary J/ψ; as b is increased the $\gamma_{J/\psi}$ dependence increases significantly. This is less the case for a, which seems much more important for the width of the peak in general.

Figure 5.2 shows the same for the three-body decay $J/\psi \to \pi^0\pi^+\pi^-$, from which the same conclusions can be drawn. However, we should note the difference that for three-body decays the peaks are slightly sharper. This is due to the fact that for three-body decays, each particle will on average have lower momentum than particles produced in two-body decays, which yields lower relative errors. Moreover, as more particles are produced in a decay, their average momentum error will converge to zero by the law of large numbers, so that the error on the reconstructed mass will be lower.

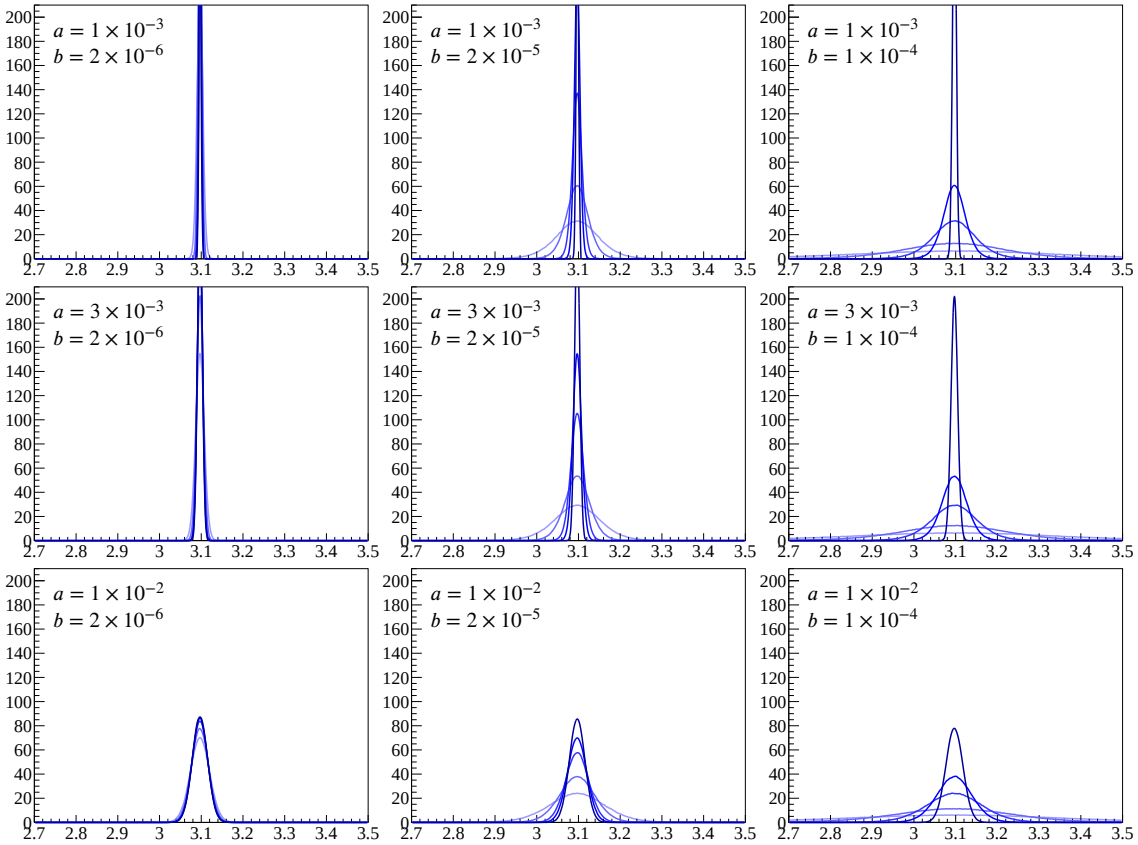

Figure 5.2
Same figure as Figure 5.1, but for $J/\psi \to \pi^0\pi^+\pi^-$.

Simulating these effects is straightforward: after a decay is generated, from which a set of momenta p_i is obtained, we simply replace the true momenteum with the observed momentum (i.e. $p_i \to \tilde{p}_i$), where the observed momentum $\tilde{p}_i$ is drawn from a Gaussian distribution with mean p_i and standard deviation $\sigma_p(p_i)$.[2]

5.2 Acceptances

Not all decays will be registered by the detector, simply because the detector has physical limitations, or because it has been tuned to ignore events if they do not match certain criteria. In such cases, it can be assumed that events that do not match these criteria are never registered and reconstructed. For example, for the LHCb detector, particles outside the pseudorapidity range $2 < \eta < 5$ (which roughly corresponds to $|\theta| < 0.27$ rad) will fly out of the detector and will thus not be observed. Generally, in simulations such acceptances can just be implemented in terms of simple cut-offs; that is, the event is only registered and processed if all user-defined criteria are met.

[2]Currently, we consider only the momentum magnitude p_i in our simulation. However, it is possible to extend the simulation such that each momentum component is smeared individually, or such that the smearing is non-Gaussian.

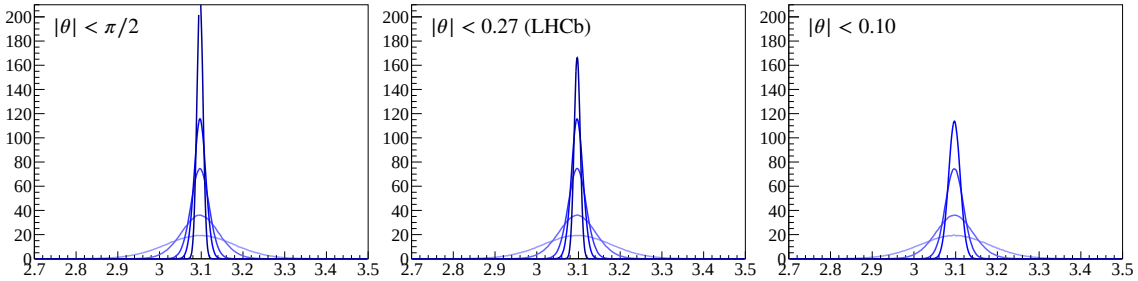

Figure 5.3
Mass spectra for $J/\psi \to \mu\mu$, for different angular acceptances. All particles must fall within the acceptance for the event to be registered. Each panel shows the spectrum for different J/ψ momenta: from dark to light blue, the curves are for $\gamma_{J/\psi} = 10, 100, 200, 500, 1000$. A Gaussian mass resolution with error (5.1) is assumed, with parameters (5.2). The x-axis shows $m(\mu\mu)$; the y-axis shows the relative frequency in arbitrary units.

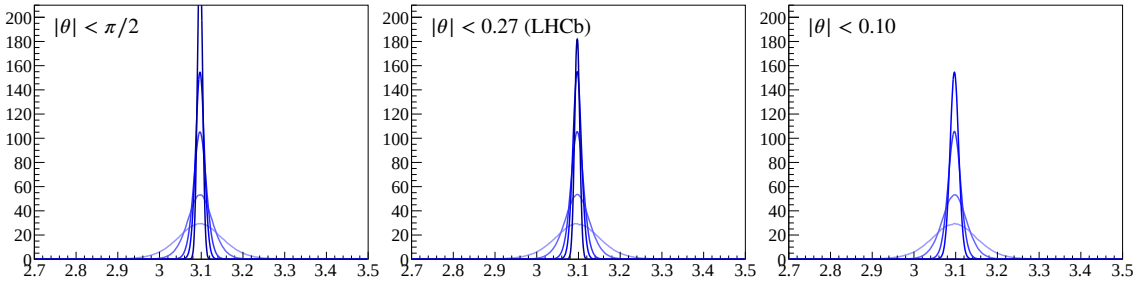

Figure 5.4
Same figure as Figure 5.3, but for $J/\psi \to \pi^0\pi^+\pi^-$.

As before, we use $J/\psi \to \mu\mu$ and $J/\psi \to \pi^0\pi^+\pi^-$ decays to illustrate how acceptances affect the mass spectrum shape. For angular acceptance, this is shown in Figures 5.3 and 5.4, respectively. From these figures we can clearly see a crucial dependence on the decay energy. More specifically, it can be seen that the acceptance angle only affects low-energy decays, while it does not change the shape or amplitude of high-energy events. That is, as the acceptance angle is made smaller, the $\gamma_{J/\psi} = 10$ peak gradually shrinks, whereas the other peaks remain unaltered. In principle, this is to be expected, as the decay energy defines a maximum angle in which the decay products can be produced (in the lab frame). Then, high-energy decays will always emit their decay products within the acceptance, whereas this will not always be the case for lower-energies decays.

For momentum acceptance, see Figures 5.5 and 5.6. A similar dependence on decay energy (i.e. $\gamma_{J/\psi}$) can be seen, where the acceptance region affects only those decays which have primary particle energies below a certain threshold, as such low-energy primary particles are the only particles for which there are decay configurations where not all particles meet the acceptance requirements. That is, the higher the decay energy, the smaller the probability of producing a low-momentum secondary particle; then, if the decay energy is sufficiently high, the probability of producing a particle that has momentum below the acceptance limit becomes negligible or even

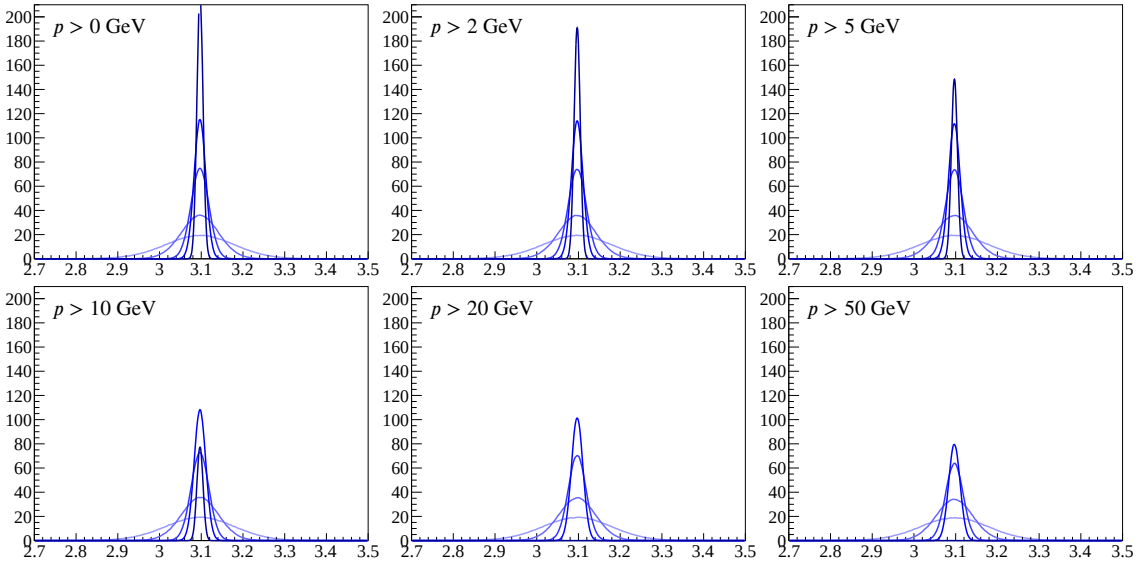

Figure 5.5
Mass spectra for $J/\psi \to \mu\mu$, for different momentum acceptances. All particles must fall within the acceptance for the event to be registered. Each panel shows the spectrum for different J/ψ momenta: from dark to light blue, the curves are for $\gamma_{J/\psi} = 10, 100, 200, 500, 1000$. A Gaussian mass resolution with error (5.1) is assumed, with parameters (5.2). The x-axis shows $m(\mu\mu)$; the y-axis shows the relative frequency in arbitrary units.

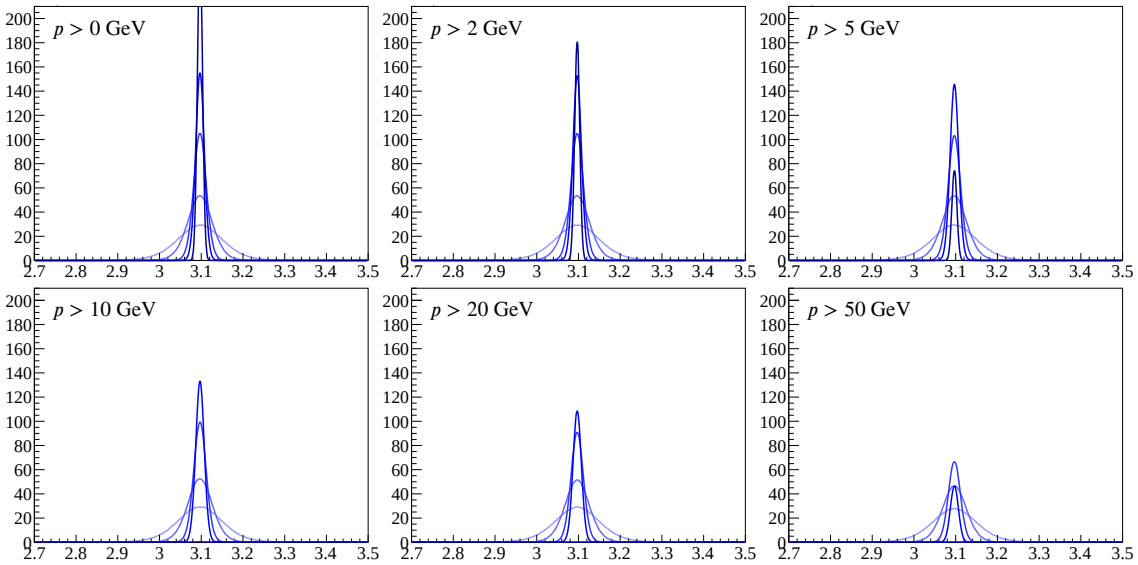

Figure 5.6
Same figure as Figure 5.5, but for $J/\psi \to \pi^0\pi^+\pi^-$.

zero if such configurations are no longer kinemetically allowed.

As with resolution effects, acceptances are relatively easy to simulate. In principle, after a decay has been generated, the simulation should simply check whether all acceptance requirements are met. If all variables are within the acceptance, the simulation can proceed to simulate detector effects and reconstruct the primary mass; otherwise, the event will be ignored and no primary mass will be reconstructed.

Chapter 6

Simulating energy loss effects

When particles pass through detector material, they will lose energy by interacting with atoms in the material. This is particularly important for electrons, which can lose a significant fraction of their energy through such interactions (by photon emission); in fact, for high-energy electrons ($E \gg 10$ MeV) bremsstrahlung is the dominating process for energy loss, such that the effects of other processes can be ignored [76]. Therefore, for electrons and positrons, only energy loss effects related to bremsstrahlung will be considered.

Since the rate of energy loss in general depends crucially on the particle's mass, energy losses for heavier particles are orders of magnitude smaller than the loss rates for electrons. Then, especially if the amount of material is kept to a minimum, as should be the case in detectors, energy losses for particles other than electrons or positrons can generally be ignored.[1]

6.1 Theory of bremsstrahlung energy loss

In the limit of (ultra)relativistic electrons, we can use the complete screening approximation to obtain the differential bremsstrahlung cross section [76, 78–81][2]

$$\frac{\mathrm{d}\sigma}{\mathrm{d}k} = 4\alpha r_e^2 \frac{1}{k}\left\{\left[\frac{4}{3} - \frac{4}{3}y + y^2\right]\left[Z^2\left(F_{\mathrm{el}} - f(Z)\right) + Z F_{\mathrm{inel}}\right] + \frac{1}{9}\left[1 - y\right]\left[Z^2 + Z\right]\right\} \tag{6.1}$$

where k is the momentum of the emitted photon, $y \equiv k/T$ is the fraction of electron kinetic energy T (in the rest frame of the nucleus, which in our case is also the detector rest frame) lost to the photon, $\alpha = e^2/4\pi\epsilon_0\hbar c$ is the fine-structure constant, $r_e = e^2/4\pi\epsilon_0 m_e c^2$ the classical electron

[1] For example, the energy loss by bremsstrahlung scales as $\mathrm{d}E/\mathrm{d}x \sim 1/m^2$, where m is the particle mass. Bremsstrahlung energy losses in muons then become important only at the TeV scale [77].

[2] Note that this cross section diverges for $k \to 0$. This infrared divergence will be removed by the Ter-Mikaelian effect discussed below.

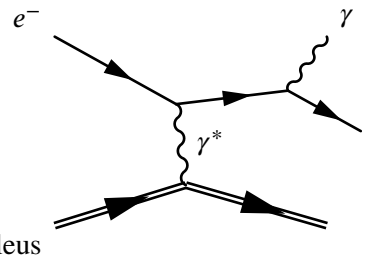

Figure 6.1
Feynman diagram of bremsstrahlung.

radius, F_{el} and F_{inel} are the elastic and inelastic atomic form factors related to screening effects,[3]

$$F_{\text{el}} = \ln\Big(\frac{184.15}{Z^{1/3}}\Big) \qquad\qquad F_{\text{inel}} = \ln\Big(\frac{1194}{Z^{2/3}}\Big),$$

and $f(Z)$ is the Coulomb correction[4] [80, 82]

$$f(Z) = \sum_{n=1}^{\infty} \frac{(\alpha Z)^2}{n\left[n^2 + (\alpha Z)^2\right]}. \tag{6.2}$$

In equation (6.1), the contributions proportional to Z^2 are due to scattering of the incident electrons off the nucleus; those proportional to Z are due to electron-electron bremsstrahlung.[5]

The expected number N_k of photons emitted with energy between k and $k+\mathrm{d}k$ while traversing a unit of length $\mathrm{d}x$ is then

$$\frac{\mathrm{d}N_k}{\mathrm{d}x} = N_{\text{atoms}}\,\frac{\mathrm{d}\sigma}{\mathrm{d}k}, \tag{6.3}$$

with N_{atoms} the number of atoms per unit volume. In terms of Avogadro's number N_A, the atomic mass A, and the material mass density ρ, we can write $N_{\text{atoms}} = N_A\rho/A$ so that we have an expected energy loss per unit length of

$$\left\langle \frac{\mathrm{d}E}{\mathrm{d}x} \right\rangle = -\frac{N_A\rho}{A}\int_0^T \mathrm{d}k\; k\,\frac{\mathrm{d}\sigma}{\mathrm{d}k}, \tag{6.4}$$

and we directly obtain

$$\frac{1}{E}\left\langle \frac{\mathrm{d}E}{\mathrm{d}x} \right\rangle = -4\alpha r_e^2 \frac{N_A\rho}{A}\left[Z^2\left(F_{\text{el}} - f(Z)\right) + Z F_{\text{inel}} + \frac{Z^2+Z}{18}\right]. \tag{6.5}$$

[3]Screening is related to the distance scale of the collisions: at large collision distances (large impact parameters) the potential due to the positive charge in the atom nucleus is screened by the negatively charged electron cloud (intuitively, from an infinite distance an atom looks like a neutral particle rather than a charged nucleus with an oppositely charged cloud surrounding it); if the collision is very close to the nucleus, no such screening takes place. For relativistic applications, we consider complete screening.

[4]The Coulomb correction accounts for the higher-order diagrams through which bremsstrahlung can take place. The (classical) Coulomb potential follows from the Feynman diagram where a single photon is exchanged without any intermediate loops; however, in reality, more complicated diagrams are also possible, and these should be included as well by means of the aforementioned Coulomb correction.

[5]Note that in many texts, the electron-electron contributions are neglected (e.g. Ref. [78]), so that equation (6.1) is proportional only to Z^2.

Defining the radiation length X_0 (in units of g/cm^2) of a material[6] as

$$\frac{1}{X_0} = 4\alpha r_e^2 \frac{N_A}{A} \left[Z^2 \left(F_{\text{el}} - f(Z) \right) + Z F_{\text{inel}} \right], \tag{6.6}$$

and ignoring the last term in (6.5) (this term is more than one order of magnitude smaller than the other terms, and is therefore often ignored in other texts), we obtain the well-known

$$\langle E(x) \rangle \approx E_0 e^{-t/X_0}, \tag{6.7}$$

where $t = x\rho$ is the thickness traversed in terms of the distance x crossed and the material density ρ, so that an electron on average loses half its energy after traversing a material of thickness $t_{1/2} = X_0 \ln 2$.

The cross section (6.1) is accurate for all values of y except at the endpoints. For high photon energies ($y \approx 1$), the complete screening approximation may become invalid (it assumes that both the incoming and outgoing electron are relativistic) and one should resort to more complicated empirically determined functions (see e.g. Refs. [78, 80, 83]). However, given that the error introduced by ignoring this issue is relatively small, the approximation will nonetheless be used for large y as well. For low photon energies ($y \approx 0$), the Landau–Pomeranchuk–Migdal (LPM) effect [84] and the Ter-Mikaelian effect [85] will play a role by suppressing the cross section for low-energy photon emission.

6.1.1 Landau–Pomeranchuk–Migdal effect

The intuition behind the LPM effect is as follows [81]. Due to the uncertainty relation (with regard to the momentum transfer from the electron to the photon), the process of bremsstrahlung formation takes place over a non-zero path length of the electron.[7] As such, in sufficiently dense materials, the electron wave function passes multiple atoms when producing a bremsstrahlung photon. However, during this process, the electron wave function scatters off these atoms, such that any created photon wave functions may destructively interfere, thus disturbing the photon production process. This happens in particular for photons with wavelengths over a certain threshold, such that the momentum k of the emitted photon is below [81]

$$k < k_{\text{LPM}} = \frac{8\pi\hbar c\gamma^2}{\alpha X_0/\rho}, \tag{6.8}$$

where γ is the gamma factor of the incident electron. Typically, for a 25 GeV electron, k_{LPM} is a few MeV for materials consisting of light elements, and around 500 MeV for the heaviest materials.

[6] In the case of compound materials with each material j having fractional weight w_j, one can compute each material's radiation length X_j using equation (6.6) and combine them according to $1/X_0 = \sum_j w_j/X_j$.

[7] Qualitatively, as the electron traverses the material, its wave function splits up in a photon wave function and an electron wave function, which both move in separate directions. Initially, at the point at which the electron wave function splits, these two wave functions overlap to such extent that they should still be considered as a single particle; only after the wave functions have separated sufficiently (one Compton wavelength) can they be considered as two separate particles. As a consequence, one considers the photon to be emitted over a finite distance, referred to as the formation length.

Note that if we define

$$y_{\mathrm{LPM}} = \frac{k_{\mathrm{LPM}}}{E} = \frac{8\pi\hbar c}{\alpha X_0/\rho}\frac{E}{\left(m_e c^2\right)^2}, \tag{6.9}$$

we find that the LPM effect indeed becomes increasingly important for higher electron energies, and can reduce the energy loss significantly for ultrarelativistic particles.

The LPM effect enters the bremsstrahlung cross section through the energy-dependent corrections $\xi(s)$, $\phi(s)$ and $\psi(s)$ such that [84, 86]

$$\frac{\mathrm{d}\sigma}{\mathrm{d}k} = 4\alpha r_e^2 \frac{1}{k}\left\{\xi(s)\left[\frac{4}{3}\phi(s) - \frac{4}{3}y\phi(s) + y^2\psi(s)\right]\left[Z^2\left(F_{\mathrm{el}} - f(Z)\right) + ZF_{\mathrm{inel}}\right] + \frac{1}{9}\left[1-y\right]\left[Z^2+Z\right]\right\}, \tag{6.10}$$

where s is recursively defined by the relations

$$s\sqrt{\xi(s)} = \frac{1}{2}\left[\frac{E}{E-k}\frac{k}{k_{\mathrm{LPM}}}\right]^{1/2} \tag{6.11}$$

$$\xi(s) = \begin{cases} 2 & \text{for } s \le s_1 \\ 1 + \ln(s)/\ln(s_1) & \text{for } s_1 < s < 1 \\ 1 & \text{for } s \ge 1 \end{cases} \tag{6.12}$$

$$s_1 = \left(\frac{Z^{1/3}}{184.15}\right)^2 \tag{6.13}$$

Note that s is defined such that the LPM effect becomes important for $s < 1$. Ref. [86] presents a direct method for approximating s: first, define (note the omission of $\xi(s)$ in the definition of s')

$$s' = \frac{1}{2}\left[\frac{E}{E-k}\frac{k}{k_{\mathrm{LPM}}}\right]^{1/2} \tag{6.14}$$

$$\xi'(s') = \begin{cases} 2 & \text{for } s' \le \sqrt{2}s_1 \\ 1 + h - 0.08(1-h)\left[1-(1-h)^2\right]/\ln\left(\sqrt{2}s_1\right) & \text{for } \sqrt{2}s_1 < s' < 1 \\ 1 & \text{for } s' \ge 1 \end{cases} \tag{6.15}$$

$$h(s') = \ln\left(s'\right)/\ln\left(\sqrt{2}s_1\right) \tag{6.16}$$

after which s can be approximated (up to 0.05% accuracy) by

$$s \approx s'/\sqrt{\xi'(s')}. \tag{6.17}$$

The obtained value for s can then be used to obtain the values of the functions (again with approx-

imations, up to 0.15% accuracy, from Ref. [86])

$$\phi(s) = 12s^2\left[-\frac{\pi}{2} + \int_0^\infty \mathrm{d}t\, e^{-st}\coth(t/2)\sin(st)\right] \tag{6.18a}$$

$$\approx \begin{cases} 4s & \text{for } s \leq 0.01 \\ 1 - \exp\left\{-6s\left[1 + (3-\pi)s\right] + s^3/(0.623 + 0.796s + 0.658s^2)\right\} & \text{for } 0.01 < s < 2 \\ 1 & \text{for } s \geq 2 \end{cases} \tag{6.18b}$$

and

$$\psi(s) = \frac{1}{3}\left\{2\phi(s) + 24s^2\left[\frac{\pi}{2} - \int_0^\infty \mathrm{d}t\, e^{-st}\frac{\sin(st)}{\sinh(t/2)}\right]\right\} \tag{6.19a}$$

$$\approx \begin{cases} 4s & \text{for } s \leq 0.01 \\ 1 - \exp\left\{-4s - 8s^2/(1 + 3.936s + 4.97s^2 - 0.05s^3 + 7.50s^4)\right\} & \text{for } 0.01 < s < 2 \\ 1 & \text{for } s \geq 2 \end{cases} \tag{6.19b}$$

Indeed, for $s \gg 1$ we see that $\xi(s)$, $\phi(s)$ and $\psi(s)$ all go to unity and the LPM cross section (6.10) reduces to the original Bethe-Heitler cross section (6.1).

6.1.2 Ter-Mikaelian effect

At the lowest photon energies, the leading cause for suppression is the Ter-Mikaelian effect [85], also called the dielectric effect or longitudinal density effect. As with the LPM effect, this is a consequence of the fact that the photon emission takes place over a finite (i.e. non-zero) distance. In the context of the Ter-Mikaelian effect, when a photon is emitted by the electron, the wave function of the photon is phase-shifted (with respect to the electron wave function) due to the dielectric constant of the medium, and thus loses coherence with the electron wave function; as a consequence, the photon emission is suppressed [87].

Decoherence due to such phase shifts occurs only for photons with sufficiently long wavelengths. In particular, the suppression occurs for photon emissions with a momentum below the (boosted) plasma momentum of the medium [88],

$$k_p = \gamma\hbar\left[4\pi\alpha\hbar c\,\frac{N_A\rho}{A}\frac{Z}{m_e}\right]^{1/2}, \tag{6.20}$$

which is typically of order $\gamma \cdot \mathrm{eV}$. As such, the fractional energy loss suppression due to this effect is of order $1\ \mathrm{eV}/m_e \sim 10^{-6}$ and can thus be safely neglected.

However, as the suppression has the effect of decreasing the LPM cross section (6.10) by a multiplicative factor

$$S(k) = \frac{k^2}{k^2 + k_p^2}, \tag{6.21}$$

the Ter-Mikaelian effect does provide a mechanism for removing the infrared divergence present in the Bethe-Heitler cross section (6.1), and is therefore included nonetheless.

We should finally also account for the effects of the LPM and Ter-Mikaelian effects on each other (the suppressions do not simply add up or multiply). As the latter dominates for the lowest photon energies, one can solve this by 'turning off' the LPM suppression for photon momenta below the plasma momentum, by making the replacement [88][8]

$$s \to \hat{s} = s \cdot \left(1 + \frac{k_p^2}{k^2}\right). \tag{6.22}$$

Explicitly, the final bremsstrahlung cross section is then

$$\frac{\mathrm{d}\sigma}{\mathrm{d}k} = 4\alpha r_e^2 \frac{S(k)}{k} \left\{ \xi(\hat{s}) \left[\frac{4}{3}\phi(\hat{s}) - \frac{4}{3}y\phi(\hat{s}) + y^2\psi(\hat{s})\right] \left[Z^2\left(F_{\mathrm{el}} - f(Z)\right) + ZF_{\mathrm{inel}}\right] + \frac{1}{9}\left[1-y\right]\left[Z^2 + Z\right]\right\}. \tag{6.23}$$

6.1.3 Electron-positron asymmetries

So far, our discussion has only considered electrons. Positrons, on the other hand, behave slightly different: first, because they can annihilate with the atomic electrons; and second, because they are oppositely charged, and are thus repelled rather than attracted by the atomic nuclei (and vice versa for the atomic electrons). This is especially important for low-energy positrons, which often cannot reach the nucleus as a result, and thus have significantly lower bremsstrahlung cross sections compared with electrons [89].

For high-energy positrons, however, the positron bremsstrahlung cross section converges to that of the electron, as the positron-electron annihilation cross section decreases with energy [76] and the positron-nucleus repulsion is more likely overcome. Consequently, any asymmetries in the bremsstrahlung cross section between electrons and positrons vanish as the incoming particle's energy is sufficiently large.

The ratio between the energy loss spectra of electrons and positrons was examined by Kim et al. [90], and is found to be well described as a function of T/Z^2. In particular, this ratio approaches unity as T/Z^2 increases, and the energy losses differ less than 10 percent for

$$T_{e^\pm} > 10^{-2} \cdot Z^2 \text{ MeV}. \tag{6.24}$$

For instance for silicon ($Z = 14$), we can then safely ignore electron-positron asymmetries for $T_{e^\pm} \gg 2$ MeV, which is almost certainly satisfied in high-energy collisions. For the heaviest elements, this limit is around 100 MeV, which is also generally satisfied.

In the remainder, if electrons are mentioned, positrons are implied as well.

[8] The procedure is then to first approximate s using the procedure presented in equations (6.14)–(6.17) and subsequently transform that value using (6.22).

6.1.4 Bremsstrahlung for muons

In principle, the process of bremsstrahlung energy loss works the same for muons (or other charged particles) as for electrons, the only difference being the rate at which energy is lost. Specifically, because the energy loss rate is proportional to the square of the particle's classical radius [see equations (6.1) or (6.23)], which in turn is proportional to the inverse of the particle's mass, we have

$$\frac{\text{energy loss rate muon}}{\text{energy loss rate electron}} = \frac{r_\mu^2}{r_e^2} = \frac{m_e^2}{m_\mu^2} \approx \frac{1}{40{,}000}. \tag{6.25}$$

As a result, bremsstrahlung energy loss is only significant for muons if the muon energies far exceed 400 GeV [91]. Similarly, for other charged particles, which are all heavier than the muon, bremsstrahlung will be even less relevant. Since such high-energy particles are only very rarely produced, it will be sufficient to take only electron bremsstrahlung into account.

6.2 Modelling bremsstrahlung energy loss

The total amount of energy an electron loses as it traverses a material is a random variable, of which the probability distribution is determined by the material properties and the differential cross section (6.23). This probability distribution for the total energy loss due to traversing a material of certain thickness is often called the straggling distribution.

Two approaches can then be taken to model bremsstrahlung energy loss for an ensemble of electrons. On the one hand, a physics-based approach can be taken where the cross section (6.23) is used to explicitly model each individual scattering as an electron traverses the material. On the other hand, if the straggling distribution is known in advance, a statistics-based approach may directly use this distribution to model all scatterings of an electron with the entire material at once, without explicitly using cross sections, but instead by simply making a random draw from this straggling distribution.

That is, to obtain the distribution of energy loss due to traversing some layer of material for an electron of given energy, the former approach explicitly models the underlying physical processes on a microscopic scale (e.g. through a Monte Carlo simulation), whereas the latter approach directly assumes the energy loss to be distributed according to some (parametrized) probability distribution function based on the properties of the material.

Both approaches have certain advantages and disadvantages. The most important drawback is that the straggling distribution cannot be analytically derived from the underlying physics; the best we can do is to assume some distribution that happens to (approximately) reproduce experimental results or the outcomes produced by the physical model. Consequently, the physical model will be more accurate and reliable than a statistical model. However, on the other hand, the physical model has the large drawback that it requires more intensive computations, and is therefore less suitable for simulations where speed is important.

Because for our purposes computational speed is essential, the statistical approach will thus be taken. The remainder of this section therefore focuses on determining a suitable straggling distribution. This is done by first calculating energy loss distributions for some sample materials using the physical approach, and subsequently these distributions are parametrized in terms of the

material properties to obtain a straggling distribution that approximately reproduces the results of the physical model. This way, we obtain a model that is both accurate, like the physics-based microscopic approach, and fast, like the statistical approach.

6.2.1 Physical model for bremsstrahlung energy loss

Using the cross section (6.23) we can simulate the total energy loss distribution of electrons with initial energy E_0 after passing through a material of thickness τ (the thickness in units of radiation length, i.e. $\tau = t/X_0$), atomic mass A, atomic number Z and mass density ρ, according to the following steps.

1. Given the parameters of the material and the electron kinetic energy T, calculate the total cross section $\sigma = \int_0^T \mathrm{d}k(\mathrm{d}\sigma/\mathrm{d}k)$ [i.e. integrate equation (6.23) over k].

2. Assuming that the distance traversed by the electron before emitting a photon is exponentially distributed with rate $N_{\text{atoms}}\sigma$ (= 1/mean free path), draw a path length r from the distribution

$$\mathrm{pdf}(r) = \frac{N_A \rho \sigma}{A} \exp\left[-\frac{N_A \rho}{A}\sigma r\right], \tag{6.26}$$

   and add this to the distance already traversed.[9]

3. If the electron has not yet escaped the material, draw an energy loss from the distribution (6.23), and subtract it from the electron's kinetic energy. Explicitly, draw a fraction of electron energy lost,

$$\mathrm{pdf}(1 - z') \propto \frac{1}{E}\frac{\mathrm{d}\sigma}{\mathrm{d}k}, \tag{6.27}$$

   where z' denotes the fraction of energy remaining after a single collision.

4. Repeat these steps until the electron has escaped the material (i.e. until the distance traversed exceeds $\tau X_0/\rho$).

It should be stressed that the cross section depends on the electron energy, and consequently σ must be recalculated after each bremsstrahlung interaction. Simulating energy loss according to this physics-based method therefore requires repeated numerical integrations (one per path segment), and, as a result, this method is indeed computationally relatively expensive. The implementation of this simulation procedure can be found in Appendix A.6.1.

6.2.2 Statistical model for bremsstrahlung energy loss

The straggling distribution should (implicitly) include information on how the number of collisions an electron has is distributed; how much energy the electron loses each collision; and how these

[9] As we focus on relativistic electrons, we can ignore the increased path length to be traversed due to the scattering of the electron, as the scattering angle will be approximately zero (the scattering angle typically scales as $\theta \sim 1/\gamma$.

energy losses compound. Regarding the latter, the straggling distribution, $\mathrm{pdf}(z;\tau,\dots)$ should have the property that

$$\mathrm{pdf}(\ln z;\tau_1+\tau_2,\dots)=\int_{-\infty}^{\ln z}\mathrm{d}(\ln z')\,\mathrm{pdf}(\ln z';\tau_1,\dots)\,\mathrm{pdf}(\ln z-\ln z',\tau_2;\dots) \tag{6.28}$$

Here, the transformation $\ln z$ is taken because then the fractional energy losses of different materials become additive rather than multiplicative, allowing us to write the compounding property (6.28) in terms of a convolution-like operation.

The main complication for constructing the straggling distribution, is that the differential cross section (6.23) is energy dependent, so that the single-collision energy loss distribution changes after each collision. As a consequence, the two probability distributions in (6.28) need not be of the same functional form.

An approximate straggling distribution based on a simplification of the differential cross section (6.1), and satisfying condition (6.28), is given by the Bethe-Heitler straggling distribution [92][10]

$$\mathrm{pdf}(z)=\frac{(-\ln z)^{\tau/\ln 2-1}}{\Gamma(\tau/\ln 2)}, \tag{6.29}$$

which depends directly on the material properties and thickness through $\tau = t/X_0$; however, it does not depend on the electron energy, which is inconsistent with the energy dependence of the cross section (6.23), indicating that this approximate straggling distribution indeed excludes some essential features.

Another property of the Bethe-Heitler straggling distribution (6.29) is that it diverges at one of the endpoints: for $\tau < \ln 2$, we find that $\mathrm{pdf}(z) \to \infty$ as $z \to 1^-$; and for $\tau > \ln 2$, we have $\mathrm{pdf}(z) \to \infty$ as $z \to 0^+$ ($\mathrm{pdf}(z) \to 0$ at the other endpoints). Now, it is exactly the former divergence (at $z = 1$) that should be suppressed by the LPM and Ter-Mikaelian effects, which is another essential feature excluded by (6.29).

Finally, the process described in Section 6.2.1 predicts a non-zero probability that the electron does not scatter on the material at all, and thus loses no energy. The distribution (6.29), on the other hand, does not predict a non-zero (i.e. not infinitely small) probability that z is exactly 1.

The implementation of the statistics-based model is shown in Appendix A.6.2.

6.2.3 Comparison of the physics-based and statistics-based models

A comparison between the physics-based [i.e. based on the differential cross section (6.23)] and the statistics-based [i.e. based on the straggling distribution (6.29)] models is shown in Figure 6.2, which shows simulations of the probability distributions of the fraction of electron energy remaining (z) after traversing a layer of silicon of various thicknesses τ (ranging from 0.05 to 2.50), and for different incident electron energies (ranging from 100 to 2000 GeV). These ranges should be sufficient to cover all circumstances occurring in high-energy experiments: detector material budgets are kept to a minimum to minimize signal distortions, and electron energies generally do not

[10] Computationally, noting that z follows a gamma distribution, $\mathrm{pdf}(z) = \mathrm{Gamma}(-\ln z; t/\ln 2, 1)$, one can draw from this distribution by drawing x from a gamma distribution $\mathrm{Gamma}(x; t/\ln 2, 1)$ and making the transformation $z = \exp(-x)$ [93]. A computational algorithm for drawing from a gamma distribution can be found in e.g. Ref. [94].

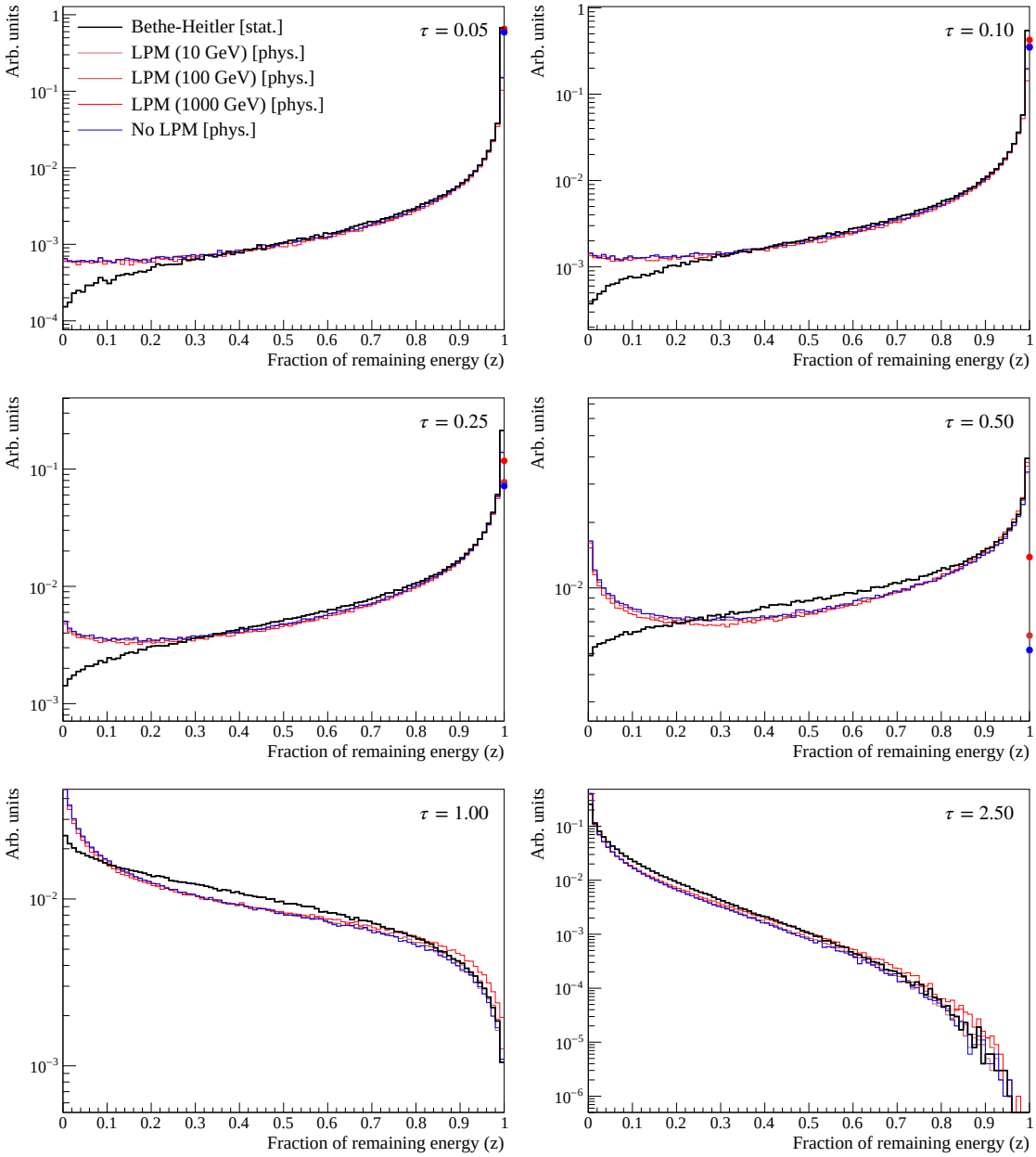

Figure 6.2
Comparison of statistics-based versus physics-based bremsstrahlung simulation models. Each panel shows the probability distribution function of the fraction of energy remaining after traversing a silicon material of given thickness, for (black line) the Bethe-Heitler model (6.29); (red lines) the physics model from Section 6.2.1 including the LPM and Ter-Mikaelian effects, for different electron energies; and (blue line) the same physics model without the LPM effect (but including the Ter-Mikaelian effect). The dots on the right axis show the probability that an electron does not interact with the material at all.

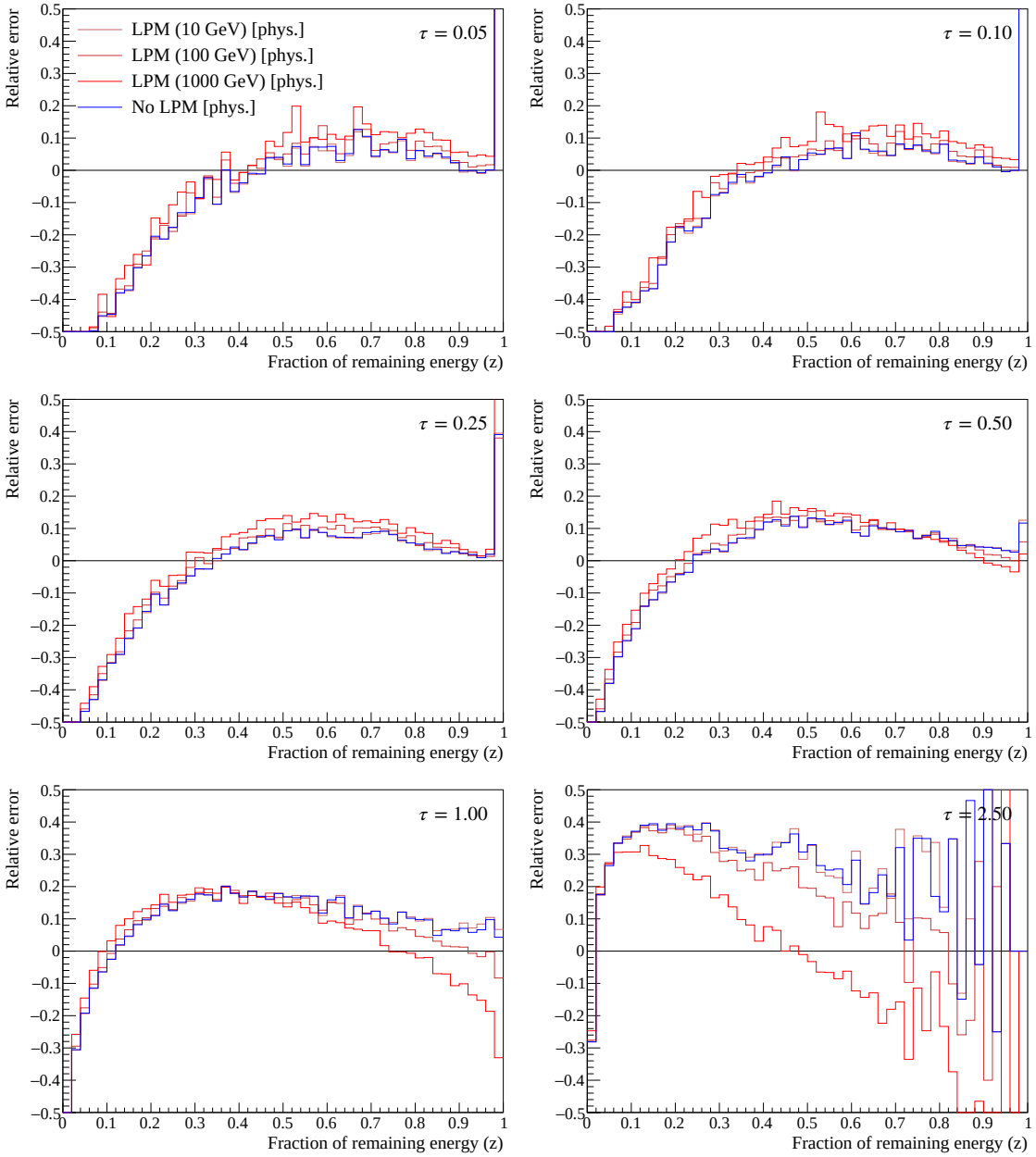

Figure 6.3
Comparison of statistics-based versus physics-based bremsstrahlung simulation models. Each panel shows the relative error of the Bethe-Heitler straggling distribution for the fraction of energy remaining after traversing a silicon material of given thickness with respect to (red lines) the physics model from Section 6.2.1 including the LPM and Ter-Mikaelian effects, for different electron energies; and (blue line) the same physics model without the LPM effect (but including the Ter-Mikaelian effect).

exceed a few hundred GeV. In addition, Figure 6.3 shows the relative error of the straggling distribution (6.29) compared to the physics-based distribution for different incident electron energies.

From Figures 6.2 and 6.3, it can be seen that the physics-based microscopic model and the statistics-based macroscopic model are only in moderate agreement (errors are of the order of 10–20 percent for most z). Especially for low values of z, i.e. where the electrons lose most of their energy, the microscopic model deviates considerably. However, it should be noted that in the context of mass spectra, these events give yield to relatively low reconstructed invariant masses, such that these events are in practice often excluded from analysis, as only decays within a certain invariant mass window are included. For that reason, a relatively large error for low z is not problematic.[11]

There are also relatively large deviations around $z \approx 1$, where the straggling distribution (6.29) predicts probabilities that are too high. To a large extent, this is due to the divergence in the distribution. Since these events end up around the peak of the mass spectrum, deviations in this region do affect the analysis and should therefore ideally be corrected for.

The probabilities that there are no collisions (and thus $z = 1$) for the different models are marked with dots in Figure 6.2. Indeed, especially for low-thickness materials, these probabilities can be significant. Moreover, it is observed that the probability of zero collisions increases in incident electron energy, which follows from the fact that bremsstrahlung emission is more strongly suppressed for higher-energy electrons. Again, the Bethe-Heitler straggling model does not include this feature, and since the high-z region affects the relevant part of the mass spectrum, this should be taken into account.

Moreover, the body of the distribution (i.e. the region $z < 1$) is indeed energy dependent. This effect is especially clear for higher thicknesses, although it is also significant for thinner materials.

6.2.4 A phenomenologically corrected Bethe-Heitler straggling model

To obtain a model that is both accurate and computationally inexpensive, we would ideally parametrize the straggling distributions obtained from the physics model from Section 6.2.1 in terms of the material thickness τ (which implicitly includes the material properties through $\tau = t/X_0$) and incident electron energy E.

To find such a parametrization, we use the Bethe-Heitler straggling distribution (6.29) as a starting point, because that distribution is already in relatively good agreement with the physics-based simulated distributions from Figure 6.2. Then, we will apply several corrections to compensate for the three major differences between these distributions that we have found in the previous sections: (i) the non-zero probability that $z = 1$; (ii) the deviations of the Bethe-Heitler distribution from the

[11] In the case that a decay of some heavier parent particle is misidentified as a signal decay, it is possible that low-z events of that misidentified decay do affect the observed spectrum. For example, consider that we are looking for $J/\psi \to ee$ events. Now, it may occur that a $\psi(2S) \to ee$ (or even $\Upsilon \to ee$) is incorrectly identified as $J/\psi \to ee$. However, the reconstructed mass of this misidentified decay will only be in the range of the J/ψ mass if the electrons have lost a sufficient amount of energy. That, in turn, means that the misidentified event is located far in the left tail of the $\psi(2S) \to ee$ spectrum, which generally is approximately flat. But if the spectrum of the misidentified decay is flat, in terms of signal shapes, these events will just add to the combinatorial background, which is also approximately flat. As a result, we need not exactly know the low-z part of the straggling distribution for potential misidentified decays either. Instead, if low-z misidentified decays would potentially contaminate our signal, it would also be safe to treat it as combinatorial background.

physics-based distributions for $z < 1$; and (iii) the non-physical divergence of the Bethe-Heitler distribution. The resulting parametrization is of the form

$$\mathrm{pdf}(z;\tau,E) = \begin{cases} P_0(\tau,E)\,\delta(z-1) & \text{for } z = 1 \\ \varepsilon(z;\tau,E)\,\frac{(-\ln z)^{\tau/\ln 2-1}}{\Gamma(\tau/\ln 2)} & \text{for } 0 < z < \lambda(\tau,E) \end{cases} \tag{6.30}$$

where $P_0(\tau, E)$ is the probability that there is no bremsstrahlung emission, $\varepsilon(z;\tau, E)$ is an empirical correction factor for the region ($z \neq 1$), and $\lambda(\tau, E) < 1$ is a divergence cut-off. Note that each of these correction depends both on thickness and electron energy.

In the following, we will briefly discuss each correction. It should be emphasized that these corrections are not derived from first principles, or are otherwise derived from physical grounds. Instead, they are purely phenomenological. However, this is in principle not an issue, because the purpose is solely to find a sufficiently accurate parametrization for the straggling distributions in Figure 6.2, and not to derive insights into the physics that generate these distributions.

Therefore, the general approach is to consider different functional forms for the corrections, and fit these parametrizations to the physics-based distributions in Figure 6.2. This ensures that we obtain a model that is sufficiently accurate at least up to $\tau = 2.5$ and $E = 2000$ GeV. The parametrization that yields the best fit is then used in the final model. Note that only the final model is reported here, as reporting alternative models would offer little to no additional insight.

Probability of no energy loss

As long as an electron does not collide and lose energy, it faces a constant cross section as it traverses the material, and thus a constant collision rate (i.e. expected number of collisions per path length). As such, the probability that the electron scatters zero times is given by the Poisson probability

$$P(0\text{ collisions}) = e^{-a\tau}\frac{(a\tau)^0}{0!}, \tag{6.31}$$

where a is the collision rate per unit thickness. Given this assumption, and the fact that the probability of zero collisions should go to unity as the thickness goes to zero, we arrive at an empirical parametrization for the probability of no energy loss of the form

$$P_0(\tau,E) = \exp\left[\left(\tau + p_{00}\tau^2\right)\sum_{n=0}^{3} p_{1n}E^n\right]. \tag{6.32}$$

The fitted coefficients are shown in Table 6.1. As expected, the coefficients are such that $P_0(\tau, E)$ decreases in τ (more material means more collisions), and increases in E (higher energies imply stronger suppression effects).

Empirical correction factor

As correction factor for the body of the straggling distribution, we assume a parametrization of the form

$$\tilde{\varepsilon}(z;\tau,E) = \frac{1 + \sum_{n=1}^{3}\sum_{m=0}^{2} c_{nm} z^n \tau^m \left(1 + k_n\sqrt{E}\right)}{1 + \sum_{n=0}^{3}\sum_{m=0}^{2} c'_{nm} z^n \tau^m \left(1 + k'_n\sqrt{E}\right)}. \tag{6.33}$$

Empirical correction factor $\tilde{\varepsilon}(z;\tau,E)$						**Prob. of no energy loss** $P_0(\tau,E)$		
			c'_{00}	-0.5668		p_{00}	0.0062	
			c'_{01}	-0.0673		p_{10}	-10.5863	
			c'_{02}	0.0768		p_{11}	0.0040	GeV^{-1}
c_{10}	-0.6013		c'_{10}	-0.0984		p_{12}	-2.58×10^{-6}	GeV^{-2}
c_{11}	24.4468		c'_{11}	41.7926		p_{13}	6.40×10^{-10}	GeV^{-3}
c_{12}	-7.5412		c'_{12}	-10.6599				
c_{20}	-1.2408		c'_{20}	-0.8901		**Divergence cut-off** $\lambda(\tau,E)$		
c_{21}	11.2490		c'_{21}	16.6239		q_0	0.0005	
c_{22}	-12.7859		c'_{22}	-28.2874		q_1	-2.0022	
c_{30}	0.0038		c'_{30}	-0.8260		q_2	0.0184	$\mathrm{GeV}^{-1/2}$
c_{31}	0.4423		c'_{31}	-8.5567				
c_{32}	-0.3431		c'_{32}	13.7098				
			k'_0	-0.0011				
k_1	0.0093	$\mathrm{GeV}^{-1/2}$	k'_1	0.0054	$\mathrm{GeV}^{-1/2}$			
k_2	-0.2590	$\mathrm{GeV}^{-1/2}$	k'_2	-0.1650	$\mathrm{GeV}^{-1/2}$			
k_3	3.9742	$\mathrm{GeV}^{-1/2}$	k'_3	-0.2326	$\mathrm{GeV}^{-1/2}$			

Table 6.1
Coefficients for the empirically corrected Bethe-Heitler model, corresponding to the corrections (6.32), (6.33) and (6.34).

Both the numerator and the denominator consist of a third-order polynomial in the thickness, and depend on the square root of the electron energy. The polynomial expansion is chosen because this corresponds to a general Taylor expansion in τ, of which we keep only the first few leading orders. The square root dependence for the energy is taken because the energy dependence of the errors in Figure 6.3 are less than linear.

Note that $\tilde{\varepsilon}(z;\tau,E)$ is not normalized. Therefore we define $\varepsilon(z;\tau,E)$ to be proportional to $\tilde{\varepsilon}(z;\tau,E)$, where the factor of proportionality is such that it normalizes the corrected distribution (6.30).

The resulting coefficients for the (non-normalized) correction factor are given in Table 6.1. Because low-z events fall outside of the analysis window, and to exclude the high-z divergence, the coefficients are obtained by fitting the model to the physics-based model only on the range $0.050 < z < 0.985$. The resulting correction factors are visualized in Figure 6.4, from which we can explicitly see the thickness and energy dependences.

Divergence cut-off

The divergence cut-off should be close to but smaller than unity. In addition, it should become closer to unity for larger τ (for $\tau > \ln 2$ the divergence in the Bethe-Heitler straggling distribution disappears, although we do not include this fact), and it should move away from unity for larger energies (for higher energies the suppression effects become stronger). A form that satisfies these

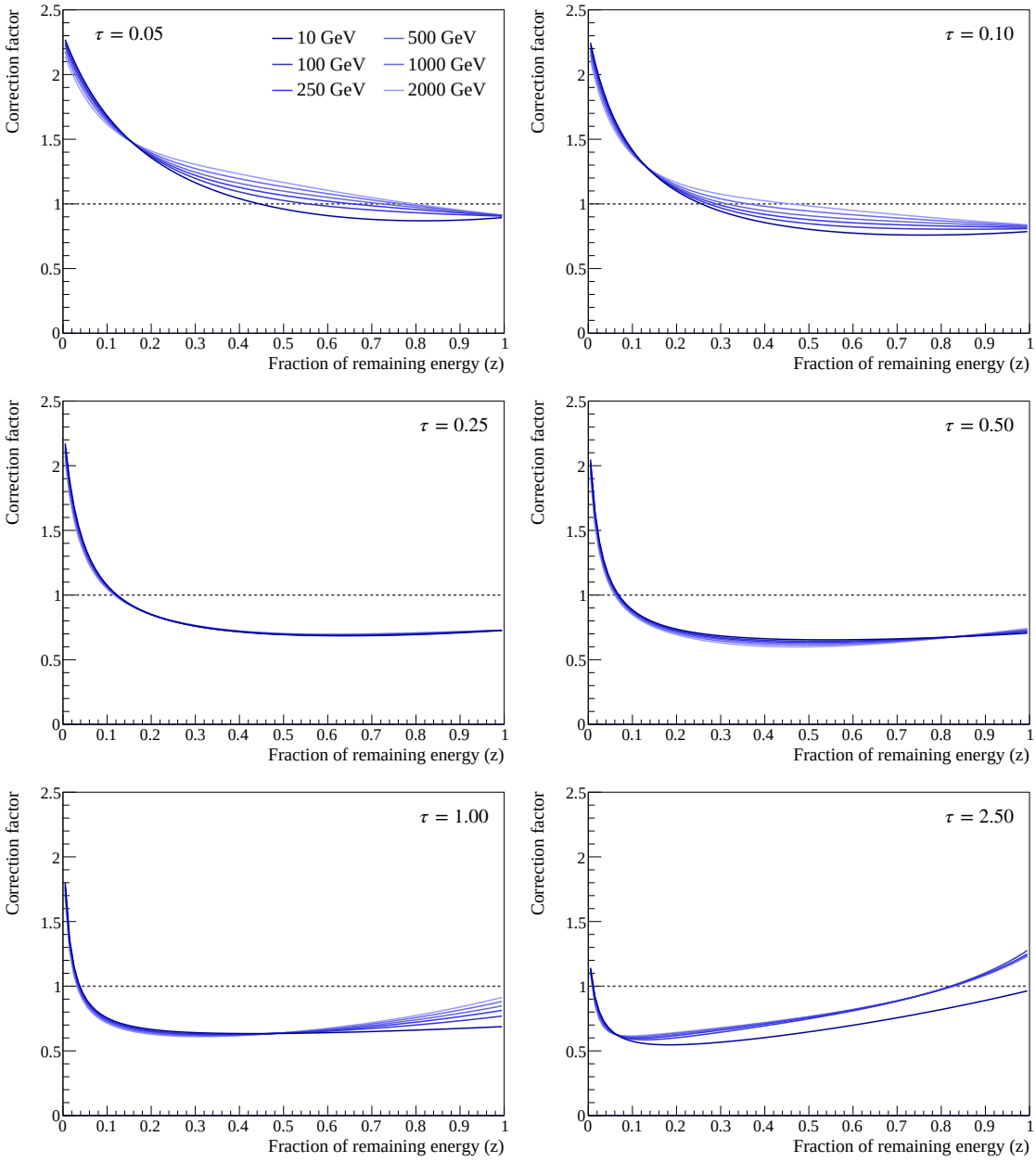

Figure 6.4
Empirical correction factors $\varepsilon(z;\tau,E)$.

requirements is

$$\lambda(\tau, E) = 1 - q_0 \exp\left[q_1\tau + q_2\sqrt{E}\right], \tag{6.34}$$

with the coefficients again given in Table 6.1.

The resulting corrected straggling distribution

The resulting straggling distribution of the form (6.30) are compared with the physics-based model in Figures 6.5–6.10. The agreement with the physics-based simulation is good for all values of z, except at the low-z endpoint, where deviations can be safely ignored. The fitted model is also inaccurate at the high-z endpoint for high-τ, high-E configurations; however, because the probability distribution has a low value at this point, this small error will not significantly affect the produced mass spectrum, and can therefore also be neglected. Indeed, for all values of τ and E shown, the empirically corrected Bethe-Heitler model performs better than the original Bethe-Heitler straggling distribution.

The implementation of the corrected Bethe-Heitler straggling model can be found in Appendix A.6.3.

In Figure 6.11 mass spectra generated with the Bethe-Heitler straggling distribution with and without corrections are compared. The error of the original model with respect to the corrected model are shown in Figure 6.12. These figures show that the mass spectra differ only marginally for low-thickness materials, but that the effect of the empirical corrections do become considerable for thicker materials. More specifically, the error of the original model is of the order of 10 percent as $\tau > 0.25$. The error is especially large in the low-energy tail, but is also substantial in the core of the spectrum. Given that the combined thickness of the material in the LHCb detector (before the magnet) is also of the order $\tau \sim 0.3$ [95], the corrections constructed in this section then indeed improve the quality of our simulated mass spectrum shapes significantly.

6.3 LHCb material budget

In order to correctly simulate bremsstrahlung for the LHCb detector, it must be known how much material electrons generally have to traverse. In particular, it must be known how much material is traversed before entering the detector's magnetic field, because only bremsstrahlung emitted before the magnet may go missing.

To understand this, refer to Figure 3.5. Because bremsstrahlung photon emission is always in the flight direction of the electron for high-energy electrons (the maximum angle of emission is proportional to $1/\gamma$), if bremsstrahlung is emitted after the magnetic field, the photon and the electron will end up in the same ECAL cluster (E_2 in Figure 3.5). As a result, the electron is assigned an energy equal to the final electron energy plus the photon energy, i.e. it is assigned exactly the energy it had before emitting bremsstrahlung radiation.

On the other hand, if the bremsstrahlung is emitted before the magnet, the magnet will subsequently curve the trajectory of the electron, but not that of the photon. Consequently, the electron and the emitted photon will not end up in the same ECAL cluster, and the electron is assigned only the final electron energy; i.e. the emitted photon energy is lost.

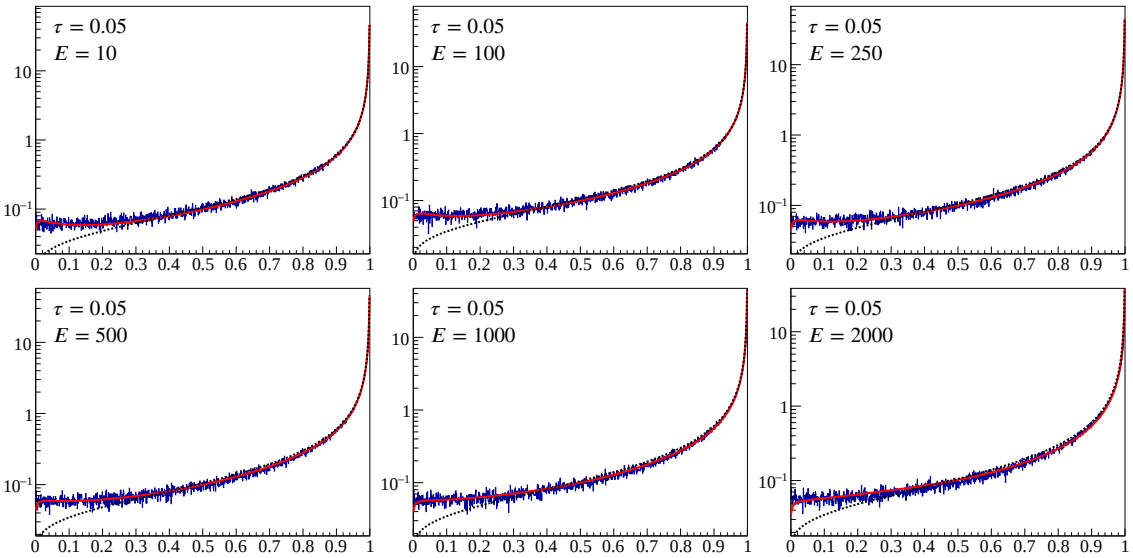

Figure 6.5
Comparison of the physics-based bremsstrahlung model from Section 6.2.1 (blue histogram) with the statistics-based Bethe-Heitler model (black dashed line) and the corrected Bethe-Heitler model (solid red line), for different electron energies E (in GeV). The material is silicon of thickness $\tau = 0.05$. The x-axis shows the fraction of energy remaining; the y-axis shows the value of the probability distribution function.

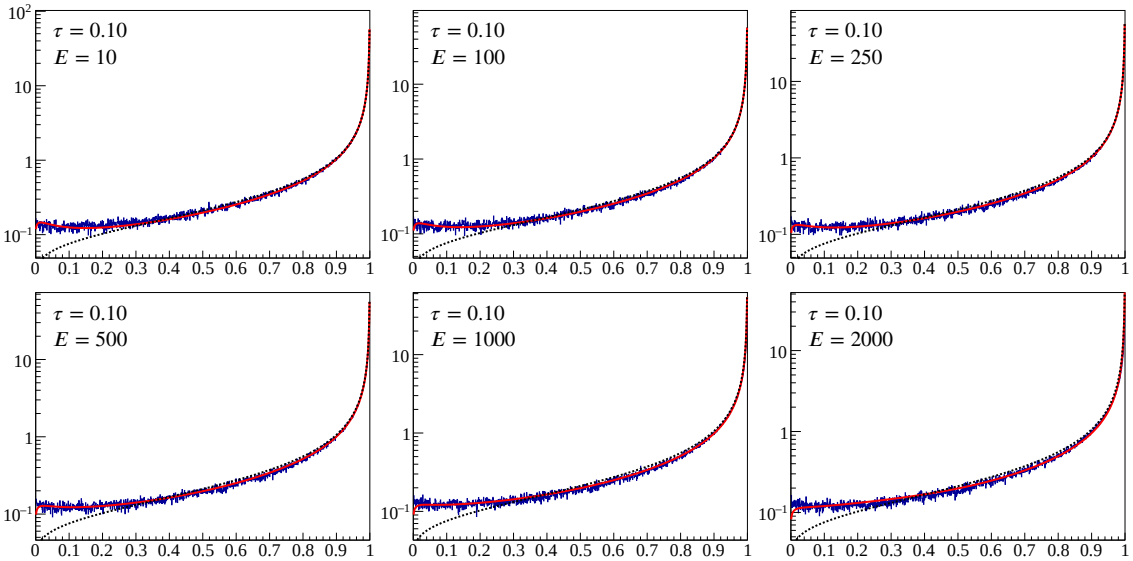

Figure 6.6
Same figure as Figure 6.5, but for $\tau = 0.10$.

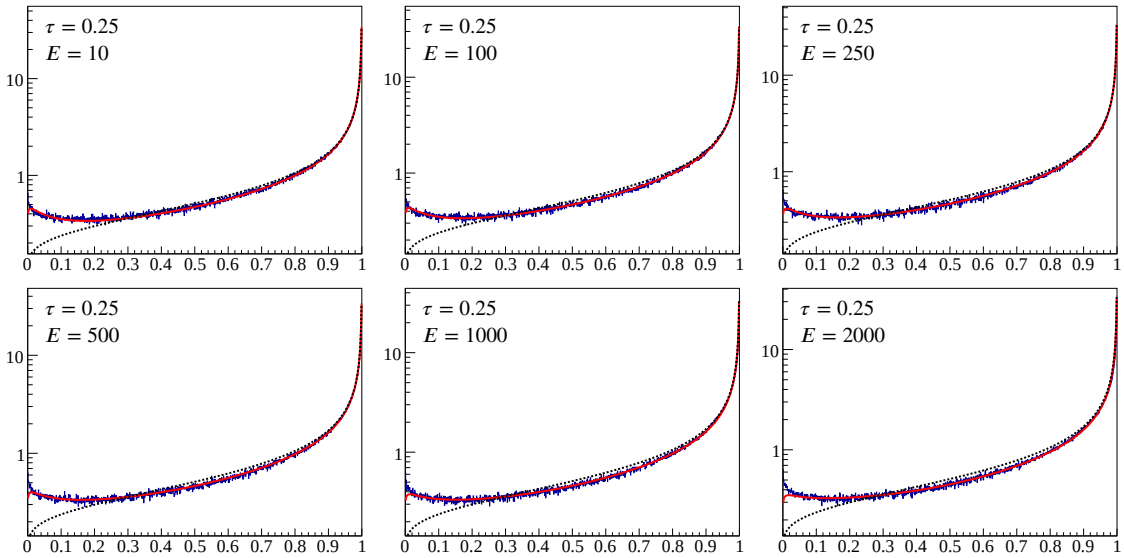

Figure 6.7
Same figure as Figure 6.5, but for $\tau = 0.25$.

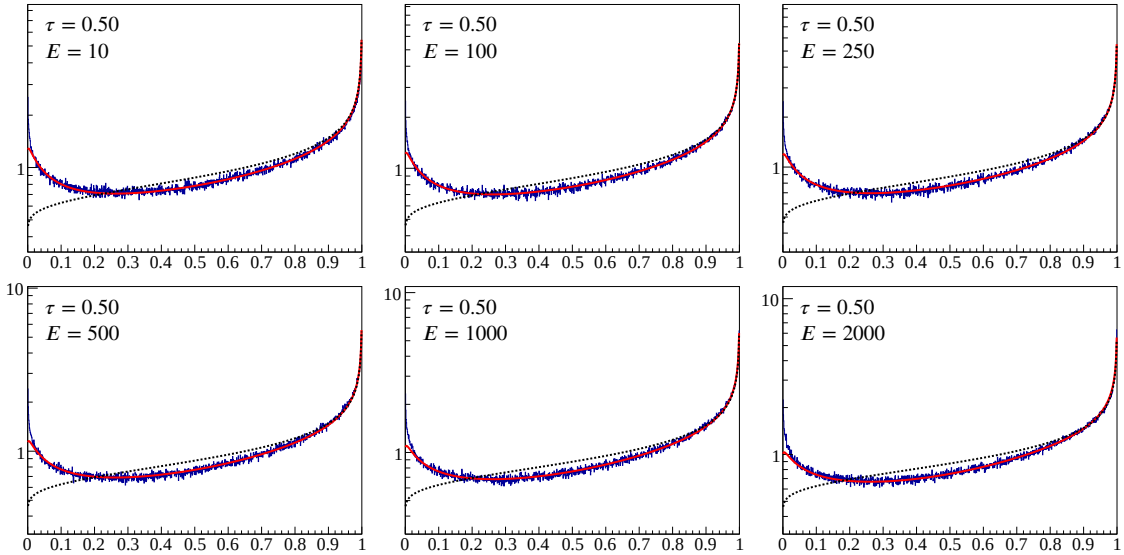

Figure 6.8
Same figure as Figure 6.5, but for $\tau = 0.50$.

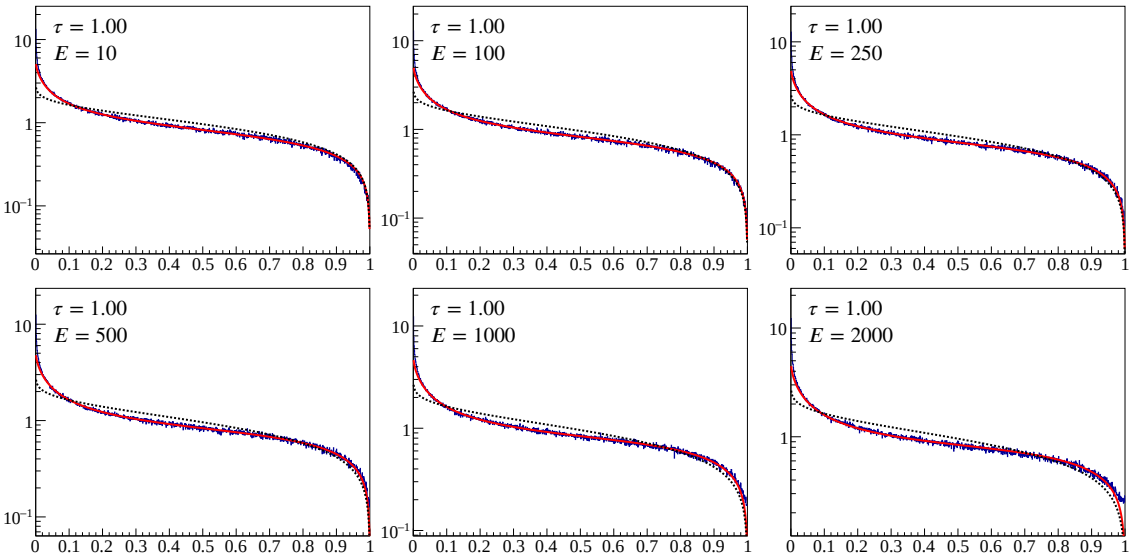

Figure 6.9
Same figure as Figure 6.5, but for $\tau = 1.00$.

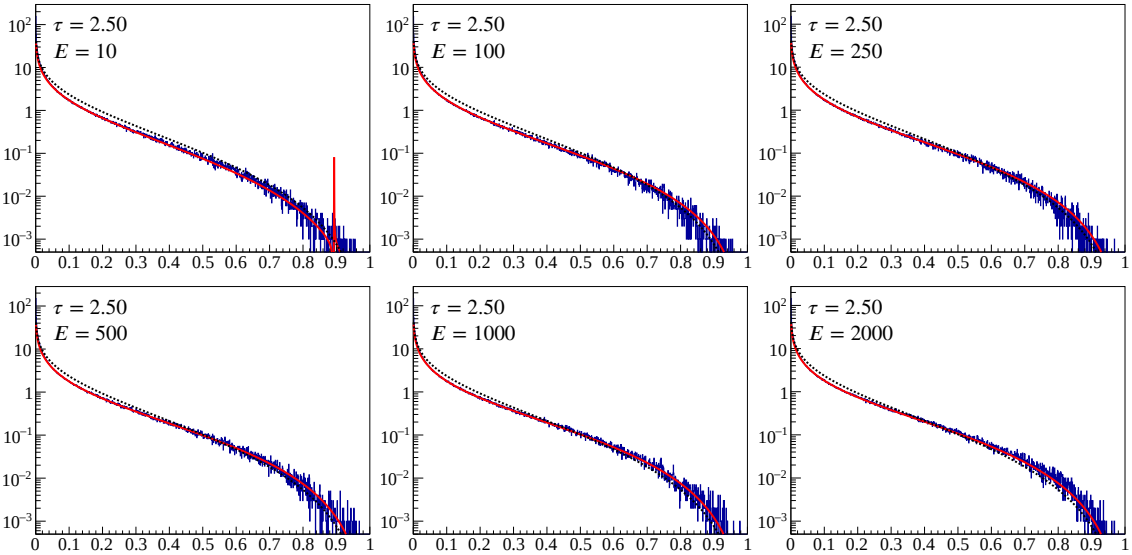

Figure 6.10
Same figure as Figure 6.5, but for $\tau = 2.50$.

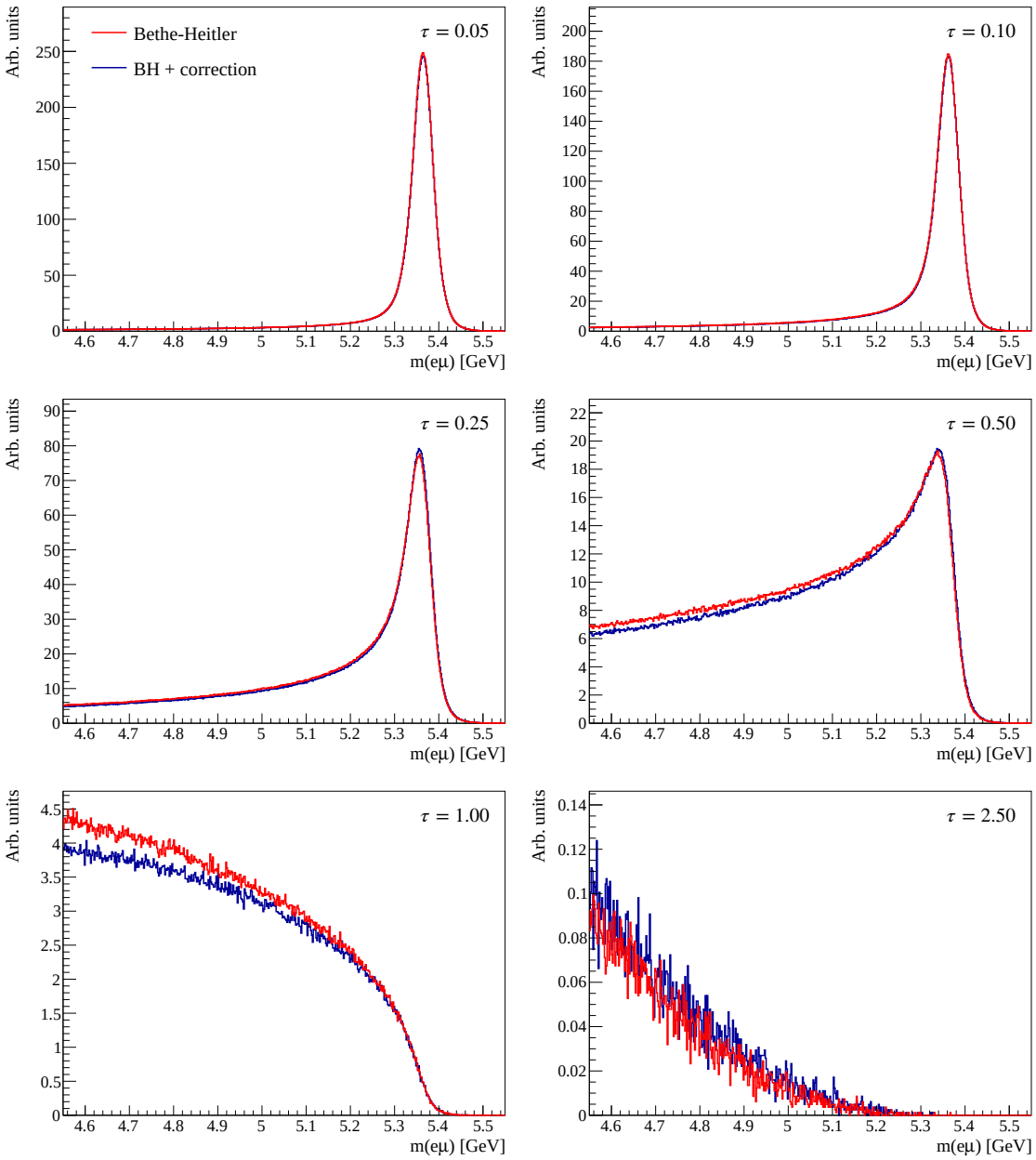

Figure 6.11
Mass spectra for $B_s^0 \to e\mu$ where the electron emits bremsstrahlung radiation according to the Bethe-Heitler model with (blue line) and without empirical corrections (red line). The distribution of B_s^0 energies is taken from Monte Carlo simulations by LHCb.

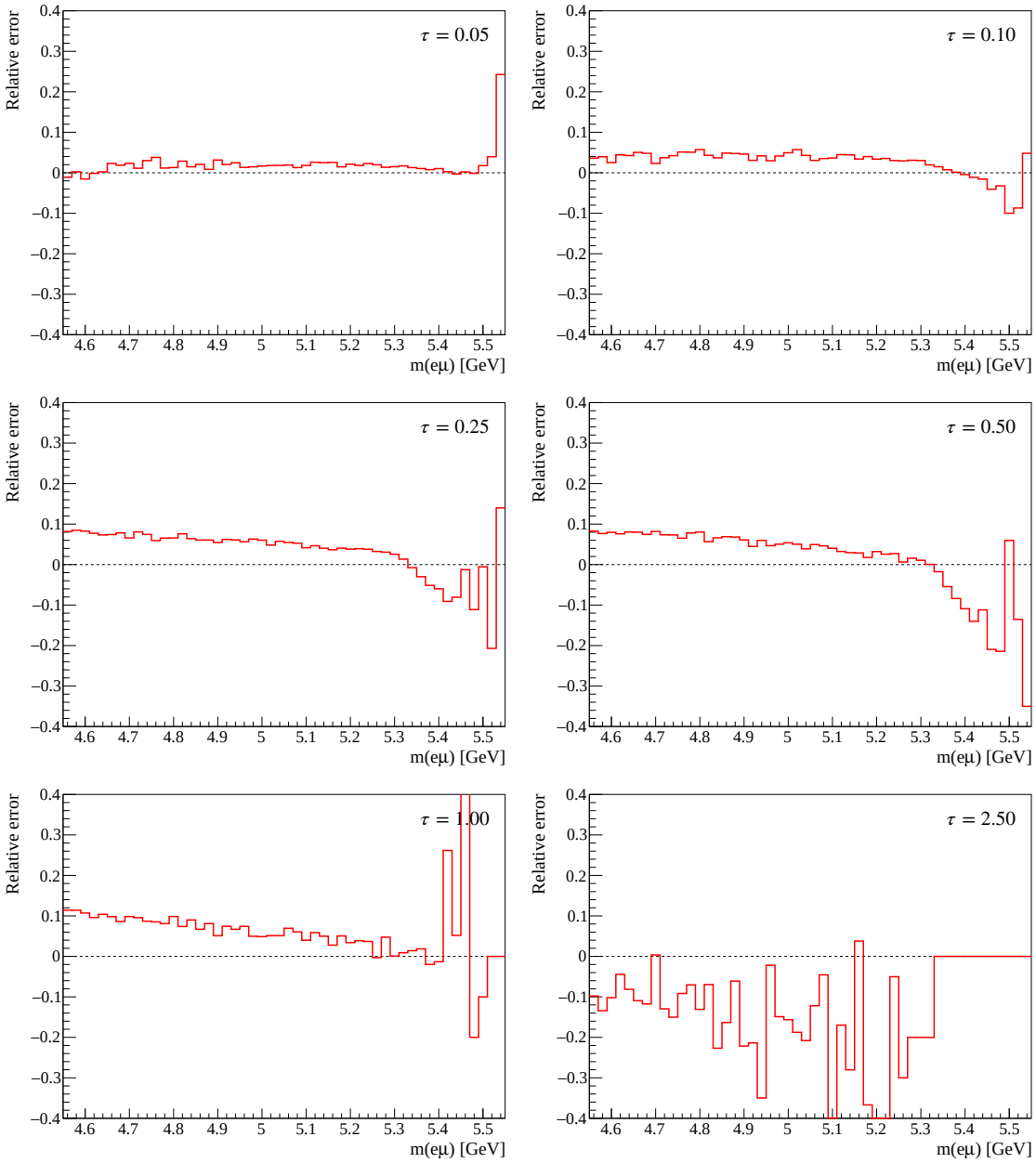

Figure 6.12
Relative errors of the mass spectra for $B_s^0 \to e\mu$ generated according to the Bethe-Heitler without corrections with respect to the mass spectra generated according to the empirically corrected model, as shown in Figure 6.11. For reference, $M(B_s^0) = 5.367$ GeV [36].

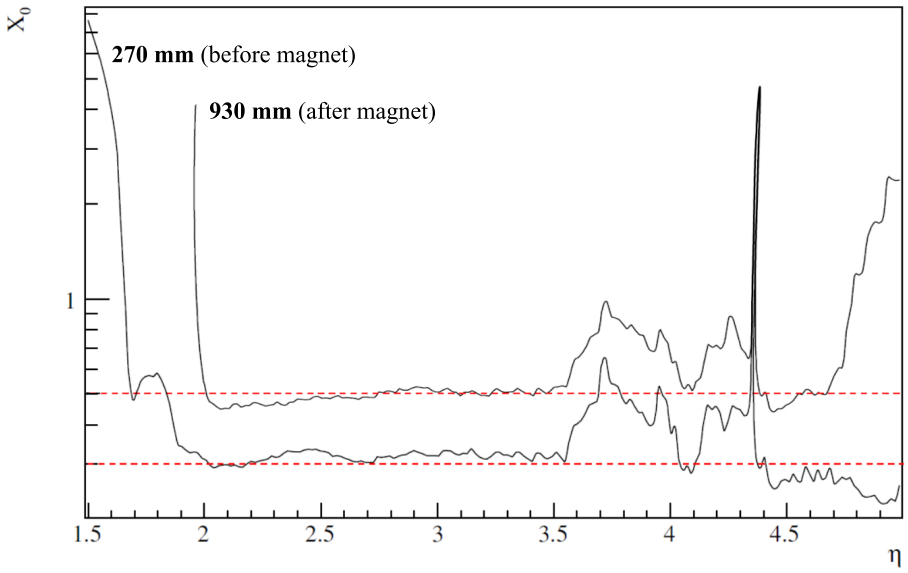

Figure 6.13
Radiation lengths by pseudorapidity for the LHCb detector, integrated along the beam direction up to $z = 270$ mm (i.e. from the primary vertex until the magnet), and up to $z = 930$ mm (i.e. until after the magnetic field). The dashed lines indicate $X = 0.3X_0$ and $X = 0.5X_0$ (i.e. respectively $\tau = 0.3$ and $\tau = 0.5$). Adapted from Ref. [96].

The distribution of material within the LHCb detector is shown in Figure 6.13, where the material budget up to the magnet is shown separately. We can see that for a large pseudorapidity range ($2.0 < \eta < 3.6$, or $55 < |\theta| < 270$ mrad), the pre-magnet radiation length is approximately constant at $\tau = 0.3$. For larger pseudorapidities (smaller polar angles θ), there is more variation in the integrated radiation length, and the radiation length becomes higher. At the highest pseudorapidities ($\eta > 4.4$ or $|\theta| < 25$ mrad), taking $\tau = 0.3$ is again a good approximation.

The observation that the integrated radiation length is approximately constant for most values of the pseudorapidity, suggests that it would be a sensible simplification to assume the radiation length to be pseudorapidity-independent. Therefore, our model will assume $\tau = 0.3$, unless stated otherwise.

6.4 Test case: bremsstrahlung in $B^0_{(s)} \to e\mu$

Figure 6.14 shows the performance of our bremsstrahlung model with respect to Monte Carlo simulations by LHCb (i.e. using the extensive simulation software discussed in Chapter 3). It can be seen that, although the cores of the distributions match relatively well, there is still room for improvement with respect to the tails.

Note that two different samples are shown in Figure 6.14: one sample of events where no bremsstrahlung photons are reconstructed (right); and one full sample (left). The difference between these spectra illustrates the effect of the bremsstrahlung recovery tool on the spectrum. In particular, by recombining the electrons that have lost a significant amount of energy through

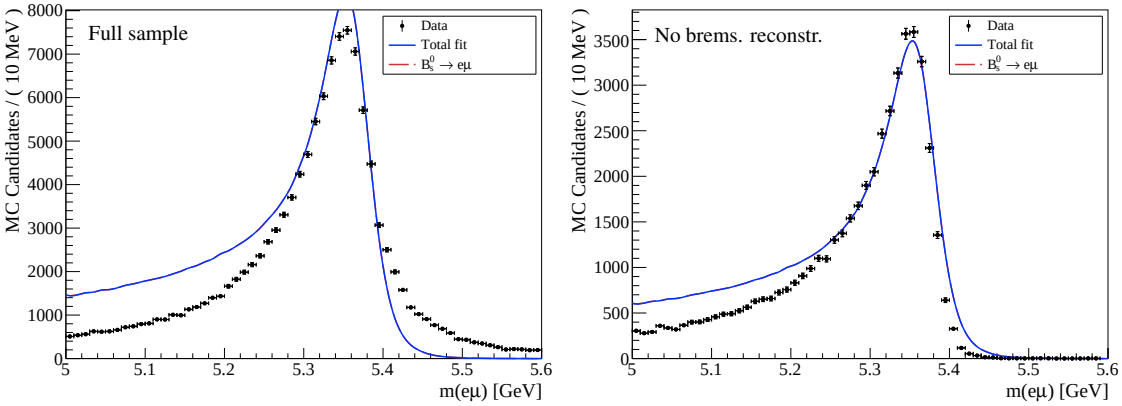

Figure 6.14
Comparison of the simulated for $B^0_s \to e\mu$ spectrum with the LHCb (Monte Carlo) data. The left figure includes all events; the right figure includes only events where no bremsstrahlung has been added to the electron. In line with the LHCb material budget, we have assumed a relative material thickness $\tau = 0.3$.

bremsstrahlung—which end up in the left tail of the spectrum—with this lost energy, it moves events in the left tail back towards the core of the spectrum.

However, there may also be events in which a background photon is reconstructed as a bremsstrahlung photon. In this case, an electron is recombined with energy that it has not lost. This may happen relatively often, and consequently a tail on the right-hand side of the core may arise, consisting of events that have been overcompensated for bremsstrahlung loss. Indeed, while this right-hand tail can be clearly seen in the sample with bremsstrahlung reconstruction, it is absent in the sample where such recombination has not been done.

It is then evident that the bremsstrahlung recovery tool has a relatively strong effect on the final mass spectrum. In fact, the difference between the left and right panels of Figure 6.14 is considerably larger than the effect of correcting the Bethe-Heitler model (Figure 6.11); as such, to improve the accuracy of our simulation model, it is important that it includes this feature as well. For that reason, further development of our model should particularly focus on understanding bremsstrahlung photon recovery.

Moreover, the imperfect match between the spectra in the right panel of Figure 6.14 suggests that additional improvements could be made as well. In particular, we see that the core of the simulated spectrum (i.e. from the simulation presented in this thesis, as opposed to the LHCb simulation) is slightly too wide. Why this is the case, is at present unclear, but this can possibly be corrected for by calibrating the model to this specific decay: the model is now calibrated to the channel $J/\psi \to \mu\mu$—see Chapter 8—but it is plausible that the momentum resolution parameters differ considerably between electrons and muons, in which case we would indeed expect the width of the core to be different than what we have simulated.

In addition, we see that the left tails, which contain the electrons that have emitted bremsstrahlung, do not agree either. Specifically, our simulation model predicts more events in this left tail than the LHCb simulation. Again, it is presently not understood what causes this. For example, it could possibly be caused by certain cuts that were not included in the model (e.g. cuts on

total or transverse momentum) or if the efficiency for electron detection is (transverse) momentum dependent. Clearly, our model would benefit from further research into this issue.

Chapter 7

Background shapes

As we have seen in Chapter 3, correctly registering and reconstructing events produced in high-energy collisions is not straightforward. Consequently, the process of event reconstruction is considerably prone to errors, and such errors will distort the observed mass spectrum in a systematic way. Ideally, in order to optimize the quality of our signal, we would like to be able to subtract such backgrounds from the signal, so that possible errors during event reconstruction are corrected for. That is, if the shapes of these potential backgrounds are known, it becomes possible to identify these backgrounds in mass spectra, and include them in the signal fit, so that they can be corrected for.

To that end, it is useful to study these background shapes in some detail. Generally, three categories of backgrounds are distinguished: peaking backgrounds, which arise if some decay is erroneously identified as a signal decay because one or more particles are misidentified (Section 7.1); semi-peaking backgrounds, which arise if some decay is erroneously identified as a signal decay because one or more particles that are produced in that decay are not observed (Section 7.2); and combinatorial backgrounds, which arise if particles produced in independent decays coincidentally combine such that they fake the signature of a signal decay (Section 7.3).

7.1 Peaking backgrounds due to misidentification

If an experiment is studying a certain decay, one criterion for identifying this specific decay is that a candidate decay produces particles of the expected identities. But then, if some similar decay takes place, and its secondary particles are misidentified as the particles that the experiment is searching for, this similar decay will be incorrectly classified as a signal decay.

For example, suppose one is looking for

$$B^{\pm} \to J/\psi K^{\pm} \tag{7.1}$$

decays. Then, we would roughly consider an event to contain such a decay if both a J/ψ (or rather its decay products that are reconstructed as a J/ψ) and a $K^{\pm}$ are observed; if these particles originate from the same vertex; if no other particles originate from the same vertex; and if the invariant mass of these particles is approximately equal to the $B^{\pm}$ mass. However, all but the first requirement are

also met by

$$B^{\pm} \to J/\psi \pi^{\pm} \tag{7.2}$$

decays. But given that particle identification is not particularly straightforward, identification errors occur relatively frequently. Consequently, it may occur that the $\pi^{\pm}$ produced in a $B^{\pm} \to J/\psi \pi^{\pm}$ is misidentified as a $K^{\pm}$, so that the decay will be incorrectly treated as a $B^{\pm} \to J/\psi K^{\pm}$. As a result, particles are assigned incorrect masses in the reconstruction, which affects the observed mass spectrum.

The goal of this section is to derive how particle misidentifications alter the observed mass spectrum. To that end, we will start by stating the problem in terms of infinitesimal misidentifications in Section 7.1.1. Subsequently, we will use the result of that section to derive the effects on the mass spectrum of a single misidentification in Section 7.1.2, after which Section 7.1.3 will treat the general case of N misidentifications.

7.1.1 Infinitesimal form

Using that the reconstructed particle masses is calculated as

$$M = \sqrt{\left(\sum_i E_i\right)^2 - \left|\sum_i \mathbf{p}_i\right|^2}, \quad \text{with } E_i = \sqrt{p_i^2 + m_i^2}, \tag{7.3}$$

an infinitesimal change in the observed secondary particle masses m_i^2, i.e. $\delta(m_i^2)$, gives a change in observed primary mass

$$\begin{aligned}
\delta(M^2) &= \delta\left[\left(\sum_i E_i\right)^2 - \left|\sum_i \mathbf{p}_i\right|^2\right] \\
&= \delta\left[\sum_i \sum_j E_i E_j\right] \\
&= \sum_i \sum_j \left(E_i \delta E_j + E_j \delta E_i\right) \\
&= 2 \sum_i \sum_j E_i \delta E_j,
\end{aligned} \tag{7.4}$$

where we have used that $\delta \mathbf{p}_i = 0$ (the momentum is measured independent of mass). Writing out E_i, and using

$$\begin{aligned}
\delta E_j &= \delta \sqrt{m_j^2 + p_j^2} \\
&= \frac{\delta(m_j^2)}{2\sqrt{m_j^2 + p_j^2}}
\end{aligned} \tag{7.5}$$

we have

$$\begin{aligned}\delta(M^2) &= \sum_i \sum_j \frac{\sqrt{m_i^2 + p_i^2}}{\sqrt{m_j^2 + p_j^2}}\, \delta(m_j^2) \\ &= \sum_i \sum_j \frac{E_i}{E_j}\, \delta(m_j^2). \end{aligned} \tag{7.6}$$

Note that thus far, we have not made any approximations to obtain equation (7.6). In the next sections, we will determine finite differences due to misidentifications by integrating this equation. The final result is given by (7.13) below.

7.1.2 Single misidentification

For a single misidentification of particle j, we can calculate the finite change $\Delta(M^2)$ by integrating:

$$\begin{aligned}\Delta(M^2) &= \int_{m_j^2}^{m_j^2+\Delta(m_j^2)} d(m_j^2) \sum_i \frac{\sqrt{m_i^2 + p_i^2}}{\sqrt{m_j^2 + p_j^2}} \\ &= \Delta(m_j^2) + \int_{m_j^2}^{m_j^2+\Delta(m_j^2)} d(m_j^2) \sum_{i\neq j} \frac{\sqrt{m_i^2 + p_i^2}}{\sqrt{m_j^2 + p_j^2}} \\ &= \Delta(m_j^2) + 2\left(\sqrt{m_j^2 + \Delta(m_j^2) + p_j^2} - \sqrt{m_j^2 + p_j^2}\right) \sum_{i\neq j} \sqrt{m_i^2 + p_i^2} \\ &= \Delta(m_j^2) + 2\left(\sum_i E_i - E_j\right) \Delta E_j. \end{aligned} \tag{7.7}$$

See Figure 7.1 for an application of this equation in the two particle decay $B^\pm \to J/\psi\, K^\pm$, where the $K^\pm$ is actually a misidentified $\pi^\pm$. Similarly, Figure 7.2 shows the same, but for the three-body decay $B^0 \to K^+K^-K^0$ where the K^0 is a misidentified π^0. In both cases the correspondence between a straightforward reconstruction assuming misidentification and directly applying equation (7.7) verifies that equation (7.7) is indeed correct.

In the relativistic limit we can make the approximation

$$\begin{aligned}\Delta E_j &= \sqrt{m_j^2 + \Delta(m_j^2) + p_j^2} - \sqrt{m_j^2 + p_j^2} \\ &\approx \frac{\Delta(m_j^2)}{2p_j} \end{aligned} \tag{7.8}$$

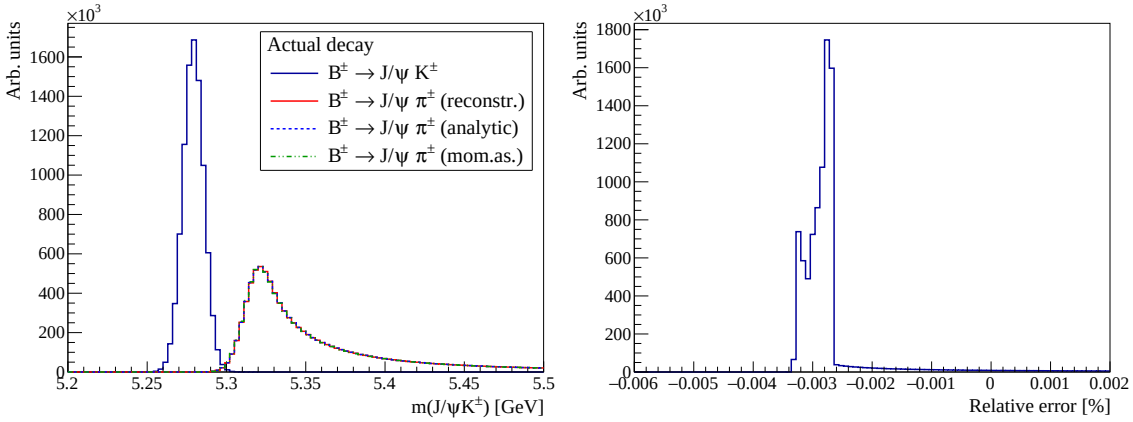

Figure 7.1
Left: Comparison of $B^{\pm} \to J/\psi K^{\pm}$ mass spectra due to a single misidentification of a $\pi^{\pm}$ as $K^{\pm}$, as calculated by (i) reconstruction, i.e. redoing the whole calculation while assuming a different secondary particle mass; by (ii) applying the exact equation (7.7); and by (iii) applying the approximate momentum asymmetry equation (7.11). The simulation is done assuming $B^{\pm}$ with $\gamma = 10$. Right: The relative error of the momentum asymmetry approximation (7.11).

so that (7.7) becomes

$$\begin{aligned}\Delta(M^2) &\approx \Delta(m_j^2)\left(1 + \frac{\sum_{i\neq j} E_i}{p_j}\right) \\ &\approx \Delta(m_j^2)\left(1 + \frac{\sum_{i\neq j} p_i}{p_j}\right) \\ &= \Delta(m_j^2)\frac{\sum_i p_i}{p_j}.\end{aligned} \tag{7.9}$$

Alternatively, if we introduce momentum asymmetry [72]

$$\beta_j = \frac{\sum_{i\neq j} p_i - p_j}{\sum_i p_i} = 1 - \frac{2p_j}{\sum_i p_i}, \tag{7.10}$$

we can express (7.9) as

$$\Delta(M^2) \approx \frac{2\Delta(m_j^2)}{1-\beta_j}. \tag{7.11}$$

The approximations (7.9) and (7.11) give considerably simpler expressions, and are therefore more appropriate for possible further analytical applications [72]. Their accuracy is shown in Figures 7.1 and 7.2, from which it can be seen that the momentum asymmetry approximation is especially accurate for misidentifications in two-body decays, where it has a relative error of about 10^{-5} (for $\gamma = 10$). However, caution must be taken when using this approximation, because its error distribution is not centred about zero. As a result, it will give a biased result. This bias is especially pronounced in the case of the three-body decay from Figure 7.2: while the relative error is only of

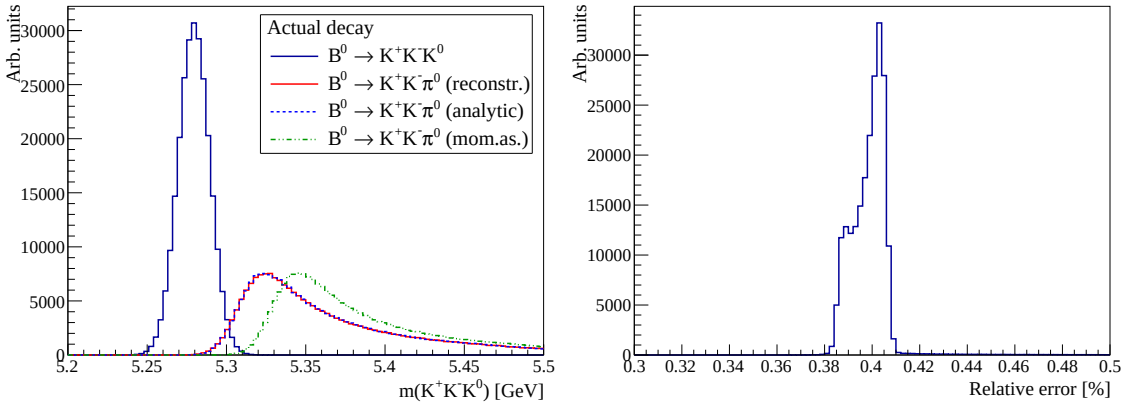

Figure 7.2
Same as Figure 7.1, but for the decay $B^0 \to K^+K^-K^0$, with a single misidentification of a π^0 as a K^0. The primary B^0 has $\gamma = 10$.

the order 10^{-3}, the error is always positive, i.e. the reconstructed primary mass M is always overestimated. This latter property significantly reduces the usefulness of the approximation, especially for decays with $N > 0$, and this should be taken into account.

7.1.3 Multiple misidentifications

If there are multiple misidentifications one can consecutively apply equation (7.7) for each misidentified particle, while adjusting the masses of the misidentified particles between step. That is, if without any misidentifications we reconstruct the primary mass M from the actual masses m_j, for each particle $j = 1, ..., N$:

1. Given the masses (and thus their energies) of all secondary particles, and given the change in the mass of particle j due to misidentification, $\Delta(m_j^2)$, calculate $\Delta(M^2)$ due to this misidentification using equation (7.7).

2. Set $M^2 \to M^2 + \Delta(M^2)$ and $m_j^2 \to m_j^2 + \Delta(m_j^2)$ to use in subsequent steps. Note that this also changes the calculated E_j in subsequent steps.

These steps are equivalent to calculating the path integral derived from equation (7.6) from the point $(m_1^2, \dots, m_N^2)$ to the point $(m_1^2 + \Delta(m_1^2), \dots, m_N^2 + \Delta(m_N^2))$ in the secondary particle mass space:

$$\Delta(M^2) = \sum_i \sum_j \int_{(m_1^2+\Delta(m_1^2),\dots,m_{j-1}^2+\Delta(m_{j-1}^2),\, \mathbf{m_j^2},\, m_{j+1}^2,\dots,m_N^2)}^{(m_1^2+\Delta(m_1^2),\dots,m_{j-1}^2+\Delta(m_{j-1}^2),\, \mathbf{m_j^2+\Delta(m_j^2)},\, m_{j+1}^2,\dots,m_N^2)} d(m_j^2)\,\frac{E_i}{E_j} \tag{7.12}$$

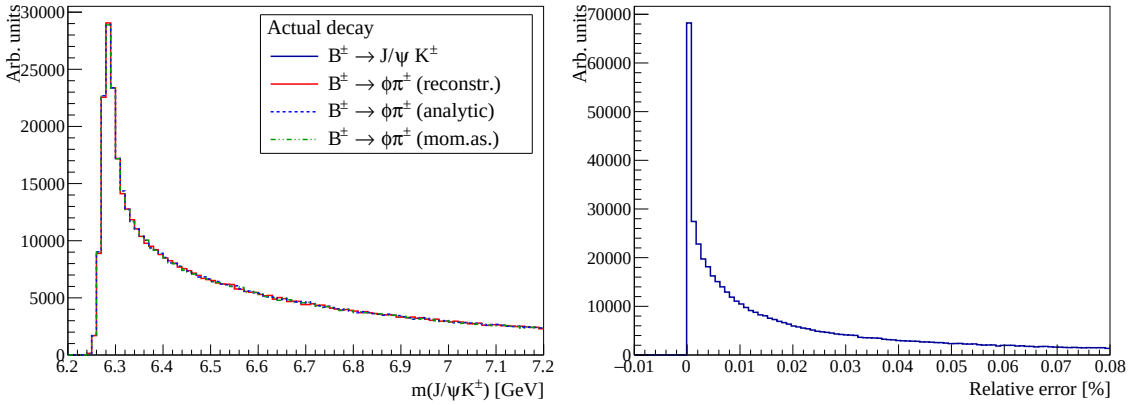

Figure 7.3
Left: Comparison of $B^{\pm} \to J/\psi K^{\pm}$ mass spectra due to a single misidentification of a $\pi^{\pm}$ as $K^{\pm}$, as calculated by (i) reconstruction, i.e. redoing the whole calculation while assuming a different secondary particle mass; by (ii) applying the exact equation (7.7); and by (iii) applying the approximate momentum asymmetry equation (7.11). The simulation is done assuming $B^{\pm}$ with $\gamma = 10$. Right: The relative error of the momentum asymmetry approximation (7.11).

or, more explicitly stated,

$$\begin{aligned}\Delta(M^2) &= \sum_i \sum_j \int_{m_j^2}^{m_j^2+\Delta(m_j^2)} \frac{d(m_j^2)}{\sqrt{m_j^2+p_j^2}} \times \begin{cases} \sqrt{m_i^2+\Delta(m_i^2)+p_i^2} & \text{for } i<j \\ \sqrt{m_i^2+p_i^2} & \text{for } i \geq j \end{cases} \\ &= \sum_j \left[\Delta(m_j^2) + 2\Delta E_j \sum_{i\neq j} \left\{ \begin{array}{ll} \sqrt{m_i^2+\Delta(m_i^2)+p_i^2} & \text{for } i<j \\ \sqrt{m_i^2+p_i^2} & \text{for } i>j \end{array} \right\} \right]. \end{aligned} \tag{7.13}$$

As an approximation, we can skip the updating of the masses m_j after each j-th step (i.e. skip step 2), and obtain an equation similar to 7.7,

$$\begin{aligned}\Delta(M^2) &= \sum_j \left[\Delta(m_j^2) + 2\Delta E_j \sum_{i\neq j} \sqrt{m_i^2+p_i^2} \right] \\ &= \sum_j \left[\Delta(m_j^2) + 2\left(\sum_i E_i - E_j \right) \Delta E_j \right], \end{aligned} \tag{7.14}$$

which is to treat a multiple misidentification as a set of independent single misidentifications. Or, if we assume $p_i \approx E_i$, the approximation can be expressed in terms of momentum asymmetries (7.10) as

$$\Delta(M^2) \approx \sum_j \frac{2\Delta(m_j^2)}{1-\beta_j}. \tag{7.15}$$

To illustrate the validity of equation (7.13), we take the same decays as in the previous section, but now with two misidentified particles. More specifically, Figures 7.3 and 7.4 show mass spectra

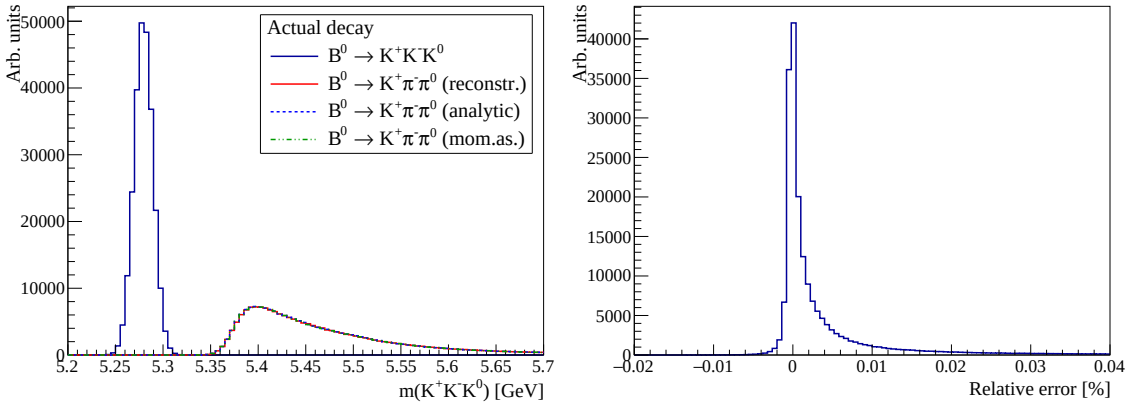

Figure 7.4
Same as Figure 7.1, but for the decay $B^0 \to K^+K^-K^0$, with a single misidentification of a π^0 as a K^0. The primary B^0 has $\gamma = 10$.

for, respectively, $B^\pm \to \phi\pi^\pm$ misidentified as $B^\pm \to J/\psi K^\pm$, and $B^0 \to K^+\pi^-\pi^0$ misidentified as $B^0 \to K^+K^-K^0$. The accuracy of the momentum asymmetry approximation (7.15) is also shown in these figures.

Again, we verify that the derived equation for the transformation of the mass spectrum is correct. In addition, we again find the error on the momentum asymmetry approximation to be low. Interestingly, however, for the three-body decay case, the approximation actually performs considerably better now that more particles are misidentified. This can be understood by noting that the bias observed in the two-body decay example is caused by the fact that the error introduced by switching to the momentum asymmetry always has the same sign for a given particle. Then, for a single misidentification, this error can never average out to zero. However, if multiple particles are misidentified, the errors on the momentum asymmetry are anti-correlated (because a high β for one particle implies a low β for another), and can thus cancel each other. As a consequence, the momentum asymmetry will be more accurate as more particles are misidentified.

7.1.4 Simulating misidentifications

The simulation of misidentifications could in principle be done by implementing equation (7.13), but it is more straightforward to mimic what happens in the detector by following the steps below.

1. Simulate the decay that is actually taking place, which yields a set of four-momenta $(E_i, \mathbf{p}_i)$.
2. Simulate all detector effects (acceptance, resolution, energy loss, etcetera) which affects the four-momenta to give a set $(E_i, \mathbf{p}_i) \to (\tilde{E}_i, \tilde{\mathbf{p}}_i)$.
3. Implement the misidentification step by changing the mass of the actual particle (m_i) to the mass of the particle as which it is misidentified (change $m_i \to m_i'$). This changes the observed energy as mentioned before:

$$\tilde{E}_i = \sqrt{m_i^2 + p_i^2} \to \tilde{E}_i' = \sqrt{m_i'^2 + p_i^2}. \tag{7.16}$$

The momentum remains unchanged.

4. Reconstruct the primary particle's mass as usual, i.e. using equation (7.3).

The implementation of misidentifications can be found in Appendix A.8.1.

7.2 Semi-peaking backgrounds due to missing particles

Alternatively, it may occur that not all particles produced in a decay are observed by the detector.[1] For neutrinos, this is the case by default (neutrinos only interact weakly, so that the probability that they interact with the detector components is basically zero); for example, in the decay

$$B_s^0 \to K^- \mu^+ \nu_\mu, \tag{7.17}$$

the neutrino is never directly observed.

However, this may also occur for other types of particles. For instance, some particle may produce insufficient hits in the detector to be reconstructed, and is consequently missed. As an example, it may happen that we have a

$$B^+ \to \pi^+ \mu^+ \mu^-, \tag{7.18}$$

but that the detector somehow fails to reconstruct the π^+. As a consequence, the decay may be identified as a $B^+ \to \mu^+ \mu^-$ decay instead.

In addition, in both cases the primary particle will be reconstructed with too little energy, as the energy of the missing particles are not taken into account. Mathematically, if X represents the set of missing particles, we will reconstruct a primary mass

$$M' = \sqrt{\left(\sum_{i \notin X} E_i\right)^2 - \left|\sum_{i \notin X} \mathbf{p}_i\right|^2}, \quad \text{with } E_i = \sqrt{p_i^2 + m_i^2}. \tag{7.19}$$

That is, the primary mass reconstruction is done as usual, but without taking into account the missing particles.

Because the momentum that goes missing through missing particle varies considerably between individual decays, the mass spectrum of M' that is produced as a result is relatively smooth. This makes such backgrounds less problematic than backgrounds due to misidentifications. However, these backgrounds do have a particular mass at which they pop up sharply. This mass is given by the configuration where the missing particle has the minimum amount of energy, i.e. by the configuration that the missing particle has zero momentum. Plugging this requirement (i.e. $p_j \geq 0$, where j indicates the missing particle) into (7.19), we obtain a minimal error in the reconstructed mass

$$\Delta M \leq -m_j, \tag{7.20}$$

[1] Note that we must always find at least two particles to consider the event to contain a decay. Therefore, the problem treated in this section does not occur for two-body decays.

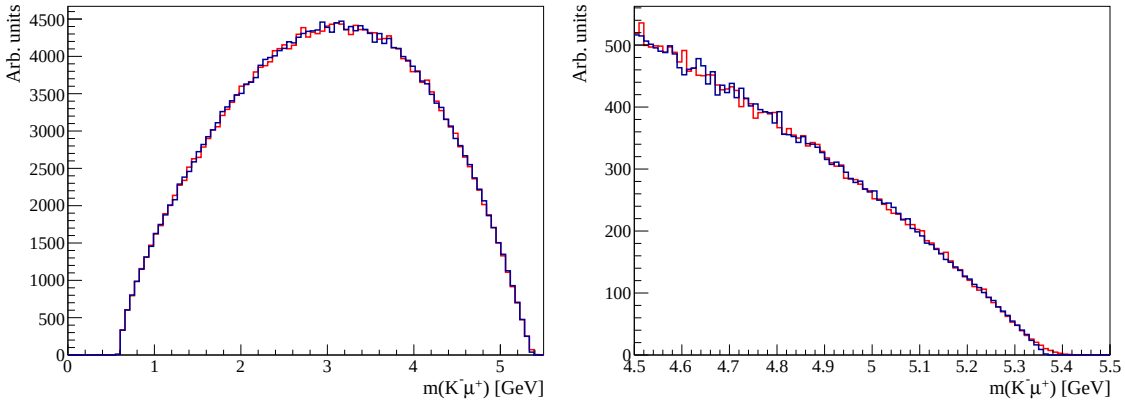

Figure 7.5
Mass spectrum for $B_s^0 \to K^- \mu^+ \nu_\mu$ decays, where the ν_μ is missing. The blue line shows the spectrum based on kinematics; the red line shows the effect of detector resolution. Note that $m(B_s^0) = 5367$ MeV [36].

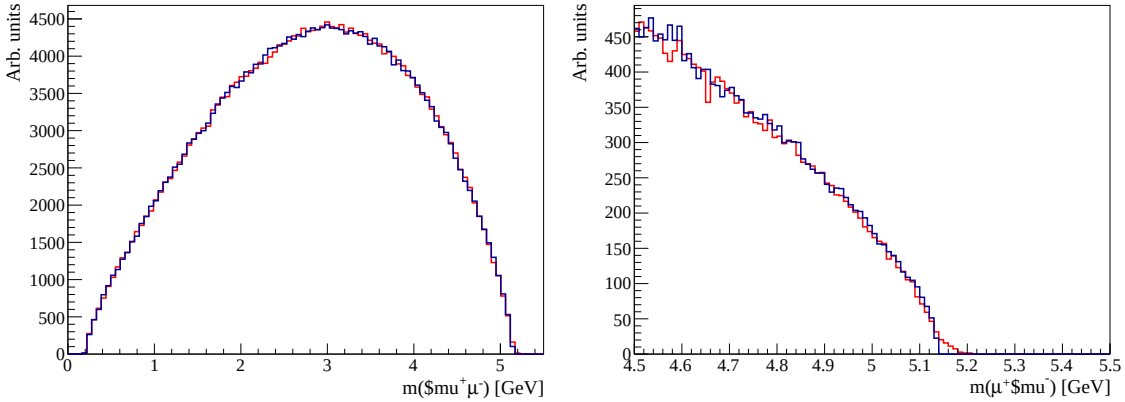

Figure 7.6
Mass spectrum for $B^+ \to \pi^+ \mu^+ \mu^-$ decays, where the π^+ is missing. The blue line shows the spectrum based on kinematics; the red line shows the effect of detector resolution. Note that $m(B^+) = 5279$ MeV, and $m(\pi^+) = 140$ [36], so that the spectrum should indeed vanish around $m(\mu^+\mu^-) \approx 5139$ MeV.

i.e. the minimum reconstructed mass that is missing is equal to the mass of the missing particle. Of course, this does not take into account any detector effects; for instance, the momentum resolution will smear the background, such that we actually have $\Delta M \lesssim -m_j$.

The resulting mass spectra for the example decays (7.17) and (7.18) are shown in Figures 7.5 and 7.6, respectively. Indeed, the backgrounds are found to sharply pop up around $M - m_{\text{missing}}$. Other than that, the distribution is wide and smooth. As such, these backgrounds are only problematic if the region around $M - m_{\text{missing}}$ is included in the experiment's mass window, because in that case these background affect the obtained signal in a non-smooth way. If, on the other hand, the region around $M - m_{\text{missing}}$ is located far from the mass window, the background will be smooth, and it can be easily corrected for.

7.2.1 Simulating missing particles

To exclude a particle from reconstruction in the simulation, we set its energy and momentum equal to zero; it can be easily seen that this is equivalent to using (7.19). Explicitly, the simulation executes the following steps.

1. Simulate the decay that is actually taking place, which yields a set of four-momenta $(E_i, \mathbf{p}_i)$.
2. Set the mass and momentum of the particle that is missing to zero. That is, for each missing particle j, set $(E_i, \mathbf{p}_i) \to (0, \mathbf{0})$.
3. Reconstruct the primary particle's mass as usual, i.e. using equation (7.3).

The implementation of missing particles can be found in Appendix A.8.2.

7.3 Combinatorial backgrounds

As mentioned earlier, combinatorial backgrounds arise if some combination of decays produces the same particles (possibly among other particles) as the signal decay that the experiment is selecting on, such that these particles have the same signature as the signal decay (i.e. they seem to come from a similar vertex as a signal decay would come from; have similar invariant mass as the signal decay, etcetera), and are thus misinterpreted as a signal decay.

Given the large number of interactions and decays that occur within a high-energy collision event, it is not uncommon for some of these decays to coincidentally fake a signal decay. Consequently, the amplitude of the combinatorial background is often relatively large. However, in general, many combinations of such decays can be made, where each of these decays can have many different kinematic configurations. Therefore, the combinatorial background will be made up of many different kinds of events, which all reconstruct to different primary masses. As a result, the combinatorial background is generally relatively smooth, and can often be corrected for by assuming them to have an exponential or polynomial shape.

While the particular origin of the combinatorial background is thus rather complex, it generally does not significantly affect the quality of the experimental mass spectrum because of its smooth nature. As such, simulating the combinatorial background would be an unnecessary complication of our model, and it is therefore not considered further.

Chapter 8

Calibration of the simulation model

For preliminary studies, it is useful if the parameters in the model designed in the previous chapters can be calibrated to reflect the characteristics of the LHCb detector. In this chapter, the method of simulated annealing will be used to find the set of parameters for the Monte Carlo model that best reflect the actual LHCb setup.

8.1 Optimizing Monte Carlo parameters

The traditional procedure for optimizing a set of model parameters to a dataset is to define a goodness-of-fit measure, and use the derivative of this measure with respect to the model parameters to iteratively find candidate parameters that improve the goodness-of-fit measure, until a (sufficiently) optimal set of parameters is found. While this procedure is generally successful, it has two main drawbacks: first, such methods may converge to local rather than global optima if the initial parameters are not chosen carefully; and, second, these methods require a well-defined objective function (the goodness-of-fit measure as function of the model parameters) as they primarily rely on its derivative to find the optimal parameters.

When optimizing the fit between a set of Monte Carlo parameters and observed data, especially the second point is problematic: because each evaluation of the objective function requires drawing a random sample from the Monte Carlo model, the objective function itself is random. As a consequence, its derivative with respect to the model parameters is random as well, and the traditional procedure will fail.[1]

To ensure convergence, other techniques should be employed that can deal with such stochastic objective functions. Moreover, to reduce computational efforts, a method is preferred that can also perform well if the objective function is evaluated using only a small Monte Carlo sample. That is, in our context, each iteration of the optimization algorithm should rely on only a handful of simulated events. Finally, since there may be multiple local optima, the chosen method should by design converge to the global optimum. One relatively simple method that satisfies all these criteria is simulated annealing, which is discussed in the next section.

[1] Because the derivative of the objective function will *on average* point in the right direction, the procedure may still work; however, convergence will generally be slow.

8.2 Simulated annealing

Simulated annealing [97] is an adaptation of the Metropolis-Hastings algorithm, and is analogous to the process of cooling a material in statistical mechanics (hence its name). Qualitatively, the main idea is to make a random walk through parameter space such that moves toward better parameters are accepted with higher probability than moves toward worse parameters. On average, then, the algorithm will gradually move toward the optimal set of parameters.

Given a dataset to fit the model parameter vector $\mathbf{p}$ to, and a goodness-of-fit measure, we can define our objective function GOF($\mathbf{p}\,|\,$data): the goodness-of-fit measure between a Monte Carlo sample generated using parameter set $\mathbf{p}$ and the experimental data, chosen such that lower values of the objective function correspond to better fits. That is, we seek the optimal set of parameters $\mathbf{p}^*$ such that

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}} \left\langle \mathrm{GOF}(\mathbf{p}\,|\,\mathrm{data}) \right\rangle , \tag{8.1}$$

where the brackets denote the expectation value.

Given some user-defined initial parameter set $\mathbf{p}_0^{(0)}$, and with $k = 0, 1, \ldots$ denoting the iteration, the simulated annealing algorithm is then as follows.

1. Pick a set of initial candidate parameter values $\mathbf{p}_0^{(k)}$.

2. Draw a sample $S_0^{(k)}$ from the Monte Carlo model with these initial parameter values and use it to calculate a value for GOF($\mathbf{p}_0^{(k)}|$data).

3. Draw a random new set of candidate parameter values $\mathbf{p}_1^{(k)}$. From what distribution to draw these new values is subject to choice; some popular choices are discussed below.

4. Draw a sample $S_1^{(k)}$ from the Monte Carlo model with the new parameter values and use it to calculate a value for GOF($\mathbf{p}_1^{(k)}|$data).

5. Accept the move towards parameter set $\mathbf{p}_1^{(k)}$ with probability

$$\text{acceptance prob.} = \min \left\{ 1,\ \exp \left[- \frac{\mathrm{GOF}(\mathbf{p}_1^{(k)}\,|\,\mathrm{data}) - \mathrm{GOF}(\mathbf{p}_0^{(k)}\,|\,\mathrm{data})}{T^{(k)}} \right] \right\} , \tag{8.2}$$

   otherwise stay with $\mathbf{p}_0^{(k)}$. $T^{(k)}$, in analogy with annealing often referred to as the temperature, is a (positive) parameter that may differ between iterations: the sequence $\{T^{(k)}\}$ is called the annealing (or cooling) schedule and is discussed below.

6. Repeat these steps with, depending on the previous step, $\mathbf{p}_0^{(k+1)} = \mathbf{p}_0^{(k)}$ or $\mathbf{p}_0^{(k+1)} = \mathbf{p}_1^{(k)}$ as initial parameter set in Step 1. The algorithm stops once some stopping criterion is met, which is discussed below.

One crucial feature of the algorithm is Step 5. The acceptance probability is designed such that an improvement [i.e. GOF($\mathbf{p}_1\,|\,$data) < GOF($\mathbf{p}_0\,|\,$data)] is accepted with certainty; on the other hand, bad moves [i.e. GOF($\mathbf{p}_1|$data) > GOF($\mathbf{p}_0|$data)] are also accepted with limited but non-zero

probability. The extent to which bad moves are accepted is determined by the parameter T: for large T, relatively large deteriorations of the goodness-of-fit are still accepted with considerable probability, while for small T the exponent in (8.2) quickly blows up even for small differences.

The strategy is then to start with a large value for T, which allows the algorithm to scan large areas of the parameter space, and gradually decrease T in order to gradually restrict the parameter search to areas which produce better goodness-of-fit values. If the scheme for decreasing T is chosen appropriately, the algorithm will converge to the (globally) optimal set $\mathbf{p}^*$ [98].

Thus, we can indeed infer that the algorithm is to make a random walk through the parameter space (generated by Step 3), which on average moves towards $\mathbf{p}^*$ because moves towards improvements in the candidate parameter $\mathbf{p}$ occur more often than moves towards worse $\mathbf{p}$ (due to Step 5).

Choosing new candidate parameters

In many cases, for drawing a new candidate set of parameters, a normal distribution centred about the current values is chosen, i.e. $\mathbf{p}_1^{(k)} \sim N(\mathbf{p}_0^{(k)}, \sigma^{(k)})$ with some chosen covariance matrix $\sigma^{(k)}$. Generally, the covariance matrix will be diagonal, and the elements are chosen such that they resemble reasonable step sizes. Choosing such step sizes is mostly a process of trial-and-error, and is not always straightforward: if the step size is too large, the candidate parameters may continuously oscillate about the optimal set rather than converge to it, and if it is too small, convergence may simply be too slow, and barriers between different local optima may not be overcome.

One strategy to counteract this is to gradually reduce the size of the elements in $\sigma^{(k)}$. This allows the algorithm to approach the region of the optimum relatively fast, after which it is allowed to converge to this optimum as the step size becomes more fine-grained. For example, Neal [99] suggests that the step sizes should scale as $\sqrt{T}$ to keep the acceptance rate approximately constant.

Alternatively, any other (preferably symmetric) distribution may be used for generating new candidates. For example, a discrete random walk can be implemented by moving to $p_{1i}^{(k)} = p_{0i}^{(k)} \pm d_i^{(k)}$, both with probability 0.5. Again, the same considerations apply to the choice of the step sizes $d_i^{(k)}$ as to the choice of the matrix $\sigma^{(k)}$.

Selecting an annealing schedule for T

Regarding the annealing schedule $\{T^{(k)}\}$, several decisions are to be made: (i) the initial temperature $T^{(0)}$; (ii) the final temperature; and (iii) how to decrease the temperature over the iterations.

As a rule of thumb, at the beginning the temperature should be sufficiently large as to explore a large region of the parameter space. This is achieved by setting $T^{(0)}$ of order GOF(no overlap between distributions) – GOF(perfect fit). In that case, even the worst possible moves are accepted with probability of order $1/e$.

Ideally, the algorithm converges to a perfect fit. However, even if the parameters are chosen perfectly, there will still be some variance in the goodness-of-fit statistic. These parameters should in principle always be accepted with high probability. As such, the minimum temperature must be of the order of the statistical deviation of the goodness-of-fit statistic under the hypothesis that both samples are drawn from the same distribution. The specific values for the minimum and maximum

temperatures obviously depend on the specific goodness-of-fit measure chosen; we turn to this in the next section.

Several schedules for decreasing the temperature are discussed in Ref. [99]. The most common, and often most appropriate, choice is to multiply T with some constant $0 < \alpha < 1$ after each iteration, i.e.

$$T^{(k+1)} = \alpha T^{(k)}, \qquad 0 < \alpha < 1. \tag{8.3}$$

Generally, an appropriate α is determined by trial-and-error, as there are no good systematic ways of choosing this constant.

Stopping criterion

One could in principle use stopping criteria that are coupled to the convergence of the algorithm. For instance, the algorithm can be instructed to stop once it has stayed within a sufficiently small region of the parameter space for a sufficient number of iterations. However, this may be complicated to implement and require additional computations. Instead, it is generally also sufficient to let the algorithm run for some maximum number of iterations, and check for convergence afterwards using e.g. trace plots of the parameters.

8.3 Goodness-of-fit between two samples

Specifically, we are interested in comparing two empirical distributions, namely the distribution of N experimental observations $\{x_1, \ldots, x_N\}$, and the distribution of a Monte Carlo sample $\{y_1, \ldots, y_M\}$ of size M. Then, if the data points are labelled such that $x_1 \leq x_2 \leq \cdots \leq x_N$ and $y_1 \leq y_2 \leq \cdots \leq y_M$, we have the empirical cumulative distribution functions (ECDFs)

$$F_N(x) = \begin{cases} 0 & \text{for } x < x_1 \\ i/N & \text{for } x_i \leq x < x_{i+1} \\ 1 & \text{for } x \geq x_N \end{cases} \qquad G_M(x) = \begin{cases} 0 & \text{for } x < y_1 \\ i/M & \text{for } y_i \leq x < y_{i+1} \\ 1 & \text{for } x \geq y_M \end{cases} \tag{8.4}$$

These can be compared using several often-used statistics which are discussed below.

Instead of using ECDFs, the sample can also be binned into histograms, which can subsequently be compared with the two-sample chi-squared statistic [100]

$$\chi^2 = \sum_i \frac{\left(\sqrt{N/M}\, H_i^{(\text{data})} - \sqrt{M/N}\, H_i^{(\text{sample})}\right)^2}{H_i^{(\text{data})} + H_i^{(\text{sample})}}, \tag{8.5}$$

where the summation is over all bins i, and $H_i^{(\cdot)}$ denotes the contents of the ith bin. Although this is an often-used approach to such problems, this statistic has several drawbacks. First of all, binning the data unnecessarily destroys information in an arbitrary manner, which may decrease the quality of the statistic. Second, the assumptions underlying the chi-squared statistic are only valid if each bin is sufficiently populated.[2] Both these problems arise in particular when sample

[2] As a solution, the histograms could be binned equiprobably (under the null hypothesis) [101], but this is computationally more expensive, and is still somewhat arbitrary.

sizes are small, which thus requires repeatedly drawing relatively large Monte Carlo samples for each candidate parameter set. This is undesirable, and we therefore explicitly depart from binning the samples in favour of using ECDFs.

The most-used statistics for comparing two ECDFs are the Kolmogorov–Smirnov statistic, which is the maximal vertical distance between the ECDFs,

$$\mathrm{KS} = \sup_x \left| F_N(x) - G_M(x) \right|, \tag{8.6}$$

and statistics with a quadratic loss function,

$$L = \frac{NM}{N+M} \int_{-\infty}^{\infty} \mathrm{d}H_{N+M}(x)\, w(x) \left[F_N(x) - G_M(x) \right]^2, \tag{8.7}$$

where

$$H_{N+M}(x) = \frac{1}{N+M} \left[N F_N(x) + M G_M(x) \right] \tag{8.8}$$

and $w(x)$ is some weighting function. Two popular choices for this weighting function are

$$w(x) = 1 \qquad \text{(Cramér–von Mises)} \tag{8.9a}$$

$$w(x) = \left\{ H_{N+M}(x) \left[1 - H_{N+M}(x) \right] \right\}^{-1} \qquad \text{(Anderson–Darling)} \tag{8.9b}$$

While the Kolmogorov–Smirnov statistic (8.6) is slightly less expensive to calculate, the maximal vertical distance often occurs somewhere in the middle of the distributions, and is consequently less sensitive to differences in the shape of the distributions. A quadratic loss function (8.7), on the other hand, evidently includes more information on these shapes, and is therefore often superior [102]. It is not straightforward to say which weight function (8.9a) or (8.9b) is more appropriate; however, because the Anderson–Darling statistic places more weight on the tails of the distribution, which is important if we want to be sensitive to e.g. the momentum-dependence of the measurement error (as we saw in Chapter 5, the shape of the tails of the mass spectrum depends crucially on the momentum resolution parameters), this statistic seems better suited.

Finally, we should consider the case where the chosen set of candidate parameters is far off from the actual parameters, such that there is no overlap between the sample and data distributions. In that event, the quadric loss (8.7) is maximal, and may not change if the candidate parameters are not sufficiently changed (because there is still no overlap then). As a consequence, these statistics can only rank the quality of two alternative candidate parameter sets if at least one of these sets is sufficiently close to the true parameter set. Since the ability to rank two distributions is essential for the numerical algorithm to proceed, this should be counteracted. Therefore, we extend these statistics by including the horizontal distance between the two distributions in case there is no overlap.

Rewriting equation (8.7) in terms of the experimental data and Monte Carlo samples, we obtain

$$L = \frac{NM}{(N+M)^2} \left\{ \sum_{i=1}^{N} w(x_i) \left[G_M(x_i) - \frac{i}{N} \right]^2 + \sum_{j=1}^{M} w(y_j) \left[F_N(y_j) - \frac{j}{M} \right]^2 \right\}. \tag{8.10}$$

Here we have used that $\mathrm{d}H_{N+M}(x)/\mathrm{d}x$ is a sum of delta functions (one delta function for each observation), and we have inserted (8.4) where possible.

If there is no overlap between the samples, i.e. $x_N < y_1$ or $y_M < x_1$, for the Cramér–von Mises statistic, we get [103]

$$L_{\max}^{\mathrm{CVM}} = \frac{NM}{N+M} - \frac{4NM-1}{6(N+M)} \tag{8.11a}$$

and for the Anderson–Darling statistic we obtain [103]

$$L_{\max}^{\mathrm{AD}} = \frac{M}{N}\sum_{i=1}^{N}\frac{i}{N+M-i} + \frac{N}{M}\sum_{j=1}^{M}\frac{M-j}{N+j}. \tag{8.11b}$$

Then, to account for the horizontal distance between non-overlapping distributions, we add the difference in sample means to L if $L = L_{\max}$, to obtain the extended statistic L':

$$L' = \begin{cases} L & \text{for } L < L_{\max} \\ L_{\max} + \left[\sum_{i=1}^{N} x_i/N - \sum_{j=1}^{M} y_j/M\right] & \text{for } L = L_{\max} \end{cases}. \tag{8.12}$$

Again, it is emphasised that this extension is made only to ensure proper behaviour of the numerical algorithm in the case that the two distributions do not overlap.

On the other hand, if the underlying distributions are equal, each ordering of the pooled sample is equally likely, and one can calculate the corresponding expected value of the statistic in that case, L_0. Conveniently [and one of the primary reasons for choosing a statistic of the form (8.7)], the value L_0 is independent of the underlying distribution, and is given (in the limit that $N, M \to \infty$) [103, 104]

$$L_0^{\mathrm{CVM}} \to 1/6, \tag{8.13a}$$

$$L_0^{\mathrm{AD}} \to 1. \tag{8.13b}$$

Similarly, the corresponding variances V_0 of the statistics under the null hypothesis that the distributions are equal, are distribution-independent, and are given by (again in the limit of infinite sample sizes) [103, 104]

$$V_0^{\mathrm{CVM}} \to 1/45, \tag{8.14a}$$

$$V_0^{\mathrm{AD}} \to 2(\pi^2 - 9)/3 \approx 0.58. \tag{8.14b}$$

As discussed in the previous section, we can use these properties to derive a suitable annealing schedule: it should have a starting temperature of order $L_{\max}$ at the most, as given by equations (8.11), and a final temperature equal to (or smaller than) $\sqrt{V_0}$ as given by (8.14).

8.4 Test case: the LHCb momentum resolution

To validate the simulated annealing algorithm described in the previous sections, and to be able to assess the effects of certain parameter choices [e.g. α in the annealing schedule (8.3)], we should first use the model to solve a relatively simple case for which the solution is (to good extent) known. As a test, therefore, we assess the LHCb momentum resolution using data on the decay

$$J/\psi \to \mu\mu. \tag{8.15}$$

This decay is used for two main reasons. First, the J/ψ is relatively long-lived, and thus has a very narrow decay width. As a result, the variance in the obtained mass shape of the J/ψ, i.e. the mass resolution, will be almost entirely due to momentum resolution effects. As such, it is relatively straightforward to extract the momentum resolution from the obtained mass spectrum.[3] Second, the production rate for J/ψs is relatively high, as is the branching ratio to $\mu\mu$, so that the dataset contains sufficiently many observations for our purpose.

In line with Refs. [51, 75], particle momentum errors are assumed to be normally distributed with standard deviation

$$\frac{\sigma(p)}{p} = a + bp, \tag{8.16}$$

with a resembling multiple scattering effects (which dominate for low particle momenta), and b resembling hit resolution effects (which dominate for high momenta); observed values for these parameters are $a \approx 3 \times 10^{-3}$ and $b \approx 2 \times 10^{-5}$ GeV^{-1} [75].

We can then validate our method by checking whether it is able to reproduce these values for a and b if a Gaussian distribution (8.16) is assumed in the Monte Carlo model. To that end, we first summarize the choices made regarding the simulated annealing algorithm, and subsequently discuss its results.

8.4.1 The simulated annealing setup

As mentioned above, the Anderson–Darling goodness-of-fit statistic seems more appropriate for our purposes, as the results from Chapter 5 have illustrated that the effects of many parameters manifest themselves in the tails of the observed mass spectrum. For that reason, in the following we will limit ourselves to the use of only the Anderson–Darling statistic.

The further setup is then as follows. The initial temperature is given by $L_{\max}/100$ [i.e. equation (8.11b)], and the final temperature by $\sqrt{V_0}/50$ [i.e. from equation (8.14b)]. Here, the numerical factors are determined by trial-and-error, i.e. by doing the annealing procedure for different factors and seeing for what factors the procedure gives the best results (e.g. in terms of convergence). For decreasing the temperature T, we use the geometric annealing schedule (8.3), where α is chosen such that decreasing from the initial to the final temperature takes 10^5 iterations.

Regarding the parameters a and b, we choose the initial parameters

$$a^{(0)} = 10^{-3} \quad \text{and} \quad b^{(0)} = 10^{-5}\ \text{GeV}^{-1}. \tag{8.17}$$

Subsequent candidate parameters are drawn from the normal distributions

$$a^{(k+1)} \sim N\left(a^{(k)}, 10^{-3} \times \sqrt{T^{(k)}/T^{(0)}}\right) \quad \text{and} \quad b^{(k+1)} \sim N\left(b^{(k)}, 10^{-5} \times \sqrt{T^{(k)}/T^{(0)}}\right).$$

Moreover, we will not use all events observed in the LHCb dataset, but only those for which the likelihood that the event was indeed a $J/\psi \to \mu\mu$ decay.[4] This will reduce the number of

[3] If the decay width is large, the mass resolution can be modelled by e.g. a Gaussian momentum smearing convolved with a Breit–Wigner function with a fixed width [105].

[4] As explained in Chapter 3, each observed particle is assigned a probability of being a certain particle (e.g. μ, π, p, ...) based on the combined information from different detector components. In our particular Ntuple, this is included in the variable `ProbNNmu`, which for each particle in each event states what the likelihood is (on a scale from 0 to 1) that the particle was a muon. For the annealing procedure, we use only events for which `ProbNNmu` > 0.9 for both muons.

misidentified events in the dataset, which is important because these change the shape of the mass spectrum, and thus affect our estimation. Of course, this will also eliminate correctly identified events, but this will only affect the amplitude and not the shape of the mass spectrum; and given that many events remain, even after relatively stringent cuts, this is not problematic. After these cuts, we have a dataset containing 63,386 events (of 559,084 candidate events in the original Ntuple). This subset is also used for determining the energy distribution of the primary J/ψs in the simulation (again, since the energy distribution of the J/ψs affects the mass spectrum shape, it is essential that the energy distributions of the data and the simulated events are the same).

8.4.2 Performance of the simulated annealing method

The results of the annealing procedure are shown in Figure 8.1; this figure includes both the resulting optimized model, and information on how the procedure has progressed. For example, from the trace plots (middle row) we can see that, as expected, initially a large part of the parameter space is scanned (there is large variation in the candidate values for early iterations), and, as the algorithm proceeds, it is constrained to an increasingly small part of the parameter space; i.e. as the temperature parameter decreases, the algorithm indeed converges to some set of parameters. The top row of Figure 8.1, that shows histograms of the candidate parameter values for the last 20,000 iterations, also illustrates this convergence, by showing that for low temperatures, some parameter values indeed become strongly preferred over others.

As can be seen from the bottom row of Figure 8.1, the simulated spectrum based on the optimized parameters,

$$a^* = 2.95 \times 10^{-3} \quad \text{and} \quad b^* = 7.89 \times 10^{-5}\ \text{GeV}^{-1}, \tag{8.18}$$

agrees well with the experimental spectrum. In addition, the found value for a^* is almost exactly the value suggested in the literature; the value for b^* is also of the right order of magnitude. Given these observations, and especially given the fact that the annealing algorithm allows us to produce a spectrum that fits the experimental data almost exactly, we may conclude that the algorithm works as intended.

Now, in this section we have only presented a relatively simple problem. For such problems, it may seem more appropriate to find a solution by simply using samples that are large enough to sufficiently reduce any random variations in the goodness-of-fit measure, so that traditional derivative-based methods can be employed. However, regarding this point, we should stress two issues: first, the parameter space may contain many local optima, in which traditional methods may get stuck; and, second, as the number of function evaluations (which requires drawing a new Monte Carlo sample) required to find the optimum rapidly increases with the number of parameters, traditional methods become prohibitively slow as the complexity of the model to be optimized increases. As such, traditional methods are not in general suitable for the particular problem of optimizing Monte Carlo parameters, and therefore it is desirable to have a method that specifically addresses this problem available.

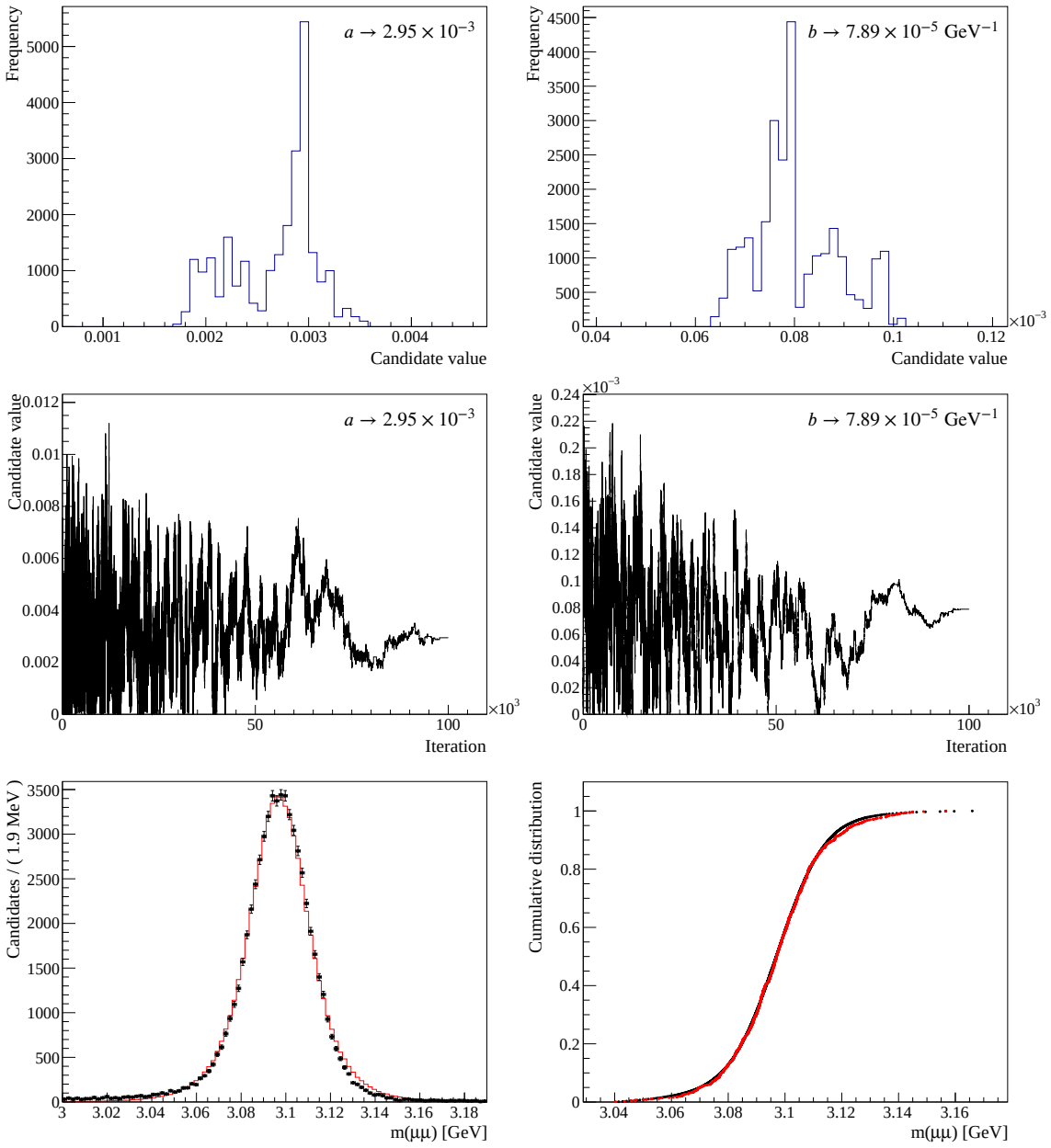

Figure 8.1
The results from the annealing setup described in Section 8.4.1. Top row: histograms of the candidate values for both parameters over the last 20,000 iterations. Middle row: trace plots of the candidate values, i.e. plots that show the candidate values per iteration. Bottom row: the resulting fits, where the black markers denote the data, and the red markers denote events simulated using the optimized parameters.

Chapter 9

Fitting experimental mass spectra

As we have mentioned before, because the detection process is inherently imperfect—besides resolution, acceptance and energy loss effects, we may misidentify or miss particles—a real experimental mass spectrum will contain certain backgrounds. In many new physics searches, in particular when looking for processes that are suppressed (e.g. $B^0_{(s)} \to \mu\mu$, which cannot decay via tree-level decays, but only via higher-order diagrams [106]) or even forbidden (e.g. $D^0 \to e\mu$, which is forbidden by lepton flavour number conservation [107]) by the Standard Model, the expected signal due to events of interest is much smaller than the expected signal due to background events, in which case it becomes important to correct for these backgrounds in order to extract the true signal from the observed spectrum.

To illustrate the problem, see Figure 9.1, which shows the experimental mass spectrum for the $B^0_{(s)} \to \mu\mu$ decays from LHCb [64]. In particular, it shows the spectra for each type of event separately, and how these separate spectra make up the observed spectrum. Now, these separate event spectra are of course not directly observable, but must somehow be estimated (otherwise we could just trivially subtract all background spectra to obtain the true signal).

Estimating these separate event spectra using the simulation model constructed in the previous chapters is the subject of this chapter. Qualitatively, this estimation comes down to finding the combination of separate event spectra that best reproduces the observed spectrum (in Figure 9.1 this best fit is shown by the solid blue line). In the next section (Section 9.1), we will formulate the problem of finding this best in the context of statistics, to set the stage for the two different estimation methods discussed in Sections 9.2 and 9.2. Section 9.4 then outlines the actual procedure, and in Section 9.5 two test cases are discussed to examine the performance of the fitting procedure.

9.1 The problem of fitting

Suppose an experiment observes some mass spectrum, i.e. a set of N reconstructed invariant masses $\{M_1, \ldots, M_N\}$. In the context of statistics, these masses can be interpreted as being draws from an underlying distribution $f(M)$,

$$\{M_1, \ldots, M_N\} = N \text{ draws from } f(M). \tag{9.1}$$

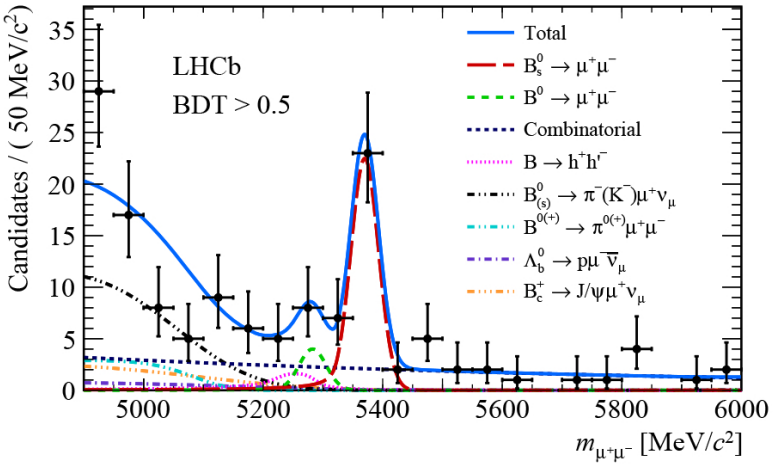

Figure 9.1
Example of an experimental mass spectrum (black markers) where all types of events that make up the spectrum (i.e. the true signal and all backgrounds; the dashed lines) are shown separately. From Ref. [64]

Then, since the observed mass spectrum includes several types of events (i.e. one or more signal events plus background events), labelled i, we can make the decomposition

$$f(M) = \sum_i n_i s_i(M), \tag{9.2}$$

where n_i is the relative abundance of event i (such that $\sum_i n_i = 1$) in the spectrum, and $s_i(M)$ is the mass spectrum distribution for event i (which, if event i is a background, also includes e.g. misidentification or partial reconstruction effects). Indeed, it is exactly these separate event spectrum distributions $s_i(M)$ that we have studied in previous chapters.

The decomposition (9.2) then implies that we can also decompose our observed spectrum (9.1),

$$\begin{aligned} \{M_1, \ldots, M_N\} = {} & N_1 \text{ draws from } s_1(M) \\ & + N_2 \text{ draws from } s_2(M) \\ & + \ldots \end{aligned} \tag{9.3}$$

where $\sum_i N_i = N$. Obtaining this decomposition (i.e. the numbers N_i) is often the objective of experiments. For example, in the context of the $B^0_{(s)} \to \mu\mu$ search of Figure 9.1, knowledge of $N_{B^0_{(s)}\to\mu\mu}$ and of the total number of $B^0_{(s)}$ particles produced would allow the experimenter to calculate the branching fractions $\mathrm{BR}(B^0_{(s)} \to \mu\mu)$.

It is therefore necessary to have a procedure for estimating the parameters N_i based on the observed set of masses $\{M_1, \ldots, M_N\}$. In the following, we will treat two methods that are most often used: the chi-squared method, which is based on binned data; and the maximum likelihood method, based on unbinned data. The former is generally used if many candidate events have been observed, because it is computationally less expensive to use binned data, whereas the latter is generally used in cases where statistics are low, because in such cases one can often not afford

any loss of data incurred by binning (we have also mentioned this issue in Chapter 8). For both methods, it can be shown that in the limit of infinitely many observations ($N \to \infty$), they will yield parameter estimates $\{N_i\}$ that converge to the true values of $\{N_i\}$ (i.e. both methods are consistent).

9.2 Chi-squared estimation

If we bin the observed invariant masses, we obtain a histogram $\tilde{F}$. Denoting the j-th bin, that ranges from $m_j + \Delta m$,[1] as $\tilde{F}_j$, the expectation value of its content is

$$\left\langle \tilde{F}_j \right\rangle = N \int_{m_j}^{m_j+\Delta m} \mathrm{d}m\, f(m), \tag{9.4}$$

which, using the decomposition (9.3), becomes

$$\left\langle \tilde{F}_j \right\rangle = \sum_i N_i \int_{m_j}^{m_j+\Delta m} \mathrm{d}m\, s_i(m). \tag{9.5}$$

Introducing the shorthand $\int_{m_j}^{m_j+\Delta m} \mathrm{d}m\, s_i(m) = S_{ij}$, we simply obtain

$$\left\langle \tilde{F}_j \right\rangle = \sum_i N_i S_{ij}. \tag{9.6}$$

Now, given the values for S_{ij} (we can calculate these values up to arbitrary precision using our simulation model), our objective is to find the set of $\{N_i\}$ that simultaneously fit all expected bin values $\left\langle \tilde{F}_j \right\rangle$ (which can be seen as functions of the set $\{N_i\}$) as closely as possible to the observed bin values $\tilde{F}_j$.

The chi-squared method in particular achieves this by choosing the coefficients N_i^* that minimize a quadratic loss function [101, 108], i.e.[2]

$$\{N_i^*\} = \underset{\{N_i\}}{\operatorname{argmin}} \sum_j \left(\tilde{F}_j - \sum_i N_i S_{ij} \right)^2. \tag{9.7}$$

[1] For simplicity, we assume equal bin widths, i.e. Δm is equal for all bins j. Unequal bin widths could trivially be included by replacing the constant width Δm with a bin-specific width Δm_j throughout this section.

[2] More precisely, the chi-squared method also weighs each bin with the inverse of its variance. Because the filling of bins is a Poisson process, the standard error on the bin content quickly converges to the normal distribution (by the central limit theorem) with deviation $\left\langle \tilde{F}_j \right\rangle$. Taking this into account, we should have

$$\{N_i^*\} = \underset{\{N_i\}}{\operatorname{argmin}} \sum_j \left(\frac{\tilde{F}_j - \sum_i N_i S_{ij}}{\sum_i N_i S_{ij}} \right)^2.$$

However, we ignore this point for two reasons: first, especially when the sample size is relatively small, the error assigned to a bin depends on one's assumptions and methods (e.g. frequentist errors may differ from Bayesian errors) and is therefore ambiguous; and, second, it is also common for many physics analyses to ignore this point [109].

As mentioned before, the chi-squared method has the advantage that it is computationally relatively inexpensive. In addition, because the function to be minimized in equation (9.7) is relatively simple, numerical algorithms for minimizing this loss function will converge relatively easily. However, it does have the drawback that binning the data will destroy some of the information in the dataset, making it inherently less appropriate for applications in which statistics are low or in which high accuracies are required. For such applications, one usually turns to maximum likelihood estimation instead.

9.3 Maximum likelihood estimation

Alternatively, maximum likelihood estimation (MLE) [101, 108] makes use of the unbinned set of observed masses $\{M_1, \ldots, M_N\}$. The main idea behind MLE is to determine the probability density function $f^*(M)$ that is most likely to generate this observed set $\{M_1, \ldots, M_N\}$. To that end, we define the likelihood as

$$\begin{aligned}\mathcal{L}[f(M)|\{M_1, \ldots, M_N\}] &= \text{Prob(observing } \{M_1, \ldots, M_N\}|f(M)) \\ &= \prod_{k=1}^{N} \text{Prob(observing } M_k|f(M)) \\ &= \prod_{k=1}^{N} f(M_k) \end{aligned} \tag{9.8}$$

where we have used that the observations are identically and independently distributed. Now, as before, we assume the probability density function that generated our observed spectrum to be of the form of equation (9.2), to write the likelihood in terms of our coefficients of interest, N_i,

$$\mathcal{L}(N_i|\{M_1, \ldots, M_N\}) = \prod_{k=1}^{N} \left(\frac{1}{N} \sum_i N_i s_i(M_k) \right). \tag{9.9}$$

Note that the likelihood is a function of the set of parameters $\{N_i\}$, and takes the observed data $\{M_1, \ldots, M_N\}$ as given. Moreover, because the term in parentheses should be a probability distribution, we have inserted a term $1/N$ to ensure that it is normalized. The separate event spectra $s_i(M)$ can again be calculated up to arbitrary precision using the developed simulation model.

The MLE method is then to determine the set of coefficients $\{N_i\}$ that maximizes this likelihood, i.e.

$$\{N_i^*\} = \underset{\{N_i\}}{\text{argmax}} \prod_{k=1}^{N} \left(\frac{1}{N} \sum_i N_i s_i(M_k) \right). \tag{9.10}$$

Alternatively, because the logarithm is a monotonously increasing transformation, it is equivalent to maximize the logarithm of the likelihood (called the log-likelihood, for short). The problem can then be stated as, after eliminating the constant $\ln(1/N)$ term from the expression,

$$\{N_i^*\} = \underset{\{N_i\}}{\text{argmax}} \sum_{k=1}^{N} \ln \left(\sum_i N_i s_i(M_k) \right). \tag{9.11}$$

Because the product in (9.10) will produce extremely small numbers, that are generally beyond the machine precision of any computer, for numerical optimization equation (9.11) is preferred.

9.4 The fitting procedure

Now that we have define two conditions for finding estimates $\{N_i^*\}$ for the coefficients $\{N_i\}$, i.e. equations (9.7) and (9.11), we can explicitly outline our fitting procedure.

1. Define a set of N observed reconstructed masses $\{M_1, \ldots, M_N\}$. If the chi-squared method is chosen, bin these observations into a histogram $\tilde{F}$.

2. Define all types of events than must be included in the fit. That is, define what the signal event is, and define all background signals that follow from misidentifications or partial reconstructions. From these definitions, the separate event spectra $s_i(M)$ are calculated. In both cases, these spectra will be approximated by a histogram. For chi-squared fits, this will directly yield the values for S_{ij} in equation (9.7) without introducing any errors. For likelihood fits, on the other hand, this implies that we discretize the (otherwise continuous) probability density function in (9.11), which does give rise to a small error; nonetheless, as long as the bin width is relatively small (and we produce a sufficiently large Monte Carlo sample), this error will be negligible.

3. Define the combinatorial background. Since this background is generally smooth, it does not need to be simulated, but can instead simply be described by an exponential or polynomial function; this function will then be used to describe the combinatorial spectrum $s_{\text{combinatorial}}(M)$. While for other events only the amplitudes N_i are fitted, for the combinatorial other parameters can be fitted as well (e.g. the combinatorial can be defined to be $s_{\text{combinatorial}}(M) = \alpha_0[1 + \alpha_1 \exp(\alpha_2 M)]$, and the procedure will not only estimate α_0 but α_1 and α_2 as well).

4. Find the best estimates according to (9.7) or (9.11), depending on the method chosen. This step is implemented in terms of ROOT's `TMinuit` class [110], which is a class specifically designed for the numerical optimization of functions.

The implementation of this procedure can be found in Appendix A.11. For the remainder of this chapter, we will discuss several example applications of this fitting procedure.

9.5 Test cases: fitting LHCb data

In this section we apply the procedure from the previous sections to actual LHCb datasets. As with the calibration in Chapter 8, we start out with the decay $J/\psi \to \mu\mu$, because for this decay the actual $J/\psi \to \mu\mu$ events strongly dominate any backgrounds in the spectrum. As such, we can ignore all these backgrounds and simply fit the simulated $J/\psi \to \mu\mu$ spectrum to the data (we do however include a smooth combinatorial background).

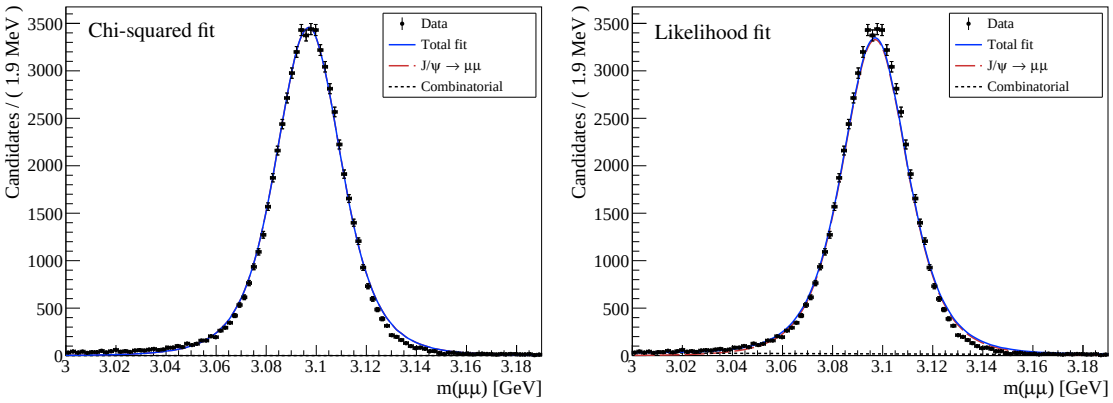

Figure 9.2
Fits to the LHCb data for $J/\psi \to \mu\mu$, using both the chi-squared method (left) and the maximum likelihood method (right). To select the events in the dataset, the same cuts are applied as in Section 8.4.

Then, we will compare the results from our fitting procedure to the results from an actual analysis, by examining data for $B_s^0 \to e\mu$. This dataset provides an ideal test for the model, as it contains both a well-detectable signal and several different types of backgrounds (see Figure 9.1).

9.5.1 Fitting $J/\psi \to \mu\mu$

The fit to the experimental data for $J/\psi \to \mu\mu$ is shown in Figure 9.2. As expected, given the calibration of our model to this channel, a good fit is found. In addition, we can compare the chi-squared fit with the likelihood fit, and find that they indeed give similar results. There is, however, a small difference: whereas the chi-squared fit finds no combinatorial background,[3] the likelihood fit does find a small combinatorial background contribution. Nonetheless, this contribution is only marginal, and we can therefore conclude that for large samples, both methods will give approximately equal results.

9.5.2 Fitting $B^0_{(s)} \to \mu\mu$

Contrary to the $J/\psi \to \mu\mu$ fit, the $B^0_{(s)} \to \mu\mu$ suffers from the facts that (i) statistics are low; and (ii) there are several backgrounds in the analysis region (4.9–6.0 GeV) that need to be corrected for. Both these issues make the fitting procedure inherently more complicated, and are therefore a good test of our model.

The results for the fit to the experimental data for $B^0_{(s)} \to \mu\mu$ are shown in Figure 9.3. While for both the chi-squared and likelihood fits, the total fit agrees well with the total fit found by the LHCb analysis [64], there are some notable discrepancies between the two estimation methods and between our model and the actual LHCb analysis.

First, there is a qualitative difference between the results of the two estimation methods that becomes apparent in this fit. Because the likelihood method estimates only a probability distribution

[3] In the fit, we have inserted a generic combinatorial background of the form $s_{\text{combinatorial}}(M) = \alpha_0[1+\alpha_1 \exp(\alpha_2 M)]$.

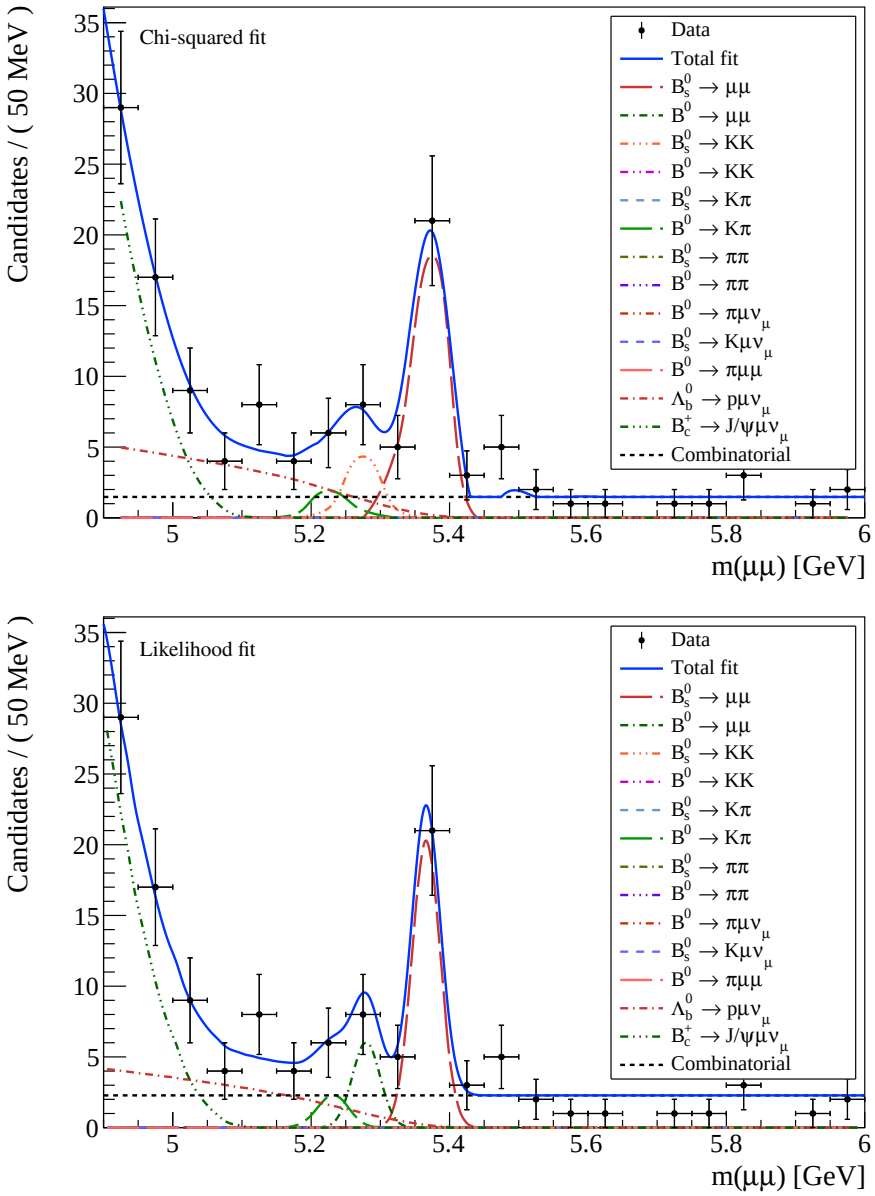

Figure 9.3
Fits to the LHCb data for $B^0_{(s)} \to \mu\mu$, using both the chi-squared method (top) and the maximum likelihood method (bottom). To select the events in the dataset, the same cuts are applied as in Ref. [64]. Note that the fitted shapes in both fits are not equal; this is a programming artefact of the simulation model (specifically, the shapes shown are numerical interpolations of the simulated spectra—but for the chi-squared fit these probability distributions need to be calculated only with resolution equal to the bin width, which gives results in inaccurate interpolations). The bump at 5.5 GeV in the chi-squared fit is a similar artefact.

function for the total fit, and not its amplitude, we obtain a total fit curve that is normalized to unity; this curve is subsequently normalized to the number of observations in the dataset. However, this is not the case for the chi-squared fit, in which the amplitude of the total fit is estimated as well. In other words, the integral of the total fit is fixed in case of the likelihood estimation, but variable in case of the chi-squared estimation. Therefore the likelihood fit may be more adequate, as this forces the fit to explain all events in the sample.

This difference aside, there are also some discrepancies between the two methods in terms of which types of event are found, and the amplitudes associated with them. In particular, a disturbing error in the chi-squared fit seems that it fails to find a peak for $B^0 \to \mu\mu$ events at 5.280 GeV. Instead, the chi-squared method finds a peaking background at this point in the spectrum ($B_s^0 \to KK$ where the kaons are misidentified as muons). Given that the spectra of both these events fall mostly in one (and the same) bin, this discrepancy is most likely due to the binned nature of the chi-squared estimation. The likelihood method does find the correct peak (and finds a zero amplitude for $B_s^0 \to KK$, which is consistent with the LHCb analysis [64]), which therefore again seems the more appropriate estimation method if statistics are low.

If we compare our likelihood fit with the fit from the LHCb analysis in Figure 9.1 (which is also a likelihood fit), we find, most importantly, that there is relatively good agreement between the amplitudes found for the true signal events (i.e. actual $B_{(s)}^0 \to \mu\mu$ events). This is especially true for the $B_s^0 \to \mu\mu$ peak, for which we find an amplitude that is approximately equal to the amplitude found by the LHCb analysis; for the $B^0 \to \mu\mu$ peak, however, we find an amplitude that is 35–40 percent too high.[4]

Regarding the backgrounds, we find approximately the same $B^0 \to K\pi$ background. However, we do not find the partial reconstructed backgrounds for $B^0 \to \pi\mu\nu_\mu$ (misidentified pion and missing neutrino), $B_s^0 \to K\mu\nu_\mu$ (misidentified kaon and missing neutrino), and $B^0 \to \pi\mu\mu$ (missing pion). Moreover, we find amplitudes for the $\Lambda_b^0 \to p\mu\nu_\mu$ and $B_c^+ \to J/\psi\mu\nu_\mu$ backgrounds that are too high (although these amplitudes fit the data well). The found combinatorial background is of the correct amplitude, but does have a slightly incorrect slope.

Although the general agreement is already quite good, especially given the simplicity of our simulation model, these discrepancies imply that there is still room for improving the fitting procedure. A possible extension may be to impose physics-based constraints on the background amplitudes. For example, if the relative amplitudes of different backgrounds are approximately known, or if we have knowledge on the expected amplitude of some background (both these things are generally calculated in advance), it is likely that the fit will be improved considerably.

[4]Specifically, we find approximately 20.7 $B_s^0 \to \mu\mu$ events (of 139 candidate events) and 6.0 $B^0 \to \mu\mu$ events. Unfortunately, we have not yet implemented a way to consistently estimate the errors on these coefficients.

Chapter 10

Conclusion

In this thesis we have developed a framework for simulating experimental mass spectra in high-energy physics experiments. Contrary to existing simulation software, our framework has focused primarily on computational speed, rather than on extreme accuracy. A sufficient level of accuracy has been ensured, however, by including those elements that are identified to be the most essential for determining the shapes of these mass spectra. In particular, in addition to decay kinematics, the model includes the most important detection effects (acceptance, resolution and bremsstrahlung effects), and can simulate the effects of particle misidentifications or of the partial reconstruction of events.

Those elements that are deemed to have only marginal effects (e.g. detector misalignments) have been ignored. As a result, we have been able to reduce the simulation model to only a handful of essential elements, thus obtaining a model that is fast and relatively simple. As such, the model is ideal for gaining quick insights into the effects of these essential elements, as long as extreme accuracy is not required. For example, it becomes relatively straightforward to examine how an increase in the momentum resolution would affect certain mass spectra; how a mass spectrum would be affected by imposing certain momentum cuts; or how a decrease in detector material would affect the bremsstrahlung tail of a spectrum. Existing simulation software may be unnecessarily complex for addressing such questions, and, more importantly, may be prohibitively slow.

The framework is completed by also providing a method to optimize the parameters of the simulation model with respect to the characteristics of an experimental setup. As a result, and given that the model is designed for generic collider experiments (although the model has been developed in the context of the LHCb experiment), it can in principle be readily adapted to any collider experiment.

In addition, we have demonstrated an application of the simulation model, by using it for fitting experimental spectra. Given the relative simplicity of the model, we have found the model to work surprisingly well, illustrating that our approach of prioritizing speed over accuracy can indeed be useful in practice. Nonetheless, there is considerable room for improvement of the fitting procedure if physics-based constraints (such as ratios between background amplitudes, if these are known) can be included.

However, the simplifications made when developing the model inherently introduce several inaccuracies and shortcomings. For instance, the mass spectrum shape depends relatively strongly

on the energy distribution of the primary particle. The model currently provides no solution for generating such distributions. Instead, such distributions must be put in by hand, i.e. defined by the user. To circumvent this issue, it would e.g. interesting to investigate whether it is feasible to pair our model with event generation software like EvtGen. Although this would decrease the speed of our model slightly (remember from Chapter 3 that the event generators are still relatively fast—the speed issue that motivates our research lies mainly with Geant), such a pairing may potentially improve the accuracy of our model considerably.

Another drawback is that many parameters are currently particle-independent. In reality, though, this may not be the case. For instance, it is not unlikely that the momentum resolution parameters differ across particles. Moreover, all detection efficiencies are assumed to be constant (but particle-dependent), whereas these may be dependent on e.g. momentum or production angle. Now, these features are relatively straightforward to implement, but do require knowledge of additional parameters. These parameters should then either be measured, or one can possibly obtain these parameters by calibrating the model to some dataset.

Finally, an important shortcoming is that bremsstrahlung, and especially bremsstrahlung photon recovery, is still poorly understood. While we have been able to develop a procedure for the fast simulation of total bremsstrahlung energy loss, we do not yet fully understand the effects of bremsstrahlung on the mass spectrum shape. Moreover, it is not yet researched how the spectrum shapes are affected by bremsstrahlung recovery. Before the model can be used for applications where electrons are studied, it is necessary that these issues are further investigated.

All in all, these shortcomings aside, we may then conclude that a fast model for simulating mass spectra can indeed be sufficiently accurate, and therefore useful. In developing such a model, promising first steps have been taken that already give relatively good results. However, there are still several elements that could refine the quality of the calculated spectra, without the need to make significant sacrifices in terms of computational speed. Further research should therefore focus on developing and implementing such improvements.

Once these extensions have been included, it would be interesting to see how well the model performs in real analyses; for example, the model may then be readily applicable to recent experiments regarding e.g. Ω_c^0 decays (via the $\Omega_c^0 \to \Xi_c^+ K^-$ channel [111]), CP violation in Λ_b^0 decays (the comparison of $\Lambda_b^0 \to p\pi^-\pi^+\pi^-$ versus $\Lambda_b^0 \to \pi^- K^+ K^-$ decays [112]), or lepton universality in B^0 decays (the comparison of $B^0 \to K^{*0}\ell\ell$ branching fractions [113]). In such experiments, where backgrounds are abundant, a fast simulation framework for predicting mass spectrum shapes for both signal and background event spectra would certainly be a valuable addition to the toolkit of the experimental physicist.

Acknowledgements

I would first and foremost like to express my gratitude to my supervisor, Gerco Onderwater—not only for all his help and valuable feedback throughout the year, but also especially for his encouragements and enthusiasm.

In addition, I am thankful for the help from the LHCb/Bfys group at Nikhef, Amsterdam. They provided me with the datasets that I used for my research, were available for help when necessary, had the patience to listen to my presentations and provided useful feedback.

Bibliography

[1] E. Roulet, CERN-2003-003 (2007).

[2] F.R. Klinkhamer, Mod. Phys. Lett. A 28, 1350010 (2013).

[3] V.A. Mitsou, Int. J. Mod. Phys. A 28, 31 (2013).

[4] G.F. Giudice, in *Perspectives on LHC physics*, edited by G. Kane and A. Pierce (World Scientific Publishing, 2008).

[5] J.C. Pati and A. Salam, Phys. Rev. D 10, 1 (1974).

[6] H. Georgi and S.L. Glashow, Phys. Rev. Lett. 32, 8 (1974).

[7] R.D. Peccei and H.R. Quinn, Phys. Rev. Lett. 28, 25 (1997).

[8] P. Fayet, Phys. Lett. 64B, 2 (1976).

[9] P. Fayet and S. Ferrara, Phys. Reports 32, 5 (1977).

[10] S. Dimopoulos and H. Georgi, Nucl. Phys. B 193, 1 (1981).

[11] B. Schrempp and F. Schrempp, Phys. Lett. 153B, 1–2 (1985).

[12] J. Wudka, Phys. Lett. 167B, 3 (1986).

[13] S.P. Martin, *A supersymmetry primer* (unpublished). [arXiv:hep-ph/9709356v7]

[14] J.D. Lykken, CERN-2010-002 (2010).

[15] K. Freese, in *Proceedings of 14th Marcel Grossman Meeting, MG14, University of Rome, Rome, July 2015* (2017).

[16] B. Schellekens, *Beyond the Standard Model* (unpublished). [nikhef.nl/∼t58/BSM.pdf]

[17] Y. Kuno and Y. Okada, Rev. Mod. Phys. 73, 1 (2001).

[18] J. Ellis, Nature 448, 7151 (2007).

[19] A. de Gouvêa and N. Saoulidou, Annu. Rev. Nucl. Part. Sci. 60, 1 (2010).

[20] A. de Gouvêa and P. Vogel, Prog. Part. Nucl. Phys. 71, 1 (2013).

[21] R.H. Bernstein and P.S. Cooper, Phys. Reports 7, 2 (2013).

[22] R. Aaij *et al.* (The LHCb Collaboration), Phys. Rev. Lett. 117, 26 (2016).

[23] F. Kahlhoefer, Int. J. Mod. Phys. A 32, 1730006 (2017).

[24] T. Gershon and V.V. Gligorov, Rep. Prog. Phys. 80, 046201 (2017)

[25] T. Sjöstrand, S. Mrenna and P. Skands, Comp. Phys. Comm. 178, 11 (2008).

[26] D.J. Lange, Nucl. Instrum. Methods A 462, 1–2 (2001).

[27] S. Agostinelli *et al.*, Nucl. Instrum. Methods A 506, 3 (2003).

[28] The LHCb Collaboration, *The Boole project*, lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/boole/, retrieved June 19, 2017.

[29] The LHCb Collaboration, *LHCb starter kit*, lhcb.github.io/first-analysis-steps/, retrieved June 6, 2017.

[30] A. Bettini, *Introduction to elementary particle physics* (Cambridge University Press, Cambridge, 2008).

[31] M.E. Peskin and D.V. Schroeder, *Introduction to quantum field theory* (Westview Press, 1995).

[32] G. Martinus, J. Rijfkogel, D. Emes, M. de Roo and D. Roest, *Elementary particle physics* (unpublished).

[33] A. Das and T. Ferbel, *Introduction to nuclear and particle physics*, 2nd ed. (World Scientific Publishing, Singapore, 1994).

[34] B.R. Martin, *Nuclear and particle physics* (John Wiley & Sons, Chichester, 2006).

[35] D. Galbraith and C. Burgard, texample.net/tikz/examples/model-physics/, retrieved May 25, 2017.

[36] C. Patrignani *et al.* (Particle Data Group), Chin. Phys. C 40, 100001 (2016).

[37] A. Quadt, Eur. Phys. J. C 48, 1 (2006).

[38] G. Martinez, in *Proceedings of the 2011 Joliot Curie School September 12-17th 2011, La Colle sur Loup, France* (2013).

[39] K. Aamodt *et al.* (The ALICE Collaboration), J. Instrum. 3, S08002 (2008).

[40] H. Song, S.A. Bass, U. Heinz, T. Hirano and C. Shen, Phys. Rev. Lett. 106, 192301 (2011).

[41] S.F. Novaes, in *Particle and Fields, Proceedings of the X.J.A. Swieca Summer School* (World Scientific, Singapore, 2000).

[42] R. Aaij *et al.* (The LHCb Collaboration), Phys. Rev. D 95, 012002 (2017).

[43] R. Aaij *et al.* (The LHCb Collaboration), Phys. Rev. Lett. 115, 072001 (2015).

[44] B.R. Webber, Int. J. Mod. Phys. A 15, 1 (2000).

[45] F. Wilczek, Rev. Mod. Phys. 71, 2 (1999).

[46] U. Mosel, *Fields, symmetries and quarks* (Springer, Berlin, 1999).

[47] V. Crede and W. Roberts, Rep. Prog. Phys. 76, 076301 (2013).

[48] J.D. Jackson and D.R. Tovey, *Kinematics*, in Chin. Phys. C 40, 100001 (2016).

[49] J.J. Sakurai and J. Napolitano, *Modern quantum mechanics* (Addison-Wesley, San Francisco, 1994).

[50] J.W. Rohlf, *Modern physics from α to* Z^0 (Wiley-VCH, 1994).

[51] A. Alves *et al.* (The LHCb Collaboration), J. Instrum. 3, S08005 (2008).

[52] The LHCb Collaboration, lhcb.web.cern.ch/lhcb/speakersbureau/html/bb_ProductionAngles.html, retrieved June 5, 2017.

[53] T. Head, J. Instrum. 9, C09015 (2014).

[54] B. Sciascia, LHCb-PROC-2016-020 (2016).

[55] R. Aaij *et al.* (The LHCb Collaboration), J. High Energy Phys. 5, 159 (2013).

[56] R. Aaij *et al.* (The LHCb Collaboration), Phys. Rev. Lett. 113, 15 (2014).

[57] R. Aaij *et al.* (The LHCb Collaboration), J. High Energy Phys. 4, 64 (2015).

[58] M. Needham, LHCb-NOTE-2001-102 (2001).

[59] R. Aaij *et al.* (The LHCb Collaboration), J. High Energy Phys. 2, 106 (2013).

[60] J.H. Friedman, Ann. Stat. 29, 5 (2001).

[61] S. Radhakrishna, kgpdag.wordpress.com, retrieved June 6, 2017.

[62] T. Hastie, R. Tibshirani and J.H. Friedman, *The elements of statistical learning* (Springer, Stanford, 2008).

[63] B.P. Roe, H.J. Yang, J. Zhu, Y. Liu, I. Stancu and G. McGregor, Nucl. Instrum. Methods A 543, 2 (2005).

[64] M. Mulder, LHCb-PROC-2017-014 (2017).

[65] The LHCb Collaboration, *The Brunel project*, lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/brunel/, retrieved June 19, 2017.

[66] The LHCb Collaboration, *The DaVinci project*, lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/davinci/, retrieved June 19, 2017.

[67] I. Antcheva *et al.*, Comp. Phys. Comm. 180, 6 (2011).

[68] The LHCb Collaboration, *The Gauss project*, lhcb-release-area.web.cern.ch/LHCb-release-area/DOC/gauss/, retrieved June 19, 2017.

[69] A. Ryd *et al.*, EvtGen V00-11-06 (2004).

[70] J. Apostolakis *et al.*, Rad. Phys. Chem. 78, 10 (2009).

[71] J.G. Körner, in *Proceedings of the Helmholtz International School: Physics of Heavy Quarks and Hadrons* (2014).

[72] M. van Veghel, LHCb internal note (in preparation).

[73] F. James, CERN-68-15 (1968).

[74] R.E. Mitchell *et al.* (The CLEO Collaboration), Phys. Rev. D 79, 072008 (2009).

[75] D. Martínez Santos and F. Dupertuis, Nucl. Instrum. Methods A 764, 1 (2014).

[76] H. Bichsel, D.E. Groom and S.R. Klein, *Passage of particles through matter*, in Chin. Phys. C 40, 100001 (2016).

[77] D.E. Groom, N.V. Mokhov and S.I. Striganov, Atomic Data and Nucl. Data Tables 78, 2 (2001).

[78] H.W. Koch and J.W. Motz, Rev. Mod. Phys. 31, 4 (1959).

[79] Y.S. Tsai, Rev. Mod. Phys. 46, 4 (1974).

[80] S.M. Seltzer and M.J. Berger, Nucl. Instrum. Methods B 12, 1 (1985).

[81] M.L. Perl, in *Proceedings of 8th Les Rencontres de Physique de la Vallee d'Aoste: Results and Perspectives in Particle Physics La Thuile, Italy, March 6-12, 1994* (1994).

[82] H. Davies, H.A. Bethe and L.C. Maximon, Phys. Rev. Lett. 93, 4 (1954).

[83] S.M. Seltzer and M.J. Berger, Atomic Data and Nucl. Data Tables 35, 3 (1986).

[84] A.B. Migdal, Phys. Rev. 103, 6 (1956).

[85] M.L. Ter-Mikaelian, Zu Eksper. Teor. Fiz. 25, 1 (1954).

[86] T. Stanev, Ch. Vankov, R.E. Streitmatter, R.W. Ellsworth and T. Bowen, Phys. Rev. D 25, 5 (1982).

[87] P.L. Anthony *et al.*, Phys. Rev. Lett. 76, 19 (1996).

[88] A. Schälicke, V. Ivanchenko, M. Maire and L. Urban, in *IEEE Nuclear Science Symposium Conference Record* (2008).

[89] I.J. Feng, R.H. Pratt and H.K. Tseng, Phys. Rev. A 24, 3 (1981).

[90] L. Kin, R.H. Pratt, S.M. Seltzer and M.J. Berger, Phys. Rev. A 33, 5 (1986).

[91] W. Riegler, CERN-2014-001 (2014).

[92] H.A. Bethe and W. Heitler, in *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* (1934).

[93] A. Strandlie, *Lecture notes: experimental high energy physics* (unpublished). [folk.uio.no/ares/FYS4550/Lectures_H14_2.pdf]

[94] G. Cowan, *Monte Carlo Techniques*, in Chin. Phys. C 40, 100001 (2016).

[95] E. Bos, Ph.D. thesis, Vrije Universiteit (2010).

[96] M. Needham and T. Ruf, LHCb-NOTE-2007-025 (2007).

[97] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, Science 220, 4598 (1983).

[98] D. Mitra, F. Romeo and A. Sangiovanni-Vincentelli, in *Proceedings of 24th IEEE Conference on Decision and Control* (1985).

[99] R. Neal, *Probabilistic inference using Markov Chain Monte Carlo methods* (unpublished).

[100] A.W. van der Vaart, *Asymptotic statistics* (Cambridge University Press, Cambridge, 1998).

[101] F. James, *Statistical methods in experimental physics* (World Scientific, Singapore, 2006).

[102] G. Bohm and G. Zech, *Introduction to statistics and data analysis for physicists* (unpublished). [doi:10.3204/DESY-BOOK/statistics]

[103] T.W. Anderson, Ann. Math. Stat. 33, 3 (1962).

[104] A. Pettitt, Biometrika 63, 1 (1976).

[105] R. Aaij *et al.* (The LHCb Collaboration), Int. J. Mod. Phys. A 30, 1530022 (2015).

[106] The CMS and LHCb Collaborations, Nature 544, 7554 (2015).

[107] R. Aaij *et al.* (The LHCb Collaboration), Phys. Lett. B 754, 1 (2016).

[108] F.M. Dekking, C. Kraaikamp, H.P. Lopuhaä and L.E. Meester, *A modern introduction to probability and statistics* (Springer, London, 2005).

[109] I. van Vulpen, Bfys presentation 24 February 2017, Nikhef, Amsterdam (unpublished). [https://www.nikhef.nl/∼ivov/Statistics/PoissonError/2017_02_24_PoissonError_LHCb_IvovanVulpen.pdf]

[110] F. James, CERN Program Library Long Writeup D506 (1994).

[111] R. Aaij *et al.* (The LHCb Collaboration), Phys. Rev. Lett. 118, 18 (2017).

[112] R. Aaij *et al.* (The LHCb Collaboration), Nature Phys. 13, 1 (2017).

[113] R. Aaij *et al.* (The LHCb Collaboration), LHCb-PAPER-2017-013 (2017).

Appendices

Appendix A

C++ code

A.1 Program layout

The class structure of the simulation framework is shown in Figure A.1. The classes in parentheses are auxiliary classes; these are in principle never used directly by the user.

In addition, we have designed classes for particles (`Particle` and `ParticlePDG`), for calibration (`Annealing`), and for fitting (`SpectrumFit`). These can all be found in the following sections.

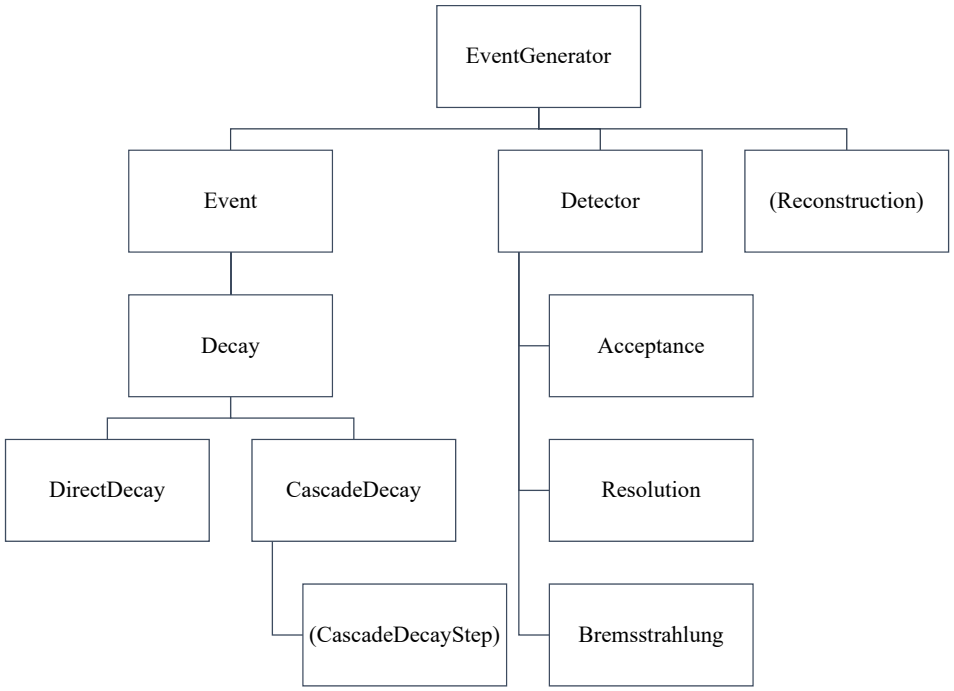

Figure A.1
The class structure of the simulation framework.

A.2 Particles

```
class Particle {
   /*
      Class for describing particles
   */

   friend class ParticlePDG;
            // ParticlePDG inherits from Particle
   friend std::ostream &::operator<<(std::ostream &output, Particle const &particle);
      // implementation in particle.cc

   int d_code;
   double d_m;
   double d_p;
   double d_theta;
   double d_phi;

   public:
      Particle();
      Particle(double m, double p = 0, double theta = 0, double phi = 0, int code = 0);
      Particle(Particle const &other);
      ~Particle() = default;

      Particle &operator=(Particle const &rhs);

      Particle(TLorentzVector const &lv);
      Particle &operator=(TLorentzVector const &rhs);
      TLorentzVector LorentzVector() const;

         int id() const;
        void id(int code);
        void id(PDGCode code);
      TParticlePDG *PDG_entry();

      double m() const; // GeV
        void m(double new_m);

      double E() const;
        void E(double new_E);
      double p() const; // GeV
        void p(double new_p);
      double pT() const; // GeV
        void pT(double new_p);
      double p_x() const;
      double p_y() const;
      double p_z() const;
      double theta() const; // rad
        void theta(double new_theta);
      double phi() const;
        void phi(double new_phi);
      double pseudorapidity() const;
      double gamma() const;
        void gamma(double new_gamma);
      double v() const; // velocity / c
        void v(double new_v);
      double T() const; // kinematic energy
        void T(double new_T);
};
```

```
class ParticlePDG: public Particle {
   /*
      Particle class with the extension that it is coupled
      to a TDatabase entry, so that PDG data are available
      to the object
   */

   TParticlePDG *d_pdg;

   public:
      ParticlePDG();
      ParticlePDG(int   code, double p = 0, double theta = 0, double phi = 0);
      ParticlePDG(PDGCode code, double p = 0, double theta = 0, double phi = 0);
      ParticlePDG(ParticlePDG const &other); // copy constructor
      ~ParticlePDG() = default;

      ParticlePDG &operator=(ParticlePDG const &rhs);
      ParticlePDG &operator=(TLorentzVector const &rhs);

      ParticlePDG(int   code, TLorentzVector const &lv);
      ParticlePDG(PDGCode code, TLorentzVector const &lv);

         int id() const;
        void id(int code);
        void id(PDGCode code);

      double m() const; // GeV
      double width() const;
      double Stable() const;

};
```

A.3 Generating two-body decays

```
class TwoDecayGenerator {
    /*
        Class that simulates two-body decays, given a primary particle
    */

   friend std::ostream &::operator<<(std::ostream &output, TwoDecayGenerator const &tdg);

    ParticlePDG d_primary;
    ParticlePDG d_secondary[2];
    TRandom3 rand_engine;

    public:
        TwoDecayGenerator();
        TwoDecayGenerator(TwoDecayGenerator const &other);

        ParticlePDG *GenerateDecay();
       std::ostream &GenerateDecay(std::ostream &outstream, size_t iterations = 1);
              void  GenerateDecay(std::string const &outpath, size_t iterations = 1);

```

```
        ParticlePDG *Primary();
              void  Primary(ParticlePDG const &primary);

        ParticlePDG *Secondary(size_t index);
              void  Secondary(size_t index,  int const code);
              void  Secondary(size_t index, PDGCode const code);

    protected:
        double p(double E, double m);
        double Atan(double num, double den);
        double mod2pi(double angle);
};
```

```
ParticlePDG *TwoDecayGenerator::GenerateDecay() {

    // Draw a mass for the primary particle if it has a BW line shape
    double M;
    if (d_primary.width() <= 0)
        M = d_primary.m();
    else
        while ((M = rand_engine.BreitWigner(d_primary.m(), d_primary.width())))
            if (M < d_primary.E() && M > 0)
                break;

    // Draw a uniform angle for one of the particles
    double cos_theta = rand_engine.Uniform(2.) - 1.; // [-1, 1]
    double sin_theta = std::sqrt(1. - cos_theta * cos_theta);
    double phi     = rand_engine.Uniform(2. * M_PI) - M_PI;

    for (size_t i = 0; i < 2; ++i) {
        double E_star, p_star;

        // Calculate energy and momentum in COM frame
        E_star = (1/(2*M)) * (M*M + d_secondary[ i].m() * d_secondary[ i].m()
                              - d_secondary[1-i].m() * d_secondary[1-i].m());
        p_star = p(E_star, d_secondary[i].m());

        // Assign boosted energy and momentum to secondary particles
        d_secondary[i].E( d_primary.gamma() * (E_star
                                + d_primary.v() * p_star * cos_theta) );
        d_secondary[i].theta( mod2pi(Atan(p_star * sin_theta ,
                                          (d_primary.gamma() * (p_star * cos_theta
                                           + d_primary.v() * E_star)))
                                         + d_primary.theta())
                                    );
        d_secondary[i].phi(phi); // Fix to xz-plane
        cos_theta *= -1.;        // Back-to-back decay (in COM)
    }

    return d_secondary;
}
```

A.4 Generating N-body decays

```
class Decay {
   /*
      Abstract base class for the simulation of decays.
      Used in DirectDecay and CascadeDecay classes.
   */

   friend std::ostream &::operator<<(std::ostream &output, Decay const &decay);

   protected:
      size_t const   d_N;
       ParticlePDG    d_primary;
       TLorentzVector d_primary_LV;
       ParticlePDG   *d_secondary;
       double        *d_secondary_masses;

   public:
      Decay(size_t N);
      Decay(Decay const &other);
      virtual ~Decay();

          size_t N() const;

        ParticlePDG *Primary();
              void Primary(ParticlePDG const &primary);

        ParticlePDG *Secondary(size_t index = 0);
              void Secondary(size_t index,  int const code);
              void Secondary(size_t index, PDGCode const code);

        virtual double GenerateDecay() = 0;
};
```

A.4.1 Generating direct decays

```
class DirectDecay :
    public Decay
{
   /*
      Class for generating direct N-body decays,
      i.e. Primary --> 1 + 2 + ... + N
   */

    TRandom3 rand;
    TGenPhaseSpace EventGenerator;

    public:
        DirectDecay(size_t N);
        DirectDecay(DirectDecay const &other);
        ~DirectDecay() = default;

        double GenerateDecay() override;
};
```

```
double DirectDecay::GenerateDecay() {

```

```
    // Draw a mass for the primary particle if it has non-zero
    // decay width
    double M;
    if (d_primary.width() == 0)
        M = d_primary.m();
    else
        while ((M = rand.Gaus(d_primary.m(), d_primary.width())))
            if (M < d_primary.E())
                break;

    // Set the mass of the primary particle to the mass drawn
    // above. (This does not change the momentum components)
    d_primary_LV.SetE( std::sqrt(d_primary.p() * d_primary.p() + M * M) );

    // Initialize the decay with the TGenPhaseSpace object
    // and generate an event
    EventGenerator.SetDecay(d_primary_LV, d_N, d_secondary_masses);
    double weight = EventGenerator.Generate();

    // Save the generated secondary particles (in ParticlePDG format)
    for (size_t i = 0; i < d_N; ++i) {
        TLorentzVector lv = *(EventGenerator.GetDecay(i));
        d_secondary[i] = lv;
    }

    return weight;
}
```

A.4.2 Generating cascade decays

```
class CascadeDecay:
   public Decay
{
   /*
      Class for simulating cascade decays, by treating them as
      sequences of direct N-body decays
   */

   friend CascadeDecayStep;

   TLorentzVector *d_secondary_LV;

   size_t d_NSteps;
   std::vector<CascadeDecayStep *> d_steps;

   public:
      CascadeDecay(size_t NFinalParticles);
      ~CascadeDecay();

        void AddStep(size_t primary_index,
                std::initializer_list<size_t> secondary_index,
                std::initializer_list<int> secondary_code);

        double GenerateDecay() override;
};
```

```
class CascadeDecayStep {
   /*
      One step of a cascade decay
   */

   CascadeDecay *d_cascade;
   size_t const d_NSecondaries;

   TLorentzVector *d_primary;  // pointer to the particle that should decay
                        // i.e. the intermediate particle
                        // - possibly points to an earlier created secondary
   ParticlePDG *d_secondary_copy; // contains info about secondaries to be produced
                        // in this decay step
   TLorentzVector **d_secondary_target; // where the produced particles should be saved

   TGenPhaseSpace EventGenerator;
   TRandom3 rand;

   public:
      CascadeDecayStep(CascadeDecay *cascade,
                size_t primary_index,
                std::initializer_list<size_t> secondary_index,
                std::initializer_list<int> secondary_code);
      ~CascadeDecayStep();
      double DoDecayStep();
};
```

```
double CascadeDecay::GenerateDecay() {
   d_secondary_LV[0] = d_primary.LorentzVector();
      // the primary particle is always initialized at index 0

   double weight = 1.; // it is assumed that the weights of different
                // decay steps multiply

   try {
      for (size_t i = 0; i < d_NSteps; ++i)
         weight *= d_steps[i]->DoDecayStep();
   } catch (CascadeDecayStepNotKinematicallyAllowed) {
      // retry if secondary masses were drawn that are not
      // kinematically allowed
      return GenerateDecay();
   }

   for (size_t i = 0; i < d_N; ++i)
      d_secondary[i] = d_secondary_LV[i];

   return weight;
}
```

```
double CascadeDecayStep::DoDecayStep() {

   // Generate masses for the particles to be produced in this step
   double masses[d_NSecondaries];
   for (size_t i = 0; i < d_NSecondaries; ++i)
      if (d_secondary_copy[i].width() <= 0)
         masses[i] = d_secondary_copy[i].m();
```

```
      else
         do {
            masses[i] = rand.BreitWigner(d_secondary_copy[i].m(),
                                         d_secondary_copy[i].width());
         } while (masses[i] < 0.);

   // Check if the decay is kinematically allowed, given the masses
   // of the secondary particles. If not, throw an error for the
   // CascadeDecay class to catch, which will then retry with different
   // masses for the secondary particles.
   bool kin_allowed = EventGenerator.SetDecay(*d_primary, d_NSecondaries, masses);
   if (!kin_allowed)
      throw CascadeDecayStepNotKinematicallyAllowed();

   // The actual decay
   double weight = EventGenerator.Generate();

   for (size_t i = 0; i < d_NSecondaries; ++i)
      *d_secondary_target[i] = *(EventGenerator.GetDecay(i));

   return weight;

}
```

A.5 Simulating the detector

```
class Detector {
   /*
      Class for simulating detector behaviour and detection effects:
      - acceptance effects
      - resolution (uncertainty) effects
      - bremsstrahlung
   */

   friend std::ostream &::operator<<(std::ostream &outstream, Detector &det);

   public:

      static Detector DetectorLHCb();

      enum class Parameter {
         E,
         p,
         pT,
         theta,
         phi,
         m
      };
      struct Acceptance {
         Parameter type;
              bool set = false;
            double min;
            double max;

         Acceptance()            = default;
         Acceptance(Acceptance const &) = default;
```

```
    };
    struct Uncertainty {
       Parameter type;
       double (*uncertainty_func)(double without_uncertainty, double *p);
       double *paramvals;

       Uncertainty(Parameter init_type, double (*init_func)(double, double *),
          double *init_paramvals = nullptr)
       :
          type(init_type),
          uncertainty_func(init_func),
          paramvals(init_paramvals)
       { }
       Uncertainty(Uncertainty const &other) :
          type(other.type),
          uncertainty_func(other.uncertainty_func),
          paramvals(nullptr)
       { }
       Uncertainty() = default;
    };

  private:
    std::vector<Acceptance> d_acceptance;
    std::vector<Uncertainty> d_uncertainty;
    std::vector<Bremsstrahlung *> d_brems;

    std::vector<std::string> d_acceptance_desc;
    std::vector<std::string> d_uncertainty_desc;
    std::vector<std::string> d_brems_desc;

    TRandom3 d_rand;

  public:
    Detector() = default;
    Detector(Detector const &other);
    ~Detector();

    Detector &operator=(Detector const &rhs);

    bool Accept(size_t NParticles,
                               ParticlePDG *particles,
                               Particle *signal);

    void GenerateUncertainty(size_t NParticles,
                               ParticlePDG *particles,
                               Particle *signal);

    void GenerateBremsstrahlung(size_t NParticles,
                               ParticlePDG *particles,
                               Particle *signal);


    void  AddAcceptance(Parameter type, double min, double max,
                    std::string const &description = "");
    void  AddUncertainty(Uncertainty const &uncertainty,
                    std::string const &description = "");
    void  AddUncertainty(Parameter init_type, double (*init_func)(double, double *),
                    double *init_paramvals,
                    std::string const &description = "");
    void  AddUncertainty(Parameter init_type, double (*init_func)(double, double *),
                    std::string const &description = "");
```

```
92
93      void  AddBremsstrahlung(Bremsstrahlung const &brems,
94                      std::string const &description = "");
95      Bremsstrahlung *GetBremsstrahlung(size_t const index);
96
97      int Detect(size_t NParticles, ParticlePDG *particles, Particle *signal);
98      int Detect(Event *event);
99
100   private:
101      double ParameterValue(Particle *particle, Parameter type);
102      void SetParameterValue(Particle *particle, Parameter type, double val);
103 };
```

```
1  int Detector::Detect(size_t NParticles, ParticlePDG *particles, Particle *signal) {
2     for (size_t i = 0; i < NParticles; ++i)
3        signal[i] = particles[i];
4
5     GenerateBremsstrahlung(NParticles, particles, signal);
6
7     if (Accept(NParticles, particles, signal)) {
8
9        GenerateUncertainty(NParticles, particles, signal);
10
11    } else {
12       return 1; // msg: not within acceptance
13    }
14
15    return 0;
16 }
```

```
1  bool Detector::Accept(size_t NParticles,
2                        ParticlePDG *particles,
3                        Particle *signal)
4  {
5     bool accept_decision = true;
6     for (size_t i = 0; i < NParticles; ++i) {
7        // for (Acceptance i_acceptance : d_acceptance) {
8        for (std::vector<Acceptance>::iterator i_acceptance = d_acceptance.begin();
9           i_acceptance != d_acceptance.end(); ++i_acceptance)
10       {
11          if (!i_acceptance->set)
12             continue;
13          accept_decision *= (
14                  ParameterValue(particles + i, i_acceptance->type)
15                                          >= i_acceptance->min
16                && ParameterValue(particles + i, i_acceptance->type)
17                                          <= i_acceptance->max
18              );
19       }
20    }
21    return accept_decision;
22 }
```

```
1  void Detector::GenerateUncertainty(size_t NParticles,
2                                     ParticlePDG *particles,
```

```
                                Particle *signal) {
  for (size_t i = 0; i < NParticles; ++i) {
    for (std::vector<Uncertainty>::iterator i_uncertainty = d_uncertainty.begin();
      i_uncertainty != d_uncertainty.end(); ++i_uncertainty)
    {
      SetParameterValue(
        signal + i,
        i_uncertainty->type,
        i_uncertainty->uncertainty_func(
          ParameterValue(signal + i, i_uncertainty->type),
          i_uncertainty->paramvals
        )
      );
    }
  }
}
```

```
void Detector::GenerateBremsstrahlung(size_t NParticles,
                                ParticlePDG *particles,
                                Particle *signal) {
  for (size_t i = 0; i < NParticles; ++i) {
    if (particles[i].id() == 11 || particles[i].id() == -11) {
      for (std::vector<Bremsstrahlung *>::iterator i_brems = d_brems.begin();
        i_brems != d_brems.end(); ++i_brems)
      {
        (*i_brems)->Generate(signal + i);
      }
    }
  }
}
```

A.6 Simulating bremsstrahlung

```
class Bremsstrahlung {
  /*
    Base class for simulating bremsstrahlung according to different models:
    - BremsstrahlungMicro:  physics-based (microscopic) model
    - BremsstrahlungMacroBH: statistics-based (macroscopic) Bethe-Heitler model
    - BremsstrahlungMacro:  corrected Bethe-Heitler model
    - BremsstrahlungMicroNoLPM: same as BremsstrahlungMicro but without LPM effect
  */

  friend std::ostream &::operator<<(std::ostream &os, Bremsstrahlung const &bs);

  public:
     // PHYSICAL CONSTANTS
     static constexpr double pi = 3.14159265358979323846264338327950288419716939937510582;
     static constexpr double hbar = 6.582119e-25; // [GeV s]
     static constexpr double c = 3e10;           // [cm/s]
     static constexpr double alpha = 1. / 137.; // [1]
     static constexpr double N_A = 6.022140857e23; // [1]
     static constexpr double r_e = 2.81794032e-13; // [cm]
     static constexpr double ln2 = 0.693147180559945309417232121458;

```

```
  protected:
      // RADIATION LENGTH AND AUX FUNCTIONS
      static double X0(double Z, double A);  // radiation length
      static double X0_prime(double Z, double A); // radiation length (small correction)
      static double f(double Z);           // Coulomb correction
      static double F_el(double Z);        // screening correction (nucleus)
      static double F_inel(double Z);      // screening correction (electrons)

      // BREMSSTRAHLUNG LPM AND TM EFFECTS
      static double kplasma(double Z, double A, double rho); // plasma momentum
      static double kLPM( double Z, double A, double rho); // LPM momentum

      // OTHER (AUXILIARY) FUNCTIONS
       static double *FindLargestBelowValue(double val, double *arr, size_t arr_len);

   //=========================================================

  protected:
      // MATERIAL CHARACTERISTICS
      // set by user
      size_t const d_Z;       // [1]
      double const d_A;       // [g/mol]
      double const d_massdensity; // [g/cm3]

      double const d_X0;       // [g/cm2]
      double const d_X0_prime; // [g/cm2]
      double d_thickness;      // [g/cm2]

      // set/used by class
      double d_kplasma; // multiply by gamma
      double d_kLPM;   // multiply by gamma^2
      TRandom3 rand;

   //=========================================================

   public:

     Bremsstrahlung(size_t const Z, double const A,
                    double const massdensity, double const thickness);
     Bremsstrahlung(); // sets silicon with thickness = 1 * X0
     Bremsstrahlung(Bremsstrahlung const &other)    = delete;
     Bremsstrahlung &operator=(Bremsstrahlung const& other)  = delete;
     virtual ~Bremsstrahlung() = 0 ;

     virtual Bremsstrahlung *Clone() const = 0;

            void Tau(double tau);
     virtual void Thickness(double thickness);
     virtual void Generate(Particle *particle) = 0;

     double RadiationLength() const;
     double X0() const;
     double NAtoms() const;
     double Thickness() const;

};
```

A.6.1 Physics-based bremsstrahlung

```
class BremsstrahlungMicro:
  public Bremsstrahlung
{
   /*
      Physics-based microscopic bremsstrahlung simulation, including
      LPM and Ter-Mikaelian effects. Generates bremsstrahlung energy
      loss by simulating the collisions as an electron traverses a
      material
   */

  static double s1(double Z);   // LPM variable s1
   static double phi(double s); // LPM function phi
   static double psi(double s); // LPM function psi

  double d_s1;  // set by s1(Z)

   double const d_F_el;
   double const d_F_inel;
   double const d_f;

  public:
    BremsstrahlungMicro(size_t const Z, double const A,
                        double const massdensity, double const thickness);
      ~BremsstrahlungMicro() override;

      BremsstrahlungMicro(BremsstrahlungMicro const &other);
      virtual BremsstrahlungMicro *Clone() const override;

    void Thickness(double thickness) override;
    void Generate(Particle *particle) override;

      double TotalCrossSection(Particle *particle, size_t const Npoints = 101) const;
             // numerically integrates DifferentialCrossSection()
    double DifferentialCrossSection(double k, Particle *particle) const;

  private:
      double PathLength(Particle *particle); // draws a path length between collisions
      double MeanFreePath(Particle *particle, size_t const Npoints = 2001);
             // calculate the mean free path for a particle (requires num. integration)
      double FractionalLossDraw(Particle *particle, size_t const Nbins = 2048);
             // draw a fractional loss using the differential cross section as pdf
             // (this is a computationally expensive step)

      // Ter-Mikaelian effect
      double TM_suppression(double k, Particle *particle) const;

    // LPM effect
      double sLPM(double k, double T, double gamma) const;
      double sLPM_prime(double k, double T, double gamma) const;
      double sLPM_hat(double k, double T, double gamma) const;
      double xi(double s) const;
      double xi_prime(double sLPM_prime) const;
      // in addition: the static functions s1, phi, psi

};
```

```
void BremsstrahlungMicro::Generate(Particle *particle) {
  double x = 0;
```

```
   while (true) {
      x += PathLength(particle); // draw a path length before a new collision
      if (x < d_thickness / d_massdensity) // check if material is escaped
         particle->T( particle->T() * FractionalLossDraw(particle) );
      else
         return;
   }
}
```

```
/*
   A preset for the LHCb detector
*/

TRandom3 DetectorLHCbMomentumUncertaintyRandom;
double DetectorLHCbMomentumUncertainty(double real_p, double *p = nullptr) {
   double a = 0.00294632, b = 7.8905e-05;
   return DetectorLHCbMomentumUncertaintyRandom.Gaus(
      real_p, a * real_p + b * real_p * real_p);
}

Detector Detector::DetectorLHCb() {
   Detector detector;

   BremsstrahlungMacro brems(14, 28.0855, 2.33, 4.5);
   brems.Tau(0.30);
   detector.AddBremsstrahlung(brems);

   detector.AddAcceptance(Parameter::theta, -0.27, 0.27, "|theta|<0.27rad");
   detector.AddAcceptance(Parameter::p,  1.0, 1e6 , "|p|>1GeV");
   detector.AddUncertainty(Parameter::p,
      &DetectorLHCbMomentumUncertainty, "{#Delta}p/p = a + bp");

   return detector;
}
```

A.6.2 The original Bethe-Heitler straggling distribution

```
class BremsstrahlungMacroBH:
   public Bremsstrahlung
{
   /*
      Statistics-based macroscopic simulation of bremsstrahlung energy loss,
      based on the Bethe-Heitler straggling distribution.
   */

   public:
      BremsstrahlungMacroBH(size_t const Z, double const A,
                            double const massdensity, double const thickness);
      ~BremsstrahlungMacroBH() override;

      BremsstrahlungMacroBH(BremsstrahlungMacroBH const &other);
      virtual BremsstrahlungMacroBH *Clone() const override;

      void Thickness(double thickness) override;
      void Generate(Particle *particle) override;
```

```
19
20      double ProbNoCollision(Particle *particle) const;
21
22    private:
23        double GammaDraw(double const k, double const lambda = 1.);
24            // draws from a gamma distribution [Gamma(k, lambda)]
25        double FractionalLossDraw();
26
27 };
```

```
1  void BremsstrahlungMacroBH::Generate(Particle *particle) {
2     particle->T(
3        particle->T() * FractionalLossDraw()
4     );
5  }
```

```
1  double BremsstrahlungMacroBH::FractionalLossDraw() {
2     // [http://folk.uio.no/ares/FYS4550/Lectures_H14_2.pdf]
3     // Draw a number from a Gamma distribution and transform it
4     // to obtain the Bethe-Heitler straggling distribution
5     return exp(- GammaDraw( (d_thickness / d_X0) / ln2, 1. ));
6  }
```

A.6.3 The corrected Bethe-Heitler straggling distribution

```
1  class BremsstrahlungMacro:
2    public Bremsstrahlung
3  {
4    /*
5       Statistics-based macroscopic bremsstrahlung simulation according to
6       the corrected Bethe-Heitler straggling distribution. Corrections
7       included are
8       - non-zero probability of no energy loss (i.e. zero collisions)
9       - correction factor for the body of the distribution (z < 1)
10      - cut-off of the divergence at z -> 1
11   */
12
13   static double constexpr exp1 = 2.71828182845904523536028747135 27;
14
15   // for the correction factors
16   static double constexpr CFPar[28] = {
17      -0.6013280359,
18      24.44678439,
19      -7.541210632,
20      -1.240833341,
21      11.24895438,
22      -12.78589071,
23      -0.566796301,
24      -0.06730110736,
25      0.07679095859,
26      -0.09840681114,
27      41.79257084,
28      -10.65990833,
```

```
    -0.8901364263,
    16.62387151,
    -28.28740161,
    0.009298503728,
    -0.2590012305,
    -0.001083091556,
    0.005370860486,
    -0.1650103521,
    0.003757009009,
    0.442290456,
    -0.3430730913,
    3.974151109,
    -0.8260459178,
    -8.556720771,
    13.70980602,
    -0.2325500516
  };

  static double constexpr CFNoLossPar[5] = {
    -10.58628634,
    0.003984238263,
    -2.582939419e-06,
    6.400747252e-10,
    0.006236080973
  };

  static double constexpr CFCutoffPar[3] = {
    0.000481060519091,
    -2.00223676531,
    0.018354967156
  };

  // make code more readable and save repeated computations:
  double d_tau;      // relative thickness = thickness / X0
  double d_tau2;     // d_tau squared

  public:
    BremsstrahlungMacro(size_t const Z, double const A,
                        double const massdensity, double const thickness);
    BremsstrahlungMacro();
    ~BremsstrahlungMacro() override;

    BremsstrahlungMacro(BremsstrahlungMacro const &other);
    virtual BremsstrahlungMacro *Clone() const override;

    void Generate(Particle *particle) override;
    void Thickness(double thickness) override;

    double BetheHeithlerPDF(double *z, double *p = nullptr) const;
    double BetheHeithlerPDF(double z) const;

  private:
      double GammaDraw(double const k, double const lambda = 1.);
      double FractionalLossDraw(); // draws from original Bethe-Heitler distribution

      // Corrections:
      double CorrectionFactor(double z, double E) const;
      double ProbNoEnergyLoss(double E) const;
      double RightCutoff(double E) const;

};
```

```
void BremsstrahlungMacro::Generate(Particle *particle) {
    if (rand.Uniform(1.) < ProbNoEnergyLoss(particle->E()))
        // No collisions, z = 1
        return;
    else {
        // z < 1

        // z cannot exceed the cut-off
        double z_cutoff = RightCutoff(particle->E());

        double z;
        do {
            z = FractionalLossDraw();
        } while (
                z > z_cutoff
             || rand.Uniform(1.) * 2.5 > CorrectionFactor(z, particle->E())
            );
        // CorrectionFactor is implemented in terms of accept-reject
        // For tau < 2.5 and E < 2000 the correction factor does not exceed 2.5
        // Also, it is generally around 1. Therefore, using an accept-reject
        // algorithm to implement the correction factor, where the "box" has a
        // height of 2.5 is reasonably efficient, although more sophisticated
        // methods would definitely lead to a more efficient algorithm.
        // (This approach is taken simply because it is quick to implement)

        particle->T( particle->T() * z );
    }
}
```

A.7 Reconstructing primary particles

```
class Reconstruction {

  public:
     Reconstruction() = default;

       static Particle Reconstruct(size_t NParticles, Particle *secondary);
       static Particle Reconstruct(size_t NParticles, ParticlePDG *secondary);

};
```

```
Particle Reconstruction::Reconstruct(size_t NParticles, Particle *secondary) {
    /*
       Return a particle with four-momentum equal to the sum
       of the four-momenta of the secondary particles
    */

    double E = 0,
           px = 0,
           py = 0,
           pz = 0;

    for (size_t i = 0; i < NParticles; ++i) {
        E  += secondary[i].E();
```

```
      px += secondary[i].p_x();
      py += secondary[i].p_y();
      pz += secondary[i].p_z();
   }

   double p2   = px*px + py*py + pz*pz;
   double p    = std::sqrt( p2 );
   double theta = std::acos( pz / p );
   double phi  = std::atan2( py , px );

   double m = std::sqrt( E*E - p2 );

   return Particle(m, p, theta, phi);
}
```

A.8 Defining events with misidentified and missing particles

```
class Event {
  /*
     Class containing the entire specification of an event:
     - The specification of the decay (Decay *d_decay)
     - Whether particles are misidentified
     - Whether particles are missing (i.e. not detected)
  */

  size_t const d_N;

   Decay *d_decay;
     int *d_misid;
    bool *d_missing;

  Particle *d_signal;

  public:
     Event(size_t N);
     Event(Decay *decay);
     ~Event();

     size_t N() const;
     Particle *Signal(size_t index = 0);

     Decay *GetDecay();
      void  SetDecay(Decay *decay);

     void SetMisID(size_t index,  int misid);
     void SetMisID(size_t index, PDGCode misid);
     void UnsetMisID();
     void UnsetMisID(size_t index);

     void SetMissing(size_t index, bool missing = true);
     void UnsetMissing();
     void UnsetMissing(size_t index);

     void ApplyMisID();
     void ApplyMissing();

```

```
40      double GenerateDecay(); // calls d_decay->GenerateDecay()
41  };
```

A.8.1 Misidentifications

```
1  void Event::ApplyMisID() {
2    for (size_t i = 0; i < d_N; ++i) {
3       if (d_misid[i] != 0)
4          d_signal[i].m( TDatabasePDG::Instance()->GetParticle(d_misid[i])->Mass() );
5          // d_misid contains PDG codes for the observed identities
6          // (d_signal originally has masses corresponding to the true decay)
7    }
8  }
```

A.8.2 Missing particles

```
1  void Event::ApplyMissing() {
2    for (size_t i = 0; i < d_N; ++i) {
3       if (d_missing[i])
4          d_signal[i] = Particle(0., 0.);
5          // Replace the signal with a particle with zero mass and momentum
6    }
7  }
```

A.9 Generating events

```
1  class EventGenerator {
2    /*
3       Class for generating events. It is mostly convenient for
4       doing related bookkeeping---it does not simulate any physics
5       itself, but only through its data members (d_detector) and
6       arguments to function calls (Generate(Event *, ...)).
7    */
8
9    Detector d_detector;
10
11   TH1D *d_hist;
12
13   TH2D  *d_dalitz;
14   size_t d_dalitz_axes[4];
15
16   public:
17      EventGenerator();
18      EventGenerator(Detector const &detector);
19
20      void SetHistogram(TH1D *h);
21      void UnsetHistogram();
22
23      void SetDalitz(TH2D *h, size_t i_x1, size_t i_x2, size_t i_y1, size_t i_y2);
```

```
    void UnsetDalitz();

    Detector *GetDetector();

    double Generate(Event *event, size_t iterations = 1);
    double Generate(Event *event, size_t iterations, TH1D *h);
    double Generate(Decay *decay, size_t iterations = 1);
    double Generate(Decay *decay, size_t iterations, TH1D *h);
};
```

```
double EventGenerator::Generate(Event *event, size_t iterations) {
  double event_weight;

  double sum_weight = 0;
  TH1D *tmphist;
  if (d_hist != nullptr)
    tmphist = new TH1D(*d_hist);

  for (size_t iteration = 0; iteration < iterations; ++iteration) {

    event_weight = event->GetDecay()->GenerateDecay();

    if ( d_detector.Detect(event) == 0 ) {
      // event accepted by Detector

      event->ApplyMisID();
      event->ApplyMissing();

      if (d_hist != nullptr) {
        // Do tasks if a histogram is to be filled
        tmphist->Fill( event->ReconstructedMass() , event_weight );
        sum_weight += event_weight;
      }

      if (d_dalitz != nullptr) {
        // Do tasks if a Dalitz plot is to be filled
        TLorentzVector lv[4];
        for (size_t i = 0; i < 4; ++i)
          lv[i] = (event->Signal() + d_dalitz_axes[i])->LorentzVector();
        TLorentzVector lv_x = lv[0] + lv[1],
                       lv_y = lv[2] + lv[3];
        d_dalitz->Fill( lv_x.M2(), lv_y.M2(), event_weight );
      }

    }

  }
  if (d_hist != nullptr) {
    // This is the purpose of tmphist: for N-body decays with N > 2,
    // event_weights are smaller than 1. But for consistency with other
    // parts of the program, we want this function to always add exactly
    // n=iterations to the histogram, and not the sum of weights which
    // is n'<iterations.
    *d_hist = *d_hist + ((double)iterations / sum_weight * *tmphist);
    delete tmphist;
  }


  return ( d_detector.Detect(event) == 0 ) ? event_weight : 0.;
```

```
    // This is important if single events are generated; then you
    // want to be able to check whether the particle has been
    // accepted and all effects have been applied.
}
```

A.10 Simulated annealing

```
class Annealing {
  friend std::ostream &operator<<(std::ostream &out, Annealing &annealing);

  size_t const d_Nparameters;
  size_t d_Ndata;
  size_t d_Nsample;

    ECDF *d_data;
    ECDF *d_sample;

  double d_Tstart;
  size_t d_Titerations;
  size_t d_iterations;
  size_t d_total_iterations;

    bool d_Ndata_set;
    bool d_Nsample_set;
    bool d_iterations_set;

    bool d_initialized;

  double *d_candidates;
  double *d_statistics;

  double *d_best_candidate;
  double d_best_statistic;

  protected:
    TRandom3 rand_engine;

  protected:
    Annealing(size_t const Nparameters);
    Annealing(Annealing const &other) = delete;
    Annealing &operator=(Annealing const &rhs) = delete;

  public:
    ~Annealing();

    size_t NParameters() const;                  // inline
    size_t NData() const;                        // inline
      void NSample(size_t const Nsample);
    size_t NSample() const;                      // inline

      void Data(size_t const Ndata, double const *data);
    double *Data();                              // inline
    double *Sample();                           // inline

    double InitialValues(double *const initialvalues);
    double *LastCandidate();                     // inline
```

```
50      double LastStatistic() const;                   // inline
51      double *BestCandidate();                      // inline
52      double BestStatistic() const;                   // inline
53
54      size_t IterationsPerT() const;                  // inline
55      double TStart() const;                        // inline
56      double TStop();
57      size_t TIterations() const;                   // inline
58      size_t TotalIterations() const;               // inline
59
60        void SetIterations(double const Tstart,
61                           double const iterationsofT,
62                           size_t const iterationsperT);
63
64        void Anneal();
65        void GenerateSampleCDF(size_t const iteration);
66
67      TGraph GraphDataCDF( int linestyle = 1, int linecolor = 1, int linewidth = 1) const;
68      TGraph GraphSampleCDF(int linestyle = 1, int linecolor = 1, int linewidth = 1) const;
69      TGraph GraphStatistic(size_t const i_stat, double const xmin,
70                            double const xmax, size_t const Npoints);
71        void Plot(std::string const &filename, size_t N_candidates = 0) const;
72        TH1D CandidateDistribution(size_t candidate, size_t N_candidates = 0) const;
73      TGraph TracePlot(size_t candidate) const;
74
75      static double AndersonDarlingLmax(size_t n, size_t m);
76      static double CramerVonMisesLmax(size_t n, size_t m);
77
78
79    protected:
80      virtual void GenerateCandidates(size_t const iteration, double const T) = 0;
81      virtual void GenerateSample(size_t const iteration)        = 0;
82      virtual void UpdateTemperature(double &T)                  = 0;
83
84      double *Candidates(size_t const iteration = 0);    // inline
85      double &Statistics(size_t const iteration = 0);    // inline
86
87
88    private:
89      double Statistic();
90      double StatisticAreaTails(double const x0,      // inline
91                                double const x1,
92                                double const xi,
93                                double const dx,
94                                double const N);
95      double StatisticAreaCore( double const x0,      // inline
96                                double const x1,
97                                double const xiR,
98                                double const xiS,
99                                double const dxR,
100                               double const dxS,
101                               double const NR,
102                               double const NS);
103     double AcceptanceProbability(size_t const iteration, double const T);
104       void RejectCandidate(size_t const iteration);
105       void SetBestCandidate(size_t const iteration);
106       void ReduceDataToSampleSize();
107
108     double LastCandidatesMean(size_t const parameter, size_t N_candidates) const;
109     double LastCandidatesSD(size_t const parameter, size_t N_candidates) const;
110
```

```
};
```

```
void Annealing::Anneal() {
   if (!d_Ndata_set || !d_Nsample_set || !d_iterations_set) {
      std::cout << "Annealing::Anneal - Not initialized:\n";
      if (!d_Ndata_set)
         std::cout << " Data not set      - Use Data(size_t const Ndata, double const *data)\n";
      if (!d_Nsample_set)
         std::cout << " Sample size not set - Use NSample(size_t const Nsample)\n";
      if (!d_iterations_set)
         std::cout << " Iterations not set - Use SetIterations(double const Tstart,\n"
                   << "                                       double const iterationsofT,\n"
                   << "                                       size_t const iterationsperT)\n";
      return;
   }
   d_initialized = true;

   double T = d_Tstart;

   for (size_t T_iter = 0, idx = 1; T_iter < d_Titerations; ++T_iter) {

      for (size_t iter = 0; iter < d_iterations; ++iter) {

         GenerateCandidates(idx, T); // implemented by user
         GenerateSampleCDF(idx);  // implemented by user

         d_statistics[idx] = Statistic();

         if (AcceptanceProbability(idx, T) < rand_engine.Uniform(0,1))
            RejectCandidate(idx);
         else if (d_statistics[idx] < d_best_statistic) {
            SetBestCandidate(idx);
         }

         ++idx;
      }

      if (idx % 1000 == 0) {
         // Progress report: Print current parameter values
         std::cout << "(" << T_iter + 1 << "/" << d_Titerations << ")\tT = " << T << '\t';
         for (size_t j = 0; j < d_Nparameters; ++j)
            std::cout << d_candidates[d_Nparameters * (idx - 1) + j] << '\t';
         std::cout << d_statistics[idx - 1] << '\n';
      }

      UpdateTemperature(T); // implemented by user
   }

   // Print final parameters and best fit
   std::cout << "BEST FIT: Statistic = " << d_best_statistic << '\n';
   for (size_t k = 0; k < d_Nparameters; ++k)
       std::cout << "        Param. " << k << " = " << d_best_candidate[k] << '\n';
}
```

```
double Annealing::AcceptanceProbability(size_t const iteration, double const T) {
   return std::exp((d_statistics[iteration - 1] - d_statistics[iteration]) / T);
}
```

```
double Annealing::Statistic() {
   return ECDF::AndersonDarling(*d_data, *d_sample);
}
```

A.10.1 Empirical cumulative distribution functions

```
struct ECDF {
   size_t N;
   double *cdf_x;
   double *cdf_y;

   ECDF(size_t ECDF_N);
   ~ECDF();

   static size_t SimilarY(ECDF &ecdf1, ECDF &ecdf2);
   static double CramerVonMises(ECDF const &dist1, ECDF const &dist2);
   static double CramerVonMisesMax(double N, double M);
   static double CramerVonMisesMax(ECDF const &dist1, ECDF const &dist2);
   static double AndersonDarling(ECDF const &dist1, ECDF const &dist2);
   static double AndersonDarlingElement(size_t &rank, size_t &i, double N, double M);
   static double AndersonDarlingMax(size_t N, size_t M);
   static double AndersonDarlingMax(ECDF const &ecdf1, ECDF const &ecdf2);

   void sortx();
   void makey();
   TGraph graph(int linestyle = 1, int linecolor = 1, int linewidth = 1) const;

   double operator()(double const x) const;

   void FillNormal(double mu, double sigma, TRandom3 &rand);
};
```

```
double ECDF::AndersonDarling(ECDF const &dist1, ECDF const &dist2) {
   double A = 0;
   size_t i = 0, j = 0, rank = 1;
   size_t const &N = dist1.N;
   size_t const &M = dist2.N;
   while (i < N && j < M) {
      if (dist1.cdf_x[i] <= dist2.cdf_x[j])
         A += AndersonDarlingElement(rank, i, N, M);
      else
         A += AndersonDarlingElement(rank, j, M, N);
   }
   if (i == N) {
      while (j < M - 1)
         A += AndersonDarlingElement(rank, j, M, N);
   }
   if (j == M) {
      while (i < N - 1)
         A += AndersonDarlingElement(rank, i, N, M);
   }
   return A;
}

double ECDF::AndersonDarlingElement(size_t &rank, size_t &i, double N, double M) {
```

```
  ++i;
  double dA = rank - (N + M) / N * i;
  double A = N / M * dA * dA / (double)(rank * (N + M - rank));
  ++rank;
  return A;
}
```

A.11 Fitting experimental spectra

```
class SpectrumFit {
  /*
    Class that fits experimental mass spectra, given
    - an EventGenerator object (including a Detector)
    - Event objects (including e.g. misID and partial reconstr.)
    Can fit both with Chi-squared (if given a histogram) or with
    Likelihood (if given an array of observations).
  */

  enum FitMode {
    UNINITIALIZED,
    CHISQUARED,
    LIKELIHOOD
  };

  static SpectrumFit *ActiveSpectrumFitObject;
  static void MinuitFCN_ChiSquared(int &, double *, double &, double *, int);
  static void MinuitFCN_Likelihood(int &, double *, double &, double *, int);

  // Event Generator
  EventGenerator *d_eg;

  // Spectrum
  TH1D *d_template_hist;
  TF1  *d_combinatorial;

  FitMode d_fitmode;
    TH1D  *d_spectrum;
  double *d_datapoints;
  size_t  d_Ndatapoints;

  // Events
  size_t d_NEvents;
  std::vector<Event *>   d_events;
  std::vector<TH1D *>  d_event_hists;
  std::vector<double>    d_event_hists_probability_in_window;
  std::vector<TGraph *>  d_event_graphs;
  std::vector<std::string> d_event_descriptions;

  // Fitting
  bool  d_initialized;
  TMinuit *d_minuit;

  public:
    SpectrumFit(TH1D     *template_hist);
    SpectrumFit(TH1D const &template_hist);
    ~SpectrumFit();
```

```
48
49      SpectrumFit *SetCombinatorial(TF1 *combinatorial);
50      SpectrumFit *AddCombinatorial(TF1 *combinatorial); //alias for Set
51      TF1        *GetCombinatorial();
52
53      SpectrumFit   *SetEventGenerator(EventGenerator *eg);
54      EventGenerator *GetEventGenerator();
55
56      SpectrumFit *AddEvent(Event *event,
57                            std::string const &description = "",
58                            size_t iterations = 1e5);
59      SpectrumFit *AddEvent(Event *event,
60                            size_t iterations,
61                            std::string const &description = "");
62      Event      *GetEvent(size_t index = 0);
63      TH1D       *GetEventHistogram(size_t index = 0);
64
65      TMinuit    *GetMinuit();
66      TMinuit    *Fit(TH1D *spectrum);
67      TMinuit    *Fit(size_t N, double *datapoints);
68
69      double      GetParameter(size_t index) const;
70      double      GetParError(size_t index) const;
71
72      TCanvas    *Plot(std::string const &draw_options = "", TH1D *spectrum = nullptr);
73      TCanvas    *Plot(TH1D *spectrum, std::string const &draw_options = "");
74
75      SpectrumFit *Print(std::ostream &os = std::cout);
76
77    private:
78      void InitializeFit(FitMode fitmode);
79      void UninitializeFit();
80
81      void MinuitFixAllExcept(size_t index);
82      void MinuitFixAll();
83      void MinuitReleaseAll();
84
85      void FitCombinatorialOnly();
86      void FitRoutine(FitMode fitmode);
87      double CurrentPDFIntegral(double *par);
88      double CurrentPDFIntegral();
89
90      size_t NParameters() const;
91
92  };
```

```
1   SpectrumFit *SpectrumFit::AddEvent(Event *event,
2                                      std::string const &description,
3                                      size_t iterations)
4   {
5     UninitializeFit();
6
7     TH1D *hist = new TH1D(*d_template_hist);
8     for (size_t i = 0; i <= hist->GetNbinsX(); ++i)
9       hist->SetBinContent(i, 0);
10
11    d_eg->Generate(event, iterations, hist);
12
13    if (hist->Integral() > 0) {
```

```
    double probability_in_window = hist->Integral() / (double)iterations;
    hist->Scale( (double)hist->GetNbinsX()
            / (double)hist->Integral()
            / (hist->GetXaxis()->GetXmax() - hist->GetXaxis()->GetXmin()));
    d_event_descriptions.push_back(description);
    d_event_hists.push_back(hist);
    d_event_hists_probability_in_window.push_back(probability_in_window);
    d_event_graphs.push_back(new TGraph(hist));
    d_events.push_back(event);
    ++d_NEvents;
  } else {
    std::cerr << "ERROR: Event \'" << description << '\''
           << " falls outside spectrum window -> Ignored!\n";
  }

  return this;
}
```

```
void SpectrumFit::MinuitFCN_ChiSquared(int &npar, double *gin, double &f, double *par, int iflag) {
  // Note: MinuitFCN_ChiSquared(..) is static, and requires use of ActiveSpectrumFitObject
  // Calculate chi-squared of fit
  double chisq = 0;

  for (size_t i = 1; i <= ActiveSpectrumFitObject->d_spectrum->GetNbinsX(); ++i) {
    double d = ActiveSpectrumFitObject->d_spectrum->GetBinContent(i);
    for (size_t j = 0; j < ActiveSpectrumFitObject->d_NEvents; ++j)
      d -= par[j] * ActiveSpectrumFitObject->d_event_hists[j]->GetBinContent(i);

    if (ActiveSpectrumFitObject->d_combinatorial != nullptr) {
      ActiveSpectrumFitObject->d_combinatorial->SetParameters(
        par + ActiveSpectrumFitObject->d_NEvents);
      d -= ActiveSpectrumFitObject->d_combinatorial->Eval(
                ActiveSpectrumFitObject->d_spectrum->GetBinCenter(i)
              );
    }
    chisq += d * d;
  }

  f = chisq;
}
```

```
void SpectrumFit::MinuitFCN_Likelihood(int &npar, double *gin, double &f, double *par, int iflag) {
  // Note: MinuitFCN_Likelihood(..) is static, and requires use of ActiveSpectrumFitObject
  // Calculate log likelihood of fit

  // Update the combinatorial
  if (ActiveSpectrumFitObject->d_combinatorial != nullptr)
    ActiveSpectrumFitObject->d_combinatorial->SetParameters(
      par + ActiveSpectrumFitObject->d_NEvents);

  double normalization = 1. / ActiveSpectrumFitObject->CurrentPDFIntegral(par);
  double ll = 0;

  for (size_t i = 0; i < ActiveSpectrumFitObject->d_Ndatapoints; ++i) {
    double x = ActiveSpectrumFitObject->d_datapoints[i];
```

```
15      double pdf = 0;
16      for (size_t j = 0; j < ActiveSpectrumFitObject->d_NEvents; ++j)
17         pdf += par[j] * ActiveSpectrumFitObject->d_event_graphs[j]
18                       ->Eval(x, nullptr, "S");
19
20      if (ActiveSpectrumFitObject->d_combinatorial != nullptr)
21         pdf += ActiveSpectrumFitObject->d_combinatorial->Eval(x);
22
23      pdf *= normalization;
24
25      ll += (pdf > 0) ? std::log(pdf) : -1e10;
26   }
27
28   f = - 2. * ll;
29 }
```