Monte Carlo Simulations for Particle Physics

Victor Manuel Granados Pinto

August 2024

Abstract

In this project, a Monte Carlo event generator for the simulation of the process $e^+e^- \to q\bar{q}$ has been developed. In the first section, the integration of the cross-section for the fixed-order description of the $2 \to 2$ scattering process at leading order in the Matrix element is discussed. Then, in section 2, the scattering process is supplemented with an all-order simulation of the cascade of bremsstrahlung emitted by the quark-antiquark pair.

Contents

1 Simulation of the $e^+e^- \to q\bar{q}$ scattering process at fixed-order QED

For the first part of this project, a Monte Carlo integrator was developed to evaluate the cross-section for the process $e^-e^+ \to q\bar{q}$ to leading-order. In this section, after a brief introduction to the relevant properties of the differential cross-section to be integrated, the general idea for the Monte Carlo integrator here developed is discussed. Then, some results for the values of the cross-section and their associated error estimates are presented. Finally, as an example of more advanced implementations of Monte Carlo integrators, the `vegas` package is used to integrate the differential cross-section, for which the results and its algorithm are commented on.

1.1 The differential cross-section at leading order

At leading order, the matrix element for the process $e^-e^+ \to q\bar{q}$ consists of two s-channel contributions which differ by having either a photon or a Z boson as the "intermediary" particle, as shown in the Feynman diagram in Figure 1.

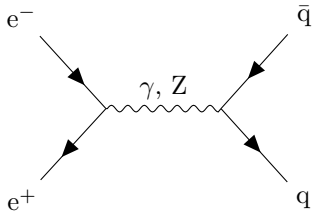


Figure 1: An s-channel process mediated by either a photon or a Z boson.

The resulting squared matrix element is then given by

$$\begin{aligned} |\mathcal{M}_{q\bar{q}}|^2 = (4\pi\alpha)^2 N_C \big[&\left(1+\cos^2\theta\right) \left\{ Q_e^2 Q_q^2 + 2Q_e Q_q V_e V_q \chi_1(s) + \left(A_e^2+V_e^2\right)\left(A_q^2+V_q^2\right)\chi_2(s) \right\} \\ &+ \cos\theta \;\; \left\{ 4Q_e Q_q A_e A_q \chi_1(s) + 8 A_e V_e A_q V_q \chi_2(s) \right\} \big]; \end{aligned} \tag{1}$$

$$\chi_1(s) = \kappa \frac{s(s-M_Z^2)}{(s-M_Z^2)^2+\Gamma_Z^2 M_Z^2}, \qquad \chi_2(s) = \kappa^2 \frac{s^2}{(s-M_Z^2)^2+\Gamma_Z^2 M_Z^2}; \tag{2}$$

and depends on three kinematic phase-space quantities:

i) s, the squared total energy in the centre-of-mass;

ii) $\cos\theta$, the cosine of the scattering angle θ between the incoming positron and the outgoing quark, and

iii) ϕ, the azimuthal angle of the outgoing quark in the plane perpendicular to the beam.

The meaning and values of the constants in expressions (1) and (2) above are not relevant for the moment but they are given in Appendix A. Instead, the following properties are important:

- There is no dependence on ϕ.

- The dependence on $\cos\theta$ comes only in two forms, $(1+\cos^2\theta)$ and $\cos\theta$. When integrating over $-1 < \cos\theta < 1$, the former term yields $8/3$ while the latter is zero. Thus, the interesting part is the dependence on s.
- The dependence on s comes in two forms (besides the trivial one), given by $\chi_1(s)$ and $\chi_2(s)$.

Finally, the differential cross-section to be integrated is

$$\frac{\mathrm{d}\sigma}{\mathrm{d}s\,\mathrm{d}(\cos\theta)\,\mathrm{d}\phi} = \frac{f_{\mathrm{conv}}}{64\pi^2}\frac{f(s)}{s}\sum_{\mathrm{q}=1}^{N_{\mathrm{q}}}|\mathcal{M}_{\mathrm{q\bar{q}}}(s,\cos\theta,\phi)|^2 \tag{3}$$

where $f(s)$ is a normalized distribution that defines the beam spectrum, N_q is the number of light quark flavours that can be produced, and f_{conv} is a conversion factor to give the cross-section in picobarns. Therefore, the task of this part will be to integrate (3) via Monte Carlo integration and, in the process, apply importance sampling for s as a technique for reducing the variance of the estimated value.

1.2 Monte Carlo integration with importance sampling

At a basic level, a Monte Carlo integrator is composed of two ingredients: an integrand and distributions to sample the integration variables. While the integrand for the problem at hand is given by (3), there are three distributions that require to be set: two distributions for the kinematic variables, s and $\cos\theta$ (a distribution for ϕ is unnecessary, since there is no dependence on it), and the beam distribution $f(s)$. The last one is determined by the specifics of the beam, but the first two require some special consideration, as being able to properly sample the integration space will determine how big the sample size needs to be to compute the integral with a certain accuracy.

A naive first approach could be to sample each variable according to a uniform distribution along its integration interval, a method known as *uniform sampling*. Although easy to implement, this method has the disadvantage of requiring a possibly too large sample size to compute the integrand to a decent accuracy, as it samples regions with small and large contributions to the integral equally. An improvement over this method, could be to allow for specific distributions that attempt to sample regions of the integrand according to their contribution (or importance) to the integral. This method is referred to as *importance sampling*, and it contains uniform sampling as a special case. *For this reason, the integrator here developed was written from the beginning as an importance sampling integrator.*

Importance sampling consists of the following: having an integrand $f(\mathbf{x})$ and a distribution $g(\mathbf{x})$ for the integration variables, N points $\mathbf{x}$ are sampled from $g(\mathbf{x})$, and the estimate of the integral is given by

$$E_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(\mathbf{x}_i)}{g(\mathbf{x}_i)}. \tag{4}$$

The estimated error for the Monte Carlo result is then given by

$$|\delta E_N| = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{f(\mathbf{x}_i)}{g(\mathbf{x}_i)}\right)^2 - E_N^2} \tag{5}$$

and decreases with sampling size as $1/\sqrt{N}$ as long as the sequence of variances for different N is bounded.

Thus, the formulas above establish the sampling method but, alas, one still needs to specify the distribution $g(\mathbf{x})$. As discussed earlier, the most interesting part of the integrand is the dependence on s, so for the remainder of this project, the distribution for $\cos\theta$ was set to a uniform distribution between -1 and $+1$ (excluding the latter), and different distributions for s were chosen, to exemplify the effects of sampling the integrand on the estimation of the integral and its error estimate.

Comments on the implementation

The Monte Carlo integrator developed was written in Python3.10 (but compatibility was checked for 3.8 and 3.9 as well) as the `integrator` module of the python package `simulator` (great names, right?). The main components for the integrator consist of the following:

1. Particle (pseudo-data) classes `Electron`, `LightQuarks` and `ZBoson`, containing the relevant properties of these particles such as charge, weak isospin, etc. These could all have been written as child classes of a parent class `Particle`. Hindsight is 20/20 and time is running out.

2. A function defining the squared matrix element in (1). It takes as input values for s, $\cos\theta$ and quark flavour q. For s and $\cos\theta$ one can pass either single values, single value and array or two arrays of equal length, whereas for q one can pass a single value or an array of equal length to s and $\cos\theta$.

3. A parent class `Distribution`, defined by the methods `sample` and `evaluate_distro`, together with three child classes of it: `Dirac`, `Uniform` and `BreitWigner`. The idea is that the Monte Carlo integrator requires only the two methods in the parent class, so one can define any distribution as a child of `Distribution` and therefore avoid having to modify the integrator to accommodate for other distributions. Additionally, to ensure reproducibility, each `Distribution` child class should allow the setting of a random number generator if necessary, so upon passing the distributions to the Monte Carlo integrator the generation of random numbers is tied to the integrator instead of independently for each distribution.

4. A `MonteCarloIntegrator` class, which can be instanced by passing distributions for s, $\cos\theta$, $f(s)$ (and even ϕ just for the sake of it) and specifying a method to sum over light quark flavours (at the moment the only options are "explicit" and "random"). All of these are optional, defaulting to $f(s) = \delta(s - M_Z^2)$, uniform distributions for $\cos\theta$ and ϕ, and explicitly summing over all quark flavours. Each instance has a default seed number (integer) which can be modified through the keyword `seed`. The squared matrix element is hard-coded within the integrator so that's the only thing it can integrate.

All the code, data and figures, as well as a few examples on how to instantiate the integrator and compute the cross-section, are available at the GitHub repository https://github.com/VictorG20/mc-particle-physics.

1.3 Results

The total cross-section for the squared matrix element in (1) was computed with the Monte Carlo integrator for the following 3 scenarios, which differ on the sampling methods used for s and the choice of beam distribution $f(s)$:

1. Fixed beam energy: $f(s) = \delta(s - M_{\mathrm{Z}}^2)$, where M_{Z} is the mass of the Z boson;
2. Flat beam spectrum and flat s sampling: $f(s)$ is a uniform distribution between $s_{\min} = (M_{\mathrm{Z}} - 3\Gamma_{\mathrm{Z}})^2$ and $s_{\max} = (M_{\mathrm{Z}} + 3\Gamma_{\mathrm{Z}})^2$ and, similarly, s is drawn from such uniform distribution;
3. Flat beam spectrum and Breit-Wigner importance sampling for s: $f(s)$ is the same as in the previous point but now importance sampling is used for s, using a Breit-Wigner distribution

$$g(s) = \frac{M_{\mathrm{Z}}\Gamma_{\mathrm{Z}}}{\rho(s_{\max}) - \rho(s_{\min})} \frac{1}{(s - M_{\mathrm{Z}}^2)^2 + M_{\mathrm{Z}}^2\Gamma_{\mathrm{Z}}^2}, \qquad \rho(s) = \arctan\left(\frac{s - M_{\mathrm{Z}}^2}{M_{\mathrm{Z}}\Gamma_{\mathrm{Z}}}\right). \tag{6}$$

Furthermore, the sum over quark flavours in (3) has been done in two ways: (i) "explicit", meaning that the sum is carried out as implied in the equation, and (ii) "random", consisting on picking a random flavour q for each Monte Carlo point and taking $\sum_{\mathrm{q}} |\mathcal{M}|^2 = N_f \langle |\mathcal{M}|^2 \rangle_{\mathrm{q}}$.

The results for the cross-section for each of the configurations described and both methods of summing over quark flavours are given in Table 1, together with the exact values obtained from solving the integrals analytically. Comparing the columns for both methods of evaluating the quark sum, it is clear that *both methods are statistically equivalent: their values and estimated errors yield intersecting intervals.* The importance of this equivalence lies in the possibility of avoiding the explicit sum in favour of the random method for the purpose of computing the cross-section, as the former requires to compute the squared matrix element for each quark flavour. Additionally, as will be discussed in the next section, it allows to classify events with a specific quark flavor for processes such as showering.

Part b) Answer 1.

		Cross-section (pb)		
s distribution	$f(s)$	Explicit quark sum	Random quark flavour	Exact value
$s = M_{\mathrm{Z}}^2$	$\delta(s - M_{\mathrm{Z}}^2)$	42,216(37)	42,236(40)	42,213
Uniform	Uniform	9,976(39)	9,984(40)	9,929
Breit-Wigner	Uniform	9,932(9)	9,939(10)	9,929

Table 1: Results for the cross-section for different distributions of s and $f(s)$ with a sample size of 100,000. The terms in parentheses are the associated Monte Carlo error estimate.

A comparison of the Monte Carlo error estimate for different sample sizes, for all cases, is shown in Figure 2. Here, *the log-log plot shows a straight-line trend in all cases, confirming the reduction of the error estimate as* $\propto 1/\sqrt{N}$, as mentioned in the previous subsection.

Part b) Answer 2.

In the results of Table 1, the cross-section for case 1 (fixed beam energy at Z boson mass) is significantly larger than that of cases 2 and 3. Mathematically, the fact that the results are different makes sense because the beam distribution $f(s)$ *is part of the integrand*, not a method for sampling

values of the integration variables. Physically, a smaller value for the cross-section when considering a broader range of beam energies reflects that *values of the centre-of-mass energy away from the mass of the* Z *boson lead to a decrease in probability for the interaction to take place*, at least for the two s-channel interactions depicted in Figure 1. This can be confirmed by comparing the two methods: the fixed beam energy method can be used to "scan" several values $s_{\min} \leq s_0 \leq s_{\max}$, obtaining the total cross-section at each of them, while the flat beam spectrum is run once and binned in a histogram, with each sampled point weighted according to its Monte Carlo weight. However, the histogram needs to be divided by the flat beam spectrum distribution *in order to get the histogram bar heights to correspond to cross-section values*. The result is shown in Figure 3, where the (green) dashed line indicates that, indeed, a maximum for the cross-section takes place at $s = M_Z^2$.

Part d) Answer 1.

Part d) Answer 2.

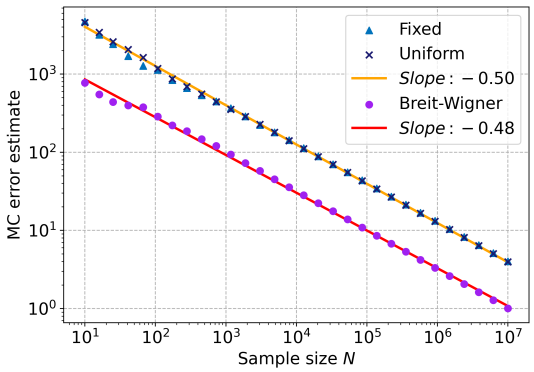


Figure 2: Monte Carlo error estimate of the cross-section vs. sample size for the three different cases of s sampling.

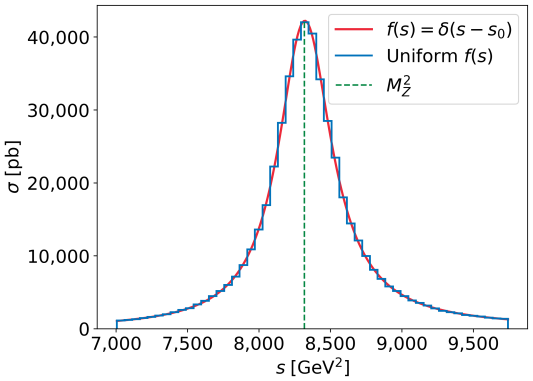


Figure 3: Cross-section values from binning the flat beam spectrum vs. integration at fixed beam energy.

The fact that the values of the cross-section are distributed along s in the shape found in Figure 3, is precisely what motivates sampling s according to the Breit-Wigner distribution with the Z boson as the resonance (well, that and the whole theoretical background, of course), equation (6). This distribution has a global maximum at M_Z^2, decaying to both sides, besides resembling the common factor present in both $\chi_1(s)$ and $\chi_2(s)$, equation (2). Thus, the Breit-Wigner distribution seems to capture well the importance of different s values to the integral for the cross-section, providing a more reasonable distribution for sampling s rather than a uniform distribution. This is indeed the case, as can be seen from the Monte Carlo error estimates in Table 1 and Figure 2, where *the errors are four times smaller than the uniform case for the same sample size.*

Part g)

1.4 Vegas

The improvement in accuracy of Monte Carlo methods when estimating integrals, as a result of using importance sampling with a distribution that captures enough features of the integrand, has led to develop sophisticated algorithms to determine such distributions. An example of these, is the implementation of Monte Carlo integration in the `vegas` package for python. This package implements the VEGAS algorithm, consisting on "scanning" over several passes the integration region and making a histogram that approximates the distribution of the integrand. In particular, the package implements two adaptive strategies: importance sampling and adaptive

stratified sampling. Thus, an integration problem using the `vegas.Integrator`™ is characterized solely by a region of integration and an integrand. After this, a number of evaluations `neval` (how many times to evaluate the integrand) and a number of iterations `nint` (number of times to perform `neval` evaluations of the integrand) yield independent estimations of the integral. A complete discussion of its implementation and use of the package can be found in https://vegas.readthedocs.io/en/latest/tutorial.html.

Using 10 iterations with 1,000 Monte Carlo points each, the differential cross-section in equation (3) was integrated with `vegas`, obtaining the results shown in Table 2. Here, the first two iterations are not to be taken into account, shown by the relatively high values of Q (the p-value of the χ^2) as compared with the rest of the subsequent iterations. This is because during these iterations vegas has not yet been able to fully optimize the sampling for the integrand. However, for every iteration after the second, the error decreases consistently, reaching a weighted average result of 9,930 pb with an error estimate of 14 pb, only 1pb of difference from the exact value of 9,929 pb.

Iteration	Integral	Weighted average	χ^2/ d.o.f.	Q
1	9,720(17)	9,720(17)	0.00	1.00
2	10,030(10)	9,948(87)	2.54	0.11
3	9,653(79)	9,787(59)	4.42	0.01
4	9,952(70)	9,855(45)	4.05	0.01
5	9,933(54)	9,887(35)	3.34	0.01
6	9,933(52)	9,901(29)	2.79	0.02
7	9,878(42)	9,894(24)	2.35	0.03
8	9,899(38)	9,895(20)	2.02	0.05
9	9950(33)	9,910(17)	2.01	0.04
10	9,977(26)	9,930(14)	2.29	0.01

Table 2: Summary of the vegas integration for 10 iterations of 1,000 Monte Carlo points each. The "Integral" and "Weighted average" columns correspond to estimations for the cross-section in picobarns. Q represents the p-value of the χ^2.

The remarkably good estimation for the cross-section with relatively few evaluations is made possible due to the fact that `vegas` has been able to "flatten" the integral, i.e., find a transformation of the integration variables such that their sampling captures better the integrand. This can be seen in Figure 4, which shows the optimal integration grid found by vegas for the variables s and $\cos\theta$ (since there is no dependence on ϕ). The other two integration grids, for the pairs (s, ϕ) and $(\cos\theta, \phi)$, are shown in Appendix B. Although the accumulation of s values close to M_Z^2 was to be expected, *smaller binning for values of* $\cos\theta$ *close to +1 and -1 come as a surprise.* Indeed, this is made clear by plotting the integrand, as shown in Figure 5, where peaks towards these values can be seen, and the peak at $+1$ is larger than that at -1. Part e)

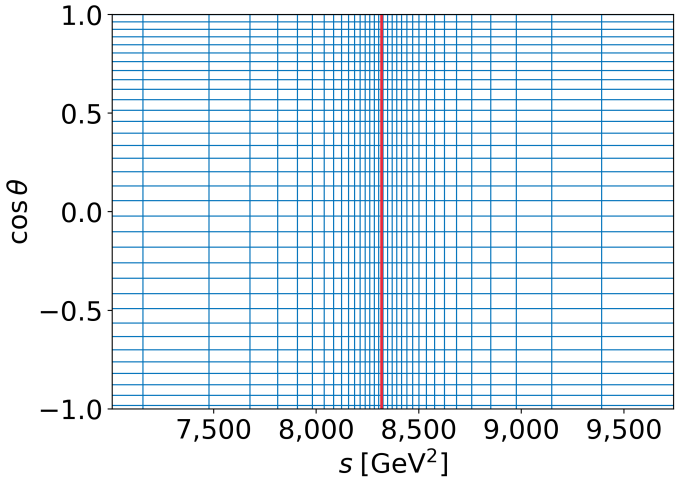


Figure 4: Binning of the integration grid found by `vegas` after 10 iterations of 1,000 evaluations each.

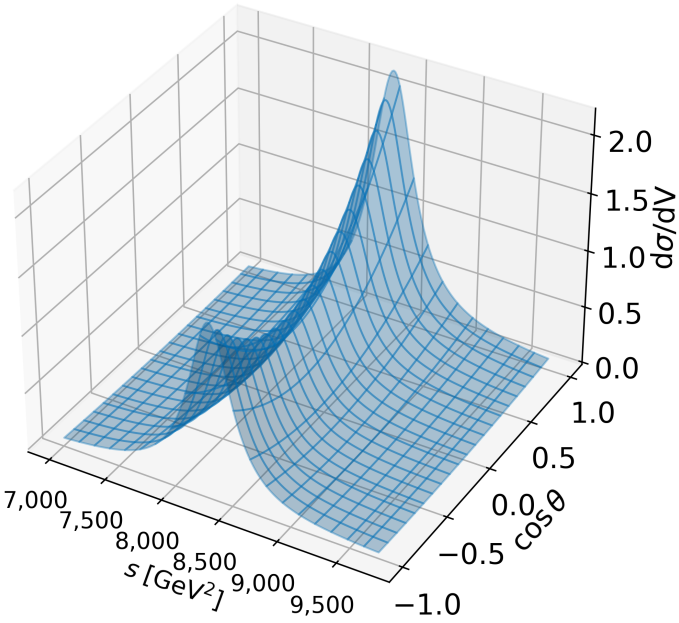


Figure 5: Differential cross-section per volume. Here $\mathrm{d}V = \mathrm{d}s\,\mathrm{d}(\cos\theta)\mathrm{d}\phi$.

2 Simulating the bremsstrahlung cascade at all-order QCD

In the second part of this project, the Monte Carlo integrator from the first part is supplemented with a parton shower, in order to add the effects of additional QCD bremsstrahlung produced by the final-state quark-antiquark pair. After a brief comment on parton showers and the kinematic limit in which it is possible to simulate this cascade at all orders in QCD, the use of the Monte Carlo integrator together with the Parton shower class provided is explained, discussing the change in final-state particles at the end. Finally, jets and jet clustering algorithms are introduced, together with histograms for the differential jet rates produced by the simulation, using as a baseline data produced with SHERPA.

2.1 Parton Shower

In the previous section, the leading-order squared matrix element, equation (1), is quadratic on the QED coupling α. The term "leading-order" implying, in this context, that we are dealing with the smallest possible power in the coupling, thereby neglecting further terms proportional to, e.g. α^4. This coupling is not constant, however, and its value varies with the energy scale. Fortunately, the variation of the QED coupling at the energy scales relevant for the process here studied, is small: at low energy scales one has $\alpha \approx 1/137$, reaching $\alpha \approx 1/127$ at the scale of the Z boson.

Once the quark-antiquark pair is emitted, these accelerated color-charged particles can radiate gluons (*bremsstrahlung*), which in turn can decay emitting more particles, giving rise to a "cascade". This interaction between quarks and gluons is determined by the QCD coupling, α_s, which, unlike the QED one, *increases with decreasing energy scale*, see Figure 6. This means that some care is needed when trying to implement perturbation theory: *if the energy scales become too small, the terms in the perturbative series cease to be smaller with increasing order.*

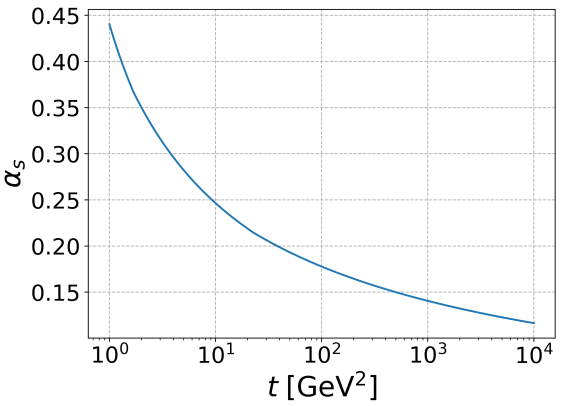


Figure 6: QCD coupling as a function of the squared energy scale.

One way to still apply perturbation theory and include this bremsstrahlung in the simulation, is to study this cascade in the kinematic limit where the emitted parton (i.e. a gluon or quark) is either collinear with its mother parton and/or soft, in other words, either the emission angle is small or the daughter's energy is small. In such case, the cross-section factorises into a part without the emission (the 2 -> 2 process of the previous section) and a universal emission factor, the latter enabling to treat subsequent emissions as a Markov Chain process, the simulation of which is called a "parton shower". Furthermore, the restriction to small emission angles for the daughter particles gives rise to cone-shaped cascades, commonly referred to as *partonic jets*.

2.2 Monte Carlo integrator with parton shower

Although the Monte Carlo integrator was developed from scratch, a `Shower` python class was provided for this project, the task being simply to "connect it" to the integrator.

An instance of `Shower` requires, for initialization, an instance of the class `AlphaS` and a lower cut-off scale t_0 (in GeV^2). The former is used to keep track of the "running" of the coupling α_s at different energy scales, while the latter represents a squared energy scale at which the emissions will be stopped when running the shower. Having created a single instance of it, the shower can be run with two arguments: (i) an `event`, consisting on a list of the incoming and outgoing particles, and (ii) the initial squared energy scale t. Then, the shower runs from the initial scale t by generating emissions which lower the scale and modify in-place the `event` list (i.e. adds more particles to it), until the energy scale is below the cutoff, at which point the running stops.

The Monte Carlo integrator, being responsible for the cross-section factor without the emission, is precisely what is used to create the initial `event` list of particles. Each Monte Carlo point for the integrator has definite values of s, $\cos\theta$, ϕ and, performing a random sum over quark flavours, also a definite flavour q. From the kinematic variables, the momenta of the initial (electron-positron) and final particles (quark-antiquark) can be set, in the centre-of-mass frame, according to

$$\begin{aligned} p_{\mathrm{e}^-} &= \frac{\sqrt{s}}{2}(1,0,0,-1), & p_{\mathrm{q}} &= \frac{\sqrt{s}}{2}(1,-\sin\theta\cos\phi,-\sin\theta\sin\phi,-\cos\theta) \\ p_{\mathrm{e}^+} &= \frac{\sqrt{s}}{2}(1,0,0,+1), & p_{\bar{\mathrm{q}}} &= \frac{\sqrt{s}}{2}(1,+\sin\theta\cos\phi,+\sin\theta\sin\phi,+\cos\theta). \end{aligned} \tag{7}$$

Besides the information of the momenta, the colour content of each particle also needs to be specified: the electron and the positron are colourless particles, the quark carries some colour and the antiquark carries the corresponding anti-colour. All this information is used to create instances of the `Particle` class (another class provided for the project), which form the elements of the initial `event` list.

Hence, the algorithm for the parton shower consists of sampling N Monte Carlo points with the integrator, using the information of the kinematic variables and quark flavour to create one `event` per Monte Carlo point, to then pass these events to the shower, which modifies each of them in-place to yield events with final-state particles in them.

Following the above algorithm, the Monte Carlo integrator for a fixed beam distribution at the mass of the Z boson ($f(s) = \delta(s - M_{\mathrm{Z}}^2)$) was used to generate 1,000 Monte Carlo points, which corresponded to the same number of events for which the parton shower was run. The cutoff scale was set to $t_0 = 1\mathrm{GeV}^2$ and the initial scale at $t = M_{\mathrm{Z}}^2$. For each of these events, the number of final-state particles was counted and weighted according to its Monte Carlo weight, giving a *weighted average of 5.25(6) particles.* Part c).

2.3 Jets and the Durham algorithm

In the initialization of the parton shower discussed in the previous section, one of the required parameters is the energy scale cutoff t_0, below which the perturbative method no longer applies and the running of the shower is stopped. This unphysical parameter is introduced because the probabilities to emit additional quark or gluons at small angles or small energies diverges. However, in doing so, quantities like the number of particles after showering are ill-defined, as they depend on the choice of cutoff. At the same time, actual experiments have a limited resolution, which gives the possibility that additional collinear or soft partons are emitted, with the experimental signature being left unchanged.

In order to address both of these issues, jet algorithms have been developed. These algorithms cluster particles into jets that are well-separated, allowing for well-defined comparisons between experimental data and theory calculations.

For clustering the particles, two generic types of algorithms are commonly used: (i) cone algorithms and (ii) sequential recombination algorithms. In this project, the jet algorithm implemented for the analysis of the final state is the *Durham k_T algorithm*, which is of the second kind. This (final state) jet finding algorithm consists of the following:

1. For each pair (i, j) of final-state particles, compute the distance measure

$$y_{ij} = \frac{2\min(E_i^2, E_j^2)(1 - \cos\theta_{ij})}{Q^2}, \tag{8}$$

   where E_i is the energy of the i-th particle, $\theta_{ij} = \langle \mathbf{p}_i, \mathbf{p}_j \rangle$ is the angle between the spatial momenta, and Q^2 is a reference scale which, for this project, corresponds to the squared mass of the Z boson.

2. Determine the pair of particles (i, j) with the minimum y_{ij} and combine them, i.e., sum their (four-)momenta.

3. Repeat until all final-state particles are clustered into jets.

The result of the above algorithm is a series of splitting scales, corresponding to the minimum values y_{ij} found at each iteration. These splitting scales can then be used to quantify the contributions to the cross-section from different numbers of jets.

After implementing the Durham k_T algorithm to cluster the final-state particles, an analysis of 1 million of events was carried out: The events were created using the Monte Carlo integrator at fixed beam energy, $s = M_Z^2$, for obtaining the initial and final-state particles properties, such as momentum and quark flavour, together with the Monte Carlo weights for each differential cross-section. The parton shower was then run for each of these events with a cutoff scale of $t_0 = 1\,\mathrm{GeV}^2$ and an initial scale $t = M_Z^2$. The results can be seen in Figure 7 and Figure 8, where the results of the simulator are compared to those of SHERPA for differentials from $2 \to 3$ up to $5 \to 6$.

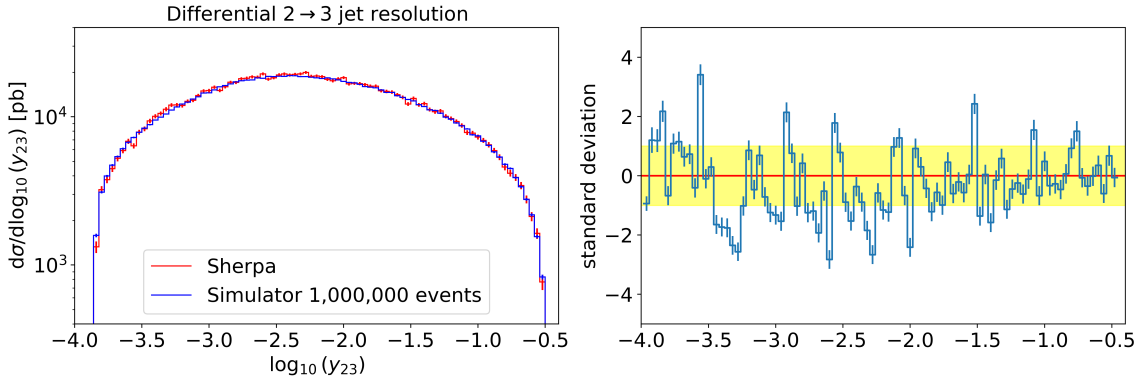
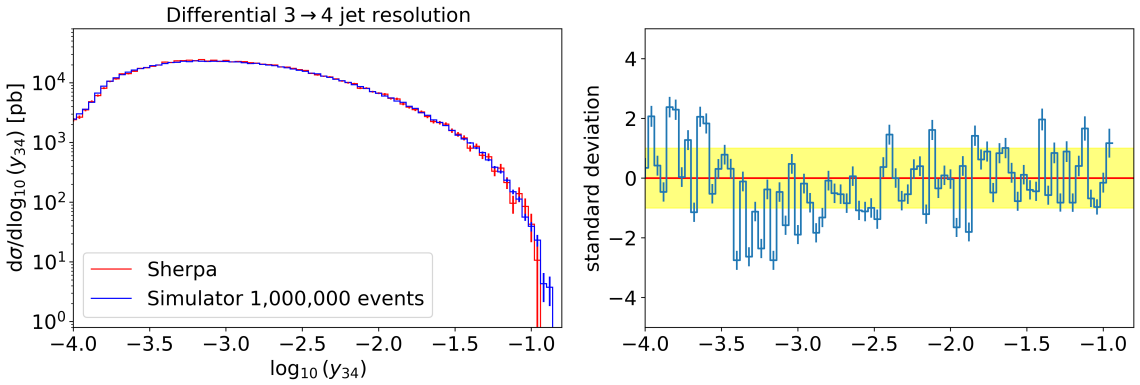
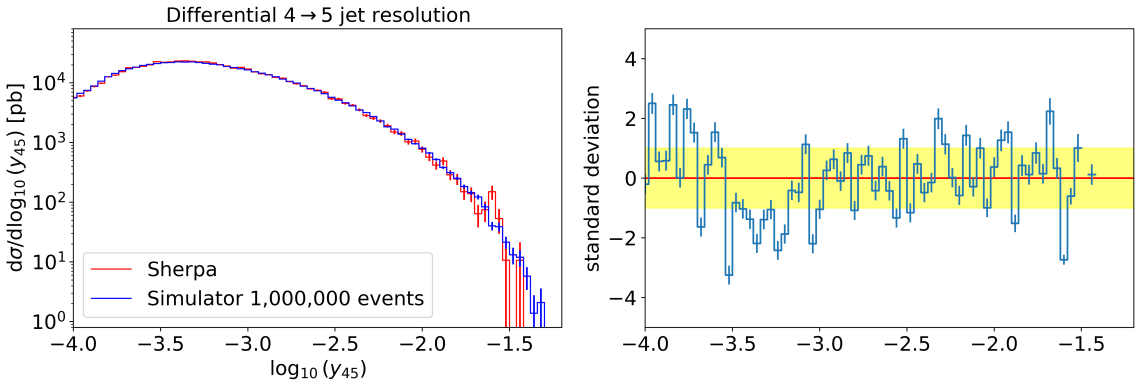


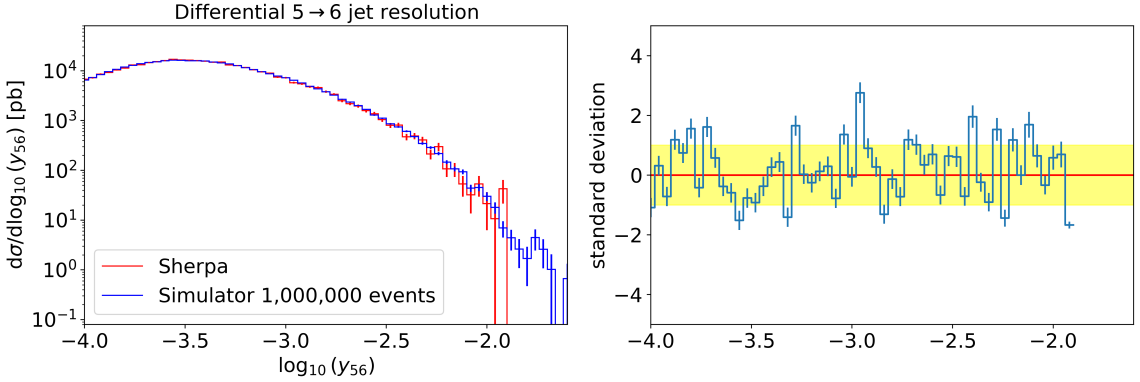
Figure 7



(a)



(b)



(c)

Figure 8: Jet resolution for several differentials. The plots on the right are the associated errors, in standard deviations, to the values of SHERPA.

A Numerical values and squared matrix element

The physical constants used throughout the project are given in Table 3.

Symbol	Value	Description
α	1/129	QED coupling
$\alpha_s(M_\mathrm{Z})$	0.118	QCD coupling at the Z mass scale
M_Z	91.2	Z boson mass
Γ_Z	2.5	Z boson decay width
$\sin^2\theta_W$	0.223	squared sine of the Weinberg angle
Q_e	-1	electric charge of the electron
$Q_\mathrm{u,c}$	2/3	electric charge of (light) up-type quarks
$Q_\mathrm{d,s,b}$	-1/3	electric charge of down-type quark
$T^3_\mathrm{u,c}$	1/2	weak isospin of (light) up-type quarks
$T^3_\mathrm{d,s,b,e}$	-1/2	weak isospin of down-type quarks and the electron
N_q	5	number of light quark flavours (i.e. u, d, s, c, and b)
N_C	3	number of QCD colours
f_conv	$3.89379656 \cdot 10^8$	physical units conversion factor

Table 3: Numerical constants used throughout the project. Masses and decay widths are given in GeV.

To leading order, the squared matrix element for the process $\mathrm{e^-e^+} \to \mathrm{q\bar{q}}$ is given by

$$|\mathcal{M}_{\mathrm{q\bar{q}}}|^2 = (4\pi\alpha)^2 N_C \left[\left(1+\cos^2\theta\right)\left\{Q_\mathrm{e}^2 Q_\mathrm{q}^2 + 2Q_\mathrm{e}Q_\mathrm{q}V_\mathrm{e}V_\mathrm{q}\chi_1(s) + \left(A_\mathrm{e}^2+V_\mathrm{e}^2\right)\left(A_\mathrm{q}^2+V_\mathrm{q}^2\right)\chi_2(s)\right\} + \cos\theta\left\{4Q_\mathrm{e}Q_\mathrm{q}A_\mathrm{e}A_\mathrm{q}\chi_1(s) + 8A_\mathrm{e}V_\mathrm{e}A_\mathrm{q}V_\mathrm{q}\chi_2(s)\right\}\right], \tag{9}$$

where

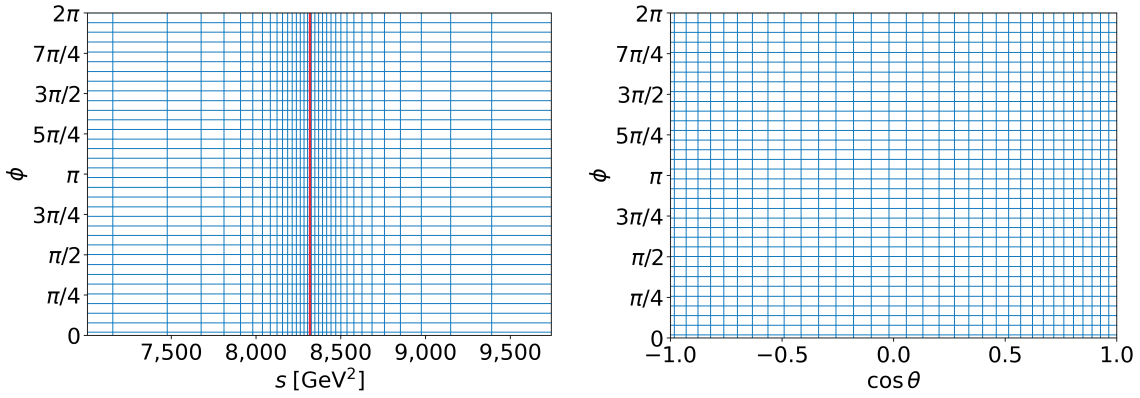
$$V_\mathrm{f} = T^3_\mathrm{f} - 2Q_\mathrm{f}\sin^2\theta_W \qquad \text{and} \qquad A_\mathrm{f} = T^3_\mathrm{f} \tag{10}$$

are the vector and axial couplings of the fermions to the Z boson, respectively, and

$$\chi_1(s) = \frac{\kappa s(s-M_\mathrm{Z}^2)}{(s-M_\mathrm{Z}^2)^2+\Gamma_\mathrm{Z}^2M_\mathrm{Z}^2}, \quad \chi_2(s) = \frac{\kappa^2 s^2}{(s-M_\mathrm{Z}^2)^2+\Gamma_\mathrm{Z}^2M_\mathrm{Z}^2}, \quad \kappa = \frac{1}{4\sin^2\theta_W(1-\sin^2\theta_W)}. \tag{11}$$

B Integration grids generated by Vegas

The adaptive algorithm implemented in the `vegas` package approximates the distributions for each integration variable to improve the evaluations of the integral. From these distributions, different grids for pairs of integration variables can be shown, helping to understand for which points in phase space do higher contributions to the integral occur. The first of these integration grids was shown in Figure 4, corresponding to the pair of kinematic variables s and $\cos\theta$. Since the integrand depends only on these two variables, it is arguably the most insightful one. Here, however, the other two integration grids are shown in Figure 9 for completeness, as they allow to confirm that `vegas` did find a uniform distribution along ϕ. Furthermore, these two integration grids also display more clearly the higher contributions of s for values closer to M_Z^2, the mass of the Z boson, as well as the asymmetrical contributions for $\cos\theta$ towards $+1$ and -1, the distribution being slightly more dense towards 1.



(a) (b)

Figure 9: Vegas integration grids for the other two pairs of kinematic variables. The red line corresponds to $s = M_Z^2$.