The Title

By

Firstname Middlename Surname (email@aims.ac.rw)

June 2017

AN ESSAY PRESENTED TO AIMS RWANDA IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE IN MATHEMATICAL SCIENCES

DECLARATION

This work was carried out at AIMS Rwanda in partial fulfilment of the requirements for a Master of Science Degree.

I hereby declare that except where due acknowledgement is made, this work has never been presented wholly or in part for the award of a degree at AIMS Rwanda or any other University.

Scan your signature

Student: Firstname Middlename Surname

Scan your signature

Supervisor: Firstname Middlename Surname

ACKNOWLEDGEMENTS

This is optional and should be at most half a page. Thanks Ma, Thanks Pa. One paragraph in normal language is the most respectful.

Do not use too much bold, any figures, or sign at the bottom.

DEDICATION

This is optional.

Abstract

A short, abstracted description of your essay goes here. It should be about 100 words long. But write it last.

An abstract is not a summary of your essay: it's an abstraction of that. It tells the readers why they should be interested in your essay but summarises all they need to know if they read no further.

The writing style used in an abstract is like the style used in the rest of your essay: concise, clear and direct. In the rest of the essay, however, you will introduce and use technical terms. In the abstract you should avoid them in order to make the result comprehensible to all.

You may like to repeat the abstract in your mother tongue.

Contents

1. General Introduction

Particle Physics or sometimes called High Energy Physics, is the field of physics that pursues the ultimate structure of matter. This is possible in two ways, one is to look for elementary particles, the ultimate constituents of matter at their smallest scale. The other is to clarify what interactions are acting among the *(forces)* to construct matter.

1.1 Large Hadron Collider (LHC)

Hadron colliders are devices made to explore the world of particle physics, they work as theories testers and also as a discovery machines, an example of these hadron colliders is the Large Hadron Collider in CERN.

Inside the Large hadron Collider, 10^{11} protons are being accelerated to a high kinetic energy and then collided up 40 million times per second to provide 14 TeV proton-proton collision. The LHC also collides with heavy ions. Two general purpose detectors, ATLAS (A Toroidal LHC ApparatuS) and CMS (Compact Muon Solenoid) have been built for probing proton-proton and ion-ion collisions. (Aad et al., 2012).

The main mission of the LHC is to investigate the properties of the Higgs boson. The importance of the Higgs boson from investigating the validity of the standard model suggests that the masses of the elementary particles are generated through the Higgs mechanism. In July 2012, the ATLAS and CMS experiments at CERN's Large Hadron Collider announced that they have had each observed new particle in the mass region around 126 GeV. This particle is consistent with the Higgs boson predicted by the Standard Model.

Most of the interesting physics at LHC involves investigating the results of these interactions (collisions). It has been observed that as a result of these collisions, stable partons are formed, partons consists of quarks and qluons. From experimental point of view, these collisions lead to the production of these partons along side the production of another particles like the $Higgs$ boson. However, new particles are not detected. Instead, the first parton (quark or gluon) that is being produced will produce more partons which will also produce more partons and eventually leading to a spray of partons. This is called *the parton shower*.

The study of quarks and gluons in LHC is challenging because of the production of a single quark or a gluon will actually appear in the detectors as many particles collimated in the same general direction, all arriving at once. The detection of a collimated flow of particles of this nature is called a **jet** (Ellis et al., 2008).

1.2 Quarks and Gluons

In our current view, all matter is made of three kind of elementary particles, *leptons*, *quarks* and *mediators*.

There are six "flavours" of quarks, these are up, down charm strange, top and bottom. Quarks can successfully account for all known mesons and baryon which are known as hadrons, Quarks carry colour charge: red, green or blue, the colour name here is an analogy that is famously used among physicists to describe a three kind of generating force. They are called by their number or any other index. We can also we think of it as a three primary additive colours.

The other partons (glouns) mediate the interaction between quarks. There are eight gluons, which come from the gauge group $SU(3)$ and has eight generators.

Gluon is a massless spin 1 particle, carrying charge called colour charges as well as the quarks, because gluons can interact among themselves.

The colour charge has strange property that it exerts a constant force that binds colour carrying particles together, this can be visualized using the analogy of a rubber band, the stronger you pull on the rubber band the tighter it feels. If we do not pull on it at all, it hangs loose. The same thing happens for the particles, that means at a very short distance, the force is relaxed and the particles behave as free particles. As the distance between them increases, the force acts like a rubber band and pulls them back stronger. When the rubber band is stretched beyond its limits, it cuts into many pieces producing more particles. This phenomenon is known as the colour confinement. In other words, these particles tend not to be separated by a macroscopic distance. This limits the range of strong force, which is believed to be of order 10^{-15}m - the dimension of a nuclear particle.

The theory which describes this force is called $Quantum\ Chromodynamics$ (Nagashima and Nambu, 2010).

1.3 Monte Carlo Simulation

Due to the complex nature of the event[1] at the Large Hadrons Collider , the description of the final state involves a multi-particle calculations. The accurate prediction of the final state in hadron colliders is still one of the hardest problems. This problem roots to the non-abelian nature of QCD, which leads to a colour confinement at a long distance. The two main problems are the description of the hadron formation and the evolution of QCD final states from short to long distances. Those problems can be tackled to a good approximation by Monte-Carlo event generators. The figure 1.1 illustrates the collision of two protons and the splitting of the particles (parton shower) after the collisions.

[1]Event in this context refers to the result of the particle interaction (collision).

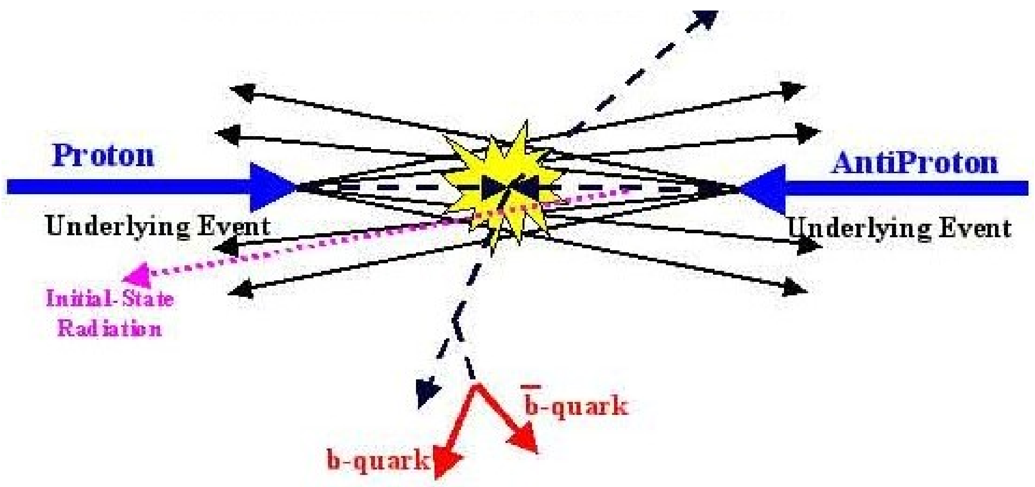


Figure 1.1: Illustration of the parton shower

2. Monte Carlo Computational Techniques

The name Monte Carlo method is a set of mathematical tools that was first used by scientists working on nuclear projects in Los Alamos. The essence of this is to generate numbers with probability that can be used to study physical phenomena. In our context, the definition of Monte Carlo method is the use of randomly generated numbers to imitate a physical behaviour that is not necessarily considered to be random (Kalos and Whitlock, 1986).

2.1 Generating Random Numbers with different PDFs

Generating samples of different probability distribution function(pdf) is essential since we are simulating various variables that have different numerical behaviour. For example if our pseudo-random numbers are uniformly[1] distributed in the interval $[0, 1]$ and instead we need numbers that have normal distribution restricted to the same interval. This part of the essay discusses two methods which are widely used.

2.2 The Accept-reject method

Let x be a uniformly distributed variable in the interval $[0, 1]$, which is the variable of interest. A different distribution of x is required, let $p(x)$ denotes the pdf of the required distribution. Another random uniformly distributed variable is generated, this will serve as the accept-reject tool. First we calculate the maximum of $pdf(x)$, hence the y is generated in the interval $[0, pdf(x)_{max}]$. Now we check if $y \leq pdf(x)$. If this is the case, accept x, otherwise reject and start again.

```
$x \leftarrow$ uniform in $[0, 1]$
$y \leftarrow$ uniform in $[0, pdf_{max(x)}]$
**if** $y \leq pdf(x)$ **then**
    accept $x$
**else**
    Reject
**end if**
Begin again
```

For example, if we have a number x that falls under uniform distribution and we want to reshape this so that we get a number that has the distribution $\frac{1}{x}$, then we find the maximum value of probability density function $(p(x))$ for x. Here, we add small number to x so that we avoid the singular point when $x = 0$. After that we generate another sample y that is also uniformly

[1]Uniform means that every value in the range of the distribution is equally likely to occur. This distribution is widely used for generating random numbers for other distributions, it is denoted by U.

distributed in the interval $[0, pdf_{max}]$. Now we check if y is less than $p(x)$. If so, we add x to our distribution, if not we start again (Weinzierl, 2000).

The histogram in figure 2.1 demonstrate the example above. For the python code see `Accept_reject.py`.

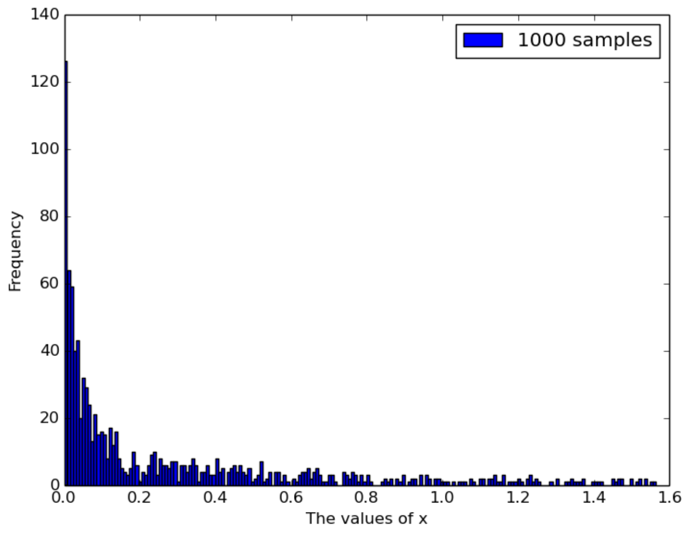


Figure 2.1: Generating values with distribution $\frac{1}{x}$. The small values of x are more likely to occur.

2.3 The Inverse-transform method

The inverse transform method is used for the same purpose as the accept-reject method, generating random numbers that are distributed according to a specific distribution.

Let x be a random variable distributed with probability density function $p(x)$. Let u be a random variable that is uniformly distributed in $[0, 1]$ and $P(x)$ is the cumulative density function (CDF), then we set

$$x = P^{-1}(u) \tag{2.3.1}$$

(Weinzierl, 2000).

The following pseudo-code shows the inverse-transform method algorithm

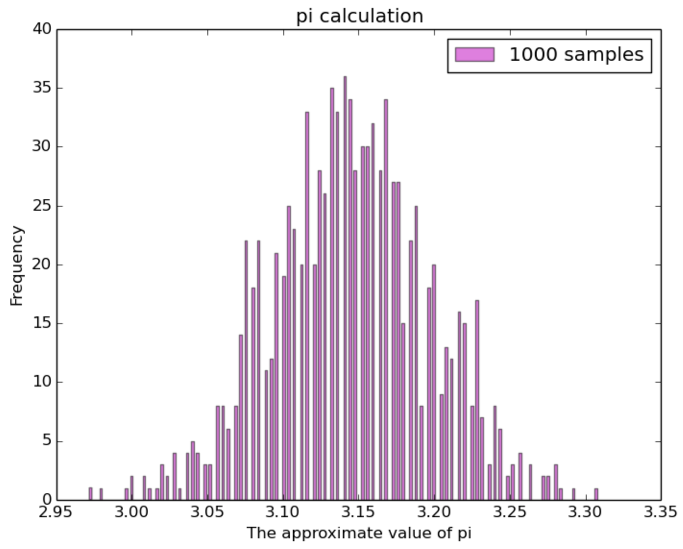
```
$u \leftarrow$ uniform in $[0, 1]$
Generate a uniform $[0, 1]$ random $u$
**return** $x = P^{-1}(u)$
```

The inversion method is exact when an explicit form of the CDF is known. The CDF is a continuous and a strictly increasing function. It can be obtained either analytically through the integration of the PDF or numerically. For example, when the pdf $= 0$, the CDF is not well defined, which means that the CDF is constant. In such cases the evaluation of the CDF can be done numerically, which can be an infinite time. This causes the inverse transform method to become less efficient (Devroye, 1986).

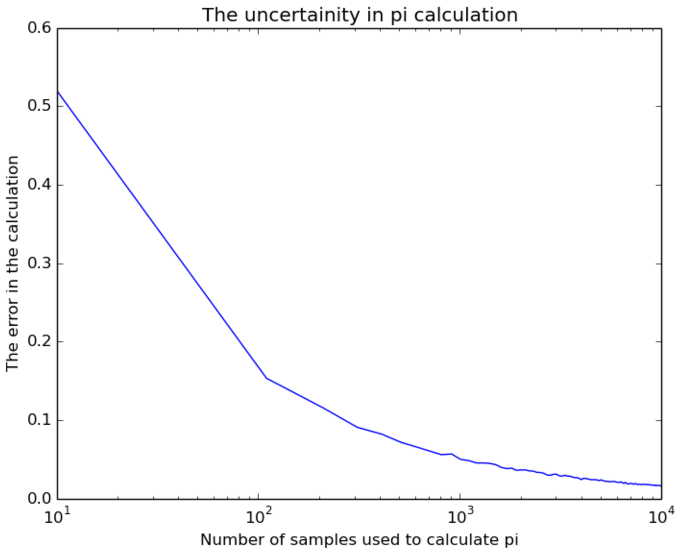
In contrast, the efficiency of accept -reject algorithm does not depend that much on the shape of the pdf. Moreover, the accept - reject method does not require the evaluation of the CDF.

The following example demonstrate the efficiency of accept - reject method in which the value of π is calculated using the accept - reject method.

Assume we have a box of side length D and a circle of diameter D inside the box, the probability that a point in the box is also in the circle is approximately the area of the circle over the area of the box, that is, π over 4. From this we can approximate the value of π. The histograms in Figure 2.2 exhibits this calculation and also the error in the calculation. For the python code see `Calculation_of_pi.py` and `uncertainity_in_pi_calculation.py`.



(a) pi value



(b) The uncertainty

Figure 2.2: At the top, this histogram shows the values of π that are generated from sampling 1000 numbers. At the bottom the plot shows the error in the calculation when we use different number of samples. From the uncertainty plot, it can be seen that as we increase the number of samples the uncertainty decreases.

3. The Parton Shower

In general parton showers are approximations of the higher order real emission corrections (this refers to the stable hadrons) to the hard scattering. The word "hard", means the process involves a transfer of large momentum, either a violent scatter or creation of large mass. They locally conserve flavour and four momentum, and also they are consistent, which means, the particle either splits into two or not. Since the parton showers are simulations of the branching and splitting processes, the quality of their predictions depend on how precise is the implementation For example, one can ensure the colour coherence through selecting an evolution variable representing the angular ordering, though this in not the only choice to ensure the colour coherence (Höche, 2014).

3.1 The Parton Shower Model

The parton shower model is built around the idea of successive one-to-two splittings, which are combined together forming a tree like sequence as in figure 3.1.

The Monte Carlo model of the parton shower can be described as a sequence of stochastic and deterministic processes.

The deterministic stage is where the particle is produced with some four momentum and travels for a finite time and distance.

After the time distance travel, the particle will spilt into two, soft particle (radiated) with angle θ taking Z fraction of the initial particle energy. The other particle with the rest of the energy θ and Z represents the stochastic part of the model.

There is one QCD approximation that will be useful regarding θ and Z distributions. This is the *soft* and *collinear* approximation. *Soft* implies that an emitted particle has a very little energy compare to the particle that emitted it. *Collinear* means that it is emitted with angle very small relative to another particle in the event (Salam, 2010b).

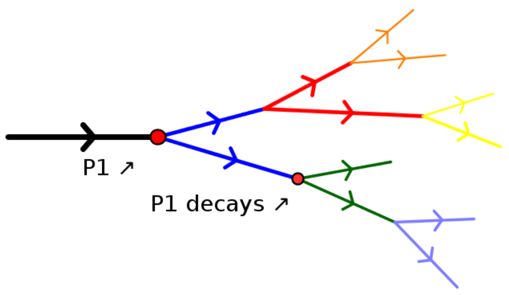


Figure 3.1: Successive one-to-two splitting

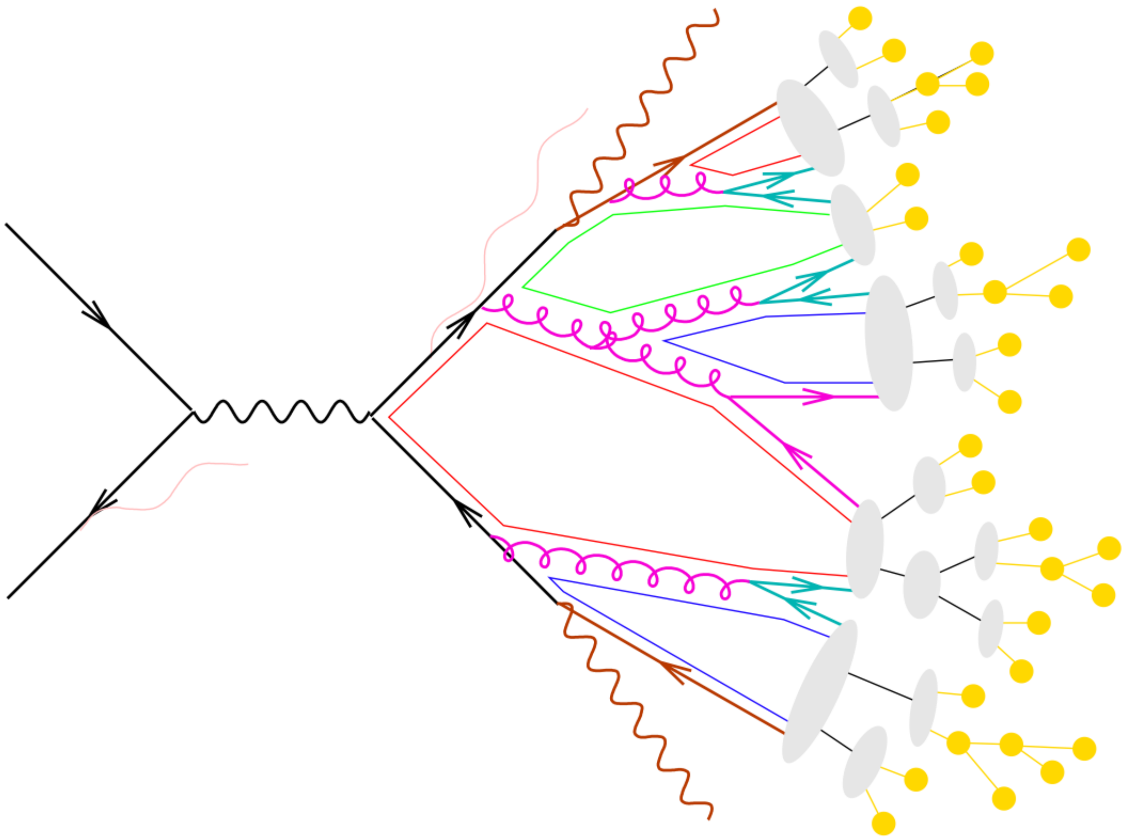


Figure 3.2: Illustration of the subsequent dynamics of the parton shower (red and purple lines) and the hadronization (gray clusters and yellow circles) of the hard partons

3.2 Hadronization

To reflect the colour neutrality of the particles in our model the partons will be transformed into a stable hadrons which are colour neutral. This process is called hadronization. The process in which this happens is not well understood. For this model, a very simple approach is taken, where a direct translation between each parton and hadron is made. To achieve this, a threshold energy is defined. Below it, the partons will begin to hadronize or stop splitting (Salam, 2010b). The figure 3.2 illustrate the process of splitting and hadronization.

3.3 Parton Shower Simulation

The following is a simple simulation for a splitting of a single particle into two. The general idea of the parton shower simulation can be expressed by the following algorithm

```
Stability limit (hadronization limit) $S$
List $\leftarrow$ 4 momenta
$List \leftarrow$ initial particle $P_i$
**if** energy of $P_i \geq S$ **then**
```

```
        Split into two and add to list and check again
    else
        Begin hadronization
    end if
```

First, we will start by describing a tow simple dimensional model. After will describe the 3 dimensional model.

3.4 2 Dimensions Parton Shower

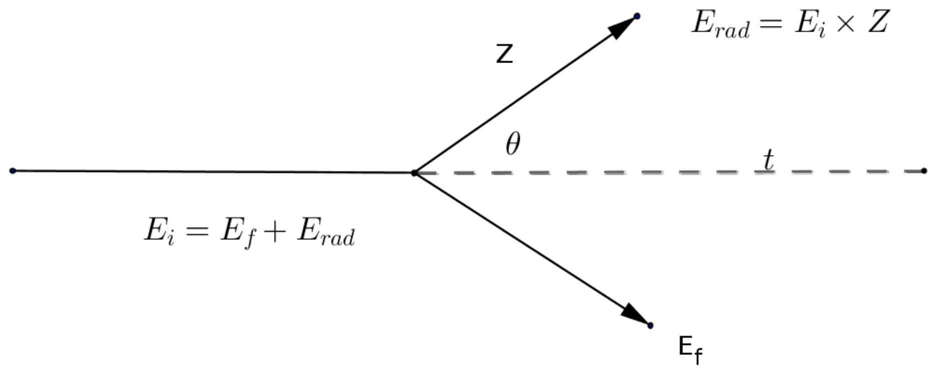


Figure 3.3: A pictorial describing the splitting of in two dimensions.

The two dimensional model accounts for one rotation matrix θ (see figure 3.3). It evolves generation of two random numbers, one of them represent the angle and the other represents the energy of the radiated particle. Here, both the energy fraction z and the angle θ are following the distribution $1/z$, this comes from the QCD approximation that was mentioned earlier. Where the former lies in the interval $[0, 1]$. To avoid the singularity at $z = 0$, a cutoff value of ϵ is added to the denominator by setting $PDF(z) = \frac{1}{\epsilon+z}$.

Considering the later in the interval [0,$\pi/2$], θ is modified in a similar way as z. θ and z are generated by applying the accept-reject method on a set of numbers that are uniformly distributed.

As for the kinematics description, the initial particle has four momentum (E, p_x, p_y, p_z) [1], for simplicity we assume that the particle is travelling in x direction with energy E. Therefore, the four momentum which describes this particle is the following

$$P_i^\mu = \begin{pmatrix} E \\ p_x \\ 0 \\ 0 \end{pmatrix} \tag{3.4.1}$$

[1]The four momentum vector is usually written as $(E/c, p_x, p_y, p_z)$. Henceforth, we are considering the natural units in which $c = 1$

Where p_y and p_z are equal zero following our direction assumption. In a given splitting, with generated pair (θ, z), the kinematics of the radiated particle are determined.

To continue the showering process and to allow the model to proceed, it is necessary to determine the kinematics of the final state particle by applying the conservation of energy and momentum

$$P_i^\mu = P_f^\mu \tag{3.4.2}$$

And Since

$$P^\mu P_\mu = E^2 - (p_x^2 + p_y^2 + p_z^2) = E^2 - ||p|| = m_0^2 \tag{3.4.3}$$

Which is lorentz invariant quantity, $i.e$, it does not depend on the frame. Here, we can make a simplifying assumption, in which we assume that the mass of the radiated particle is zero. This assumption is based on the fact that, in LHC the quark mass is $\sim$ 1MeV [2] where the hadron masses is $\sim$ 1 TeV. Now equation 3.4.3 will become

$$E^2 = ||p|| \tag{3.4.4}$$

In other words the final state particles energy and momentum summation is zero (Salam, 2010b).

In describing the direction of the radiated particle after it is being produced, we use the rotation matrix in two dimensions which rotates the radiated particle with the angle θ

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{3.4.5}$$

From this we can determine the direction of the another particle here we will denote its four momentum by P_{part}^μ . Since P_f^μ after the splitting is

$$P_{rad}^\mu + P_{part}^\mu \tag{3.4.6}$$

and from equation 3.4.3 then

$$P_i^\mu = P_{rad}^\mu + P_{part}^\mu \tag{3.4.7}$$

Therefore,

$$P_{part}^\mu = P_i^\mu - P_{rad}^\mu \tag{3.4.8}$$

For the hadronization, a tunable parameter is made to account for the hadronization. The figure 3.4 shows a graphical representation of the two dimensions parton shower as explained above. For the python code see `two_D_partonshower.py`.

3.5 3 Dimensions Parton Shower

Since the real physics we are modelling is in three dimension, we need to modify our model to account for this. An additional splitting angle ϕ will be included, which is the azimuthal angle of the decay around the direction of travel of the initial particle, as shown in figure 3.5.

[2]1 Mev = 10^6 ev and 1 Tev = 10^12 ev.

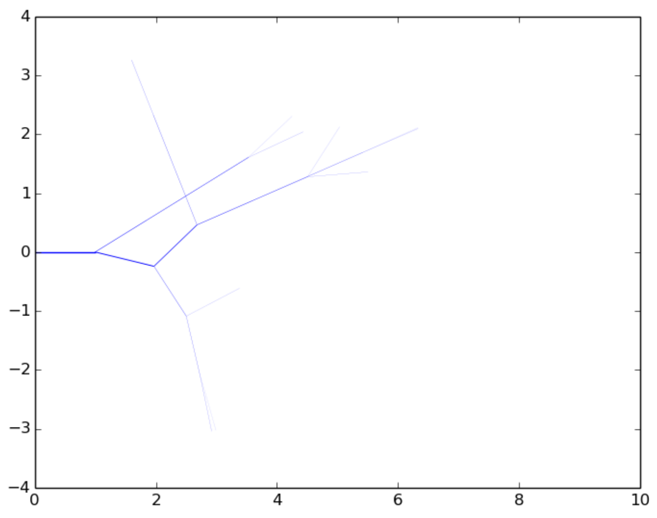


Figure 3.4: 2D simulation of the spliting of a single parton. Here, the colour fades as the energy decreases

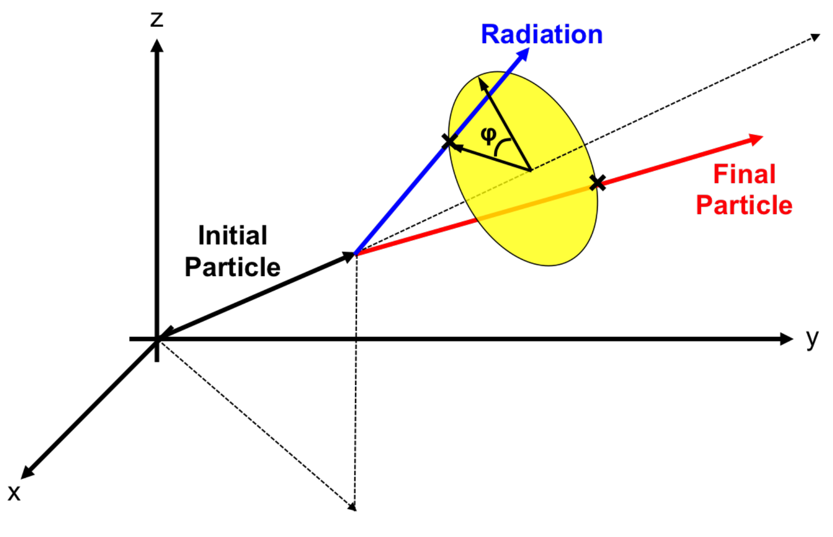


Figure 3.5: An illustration of the azimuthal angle ϕ of a one-to-two splitting occurring in three dimensions.

Unlike the angle θ, there is no preference for the value of this angle and it should therefore be chosen from a uniform distribution within the interval $[0, 2\pi]$ (Salam, 2010b).

There are no major differences between the three and two dimensional models, except that in the three dimensional model the particle will be rotated twice with the angles θ and ϕ. This is done by applying the three dimensional matrix

$$\begin{pmatrix} \cos\theta + u_x^2(1-\cos\theta) & u_x u_y(1-\cos\theta) - u_z\sin\theta & u_x u_z(1-\cos\theta) + u_y\sin\theta \\ u_y u_x(1-\cos\theta) + u_z\sin\theta & \cos\theta + u_y^2(1-\cos\theta) & u_y u_z(1-\cos\theta) - u_x\sin\theta \\ u_z u_x(1-\cos\theta) - u_y\sin\theta & u_z u_y(1-cos\theta) + u_x\sin\theta & \cos\theta + u_z^2(1-\cos\theta) \end{pmatrix} \tag{3.5.1}$$

The figure 3.6 shows a graphical representation of the three dimensional parton shower. For the python code see `partonshower3d.py`.

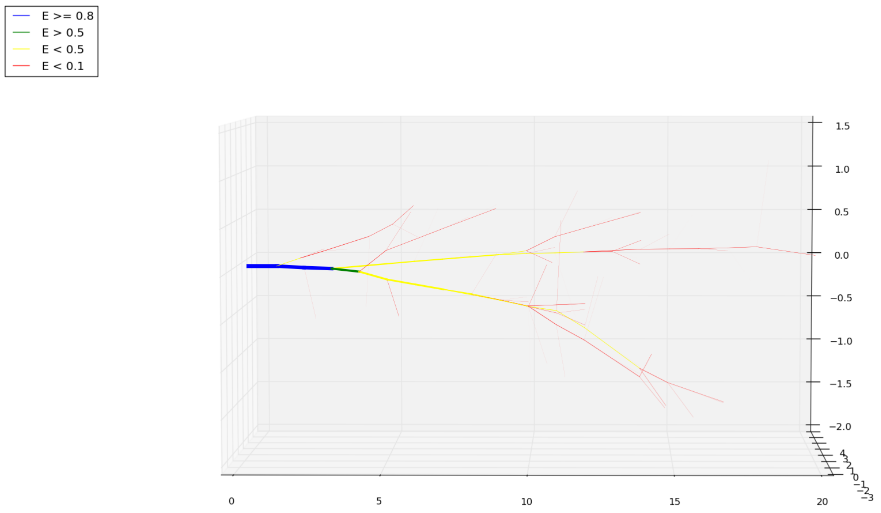


Figure 3.6: 3D simulation of the spliting of a single parton

3.6 A Jetty Event

In the LHC collision, usually two partons are emitted, it is very rare to produce a single parton. Since the collision was between two particles travelling opposite to each other, a simplification assumption can be made. That the particles are in a configuration that conserve the momentum, this is as a result of the initial momentum $= 0$. There are some aspects of this configuration which are stochastic, these are :

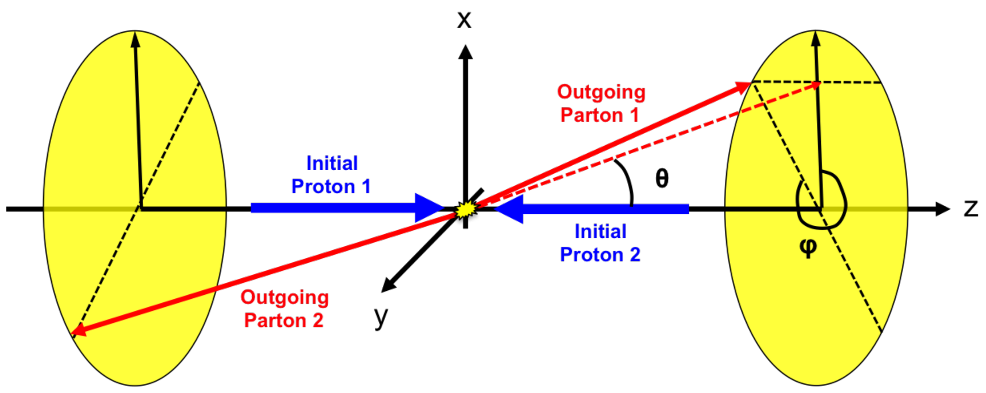


Figure 3.7: A pictorial representation of a proton-proton collision event similar to that in the LHC.

- Energy : the emitted particle has a randomly distributed energy which falls under the exponential distribution $e^{-\alpha E_{parton}}$, where α is a physically meaningful parameter that characterizes the collisions at the LHC and E_{parton} is randomly chosen energy of the parton.

- The azimuthal and the polar direction. The collision has no preferred angle in terms of the both angles θ and ϕ as shown in figure 3.7, hence, the former is uniformly distributed in the range $[0, \pi]$ and the later is uniformly distributed in the range $[0, 2\pi]$ (Salam, 2010b).

There are three main programs in use at present for the generation of simulated collider events. They incorporate different combinations of the approaches of the model described above. These are HERWIG, PYTHIA and SHERPA (Buckley et al., 2011).

4. Jet Algorithms

4.1 Jet Reconstruction

After the process of the splitting and branching, the quarks and gluons start to hadronize leading to a collimated spray of stable colourless hadrons called jets.

The definition of the jet is central in comparing the data and the theoretical predictions. The definition is provided in the form of the jet algorithm, this means the jet algorithm and its corresponding parameters and recombination scheme. The paragraphs below gives a detailed description of the algorithm.

The jet reconstruction is essential in understanding the link between the observed physics or the long distance physics and the underlying physics (short distance physics) in the parton level. Also, the accurate reconstruction is very important in comparing the theoretical predictions and the data.

Jet algorithms basically rely on merging. This means that they merge objects that are near to each other. As for the accuracy of the algorithm, we assume that the kinematics of the clustered jet provide a useful measure of the kinematics of the underlying short distance physics. In particular, we assume the basic mismatch between the long observed hadrons and short distance unobserved partons which does present any numerical limitations (Ellis et al., 2008).

4.2 Jet Algorithms

There are two broad classes of jet algorithms, the **Cone** algorithms and the **Sequential** algorithms. Both algorithm works on defining the jets by the idea of nearness.

It is important to recognize that jet algorithms involve two distinct steps. The first step is to identify the members of the jet, *i.e*, the partons that make-up the final stable jets. The second step is to construct the kinematic properties that will characterize the jet (Berger et al., 2001). Here, we focus on the second class.

4.3 The Cone Algorithms

The cone algorithms associate hadrons into jets by identifying those that are nearby in angle, $i.e$, they follow the geometrical intuition in defining the jets, where the jet is composed of hadrons and partons whose momenta lie within a cone defined by a circle in $\eta - \phi$ plane. Where η is the pseudo-rapidity and can be calculated by $\ln(\cot \frac{\phi}{2})$, and ϕ is the azimuthal angle (Berger et al., 2001).

For the first step in the algorithm, *i.e*, identifying the members of the jet, it is a simple sum over the all (short and long distances) within a cone centred at $\eta - \phi$ plane. Here, one introduces the concept of *stable* cones as a circle of fixed radius R in the plane $\eta - \phi$ such that the sum of all four momenta within the cone points to the same direction as the centre of the circle. The cone algorithms attempts to identify the stable cones. Thus, at least in principle one can think in terms of placing trial cones randomly in the plane $\eta - \phi$ and allowing them to follow until a stable cone or a jet is found (Ellis et al., 2008).

Cone algorithms differ in the way they deal with the fact that the stable cones may overlap. There are two famous cone algorithms. They are, the midpoint cone algorithm and SISCone (seedless infra-red safe cone) (Ellis et al., 2008).

4.4 Sequential Algorithms

These algorithms work by defining a distance between pairs of objects, performing a successive recombination of the pair of closest objects. And stopping when all objects are further apart.

One starts by first defining these distances, d_{ij} and d_{iB}, where d_{ij} is the distance between objects (pseudo-jets) i and j [1]. And d_{iB} is the distance between the object (pseudo-jet) i and the beam B. The clustering proceeds by identifying the smallest of the distances and if it is d_{ij} recombine the objects (pseudo-jets) i and j. Otherwise, if it is d_{iB} calling i a jet and removing it from the list of objects. The distances are recalculated and the procedure repeated until no objects left (Cacciari et al., 2008).

For merging and combining the objects, we use the 4 -vector recombination scheme, whereby to combine two particles we add their four momenta (Blazey et al., 2000).

The difference between the sequential algorithms lies in the definition of the distances it measures:

$$d_{ij} = min(k_{ti}^{2p}, k_{tj}^{2p})\frac{\Delta_{ij}}{R^2}, \tag{4.4.1}$$

$$d_{iB} = k_{ti}^{2p}, \tag{4.4.2}$$

where $\Delta_{ij}^2 = (\eta_i - \eta_j)^2 + (\phi_i - \phi_j)^2$ and k_{ti}, η_i and ϕ_i are the transverse momentum, pseudo-rapidity and the azimuth of the particle i (Cacciari et al., 2008). The exact formula for the transverse momentum depends on the beam axis, which is z hence, $k_t = \sqrt{p_x^2 + p_y^2}$. The pseudo-rapidity describes the angle relative to the beam axis. It is defined as $-\ln\left[\tan\left(\frac{\theta}{2}\right)\right]$ where θ is the polar angle (Salam, 2010a). Sometimes the rapidity y_i is used instead of the pseudo-rapidity, the rapidity is defined as $\frac{1}{2}\ln\frac{E_i+p_{zi}}{E_i-p_{zi}}$, where E_i and p_{zi} are the energy and component of the momentum along the beam axis of the particle i(Cacciari et al., 2012). The parameter R describes the jet radius, it is a parameter of the algorithm that determines its angular reach (Cacciari et al., 2012). The parameter p govern the relative of the energy versus geometrical (Δ_{ij} scales (Cacciari et al., 2008).

[1]Pseudo-jet since it is neither a particle, nor yet a full particle. It can be a single particle or a composition of particles.

For $p = 1$, one defines the inclusive k_t algorithm. The case where $p = 0$ corresponds to the inclusive Cambridge/Aachen algorithm. $p = -1$ refers to anti-k_t jet clustering algorithm (Cacciari et al., 2008).

The behaviours of different jet algorithms are illustrated in figure 4.1 where a parton level event has been taken together with random soft (radiated) particles and then clustered with 4 different jet algorithms. The figure shows the region which within the random soft particles are clustered. For the k_t and Cambridge/Aachen algorithms, that region depends somewhat on the specific set of soft particles and the irregular borders of the jet are consequence of the randomness of the soft particles. For SISCone it can be seen that single-particle jets are regular (though with a radius $R/2$), while composite jets have varied shapes. With the anti-k_t algorithms, the hard jets are circular and the softer jets have more complex shapes. The pair of jets near $\phi = 5$ and $y = -2$ provides an example in this respect. The former is much softer than the later. The circular hard jets clips a lens shaped region out of the soft one, creating the crescent shape (Cacciari et al., 2008).

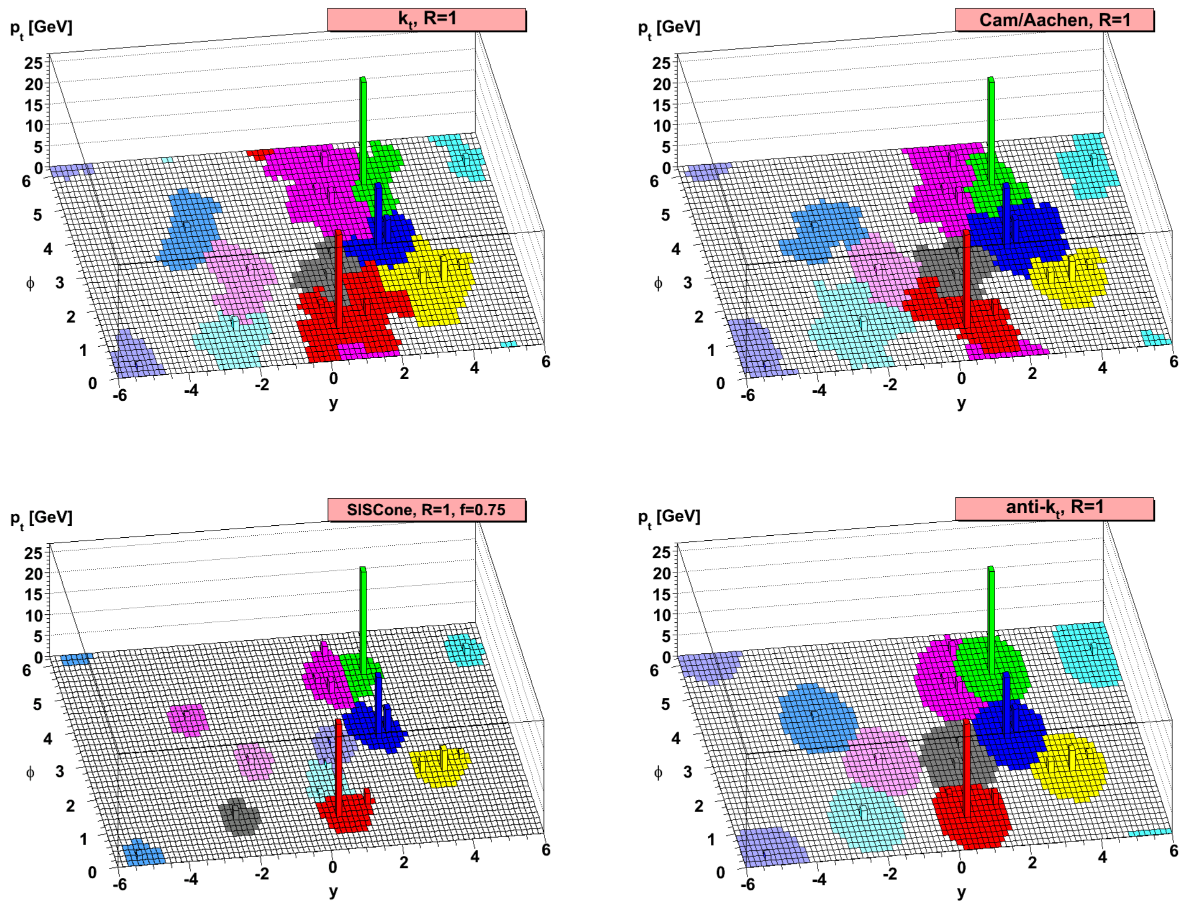


Figure 4.1: A sample parton-level event (generated with Herwig), together with many random soft, clustered with four different jets algorithms, illustrating the "active" catchment areas of the resulting hard jets.

4.5 Implementation of Anti-k_t

The implementation of anti-k_t clustering algorithm can be illustrated by the following algorithm

```
Calculate all the $d_{ij}$ and $d_{iB}$
Find the minimum of $d_{ij}$ and $d_{iB}$
if minimum distance is $d_{ij}$ then
    Recombine $i$ and $j$ in a single object
    return step one
else
    the minimum is $d_{iB}$
    Declare $i$ as a jet and remove from the list
    return step one
end if
Repeat until no objects left
```

The algorithm for this work was implemented in Python. The main step in the algorithm decides the nature of the object, $i.e$, finding the minimum in the list of the distances. We used the Python function `heap queue` which is an implementation of the priority queue algorithm. This function gives the smallest element in a list, maintaining the list invariant.

For the code and the documentation see `anti_kT_algorithm.py`.

5. Jet Observables

After performing the jet clustering on the generated parton shower. The next step is to define final state observables, that can be used to extract specific properties of the final state. Here, there are two kinds of observables, event observables and jet observables.

The most important feature of the observables is that they can be defined both for the Monte Carlo simulation and the real collision data, allowing for tuning the underlying parameters of the model used to generate the parton shower.

5.1 Single Jet Observables

As an example of the jet observables, the following association can be made $(E, p_x, p_y, p_z)_{parton}$ $\sim (E, p_x, p_y, p_z)_{jet}$. Hence, the jets were formed by combining four momentum vectors together and the jet itself is composed of multiple particles. Focusing on the first entry E, one can recognize that the parton energy is represented as $E_{parton} \sim E_{jet} = \sum_{i \in jetconstituents} E_i$, where the jet constituents are the stable particles that the jet clustering initially started with.

Another jet observable is the number of the constituents in the jet with highest energy. This is an intuition about the formation of the jet. The histogram in figure 5.1 shows the data taken from running a jetty event 1000 times and then clustering using anti-k_t algorithm. Here, different values of the parameter R were chosen.

For the code and the documentation see `number_of_constituents_observable.py` and for the histogram see `hist_of_n_of_constituents.py`.

Another single jet observable that we have worked on is *pseudo-mass* observable. This observable is described as follows

$$pseudo - mass(J) = E(j_1) \times E(j_2) \times \Delta_{ij}, \tag{5.1.1}$$

Where j_1 and j_2 are the two highest momenta of the jet J and Δ_{ij} as given above. These calculations were performed on the jets with highest energy. Conditions where the jet with the highest energy has one constituent, this observable was not calculated. The histogram in figure 5.2 illustrates the results of this calculation for 1000 events clustered with anti-k_t algorithm with different values of R.

For the code and the documentation see `Pseudomass_observable.py` and for the histogram see `hist_pseudomass.py`.

5.2 Event Observables

These observables associate the elements of initial event and final state particles (jets). An example of this is the number of jets in the event. This exhibits the number of parton in the

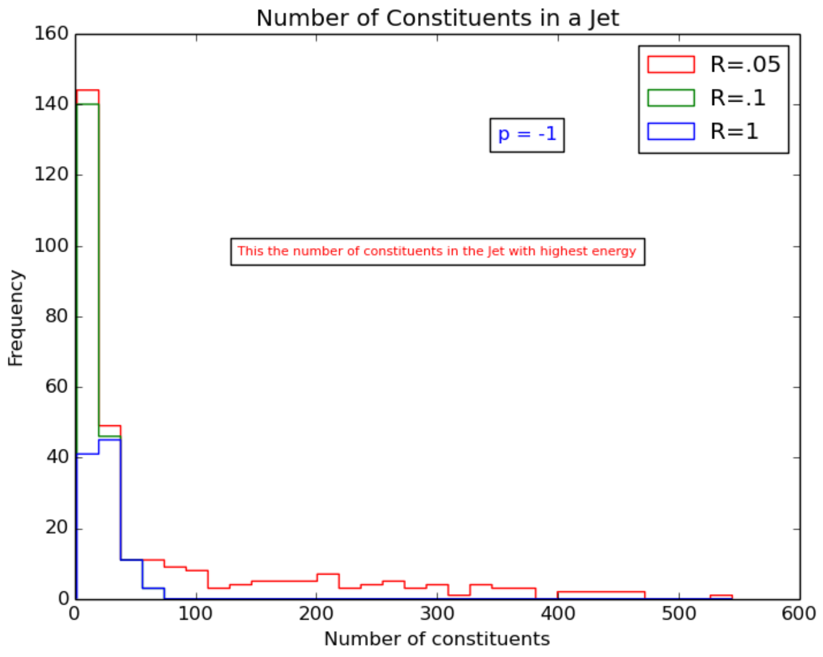


Figure 5.1: Number of constituents of a jet for differnet values of R. Note that here we are looking at the difference on the y axis. The values that correspond to $R = .1$ are the differnce between the blue line $R = 1$ and maximum of $R = .1$.

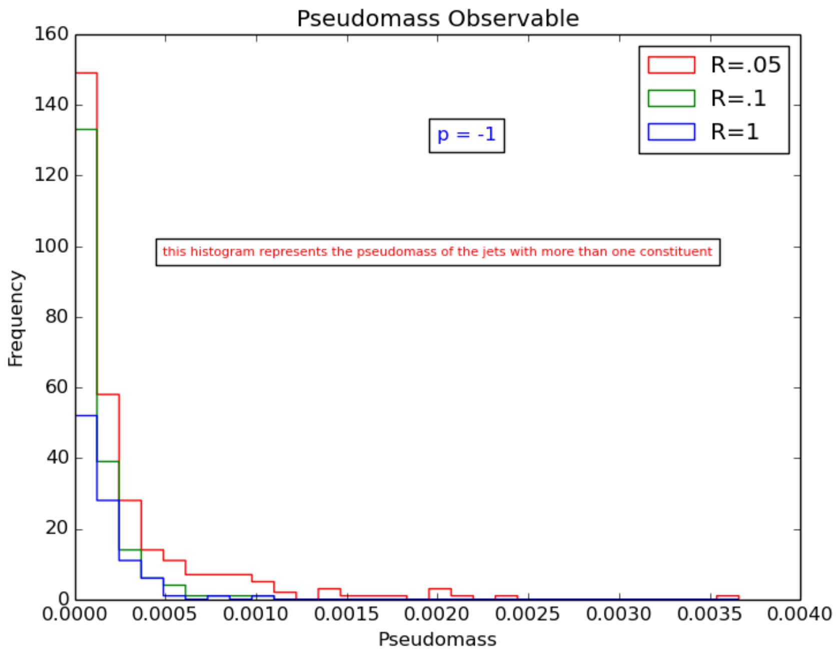


Figure 5.2: Pseudo-mass of jet when uses differnt values of R. Note that here we are looking at the difference on the y axis. The values that correspond to $R = .1$ are the differnce between the blue line $R = 1$ and maximum of $R = .1$.

generated parton shower. The histogram in figure 5.3 shows the results of clustering 1000 events using anti-k_t with different values of R. One sees the number of jets if it is affected by the choice of R. It worth noting that as R gets smaller, the number of jets in the event increases.

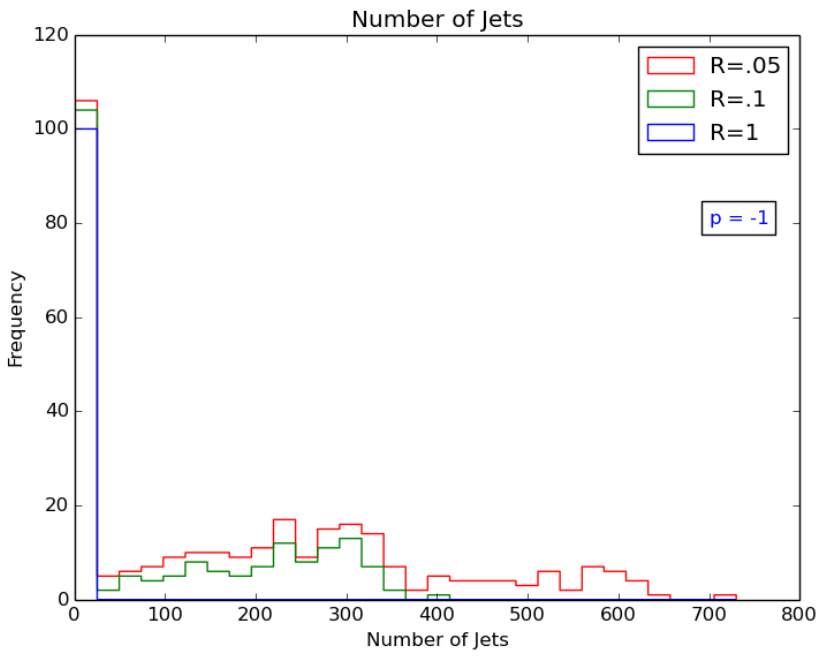


Figure 5.3: Number of jets for different values of R. Note that here we are looking at the difference on the y axis. The values that correspond to $R = .1$ are the differnce between the blue line $R = 1$ and maximum of $R = .1$.

For the code and the documentation see `number_of_Jets_observable.py` and for the histogram see `hist_n_of_Jets.py`.

6. Pythia

Pythia is a general-purpose event generator. It has been used for electron-positron and proton-proton collisions and for other physics studies. It is extensively used for studying physics at LHC. At the beginning the code was written in `Fortran 77`, for the LHC era the experimental community made a decision to move the code to `C++`. The latest version of Pythia (8.1) was released in 2007, which is written in `C++` (Buckley et al., 2011).

In general Pythia performs on idea of the program designed here for the parton shower. However, it accounts for more of the underlying physics.

6.1 FastJet

`FastJet` is a `C++` package provides a wide range of jet finding and analysis tools. It includes efficient implementation of all widely used sequential recombination jet algorithms for proton-proton (pp) and electron-positron (ee^+) collisions. In the case of jet clustering, the working principle is the same as the algorithm that we implemented in chapter 4, but it is much faster (Buckley et al., 2011).

In our work we generated 1000 events using Pythia 8.1 and analysed them using `FastJet`.

We extracted two observables the mass of the jet observable and the pseudo-mass observable. And also analysed some physical quantities, the transverse momentum k_t, pseudo-rapidity η and the azimuth ϕ. Here the value of the parameter R is set to 1. For more about the documentation of `FastJet` see (Buckley et al., 2011).

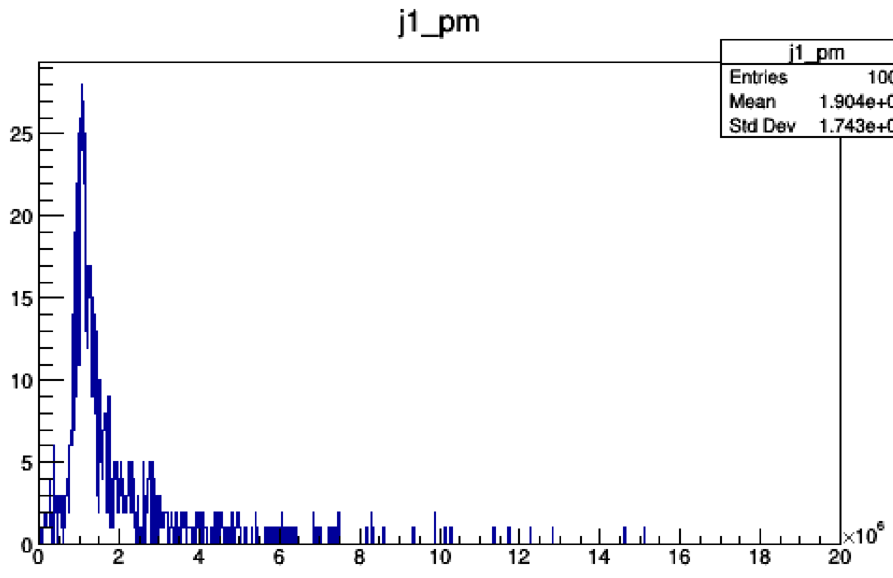


Figure 6.1: The pseudo-mass of jets obtained from analysing 1000 events. R is chosen to be 1

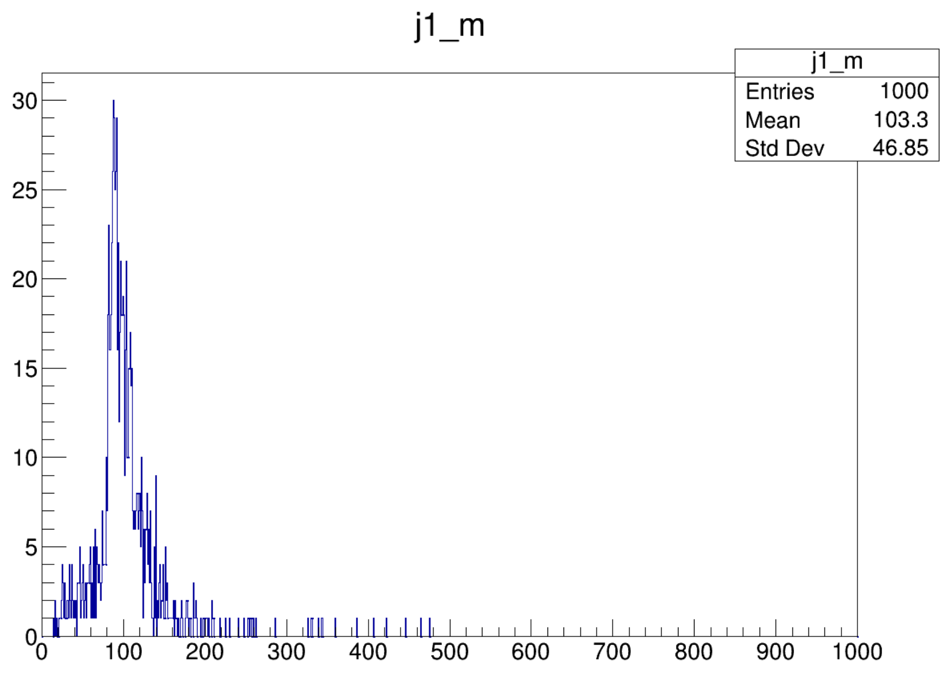


Figure 6.2: The mass of the jets obtained from analysing 1000 events.

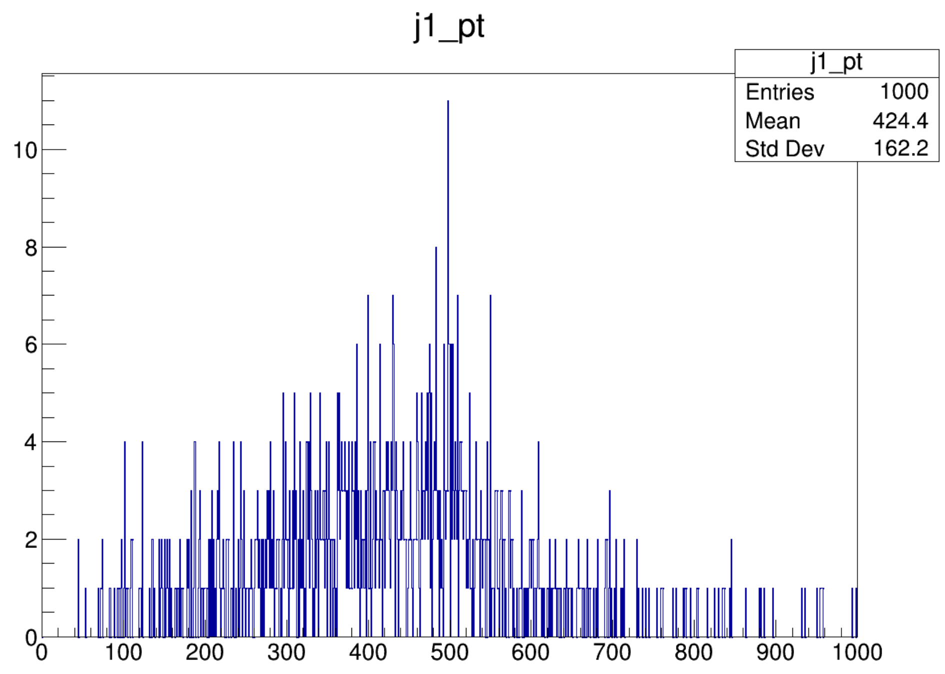


Figure 6.3: The transverse momentum as it is obtained analysing from 1000 events.

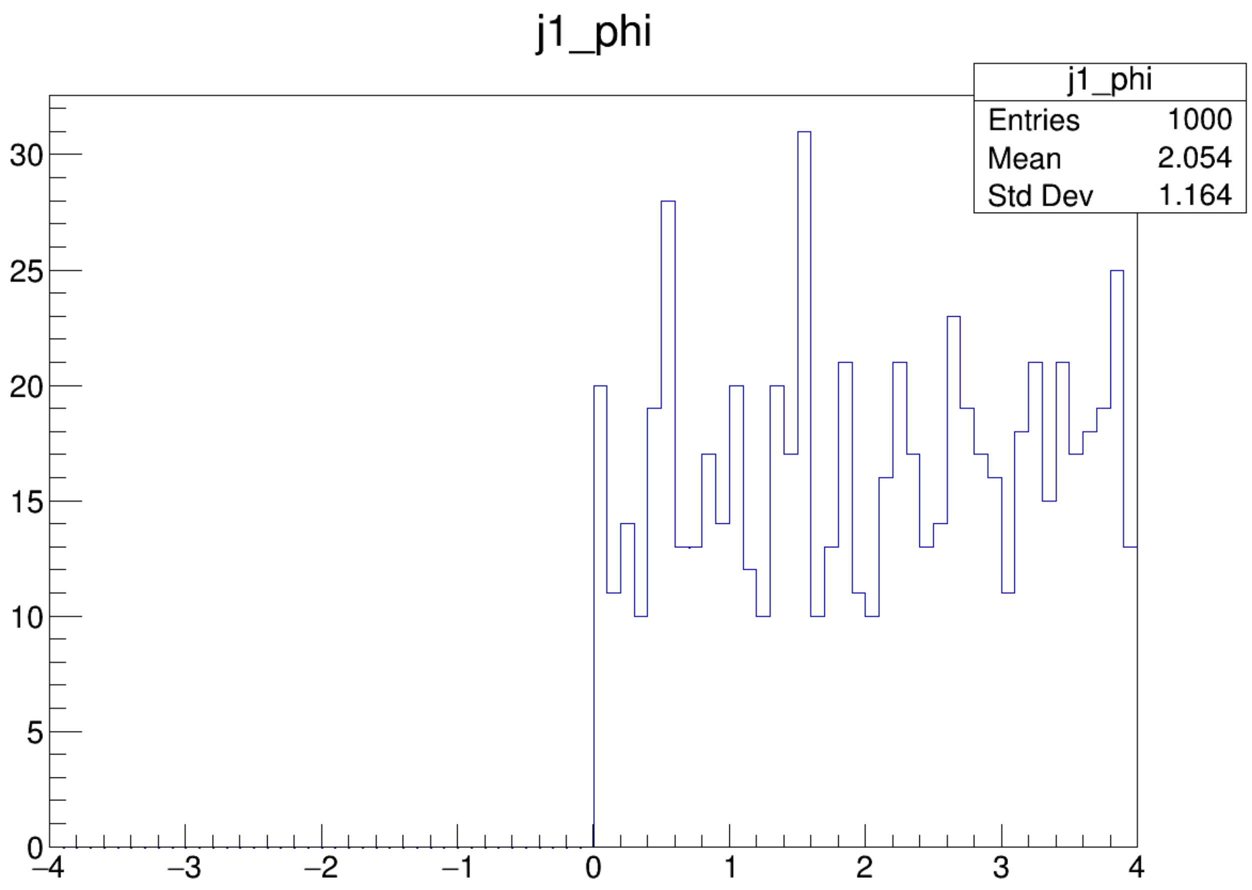


Figure 6.4: The Azimuth distance.

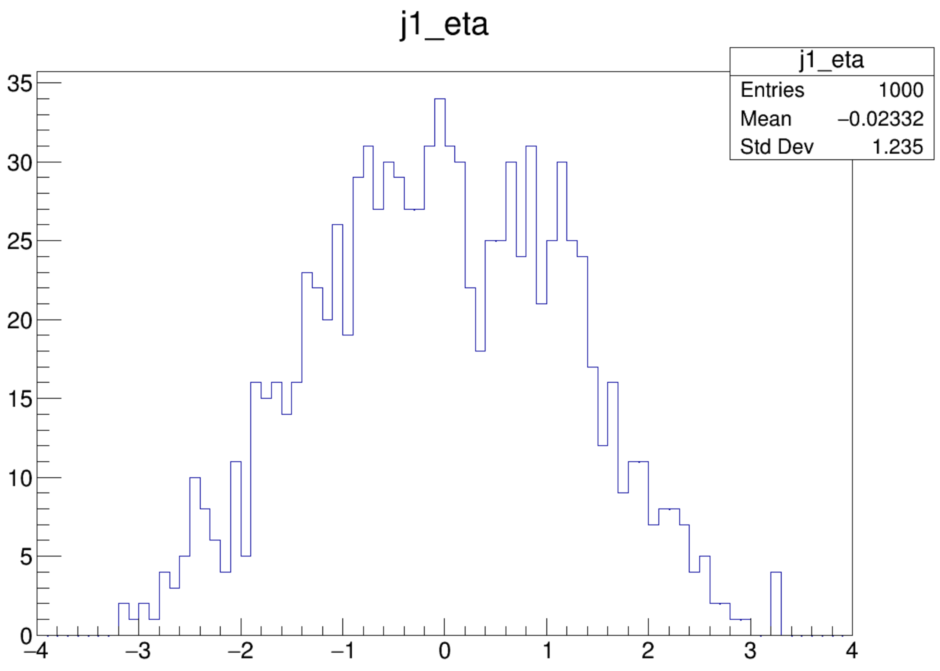


Figure 6.5: The pseudo-rapidity

References

Georges Aad et al. Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. *Phys. Lett.*, B716:1–29, 2012. doi: 10.1016/j.physletb.2012.08.020.

C. F. Berger et al. Snowmass 2001: Jet energy flow project. *eConf*, C010630:P512, 2001.

Gerald C. Blazey et al. Run II jet physics. In *QCD and weak boson physics in Run II. Proceedings, Batavia, USA, March 4-6, June 3-4, November 4-6, 1999*, pages 47–77, 2000. URL http://lss.fnal.gov/cgi-bin/find_paper.pl?conf-00-092.

Andy Buckley et al. General-purpose event generators for LHC physics. *Phys. Rept.*, 504:145–233, 2011. doi: 10.1016/j.physrep.2011.03.005.

Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. The Anti-k(t) jet clustering algorithm. *JHEP*, 04:063, 2008. doi: 10.1088/1126-6708/2008/04/063.

Matteo Cacciari, Gavin P. Salam, and Gregory Soyez. FastJet User Manual. *Eur. Phys. J.*, C72:1896, 2012. doi: 10.1140/epjc/s10052-012-1896-2.

Luc Devroye. Sample-based non-uniform random variate generation. In *Proceedings of the 18th Conference on Winter Simulation*, WSC '86, pages 260–265, New York, NY, USA, 1986. ACM. ISBN 0-911801-11-1. doi: 10.1145/318242.318443. URL http://doi.acm.org/10.1145/318242.318443.

S. D. Ellis, J. Huston, K. Hatakeyama, P. Loch, and M. Tonnesmann. Jets in hadron-hadron collisions. *Prog. Part. Nucl. Phys.*, 60:484–551, 2008. doi: 10.1016/j.ppnp.2007.12.002.

Stefan Höche. Introduction to parton-shower event generators. 2014. URL https://inspirehep.net/record/1328513/files/arXiv:1411.4085.pdf.

M.H. Kalos and P.A. Whitlock. *Monte Carlo Methods*. Monte Carlo Methods. Wiley, 1986. ISBN 9780471898399. URL https://books.google.rw/books?id=GyfvAAAAMAAJ.

Y. Nagashima and Y. Nambu. *Elementary Particle Physics: Quantum Field Theory and Particles*. Number v. 1. Wiley, 2010. ISBN 9783527630103. URL https://books.google.rw/books?id=J0l8s3pdOksC.

www.python.org Python foundation. Python reference manual. Technical report, The Us, 2007.

Gavin P. Salam. Towards Jetography. *Eur. Phys. J.*, C67:637–686, 2010a. doi: 10.1140/epjc/s10052-010-1314-6.

Gavin P. Salam. Elements of QCD for hadron colliders. In *High-energy physics. Proceedings, 17th European School, ESHEP 2009, Bautzen, Germany, June 14-27, 2009*, 2010b. URL https://inspirehep.net/record/880643/files/arXiv:1011.5131.pdf.

Stefan Weinzierl. Introduction to Monte Carlo methods. 2000.