Universidad Técnica Federico Santa María
Departamento de Física
Doctorado en Ciencias Físicas

# Simulation and analysis of the production of the Higgs boson in $pp$ collisions using `Pythia` and `ROOT`

Proyecto final Física Computacional

by

Sebastián Fuenzalida Garrido

Professors : Dr. Edson Carquin Lopez
Dr. Juan Manuel Flores Uribe

**VALPARAISO • CHILE**
**January 2021**

*Corta una flor en la Tierra, y moverás la estrella más lejana.*

Paul A.M. Dirac (1902-1984)

# Table of content

# List of Figures

# List of Tables

# Abstract

In this project, we analyse the SM decay processes $H \to \gamma\gamma$ and $H \to c\bar{c}$. In particular, in both decay channels we reconstruct the SM Higgs boson mass by using the `ROOT C++` framework and the `PYTHIA` event generator. In this way, we simulate $pp$ collision at the center-of-mass energy of $\sqrt{s} = 13$ TeV, which is the energy used by today in the LHC. In this way, by using a filter method we obtain via the simulation that the coupling of the Higgs with the third generation quarks $c\bar{c}$ is very weak in comparison with the coupling with the photons $\gamma$.

# Chapter 1

# Introduction and motivations

The standard model (SM) of particle physics [1–3] is a collection of experimental results that describes the behaviour of elementary particles. It describes accurately the interactions between them up to an energy scale of a few hundred GeV [4]. In the SM, the elementary particles or the building blocks of matter are the fermions, which are classified as quarks and leptons. These particles interact with each other through force carriers, which are the photon ($\gamma$), for the electromagnetic interaccions, the $W$ and $Z$ bosons, for the weak interactions, and the gluons ($g$), for the strong interactions. On the other hand, all of these particles acquire mass through their interaction with the Higgs field [5–13]. This is shown in the following table:

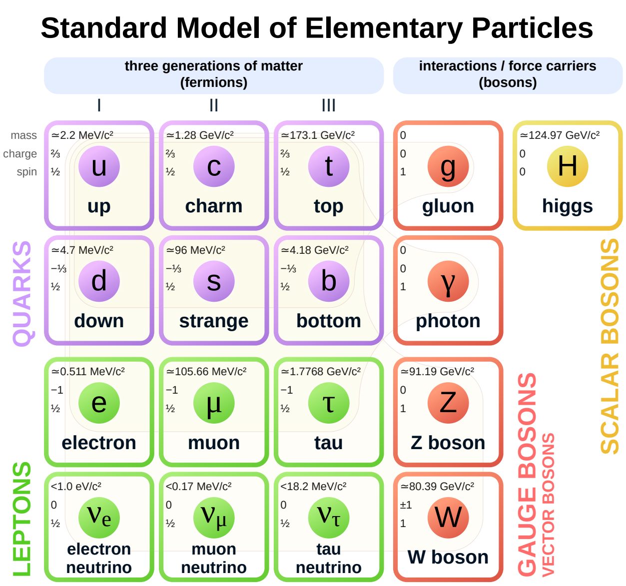


Figure 1.1: Standard Model of Elementary Particles. *Source: taken from* ***https://en.wikipedia.org/wiki/Standard_Model***.

The mechanism by which all the elementary particles acquire mass is the "Higgs" or "BEG" mechanism first proposed in the 1960's [5–10]. In this way, throught this mechanism the $W$ and $Z$ bosons acquire mass whilst the photon and the gluons remains massless, and adds to the model one new scalar particle, the SM Higgs boson. Furthermore, the Higgs field gives mass to the fermions, through the Yukawa interactions [11–13]. Moreover, is very important to note that the SM doesn't directly predict the value of the masses of the elementary particles, and due this there is no theoretical prediction for the mass of the Higgs boson. Rather the particle masses are considered parameters to be determined experimentally. Nevertheless, a numerous arguments were given to restrict the range of possibles values for the Higgs boson mass to bellow approximately 1 TeV. Additionally, the big number of data taken from the LEP and SLC colliders, the Tevatron, and other experiments predecited that the range of the mass of the Higgs boson go approximately from the 90 GeV to the uppper limit of 152 GeV at the 95% confidence level (CL) [4]. Furthermore, direct dearches at LEP excluded values lower than 114.4 GeV at 95% CL [14], and the measurements from the Tevatron excluded the mass range $162-166$ GeV at 95% CL [15].

The discovery or exclusion of the SM Higgs boson was one of the primary scientific goals of the Large Hadron Collider (LHC) when was under construction, which is the world's highest energy particle accelerator.

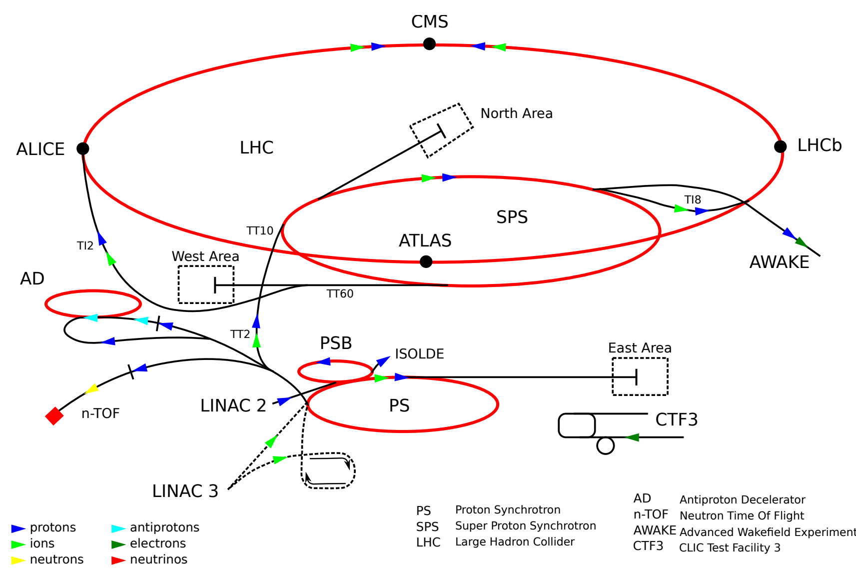


Figure 1.2: CERN accelerator complex. *Source: taken from* *https://en.wikipedia.org/wiki/CERN.*

The first direct searches at the LHC were based on data taken from proton-proton ($pp$) collision corresponding to an integrated luminosity of 5.1 fb$^{-1}$ collected at the center-of-mass energy of 7 TeV. In particular, the CMS experiment excluded the range of masses $127-600$ GeV at 95% CL [16], and the ATLAS colaboration excluded the ranges $111.4-116.4$ GeV, $119.4-122.1$ GeV, and $129.2-541$ GeV also at 95% CL [17]. Thereby, whitin the remaining allowed mass region, an excess of events between the 2 and 3 standard deviations ($\sigma$) near 125 GeV was reported by both experiments. Later, in 2012 the $pp$ center-of-mass energy was increased to 8 TeV, and by the end of June, an additional integrated luminosity of more than 5.3 fb$^{-1}$ had been recorded by both experiments, enhancing in this way significantly the sensitivity of the search of the Higgs boson. The final result was the observation by the ATLAS and CMS Collaborations of a new boson with a mass of approximately 125 GeV. In this way, the two Collaborations simultaneously published the final observation of the Higgs boson in the articles [18, 19]. In particular, these experiments focused on the observation in the five main decay channels: $H \to \gamma\gamma$, $H \to ZZ \to 4l$, $H \to WW \to l\nu l\nu$, $H \to \tau\tau$, and $H \to bb$, where $l$ is a lepton that can be a electron ($e^-$) or a muon ($\mu^-$), and for simplicity there is no distinction between particles and antiparticles. In addition, in the summer 2012 the analysis of the full data set by the CDF and D0 Collaboration resulted in an excess of events of about $3\sigma$ in the mass range $120-135$ GeV, while searching for a SM Higgs boson decaying into $b$ quarks [20].

Among the decay channels named above, the ones that have the highest sensitivity for discovering the SM Higgs boson with a mass near 125 GeV are $H \to \gamma\gamma$ and $H \to ZZ \to 4l$. The other three decay channels have poorer mass resolution and, in consequence needs more data to achieve a similar sensitivity. Among them, the decay channel $H \to WW \to l\nu l\nu$ has the largest signal-to-background ratio. Furthermore, these five channels are complementary in the way they are measured in the detector, as is the information they can provide about the SM Higgs boson.

In this project we will study the production of the Higgs boson by using the `Pythia` simulator, which is a general-purpose event generator for high-energy particle collisions written from scratch in `C++`. Furthermore, in order to make easy the analysis of the Higgs boson we will also use `ROOT`, which is an object-oriented framework aimed at solving the data analysis challenges of high-energy physics. In particular, we will simulate the production of the Higgs particle via the decays channels $H \to \gamma\gamma$ and $H \to c\bar{c}$ and we will try to reconstruct the mass of the Higgs by using the aforementioned tools.

This project is organized as follows: In chapter 2, we give a general overwiew of the `ROOT` framework and the `PYTHIA` simulator. Chapter 3 is devoted to analyse the decays channels $H \to \gamma\gamma$ and $H \to c\bar{c}$, which constitutes the principal results of this project. Finally, some conclusions and further comments are given in the chapter 4.

# Chapter 2

# ROOT and Pythia

## 2.1 ROOT

In the 1990's, René Brun and Fons Rademakers after many years of experienve developing interactive tools and simulations packages, they noticed that the PAW, PIAF, and GEANT libraries of FORTRAN had reached their limits. This due to the very big challenges offered by the LHC, where the data produced is a few orders of magnitude greater than anything seen before. Futhermore, in parallel the field of computer science had made a greater progress in the area of Object Oriented Desing, and René and Fons were ready to take advantage of it.

Under this context ROOT was born, which was developed under the context of the NA49 experiment at CERN. This was due to the fact that this experiment was generated an impresive amount of data, around 10 Terabytes per run. This was provided an arena to develop and test the next generation of data analysis.

The final result is that the physicist developed ROOT for themselves; this made it specific, appropiate, useful, and over time refined and very powerful. In this way, due to the powerful attributes of ROOT, this framework have finded its way in other fields such as medical and financial industries, among others.

### 2.1.1 The ROOT framework

ROOT is an object-oriented framework which is aimed to solve the challenges that arises in the field of high-energy physics (HEP) (particle and nuclear physics). Programing in the context of a framework is very like to living in a city. When we face the every day life in a city we use services such as plumbing, electricity outlets, and telephones. The details, for example, of how are the initimate details of the phone switching system aren't of interest for us. We aren't interested in these details; instead we are only interested in using the phone to communicate with your collaborators to solve your domain specific problems.

In sotware engineering the idea is very similar. In a framework, the basic utilities and services, such as input/output (I/O) and graphics are provided by the framework, or in other words we are not interested in the intimate details of such utilities, we simple use them to solve a particular problem. In this way, `ROOT` as a HEP analysis framework, have a lot of utilities such as histograms and fitting tools. When we stick to a framework to try to solve a problem we need to get used to it, in the sense that we need to learn the sintax associated with such framework. Similarly when we live in a city, to be able to use the services it offers to us we need in first intance to learn how to use such services in order to survive in such city.

The services or better said the more commonly used components offered by the `ROOT` framework are: command line interpreter, histogram and fitting, wrtting a graphical user interface, $2D$ graphics, I/O, collection classes, script processor, etc.

On the other hand, the benefits of working in a framework are:

- Less code to write.
- More reliable and robust code.
- More consistent and modular code.
- More focus on areas of expertise.

Besides, object-oriented programming can be defined as a model of programming based on the concept of objects. This objets contain data in the form of attributes and code in the form of methods. In object oriented programming, the algorithms are designed using the concept of objects that interact with the real world. Between the benefit of object-oriented programming we find:

- Encapsulation enforces data abstraction and increases opportunity for reuse.
- Sub classing and inheritance make it possible to extend and modify objects.
- Class hierarchies and containment containment hierarchies provide a flexible mechanism for modeling real-world objects and the relationships among them.
- Complexity is reduced because there is little growth of the global state, the state is contained within each object, rather than scattered through the program in the form of global variables.
- Objects may come and go, but the basic structure of the program remains relatively static, increases opportunity for reuse of design.

### 2.1.2 The organization of the `ROOT` framework

Now, having said in general terms what the `ROOT` framework is, let's look how the physical directories and files are organized after we install in our computer the `ROOT` framework (see the reference [21] or the detailed intructions of installation). In any case, we need an environment variable called `ROOTSYS`, which gives the path of the top `ROOT` directory. This can be obtained with the following commands in the terminal shell:

```
> echo $ROOTSYS
/opt/root
```

Thus, in the `ROOTSYS` directory are examples, executables, tutorial, header tutorials files, and, if we want to download it, we can get the source. The most important directories of special interest to us are `bin`, `tutorials`, `lib`, `test`, and `include`. The figure bellow shows the contents of these directories:

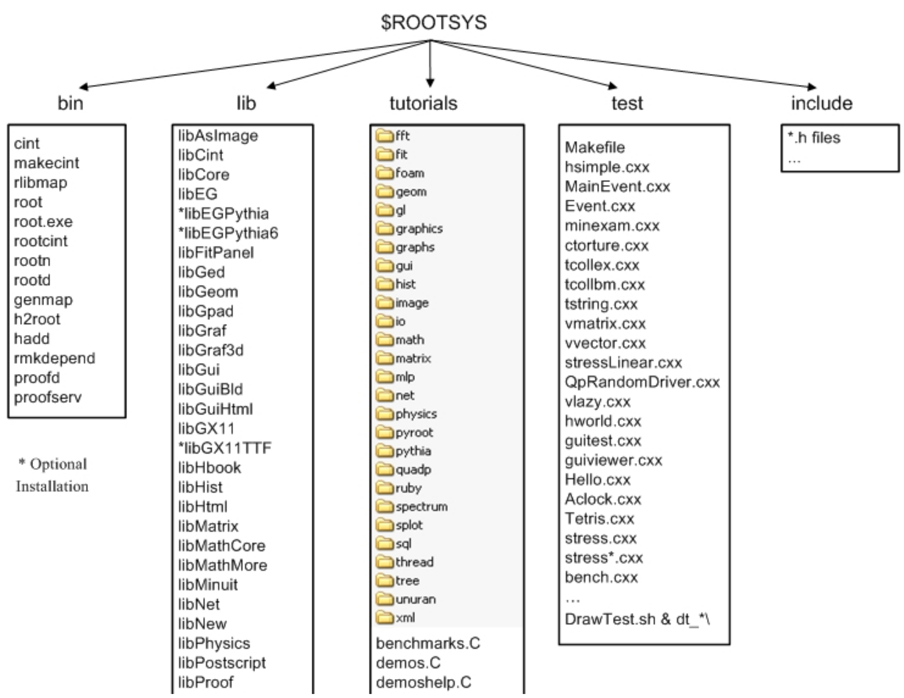


Figure 2.1: ROOT framework directories. *Source: taken from* ***https://root.cern/root/htmldoc/guides/users-guide/ROOTUsersGuide.html***.

Among the directories shown above let's focus on the `lib` directory. To obtain more information about the other directories see [21].

#### 2.1.2.1 `$ROOTSYS/lib`

In the `lib` directory we can find the libraries of the `ROOT` framework. Here is a short description of the most relevant libraries:

- `libCling` is the `C++` interpreter (`Cling`).
- `libGraf` is the $2D$ graphics primitives (can be used independent of `libGpad`).
- `libGraf3d` is the $3D$ graphics primitives.
- `libHist` is the histogram classes (with accompanying painter library).

- libMatrix is the matrix and vector manipulation.
- libMathCore contains the core mathematics and physics vector classes.
- libMinuit is the MINUIT fitter.
- libPhysics contains the legacy physics classes (TLorentzVector, etc.).
- libRIO provides the functionality to write and read objects to and from ROOT files.
- libRooFit is the RooFit fitting framework.

### 2.1.3 Histograms

Now, let's see the histogram classes of ROOT, which belong to the libHist library. This library will be the more used along this project. We have that ROOT support histograms up to three dimensions. Furthermore, separate concrete classes are provided or 1−dimensional, 2−dimensional and 3−dimensional classes. The histogram classes are split into further categories, depending on the set of possible bin values:

- TH1C, TH2C and TH3C contain one byte per bin (maximum bin content = 255)
- TH1S, TH2S and TH3S contain one short per bin (maximum bin content = 65.535).
- TH1I, TH2I and TH3I contain one integer per bin (maximum bin content = 2.147.483.647).
- TH1F, TH2F and TH3F contain one float per bin (maximum precision = 7 digits).
- TH1D, TH2D and TH3D contain one double per bin (maximum precision = 14 digits).

Besides ROOT supports profile histograms, which is a fancy way of replacing the 2−dimensional histograms in many cases. It is worth mentioning that the relation between two measured quantities $X$ and $Y$ can always be vizualized with a 2−dimensional histogram or scatter-plot. On the other hand, the profile histograms are used to display the mean value of $Y$ and it RMS for each bin in $X$. Additionally, we have the following classes:

- TProfile for 1−dimensional profiles.
- TProfile2D for 2−dimensional profiles.

All the aforementioned classes are organized in a hierarchy way as we see below:

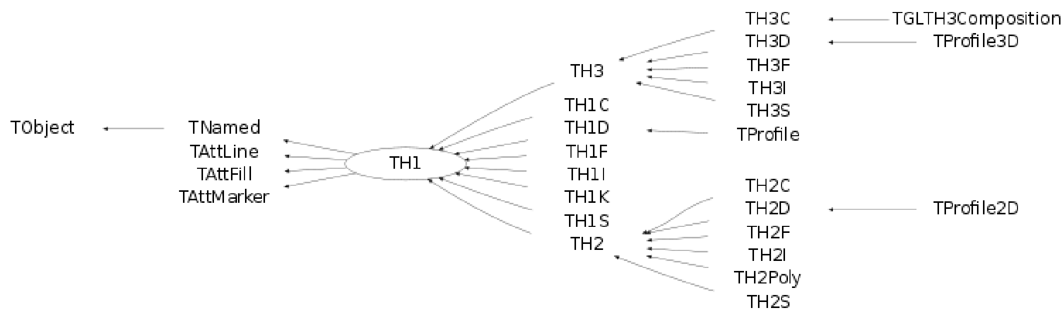


Figure 2.2: The class hierarchy of histogram classes. *Source: taken from* ***https://root.cern/root/htmldoc/guides/users-guide/ROOTUsersGuide.html*** *.*

From the diagram above we can see that all the `ROOT` histogram classes are derived from the base class `TH1`. This means that 2−dimensional and 3−dimensional histograms are seen as objects derived from a 1−dimensional histogram, in the same way that a multidimensional `C` array is just an abstraction of a 1−dimensional contiguous block of memory. The same holds for all the other `ROOT` clases which belong to the different libraries of such framework. For more information about other classes in root see the reference class documentation avalaible in [22].

Next we provide an example of an algorithm written in the `ROOT` framework that takes as input the convolution of two random number which are constructed from the class `TRandom3`. These two random numbers follows a gaussian and a exponential probability distribution. Then, we fill a histogram which belong to the `TH1F` class with these two random numbers, and we fit a function to the data of the histogram. The algorithm is shown bellow:

```
#include "TCanvas.h"
#include "TF1.h"
#include "TH1F.h"
#include "TLegend.h"
#include "TMath.h"

#include <iostream>
#include <math.h>

using namespace std;

void histogram_algorithm()
{
    Int_t number_of_simulations = 50000;

    // Gaussian distribution parameters
    Double_t mu = 0.0;
    Double_t sigma = 2.0;

    // Exponential distribution parameters
    Double_t tau = 3.0;

    TRandom3 r1, r2;
```

```
    vector<double> x1;

    for (Int_t i = 0; i < number_of_simulations; ++i)
    {
        x1.push_back(r1.Gaus(mu, sigma) + r2.Exp(tau));
    }

    auto *c2 = new TCanvas();
    c2 -> SetGrid();

    TF1 *func = new TF1("m1", "gaus", -4, 6);
    TH1F *histogram_convolution = new TH1F("Histogram convolution", "
    Histogram convolution", 100, -4, 6);
    histogram_convolution -> SetMarkerSize(2.0);
    histogram_convolution -> SetLineColor(kBlue);
    histogram_convolution -> SetFillColor(kBlue - 10);

    for (Int_t i = 0; i < number_of_simulations; ++i)
    {
        histogram_convolution -> Fill(x1[i]);
    }

    histogram_convolution -> GetXaxis() -> SetTitle("f(x)");
    histogram_convolution -> GetYaxis() -> SetTitle("Events");
    histogram_convolution -> GetXaxis() -> CenterTitle();
    histogram_convolution -> GetYaxis() -> CenterTitle();
    histogram_convolution -> Fit(func);
    c2 -> Draw();
    histogram_convolution -> Draw();

    c2 -> Print("histogram_convolution.pdf");
}
```

Consequently we get the following output histogram:

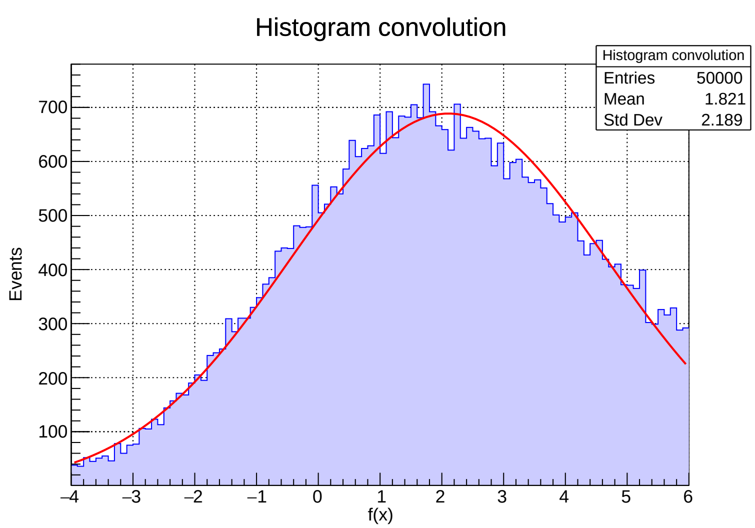


Figure 2.3: Output of the example `ROOT` algorithm for fitting a histogram. *Source: made by the author.*

The code showed previously will serve as a basic model for the analysis of the Higgs boson that we will make in this project, so at this point is very important to understand the syntax of the code. To obtain more details of the classes used in the code you can see the reference class documentation.

Finally, it is worth mentioning that in order to run a ROOT macro or script we need in first place that the name of the main function of our program have the same name as our file name. For example, if we create a macro with the name histogram_script.C, then the main function of our algorithm must be called as

```
void histogram_script()
{}
```

Having written our code we can run it via the terminal shell with any of the following commands:

```
> root histogram_script
> root histogram_script +g
```

where the first one run the macro by using the Cling interpreter and the second one in a compiled way.

## 2.2 PYTHIA

PYTHIA is a general-purpose event generator which serves as a tool for the analysis in high-energy particle collisions. This event generator or simulator was rewritten from scratch in the C++ language. It have implemented a lot of features, such as soft and hard processes, parton showers, multiparton interactions and string fragmentation.

More specifically, the PYTHIA program is a very powerful tool for the generation and simulation of high-energy collisions. The simulator focuses in collision with center-of-mass energies greater than about 10 GeV. Furthermore, PYTHIA have a coherent set of physics models for the evolution from a few-body high-energy ("hard´´) scattering process to a complex multihadronic final state. Also, the particles are produced in vaccum. However, it is possible to produce the particles with detector material effects via the interfacing with external detector simulation codes.

In addition, the PYTHIA package, in particular the 8th version, contains a library of hard interactions and models for initial- and final-state parton showers, multiple parton-parton interaction, beam remnants, string fragmentation and particle decays. It also has a set of utilities and interfaces to external programs.

The PYTHIA simulator, in particular the 8th version, is by today's standars, a small package. It's completely self-contained, and is therefore very easy to install for own use, e.g., if you want to use it on your own laptop, then yo can use it in a freely way without the concern of have a destructive interference between different libraries (see the reference [23] for the installation tutorial).

Finally, while `PYTHIA` can be run standalone, it is possible to interface it with a set of other libraries. An example is `HEPMC`, which is a standard tool used by experimentalist to store generated events. Furthermore, it is possible to link the `PYTHIA` generator with other libraries and frameworks such as: `ALPGEN`, `MADGRAPH`, `POWHEG`, `FASTJET`, `LHEF`, `LHAPDF`, `SLHA`, and of course `ROOT`. Thereby, in this proyect we will make the interface of `ROOT` and `PYTHIA` in order to make the analysis of the SM Higgs boson.

### 2.2.1 Particle properties

In the context of the `PYTHIA` simulator a particle corresponds to one entry/slot in the event record. The properties that are stored in this slot correspond to the ones belonging to a particle-as-such, like its identity code or $4-$momentum, and ones related to the event-as-a-whole, like which mother or daughter the particle has. It is important to say that the energies, momenta and masses are all given in GeV, and the space-time coordinates all in mm. In other words, the units are chosen such that the speed of light $c$ is the unity. In particular, times are also in mm, not in seconds.

The information that is stored for each particle is

- The identity code.
- The status code.
- Two mother indices.
- Two daughter indices.
- A colour and an anticoluor index.
- The four-momentum (energy and $3-$momentum), mass, angle $\phi$, pseudo-rapidity, rapidity.
- The scale at which the particle was produced (optional).
- The polarization/spin/helicity of the particle (optional).
- The production vertex and proper lifetime (optional).
- A pointer to the particle kind in the particle data table.
- A pointer to the event the particle belongs to.

In this way, from these properties a lot of other quantities can be derived.

On the other hand, `PYTHIA` have the following member functions which can be used to extract the most important information concerning each particle:

- `int Particle::id()` The identity code of a particle, acording to the PDG particle codes (below we will give a table with the most important identity codes).

- `int Particle::status()` The status code includes information about how each particle in the event record was produced, e.g., if it is a decay product of other particle (see [23] for more information).

- `int Particle::mother1()`, `int Particle::mother2()` The indices in the event record where the first and last mothers are stored.

- `int Particle::daughter1()`, `int Particle::daughter2()` The indices in the event record where the first and last daughter are stored.

- `double Particle::px()`, `double Particle::py()`, `double Particle::pz()`, `double Particle::e()` The particle 4−momentum components.

- `double Particle::m()` The particle mass, stored with a minus sign (times the absolute value) for spacelike virtual particles.

- `double Particle::pT()` The transverse momentum.

- `double Particle::eT()` The transverse energy.

- `double Particle::mT()` The transverse mass.

- `double Particle::theta()`, `double Particle::phi()` Polar and azimuthal angle.

- `double Particle::y()`, `double Particle::eta()` Rapidity and pseudorapidity.

- `string Particle::name()` The name of the particle.

Now, bellow we give a table with the identity codes of some particles according the de PDG standars:

| | | | | | |
|---|---|---|---|---|---|
| 1 | $d$ | 11 | $e^-$ | 21 | g |
| 2 | $u$ | 12 | $\nu_e$ | 22 | $\gamma$ |
| 3 | $s$ | 13 | $\mu^-$ | 23 | $Z^0$ |
| 4 | $c$ | 14 | $\nu_\mu$ | 24 | $W^+$ |
| 5 | $b$ | 15 | $\tau^-$ | 25 | $H^0$ |
| 6 | $t$ | 16 | $\nu_\tau$ | | |

Table 2.1: Table with the PDG ID codes of some particles. *Source: ID codes taken from [24].*

Furthermore, bellow we give a table with the meaning of the different status codes:

| code range | explanation |
|---|---|
| 11 - 19 | beam particles |
| 21 - 29 | particles of the hardest subprocess |
| 31 - 39 | particles of the subsequent subprocesses in multiparton interactions |
| 41 - 49 | particles produced by initial-state-showers |
| 51 - 59 | particles produced by final-state-showers |
| 61 - 69 | particles produced by beam-remnant process |
| 71 - 79 | partons in preparation of hadronization process |
| 81 - 89 | primary hadrons produced by hadronization process |
| 91 - 99 | particles produced in decay process, or by Bose-Eintein effects |

Table 2.2: Table with the status codes. *Source: taken from [23].*

### 2.2.2 A simple `PYTHIA` algorithm

Now we give an example of a simple algorithm written in the `ROOT` framework and also that makes use of the tools of the `PYTHIA` simulator, i.e., we make the link beetween the `PYTHIA` generator and the `ROOT` framework. In this algorithm we compute the mass of the electron $e^-$, the muon $\mu^-$, the $Z$ boson, and the $W$ boson. This masses are obtained from the member function `double Particle::m()`. In this way, by obtaining the masses of the previous particles with the aforementioned command we fill four histograms, which are contructed with the class `TH1F` of the `ROOT` framework. The algorithm is shown bellow:

```
// PYTHIA
#include "Pythia8/Pythia.h"

// ROOT, for histogramming.
#include "TH1.h"
#include "TCanvas.h"

// ROOT, for interactive graphics.
#include "TVirtualPad.h"
#include "TApplication.h"

// More ROOT packages.
#include "TLatex.h"

using namespace Pythia8;

void pythia_higgs_events_generator(Int_t number_of_events)
{
    TCanvas *canvas_1 = new TCanvas("canvas_1", "canvas_1", 10, 10,
    900, 600);
    canvas_1 -> Divide(2, 2);

    TH1F *m_e_histogram = new TH1F("Information", "e mass histogram",
    10, 0, 0.0009);
    m_e_histogram -> SetMarkerSize(2.0);
    m_e_histogram -> SetLineColor(kBlue);
```

```
    m_e_histogram -> SetFillColor(kBlue - 10);

    TH1F *m_mu_histogram = new TH1F("Information", "#mu mass histogram
", 100, 0, 0.2);
    m_mu_histogram -> SetMarkerSize(2.0);
    m_mu_histogram -> SetLineColor(kRed);
    m_mu_histogram -> SetFillColor(kRed - 10);

    TH1F *m_Z_histogram = new TH1F("Information", "Z mass histogram",
100, 70, 110);
    m_Z_histogram -> SetMarkerSize(2.0);
    m_Z_histogram -> SetLineColor(kOrange);
    m_Z_histogram -> SetFillColor(kOrange - 10);

    TH1F *m_W_histogram = new TH1F("Information", "W mass histogram",
100, 60, 100);
    m_W_histogram -> SetMarkerSize(2.0);
    m_W_histogram -> SetLineColor(kGreen);
    m_W_histogram -> SetFillColor(kGreen - 10);

    Pythia pythia;

    // Switch on processes:
    pythia.readString("HiggsSM:all = on");

    // Energy of the collitions: 13 [TeV] CM energy
    pythia.readString("Beams:eCM = 13000.");

    // How many events you want to see

    pythia.readString("Next:numberShowEvent = 2");
    pythia.init();

    for (int iEvent = 0; iEvent < number_of_events; ++iEvent) // Event
 loop.
    {
        pythia.next();

        Int_t ie = 0;
        Int_t imu = 0;
        Int_t iZ = 0;
        Int_t iW = 0;
        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle
loop.
        {

            if (pythia.event[i].id() == 11) // Fill electron mass
histogram
            {
                ie = i;
                m_e_histogram -> Fill(pythia.event[ie].m());
            }

            else if (pythia.event[i].id() == 13) // Fill muon mass
histogram
            {
                imu = i;
```

```
            m_mu_histogram -> Fill(pythia.event[imu].m());
        }

        else if (pythia.event[i].id() == 23) // Fill Z mass histogram
        {
            iZ = i;
            m_Z_histogram -> Fill(pythia.event[iZ].m());
        }

        else if (pythia.event[i].id() == 24) // Fill W mass histogram
        {
            iW = i;
            m_W_histogram -> Fill(pythia.event[iW].m());
        }
    }
}

canvas_1 -> cd(1);
canvas_1 -> Draw();
canvas_1 -> SetGrid();
m_e_histogram -> Draw();
m_e_histogram -> GetXaxis() -> SetTitle("e mass");
m_e_histogram -> GetXaxis() -> CenterTitle();
m_e_histogram -> GetYaxis() -> SetTitle("Events");
m_e_histogram -> GetYaxis() -> CenterTitle();

canvas_1 -> cd(2);
canvas_1 -> Draw();
canvas_1 -> SetGrid();
m_mu_histogram -> Draw();
m_mu_histogram -> GetXaxis() -> SetTitle("#mu mass");
m_mu_histogram -> GetXaxis() -> CenterTitle();
m_mu_histogram -> GetYaxis() -> SetTitle("Events");
m_mu_histogram -> GetYaxis() -> CenterTitle();

canvas_1 -> cd(3);
canvas_1 -> Draw();
canvas_1 -> SetGrid();
m_Z_histogram -> Draw();
m_Z_histogram -> GetXaxis() -> SetTitle("Z mass");
m_Z_histogram -> GetXaxis() -> CenterTitle();
m_Z_histogram -> GetYaxis() -> SetTitle("Events");
m_Z_histogram -> GetYaxis() -> CenterTitle();

canvas_1 -> cd(4);
canvas_1 -> Draw();
canvas_1 -> SetGrid();
m_W_histogram -> Draw();
m_W_histogram -> GetXaxis() -> SetTitle("W mass");
m_W_histogram -> GetXaxis() -> CenterTitle();
m_W_histogram -> GetYaxis() -> SetTitle("Events");
m_W_histogram -> GetYaxis() -> CenterTitle();

canvas_1 -> Print("mass_e_mu_Z_W_histogram.png");
```

```
    pythia.stat();

}

int main()
{
    Int_t simulated_events = 15000;

    pythia_higgs_events_generator(simulated_events);

    return 0;
}
```

In this way, the above algorithm deliver as output the following histograms:

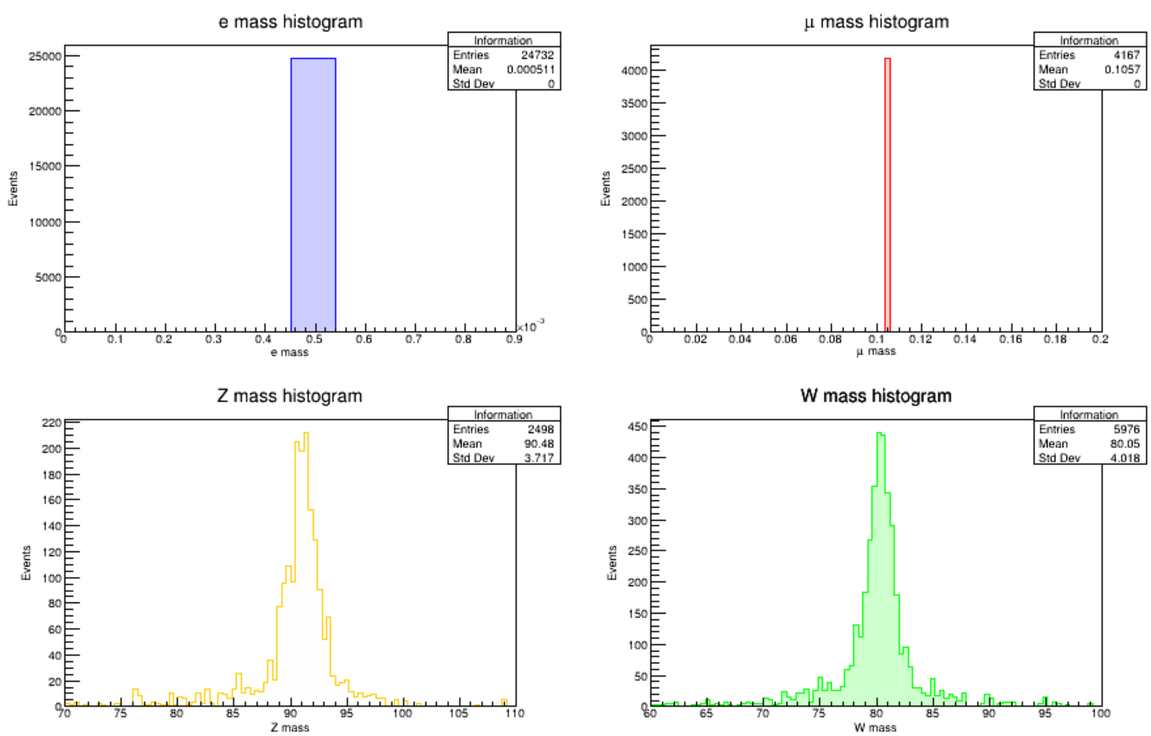


Figure 2.4: Output of the example PYTHIA algorithm for the $e^-$, $\mu^-$, $Z$ and $W$ masses. *Source: made by the author.*

On the other hand, in order to run the above algorithm we need to put the following orders in the terminal:

```
> make mymain02
> ./mymain02 > myout02
```

where we have choosen to put the name `mymain02.cc` to the previous algorithm. Also, in order to run the program we need to modify the `Makefile` located in `pythia8303/examples`. In particular, we need to modify the `ROOT` section (which makes possible the link between `PYTHIA` and `ROOT`) as follows:

```
# ROOT (turn off all warnings for readability).
main91 mymain02: $(PYTHIA) $$@.cc
ifeq ($(ROOT_USE),true)
	$(CXX) $@.cc -o $@ -w $(CXX_COMMON) $(ROOT_LIB)\
	 `$(ROOT_CONFIG) --cflags --glibs`
else
	$(error Error: $@ requires ROOT)
endif
```

In this way, when the terminal shell finish the compilation process the algorithm deliver also as output an event record with the name `myout01`. In this record we have information about the particles produced in each $pp$ colition. In particular, in the line 136 of the above `PYTHIA` algorithm we set the number of simulations or $pp$ colitions. Then, for each $pp$ colition we will have a number of particles produced after each collition, which will be stored in the section `Event Listing`. Futhermore, in the line 48 we fix the center-of-mass energy, which we fix as $\sqrt{s} = 13$ GeV, because this is the actual energy used in the LHC. Finally, in the line 45 we focus on the production of the SM Higgs boson, specifically we analyse the Higgs with all the subprocess production channels switched on. These channels correspond to:

- `HiggsSM:ffbar2H (default = off)` Scattering $f\bar{f} \to H^0$, where $f$ sums over available flavours exept top. Code 901.
- `HiggsSM:gg2H (default = off)` Scattering $gg \to H^0$ via loop contributions primarily from top. Code 902.
- `HiggsSM:gmgm2H (default = off)` Scattering $\gamma\gamma \to H^0$ via loop contributions primarily from top and $W$. Code 903.
- `HiggsSM:ffbar2HZ (default = off)` Scattering $f\bar{f} \to H^0 Z^0$ via $s-$channel $Z^0$ exchange. Code 904.
- `HiggsSM:ffbar2HW (default = off)` Scattering $f\bar{f} \to H^0 W^{\pm}$ via $s-$channel $W^{\pm}$ exchange. Code 905.
- `HiggsSM:ff2Hff(t:ZZ) (default = off)` Scattering $ff' \to H^0 ff'$ via $Z^0 Z^0$ fusion. Code 906.
- `HiggsSM:ff2Hff(t:WW) (default = off)` Scattering $f_1 f_2 \to H^0 f_3 f_4$ via $W^+W^-$ fusion. Code 907.
- `HiggsSM:gg2Httbar (default = off)` Scattering $gg \to H^0 t\bar{t}$ via $t\bar{t}$ fusion (or, alternatively put, Higgs radiation off a top line). Code 908.
- `HiggsSM:qqbar2Httbar (default = off)` Scattering $q\bar{q} \to H^0 t\bar{t}$ via $t\bar{t}$ fusion (or, alternatively put, Higgs radiation off a top line). Code 909.

  Finally, let's see how the event record `myout02` looks like. Indeed, after running the above `PYTHIA` algorithm we obtain the following output shown in the images below (to obtain more details about the information stored in the event record see the refenrece [23]):

```
*-------  PYTHIA Process Initialization  --------------------------*
|                                                                  |
| We collide p+ with p+ at a CM energy of 1.300e+04 GeV            |
|                                                                  |
|------------------------------------------------------------------|
|                                                    |             |
| Subprocess                                    Code |   Estimated |
|                                                    |    max (mb) |
|                                                    |             |
|------------------------------------------------------------------|
|                                                    |             |
| f fbar -> H (SM)                               901 |   9.030e-09 |
| g g -> H (SM)                                  902 |   2.524e-07 |
| gamma gamma -> H (SM)                          903 |   8.477e-12 |
| f fbar -> H0 Z0 (SM)                           904 |   5.187e-09 |
| f fbar -> H0 W+- (SM)                          905 |   9.431e-09 |
| f f' -> H0 f f'(Z0 Z0 fusion) (SM)             906 |   1.835e-08 |
| f_1 f_2 -> H0 f_3 f_4 (W+ W- fusion) (SM)      907 |   4.832e-08 |
| g g -> H t tbar (SM)                           908 |   5.102e-08 |
| q qbar -> H t tbar (SM)                        909 |   2.603e-08 |
|                                                                  |
*-------  End PYTHIA Process Initialization -----------------------*
```

Figure 2.5: `PYTHIA` Process Initialization. *Source: made by the author.*

```
 --------  PYTHIA Info Listing  ----------------------------------------

 Beam A: id =   2212, pz =  6.500e+03, e =  6.500e+03, m =  9.383e-01.
 Beam B: id =   2212, pz = -6.500e+03, e =  6.500e+03, m =  9.383e-01.

 In 1: id =   21, x =  1.508e-03, pdf =  2.813e+01 at Q2 =  1.562e+04.
 In 2: id =   21, x =  6.131e-02, pdf =  1.561e+00 at same Q2.

 Subprocess g g -> H (SM) with code 902 is 2 -> 1.
 It has sHat =  1.562e+04.
     alphaEM =  7.846e-03,  alphaS =  1.238e-01    at Q2 =  1.562e+04.

 Impact parameter b =  6.777e-01 gives enhancement factor =  1.643e+00.
 Max pT scale for MPI =  1.300e+04, ISR =  1.300e+04, FSR =  1.300e+04.
 Number of MPI =     5, ISR =    12, FSRproc =    54, FSRreson =    12.

 --------  End PYTHIA Info Listing  ------------------------------------
```

Figure 2.6: `PYTHIA` Info Listing. *Source: made by the author.*

```
 --------  PYTHIA Event Listing  (hard process)  -----------------------------------------------------------------

    no         id  name            status     mothers   daughters     colours      p_x        p_y        p_z         e          m
     0         90  (system)           -11     0     0     0     0     0     0      0.000      0.000      0.000  13000.000  13000.000
     1       2212  (p+)               -12     0     0     3     0     0     0      0.000      0.000   6500.000   6500.000      0.938
     2       2212  (p+)               -12     0     0     4     0     0     0      0.000      0.000  -6500.000   6500.000      0.938
     3         21  (g)                -21     1     0     5     0   101   102      0.000      0.000      9.802      9.802      0.000
     4         21  (g)                -21     2     0     5     0   102   101      0.000      0.000   -398.531    398.531      0.000
     5         25  (h0)               -22     3     4     6     7     0     0      0.000      0.000   -388.730    408.333    124.999
     6          4  c                   23     5     0     0     0   103     0     49.441      8.740   -315.857    319.826      1.500
     7         -4  cbar                23     5     0     0     0     0   103    -49.441     -8.740    -72.873     88.507      1.500
                                   Charge sum:  0.000           Momentum sum:      0.000      0.000   -388.730    408.333    124.999

 --------  End PYTHIA Event Listing  -----------------------------------------------------------------------------
```

Figure 2.7: `PYTHIA` Event Listing (hard process). *Source: made by the author.*

```
 --------  PYTHIA Event Listing  (complete event)  ---------------------------------------------------------------------------------

    no         id  name            status     mothers   daughters     colours      p_x        p_y        p_z         e          m
     0         90  (system)           -11     0     0     0     0     0     0      0.000      0.000      0.000  13000.000  13000.000
     1       2212  (p+)               -12     0     0   270     0     0     0      0.000      0.000   6500.000   6500.000      0.938
     2       2212  (p+)               -12     0     0   271     0     0     0      0.000      0.000  -6500.000   6500.000      0.938
     3         21  (g)                -21     6     0     5     0   101   102      0.000      0.000      9.802      9.802      0.000
     4         21  (g)                -21     7     7     5     0   102   101      0.000      0.000   -398.531    398.531      0.000
     5         25  (h0)               -22     3     4     8     8     0     0      0.000      0.000   -388.730    408.333    124.999
     6         21  (g)                -41    10    10     9     3   101   104      0.000      0.000     38.246     38.246      0.000
     7         21  (g)                -42    11     0     4     4   102   101     -0.000     -0.000   -398.531    398.531      0.000
     8         25  (h0)               -44     5     5    12    12     0     0     44.676     28.554   -358.914    383.739    124.999
     9         21  (g)                -43     6     0    13    13   102   104    -44.676    -28.554     -1.371     53.039      0.000
    10         21  (g)                -42    15    15     6     6   101   104      0.000     -0.000     38.246     38.246      0.000
    11         21  (g)                -41    16     0    14     7   102   105     -0.000      0.000   -446.740    446.740      0.000
    12         25  (h0)               -44     8     8    17    17     0     0     44.664     40.774   -390.060    414.040    124.999
    13         21  (g)                -44     9     9    18    18   102   104    -44.677    -27.658     -6.023     52.889      0.000
    14         21  (g)                -43    11     0    19    19   101   105      0.013    -13.116    -12.411     18.057      0.000
    15         21  (g)                -42    21     0    10    10   101   104     -0.000     -0.000     38.246     38.246      0.000
    16         21  (g)                -41    22    22    20    11   106   105      0.000      0.000   -722.639    722.639      0.000
    17         25  (h0)               -44    12    12    23    23     0     0     37.479     31.810   -393.439    415.735    124.999
    18         21  (g)                -44    13    13    24    24   102   104    -45.203    -28.315     -5.647     53.637      0.000
    19         21  (g)                -44    14    14    25    25   101   105     -0.259    -13.455    -12.376     18.283      0.000
    20         21  (g)                -43    16     0    26    26   106   102      7.983      9.960   -272.931    273.229      0.000
    21         21  (g)                -41    34    34    27    15   107   104     -0.000      0.000     59.555     59.555      0.000
    22         21  (g)                -42    35     0    16    16   106   105     -0.000     -0.000   -722.639    722.639      0.000
    23         25  (h0)               -44    17    17    36    36     0     0     35.355     30.136   -386.750    409.095    124.999
    24         21  (g)                -44    18    18    37    37   102   104    -49.775    -31.919    -12.300     60.396      0.000
    25         21  (g)                -44    19    19    28    29   101   105     -0.822    -13.899    -13.412     19.333      0.000
    26         21  (g)                -44    20    20    39    39   106   102      7.955      9.938   -270.832    271.131      0.000
    27         21  (g)                -43    21     0    30    30   107   101      7.287      5.744     20.211     22.239      0.000
    28         21  (g)                -51    25     0    38    38   108   105     -0.160    -16.166     -4.996     16.921      0.000
```

Figure 2.8: `PYTHIA` Event Listing (complete event). *Source: made by the author.*

```
*-------  PYTHIA Error and Warning Messages Statistics  ----------------------------------------------------------*
|                                                                                                                 |
|  times   message                                                                                                |
|                                                                                                                 |
|     19   Error in Pythia::next: hadronLevel failed; try again                                                   |
|     15   Error in StringFragmentation::fragment: stuck in joining                                               |
|      4   Error in StringFragmentation::fragmentToJunction: caught in junction flavour loop                      |
|      2   Warning in HadronWidths::pickMasses: angular momentum and running widths not used                      |
|      4   Warning in PhaseSpace2to2tauyz::trialKin: maximum for cross section violated                           |
|      1   Warning in Pythia::check: energy-momentum not quite conserved                                          |
|      5   Warning in Pythia::check: not quite matched particle energy/momentum/mass                              |
|      2   Warning in SimpleSpaceShower::pT2nextQCD: small daughter PDF                                           |
|      4   Warning in SimpleSpaceShower::pT2nextQCD: weight above unity                                           |
|      7   Warning in SimpleTimeShower::findMEcorr: ME weight above PS one                                        |
|      1   Warning in SimpleTimeShower::pTnext: negative dipole mass.                                             |
|     71   Warning in StringFragmentation::fragmentToJunction: bad convergence junction rest frame                |
|                                                                                                                 |
*-------  End PYTHIA Error and Warning Messages Statistics  ------------------------------------------------------*
```

Figure 2.9: `PYTHIA` Error and Warning Messages Statistics. *Source: made by the author.*

# Chapter 3

# Reconstruction of Higgs boson mass using the $H \to \gamma\gamma$ and $H \to c\bar{c}$ decay channels

In this chapter we will report the results obtained in this project. The main objetive of this project is to try to recontruct the Higgs boson mass by using the decay channels $H \to \gamma\gamma$ and $H \to c\bar{c}$. Indeed, before to make the previous analysis let's do a simple analysis of the kinematical variables associated with the Higgs boson. By using the first algorithm of the appendix A we obtain that the transversal momentum $p_T$, the pseudo-rapidity $\eta$, the azimuthal angle $\phi$ and the energy $E$ of the Higgs boson behaves as follows:

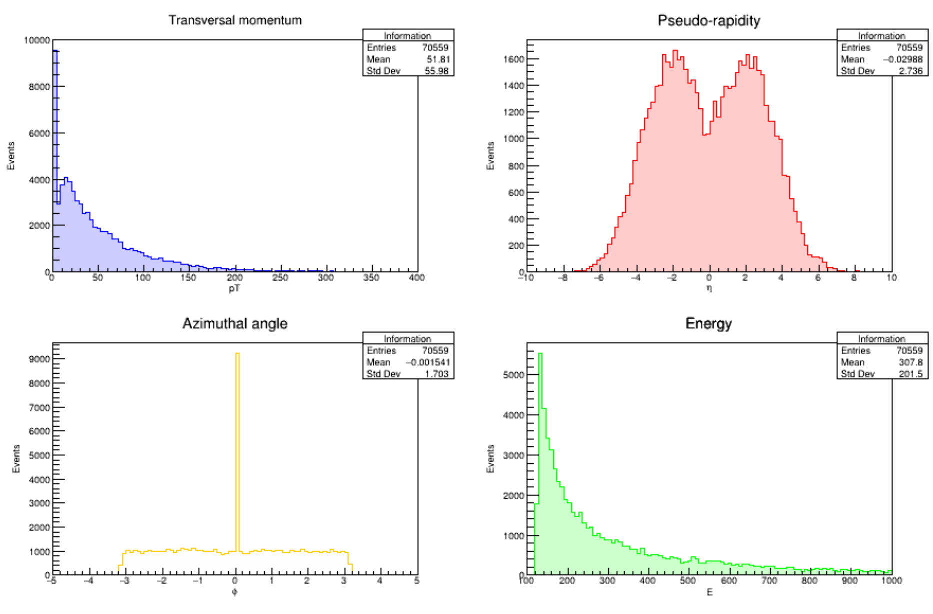


Figure 3.1: Histograms with the Higgs $p_T$, $\eta$, $\phi$ and $E$ (see the first algorithm of Appendix A). *Source: made by the author.*

Furthermore, from the same algorithm we can obtain the following behaviors for the rapidity $y$, the polar angle $\theta$, the transversal energy $E_T$ and the transversal mass $m_T$:

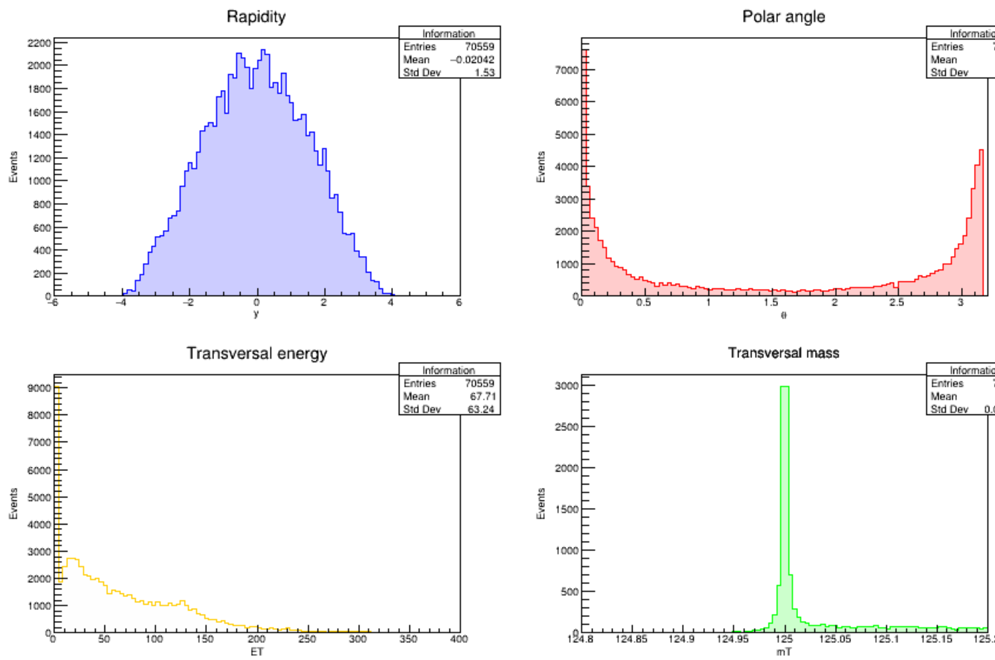


Figure 3.2: Histograms with the Higgs $y$, $\theta$, $E_T$ and $m_T$ (see the first algorithm of Appendix A). *Source: made by the author.*

In addition, from the aforementioned algorithm we can obtain the invariant mass of the SM Higgs boson, which is showed in the histogram bellow:

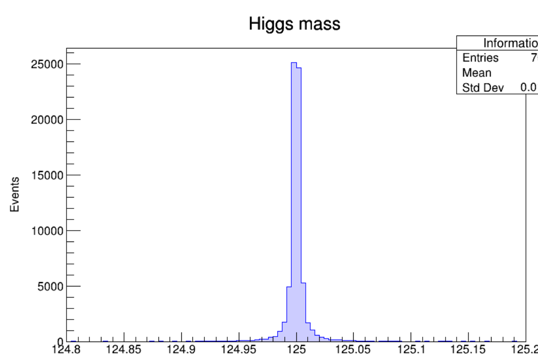


Figure 3.3: Histogram with the Higgs mass (see the first algorithm of Appendix A). *Source: made by the author.*

Furthermore, from the second algorithm of the appendix A we obtain an stacked histogram for the Higgs boson mass due to the different subprocess production of the Higgs boson, which as we said in the chapter 2 are the scattering processes $f\bar{f} \to H^0$, $gg \to H^0$, $\gamma\gamma \to H^0$, $f\bar{f} \to H^0 Z^0$, $f\bar{f} \to H^0 W^{\pm}$, $ff' \to H^0 ff'$, $f_1 f_2 \to H^0 f_3 f_4$,

$gg \to H^0 t\bar{t}$ and $q\bar{q} \to H^0 t\bar{t}$:

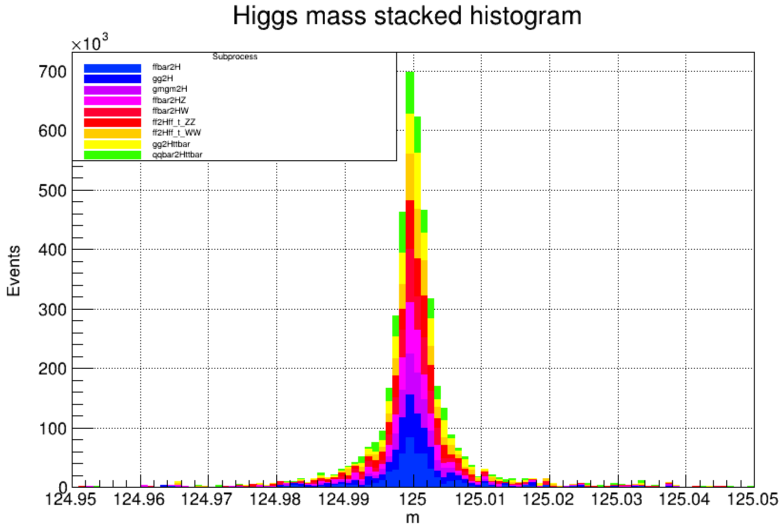


Figure 3.4: Stacked histogram with the Higgs mass due to different subprocess (see the second algorithm of Appendix A). *Source: made by the author.*

In this way, from the previous histograms we can get a general idea of the kinematical quantities associated with the SM Higgs boson. With these properties at hand we have a solid ground to make a more detail analysis. In our case the reconstruction of the Higgs boson mass due to the decay channels $H \to \gamma\gamma$. Furthermore, with the first and second algorithms given in the appendix A and this added to the algorithms seen in the chapter 2 we are ready to make our desired analysis. To obtain a complete review of the concepts of the kinematical variables seen above see the reference [25].

## 3.1 $H \to \gamma\gamma$ decay channel

As said in the reference [26] in the $H \to \gamma\gamma$ analysis a search is made for a narrow peak in the diphoton invariant mass distribution in the range $110-150$ GeV. Indeed, this diphoton invariant mass is given by the formula

$$m_{\gamma\gamma} = \sqrt{2p_{T,1}p_{T,2}\left(\cosh\left(\eta_1-\eta_2\right)-\cos\left(\phi_1-\phi_2\right)\right)}\ , \tag{3.1}$$

where $p_T$ is the transversal momentum, $\eta$ is the pseudo rapidity and $\phi$ is the azimuthal angle.

On the other hand, we have that the diphoton mass receives contribution from a large irreducible background from quantum chromodynamics QCD production of two photon. Furthermore, there is also a reducible background where one or more of the reconstructed photon candidates originate from misidentiication of jet fragments. In addition, the decay channel $H \to \gamma\gamma$ is one of the most promising channels in the search of a SM Higgs boson in the low-mass range [27].

Now, let's try to reconstruct the Higgs boson mass in the $H \to \gamma\gamma$ decay channel by using the `PYTHIA` simulator. In first place, let's see what are all the possibles daughters of the Higgs boson when we have all the scattering subprocess switched on. Indeed, by making the analysis of the possible daughter masses we obtain the following histogram:

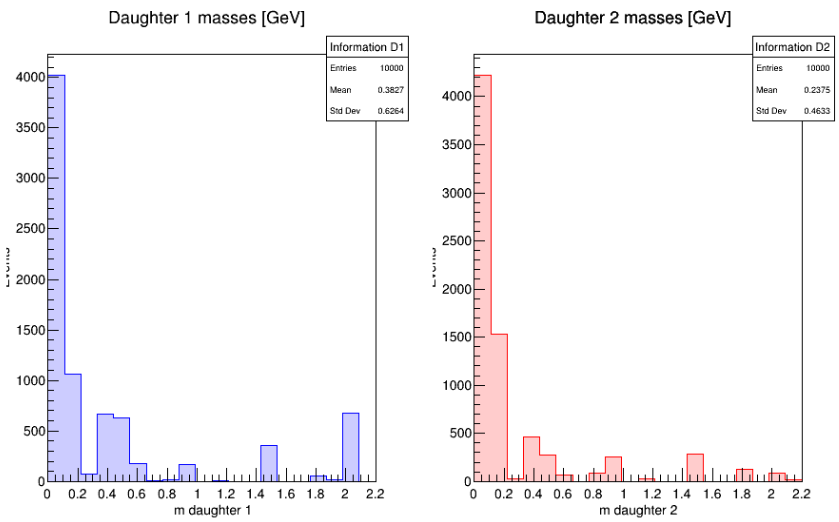


Figure 3.5: Histograms with the masses of the daughters one and two of the Higgs (see the third algorithm of Appendix A). *Source: made by the author.*

From the previous histogram we see that the majority of the events corresponds to daughters that are massless. From the SM we know that the massless boson are the gluon and the photon. In this way, we need to discriminate photons over the gluon background. Indeed, if we make such discrimination, then we obtain the following histogram for the ID of the photon and gluons:

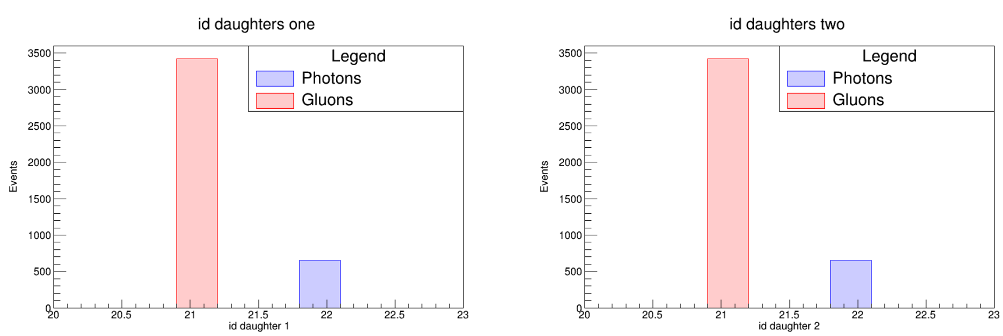


Figure 3.6: Histograms with the id of the photon and gluon daughters of the Higgs (see the third algorithm of Appendix A). *Source: made by the author.*

Clearly, we see that there is big prduction of the Higgs boson due to the decay channel $H \to gg$. However, we are interested only in the decay channel $H \to \gamma\gamma$. Therefore, by considering 10000 simulations and by using the diphoton invariant mass formula (3.1) we obtain the following histogram for the invariant mass of the Higgs boson:

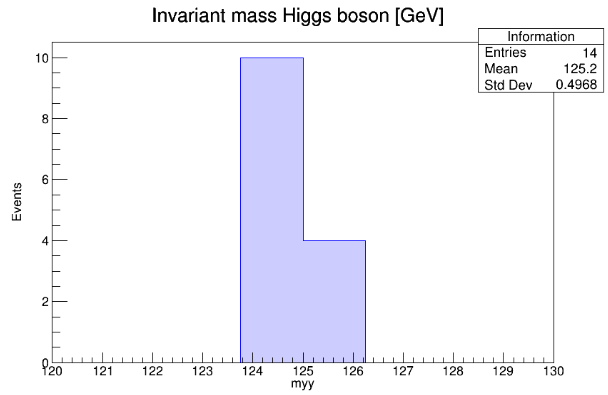


Figure 3.7: Histogram with the reconstructed Higgs mass (see the third algorithm of Appendix A). *Source: made by the author.*

where we have a little peak around the 125 GeV as expected.

## 3.2 $H \to c\bar{c}$ decay channel

The discovery of the Higgs boson, $H$, by the ATLAS and CMS experiments in the year 2012 represented a major step toward the characterisation of the electroweak symmetry breaking mechanism [5, 7, 8]. The mass of this particle is measured to be $m_H \sim 125$ GeV [28–30] and its decays in the $\gamma\gamma$, $ZZ$, $WW$, and $\tau\tau$ modes have been observed [31–40]. From the measured properties so far it can be concluded that this new particle is consistent with the expectations of the SM. Nevertheless, there remains much to be learned about the properties of this new particle. Indeed, one of the highest priorities of the LHC is the measurement of the coupling of the $H$ boson to other SM particles. Recently both ATLAS and CMS Collaborations reported the first direct measurements of the $H$ boson to third-generation quarks ($t$ and $b$) and found them to be also compatible with the SM prediction. Then, the next objetive is the measurement of the coupling of the the $H$ boson to second generation leptons ($\mu$ and $\nu_\mu$) and quarks ($c$ and $s$).

In this way, let's analyse the decay channel $H \to c\bar{c}$ by using the `PYTHIA` event generator. Again, by making a histogram of the daughter masses of the Higgs we get

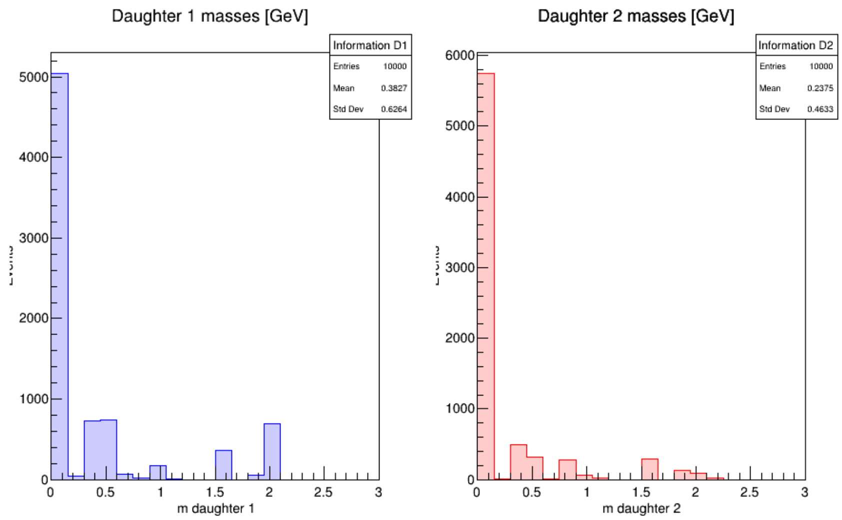


Figure 3.8: Histograms with the masses of the daughters one and two of the Higgs (see the fourth algorithm of Appendix A). *Source: made by the author.*

We see that there is a very low production of daughter particles between mass range $1 - 1.5$ GeV, in which the charm quark is located. This is expected because according to the SM the branching fraction of this process is $\mathcal{B}\left(H \to c\bar{c}\right) = 0.0288^{+0.0016}_{-0.0006}$ [41], which is factor of 20 smaller that that of $H \to b\bar{b}$, and also there is a very largebackground from SM processes comprised uniquely of jets produced through the strong interaction, which are know as QCD multijet events.

Indeed, returning to our analysis we get the following histogram for the ID codes of the quarks $c\bar{c}$ and the photon $\gamma$:

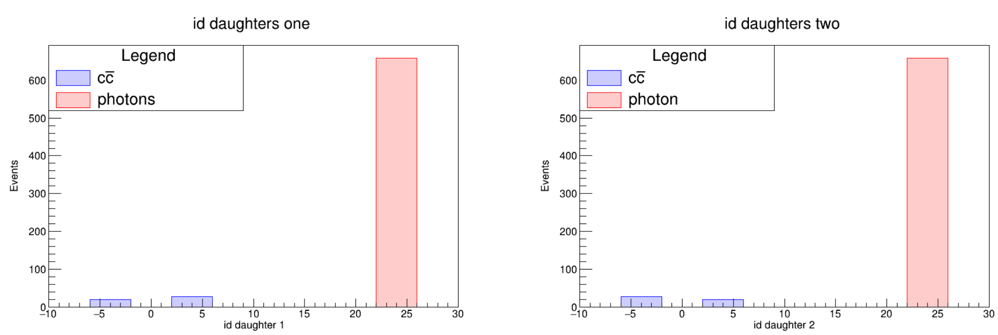


Figure 3.9: Histograms with the id of the photon and $c\bar{c}$ daughters of the Higgs (see the fourth algorithm of Appendix A). *Source: made by the author.*

Clearly we see that there is a very low production of the pair $c\bar{c}$ in comparison with the photon background. Therefore, the above histogram indirectly probes the weakness of the coupling between the Higgs and the second generation of quarks. Finally, by calculating the mass of the Higgs due to the decay $H \to$ we get the following histogram for the mass:

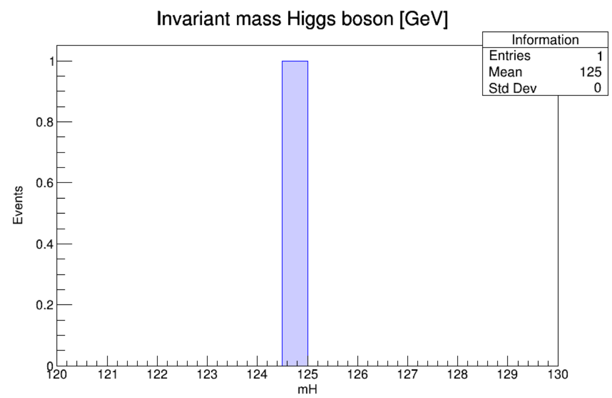


Figure 3.10: Histogram with the reconstructed Higgs mass (see the fourth algorithm of Appendix A). *Source: made by the author.*

where we have only one event when we consider 10000 simulations. This mass can be calculated with the formula:

$$M = \sqrt{m_1^2 + m_2^2 + 2\left(E_1 E_2 - \vec{p}_1 \cdot \vec{p}_2\right)}, \tag{3.2}$$

where $m_1$ and $m_2$ are masses of the daughter 1 and 2, $E_1$ and $E_2$ are the energies of the daughters 1 and 2, and $\vec{p}_1$ and $\vec{p}_2$ are the spatial momenta of the daughter 1 and 2.

# Chapter 4

# Conclusions and remarks

In this project we have tackled the problem of analyse the decays channels $H \to \gamma\gamma$ and $H \to c\bar{c}$. In particular, we have reconstructed the SM Higgs mass in these two decays channels obtaining a resonance around the mass $m \sim 125$ GeV. This resonances were reported in the histograms (3.7) for the decay $H \to \gamma\gamma$, and (3.10) for the decay $H \to c\bar{c}$. From these results we can conclude that the coupling of the SM Higgs boson with the photons is much greater than the coupling with the second generations quarks $c\bar{c}$, because from the simulation we have obtained 14 events where the Higgs were produced in the $H \to \gamma\gamma$ channel, and only one event for the $H \to c\bar{c}$ case. Furthermore, this simulations were made with 10000 $pp$ collisions, but clearly we can go further and make the simulation with more $pp$ collisions. However, the resources of my computer are not the optimal to make this number of simulations. In order to make a simulation with more $pp$ colission I can access to `LXPLUS` (Linux Public Login User Service), which is the interactive logon service to Linux for all CERN users. The cluster `LXPLUS` consists of public machines provided by the IT Departament for interactive work (see [43] for more details). Anyway, the results reported in this project are consistent with the ones reported in the articles [26, 42]. This is due to the fact that according to the SM the branching fraction for the decay $H \to c\bar{c}$ is very tiny in comparison with for example the branching fraction for the decay $H \to b\bar{b}$. This make the search of this process at the LHC a very challenging task.

Finally, I would like to thanks to the professors Edson Carquin and Juan Manuel Flores for the very enlightening lectures on Computational Physics imparted during this semester. This work is supported by Beca Interna del programa de doctorado de la Universidad Técnica Federico Santa María.

# Appendix A

# Code

## A.1 First code

```
#include <iostream>
#include <vector>
#include <string>
#include <stdio.h>

// PYTHIA
#include "Pythia8/Pythia.h"

// ROOT, for histogramming.
#include "TH1.h"
#include "TCanvas.h"
#include "THStack.h"

// ROOT, for interactive graphics.
#include "TVirtualPad.h"
#include "TApplication.h"

// ROOT, for saving file.
#include "TFile.h"

// More ROOT packages.
#include <TChain.h>
#include "TLatex.h"
#include <TMath.h>
#include <TLorentzVector.h>

using namespace Pythia8;

void pythia_higgs_events_generator(Int_t number_of_events)
{
    TCanvas *canvas_1 = new TCanvas("canvas_1", "canvas_1", 10, 10,
   900, 600);
    canvas_1 -> Divide(2, 2);

    TH1F *pT_histogram = new TH1F("Information", "Transversal momentum
   ", 100, 0, 400);
    pT_histogram -> SetMarkerSize(2.0);
    pT_histogram -> SetLineColor(kBlue);
```

```
    pT_histogram -> SetFillColor(kBlue - 10);

    TH1F *eta_histogram = new TH1F("Information", "Pseudo-rapidity",
    100, -10, 10);
    eta_histogram -> SetMarkerSize(2.0);
    eta_histogram -> SetLineColor(kRed);
    eta_histogram -> SetFillColor(kRed - 10);

    TH1F *phi_histogram = new TH1F("Information", "Azimuthal angle",
    100, -5, 5);
    phi_histogram -> SetMarkerSize(2.0);
    phi_histogram -> SetLineColor(kOrange);
    phi_histogram -> SetFillColor(kOrange - 10);

    TH1F *E_histogram = new TH1F("Information", "Energy", 100, 100,
    1000);
    E_histogram -> SetMarkerSize(2.0);
    E_histogram -> SetLineColor(kGreen);
    E_histogram -> SetFillColor(kGreen - 10);

    TCanvas *canvas_2 = new TCanvas("canvas_2", "canvas_2", 10, 10,
    900, 600);
    canvas_2 -> Divide(2, 2);

    TH1F *rapidity_histogram = new TH1F("Information", "Rapidity",
    100, -6, 6);
    rapidity_histogram -> SetMarkerSize(2.0);
    rapidity_histogram -> SetLineColor(kBlue);
    rapidity_histogram -> SetFillColor(kBlue - 10);

    TH1F *theta_histogram = new TH1F("Information", "Polar angle",
    100, 0, 3.2);
    theta_histogram -> SetMarkerSize(2.0);
    theta_histogram -> SetLineColor(kRed);
    theta_histogram -> SetFillColor(kRed - 10);

    TH1F *eT_histogram = new TH1F("Information", "Transversal energy",
     100, 0, 400);
    eT_histogram -> SetMarkerSize(2.0);
    eT_histogram -> SetLineColor(kOrange);
    eT_histogram -> SetFillColor(kOrange - 10);

    TH1F *mT_histogram = new TH1F("Information", "Transversal mass",
    100, 124.8, 125.2);
    mT_histogram -> SetMarkerSize(2.0);
    mT_histogram -> SetLineColor(kGreen);
    mT_histogram -> SetFillColor(kGreen - 10);

    TCanvas *canvas_3 = new TCanvas("canvas_3", "canvas_3", 10, 10,
    900, 600);
    canvas_3 -> Divide(1, 1);

    TH1F *invariant_mass_histogram = new TH1F("Information", "Higgs
    mass", 100, 124.8, 125.2);
    invariant_mass_histogram -> SetMarkerSize(2.0);
    invariant_mass_histogram -> SetLineColor(kBlue);
    invariant_mass_histogram -> SetFillColor(kBlue - 10);
```

```

    Pythia pythia;

    pythia.readString("Higgs:cubicWidth = on");
    pythia.readString("Higgs:runningLoopMass = on");
    pythia.readString("Higgs:clipWings = on");
    pythia.readString("Higgs:wingsFac = 50");
    pythia.readString("HiggsSM:NLOWidths = on");

    // Switch on processes:
    pythia.readString("HiggsSM:all = on");

    // Energy of the collitions: 13 [TeV] CM energy
    pythia.readString("Beams:eCM = 13000.");

    // How many events you want to see

    pythia.readString("Next:numberShowEvent = 2");
    pythia.init();

    for (int iEvent = 0; iEvent < number_of_events; ++iEvent) // Event loop.
    {
        pythia.next();

        Int_t iH = 0;
        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle loop.
        {

            if (pythia.event[i].id() == 25)
            {
                iH = i;
                pT_histogram -> Fill(pythia.event[iH].pT());
                eta_histogram -> Fill(pythia.event[iH].eta());
                phi_histogram -> Fill(pythia.event[iH].phi());
                E_histogram -> Fill(pythia.event[iH].e());
                rapidity_histogram -> Fill(pythia.event[iH].y());
                theta_histogram -> Fill(pythia.event[iH].theta());
                eT_histogram -> Fill(pythia.event[iH].eT());
                mT_histogram -> Fill(pythia.event[iH].mT());
                invariant_mass_histogram -> Fill(pythia.event[iH].m());
            }
        }
    }

    canvas_1 -> cd(1);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    pT_histogram -> Draw();
    pT_histogram -> GetXaxis() -> SetTitle("pT");
    pT_histogram -> GetXaxis() -> CenterTitle();
    pT_histogram -> GetYaxis() -> SetTitle("Events");
    pT_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> cd(2);
```

```
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    eta_histogram -> Draw();
    eta_histogram -> GetXaxis() -> SetTitle("#eta");
    eta_histogram -> GetXaxis() -> CenterTitle();
    eta_histogram -> GetYaxis() -> SetTitle("Events");
    eta_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> cd(3);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    phi_histogram -> Draw();
    phi_histogram -> GetXaxis() -> SetTitle("#phi");
    phi_histogram -> GetXaxis() -> CenterTitle();
    phi_histogram -> GetYaxis() -> SetTitle("Events");
    phi_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> cd(4);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    E_histogram -> Draw();
    E_histogram -> GetXaxis() -> SetTitle("E");
    E_histogram -> GetXaxis() -> CenterTitle();
    E_histogram -> GetYaxis() -> SetTitle("Events");
    E_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> Print("higgs_pt_eta_phi_e.png");

    canvas_2 -> cd(1);
    canvas_2 -> Draw();
    canvas_2 -> SetGrid();
    rapidity_histogram -> Draw();
    rapidity_histogram -> GetXaxis() -> SetTitle("y");
    rapidity_histogram -> GetXaxis() -> CenterTitle();
    rapidity_histogram -> GetYaxis() -> SetTitle("Events");
    rapidity_histogram -> GetYaxis() -> CenterTitle();

    canvas_2 -> cd(2);
    canvas_2 -> Draw();
    canvas_2 -> SetGrid();
    theta_histogram -> Draw();
    theta_histogram -> GetXaxis() -> SetTitle("#theta");
    theta_histogram -> GetXaxis() -> CenterTitle();
    theta_histogram -> GetYaxis() -> SetTitle("Events");
    theta_histogram -> GetYaxis() -> CenterTitle();

    canvas_2 -> cd(3);
    canvas_2 -> Draw();
    canvas_2 -> SetGrid();
    eT_histogram -> Draw();
    eT_histogram -> GetXaxis() -> SetTitle("ET");
    eT_histogram -> GetXaxis() -> CenterTitle();
    eT_histogram -> GetYaxis() -> SetTitle("Events");
    eT_histogram -> GetYaxis() -> CenterTitle();

    canvas_2 -> cd(4);
    canvas_2 -> Draw();
```

```
    canvas_2 -> SetGrid();
    mT_histogram -> Draw();
    mT_histogram -> GetXaxis() -> SetTitle("mT");
    mT_histogram -> GetXaxis() -> CenterTitle();
    mT_histogram -> GetYaxis() -> SetTitle("Events");
    mT_histogram -> GetYaxis() -> CenterTitle();

    canvas_2 -> Print("higgs_y_theta_et_mt.png");

    canvas_3 -> cd(1);
    canvas_3 -> Draw();
    canvas_3 -> SetGrid();
    invariant_mass_histogram -> Draw();
    invariant_mass_histogram -> Draw();
    invariant_mass_histogram -> GetXaxis() -> SetTitle("m");
    invariant_mass_histogram -> GetXaxis() -> CenterTitle();
    invariant_mass_histogram -> GetYaxis() -> SetTitle("Events");
    invariant_mass_histogram -> GetYaxis() -> CenterTitle();

    canvas_3 -> Print("higgs_invariant_mass.png");

    pythia.stat();
}

int main()
{
    Int_t simulated_events = 10000;

    pythia_higgs_events_generator(simulated_events);

    return 0;
}
```

## A.2 Second code

```
#include <iostream>
#include <vector>
#include <string>
#include <stdio.h>
#include <iomanip>

// PYTHIA
#include "Pythia8/Pythia.h"

// ROOT, for histogramming.
#include "TH1.h"
#include "TCanvas.h"
#include "THStack.h"
#include "TLegend.h"

// ROOT, for interactive graphics.
#include "TVirtualPad.h"
#include "TApplication.h"

// ROOT, for saving file.
#include "TFile.h"

```

```
// More ROOT packages.
#include <TChain.h>
#include "TLatex.h"
#include <TMath.h>
#include <TGraph.h>
#include <TLorentzVector.h>

using namespace Pythia8;
using namespace std;

void pythia_higgs_events_generator(Int_t number_of_events)
{
    Double_t a = 124.95;
    Double_t b = 125.05;
    Int_t bins = 10;

    TCanvas *canvas_1 = new TCanvas("canvas_1", "canvas_1", 10, 10,
    900, 600);
    THStack *hs = new THStack("Information", "Higgs mass stacked
    histogram");

    TH1F *histogram_1_mass_higgs = new TH1F("histogram_1_mass_higgs",
    "ffbar2H", bins, a, b);
    histogram_1_mass_higgs -> SetMarkerSize(2.0);
    histogram_1_mass_higgs -> SetLineColor(kAzure);
    histogram_1_mass_higgs -> SetFillColor(kAzure);

    TH1F *histogram_2_mass_higgs = new TH1F("histogram_2_mass_higgs",
    "gg2H", bins, a, b);
    histogram_2_mass_higgs -> SetMarkerSize(2.0);
    histogram_2_mass_higgs -> SetLineColor(kBlue);
    histogram_2_mass_higgs -> SetFillColor(kBlue);

    TH1F *histogram_3_mass_higgs = new TH1F("histogram_3_mass_higgs",
    "gmgm2H", bins, a, b);
    histogram_3_mass_higgs -> SetMarkerSize(2.0);
    histogram_3_mass_higgs -> SetLineColor(kViolet);
    histogram_3_mass_higgs -> SetFillColor(kViolet);

    TH1F *histogram_4_mass_higgs = new TH1F("histogram_4_mass_higgs",
    "ffbar2HZ", bins, a, b);
    histogram_4_mass_higgs -> SetMarkerSize(2.0);
    histogram_4_mass_higgs -> SetLineColor(kMagenta);
    histogram_4_mass_higgs -> SetFillColor(kMagenta);

    TH1F *histogram_5_mass_higgs = new TH1F("histogram_5_mass_higgs",
    "ffbar2HW", bins, a, b);
    histogram_5_mass_higgs -> SetMarkerSize(2.0);
    histogram_5_mass_higgs -> SetLineColor(kPink);
    histogram_5_mass_higgs -> SetFillColor(kPink);

    TH1F *histogram_6_mass_higgs = new TH1F("histogram_6_mass_higgs",
    "ff2Hff(t:ZZ)", bins, a, b);
    histogram_6_mass_higgs -> SetMarkerSize(2.0);
    histogram_6_mass_higgs -> SetLineColor(kRed);
    histogram_6_mass_higgs -> SetFillColor(kRed);

```

```
    TH1F *histogram_7_mass_higgs = new TH1F("histogram_7_mass_higgs",
"ff2Hff(t:WW)", bins, a, b);
    histogram_7_mass_higgs -> SetMarkerSize(2.0);
    histogram_7_mass_higgs -> SetLineColor(kOrange);
    histogram_7_mass_higgs -> SetFillColor(kOrange);

    TH1F *histogram_8_mass_higgs = new TH1F("histogram_8_mass_higgs",
"gg2Httbar", bins, a, b);
    histogram_8_mass_higgs -> SetMarkerSize(2.0);
    histogram_8_mass_higgs -> SetLineColor(kYellow);
    histogram_8_mass_higgs -> SetFillColor(kYellow);

    TH1F *histogram_9_mass_higgs = new TH1F("histogram_9_mass_higgs",
"qqbar2Httbar", bins, a, b);
    histogram_9_mass_higgs -> SetMarkerSize(2.0);
    histogram_9_mass_higgs -> SetLineColor(kSpring);
    histogram_9_mass_higgs -> SetFillColor(kSpring);

    vector<string> subprocess = {"HiggsSM:ffbar2H = on", "HiggsSM:gg2H
 = on", "HiggsSM:gmgm2H = on", "HiggsSM:ffbar2HZ = on", "HiggsSM:
ffbar2HW = on", "HiggsSM:ff2Hff(t:ZZ) = on", "HiggsSM:ff2Hff(t:WW)
= on", "HiggsSM:gg2Httbar = on", "HiggsSM:qqbar2Httbar = on"};
    Double_t matrix_with_higgs_masses[subprocess.size()][10000];

    vector<double> higgs_mass_ffbar2H;
    vector<double> higgs_mass_gg2H;
    vector<double> higgs_mass_gmgm2H;
    vector<double> higgs_mass_ffbar2HZ;
    vector<double> higgs_mass_ffbar2HW;
    vector<double> higgs_mass_ff2Hff_t_ZZ;
    vector<double> higgs_mass_ff2Hff_t_WW;
    vector<double> higgs_mass_gg2Httbar;
    vector<double> higgs_mass_qqbar2Httbar;

    vector<double> higgs_mass_ffbar2H_filter;
    vector<double> higgs_mass_gg2H_filter;
    vector<double> higgs_mass_gmgm2H_filter;
    vector<double> higgs_mass_ffbar2HZ_filter;
    vector<double> higgs_mass_ffbar2HW_filter;
    vector<double> higgs_mass_ff2Hff_t_ZZ_filter;
    vector<double> higgs_mass_ff2Hff_t_WW_filter;
    vector<double> higgs_mass_gg2Httbar_filter;
    vector<double> higgs_mass_qqbar2Httbar_filter;

    for(Int_t ii = 0; ii < subprocess.size(); ++ii)
    {
        Pythia pythia;

        pythia.readString(subprocess[ii]);
        pythia.readString("Beams:eCM = 13000.");
        pythia.readString("Next:numberShowEvent = 0");
        pythia.init();

        vector<double> temporal_vector_for_counters_mass_higgs;

        for (Int_t iEvent = 0; iEvent < number_of_events; ++iEvent)
        {
```

```
            pythia.next();

            Int_t iH = 0;
            Int_t iH2 = 0;
            for (Int_t i = 0; i < pythia.event.size(); ++i)
            {
                if (pythia.event[i].id() == 25)
                {
                    iH = i;
                    temporal_vector_for_counters_mass_higgs.push_back(pythia.event[iH].m());
                    matrix_with_higgs_masses[ii][iH] = pythia.event[iH].m();
                }

                if (pythia.event[i].id() == 25)
                {
                    iH2 = i;
                    matrix_with_higgs_masses[ii][i] = pythia.event[iH2].m();
                }
            }
        }

        cout << "Number of Higgs boson produced = " << temporal_vector_for_counters_mass_higgs.size() << endl;

        temporal_vector_for_counters_mass_higgs.clear();
    }

    for (Int_t i = 0; i < 10000; ++i)
    {
        higgs_mass_ffbar2H.push_back(matrix_with_higgs_masses[0][i]);
        higgs_mass_gg2H.push_back(matrix_with_higgs_masses[1][i]);
        higgs_mass_gmgm2H.push_back(matrix_with_higgs_masses[2][i]);
        higgs_mass_ffbar2HZ.push_back(matrix_with_higgs_masses[3][i]);
        higgs_mass_ffbar2HW.push_back(matrix_with_higgs_masses[4][i]);
        higgs_mass_ff2Hff_t_ZZ.push_back(matrix_with_higgs_masses[5][i]);
        higgs_mass_ff2Hff_t_WW.push_back(matrix_with_higgs_masses[6][i]);
        higgs_mass_gg2Httbar.push_back(matrix_with_higgs_masses[7][i]);
        higgs_mass_qqbar2Httbar.push_back(matrix_with_higgs_masses[8][i]);
    }

    for (Int_t i = 0; i < higgs_mass_ffbar2H.size(); ++i)
    {
        histogram_1_mass_higgs -> Fill(higgs_mass_ffbar2H[i]);
        hs -> Add(histogram_1_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_gg2H.size(); ++i)
    {
        histogram_2_mass_higgs -> Fill(higgs_mass_gg2H[i]);
        hs -> Add(histogram_2_mass_higgs);
```

```
    }

    for (Int_t i = 0; i < higgs_mass_gmgm2H.size(); ++i)
    {
        histogram_3_mass_higgs -> Fill(higgs_mass_gmgm2H[i]);
        hs -> Add(histogram_3_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_ffbar2HZ.size(); ++i)
    {
        histogram_4_mass_higgs -> Fill(higgs_mass_ffbar2HZ[i]);
        hs -> Add(histogram_4_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_ffbar2HW.size(); ++i)
    {
        histogram_5_mass_higgs -> Fill(higgs_mass_ffbar2HW[i]);
        hs -> Add(histogram_5_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_ff2Hff_t_ZZ.size(); ++i)
    {
        histogram_6_mass_higgs -> Fill(higgs_mass_ff2Hff_t_ZZ[i]);
        hs -> Add(histogram_6_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_ff2Hff_t_WW.size(); ++i)
    {
        histogram_7_mass_higgs -> Fill(higgs_mass_ff2Hff_t_WW[i]);
        hs -> Add(histogram_7_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_gg2Httbar.size(); ++i)
    {
        histogram_8_mass_higgs -> Fill(higgs_mass_gg2Httbar[i]);
        hs -> Add(histogram_8_mass_higgs);
    }

    for (Int_t i = 0; i < higgs_mass_qqbar2Httbar.size(); ++i)
    {
        histogram_9_mass_higgs -> Fill(higgs_mass_qqbar2Httbar[i]);
        hs -> Add(histogram_9_mass_higgs);
    }

    canvas_1 -> SetGrid();
    canvas_1 -> Draw("nostackb");
    hs -> Draw();
    hs -> GetXaxis() -> SetTitle("m");
    hs -> GetXaxis() -> CenterTitle();
    hs -> GetYaxis() -> SetTitle("Events");
    hs -> GetYaxis() -> CenterTitle();

    auto legend = new TLegend(0.9, 0.7, 0.48, 0.9);
    legend -> SetHeader("Subprocess", "C");
    legend -> AddEntry(histogram_1_mass_higgs, "ffbar2H", "f");
    legend -> AddEntry(histogram_2_mass_higgs, "gg2H", "f");
    legend -> AddEntry(histogram_3_mass_higgs, "gmgm2H", "f");
```

```
    legend -> AddEntry(histogram_4_mass_higgs, "ffbar2HZ", "f");
    legend -> AddEntry(histogram_5_mass_higgs, "ffbar2HW", "f");
    legend -> AddEntry(histogram_6_mass_higgs, "ff2Hff_t_ZZ", "f");
    legend -> AddEntry(histogram_7_mass_higgs, "ff2Hff_t_WW", "f");
    legend -> AddEntry(histogram_8_mass_higgs, "gg2Httbar", "f");
    legend -> AddEntry(histogram_9_mass_higgs, "qqbar2Httbar", "f");
    legend -> Draw();

    canvas_1 -> Print("higgs_mass_stacked_histogram.png");
}

int main()
{
    Int_t simulated_events = 10000;

    pythia_higgs_events_generator(simulated_events);

    return 0;
}
```

## A.3 Third code

```
#include <iostream>
#include <vector>
#include <string>
#include <stdio.h>
#include <iomanip>

// PYTHIA
#include "Pythia8/Pythia.h"

// ROOT, for histogramming.
#include "TH1.h"
#include "TCanvas.h"
#include "THStack.h"
#include "TLegend.h"

// ROOT, for interactive graphics.
#include "TVirtualPad.h"
#include "TApplication.h"

// ROOT, for saving file.
#include "TFile.h"

// More ROOT packages.
#include <TChain.h>
#include "TLatex.h"
#include <TMath.h>
#include <TGraph.h>
#include <TLorentzVector.h>

using namespace Pythia8;

void pythia_higgs_events_generator(Int_t number_of_events)
{
    vector<int> loc_vector;
    vector<int> id_vector;
```

```
    vector<int> loc_id_daughter_1_vector;
    vector<int> loc_id_daughter_2_vector;
    vector<double> mass_vector;
    vector<double> pt_vector;
    vector<double> eta_vector;
    vector<double> phi_vector;
    vector<double> E_vector;
    vector<string> names_vector;

    int id_daughter_1_vector_filter[number_of_events];
    int id_daughter_2_vector_filter[number_of_events];
    double mass_vector_daughter_1_filter[number_of_events];
    double mass_vector_daughter_2_filter[number_of_events];
    double pt_vector_daughter_1_filter[number_of_events];
    double pt_vector_daughter_2_filter[number_of_events];
    double eta_vector_daughter_1_filter[number_of_events];
    double eta_vector_daughter_2_filter[number_of_events];
    double phi_vector_daughter_1_filter[number_of_events];
    double phi_vector_daughter_2_filter[number_of_events];
    double E_vector_daughter_1_filter[number_of_events];
    double E_vector_daughter_2_filter[number_of_events];

    vector<int> id_photon_vector_daughter_1;
    vector<int> id_photon_vector_daughter_2;
    vector<int> id_photon_vector_daughter_1_filter;
    vector<int> id_photon_vector_daughter_2_filter;

    vector<int> id_gluon_vector_daughter_1;
    vector<int> id_gluon_vector_daughter_2;
    vector<int> id_gluon_vector_daughter_1_filter;
    vector<int> id_gluon_vector_daughter_2_filter;

    vector<double> pt_photon_1;
    vector<double> pt_photon_2;
    vector<double> eta_photon_1;
    vector<double> eta_photon_2;
    vector<double> phi_photon_1;
    vector<double> phi_photon_2;
    vector<double> E_photon_1;
    vector<double> E_photon_2;

    vector<double> m_yy;
    vector<double> m_yy_filter;

    TCanvas *canvas_1 = new TCanvas("canvas_1", "canvas_1", 10, 10,
   900, 600);
    canvas_1 -> Divide(2, 1);
    TH1F *invariant_mass_daughter_1_histogram = new TH1F("Information
   D1", "Daughter 1 masses [GeV]", 20, 0, 2.2);
    invariant_mass_daughter_1_histogram -> SetMarkerSize(2.0);
    invariant_mass_daughter_1_histogram -> SetLineColor(kBlue);
    invariant_mass_daughter_1_histogram -> SetFillColor(kBlue - 10);

    TH1F *invariant_mass_daughter_2_histogram = new TH1F("Information
   D2", "Daughter 2 masses [GeV]", 20, 0, 2.2);
    invariant_mass_daughter_2_histogram -> SetMarkerSize(2.0);
    invariant_mass_daughter_2_histogram -> SetLineColor(kRed);
```

```
    invariant_mass_daughter_2_histogram -> SetFillColor(kRed - 10);

    TCanvas *canvas_2_1 = new TCanvas("canvas_2_1", "canvas_2_1", 10,
    10, 900, 600);
    canvas_2_1 -> Divide(1, 1);
    THStack *hs_1 = new THStack("hs_1", "id daughters one");
    TH1F *id_daughter_1_histogram_photon = new TH1F("Information D1 #
    gamma", "id daughter 1 photon", 10, 20, 23);
    id_daughter_1_histogram_photon -> SetMarkerSize(2.0);
    id_daughter_1_histogram_photon -> SetLineColor(kBlue);
    id_daughter_1_histogram_photon -> SetFillColor(kBlue - 10);
    TH1F *id_daughter_1_histogram_gluon = new TH1F("Information D1 g",
     "id daughter 1 gluon", 10, 20, 23);
    id_daughter_1_histogram_gluon -> SetMarkerSize(2.0);
    id_daughter_1_histogram_gluon -> SetLineColor(kRed);
    id_daughter_1_histogram_gluon -> SetFillColor(kRed - 10);

    TCanvas *canvas_2_2 = new TCanvas("canvas_2_2", "canvas_2_2", 10,
    10, 900, 600);
    canvas_2_2 -> Divide(1, 1);
    THStack *hs_2 = new THStack("hs_1", "id daughters two");
    TH1F *id_daughter_2_histogram_photon = new TH1F("Information D2 #
    gamma", "id daughter 2 photon", 10, 20, 23);
    id_daughter_2_histogram_photon -> SetMarkerSize(2.0);
    id_daughter_2_histogram_photon -> SetLineColor(kBlue);
    id_daughter_2_histogram_photon -> SetFillColor(kBlue - 10);
    TH1F *id_daughter_2_histogram_gluon = new TH1F("Information D2 g",
     "id daughter 2 gluon", 10, 20, 23);
    id_daughter_2_histogram_gluon -> SetMarkerSize(2.0);
    id_daughter_2_histogram_gluon -> SetLineColor(kRed);
    id_daughter_2_histogram_gluon -> SetFillColor(kRed - 10);

    TCanvas *canvas_3 = new TCanvas("canvas_3", "canvas_3", 10, 10,
    900, 600);
    canvas_3 -> Divide(1, 1);
    TH1F *invariant_mass_higgs = new TH1F("Information", "Invariant
    mass Higgs boson [GeV]", 8, 120, 130);
    invariant_mass_higgs -> SetMarkerSize(2.0);
    invariant_mass_higgs -> SetLineColor(kBlue);
    invariant_mass_higgs -> SetFillColor(kBlue - 10);

    Pythia pythia;

    pythia.readString("Higgs:cubicWidth = on");
    pythia.readString("Higgs:runningLoopMass = on");
    pythia.readString("Higgs:clipWings = on");
    pythia.readString("Higgs:wingsFac = 50");
    pythia.readString("HiggsSM:NLOWidths = on");

    // Switch on processes:
    pythia.readString("HiggsSM:all = on");

    // Energy of the collitions: 13 [TeV] CM energy
    pythia.readString("Beams:eCM = 13000.");

    // How many events you want to see
    pythia.readString("Next:numberShowEvent = 0");
```

```
    pythia.init();

    for (Int_t iEvent = 0; iEvent < number_of_events; ++iEvent) // Event loop.
    {
        pythia.next();

        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle loop.
        {
            loc_vector.push_back(i);
            id_vector.push_back(pythia.event[i].id());
            loc_id_daughter_1_vector.push_back(pythia.event[i].daughter1());
            loc_id_daughter_2_vector.push_back(pythia.event[i].daughter2());
            mass_vector.push_back(pythia.event[i].m());
            pt_vector.push_back(pythia.event[i].pT());
            eta_vector.push_back(pythia.event[i].eta());
            phi_vector.push_back(pythia.event[i].phi());
            E_vector.push_back(pythia.event[i].e());
            names_vector.push_back(pythia.event[i].name());
        }

        Int_t iH = 0;
        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle loop.
        {
            if (pythia.event[i].id() == 25)
            {
                if (pythia.event[i].daughter1() != pythia.event[i].daughter2())
                {
                    iH = i;
                    id_daughter_1_vector_filter[iEvent] = id_vector[loc_id_daughter_1_vector[iH]];
                    id_daughter_2_vector_filter[iEvent] = id_vector[loc_id_daughter_2_vector[iH]];
                    mass_vector_daughter_1_filter[iEvent] = mass_vector[loc_id_daughter_1_vector[iH]];
                    mass_vector_daughter_2_filter[iEvent] = mass_vector[loc_id_daughter_2_vector[iH]];
                    pt_vector_daughter_1_filter[iEvent] = pt_vector[loc_id_daughter_1_vector[iH]];
                    pt_vector_daughter_2_filter[iEvent] = pt_vector[loc_id_daughter_2_vector[iH]];
                    eta_vector_daughter_1_filter[iEvent] = eta_vector[loc_id_daughter_1_vector[iH]];
                    eta_vector_daughter_2_filter[iEvent] = eta_vector[loc_id_daughter_2_vector[iH]];
                    phi_vector_daughter_1_filter[iEvent] = phi_vector[loc_id_daughter_1_vector[iH]];
                    phi_vector_daughter_2_filter[iEvent] = phi_vector[loc_id_daughter_2_vector[iH]];
                    E_vector_daughter_1_filter[iEvent] = E_vector[loc_id_daughter_1_vector[iH]];
                    E_vector_daughter_2_filter[iEvent] = E_vector[
```

```
loc_id_daughter_2_vector[iH]];
                    invariant_mass_daughter_1_histogram  -> Fill(
mass_vector_daughter_1_filter[iEvent]);
                    invariant_mass_daughter_2_histogram  -> Fill(
mass_vector_daughter_2_filter[iEvent]);
                }
            }
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 &&
id_daughter_2_vector_filter[i] == 22) //photon
        {
            id_photon_vector_daughter_1.push_back(
id_daughter_1_vector_filter[i]);
            id_photon_vector_daughter_2.push_back(
id_daughter_2_vector_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 21 &&
id_daughter_2_vector_filter[i] == 21) //gluon
        {
            id_gluon_vector_daughter_1.push_back(
id_daughter_1_vector_filter[i]);
            id_gluon_vector_daughter_2.push_back(
id_daughter_2_vector_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 &&
id_daughter_1_vector_filter[i] == 22)
        {
            pt_photon_1.push_back(pt_vector_daughter_1_filter[i]);
            pt_photon_2.push_back(pt_vector_daughter_2_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 &&
id_daughter_1_vector_filter[i] == 22)
        {
            eta_photon_1.push_back(eta_vector_daughter_1_filter[i]);
            eta_photon_2.push_back(eta_vector_daughter_2_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 &&
```

```
id_daughter_1_vector_filter[i] == 22)
        {
            phi_photon_1.push_back(phi_vector_daughter_1_filter[i]);
            phi_photon_2.push_back(phi_vector_daughter_2_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 && id_daughter_1_vector_filter[i] == 22)
        {
            E_photon_1.push_back(E_vector_daughter_1_filter[i]);
            E_photon_2.push_back(E_vector_daughter_2_filter[i]);
        }
    }

    //filters
    copy_if (id_photon_vector_daughter_1.begin(), id_photon_vector_daughter_1.end(), back_inserter(id_photon_vector_daughter_1_filter), [](int i){return i != 0;} );
    copy_if (id_photon_vector_daughter_2.begin(), id_photon_vector_daughter_2.end(), back_inserter(id_photon_vector_daughter_2_filter), [](int i){return i != 0;} );

    copy_if (id_gluon_vector_daughter_1.begin(), id_gluon_vector_daughter_1.end(), back_inserter(id_gluon_vector_daughter_1_filter), [](int i){return i != 0;} );
    copy_if (id_gluon_vector_daughter_2.begin(), id_gluon_vector_daughter_2.end(), back_inserter(id_gluon_vector_daughter_2_filter), [](int i){return i != 0;} );

    for (Int_t i = 0; i < id_photon_vector_daughter_1_filter.size(); ++i)
    {
        id_daughter_1_histogram_photon -> Fill(id_photon_vector_daughter_1_filter[i]);
        hs_1 -> Add(id_daughter_1_histogram_photon);
    }

    for (Int_t i = 0; i < id_gluon_vector_daughter_1_filter.size(); ++i)
    {
        id_daughter_1_histogram_gluon -> Fill(id_gluon_vector_daughter_1_filter[i]);
        hs_1 -> Add(id_daughter_1_histogram_gluon);
    }

    for (Int_t i = 0; i < id_photon_vector_daughter_2_filter.size(); ++i)
    {
        id_daughter_2_histogram_photon -> Fill(id_photon_vector_daughter_2_filter[i]);
        hs_2 -> Add(id_daughter_2_histogram_photon);
    }

    for (Int_t i = 0; i < id_gluon_vector_daughter_2_filter.size(); ++
```

```
i)
    {
        id_daughter_2_histogram_gluon -> Fill(id_gluon_vector_daughter_2_filter[i]);
        hs_2 -> Add(id_daughter_2_histogram_gluon);
    }

    // diphoton mass higgs

    TLorentzVector Photon_1  = TLorentzVector();
  TLorentzVector Photon_2  = TLorentzVector();

    for (Int_t i = 0; i < pt_photon_1.size(); ++i)
    {
        Photon_1.SetPtEtaPhiE(pt_photon_1.at(i), eta_photon_1.at(i), phi_photon_1.at(i), E_photon_1.at(i));
        Photon_2.SetPtEtaPhiE(pt_photon_2.at(i), eta_photon_2.at(i), phi_photon_2.at(i), E_photon_2.at(i));
    }

    Double_t dPhi_yy[pt_photon_1.size()];

    for (Int_t i = 0; i < pt_photon_1.size(); ++i)
    {
        dPhi_yy[i] = TMath::Abs( phi_photon_1[i] - phi_photon_2[i] );
        dPhi_yy[i] = dPhi_yy[i] < TMath::Pi() ? dPhi_yy[i] : 2*TMath::Pi() - dPhi_yy[i];
    }

    for (Int_t i = 0; i < pt_photon_1.size(); ++i)
    {
        m_yy.push_back(TMath::Sqrt( 2 * Photon_1.Pt() * Photon_2.Pt() * ( TMath::CosH( Photon_1.Eta() - Photon_2.Eta() ) - TMath::Cos( dPhi_yy[i] ) ) )*1000 - 10);
    }

    copy_if (m_yy.begin(), m_yy.end(), back_inserter(m_yy_filter), [](int i){return (i > 121 && i < 129);} );

    for (Int_t i = 0; i < m_yy_filter.size(); ++i)
    {
       invariant_mass_higgs -> Fill(m_yy_filter[i]);
    }

    canvas_1 -> cd(1);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    invariant_mass_daughter_1_histogram -> Draw();
    invariant_mass_daughter_1_histogram -> GetXaxis() -> SetTitle("m daughter 1");
    invariant_mass_daughter_1_histogram -> GetXaxis() -> CenterTitle();
    invariant_mass_daughter_1_histogram -> GetYaxis() -> SetTitle("Events");
    invariant_mass_daughter_1_histogram -> GetYaxis() -> CenterTitle();

```

```
    canvas_1 -> cd(2);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    invariant_mass_daughter_2_histogram -> Draw();
    invariant_mass_daughter_2_histogram -> GetXaxis() -> SetTitle("m daughter 2");
    invariant_mass_daughter_2_histogram -> GetXaxis() -> CenterTitle();
    invariant_mass_daughter_2_histogram -> GetYaxis() -> SetTitle("Events");
    invariant_mass_daughter_2_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> Print("histogram_mass_daughters_one_and_two.png");

    canvas_2_1 -> cd(1);
    canvas_2_1 -> Draw();
    canvas_2_1 -> SetGrid();
    hs_1 -> Draw("nostack");
    hs_1 -> GetXaxis() -> SetTitle("id daughter 1");
    hs_1 -> GetXaxis() -> CenterTitle();
    hs_1 -> GetYaxis() -> SetTitle("Events");
    hs_1-> GetYaxis() -> CenterTitle();

    auto legend_2_1 = new TLegend(0.9, 0.7, 0.48, 0.9);
    legend_2_1 -> SetHeader("Legend", "C");
    legend_2_1 -> AddEntry(id_daughter_1_histogram_photon, "Photons", "f");
    legend_2_1 -> AddEntry(id_daughter_1_histogram_gluon, "Gluons", "f");
    legend_2_1 -> Draw();

    canvas_2_1 -> Print("id_histogram_daughters_1_photon_and_gluon.png");

    canvas_2_2 -> cd(1);
    canvas_2_2 -> Draw();
    canvas_2_2 -> SetGrid();
    hs_2 -> Draw("nostack");
    hs_2 -> GetXaxis() -> SetTitle("id daughter 2");
    hs_2 -> GetXaxis() -> CenterTitle();
    hs_2 -> GetYaxis() -> SetTitle("Events");
    hs_2 -> GetYaxis() -> CenterTitle();

    auto legend_2_2 = new TLegend(0.9, 0.7, 0.48, 0.9);
    legend_2_2 -> SetHeader("Legend", "C");
    legend_2_2 -> AddEntry(id_daughter_2_histogram_photon, "Photons", "f");
    legend_2_2 -> AddEntry(id_daughter_2_histogram_gluon, "Gluons", "f");
    legend_2_2 -> Draw();

    canvas_2_2 -> Print("id_histogram_daughters_2_photon_and_gluon.png");

    canvas_3 -> cd(1);
    canvas_3 -> Draw();
```

```
    canvas_3 -> SetGrid();
    invariant_mass_higgs -> Draw();
    invariant_mass_higgs -> GetXaxis() -> SetTitle("myy");
    invariant_mass_higgs -> GetXaxis() -> CenterTitle();
    invariant_mass_higgs -> GetYaxis() -> SetTitle("Events");
    invariant_mass_higgs -> GetYaxis() -> CenterTitle();

    canvas_3 -> Print("higgs_mass_histogram.png");

    pythia.stat();

    vector<int>().swap(loc_vector);
    vector<int>().swap(id_vector);
    vector<int>().swap(loc_id_daughter_1_vector);
    vector<int>().swap(loc_id_daughter_2_vector);
    vector<double>().swap(mass_vector);
    vector<double>().swap(pt_vector);
    vector<double>().swap(eta_vector);
    vector<double>().swap(phi_vector);
    vector<double>().swap(E_vector);
    vector<string>().swap(names_vector);

    loc_vector.clear();
    id_vector.clear();
    loc_id_daughter_1_vector.clear();
    loc_id_daughter_2_vector.clear();
    mass_vector.clear();
    pt_vector.clear();
    eta_vector.clear();
    phi_vector.clear();
    E_vector.clear();
    names_vector.clear();

    vector<int>().swap(id_photon_vector_daughter_1);
    vector<int>().swap(id_photon_vector_daughter_2);
    vector<int>().swap(id_photon_vector_daughter_1_filter);
    vector<int>().swap(id_photon_vector_daughter_2_filter);

    id_photon_vector_daughter_1.clear();
    id_photon_vector_daughter_2.clear();
    id_photon_vector_daughter_1_filter.clear();
    id_photon_vector_daughter_2_filter.clear();

    vector<int>().swap(id_gluon_vector_daughter_1);
    vector<int>().swap(id_gluon_vector_daughter_2);
    vector<int>().swap(id_gluon_vector_daughter_1_filter);
    vector<int>().swap(id_gluon_vector_daughter_2_filter);

    id_gluon_vector_daughter_1.clear();
    id_gluon_vector_daughter_2.clear();
    id_gluon_vector_daughter_1_filter.clear();
    id_gluon_vector_daughter_2_filter.clear();

    vector<double>().swap(pt_photon_1);
    vector<double>().swap(pt_photon_2);
    vector<double>().swap(eta_photon_1);
    vector<double>().swap(eta_photon_2);
```

```
    vector<double>().swap(phi_photon_1);
    vector<double>().swap(phi_photon_2);
    vector<double>().swap(E_photon_1);
    vector<double>().swap(E_photon_2);

    pt_photon_1.clear();
    pt_photon_2.clear();
    eta_photon_1.clear();
    eta_photon_2.clear();
    phi_photon_1.clear();
    phi_photon_2.clear();
    E_photon_1.clear();
    E_photon_2.clear();

    vector<double>().swap(m_yy);
    vector<double>().swap(m_yy_filter);

    m_yy.clear();
    m_yy_filter.clear();
}

int main()
{
    Int_t simulated_events = 10000; // each event is understood as a
    pp colition

    pythia_higgs_events_generator(simulated_events);

    return 0;
}
```

## A.4 Fourth code

```
#include <iostream>
#include <vector>
#include <string>
#include <stdio.h>
#include <iomanip>

// PYTHIA
#include "Pythia8/Pythia.h"

// ROOT, for histogramming.
#include "TH1.h"
#include "TCanvas.h"
#include "TLegend.h"
#include "THStack.h"

// ROOT, for interactive graphics.
#include "TVirtualPad.h"
#include "TApplication.h"

// ROOT, for saving file.
#include "TFile.h"

// More ROOT packages.
#include <TChain.h>
```

```
#include "TLatex.h"
#include <TMath.h>
#include <TGraph.h>
#include <TLorentzVector.h>

using namespace Pythia8;

void pythia_higgs_events_generator(Int_t number_of_events)
{
    vector<int> loc_vector;
    vector<int> id_vector;
    vector<int> loc_id_daughter_1_vector;
    vector<int> loc_id_daughter_2_vector;
    vector<double> mass_vector;
    vector<double> pt_vector;
    vector<double> eta_vector;
    vector<double> phi_vector;
    vector<double> E_vector;
    vector<string> names_vector;

    int id_daughter_1_vector_filter[number_of_events];
    int id_daughter_2_vector_filter[number_of_events];
    double mass_vector_daughter_1_filter[number_of_events];
    double mass_vector_daughter_2_filter[number_of_events];
    double pt_vector_daughter_1_filter[number_of_events];
    double pt_vector_daughter_2_filter[number_of_events];
    double eta_vector_daughter_1_filter[number_of_events];
    double eta_vector_daughter_2_filter[number_of_events];
    double phi_vector_daughter_1_filter[number_of_events];
    double phi_vector_daughter_2_filter[number_of_events];
    double E_vector_daughter_1_filter[number_of_events];
    double E_vector_daughter_2_filter[number_of_events];

    vector<int> id_c_cbar_vector_daughter_1;
    vector<int> id_c_cbar_vector_daughter_2;
    vector<int> id_c_cbar_vector_daughter_1_filter;
    vector<int> id_c_cbar_vector_daughter_2_filter;

    vector<int> id_c_vector_daughter_1_filter;
    vector<int> id_c_vector_daughter_2_filter;
    vector<int> id_cbar_vector_daughter_1_filter;
    vector<int> id_cbar_vector_daughter_2_filter;

    vector<int> id_photon_vector_daughter_1;
    vector<int> id_photon_vector_daughter_2;
    vector<int> id_photon_vector_daughter_1_filter;
    vector<int> id_photon_vector_daughter_2_filter;

    vector<double> pt_c;
    vector<double> pt_cbar;
    vector<double> eta_c;
    vector<double> eta_cbar;
    vector<double> phi_c;
    vector<double> phi_cbar;
    vector<double> E_c;
    vector<double> E_cbar;

```

```
    vector<double> m_qq;
    vector<double> m_H;

    TCanvas *canvas_1 = new TCanvas("canvas_1", "canvas_1", 10, 10,
    900, 600);
    canvas_1 -> Divide(2, 1);
    TH1F *invariant_mass_daughter_1_histogram = new TH1F("Information
    D1", "Daughter 1 masses [GeV]", 20, 0, 3);
    invariant_mass_daughter_1_histogram -> SetMarkerSize(2.0);
    invariant_mass_daughter_1_histogram -> SetLineColor(kBlue);
    invariant_mass_daughter_1_histogram -> SetFillColor(kBlue - 10);

    TH1F *invariant_mass_daughter_2_histogram = new TH1F("Information
    D2", "Daughter 2 masses [GeV]", 20, 0, 3);
    invariant_mass_daughter_2_histogram -> SetMarkerSize(2.0);
    invariant_mass_daughter_2_histogram -> SetLineColor(kRed);
    invariant_mass_daughter_2_histogram -> SetFillColor(kRed - 10);

    TCanvas *canvas_2_1 = new TCanvas("canvas_2_1", "canvas_2_1", 10,
    10, 900, 600);
    canvas_2_1 -> Divide(1, 1);
    THStack *hs_1 = new THStack("hs_1", "id daughters one");
    TH1F *id_daughter_1_histogram_c_cbar = new TH1F("Information D1 c#
    bar{c}", "id daughter 1 c#bar{c}", 10, -10, 30);
    id_daughter_1_histogram_c_cbar -> SetMarkerSize(2.0);
    id_daughter_1_histogram_c_cbar -> SetLineColor(kBlue);
    id_daughter_1_histogram_c_cbar -> SetFillColor(kBlue - 10);
    TH1F *id_daughter_1_histogram_photon = new TH1F("Information D1 #
    gammas", "id daughter 1 photon", 10, -10, 30);
    id_daughter_1_histogram_photon -> SetMarkerSize(2.0);
    id_daughter_1_histogram_photon -> SetLineColor(kRed);
    id_daughter_1_histogram_photon -> SetFillColor(kRed - 10);

    TCanvas *canvas_2_2 = new TCanvas("canvas_2_2", "canvas_2_2", 10,
    10, 900, 600);
    canvas_2_2 -> Divide(1, 1);
    THStack *hs_2 = new THStack("hs_1", "id daughters two");
    TH1F *id_daughter_2_histogram_c_cbar = new TH1F("Information D2 c#
    bar{c}", "id daughter 2 c#bar{c}", 10, -10, 30);
    id_daughter_2_histogram_c_cbar -> SetMarkerSize(2.0);
    id_daughter_2_histogram_c_cbar -> SetLineColor(kBlue);
    id_daughter_2_histogram_c_cbar -> SetFillColor(kBlue - 10);
    TH1F *id_daughter_2_histogram_photon = new TH1F("Information D2 #
    gamma", "id daughter 2 photon", 10, -10, 30);
    id_daughter_2_histogram_photon -> SetMarkerSize(2.0);
    id_daughter_2_histogram_photon -> SetLineColor(kRed);
    id_daughter_2_histogram_photon -> SetFillColor(kRed - 10);

    TCanvas *canvas_3 = new TCanvas("canvas_3", "canvas_3", 10, 10,
    900, 600);
    canvas_3 -> Divide(1, 1);
    TH1F *invariant_mass_higgs = new TH1F("Information", "Invariant
    mass Higgs boson [GeV]", 20, 120, 130);
    invariant_mass_higgs -> SetMarkerSize(2.0);
    invariant_mass_higgs -> SetLineColor(kBlue);
    invariant_mass_higgs -> SetFillColor(kBlue - 10);

```

```
    Pythia pythia;

    // Switch on processes:
    pythia.readString("HiggsSM:all = on");

    // Energy of the collitions: 13 [TeV] CM energy
    pythia.readString("Beams:eCM = 13000.");

    // How many events you want to see

    pythia.readString("Next:numberShowEvent = 0");
    pythia.init();

    for (Int_t iEvent = 0; iEvent < number_of_events; ++iEvent) // Event loop.
    {
        pythia.next();

        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle loop.
        {
            loc_vector.push_back(i);
            id_vector.push_back(pythia.event[i].id());
            loc_id_daughter_1_vector.push_back(pythia.event[i].daughter1());
            loc_id_daughter_2_vector.push_back(pythia.event[i].daughter2());
            mass_vector.push_back(pythia.event[i].m());
            pt_vector.push_back(pythia.event[i].pT());
            eta_vector.push_back(pythia.event[i].eta());
            phi_vector.push_back(pythia.event[i].phi());
            E_vector.push_back(pythia.event[i].e());
            names_vector.push_back(pythia.event[i].name());
        }

        Int_t iH = 0;
        for (Int_t i = 0; i < pythia.event.size(); ++i) // Particle loop.
        {
            if (pythia.event[i].id() == 25)
            {
                if (pythia.event[i].daughter1() != pythia.event[i].daughter2())
                {
                    iH = i;
                    id_daughter_1_vector_filter[iEvent] = id_vector[loc_id_daughter_1_vector[iH]];
                    id_daughter_2_vector_filter[iEvent] = id_vector[loc_id_daughter_2_vector[iH]];
                    mass_vector_daughter_1_filter[iEvent] = mass_vector[loc_id_daughter_1_vector[iH]];
                    mass_vector_daughter_2_filter[iEvent] = mass_vector[loc_id_daughter_2_vector[iH]];
                    pt_vector_daughter_1_filter[iEvent] = pt_vector[loc_id_daughter_1_vector[iH]];
                    pt_vector_daughter_2_filter[iEvent] = pt_vector[loc_id_daughter_2_vector[iH]];
```

```
                    eta_vector_daughter_1_filter[iEvent] = eta_vector[loc_id_daughter_1_vector[iH]];
                    eta_vector_daughter_2_filter[iEvent] = eta_vector[loc_id_daughter_2_vector[iH]];
                    phi_vector_daughter_1_filter[iEvent] = phi_vector[loc_id_daughter_1_vector[iH]];
                    phi_vector_daughter_2_filter[iEvent] = phi_vector[loc_id_daughter_2_vector[iH]];
                    E_vector_daughter_1_filter[iEvent] = E_vector[loc_id_daughter_1_vector[iH]];
                    E_vector_daughter_2_filter[iEvent] = E_vector[loc_id_daughter_2_vector[iH]];
                    invariant_mass_daughter_1_histogram  -> Fill(mass_vector_daughter_1_filter[iEvent]);
                    invariant_mass_daughter_2_histogram  -> Fill(mass_vector_daughter_2_filter[iEvent]);
                }
            }
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 4 && id_daughter_2_vector_filter[i] == -4)
        {
            id_c_cbar_vector_daughter_1.push_back(id_daughter_1_vector_filter[i]);
            id_c_cbar_vector_daughter_2.push_back(id_daughter_2_vector_filter[i]);
        }

        else if (id_daughter_1_vector_filter[i] == -4 && id_daughter_2_vector_filter[i] == 4)
        {
            id_c_cbar_vector_daughter_1.push_back(id_daughter_1_vector_filter[i]);
            id_c_cbar_vector_daughter_2.push_back(id_daughter_2_vector_filter[i]);
        }
    }

    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_daughter_1_vector_filter[i] == 22 && id_daughter_2_vector_filter[i] == 22)
        {
            id_photon_vector_daughter_1.push_back(id_daughter_1_vector_filter[i]);
            id_photon_vector_daughter_2.push_back(id_daughter_2_vector_filter[i]);
        }
    }

    copy_if (id_c_cbar_vector_daughter_1.begin(), id_c_cbar_vector_daughter_1.end(), back_inserter(id_c_cbar_vector_daughter_1_filter), [](int i){return i != 0;} );
```

```
    copy_if (id_c_cbar_vector_daughter_2.begin(),
   id_c_cbar_vector_daughter_2.end(), back_inserter(
   id_c_cbar_vector_daughter_2_filter), [](int i){return i != 0;} );

    copy_if (id_photon_vector_daughter_1.begin(),
   id_photon_vector_daughter_1.end(), back_inserter(
   id_photon_vector_daughter_1_filter), [](int i){return i != 0;} );
    copy_if (id_photon_vector_daughter_2.begin(),
   id_photon_vector_daughter_2.end(), back_inserter(
   id_photon_vector_daughter_2_filter), [](int i){return i != 0;} );

    for (Int_t i = 0; i < id_c_cbar_vector_daughter_1_filter.size();
   ++i)
    {
        id_daughter_1_histogram_c_cbar -> Fill(
   id_c_cbar_vector_daughter_1_filter[i]);
        hs_1 -> Add(id_daughter_1_histogram_c_cbar);
    }

    for (Int_t i = 0; i < id_photon_vector_daughter_1_filter.size();
   ++i)
    {
        id_daughter_1_histogram_photon -> Fill(
   id_photon_vector_daughter_1_filter[i]);
        hs_1 -> Add(id_daughter_1_histogram_photon);
    }

    for (Int_t i = 0; i < id_c_cbar_vector_daughter_2_filter.size();
   ++i)
    {
        id_daughter_2_histogram_c_cbar -> Fill(
   id_c_cbar_vector_daughter_2_filter[i]);
        hs_2 -> Add(id_daughter_2_histogram_c_cbar);
    }

    for (Int_t i = 0; i < id_photon_vector_daughter_2_filter.size();
   ++i)
    {
        id_daughter_2_histogram_photon -> Fill(
   id_photon_vector_daughter_2_filter[i]);
        hs_2 -> Add(id_daughter_2_histogram_photon);
    }

    Int_t counter_1_1 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_1_filter[i] == 4)
        {
            counter_1_1 = i;
            pt_c.push_back(pt_vector_daughter_1_filter[counter_1_1]);
        }
    }

    Int_t counter_1_2 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_2_filter[i] == -4)
```

```
        {
            counter_1_2 = i;
            pt_cbar.push_back(pt_vector_daughter_2_filter[counter_1_2]);
        }
    }

    Int_t counter_2_1 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_1_filter[i] == 4)
        {
            counter_2_1 = i;
            eta_c.push_back(eta_vector_daughter_1_filter[counter_2_1]);
        }
    }

    Int_t counter_2_2 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_2_filter[i] == -4)
        {
            counter_2_2 = i;
            eta_cbar.push_back(eta_vector_daughter_2_filter[counter_2_2]);
        }
    }

    Int_t counter_3_1 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_1_filter[i] == 4)
        {
            counter_3_1 = i;
            phi_c.push_back(phi_vector_daughter_1_filter[counter_3_1]);
        }
    }

    Int_t counter_3_2 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_2_filter[i] == -4)
        {
            counter_3_2 = i;
            phi_cbar.push_back(phi_vector_daughter_2_filter[counter_3_2]);
        }
    }

    Int_t counter_4_1 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_1_filter[i] == 4)
        {
            counter_4_1 = i;
```

```
            E_c.push_back(E_vector_daughter_1_filter[counter_4_1]);
        }
    }

    Int_t counter_4_2 = 0;
    for (Int_t i = 0; i < number_of_events; ++i)
    {
        if (id_c_cbar_vector_daughter_2_filter[i] == -4)
        {
            counter_4_2 = i;
            E_cbar.push_back(E_vector_daughter_2_filter[counter_4_2]);
        }
    }

    TLorentzVector quark_1  = TLorentzVector();
  TLorentzVector quark_2  = TLorentzVector();
    TLorentzVector quark_12 = TLorentzVector();

    for (Int_t i = 0; i < pt_c.size(); ++i)
    {
        quark_1.SetPtEtaPhiE(pt_c.at(i), eta_c.at(i), phi_c.at(i), E_c
    .at(i));
        quark_2.SetPtEtaPhiE(pt_cbar.at(i), eta_cbar.at(i), phi_cbar.
    at(i), E_cbar.at(i));
        quark_12 = quark_1 + quark_2;
        m_qq.push_back(quark_12.Mag());
    }

    Int_t counter_5 = 0;
    for (Int_t i = 0; i < pt_c.size(); ++i)
    {
        if (m_qq[i] > 120 && m_qq[i] < 130)
        {
            counter_5 = i;
            m_H.push_back(m_qq[counter_5]);
        }
    }

    for (Int_t i = 0; i < m_H.size(); ++i)
    {
        invariant_mass_higgs -> Fill(m_H[i]);
    }

    canvas_1 -> cd(1);
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    invariant_mass_daughter_1_histogram -> Draw();
    invariant_mass_daughter_1_histogram -> GetXaxis() -> SetTitle("m
    daughter 1");
    invariant_mass_daughter_1_histogram -> GetXaxis() -> CenterTitle()
    ;
    invariant_mass_daughter_1_histogram -> GetYaxis() -> SetTitle("
    Events");
    invariant_mass_daughter_1_histogram -> GetYaxis() -> CenterTitle()
    ;

    canvas_1 -> cd(2);
```

```
    canvas_1 -> Draw();
    canvas_1 -> SetGrid();
    invariant_mass_daughter_2_histogram -> Draw();
    invariant_mass_daughter_2_histogram -> GetXaxis() -> SetTitle("m daughter 2");
    invariant_mass_daughter_2_histogram -> GetXaxis() -> CenterTitle();
    invariant_mass_daughter_2_histogram -> GetYaxis() -> SetTitle("Events");
    invariant_mass_daughter_2_histogram -> GetYaxis() -> CenterTitle();

    canvas_1 -> Print("histogram_mass_daughters_one_and_two_second_analysis.png");

    canvas_2_1 -> cd(1);
    canvas_2_1 -> Draw();
    hs_1 -> Draw("nostack");
    hs_1 -> GetXaxis() -> SetTitle("id daughter 1");
    hs_1 -> GetXaxis() -> CenterTitle();
    hs_1 -> GetYaxis() -> SetTitle("Events");
    hs_1-> GetYaxis() -> CenterTitle();

    auto legend_2_1 = new TLegend(0.1, 0.7, 0.48, 0.9);
    legend_2_1 -> SetHeader("Legend", "C");
    legend_2_1 -> AddEntry(id_daughter_1_histogram_c_cbar, "c#bar{c}", "f");
    legend_2_1 -> AddEntry(id_daughter_1_histogram_photon, "photons", "f");
    legend_2_1 -> Draw();

    canvas_2_1 -> Print("id_histogram_daughters_1_c_cbar_and_photon.png");

    canvas_2_2 -> cd(1);
    canvas_2_2 -> Draw();
    hs_2 -> Draw("nostack");
    hs_2 -> GetXaxis() -> SetTitle("id daughter 2");
    hs_2 -> GetXaxis() -> CenterTitle();
    hs_2 -> GetYaxis() -> SetTitle("Events");
    hs_2 -> GetYaxis() -> CenterTitle();

    auto legend_2_2 = new TLegend(0.1, 0.7, 0.48, 0.9);
    legend_2_2 -> SetHeader("Legend", "C");
    legend_2_2 -> AddEntry(id_daughter_2_histogram_c_cbar, "c#bar{c}", "f");
    legend_2_2 -> AddEntry(id_daughter_2_histogram_photon, "photon", "f");
    legend_2_2 -> Draw();

    canvas_2_2 -> Print("id_histogram_daughters_2_c_cbar_and_photon.png");

    canvas_3 -> cd(1);
    canvas_3 -> Draw();
    invariant_mass_higgs -> Draw();
    invariant_mass_higgs -> GetXaxis() -> SetTitle("mH");
```

```
    invariant_mass_higgs -> GetYaxis() -> SetTitle("Events");
    invariant_mass_higgs -> GetXaxis() -> CenterTitle();
    invariant_mass_higgs -> GetYaxis() -> CenterTitle();

    canvas_3 -> Print("higgs_mass_histogram_second_analysis.png");

    pythia.stat();

    vector<int>().swap(loc_vector);
    vector<int>().swap(id_vector);
    vector<int>().swap(loc_id_daughter_1_vector);
    vector<int>().swap(loc_id_daughter_2_vector);
    vector<double>().swap(mass_vector);
    vector<double>().swap(pt_vector);
    vector<double>().swap(eta_vector);
    vector<double>().swap(phi_vector);
    vector<double>().swap(E_vector);
    vector<string>().swap(names_vector);

    loc_vector.clear();
    id_vector.clear();
    loc_id_daughter_1_vector.clear();
    loc_id_daughter_2_vector.clear();
    mass_vector.clear();
    pt_vector.clear();
    eta_vector.clear();
    phi_vector.clear();
    E_vector.clear();
    names_vector.clear();

    vector<int>().swap(id_c_cbar_vector_daughter_1);
    vector<int>().swap(id_c_cbar_vector_daughter_2);
    vector<int>().swap(id_c_cbar_vector_daughter_1_filter);
    vector<int>().swap(id_c_cbar_vector_daughter_2_filter);

    id_c_cbar_vector_daughter_1.clear();
    id_c_cbar_vector_daughter_2.clear();
    id_c_cbar_vector_daughter_1_filter.clear();
    id_c_cbar_vector_daughter_2_filter.clear();

    vector<int>().swap(id_photon_vector_daughter_1);
    vector<int>().swap(id_photon_vector_daughter_2);
    vector<int>().swap(id_photon_vector_daughter_1_filter);
    vector<int>().swap(id_photon_vector_daughter_2_filter);

    id_photon_vector_daughter_1.clear();
    id_photon_vector_daughter_2.clear();
    id_photon_vector_daughter_1_filter.clear();
    id_photon_vector_daughter_2_filter.clear();

    vector<double>().swap(pt_c);
    vector<double>().swap(pt_cbar);
    vector<double>().swap(eta_c);
    vector<double>().swap(eta_cbar);
    vector<double>().swap(phi_c);
    vector<double>().swap(phi_cbar);
    vector<double>().swap(E_c);
```

```
    vector<double>().swap(E_cbar);

    pt_c.clear();
    pt_cbar.clear();
    eta_c.clear();
    eta_cbar.clear();
    phi_c.clear();
    phi_cbar.clear();
    E_c.clear();
    E_cbar.clear();

    vector<double>().swap(m_qq);
    vector<double>().swap(m_H);

    m_qq.clear();
    m_H.clear();
}

int main()
{
    Int_t simulated_events = 10000; // each event is understood as a
    pp colition

    pythia_higgs_events_generator(simulated_events);

    return 0;
}
```

# Bibliography


[1] Sheldon L Glashow. Partial-symmetries of weak interactions. *Nuclear physics*, 22(4):579–588, 1961.

[2] Steven Weinberg. A model of leptons. *Physical review letters*, 19(21):1264, 1967.

[3] Abdus Salam. Weak and electromagnetic interactions. In *Selected Papers Of Abdus Salam: (With Commentary)*, pages 244–254. World Scientific, 1994.

[4] CDF ALEPH. D0, delphi, l3, opal, sld collaborations, the lep electroweak working group, the tevatron electroweak working group, and the sld electroweak and heavy flavour groups, precision electroweak measurements and constraints on the standard model. *arXiv preprint arXiv:1012.2367*, 2010.

[5] François Englert and Robert Brout. Broken symmetry and the mass of gauge vector mesons. *Physical Review Letters*, 13(9):321, 1964.

[6] Peter Ware Higgs. Broken symmetries, massless particles and gauge fields. *Phys. Lett.*, 12:132–133, 1964.

[7] Peter W Higgs. Broken symmetries and the masses of gauge bosons. *Physical Review Letters*, 13(16):508, 1964.

[8] Gerald S Guralnik, Carl R Hagen, and Thomas WB Kibble. Global conservation laws and massless particles. *Physical Review Letters*, 13(20):585, 1964.

[9] Peter W Higgs. Spontaneous symmetry breakdown without massless bosons. *Physical Review*, 145(4):1156, 1966.

[10] Tom WB Kibble. Symmetry breaking in non-abelian gauge theories. *Physical Review*, 155(5):1554, 1967.

[11] Yoichiro Nambu and Giovanni Jona-Lasinio. Dynamical model of elementary particles based on an analogy with superconductivity. i. *Physical review*, 122(1):345, 1961.

[12] Yoichiro Nambu and Nobel Lecture. Spontaneous symmetry breaking in particle physics: a case of cross fertilization, 2008.

[13] Murray Gell-Mann and Maurice Lévy. The axial vector current in beta decay. *Il Nuovo Cimento (1955-1965)*, 16(4):705–726, 1960.

[14] The LEP Working Group for Higgs, ALEPH Collaboration, DELPHI Collaboration, L3 Collaboration, OPAL Collaboration, et al. Search for the standard model higgs boson at lep. *Physics Letters B*, 565:61–75, 2003.

[15] Timo Aaltonen, VM Abazov, B Abbott, M Abolins, BS Acharya, M Adams, T Adams, J Adelman, E Aguilo, GD Alexeev, et al. Combination of tevatron searches for the standard model higgs boson in the w+ w- decay mode. *Physical review letters*, 104(6):061802, 2010.

[16] Serguei Chatrchyan, V Khachatryan, AM Sirunyan, A Tumasyan, W Adam, T Bergauer, M Dragicevic, J Erö, C Fabjan, M Friedl, et al. Combined results of searches for the standard model higgs boson in pp collisions at s= 7 tev. *Physics Letters B*, 710(1):26–48, 2012.

[17] Georges Aad, T Abajyan, B Abbott, J Abdallah, S Abdel Khalek, AA Abdelalim, R Aben, B Abi, M Abolins, OS AbouZeid, et al. Combined search for the standard model higgs boson in p p collisions at s= 7 tev with the atlas detector. *Physical Review D*, 86(3):032003, 2012.

[18] Georges Aad, Tatevik Abajyan, B Abbott, J Abdallah, S Abdel Khalek, Ahmed Ali Abdelalim, R Aben, B Abi, M Abolins, OS AbouZeid, et al. Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc. *Physics Letters B*, 716(1):1–29, 2012.

[19] Serguei Chatrchyan, Vardan Khachatryan, Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Ernest Aguilo, Thomas Bergauer, M Dragicevic, J Erö, C Fabjan, et al. Observation of a new boson at a mass of 125 gev with the cms experiment at the lhc. *Physics Letters B*, 716(1):30–61, 2012.

[20] Timo Aaltonen, VM Abazov, B Abbott, BS Acharya, M Adams, T Adams, GD Alexeev, G Alkhazov, A Alton, B Álvarez González, et al. Evidence for a particle produced in association with weak bosons and decaying to a bottom-antibottom quark pair in higgs boson searches at the tevatron. *Physical review letters*, 109(7):071804, 2012.

[21] `ROOT` User's Guide. `https://root.cern.ch/root/htmldoc/guides/users-guide/ROOTUsersGuide.html`.

[22] `ROOT` Reference Documentation. `https://root.cern/doc/master/`.

[23] `PYTHIA` Reference Documentation. `http://home.thep.lu.se/~torbjorn/pythia83html/Welcome.html`.

[24] Particle Data Group, PA Zyla, RM Barnett, J Beringer, O Dahl, DA Dwyer, DE Groom, C-J Lin, KS Lugovsky, E Pianori, et al. Review of particle physics. *Progress of Theoretical and Experimental Physics*, 2020(8):083C01, 2020.

[25] Raghunath Sahoo. Relativistic kinematics. *arXiv preprint arXiv:1604.02651*, 2016.

[26] Morad Aaboud, Georges Aad, Brad Abbott, B Abeloos, SH Abidi, OS AbouZeid, NL Abraham, H Abramowicz, H Abreu, Y Abulaiti, et al. Measurements of higgs boson properties in the diphoton decay channel with 36 fb- 1 of p p collision data at s= 13 tev with the atlas detector. *Physical Review D*, 98(5):052005, 2018.

[27] H Satz. Proceedings of the large hadron collider workshop. 1990.

[28] Georges Aad, Brad Abbott, Jalal Abdallah, R Aben, M Abolins, OS AbouZeid, H Abramowicz, H Abreu, R Abreu, Y Abulaiti, et al. Combined measurement of the higgs boson mass in p p collisions at s= 7 and 8 tev with the atlas and cms experiments. *Physical review letters*, 114(19):191803, 2015.

[29] CMS collaboration et al. Measurements of properties of the higgs boson decaying into the four-lepton final state in pp collisions at sqrt (s)= 13 tev. *arXiv preprint arXiv:1706.09936*, 2017.

[30] Morad Aaboud, Georges Aad, Brad Abbott, B Abeloos, DK Abhayasinghe, SH Abidi, OS AbouZeid, NL Abraham, H Abramowicz, H Abreu, et al. Measurement of the four-lepton invariant mass spectrum in 13 tev proton-proton collisions with the atlas detector. *Journal of High Energy Physics*, 2019(4):48, 2019.

[31] G Aad et al. Measurement of higgs boson production in the diphoton decay channel in pp collisions at center-of-mass energies of 7 and 8 tev with the atlas detector. *PHYSICAL REVIEW D Phys Rev D*, 90:112015, 2014.

[32] Vardan Khachatryan, Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Thomas Bergauer, Marko Dragicevic, Janos Erö, Christian Fabjan, Markus Friedl, Rudolf Fruehwirth, et al. Observation of the diphoton decay of the higgs boson and measurement of its properties. *The European Physical Journal C*, 74(10):3076, 2014.

[33] Georges Aad, B Abbott, J Abdallah, S Abdel Khalek, R Aben, B Abi, M Abolins, OS AbouZeid, H Abramowicz, H Abreu, et al. Measurements of higgs boson production and couplings in the four-lepton channel in p p collisions at center-of-mass energies of 7 and 8 tev with the atlas detector. *Physical Review D*, 91(1):012006, 2015.

[34] Serguei Chatrchyan, Vardan Khachatryan, Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Thomas Bergauer, Marko Dragicevic, Janos Erö, Christian Fabjan, Markus Friedl, et al. Measurement of the properties of a higgs boson in the four-lepton final state. *Physical Review D*, 89(9):092007, 2014.

[35] Collaboration ATLAS, Marco Agustoni, Hans Peter Beck, Alberto Cervelli, Antonio Ereditato, Sigve Haug, Sonja Kabana, Lukas Marti, Federico Meloni, Klaus-Peter Pretzl, et al. Observation and measurement of higgs boson decays to ww* with the atlas detector. *Physical review. D-particles, fields, gravitation, and cosmology*, 92(1):012006, 2015.

[36] Georges Aad, B Abbott, J Abdallah, R Aben, M Abolins, OS AbouZeid, H Abramowicz, H Abreu, R Abreu, Y Abulaiti, et al. Study of (w/z) h production and higgs boson couplings using h→ ww∗ decays with the atlas detector. *Journal of High Energy Physics*, 2015(8):137, 2015.

[37] Georges Aad, Brad Abbott, Jalal Abdallah, S Abdel Khalek, R Aben, B Abi, M Abolins, OS AbouZeid, H Abramowicz, H Abreu, et al. Evidence for the higgs-boson yukawa coupling to tau leptons with the atlas detector. *Journal of high energy physics*, 2015(4):117, 2015.

[38] Serguei Chatrchyan, Vardan Khachatryan, Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Thomas Bergauer, Marko Dragicevic, Janos Erö, Christian Fabjan, Markus Friedl, et al. Evidence for the 125 gev higgs boson decaying to a pair of $\tau$ leptons. *Journal of High Energy Physics*, 2014(5):104, 2014.

[39] CMS collaboration et al. Observation of the higgs boson decay to a pair of tau leptons with the cms detector. *arXiv preprint arXiv:1708.00373*, 2017.

[40] Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Federico Ambrogi, Ece Asilar, Thomas Bergauer, Johannes Brandstetter, Marko Dragicevic, Janos Erö, A Escalante Del Valle, et al. Measurements of properties of the higgs boson decaying to a w boson pair in pp collisions at s= 13tev. *Physics letters B*, 791:96–129, 2019.

[41] Daniel de Florian, Christophe Grojean, F Maltoni, C Mariotti, A Nikitenko, M Pieri, P Savard, M Schumacher, R Tanaka, R Aggleton, et al. Handbook of lhc higgs cross sections: 4. deciphering the nature of the higgs sector. *arXiv. org*, 2016.

[42] Albert M Sirunyan, Armen Tumasyan, Wolfgang Adam, Federico Ambrogi, Thomas Bergauer, Marko Dragicevic, Janos Erö, A Escalante Del Valle, Martin Flechl, Rudolf Fruehwirth, et al. A search for the standard model higgs boson decaying to charm quarks. *Journal of High Energy Physics*, 2020(3):1–54, 2020.

[43] The `LXPLUS` Service. `https://lxplusdoc.web.cern.ch/`.