POLITECNICO DI MILANO

School of Industrial and Information Engineering Master of Science in

Mathematical Engineering

STATISTICAL ANALYSIS OF SIGNALS PRODUCED IN HIGH ENERGY PHYSICS EXPERIMENTS

Supervisor: Prof. Anna Maria Paganoni

Candidate:
Tatti Paolo, matricola 862883

Academic Year 2017-2018

A chi mi ha seguito e ha sempre creduto in me

Contents

List of Figures

List of Tables

Sommario

Nel mio elaborato vengono esaminati dei metodi con lo scopo di calcolare e migliorare stime di densità relative alla massa invariante in un determinato framework e alla distribuzione angolari dell'elettrone e muone nel sistema di coordinate conosciuto come Rest Boson Frame attraverso l'utilizzo dell'analisi funzionale. Queste analisi sono state fatte su diversi dataset generati tramite Phantom, un simulatore di eventi basato sul metodo di Montecarlo, i quali eventi sono generati dal fenomeno di collisione che di verifica al *Large Hadron Collider* (LHC).
La tesi è strutturata in due parti.
La prima parte dell'elaborato è dedicata al calcolo della massa invariante delle coppie muone-neutrino (μ, ν_μ) e dei quark legati al decadimento del bosone W. Uno degli obiettivi della tesi é quello di riuscire ad identificare la coppia di quark sopracitata attraverso algortimi differenti e, succesivamente, di stimare queste distribuzioni, basando l'analisi su tecniche di Density Estimation, quali le spline. Nella tesi sono quindi proposti diversi metodi di analisi, sfruttando in ultima analisi le *Free Knot Splines*.
La seconda parte del lavoro è dedicata allo studio del prodotto del decadimento dei bosoni W^+ e W^- nel *Vector Boson Scattering* (VBS) con differenti polarizzazioni. In particolare l'analisi si concentra sulle stime delle distribuzioni angolari relative all'elettrone e al muone. Il fine dell'analisi comprende la stima di densitá multivariate, dove la ricostruzione di quest'ultime viene fatta attraverso l'utilizzo della teoria della Copula. Infine, sfruttando i risultati ottenuti, vengono riportate le analisi eseguite con lo scopo di determinare la cross-section degli eventi polarizzati tramite regressione lineare.

Parole Chiave: Large Hadron Collider, Vector Boson Scattering, Analisi Funzionale, Spline a nodi liberi, Copula, Copula non parametrica, Sezione d'urto

Abstract

In this thesis I will examine a number of methods to improve the estimates of the invariant mass distributions in a particular framework and the angular distributions of the electrons and muons in the W+W− bosons rest frame through functional analysis, working with different datasets generated by Phantom, a Monte Carlo event generator, simulating the collisions that occur in the *Large Hadron Collider* (LHC).
This thesis is divided into two different parts.
In the first part my efforts were dedicated to the computation of the invariant mass distributions with different algorithms and the analysis of those distributions via Density Estimation, relying on splines. I will then propose different methods to achieve the best estimated ending up using a technique called *Free Knot Splines*.
The second part focuses on the analysis of the products of the decay of W^+ W^- in *Vector Boson Scattering* (VBS) in order to analyze and find the best approximation of the density of the angular distributions of the two bosons, initially finding the uni-dimensional distribution related to each one of the bosons and then working with bi-dimensional distributions. In this section I have used different approaches concerning the Copula's Theory.
By using the aforementioned estimates, the last part of the thesis will shown the results leading to the reconstruction of the cross-section of the polarized distributions.

Keywords: Large Hadron Collider, Vector Boson Scattering, Functional Analysis, Free Knot Splines, Copula, Non parametric Copula , Cross-section

Chapter 1

Introduction

This chapter aims to explain the context of the work and the physical background.

1.1 Context

The Standard Model of Particle Physics (SM) is a Quantum Field Theory (QFT) describing the interactions of the smallest building blocks of the universe that are accessible to current particle physics experiments. As a QFT it describes systems in the relativistic and the microscopic limit. Particles are introduced as quantized fields acting according to a Lagrangian density formulation. The Lagrangians governing the interactions of the fields are required to obey a set of local gauge symmetries.
The development of QFT started in the late 1920s with the formulation of Dirac's equation and continued with the foundation of the ingredients of the SM: Quantum Electrodynamics, electroweak symmetry breaking electroweak theory (EWK) and Quantum Chromodynamics (QCD).
Today, the Standard Model is largely accepted as it provides precise predictions for data at the current and precursory high-energy experiments, such as the Large Hadron Collider (LHC), Tevatron, and LEP.
The SM includes a unified description of the electromagnetic and weak interactions, referred to as the electroweak interactions. The EW sector and Quantum chromodynamics (QCD) are unified in a framework (the SM) that does not introduce any dependence of their coupling constants.
A crucial prediction of the model is the presence of three vector bosons, W^+, W^- and Z^0, which are the mediators of the weak interaction.

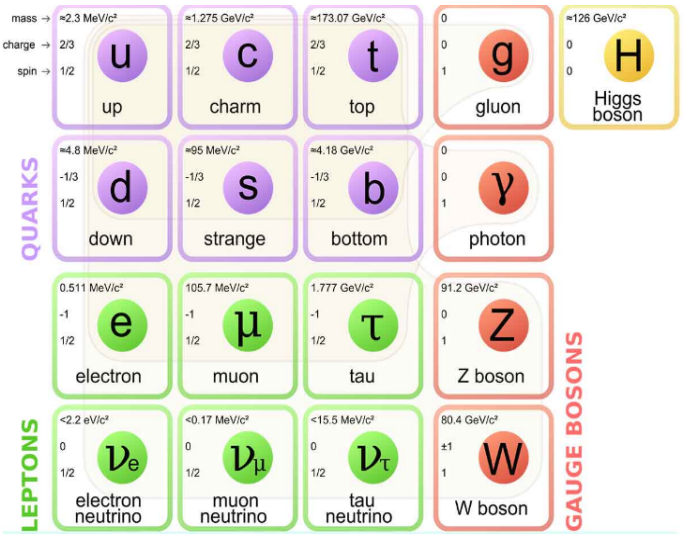

Figure 1.1: The Standard Model particles

Figure 1.1 summarizes the known fundamental particles, the four vector bosons, and the recently discovered Higgs boson.
The solutions of the problems remain still open and might be found at the heart of the SM, the breaking of the electroweak symmetry: with this mechanism, vector bosons acquire mass through their coupling to the Higgs field. At the same time, the breaking of the electroweak symmetry also rules the vector bosons scattring (VBS), avoiding its divergence at high energy.

1.2 Physics Problem

The principle behind the LHC is pretty simple. First, two beams of particles are fired along two pathways, one going clockwise and the other going counterclockwise. You accelerate both beams to near the speed of light. Then, you direct both beams toward each other leading to the collision of the protons. When the collision happens they break apart into even smaller particles, that include subatomic particles called quarks and a mitigating force called gluon.
W particles are produced in different ways during the proton-proton collision. They are heavier and they decay immediately after the collision, which makes it almost impossible to study them.
Usually in two-thirds of decays, a pair of *jets* is produced, i.e. quarks in final state. In one third of the W decays, a lepton and a neutrino are produced. However the possible output of a process can vary greatly, as I will discuss later in the elaborate.

1.3 State of art

While the first part is not related to any previous work, at least from a statistical point of view, the second part is a continuation of a previous work [1], which aimed to create a parser for *LHE files*, see SubSection 2.1, to compute distributions of $\cos\theta_e$, with respect to the W^- boson, and to estimate these distributions and the estimation of the cross-sections of the unpolarized signal, in addition to other statistical analysis not relevant to the analysis carried out by me.
The purpose of my work is to compute those distributions for both the bosons present in the process achievement of the joint distribution between them in order to find the best estimate of a bi-dimensional distribution and to be able to make several different analysis in the future. In particular the idea behind this part of the project is to be able to isolate a specific polarization of the couple of bosons. Therefore a bi-dimensional distribution is required since it is related to both bosons present in the process.
Then all the estimates found will be used to estimate the cross-sections of the bivariate unpolarized signal.

1.4 Main Issues

All the datasets provided to me are generated using *Phantom*, a Monte Carlo event generator for six parton final states at high energy colliders. The different datasets are made up of a large number of events, ranging from 500000 to 4000000. Those represent the behaviour of the phenomenon caused by the collision of the two protons in a peculiar format for this kind of data, known as *Les Houches Event* (LHE). Thanks to the previously aforementioned work, [1], a parser is already existing, which is able to translate this data into a more comfortable format, readable by the program R[19], used for the majority of the analysis.
Thus in the first part of the project I managed with the computation of the invariant mass distribution related to the muonic-leptons and the quarks related to the decay of the bosons. However no algorithms allow me to determine which quark couple is linked to the decay of the bosons, so I had to implement four different methods and choose the more effective one.
The second part of the work focused on the computation and representation of the bi-dimensional distribution formed by the angular distribution on both the electrons related to either W^- and W^+, which is not obvious because after the computation of the two marginals distribution the main problem was to achieve the joint distribution and the best estimate thereof.

1.5 Outline

The thesis is structured as follows :

- **Chapter 2** focuses on a brief explanation of the *LHE* files, the presentation of the various datasets used in the work with a quick introduction of the underlying physics and a summary of the main variable used during the process and the preprocessing analysis.
- **Chapter 3** deals with the first part of the work, containing the explanation of problems and goals and the description of the analysis, the tests and the algorithms implemented on the data, the presentation of the different techniques used and the results accomplished.
- **Chapter 4** focuses on the estimation of multivariate densities, which concerns the preprocessing made on the data in order to reach a suitable and different coordinate system, the computation of the marginal distribution and the analysis made in order to find the joint distribution and the subsequent investigation carried out, which includes the trials made with parametric Copulas and non-parametric.
- **Chapter 5** aims to describe the work made in order to rebuild the unpolarized signal coming from the dataset `noHiggs` by using the estimation made on the bivariate distributions of the cosine coming from the polarized datasets. Then it focuses on the comparison between the results obtained and he results coming from the reconstruction of the Standard Model dataset.
- **Chapter 6** summarizes the results obtained and tries to pave the way for future analysis or possible improvements on the one presented in this paper.

Chapter 2

Datasets and Preprocessing

2.1 Les Houches Event Files

The *Les Houches Event Files* are used routinely to pass information from matrix element-based generators to general-purpose ones, in order to generate complete events for a multitude of processes (see [2], [3]). The original standard was based on two Fortran common blocks, while the actual usage has tended mainly to focus on files with parton-level events and will increasingly be used by C++ generators.
The format of these event files is not specified by the standard, different format are currently being used requiring a considerable effort when such files are to be parsed. Based on the parser that was given to me, I made some minor adjustments in order to be able to process correctly all the different data and to convert them into *csv* files, readable by R [19].
A *LHE* file contains the output of the calculation of a Monte Carlo program, it is encoded in the XML format and is divided into a `header` section, a `init` section and the body containing the list of events. In the `init` section and in the event listing, physical quantities are identified by variables with a predefined name. The `header` section content depends on the generator which has been used for the event production, which usually fills it with the parameters that have been used for the event generation (not shown in Figure 2.1). The `init` section contains information to be read only once and valid for all the events stored in the *LHE* file.
It is then followed by a set of lines, which describe the processes generated by the Monte Carlo program. Each of these lines contains the information relative to one single process, therefore there are as many lines as processes generated by the Monte Carlo program. Hence, the total cross-section is the sum of the the cross-sections of the single processes. The `init` section is followed by the list of events. In this list, each event is stored in an *XML* event tag, which has one first line containing information valid for the entire

event, followed by several lines, one for each particle present in the event.
Below, Figure 2.1, it is shown an example of event in the *LHE* file, comprising the `init` section and the event simulated.

```
<LesHouchesEvents version="1.0">
<init>
     2212     2212           6500           6500    4    4   46   46    3    1
              1              0              1    661
</init>
<event>
   10    661             1      293.19061    0.007546193     0.10968293
       -1 -1    0    0    0  501             0             0      1479.1356      1479.1356              0 0 0
        2 -1    0    0  502    0             0             0     -1386.9011      1386.9011              0 0 0
       24  2    1    2    0    0      48.947344      3.8584059      378.12771      391.56847      89.076183 0 9
      -24  2    1    2    0    0      68.980917      37.187873      -71.85068      138.82663      89.269303 0 9
       -1  1    1    2    0  501     -175.21645     -26.177213      1052.7069      1067.5102              0 0 9
        2  1    1    2  502    0      57.288186     -14.869066     -1266.7495      1268.1314              0 0 9
       14  1    3    3    0    0      39.106901     -38.743299      255.94473      261.79782              0 0 9
      -13  1    3    3    0    0      9.8404431      42.601705      122.18298      129.77064              0 0 9
       11  1    4    4    0    0      4.1653606      -16.21178     -49.019967       51.79893              0 0 9
      -12  1    4    4    0    0      64.815557      53.399653     -22.830713      87.027701              0 0 9
<weights></weights>
#pdf   -1    2  0.227559E+00  0.213369E+00  0.293191E+03  0.238408E-01  0.437671E+00
</event>
```

Figure 2.1: Example of event in the LHE file

2.1.1 Events

The main part used for the analysis is divided into two common blocks called `init` and `event`.
Here I describe the various parts of an event in the *LHE* file.

1. Initialization block :
    - one line with process-number-independent information.
2. Event block :
    - one line with common event information, including the number of particles (NUP);
    - NUP lines, one for each particle i:
        - IDUP(i), identity code for particle i, each label corresponds to a physical particle.
        - ISTUP(i), status code of particle i:
            * -1 = incoming state;
            * 1 = final state;
            * 2 = intermediate state.
        - MOTHUP(1,i), the first particle from which the particle studied comes from.

- MOTHUP(2,i), the second particle from which the particle studied comes from.
- ICOLUP(1,i), the "colour" of the first mother.
- ICOLUP(2,i), the "colour" of the second mother.
- PUP(1,i), momentum along x-axis (p_x).
- PUP(2,i), momentum along y-axis (p_y).
- PUP(3,i), momentum along z-axis (p_z).
- PUP(4,i), energy (E)
- PUP(5,i), mass (M)
- VTIMUP(i), invariant lifetime.
- SPINUP(i), spin information.

We must emphasize that the variables in the *LHE* files are identified by their position; the "colour" variable mentioned before is a property related to the particles' strong interaction.
Table 2.1 shows the legend of the IDUP labels.

1 d	11 e^-	21 g
2 u	12 ν_e	22 γ
3 s	13 μ^-	23 Z^0
4 c	14 ν_μ	24 W^+
5 b	15 τ^-	25 h^0
6 t	16 ν_τ	

Table 2.1: Legend of IDUP labels

The first column describes the different quarks, the second concerns the leptons and, in the end, while the third relates to the Gauge and Higgs Bosons.
After using the parser the files are translated and ready for use.

2.2 Physical Concepts

The data on which the analysis were performed aims to simulate what happened with the collision of the quarks whose products are a pair of W bosons of opposite sign (W^+, W^-) along with two quarks, by the effect of the electroweak interaction. One issues is that the bosons are not visible due to their short half-life, thus the analysis are conducted on

the quarks and leptons found in the final state.
In the following pages I will define the main physical variables I dealt with in my computations.

2.2.1 Variables of interest

When the two beams of the particles are fired along two pathways, a special axis is defined through the geometry of particle physics experiments, namely the *beam axis*, i.e the axis parallel to the incoming beams. This particular axis should be uniquely defined and it's common practice to choose the z-axis and also to fix the origin of the coordinate system where the beams collide.
The variables used for these analysis are :

- **Quadri Momentum**
$$p = (E, p_x, p_y, p_z) \tag{2.1}$$
is the generalization of the classical three-dimensional momentum to four-dimensional spacetime; in the first position we can find the energy followed by the components of the momentum with respect to the x,y,z axes.

- **Invariant Mass**
$$m_0^2 = E^2 - \|p\|^2 \tag{2.2}$$
is the mass in the rest frame of the particle; it's calculated by using the energy E and its momentum p as measured in any frame.

- **Pseudorapidity**
$$\eta = \tanh^{-1}\left(\frac{p_z}{\|p\|}\right) \tag{2.3}$$
is a spacial coordinate used to describe the angle of a particle trajectory with respect to the beam axis.

- **Transversal Momentum**
$$p_t = \sqrt{p_x^2 + p_y^2} \tag{2.4}$$
is the component of momentum perpendicular to the beam line.

- **Polar Angle**
$$\theta = 2\arctan(e^{-\eta}) \tag{2.5}$$
angle of a particle with respect to the z-axis.

- **Azimuthal Angle**
$$\phi = \arctan\left(\frac{p_y}{p_x}\right) \tag{2.6}$$
angle around the beam axis.

- **Velocity**

$$\beta = \frac{\|p\|}{E} \tag{2.7}$$

  is the velocity of the particle, normalized with respect to the velocity of light.

- **Relativistic Factor**

$$\gamma = \frac{1}{\sqrt{1-\beta^2}} \tag{2.8}$$

  is the factor by which time, length, and relativistic mass change for an object while that object is moving.

where $\|p\| = \sqrt{p_x^2 + p_y^2 + p_z^2}$.
The first part of the work will focus on the first variable, i.e invariant mass, quadri-momentum, pseudorapidity and transversal momentum, while, during the second part, all the variables will be used for different purposes.

2.2.2 Boson Rest Frame

During the computation of the angular distribution I was interested in the electrons and muons produced by the decay of, respectively, boson W^- and W^+. This process cannot be performed in the frame provided by the laboratory, as it is necessary to change the coordinate system. The frame in which all the three momentum (p_x, p_y, p_z) for the boson are null, namely the *Boson Rest Frame*.
First of all I had to perform a rotation of $-\phi$ along the z-axis, ϕ being the azimuthal angle. Then I did a rotation of $-\theta$ along the y-axis, with θ representing the polar angle. Lastly there is a boost along the z-axis, that is a traslation without rotation, as in (2.9).

$$M = \begin{bmatrix} \gamma & 0 & 0 & -\beta\gamma \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\beta\gamma & 0 & 0 & \gamma \end{bmatrix} \tag{2.9}$$

After these particular transformations we reach the searched frame, related to the W^- boson. Since the same procedure must apply to the other boson, analogous transformations have been made in order to reach the Boson Rest Frame concerning the W^+ boson.
It is important to remember that after each transformation the variable of interest must be computed again.
In Figure 2.2 below you can see the transformations that must be made.

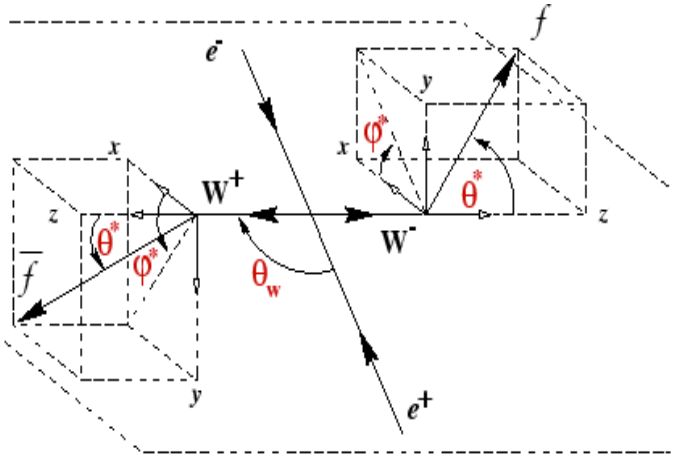

Figure 2.2: Production and decay angles of W bosons

2.2.3 Polarization

The polarization of an electromagnetic wave is determined by a quantum mechanical property, the *spin*. It is known that a massless 1-spin boson can exist in two transverse polarization states, labeled right and left. On the other hand, a massive 1-spin boson, such as W^-, besides the right and left polarization states is characterized by a third one, denoted as *longitudinal*. A boson is simply called polarized when its polarization is known and it is named unpolarized when it remains unknown.
To measure the polarization of a vector boson we need to reconstruct the four-momenta of its decay products and measure their distribution with respect to a polarization axis. In general, the polarization of a gauge boson can be determined from the angular distribution of its decay products (see [25]). The differential cross section of a leptonically-decaying W boson is related to the polarization fractions. According to the theory, for any model M, the unpolarized $\cos(\theta_e)$ distribution is a linear combination of 4 terms: 3 polarized and 1 additional.
Let x be equal to $\cos(\theta_e)$

$$\frac{d\sigma}{d\cos\theta_e} = f^M_{NP}(x) = \alpha^M_0 f^M_0(x) + \alpha^M_L f^M_L(x) + \alpha^M_R f^M_R(x) + \alpha^M_I f^M_I(x) \tag{2.10}$$

where the indices NP,0,R,L,I indicate unpolarized, longitudinal, left, right and interference, respectively.
The same can be said about the boson W^+, by setting y equal to $\cos(\theta_\mu)$, we have :

$$\frac{d\sigma}{d\cos\theta_\mu} = f^M_{NP}(y) = \alpha^M_0 f^M_0(y) + \alpha^M_L f^M_L(y) + \alpha^M_R f^M_R(y) + \alpha^M_I f^M_I(y) \tag{2.11}$$

using the same indexes as before.
W bosons can have helicity ±1 or 0. A W boson with helicity ±1 is said to be transversely polarised and one with zero helicity is longitudinally polarized, where helicity is the projection of the spin onto the direction of momentum. This means that there are four possible final polarized states of the W boson pair: transverse-transverse (TT), longitudinal-longitudinal (LL), transverse-longitudinal (TL) and longitudinal-transverse (LT), see [18].
The unpolarized distribution can be written as:

$$\frac{d^2\sigma}{d\cos\theta_e d\cos\theta_\mu} = f_{NP}^M(x,y) = \sum_{i,j=0,T} \alpha_{ij}^M f_{ij}^M(x,y) + \alpha_I^M f_I^M(x,y) \tag{2.12}$$

where $x = \cos\theta_e$, $y = \cos\theta_\mu$ and all the other indices have the same notation as in 2.10.

2.3 Datasets

As I was mentioning before, I had to work with several different datasets related to different phenomena, which can be shown by using the Feynman diagram.
Below, Figure 2.3, you can see an example of a Feynman diagram, that is the graphical representation of the mathematical expressions describing the behavior of subatomic particles.

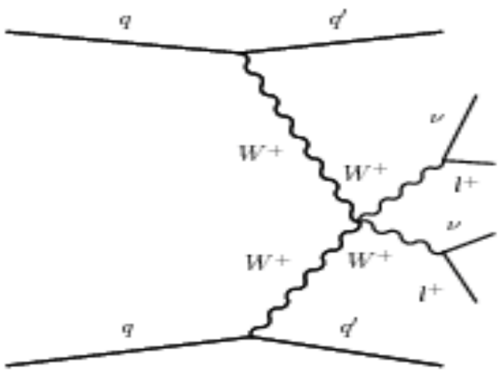

Figure 2.3: Example of Feynman's diagram

After the collision of the two quarks a pair of Boson W has been produced.
In the first part of the work I studied three different datasets linked with particular phenomena : *Electroweak* and *Quantum Chromodynamics*.

2.3.1 Electroweak & Quantum Chromodynamics

The electroweak theory comprises the description of electromagnetic and weak interactions, combined in one gauge theory. Here electromagnetic and weak interactions are combined by embedding the symmetry groups $SU(2)_I$ and $U(1)_Q$ into the new group $SU(2)_L \otimes U(1)_Y$ according to the newly introduced weak hypercharge $Y = 2(Q - I_3)$.
The theory known as Quantum Chromodynamics describes the strong interaction of quarks, important features of this particular research field are asymptotic freedom and confinement (for more details [24]).
The third dataset is a coherent sum of the two processes, that is supposed to be compared with the incoherent sum coming from the other two datasets previously mentioned.
In Figure 2.4 below, you can see a Feynman diagram related to the Electroweak theory.

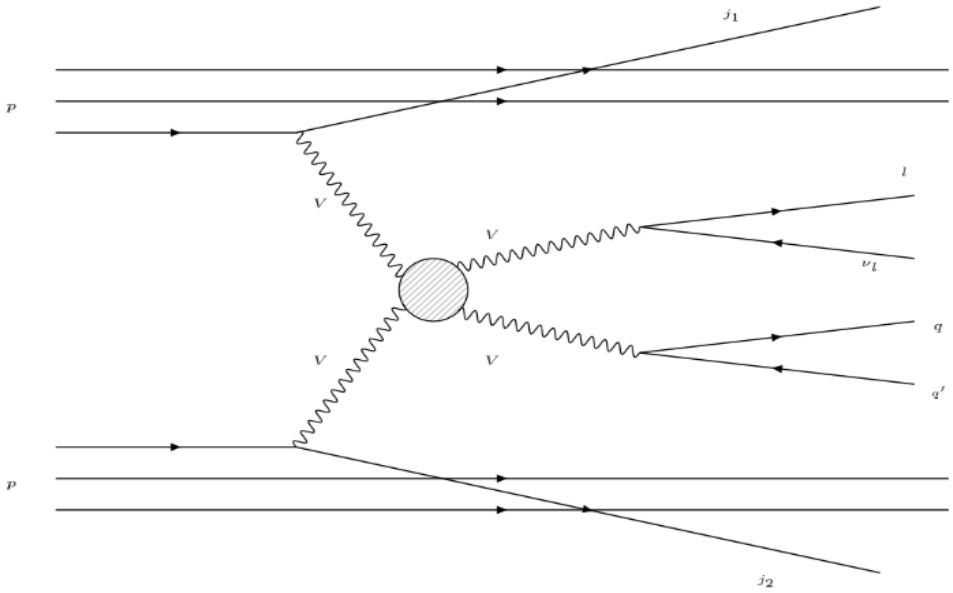

Figure 2.4: Feynman diagram of Electroweak Vector Boson Scattering

Each event is characterized by the presence of six fermions at the final state, which can be written as $pp \rightarrow WW \rightarrow lvqqqq$, where p are protons and lv represents the lepton-neutrino couple. In a nutshell, at the final state we will find:

- 2 tag quarks,
- 2 quark related to the decay of one Boson,
- a lepton-neutrino pair, related to the decay of the other Boson.

2.3.2 Standard Model and No Higgs

In all the work made I am dealing with simulation related to the Vector Boson Scattering. The difference between the first and the second part is that the six fermions at the final state are different. Therefore, in the second part, I will work with the following process: $pp \to WW \to jj \quad e^-\nu_e \quad \mu^+\nu_\mu$, where p stands for proton and j stands for *jet*. The elementary process is $qq \to qq \quad e^-\nu_e \quad \mu^+\nu_\mu$, where q stands for quark, as mentioned before.
The pair (e^-, ν_e) is the product of the decay of the boson $W-$, while the couple (μ^+, ν_μ), in a very rough approximation, is the product of the decay of the other boson. To better understand the problem, one can imagine that the boson will be irradiated by the quarks extracted from the protons and, after the scattering between themselves (W^+W^- $\to W^+W^-$), they will decay leptonically.
From a theoretical point of view, the simulation of this particular process underlines a precise dynamics related to the Standard Model, which expects the presence of the Higgs Boson, which is also able to regulate the asymptotic behaviour of the scattering $W^+W^- \to W^+W^-$. Each of the four datasets, the ones related to the Standard Model, is characterized by a specific double polarization, i.e each of the boson will be polarized. As mentioned before there are four possible final polarized states of the boson W. In addition to the four datasets which include the Higgs Boson, there is another different dataset, called `noHiggs`, which displays the same characteristics, without those of the aforementioned famous boson. This particular set of data is a complete generation of the process and it is not polarized.

2.4 Preprocessing

As mentioned before, given the magnitude of the data, the parser was not able to convert instantaneously the *LHE files*, so I have to factor the initial file into smaller files containing a tenth of the data, in order to make the conversion feasible without having an exaggerated computational effort.
Given the amount of datasets on which I had to work on, the procedure has been repeated for each of them.

2.4.1 Cuts

Since the data were simulation by using the *Phantom* generation, some cut have to be imposed even in the creation part, since we should be able to recreate a phenomenon that could actually appear in the reality and that goes in accord to the related theory developed in this particular field of physic.

Many of them are related to the geometry and the ability of the detector to reveal the particles. For instance, at the LHC, it is not be possible to see a jet (quark) with $\eta > 5.5$ or $p_t < 20$ GeV so this cut have to be imposed.
Other cuts are imposed in order to define the signals that we want to find. In this case, since the idea is to work the Vector Boson Scattering, cut on M_{jj} and $\Delta_{\eta_{jj}}$ are used.
Entering more in the details, from a experimental point of view, jets with $|\eta| > 5$ are not built because it does not exist the detector, so it is consider 5.5 as limit point, with the addition of 0.5 for security policy.
On the transversal momentum is required due to the impossibility to distinguish the of the real jets from the ones generated from the contamination sources (whose represents noise or the ones not coming from the principal collision). In this case there is more freedom of choice, since the value chosen for the cut is related to how much contamination we are willing to accept.
Below, Table 2.2, you can find the legend of the variables on which the cut where made :

Variable	Explanation
p_t^W	transversal momentum for W boson
p_t^j	transversal momentum for *jet* particles
p_t^l	transversal momentum for *lepton* particles
p_t^ν	transversal momentum for *neutrino* particles
M_{WW}	invariant mass for W+W-
M_{jj}	invariant mass for *jet* particles
η_j	pseudorapidity for *jet* particles
η_e	pseudorapidity for *electron* particles
η_μ	pseudorapidity for *muon* particles
$\Delta_{\eta_{jj}}$	variation of η for *jet* particles

Table 2.2: Legend of cut variables

The first part of the work is characterized with some imposed cut, i.e used in order to create the datasets :

- $p_t^l > 25$ GeV;
- $p_t^\nu > 25$ GeV;
- $p_t^j > 20$ GeV;

- $M_{jj} > 30$ GeV;
- $\eta_j < 5.4$;
- $\eta_j < 3$.

Even in the second part of the work the data has been generated with some imposed cut, summarized as follows :

- $p_t^j > 20$ GeV;
- $M_{jj} > 600$ GeV;
- $|\eta_j| < 5$;
- $|\Delta_{\eta_{jj}}| > 3.6$.

2.4.2 Additional cut

Regarding the second part of the work an additional cut was used in the analysis :

- $p_t^e > 20$ GeV;
- $p_t^\mu > 20$ GeV;
- $|\eta_e| < 2.5$;
- $|\eta_\mu| < 2.5$.

It is important to remind that after having imposed the additional cut just mentioned, I have move from the laboratory rest frame and to compute once more the variables of interest (i.e $\cos\theta_e$ and $\cos\theta_\mu$).
After applied the constraints on the datasets, they have been reduce, on average, to almost 60% of the original sampling size.

Chapter 3

Density Estimation: Invariant Mass Distribution

This chapter aims to describe the first part of the work, before analyzing it and describing the main results concerning the density estimation of the various distributions computed.

3.1 Goal of the analysis

The main goal, as mentioned before, of this part is to be able to estimate correctly the invariant mass distribution computed from the datasets.
I can rely on three datasets, referring to a specific process in quantum physics, known as *Electroweak* theory and Quantum Chromodynamics, plus a dataset made by the coherent sum of these two particular phenomena. To this end the datasets will be labelled as follows :

- EWK, for the Electroweak dataset,
- QCD, for the Quantum Chromodynamic dataset,
- EWK + QCD, for the coherent sum of the two above.

Each of them comprises 500000 events, made up of 10/11 particles, divided in the three possible states, as discussed before.
For each dataset, the analysis deals with the computation of several different distributions related to the couple formed by the muon (μ) and the muon neutrino (ν_μ), the quark decaying from the other boson (from now on defined as DECAY QUARK) and, ultimately, the computation of the distribution related to the total system of the two bosons, i.e the system comprising the muonic-lepton and the decay quark. All the figures related to this last part will be in Appendix A since there are not useful to the main statistical

analysis.
The events are a simulation of a process performed at LHC whose final state is the creation of six fermions, two tag quarks, two decay quarks and couple muonic-lepton. It can write as $pp \to WWq_1q_2 \to l\nu q_1q_2q_3q_4$, where q_i is the quark, l is the lepton and ν is the neutrino.
In Figure 3.1, you can see how an event of the dataset is composed.

	IDUP	ISTUP	MOTHUP1	MOTHUP2	ICOLUP1	ICOLUP2	Px	Py	Pz	E	M	VTIMUP	SPINUP	EVENT
1	1	-1	0	0	501	0	0.000000	0.00000	943.756470	943.75647	0.00000	0	0	16
2	2	-1	0	0	502	0	0.000000	0.00000	-385.526090	385.52609	0.00000	0	0	16
3	24	2	1	2	0	0	-2.845709	17.70594	-70.823925	105.93603	76.71256	0	9	16
4	-24	2	1	2	0	0	-18.634875	-27.07968	-65.533493	108.22205	79.60406	0	9	16
5	1	1	1	2	501	0	-387.106020	-91.45989	531.330520	663.72291	0.00000	0	9	16
6	2	1	1	2	502	0	408.586610	100.83363	163.257270	451.40156	0.00000	0	9	16
7	4	1	3	3	503	0	20.801546	-19.76642	-48.419949	56.28416	0.00000	0	9	16
8	-3	1	3	3	0	503	-23.647255	37.47236	-22.403976	49.65187	0.00000	0	9	16
9	13	1	4	4	0	0	5.425820	-48.22270	-57.421509	75.18044	0.00000	0	9	16
10	-14	1	4	4	0	0	-24.060695	21.14302	-8.111983	33.04162	0.00000	0	9	16

Figure 3.1: Event from Electroweak dataset

This is the result of the work made by the parser, which copies all the variables that were also in the *Les Houce file*. The only variable of interest in this stage of the work are the quadri-momentum, IDUP, ISTUP (see Subsection 2.1.1), that were used in order to diversify various particles in different states, and the variable EVENT, which is necessary to obtain the final distribution.

3.2 Searching for the Decay Quarks

The muonic-lepton pair is easily traceable just by looking at the IDUP variable and it is related to the decay of one of the bosons.
Unfortunately there is no way to find the decay quarks, because for the quantum fields theory there is only a certain probability that the quarks are decaying from the boson W. In fact, there is the chance that those quarks are decaying from other boson, for example Z boson or a Higgs Boson, and there is no way to distinguish the jets from each other. So it is indispensable to choose an algorithm a priori in order to define which define the source of the jets. Furthermore is possible to choose first the decay quark pair instead of the tags couple.
In order to proceed with the analysis a step is required where I have to evaluate and choose these specific pairs. The idea is to test four different methods and compare the different distributions of the invariant mass calculated in order to find a difference between the methods or to validate one of them or more.
From now on the different procedures will be called METHOD, followed by a number, for

the sake of simplicity.

3.2.1 Muonic-Lepton

As mentioned before the computation of the invariant mass distribution related to the muonic-lepton pair is trivial. In Figure 3.2 you can see the results, represented with an histogram for each of the datasets, from left to right EWK, QCD, EWK $_{+}$QCD.

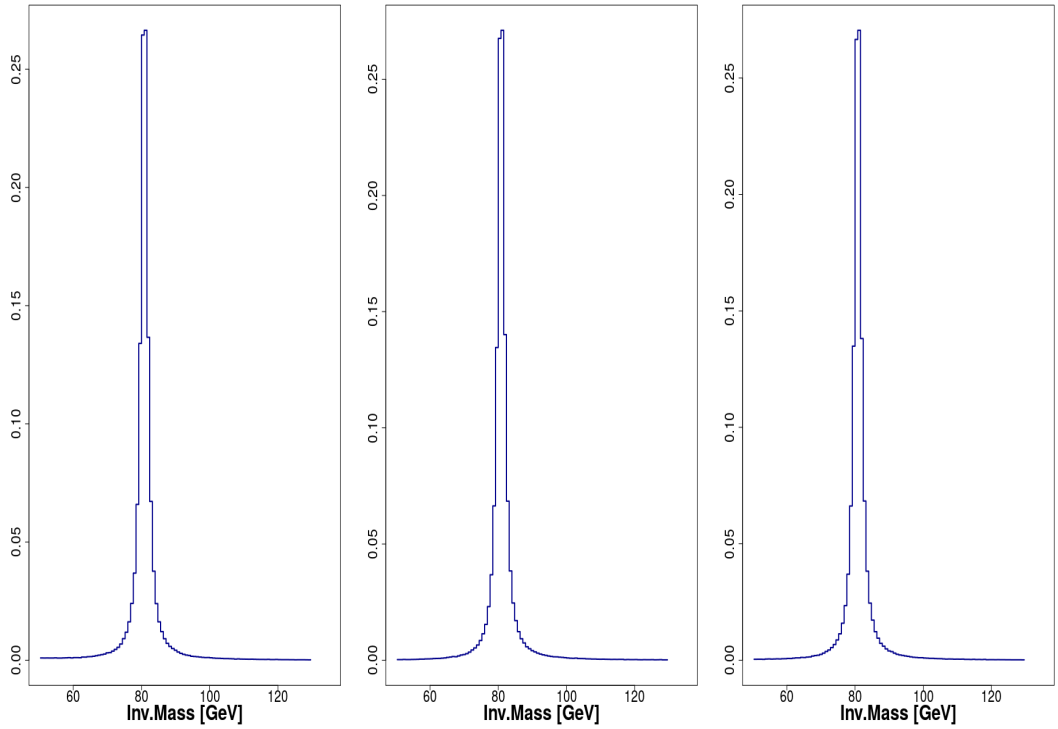

Figure 3.2: Invariant Mass: (μ, ν_{mu})

It is very clear that the behaviour of this particular pair is almost the same in each framework used. It is a very evident mode in the neighborhood of 80 GeV, which corresponds to the mass of the W boson, as we could expect by looking at the theory.

3.2.2 Methods

As mentioned, the first step with these methods is to choose the selection order of the tags or the decay quark couple. With the first two methods the jet couple has been selected as the first step, followed by the V couple, whereas with the last two methods the opposite strategy is retained. Therefore the first pair selected is the variable of interest.
After choosing the different couples, it was possible to evaluate the distribution for each of the method, relying on different variables used to distinguish different methods.
The algorithms used are the following:

- Method 1 :
  - the couple with maximum $\Delta\eta_{jj}$ as tag jets
  - the remaining couple is the V jets.
- Method 2 :
  - the couple with maximum invariant mass m_{jj} as tag jets
  - the remaining couple is the V jets.
- Method 3 :
  - the couple with maximum transversal momentum P_t as V jets
  - the remaining couple is the tag jets.
- Method 4 :
  - the couple with an invariant mass closer to 80 GeV (the W mass) as V jets
  - the remaining couple is the tag jets.

Due to the large dimension of each dataset the computation of the distributions required several hours, even using parallel computation, by relying on the `parallel` package in R [20].
Some of the plots made are visible in Figures 3.3, 3.4, 3.5 below.
Since the Methods 1,2 and 4 return more or less the same results, as detailed in 3.3, in this section I have decided to show only two plots for each method, the first Method and the third Method that is the only one that seems to have different

results with respect to the others. For the sake of completeness all the other plots will be shown in the Appendix A.

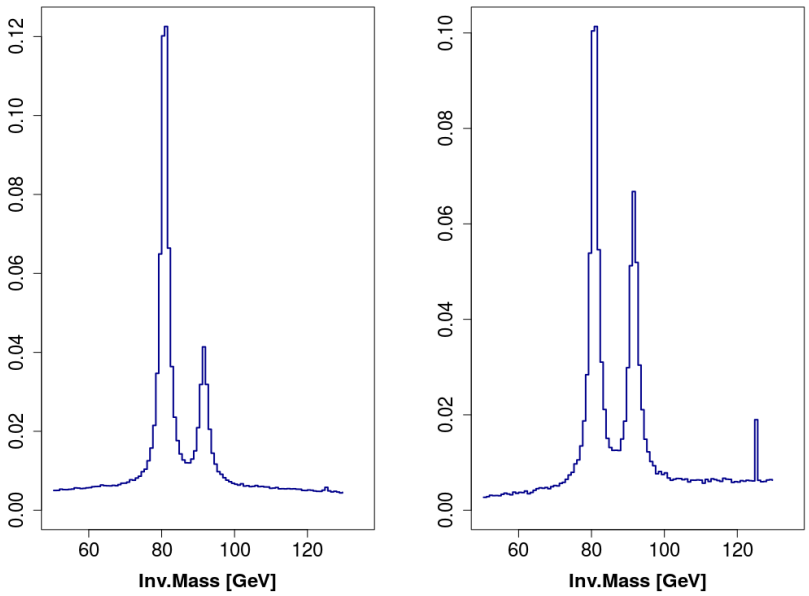

Figure 3.3: Invariant Mass Distribution (EWK): Method 1, Method 3

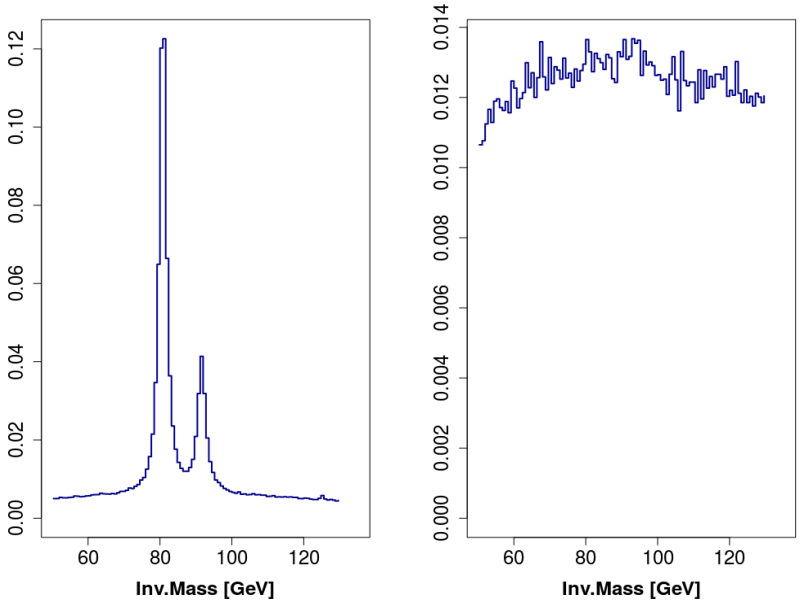

Figure 3.4: Invariant Mass Distribution (QCD): Method 1, Method 3

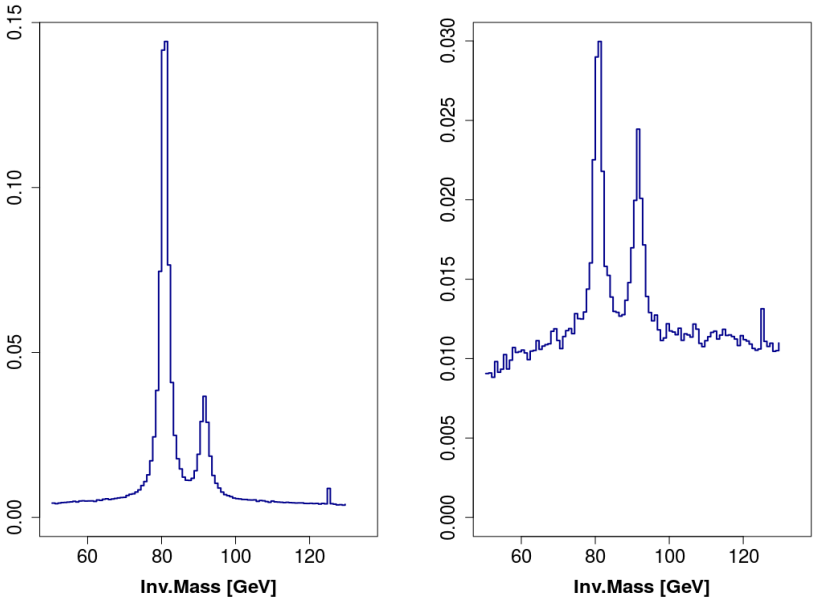

Figure 3.5: Invariant Mass Distribution (EWK + QCD): Method 1, Method 3

As I already said, the main goal behind the computation of these histograms is

to find the best method to determine which quarks can be linked to the decaying of the boson W and which can be considered as tags. Knowing the process we can guess that two W bosons will be produced in the intermediate state, between the initial state (two protons p) and the final state (charged lepton, neutrino and the four quarks). However since quantum mechanics theory does not allow to make this assumption, we can only be sure that there will be a certain probability on which we will have at the intermediate state.
By looking at the different distributions, in most cases we can notice a particular behaviour, that is the presence of three different modes at 80 GeV, 90 GeV and 125 GeV. This is very important because those peaks correspond to the mass of the boson W ($\approx$ 80 GeV), the boson Z ($\approx$ 91 GeV) and the Higgs Boson ($\approx$ 125 GeV), thus we can assume that in the intermediate state we can find not only the usual W boson but also the other two, with a given probability. By looking at those odds we can see that the probability to find a W boson is much higher than the others and also the presence of the peak with respect to the invariant mass of Higgs Boson is another prove of its existence.
They seem to be reliable methods that can be used to find the decay quarks in those type of events.
The only differences can be seen looking at the third method, which seems not to be a efficient method, as it is underlined in Figure 3.4, which represents the histogram of the invariant mass related to the QCD dataset.
Each distribution will be limited to the $[50, 130]$ GeV range, at least in this part of the analysis. For the computation of the total system of the two bosons, the range will be set to $[100, 1000]$ GeV in order to have a better image of the results since beyond a certain amount the behaviour tends to zero and thus is not useful to the analysis. Furthermore it has been decided that the histograms will have a bins equal to 100. This is important in order to be able to reproduce a particular behaviour typical of the distribution of the Boson W. In fact it has a peculiar behaviour, such as a Cauchy distribution function,also knows as Lorentz(ian) function, because it is an unstable particle that decays inside the cinematic variability allowed by the Heisenberg's uncertainty principle. The width of the resonance curve is a property of each singular particle. For the Higgs Boson in particular it is 4 MeV (with respect to its mass, 125 GeV), therefore we should have an impulse in the plots as shown with the bin used.

3.3 Testing equality between distributions

One of the goals of the analysis was to be able to evaluate the different methods in order to understand which one is different from the others or if there is a difference between them.
The idea is quite simple: to evaluate a distance between the measured distributions in order to understand which one is actually different. To do so I have decided, for each of the three datasets, to evaluate the *Earth Mover's Distance* (EMD), see Subsection 3.3.1, by mixing up the methods. This will result into six different distances related to the mutual comparison.
In order to make a more robust analysis I have decided to bootstrap, from each one of the datasets, 500 times taking, at each simulation a sample of 20000 elements. At each step I have computed the invariant mass distribution, thus the density shown before Figure 3.3, and then I have calculated the distance between the distributions obtained by using different methods.

3.3.1 Earth Mover's Distance

The EMD (see for more details [5]) is a method used to evaluate dissimilarity between two distributions in some feature space where a distance measure between single features is given. The EMD "lifts" this distance from individual features to full distributions. In mathematics it is also known as the Wasserstein metric.
Intuitively, given two distributions, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Thus, EMD measures the least amount of work needed to fill the holes with earth. A unit of work corresponds to transporting a unit of earth by a unit of *ground distance*, i.e the metric used to evaluate distances between features.
One way to represent a distribution is to see it as a set of clusters where each cluster is represented by its mean (or mode), and by the fraction of the distribution that belongs to that cluster. We call such a representation the *signature* of the distribution.
The computation of EMD is based on the solution of the transportation problem. The main idea is the following: we can think that we have several suppliers, each with a given amount of goods, are required to supply several consumers, each with a given limited capacity. For each supplier-consumer pair, the cost of transporting a single unit of goods is given. The transportation problem is then to find a

least-expensive flow of goods from the suppliers to the consumers that satisfies the consumers' demand. Matching signatures can be naturally cast as a transportation problem by defining one signature as the supplier and the other as the consumer, and by setting the cost for a supplier-consumer pair to equal the ground distance between an element in the first signature and an element in the second. Thus, the solution is the minimum amount of "work" required to transform one signature into the another.
This idea leads us to a formalization of the following linear problem : let $P = \{(p_1, w_{p_1}), \ldots, (p_m, w_{p_m})\}$ be the first signature with m clusters, where p_i is the cluster representative and w_{p_i} is the weight of the cluster.
Let $Q = \{(q_1, w_{q_1}), \ldots, (q_n, w_{q_n})\}$ be the second signature. Let $\mathbf{D} = [d_{ij}]$ be the *ground distance* between the signatures, that can be any distance chosen in according to the problem.
Let $\mathbf{F} = [f_{ij}]$ be the flow that we want to minimizes the overall cost. Below, equation 3.1, you can find the linear program problem.

$$\text{minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}d_{ij} \tag{3.1}$$

with the following constraints :

$$f_{ij} \geq 0 \qquad \forall m, \forall n \tag{3.2}$$

$$\sum_{j=1}^{n} f_{ij} \leq w_{p_i} \qquad \forall m \tag{3.3}$$

$$\sum_{i=1}^{m} f_{ij} \leq w_{q_j} \qquad \forall n \tag{3.4}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min(\sum_{i=1}^{m} w_{p_i}, \sum_{j=1}^{n} w_{q_j}) \tag{3.5}$$

Constraint (3.2) allows the movement of the supplies from P to Q and not the other way around. The purpose constraint (3.3) limits the amount of supplies that can be sent by the clusters in P to their weights. Vice versa for constraint (3.4). Constraint (3.5) forces to move the maximum amount of supplies possible.
Once the transportation problem is solved and we have found the optimal flow $\mathbf{F}$, the earth mover's distance is evaluated by normalizing the objective function with the total flow, thus:

$$\text{EMD(P, Q)} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}d_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}} \tag{3.6}$$

The total flow, i.e the normalization factor, is the total weight of the smaller signature, because of constraint (3.5). This factor is needed when the two signatures have different total weight, in order to avoid favoring smaller signatures.
By relying on the R package `earthmovdist` [13] I was able to compute the aforementioned distances between the methods, for each dataset. The package considers the L^1 distance as *ground distance* and the results are visible in Table 3.1 below.

	EWK	QCD	EWK +QCD
Method 1 vs 2	1.121	2.242	1.627
Method 1 vs 3	9.363	14.698	14.186
Method 1 vs 4	2.416	4.602	3.404
Method 2 vs 3	10.059	16.863	15.660
Method 2 vs 4	2.789	2.515	2.037
Method 3 vs 4	10.468	19.251	17.559

Table 3.1: EMD comparison between Methods

The results summarized in the table seem to show that each distance related to Method 3 is larger than the others not related to that proving what stated before, see Subsection 3.2.2.

3.4 Density Estimation

The last step of this first part is the estimation of the density. In this section I will introduce and show the results obtained with two different proceedings : *Kernel Density Estimate* and *Free Knot Splines*.
All the plots shown will be related to two methods, thus there will not be plots of the muonic-lepton couple of the plots or the total system, that can be found in Appendix A.

3.4.1 Kernel Density Estimate

Kernel Density Estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. Kernel density estimation is a funda-

mental data smoothing problem where inferences about the population are made, based on a finite data sample.
Let $(x_1, x_2, \ldots, x_n)$ be a univariate i.i.d sample drawn from some distribution with an unknown density f. The idea is to estimate the shape of a given function.
Its kernel density estimator is, as follows :

$$\hat{f}_h = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{3.7}$$

where K is the kernel, i.e a non-negative real-valued integrable function which satisfies Normalization and Symmetry properties, $h(> 0)$ is a smoothing parameters called bandwidth, that controls the degree of smoothing applied to the data.
The analysis were made relaying on the R package `KernSmooth` [29] which allows different choices of kernel and a function that tackles the bandwidth selection problem.
In order to have a more complete analysis, besides using the kernels available I decided to test a estimation based on local polynomials, still granted by the package. To compare the results and choose the best one I have used Mean Squared Error.
The first step was the bandwidth selection (see [17]). To try and solve this problem I chose the bandwidth that minimizes the asymptotic mean integrated square error, or AMISE.

$$\min_h \quad \text{AMISE} = \min_h \int_{-\infty}^{+\infty} \left[\left(\frac{1}{2} h^2 f'' k_2\right)^2 + \frac{f(x)}{nh} \int_{-\infty}^{\infty} K^2(x) dt \right] dx \tag{3.8}$$

where the first addendum is the $Bias\hat{f}(x)$ and the second one is the variance $Var[\hat{f}(x)]$.
I have decided to test more than one kernel, the Gaussian kernel, the Epanechnikov kernel, based on the centered $\mathcal{B}(2, 2)$ distribution and the triweight kernel, i.e centered $\mathcal{B}(4, 4)$ distribution.
Below, Tables 3.2, 3.3, 3.4, it is shown the Mean Squared Error related to the different estimate made with different kernel and local polynomial.

	Gaussian	Triweight	Epanechnikov	Polynomial
Method 1	0.0000700870	$6.866614e-05$	$8.079531e-05$	0.0002267953
Method 3	0.0001834522	$1.835892e-04$	$9.050061e-06$	0.0001713488

Table 3.2: MSE with different Kernels(EWK)

	Gaussian	Triweight	Epanechnikov	Polynomial
Method 1	$8.287853e-05$	$6.124896e-05$	$3.589318e-05$	$1.056905e-04$
Method 3	$1.064558e-04$	$7.938261e-06$	$7.482146e-06$	$1.157013e-05$

Table 3.3: MSE with different Kernels(QCD)

	Gaussian	Triweight	Epanechnikov	Polynomial
Method 1	$1.039075e-04$	$7.758953e-05$	$4.831625e-05$	$1.304585e-04$
Method 3	$7.856417e-04$	$5.849997e-06$	$5.709710e-06$	$1.119277e-05$

Table 3.4: MSE with different Kernels(EWK $_+$ QCD)

The results seem to suggest that the use of a triweight kernel or the Epanechnikov kernel will bring the same results, while the use of Gaussian kernel or local polynomials will carry out worse results, especially concerning the last option mentioned.
Taking into account the analysis just shown I have computed all the estimate using the Epanechnikov kernel.
Below, Figures 3.6, 3.7, 3.8, 3.9, 3.10, 3.11, there are the plots of the distributions followed by the figures of the first and second derivatives.

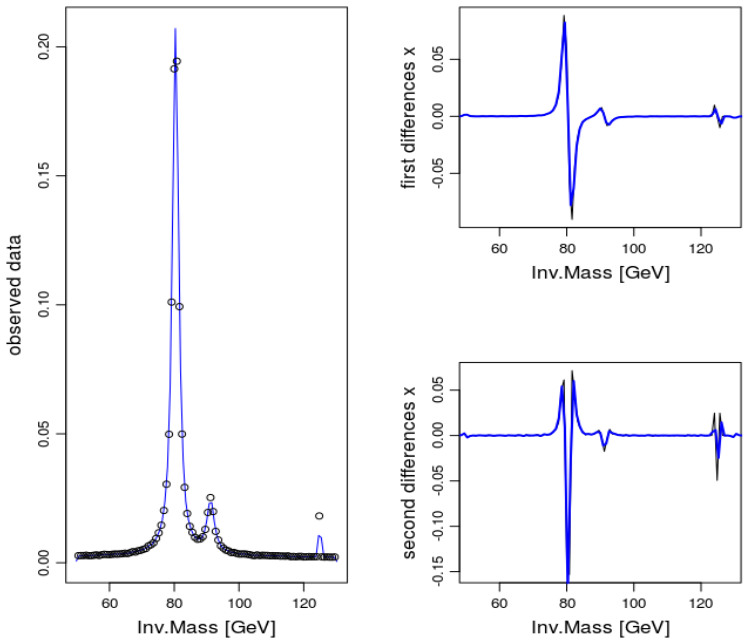

Figure 3.6: Density Estimation (KDE) : Method 1 (EWK)

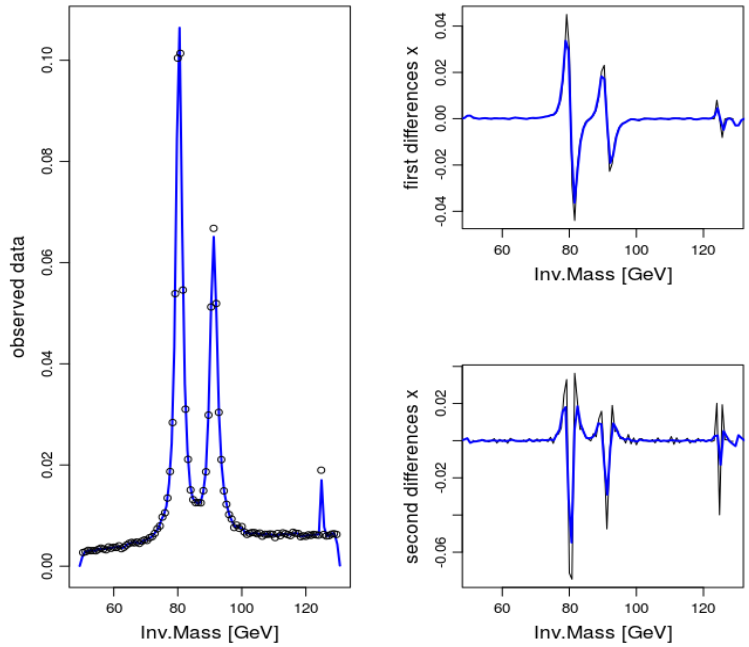

Figure 3.7: Kernel Density Estimation : Method 3 (EWK)

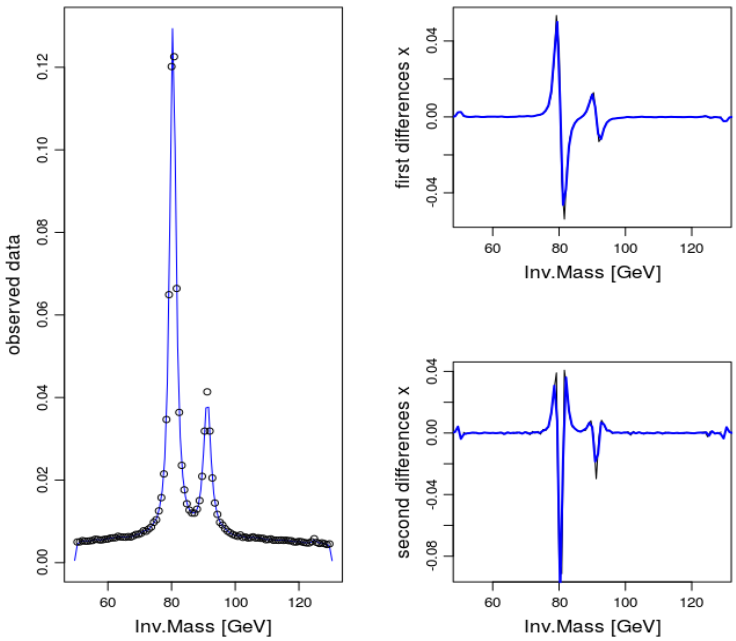

Figure 3.8: Kernel Density Estimation : Method 1 (QCD)

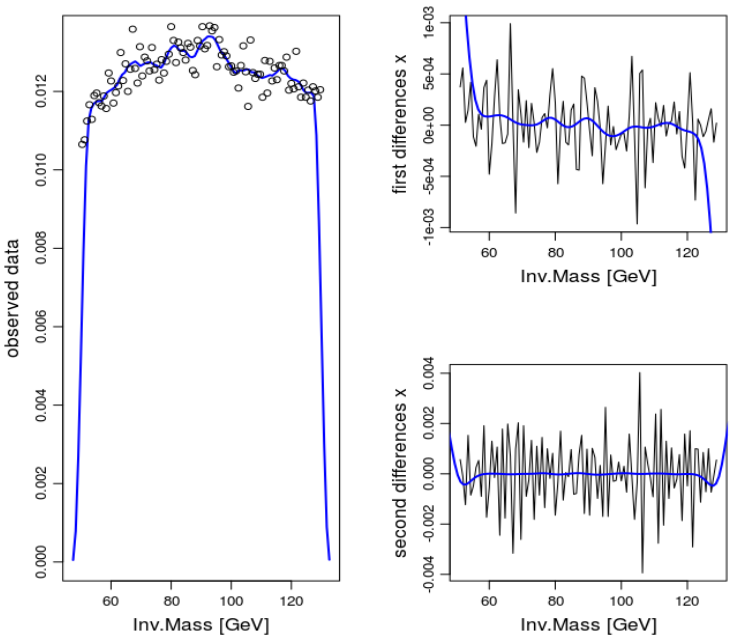

Figure 3.9: Kernel Density Estimation: Method 3 (QCD)

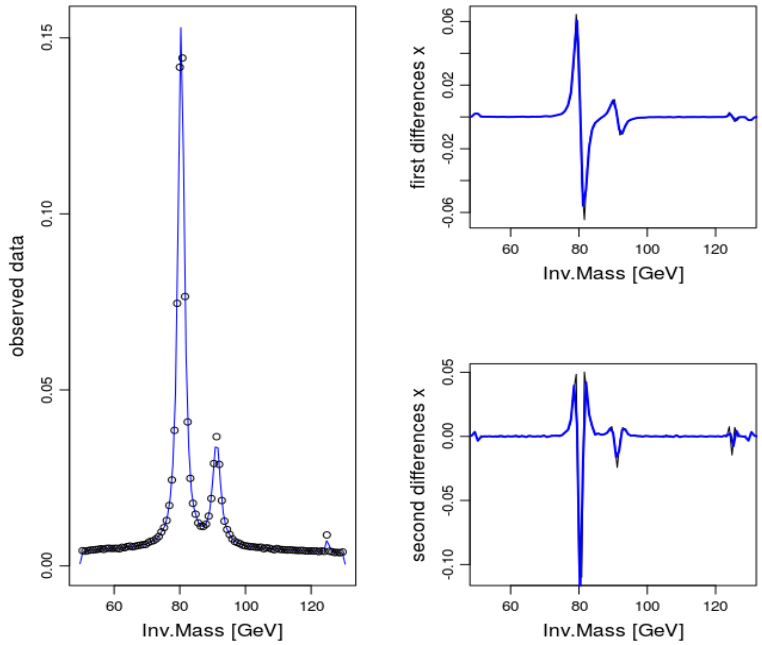

Figure 3.10: Kernel Density Estimation: Method 1 (EWK + QCD)

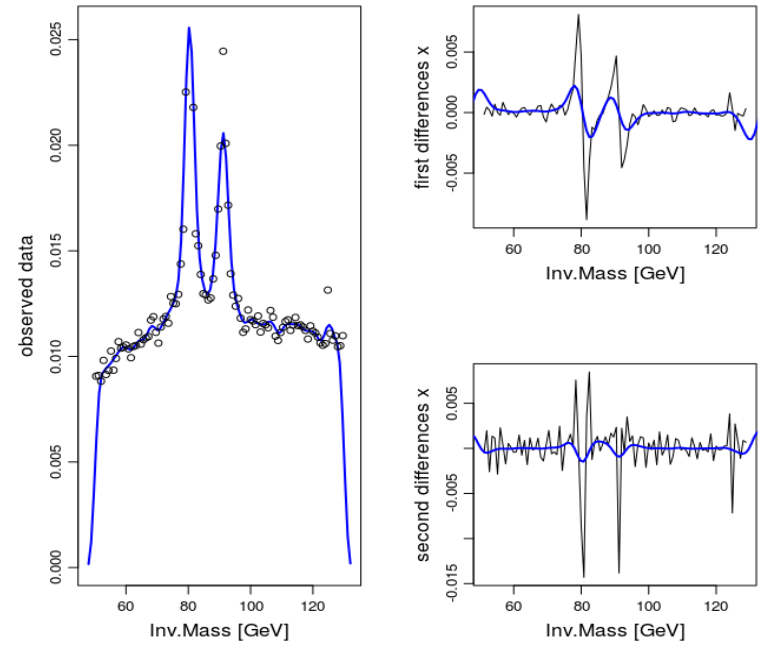

Figure 3.11: Kernel Density Estimation: Method 3 (EWK + QCD)

By looking overall at all the pictures seems that the estimates are valuable.

The fit on the observed data makes it possible to reproduce almost the same behaviour in almost all the cases, only in few plots seems that the estimates are not able to capture the shapes, especially in the neighborhoods of the peaks or in the neighborhood of the boundaries.
The same can be said for the derivatives; although bounded they are still able to follow the wanted behaviour.

3.4.2 B-splines with Free Knot

Another common choice to face this kind of problem is the use of Splines. They are used to make an approximation of non-periodic functional data or parameters. Splines are piecewise polynomial functions, where polynomial segments are joined end to end. Splines combine the fast computation of polynomials with substantial greater flexibility and are preferred to polynomial interpolation because they yields similar results, even when using low degree polynomials, while avoiding Runge's phenomenon for higher degrees.
The basic idea behind the construction of the splines is pretty straightforward. Let the spline be defined as $S : [a, b] \to \mathbb{R}$. The first step is to divide the interval $[a, b]$, where the estimated function is defined, into K sub-intervals such as to create a partition $[t_i, t_{i+1}]$ such that $[a, b] = [t_0, t_1] \cup [t_1, t_2] \cup \cdots \cup [t_{k-2}, t_{k-1}] \cup [t_{k-1}, t_k]$ and $a = t_0 \leq t1 \leq \cdots \leq t_{k-1} \leq t_k = b$. Each of these intervals is associated with a polynomial $P_k : [t_k, t_{k+1}] \to \mathbb{R}$ such that $S(t) = P_k(t)$ in the interval $t_{k-1} \leq t \leq t_k$. Each polynomial will have a given degree n and an order that is the number of constants required to define it. The given $k + 1$ points t_j $0 \leq j \leq k$) are called *knots*.
As mentioned I have decide to use a free knot spline, by relying on a particular type of spline, namely *B-Splines* (see [7]). These are spline functions that have minimal support with respect to a given degree, smoothness, and domain partition. Any spline function of a given degree can be expressed as a linear combination of B-splines of that degree. A B-spline of order n is a piecewise polynomial function of degree $n - 1$. It is defined over $n + 1$ locations t_k, i.e the knots. The B-spline function is a combination of bands that passes through the number of points, namely control points, and creates smooth curves.
A p-degree B-spline is defined as:

$$S_{n,t}(x) = \sum_i p_i B_{i,n}(x) \tag{3.9}$$

where p_i represents the control point and $B_{i,p}$ is the B-spline basis function, defined as follows by the Cox-de Boor recursion formula, see [7]:

$$B_{i,0}(x) = \begin{cases} 1, & \text{if } t_k \leq t \leq t_{k+1} \\ 0, & \text{otherwise} \end{cases} \tag{3.10}$$

$$B_{i,k}(x) = \frac{x - t_i}{t_{i+k} - t_i} B_{i,k-1}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_i} B_{i+1,k-1}(x) \tag{3.11}$$

where (3.10) is piecewise constant one or zero indicating which knot span it is in, while in (3.11) shows the recursive part of the equation.
Usually a knot vector is defined in advance, however the choice of the knots is crucial in order to get the best estimate. For example, uniform spaced knots, such as Chebyshev points, might result in an overshooting problem when the curves contain non-trivial cases, e.g discontinuous points. In order to overcome the problem, a non-uniform knot space is introduced, i.e *Free Knot Splines* (FKS). Knowing the number and the locations of the knots is critical for spline estimation. Such knowledge is typically unavailable in practice however.
Ideally, knot selection should be performed jointly instead of marginally, to obtain the global optimal knots. Unfortunately, this task is unfeasible even for small datasets, because the number of numerical evaluations grows exponentially with the number of knots.
One idea to overcome this issue and to reduce the computational cost (see [30], [15]) can be found by exploiting special local properties of the spline estimators:

- Knot addiction of a single point does not change the value of the spline outside a local neighborhood.
- The value of the spline estimate at a particular point can be well approximated by a locally fitted spline.

The B-Splines are useful in this particular framework because they are relatively well conditioned and yield an estimate that is numerically more stable than the power series representations alongside the low computational cost.
Without entering into the details, the basic idea behind the algorithm can be expressed in four step, as follows :

- Knot Initialization
- Knot Search

- Knot Relocation and Deletion
- Refinement

In order to perform the analysis I end up relying on the R package `freeknotspline` [26] that is able to elaborate the algorithm mentioned and, using `fda` [12] package, to create the first and second derivatives of the distributions. Below you can find the the plots related to the different datasets.

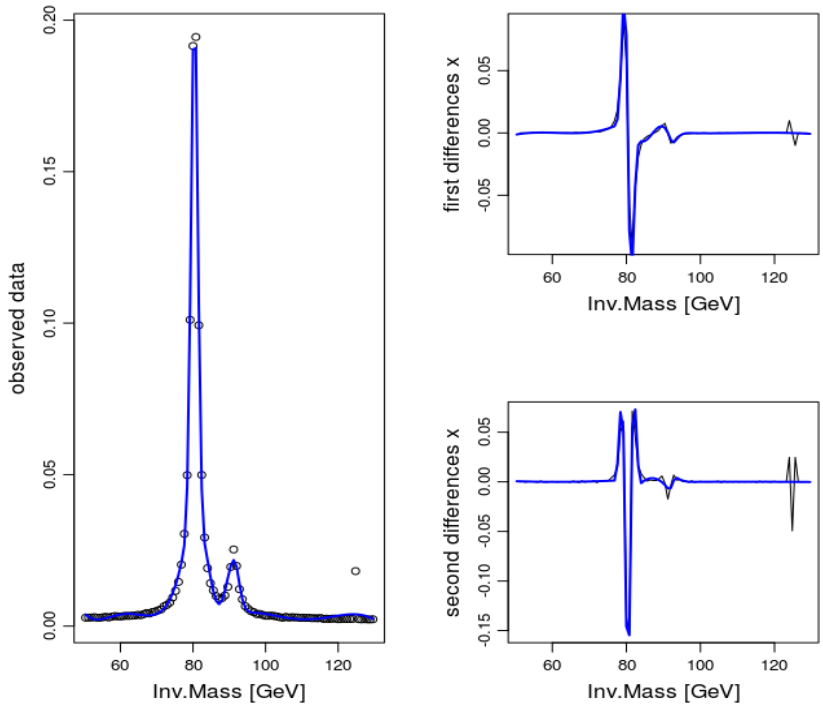

Figure 3.12: Free Knot Splines : Method 1 (EWK)

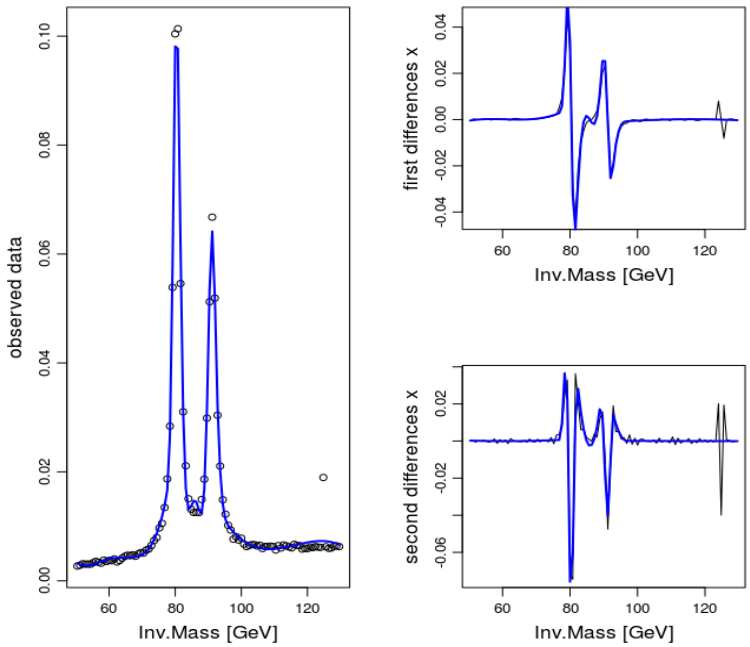

Figure 3.13: Free Knot Splines : Method 3 (EWK)

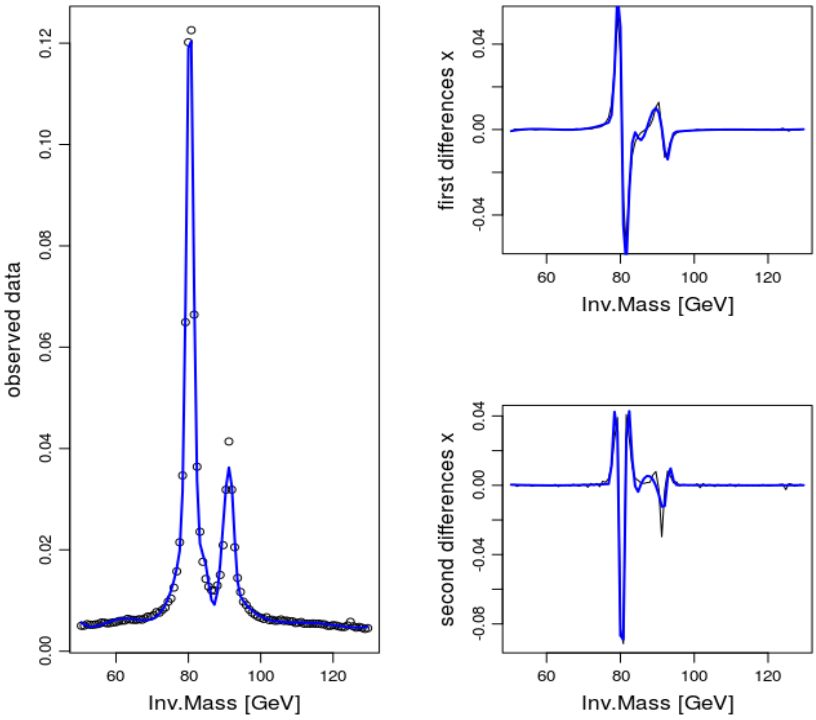

Figure 3.14: Free Knot Splines : Method 1 (QCD)

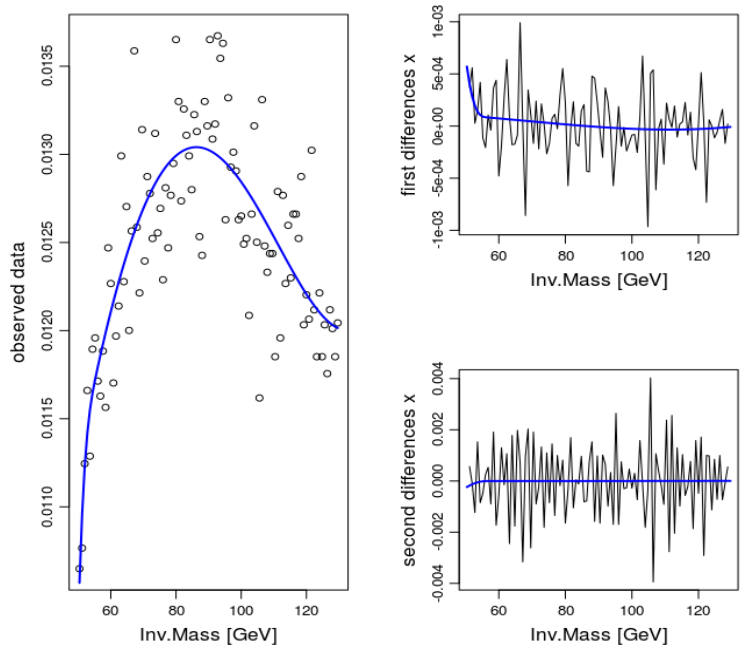

Figure 3.15: Free Knot Splines : Method 3 (QCD)

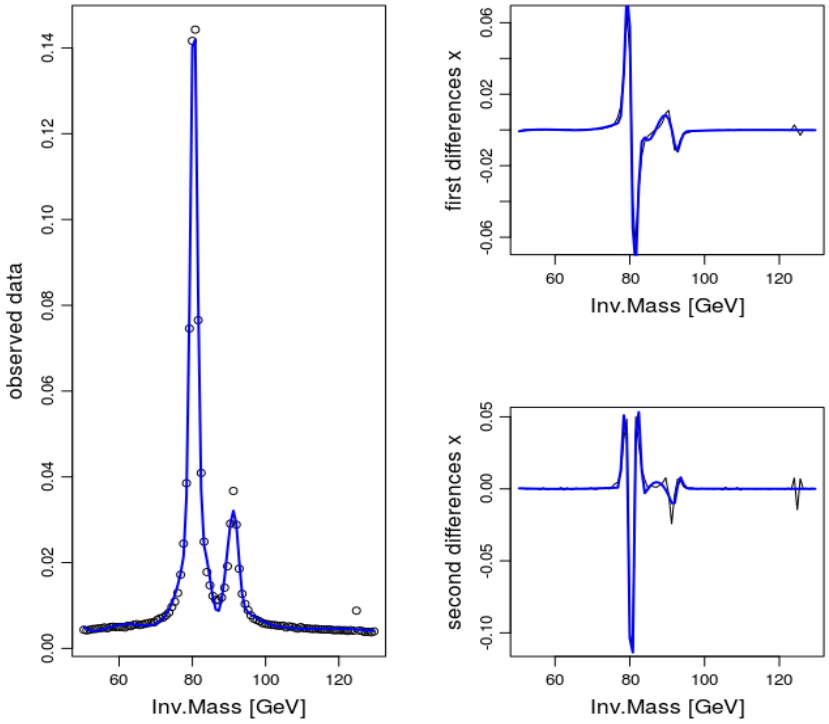

Figure 3.16: Free Knot Splines : Method 1 (EWK + QCD)

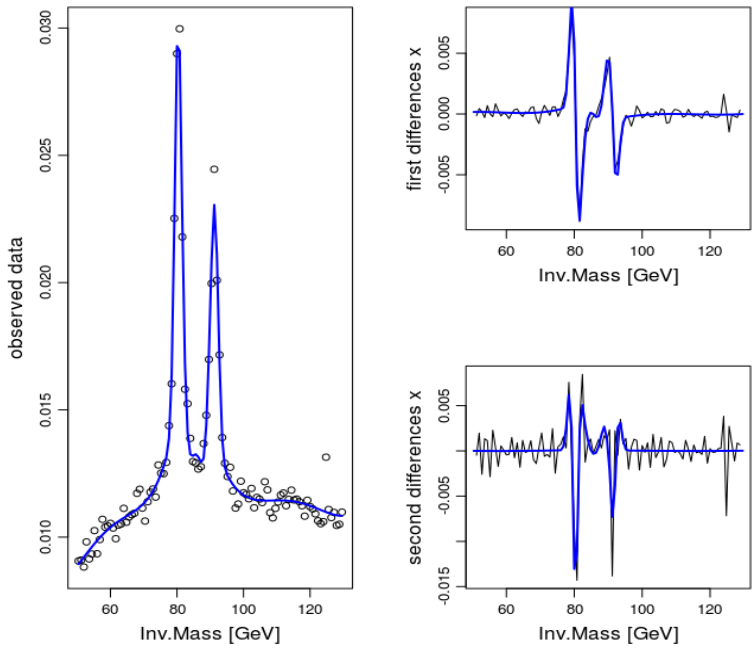

Figure 3.17: Free Knot Splines : Method 3 (EWK + QCD)

All the estimates seem to be valuable, except perhaps in the Figure 3.15 where, given the concentration of the data, it seems that the algorithm is not able to capture the behaviour shown. The same can be said about the derivatives which are surely bounded and do not show any unusual behaviour.
The function used to find the knot requires some variables : order, degree, minimum and maximum number of nodes and a goodness of the fit criterion used to evaluate the different estimates made with various numbers of knots. To this end, after some trials, I have decide to opt for cubic splines, i.e degree 3 and order 4, a range of 15 possible knots, i.e from 1 to 15, and the adjusted generalized cross-validation as index to look at the goodness of the fits. For the sake of completeness the order entered into the function is considered as real degree plus 1, therefore order 5 will appear instead of 4.
Below, Tables 3.5, 3.6, 3.7 , the are some results summarized.

	GCV	# Knot	Order
Method 1	6.08033382019761e-06	8	5
Method 3	4.35664503057913e-06	9	5

Table 3.5: Free Knot Splines Parameters (EWK)

	GCV	# Knot	Order
Method 1	2.14462463204209e-06	9	5
Method 3	1.48137056974069e-07	1	5

Table 3.6: Free Knot Splines Parameters (QCD)

	GCV	# Knot	Order
Method 1	2.60172092925441e-06	8	5
Method 3	3.24936515558523e-07	10	5

Table 3.7: Free Knot Splines Parameters (EWK + QCD)

3.4.3 Goodness of Fit

Since both methods, i.e *Free Knot Splines* and *Kernel Density Estimation* with Epanechnikov kernel, end up giving good and valuable estimates I have decided to make a comparison between them. In order to be able to evaluate which one is better I compared the Mean Squared Error of each estimates.
Below are the results divided into datasets, algorithms and methods.

	Free Knot Splines	Kernel Density
Method 1	4.393041e-06	8.079531e-05
Method 3	3.147676e-06	9.050061e-06

Table 3.8: MSE of the algorithms (EWK)

	Free Knot Splines	Kernel Density
Method 1	1.549491e-06	3.589318e-05
Method 3	1.308939e-07	7.482146e-06

Table 3.9: MSE of the algorithms (QCD)

	Free Knot Splines	Kernel Density
Method 1	1.879743e-06	4.831625e-05
Method 3	2.347666e-07	5.709710e-06

Table 3.10: MSE of the algorithms (EWK $_+$ QCD)

By looking at these results seem to indicate that the algorithm relying on the B-Splines is better than the other ones, which also return good results either.

Chapter 4

Multivariate Density Estimation using Copulas

This chapter aims to describe the second part of the work, firstly analyzing it and describing the main results concerning the computation of the desired distributions: the bi-dimensional density estimation of the various estimates.

4.1 Goal of the analysis

The main goal of this part, as mentioned before, is to correctly estimate the bi-dimensional distribution of the cosine of the θ angle, where θ is the polar angle previously mentioned, see Subsection 2.2.1.
By looking at the natural decay of the bosons I expect to see a decay to the electron (e^-) for the boson W^- and to the muon (μ^+) for the boson W^+, thus the variable of interest will be the $\cos\theta_e$ and the $\cos\theta_\mu$
The analysis focuses on 5 different datasets, which describe the following kind of event: $qq \to qq \quad e^-\nu_e \quad \mu^+\nu_\mu$.
Four datasets are related to the Standard Model and have different polarization, summarized below:

- W+W- longitudinal-longitudinal polarization (LL).
- W+W- longitudinal-transversal polarization (LT),
- W+W- transversal-longitudinal polarization (TL),
- W+W- transversal-transversal polarization (TT).

Instead the last dataset is unpolarized and does not contain the Higgs boson (from now on `noHiggs`).
The estimation techniques will be tested on all datasets.

4.2 Preprocessing and preliminary analysis

As for the previous analysis, the first part was made by relying on the parser in order to have a format readable by R.
In Figure 4.1, you can see how an event is composed.

	IDUP	ISTUP	MOTHUP1	MOTHUP2	ICOLUP1	ICOLUP2	Px	Py	Pz	E	M	VTIMUP	SPINUP	EVENT
1	3	-1	0	0	502	0	0.000000	0.000000	276.19462	276.19462	0.00000	0	0	100
2	2	-1	0	0	501	0	0.000000	0.000000	-3158.00440	3158.00440	0.00000	0	0	100
3	24	2	1	2	0	0	-21.363149	11.180442	67.04169	108.01690	81.18913	0	9	100
4	-24	2	1	2	0	0	-41.489774	23.404489	-1175.51000	1179.32810	81.98722	0	9	100
5	2	1	1	2	501	0	-95.535841	49.600700	-1915.16520	1918.18800	0.00000	0	9	100
6	3	1	1	2	502	0	158.388760	-84.185631	141.82367	228.66608	0.00000	0	9	100
7	14	1	3	3	0	0	-9.541273	8.588334	85.21824	86.17971	0.00000	0	9	100
8	-13	1	3	3	0	0	-11.821877	2.592108	-18.17654	21.83718	0.00000	0	9	100
9	11	1	4	4	0	0	-18.469061	-30.412871	-376.88199	378.55790	0.00000	0	9	100
10	-12	1	4	4	0	0	-23.020712	53.817360	-798.62798	800.77020	0.00000	0	9	100

Figure 4.1: Event from LL dataset

The main difference, with respect to the datasets in Chapter 3, is that these events are characterized by the presence, in the final state, of just one couple of quarks and the pairs of interest, electron - electron neutrino (e, ν_e) and muon - muon neutrino (μ, ν_μ).
The second step of the preprocessing is to change the coordinates system, which means switching from the Laboratory Rest frame to the Boson Rest Frame, as mentioned in Subsection 2.2.2.
In order to apply this particular change of framework it is necessary to :

- rotation of $-\phi$ along z-axis,
- rotation of $-\theta$ along y-axis,
- boost along z-axis (traslation).

The same transformation will be applied for each of the bosons, independently one from each other, in order to be able to compute correctly the distributions.
After this, remain only the calculation of cosine of the angle θ for both the bosons.
Below, in Figures 4.2, 4.4, 4.6, 4.8, 4.10, you can find a 3-D histogram (with 100

bins selected) concerning the densities for each datasets and the contour plots, Figures 4.3, 4.5, 4.7, 4.9, 4.11.

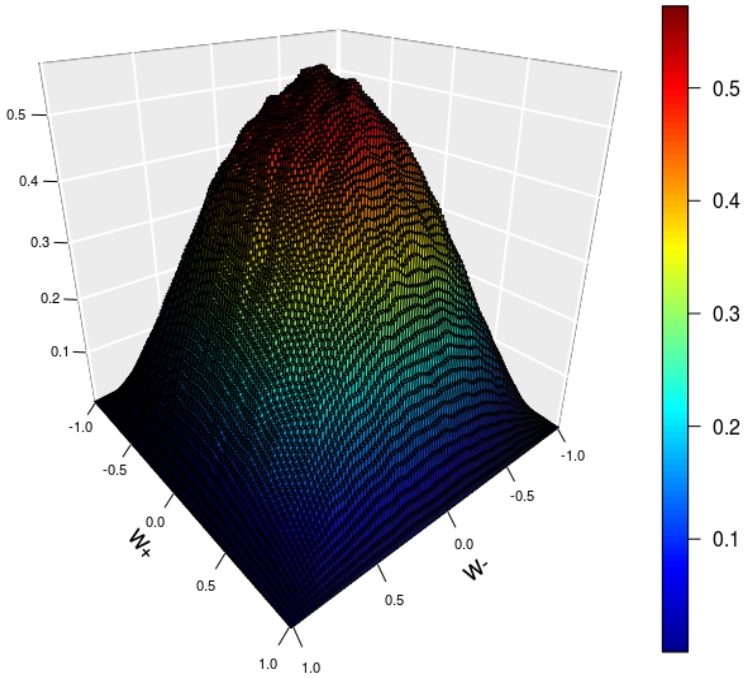

Figure 4.2: Histogram of the distribution (LL)

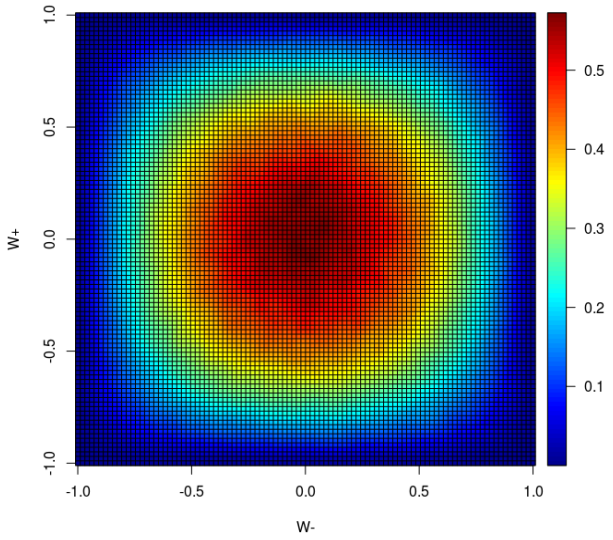

Figure 4.3: Image of the distribution (LL)

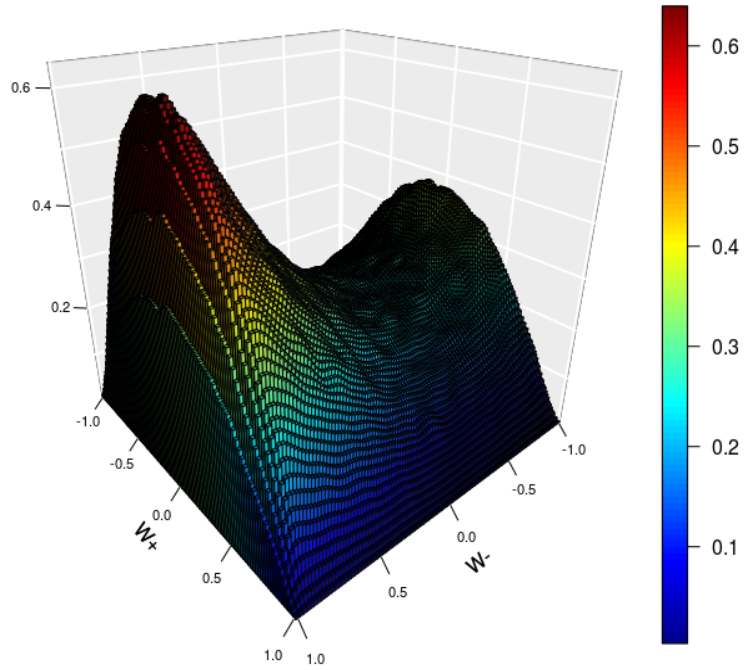

Figure 4.4: Histogram of the distribution (LT)

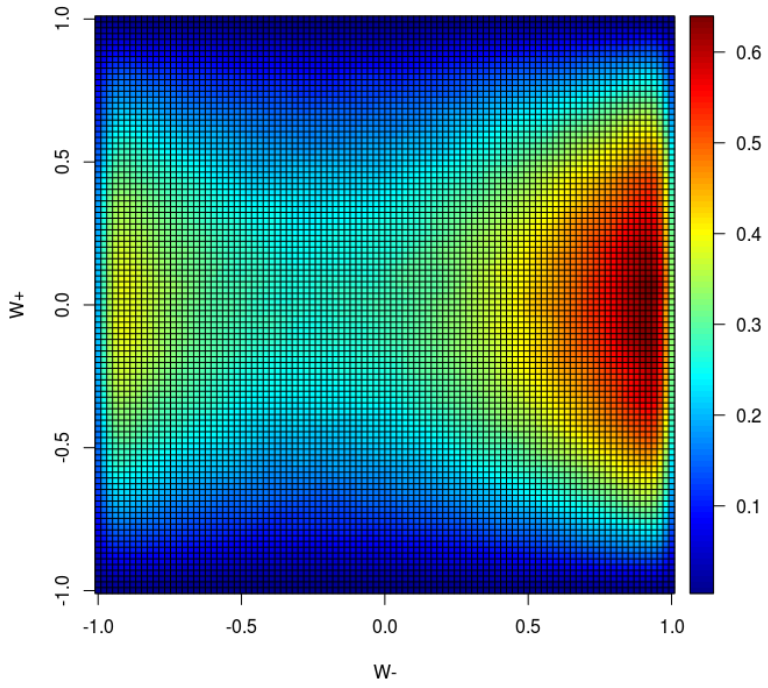

Figure 4.5: Image of the distribution (LT)

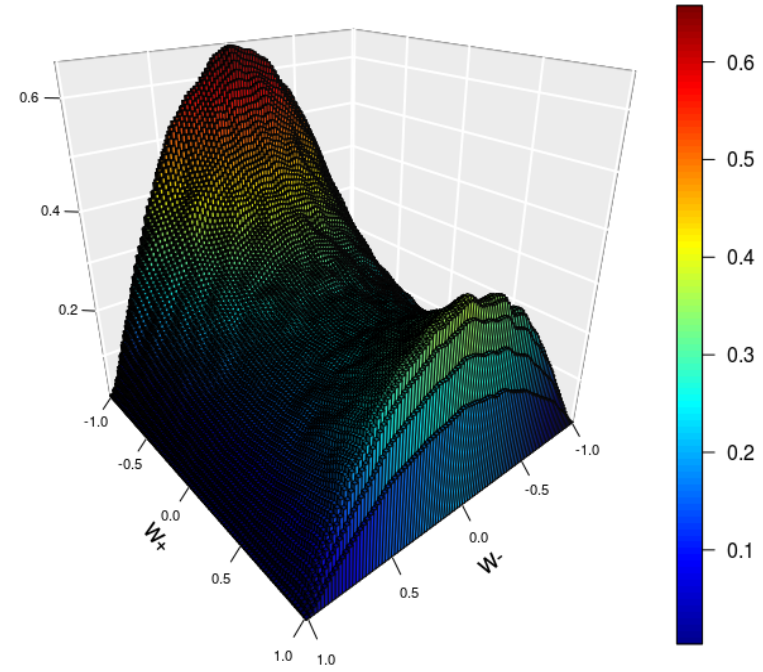

Figure 4.6: Histogram of the distribution (TL)

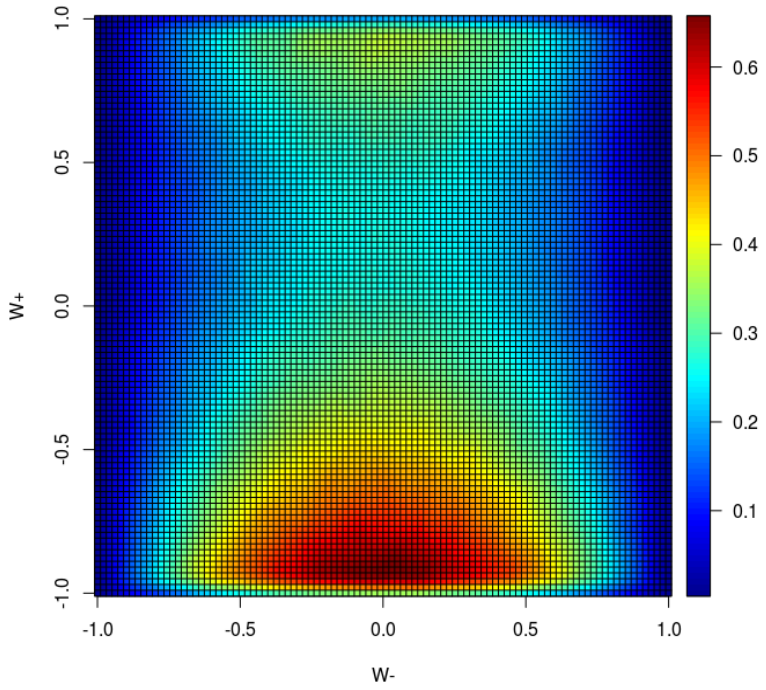

Figure 4.7: Image of the distribution (TL)

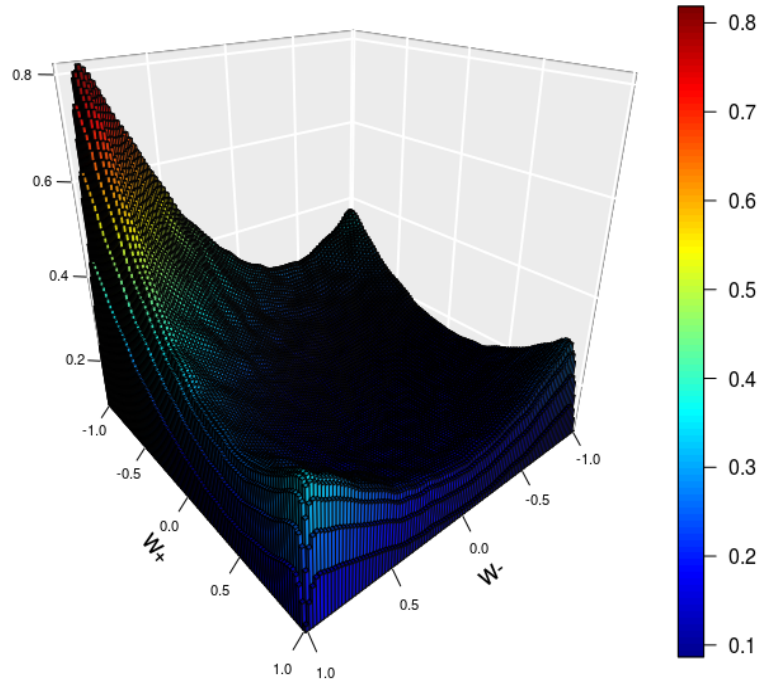

Figure 4.8: Histogram of the distribution (TT)

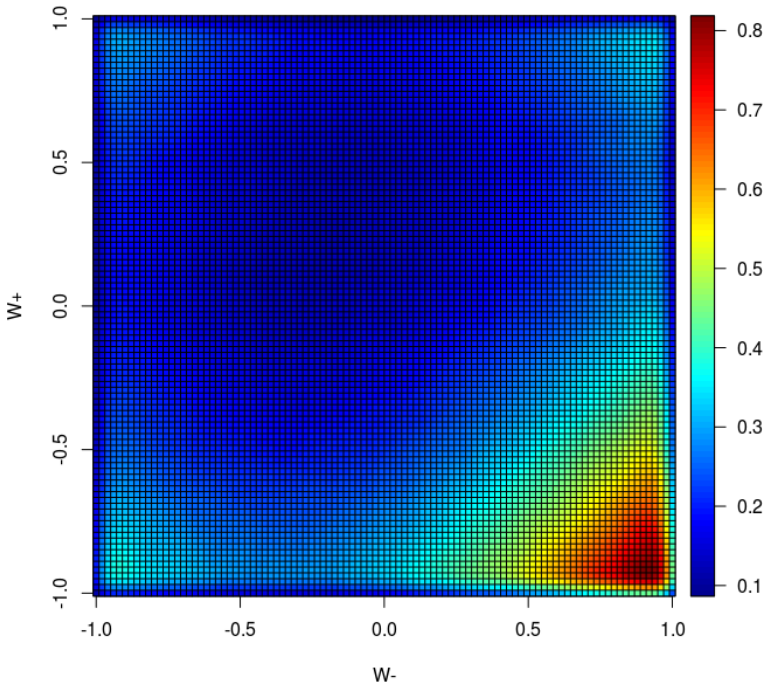

Figure 4.9: Image of the distribution (TT)

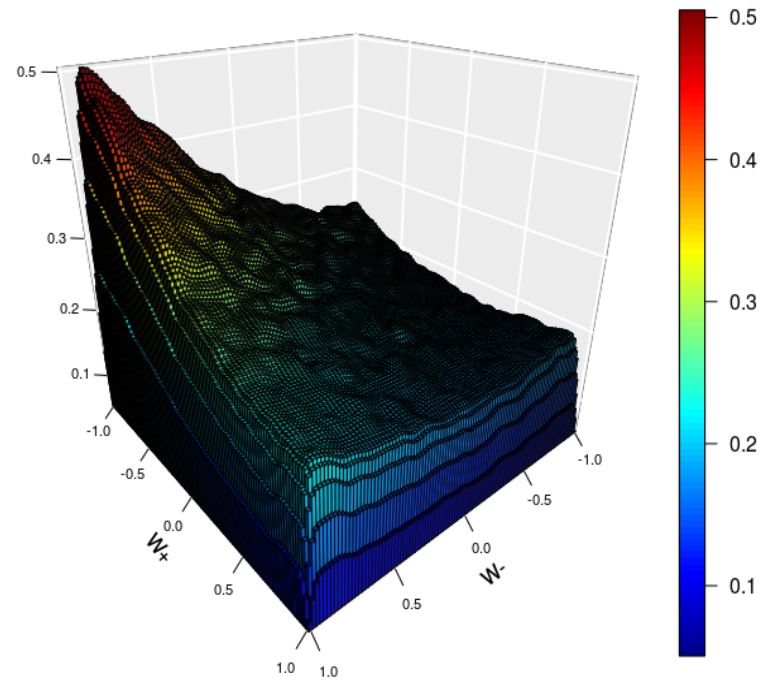

Figure 4.10: Histogram of the distribution (noHiggs)

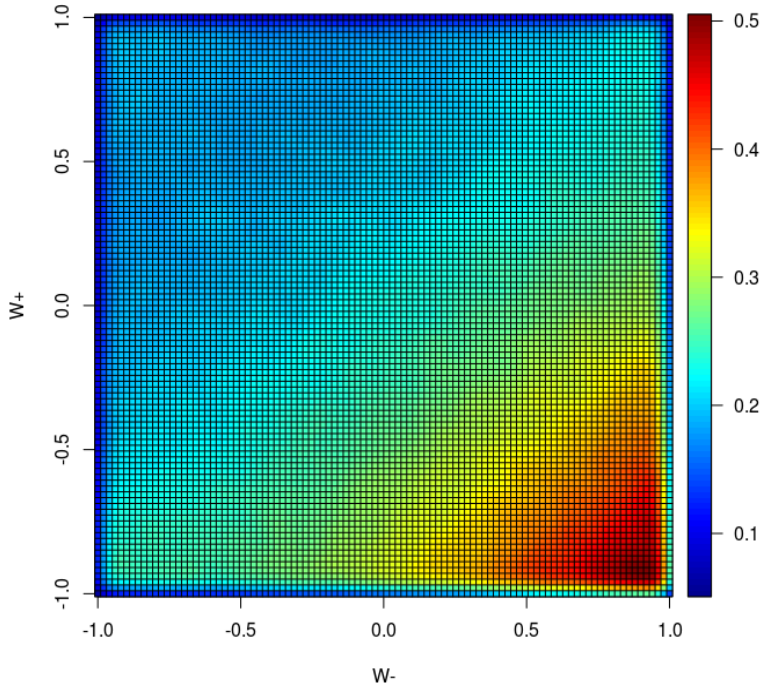

Figure 4.11: Image of the distribution (noHiggs)

In each plot the x-axis represents the cosine distribution with respect to the Boson W^-, whereas the y-axis is the cosine distribution with respect to the boson W^+.
Since the transformation used to change the coordinate system was made independently for both the bosons, I have tested the independence between the two distributions computed so that I could be able to have a better understanding of the process and the phenomenon that I was dealing with. In order to achieve that, I have decided to compare the results of two statistical test: *Kendall's Rank Correlation* test and the *Hoeffding's Independence* test.
Let $X = x_1, \ldots, x_n$ and $Y = y_1, \ldots, y_n$ be the observations on which we want to establish the independence, based on a random sample of size n.
As regards the Kendall's test (for more details see [14]), it relays on two assumptions : Independence assumption and Continuity assumption.
It has as hypothesis :

$$H_0 : \quad \tau = 0 \tag{4.1}$$

$$H_1 : \quad \tau \neq 0 \tag{4.2}$$

where H_0 meaning is the independence between variables.
It uses as test-statistic :

$$\tau = \frac{1}{n(n-1)} \sum_{i \neq j} \text{sign}(x_i - x_j)\, \text{sign}(y_i - y_j) \tag{4.3}$$

Since it appears that in the distribution there are some ties the test-statistic, the estimation used for τ it will be $\hat{\tau}_b$:

$$\hat{\tau}_b = \frac{1}{\sqrt{([n(n-1)/2] - n_x)([n(n-1)/2] - n_y)}} \tag{4.4}$$

where :

- $K = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} Q_{ij}^*$,
- $n = \#$ of observations,
- $n_x = \sum_i t_i(t_i - 1)/2$ with t_i denoting the size of i-th group of ties on X,
- $n_y = \sum_i u_i(u_i - 1)/2$ with u_i denoting the size of i-th group of ties on Y.

while the matrix Q^* can be built by looking at the following scheme :

$$Q^*[(a,b),(c,d)] = \begin{cases} 1, & \text{if } (d-b)(a-c) > 0 \\ 0, & \text{if } (d-b)(a-c) = 0 \\ -1, & \text{if } (d-b)(a-c) < 0 \end{cases} \tag{4.5}$$

Regarding the Hoeffding's test (for more details see [10]) is a non parametric test, thus it does not make any assumptions on the functional form of the population distribution. The test aims to prove the independence of two random variables assumed to have continuous distribution. Furthermore the test is consistent with the class Ω'', i.e the class of density function having continuous joint and marginal probability density.
Let $F(x,y)$ be the continuous joint distribution. In case of independence we can say that $F(x,y) = F(x,\infty)F(\infty,y)$.
The test-statistic $D(x,y)$ depends only on the rank order of the observation and it has the following form:

$$D(x,y) = F(x,y) - F(x,\infty)F(\infty,y) \tag{4.6}$$

and it can be proved that it has normal limiting distribution for any parent distribution.
In case of independence it is shown that the limiting distribution is degenerate and nD has no normal limiting distribution.
The hypothesis of the test H_0 can be seen as : the random variable are independent, and that can be summarized as follows:

$$H_0 : \quad D = 0 \tag{4.7}$$

$$H_1 : \quad D \neq 0 \tag{4.8}$$

The estimated test-statistic D_n can be written as follows:

$$D_n = \frac{A - 2(n-2)B + (n-2)(n-3)C}{n(n-1)(n-2)(n-3)(n-4)} \tag{4.9}$$

where:

- $A = \sum_{\alpha=1}^{n} a_\alpha(a_\alpha - 1)b_\alpha(b_\alpha - 1)$,
- $B = \sum_{\alpha=1}^{n}(a_\alpha - 1)(b_\alpha - 1)c_\alpha$,
- $C = \sum_{\alpha=1}^{n} c_\alpha(c_\alpha - 1)$,

- $a_\alpha = \sum_{\beta=1}^{m} C(X_\alpha - X_\beta) - 1$,
- $b_\alpha = \sum_{\beta=1}^{m} C(Y_\alpha - Y_\beta) - 1$,
- $c_\alpha = \sum_{\beta=1}^{m} C(X_\alpha - X_\beta)C(Y_\alpha - Y_\beta) - 1$

For the sake of completeness, a_α (b_α) is the rank of X_α (Y_α), while c_α is the number of observations of sample members (X_β,Y_β) for which both $X_\beta < X_\alpha$ and $Y_\beta < Y_\alpha$ (since they are assumed to be continuous we can say that they are at least different from each other).
In order to make a more robust analysis, I have decided to bootstrap, from each one of the datasets, by taking at each simulation a sample of 100000 elements, and cycled on it 200 times. Then, on each simulation, I have computed the p-value.
Below, Table 4.1, you can see the results.

	LL	LT	TL	TT
Kendall	0.25884	0.50551	0.45172	0.00000
Hoeffding	0.10544	0.14436	0.20984	0.00000

Table 4.1: P-value of the independence test

By looking at the p-value seems that almost all the marginal distributions can be considered independent.
The only dataset where an independence relationship cannot be accepted is TT.

4.3 Copula's analysis

In this section there will be a brief introduction of the Copula's theory (for more details [27]) and of the Non-parametric Copula theory (for more details [16]), which will be used to find the better density estimation of the distribution previously mentioned. Moreover there will be listed all the results achieved both the complete datasets and the datasets submitted to the cuts. More details on the bandwidth selection method used see [27].

4.3.1 Copula's Theory

Let $X = \{X_1, \ldots, X_n\}$ be a random vector with distribution F and with marginal distribution functions F_i, $X_i \sim F_i$, $1 \leq i \leq n$. A distribution function C with uniform marginals on $[0, 1]$ is called *Copula* of X if :

$$F = C(F_1, \ldots, F_n) \tag{4.10}$$

for the case that the marginal distributions are continuous the copula, C is the distribution function of $(F_1(X_1), \ldots, F_n(X_n))$. Since $F_i(X_i) \sim \mathcal{U}(0, 1)$, C is a copula and furthermore we obtain the following representation :

$$C(u_1, \ldots, u_n) = P\{(F_1(X_1) \leq u_1, \ldots, (F_n(X_n) \leq u_n)\} \tag{4.11}$$

$$= F_X(F_1^{-1}(u_1), \ldots, F_X(F_n^{-1}(u_n)) \tag{4.12}$$

where $F_i^{-1}(t) = \inf\{x \in \mathbb{R}^1; F_i(x) \leq t\}$ is the generalized inverse of F_i.
It is important to remark that any continuous random variable can be transformed to be $\mathcal{U}(0, 1)$ by its probability integral transformation, thus copulas can be used to provide multivariate dependence structure separately from the marginal distributions. Therefore a more general representation of a copula can be written as:

$$F(x_1, \ldots, x_n) = P\{X_1 \leq x_1, \ldots, X_n \leq x_n\} \tag{4.13}$$

$$= C(F_1(x_1), \ldots, F_n(x_n)) \tag{4.14}$$

The most important theorem of the Copula's theory is the Sklar's theorem which states that:

Theorem 4.3.1 (Sklar's theorem) *Let $F \in \mathcal{F}(F_1, \ldots, F_n)$ be an n-dimensional distribution function with marginals $F_1, \ldots, F_n$. Then exist a copula $C \in \mathcal{F}(\mathcal{U}, \ldots, \mathcal{U})$ with uniform marginals such that $F(x_1, \ldots, x_n) = C(F_1(x_1), \ldots, F_n(x_n))$.*
Moreover if each F_i is continuous then the copula C is continuous.

It states that any multivariate joint distributions can be written in terms of univariate marginal distribution functions and a copula which describes the dependence structure between the variables. It allows us to separate the modeling of the marginal distribution F_i from the dependence structure, which is expressed in C. Furthermore if $F(\cdot)$ and $C(\cdot)$ are differentiable the probability density function satisfies:

$$\frac{f(x_1, \ldots, x_n)}{f(x_1) \cdot \cdots \cdot f(x_n)} = c[F_1(x_1), \ldots, F_n(x_n)] \tag{4.15}$$

where c is the probability density function of C and it can be defined as follows:

$$c(u_1, \ldots, u_n) = \frac{\partial^n}{\partial u_1 \cdot \cdots \cdot \partial u_n} C(u_1, \ldots, u_n) \tag{4.16}$$

An Independence copula is defined as follows. Let's have $U_1, \ldots, \overset{i.i.d}{\sim} \mathcal{U}[0,1]$, we have :

$$P(U_1 \leq u_1, \ldots, U_n \leq u_n) = \prod_{j=1}^{n} u_j = \Pi(u_1, \ldots, u_n) \tag{4.17}$$

where Π is called *Independence copula*.
Sklar's theorem gives us a simple way to construct copula functions. Thus, by inverting the independence copula, we get :

$$C(u_1, \ldots, u_n) = F(F^{-1}(u_1), \ldots, F^{-1}(u_n)). \tag{4.18}$$

From this formula we can also obtain a representation of the corresponding copula density $c(u_1, \ldots, u_n)$:

$$c(u_1, \ldots, u_d) = \frac{f(F_1^{-1}(u_1), \ldots, F_n^{-1}(u_n))}{\prod_{j=1}^{n} f_i(F_j^{-1}(u_j))} \tag{4.19}$$

In order to fit the best parametric copula for our data we can use any parametric distribution functions F, such as Gaussian, Elliptical or Archimedean .

4.3.2 Non-Parametric Copulas

As mentioned before, the copula $C : [0,1]^2 \to [0,1]$ is a bivariate of the random vector (X, Y) which has uniform marginal distribution.
Let's assume that we have *i.i.d.* observations (X_i, Y_i) from a bivariate copula C and we are interested in the estimation of the corresponding density $c(x, y)$.
The easiest way is to apply the usual kernel density estimator to this kind of problem :

$$\hat{c}_n(x, y) = \frac{1}{n} \sum_{i=1}^{n} K_{b_n}(x - X_i) K_{b_n}(y - Y_i) \tag{4.20}$$

where $K_b(\cdot) = K(\cdot/b)/b$ is the kernel function and $B_n > 0$ is the bandwidth parameter.
However, in this particular analysis, this will be a problem because it will put a considerable amount of probability mass outside of the unit square. This implies that $\hat{c}_n$ is not a density function on $[0,1]^2$, since it does not integrate to one. Moreover the estimator will suffer from bias at the boundaries.
Many different approaches are used to tackle this problem, among which :

- Mirror-Reflection method (MR),
- Transformation method (T).

The MR method is an intuitive way to adapt the estimator $\hat{c}_n$ to make sure that it is a density on the domain of definition.
The idea is the following: gather all the probability mass that was put outside of the unit square and redistribute it back to $[0,1]^2$.
Thus all data are reflected at the corners and the edges of the boundary region and, by doing this, also the probability mass outside of the unit square gets reflected back to the interior.
The augmented dataset containing all the reflections is given by :

$$\begin{aligned}(\tilde{X}_{ik}, \tilde{Y}_{ik}) = \{&(X_i, Y_i), (-X_i, Y_i), (X_i, -Y_i), (-X_i, -Y_i), (X_i, 2 - Y_i), \\ &(-X_i, 2 - Y_i), (2 - X_i, Y_i), (2 - X_i, -Y_i), (2 - X_i, 2 - Y_i)\}\end{aligned} \tag{4.21}$$

and the *mirror-reflection estimator* is defined as follows :

$$\hat{c}_n^{(MR)}(x, y) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{9} K_{b_n}(x - \tilde{X}_{ik}) K_{b_n}(y - \tilde{Y}_{ik}) \tag{4.22}$$

The idea behind the T method is to transform the data so that it is supported on the full $\mathbb{R}^2$, instead of the unit cube as stated by the theory.
On this transformed domain, standard kernel techniques can be used to estimate the density. An adequate back-transformation then yields an estimate of the copula density. The most common choice for elaborate this transformation is the inverse of the standard Gaussian cumulative density function.
Denote Φ as the standard Gaussian cumulative density function and ϕ as its derivative. Then $(U_i, V_i) = (\Phi^{-1}(X_i), \Phi^{-1}(Y_i))$ is a random vector with Gaussian margins and copula C. By Sklar's theorem, the density f can be written as follows :

$$f(u, v) = c(\Phi(u), \Phi(v))\phi(u)\phi(v) \tag{4.23}$$

and this density can be estimated with a standard kernel estimator ($\hat{f}_n$).
Thus *Transformation estimator* is defined as follows :

$$\hat{c}_n^{T}(x, y) = \frac{\hat{f}_n(\Phi^{-1}(x), \Phi^{-1}(y))}{\phi(\Phi^{-1}(x))\phi(\Phi^{-1}(y)))} \tag{4.24}$$

4.3.3 Density Estimation

Here I will show all the results obtained by using parametric copulas and non-parametric copulas.
I have decided also to use non parametric copulas since the marginal distribution are not referable to a classic distribution, thus the estimation of the marginal in the model can cause difficulty and lead to biased estimations.
Since the tests made clearly show and independence in some cases I have decided to use Copulas only in the case where the aforementioned the H_0 hypothesis is rejected, since it is not necessary in order to evaluated the joint distribution given the independence of the distributions.
All the analysis were made by using once again R [19] and relaying of the packages `copula` [11], `VineCopula` [23] and `kdecopula` [16], which handle the non parametric analysis.
In order to build the parametric model the first step is to be able to identify the copula family, which will be used in the next computation. Since the datasets have too large dimensions the built-in functions in the packages are not able to evaluate the model, it will take too much time and the functions cannot allocate the vector due to memory issues.
So I have decided first to split the datasets into 10 parts and on those evaluate the families that can lead to the best approximation, choosing then the family that appears with more frequency.
After that, using the chosen family, I have evaluated on each chunk of data the Copula and I have averaged the estimated parameters in order to get the best estimates for the parameters that are about to be used to fit the copula.
Regarding the TT dataset, the chosen family is the *Rotated BB8 copula (90 degrees)*. This particular copula belongs to the family of the *Archimedean* copula, which are defined as follows :

$$C(u_1, u_2) = \varphi^{[-1]}((u_1) + \varphi(u_2)) \tag{4.25}$$

where $\varphi : [0, 1] \to [0, \infty]$ is a continuous strictly decreasing convex function such that $\varphi(1) = 0$ and $\varphi^{[-1]}$ is the pseudo-inverse :

$$\varphi^{[-1]}(t) = \begin{cases} \varphi^{-1}(t), & 0 \leq t \leq (0) \\ 0, & \varphi(0) \leq t \leq \infty \end{cases} \tag{4.26}$$

Moreover φ is called *generation function* of the copula C.
The BB8 copula, also known as Joe-Frank, is an Archimedean copula with two

parameters, θ and δ. The 90 degree rotation is needed since it allows the he modeling of negative dependence which is not possible with the standard non-rotated version.
Let's take $c(u_1, u_2)$ as copula density, thus the densities of the rotated version of this copula is given by $c_{90} = c(1 - u_1, u_2)$.
The generation function has the following structure:

$$\varphi(t) = -\log\left[\frac{1-(1-\delta t)^{\theta}}{1-(1-\delta)^{\theta}}\right] \tag{4.27}$$

where $\theta \in (-\infty, -1]$ and $\delta \in [-1, 0)$.
Below, Table 4.2 , it is shown the summary of the estimated copula :

Family	θ	δ	τ	AIC
30	-6	-0.13	-0.08	-37213.69

Table 4.2: Copula characteristics TT dataset

where τ is the value of *Kendall's tau* computed along with the copula.
After the family has been chosen the next step consists in choosing of the marginal that has to be used in order to create the random generator for the joint distribution.
I have decided to try with three different choices:

- uniform marginals : $\mathcal{U}[0, 1]$,
- beta marginal : $\mathcal{B}(\alpha, \beta)$,
- non parametric marginals.

The parameters in the beta distribution were computed by transforming the mean and the variance of the two cosine distributions into the parameters requested by a beta random variable.
The non parametric marginals, instead, were created by using a spline in order to get an approximation of the marginal distributions.
Below, Figures 4.12, 4.14, 4.16, show the 3D histograms and , Figures 4.13, 4.15, 4.17, the contour plots.

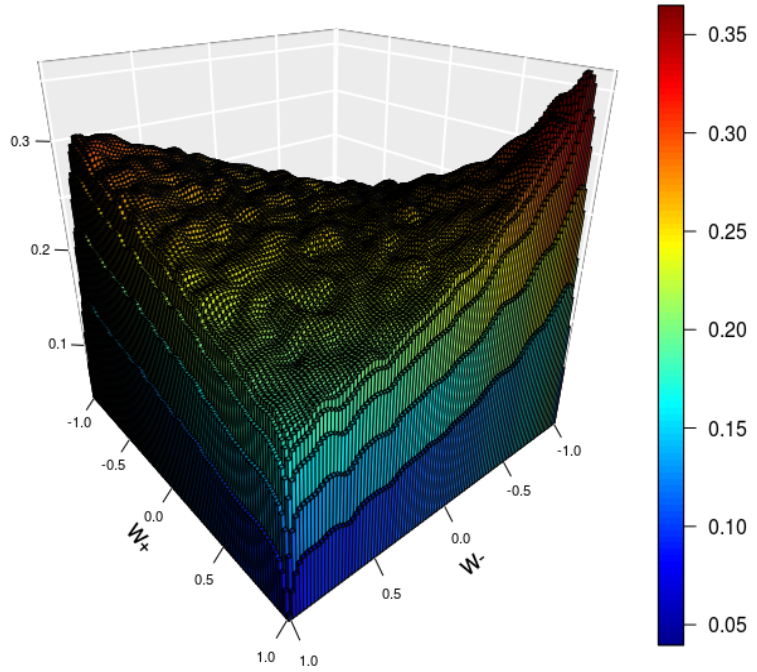

Figure 4.12: Histogram of the distribution (TT), Uniform marginals

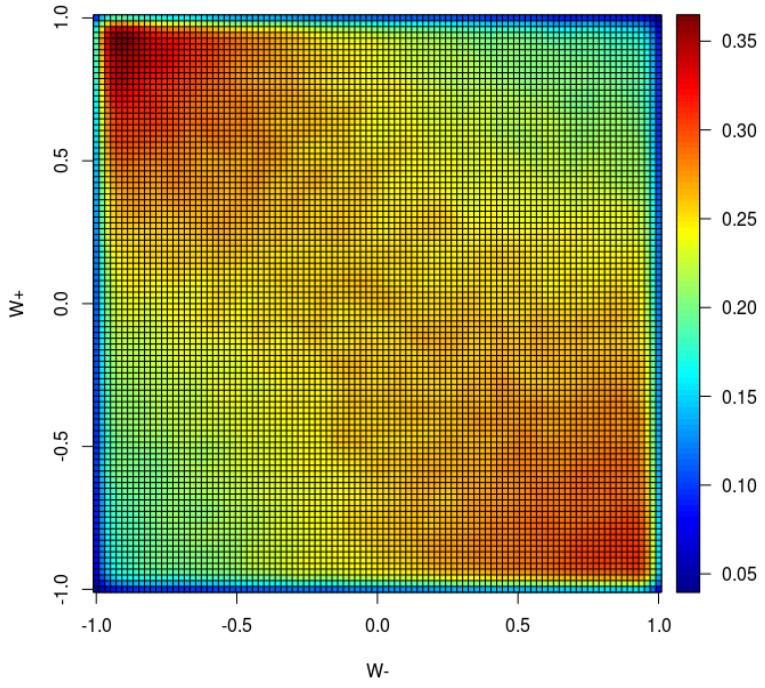

Figure 4.13: Image of the distribution (TT), Uniform marginals

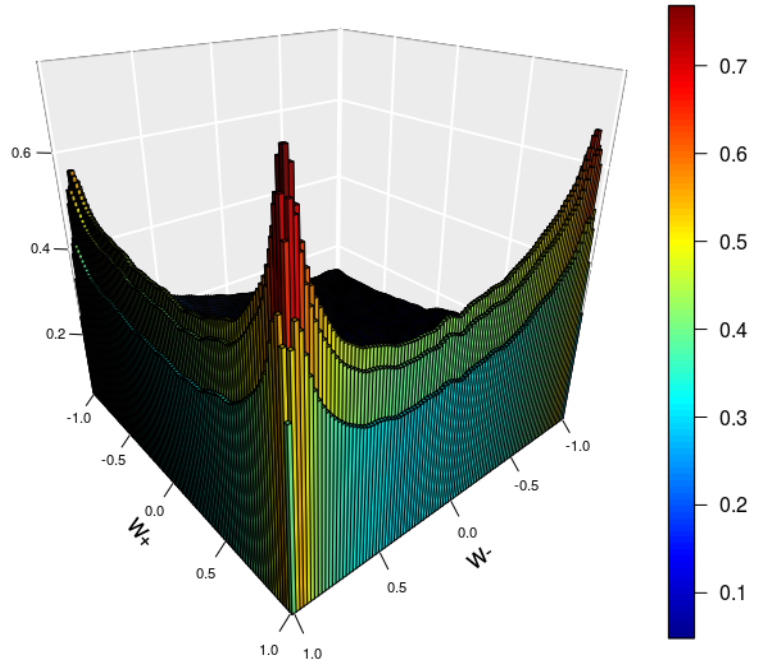

Figure 4.14: Histogram of the distribution (TT), Beta marginals

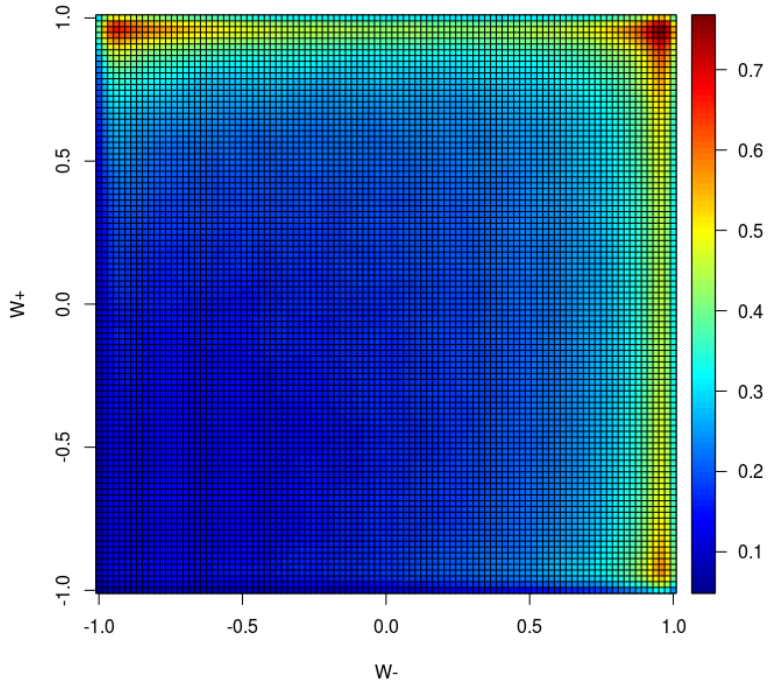

Figure 4.15: Image of the distribution (TT), Beta marginals

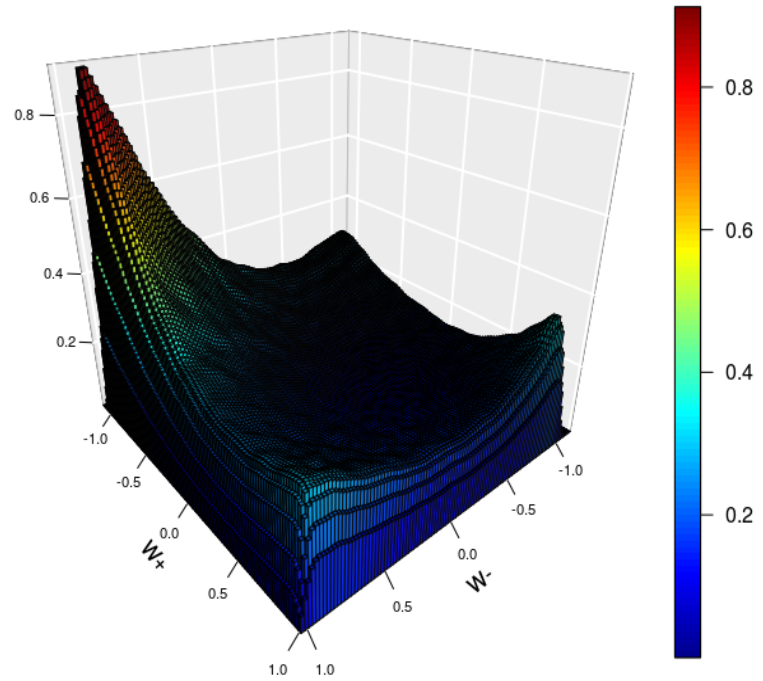

Figure 4.16: Histogram of the distribution (TT), NonParam marginals

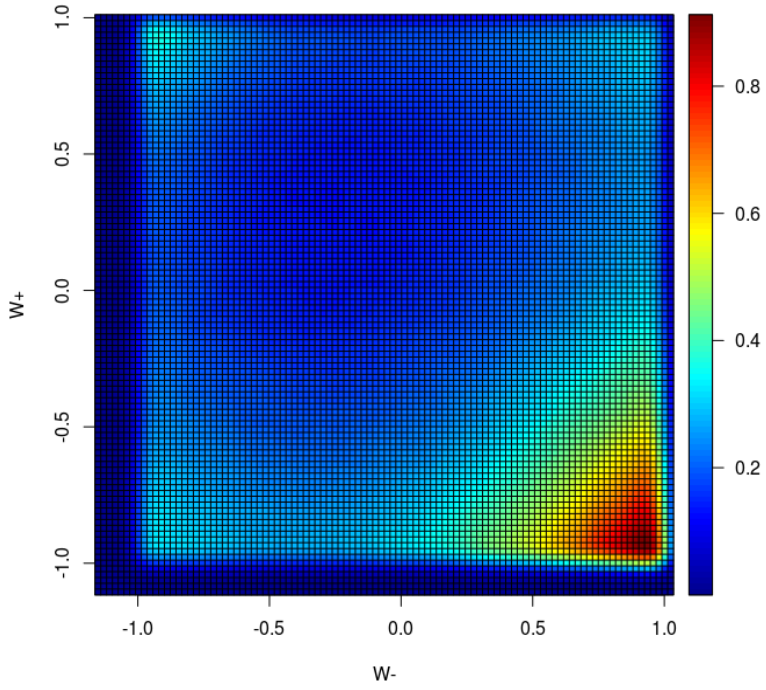

Figure 4.17: Image of the distribution (TT), NonParam marginals

As you can see the non parametric method is able to simulate almost correctly the joint distribution related to this dataset (see Figure 4.8).
As regards the estimation made by using the uniform marginals, the behaviour is almost correctly estimated, except for the peak in the north-west corner, which is not replicated in the results obtained.
Instead, the simulation obtained with Beta marginals is not able to replicate the original distribution.
In addition to those three results I have decided also to test the two non parametric method previously mentioned, Subsection 4.3.2, i.e Mirror-Reflection and Transformation method.
Below, Table 4.3, shows the summary of the fitted models:

	τ	AIC	BIC
MR	-0.056	-31654.95	-31430.16
T	-0.055	-32367.41	-32019.28

Table 4.3: Non Parametric Copula TT dataset

Below, Figures 4.18, 4.20, there are histograms of the estimates and, Figures 4.19, 4.21 , the contour plots.

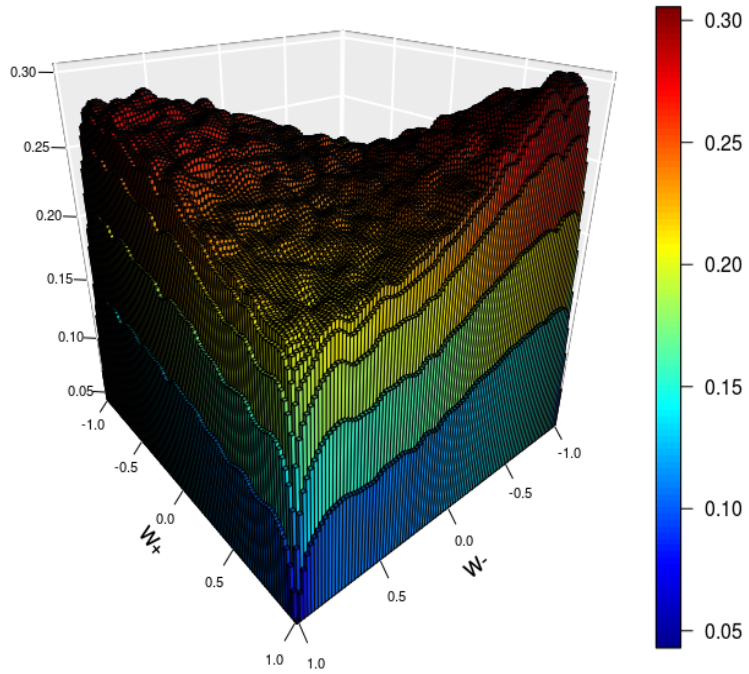

Figure 4.18: Histogram of the distribution (TT), MR method

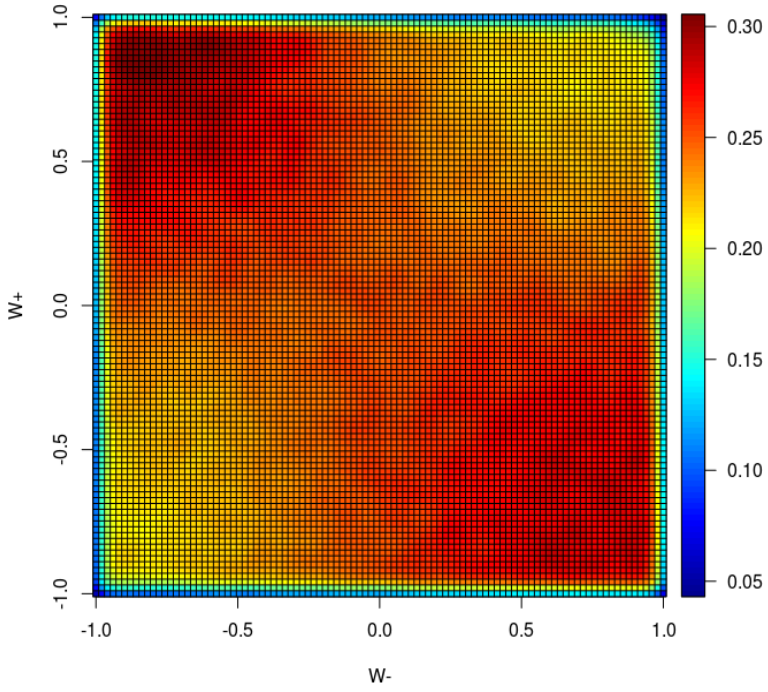

Figure 4.19: Image of the distribution (TT), MR method

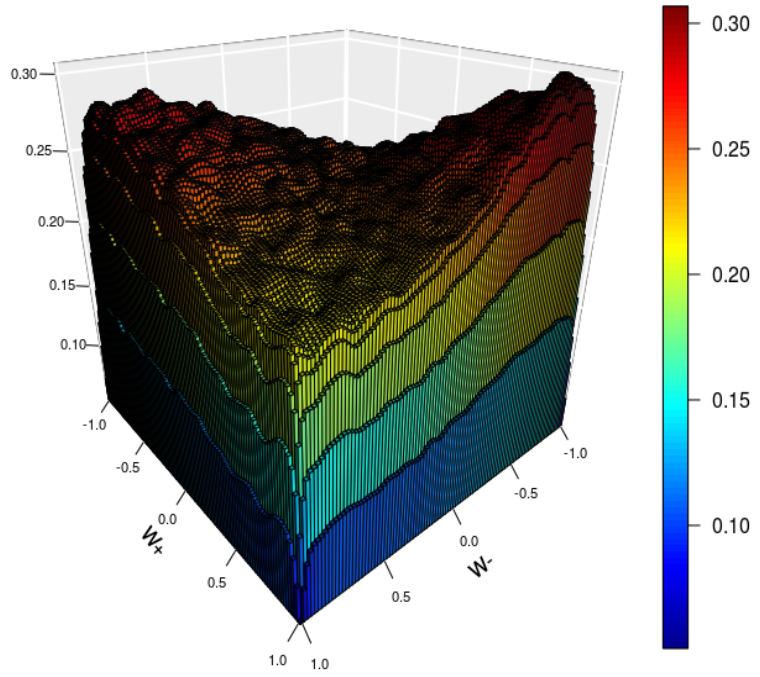

Figure 4.20: Histogram of the distribution (TT), T method

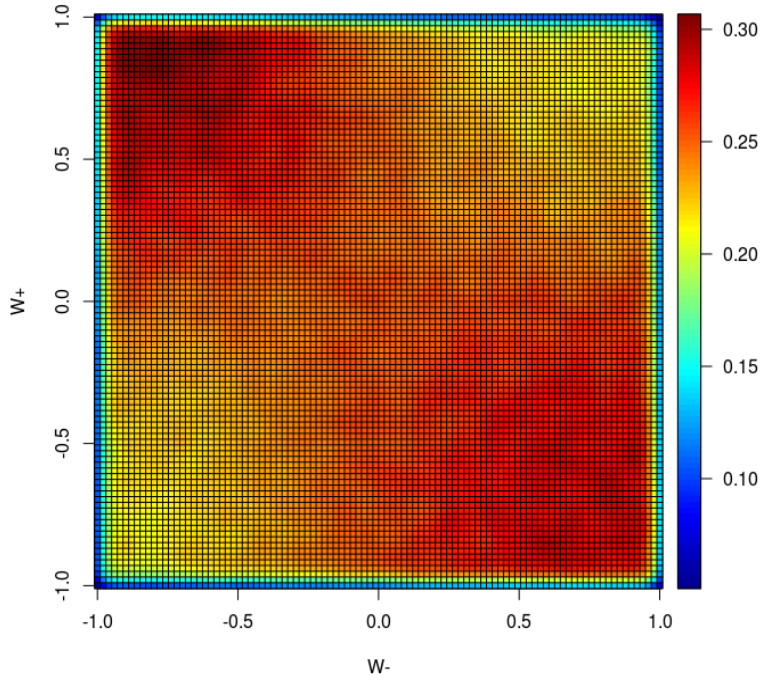

Figure 4.21: Image of the distribution (TT), T method

As expected by looking at Table 4.3, the estimated are very similar. In particular they show the same behavior as the simulation made with the Uniform marginals, i.e an adequate estimate of the general pattern and the inability to recreate the aforementioned peak in the north-west corner.

4.3.4 Cuts

Below I will show the results obtained in the analysis of the cut dataset, which are shown in Subsection 2.4.2.
Below, Figures 4.22, 4.24, 4.26, 4.28, 4.30 , you can find the 3D histograms and, Figures 4.23, 4.25, 4.27, 4.29, 4.31, the contour plot of the joint distribution built starting from the marginals, $\cos\theta_e$ and $\cos\theta_\mu$, as before.

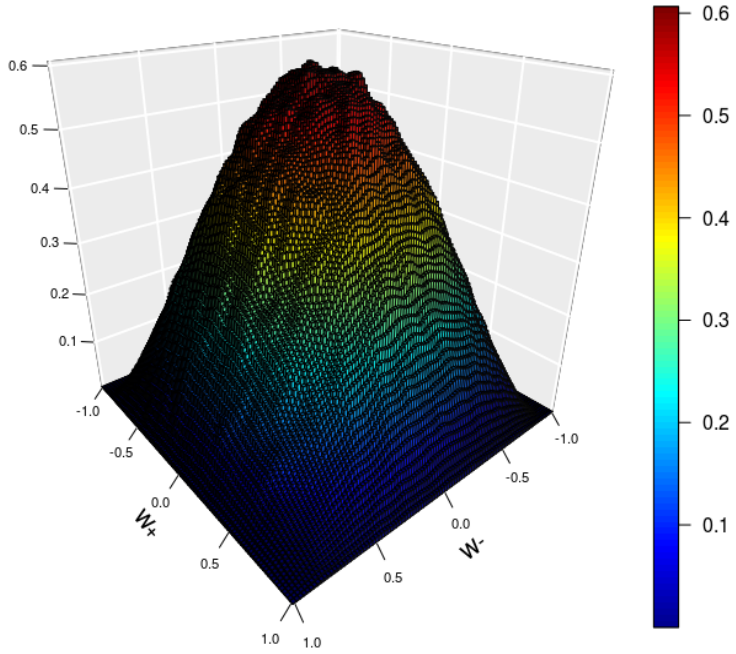

Figure 4.22: Histogram of the distribution (LL)

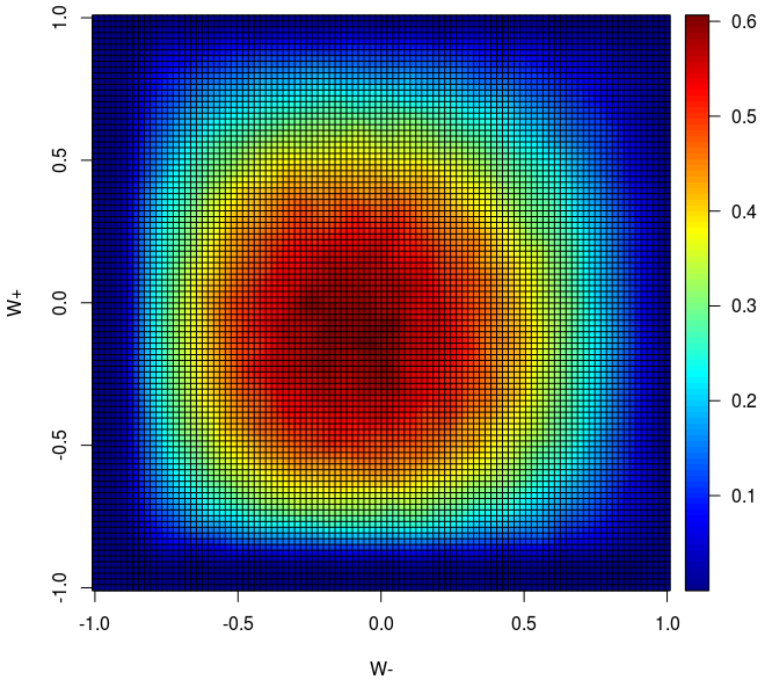

Figure 4.23: Image of the distribution (LL)

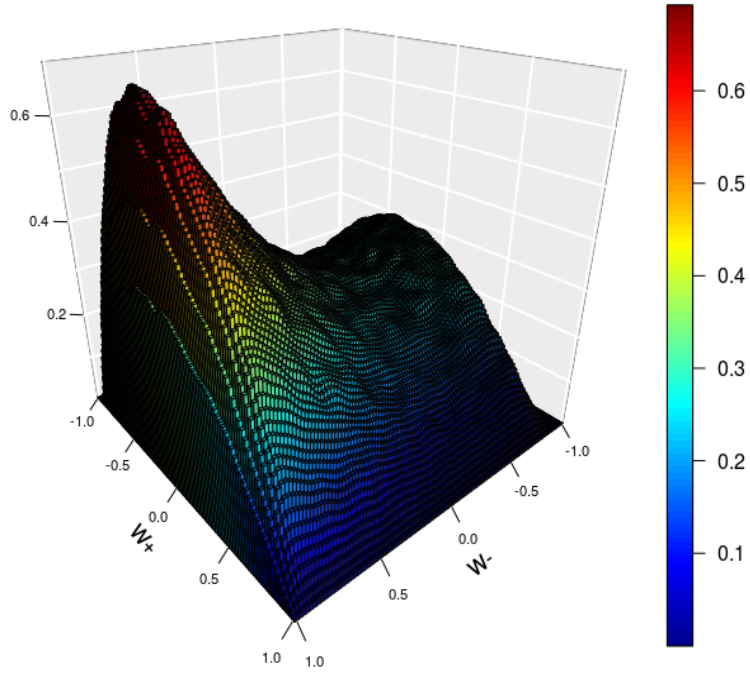

Figure 4.24: Histogram of the distribution (LT)

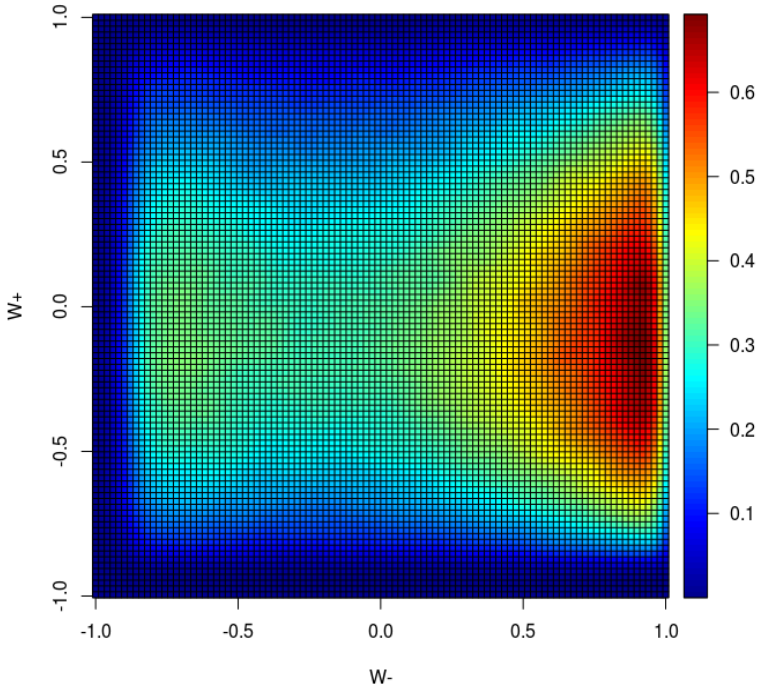

Figure 4.25: Image of the distribution (LT)

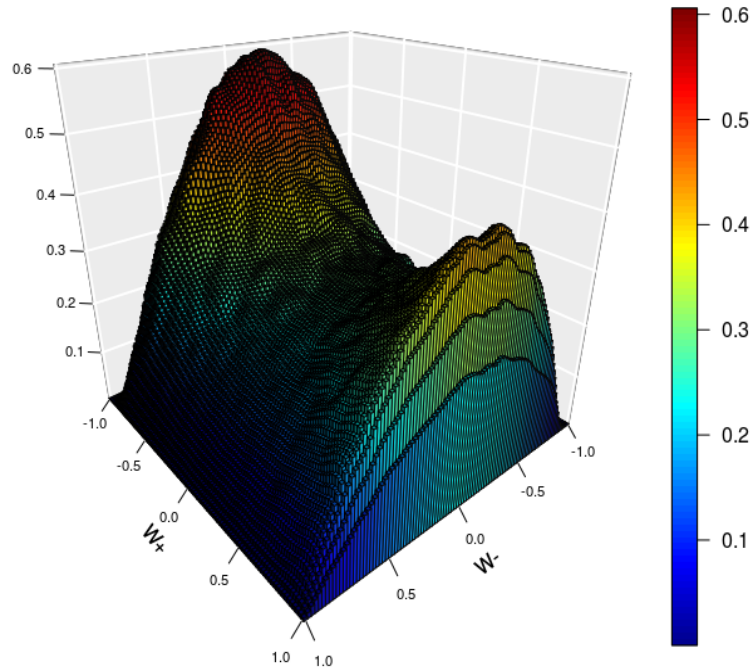

Figure 4.26: Histogram of the distribution (TL)

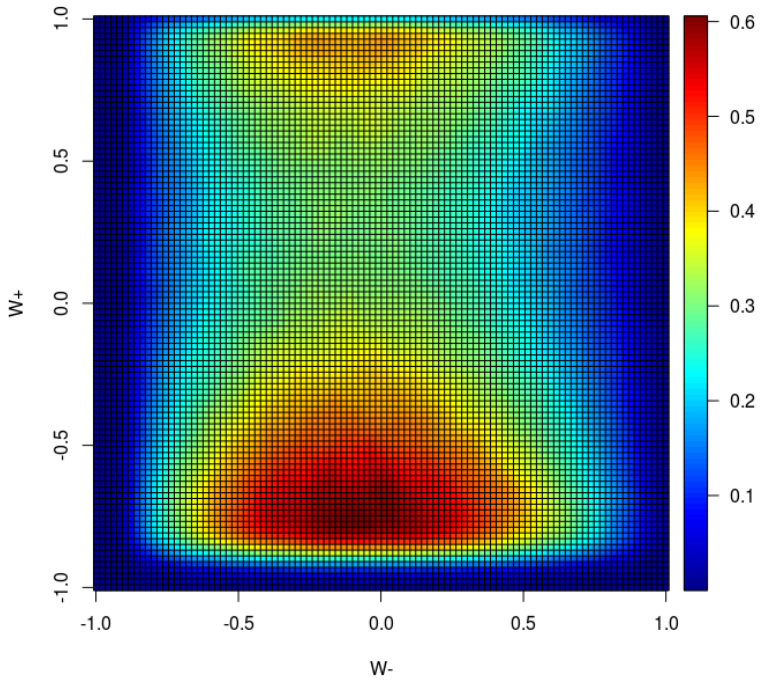

Figure 4.27: Image of the distribution (TL)

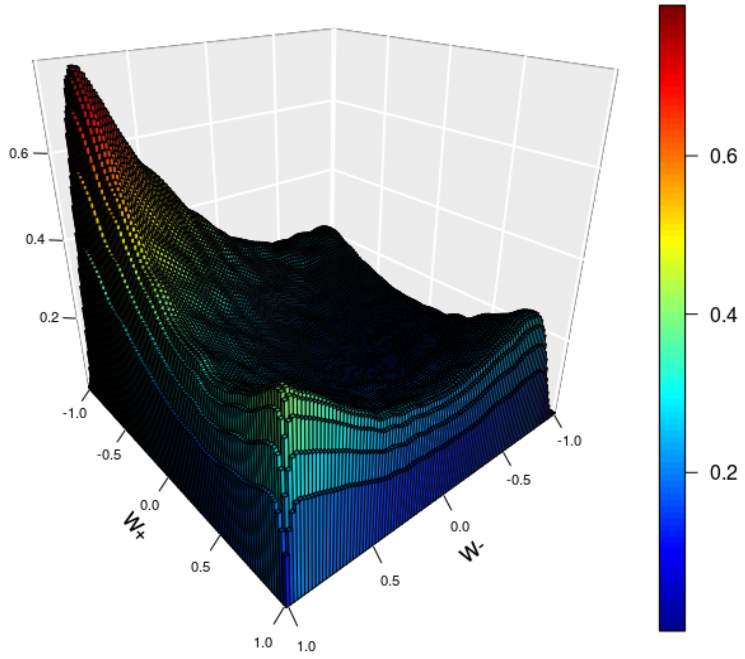

Figure 4.28: Histogram of the distribution (TT)

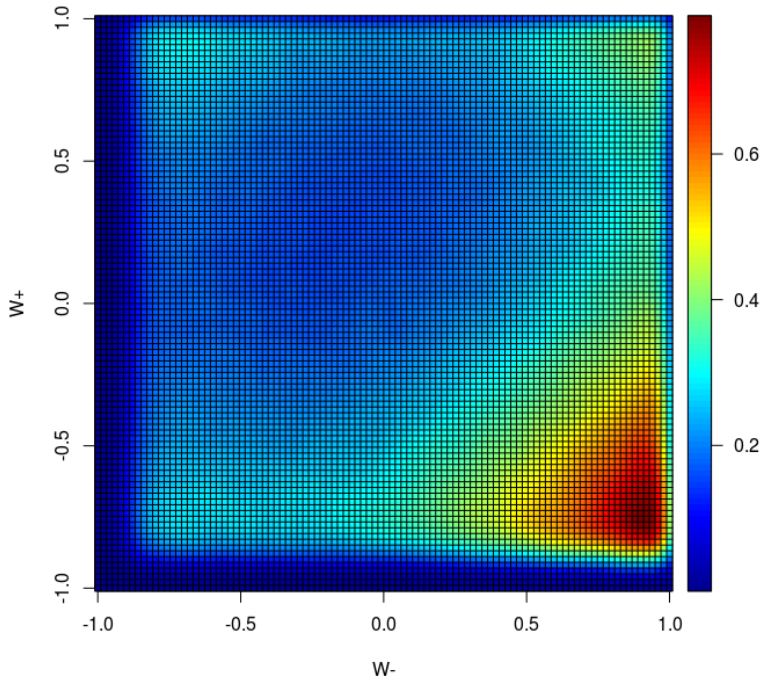

Figure 4.29: Image of the distribution (TT)

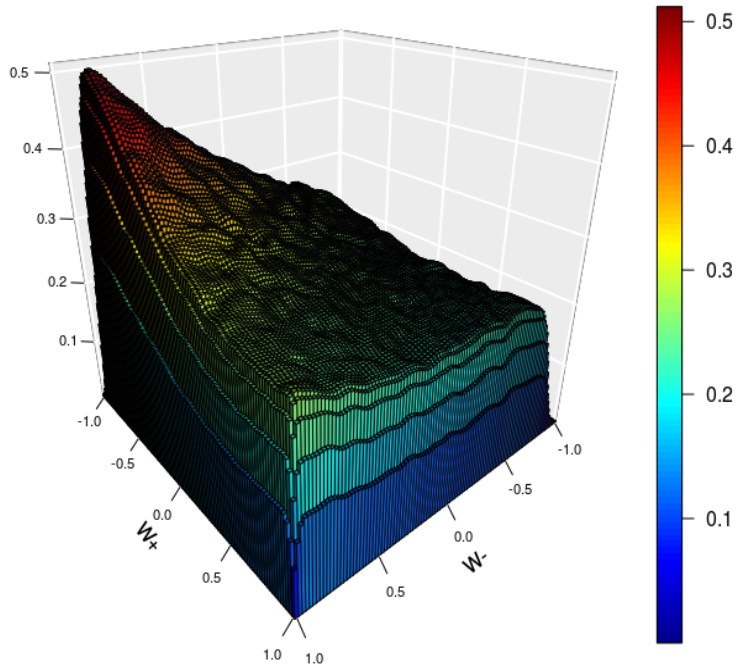

Figure 4.30: Histogram of the distribution (noHiggs)

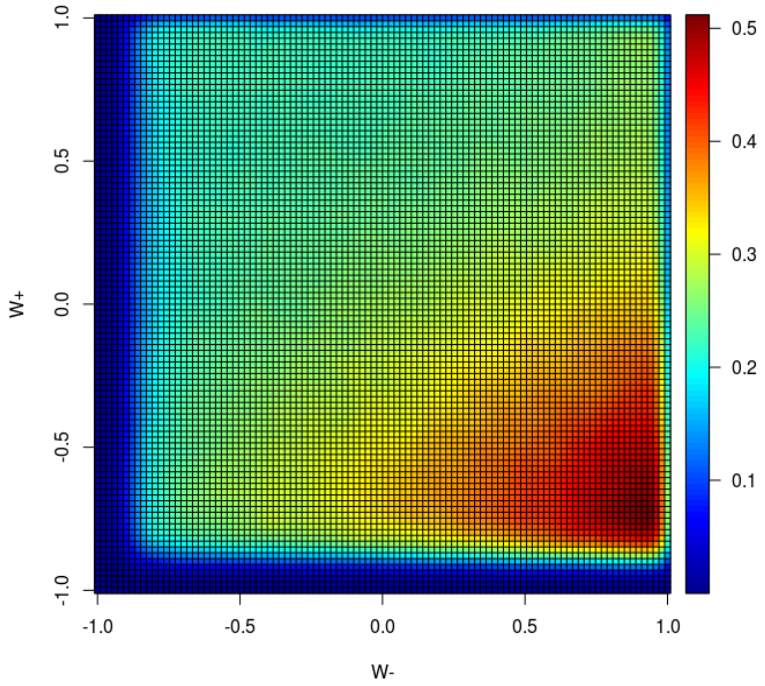

Figure 4.31: Image of the distribution (noHiggs)

The analysis will follow the procedure of the previous one, thus the first step will be the analysis of the independence of the distributions' variables of interest. I have decided to compare the p-value of the same tests used before, so *Kendall's Rank Correlation* test ad *Hoeffding's Independece* test.
Below, Table 4.4, you can see the results.

	LL	LT	TL	TT
Kendall	0.25884	0.50551	0.45172	0.00000
Hoeffding	0.10544	0.14436	0.20984	0.00000

Table 4.4: P-value of the independence test

The conclusion are the same found in the previous case, i.e the independence of the marginals in the first three cases and the presence of a dependence relationship in the Transversal-Transversal polarization datasets and in the Non-Polarized one, i.e `noHiggs`.

I have used once again the copula theory in order to evaluate the joint distribution. As before, since in case of independence the use of Copulas is useless, I have analyzed with it only the datasets that do have dependent marginals.
I have used the same techniques as before: firstly the selection of the fittest family and then the simulation of the parametric copula by relying on the same families for the marginals.
Below, Table 4.5, shows the statistics for the chosen family, once again the *BB8 copula, 90 degrees rotation.*

Family	θ	δ	τ	AIC
30	-3.15	-0.26	-0.08	-17298.27

Table 4.5: Copula characteristics Cut TT dataset

Below, as before, there are the histograms and contour plots, Figures 4.32, 4.34, 4.36 , made by relying on the parametric algorithms.

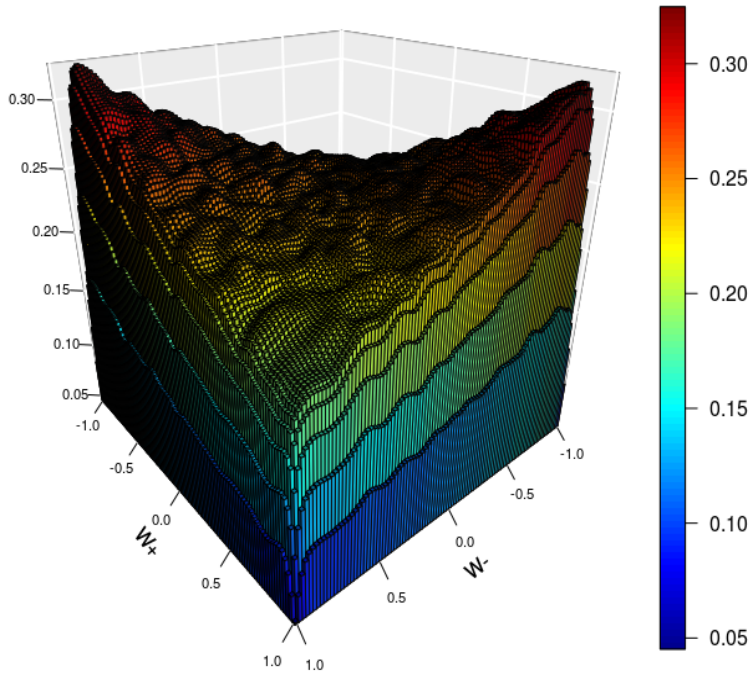

Figure 4.32: Histogram of the distribution (TT), Uniform marginals

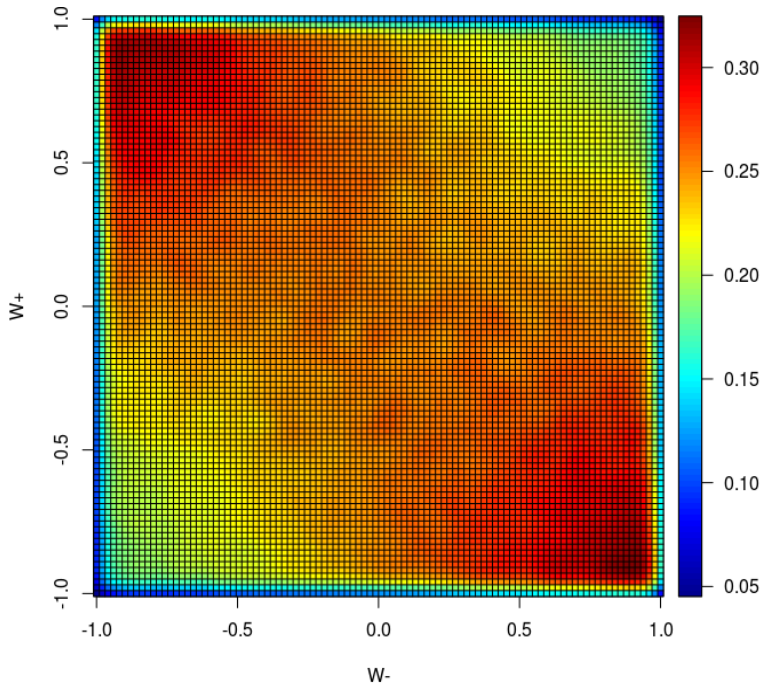

Figure 4.33: Image of the distribution (TT), Uniform marginals

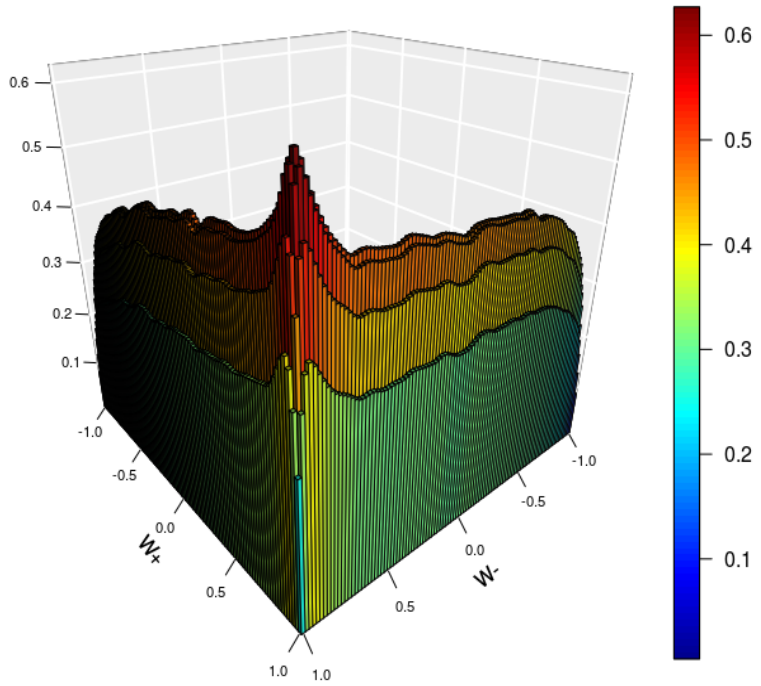

Figure 4.34: Histogram of the distribution (TT), Beta marginals

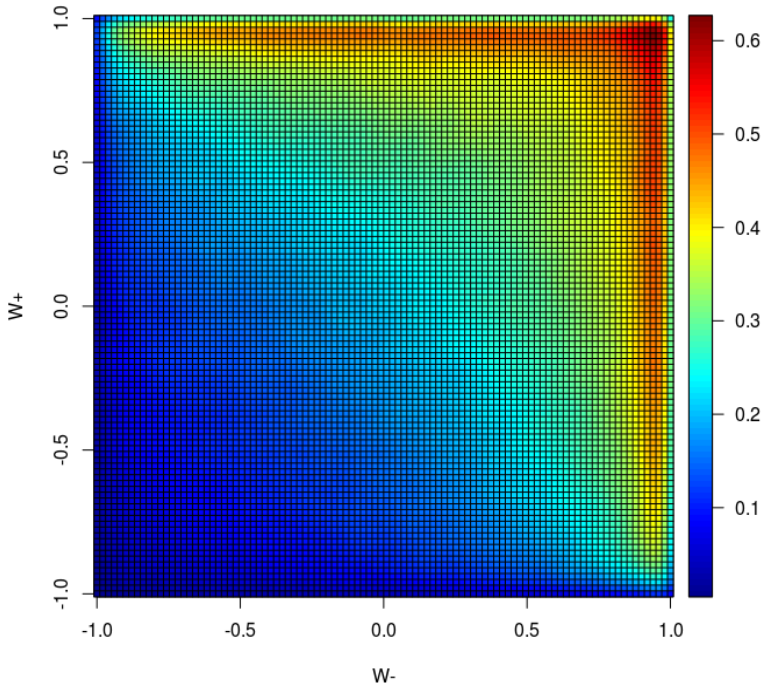

Figure 4.35: Image of the distribution (TT), Beta marginals

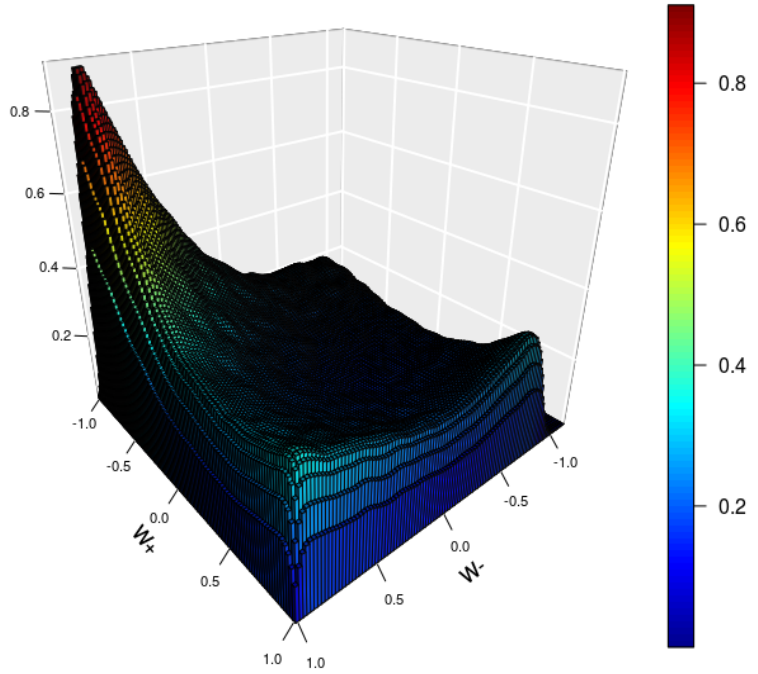

Figure 4.36: Histogram of the distribution (TT), NonParam marginals

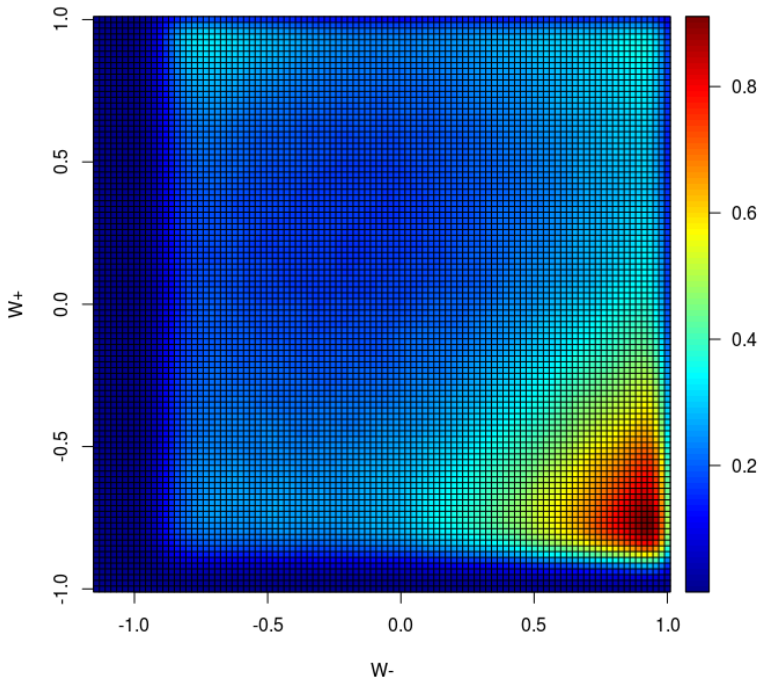

Figure 4.37: Image of the distribution (TT), NonParam marginals

As in the previous analysis seems that the method that is able to replicate the behavior shown in Figure 4.28 is the one in which I have used the non parametric marginals.
Regarding the other two estimates, they are not able to recreate completely the target distribution. The Uniform marginals seems not to be able to concentrate enough mass in the north-west corner, and the Beta marginals are simply inefficient.
Moreover I have tested also the non parametric copulas, using the same method presented before.
Below, Table 4.6, there are shown the summary of the fitted models:

	τ	AIC	BIC
MR	-0.064	-18864.2	-18694.68
T	-0.055	-17340.83	-17202.23

Table 4.6: Non Parametric Copula Cut TT dataset

Below, Figures 4.38, 4.40, there will be listed the histograms and, Figures 4.39, 4.41, the contour plots.

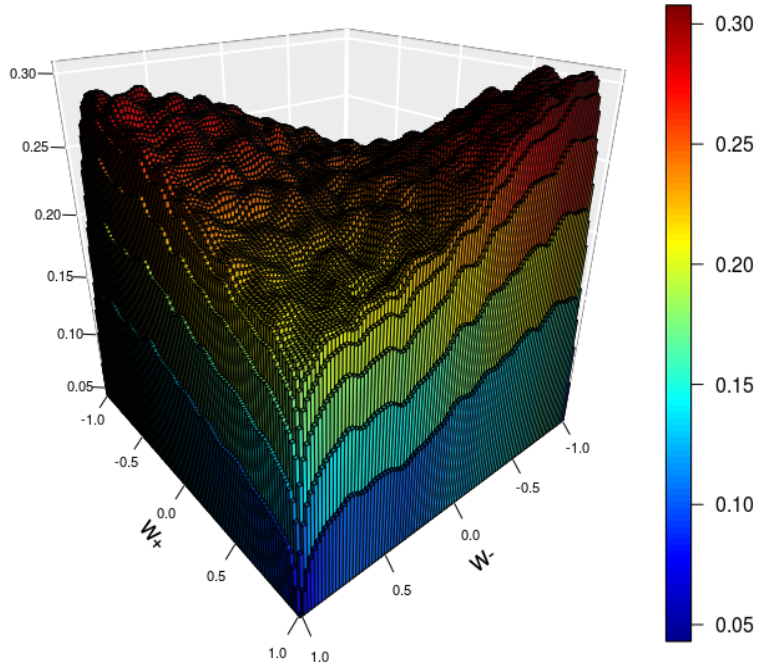

Figure 4.38: Histogram of the distribution (TT), MR method

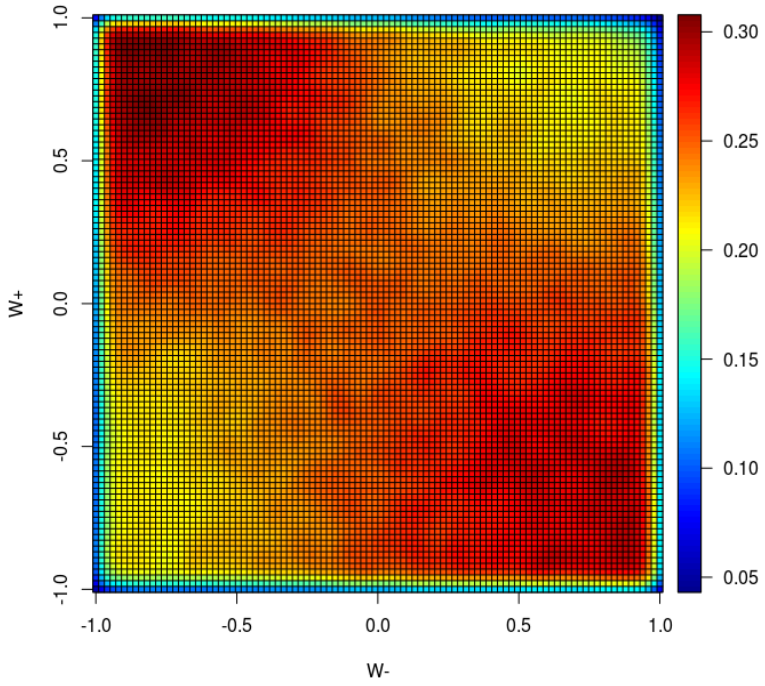

Figure 4.39: Image of the distribution (TT), MR method

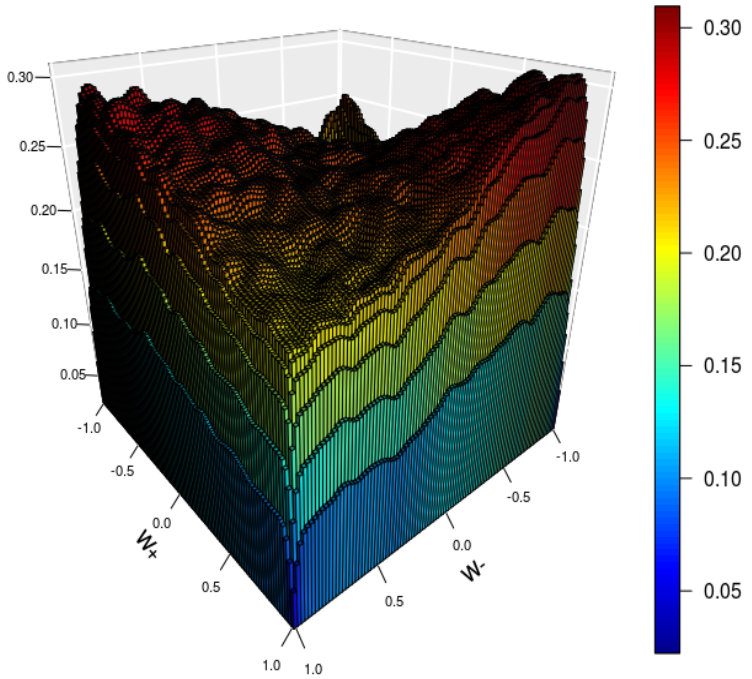

Figure 4.40: Histogram of the distribution (TT), T method

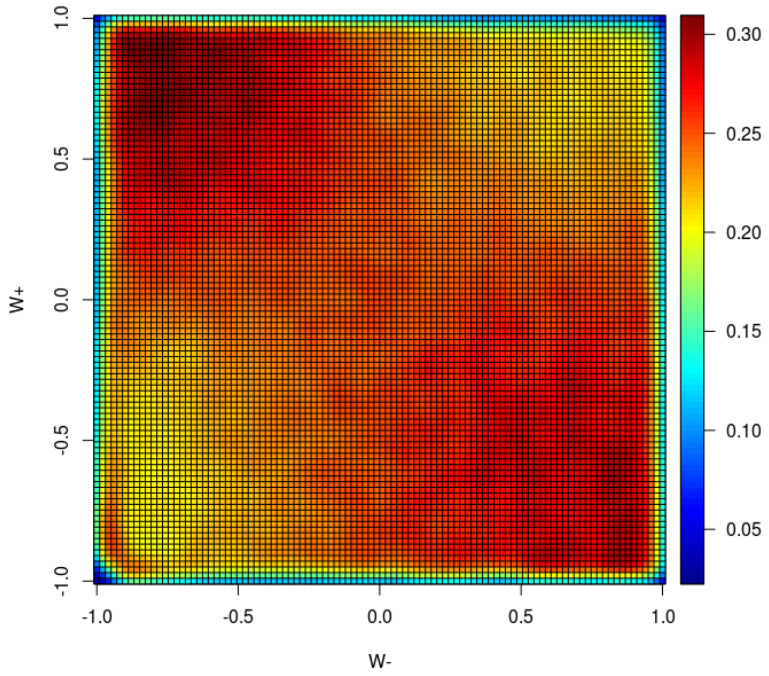

Figure 4.41: Image of the distribution (TT), T method

Both the estimates seems to show the same pattern, i.e the behaviour for the most of the domain is similar to the original, with the exception of the aforementioned peak in the north-west corner.

Chapter 5

Determination of the cross-section

The fifth chapter deals with the estimation of the unpolarized signal, i.e the bivariate distribution evaluated on the `noHiggs` datasets and the comparison with the reconstructed Standard Model dataset.

5.1 Linear Regression

As mentioned in the outline of the thesis, the last chapter will show the estimate found for the non-polarized. The bivariate distribution is found by computing the cosine of θ_e and $\theta_m u$, respectively for the W^- and W^+ boson, starting from the dataset made with the unpolarized event.
As told before, this signal can be computed by a linear combination of the four possible final polarized states of the W boson pair: transverse-transverse (TT), longitudinal-longitudinal (LL), transverse-longitudinal (TL) and longitudinal-transverse (TL), thus the signal can be evaluated as follows:

$$f_{NP}^{M}(x,y) = \sum_{i,j=0,T} \alpha_{ij}^{M} f_{ij}^{M}(x,y) + \alpha_{I}^{M} f_{I}^{M}(x,y) \tag{5.1}$$

where $x = \cos\theta_e$ and $y = \cos\theta_\mu$.
After obtaining the estimated for each bivariate densities, Section 4, I was finally able to compute the aforementioned distributions.
Since the signal is a linear combination of the three polarized distribution plus an interference term, the easiest way to reach the goal is to fit a linear regression model in order to evaluate the four coefficients, α_{ij}^{M}, using the estimated found as regressors.
It is important to remark that the coefficients are normalized with respect to the

cross-section of the unpolarized events.
Since the covariates and the response are matrices, in order to avoid computational complications, I have decided to transform them into vector, and after having fitted the models, to build them up again as matrices, so that I could be able to replicate the real form of the output variable to compare them with the data computed from the `noHiggs` datasets.
Moreover since I have found more than one estimate of the TT distribution I have decided to test them all as covariate, i.e I have fitted five linear models using as covariates the estimations carried out from the previous analysis and I have switched the covariates related to the estimation obtained regarding the TT dataset. This means that each of the models will have the same three first covariates (xLL, xLT, xTL) that are related to the dataset in which it is shown an independence between the marginal distributions, see Table 4.1 for more details, and a forth covariate (xTT) related to one of the different estimations obtained by using the results concerning the Copulas.
After that, I have compared these models in terms of *Mean Squared Error*, *Akaike Information Criterion* and *Bayesian Information Criterion* in order to select the best model.

5.1.1 Complete data results

Below, Tables 5.1, you can find the summary of the main characteristics of the estimated coefficients for each one of the models fitted:

	NonParam	Uniform	Beta	MR	T
MSE	0.002008	0.002123	0.002524	0.002257	0.002251
AIC	−33718.40	−33162.78	−31432.31	−32546.84	−32573.22
BIC	−33682.35	−33126.73	−31396.26	−32510.79	−32537.17

Table 5.1: Statistics from the fitted models : Complete dataset

As expected, the best method is the one relying on the Non Parametric marginals, despite there is a small difference between the evaluated values.
Below, Tables 5.2, there are the statistics regarding the coefficients of the selected linear model:

Coefficients	Estimate	Std.Error	t value	p-value
α_{LL}	-0.121573	0.003517	-34.56	$< 2e-16$
α_{LT}	0.387242	0.003752	103.22	$< 2e-16$
α_{TL}	0.425659	0.003459	123.06	$< 2e-16$
α_{TT}	0.315849	0.003921	80.56	$< 2e-16$

Table 5.2: Estimated Coefficients for `noHiggs`: Complete dataset

Below, Figure 5.1, there is shown the histogram of the fitted distribution from the regression model, and, Figure 5.2, there is the contour plot of the same fitted values, remembering that these estimation refers to the complete dataset, i.e the one without any cut applied.

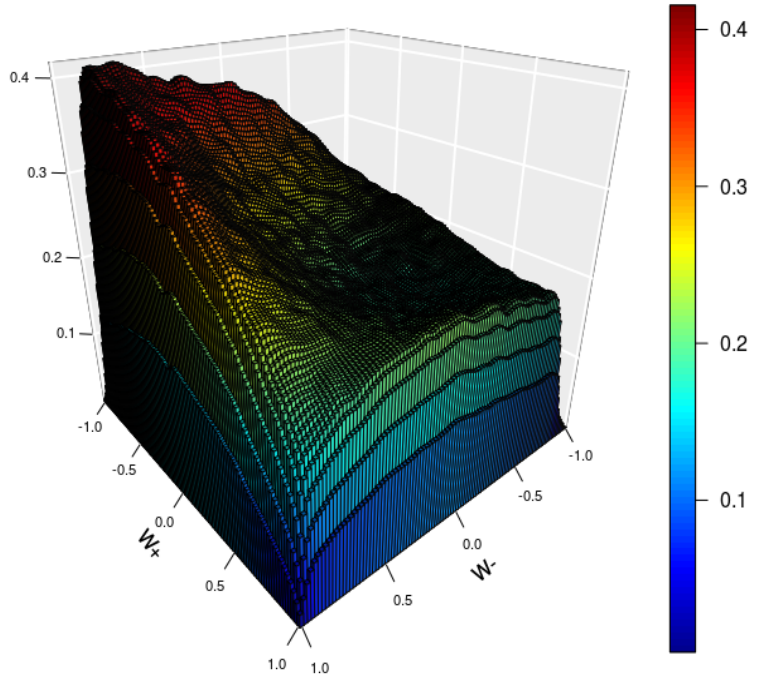

Figure 5.1: Histogram of the Estimated Distribution: Complete dataset

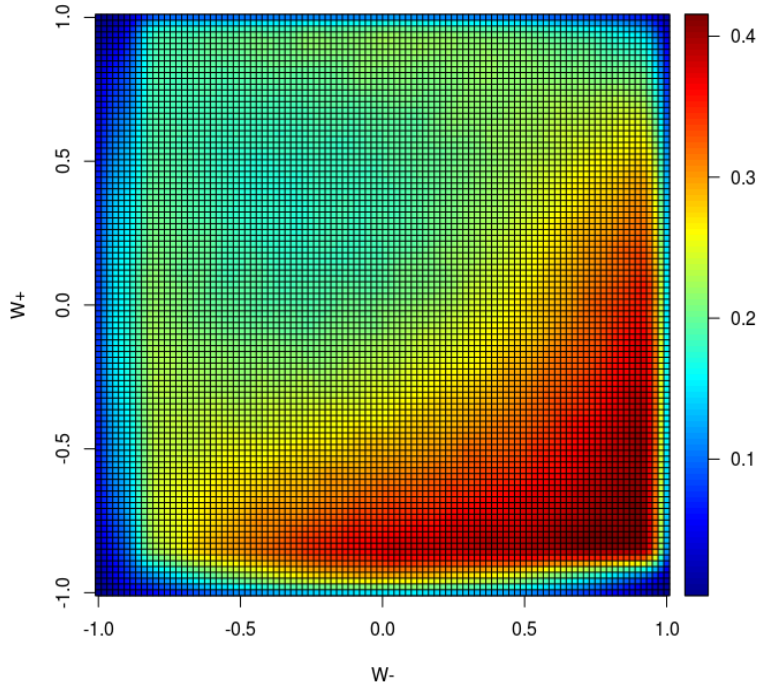

Figure 5.2: Image of the Estimated Distribution: Complete dataset

5.1.2 Cut data results

I have replicated the same analysis also for the Cut dataset, where the cuts are shown in Subsection 2.4.2.
Therefore, firstly, will be shown a Table regarding the estimation carried out from five different linear regression models in which, as before, the only covariate that changes is the one related to the TT dataset (xTT), since I want to test the goodness of the different estimations (see Subsection 4.3.4).
Below, Table 5.3, you can see the comparison of some statistics of the fitted model:

	NonParam	Uniform	Beta	MR	T
MSE	0.000433	0.001829	0.001709	0.001872	0.001867
AIC	−49062.29	−34649.46	−35326.97	−34418.03	−34446.69
BIC	−49026.24	−34613.40	−35290.92	−34381.98	−34410.64

Table 5.3: Statistics from the fitted models: Cut dataset

As expected the best model, in terms of all the statistics computed, is the one built by using the Non Parametric marginals.
The analysis will follow, using the regression model selected.
Below, Tables 5.4, there are the results obtained for the selected model:

Coefficients	Estimate	Std.Error	t value	p-value
α_{LL}	0.004820	0.002115	2.278	0.0227
α_{LT}	0.301055	0.002113	142.493	$< 2e-16$
α_{TL}	0.276960	0.002153	128.611	$< 2e-16$
α_{TT}	0.430845	0.002158	199.645	$< 2e-16$

Table 5.4: Estimated Coefficients for `noHiggs`: Cut dataset

Below, Figure 5.3, it is shown the histogram of the fitted distribution from the regression model, and, Figure 5.4, there is the contour plot of the same fitted values.

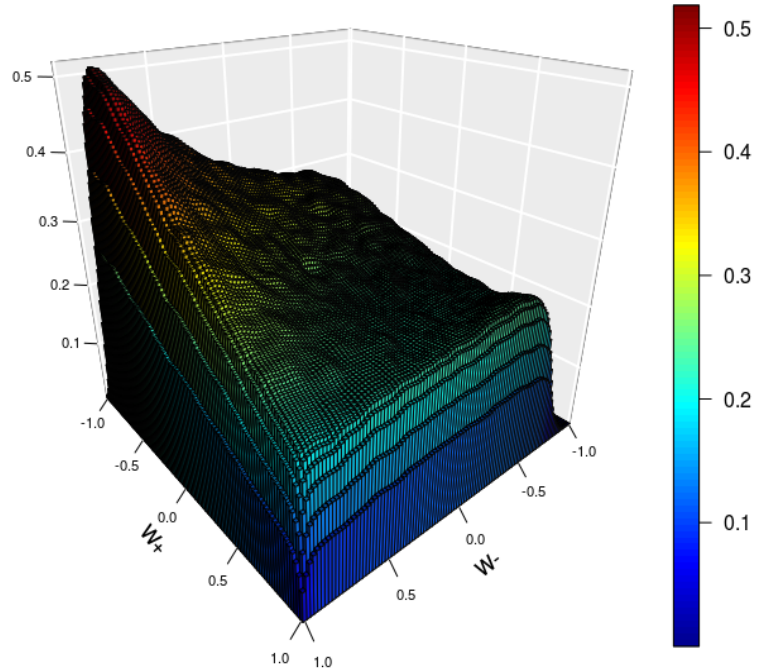

Figure 5.3: Histogram of the Estimated Distribution: Cut dataset

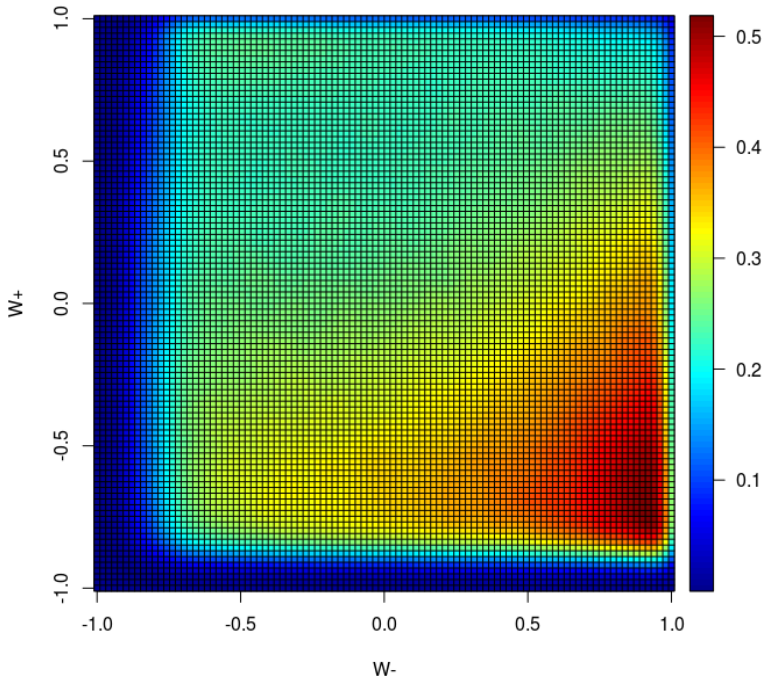

Figure 5.4: Image of the Estimated Distribution: Cut dataset

5.2 Comparison with the SM dataset

In order to evaluate the goodness of the estimated coefficients, I have decided to reconstruct the Standard Model dataset.
Therefore, since the data were not available, I have built the dataset starting from the value of the cross-sections given with the *LHE* files and the datasets characterized by the double polarization (see Section 4.1).
Since by (5.1), the unpolarized signal is described by the sum of the different polarized signal, I have extracted from each of the polarized datasets a certain amount of samples.
The size of these extractions is given by the percentage of the cross-section with respect to the normalized total cross-section.
Below, Table 5.5, you can see the aforementioned percentages:

	LL	LT	TL	TT
Cross-section percentage	0.05115813	0.16321047	0.17049171	0.61513969

Table 5.5: Cross-section percentages

The analysis are made with the complete and the data with the cut applied (see Subsection 2.4.2).

5.2.1 Complete data comparison

Therefore, by using the above results, I have built the SM dataset.
Below you can see the histogram, Figure 5.5, and the contour plot, Figure 5.6, of the distribution.

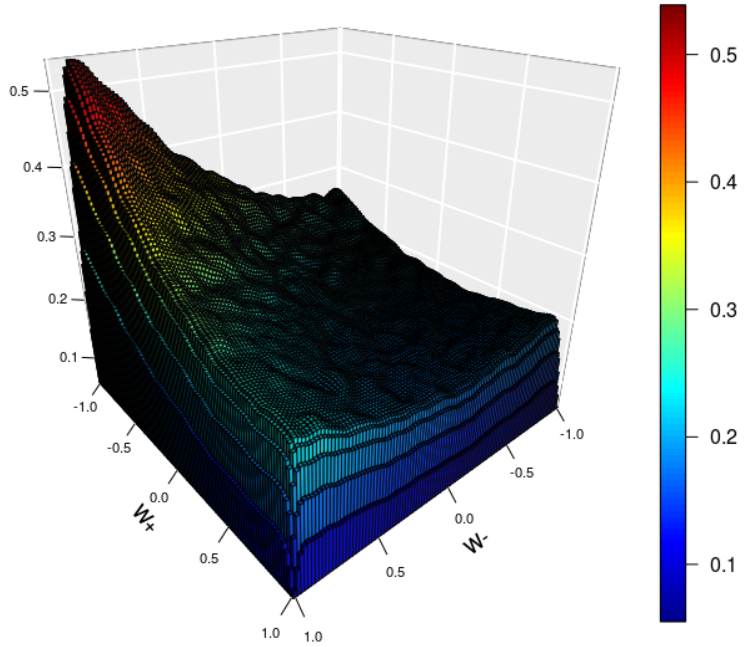

Figure 5.5: Histogram of the distribution (SM)

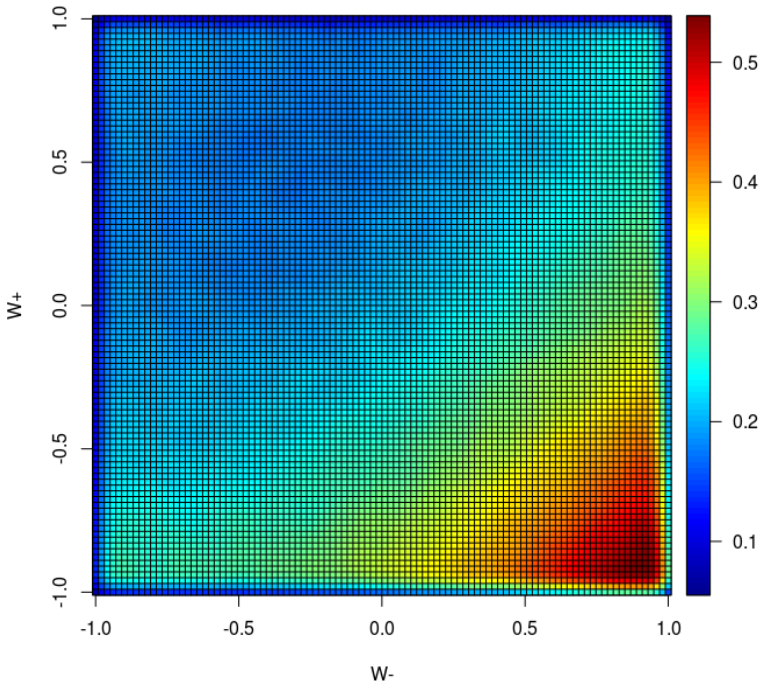

Figure 5.6: Image of the distribution (SM)

Once I have built the model I have used this distribution as response variable in the linear regression model, using the same covariates as before.
I have fitted the model using only the best estimate of the Transversal-Transversal distribution, i.e the one carried out with non parametric marginals.
Below, Table 5.6, you can see the results obtained:

Coefficients	Estimate	Std.Error	t value	p-value
α_{LL}	-0.192109	0.003796	-50.61	$< 2e-16$
α_{LT}	0.408629	0.004049	100.93	$< 2e-16$
α_{TL}	0.451037	0.003733	120.83	$< 2e-16$
α_{TT}	0.340475	0.004231	80.47	$< 2e-16$

Table 5.6: Estimated Coefficients for SM: Complete dataset

Below you can see the histogram, Figure 5.7, and the contour plot, Figure 5.8, of the distribution.

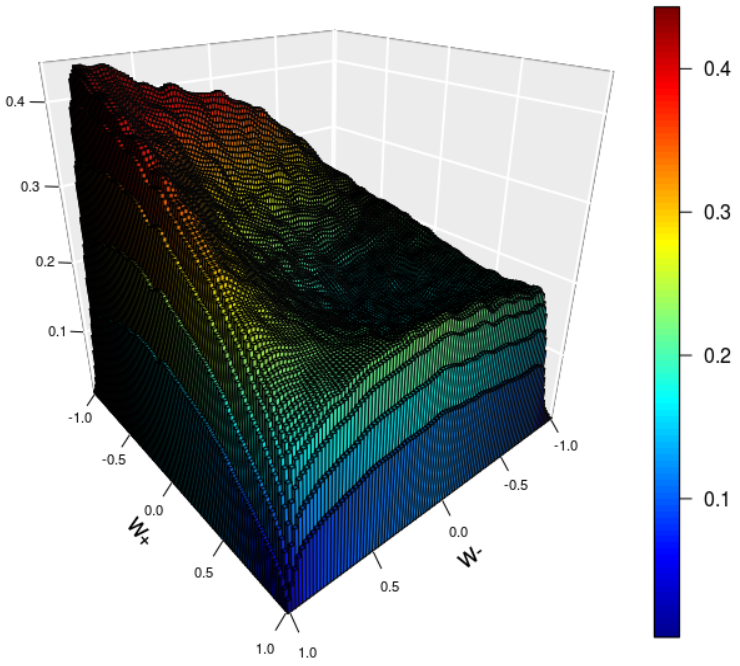

Figure 5.7: Histogram of the Estimated Distribution (SM): Complete dataset

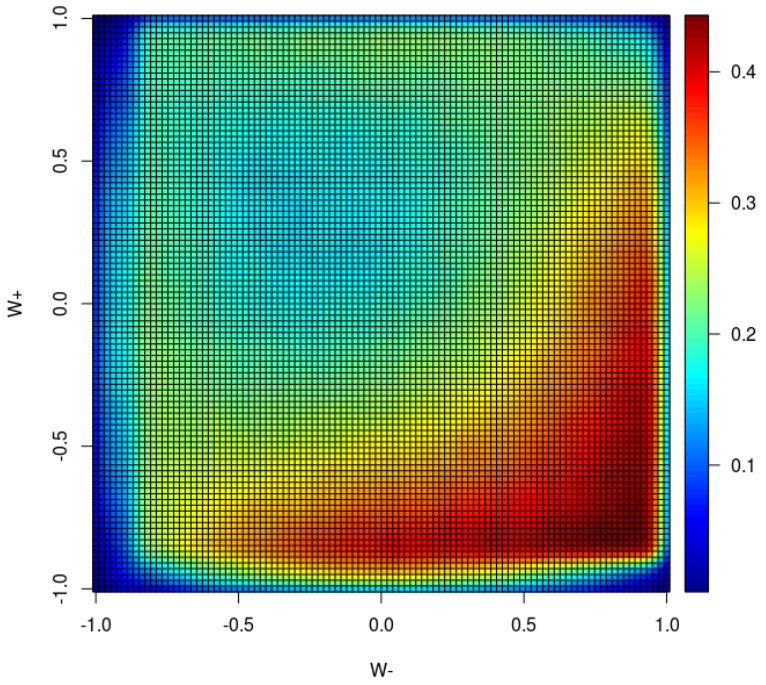

Figure 5.8: Image of the Estimated Distribution (SM): Complete dataset

By looking at the results and comparing them with the ones obtained by using the `noHiggs` dataset, it seems that all the estimates are very similar and they share the same behavior.
However, in both the cases, the coefficients related to the longitudinal-longitudinal polarized dataset (α_{LL}) are simply not correct.
In fact all the estimates should be at least positive and, in general, should be similar to the results show in Table 5.5.
Therefore this result highlights an error of the estimations carried out from the fitted linear regression model.

5.2.2 Cut data comparison

The same analysis were replicated for the Cut data.
As before, I have extracted the number of samples suggested by the percentage of the cross-section from the polarized datasets and built the Standard Model dataset for the Cut data.
Below you can see the histogram, Figure 5.9, and the contour plot, Figure 5.10, of the distribution estimated from the linear model.

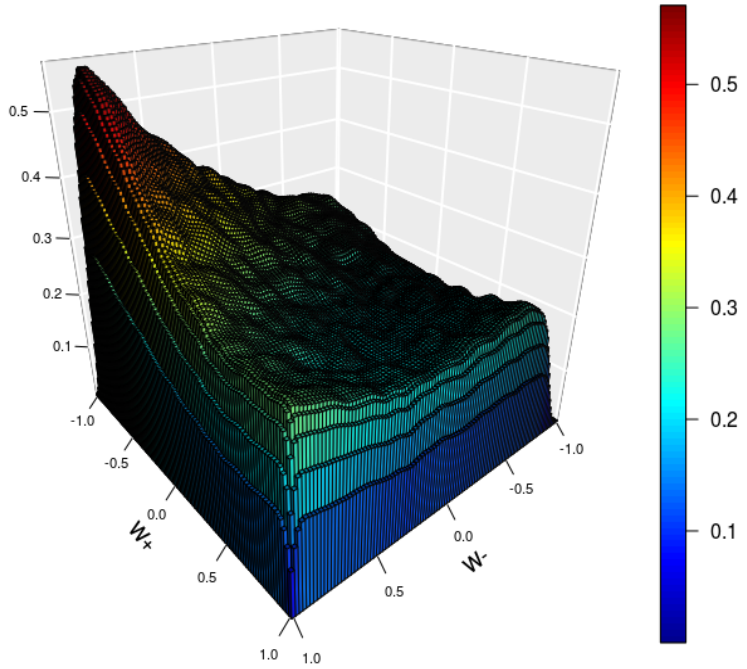

Figure 5.9: Histogram of the distribution (SM)

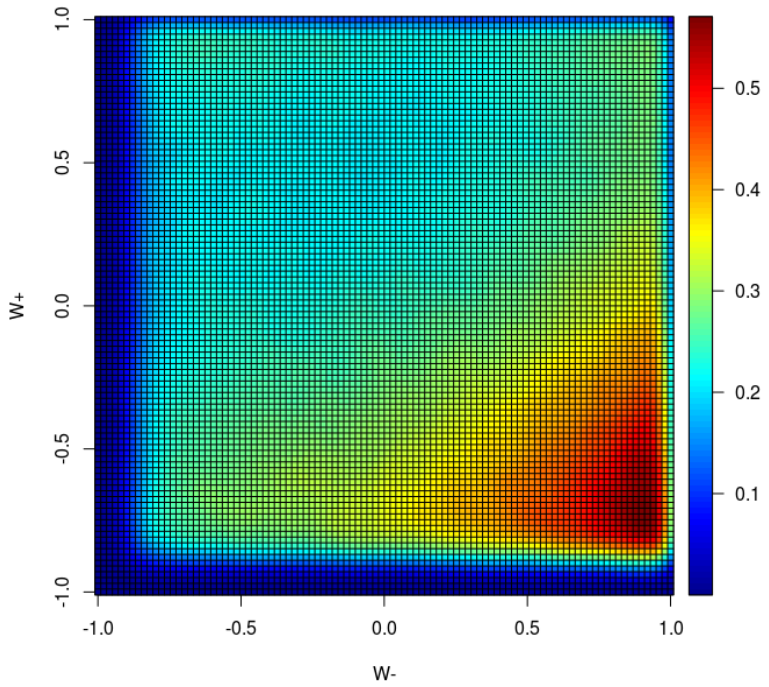

Figure 5.10: Image of the distribution (SM)

As before, I have fitted the model using only the best estimate of the Transversal-Transversal distribution, i.e the one carried out with non parametric marginals. Below, Table 5.7, you can see the results obtained :

Coefficients	Estimate	Std.Error	t value	p-value
α_{LL}	0.078364	0.002249	34.85	$< 2e-16$
α_{LT}	0.236443	0.002246	140.18	$< 2e-16$
α_{TL}	0.225829	0.002289	132.89	$< 2e-16$
α_{TT}	0.474931	0.002294	207.04	$< 2e-16$

Table 5.7: Estimated Coefficients for SM: Cut dataset

Below you can see the histogram, Figure 5.11, and the contour plot, Figure 5.12, of the distribution estimated from the linear model.

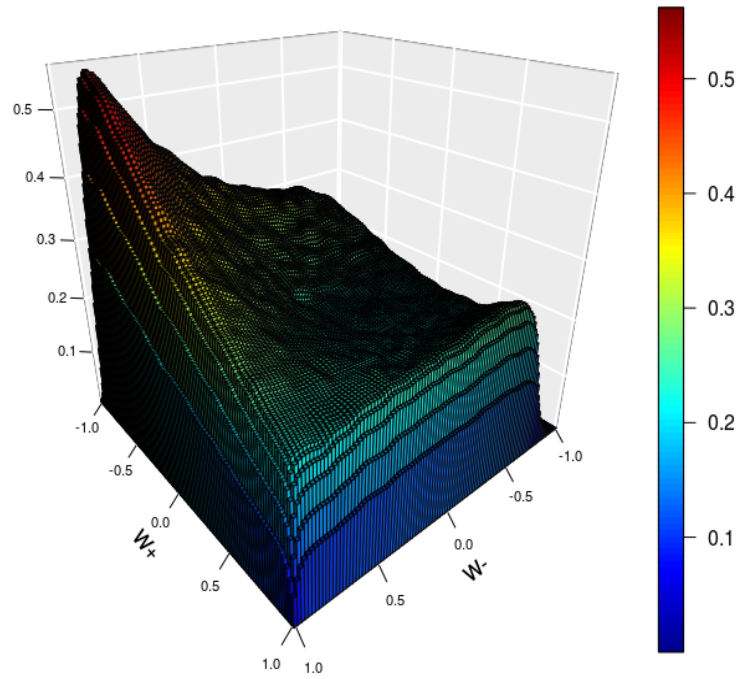

Figure 5.11: Histogram of the Estimated Distribution (SM): Complete dataset

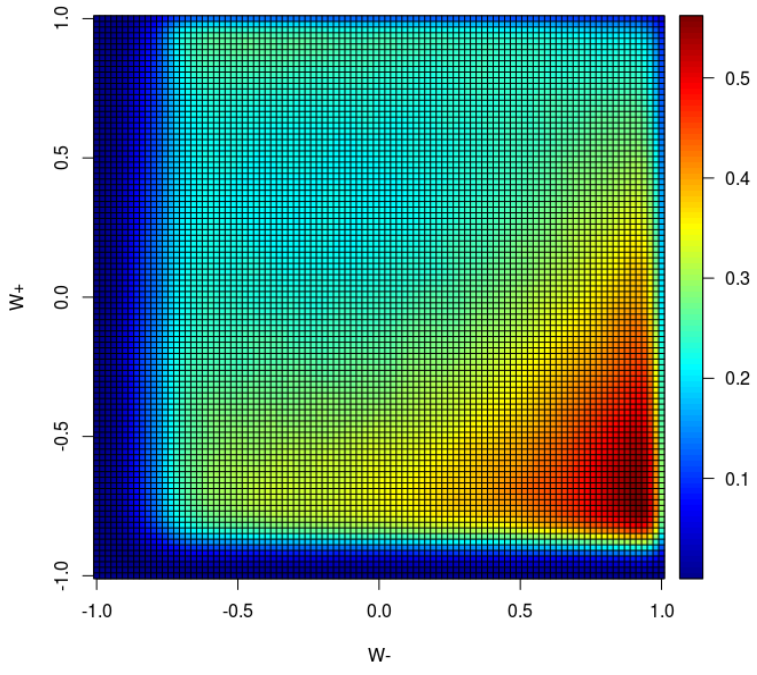

Figure 5.12: Image of the Estimated Distribution (SM): Cut dataset

By comparing these estimates with the one found by using the `noHiggs` dataset (Table 5.4), it seems that, using the two different response variable, the estimates do not show substantial modifications: all the estimates are positive, they seems to follow the cross-section percentages and they show the same behavior between them.
Thus it seems reasonable to conclude that, regarding the Cut dataset, the linear model is able to give an accurate estimate of the bivariate distributions.

Chapter 6

Conclusions

As regarding the first part, the choice of the free knot splines method seems very helpful in order to estimate the distributions of the invariant mass, but also the kernel density estimation approach has been proved to be valuable. Both the packages `freeknotsplines`[26] and `KernSmooth`[29] give us promising results, either in terms of goodness of fit and computational time, where the second package excels. The main issues with that part of the work was related to the selection of the quarks decaying from the boson, distinguishing them from the other couple, not relevant for these purposes.
By looking at the results, both plots and statistical tests used, seems that, in general, the methods were able to select the right pair of jets, i.e we can find in the computed distributions three peaks, corresponding to the mass of the three known bosons, W,Z, Higgs, highlighting the fact that the methods are able to correctly select the couple of quarks coming from the decay of the boson.
The only issues are shown with the third method and the QCD dataset, where it is shown that the algorithm fails to reach the goal of the analysis.
However the result shown is interesting for a purpose different than the one intended. In fact, besides claim that the transversal momentum is not able to divide correctly the couple of jets, it shows that the p_t of the products of the decay boson vector is basically higher in the case of signal, while this is not always the higher in the case of noise. This could be an indirect way to say that we have a variable that is able to distinguish the noise signal.
Thus, despite the inefficiency of the algorithm, this could lead to a new instruments that can be used for separating the signal from the contamination one.
One other issues was the computational time regarding the selection of the pair couple. All four methods take very long time to complete the algorithms, even

relying on a parallel version of them. I think that one possibility is to write down the code on a different language such as C++, in order to create the right object that could be able to store correctly the data and computes all the calculations relying on a MPI version of the problem, in order to speed up the process. A different solution could be to implement with `RSpark` the function needed.

The main goal of the second part of the thesis was to correctly estimate the unpolarized signal and the estimation of the cross-section related to them.
All the analysis were made by considering the decay of each of the boson that characterized the process, thus this leads to a statistical analysis with bivariate distributions, in which the marginal distributions are the cosine of the polar angle for the electron and the muon, i,e the particles in which the W^- and the W^+ naturally decay.
In order to estimate the joint distribution between the variables of interest I have used Copula's theory, which is a technique used to estimate joint distributions. Both the packages `copula`[11] and `VineCopula`[23] are reliable tools to this analysis, since they provided a large sample of functions useful to select the best copula family and establish the goodness of fit of the selected family. For these particular datasets, it was difficult to find a estimation that could simulate the correct behavior of the target distributions, given the unusual marginal distributions. However the use of non parametric marginals seems to overcome this difficulty. The `kdecopula`[16], despite the fact that they relay on a non parametric approach, has shown the same difficulty remarked before for the other packages. In addition to this it was not efficient in terms of computational time, since the functions need all the data used at the same time, leading to several hours of computations.
The estimation of the cross-section gives us conflicting results.
As regards the Cut data, all the estimate seems to be reliable and also the comparison between the `nohiggs` dataset and the Standard Model dataset gives us very similar results.
Instead, the analysis made on the complete data, highlight an important error in the estimation of the coefficients, despite the goodness of the statistics use to evaluate the models. All the coefficients should be similar to the cross-section percentage given by the double polarized datasets, but this is not shown for the coefficients related to the LL dataset (α_{LL}). This coefficients is actually negative which is a nonsense for the analysis that were made. This results is shown in each model fitted, for both the `noHiggs` and the SM dataset.
This could lead us to use another approach, different from the standard linear

regression model, in order to correctly estimate the cross-sections.

Appendix A

Figures

In Appendix A, I will list a series of plots that were not introduced during the chapter before (see Subsection 3.2.2), in order to make more linear the elaborate and because there were not meaningful to the description of the work.

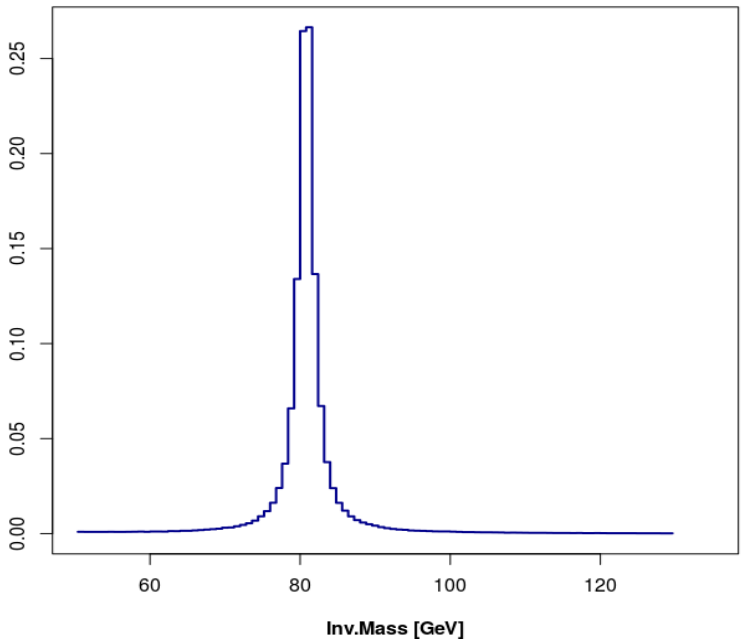

Figure A.1: Invariant Mass Muonic-Lepton couple (EWK)

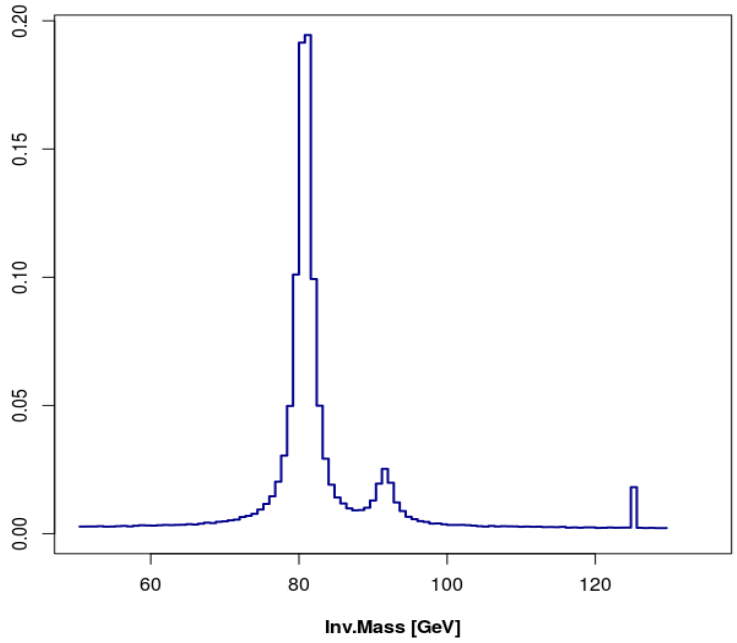

Figure A.2: Invariant Mass Distribution (EWK): Method 1

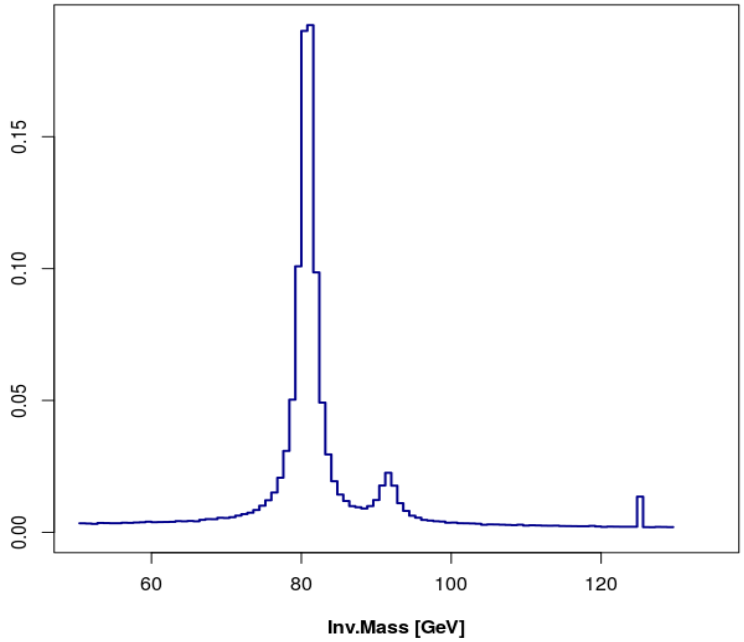

Figure A.3: Invariant Mass Distribution (EWK): Method 2

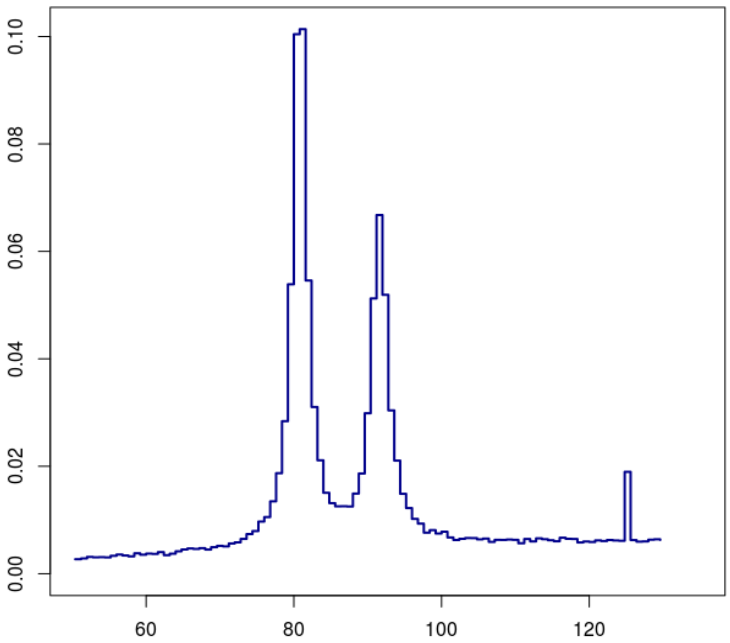

Figure A.4: Invariant Mass Distribution (EWK): Method 3

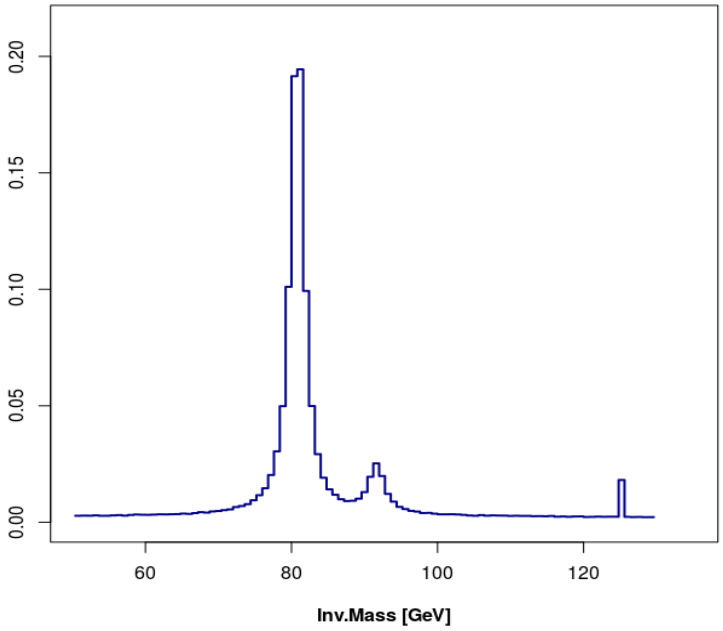

Figure A.5: Invariant Mass Distribution (EWK): Method 4

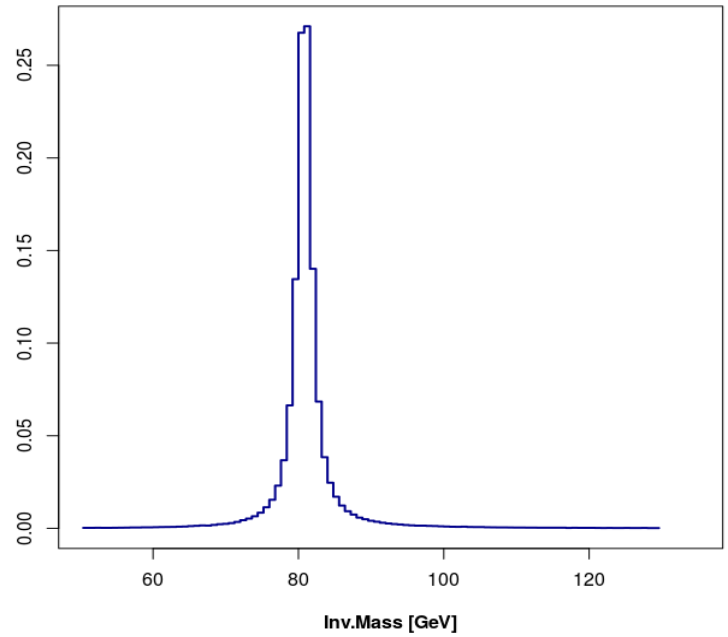

Figure A.6: Invariant Mass Muonic-Lepton couple (QCD)

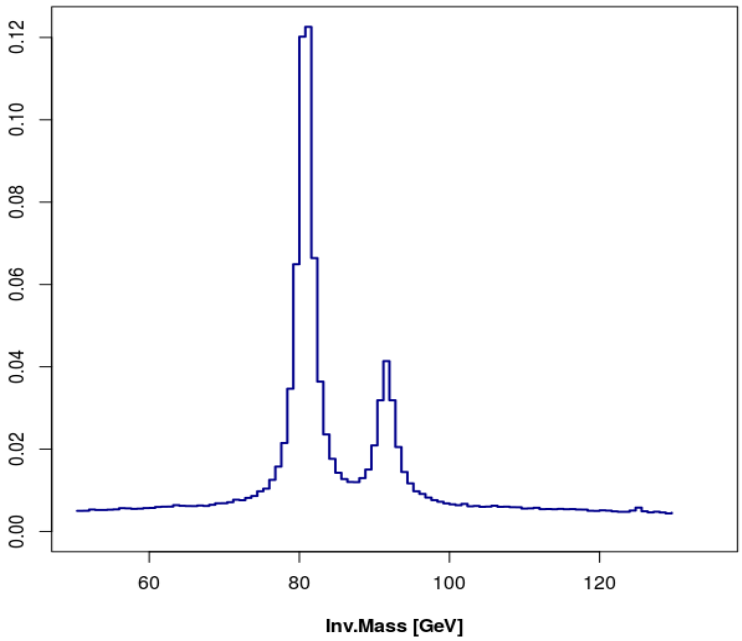

Figure A.7: Invariant Mass Distribution (QCD): Method 1

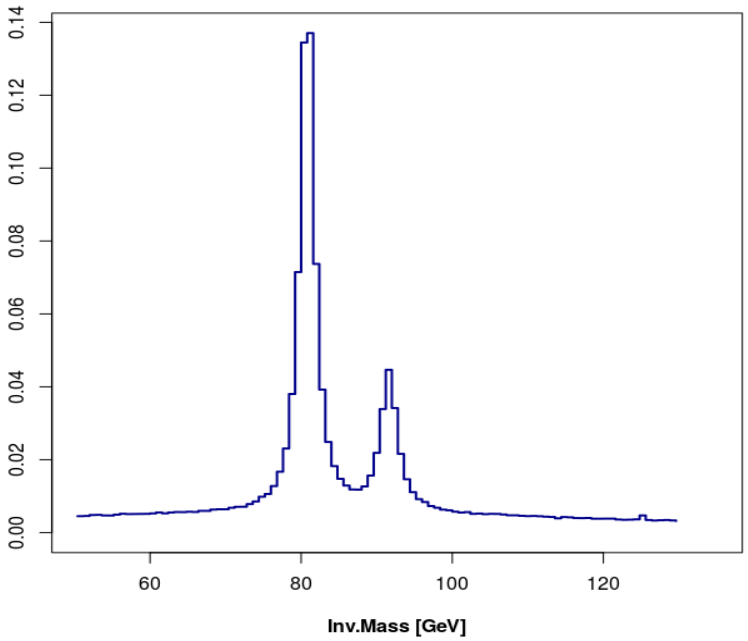

Figure A.8: Invariant Mass Distribution (QCD): Method 2

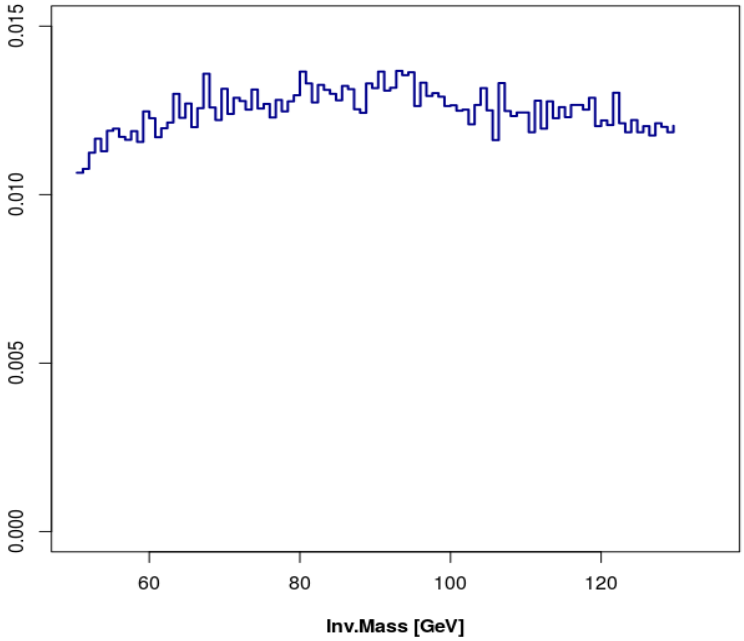

Figure A.9: Invariant Mass Distribution (QCD): Method 3

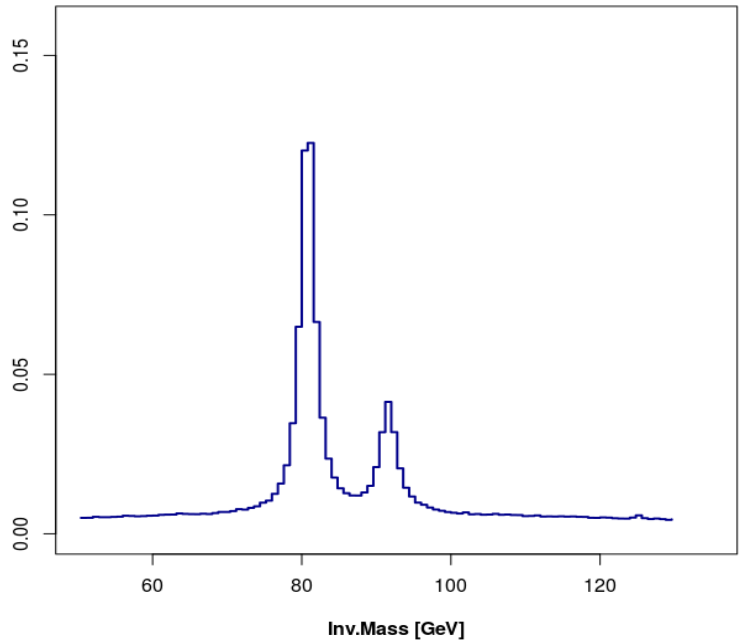

Figure A.10: Invariant Mass Distribution (QCD): Method 4

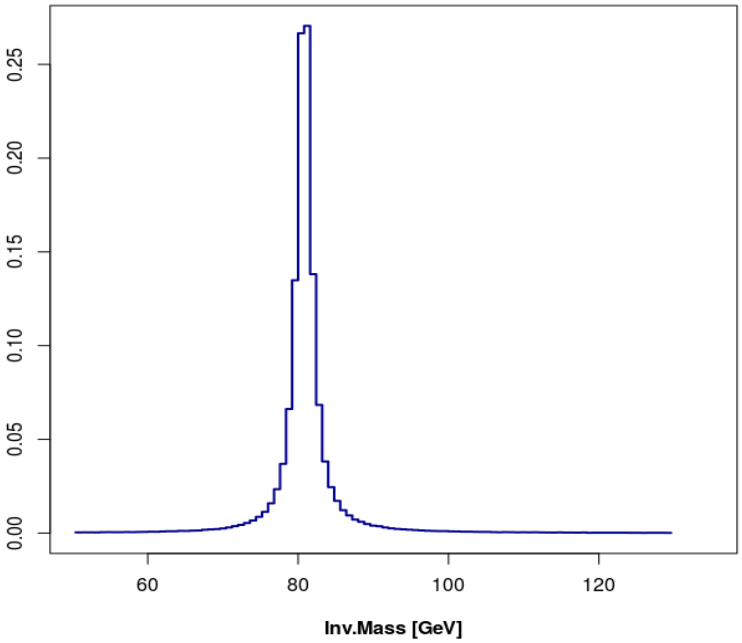

Figure A.11: Invariant Mass Muonic-Lepton couple (EWK + QCD)

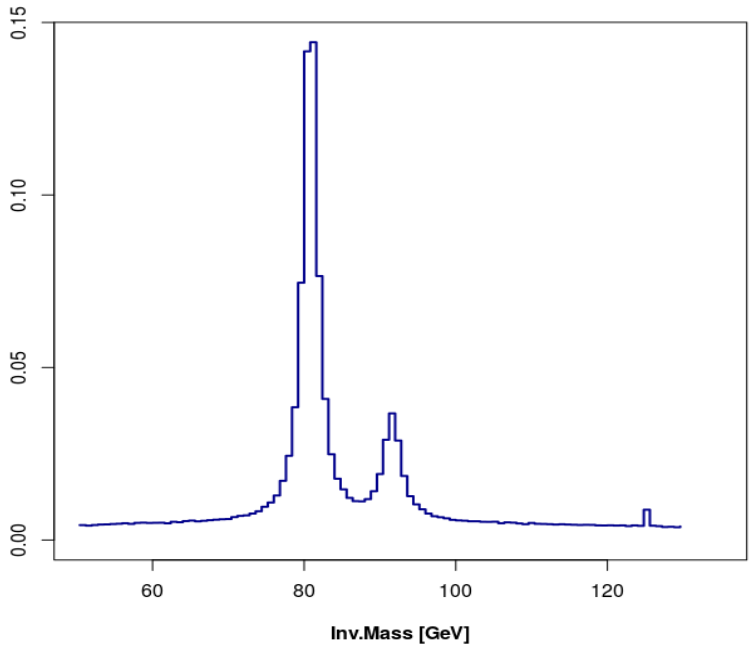

Figure A.12: Invariant Mass Distribution (EWK + QCD): Method 1

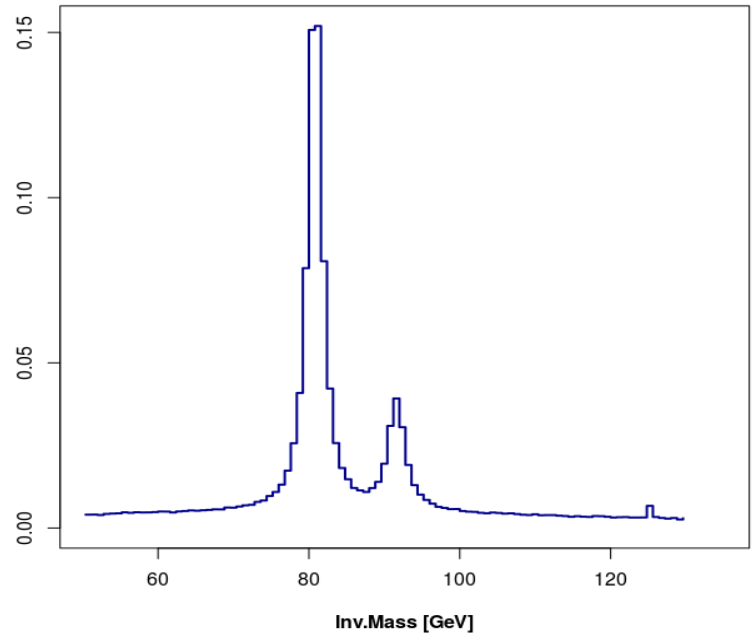

Figure A.13: Invariant Mass Distribution (EWK + QCD): Method 2

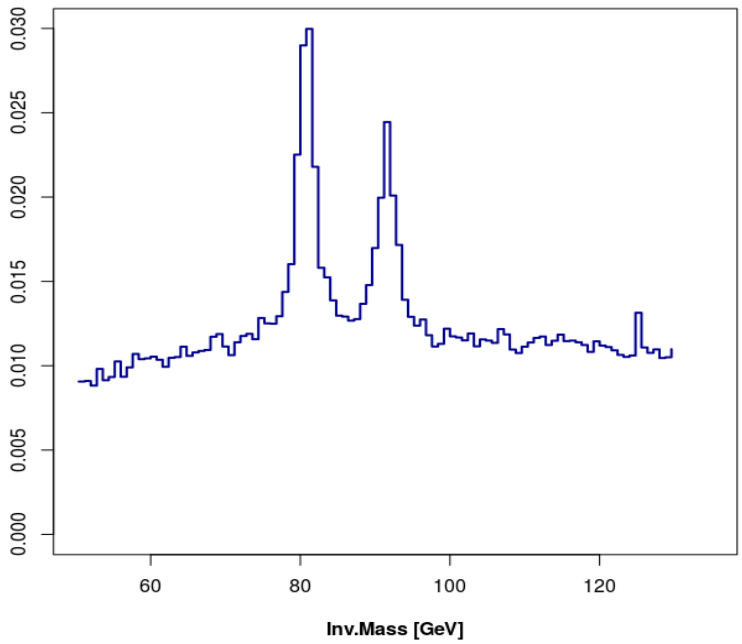

Figure A.14: Invariant Mass Distribution (EWK + QCD): Method 3

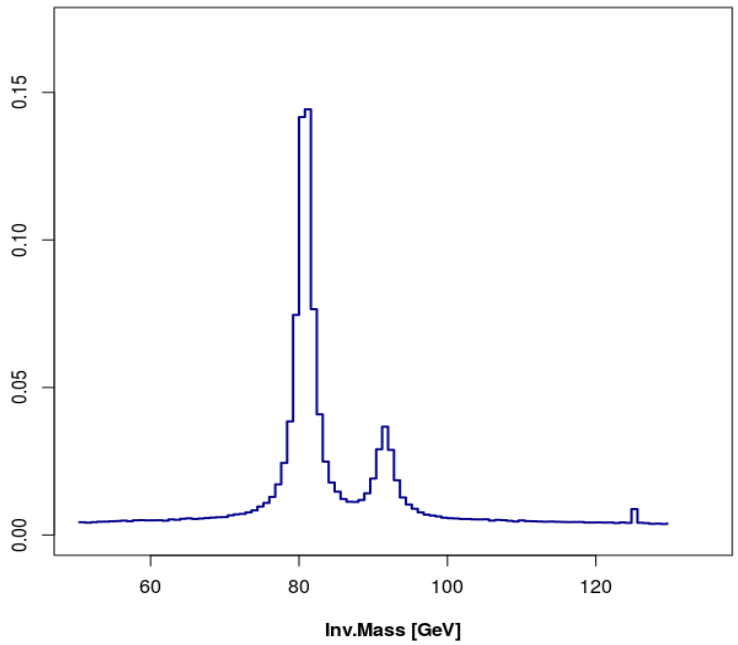

Figure A.15: Invariant Mass Distribution (EWK + QCD): Method 4

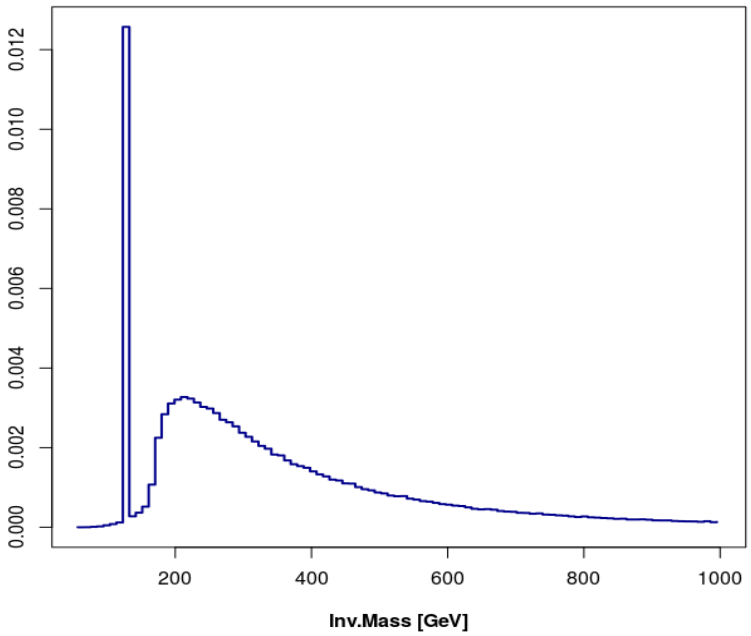

Figure A.16: Invariant Mass Distribution Total System (EWK) : Method 1

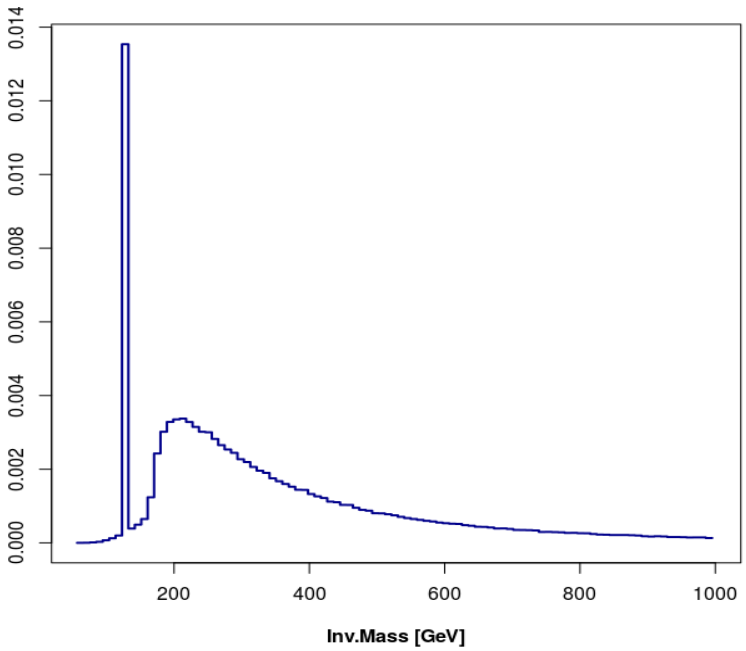

Figure A.17: Invariant Mass Distribution Total System (EWK) : Method 2

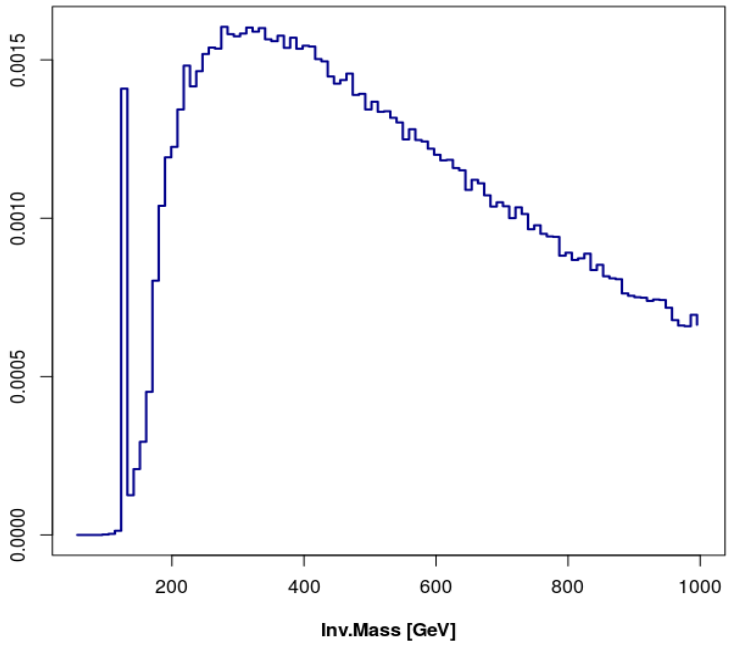

Figure A.18: Invariant Mass Distribution Total System (EWK) : Method 3

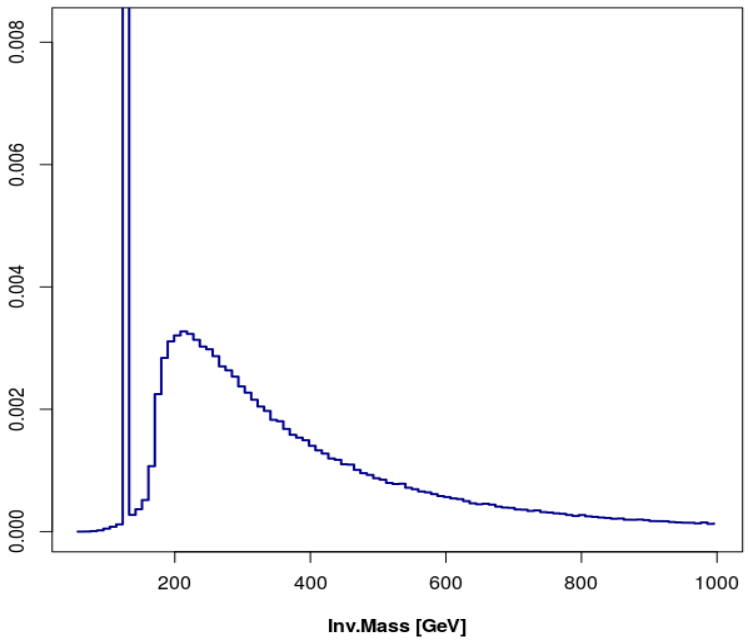

Figure A.19: Invariant Mass Distribution Total System (EWK) : Method 4

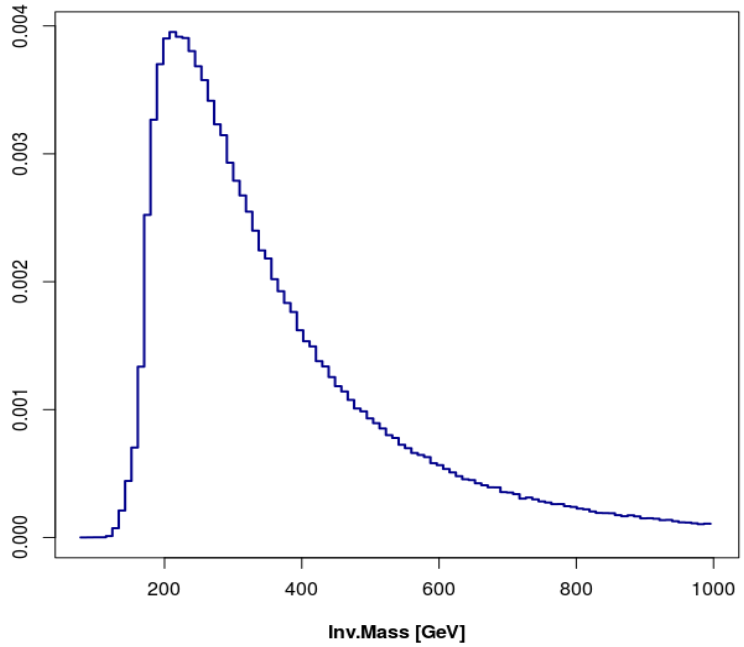

Figure A.20: Invariant Mass Distribution Total System (QCD) : Method 1

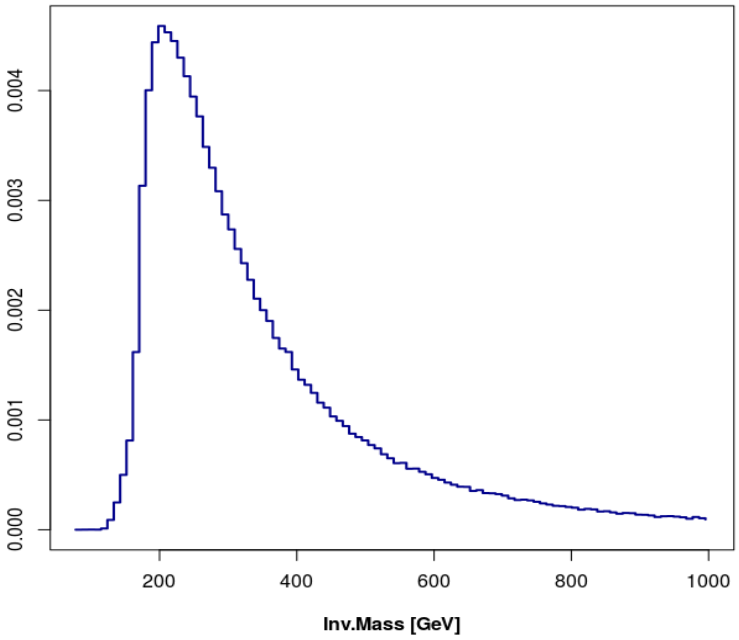

Figure A.21: Invariant Mass Distribution Total System (QCD) : Method 2

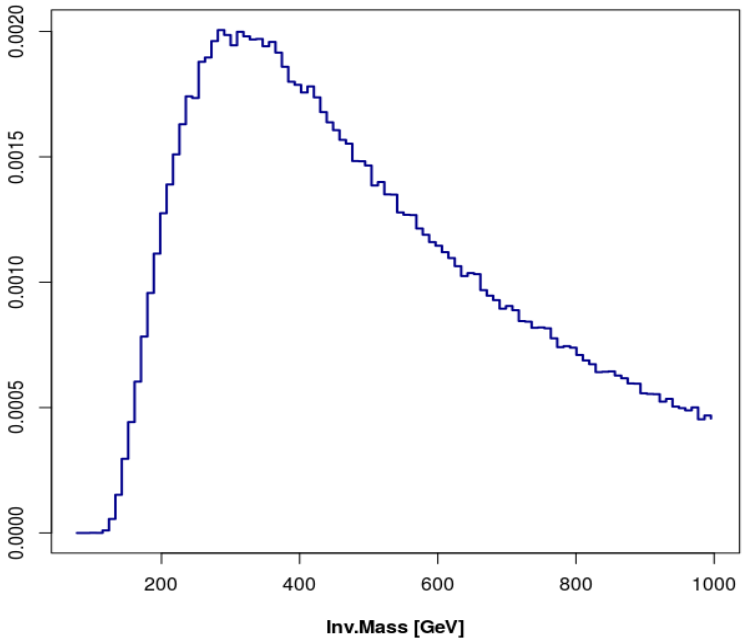

Figure A.22: Invariant Mass Distribution Total System (QCD) : Method 3

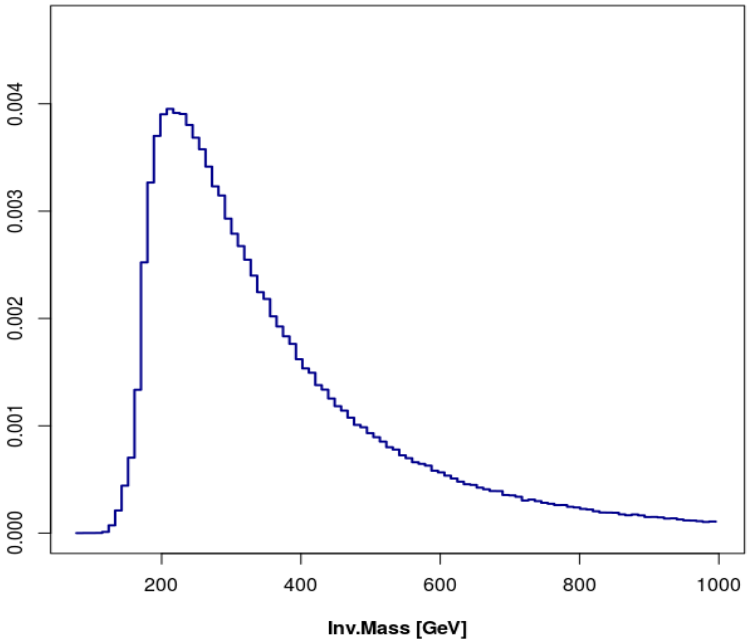

Figure A.23: Invariant Mass Distribution Total System (QCD) : Method 4

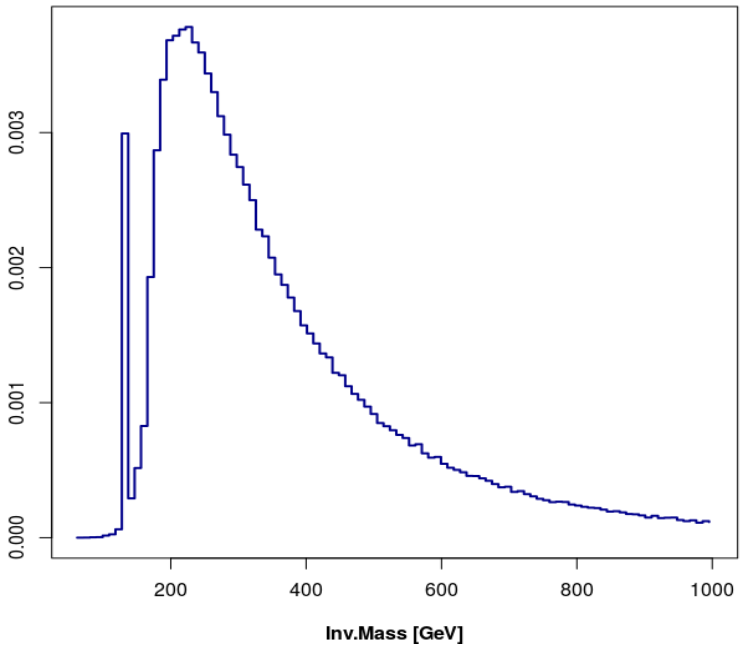

Figure A.24: Invariant Mass Distribution Total System (EWK + QCD) : Method 1

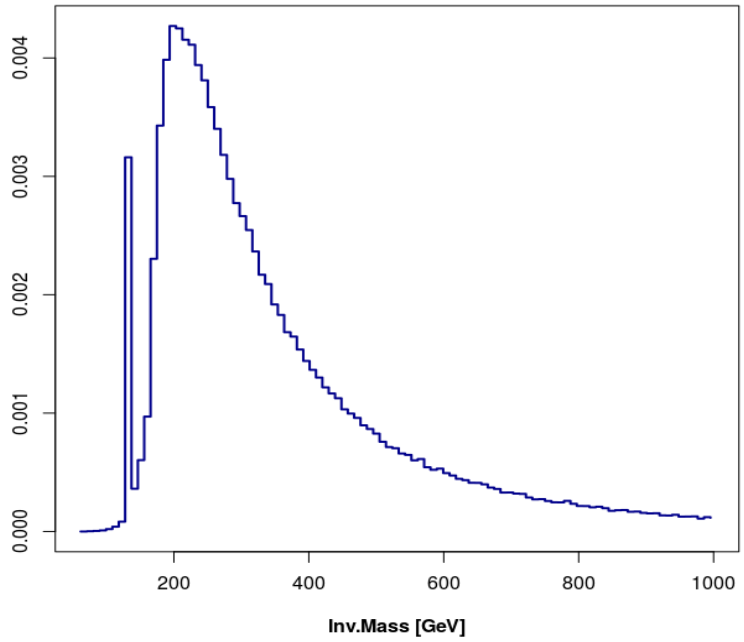

Figure A.25: Invariant Mass Distribution Total System (EWK + QCD) : Method 2

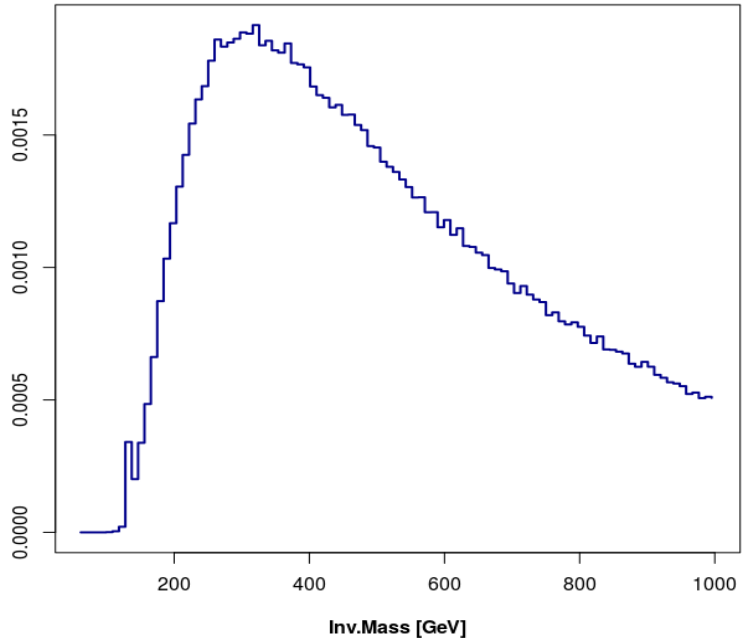

Figure A.26: Invariant Mass Distribution Total System (EWK + QCD) : Method 3

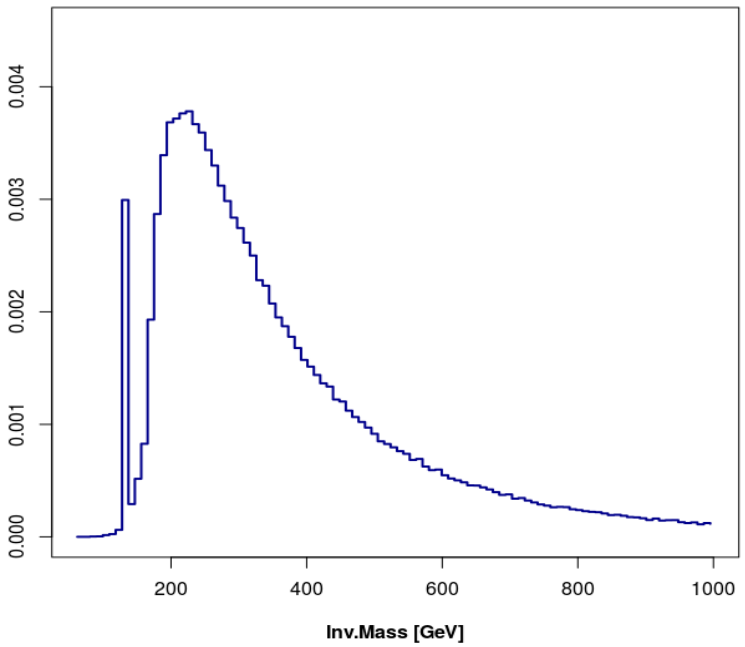

Figure A.27: Invariant Mass Distribution Total System (EWK + QCD) : Method 4

Appendix B

Codes

In Appendix B, I will show all the main functions needed to realized described in the thesis.
Firstly I will show the code used as parser, that uses a simple script made on Python[22] and then the main part of the code realized with R[19] (see Section 2.1.1).
The second script will list all the function used for the computation of the variables of interest used in the analysis (see Section 2.2.1).
The third script will include the functions in the third Chapter 3.
The forth script will introduced the analysis made on the second part of the elaborate (see Chapter 4).

```
fi = open('total.lhe', 'r')
fo1 = open('tenth1.txt','w')
fo2 = open('tenth2.txt','w')
fo3 = open('tenth3.txt','w')
fo4 = open('tenth4.txt','w')
fo5 = open('tenth5.txt','w')
fo6 = open('tenth6.txt','w')
fo7 = open('tenth7.txt','w')
fo8 = open('tenth8.txt','w')
fo9 = open('tenth9.txt','w')
fo10 = open('tenth10.txt','w')

it = 0;
for line in fi.readlines():
        if "<event>" in line:
                it +=1
                continue
        elif "</event>" in line:
                continue
        elif "#pdf" in line:
                continue
```

```
        elif "Les" in line:
23              continue
        elif "<weights>" in line:
25              continue
        elif it > 0 and it <=400000:
27              fo1.write(str(it) + line)
        elif it > 400000 and it <=800000:
29              fo2.write(str(it) + line)
        elif it > 800000 and it <=1200000:
31              fo3.write(str(it) + line)
        elif it > 1200000 and it <=1600000:
33              fo4.write(str(it) + line)
        elif it > 1600000 and it <=2000000:
35              fo5.write(str(it) + line)
        elif it > 2000000 and it <=2400000:
37              fo6.write(str(it) + line)
        elif it > 2400000 and it <=2800000:
39              fo7.write(str(it) + line)
        elif it > 2800000 and it <=3200000:
41              fo8.write(str(it) + line)
        elif it > 3200000 and it <=3600000:
43              fo9.write(str(it) + line)
        elif it > 3600000 and it <=4000000:
45              fo10.write(str(it) + line)
fi.close()
47 fo1.close()
fo2.close()
49 fo3.close()
fo4.close()
51 fo5.close()
fo6.close()
53 fo7.close()
fo8.close()
55 fo9.close()
fo10.close()
```

```
create_csv_file_complete <- function(path,name,save,total_,partition,part,big,
    small){
  if(.Platform$OS.type != "unix") stop("Linux required !")
  # set the working directory
  setwd(path)
  # check if the files needed for the operations are in the current directory
  if(!file.exists("dividetotal.py") & !file.exists("total.lhe") &!file.exists("
    dividetotal2.py"))
  {
    stop("Missing required files !")
  }
  if(missing(name)) {save = FALSE}
  if(missing(part)) {partition = FALSE}
  if((total_ == TRUE) & (partition == TRUE)){stop("Only one option is available in
     one execution")}
  if((small == TRUE) & (big == TRUE)){stop("Only one option is available in one
    execution")}
  if(missing(total_)){total_ = FALSE}
  if(missing(partition)){partition = FALSE}
  # create the partition of the *.lhe file
  print("Building .txt files")
  divide_total(big,small)
  # create and save the complete dataset
  if(total_ == TRUE)
  {
    data_file <- c("tenth1.txt","tenth2.txt","tenth3.txt","tenth4.txt","tenth5.txt
    ",
                   "tenth6.txt","tenth7.txt","tenth8.txt","tenth9.txt","tenth10.
    txt")
    dataset <- complete_data(data_file)
    if(save == TRUE){save(file = paste0(name,".RData"),dataset)}
  }

  if(partition == TRUE)
  {
    data_file <- paste0("tenth",part,".txt")
    dataset <- read_tenth(data_file)
    if(save == TRUE){save(file = paste0(name,".RData"),dataset)}
  }
  # eliminate the smallest file created
  options(warn = -1 )
  print("Deleting .txt files")
  file.remove("tenth1.txt","tenth2.txt","tenth3.txt","tenth4.txt","tenth5.txt",
              "tenth6.txt","tenth7.txt","tenth8.txt","tenth9.txt","tenth10.txt")
  options(warn = 1)
  return(dataset)
}
divide_total <- function(big,small){
  # the function calls the pyhton script used divide the .*lhe file into *.txt
    files
  # big : noHiggs event
  # small: polarized event

  if(big == TRUE){reticulate::py_run_file("dividetotal.py")}
  if(small == TRUE){reticulate::py_run_file("dividetotal2.py")}
```

```
}
read_tenth <- function(file){
  # the function reads the uncomplete *.txt file and extracts the data that will
    be used

  # read the file create with python (*.txt)
  df <- data.table::fread(file,fill=TRUE)
  # set the comlumns names
  colnames(df)<- c("EVENT","IDUP","ISTUP","MOTHUP1","MOTHUP2","ICOLUP1","ICOLUP2",
                   "Px","Py","Pz","E","M","VTIMUP","SPINUP")
  # remove the lines with NA
  df <- df[complete.cases(df),]
  df <- as.data.frame(df)
  # set the right order of the columns
  df <- df[,c(2,3,4,5,6,7,8,9,10,11,12,13,14,1)]
  return(df)

}
complete_data <- function(file){
  # the function will take all the partial files create with python an will return
     the complete dataset
  data_list <- list(NA)
  data_list <- lapply(file,read_tenth)
  dataset <- do.call(rbind,data_list)
  return(as.data.frame(dataset))
}
help_parser_complete <- function(){
  print("Create_csv_file_complete <- function(path,name,save,total_,partition,part
    ,big,small)")
  print("The function will create a readable dataset from the *lhe files")
  print("In order to use this function you must have three files in the current
    directory : ")
  print("1. dividetotal.py (it splits the main *lhe files in 10 parts)")
  print("2. dividetotal2.py (it splits the main *lhe files in 5 parts (to be used
    for polarized dataset))")
  print("3. total.lhe")
  print("--------------------Input Variables----------------------------------")
  print("Input variables : ")
  print("path : working directory to be set")
  print("name : name choose for the file(if save enabled)")
  print("save : enables save option")
  print("total_: enables the creation of the complete dataset")
  print("partition : enables the creation of 1 of the partition created with
    python ")
  print("part : name of the part to be saved")
  print("big : used only for the biggest dataset")
  print("small : for smaller datasets, polarized ones")
  print("--------------------Output Variables---------------------------------")
  print("dataset: the dataset converted from the *.lhe file")
  print("name.RData: the dataset will be saved (if save == TRUE)")
  print("--------------------Packages needed----------------------------------")
  print("1. reticulate (python.load())")
  print("2. data.table (fread())")
}
```

```
normP<- function(df){
  return(sqrt(df$Px^2+df$Py^2+df$Pz^2))
}

invmass_event <- function(df)
{
  # compute invariant mass distribution for each event
  mass <- function(.) return(sqrt((.$E)^2-(.$Px)^2-(.$Py)^2-(.$Pz)^2))
  vec <- df %>% select(E, Px, Py, Pz, EVENT) %>%
    group_by(EVENT) %>%
    summarise_all(funs(sum)) %>%
    mass

  return(vec)
}

invmass_particle <- function(df,ID){
  # Compute invariant mass distribution given a set of particles
  mass <- function(.) return(sqrt((.$E)^2-(.$Px)^2-(.$Py)^2-(.$Pz)^2))

  if(length(ID) >= 5){stop("Too much particles requested")}

  if(length(ID) == 2)
  {
    vec <- df %>% select(E, Px, Py, Pz, EVENT) %>%
      filter(EVENT == ID[1] | EVENT == ID[2]) %>%
      group_by(EVENT) %>%
      summarise_all(funs(sum)) %>%
      mass

    return(vec)
  }
  if(length(ID) == 3)
  {
    vec <- df %>% select(E, Px, Py, Pz, EVENT) %>%
      filter(EVENT == ID[1] | EVENT == ID[2] | EVENT == ID[3]) %>%
      group_by(EVENT) %>%
      summarise_all(funs(sum)) %>%
      mass

    return(vec)
  }
  if(length(ID) == 4)
  {
    vec <- df %>% select(E, Px, Py, Pz, EVENT) %>%
      filter(EVENT == ID[1] | EVENT == ID[2] | EVENT == ID[3] | EVENT == ID[4])
    %>%
      group_by(EVENT) %>%
      summarise_all(funs(sum)) %>%
      mass

    return(vec)
  }
}

```

```
invmass_limit <- function(df,limit = 80)
{
  # used to compute the invariant mass distribution closer to the set limit
  # limit is used to be set at 80 (invariant mass of the W boson)
  mass <- function(.) return(sqrt((.$E)^2-(.$Px)^2-(.$Py)^2-(.$Pz)^2))
  vec <- df %>% select(E, Px, Py, Pz, COUPLE) %>%
    group_by(COUPLE) %>%
    summarise_all(funs(sum)) %>%
    mass
  return((abs(vec-limit)))
}

eta <- function(df){
  # Compute Pseudorapidity

  if(is.na(df)){df <- df %>% complete.cases}
  eta <- atanh(df$Pz/normP(df))

  # Replacing NaN with 1 (Hp: limPz-->0 Pz/normP=1)

  eta[which(etavar=="NaN")]<-Inf

  return(eta)
}

trans_mom <- function(df)
{
  # Compute transverse momentum

  mom <- function(.) return(sqrt((.$Px)^2+(.$Py)^2))
  vec <- df %>% select(IDUP,ISTUP,E,Px,Py,Pz,EVENT) %>%
    group_by(EVENT) %>%
    summarise_all(funs(sum)) %>%
    mom

  return(vec)
}

theta <- function(df,eta){
  #Compute Polar angle
  return(2*atan(exp(-eta))*sign(df$Px))
}


phi <- function(df){
  #compute azimuthal  angle
  return(atan(df$Px/df$Py))
}

speed<-function(df){
  # Compute the velocity (beta)
  return(normP(df)/df$E)
}

gamma <- function(vel){
  # Compute the relativistic factor (gamma)
```

```
  vel[vel>=1] <- 0.99999999999
  return(1/sqrt(1-vel^2))
}

rest_boson <- function(df,ID)
{
  # Change frame from Laboratory Rest Frame to Boson Rest Frame
  if(ID == "plus")
  {
    id <- c(-13,14)
    print("Computing Boson Rest Frame for W+")
  }
  if(ID == "minus")
  {
    id <- c(11,-12)
    print("Computing Boson Rest Frame for W-")
  }
  if(missing(ID)){stop("ID is missing. You must choose on which Boson you want to
    perform the analysis !")}

  boson <- df %>% select(IDUP,E, Px, Py, Pz, EVENT) %>%
    filter(IDUP == id[1] | IDUP == id[2]) %>%
    group_by(EVENT) %>%
    summarise_all(funs(sum)) %>%
    as.data.frame %>%
    select(Px,Py,Pz,E)

  temp<-matrix(NA,nrow=dim(df)[1],ncol=4)
  windex<-unique(df$EVENT) ## Let's assume that there will be always id[1] or id
    [2]
  len<-length(windex)

  phiW_<-phi(boson)
  thetaW_<-theta(boson)
  velW_<-speed(boson)
  gammaW_<-gamma(velW_)

  opb <- pboptions(style = 1, char = ">")
  on.exit(closepb(pb))
  pb <- startpb(min = 0, max = len)

  for(i in 1:len){
    setpb(pb,i)
    ind <- which(df$EVENT==windex[i])
    cos_phi <- cos(-phiW_[i])
    sin_phi <- sin(-phiW_[i])
    cos_theta <- cos(-thetaW_[i])
    sin_theta <- sin(-thetaW_[i])

    S <- matrix(c(gammaW_[i], 0, 0, -velW_[i]*gammaW_[i],
                  velW_[i]*gammaW_[i]*sin_theta*cos_phi, cos_phi*cos_theta, sin_phi,
    gammaW_[i]*(-sin_theta*cos_phi),
                  -velW_[i]*gammaW_[i]*sin_theta*sin_phi, -sin_phi*cos_theta,cos_phi,
     gammaW_[i]*sin_theta*sin_phi,
                  -velW_[i]*gammaW_[i]*cos_theta, sin_theta, 0, gammaW_[i]*cos_theta)
    ,
```

```
                nrow=4,ncol=4,byrow = TRUE)
    temp[ind,]<-as.matrix(df[ind,c("E","Px","Py","Pz")])%*%S
  }

  invisible(NULL)

  temp<-as.data.frame(temp)
  colnames(temp)<-c("E","Px","Py","Pz")
  temp<-na.omit(temp)
  temp<-round(temp,6)

  t1 <- df[,1:6]
  t2 <- df[,11:14]
  final <- cbind(t1,temp$Px,temp$Py,temp$Pz,temp$E,t2)
  colnames(final)<-c("IDUP","ISTUP","MOTHUP1","MOTHUP2","ICOLUP1",
                     "ICOLUP2","Px","Py","Pz","E",
                     "M","VTIMUP","SPINUP","EVENT")


  return(final)
}
```

```
pl_method <- function(dataset, method_name)
{
  if(.Platform$OS.type != "unix")stop("Linux required!")
  if(method != c(1,2,3,4)){stop("Error: only four method are available")}
  df <- dataset %>% select(IDUP,ISTUP,E, Px, Py, Pz,EVENT) %>% filter(ISTUP == 1,
    abs(IDUP)<13) # only jet, not leptons
  data_list <- list(NA)
  data_list <- pbapply::pblapply((unique(method_name)), function(x) methods(df,x))
  return(data_list)

}
methods <- function(df,method_name)
{
  if(missing(method_name)) method_name <- c("method1","method2","method3","method4
    ")
  no_cores <- parallel::detectCores() - 1
  cl <- parallel::makeCluster(no_cores)
  parallel::clusterExport(cl,
                varlist = c("df","method_name",
                            "method_one","method_two","method_three","method_four"
    ))
  result <- switch(
    method_name,
    "method1"  = pbapply::pblapply(cl = cl,X = unique(df$EVENTS), function(x)
    method_one(df,x)),
    "method2"  = pbapply::pblapply(cl = cl,X = unique(df$EVENTS), function(x)
    method_two(df,x)),
    "method3"  = pbapply::pblapply(cl = cl,X = unique(df$EVENTS), function(x)
    method_three(df,x)),
    "method4"  = pbapply::pblapply(cl = cl,X = unique(df$EVENTS), function(x)
    method_four(df,x))
  )
  parallel::stopCluster(cl)
}
method_one <- function(df,i)
{
  dat <- df %>% filter(EVENT == i)
  check <- t(combn(dat$IDUP,2))
  delta <- combn(dat$ETA,2,diff)
  max_index <- which.max(abs(delta))
  to_save <- dat[which(dat$IDUP == check[max_index,1] | dat$IDUP == check[max_
    index,2]),]
  return(to_save)
}
method_two <- function(df,i)
{
  dat <- find_event(df,i)
  check <- t(combn(dat$IDUP,2))
  range <- 1:nrow(check)
  lll <- lapply(range, function(x) link(dat,check,x))
  ll <- do.call(rbind,lll) %>% na.omit(ll)
  l <- add_couple(ll)
  max_index <- invmass_event(l) %>% which.max
  to_save <- dat[which(dat$IDUP == check[max_index,1] | dat$IDUP == check[max_
    index,2]),]
```

```
  return(to_save)
}
method_three <- function(df,i)
{
  dat <- find_event(df,i)
  check <- t(combn(dat$IDUP,2))
  range <- 1:nrow(check)
  lll <- lapply(range, function(x) link(dat,check,x))
  ll <- do.call(rbind,lll) %>% na.omit(ll)
  l <- add_couple(ll)
  max_index <- trans_mom(l) %>% which.max
  to_save <- dat[which(dat$IDUP == check[max_index,1] | dat$IDUP == check[max_
    index,2]),]
  return(to_save)
}
method_four <- function(df,i)
{
  dat <- find_event(df,i)
  check <- t(combn(dat$IDUP,2))
  range <- 1:nrow(check)
  lll <- lapply(range, function(x) link(dat,check,x))
  ll <- do.call(rbind,lll) %>% na.omit(ll)
  l <- add_couple(ll)
  index <- invmass_limit(df = l, limit = 80) %>% which.min
  to_save <- dat[which(dat$IDUP == check[index,1] | dat$IDUP == check[index,2]),]
  return(to_save)
}
link <- function(df,check,i)
{
  # support function
  ll <- list(NA)
  ll[[i]] <- df %>% filter(df$IDUP == check[i,1] | df$IDUP == check[i,2])
  ll <- do.call(rbind,ll)
  return(ll)
}
add_couple <- function(df)
{
  # support funcion
  df <- df[1:12,]
  i <- rep(1:6,each = 2 , length.out = 12)
  df <- df %>% mutate(COUPLE = i)
  return(df)
}
mixup <- function(df1,df2)
{
  # df1: complete dataset to be re-sized
  # df2: dataset related with a particular method

  df1 <- df1 %>% filter(IDUP == 13 | IDUP == -14 | IDUP == -13 | IDUP == 14, ISTUP
     == 1) %>%
    select(IDUP, ISTUP, E, Px, Py, Pz, EVENT)
  inv <- NA
  inv <- rbind(df1,df2)  %>%
    invmass_event

  return(inv)
```

```
}
plot_dist <- function(item1,item2,item3,item4,item5,breaks=100)
{
  # item1 : lepmass
  # item2 : mass1 (method1)
  # item3 : mass2 (method2)
  # item4 : mass3 (method3)
  # item5 : mass4 (method4)

  item <- c(item1,item2,item3,item4,item5)

  a <- floor(min(item))
  b <- ceiling(max(item))

  if(a==b){
    pass = 0
    by = 1
  } else{
    pass <- round((b - a)/breaks,3)
    by <- seq(a,b,pass)
  }

  lep <- hist(item1,breaks=by, plot=FALSE, right=FALSE)
  m1  <- hist(item2,breaks=by, plot=FALSE, right=FALSE)
  m2  <- hist(item3,breaks=by, plot=FALSE, right=FALSE)
  m3  <- hist(item4,breaks=by, plot=FALSE, right=FALSE)
  m4  <- hist(item5,breaks=by, plot=FALSE, right=FALSE)
  ascisse <- lep$mids
  if(pass==0){ascisse<- 0}

  mylist<- list("lep" = lep, "m1" = m1,"m2" = m2,"m3" = m3, "m4" = m4,  "ascisse"
    = ascisse)

  lep_dens <- mylist$lep$density
  m1_dens <- mylist$m1$density
  m2_dens <- mylist$m2$density
  m3_dens <- mylist$m3$density
  m4_dens <- mylist$m4$density

  final <- list("lep_dens" = lep_dens, "m1_dens" = m1_dens, "m2_dens" = m2_dens,
                "m3_dens" = m3_dens, "m4_dens" = m4_dens, "ascisse" = mylist$
    ascisse)
  return(final)

}
mse <- function(x,x_){
  n <- 1/length(x)
  mse <- n*sum(((x-x_)^2))
  return(mse)
} # mean squared error
zoom <- function(item){
  item <- item[which(item > 50 & item < 130)]
  return(item)
}
compare_dist <- function(dat1,dat2,dimsample = 5000, iter = 1000,
                         test_name = c("EMD","Hellinger", "Kolmogorov-Smirnov","
```

```
    Total_Variation"))
{
  # It takes two distribution and compare it with the four choosen distances

  if(.Platform$OS.type != "unix") stop("Linux required !")
  if(missing(test_name)) test_name <- c("EMD","Hellinger", "Kolmogorov-Smirnov","
    Total_Variation")

  set.seed(22061993)

  df <- make_data(dat1,dat2)

  no_cores <- parallel::detectCores() - 1
  cl <- parallel::makeCluster(no_cores)
  parallel::clusterExport(cl,varlist = c("df","bstrap_dist","eval_dist"))
  dist_df <- pbapply::pbsapply(cl,unique(1:iter), function(x) bstrap_dist(x,
    dimsample))
  parallel::stopCluster(cl)
  ddist <- apply(pv_df,1,mean)
  distances <- data.frame("distance" = pv)
  rownames(distances) <- test_name
  return(t(distances))
}

make_data <- function(dat1,dat2)
{
  d1 <- dat1 %>% filter(EVENT %in% index) %>% invmass_event %>% zoom
  d2 <- dat2 %>% filter(EVENT %in% index) %>% invmass_event %>% zoom

  df <- data.frame(x = d1, y = d2)
  return(df)
}
bstrap_dist <- function(i,dimsample)
{
  index <- sort(base::sample(1:dimsample,dimsample,replace = TRUE))
  dat <- df[index,]
  pv_vec <- sapply(test_name, function(x) eval_dist(x,dat))
  pv_df <- do.call(rbind,list(pv_vec))

  return(pv_df)
}
eval_dist <- function(test_name, df) {
  result <- switch(
    test_name,
    "EMD"                = earthmovdist::emdL1(df[,1],df[,2]),
    "Hellinger"          = textmineR::CalcHellingerDist(df[,1],df[,2]),
    "Kolmogorov-Smirnov" = stats::ks.test(df[,1],df[,2], exact = FALSE)$p.value,
    "Total Variation"    = total_var_dist(df[,1],df[,2])
  )
}
plot_splines <- function(to_plot,NT){
  # plots the splines, first derivative, second derivative
  x11()
  par (mfrow=c(3,1),mar=c(6,5,2,1),mex=0.6, mgp=c(2.2,0.7,0),pty="m",
       font.main=1,font.lab=1, font.axis=1,cex.lab=1.3,cex.axis=1)
  layout(matrix(c(1,1,2,3), 2, 2, byrow = F))
```

```
208   plot(to_plot[[1]],to_plot[[2]],xlab="Inv.Mass [GeV]",ylab="observed data")
      points(to_plot[[1]],to_plot[[3]] ,type="l",col="blue",lwd=2)
210   plot(abscissa[2:(NT-1)],to_plot[[6]],xlab="Inv.Mass [GeV]",ylab="first
        differences x",type="l")
      points(abscissa,to_plot[[4]] ,type="l",col="blue",lwd=2)
212   plot(abscissa[2:(NT-1)],to_plot[[7]],xlab="Inv.Mass [GeV]",ylab="second
        differences x",type="l")
      points(abscissa,to_plot[[5]] ,type="l",col="blue",lwd=2)
214 }
    specify_decimal <- function(x, k) {
216   # helps to scecify the number of decimals in the results
      trimws(format(round(x, k), nsmall=k))
218 }
    all_spline <- function(abscissa,y,minknot,maxknot,knotnumcrit,degree,to_plot,GoF)
220 {
      # creates the free knot spline, elaborates the plots and analyzes the goodness
        of fit(optional)
222   opt_knot <- fit.search.numknots(abscissa,y,minknot = minknot,maxknot = maxknot,
        knotnumcrit  = knotnumcrit,degree = degree)

224   Xobs0 <- y
      NT <- length(abscissa)
226   m <- opt_knot@degree+1
      degree <- opt_knot@degree
228
      breaks <- c(min(abscissa),opt_knot@optknot,max(abscissa))
230   basis <- create.bspline.basis(breaks, norder=m)
      functionalPar <- fdPar(fdobj=basis, Lfdobj=m-2, lambda=0)
232
      rappincX1 <- (Xobs0[3:NT]-Xobs0[1:(NT-2)])/(abscissa[3:NT]-abscissa[1:(NT-2)])
234   rappincX2 <- ((Xobs0[3:NT]-Xobs0[2:(NT-1)])/
                      (abscissa[3:NT]-abscissa[2:(NT-1)])-(Xobs0[2:(NT-1)]-Xobs0[1:(NT
        -2)])/(abscissa[2:(NT-1)]-abscissa[1:(NT-2)]))*2/(abscissa[3:(NT)]-abscissa
        [1:(NT-2)])
236

238   Xss <- smooth.basis(abscissa, Xobs0, functionalPar)

240   Xss0 <- eval.fd(abscissa, Xss$fd, Lfd=0)
      Xss1 <- eval.fd(abscissa, Xss$fd, Lfd=1)
242   Xss2 <- eval.fd(abscissa, Xss$fd, Lfd=2)


244
      if(to_plot == TRUE)
246   {
        toplot <- list(opt_knot@x , opt_knot@y , Xss0 , Xss1, Xss2, rappincX1,
        rappincX2,opt_knot@optknot)
248     plot_splines(to_plot = toplot,NT = NT)
      }
250
      if(Gof == TRUE)
252   {
        MSE <- mse(yr,Xss0)
254     Goodness <- c(MSE,opt_knot@GCV, specify_decimal(length(opt_knot@optknot),0))
        names(Goodness) <- c("MSE","GCV", "#Knot")
256     print(Goodness)
```

```
  }

  output <-list("fit" = Xss0,"der1" = Xss1,"der2" = Xss2,
                "MSE" = MSE,"GCV" = opt_knot@GCV, "#Knot" = specify_decimal(length
    (opt_knot@optknot),0))

}
```

```r
indep_p_value <- function(df,iter = 10000, dimsample = 1000,
                          test_name = c("Pearson","Kendall", "Spearman","Hoeffding"))
{
  if(.Platform$OS.type != "unix") stop("Linux required !")
  if (any(test_name != c("Pearson","Kendall", "Spearman","Hoeffding")))
    stop("Error: only these four test are can be used !")
  if(missing(test_name)) test_name <- c("Pearson","Kendall", "Spearman","Hoeffding")

  no_cores <- parallel::detectCores() - 1
  cl <- parallel::makeCluster(no_cores)
  parallel::clusterExport(cl,varlist = c("df","bstrap","eval_pvalue"))
  pv_df <- pbapply::pbsapply(cl,unique(1:iter), function(x) bstrap(x,dimsample))
  parallel::stopCluster(cl)
  pv <- apply(pv_df,1,mean)
  p_value <- data.frame("p value" = pv)
  rownames(p_value) <- test_name
  return(t(p_value))
}
bstrap <- function(i,dimsample)
{
  index <- sort(base::sample(1:dimsample,dimsample,replace = TRUE))
  dat <- df[index,]
  pv_vec <- sapply(test_name, function(x) eval_pvalue(x,dat))
  pv_df <- do.call(rbind,list(pv_vec))

  return(pv_df)
}
eval_pvalue <- function(test_name, df) {
  result <- switch(
    test_name,
    "Pearson"    = stats::cor.test(x = df[,1], y = df[,2], method = "pearson",
    exact = NULL)$p.value,
    "Kendall"    = Kendall::Kendall(df[,1],df[,2])$sl[1],
    "Spearman"   = Hmisc::spearman.test(x = df[,1],y = df[,2])[5],
    "Hoeffding"  = Hmisc::hoeffd(df[,1],df[,2])$P[1,2]
  )
}
boot_Select_Cop <- function(df,seed,iter,dimsample)
{
  if(missing(seed)) seed <- (220061993)
  if(missing(dimsample)) dimsample <- 100000
  if(missing(iter)) iter <- 2000
  set.seed(seed)
  no_cores <- parallel::detectCores()-1
  cl <- parallel::makeCluster(no_cores)
  parallel::clusterExport(cl = cl,varlist = c("cycle","df","BiCopSelect"))
  fam <- pbapply::pblapply(cl = cl,X = unique(1:iter), function(x) cycle(x,
    dimsample))
  parallel::stopCluster(cl)
  dat <- do.call(rbind,fam)
  final <- exit_result(dat)
  return(final)
}
```

```
exit_result <- function(df)
{
  return(list("family" = getmode(df[,1]), "par" = mean(df[,2]), "tau" = mean(df
    [,3]),
                "se" = mean(df[,4]), "p value" = mean(df[,5]), "all_family" = df
    [,1]))
}

cycle <- function(i,dimsample)
{
  index <- sort(sample(1:dim(df)[1], size = dimsample, replace = TRUE))
  selectedCopula <- BiCopSelect(df$u1[index], df$u2[index],indeptest = TRUE,se =
    TRUE, familyset = NA, presel = TRUE)

  family <- selectedCopula$family
  param <- selectedCopula$par
  param2 <- selectedCopula$par2
  tau <- selectedCopula$tau
  se <- selectedCopula$se
  tests <- selectedCopula$p.value.indeptest

  return(data.frame("family" = (family), "par" = (param), "par2" = param2,
                    "tau" = (tau), "se" = (se), "p value" = (tests)))
}
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
```

Bibliography

[1] Gallotti Fabio Giso Nicoló. "Statistical Analysis for the Study of High Energy Particles Decays". MA thesis. Politecnico di Milano, 2017.

[2] J. Alwall, A. Ballestrero, P. Bartalini, S. Belov, E. Boos, A. Buckley, J. M. Butterworth, L. Dudko, S. Frixione, L. Garren, S. Gieseke, A. Gusev, I. Hinchliffe, J. Huston, B. Kersevan, F. Krauss, N. Lavesson, L. Lönnblad, E. Maina, F. Maltoni, M. L. Mangano, F. Moortgat, S. Mrenna, C. G. Papadopoulos, R. Pittau, P. Richardson, M. H. Seymour, A. Sherstnev, T. Sjöstrand, P. Skands, S. R. Slabospitsky, Z. Waş, B. R. Webber, M. Worek, and D. Zeppenfeld. "A standard format for Les Houches Event Files". In: *Computer Physics Communications* 176 (Feb. 2007), pp. 300–304. eprint: `hep-ph/0609017`.

[3] A. Ballestrero, A. Belhouari, G. Bevilacqua, V. Kashkan, and E. Maina. "PHANTOM: A Monte Carlo event generator for six parton final states at high energy colliders". In: *Computer Physics Communications* 180 (Mar. 2009), pp. 401–417. arXiv: `0801.3359 [hep-ph]`.

[4] *CERN*. `http://atlas.physicsmasterclasses.org/en/wpath_lhcphysics2.htm`. Accessed: 2018-01-7.

[5] Scott Cohen. "Finding Color AND Shape Patterns in Images". PhD thesis. Standford University, 1999.

[6] Henry Deng and Hadley Wickham. "Density estimation in R". In: 2014.

[7] Van Than Dung and Tegoeh Tjahjowidodo. "A direct method to solve optimal knots of B-spline curves: An application for non-uniform B-spline curves fitting". In: *PLOS ONE* 12.3 (Mar. 2017), pp. 1–24. URL: `https://doi.org/10.1371/journal.pone.0173857`.

[8] R. Frederix and F. Maltoni. "Top pair invariant mass distribution: a window on new physics". In: *Journal of High Energy Physics* 1 (Jan. 2009), pp. 0–47. arXiv: 0712.2355 [hep-ph].

[9] T. Han, D. Krohn, L.-T. Wang, and W. Zhu. "New physics signals in longitudinal gauge boson scattering at the LHC". In: *Journal of High Energy Physics* 3, 82 (Mar. 2010), p. 82. arXiv: 0911.3656 [hep-ph].

[10] Wassily Hoeffding. "A Non-Parametric Test of Independence". In: *Ann. Math. Statist.* 19.4 (Dec. 1948), pp. 546–557.

[11] Marius Hofert, Ivan Kojadinovic, Martin Maechler, and Jun Yan. *copula: Multivariate Dependence with Copulas*. R package version 0.999-18. 2017. URL: https://CRAN.R-project.org/package=copula.

[12] J. Ramsay James and B.W. Silverman. *Functional Data Analysis*. edition 2. Spinger-Verlag New York, 2005.

[13] Rainer M Krug and Dirk Eddelbuettel. *earthmovdist: Wrapper to the Emd-L1 library by Haibin Ling and Kazunori Okada*. R package version 0.1.2/r32. 2012. URL: https://R-Forge.R-project.org/projects/earthmovdist/.

[14] Guangyu Mao. "Testing independence in high dimensions using Kendall's tau". In: *Computational Statistics Data Analysis* 117 (2018), pp. 128 –137.

[15] Satoshi Miyata and Xiaotong Shen. "Free-Knot Splines and Adaptive Kknot Selection". In: *J. Japan Statist. Soc* 35.2 (Mar. 2017), pp. 303–324.

[16] T. Nagler. "kdecopula: An R Package for the Kernel Estimation of Bivariate Copula Densities". In: *ArXiv e-prints* (Mar. 2016). arXiv: 1603.04229 [stat.CO].

[17] Abd Alrahem Shafeq Marie Mohammed H. Baker Al-Haj Ebrahem Omar M. Eidous Mohammad. "A Comparative Study for Bandwidth Selection in Kernel Density Estimation". In: *Journal of Modern Applied Statistical Methods* 9 (2010), pp. 263–273.

[18] *Polarization of the W-Pair System*. http://www.hep.ucl.ac.uk/~jpc/all/ulthesis/node44.html. Accessed: 2018-05-11.

[19] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2016. URL: https://www.R-project.org/.

[20] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2016. URL: https://www.R-project.org/.

[21] J. O. Ramsay, Hadley Wickham, Spencer Graves, and Giles Hooker. *fda: Functional Data Analysis*. R package version 2.4.7. 2017. URL: https://CRAN.R-project.org/package=fda.

[22] Guido Rossum. *Python Reference Manual*. Tech. rep. Amsterdam, The Netherlands, The Netherlands, 1995.

[23] Ulf Schepsmeier, Jakob Stoeber, Eike Christian Brechmann, Benedikt Graeler, Thomas Nagler, and Tobias Erhardt. *VineCopula: Statistical Inference of Vine Copulas*. R package version 2.1.4. 2018. URL: https://CRAN.R-project.org/package=VineCopula.

[24] Ulrike Schnoor. "Vector Boson Scattering and Electroweak Production of Two Like-Charge W Bosons and Two Jets at the Current and Future ATLAS Detector". PhD thesis. Fakultät Mathematik und Naturwissenschaften der Technischen Universit ät Dresden, 2015.

[25] J. Searcy, L. Huang, M.-A. Pleier, and J. Zhu. "Determination of the W W polarization fractions in p p $\rightarrow W^{\pm}W^{\pm}$j j using a deep machine learning technique". In: 93.9, 094033 (May 2016), p. 094033. arXiv: 1510.01691 [hep-ph].

[26] Steven Spiriti, Philip Smith, and Pierre Lecuyer. *freeknotsplines: Free-Knot Splines*. R package version 1.0. 2012. URL: https://CRAN.R-project.org/package=freeknotsplines.

[27] Thomas Nagler. "Kernel Methods for Vine Copula Estimation". MA thesis. Technische Universität München, 2014.

[28] *W Boson Decays*. ://www.hep.ucl.ac.uk/ jpc/all/ulthesis/node45.html.

[29] Matt Wand. *KernSmooth: Functions for Kernel Smoothing Supporting Wand Jones (1995)*. R package version 2.23-15. 2015. URL: https://CRAN.R-project.org/package=KernSmooth.

[30] Shanggang Zhou and Xiaotong Shen. "Spatially Adaptive Regression Splines and Accurate Knot Selection Schemes". In: *Journal of the American Statistical Association* 96.453 (2001), pp. 247–259. URL: https://doi.org/10.1198/016214501750332820.

Ringraziamenti

Dopo sei lunghissimi anni finalmente mi trovo a scrivere le mie ultime parole al Politecnico.
Sei anni fatti di sforzi, sofferenza, gratificazioni, momenti bui e momenti di gioia, sei anni che che mi hanno formato e reso la persona che sono ora.
Ma ora che finalmente, anche quasi inaspettatamente, la mia carriera universitaria sta per giungere al termine é il momento di ringraziare chi ha vissuto questi anni insieme a me.
Innanzitutto un grazie sentito alla professoressa Paganoni, per aver accettato per due volte di farmi da relatrice, per l'infinita pazienza, per l'attenzione data al mio lavoro e ai miei dubbi.
Grazie a Mamma e Papá, i miei primi sostenitori, gli ultimi ad arrendersi anche quando forse era giusto farlo. Mi siete sempre stati vicino, mi avete incoraggiato quando ne avevo piú bisogno, mi avete rimproverato quando era necessario.
Grazie a Marco, Luca, Ema e Beatrice che siete sempre presenti, per ascoltarmi parlare ore di universitá senza capire niente, per lasciarmi sfogare quando ne avevo bisogno, per dirmi senza paura quello che andava detto.
Grazie a Nicoló, Peri, Mowa, Manuela e Gra per essere stati fantistici compagni di progetto, per avermi aiutato ogni volta che ne avevo bisogno, che fossero appunti mancanti o spiegazioni in ritardo o per avermi fatto apprezzare un po' di piú l'universitá.
Grazie a Bezu e Luchino per avermi sempre ricordato chi sono e per essermi stati vicino anche quando pensavo di non volere nessuno.
Grazie a Niki, Jacopo, Andrea, Martina e Mirocle, fidati compagni di biblioteca, sempre presenti, sempre di aiuto e sempre pronti a fare pausa e farmi allontanare dal mio posto.
Grazie a Luca, Sabbo e Pibe per essere stati i migliori compagni, le migliori persone che mai avrei mai potuto immaginare di poter trovare in universitá, sempre pronti, sempre presenti, sempre in ultima fila a farmi compagnia.

Grazie a Sara e Beatrice, per avermi aiutato ad ogni volta che ne avessi avuto bisogno, per esserci sempre state presenti per qualsiasi dubbio io possa aver avuto. Un ultimo, immenso e veramente sentito grazie va a Diego, Alessandra e Gianluca ai quali, peró, un semplice grazie non puó bastare. Ai quali non puó bastare qualche semplice pensiero. Ai quali mi é davvero difficile trovare le giuste parole per descrivere quanto siete stati importanti per me. Voi piú di tutti sapete conoscete bene quanto difficile e faticoso sia stato questo percorso. Una gran parte di questo mio successo é merito vostro. GRAZIE!

Il vostro Paolo