Report CMEPDA 2022/2023

Measurement of the forward-backward asymmetry of Drell–Yan events in pp collisions at 8 TeV

Gamba Sara Nigrelli Giulia

Abstract

A software has been designed to reproduce the analysis of the weak mixing angle using the forward-backward asymmetry of Drell-Yan muon pairs produced in proton-proton collisions at $\sqrt{s} = 8$ TeV at CMS experiment of the LHC [3]. The data correspond to an integrated luminosity of 18.8 fb^{-1}. The analysis provides a comparison between the data recorded by the CMS experiment and those obtained from the Monte Carlo simulations of collision events. In particular the software produces the mass distribution of the muon pairs and their angular distribution and also the forward-backward asymmetry. The first two analyses are performed for three dimuon rapidity ranges: $0.0 < |y_{\mu\mu}| < 0.4$, $0.8 < |y_{\mu\mu}| < 1.2$, $1.6 < |y_{\mu\mu}| < 2.0$, while for the forward-backward asymmetry six ranges have been used; $0.0 < |y_{\mu\mu}| < 0.4$, $0.4 < |y_{\mu\mu}| < 0.8$, $0.8 < |y_{\mu\mu}| < 1.2$, $1.2 < |y_{\mu\mu}| < 1.6$, $1.6 < |y_{\mu\mu}| < 2.0$, $2.0 < |y_{\mu\mu}| < 2.4$. The software also allows to filter the input data to make them readable by the developed functions.

The software is stored in the following repository on GitHub: `https://github.com/sgamba2/cmepdaexam`.

Contents

1 Introduction

In this report we present a calculation of the forward-backward asymmetry (A_{FB}) in Drell–Yan events $q\bar{q} \rightarrow l^+l^-$, where l is a lepton and in our case stands for muon(μ). The calculation has been performed through the software developed by the authors.

The analysis is based on proton-proton collisions at $\sqrt{s} = 8$ TeV data from the CMS experiment at the CERN LHC. At leading order, muon pairs are produced from the annihilation of a quark with its antiquark into a Z boson or a virtual photon: $q\bar{q} \rightarrow Z/\gamma \rightarrow l^+l^-$.

The expression of A_{FB} is based on the angle θ^* of the negative lepton in the Collins-Soper frame of the dilepton system. In this frame, θ^* is the angle of the l^- relative to the axis that bisects the angle between the quark direction and the reversed antiquark direction. In proton-proton collisions, due to the PDFs (Parton Distribution Function), it is more likely to have a quark in the same direction of the Lorentz boost of the dilepton. Therefore, $cos(\theta^*)$ has been calculated as:

$$cos(\theta^*) = \frac{2(P_1^+ P_2^- - P_1^- P_2^+)}{\sqrt{m_{ll}^2(m_{ll}^2 + p_{T,ll}^2)}} \frac{p_{z,ll}}{|p_{z,ll}|}; \tag{1}$$

where $m_{ll},\ p_{z,ll},\ p_{T,ll}$ are respectively the mass, longitudinal momentum and transverse momentum of the dilepton system, and $p_i^{+,-} = (E_i \pm p_{z,i})/\sqrt{2}$ are expressed in term of the energies (E_i) and longitudinal momenta ($p_{z,i}$) of the negatively and positively charged leptons.

For each value of the dilepton invariant mass m_{ll}, the differential cross section at leading order (LO) is:

$$\frac{d\sigma}{d(cos\theta^*)} \propto 1 + cos^2\theta^* + A_4 cos\theta^*; \tag{2}$$

where the $(1 + cos^2\theta^*)$ term reflects the exchange of a spin-1 boson (Z or γ) and the $cos\theta^*$ comes from the V-A structure of weak interaction. The cross section is expressed at parton level. A_{FB} is defined as:

$$A_{FB} = \frac{3}{8}A_4 = \frac{\sigma_F - \sigma_B}{\sigma_F + \sigma_B}; \tag{3}$$

where σ_F and σ_B are, respectively, the cross sections in the forward ($cos(\theta^*) > 0$) and backward ($cos(\theta^*) < 0$) hemispheres.

This report aims to illustrate the results obtained by the developed software, partially reproducing the measurement of the weak mixing angle[3]. In addition, to compare MC data and Run data, it was necessary to develop a filtering algorithm for MC data that in principle contain more information than required for the desired analysis.

2 Data and simulated events

The analysis is based on the DoubleMuParked dataset in AOD format from 2012 RunC of the CMS experiment and on the MC simulations[1]. Both Run data and simulation data contain background events, which were not deleted during analysis. For this reason, the results obtained show discrepancies with the original analysis [3], in particular in areas far from the peaks. This can be observed in both $cos\theta^*$ and spectrum plots shown in the figures (2) and (1) respectively.

The events are selected through multiple triggers in order to consider for the analysis only dimuon from the production of a Z boson. In our software we can insert two ROOT files path, in order to filter datas: one for MC datas and one for Run datas. All the events must fulfill the following kinematic conditions:

1. Muon number trigger: $n_\mu = 2$;
2. Charge trigger: $Q_{\mu,1} \cdot Q_{\mu,2} = -1$;
3. Rapidity trigger: $|\eta| < 2.4$;
4. Transverse momentum trigger: $p_{1,T} > 25$ GeV and $p_{2,T} > 15$ GeV or viceversa;
5. Transverse distance trigger: $d_T < 0.2$ cm;
6. Muon isolation trigger: $Iso_{1,2} < 10\%\ p_T$;

Muon tracks are required to pass within a transverse distance of 0.2 cm from the primary vertex (the pp vertex with the largest $\sum p_T^2$ of its associated tracks); muon tracks must be isolated, i.e. the scalar p_T^2 sum of all tracks within a cone of $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\Phi)^2} = 0.3$ around the muon is larger than 10% of the muon p_T. The filters chosen are based on the specifications of both events and CMS detector[2].

It is useful to save this filtered variables for the two muons:

1. Trasversal momentum ($Muon\_pt$);
2. Pseudorapidity ($Muon\_\eta$);
3. ϕ coordinate ($Muon\_\phi$);
4. Mass ($Muon\_mass$).

[1]We have found datas here: https://eospublichttp01.cern.ch/eos/opendata/cms/derived-data/NanoAODRun1/01-Jul-22/

3 Dimuon mass spectrum of Z

The software produces the normalized histograms of the dimuon mass spectrum of Z boson. Manipulating the dataframes the new variables of the dimuon system are defined, from which the invariant mass of the system can be calculated and the rapidity as well. After that we have created three different filters for three different dimuon rapidity ranges: $0.0 < |y_{\mu\mu}| < 0.4$, $0.8 < |y_{\mu\mu}| < 1.2$, $1.6 < |y_{\mu\mu}| < 2.0$. The histograms are normalized with the total number of the events of MC and Run datas respectively.

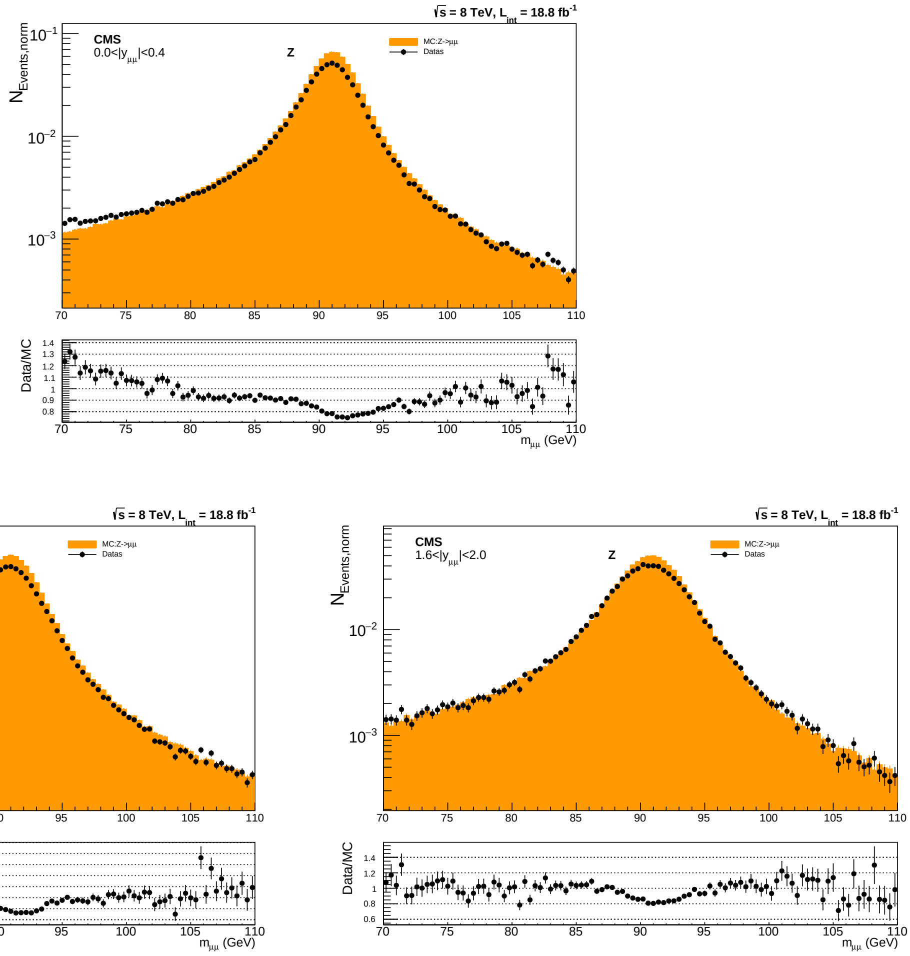

Figure 1: Dimuon mass distributions in three representative bins in rapidity: $|y_{\mu\mu}| < 0.4$ (upper),$0.8 < |y_{\mu\mu}| < 1.2$ (lower-left), and $1.6 < |y_{\mu\mu}| < 2.0$ (lower-right). MC data are plotted in yellow and Run data as black dots. Each graph also shows the ratio between the two diagrams at the bottom. We used 100 bins.

The ratio of the histograms tends to one, therefore the simulations are consistent with the data of the Run and the filters applied are correct. We can see some differences between MC and Run data: this is because we haven't considered the background. We can see these differences only when $m_{\mu\mu}$ tends to the extremes of the histogram: in these limits the MC signal is two orders of magnitude less than when $m_{\mu\mu}$ is equal to the Z mass, so the background effects is most effective.

4 Angular distribution

The developed software also produces the angular distribution in $\cos(\theta^*)$, where θ^* is the angle of the negative muon in the Collins–Soper frame of the dimuon system as explained in (1). The histograms are normalized with the total number of the events of MC and Run datas respectively. Rapidity ranges are the same as for Z-resonance graphs (1).

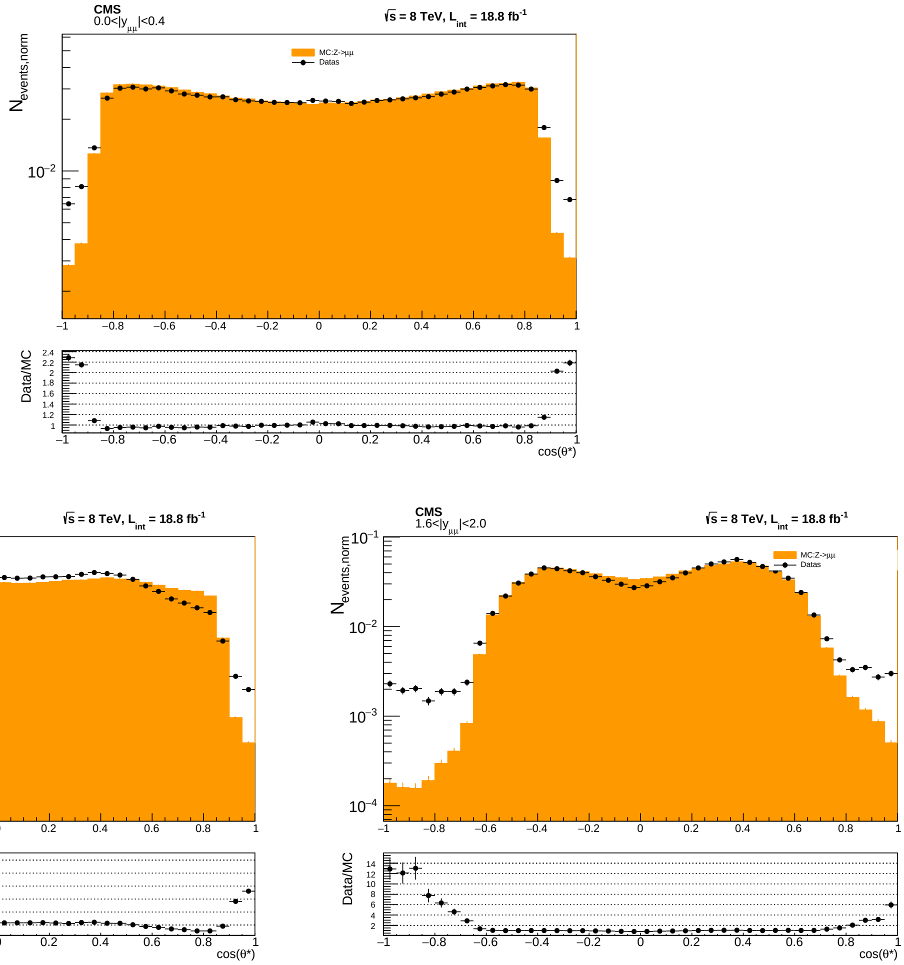

Figure 2: Muon $cos(\theta^*)$ distributions in three representative bins in rapidity: $|y_{\mu\mu}| < 0.4$ (upper), $0.8 < |y_{\mu\mu}| < 1.2$ (lower-left), and $1.6 < |y_{\mu\mu}| < 2.0$ (lower-right). MC data are plotted in yellow and Run data as black dots. Each graph also shows the ratio between the two diagrams at the bottom. We used bins of width of 0.05 -in total 40 bins.

As for Z invariant mass distributions, the ratio of the histograms tends to one, therefore the simulations are consistent with the data of the Run and the filters applied are correct. We can see some differences between MC and Run data: this is because we haven't considered the background. We can see these differences only when $|cos(\theta^*)| \to 1$: in these limits the MC signal is three orders of magnitude less than when $cos(\theta^*)$ is equal to zero, so the background effects is most effective.

5 Forward-backward asymmetry

In Section (1) is shown the LO angular distribution of dilepton events but there is also a NLO term from the p_T of the interacting partons [2]. Therefore, the angular distribution in $cos(\theta^*)$ in each bin (m_{ll}, y_{ll}) has the form:

$$\frac{1}{\sigma}\frac{d\sigma}{d(cos\theta^*)} = \frac{3}{8}[1 + cos^2\theta^* + \frac{A_0}{2}(1 - 3cos^2\theta^*) + A_4 cos\theta^*]; \tag{4}$$

Taking this into account, the value of A_{FB} can be calculated in each bin through "angular event weighting" method (described in [1]). The weighted A_{FB} of Eq. (3) can be written as:

$$A_{FB} = \frac{3}{8}\frac{N_F - N_B}{D_F + D_B}; \tag{5}$$

where D_F, D_B, N_F, N_B are denominators and numerators for forward ($cos\theta^* > 0$) and backward ($cos\theta^* < 0$) events defined as:

$$\begin{aligned} D_F &= \sum_{c>0} w_D;\ D_B = \sum_{c<0} w_D; \\ N_F &= \sum_{c>0} w_N;\ N_B = \sum_{c<0} w_N; \end{aligned} \tag{6}$$

where w_D, w_N are weights that reflect the angular dependence on $cos\theta^*$, indeed they are expressed in terms of $c = cos\theta^*$:

$$\begin{aligned} w_D &= \frac{1}{2}\frac{c^2}{(1 + c^2 + h)^3}; \\ w_N &= \frac{1}{2}\frac{|c|}{(1 + c^2 + h)^2}; \end{aligned} \tag{7}$$

where $h = 0.5A_0(1 - 3c^2)$; we set $A_0 = 0.1$ which is the $p_{T,ll}$-averaged predicted by the signal MC simulation [3].

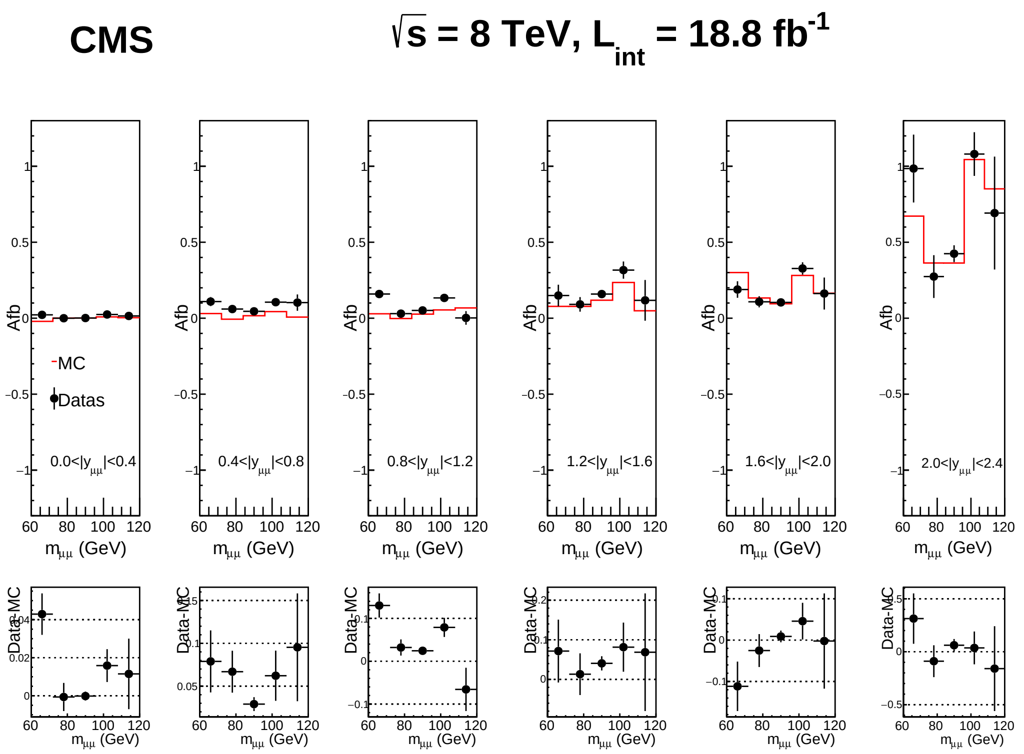

Figure 3: A_{FB} distributions for the dimuon in six ranges of rapidity $0.0 < |y_{\mu\mu}| < 0.4$, $0.4 < |y_{\mu\mu}| < 0.8$, $0.8 < |y_{\mu\mu}| < 1.2$, $1.2 < |y_{\mu\mu}| < 1.6$, $1.6 < |y_{\mu\mu}| < 2.0$, $2.0 < |y_{\mu\mu}| < 2.4$. In red we can see the MC data and in black the Run data. We have also reported the difference between the two histograms.

The software calculates the weights and produces histograms in the six rapidity ranges, as shown in figure (3). We haven't done the fit of these histograms because, differently from the article, we didn't have a model that described the value of A_{FB} in terms of the weak mixing angle θ_W. For these reasons a comparison was made between MC and Run data.

The trend of the histograms is also different from the trend in the article. One of the difference could be the binning of the histograms: we used five bins on x and on y, instead the authors of the article decided to use twelve bin for x and six for y. The reason we used less bins is that our samples are much smaller than theirs. However, both the A_{FB} of the article and ours increase in module increasing the rapidity range. In order to compare our result for A_{FB} optimally we need to increase the statistics and formulate a model for the fit.

6 Software description

6.1 Functionalities

The whole project is written in $C++$ files that are compiled in python, through the execution of the code openfiles.py. This code defines the process function that performs the analysis chosen by the user on the loaded data. The selection of the analysis and the data upload is done through the flags. (More information about how to run are in the software documentation (6.3)). The data are stored in root files through the ROOT class RDataFrame.

Multi-threading is enabled: we used EnableImplicitMT() and the number of threads is automatically decided by the implementation, but it is configurable according to the system used.

The workflow of the program can be summarized as:

1. Loading and reading the data from both MC simulations and Run root files;
2. Application/control filters on data;
3. Creation of new filtered data files;
4. Choice of the analysis;
5. Definition of new variables in the dataframes;
6. Histogram production;
7. Result saving in pdf and png.

In order to work properly filtered data must be used, for this reason the opening code includes a flag to filter or not the data before the analysis (–filter [y/n]). If the user chooses 'n', there must be already filtered files in datas folder. More information about that can be found in the software documentation (6.3). The software needs root files to work properly.

The software allows the user to choose between three types of analysis:

1. the measurement of the Z resonance, calling the **dimuonSpectrumZ** function;
2. the measurement of the angle of the negative muon in the Collins–Soper frame of the dimuon system, calling the **costheta** function;
3. the measurement of the forward-backward asymmetry, calling the **afb** function.

Each function manipulates the dataframes in a different way to obtain a better analysis and avoid the overload of the machine on which the program is running. To optimize the program, a file has been created, called utilities.h, in which are defined all the functions to modify the dataframes.

All figures are produced both in png and pdf format and automatically saved in the images folder, in the respective subfolders with the name of the chosen analysis. A dedicated file has been created to optimize graphs management, called graphicalUtilities.h. (More information about the nomenclature of files can be found in the documentation (6.3)).

6.1.1 Filter

In **filterDf function** we can insert two ROOT files path, in order to filter datas: one for MC datas and one for Run datas[2]. In our software, before the analysis, we check if the files exist and if they have the right columns: $nMuon$, $Muon\_mass$, $Muon\_charge$, $Muon\_phi$, $Muon\_eta$, $Muon\_pt$, $Muon\_dxy$, $Muon\_pfRelIso03\_chg$. If the paths don't exist, if the files haven't the right column or there are less than 50 events in the dataframes the software doesn't create the filtered files. The function applies the constraints listed in Section (2) and reports the statistics of the data sets obtained, even checking if the dataframes are almost empty. The function creates two files with the Snapshot function provided by ROOT, with the four columns listed in Section (2): one for the MC events and one for the Run events. The name of the files can be choosen by the user. These files will be created in the datas folder: if it doesn't exist, the program will create it.

[2]We have found datas here: https://eospublichttp01.cern.ch/eos/opendata/cms/derived-data/NanoAODRun1/01-Jul-22/

6.1.2 Dimuon mass spectrum analysis

The **dimuonSpectrumZ function** checks the filtered data path and columns then creates two dataframes for MC dataset and for Run dataset. Then it extracts the quadrivectot function from utilities.h, such function creates a four vector for the dimuon system of the type PtEtaPhiMVector. Two more columns are added to the dataframes by extracting mass and velocity from the quadrivector. Finally it filters the data in the three rapidity ranges, defined in the abstract, and produces the normalized histograms with respect to the total number of events as shown in figure (1). The histograms are saved in the directory images/dimuonspectrumZ: if they don't exist, then the program will create them. These operations are managed through the use of graphicalUtilities.h header.

6.1.3 Angular distribution

The **costheta function** checks the filtered data path and columns then creates two dataframes for MC dataset and for Run dataset. The following step is to extract the allquantities function from utilities.h, such function adds to the dataframes 16 columns: three columns for dimuon system and individual muon four vectors, five columns for invariant mass, rapidity, transverse momenta, longitudinal momenta and energy of the dimuon system, eight columns, four for each muon, for energy, longitudinal momenta, p^+ and p^- (defined in Section (1)), one column for $cos(\theta^*)$. Finally it filters the data in the three rapidity ranges, defined in the abstract, and produces the normalized histograms with respect to the total number of events as shown in figure (2). The histograms are saved in the directory images/costheta: if they don't exist, then the program will create them. These operations are managed through the use of graphicalUtilities.h header.

6.1.4 Forward-backward asymmetry

The function checks the filtered data path and columns then creates two dataframes for MC dataset and for Run dataset. After that, allquantities function is extracted, as for $cos(\theta^*)$, two more columns are added for the weights w_D, w_N defined in (7). The datasets are filtered in the six rapidity ranges defined in the abstract. We have done eight 2D histograms, four for MC and four for Run data. On the x there was dimuon mass, on y there was rapidity and it was weighted with w_D or w_N. Two histogram of each data group were done with the condition of $cos(\theta^*) > 0$ and the other two with $cos(\theta^*) < 0$. Then we operated on the histograms, summing, subtracting, dividing and scaling them in operationHist(managed through the use of graphicalUtilities.h). Six produced histograms are merged in a single graph as shown in figure (3). The histograms are saved in the directory images/afb: if they don't exist, then the program will create them.

6.2 Testing

The interface of the tests is the same of the analyses, therefore through a compilation in python it is possible to run the tests passing as arguments both filtered data and not using the respective flags. More information can be found in the documentation (6.3) in the file section testing.py.

The software offers five types of tests, on the main functions used in analysis:

1. test on filter function;
2. test on $cos\theta^*$ value;
3. test on energy value;
4. test on energy formulas value;
5. tests on operations between histograms.

6.2.1 Filter test

In order to test the filter functon two empty TTree are created, one with the right number of columns and one with a missing column. The testFilt function checks that the correct error flag is printed in several cases: wrong file path, wrong file extension, empty dataframe, wrong dataframe format. It prints "Test passed!" everything was done correctly or "Test failed!".

6.2.2 Angular distribution test

The $cos\theta^*$ test creates a function testCos in which two dataframes are created from MC and Run dataset, to which are added the columns necessary for the calculation of $cos\theta^*$ through the function allquantities. Applying a filter that selects events with $|cos\theta^*| > 1$, it is possible to count events in which the $cos\theta^*$ has not been correctly calculated. It prints "Test passed!" if there are no such events left or "Test failed!".

6.2.3 Energy value test

The energy value test creates a function testEnergy that tests the energy value in the same way as the $cos\theta^*$ value test, therefore it counts all the columns for which $E_1 + E_2 - E \neq 0$. We couldn't explicit write this statement, so we decided to verify the equation in a ϵ of $0.01\%E$. It prints "Test passed!" if there are no such events left or "Test failed!".

6.2.4 Energy formula test

The energy formulas test creates a function testEnergyFormulas that defines both for MC data and for Run data six new columns (p_x, p_y and $E = \sqrt{m^2 + p_x^2 + p_y^2 + p_z^2}$) and applies to them a filter that selects events with $E_{default} \neq E_{defined}$. We couldn't explicit write this statement, so we decided to verify the equation in a ϵ of $0.01\%E$. It prints "Test passed!" if there are no such events left or "Test failed!".

6.2.5 Operations on histograms test

The test on histogram operations creates a function testOperationHist that generates two files with TTree inside, which are then opened as RDataframe. Four histograms are built from dataframes. From dataframes four empty histograms are constructed, the contents of the bins are modified through the TH2D::AddBinContent() method. We added four random variables, also rearranged, to the four histograms in four different bins. Then, the operationhist function extracted from the graphicalUtilities header is applied on the four diagrams. At the end the contents of the bins are compared with the expected value. As before, we couldn't explicit write the = statement, so we decided to verify the equation in a ϵ of $0.01\%var$, where var is the random variable. If there are no differences it prints "Test passed!" vice versa if there are discrepancies it prints "Test failed!".

6.3 Documentation

The documentation was produced using **Doxygen**, which is the standard tool for generating documentation through annotations in the C++ source. We have used **Doxygen** also for the documentation of the python files. More specific information about function definitions can be found in the documentation at the following link: `https://sgamba2.github.io/`. All the software is saved at the following link on github: `https://github.com/sgamba2/cmepdaexam`.

References

[1] A. Bodek. "A simple event weighting technique: optimizing the measurement of the forward-backward asymmetry of Drell-Yan dilepton and top-antitop pairs at hadron colliders". In: *The European Physical Journal C* (2010). DOI: 10.1140/epjc/s10052-010-1287-5. URL: https://doi.org/10.1140/epjc/s10052-010-1287-5.

[2] E. Mirkes and J. Ohnemus. "W and Z polarization effects in hadronic collisions". In: *Phys. Rev. D* 50 (9 Nov. 1994), pp. 5692–5703. DOI: 10.1103/PhysRevD.50.5692. URL: https://link.aps.org/doi/10.1103/PhysRevD.50.5692.

[3] Albert M. Sirunyan et al. "Measurement of the weak mixing angle using the forward-backward asymmetry of Drell-Yan events in pp collisions at 8 TeV". In: *Eur. Phys. J. C* 78.9 (2018), p. 701. DOI: 10.1140/epjc/s10052-018-6148-7. arXiv: 1806.00863 [hep-ex].