Python - ECON107

PROJECT: Data Analysis in Python

High Energy Electron Collisions

Meliha Delalić 220302150 – Department of Mechanical Engineering

Deniz Durmus, Adjunct Instructor

Sarajevo, December 2024.

Table of Contents

1. Introduction

This paper will discuss the procedure and the findings behind data analysis done on electron-electron collision data collected by Compact Muon Solenoid (CMS) general purpose detector at CERN. The main objective of this project is using Python for statistical analysis of the data, as well as its built-in libraries such as Matplotlib and Seaborn for visualizing the results and inspect the trends of a post electron-electron collision. To better understand the results of the analysis, there will be provided a gentle introduction to the world of particle physics, and the main objectives of experiments done at Large Hadron Collider (LHC).

Software used for this project is Jupyter Notebook, which allows interactive code workflow through neatly organized cells of code and their explanations, as well as observing visualizations in real time.

2. Particle Collisions at LHC

Large Hadron Collider (LHC) is the world's most powerful particle accelerator, designed with the goal to answer some of the biggest questions about our universe, by finding ways to create new particles, as well as to explain the existance of those we are already well aware of. Even though most of the physics phenomena has already been well explained and studied, there are still numerous matters that have not gotten their scientific explanation, which require experimental work and profound insight into the world of particle collisions. Questions like origin of the mass, large absence of antimatter with respect to matter, and the true nature of dark energy are topics that have unanswered to this day. Accelerating and colliding particles is a good start to answering them, and the number of events required to occur to get a solid base for serious analysis reaches astronomical amounts every day, with LHC generating up to 1 billion proton-proton collisions each second. This amount of data is impossible to process even for the worlds fastest computers, which is why particle detectors such as CMS and ATLAS have built-in systems for filtering all the less interesting events.

However, these detectors still collect very large datasets ready for analysis each day. Understanding them requires knowledge of Standard Model which describes fundamental particles and their collision, detector technologies that are implemented, as well as statistical methods used to analyze the patterns and statistical trends in the dataset.

Picture 1 – The Large Hadron Collider, burried 100 m underground, used to accelerate particles thanks to its large radius and different magnets used to direct the particles to the collision point.

2.1. Detection of the Data – Positional Parameters ***ν*** (nu) and η (eta)

Several detectors at CERN are responsible for tracking events and collecting interesting events such as CMS, ALICE and ATLAS, where each of these employs advanced technologies such as electromagnetic calorimeters, superconducting magnets and inner tracking systems which are responsible for detecting and tracking different particles projectories. Since our dataset is gathered by CMS, we will discuss it in more detail.

CMS acts as a giant, high-speed camera, taking 3D "photographs" of particle collisions from all directions up to 40 million times each second. [1] Capturing particle trajectories,

requires several steps, from bending the charged particles trajectory, identifying it with a silicon tracker and then measuring its energy using electromagnetic calorimeters.

Our special concerning property of this detector is its coordinate system which relies on measuring two primary parameters that determine the position of the particle. These are pseudorapidity (η (eta)) and azimuthal angle *ν* (nu).

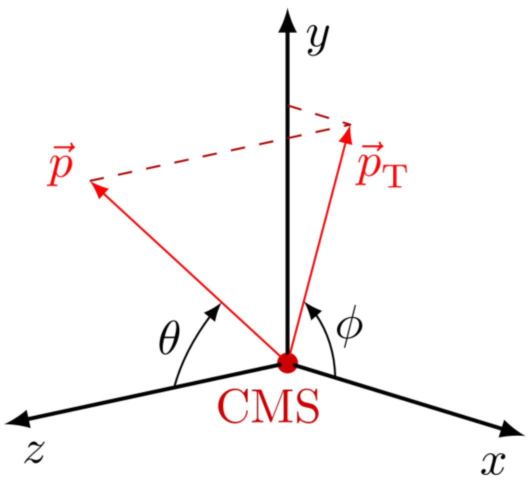


Figure 1 – CMS Coordinate system with angles theta and phi describing the position of the particle, where p_T is the transverse component of momentum.

Pseudorapidity describes the position of the particle with respect to the beam axis, which is z-axis in this case. In the figure above, phi represents the azimuthal angle, and theta is the pseudorapidity angle. These parameters will help us understand spatial distribution and the dynamics of the collision events.

[1] https://cms.cern/detector

2.2. Energy, Momentum and Invariant Mass

All experimental data which is detected belongs to either intermediate or final stages of the proton-proton collisions that are initialized at LHC. In this paper, we are analyzing electron-electron collision events that represent final decay products, produced by intermediate particles.

During this process, also known as Drell-Yan process, a quark from one proton and an antiquark from another proton annihilate to produce a virtual photon or a Z-boson, which then decay into an electron-positron pair. However, analysis in this paper focuses on electron-electron pairs only, excluding positrons and positron pairs for a more coarse analysis of electron-electron collisions.

Energy, momentum and invariant mass are the three most valuable physical indicators of the collision dynamics because they allow reconstruction of the parent particles, and serve as a test for theoretical models such as Quantum Electrodynamics, where small discrepancies between the model and the experimental results call for refinement of the theroetical postulates.

Invariant mass is one of the most important properties defining a system of particles, which describes the relationship between energy and momentum.

$$M = \sqrt{(E_1 + E_2)^2 - \|p_1 + p_2\|^2}$$

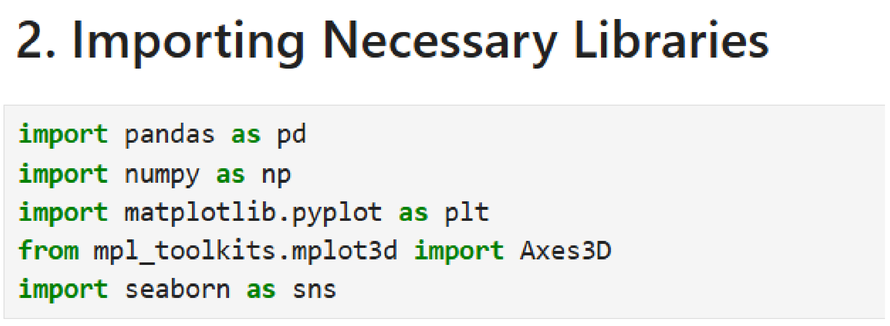
This mass is independent of the environemnt of the particle, hence the name invariant, and this is why it is an extremely powerful tool for discovering new particles such as Higgs Boson which have very high speeds and may not have mass. In the case of Higgs Boson, we know that it originates from four electrons, whose momentum and energy is easy to measure in a detector. Therefore we can calculate the mass of Higgs Boson as these electrons come from it.

Studying invariant mass is especially important when trying to discover a new particle, since any peaks in invariant mass which do not correspond to some already known particle could mean that a new particle was found.

However, the primary objective of this paper is to observe the standard statistical trends present among these properties to search for areas of high energy, look for potential peaks in invariant mass and observe the relationship between momentum components and energy histograms, scatter plots and heatmaps, which will enable us to better understand the relationship between these properties and how they describe a decay process.

3. Data Setup and Preprocessing

Before conducting any statistical methods or analysis, the raw data must be imported and properly handled. Just then, we can move to actual analysis of the data and visualizing the results.



2. Importing Necessary Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import seaborn as sns
```

Figure 2 – Importing libraries for data analysis. Axes3D class is imported from the mpl_toolkits module for creating a 3D plot of momentum.

3.1. Importing Libraries

Three libraries used for analysis include Pandas, Matplotlib, Seaborn and NumPy. Pandas provides us with the main tools for data manipulation and analysis as it is powerful, fast and easy to use. It is responsible for reading and importing the dataset, cleaning the dataset, and some standard functions for statistical methods such as mean(), max() and min().

Matplotlib and Seaborn are used for visualizing the results and data correlation. Matplotlib is used to create histograms and scatter plots necessary for studying the relationship between the key variables from the dataset such as energies of the two electrons, or transverse momentum distribution. Seaborn helps us see these relationship in a more compact way through 2D pairplots and correlation heatmaps.

Finally, NumPy is used primarily for calculating the event-based invariant mass for direct comparison with the experimental invariant mass.

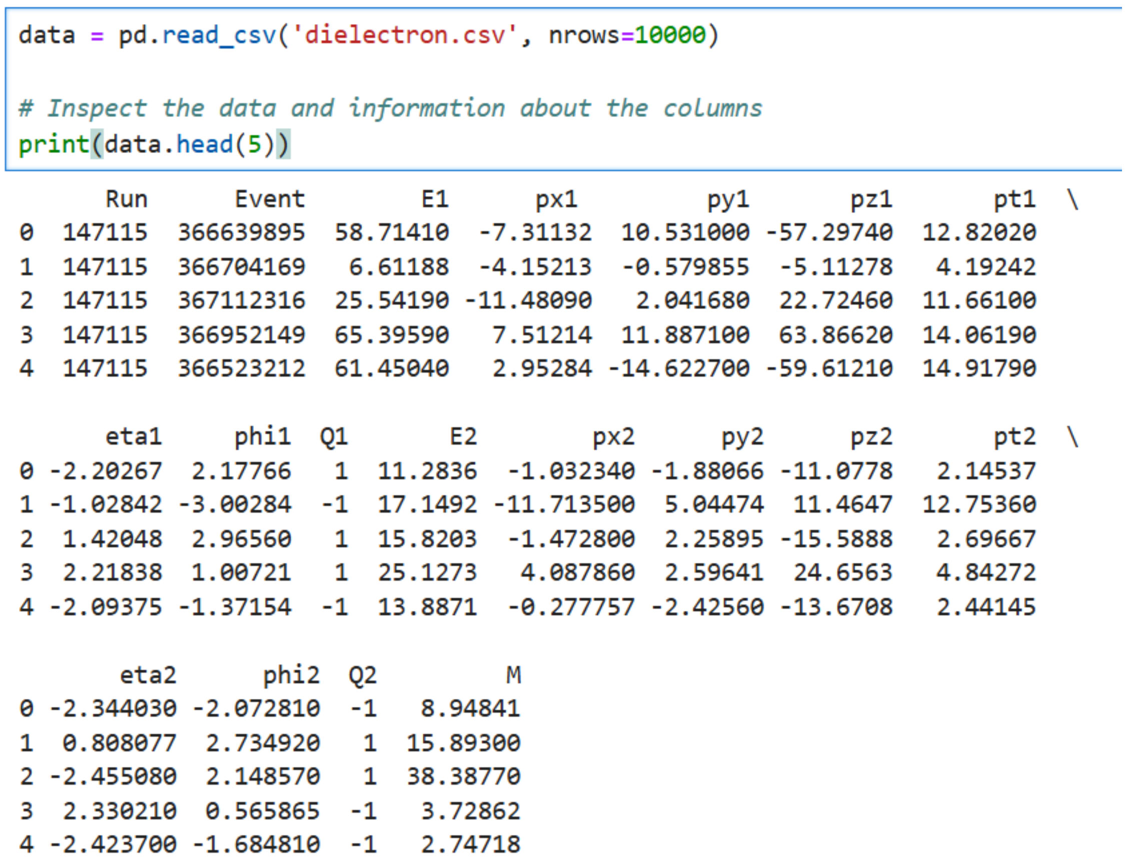
3.2. Importing Data

First step in data analysis is reading the data file into our Jupyter Notebook. This is done by read_csv() function from the Pandas library. Since the dataset we are working on is very large and contains 100000 dielectron events, we only import the first 10000 events for a more simple analysis and data management. However, it is important to note that working on a subset can often result in a lack of conclusive results or incomplete analysis.

The entire dataset is converted to a DataFrame format named „data“ which is a two-dimensional data structure containing all events and their properties. Later, every time we wish to perform any change to the dataset or analyze the results, we refer to each column in the dataframe as data[„column_name“].

After importing the data, it is useful to call head() function that allows us to see first few rows of the dataframe. This way we can see the exact order of the columns and their names. Keyword „nrows“ is used to specify the number of rows we wish to include in the DataFrame.

Columns „Run“ and „event“ give us information about the collision number. Each run refers to a specific experimental trial within which multiple events occur. E1 and E2 are for storing the electron energy values right after collision, Q1 and Q2 are the charges of each electron (or positron). The positional parameters eta1, eta2 tell us about the position of the particle with respect to the beam axis for each electron (pseudorapidity), and phi1, phi2 specify the azimuthal angle of the particle. This is because CMS is a detector that operates in a spherical coordinate system. Momentum components include the momentum in the x,y and z directions (px1, py1,pz1 and px2, py2, pz2), as well as the transverse momentum components (pt1, pt2) which are perpendicular to the beam axis. Lastly, column M contains the values of invariant mass for each electron.



```python
data = pd.read_csv('dielectron.csv', nrows=10000)

# Inspect the data and information about the columns
print(data.head(5))
```

```
      Run      Event        E1       px1         py1       pz1       pt1  \
0  147115  366639895  58.71410  -7.31132   10.531000 -57.29740  12.82020
1  147115  366704169   6.61188  -4.15213   -0.579855  -5.11278   4.19242
2  147115  367112316  25.54190 -11.48090    2.041680  22.72460  11.66100
3  147115  366952149  65.39590   7.51214   11.887100  63.86620  14.06190
4  147115  366523212  61.45040   2.95284  -14.622700 -59.61210  14.91790

      eta1     phi1  Q1       E2        px2       py2      pz2       pt2  \
0 -2.20267  2.17766   1  11.2836  -1.032340 -1.88066 -11.0778   2.14537
1 -1.02842 -3.00284  -1  17.1492 -11.713500  5.04474  11.4647  12.75360
2  1.42048  2.96560   1  15.8203  -1.472800  2.25895 -15.5888   2.69667
3  2.21838  1.00721   1  25.1273   4.087860  2.59641  24.6563   4.84272
4 -2.09375 -1.37154  -1  13.8871  -0.277757 -2.42560 -13.6708   2.44145

       eta2      phi2  Q2         M
0 -2.344030 -2.072810  -1   8.94841
1  0.808077  2.734920   1  15.89300
2 -2.455080  2.148570   1  38.38770
3  2.330210  0.565865  -1   3.72862
4 -2.423700 -1.684810  -1   2.74718
```

Figure 3 – DataFrame named „data" containing number of runs and events, and most important information for analysis for each event.

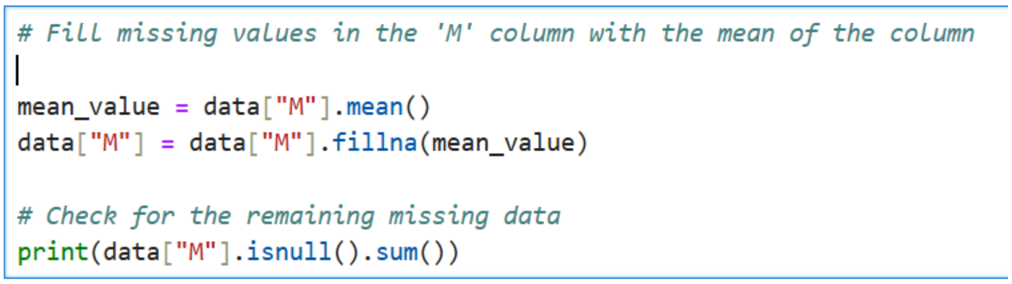
3.3. Cleaning Data

Cleaning the dataset is a crucial step before performing any type of analysis. We start this process by ensuring all column names are clear and ready to use. This is done by calling data..columns() line in our case. There was an extra space in the name of px1 which is fixed by calling rename() function.

Next is checking if there are any missing values in the dataset. By calling info() function, we saw that in the column M, there are only 9995 non-null values, indicating that the remianing 5 need to be properly addressed.

By running isnull() function on the column M, we can also confirm the exact columns numbers in which there are NaN values.

The mean of the column M is then calculated and replaced by the null values.


```
# Fill missing values in the 'M' column with the mean of the column

mean_value = data["M"].mean()
data["M"] = data["M"].fillna(mean_value)

# Check for the remaining missing data
print(data["M"].isnull().sum())
```

Figure 4 – Replacing NaN values by the mean of M values. Then again running isnull.().sum() will confirm if there are any other NaN values and sum them up.

To make sure that our dataset contains only electron pairs, we removed each row that contains either one or two positrons. Charges of the particles are stored in Q1 and Q2 columns, and we want to remove those that are +1. This is done when neither Q1 is equal to 1, and neither Q2.

data = data[(data["Q1"] != 1) & (data["Q2"] != 1)]

This line of code makes sure that both of these conditions are satisfied.

Final step includes looking for duplicate rows which is not impossible to find in such large datasets. Not removing them could affect the integrity and uniqueness of the results which is why this is done a a precaution. Same rows in this dataset would be identified if there are two rows with the same entrances in „Run“ and „event“. Because of that, we are using the following line that includes drop_duplicates() function:

data = data.drop_duplicates(subset = ["Run", "Event"])

However since the size of the DataFrame did not change after this step, we conclude that there were no duplicate rows.

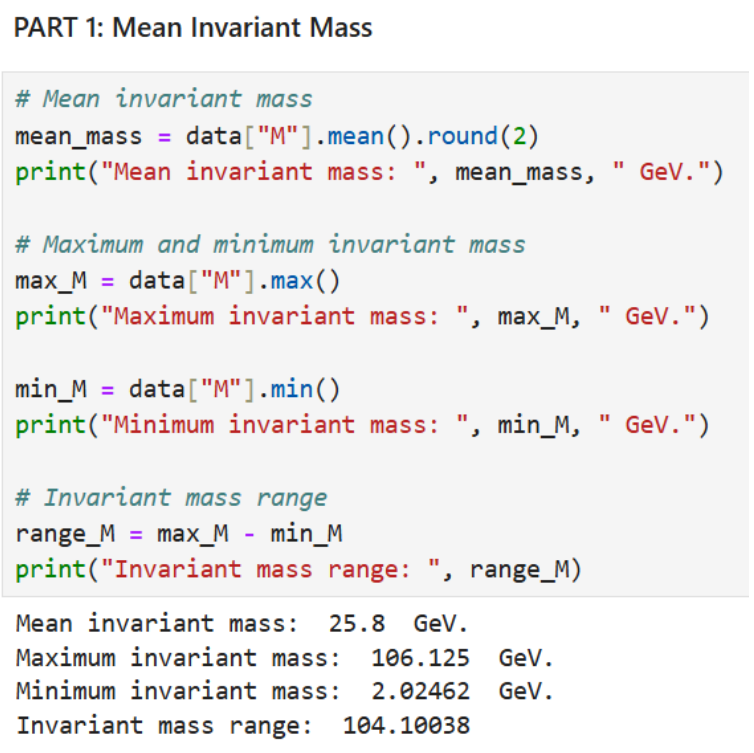
4. Data Processing

Analysis of the data is done by applying the descriptive statistical methods to the dataset, and then visualizing the results . This will help us better understand the occurrances in electron-electron collision, as well to give us insight into potentially interesting events.

4.1. Statistical Methods

Some of the statistical methods used for analysis of energy, momentum and invariant mass include calculating the central tendency (mean), maximum and minimum of a given value, range and standard deviation for measuring the spread of data from the mean value, as well as the variability measurements such as quantiles which track the general value of a given data by dividing it into equal parts.

4.1.1. Invariant Mass, and Event-Based Invariant Mass



PART 1: Mean Invariant Mass

```
# Mean invariant mass
mean_mass = data["M"].mean().round(2)
print("Mean invariant mass: ", mean_mass, " GeV.")

# Maximum and minimum invariant mass
max_M = data["M"].max()
print("Maximum invariant mass: ", max_M, " GeV.")

min_M = data["M"].min()
print("Minimum invariant mass: ", min_M, " GeV.")

# Invariant mass range
range_M = max_M - min_M
print("Invariant mass range: ", range_M)
```

```
Mean invariant mass:  25.8  GeV.
Maximum invariant mass:  106.125  GeV.
Minimum invariant mass:  2.02462  GeV.
Invariant mass range:  104.10038
```

Figure 5 – Mean, maximum, minimum and range of invariant mass.

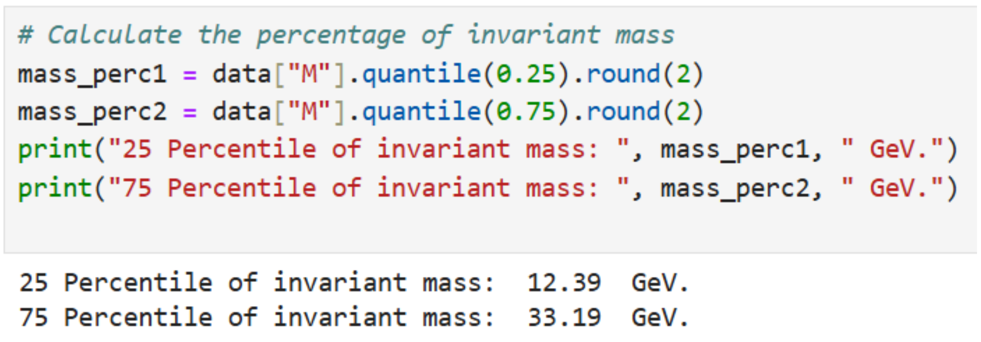
We can tell that the mean mass is on the lower end which indicates that most of the events results from background interactions which often result in less interesting particles. An important piece of information is the maximum mass which means that despite the central tendency being quite low, there are in fact some meaningful collisions which ended up in a particle with a bigger mass. We can compare this to a Z-boson which is often desired in such collisions that has mass of about 90 GeV.

The information above is also confirmed by a high range value that validates the existance of a higher-mass particles.

<u>Standard Deviation and Percentiles:</u>

To get a better grasp of the mass distribution, we also computed the standard deviation of the invariant mass by calling std() function. When compared to the mean of the mass, the resulting value of 19.95 tells us that most of the mass will be distributed between the values

$$25.8 - 19.95 = 5.85\ GeV \quad and\ 25.8 + 19.95 = 45.75\ GeV$$



```
# Calculate the percentage of invariant mass
mass_perc1 = data["M"].quantile(0.25).round(2)
mass_perc2 = data["M"].quantile(0.75).round(2)
print("25 Percentile of invariant mass: ", mass_perc1, " GeV.")
print("75 Percentile of invariant mass: ", mass_perc2, " GeV.")
```

```
25 Percentile of invariant mass:  12.39  GeV.
75 Percentile of invariant mass:  33.19  GeV.
```

Figure 6 - Calculating quantiles shows that 25% of events have an invariant mass less than or equal to 12.39 GeV 75% of events have an invariant mass less than or equal to 33.19 GeV

<u>Event-Based Invariant Mass:</u>

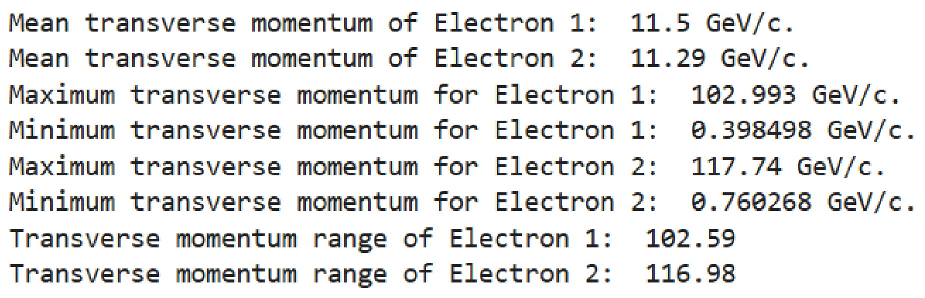
In the following line, we also calculate the invariant mass by using the information about energy and momentum. This is done to compare the distribution of both invariant mass as it is recorded by the detector, and the computed values.

data["Event_M"] = np.sqrt((data["E1"] + data["E2"])**2 - (data["px1"] + data["px2"])**2 - (data["py1"] + data["py2"])**2 - (data["pz1"] + data["pz2"])**2)

In the formula above, new column „Event_M" is created to store the new values of computed mass. To conduct the following mathematical operations, we refer to the NumPy library and call the function by np.function_name().

4.1.2. Transverse Momentum (pt1, pt2)

During an analysis of paricle collision, high transverse momentum can indicate the presence of hard scattering processes, whereas low momentum indicates dominance of softer interactions. Similar application of the statistical methods for transverse momentum such as for mass above yields the following results:



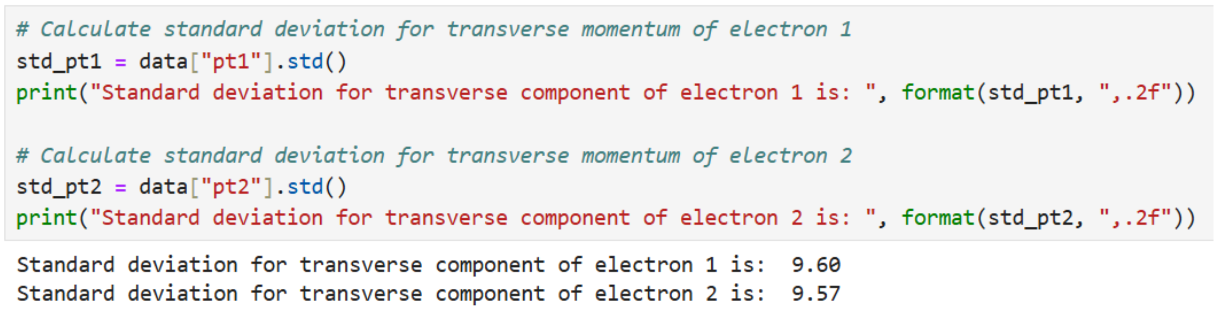
```
Mean transverse momentum of Electron 1:  11.5 GeV/c.
Mean transverse momentum of Electron 2:  11.29 GeV/c.
Maximum transverse momentum for Electron 1:  102.993 GeV/c.
Minimum transverse momentum for Electron 1:  0.398498 GeV/c.
Maximum transverse momentum for Electron 2:  117.74 GeV/c.
Minimum transverse momentum for Electron 2:  0.760268 GeV/c.
Transverse momentum range of Electron 1:  102.59
Transverse momentum range of Electron 2:  116.98
```

Mean values of 11.5 and 11.29 for both electrons are considered substantially high for a decay process as it is usually in the range of 10-15 that the electrons are a result of parton interactions in a proton-proton collision. That being said, it is very likely that the electrons are actually decay products of either some heavy bosons such s Z-boson, or a virtual photon.

Similarly, maximum momentum values found among the electrons imply that significant events occurred. Such high values of transverse momentum could mean that, if in fact these electrons are Z-boson decay products, then the boson itself had initially had very high momentum. This is common when the particles recoil off of other particles or collide with the jets to gain additional momentum. Very low momentum is present due to less significant events, and high range again confirms heavy particles origins.

<u>Standard Deviation and Percentiles</u>

As for standard deviation, the results show how momentum is fairly distributed around the mean value. Even though most of the particles' momentum is close to the mean value, still there are parts with much higher and lower transverse momentum, which again confirms the findings from above.



```python
# Calculate standard deviation for transverse momentum of electron 1
std_pt1 = data["pt1"].std()
print("Standard deviation for transverse component of electron 1 is: ", format(std_pt1, ",.2f"))

# Calculate standard deviation for transverse momentum of electron 2
std_pt2 = data["pt2"].std()
print("Standard deviation for transverse component of electron 2 is: ", format(std_pt2, ",.2f"))
```

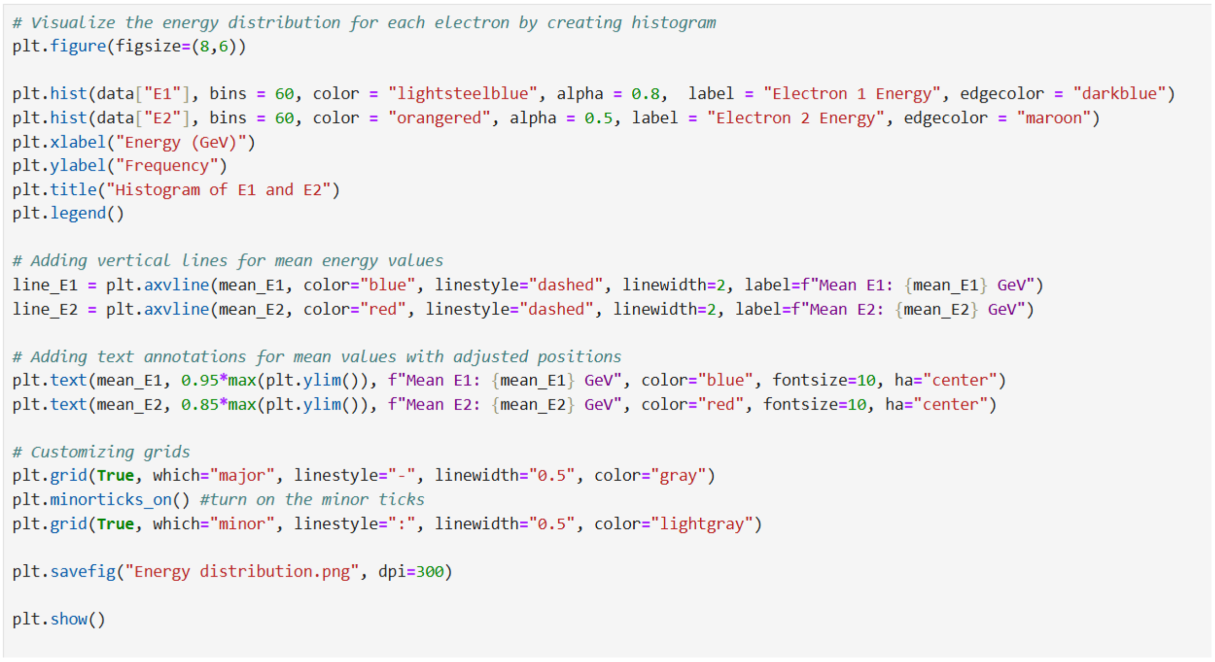
```
Standard deviation for transverse component of electron 1 is:  9.60
Standard deviation for transverse component of electron 2 is:  9.57
```

Figure 7 – Transverse momentum standard deviation.

For Electron 1, the 25th percentile of the transverse momentum is 3.27 GeV/c, meaning that 25% of the electrons have a pt1 less than or equal to this value. The 75th percentile is 16.92 GeV/c, meaning that 75% of the electrons have a pt1 less than or equal to this value. Similar interpretations can be made for Electron 2.

4.2. Visualization of Results

First type of visualizing technique used are histograms which plot the variable distribution with respect to the frequency, using bins to represent the variations. Code for creating a histogram for energy distributions between the two electrons is shown below:



```python
# Visualize the energy distribution for each electron by creating histogram
plt.figure(figsize=(8,6))

plt.hist(data["E1"], bins = 60, color = "lightsteelblue", alpha = 0.8,  label = "Electron 1 Energy", edgecolor = "darkblue")
plt.hist(data["E2"], bins = 60, color = "orangered", alpha = 0.5, label = "Electron 2 Energy", edgecolor = "maroon")
plt.xlabel("Energy (GeV)")
plt.ylabel("Frequency")
plt.title("Histogram of E1 and E2")
plt.legend()

# Adding vertical lines for mean energy values
line_E1 = plt.axvline(mean_E1, color="blue", linestyle="dashed", linewidth=2, label=f"Mean E1: {mean_E1} GeV")
line_E2 = plt.axvline(mean_E2, color="red", linestyle="dashed", linewidth=2, label=f"Mean E2: {mean_E2} GeV")

# Adding text annotations for mean values with adjusted positions
plt.text(mean_E1, 0.95*max(plt.ylim()), f"Mean E1: {mean_E1} GeV", color="blue", fontsize=10, ha="center")
plt.text(mean_E2, 0.85*max(plt.ylim()), f"Mean E2: {mean_E2} GeV", color="red", fontsize=10, ha="center")

# Customizing grids
plt.grid(True, which="major", linestyle="-", linewidth="0.5", color="gray")
plt.minorticks_on() #turn on the minor ticks
plt.grid(True, which="minor", linestyle=":", linewidth="0.5", color="lightgray")

plt.savefig("Energy distribution.png", dpi=300)

plt.show()
```

Figure 8 – Creating a histogram for energy distribution of the two electrons simultaneously. Grids and 60 bins are included to ensure better insight into data distribution.

In the code above, we define labels for both x and y axis, as well as for the both energy values. Lines are added using axvline() function to mark the mean values of energy, as well as function txt() for adding line annotations. This format will be extensively used for most of the visualization methods in the analysis.

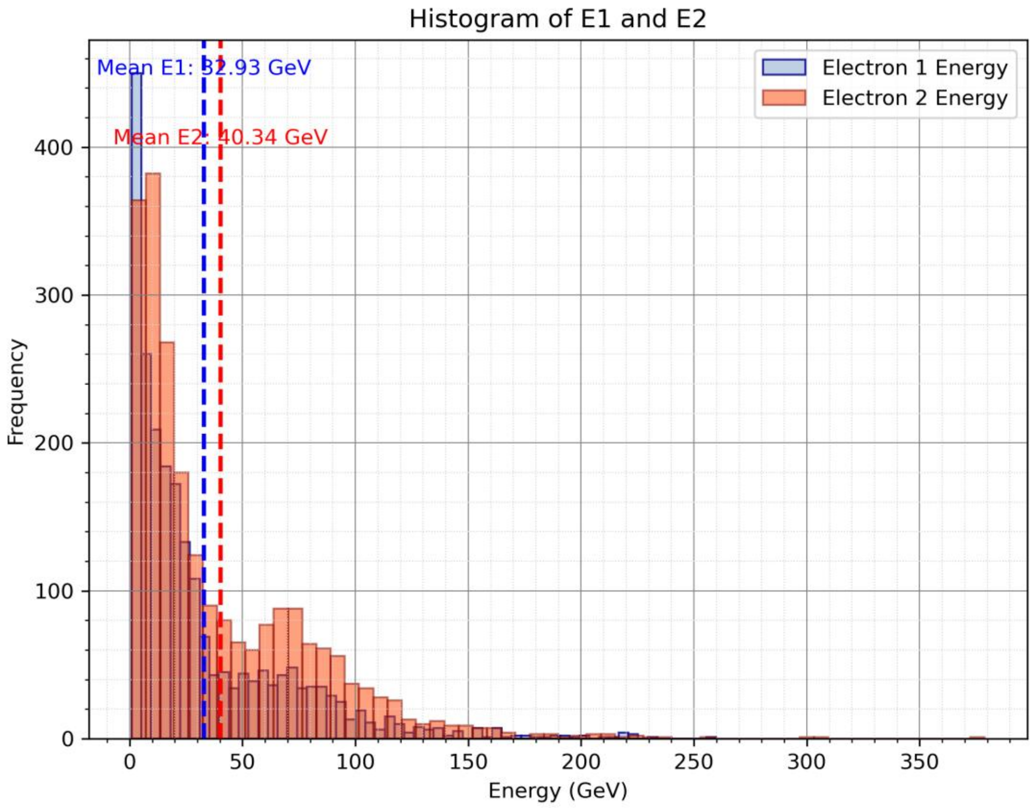


Figure 9 – Energy distribution histogram of both electrons.

It is obvious that majority of the particles lie on the lower spectrum of energy, with electron 2 having more frequently higher energy values. Much smaller number of electrons have energy higher than 150 GeV.

Mean energies for both electrons lie close to each other, indicating that their central tendencies are quite similar.

As for mass, both histograms show potential heavy particle parents, due to high invariant mass values, specifically at frequency lower than 5. Just like in the case of energy observation, most particles have lower invariant mass and are much more frequent as opposed to more rare events with invariant mass going over 80 GeV.

Seeing how both histograms give almost identical mass distribution, we can also confirm that computed mass values give highly accurate results.

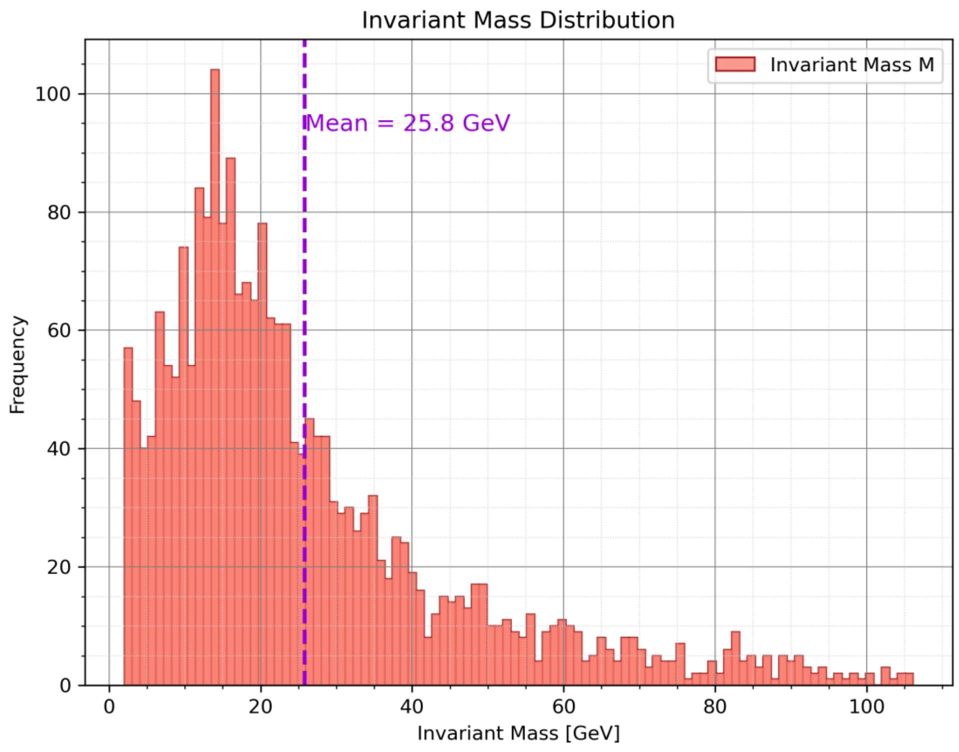


Figure 10 – Invariant mass histogram, with mean mass of 25.8 GeV

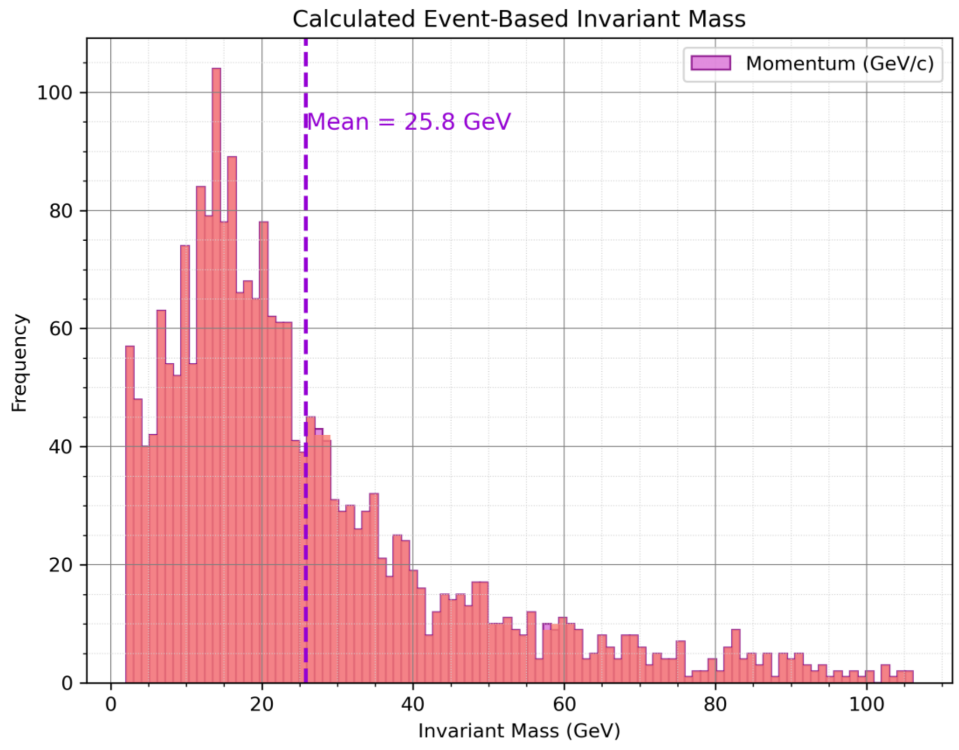


Figure 11 – Invariant mass calculated using energy and momentum relation.

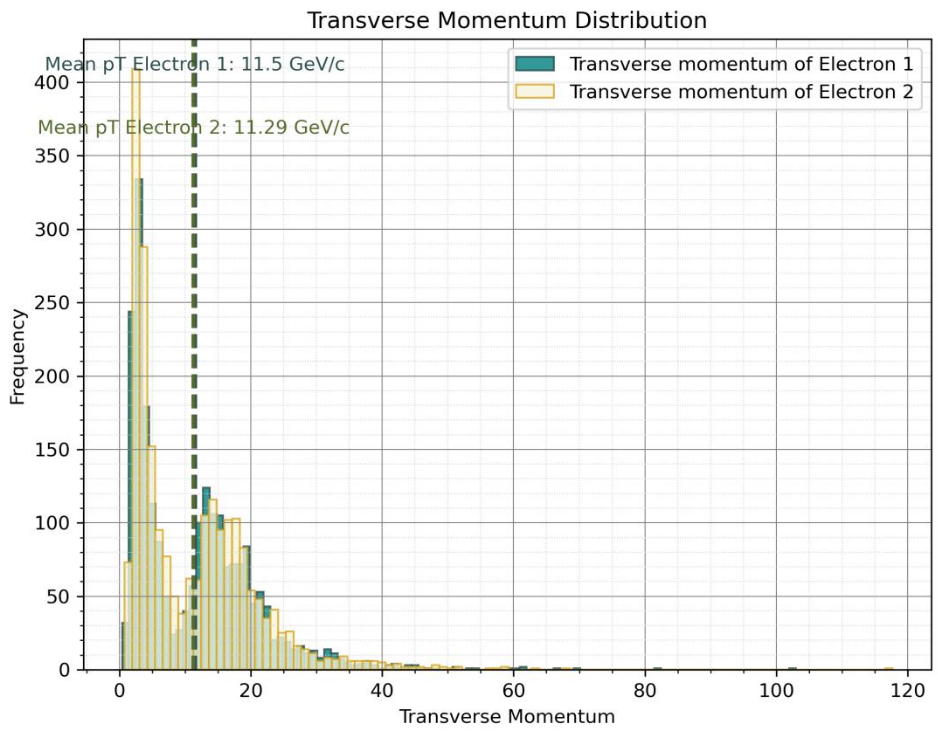


Figure 11 – Invariant mass calculated using energy and momentum relation.

We can see that the majority of particles fall in the range of transverse momentum from 0 to 10 which is common for soft interactions. Transverse momentum with values over 10 and up to 30 is an indication of some medium energy processes such as Z-boson decay. Less frequent values are those with momentum higher than 40 which are common for very heavy particles such as Higgs boson.

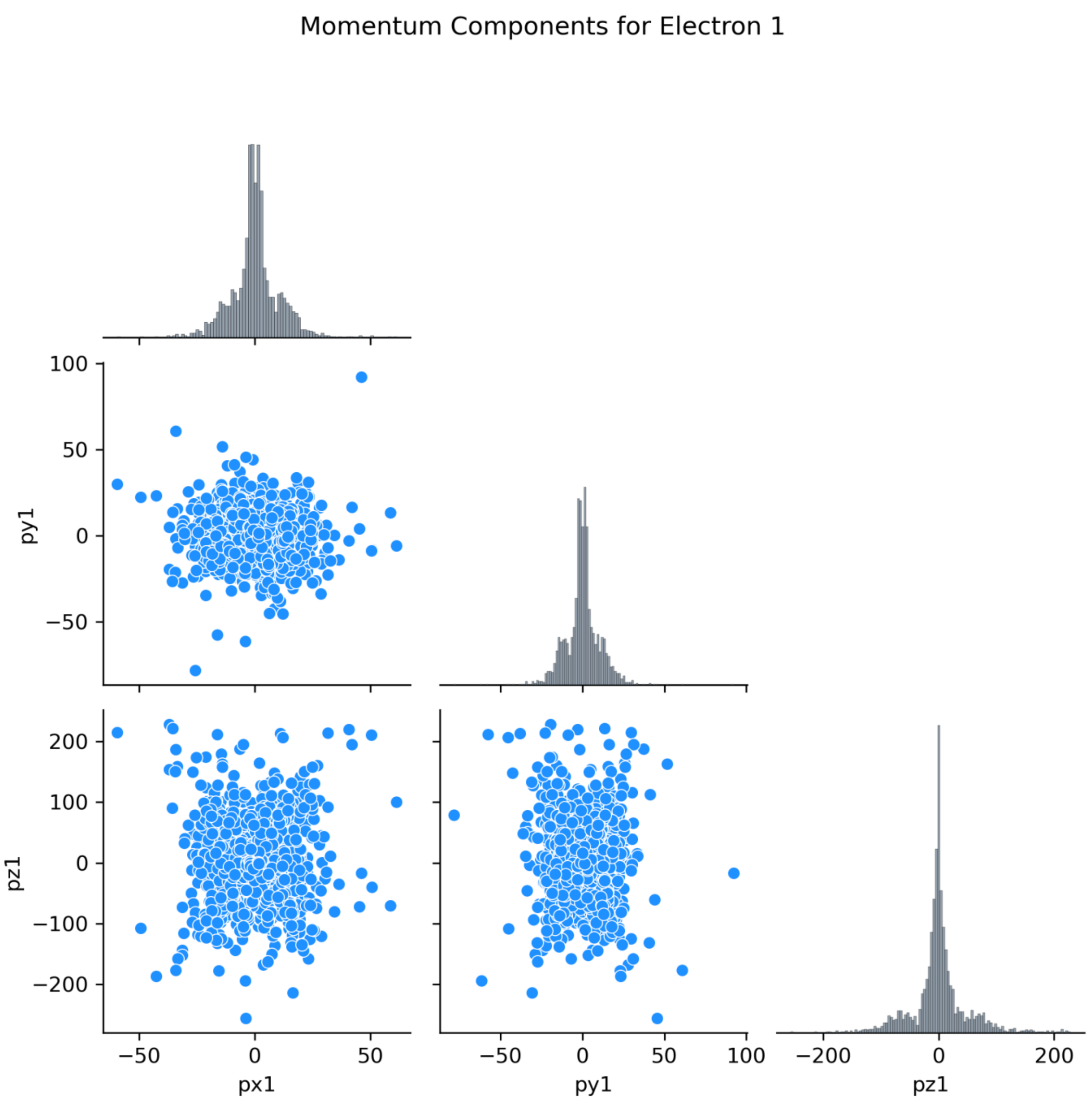


Figure 12 – X, Y and Z components of momentum for Electron 1 and their correlations.

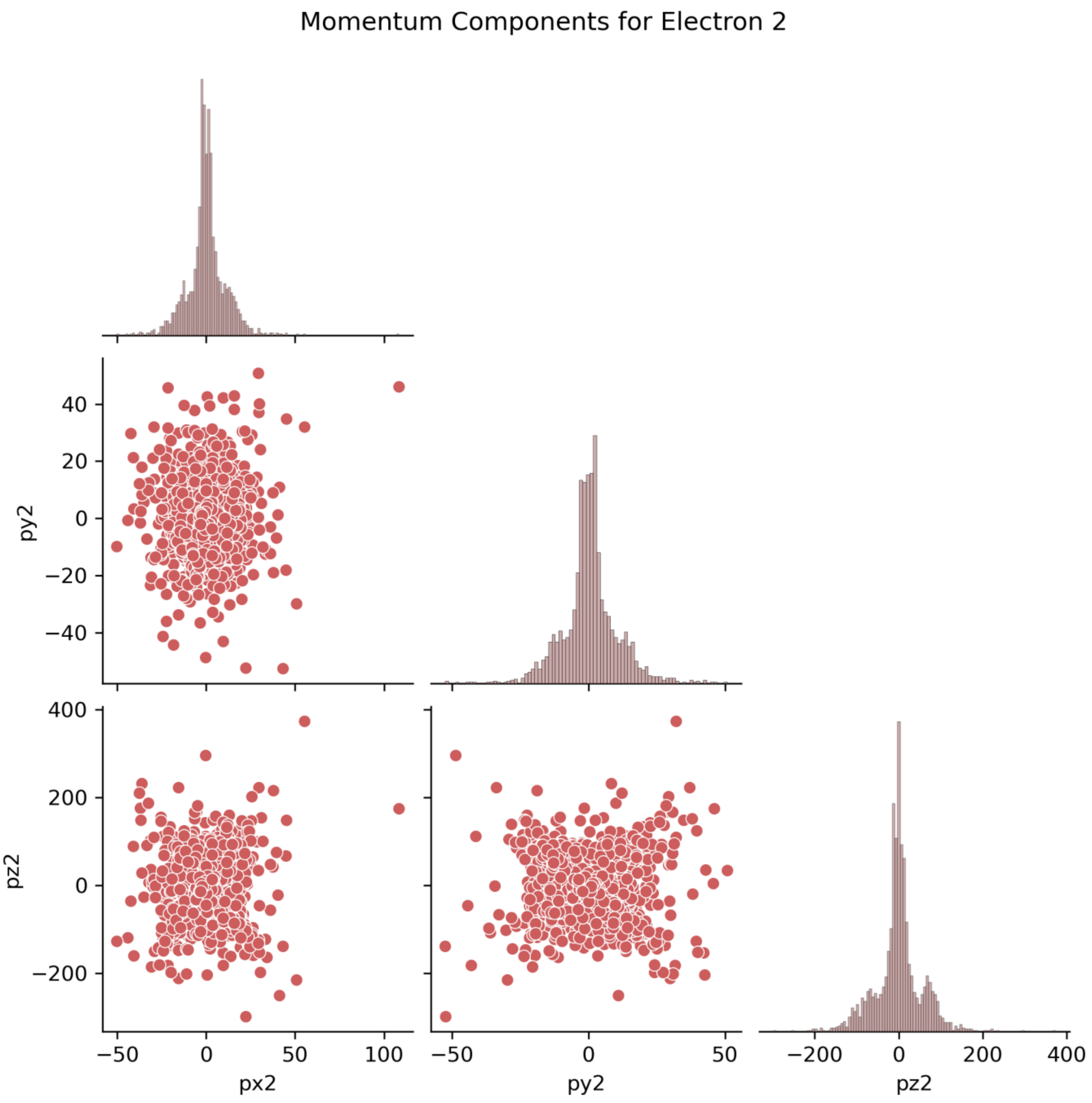


Figure 13 – X, Y and Z components of momentum for Electron 2 and their correlations.

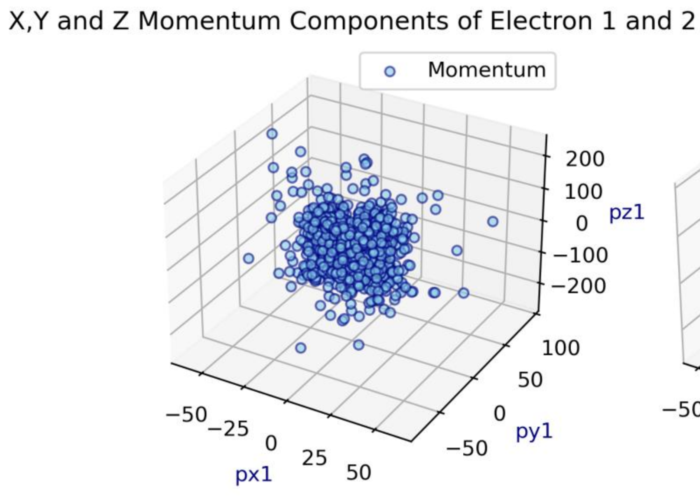


Figure 14 – 3D visualization of X, Y and Z components of momentum for both electrons.

All the three previous visualizations of momentum point to uniform distribution of momentum across the three directions.

In the LHC and CMS, the beams of particles collide along z-axis which means that initial momentum is primarily along the z-axis.

Because the collisions occur head-on (along the z-axis), there is no preferred direction in the transverse plane (the x-y plane). This leads to a roughly circular or isotropic distribution of momentum in the x-y plane. This is why the "spray" of momentum is generally uniform in all directions.

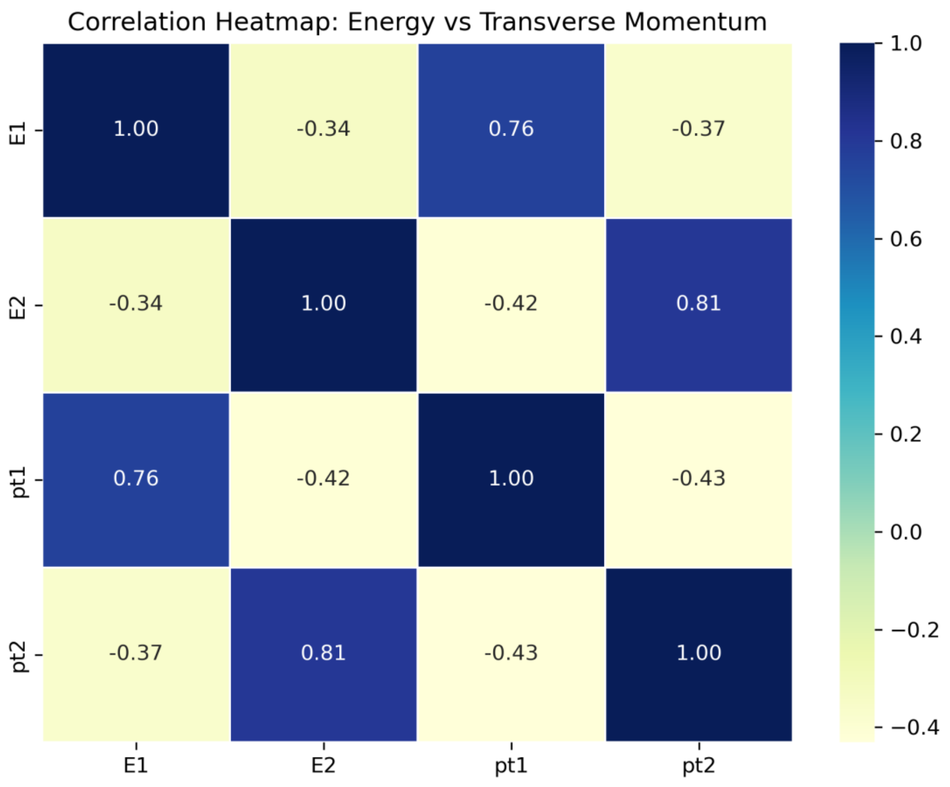


Figure 15 – Correlation heatmap which maps correlations between energy and momentum.

Highest correlations (closer to 1) are between energy and the corresponding transverse momentum of each electron (0.81). Strong correlation is also found between one electron's momentum and energy (0.76). This means that as energy of one electron increases, its transverse momentum will also increase.

The lowest correlations (negative) are between opposing components of the two electrons such as pt1 and pt2, which are of moderate negative trend.

This could be interpreted as: increase in pt1 means a decrease in pt2.

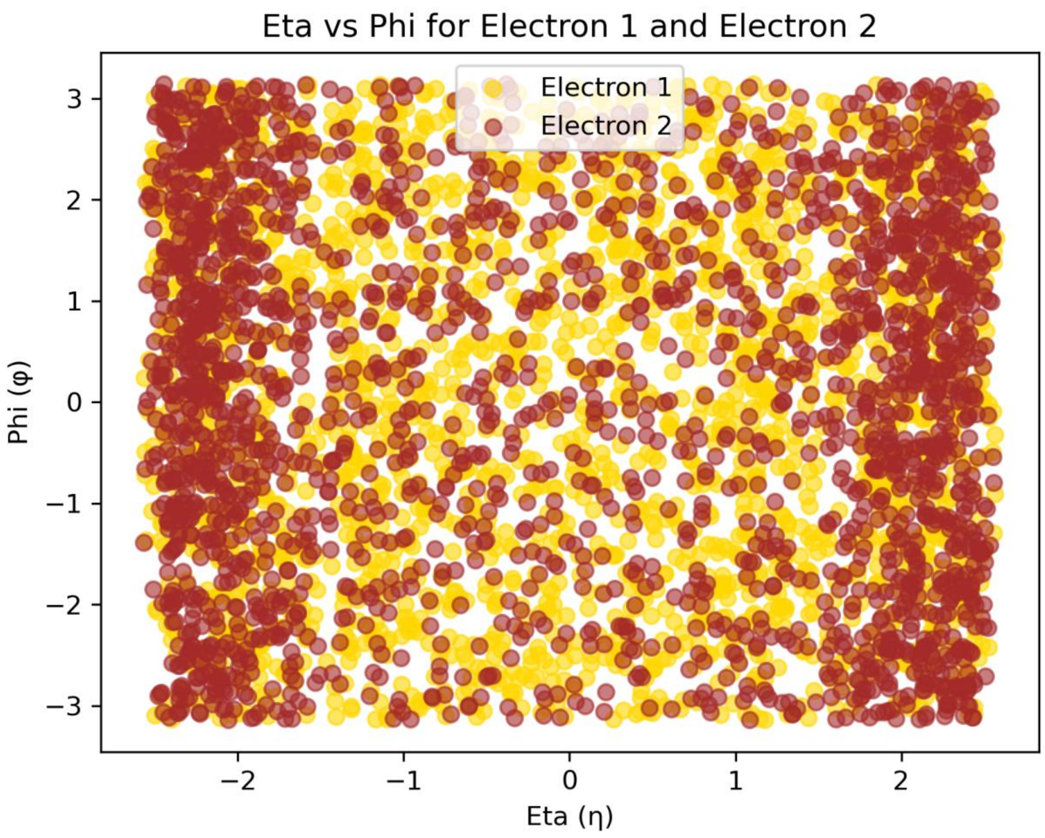


Figure 16 – Spatial distribution of particles in the detector.

Particle production is expected to be uniform in phi due to circular collider symmetry.

As for pseudorapidity (eta), clusters on the end-positive and end-negative sides tell us about the angle of the particles relative to the z-axis.

Higher values in eta result in emission of particles closer to the positive end of the beam axis. Negative is dominant for smaller values of eta.

Particles with smaller angles relative to the beam axis have larger pseudorapidity.

5. Conclusion and Further Work

Analyzing dielectron collisions in Python has been shown to be a straightforward and simple task. Thanks to Jupyter Notebooks which has been an invaluable tool for testing code directly, the workflow was very smooth and made the analysis all that simpler.

In this analysis, we investigated the properties of electron pairs that are product of intermediate decays of initial proton-proton collisions.We analyzed the invariant mass and transverse momentum distributions of the electrons, as well as the correlation between their transverse momentum. We also observed a broad invariant mass distribution with a peak at lower masses (around 10-20 GeV), suggesting a variety of production mechanisms.

The transverse momentum distributions showed the expected exponential decrease at high pt, characteristic of QCD (Quantum Chromodynamics) processes. Though we discovered several useful indicators that hard scattering processes did occur, no clear Z boson peak was observed, possibly due to low statistics, background contributions, or the suppressed decay channel. A moderate negative correlation was found between the transverse momentum of the two electrons, consistent with transverse momentum conservation.

Observing complex events such as electron collisions gave us great insight into particle dynamics, and how certain patterns align well with theoretical expectations. In this and many similar analysis, the variables of particular interest are energy, invariant mass and transverse momentum, as they often serve as best indicators of new particles or interesting decay processes. Applying standard descriptive statistics to these properties allowed us to track variations, trends and produce interesting predictions for further analysis.

This analysis is based on a limited dataset, and future work should focus on background estimation, refining event selection, and increasing the statistics to improve the sensitivity to potential signals and further investigate the high mass tail. Including detector effects and systematic uncertainties in the analysis would also improve the precision of the results.

6. Table of Pictures and Figures

7. References and Sources

CERN Electron Collision Data. (2020, December 25). Kaggle. https://www.kaggle.com/datasets/fedesoriano/cern-electron-collision-data?resource=download

Detector | CMS Experiment. (2024, December 11). https://cms.cern/detector

Facts and figures about the LHC | CERN. (2024, December 4). https://home.cern/resources/faqs/facts-and-figures-about-lhc

Wikipedia contributors. (2024, October 18). *Drell–Yan process*. Wikipedia. https://en.wikipedia.org/wiki/Drell%E2%80%93Yan_process

Keaten, J. (2024, October 1). *Mysteries of universe revealed? Hardly. But CERN still fascinates, discovers on its 70th anniversary | AP News*. AP News. https://apnews.com/article/cern-particle-accelerator-physics-geneva-anniversary-4554c83f661a53a8efa43bff3b62c253

Calculating the invariant mass — Physics. (n.d.). https://opendata-education.github.io/en_Physics/Exercises-with-open-data/Warming-up/Calculate-invariant-mass.html

Chatrchyan, S., Khachatryan, V., Sirunyan, A. M., Tumasyan, A., Adam, W., Bergauer, T., Dragicevic, M., Erö, J., Fabjan, C., Friedl, M., Frühwirth, R., Ghete, V. M., Hammer, J., Hoch, M., Hörmann, N., Hrubec, J., Jeitler, M., Kiesenhofer, W., Krammer, M., . . . Weinberg, M. (2012). Measurement of the rapidity and transverse momentum distributions ofZbosons inppcollisions at(s). *Physical Review. D. Particles, Fields, Gravitation, and Cosmology/Physical Review. D, Particles, Fields, Gravitation, and Cosmology*, *85*(3). https://doi.org/10.1103/physrevd.85.032002