# Tutorial summer 2022

## Exploring Particle Physics: Hands-on methods using LHC data

## Exercise 1

- ### Introduction

This guide here aims to provide you with detailed instructions for our tutorial. Section 2. provides the actual instructions to connect to the lyon machines for the daily work and to setup the environment for the tutorial, which you should follow in the given order. After some general information about the files needed for the data analysis, we will start in Section 3. with a few exercises using ROOT interactively. While this is useful to get familiar with the structure of the underlying data – or more general for something “fast and simple” – the ROOT browser and command line are not suitable for systematically processing large data sets. Therefore we’ll be moving on to using Python scripts rather quickly, in order to automate the work. When this is accomplished, we can dive deeper into the actual particle physics analysis.

- ### How to connect and get the analysis package

Step 2.0 needs to be repeated every time you want to work in this tutorial.

Step 2.1 you need to do only once!

#### 2.0 How to connect to the lyon

To connect to the lyon you will always need to follow the two steps

1. `ssh -Y your_user_name@cca.in2p3.fr`

#### 2.1 The analysis package

All files which are needed for this lab course are stored in the directory `/home/data/complete_set_of_ATLAS_open_data_samples_July_2021/`. These are the ATLAS data and simulated MC files in form of ROOT trees, as well as example Python code to analyse them. As in the real experiment, the data files are divided into so-called streams, depending on which triggers selected the event. For the MC files, the subdivision is based on the underlying physics process which was chosen for the event generation.

The ROOT files for the ATLAS data trees are stored on the web, but some are downloaded locally to inspect them and get familiar with the structure in the directory

`/home/data/complete_set_of_ATLAS_open_data_samples_July_2021/4lep/Data/`. The data_A.4lep.root data_B.4lep.root data_C.4lep.root data_D.4lep.root files contains 13 TeV data events selected, the labels A, B, C and D correspond to the data taking year (2015, 2016, 2017 and 2018).

In the `/home/data/complete_set_of_ATLAS_open_data_samples_July_2021/4lep//MC/` directory, various ROOT files for simulated MC processes are contained. You won't necessarily need all of them for the tutorial, it will depend on the analysis you decide to do in the non guided part of the tutorial. The ROOT files have self-explanatory names, in case of doubt please ask me.

Finally, the code you have copied in you personal folder `atlas-outreach-PyROOT-framework-13tev-master/` contains example Python code for the analysis event loop, plotting as well as fitting the final results. These files should be copied into the user's home area first day of the tutorial, and can be modified there. To copy the files and setup your environment please follow the instructions below (only once!):

1. **Copy the files to your home**

   Create a folder to store the code in your home (pay attention to the empty spaces):

```
cd
mkdir LHCDataLecture2022
cd LHCDataLecture2022
cp -r /home/data/complete_set_of_ATLAS_open_data_samples_July_2021/atlas-outreach-PyROOT-framework-13tev-master.zip .
```

2. **Setup your environment**

   Go back to your home, open your .profile (you can use any text editor you like) and add the following line at the end of the file (Please copy the dot and the empty space between the dot and /usr... ):

```
. /usr/local/root/bin/thisroot.sh
```

   If you do not have a prefered text editor, you can use emacs program that is installed in the lyon for you. The basic emacs commands can be found here. The following table show the instructions step by step:

| Command | What the command is doing |
|---|---|
| `cd` | Bring you to your home |
| `emacs -nw .profile` | Opens the .profile file, now you can now edit it. Please move with your keyboard arrows until the end of the file and copy the following line:<br>`. /usr/local/root/bin/thisroot.sh` |
| `Ctr+x Ctr+s` | Type this to save your changes in emacs |

| | |
|---|---|
| `Ctr+x Ctr+c` | Type this to close the emacs window |

3. **Setup your environment in a windows machine**

   If you will be using a machine running windows please use Putty in order to be able to see and display graphics. More details here : https://wwwpub.zih.tu-dresden.de/~baecker/teaching/cp2020/windows.html

## ● Exercise: Getting to Know the ROOT Basics

In this subsection you will be using ROOT interactively in order to make yourself familiar with the structure of the data files in the form of ROOT trees; and of course to get a first impression of ROOT itself. For this you will be using ROOT's `TBrowser` which allows to browse ROOT objects in graphical way.

ROOT can be started from the command line, issuing the command root in a terminal window. The result is the output of the ROOT banner for a few seconds and the prompt of Cling which is an interactive C++ interpreter. User input is typed just after the ROOT prompt "`root[n]`" where n is counting the accepted commands. In order to quit this interactive ROOT session type "`.q`".

Here below some exercises that will help you to get familiar with ROOT and our data:

1. Get familiar with the Open Data files provided for this course. In a terminal, open the `MC/mc_363490.llll.4lep.root` file with ROOT and use the `TBrowser` to look at its content:

   ```
   root
   /home/data/complete_set_of_ATLAS_open_data_samples_July_2021/4lep/MC/mc_363490.llll.4lep.root

   TBrowser t
   ```

   A short comment for each variable in the ROOT file can be found online at http://opendata.atlas.cern/release/2020/documentation/datasets/dataset13.html

2. Plot the `lep_eta` variable. Make yourself clear what is shown, and why the distributions looks as it does. What does the statistics box in the upper right corner tell you? What are the units of the axis? Label them correctly by right-clicking with the mouse at the proper elements. Save and print the plot using the TBrowser: File → Save (or Save As). png or pdf are the file types of choice

3. Do you remember the ATLAS coordinate system? How are this is related to the `lep_eta` plot in question 2? What are the structures on the distribution? Explain possible reasons.

4. Plot the `lep_n` variable. What is shown? Unfortunately this distribution has too coarse binning out of the box. We need to use the ROOT the command line to show n = 1 binning. Type

   ```
   root [] mini -> Draw("lep_n >> htemp(10,0,10)")
   ```

Then press enter and click with the middle mouse button on the canvas to refresh it.

5. How many leptons do you expect in the final state of a ZZ decay? Why do you see events here with more than 4 leptons? What is the dominant process for the 4-lepton final state?

6. Plot the `lep_pt` `(mini->Draw("lep_pt/1000 >> htemp(100,0,300)"))`, and `lep_phi` variables. What do you see? What does the statistics box in the upper right corner tell you now? Why are there more entries than if you plot runNumber? How many entries do you expect and why? What are the general features of the distributions? Why are there no gaps in the phi distribution? Why the $p_T$ spectrum has such a strange shape? Explain the concept of a trigger.

# Exploring Particle Physics: Hands-on methods using LHC data

# Exercise 2: Z boson analysis (part 1)

## 0. Disclaimer

This guide here aims to provide you with detailed instructions for our second tutorial, Z boson analysis. Instructions on how to connect to the lyon are given in the Exercise 1 (see above).

## ● Introduction. The Drell-Yan Process at the LHC

Particle colliders such as the Large Hadron Collider are, in a sense, very powerful microscopes. The higher the collision energy, the smaller distances can be studied. One of the most important processes that occurs in proton-proton collisions is the Drell-Yan process.

When a quark (e.g. a down quark d) from one proton collides with an antiquark (e.g. an antidown quark d) from the oncoming proton, they can annihilate into a virtual photon $\gamma^*$ or a Z boson if the net electric charge is zero. After briefly propagating, the $\gamma^*/Z$ can split into a lepton and its antiparticle partner, for example into an electron-positron pair or into a muon and antimuon. A Feynman diagram of the annihilation of a d quark and d antiquark into a muon-antimuon pair looks can be seen in Figure 1.


Figure 1: Drell-Yan diagram.

By conservation of momentum the sum of the muon and antimuon momenta will add up to the momentum of the γ∗/Z. This is nicely illustrated in figure 2 where the invariant mass distribution for any opposite-sign muon pair produced in proton collisions at the 7 TeV LHC is plotted, as measured by the CMS experiment.

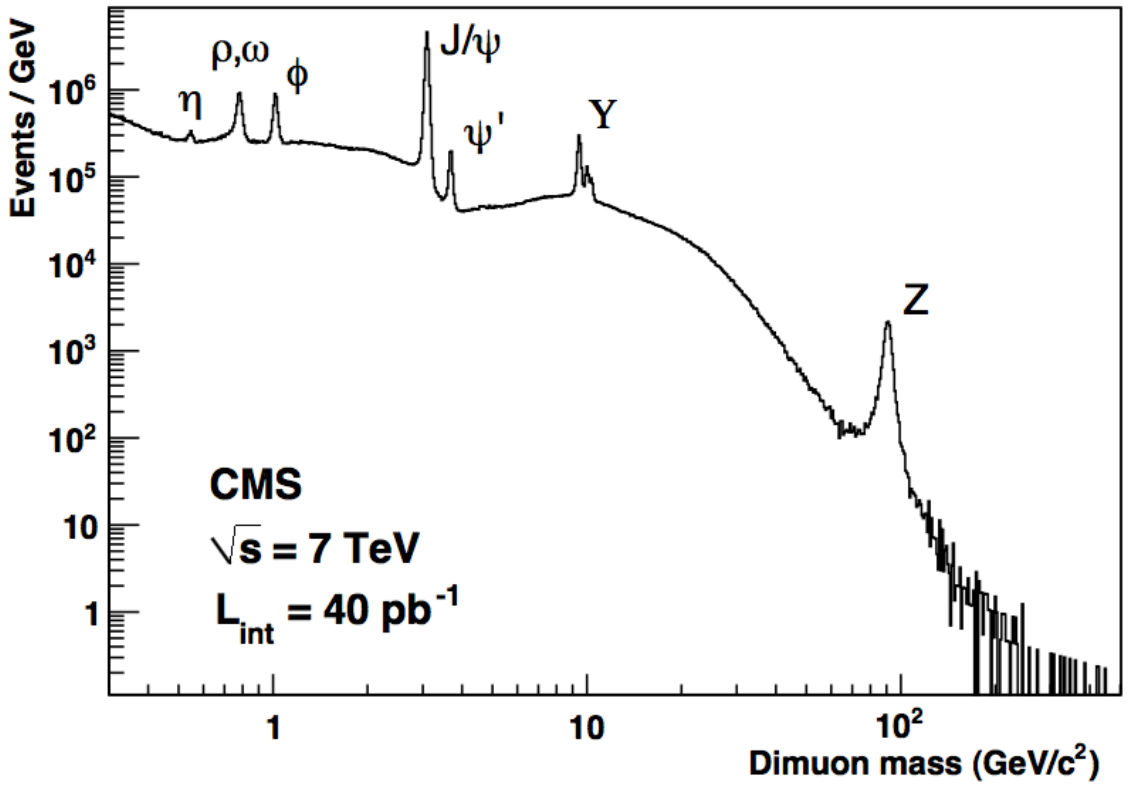


Figure 2: CMS invariant-mass spectra of opposite-sign muon pairs using 2010 data.

The rightmost peak at about 90 GeV corresponds to the production Z bosons. The other peaks represent the production of similarly well-known particles in the particle zoo that have decayed into a muon-antimuon pair. The clarity of each peak and the fact that this plot uses only about 0.2% of the total data collected during the first LHC data collection period (Run I) means that the Drell-Yan process is a very useful tool for calibrating the detectors. If the experiments are able to measure the Z boson, the ρ meson, etc. at their correct energies, then there is confidence that the detectors are working well enough to study nature at energies never before explored in a laboratory.

However, the Drell-Yan process is not as simple as drawn in figure 1. Real collisions include the remnants of the scattered protons. An example for a more realistic Drell-Yan interaction is depicted in figure 3, in which in addition to the proton remnants gluon radiation processes are included. This will lead to more particles in the final state, in particular jets of hadrons produced by the hadronization of quarks and gluons.

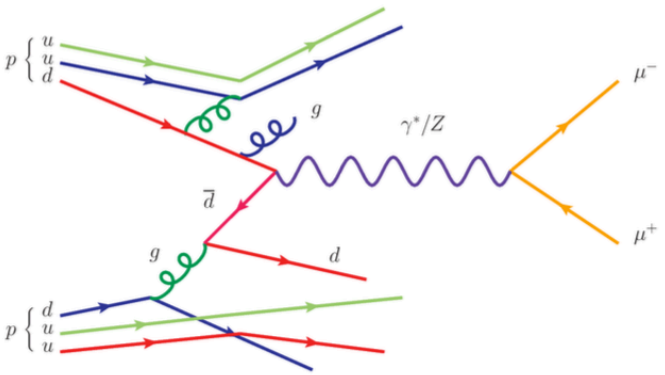


Figure 3: More realistic Drell-Yan diagram.

The focus of this exercise is to built a dilepton selection using electrons and muons, and to calculate the invariant mass of the dilepton system. We should corroborate if the Z mass peak appears and if it is located at the expected mass.

- **Get familiar with the analysis code**

Since this is our first analysis you need to get familiar with the code that you will be using

- Look at the provided Python scripts and make yourself familiar with its content. Describe what happens in the following scripts:
  1. `Configurations/Configuration.py`
  2. `Analysis/TupleReader.py`
  3. `Analysis/StandardHistograms.py`
  4. `Analysis/AnalysisHelpers.py`
  5. `Analysis/ZAnalysis.py`

  Important: Go in detail over ZAnalysis.py and make yourself familiar with its content. This is the code that you will be modifying and running in the following.

- Run the default script on the Z→ee and Z→μμ MC files and limit the number of events to a reasonable amount (hint: on the slides of the exercise 2. you will find the command lines to run the code)
- Explore the output: Your code will create a root file under `Output/xyz.root` where xyz is replaced by the name of your input file. It contains all histograms that have been defined in your ZAnalysis.py.
- The histograms you call in ZAnalysis.py are defined in StandardHistograms.py. Find the empty histograms and fill them in the ZAnalysis.py. Here there is one example of how to fill a histogram:

```
self.hist_leptn.Fill(len(GoodLeptons), weight)
```

**hist_leptn** is the name of the histogram. **Fill** the command to fill the histogram. **len(GoodLeptons)** in this case is the value you are filling in. **weight** is the normalization weight this is automatically set to 1 for data and can be different than 1 for simulation.

- Why do we fill MC histograms with a **weight** different than 1?

- **Built a Drell-Yan selection**

  Select pairs of electrons and muons of opposite sign.

  - Practise the handling the objects defined in TupleReader.py. For example the **GoodLeptons** we will be using in ZAnalysis.py are of the **Lepton** type. Check in TupleReader.py the possibilities you have and print the content of various variables, e.g. pt, eta, pdgID. Note that all energy and momentum variables are saved in MeV, but we want our final plots in GeV. (hint: Add your print after the line "# Event selection here" in ZAnalysis.py. Look in the s.12 from today for printing commands).
  - Compute the invariant mass of the two leading leptons in the event and plot it. Pay attention to the sign of $M^2_{ll}$ and restrict to a physical result. Hint: The leptons can be treated as massless. Recall the trigonometric relations between $p_x$, $p_y$, $p_z$ and $p_T$.
  - Now use ROOT's TLorentzVector class to calculate the invariant mass and compare with your results.
  - Fill a histogram with the invariant mass, run on the data_A.2lep.root file? How many peaks can you identify, and what do they belong to?
  - Now run your analysis over the Monte Carlo file mc_361106.Zee.2lep.root and compare the results for the invariant mass distribution.

# Exploring Particle Physics: Hands-on methods using LHC data

# Exercise 3: Z boson analysis (part 2)

- **Built a Drell-Yan selection**

  Since we still need to finish Exercise 2 we can focus on getting that working.
  Some hint on how to recover the information from the leptons:

```
# My Z candidates array
#print("Leading Lepton pt ",GoodLeptons[0].pt())
#print("Leading Lepton eta ",GoodLeptons[0].eta())
#print("Leading Lepton phi ",GoodLeptons[0].phi())
#print("Leading Lepton e ",GoodLeptons[0].e())
```

  Let's also discuss on how you have implemented the selection

- **Comparing Data and Monte Carlo Simulation**

  Monte Carlo simulations (MC) play an important role in particle physics analysis. Amongst others, they are used to prove that all detector effects are understood and that all physics processes contributing to the final state under study are correctly modeled and included.

  A simple but descriptive way of demonstrating the level of agreement between data and simulation is the comparison of the measured and simulated distributions in "stacked plots". Figure shows such plot for the missing transverse energy.

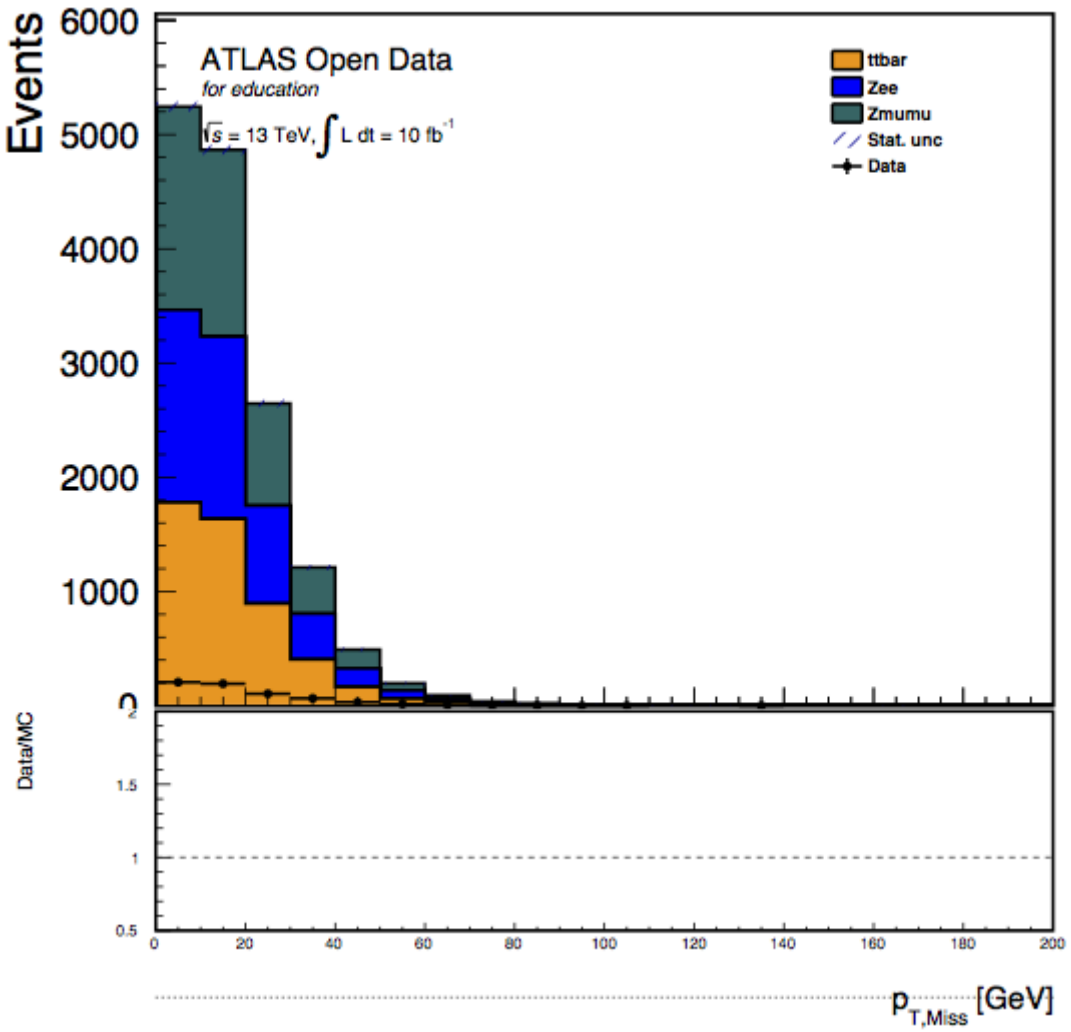


Figure 1. Missing transverse energy

To produce data and MC comparisons you need to:

1. Execute the following line:
   `python3 PlotResults.py Configurations/PlotConf_ZAnalysis.py`
2. First try this will fail so you need to modify the file `Configurations/PlotConf_ZAnalysis.py` and adapt it to the data and backgrounds you have

Next steps:

3. Produce a data/MC comparison for the Z mass
4. Figure out what is wrong with the data/mc comparison… this looks like a normalization issue!

# Exploring Particle Physics: Hands-on methods using LHC data

# Exercise 4: Z boson analysis (part 3)

### Fitting the invariant mass of the Z mass distribution

The selected event sample can be used to study the properties of the Z boson. Obviously, the invariant mass distribution provides a direct way for the determination of the Z boson mass and its decay width. For best results, the mass and width are extracted from a fit to the invariant mass spectrum, using the convolution of a Gauss and Breit-Wigner distribution, as shown in the Figure 2.

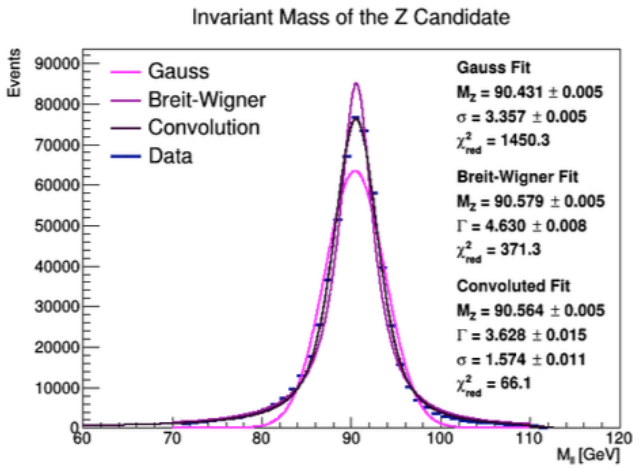


Figure 2: Fitted the invariant mass of the Z candidate with different formulas

**Gaussian:**

$$G(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

**Relativistic Breit-Wigner:**

$$B(m;M,\Gamma) = N\cdot\frac{2}{\pi}\cdot\frac{\Gamma^2M^2}{(m^2-M^2)^2+m^4(\Gamma^2/M^2)}$$

1. Why is the convolution of a Gauss and Breit-Wigner line shape expected to yield the best fit results? Which effects are taken into account by each of the two contributions?
2. Use and complete the provided script fit.C to extract the mass and width of the Z boson from the measured data sample. Compare your results with the PDG values (hint: Some examples and useful tips in the ROOT web, _Results, compare with others here)

/home/data/complete_set_of_ATLAS_open_data_samples_July_2021/

# Exploring Particle Physics: Hands-on methods using LHC data

# Exercise 5: Tag-and-probe

- **The tag-and-probe method**

The precise knowledge of the particle detection efficiencies plays a crucial role when for instance measuring cross sections or searching for new physics signatures (limit setting). Measuring the efficiencies requires a clean and unbiased sample of the particles under investigation. For electrons, the method of choice is the tag-and-probe method, which makes use of the characteristic signatures of $Z \rightarrow ee$ decays.

In this method, strict selection criteria are applied on one of the two decay electrons, called tag, and the second electron, the probe, is used for the efficiency measurements. Standard event selection criteria are applied to further reject background. The tag-and-probe pairs must also pass requirements on their reconstructed invariant mass.

The probe samples might be contaminated by background objects (for example, hadrons misidentified as electrons, electrons from semileptonic heavy flavour decays or from photon conversions). This contamination can in principle be determined using either background template shapes or combined fits. However, if the background contribution is reasonably small it can be neglected.

The tag-and-probe selection makes sure, that the probe electron is most likely a real electron and thus can be used to determine detection efficiencies. For this the number of electrons is independently estimated at the probe level and at the level where the probe electron candidate satisfies in addition the tested criteria. The efficiency ε is then defined as the fraction of probe electrons satisfying the tested criteria.

Usually the total efficiency to detect an electron in ATLAS is divided into different components, namely trigger, reconstruction and identification efficiencies, as well as the efficiency to satisfy additional analysis criteria, like isolation. The full efficiency $\varepsilon_{total}$ for a single electron can be written as :

$$\varepsilon_{total} = \varepsilon_{reconstruction} \times \varepsilon_{identification} \times \varepsilon_{trigger} \times \varepsilon_{additional} \times \varepsilon_{isolation}$$

## 1. Determining Efficiencies:

The precise knowledge of efficiencies plays an important role in many areas of physics data analysis. Hereby not necessarily the question stands in the foreground, if or when a selection or measurement is fully efficient, but also how precise the efficiency is known. If errors are small, inefficiencies can be corrected for in the measurement. Now we are going to determine the efficiency for the `tight` electron identification criteria using the tag-and-probe method.

1.1. Implement a tag-and-probe selection to determine the detection efficiency for `tight` electrons in data as function of $p_T$ and η. Some hints:
   - *The tag electron:* should be `tight` and pass all our usual `isGoodLepton` requirements (look in `AnalysisHelpers.py` for the `isGoodElectron` definition)
   - *The probe electron:* should be as loose as possible, disregard all other previously applied cuts in order to measure the efficiency of the `tight` identification (only keep a minimum $p_T$ cut of 5GeV).
   - You can compare your results with others compare with others here

1.2. Fill two histograms the numerator and denominator histogram separately in the event loop. The final efficiency can be derived by dividing both histograms. Hint: ROOT provides a TH1::Divide() method.

1.3. Compare the tight electron detection efficiencies in data and Monte Carlo simulation. Is the detector simulation able to describe the data correctly?

1.4. What type of error determination needs to be applied in order to correctly account for the statistical uncertainties in the efficiency determination? The TH1::Divide function does not know a priori how the given histograms are related and which uncertainties to assign. But it accepts options to specify this have a look at the TH1::Divide options compare with others here.

## 2. Effects of the selection criteria on the Z mass resolution:

The quality of the different objects selected in the analysis might influence the resolution of your final result.

2.1. Implemented for muons a very loose selection, i.e disregard all other previously applied cuts for `isGoodMuons` only keep a minimum $p_T$ cut of 5GeV.

2.2. Use now the two very loose leptons to built the dilepton pair (instead of the GoodLeptons) selection using those loose leptons. Do not apply the Z mass cut.

2.3. Compared with the original `isGoodLepton` selection results you had previously. Do you see a difference in the Z mass width and mass?

2.4. Plot the dilepton invariant mass distribution, how many resonance do you see now?

# Tutorial summer 2022

## Exploring Particle Physics: Hands-on methods using LHC data

## Exercise 6: Build your own measurement

### 0. Disclaimer

This is the first part of the non guided tutorial or less structured tutorial (I do not know how to call it, because of course you will have my guidance). The idea is to let you the freedom to choose and design your own analysis.

- **Define the physics you want to study**

Now is your time to allow your physicist imagination fly. Do you want to do a search or a measurement using the ATLAS data? Think about what will be interesting for you!

**1.1 Choose your analysis:** You would have to consider a few things before making your choice:

- Cross section of the process you are willing measure. How many signal events do you expect? This might require a bit of search for theory papers predicting cross sections.

**1.2 The dataset to use:** Is the data we have enough for a cross section measurement or observation? This is what we have available as data:

- The 13 TeV data:
  - We have been working until now with $10fb^{-1}$ of data at 13 TeV.
  - We have only data triggered by the electrons and/or muon available.
  - A limited amount of simulation please have a look on the Input root files

A limitation for this tutorial is the availability of the data and especially simulation for new physics signatures.

**1.3 The final state:** Think about the detector signature of the process you want to measure, the final state particles that will reach your detector.

**1.4 The possible backgrounds:** given the final state signature think about other possible physics process that could be background.

Have you chosen? Please add your choice and some of the answers to the previous questions **here**

- **Design and implement your selection**

  By now you are familiar with the use of cuts to select events of interest. Cuts are made that preferentially remove the unwanted processes (background) but leave the desired process (signal). It is useful to have a good understanding of the physics processes involved when applying cuts.

  A good way to improve your analysis selection is to look at figures of merit. For example calculate the significance (Number of signal events)/(√Number of background events) or simply the ratio of signal over background (Number of signal events)/(Number of background events). The larger the significance value is, the better job you have done extracting the signal.

  My advice:

  1. Implement a basic selection of the objects (leptons, missing transverse energy, jets depending on your selected analysis).
  2. Run over data and MC and check the composition of your final selection ("signal region"). Do MC describes well data, check the value of your figures of merit.
  3. Do you want to cut in complex variables, for example mass of the boson, transverse mass. Have a look before cutting, check the modelling.
  4. You will have to iterate several times..

  This may take more than this week tutorial to finish, so don't be frustrated if you do not have something clean and ready in one week.

- **Estimate the background**

  This is the next step, we will work on it once we all have a clear idea of the analysis we are going to develop.