$e^+e^- \rightarrow$ ZH:

Studies on the Higgs recoil mass

WorkFlow

24th May 2022

Ang LI

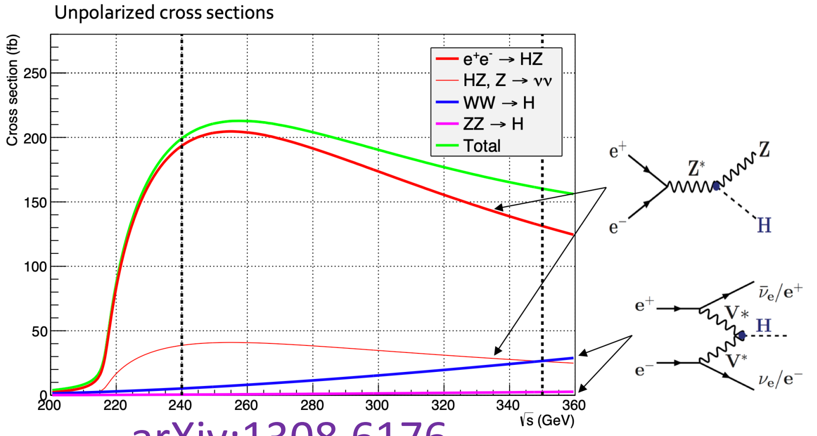
FUTURE CIRCULAR COLLIDER

Université de Paris

Plans

- **Short introduction of Higgs recoil mass**
- **Signal and background samples**
- **Software and Workflow**

Higgs recoil mass at Future Circular Collider (FCC)



arXiv:1308.6176

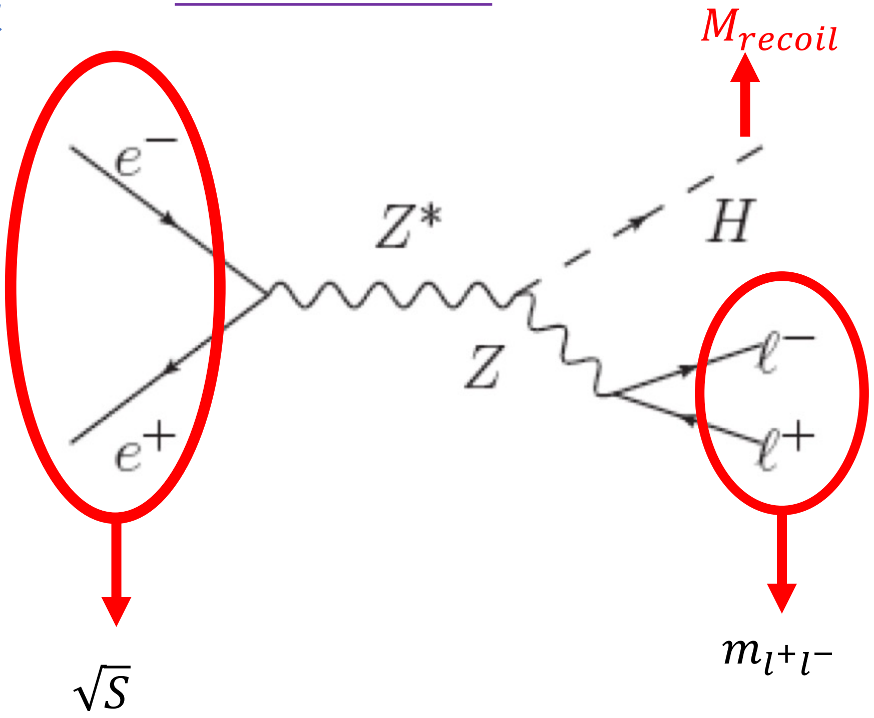
- Goal: Measurement of the ZH total cross section

- Signal: $e^+e^- \rightarrow ZH \rightarrow l\bar{l} + X$

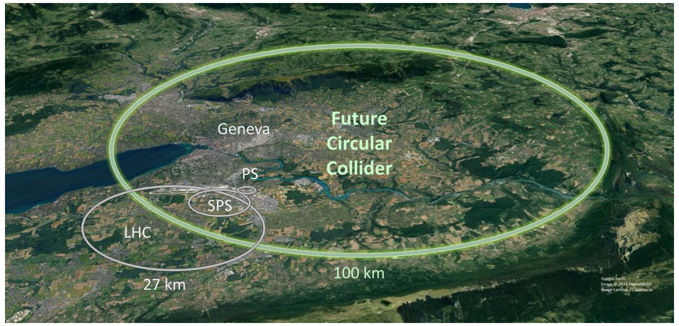
  ZH is the dominant Higgs production process @ 240 GeV e^+e^- machine

- Use events with a Z decaying leptonically, and reconstruct M_{recoil} from the Z production without measuring the Higgs production final state

$$M_{recoil}^2 = (\sqrt{s} - E_{l\bar{l}})^2 - p_{l\bar{l}}^2 = s - 2E_{l\bar{l}}\sqrt{s} + m_{l\bar{l}}^2$$



- The reconstructed M_{recoil} is sensitive to the precise knowledge of the centre-of-mass energy



- Physic independent study

- WW & ZZ Background @ 240 GeV

Signals, Backgrounds and Selections

- **Signals:**
  1. $Z(\mu^+\mu^-)H$ (Whizard)
  2. $Z(\tau^+\tau^-)H$ (Whizard)
  3. $Z(q\bar{q})H$ (Whizard)
  4. $Z(\nu\bar{\nu})H$ (Whizard)
  5. $Z(e^+e^-)H$ (Whizard)

  Or ZH inclusive (Pythia)

- $\sqrt{s} = 240$ **GeV**
- **ISR and FSR on**
- **Beam Energy Spread**
- **Luminosity:**
  $L = 5\ ab^{-1}$
- **IDEA detector**
- **Spring2021 samples**

- **Backgrounds:**
  1. ZZ(inclusive), (Pythia)
  2. $W^+(\nu\mu^+)W^-(\bar{\nu}\mu^-)$, (Pythia)
  3. $Z \to l^+l^-$, (Pythia)
  4. $Z \to q\bar{q}$, (Pythia)
  5. eeZ, (Whizard)
  6. $\gamma\gamma \to \mu^+\mu^-$, (Whizard)
  7. $\gamma\gamma \to \tau^+\tau^-$ (Whizard)

Pre-Selection:
1. At least one Z boson from a $\mu^+\mu^-$ pair
2. $m_{\mu^+\mu^-} \in [80, 100]$ GeV

	ZH(inclusive)	mumuH	WW_mumu	ZZ(inclusive)	Zll
$\sigma \cdot L$	1006580	33822	1289600	6794950	68893500
NEVENTS	10^7	10^6	10^7	10^7	$0.99 \cdot 10^7$
NEVENTS/$\sigma \cdot L$	9.93	29.57	7.75	1.47	0.14

FCC Spring 2021

The new data from the FCC group:

- One may need permission to get access to this repository
- Contact: [Emmanuel]() or [Clement]()

http://fcc-physics-events.web.cern.ch/fcc-physics-events/Delphesevents_spring2021_IDEA.php

- IDEA with Beam Energy Spread (BES)

NO	NAME	NEVENTS	NWEIGHTS	NFILES	NBAD	NEOS	SIZE (GB)	OUTPUT PATH	MAIN PROCESS
34	p8_ee_ZH_ecm240	10,000,000	0	100	0	100	97.44	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_ee_ZH_ecm240/	ZH ecm=240GeV
35	p8_ee_ZZ_ecm240	10,000,000	0	100	0	100	71.76	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_ee_ZZ_ecm240/	ZZ ecm=240GeV
36	p8_ee_WW_ecm240	10,000,000	0	100	0	100	69.21	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_ee_WW_ecm240/	WW ecm=240GeV

- IDEA without Beam Energy Spread (BES)

45	p8_noBES_ee_ZH_ecm240	10,000,000	0	100	0	100	97.07	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_noBES_ee_ZH_ecm240/	ZH ecm=240GeV, no BES
46	p8_noBES_ee_ZZ_ecm240	10,000,000	0	100	0	100	71.53	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_noBES_ee_ZZ_ecm240/	ZZ ecm=240GeV, no BES
47	p8_noBES_ee_WW_ecm240	10,000,000	0	100	0	100	68.89	/eos/experiment/fcc/ee/generation/DelphesEvents/spring2021/IDEA/p8_noBES_ee_WW_ecm240/	WW ecm=240GeV, no BES

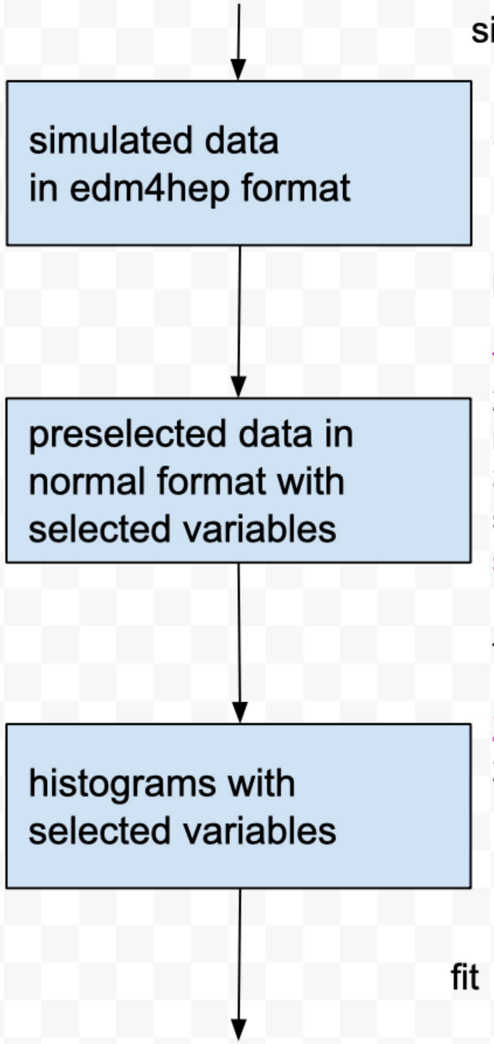
IDEA with Full Silicon tracker (called CLD in the studies of Ang and Greg)

http://fcc-physics-events.web.cern.ch/fcc-physics-events/Delphesevents_spring2021_IDEA_FullSilicon.php

IDEA with 3T magnet

http://fcc-physics-events.web.cern.ch/fcc-physics-events/Delphesevents_spring2021_IDEA_3T.php

FCCAnalyses Workflow



simulation: detector constrains (e.g. eta)

edm4hep format: e.g. <edm4hep> muons include its momentum and energy pid etc.

pre-selection :
1, constructions (m_recoil and m_Z), (e.g. Z boson in edm4hep format)
2, selection on pT. (opening angle is calculated on the m_Z reconstruction stage)
3, extract information (pT, M, E, eta) from edm4hep format and save in sample vectors (e.g. muons pt were saved as <vector> selected_muons_pt, selected_muons_y, Z_mass, Z_pt, etc.)

final-selection :
1, Selection on m_Z and N_Z (e.g. Z_mass.size() ==1 && Z_mass[0] > 80 && Z_mass[0]< 100)
2, Histograms are made. bin width axis range are fixed

FCCAnalyses Installation

My forked repositories are:
The software is available here:

https://github.com/AngLICERN/FCCAnalyses/#getting-started

APC Higgs tools (Branch ZH_recoil)

https://github.com/AngLICERN/FCCeePhysicsPerformance/tree/ZH_recoil/case-studies/higgs

Installation of FCCAnalyses:

1. git clone https://github.com/AngLICERN/FCCAnalyses.git
2. cd FCCAnalyses
3. source ./setup.sh
4. mkdir build install
5. cd build
6. cmake .. -DCMAKE_INSTALL_PREFIX=../install
7. make install
8. cd ../

FCCeePhysicsPerformance Workflow

Installation of APC Higgs tools:

1. git clone –b ZH_recoil https://github.com/AngLICERN/FCCeePhysicsPerformance.git
2. cd FCCeePhysicsPerformance/case-studies/higgs/
3. cd tools
4. source ./install.sh LOCALPATH
   LOCALPATH written as absolute like /afs/cern.ch/user/x/xyz/FCCsoft/FCCeePhysicsPerformance/case-studies/flavour/tools/localPythonTools
5. cd ../dataframe
6. source /cvmfs/fcc.cern.ch/sw/latest/setup.sh
7. source ./localSetup.sh
8. mkdir build install
9. cd build/
10. cmake .. -DCMAKE_INSTALL_PREFIX=../install -DFCCANALYSES_INCLUDE_PATH=<PATH_FCCANALYSES_INSTALL_INCLUDE_DIR>
    where PATH_FCCANALYSES_INSTALL_INCLUDE_DIR points to the install include dir of FCCAnalyes. For example /afs/cern.ch/user/h/helsens/FCCsoft/HEP-FCC/FCCAnalyses/install/include/FCCAnalyses.
11. make install
12. cd ..
13. source ./localSetup.sh

each time login

1. Go to FCCAnalysis repository
   source setup.sh
2. Go to FCCeePhysicsPerformance/case-studies/higgs/dataframe/
   source localSetup.sh
3. Go to work directory
   cd ../mH-recoil

Keep in mind our working directory is FCCeePhysicsPerformance/case-studies/higgs/mH-recoil

First stage

First stage study consists of two part:

1. Plot the basic variables
2. Produce the BDT model

We will do it the following way:

Part one:

Before running the scripts, you need to read the second python script of each step.
They are configuration files, which contain input, output path and batch information.
Pre-select the events:

python analysis/APC/FCCAnalysisRun.py FCCAnalyses-config/mumu/analysis_stage1_batch.py

1. Final selection:

   python analysis/APC/FCCAnalysisRun.py FCCAnalyses-config/mumu/analysis_stage1_final.py --final

2. Plotting:

   python analysis/APC/FCCAnalysisRun.py FCCAnalyses-config/mumu/analysis_stage1_plot.py --plots

Part two:

All scripts for BDT study use the configuration file “ FCCAnalyses-config/mumu/userConfig.py”
So please make sure you set this file properly.

1. Prepare the pickle files for BDT training:

   python process_sig_bkg_samples_for_xgb.py --Mode= "mumuH","ZZ","WWmumu","Zll","eeZ” (type one mode each time)

2. Train the BDT:

   python FCCAnalyses-config/mumu/train_xgb.py --Vars=normal

FCCAnalyses analysis.py

```python
def run(self):
  df2 = (self.df
        # define an alias for muon index collection
        .Alias("Muon0", "Muon#0.index")
        # define the muon collection
        .Define("muons",  "ReconstructedParticle::get(Muon0, ReconstructedParticles)")
        #select muons on pT
        .Define("selected_muons", "ReconstructedParticle::muon_quality_check(muons)")
        #select muons +
        .Define("selected_muons_plus", "ReconstructedParticle::sel_charge(1.0,false)(selected_muons)")
        #select muons -
        .Define("selected_muons_minus", "ReconstructedParticle::sel_charge(-1.0,false)(selected_muons)")
        #muons + numbers
        .Define("selected_muons_plus_n", "ReconstructedParticle::get_n(selected_muons_plus)")
        #muons - numbers
        .Define("selected_muons_minus_n", "ReconstructedParticle::get_n(selected_muons_minus)")
        #muons numbers
        .Define("selected_muons_n", "ReconstructedParticle::get_n(selected_muons)")
        # create branch with muon transverse momentum
        .Define("selected_muons_pt", "ReconstructedParticle::get_pt(selected_muons)")
        # create branch with muon rapidity
        .Define("selected_muons_y",  "ReconstructedParticle::get_y(selected_muons)")
        # create branch with muon total momentum
        .Define("selected_muons_p",     "ReconstructedParticle::get_p(selected_muons)")
        # create branch with muon energy
        .Define("selected_muons_e",     "ReconstructedParticle::get_e(selected_muons)")
        # find zed candidates from  di-muon resonances
        .Define("selected_muons_tlv",   "ReconstructedParticle::get_tlv(selected_muons)")
        #.Define("zed_leptonic",        "ResonanceBuilder(23, 91)(selected_muons)")
        .Define("zed_leptonic",         "ReconstructedParticle::resonanceZBuilder(91.1876)(selected_muons)")
        # write branch with zed mass
        .Define("zed_leptonic_m",       "ReconstructedParticle::get_mass(zed_leptonic)")
        # write branch with zed number
        .Define("zed_leptonic_n",       "ReconstructedParticle::get_n(zed_leptonic)")
        # write branch with zed charge
        .Define("zed_leptonic_charge",  "ReconstructedParticle::get_charge(zed_leptonic)")
        # write branch with zed transverse momenta
        .Define("zed_leptonic_pt",      "ReconstructedParticle::get_pt(zed_leptonic)")
        # calculate recoil of zed_leptonic
        .Define("zed_leptonic_recoil",  "ReconstructedParticle::recoilBuilder(240)(zed_leptonic)")
        # write branch with recoil mass
        .Define("zed_leptonic_recoil_m","ReconstructedParticle::get_mass(zed_leptonic_recoil)")
  )
```

- The first parameter is the name of the branch you want
- The second parameter is the name of the function
- One need to implement customized algorithm in
  analyzers/dataframe/ ReconstructedParticle.cc
  (e.g. muon_quality_check(muons) is programmed by me)
- There are already lots of very useful functions

```python
    # select branches for output file
    branchList = ROOT.vector('string')()
    for branchName in [
            "selected_muons_pt",
            "selected_muons_y",
            "selected_muons_p",
            "selected_muons_e",
            "selected_muons_n",
            "selected_muons_plus_n",
            "selected_muons_minus_n",
            "selected_muons_tlv",
            "zed_leptonic_pt",
            "zed_leptonic_m",
            "zed_leptonic_n",
            "zed_leptonic_charge",
            "zed_leptonic_recoil_m",
            #"missingET_px",
            #"missingET_py",
            #"missingET_pz",
            #"missingET_costheta",
            #"missingET_n"
            ]:
        branchList.push_back(branchName)
    df2.Snapshot("events", self.outname, branchList)
```

The branches you want to save in the output file for the next-step of the analysis

- The outputs (outputs/FCCee/higgs/mH-recoil/mumu)
  p8_ee_ZH_ecm240.root
  p8_ee_ZZ_ecm240.root
  P8_ee_WW_ecm240.root

FCCAnalyses finalSel.py

```
ROOT.gInterpreter.Declare("""
bool myFilter1(ROOT::VecOps::RVec<float> mass) {
  if (mass.size()<2) return false;
  for (size_t i = 0; i < mass.size(); ++i) {
  if (mass.at(i)<80. || mass.at(i)>100.)
    return false;
  }
  return true;
}
""")
```

- One can make their own selection this way

```
process_list=['p8_ee_ZZ_ecm240','p8_ee_WW_ecm240','p8_ee_ZH_ecm240']
```

List of the process

```
###Dictionnay of the list of cuts. The key is the name of the selection that will be added to the output file
cut_list = {"sel0":"return true;",
            "sel1":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 73 &&  zed_leptonic_m[0] < 120",
            "sel2":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 80 &&  zed_leptonic_m[0] < 110",
            "sel3":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 80 &&  zed_leptonic_m[0] < 100",
            "sel4":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 84 &&  zed_leptonic_m[0] < 98",
            "sel5":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96",
            "sel6":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20",
            "sel7":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 116 && zed_leptonic_recoil_m[0] <140 ",
            "sel8":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 120 && zed_leptonic_recoil_m[0] <140 ",
            "sel9":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 116 && zed_leptonic_recoil_m[0] <136 ",
            "sel10":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 80 && zed_leptonic_recoil_m[0] <110 ",
            "sel11":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 80 && zed_leptonic_recoil_m[0] <100 ",
            "sel12":"zed_leptonic_m.size() == 1 && zed_leptonic_m[0] > 86 &&  zed_leptonic_m[0] < 96 && zed_leptonic_pt[0] > 20 && zed_leptonic_recoil_m[0] > 86 && zed_leptonic_recoil_m[0] <96 "
            #"sel3":"myFilter1(zed_leptonic_m)"
        }
```

List of the cut, the program will run over all these selections independently

The way to use the customized selection

```
variables = {
    #"missingET_costheta":{"name":"missingET_costheta","title":"\theta_{missing}","bin":200,"xmin":-1,"xmax":1},
    #"missingET_n":{"name":"missingET_n","title":"missingET number","bin":12,"xmin":-1.5,"xmax":10.5},
    "Nmu_plus":{"name":"selected_muons_plus_n","title":"\mu^{+} number","bin":12,"xmin":-1.5,"xmax":10.5},
    "Nmu_minus":{"name":"selected_muons_minus_n","title":"\mu^{-} number","bin":12,"xmin":-1.5,"xmax":10.5},
    "Nmu":{"name":"selected_muons_n","title":"\mu number","bin":12,"xmin":-1.5,"xmax":10.5},
    "Cz":{"name":"zed_leptonic_charge","title":"Z charge","bin":23,"xmin":-11.5,"xmax":11.5},
    "Nz":{"name":"zed_leptonic_n","title":"Z number","bin":12,"xmin":-1.5,"xmax":10.5},
    "mz":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":250,"xmin":0,"xmax":250},
    "mz_zoom1":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":40,"xmin":80,"xmax":100},
    "mz_zoom2":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":470,"xmin":73,"xmax":120},
    "mz_zoom3":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":300,"xmin":80,"xmax":110},
    "mz_zoom4":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":200,"xmin":80,"xmax":100},
    "mz_zoom5":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":140,"xmin":84,"xmax":98},
    "mz_zoom6":{"name":"zed_leptonic_m","title":"m_{Z} [GeV]","bin":100,"xmin":86,"xmax":96},
    "leptonic_recoil_m":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":100,"xmin":0,"xmax":200},
    "leptonic_recoil_m_zoom1":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":240,"xmin":116,"xmax":140},
    "leptonic_recoil_m_zoom2":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":200,"xmin":120,"xmax":140},
    "leptonic_recoil_m_zoom3":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":200,"xmin":116,"xmax":136},
    "leptonic_recoil_m_zoom4":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":40,"xmin":124,"xmax":128},
    "leptonic_recoil_m_zoom5":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":120,"xmin":128,"xmax":140},
    "leptonic_recoil_m_zoom6":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":300,"xmin":80,"xmax":110},
    "leptonic_recoil_m_zoom7":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":200,"xmin":80,"xmax":100},
    "leptonic_recoil_m_zoom8":{"name":"zed_leptonic_recoil_m","title":"M_{recoil} [GeV]","bin":100,"xmin":86,"xmax":96},
    "muon_y":{"name":"selected_muons_y","title":"y_{#mu}","bin":100,"xmin":-4,"xmax":4},
    "muon_pT":{"name":"selected_muons_pt","title":"p_{T_{#mu}}","bin":200,"xmin":0,"xmax":200},
    "muon_E":{"name":"selected_muons_e","title":"E_{#mu}","bin":200,"xmin":0,"xmax":200}
}
```

- Here to define the name of the histogram you are going to save
- Choose the name of the varaible
- The number of bins
- Name, maximun and minimum of the X axixs

- The outputs (outputs/FCCee/higgs/mH-recoil/mumu)
  p8_ee_ZH_ecm240_sel0_histo.root
  p8_ee_ZZ_ecm240 _sel0_histo.root
  P8_ee_WW_ecm240 _sel0_histo.root

FCCAnalyses plot.py

```
# global parameters
intLumi        = 5.0e+06 #in pb-1
ana_tex        = 'e^{+}e^{-} #rightarrow ZH #rightarrow #mu^{+}#mu^{-} + X'
delphesVersion = '3.4.2'
energy         = 240.0
collider       = 'FCC-ee'
inputDir       = 'outputs/FCCee/higgs/mH-recoil/mumu/'
formats        = ['png','pdf']
yaxis          = ['lin','log']
stacksig       = ['stack','nostack']
outdir         = 'outputs/FCCee/higgs/mH-recoil/mumu/plots/'
```

- Parameters for plotting

```
variables = ["Nmu", "Nmu_plus", "Nmu_minus", 'Cz', 'Nz', 'mz','mz_zoom1','mz_zoom2', 'mz_zoom3', 'mz_zoom4', 'mz_zoom5', 'mz_zoom6', 'leptonic_recoil_m','leptonic_recoil_m_zoom1', 'leptonic_recoil_m_zoom2', 'leptonic_recoil_m_zoom3',
  'leptonic_recoil_m_zoom4', 'leptonic_recoil_m_zoom5', 'leptonic_recoil_m_zoom6', 'leptonic_recoil_m_zoom7', 'leptonic_recoil_m_zoom8', 'muon_y', 'muon_pT', 'muon_E']
```

The variables one wants to plot

```
selections['ZH']  = ["sel0", "sel1", "sel2", "sel3", "sel4", "sel5", "sel6", "sel7", "sel8", "sel9", "sel10", "sel11", "sel12"]
```

Selections one wants to plot

```
extralabel['sel0'] = "Selection: No Selection"
extralabel['sel1'] = "Sel.1"
extralabel['sel2'] = "Sel.2"
extralabel['sel3'] = "Sel.3"
extralabel['sel4'] = "Sel.4"
extralabel['sel5'] = "Sel.5"
extralabel['sel6'] = "Sel.6"
extralabel['sel7'] = "Sel.7"
extralabel['sel8'] = "Sel.8"
extralabel['sel9'] = "Sel.9"
extralabel['sel10'] = "Sel.10"
extralabel['sel11'] = "Sel.11"
extralabel['sel12'] = "Sel.12"
```

- Name of the selection
- The outputs are here
  outputs/FCCee/higgs/mH-recoil/mumu/plots

BackUp