Introduction to the Terascale

20 - 24 April 2015 Desy - Hamburg

DELPHES

A modular framework for fast simulation of a collider detector

Lisa Borgonovi - Università e INFN Bologna (Italy)

INFN sezione Bologna

Outline

- **Delphes**
  - Introduction
  - What is Delphes?
  - Why do we use Delphes?
  - Structure of the simulated detector
  - Object Reconstruction
  - High-level Corrections
  - Validation
  - Software Implementation
  - To summarise
- **Delphes for the upgrades**
- **$H \rightarrow ZZ \rightarrow 4l$ analysis with Delphes**
- **Exercise**

Introduction to Delphes

Developed by S. Ovyn, X. Rouby and V. Lemaître
Center for Particle Physics and Phenomenology (CP3)
Université Catholique de Louvain - Belgium, 2010

JHEP 02 (2014) 057 [arXiv:1307.6346 [hep-ex]]
Delphes web-page: https://cp3.irmp.ucl.ac.be/projects/delphes

Input

Output of event generators (MadGraph, Pythia, Herwig, ..) in various formats: ProMC, HepMC, LHEF, ..

Output

ROOT ntuple (generated and reconstructed objects)

What is Delphes?

C++ modular framework
FAST and **PARAMETRISED**
multipurpose detector response **simulation**

- Tracking system embedded in a magnetic field
- Electromagnetic and hadron calorimeters with their granularity
- Muon identification system
- Reconstruction of physics objects (charged leptons, photons, jets, missing energy, ..)

Why do we use Delphes? (1)

Very high energy particle collisions produce a large variety of final states → Multipurpose detectors are complex and sophisticated

To tune and optimise specific analysis:

FULL DETECTOR SIMULATION

GEANT package, dedicated routines, complex algorithms

↓

- High level expertise
- Large scale computing resources

↓

Time consuming!

Why do we use Delphes? (2)

- Production of large background samples
- Phenomenological studies (e.g. comparison between several different configurations of a detector)

Faster and **simpler** tool:

DELPHES

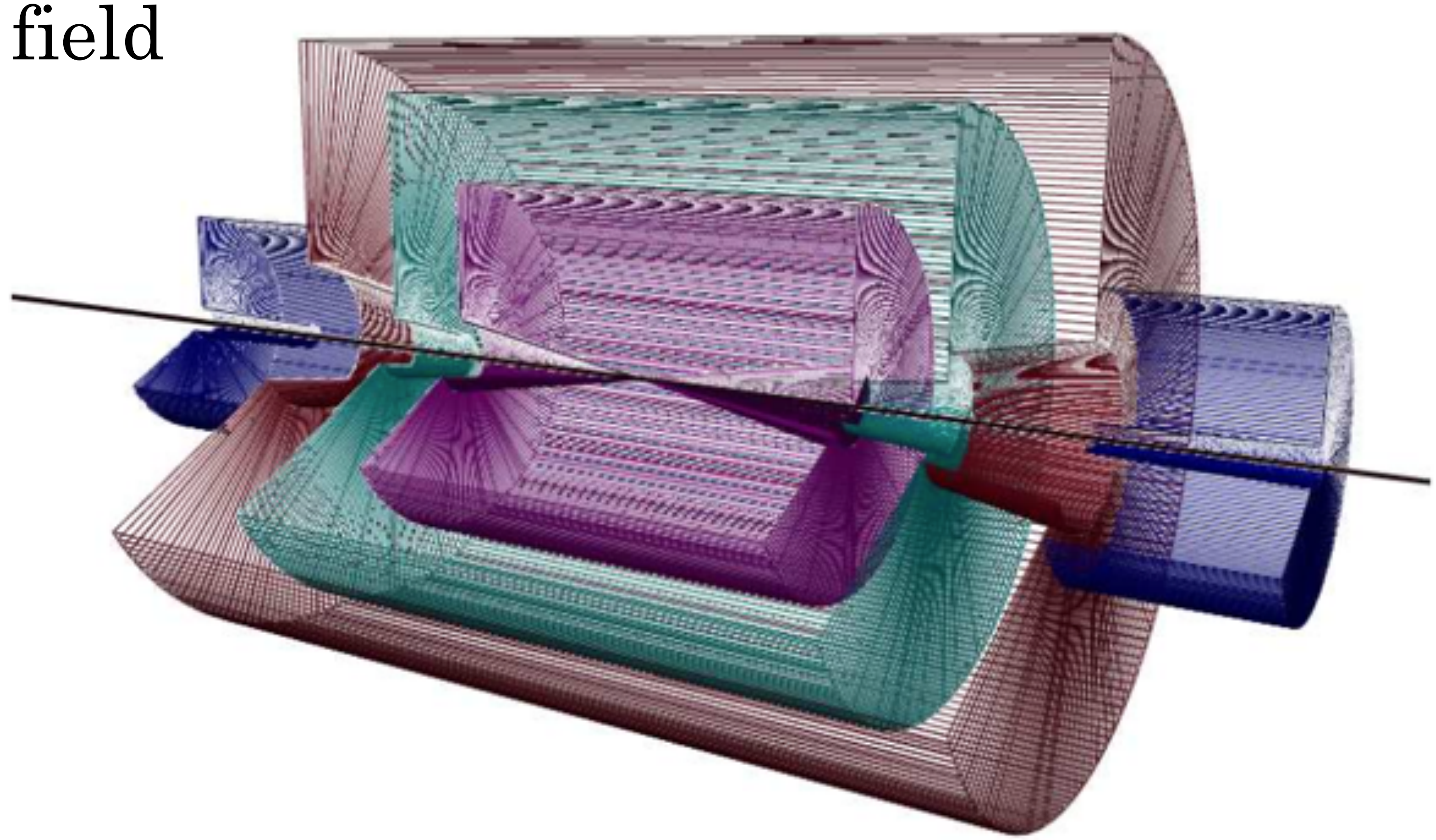
Limitations:

- Detector geometry is uniform, symmetric around the beam axis, ideal (without cracks or dead material)
- Secondary interactions, multiple scattering, photon conversion and Bremsstrahlung are not simulated

Structure of the simulated detector (1)

Delphes simulates the structure of a general purpose detector thanks to a parametrisation:

- Inner tracking system embedded in a magnetic field (TRACKER)
- Electromagnetic and hadron calorimeters (ECAL and HCAL)
- Muon system (MUON)
- Forward calorimeters (FCAL)



Structure of the simulated detector (2)

TRACKER:

Propagation of particles through the inner tracker volume in a uniform axial magnetic field along the beam axis

- only charged particles have a user-defined probability to be reconstructed as tracks in the tracker volume
- NO smearing is applied to the track parameters (except for the module of the transverse momentum)
- Particles that originate outside the tracker volume are neglected

Structure of the simulated detector (3)

ELECTROMAGNETIC AND HADRON CALORIMETERS:

Measurement of the particle energy

ECAL	**HCAL**
Measures the energy deposit of electrons and photons	Measures the energy deposit of charged and neutral hadrons
$f_{ECAL}(e, \gamma) = 1$	$f_{HCAL}(e, \gamma) = 0$
$f_{ECAL}(h) = 0$	$f_{HCAL}(h) = 1$
$f_{ECAL}(K, \Lambda) = 0.3$	$f_{HCAL}(K, \Lambda) = 0.7$
$f_{ECAL}(\mu, \nu) = 0$	$f_{HCAL}(\mu, \nu) = 0$

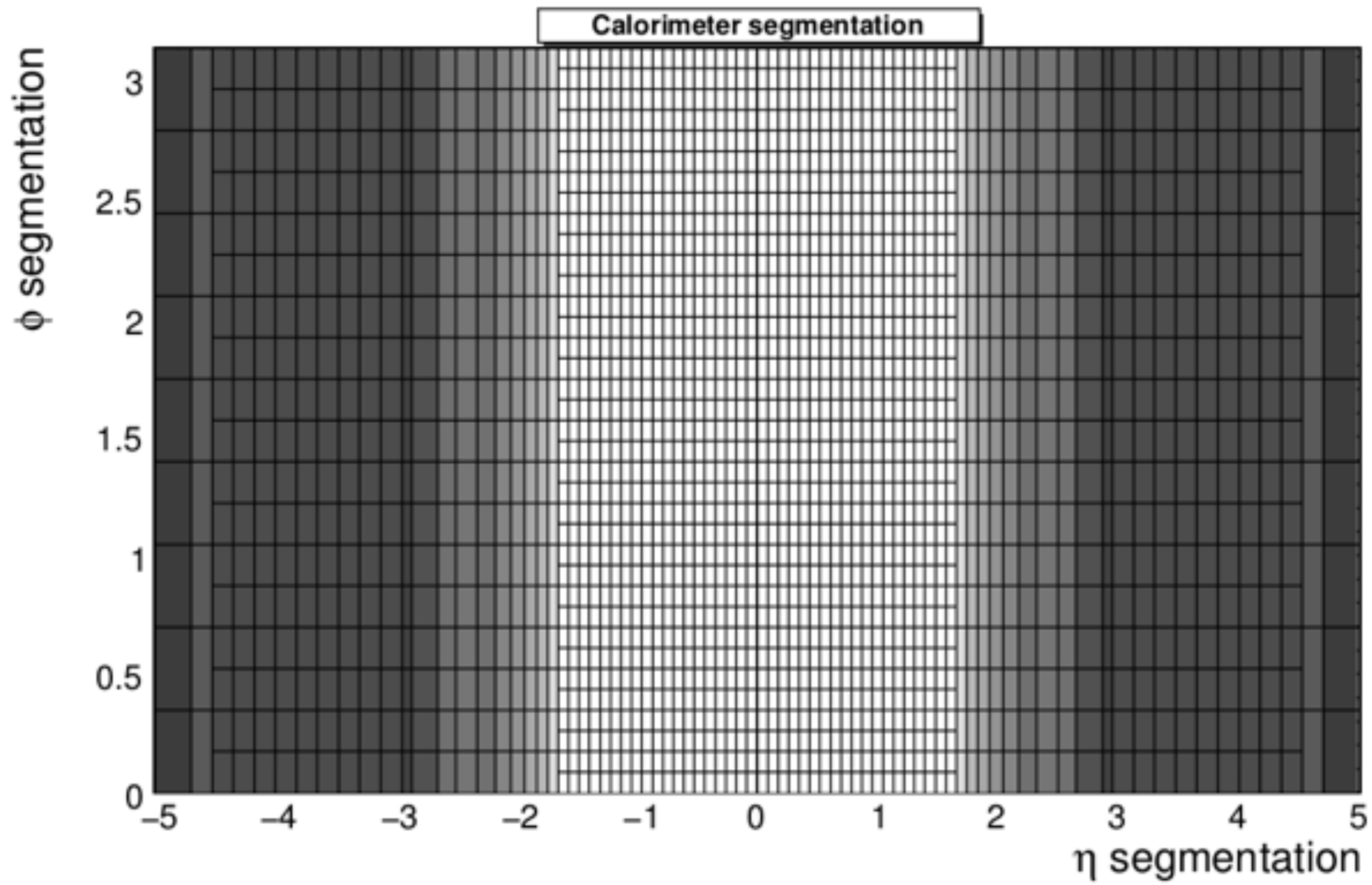
Default values, user can modify them

ELECTROMAGNETIC AND HADRON CALORIMETERS

Segmented in the (η,ϕ) plane in user-defined size cells

$$\eta \equiv -\ln\left[\tan\left(\frac{\theta}{2}\right)\right]$$

Same granularity for ECAL and HCAL



- Uniform segmentation in ϕ
- No longitudinal segmentation
- Response simulated through a Gaussian smearing of the cell energy

Structure of the simulated detector (5)

ENERGY FLOW

Based on Tracker and Calorimeter information with the assumption that the momentum resolution of the Tracker is better than the energy resolution of the Calorimeter

The algorithm counts E_{ECAL} , E_{HCAL} , $E_{ECAL,trck}$, $E_{HCAL,trck}$ and produces:

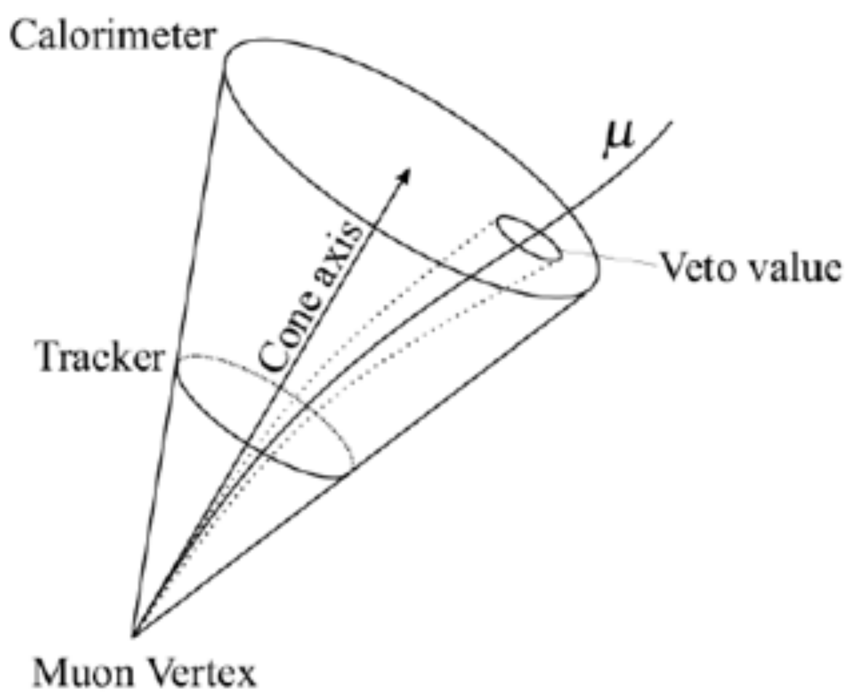
- EnergyFlow tracks from reconstructed tracks: EF_E_{Track}
- If $EF\_E_{Tower} > 0 \rightarrow EF\_E_{Tower} = \max(0, \Delta_{ECAL}) + \max(0, \Delta_{HCAL})$

Where $\Delta_{ECAL} = E_{ECAL} - E_{ECAL,trck}$ and $\Delta_{HCAL} = E_{HCAL} - E_{HCAL,\mathit{trck}}$

Object reconstruction (1)

Muons, electrons, photons

- reconstructed with a user-defined efficiency f(η, p_T) (passing cuts: e.g. tracker acceptance, p_T threshold, ..)
- reconstructed momentum/energy: Gaussian smearing of the generated momentum/energy → the resolution is user-defined f(η, p_T)



Isolation

$$I(P) = \frac{\sum_{i \neq P} p_T(i)}{p_T(P)}$$

e, μ isolated if $I < I_{min}$ within a small cone of radius ΔR in the (η,ϕ) plane

Object reconstruction (2)

Jets

Final states dominated by Jets: need an accurate reconstruction

It is possible to produce Jets starting from different input collections:

- Generated Jets: clustered from generator-level particles after parton shower and hadronization
- Calorimeter Jets: use calorimeter towers
- Energy-flow Jets: result of clustering energy-flow tracks and towers

Many clustering algorithms included in the FastJet package (Longitudinally invariant k_t jet, Cambridge/Aachen jet, Anti k_t jet, CDF MidPoint, ...)

Object reconstruction (3)

b-jets and τ-jets

- A jet is a potential b/τ jet if a generated b/τ is found within some distance along the jet axis:

$$\Delta R = \sqrt{(\eta^{jet} - \eta^{b,\tau})^2 + (\phi^{jet} - \phi^{b,\tau})^2}$$

- User-defined efficiency to identify that candidate as a real b/τ-jet

Mis-tagging efficiency can be specified!

High-level corrections

Resulting collections not yet ready for the analysis:

Jet Energy scale correction

- Only for Jets (not for non-composite objects)
- Discrepancy between average momenta of reconstructed objects and generator-level objects
- $f(\eta_{RECO}$ jet, $p_{T\ RECO}$ jet)

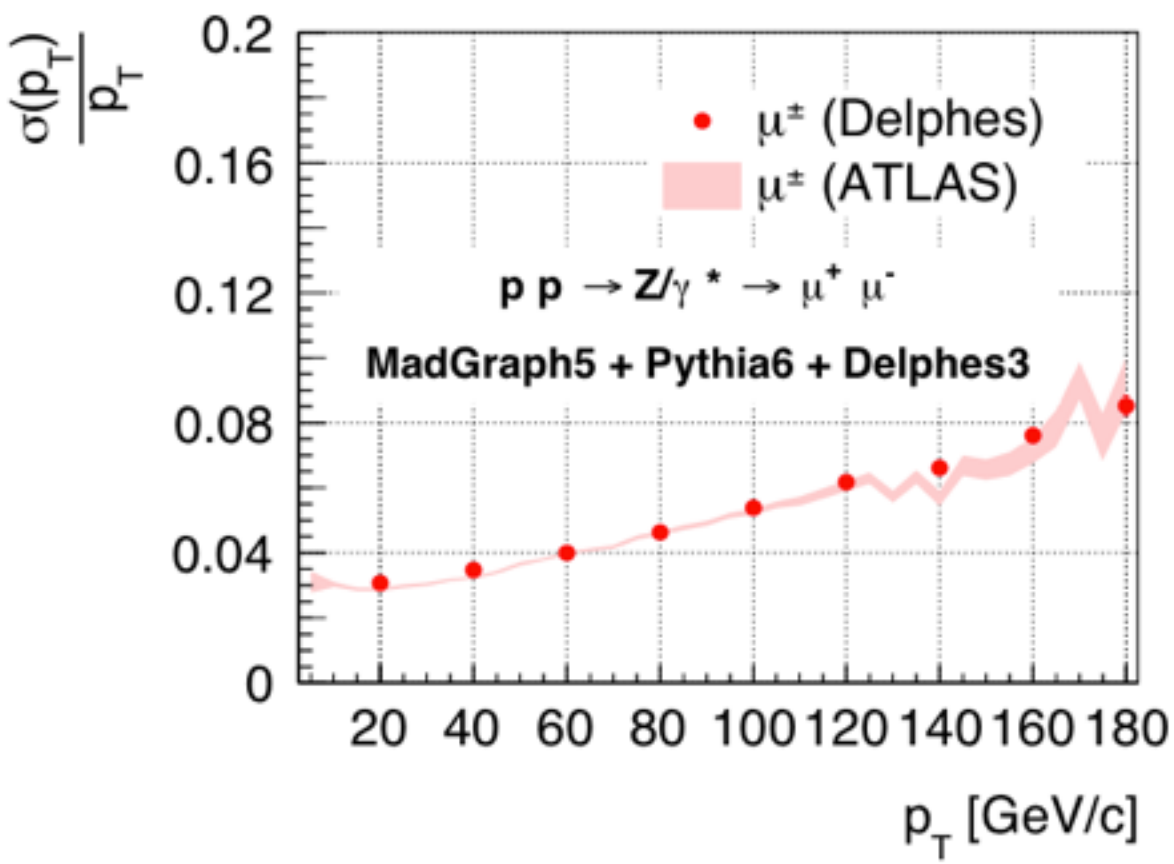
PileUp Subtraction

- PU randomly extracted from a file containing Minimum Bias interactions
- Only for Jets and Isolation using vertex reconstruction (not for MET because too difficult to account for neutral contamination)

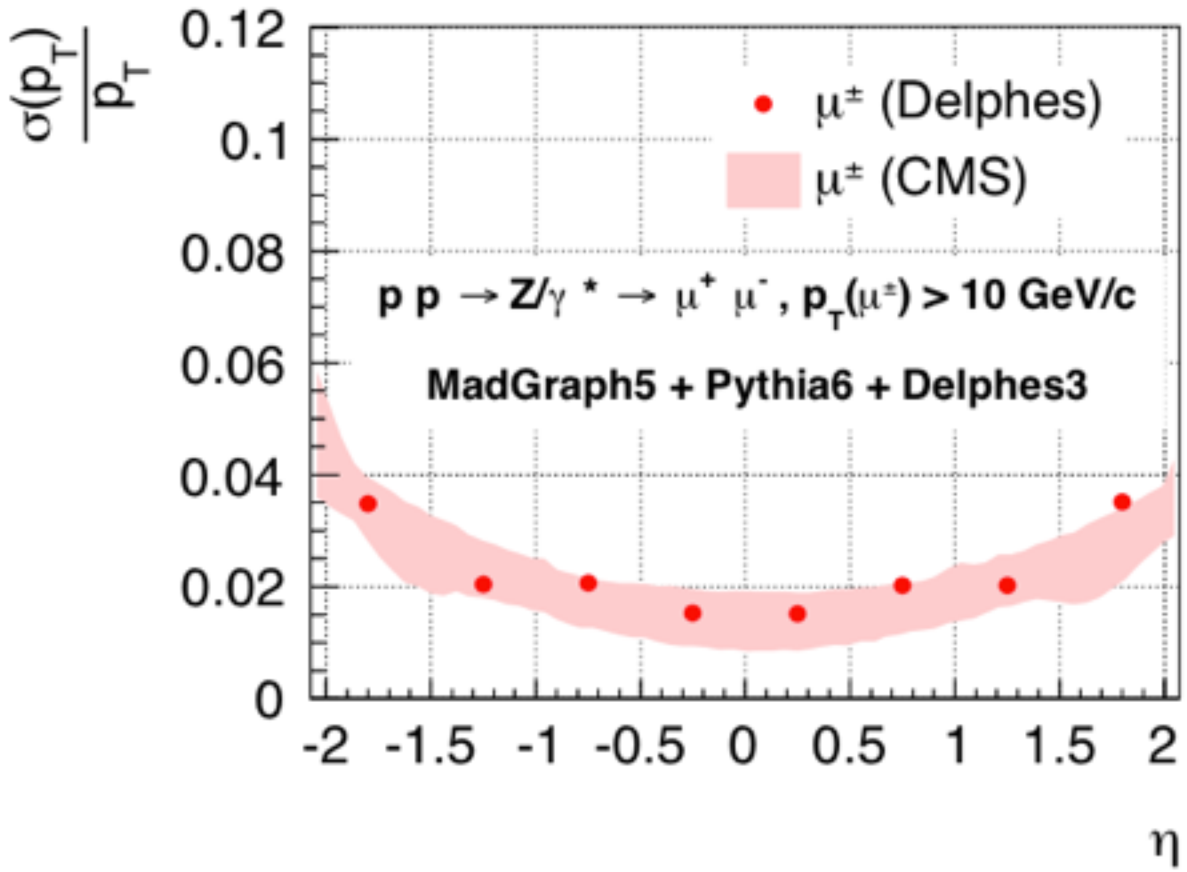
Validation

The simulation and reconstruction process executed by Delphes has to be validated with real experiment data:

ATLAS

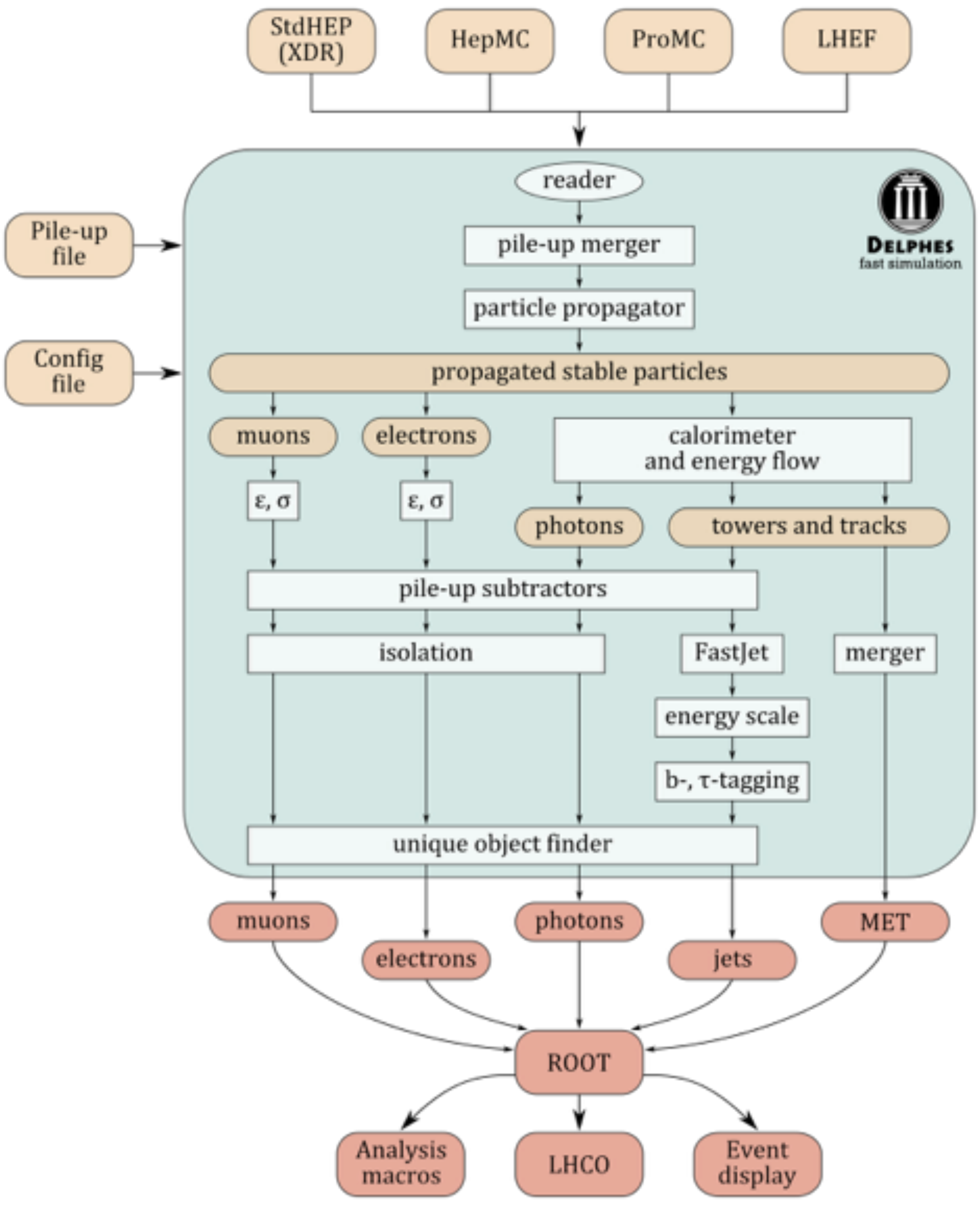


CMS



Software implementation

- input files decoded by a reader module
- pile-up events extracted from low-QCD interactions file and overlaid to the hard scattering event
- stable particles propagated from the tracker to the calorimeters, within the magnetic field
- objects reconstruction: different modules (isolation is computed)
- duplicates of reconstructed objects are removed
- resulting collections of global event quantities and physics objects stored in a ROOT Tree file format, together with the initial Monte Carlo generated object collections.

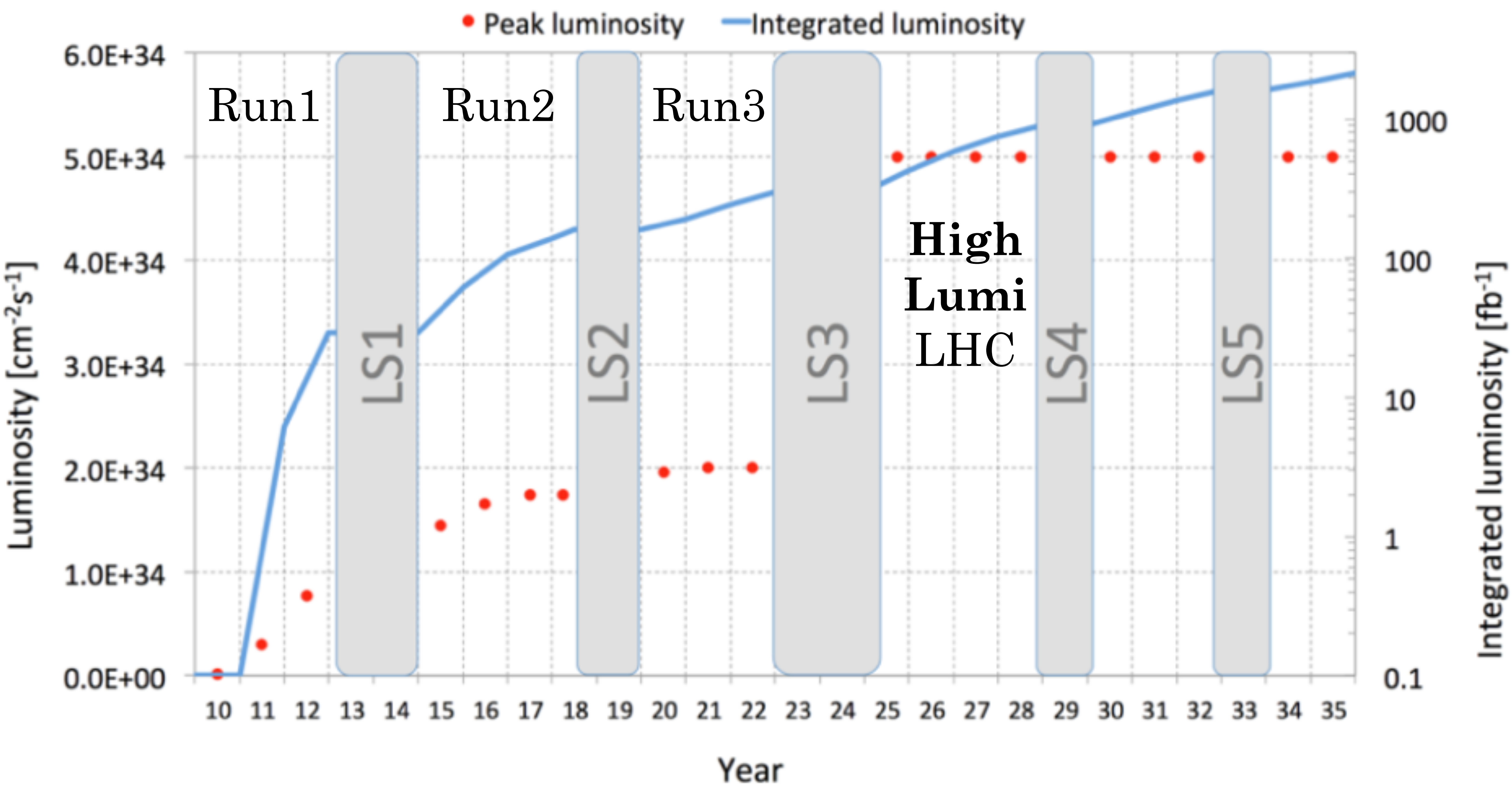


To summarise

- Delphes is a modular framework that performs a **fast** and **parametrised simulation** of a multipurpose detector (e.g. CMS and ATLAS)
- This tool has been completely **validated** (with CMS and ATLAS real data)
- Useful in **phenomenological studies** in which a fast simulation is needed

Delphes for the upgrade (1)

LHC future perfomances:



Delphes for the upgrade (2)

CMS/ATLAS → Substantial **upgrades** to cope with these challenging conditions

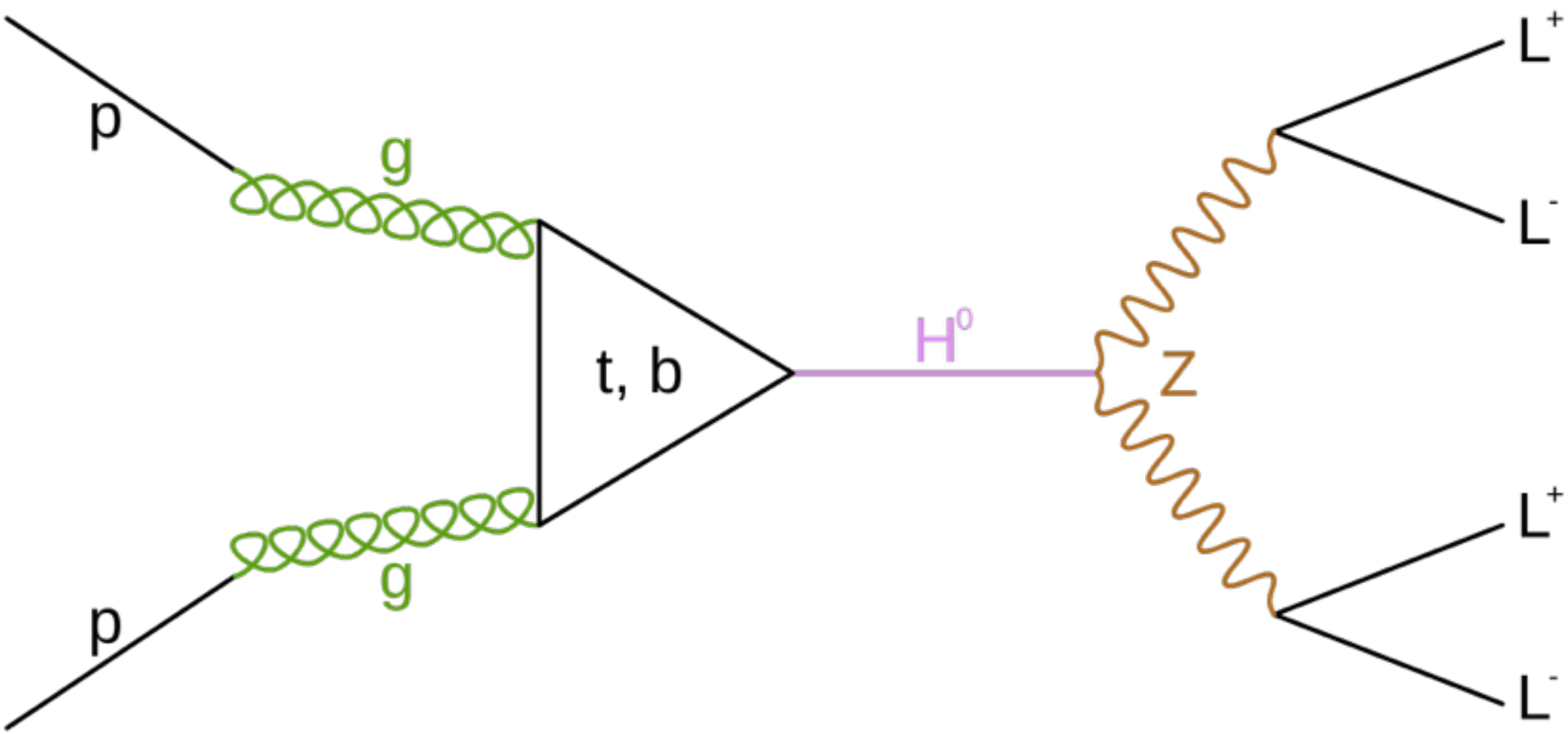
- Studies and comparisons of different detector configurations
- NO full simulation samples available
- production of large background samples to compute signal selection efficiency and fake rate

DELPHES!

$H \rightarrow ZZ \rightarrow 4l$ analysis with Delphes (1)

Higgs production mechanism: gluon-gluon fusion

Higgs decay mode: $ZZ \rightarrow 4l$ ($4e/4\mu$)



Exercise: goals

Goal of this exercise:

Basic knowledge about:

- the Delphes structure
- how Delphes works
- how to use Delphes to produce root samples of given events
- the ROOT samples structure
- how to analyse the samples with ROOT

Exercise: how to run Delphes

Getting started with Delphes

Setup GCC and ROOT

Download Delphes

Compile the code

You need one or more file(s) with Generated Events (in .hepMC format or other)

You need a file with Minimum Bias Events for the PU (MinBias.pileup)

Run Delphes:

./DelphesHepMC cards/delphescard.tcl path/outputfile.root path/hepmcfile.hepmc

you need root to analyse the .root file produced

Exercise: hepMC file

File with Generated Events .hepMC (or other)

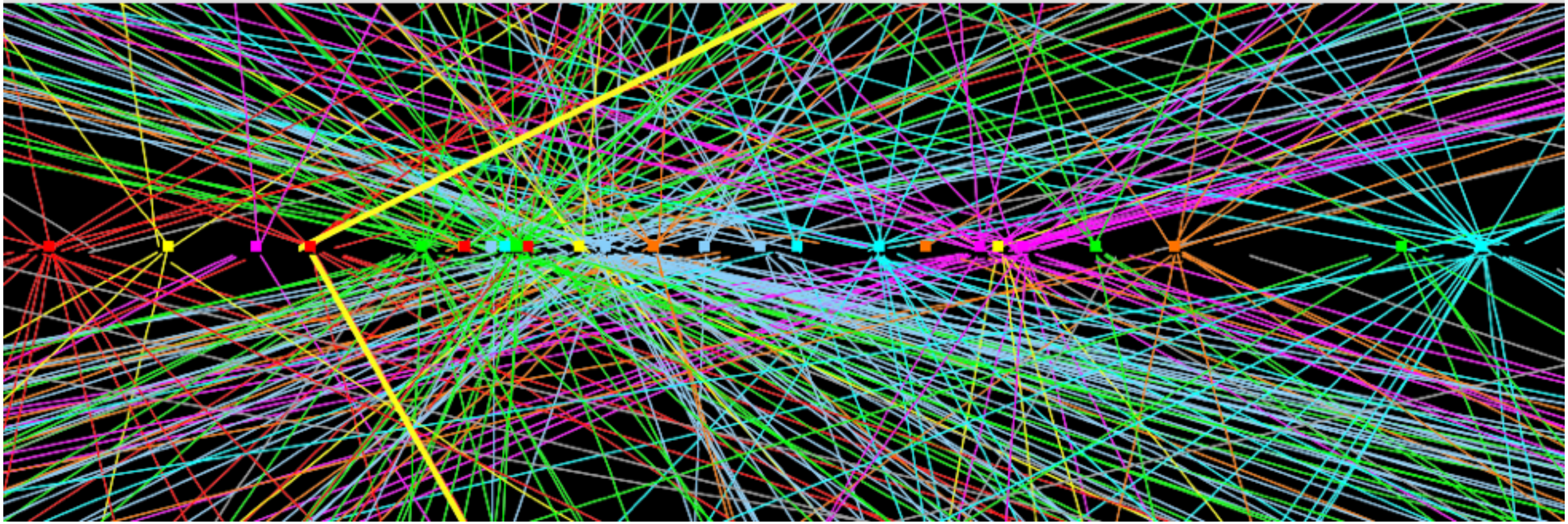
- Output of events generators
- File with all the characteristics of generated events: η, p_T, E, …

```
E 0 -1 4.6660200000000003e+01 1.4078099999999999e-01 7.7585830116479182e-03 9999 0 2024 1 2 0 1 1.0000000000000000e+00
N 1 "0"
U GEV MM
C 3.4466500000000003e+01 3.4466500000000003e+01
F -4 21 8.7436883742857147e-03 1.9855286214285714e-02 4.6660200000000003e+01 0 0 0 0
V -1 0 0 0 0 0 0 0 1 0
P 3 -4 0 0 6.1205818620000002e+01 6.1205818620000002e+01 0 21 0 0 -3 1 2 501
V -2 0 0 0 0 0 0 0 2 0
P 4 21 0 0 -1.3898700349999999e+02 1.3898700349999999e+02 0 21 0 0 -3 2 1 501 2 502
P 24 21 5.7185177904743236e+00 -6.1609436926723209e+00 -2.8704929448187880e+01 2.9910393637157195e+01 0 43 0 0 -23 2 1 507 2 501
V -3 0 0 0 0 0 0 0 2 0
P 5 25 4.0718400500000001e+01 -2.2785643319999998e+01 -7.5882461910000004e+01 1.5349401240000000e+02 1.2500036050000000e+02 22 0 0 -15 0
P 6 -4 -4.0718400500000001e+01 2.2785643319999998e+01 -1.8987229400000001e+00 4.6698809730000001e+01 7.3666614860000002e-07 23 0 0 -11 1 2 502
V -4 0 0 0 0 0 0 0 1 0
P 7 2 0 0 3.8552334991618829e+02 3.8552334991618818e+02 0 31 0 0 -6 1 1 503
V -5 0 0 0 0 0 0 0 2 0
P 8 21 0 0 -5.5383686130864440e-01 5.5383686130864385e-01 0 31 0 0 -6 2 1 504 2 505
P 15 21 -5.8817756360796549e+00 8.4497209454207436e+00 2.7410514378725580e+01 2.9280187280736172e+01 0 43 0 0 -77 2 1 505 2 506
V -6 0 0 0 0 0 0 0 2 0
P 9 2 1.1737916413747280e+01 5.0540118425919403e+00 9.8963597450047828e+01 9.9785891977254437e+01 3.3000000000000002e-01 33 0 0 -9 1 1 504
P 10 21 -1.1737916413747280e+01 -5.0540118425919403e+00 2.8600591560483167e+02 2.8629129480024238e+02 0 33 0 0 -10 2 1 503 2 505
V -7 0 0 0 0 0 0 0 1 0
P 11 2 -8.4376949871511897e-15 1.1546319456101628e-14 3.8552334991619614e+02 3.8552334991619614e+02 0 42 0 0 -4 1 1 503
V -8 0 0 0 0 0 0 0 1 0
P 12 21 7.9936057773011271e-15 -8.8817841970012523e-15 -1.6068131961151266e+00 1.6068131961150129e+00 0 41 0 0 -5 2 1 504 2 506
V -9 0 0 0 0 0 0 0 1 0
P 13 2 1.6104321264787131e+01 -1.2187372061304131e+00 1.3027367477813596e+02 1.3127137435091254e+02 3.3000000000000002e-01 44 0 0 -24 1 1 504
V -10 0 0 0 0 0 0 0 1 0
P 14 21 -1.0222545628707394e+01 -7.2309837392899681e+00 2.2623234756322017e+02 2.2657860148067448e+02 0 44 0 0 -76 2 1 503 2 505
```

Exercise: Minimum Bias Events file

File with Minimum Bias events

- File with a collection of PU events
- Delphes accesses this file randomly for every signal event to take the selected number of PU events and superimpose them to the main one



Exercise: DataCard (1)

Delphes DataCard

File with ALL the settings of the simulation: efficiencies, resolutions, parameters for isolation, jets, ...

Structure:

- Sequence of the processing modules to be called during the execution
- Definition of the parametrizations for the modules (Muon Momentum Smearing, Muon efficiency, Electron Energy Resolution, Muon Isolation, ...)

  Last Module: TreeWriter
  (choice of the Branches to write in the ROOT Tree)

Exercise: DataCard (2)

```
#######################################
# Order of execution of various modules
#######################################

set ExecutionPath {

  PileUpMerger
  ParticlePropagator

  ChargedHadronTrackingEfficiency
  ElectronTrackingEfficiency
  MuonTrackingEfficiency

  ChargedHadronMomentumSmearing
  ElectronEnergySmearing
  MuonMomentumSmearing

  TrackMerger
  Calorimeter
  TrackPileUpSubtractor
  NeutralTowerMerger
  EFlowMergerAllTracks
  EFlowMerger

  NeutrinoFilter
  GenJetFinder

  Rho
  FastJetFinder
  PileUpJetID
  JetPileUpSubtractor

  JetEnergyScale

  PhotonEfficiency
  PhotonIsolation

  ElectronEfficiency
  ElectronIsolation

  MuonEfficiency
  MuonIsolation

  MissingET

  BTagging
  TauTagging

  UniqueObjectFinder

  ScalarHT

  TreeWriter
}
```

```
##########################
# Muon tracking efficiency
##########################

module Efficiency MuonTrackingEfficiency {
  set InputArray ParticlePropagator/muons
  set OutputArray muons

  # set EfficiencyFormula {efficiency formula as a function of eta and pt}

  # tracking efficiency formula for muons
  set EfficiencyFormula {                                        (pt <= 0.1)   * (0.00) +
                                         (abs(eta) <= 1.5) * (pt > 0.1 && pt <= 1.0) * (0.75) +
                                         (abs(eta) <= 1.5) * (pt > 1.0)              * (0.99) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 0.1 && pt <= 1.0) * (0.70) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 1.0)              * (0.98) +
                         (abs(eta) > 2.5)                                              * (0.00)}
}
```

```
################################
# Momentum resolution for muons
################################

module MomentumSmearing MuonMomentumSmearing {
  set InputArray MuonTrackingEfficiency/muons
  set OutputArray muons

  # set ResolutionFormula {resolution formula as a function of eta and pt}

  # resolution formula for muons
  set ResolutionFormula {                    (abs(eta) <= 0.5) * (pt > 0.1   && pt <= 5.0)   * (0.02) +
                                             (abs(eta) <= 0.5) * (pt > 5.0   && pt <= 1.0e2) * (0.015) +
                                             (abs(eta) <= 0.5) * (pt > 1.0e2 && pt <= 2.0e2) * (0.03) +
                                             (abs(eta) <= 0.5) * (pt > 2.0e2)                * (0.05 + pt*1.e-4) +
                         (abs(eta) > 0.5 && abs(eta) <= 1.5) * (pt > 0.1   && pt <= 5.0)   * (0.03) +
                         (abs(eta) > 0.5 && abs(eta) <= 1.5) * (pt > 5.0   && pt <= 1.0e2) * (0.02) +
                         (abs(eta) > 0.5 && abs(eta) <= 1.5) * (pt > 1.0e2 && pt <= 2.0e2) * (0.04) +
                         (abs(eta) > 0.5 && abs(eta) <= 1.5) * (pt > 2.0e2)                * (0.05 + pt*1.e-4) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 0.1   && pt <= 5.0)   * (0.04) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 5.0   && pt <= 1.0e2) * (0.035) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 1.0e2 && pt <= 2.0e2) * (0.05) +
                         (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 2.0e2)                * (0.05 + pt*1.e-4)}
}
```

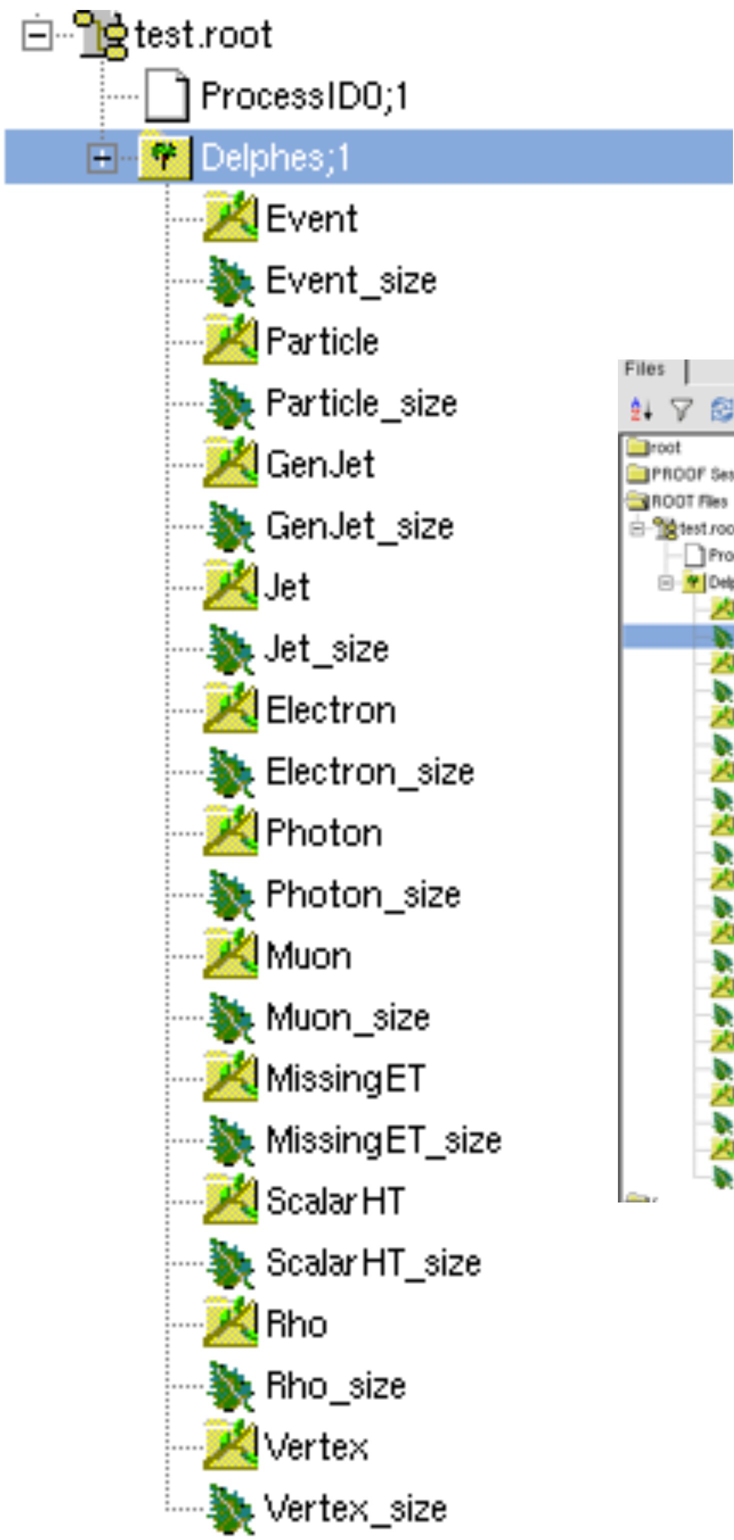
Exercise: DataCard (3)

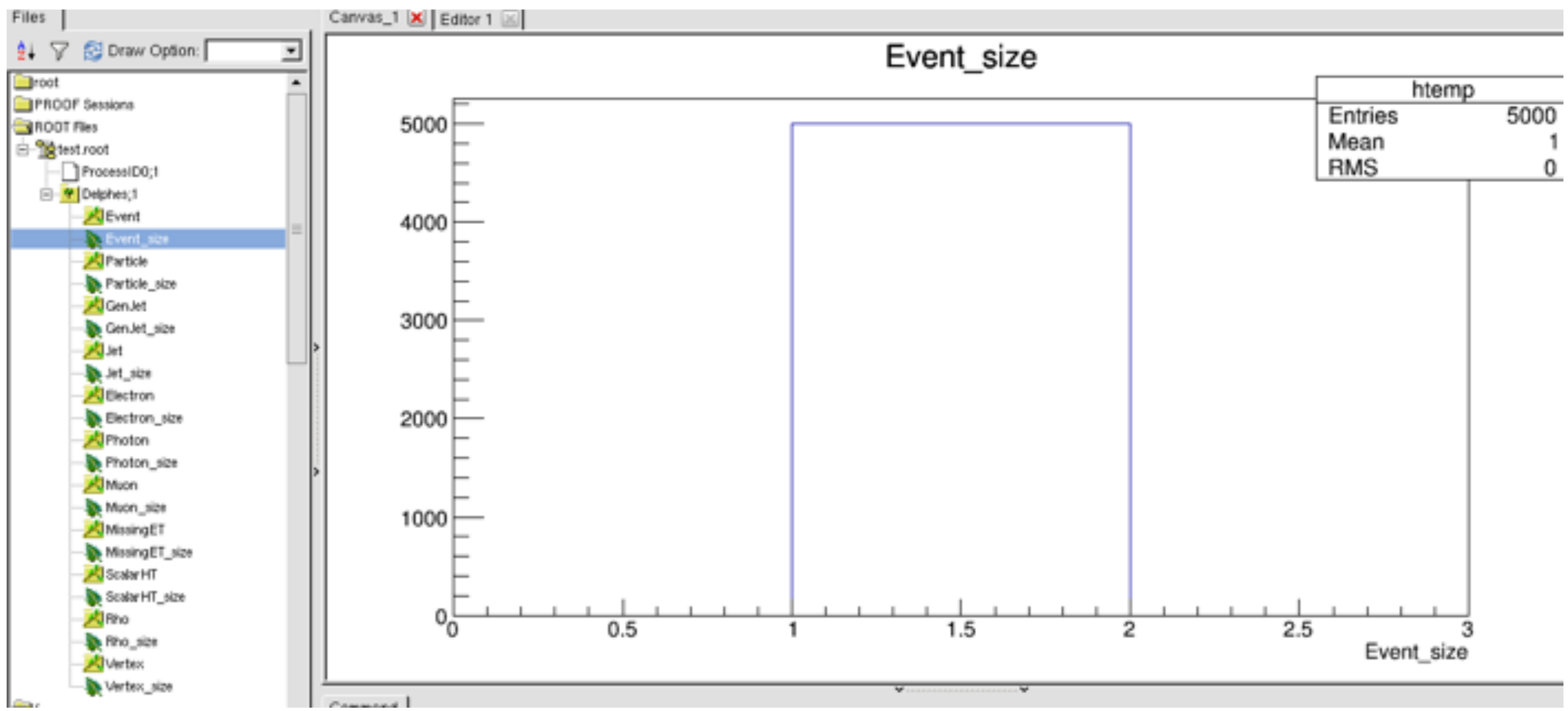
```
#################
# Muon efficiency
#################

module Efficiency MuonEfficiency {
  set InputArray TrackPileUpSubtractor/muons
  set OutputArray muons

  # set EfficiencyFormula {efficiency as a function of eta and pt}

  # efficiency formula for muons
  set EfficiencyFormula {                                    (pt <= 10.0)               * (0.00) +
                                        (abs(eta) <= 1.5) * (pt > 10.0 && pt <= 1.0e3) * (0.95) +
                                        (abs(eta) <= 1.5) * (pt > 1.0e3)               * (0.95 * exp(0.5 - pt*5.0e-4)) +
                        (abs(eta) > 1.5 && abs(eta) <= 2.4) * (pt > 10.0 && pt <= 1.0e3) * (0.95) +
                        (abs(eta) > 1.5 && abs(eta) <= 2.4) * (pt > 1.0e3)               * (0.95 * exp(0.5 - pt*5.0e-4)) +
                        (abs(eta) > 2.4)                                                 * (0.00)}
}
```

```
################
# Muon isolation
################

module Isolation MuonIsolation {
  set CandidateInputArray MuonEfficiency/muons
  set IsolationInputArray EFlowMerger/eflow
  set RhoInputArray Rho/rho

  set OutputArray muons

  set DeltaRMax 0.5

  set PTMin 0.5

  set PTRatioMax 0.1
}
```

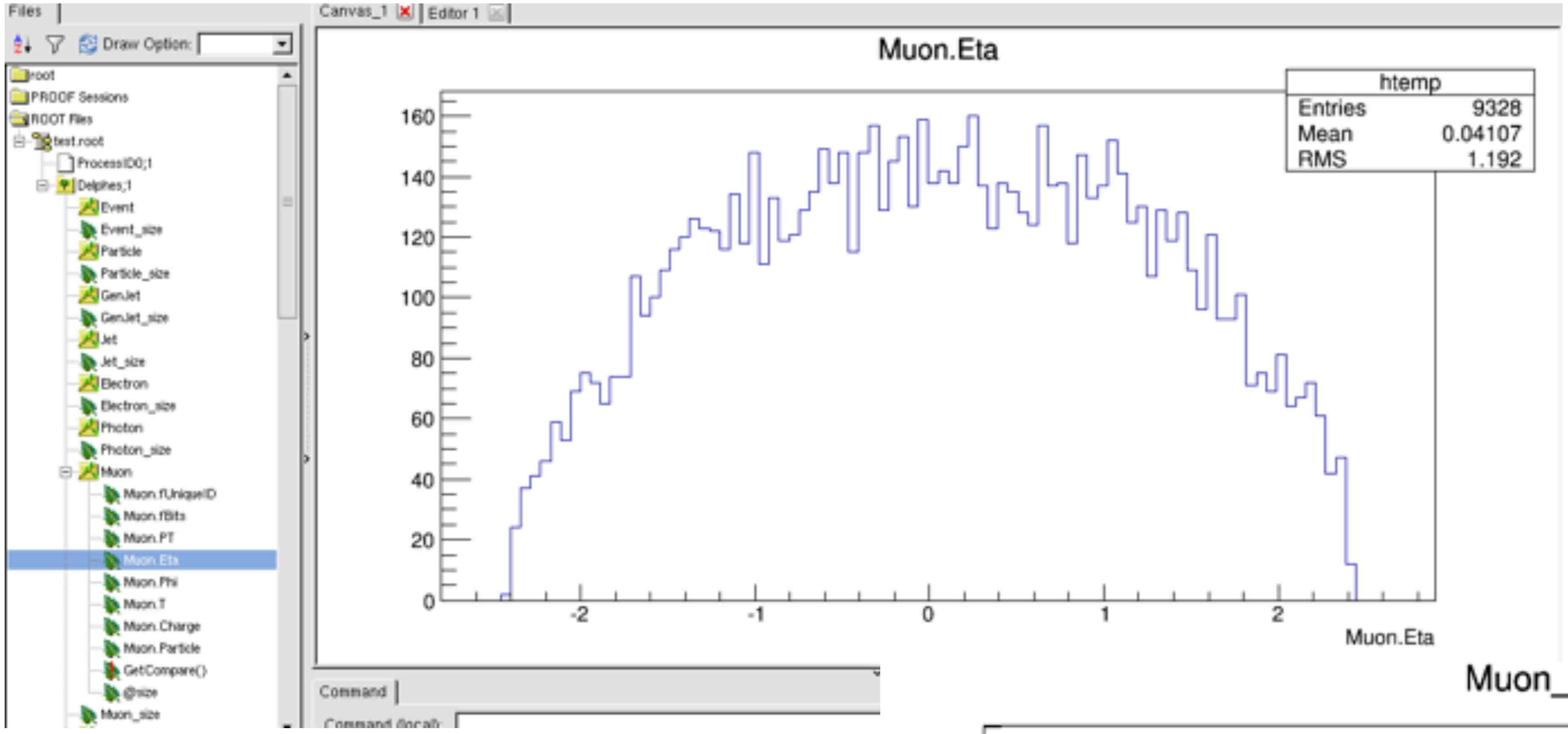
```
module TreeWriter TreeWriter {
# add Branch InputArray BranchName BranchClass
  add Branch Delphes/allParticles Particle GenParticle

#  add Branch TrackMerger/tracks Track Track
#  add Branch Calorimeter/towers Tower Tower

#  add Branch Calorimeter/eflowTracks EFlowTrack Track
#  add Branch Calorimeter/eflowPhotons EFlowPhoton Tower
#  add Branch Calorimeter/eflowNeutralHadrons EFlowNeutralHadron Tower

  add Branch GenJetFinder/jets GenJet Jet
  add Branch UniqueObjectFinder/jets Jet Jet
  add Branch UniqueObjectFinder/electrons Electron Electron
  add Branch UniqueObjectFinder/photons Photon Photon
  add Branch UniqueObjectFinder/muons Muon Muon
  add Branch MissingET/momentum MissingET MissingET
  add Branch ScalarHT/energy ScalarHT ScalarHT
  add Branch Rho/rho Rho Rho
  add Branch PileUpMerger/vertices Vertex Vertex
}
```

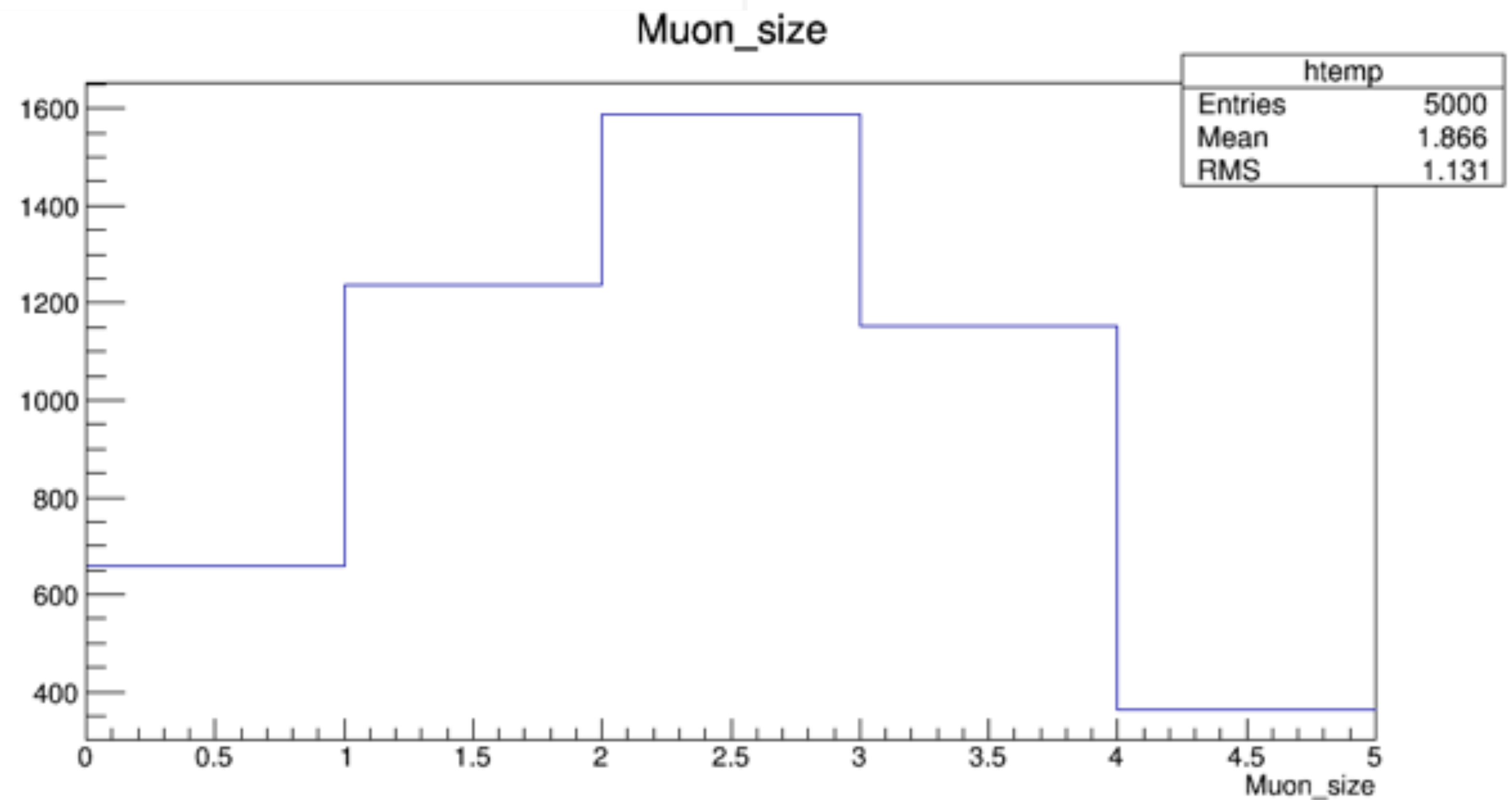
Exercise: Delphes ROOT Tree (1)





Exercise: Delphes ROOT Tree (2)





Exercise: H → ZZ → 4l analysis macros

DelphesMU_analysis.C - **DelphesMU_distributions.C**

Opening of the output file

Reading of the Delphes Tree

Declarations of histograms (Higgs invariant mass and Cutflow table / Efficiency vs η and vs p_T , p_T Resolution)

Loop on every event (sequential cuts to identify events $H \rightarrow ZZ^* \rightarrow 4l$ and filling of histrograms)

Writing of the .histo file

Plots.C

Drawing and comparing of the histograms

Exercise: instructions (1)

Login:

```
ssh schoolXX@naf-schoolXX.desy.de -Y
```

(pwd)

Main steps to start using Delphes in Desy devices.
In **BLUE**: steps you have to do only the first time that you follow these instructions
In **RED**: steps you have to repeat EVERY TIME you start a new session in your machine

Setup GCC and ROOT:

```
source /afs/cern.ch/sw/lcg/external/gcc/4.7/x86_64-slc6/setup.sh

cd /afs/cern.ch/sw/lcg/app/releases/ROOT/5.34.18/x86_64-slc6-gcc47-opt/root/

source bin/thisroot.sh

cd
```

Exercise: instructions (2)

Download Delphes and compile it:

```
wget http://cp3.irmp.ucl.ac.be/downloads/Delphes-3.2.0.tar.gz

tar -zxf Delphes-3.2.0.tar.gz

cd Delphes-3.2.0/

make
```

From inside the folder Delphes-3.2.0 (useful to use the macros to analyze the ROOT Tree):

```
export DELPHES_PATH=$(pwd)

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$DELPHES_PATH
```

Exercise: instructions (3)

To run Delphes, you need:

1) a file with generated events, in .hepMC format (or other): **be sure to know its path (don't copy it!)**. The .hepMC files we are going to use for this exercise are in:

/afs/desy.de/project/school/Delphes_tutorial/hepMC/

h_ggH_ZZ_4mu_125_5k.hepmc (4-muon channel)
h_ggH_ZZ_4elec_125_5k.hepmc (4-electron channel)

2) a file in .tcl format (Delphes Card). For this exercise use the default cards inside the folder cards:

delphes_card_ATLAS_PileUp.tcl
or
delphes_card_CMS_PileUp.tcl

Exercise: instructions (4)

3) If you run with Pile Up (our case!) you need one file with a collection of Minimum Bias Events (MinBias.pileup). Be sure to know its path: in the Delphes Card, you have to indicate the MinBias.pileup path. **Copy it in your folder Delphes-3.2.0** from:

/afs/desy.de/project/school/Delphes_tutorial

so that the path indicated in the card is already correct.

After the production of the .root ntuples with Delphes, you will have to analyze them with some macros that I've prepared which perform a simple analysis, **copy** the folder macros from:

/afs/desy.de/project/school/Delphes_tutorial/

Exercise: instructions (5)

Run Delphes using the CMS or ATLAS card with Pile Up (default: PU=50).

```
./DelphesHepMC cards/delphes_card_CMS_PileUp.tcl
test.root /afs/desy.de/project/school/
Delphes_tutorial/hepMC/h_ggH_ZZ_4mu_125_5k.hepmc
```

Exercise: your turn!

All the instructions in file: Delphes_Exercise_Instruction.pdf

Thank you for the attention!

For any question: lisa.borgonovi@bo.infn.it

Backup

Exercise_1

Exercise: MUONS

1) Run Delphes using the CMS or ATLAS card with Pile Up (default: PU=50).

The structure of the shell command line to run Delphes (if you are using a hepMC file) is:

```
./DulphesHepMC cards/cardname.tcl path/outputfile.root path/hepmcfile.hepmc
```

In this case the command line is:

```
./DelphesHepMC cards/delphes_card_CMS_PileUp.tcl test.root /afs/desy.de/project/school/
Delphes_tutorial/hepMC/h_ggH_ZZ_4mu_125_5k.hepmc
```

When the .root file is produced, look at its structure with the ROOT TBrowser:

```
root -l
gSystem->Load("libDelphes")
TFile *_file0 = TFile::Open("name.root")
new TBrowser
```

Now you can analyze the .root file with the H → ZZ → 4mu macros:

DelphesMU_analysis.C
DelphesMU_distributions.C

Exercise_2

To compile the macros:

```
source /compileDelphesMU_analysis
or
source /compileDelphesMU_distributions
```

To run the macros on one .root file:

```
./RunDelphesMU_analysis path/name.root
```
or
./RunDelphesMU_distributions path/name.root

Look at the .histo produced with the TBrowser.

2) Produce three new cards starting from the one you just used:
In the first one change the value of "PTRatioMax" for muons (it is a cut on the particle isolation: with the default value, only particle with an "isolation value" less than 0.1 are saved. This cut is very tight, so try with no cut e.g. 9999).

In the second one maintain the value of "PTRatioMax" to 9999 and extend the muon reconstruction efficiency down to pT = 3 GeV.

In the third one maintain "PTRatioMax" at 9999 and the muon reconstruction efficiency down to pT = 3 GeV as before and then try to simulate an upgraded detector, with better muon tracking efficiency, muon reconstruction efficiency, muon pT resolution and eta coverage extension up to 3.5.

After producing these 3 new samples, use the macro to apply the $H \rightarrow ZZ \rightarrow 4mu$ analysis on each one of them. Then compare the three results using the macro Plots.C. To use it, open the macro and insert the correct name of the files you want to read, then:

```
root -l

.x Plots.C
```

Exercise_3

Exercise: ELECTRONS

1) Run Delphes using the CMS or ATLAS card with Pile Up (default: PU=50), changing the electron "PTRatioMax" value to 9999 and extending the electron reconstruction efficiency down to pT = 3 GeV.

2) Produce a new card starting from the one just used, modifying the calorimeter granularity

3) Produce a new card with the original calorimeter granularity, but with better electron tracking efficiency, reconstruction efficiency, pT resolution and eta extension

After producing these three new samples, modify the macros used before to apply the H → ZZ → 4ele analysis on each one of them.

Tips:

PID(ele) = 11
Mass(ele) = 0.00051 GeV
Remember to modify also the compiler and the .h!

Then use the macro Plots.C to compare the two results.