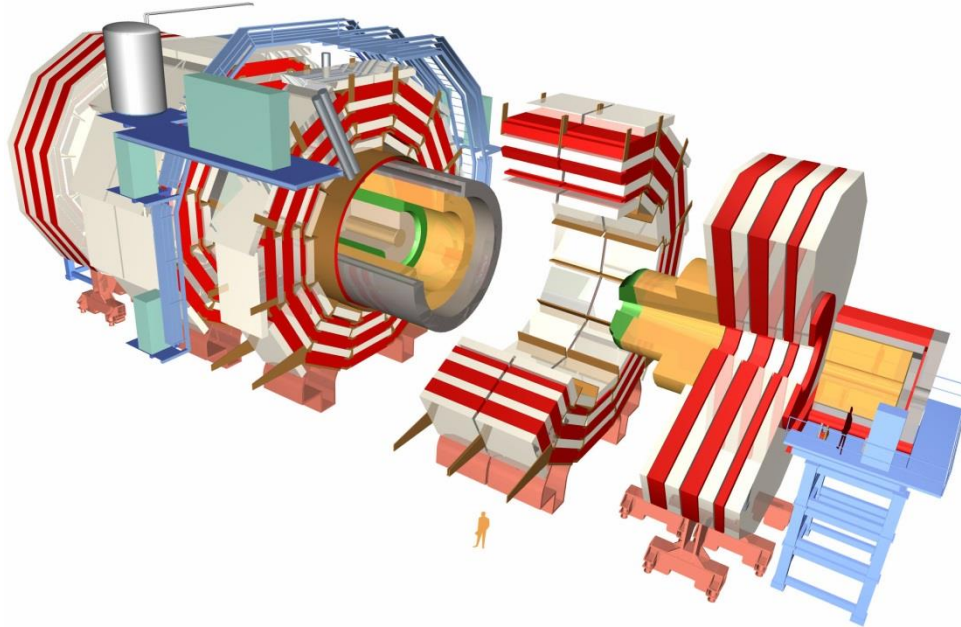




Detector simulation for LHC analyses recasting - Part 1



Eric CONTE

The second MadAnalysis 5 workshop on LHC recasting @ Korea
13-20 February 2020

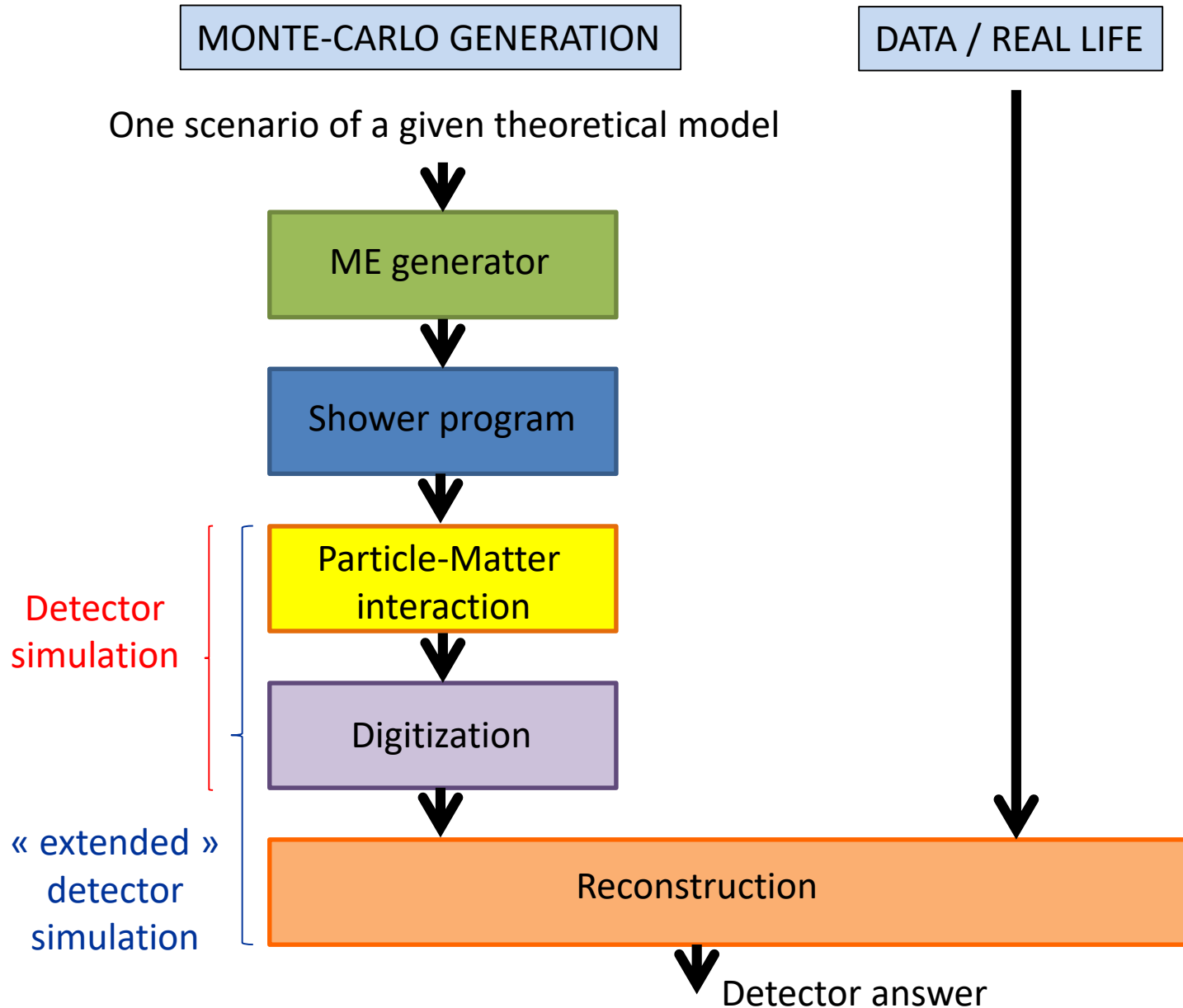
Sunday

- 1. Different categories of detector simulation**
- 2. General concepts on Delphes**
- 3. Delphes sequence for simulating the ATLAS or CMS detector**
- 4. Validation & limitations of Delphes**
- 5. Application to the recasting of the analysis SUSY-2018-32**

Monday

- 1. Pile-up simulation**
- 2. Validation of your analysis**

1. Different categories of detector simulation

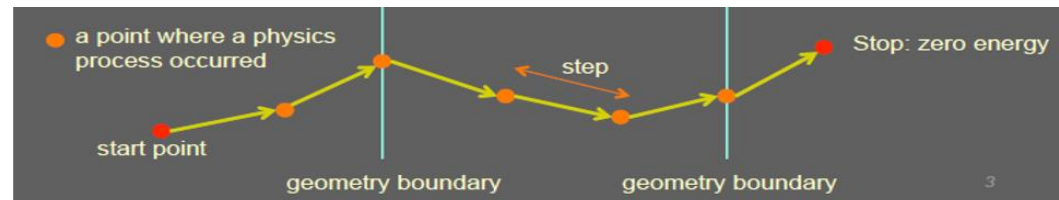
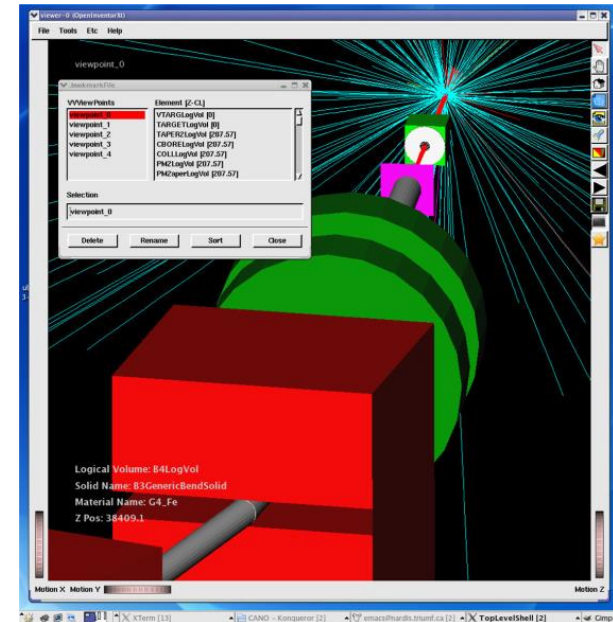


Category 1: fullsim ATLAS/CMS

Particle-Matter
interaction

Based on Monte-Carlo radiation transportation codes
The most known: **Geant4**

- Describe the **geometry** and the **material budget** of the detector.
- Treat a particle at the time.
- Trajectory of the particle is split in steps.
A **step** =
 - Physics process (interaction)
 - Volume boundary
- Simulate the physics along a step and at the end of each step.



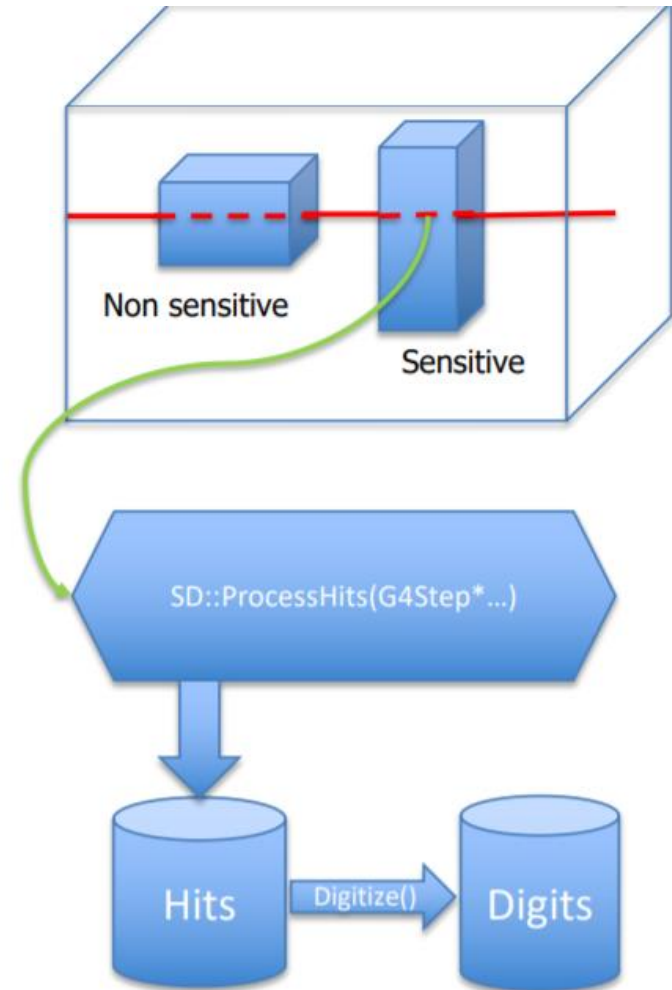


Digitization

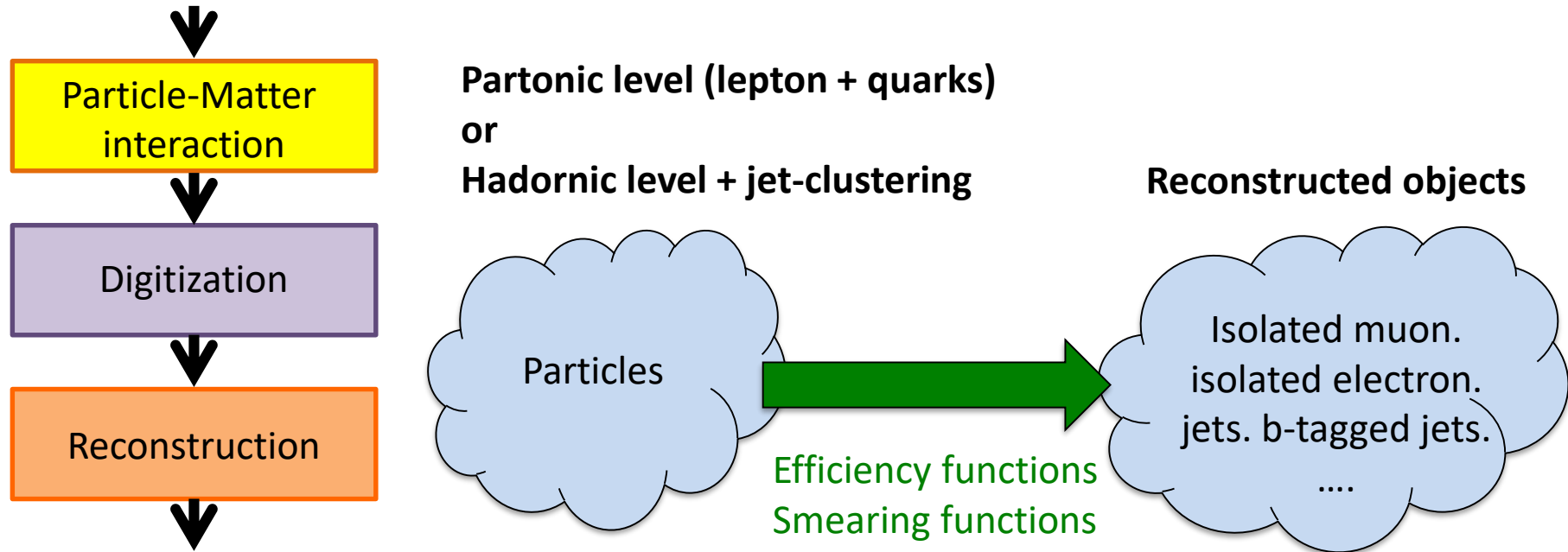


The most known tool:
Geant4

- Goal: **convert the energy deposit** into:
 - Electric current
 - Voltage signals
- Identifying the **sensitive part** of the detector
- Modelizing its answer by a **digitizer**:
 - simulate ADC or TDC
 - simulate readout scheme
 - generate raw data
 - simulate trigger logics

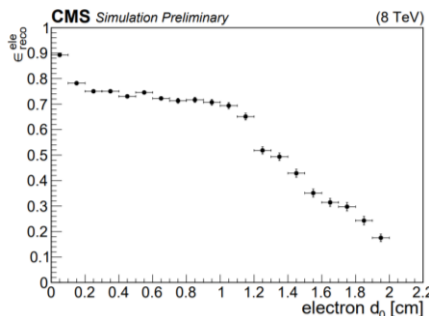


Category 2: parametric functions



Example: parametrization for displaced leptons within the CMS detector

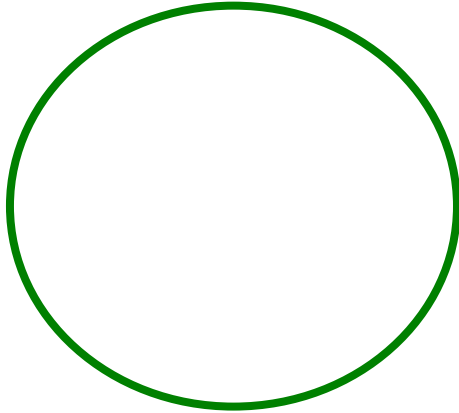
<https://twiki.cern.ch/twiki/bin/view/CMSPublic/DisplacedSusyParametrisationStudyForUser>



Can be directly applied on the top of LHE events produced by MadGraph

Categories of tools

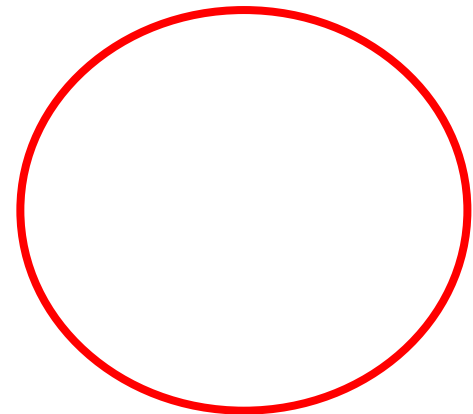
realism



Full simulation:

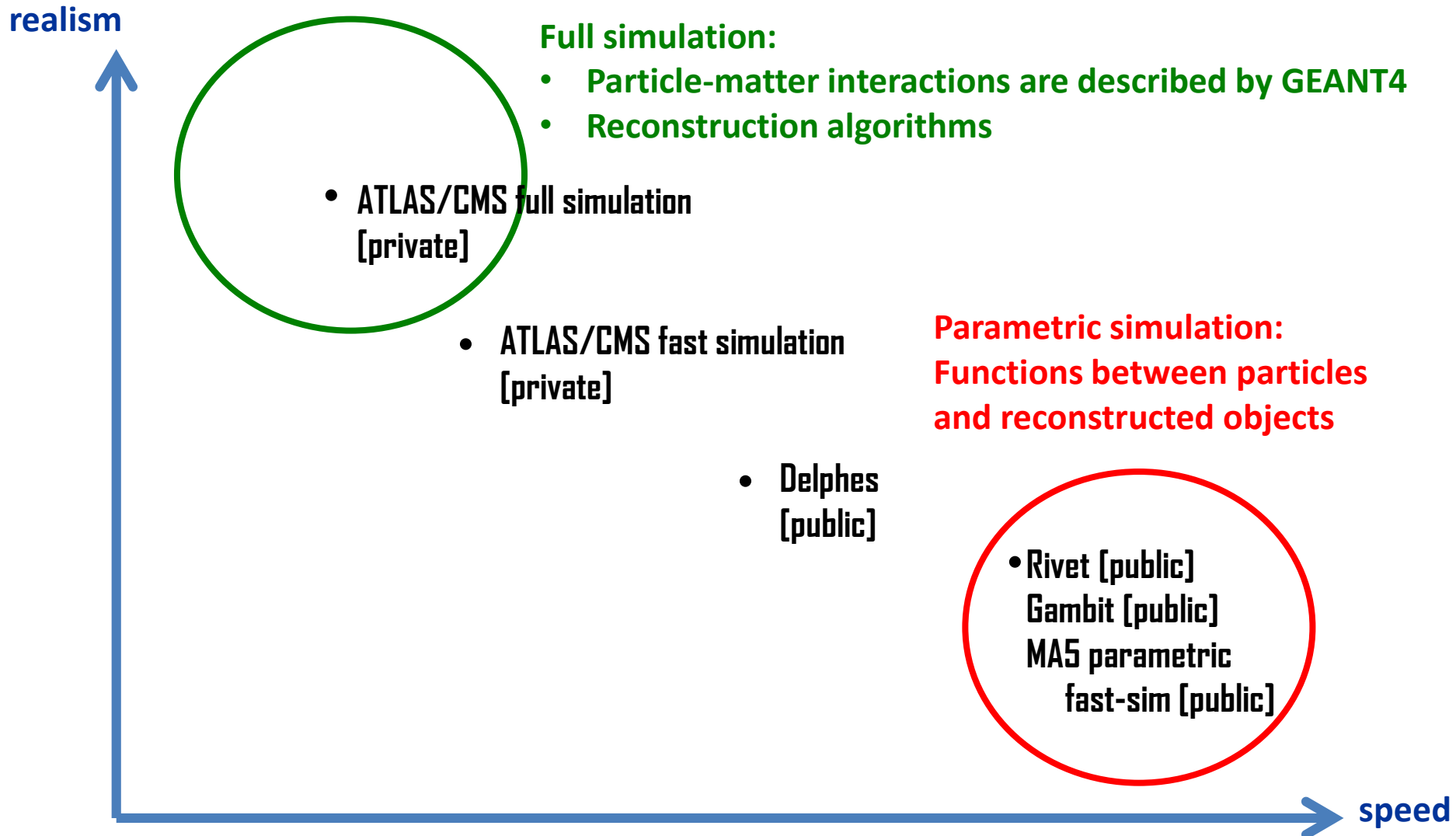
- Particle-matter interactions are described by GEANT4
- Reconstruction algorithms

Parametric simulation:
Functions between particles
and reconstructed objects

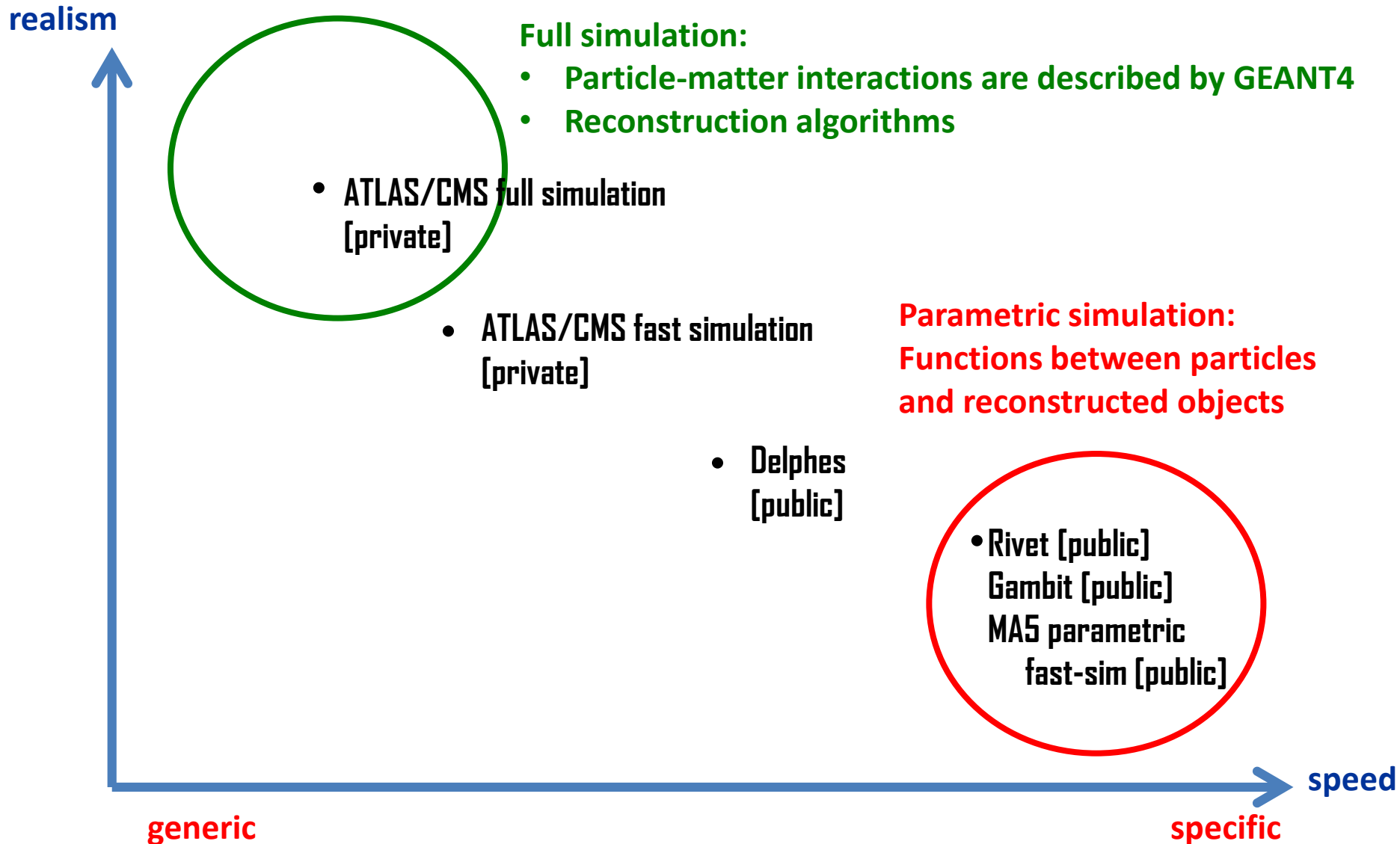


speed

Categories of tools



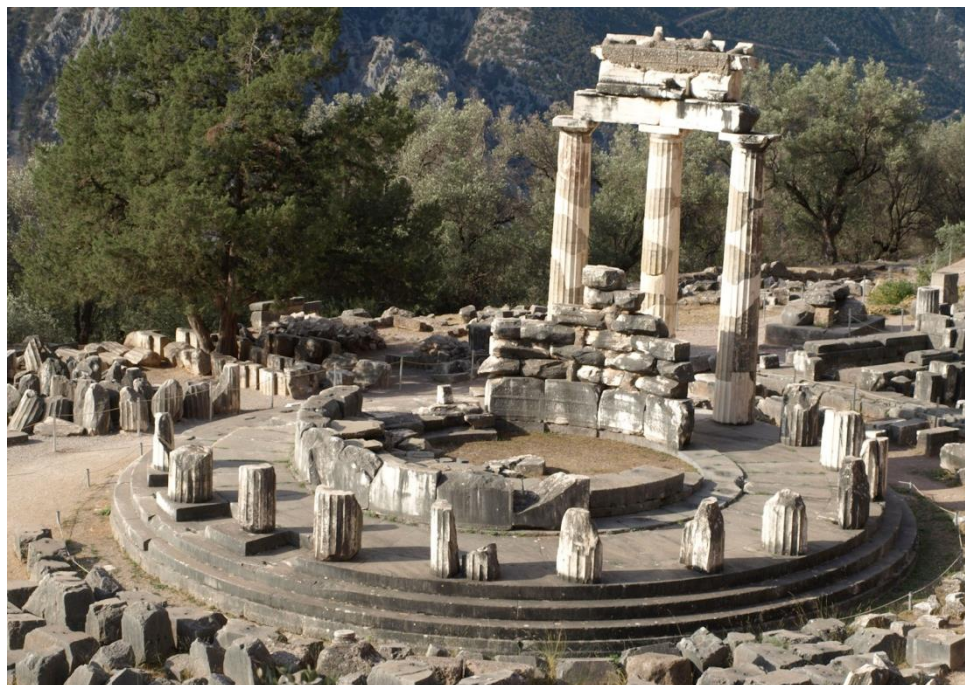
Categories of tools



2. General concepts of Delphes

What is Delphes?

- **DELPHES** is a very-fast-simulation for generic detector:
 - ATLAS & CMS detectors
 - Upgrade of ATLAS & CMS
 - LHCb
 - Future detectors: FCC
- **Output in ROOT format**



DELPHES
fast simulation

- [JHEP 02 \(2014\) 057](#)
- [J.Phys.Conf.Ser. 523 \(2014\) 012033](#)
- [J.Phys.Conf.Ser. 608 \(2015\) 1. 012045](#)

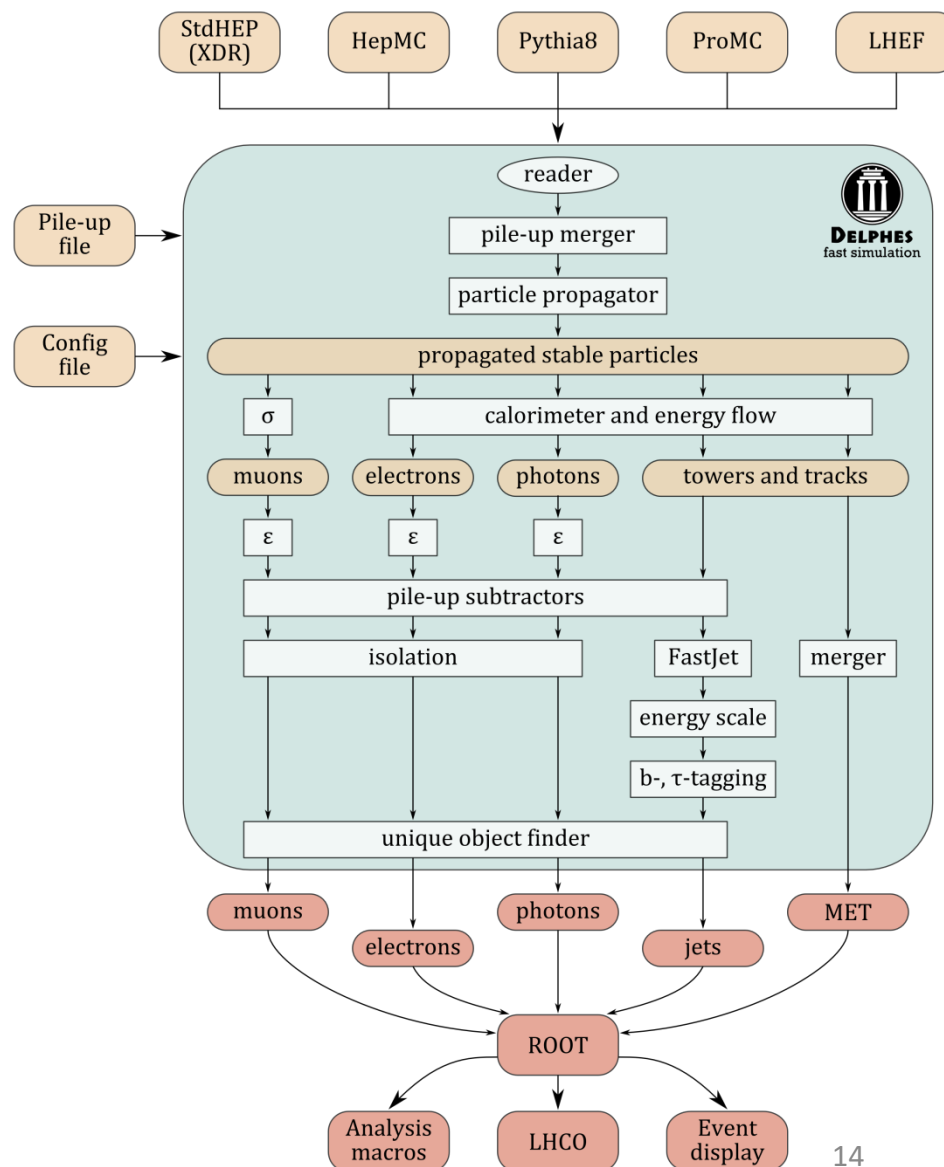
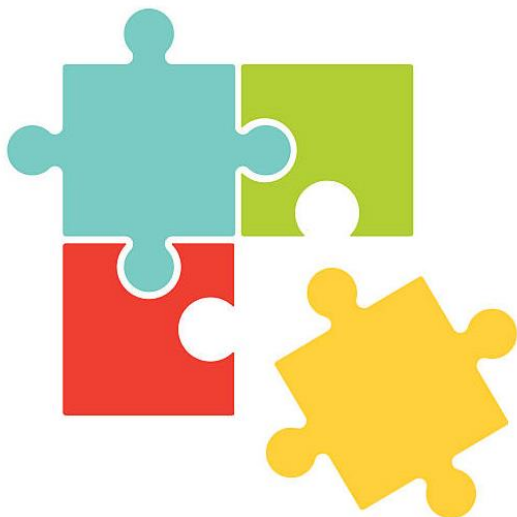
A question of linguistics:

What does "Delphes" mean?


Modular architecture


The detector simulation is split into modules.


→ Each module is devoted to a function.





List of modules


 AngularSmearing.h

 BeamSpotFilter.h


 BTagging.h


 Calorimeter.h


 Cloner.h


 ConstituentFilter.h


 Delphes.h

 Efficiency.h


 EnergyScale.h


 EnergySmearing.h


 ExampleModule.h


 FastJetFinder.h


 FastJetGridMedianEstimator.h


 FastJetLinkDef.h


 Hector.h


 IdentificationMap.h


 ImpactParameterSmearing.h


 Isolation.h


 JetFakeParticle.h


 JetFlavorAssociation.h


 JetPileUpSubtractor.h

 LeptonDressing.h


 Merger.h


 ModulesLinkDef.h


 MomentumSmearing.h


 OldCalorimeter.h


 ParticlePropagator.h


 PdgCodeFilter.h


 PhotonConversions.h


 PileUpJetID.h


 PileUpMerger.h


 PileUpMergerPythia8.h


 Pythia8LinkDef.h


 RecoPuFilter.h


 RunPUPPI.h


 SimpleCalorimeter.h


 StatusPidFilter.h


 TaggingParticlesSkimmer.h


 TauTagging.h


 TimeSmearing.h


 TrackCountingBTagging.h


 TrackCountingTauTagging.h


 TrackPileUpSubtractor.h


 TrackSmearing.h


 TreeWriter.h

 UniqueObjectFinder.h

 VertexFinder.h

 VertexFinderDA4D.h

 VertexSorter.h

 Weighter.h

Detector simulation is totally described by a card (text file in **tcl** language).

This card contains:

- the sequence of the modules that you need
- how they interact between themselves.

```
set ExecutionPath {  
    ParticlePropagator  
    ChargedHadronTrackingEfficiency  
    ElectronTrackingEfficiency  
    MuonTrackingEfficiency  
    ChargedHadronMomentumSmearing  
    ElectronMomentumSmearing  
    MuonMomentumSmearing  
    TrackMerger  
    ECal  
    HCal  
    Calorimeter  
    EFlowMerger  
    EFlowFilter  
    PhotonEfficiency  
    PhotonIsolation  
    ElectronFilter  
    ElectronEfficiency  
    ElectronIsolation  
    ChargedHadronFilter  
    MuonEfficiency  
    MuonIsolation  
    MissingET  
    NeutrinoFilter  
    GenJetFinder  
    GenMissingET  
    FastJetFinder  
}
```

Order of execution
of the modules use
in the simulation

Detector description

Detector simulation is totally described by a card (text file in **tcl** language).

This card contains:

- the sequence of the modules that you need
- how they interact between themselves.

Syntax of
module
declaration

```
module FastJetFinder FastJetFinder
{
  set InputArray EFlowMerger/eflow

  set OutputArray jets

  set JetAlgorithm 6
  set ParameterR 0.5
  set JetPTMin 20.0
}
```

Name & type
of module

Input collection

Output collection

Parameters

Requirements:

Package	Utility
ROOT 6	Main framework & data format
TCL	Language of detector card
FastJet	Jet-clustering algorithm. pile-up

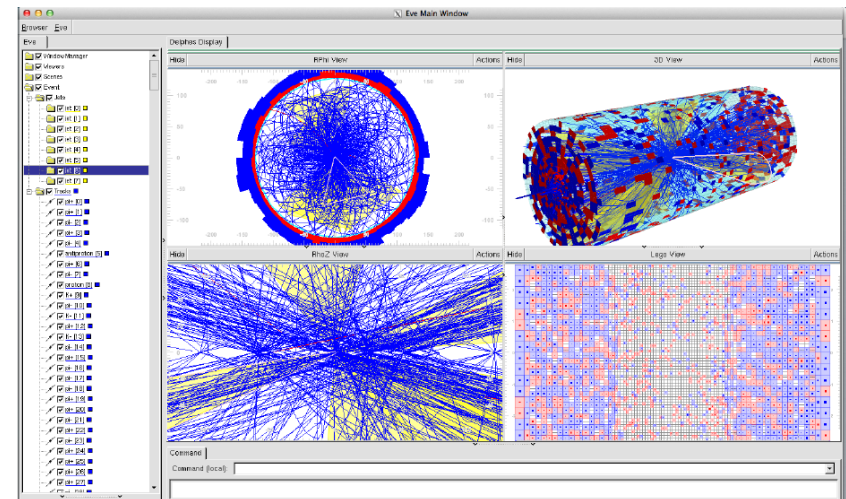
→ To be installed

→ To be installed

→ Encapsulated in the delphes package

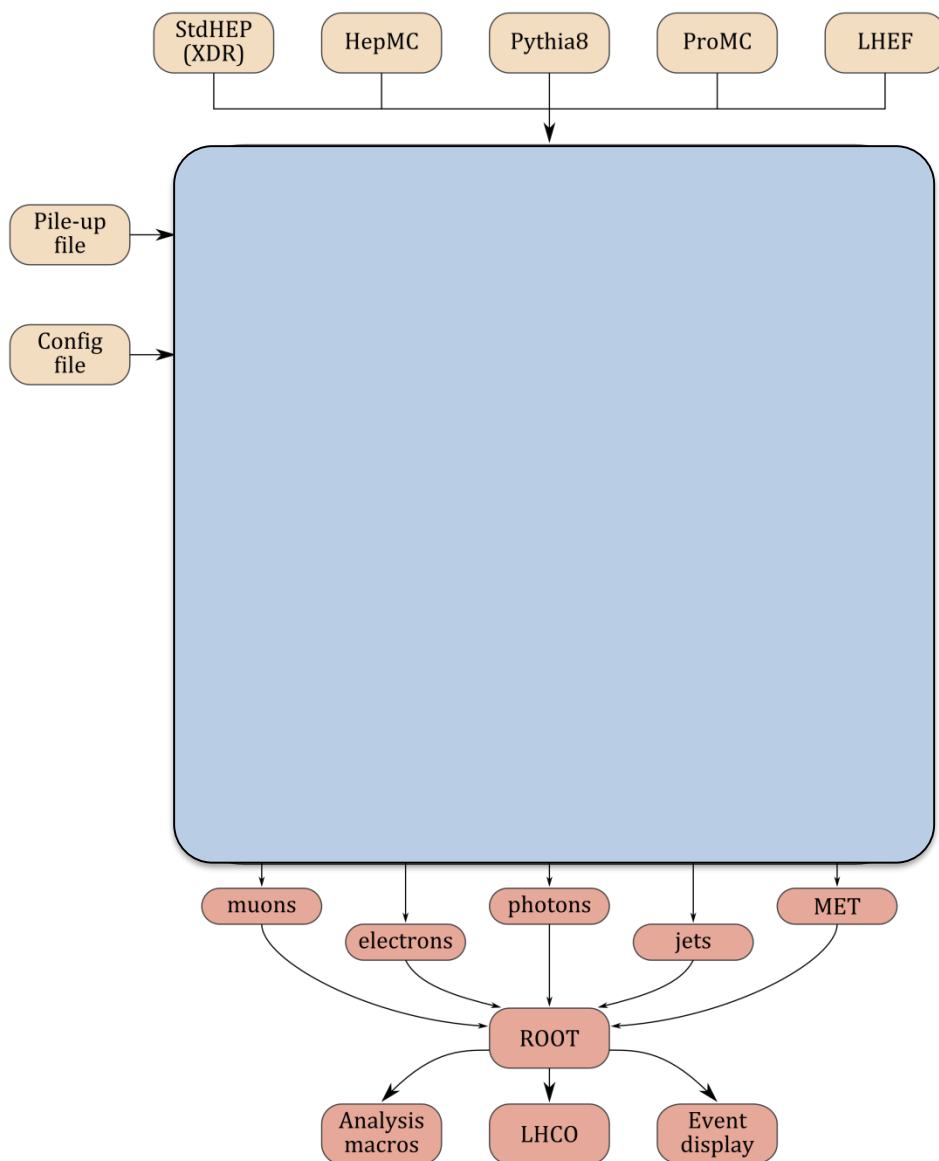
Extra programs:

- **EVE** (former FROG): program of event vizualisation
- **DelphesAnalysis**: reading Delphes ROOT file with Python



3. Delphes sequence for simulating the ATLAS or CMS detector

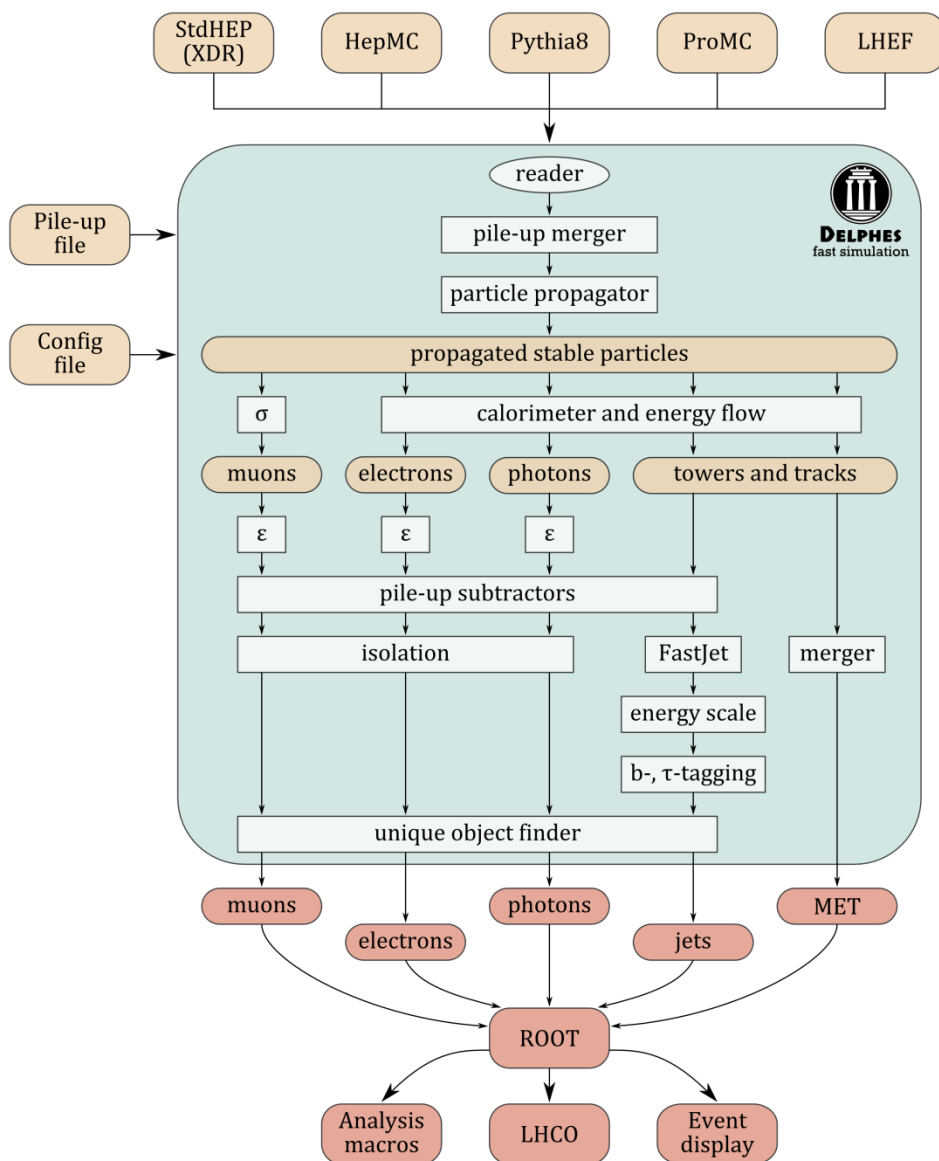
Dataflow diagram



Official dataflow diagram:

<https://cp3.irmp.ucl.ac.be/projects/delphes/wiki/Workbook/DataFlowDiagram>

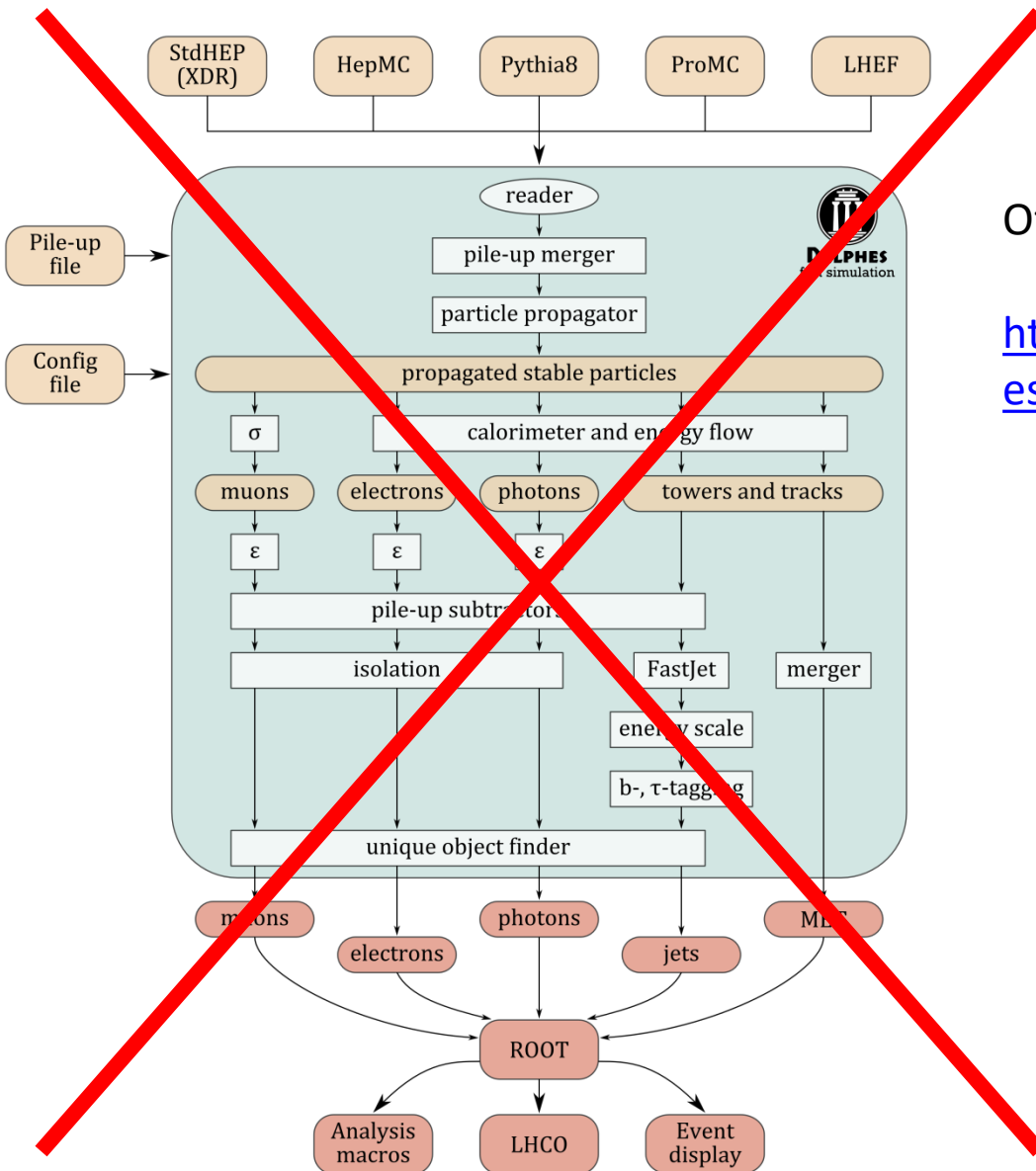
Dataflow diagram



Official dataflow diagram:

<https://cp3.irmp.ucl.ac.be/projects/delphes/wiki/WorkBook/DataFlowDiagram>

Dataflow diagram

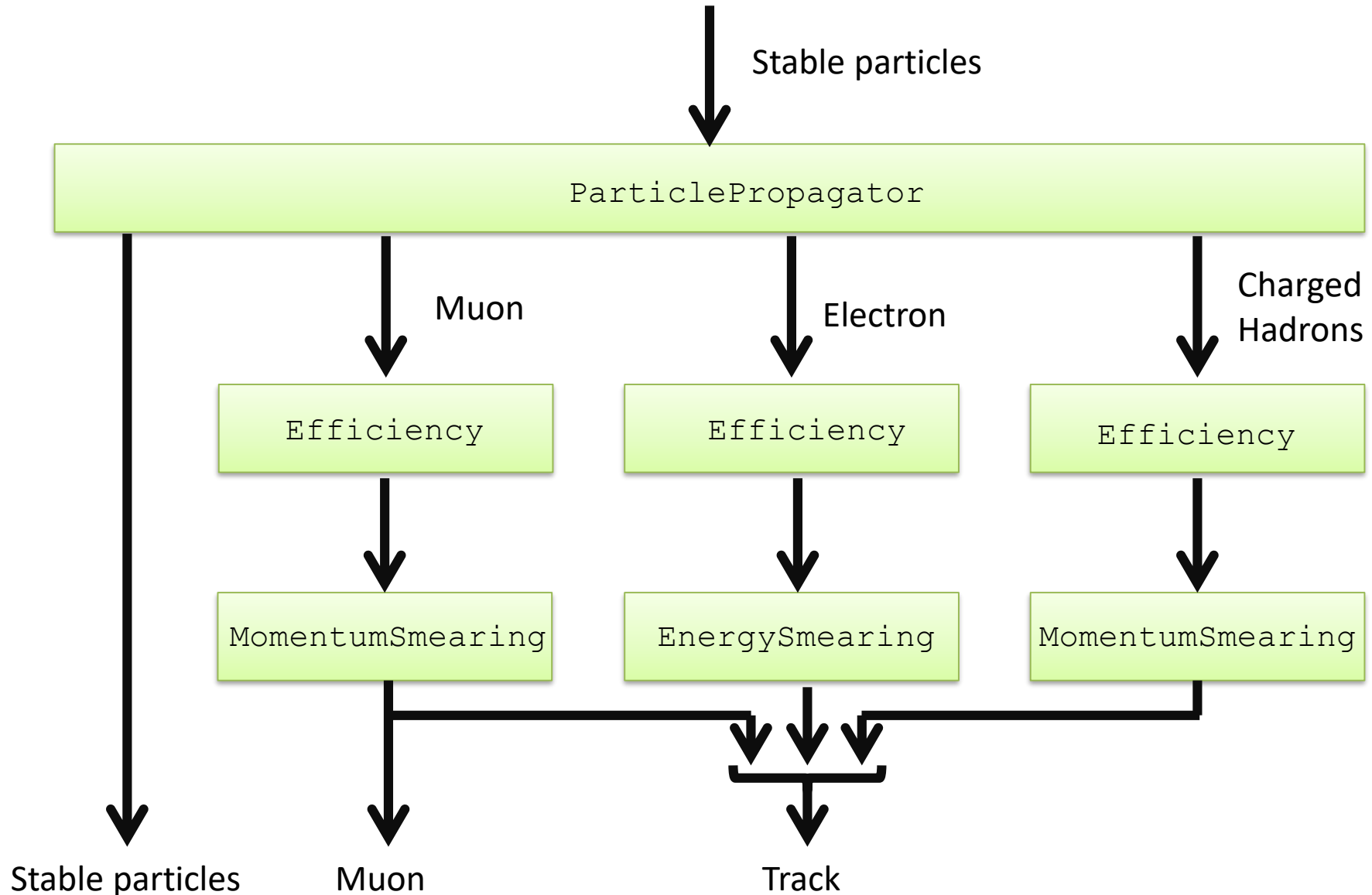


Official dataflow diagram:

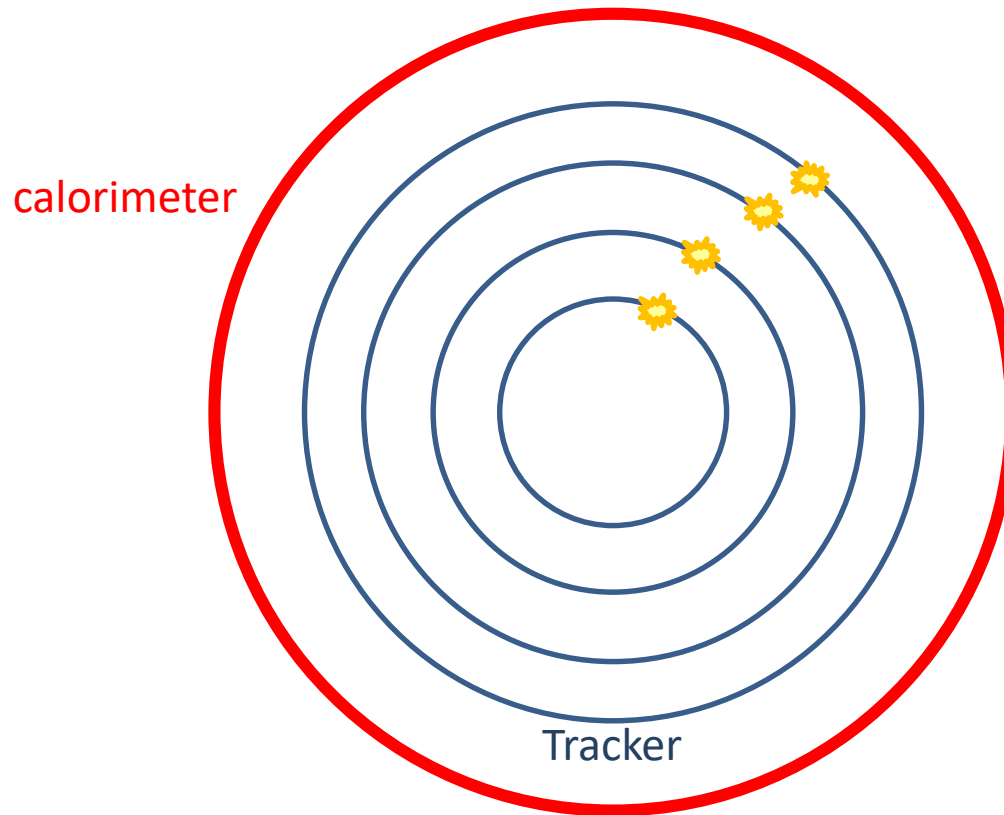
<https://cp3.irmp.ucl.ac.be/projects/delphes/wiki/WorkBook/DataFlowDiagram>

Obsolete with Delphes 3.4.2

Workflow part 1: tracking



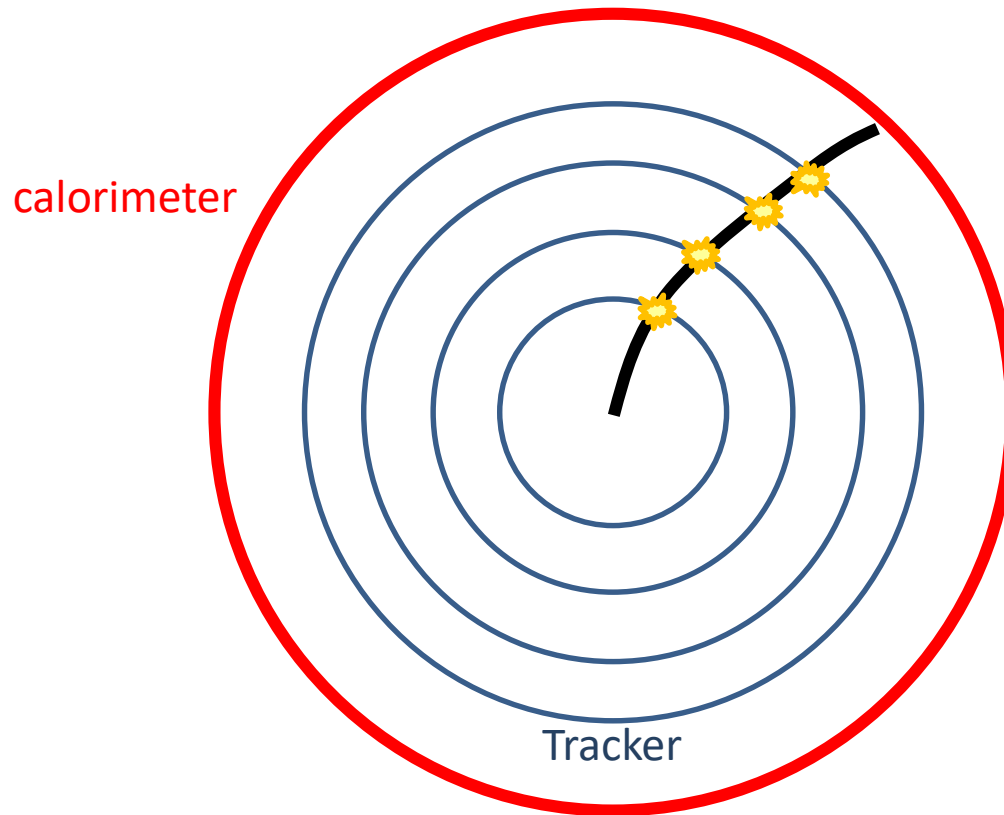
In the real life:



In Delphes. there is no real tracking or vertexing.

Workflow part 1: tracking

In the real life:

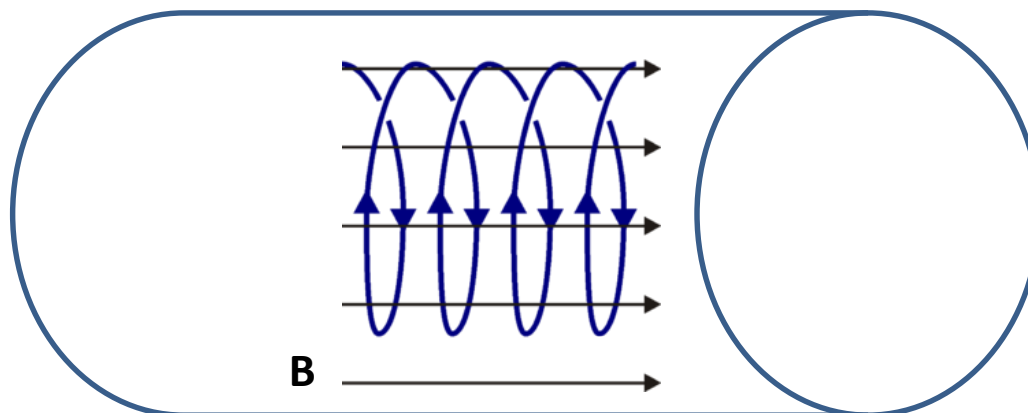


In Delphes. there is no real tracking or vertexing.

ParticlePropagator

- Charged particles are propagated in the magnetic field until they reach calorimeters.
We apply the following movement equations in a cylinder:

$$\frac{d\vec{p}}{dt} = q(\vec{v} \times \vec{B})$$



- The result of this computation is to compute the position @ first calorimeter layer.

ParticlePropagator

```
module ParticlePropagator ParticlePropagator
{
  # input
  set InputArray Delphes/stableParticles

  # outputs
  set OutputArray stableParticles
  set ChargedHadronOutputArray chargedHadrons
  set ElectronOutputArray electrons
  set MuonOutputArray muons

  # radius of the magnetic field coverage, in m
  set Radius 1.29
  # half-length of the magnetic field coverage, in m
  set HalfLength 3.00

  # magnetic field
  set Bz 3.8
}
```

Definition of the cylindral volume of the tracker.

Magnitude of the magnetic field

PS: for ATLAS

```
set Radius 1.15
set HalfLength 3.51
set Bz 2.0
```

Workflow part 1: tracking

Efficiency

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

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

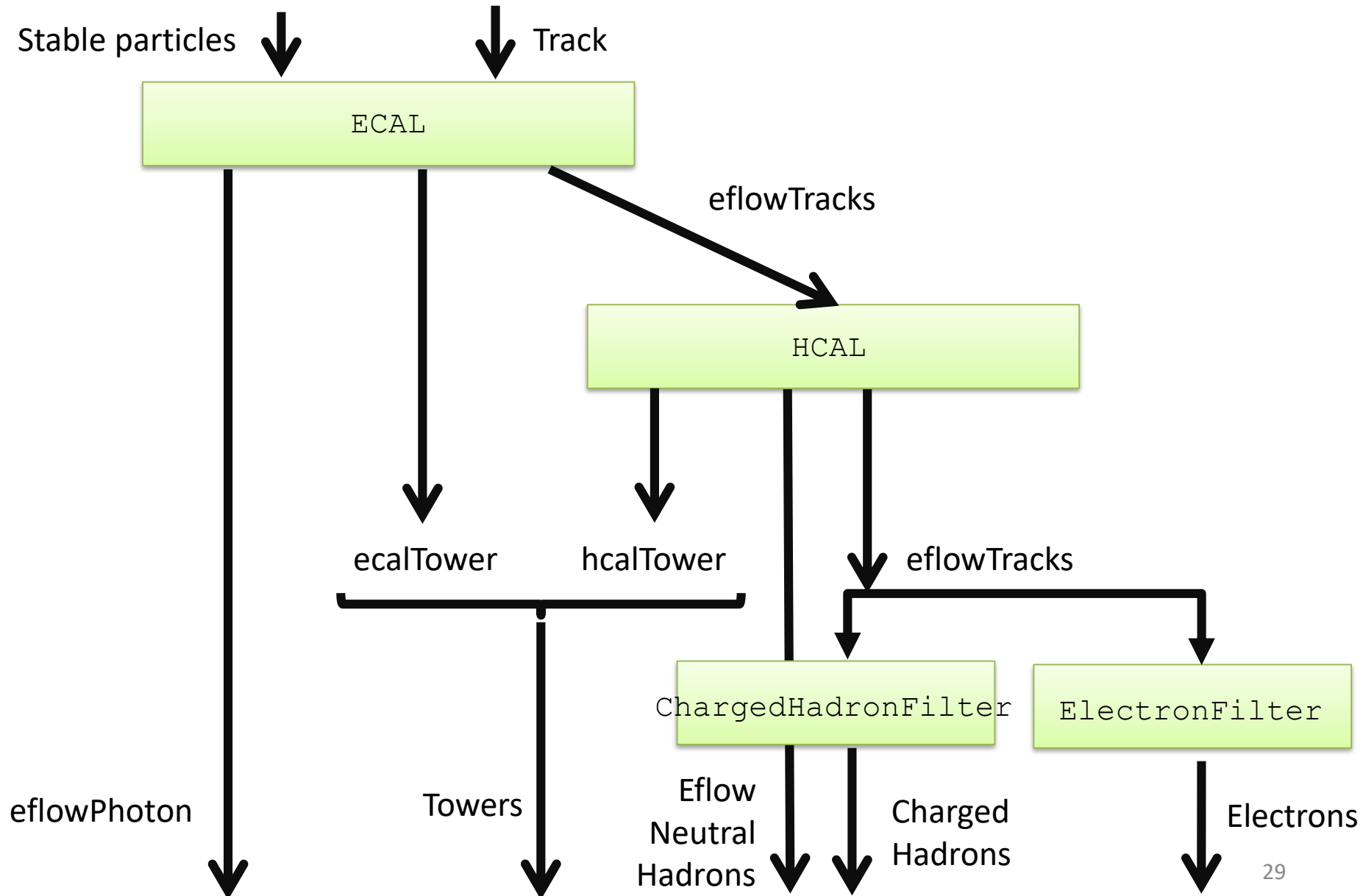
MomentumSmearing

```
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) * sqrt(0.01^2 + pt^2*1.0e-4^2) +
    (abs(eta) > 0.5 && abs(eta) <= 1.5) * (pt > 0.1) * sqrt(0.015^2 + pt^2*1.5e-4^2) +
    (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 0.1) * sqrt(0.025^2 + pt^2*3.5e-4^2)
  }
}
```

Resolution on pT

Workflow part 2: calorimetry

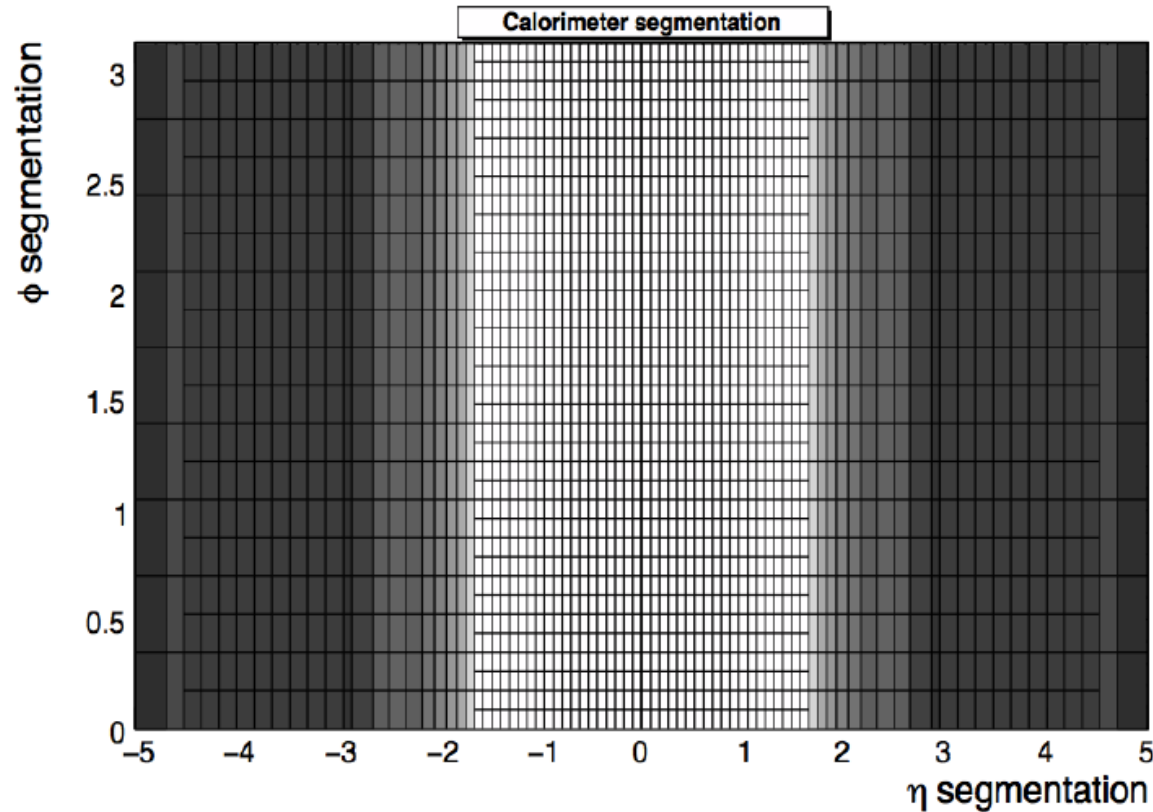


ECAL

HCAL

= inheritance from the same module :
Calorimeter

1) Segmentation of the calorimeter into cells



ECAL

HCAL

= inheritance from the same module :
Calorimeter

2) Energy fraction absorbed by the calorimeter

ECAL case

```
add EnergyFraction {0} {0.0}
# energy fractions for e, gamma and pi0
add EnergyFraction {11} {1.0}
add EnergyFraction {22} {1.0}
add EnergyFraction {111} {1.0}
# energy fractions for muon, neutrinos and neutralinos
add EnergyFraction {12} {0.0}
add EnergyFraction {13} {0.0}
add EnergyFraction {14} {0.0}
add EnergyFraction {16} {0.0}
add EnergyFraction {1000022} {0.0}
add EnergyFraction {1000023} {0.0}
add EnergyFraction {1000025} {0.0}
add EnergyFraction {1000035} {0.0}
add EnergyFraction {1000045} {0.0}
# energy fractions for K0short and Lambda
add EnergyFraction {310} {0.3}
add EnergyFraction {3122} {0.3}
```

HCAL case

```
# default energy fractions {abs(PDG code)} {Fecal Fhcal}
add EnergyFraction {0} {1.0}
# energy fractions for e, gamma and pi0
add EnergyFraction {11} {0.0}
add EnergyFraction {22} {0.0}
add EnergyFraction {111} {0.0}
# energy fractions for muon, neutrinos and neutralinos
add EnergyFraction {12} {0.0}
add EnergyFraction {13} {0.0}
add EnergyFraction {14} {0.0}
add EnergyFraction {16} {0.0}
add EnergyFraction {1000022} {0.0}
add EnergyFraction {1000023} {0.0}
add EnergyFraction {1000025} {0.0}
add EnergyFraction {1000035} {0.0}
add EnergyFraction {1000045} {0.0}
# energy fractions for K0short and Lambda
add EnergyFraction {310} {0.7}
add EnergyFraction {3122} {0.7}
```

WARNING: if you had exotic particle in your sample. declare its EnergyFractions.

ECAL

HCAL

= inheritance from the same module :
Calorimeter

3) Smearing of cell energy

$$\left(\frac{\sigma}{E}\right)^2 = \left(\frac{S(\eta)}{\sqrt{E}}\right)^2 + \left(\frac{N(\eta)}{E}\right)^2 + C(\eta)^2$$

ECAL case

```
set ResolutionFormula {  
    (abs(eta) <= 1.5) * (1+0.64*eta^2) * sqrt(energy^2*0.008^2 + energy*0.11^2 + 0.40^2) +  
    (abs(eta) > 1.5 && abs(eta) <= 2.5) * (2.16 + 5.6*(abs(eta)-2)^2) * sqrt(energy^2*0.008^2 + energy*0.11^2 + 0.40^2) +  
    (abs(eta) > 2.5 && abs(eta) <= 5.0) * sqrt(energy^2*0.107^2 + energy*2.08^2)}  
}
```

[arXiv:1306.2016](#). [Xiv:1502.02701](#)

HCAL case

```
# set HCalResolutionFormula {resolution formula as a function of eta and energy}  
set ResolutionFormula {  
    (abs(eta) <= 3.0) * sqrt(energy^2*0.050^2 + energy*1.50^2) +  
    (abs(eta) > 3.0 && abs(eta) <= 5.0) * sqrt(energy^2*0.130^2 + energy*2.70^2)}  
}
```


+ Min value on energy cell

ECAL

HCAL

= inheritance from the same module :
Calorimeter

4) Two kinds of output collection

- 
- Calorimeter information: **towers**
`ecalTower. hcalTower`

- 
- Calorimeter + tracker information: **particle (or energy) flow**

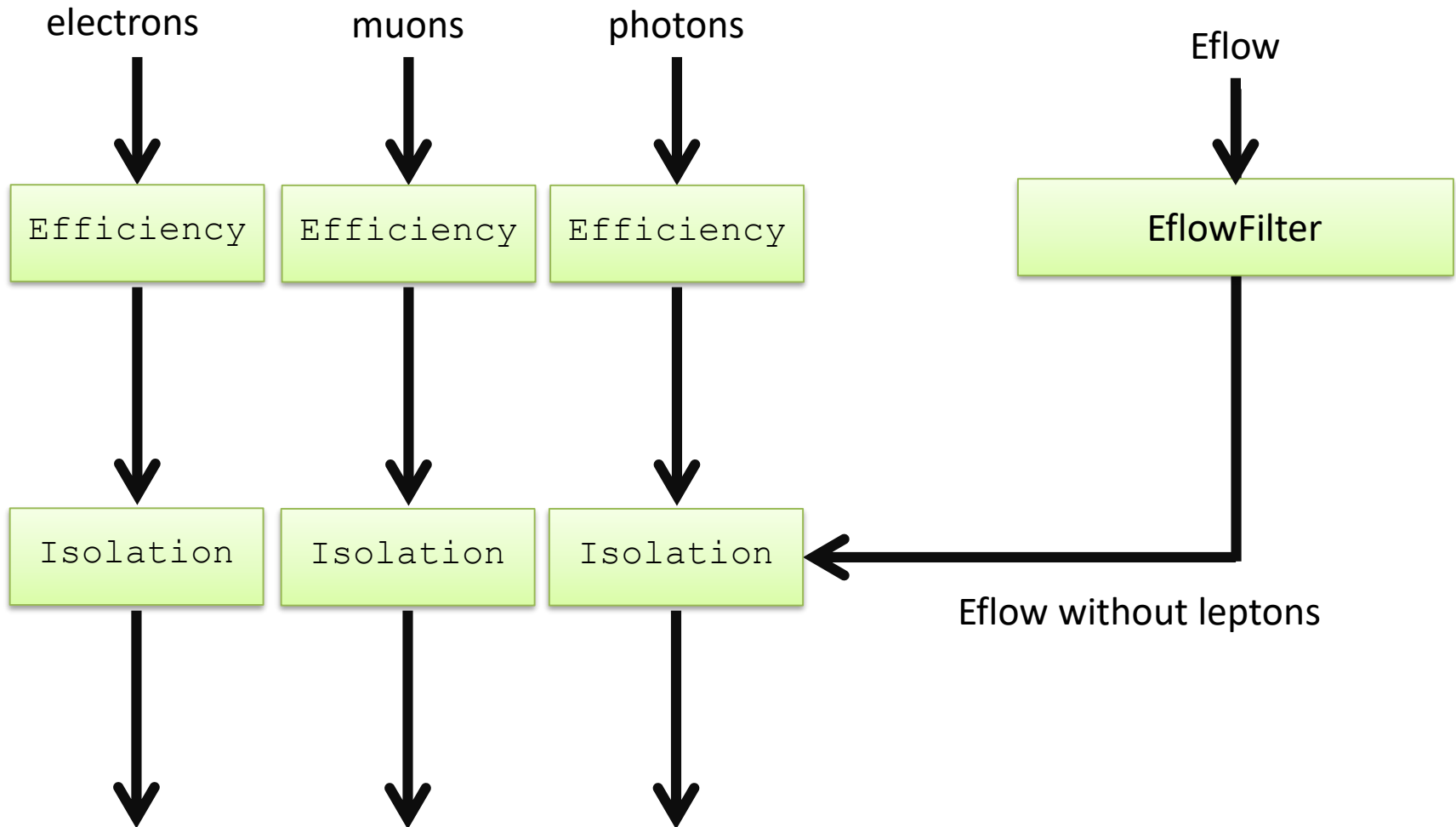
	With track	Without track
ECAL	<code>eflowElectron</code>	<code>eflowPhoton</code>
HCAL	<code>eflowChargedHadron</code>	<code>eflowNeutralHadron</code>



Characteristic of objects are corrected:

- tracking provides good measurement of momenta at low energy
- calorimeter provides good measurement of momenta at high energy

Workflow part 3: e . μ . γ



Workflow part 3: e . μ . γ

Efficiency

Applying efficiency corresponding to identification

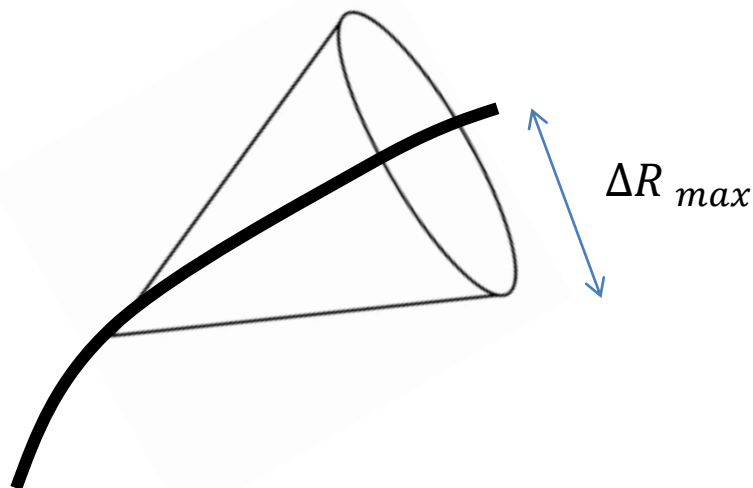
```
module Efficiency MuonEfficiency {  
  set InputArray MuonMomentumSmearing/muons  
  set OutputArray muons  
  
  # efficiency formula for muons  
  set EfficiencyFormula {  
    (pt <= 10.0) * (0.00) +  
    (abs(eta) <= 1.5) * (pt > 10.0) * (0.95) +  
    (abs(eta) > 1.5 && abs(eta) <= 2.4) * (pt > 10.0) * (0.95) +  
    (abs(eta) > 2.4) * (0.00) }  
}
```

```
module Efficiency ElectronEfficiency {  
  set InputArray ElectronFilter/electrons  
  set OutputArray electrons  
  
  # efficiency formula for electrons  
  set EfficiencyFormula {  
    (pt <= 10.0) * (0.00) +  
    (abs(eta) <= 1.5) * (pt > 10.0) * (0.95) +  
    (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 10.0) * (0.85) +  
    (abs(eta) > 2.5) * (0.00) }  
}
```

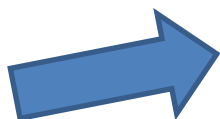
```
module Efficiency PhotonEfficiency {  
  set InputArray ECal/eflowPhotons  
  set OutputArray photons  
  
  # efficiency formula for photons  
  set EfficiencyFormula {  
    (pt <= 10.0) * (0.00) +  
    (abs(eta) <= 1.5) * (pt > 10.0) * (0.95) +  
    (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 10.0) * (0.85) +  
    (abs(eta) > 2.5) * (0.00) }  
}
```

Isolation

Applying an isolation criterion to leptons & photons wrt jets



2 methods



UsePTSum=1

Scalar sum of track PT < threshold



UsePTSum=0 [default]

Scalar sum of track PT / lepton PT
< threshold

Isolation

Applying an isolation criterion to leptons & photons wrt jets

```
module Isolation MuonIsolation
{
  set CandidateInputArray MuonEfficiency/muons
  set IsolationInputArray EFlowFilter/eflow

  set OutputArray muons

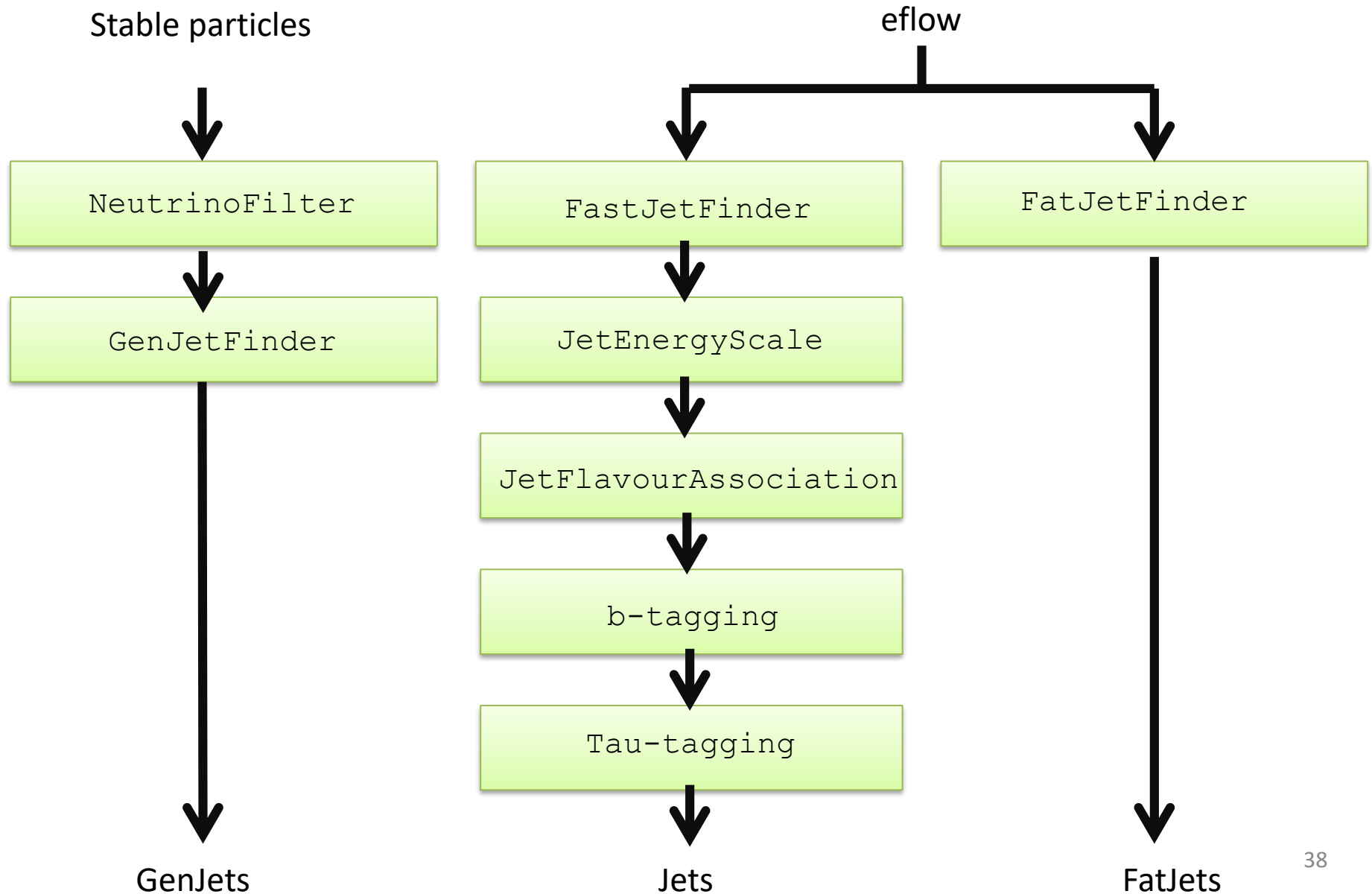
  set DeltaRMax 0.5
  set PTMin 0.5
  set PTRatioMax 0.25
}
```

Size of the isolation cone

Remove muons with low PT

Threshold on the ratio

Workflow part 4: jets



Workflow part 4: jets

Stable particles



NeutrinoFilter



GenJetFinder



GenJets

Remove invisible particle from the stable particles.

Apply a jet –clustering algorithm

- 1 CDFJetClu
- 2 MidPoint
- 3 SIScone
- 4 kt
- 5 Cambridge/Aachen
- 6 antikt

And remove low-PT jets

Reconstructed jets with a perfect calorimeter

Workflow part 4: jets

Stable particles



NeutrinoFilter



GenJetFinder



GenJets

```
module PdgCodeFilter NeutrinoFilter {  
  set InputArray Delphes/stableParticles  
  set OutputArray filteredParticles  
  
  add PdgCode {12}  
  add PdgCode {14}  
  add PdgCode {16}  
  add PdgCode {-12}  
  add PdgCode {-14}  
  add PdgCode {-16}  
}
```

```
module FastJetFinder GenJetFinder {  
  set InputArray NeutrinoFilter/filteredParticles  
  set OutputArray jets  
  
  set JetAlgorithm 6  
  set ParameterR 0.5  
  set JetPTMin 20.0  
}
```


Workflow part 4: jets

eflow



FastJetFinder



JetEnergyScale



JetFlavourAssociation



b-tagging



Tau-tagging



Jets

Apply a jet –clustering algorithm

- 1 CDFJetClu
- 2 MidPoint
- 3 SIScone
- 4 kt
- 5 Cambridge/Aachen
- 6 antikt

And remove low-PT jets

Apply correction to jet energy

Match jets to partons and
determine the « true » b-jets and
taus

B-tagging id and mis-id

tau-tagging id and mis-id

Workflow part 4: jets

eflow



FastJetFinder



JetEnergyScale



JetFlavourAssociation



b-tagging



Tau-tagging



Jets

```
module BTagging BTagging {  
  set JetInputArray JetEnergyScale/jets  
  
  set BitNumber 0  
  
  # default efficiency formula (misidentification rate)  
  add EfficiencyFormula {0} {0.01+0.000038*pt}  
  
  # efficiency formula for c-jets (misidentification rate)  
  add EfficiencyFormula {4} {0.25*tanh(0.018*pt)*(1/(1+ 0.0013*pt))}  
  
  # efficiency formula for b-jets  
  add EfficiencyFormula {5} {0.85*tanh(0.0025*pt)*(25.0/(1+0.063*pt))}  
}
```

arXiv:1211.4462

Workflow part 4: jets

eflow



FatJetFinder

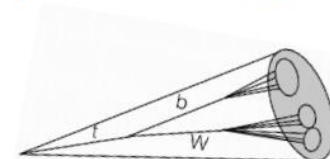


FatJets

```
module FastJetFinder FatJetFinder {  
  set InputArray EFlowMerger/eflow  
  
  set OutputArray jets  
  
  set JetAlgorithm 6  
  set ParameterR 0.8  
  
  set ComputeNsubjettiness 1  
  set Beta 1.0  
  set AxisMode 4  
  
  set ComputeTrimming 1  
  set RTrim 0.2  
  set PtFracTrim 0.05  
  
  set ComputePruning 1  
  set ZcutPrun 0.1  
  set RcutPrun 0.5  
  set RPrun 0.8  
  
  set ComputeSoftDrop 1  
  set BetaSoftDrop 0.0  
  set SymmetryCutSoftDrop 0.1  
  set R0SoftDrop 0.8  
  
  set JetPTMin 200.0  
}
```

Collection
devoted to
boosted objects

tops in a single jet



ParameterR is
bigger than the
normal one.
Therefore these
« fat » jets has a
substructure.

Workflow part 4: jets

eflow



FatJetFinder



FatJets

```
module FastJetFinder FatJetFinder {  
  set InputArray EFlowMerger/eflow  
  
  set OutputArray jets  
  
  set JetAlgorithm 6  
  set ParameterR 0.8  
  
  set ComputeNsubjettiness 1  
  set Beta 1.0  
  set AxisMode 4  
  
  set ComputeTrimming 1  
  set RTrim 0.2  
  set PtFracTrim 0.05  
  
  set ComputePruning 1  
  set ZcutPrun 0.1  
  set RcutPrun 0.5  
  set RPrun 0.8  
  
  set ComputeSoftDrop 1  
  set BetaSoftDrop 0.0  
  set SymmetryCutSoftDrop 0.1  
  set R0SoftDrop 0.8  
  
  set JetPTMin 200.0  
}
```

Probing jet
substructure with N-
subjettiness algo with
N=1.2.3.4.5

Trimming algo

Pruning algo

SoftDrop algo

Workflow part 5: output

Towers. Tracks. eflowPhotons.
eflowTracks. eflowNeutralHadrons.
GenJet. FatJet. MissingEt. ScalarHT.
GenParticle

electrons

muons

photons

jets

UniqueObjectFinder

TreeWriter

UniqueObjectFinder

```
module UniqueObjectFinder UniqueObjectFinder
{
  add InputArray PhotonIsolation/photons photons
  add InputArray ElectronIsolation/electrons electrons
  add InputArray MuonIsolation/muons muons
  add InputArray JetEnergyScale/jets jets
}
```

Jet collection can contain photons. electrons & muons.

→ Cleaning collections by removing redundancies.

Workflow part 5: output

TreeWriter

```
module TreeWriter TreeWriter
{
  add Branch Delphes/allParticles Particle GenParticle
  add Branch TrackMerger/tracks Track Track
  add Branch Calorimeter/towers Tower Tower
  add Branch HCal/eflowTracks EFlowTrack Track
  add Branch ECal/eflowPhotons EFlowPhoton Tower
  add Branch HCal/eflowNeutralHadrons EFlowNeutralHadron Tower
  add Branch GenJetFinder/jets GenJet Jet
  add Branch GenMissingET/momentum GenMissingET MissingET
  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 FatJetFinder/jets FatJet Jet
  add Branch MissingET/momentum MissingET MissingET
  add Branch ScalarHT/energy ScalarHT ScalarHT
}
```

List of all
(temporary or final)
collection of objects
saved in the ROOT files

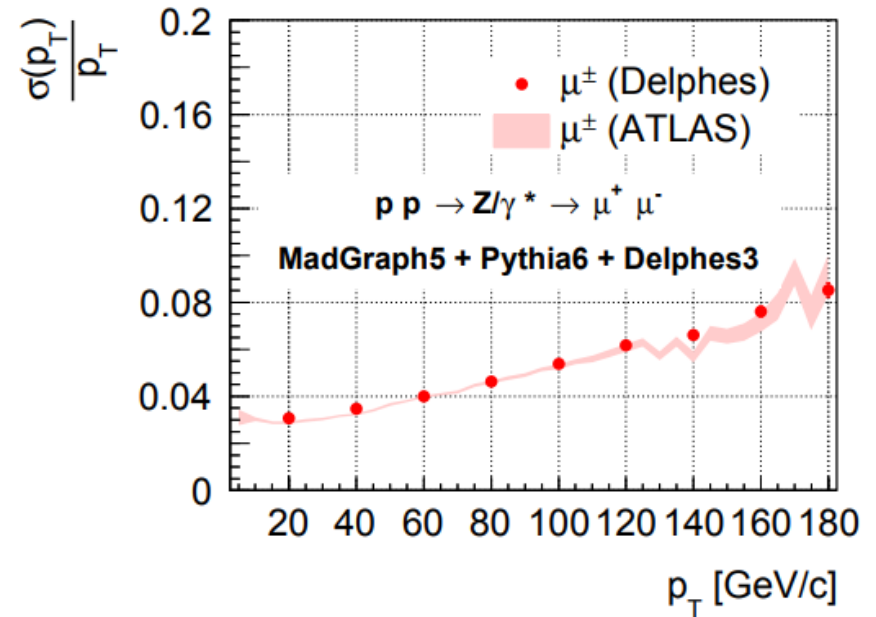
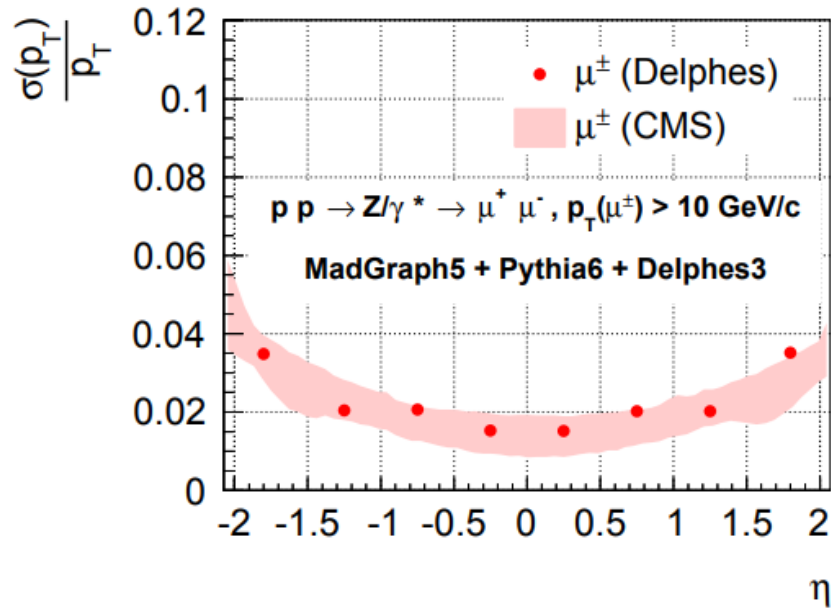
Workflow part 5: output

List of Arrays

Array name	Description
Delphes/allParticles	All generated particles
Delphes/stableParticles	Final state particles (Status==1)
Delphes/partons	Decayed particles or partons produced in shower (Status==2)
ParticlePropagator/stableParticles	All propagated particles
ParticlePropagator/chargedHadrons	Propagated charged hadrons
ParticlePropagator/electrons	Propagated electrons
ParticlePropagator/muons	Propagated muons
ChargedHadronTrackingEfficiency/chargedHadrons	Propagated charged hadrons that pass the efficiency selection
ChargedHadronMomentumSmearing/chargedHadrons	Tracks with momentum smeared according to the charged hadrons momentum resolution
ElectronTrackingEfficiency/electrons	Propagated electrons that pass the efficiency selection
ElectronEnergySmearing/electrons	Tracks with energy smeared according to the electron energy resolution
MuonTrackingEfficiency/muons	Propagated muons that pass the efficiency selection
MuonMomentumSmearing/muons	Tracks with momentum smeared according to the muon momentum resolution
TrackMerger/tracks	Combination of charged hadrons and electrons
Calorimeter/towers	All calorimeter towers
Calorimeter/photons	Calorimeter towers associated with the photons
Calorimeter/eflowTracks	Tracks output from the energy flow algorithm
Calorimeter/eflowPhotons	Photon output from the energy flow algorithm
Calorimeter/eflowNeutralHadrons	Neutral hadron output from the energy flow algorithm
EFLOWMerger/eflow	Combination of tracks, calorimeter towers and muons required for jet finding
ElectronEfficiency/electrons	Electrons that pass the efficiency selection
ElectronIsolation/electrons	Isolated electrons
PhotonEfficiency/photons	Photons that pass the efficiency selection
PhotonIsolation/photons	Isolated photons
MuonEfficiency/muons	Muons that pass the efficiency selection
MuonIsolation/muons	Isolated muons
FastJetFinder/jets	Reconstructed jets
MissingET/momentum	Missing transverse energy
ScalarHT/energy	Scalar sum of transverse momenta and energy of all reconstructed objects
UniqueObjectFinder/photons	Uniquely identified photons
UniqueObjectFinder/electrons	Uniquely identified electrons
UniqueObjectFinder/jets	Uniquely identified jets

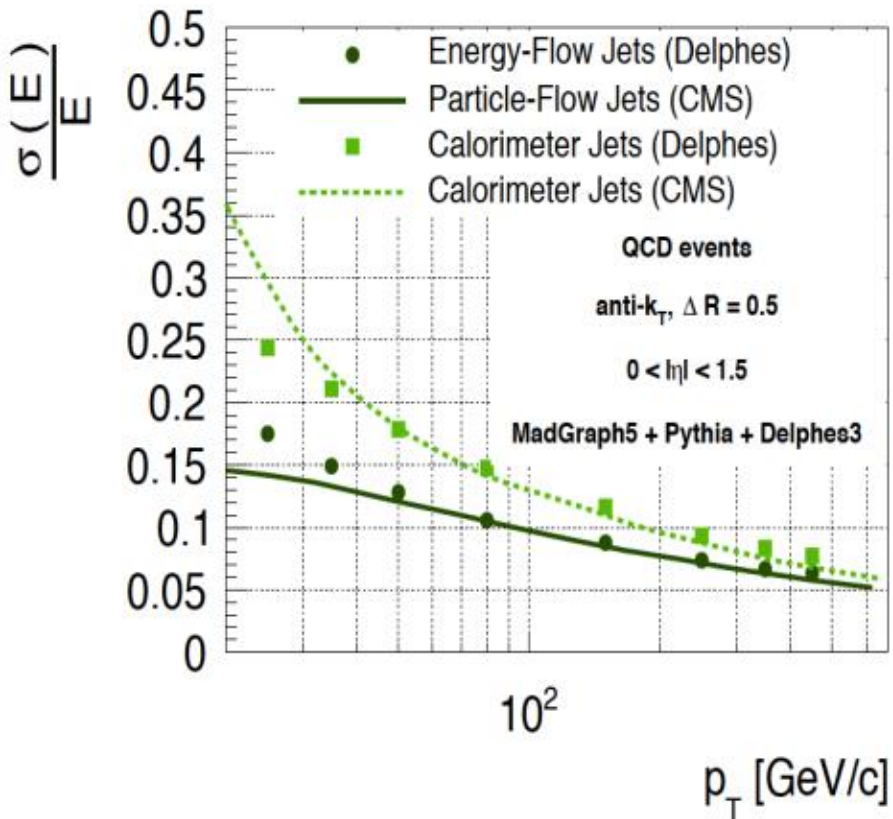
4. Validation and limitations of Delphes

PT resolution of reconstructed muons

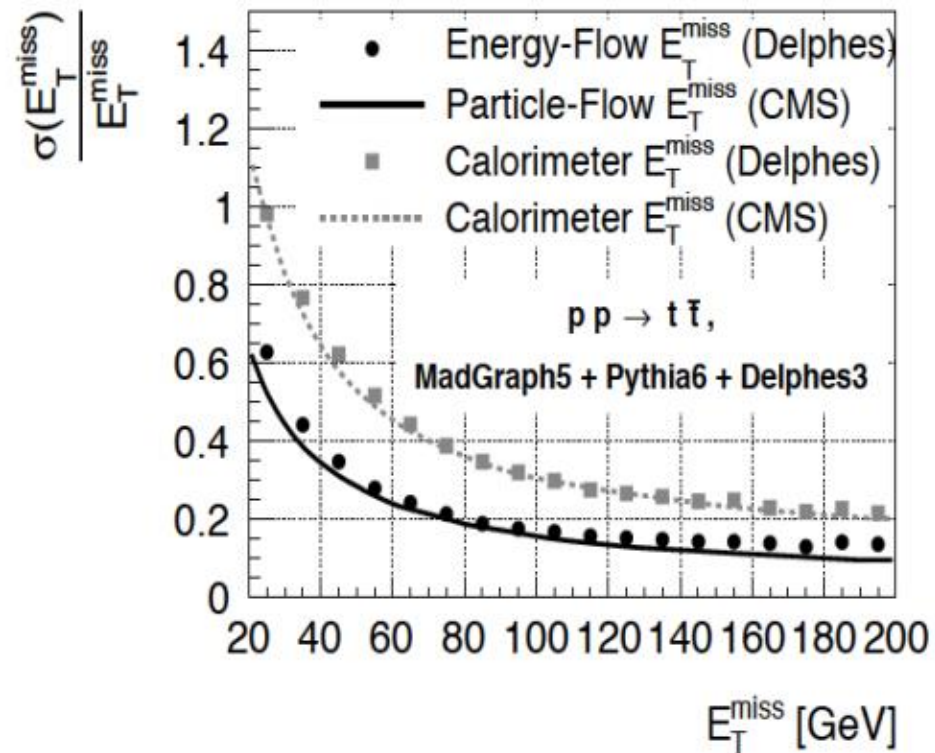


ParticleFlow validation

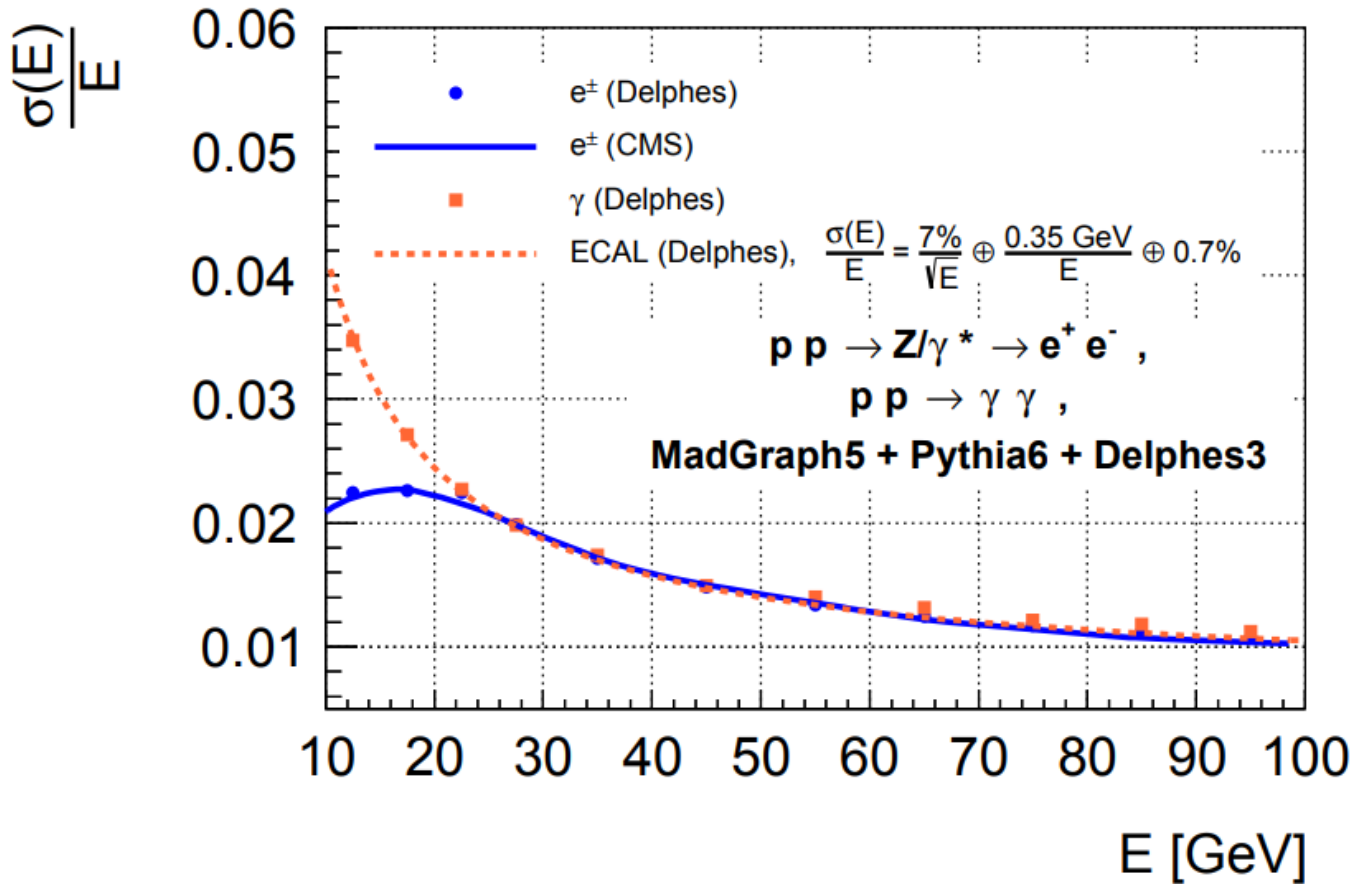
Jet energy resolution



MET resolution



Energy resolution of reconstructed electrons & photons



- The **simulation of the trigger** (online selection) is missing.
BUT usually the offline selection includes the effects of the trigger.
- There is no tracker simulation:
 - **No reconstructed vertices**
 - **No fake tracks**
 - **No track quality**
- **No noise in the calorimeters**
- **No photon conversion** (but included for LHCb simulation)
- **No fake muons. electrons or photons**
- **Does not convient for exotics topology.**
But some developments are ongoing. in particular for long-lived particles:
 - Displaced tracks: OK
 - Displaced vertices: feasible
 - Displaced jets: to do

- **The Delphes development model is community-based.**

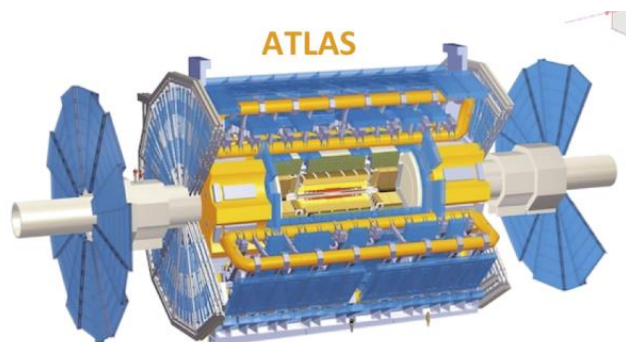
People are encouraged to:

- Tune their detector cards according to their usage
- Develop their own modules
- Modify the code if necessary

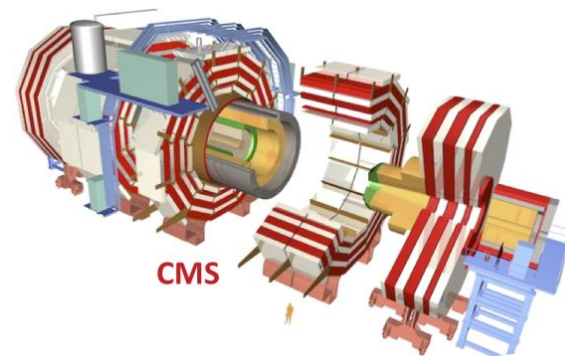
- **Proliferation of tunes of Delphes. Some example:**

- **CMS or ATLAS tunes of Delphes** [private]
- **Rivet tune of Delphes:**
 - Improving ATLAS simulation realism
 - Adding a lot of tags in the data format
- **CheckMate tune**
- **MadAnalysis 5 tune(s) of the Delphes cards**

The several tunes of Delphes



Detector very-fast-simulation



old
way

Delphes MA5-Tune

Special tuning of the
Delphes 3.0 package
provided by MadAnalysis 5



new
way

from
MA5
v1.2

Delphes + MA5 card

Official Delphes release using
special CMS/ATLAS detector
cards provided by MadAnalysis 5

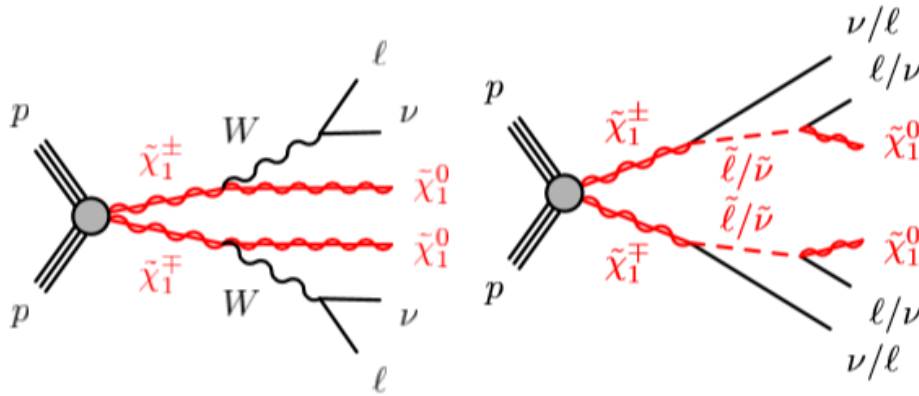
- Reducing the ROOT size.
 - Lepton & photon isolation done @ analysis level.
 - More realistic parametrization of the b-tagging(mis-)efficiency @ analysis level.
 - More info on generated particles.
-
- Most of the features implemented in the official Delphes release.
 - Other features are encapsulated into external Delphes modules.
 - Lepton & photon isolation always done @ analysis level + **improvement**.

5. Application to the recasting of the analysis SUSY-2018-32

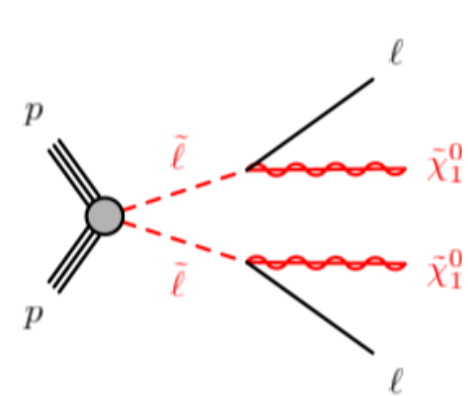
Reminder of SUSY-2018-32

ATLAS analysis : $\sqrt{s}=13$ TeV proton-proton collision. $L^{int}=139 \text{ fb}^{-1}$

Signal: SUSY simplified models



Charginos production



Sleptons production

Final states: $ee/\mu\mu + \text{MET}$

Public page: <https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PAPERS/SUSY-2018-32/>

Benjamin Fuks explained you how to implement this analysis with MA5 expert mode.

Starting point: default ATLAS card for Delphes

Section 3 – ATLAS detector

- Main information on the ATLAS detector.
- The default ATLAS card describes the detector.
- Sanity check can be done.

3 ATLAS detector

The ATLAS detector [31] at the LHC is a general-purpose detector with a forward–backward symmetric cylindrical geometry and an almost complete coverage in solid angle around the collision point.¹ It consists of an inner tracking detector surrounded by a thin superconducting solenoid, electromagnetic and hadronic calorimeters, and a muon spectrometer incorporating three large superconducting toroid magnets.

The inner-detector (ID) system is immersed in a 2 T axial magnetic field produced by the solenoid and provides charged-particle tracking in the range $|\eta| < 2.5$. It consists of a high-granularity silicon pixel detector, a silicon microstrip tracker and a transition radiation tracker, which enables radially extended track reconstruction up to $|\eta| = 2.0$. The transition radiation tracker also provides electron identification information. During the first LHC long shutdown, a new tracking layer, known as the Insertable *B*-Layer [32, 33], was added with an average sensor radius of 33 mm from the beam pipe to improve tracking and *b*-tagging performance.

Section 5: object identification → Baseline electrons

Baseline electron candidates are reconstructed using clusters of energy deposits in the electromagnetic calorimeter that are matched to an ID track. They are required to satisfy a *Loose* likelihood-based identification requirement [39], and to have $p_T > 10$ GeV and $|\eta| < 2.47$. They are also required to be

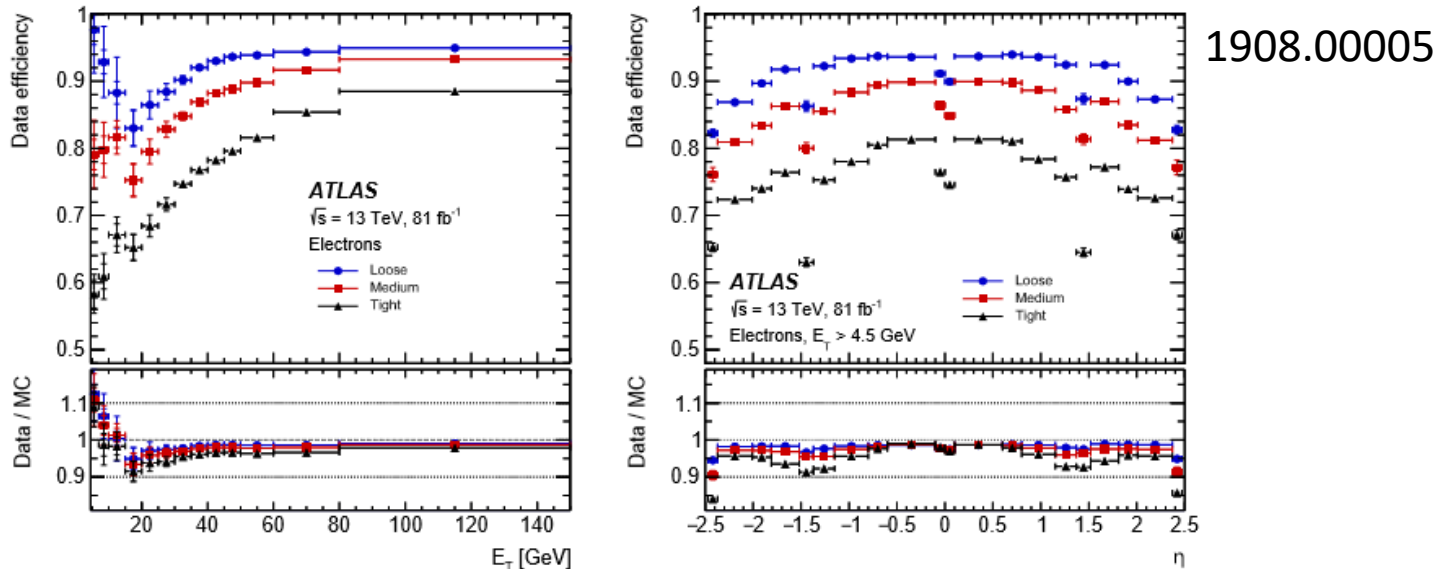
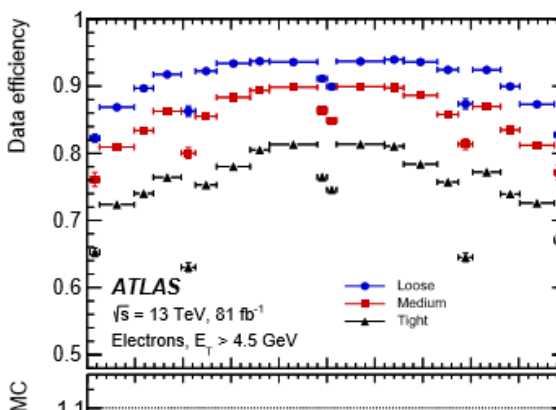
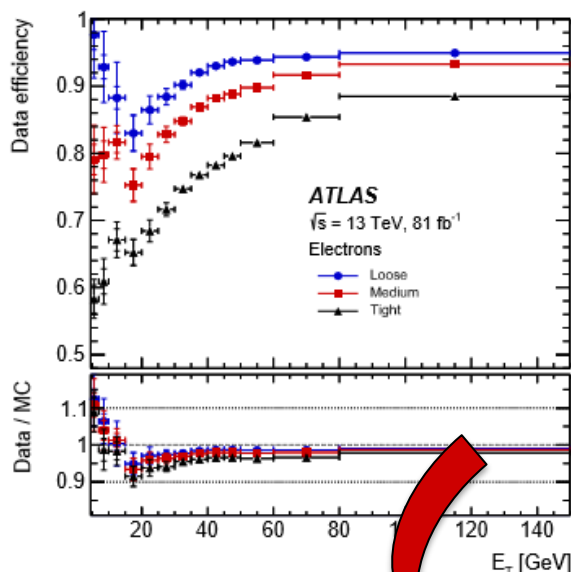


Figure 17: The electron identification efficiency in $Z \rightarrow ee$ events in data as a function of E_T (left) and as a function of η (right) for the Loose, Medium and Tight operating points. The efficiencies are obtained by applying data-to-simulation efficiency ratios measured in $J/\psi \rightarrow ee$ and $Z \rightarrow ee$ events to $Z \rightarrow ee$ simulation. The inner uncertainties are statistical and the total uncertainties are the statistical and systematic uncertainties in the data-to-simulation efficiency ratio added in quadrature. For both plots, the bottom panel shows the data-to-simulation ratios.

Section 5: object identification → Baseline electrons

Baseline electron candidates are reconstructed using clusters of energy deposits in the electromagnetic calorimeter that are matched to an ID track. They are required to satisfy a *Loose* likelihood-based identification requirement [39], and to have $p_T > 10$ GeV and $|\eta| < 2.47$. They are also required to be



1908.00005

Figure 17: The electron identification efficiency function of η (right) for the Loose, Medium and Tight selection. The data-to-simulation efficiency ratios measured in J/ψ inner uncertainties are statistical and the total uncertainty data-to-simulation efficiency ratio added in quadrature. For ratios.

```
#####
# Electron efficiency
#####

module Efficiency ElectronEfficiency {
  set InputArray ElectronFilter/electrons
  set OutputArray electrons

  # efficiency formula for electrons
  set EfficiencyFormula {
    (abs(eta) <= 1.5) * (pt <= 10.0) * (0.00) +
    (abs(eta) > 1.5 && abs(eta) <= 2.5) * (pt > 10.0) * (0.95) +
    (abs(eta) > 2.5) * (0.00)
  }
}
```

Section 5: object identification → Baseline muons

Baseline muon candidates are reconstructed in the pseudorapidity range $|\eta| < 2.7$ from MS tracks matching ID tracks. They are required to have $p_T > 10$ GeV, to be within $|z_0 \sin \theta| = 0.5$ mm of the primary vertex and to satisfy the *Medium* identification requirements defined in Ref. [40]. The *Medium* identification criterion defines requirements on the number of hits in the different ID and MS subsystems, and on the

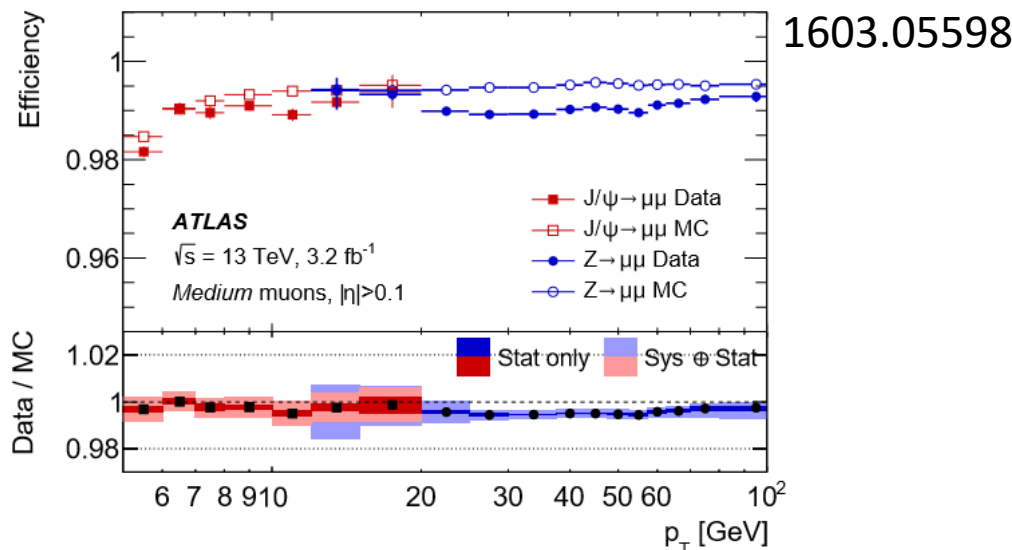


Figure 6: Reconstruction efficiency for the *Medium* muon selection as a function of the p_T of the muon, in the region $0.1 < |\eta| < 2.5$ as obtained with $Z \rightarrow \mu\mu$ and $J/\psi \rightarrow \mu\mu$ events. The error bars on the efficiencies indicate the statistical uncertainty. The panel at the bottom shows the ratio of the measured to predicted efficiencies, with statistical and systematic uncertainties.

Section 5: object identification → Baseline muons

Baseline muon candidates are reconstructed in the pseudorapidity range $|\eta| < 2.7$ from MS tracks matching ID tracks. They are required to have $p_T > 10$ GeV, to be within $|z_0 \sin \theta| = 0.5$ mm of the primary vertex and to satisfy the *Medium* identification requirements defined in Ref. [40]. The *Medium* identification criterion defines requirements on the number of hits in the different ID and MS subsystems, and on the

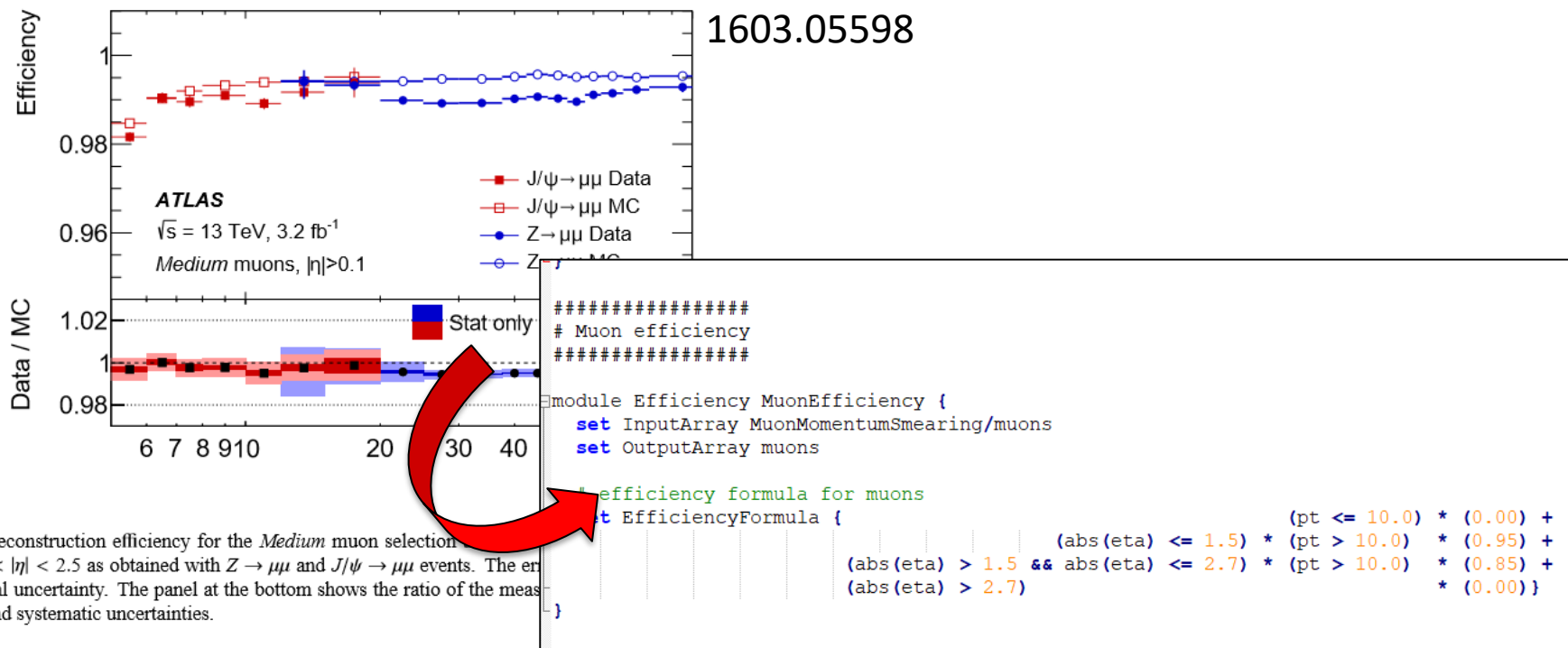


Figure 6: Reconstruction efficiency for the *Medium* muon selection in the region $0.1 < |\eta| < 2.5$ as obtained with $Z \rightarrow \mu\mu$ and $J/\psi \rightarrow \mu\mu$ events. The error bars show the statistical uncertainty. The panel at the bottom shows the ratio of the measured efficiency to the Monte Carlo prediction, with the statistical and systematic uncertainties.

Section 5: object identification → isolation of leptons

Isolation criteria are applied to signal electrons and muons. The scalar sum of the p_T of tracks inside a variable-size cone around the lepton (excluding its own track), must be less than 15% of the lepton p_T . The track isolation cone size for electrons (muons) $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$ is given by the minimum of

- Isolation performed in the analysis code
- Isolation modules must be removed

```
#####  
# Electron isolation  
#####  
  
module Isolation ElectronIsolation {  
  set CandidateInputArray ElectronEfficiency/electrons  
  set IsolationInputArray EFlowFilter/eflow  
  
  set OutputArray electrons  
  
  set DeltaRMax 0.5  
  set PTMin 0.5  
  set PTRatioMax 0.12  
}
```

```
#####  
# Muon isolation  
#####  
  
module Isolation MuonIsolation {  
  set CandidateInputArray MuonEfficiency/muons  
  set IsolationInputArray EFlowFilter/eflow  
  
  set OutputArray muons  
  
  set DeltaRMax 0.5  
  set PTMin 0.5  
  set PTRatioMax 0.25  
}
```

Section 5: object identification → Jet definition

Jets are reconstructed from topological clusters of energy in the calorimeter [86] using the anti- k_t jet clustering algorithm [87] as implemented in the FastJet package [88], with a radius parameter $R = 0.4$. The reconstructed jets are then calibrated by the application of a jet energy scale derived from 13 TeV data and simulation [89]. Only jet candidates with $p_T > 20$ GeV and $|\eta| < 2.4$ are considered,⁴ although jets with $|\eta| < 4.9$ are included in the missing transverse momentum calculation and are considered when applying the procedure to remove reconstruction ambiguities, which is described later in this Section.

```
#####  
# Jet finder  
#####  
  
module FastJetFinder FastJetFinder {  
  set InputArray Calorimeter/towers  
  
  set OutputArray jets  
  
  # algorithm: 1 CDFJetClu, 2 MidPoint, 3 SIScone, 4 kt, 5 Cambridge/Aachen, 6 antikt  
  set JetAlgorithm 6  
  set ParameterR 0.4  
  
  set JetPTMin 10.0  
}
```

← Do not set 20 GeV (because we are before the JES correction)

Section 5: object identification → Jet definition

Jets are reconstructed from topological clusters of energy in the calorimeter [86] using the anti- k_t jet clustering algorithm [87] as implemented in the FastJet package [88], with a radius parameter $R = 0.4$. The reconstructed jets are then calibrated by the application of a jet energy scale derived from 13 TeV data and simulation [89]. Only jet candidates with $p_T > 20$ GeV and $|\eta| < 2.4$ are considered,⁴ although jets with $|\eta| < 4.9$ are included in the missing transverse momentum calculation and are considered when applying the procedure to remove reconstruction ambiguities, which is described later in this Section.

```
#####  
# Jet Energy Scale  
#####  
  
module EnergyScale JetEnergyScale {  
  set InputArray FastJetFinder/jets  
  set OutputArray jets  
  
  # scale formula for jets  
  set ScaleFormula { sqrt( (3.0 - 0.2*(abs(eta)))^2 / pt + 1.0 ) }  
}
```

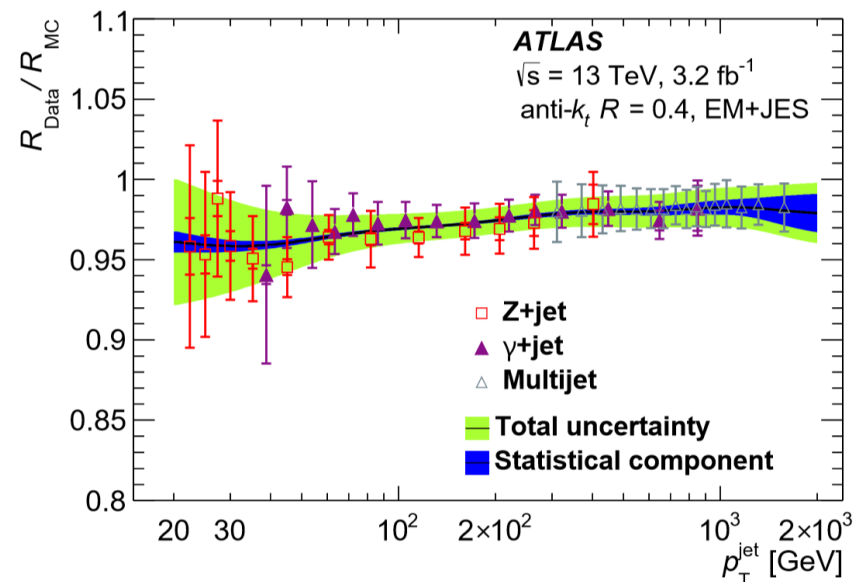
Default
parametrization in
the Delphes card
→ To test first

Section 5: object identification → Jet definition

Jets are reconstructed from topological clusters of energy in the calorimeter [86] using the anti- k_t jet clustering algorithm [87] as implemented in the FastJet package [88], with a radius parameter $R = 0.4$. The reconstructed jets are then calibrated by the application of a jet energy scale derived from 13 TeV data and simulation [89]. Only jet candidates with $p_T > 20$ GeV and $|\eta| < 2.4$ are considered,⁴ although jets with $|\eta| < 4.9$ are included in the missing transverse momentum calculation and are considered when applying the procedure to remove reconstruction ambiguities, which is described later in this Section.

Dependence in p_T
(no mapping $p_T \times \eta$ available)
→ To implement if significant disagreements are observed.

WARNING: definitions of JES in Delphes and ATLAS are different.



Section 5: object identification → Jet definition

Jets are reconstructed from topological clusters of energy in the calorimeter [86] using the anti- k_t jet clustering algorithm [87] as implemented in the FastJet package [88], with a radius parameter $R = 0.4$. The reconstructed jets are then calibrated by the application of a jet energy scale derived from 13 TeV data and simulation [89]. Only jet candidates with $p_T > 20$ GeV and $|\eta| < 2.4$ are considered,⁴ although jets with $|\eta| < 4.9$ are included in the missing transverse momentum calculation and are considered when applying the procedure to remove reconstruction ambiguities, which is described later in this Section.

MET automatically handled with DELPHES

Implemented in the analysis source file

Section 5: object identification → b-tagging

The MV2C10 boosted decision tree algorithm [41] identifies jets containing b -hadrons (b -jets') by using quantities such as the impact parameters of associated tracks, and well-reconstructed secondary vertices. A selection that provides 85% efficiency for tagging b -jets in simulated $t\bar{t}$ events is used. The corresponding rejection factors against jets originating from c -quarks, from τ -leptons, and from light quarks and gluons in the same sample at this working point are 2.7, 6.1 and 25, respectively.

```
module BTagging BTagging {  
  set JetInputArray JetEnergyScale/jets  
  
  set BitNumber 0  
  
  # default efficiency formula (misidentification rate)  
  add EfficiencyFormula {0} {0.04}  
  
  # efficiency formula for c-jets (misidentification rate)  
  add EfficiencyFormula {4} {0.37}  
  
  # efficiency formula for b-jets  
  add EfficiencyFormula {5} {0.85}  
}
```

Very simplified method

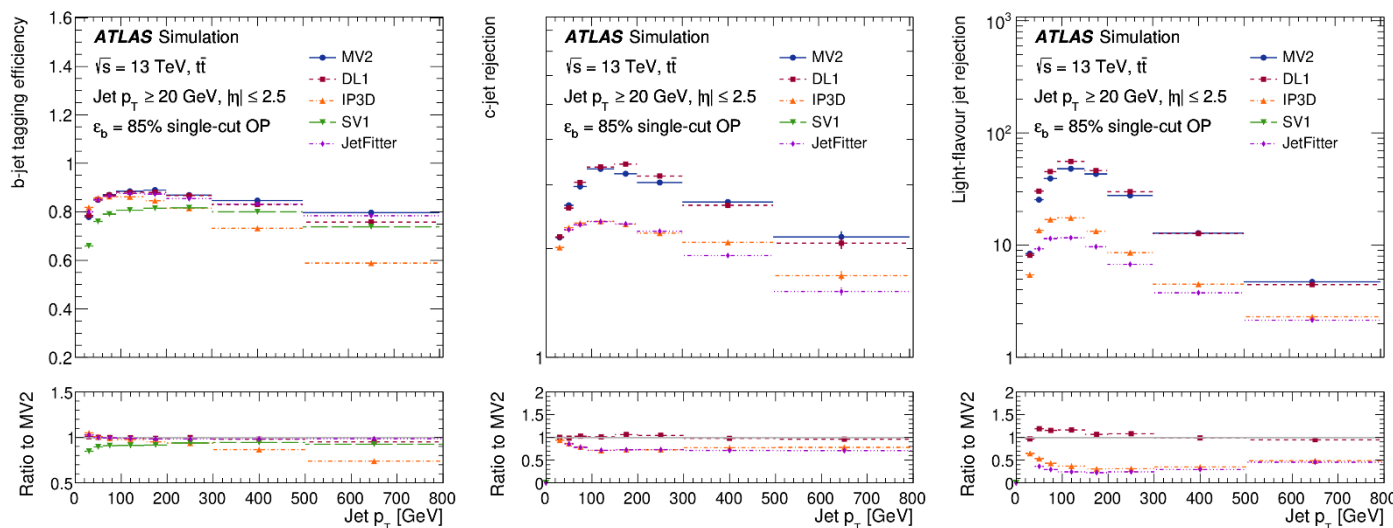
Section 5: object identification → b-tagging

The MV2C10 boosted decision tree algorithm [41] identifies jets containing b -hadrons (' b -jets') by using quantities such as the impact parameters of associated tracks, and well-reconstructed secondary vertices. A selection that provides 85% efficiency for tagging b -jets in simulated $t\bar{t}$ events is used. The corresponding rejection factors against jets originating from c -quarks, from τ -leptons, and from light quarks and gluons in the same sample at this working point are 2.7, 6.1 and 25, respectively.

arXiv:1907.05120

Devoted ATLAS page with extra plots

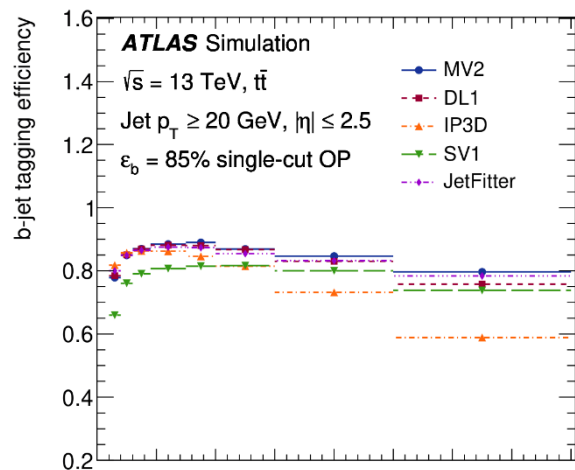
<https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/PAPERS/FTAG-2018-01/>



Figures 4c. 4b. 4a
+16a

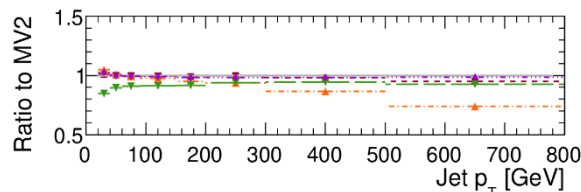
Section 5: object identification → b-tagging

The MV2C10 boosted decision tree algorithm [41] identifies jets containing b -hadrons (b -jets') by using quantities such as the impact parameters of associated tracks, and well-reconstructed secondary vertices. A selection that provides 85% efficiency for tagging b -jets in simulated $t\bar{t}$ events is used. The corresponding rejection factors against jets originating from c -quarks, from τ -leptons, and from light quarks and gluons in the same sample at this working point are 2.7, 6.1 and 25, respectively.



Point extraction for MV2C10

PT min	PT mean	PT max	Efficiency
20	30	40	0.7765
40	50	60	0.8481
60	75	90	0.8696
90	120	150	0.8910
150	250	300	0.8696
300	400	500	0.8481
500	650	800	0.7979



What about underflow?
What about overflow?

Section 5: object identification → b-tagging

The MV2C10 boosted decision tree algorithm [41] identifies jets containing b -hadrons (b -jets') by using quantities such as the impact parameters of associated tracks, and well-reconstructed secondary vertices. A selection that provides 85% efficiency for tagging b -jets in simulated $t\bar{t}$ events is used. The corresponding rejection factors against jets originating from c -quarks, from τ -leptons, and from light quarks and gluons in the same sample at this working point are 2.7, 6.1 and 25, respectively.

Point extraction for MV2C10

PT min	PT mean	PT max	Efficiency
20	30	40	0.7765
40	50	60	0.8481
60	75	90	0.8696
90	120	150	0.8910
150	250	300	0.8696
300	400	500	0.8481
500	650	800	0.7979

More accurate method

```
# efficiency formula for b-jets
add EfficiencyFormula {5} { (pt<20.0)*(0.00) +
                             (pt>=20.0 && pt<40 )*0.7765) +
                             (pt>=40.0 && pt<60 )*0.8481) +
                             (pt>=60.0 && pt<90 )*0.8696) +
                             (pt>=90.0 && pt<150)*0.8910) +
                             (pt>=150.0 && pt<300)*0.8696) +
                             (pt>=300.0 && pt<500)*0.8481) +
                             (pt>=500.0 && pt<800)*0.7979) +
                             (pt>=800.0)*0.7979) }
```

Section 5: object identification → double-counting removal

To avoid the double counting of analysis baseline objects, a procedure to remove reconstruction ambiguities is applied as follows:

- jet candidates within $\Delta R' = \sqrt{\Delta y^2 + \Delta \phi^2} = 0.2$ of an electron candidate are removed;
- jets with fewer than three tracks that lie within $\Delta R' = 0.4$ of a muon candidate are removed;
- electrons and muons within $\Delta R' = 0.4$ of the remaining jets are discarded, to reject leptons from the decay of b - or c -hadrons;
- electron candidates are rejected if they are found to share an ID track with a muon.

```
#####  
# Find uniquely identified photons/electrons/tau/jets  
#####  
  
module UniqueObjectFinder UniqueObjectFinder {  
  # earlier arrays take precedence over later ones  
  # add InputArray InputArray OutputArray  
  add InputArray PhotonIsolation/photons photons  
  add InputArray ElectronIsolation/electrons electrons  
  add InputArray MuonIsolation/muons muons  
  add InputArray JetEnergyScale/jets jets  
}
```

- Remove the module UniqueObjectFinder
- To implement in the analysis source code



Is there any invisible
exotic particle in your
final state ?

Yes. the neutralino 1

Usually with PDG-ID = 1 000 022

Energy fraction absorbed by the calorimeters ECAL & HCAL

ECAL case

```
add EnergyFraction {0} {0.0}
# energy fractions for e, gamma and pi0
add EnergyFraction {11} {1.0}
add EnergyFraction {22} {1.0}
add EnergyFraction {111} {1.0}
# energy fractions for muon, neutrinos and neutralinos
add EnergyFraction {12} {0.0}
add EnergyFraction {13} {0.0}
add EnergyFraction {14} {0.0}
add EnergyFraction {16} {0.0}
add EnergyFraction {1000022} {0.0}
add EnergyFraction {1000023} {0.0}
add EnergyFraction {1000025} {0.0}
add EnergyFraction {1000035} {0.0}
add EnergyFraction {1000045} {0.0}
# energy fractions for K0short and Lambda
add EnergyFraction {310} {0.3}
add EnergyFraction {3122} {0.3}
```

HCAL case

```
# default energy fractions {abs(PDG code)} {Fecal Fhcal}
add EnergyFraction {0} {1.0}
# energy fractions for e, gamma and pi0
add EnergyFraction {11} {0.0}
add EnergyFraction {22} {0.0}
add EnergyFraction {111} {0.0}
# energy fractions for muon, neutrinos and neutralinos
add EnergyFraction {12} {0.0}
add EnergyFraction {13} {0.0}
add EnergyFraction {14} {0.0}
add EnergyFraction {16} {0.0}
add EnergyFraction {1000022} {0.0}
add EnergyFraction {1000023} {0.0}
add EnergyFraction {1000025} {0.0}
add EnergyFraction {1000035} {0.0}
add EnergyFraction {1000045} {0.0}
# energy fractions for K0short and Lambda
add EnergyFraction {310} {0.7}
add EnergyFraction {3122} {0.7}
```

And NeutrinoFilter module if you are interested by GenJets.



**Is there any invisible
exotic particle in your
final state ?**

Yes. the neutralino 1

Usually with PDG-ID = 1 000 022

And NeutrinoFilter module if you are interested by GenJets

```
#####  
# Neutrino Filter  
#####  
  
module PdgCodeFilter NeutrinoFilter {  
  
    set InputArray Delphes/stableParticles  
    set OutputArray filteredParticles  
  
    set PTMin 0.0  
  
    add PdgCode {12}  
    add PdgCode {14}  
    add PdgCode {16}  
    add PdgCode {-12}  
    add PdgCode {-14}  
    add PdgCode {-16}  
  
}
```

If the neutralino 1 is not added to this list. it will be considered as a possible constituent for GenJets.

ADVICE

Simplyfing the ROOT output

- Selecting only the collections that you need

```
#####  
# ROOT tree writer  
#####  
  
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 HCal/eflowTracks EFlowTrack Track  
  add Branch ECal/eflowPhotons EFlowPhoton Tower  
  add Branch HCal/eflowNeutralHadrons EFlowNeutralHadron Tower  
  
  add Branch GenJetFinder/jets GenJet Jet  
  add Branch GenMissingET/momentum GenMissingET MissingET  
  
  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  
}
```

Summary

In order to recast LHC analyses. we need a **very-fast & realistic detector simulation**.

Delphes (current release: 3.4.2) suits very well:

- Generic simulation. in particular ATLAS & CMS can handled.
- Modular architecture & initially community- based
- No code to develop for tuning Delphes for your analysis → configuration card

In the context of this workshop. it is important that the students:

- find a balance between simulation realism and Delphes abilities
- tune the settings of the Delphes card according to their analysis.
- provide their tuned detector card to the supervisors

Missing point in this talk: pile-up simulation?

- Do we need to take into account pile-up into our simulation?
- See you tomorrow for the sequel of this talk.