Pavel Demin, Alexandre Mertens, Michele Selvaggi

Université catholique de Louvain (UCL)
Center for Particle Physics and Phenomenology (CP3)

JHEP 02 (2014) 057

Tutorial
8 May 2014
Outline of tutorial

• General Delphes introduction (~ 20')
  - brief overview
  - new features
    track counting b-tagging
    simple calorimeters
    N-subjettiness

• Hands-on session (~ 1h 30')
  - Higgs invariant mass in bb final state (FCC-ee)
    run Delphes, glance at output tree
    run simple macro on output
    modify configuration card (change calo granularity, resolution)
    write a Delphes module
  - Jet substructure and pile-up (FCC-hh)
    run a simple analysis with pile-up, see impact of pile-up on
    N-subjettiness for two jet topologies (light and top jets)
Outline of tutorial

• **Open discussion** *(remaining time)*

  - un-addressed topics:
    
    → using Delphes as an external library

    → possible improvements:
      - transverse energy sharing
      - longitudinal segmentation
      - frozen showers
      - calorimeter clustering (and implications on particle-flow)

    → suggestions from the audience?
Overview
The Delphes project: A bit of history

- Delphes project started back in 2007 at UCL as a side project to allow quick feasibility studies

- Since 2009, its development is **community-based**
  - ticketing system for improvement and bug-fixes
    → user proposed patches
  - **Quality control** and **core development** is done at the UCL

- In 2013, **DELPHES 3** was released:
  - modular software
  - new features
  - also included in MadGraph suite

- **Widely** tested and used by the community (pheno, Snowmass, CMS ECFA efforts, etc...)


The Delphes project: Delphes in a nutshell

- **Delphes** is a **modular framework** that simulates the response of a multipurpose detector in a **parameterized** fashion.

- **Includes:**
  - pile-up
  - charged particle **propagation** in magnetic field
  - electromagnetic and hadronic **calorimeters**
  - **muon** system

- **Provides:**
  - leptons (electrons and muons)
  - photons
  - jets and missing transverse energy (particle-flow)
  - taus and b's
The Delphes project: I/O and configurations

- **modular** C++ code

- **Uses**
  - ROOT classes [Comp. Phys. C. 180 (2009) 2499]

- **Input**
  - Pythia/Herwig output (HepMC,STDHEP)
  - LHE (MadGraph/MadEvent)
  - ProMC

- **Configuration file**
  - Define geometry
  - Resolution/reconstruction/selection criteria
  - Output object collections

- **Output**
  - ROOT trees
New features
Track Counting B-tagging
Parametric b-tagging:
- Check if there is a b,c-quark in the cone of size DeltaR
- Apply a parametrized Efficiency (PT, eta)

→ perfectly reproduces existing performances
→ not predictive
Track counting b-tagging

- b-jets have charged constituents with large impact parameter
- for each track inside jet, compute $\text{sig}(IP) = \frac{IP}{\sigma(IP)}$ in 2D (simpler)
- count tracks with $\text{sig}(IP) > \text{cut}$
- if find more than 2 (high efficiency) or 3 (high purity) such tracks
  $\rightarrow$ jet is b-tagged

inspired from CMS NOTE 2006 019 and BTV-11-002
IP smearing

- computes coordinates of point of closest approach \((X_d, Y_d)\) to vertex \((0,0)\) and smears \((X_d, Y_d)\) independently.
  \[ X_d(sm) = X_d + \text{Gaus}(0, res) \] (idem for \(Y_d(sm)\)).
- derive \(\text{IP}(sm)\) from \((X_d(sm), Y_d(sm))\)
- compute independently \(\sigma(\text{IP}) = |\text{Gaus}(0, res)|\).
- save these quantities as track members.
takes as input track and jet collections (alternatively could take simply jet collection using charged jet constituents as tracks)

parameters:

- \( p_T^{\text{min}} = 1 \) GeV of the tracks
- \( \Delta R \) cone to associate tracks with jet (typically \( \Delta R = 0.3 \))
- \( IP_{\text{max}} = 2 \) mm
- \( \text{sig}(IP)_{\text{min}} \) (algo working point)
- \( N_{\text{tracks}} \), defines algorithm. Minimum number of tracks that pass \( \text{sig}(IP) \) cut (= 2 for HE, = 3 for HP)

compute \( \text{sig}(IP) = \text{sign}(p_T^{\text{jet}} \cdot IP) \cdot IP \).

flag BTAG set to true if at least \( N_{\text{tracks}} \) with \( \text{sig}(IP) > \text{sig}(IP)_{\text{min}} \) are found.
IP btagging
IP btagging

Performance: High Purity

- c mistag rate agrees ok
- too pessimistic light rejection for low btagging efficiency

At 55% b-tagging efficiency (Medium working point):

DELPHES: l-mis = 1.5%, c-mis = 10%
CMS: l-mis = 1.0%, c-mis = 10%

\[ \text{sig}(IP)_{min} = 6.5 \]
IP btagging

Performance vs impact parameter resolution (1)

\[ \rho_T < 5 \text{GeV} \quad \sigma(\text{IP}) = 20 \mu m \]

\[ \rho_T > 5 \text{GeV} \quad \sigma(\text{IP}) = 10 \mu m \]
Performance vs impact parameter resolution (2)

\[ \rho_T < 5 \text{GeV} \] \( \sigma(\text{IP}) = 10 \, \mu m \)

\[ \rho_T > 5 \text{GeV} \] \( \sigma(\text{IP}) = 5 \, \mu m \)
Performance vs impact parameter resolution (3)

\[ \rho_T < 5 \text{GeV} \ \sigma(IP) = 5 \mu m \]

\[ \rho_T > 5 \text{GeV} \ \sigma(IP) = 1 \mu m \]
Simple calorimeters
• In **Calorimeter** module ECAL and HCAL tied together → same granularity

• New module: **SimpleCalorimeter**

  → call it once for ECAL and once for HCAL
  → can define separate:
    
    • granularity
    • resolution
    • fraction of energy deposit per particle

• It outputs the following objects:

  → Towers ( ~ full deposit)
  → **EFlow**Towers ( ~ full deposit – charged deposit)
N-subjettiness
N-subjettiness and N-jettiness

- very useful for identifying **sub-structure** of highly-boosted jets.
- build ratios $\tau_N / \tau_M$ to **discriminate** between $N$ or $M$-prong

- Embedded in FastJetFinder module
- Variables $\tau_1, \tau_2, \ldots, \tau_5$ saved as jet members (N-subjettiness)

---

Thanks to A. Larkowski for help

```cpp
# N-subjettiness

module FastJetFinder FastJetFinder {
    # set InputArray Colorimeter/towers
    set InputArray EFlowMerger/eflow

    set OutputArray jets

    # algorithm: 1 CDFJetClu, 2 MidPoint, 3 SIScone, 4 kt,
    # 5 Cambridge/Aachen, 6 antikt, 7 antikt wta, 8 Njettiness

    set JetAlgorithm 7
    set ParameterR 0.5

    set ComputeNsubjettiness 1
    set Beta 1.0

    # axis mode: 1 wta, 2 wta optimized, 3 kt, 4 kt optimized
    set AxisMode 1

    set JetPTMin 1000.0
}
```
Jerome de Favereau
Christophe Delaere
Pavel Demin
Andrea Giammanco
Vincent Lemaitre
Alexandre Mertens
Michele Selvaggi

the community ...
Back-up
The modules: Particle Propagation

- **Charged** and **neutral** particles are propagated in the magnetic field until they reach the calorimeters.

  - Propagation parameters:
    - magnetic field \( B \)
    - radius and half-length \((R_{\text{max}}, z_{\text{max}})\)

  - Efficiency/resolution depends on:
    - particle ID
    - transverse momentum
    - pseudorapidity

No real tracking/vertexing!!
→ no fake tracks/ conversions (but can be implemented)
→ no dE/dx measurements
The modules: Calorimetry

- em/had calorimeters have same segmentation in eta/phi

- Each particle that reaches the calorimeters deposits a fraction of its energy in one ECAL cell ($f_{\text{EM}}$) and HCAL cell ($f_{\text{HAD}}$), depending on its type:

<table>
<thead>
<tr>
<th>particles</th>
<th>$f_{\text{EM}}$</th>
<th>$f_{\text{HAD}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>e, γ, π°</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Long-lived neutral hadrons ($K_s^0$, $Λ^0$)</td>
<td>0.3</td>
<td>0.7</td>
</tr>
<tr>
<td>ν, μ</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>others</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- Particle energy is smeared according to the calorimeter cell it reaches

No different segmentation ECAL vs. HCAL
No Energy sharing between the neighboring cells
No longitudinal segmentation in the different calorimeters
The modules: Particle-Flow Emulation

- Idea: Reproduce realistically the performances of the Particle-Flow algorithm.

- In practice, in DELPHES use **tracking and calo** info to reconstruct high reso. input objects for later use (jets, $E_T^{miss}$, $H_T$)

  → assume $\sigma(trk) < \sigma(calo)$

**Example 1:** A pion of 10 GeV

  → $E^{HCAL}(\pi^+) = 15$ GeV
  → $E^{TRK}(\pi^+) = 11$ GeV

**Particle-Flow** algorithm creates:

  → PF-track, with energy $E^{PF-trk} = 11$ GeV
  → PF-tower, with energy $E^{PF-tower} = 4$ GeV
Example 2: A pion (10 GeV) and a photon (20 GeV)

\[ E^{\text{ECAL}}(\gamma) = 18 \text{ GeV} \]
\[ E^{\text{HCAL}}(\pi^+) = 15 \text{ GeV} \]
\[ E^{\text{TRK}}(\pi^+) = 11 \text{ GeV} \]

Particle-Flow algorithm creates:

\[ \text{PF-track, with energy } E^{\text{PF-trk}} = 11 \text{ GeV} \]
\[ \text{PF-tower, with energy } E^{\text{PF-tower}} = 4 + 18 \text{ GeV} \]

Separate neutral and charged calo deposits has crucial implications for pile-up subtraction.

No separation between “Photons” and “Neutral Hadrons” in the output.
The modules: Jets / $E_T^{\text{miss}}$ / $H_T$

- Delphes uses **FastJet** libraries for jet clustering

- Inputs **calorimeter towers** or “**particle-flow**” objects

```cpp
module FastJetFinder FastJetFinder {
    # set InputArray Calorimeter/towers
    set InputArray EFlowMerger/eflow

    set OutputArray jets

    # algorithm: 1 CDFJetClu, 2 MidPoint, 3 SIScone, 4 kt, 5 Cambridge/Aachen, 6 antikt
    set JetAlgorithm 6
    set ParameterR 0.7

    set ConeRadius 0.5
    set SeedThreshold 1.0
    set ConeAreaFraction 1.0
    set AdjacencyCut 2.0
    set OverlapThreshold 0.75

    set MaxIterations 100
    set MaxPairSize 2
    set Iratch 1

    set JetPTMin 20.0
}
```
The modules: Leptons and photons reconstruction

Muons/electrons

- Identified via their PDG id
- Muons do not deposit energy in calo (independent smearing parameterized in $p_T$ and $\eta$)
- Electrons smeared according to tracker and ECAL resolution

Isolation:

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

If $I(P) < I_{\text{min}}$, the lepton is isolated

User can specify parameters $I_{\text{min}}$, $\Delta R$, $p_T^{\text{min}}$

No fakes, punch-through, brehmstrahlung, conversions
The modules: b-jets (tau-jets)

Current b-tagging:
- Check if there is a b,c-quark in the cone of size DeltaR
- Apply a **parametrized Efficiency** (PT, eta)

```cpp
module BTagging BTagging {
    set PartonInputArray Delphes/partons
    set JetInputArray JetEnergyScale/jets

    set BitNumber 0
    set DeltaR 0.5
    set PartonPTMin 1.0
    set PartonEtaMax 2.5

    # default efficiency formula (misidentification rate)
    add EfficiencyFormula {0} [0.001]

    # efficiency formula for c-jets (misidentification rate)
    add EfficiencyFormula {4} {
        (pt <= 15.0) * (0.000) + \n        (abs(eta) <= 1.2) * (pt > 15.0) * (0.2*tanh(pt*0.03 - 0.4)) + \n        (abs(eta) > 1.2 && abs(eta) <= 2.5) * (pt > 15.0) * (0.1*tanh(pt*0.03 - 0.4)) + \n        (abs(eta) > 2.5) * (0.000)
    }

    # efficiency formula for b-jets
    add EfficiencyFormula {5} {
        (pt <= 15.0) * (0.000) + \n        (abs(eta) <= 1.2) * (pt > 15.0) * (0.5*tanh(pt*0.03 - 0.4)) + \n        (abs(eta) > 1.2 && abs(eta) <= 2.5) * (pt > 15.0) * (0.4*tanh(pt*0.03 - 0.4)) + \n        (abs(eta) > 2.5) * (0.000)
    }
}
```

No Predictive B-tagging !
Impact parameter smearing module

- In RECO, impact parameter IP is derived from track fitting.
- $sigma(IP)$ is correlated with error on $p_T$, $R_{curv}$ ... (encoded in track covariance matrix).
- Here we assume error can be simply parametrized as $f(p_T, \eta)$.

Module:

- takes as input tracks
- parameters needed for computation ($R_{curv}$, $(X_c, Y_c)$) already saved as members of Candidates in the ParticlePropagator module.
- specify (absolute) impact parameter resolution $f(p_T, \eta)$ in mm (here we take $\text{res} = 10 \, \mu m$ if $p_T < 5$ GeV and $\text{res} = 5 \, \mu m$ if $p_T > 5$ GeV)
- computes coordinates of point of closest approach $(X_d, Y_d)$ to vertex (0,0) and smears $(X_d, Y_d)$ independently $X_d(sm) = X_d + Gaus(0, \text{res})$ (idem for $Y_d(sm)$).
- derive IP(sm) from $(X_d(sm), Y_d(sm))$
- compute independently $\sigma(IP) = |Gaus(0, \text{res})|$
- save these quantities as track members.
IP btagging

takes as input track and jet collections (alternatively could take simply jet collection using charged jet constituents as tracks)

parameters:

- \( p_T^{\text{min}} = 1 \text{ GeV of the tracks} \)
- \( \Delta R \) cone to associate tracks with jet (typically \( \Delta R = 0.3 \))
- \( IP_{\text{max}} = 2 \text{ mm} \)
- \( \text{sig}(IP)_{\text{min}} \) (algo working point)
- \( N_{\text{tracks}} \), defines algorithm. Minimum number of tracks that pass sig(IP) cut (= 2 for HP, = 3 for HE)

compute \( \text{sig}(IP) = \text{sign}(p_T^{\text{jet}} \cdot IP) \) IP.

flag BTAG set to true if at least \( N_{\text{tracks}} \) with \( \text{sig}(IP) > \text{sig}(IP)_{\text{min}} \) are found.