Delphes 3
Framework for fast simulation of a generic collider experiment

Michele Selvaggi, for the Delphes Team

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

ACAT, Beijing
May 18, 2013
Introduction
**Detector simulation**

- Full simulation (GEANT):
  - **simulates** particle-matter interaction (including e.m. showering, nuclear int., brehmstrahlung, photon conversions, etc ...) → 10 s /ev

- Experiment Fast simulation (ATLAS, CMS ...):
  - **simplifies** and makes faster simulation and reconstruction → 1 s /ev

- Parametric simulation:
  **Delphes, PGS:**
  - **parameterize** detector response, reconstruct complex objects → 10 ms /ev

  **TurboSim**
  - **no detector**, parameterize object response, parton ↔ reco
Detector simulation

- Full simulation (GEANT):
  - **simulates** particle-matter interaction (including e.m. showering, nuclear int., brehmstrahlung, photon conversions, etc ...) → 10 s /ev

- Experiment Fast simulation (ATLAS, CMS ...):
  - **simplifies** and makes faster simulation and reconstruction → 1 s /ev

- Parametric simulation:
  
  **Delphes:**
  - **parameterize** detector response, reconstruct complex objects → 10 ms /ev

  **TurboSim**
  - **no detector,** parameterize object response, parton ↔ reco
Parametric simulation

- What do we expect from Delphes parametric detector simulation?
  - fast
  - realistic enough
  - flexible detector geometry
  - user-friendly
  - flexible I/O (modular)

- When do you need Delphes?
  - more advanced than parton-level studies
  - scan big parameter space (SUSY-like ...)
  - preliminary tests of new geometries/resolutions (upgrades, Snowmass)
  - testing analysis methods (multivariate/Matrix Element)
  - educational purpose (master thesis)
Workflow
What is Delphes?

- **Delphes** is a **modular framework** that simulates the response of a multipurpose detector

- **simulates:**
  - charged particle propagation in magnetic field: **tracking**
  - electromagnetic and hadronic **calorimeters**
  - **muon** system

- **reconstructs:**
  - leptons (electrons and muons)
  - photons
  - jets and missing transverse energy
  - taus and b's
**Technical features**

- **modular** C++ code, uses ROOT classes

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

- **Output**
  - ROOT trees

- **Configuration file**
  - define geometry
  - resolution/reconstruction/selection criteria
  - output object collections
Charged Particle Propagation

- Charged 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

\[
\begin{align*}
\text{Not real tracking/vertexes}  &  \text{!!} \\
\rightarrow \text{ no fake tracks/ conversions (but can be easily implemented)} \\
\rightarrow \text{ no } dE/dx \text{ measurements}
\end{align*}
\]
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, π⁰</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

$$E_{\text{smeared}} = \text{gauss}(f_{\text{EM}} E, \sigma_{\text{EM}}(\eta)) + \text{gauss}(f_{\text{HAD}} E, \sigma_{\text{HAD}}(\eta))$$

$$\sigma^2(\eta) = N^2(\eta) + S^2(\eta)E + C^2(\eta)E^2$$
“Particle-flow” approach aims at maximizing object reconstruction resolution by using all sub-detector information (in Delphes tracking, ECAL and HCAL) → particle-flow candidates can be used as input for jets and $E_T^{\text{miss}}$

• For each calorimeter cell:
Leptons, photons

- **Muons/photons/electrons**
  - **identified** via their PDG id
  - inside the **tracker coverage** for electrons and muons
  - muons do not deposit energy in calo (independent smearing parameterized in $p_T$ and $\eta$)
  - electrons and photons smeared according to electromagnetic calorimeter resolution

- **Isolation:**
  \[
  \text{rel.Iso} = \frac{\sum_{\Delta R < 0.5} p_{T}^{\text{track}}}{p_T} \quad \rightarrow \text{modular structure allows to easily define different isolation}
  \]

  If rel.Iso \~ 0, the lepton is isolated

- **Not taken into account:**
  - fakes, punch-through, brehmstrahlung, conversions
• **FastJet** library used for jet clustering
  - all clustering algos supported: anti-\(k_T\), SisCone, ...

• Jets, \(E_T^{\text{miss}}\) and \(H_T\) quantities can be formed from:
  - calorimeter towers
  - “particle-flow” candidates:

\[
E_T^{\text{miss}} = - \sum_i p_T^i(i), \quad H_T = \sum_i |p_T^i(i)|,
\]
**b and tau jets**

- **b-jets**
  - if \( b \) parton is found in a cone \( \Delta R \) w.r.t jet direction
    - apply **efficiency**
  - if \( c \) parton is found in a cone \( \Delta R \) w.r.t jet direction
    - apply **c-mistag rate**
  - if \( u,d,s,g \) parton is found in a cone \( \Delta R \) w.r.t jet direction
    - apply **light-mistag rate**

    **b-tag flag** is then stored in the jet collection

- **tau-jets**
  - if tau lepton is found in a cone \( \Delta R \) w.r.t jet direction
    - apply **efficiency**
  - else
    - apply **tau-mistag rate**

  tau jets have their own collection (no leptonic tau decays)

can define \( p_{T} \) and \( \eta \) dependent efficiency and mistag rate
Does this approach work?
**Validation: jets and $E_T^{\text{miss}}$**

- Electrons, muons and photons are auto-validated by construction
- **Jets** and **missing energy** need to be tested:

\[ \frac{\sigma \left( E_T \right)}{E_T} \]

**CMS resolution from:** The CMS Collaboration, **CMS-PAS-PFT-09-001**

\[ \frac{\sigma \left( E_T \right)}{E_T} \]

→ **excellent agreement**
Performances

- small memory footprint
- short processing time with a standard laptop
- output 50% smaller than HepMC!
Pile-up motivations

- **Pile-up** becomes an issue at **high luminosity** LHC
  - reduced **efficiency**
  - worsened **resolution** (jets, $E_{\text{miss}}$)
  - degraded **isolation**
  - **fake** tracks, jets

- **Efficiencies** and **resolutions** can be **tuned by hand** to mimic pile-up

- Fake objects need to be simulated. Also, we want to have some predictive power:
  - We therefore introduced: **tunable simulation** of pile-up
    - pile-up **subtraction** procedure.

- This new feature is being actively validated in collaboration with the groups preparing results for Snowmass 2013 (CMS and ATLAS).
Pile-up implementation

- Pile-up is implemented in Delphes since version 3.0.4
  - mixes $N$ minimum bias events with hard event sample
  - spreads $\text{poisson}(N)$ events along $z$-axis with configurable spread
    - if $z < |Z_{\text{res}}|$ keep all charged and neutrals ($\rightarrow$ ch. particles too close to hard scattering to be rejected)
    - if $z > |Z_{\text{res}}|$ keep only neutrals (perfect charged subtraction)
- With this approach:
  - charged subtraction is already done at the mixing level (faster)
  - allows user to tune amount of charged particle subtraction by adjusting $Z$ spread/resolution
- Residual pileup substraction is needed for jets and isolation.
  - Use the FastJet Area approach (add ref.)
    - compute $\rho = \text{event pile-up density}$
    - jet correction : $p_T \rightarrow p_T - \rho A$ (JetPileUpSubtractor)
    - isolation : $\sum p_T \rightarrow \sum p_T - \rho \pi R^2$ (Isolation module itself)
Performances (II)

Time consumption per module

No pile-up
- Propagation
- Calorimeter
- FastJet
- TreeWriter
- Other

53%

50 pile-up
- Propagation
- Calorimeter
- FastJet
- TreeWriter
- Other

93%
A basic **event-display** is provided, based on ROOT EVE

- Displays **tracks, calo-towers, jets**.
- Useful for debugging
- More detailed version planned.
Event-display with pile-up

\[ Z_{\text{spread}} \approx Z_{\text{res}}, \text{ 50 av. pile-up} \]
Conclusions
Development

• Delphes project started back in 2007

• 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

• Team
  - Two research scientists (P. Demin, J. de Favereau)
    Website, repository, releases
    Core developments and code optimization
    Support
  - One post-doc (M. Selvaggi) and one PhD student (A. Mertens)
    Re-optimization of the performances, validation
    Implementation of new features
    Support

• **Widely** tested and used by the community  > 100 citations!!
Conclusions

• **Delphes 3** is out, with **major improvements**:
  - modularity
  - pile-up implementation
  - revamped particle flow algorithm
  - new visualization tool based on ROOT EVE
  - default cards giving results on par with published performance from LHC experiments
  - now fully integrated within MadGraph5


• Test it, and give us feedback!

https://cp3.irmp.ucl.ac.be/projects/delphes
Event-display (Charged PU substr.)

\[ Z_{\text{spread}} \gg Z_{\text{res}}, \text{ 50 av. pile-up} \]
Event-display (Charged PU substr.)

\[ Z_{\text{spread}} \gg Z_{\text{res}}, \text{ 50 av. pile-up} \]
Backup slides
“Particle-flow” approach aims at maximizing object reconstruction resolution by using all sub-detector information (in Delphes tracking, ECAL and HCAL) → particle-flow candidates can be used as input for jets and $E_T^{\text{miss}}$.

- For each calorimeter cell:
“Particle-flow” approach aims at maximizing object reconstruction resolution by using all sub-detector information (in Delphes tracking, ECAL and HCAL) → particle-flow candidates can be used as input for jets and $E_T^{\text{miss}}$.

- For each calorimeter cell:
• “Particle-flow” approach aims at maximizing object reconstruction resolution by using all sub-detector information (in Delphes tracking, ECAL and HCAL)

→ particle-flow candidates can be used as input for jets and $E_T^{\text{miss}}$

• For each calorimeter cell:
Pile-up sanity checks

QCD : fake $E_T^{\text{miss}}$

$\ell\ell\nu\nu$ : real $E_T^{\text{miss}}$

MET

Jets
Event-display (Charged PU substr.)

\[ Z_{\text{spread}} \gg Z_{\text{res}}, \text{ 50 av. pile-up} \]
Electrons, muons and photons are auto-validated by construction.

Jets and missing energy need to be tested:

\[ \sigma \left( \frac{E_{\text{jet}}/E_{\text{MC}}}{E_{\text{jet}}/E_{\text{MC}}} \right) = \frac{102.0 \pm 5.7}{E_{\text{jet}}/E_{\text{MC}}} \]

Events: pp → gg
MidPoint cone algorithm, Δ R=0.7
MG/ME + Pythia + Delphes

\[ \sigma(E)/E = \frac{100.0 \pm 1.1 \times 434.0}{E_{\text{MC}}} \]

Events: pp → ggX
k_{T} algorithm Δ R = 0.6
MG/ME + Pythia + Delphes

→ good agreement
Electrons, muons and photons are auto-validated by construction.

Jets and **missing energy** need to be tested:

→ good agreement