DELPHES
Status and Plans

Michele Selvaggi

CERN

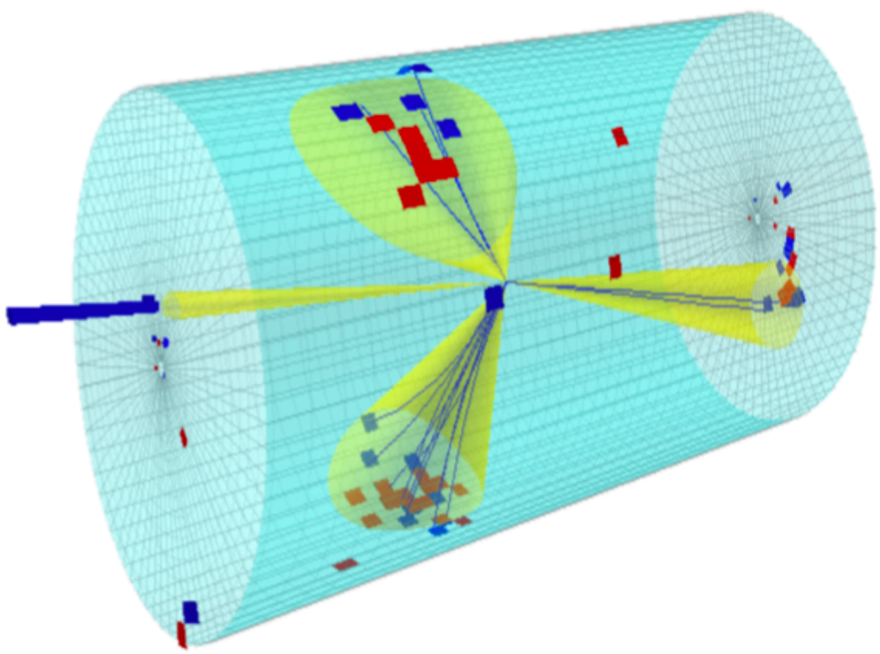
RDFA 16/12/2021

Delphes in a nutshell

- **Delphes** is a modular framework that simulates the response of a **multipurpos detector** in a parameterised fashion

- Includes:
  - pile-up
  - charged particle propagation in B field
  - EM/Had calorimeters
  - particle-flow

- Provides:
  - leptons, photons, neutral hadrons
  - jets, missing energy
  - heavy flavour tagging



- designed to deal with hadronic environment
- well-suited also for e+e- studies
- detector cards for: CMS (current/PhaseII) - ATLAS - LHCb - FCC-hh - ILD - CEPC - FCCee (IDEA/CLD)

Reconstruction performance

- Delphes is not fully parameteric, object reco. efficiency (e.g. Tracking) and Calorimeter performance is parameterised, BUT:
  - Tracking resolution , dE/dx
  - Particle Flow
  - Jet (anti-kT/Valencia exclusive)
  - Missing energy
  - HF-tagging

can be predicted (with all the caveats of a fast simulation model) …

Outline

- Recent developments:
  - Dual Readout Calorimetry
  - TrackCovariance/Cluster counting
  - Time of flight implementation
- Bug Fixes:
  - Inefficiency for low momentum tracks (fixed)
  - Inefficiency for electrons (fixed)
- Proposals:
  - for electron energy measurement
  - delphes / EDM4Hep event format

Recent Developments

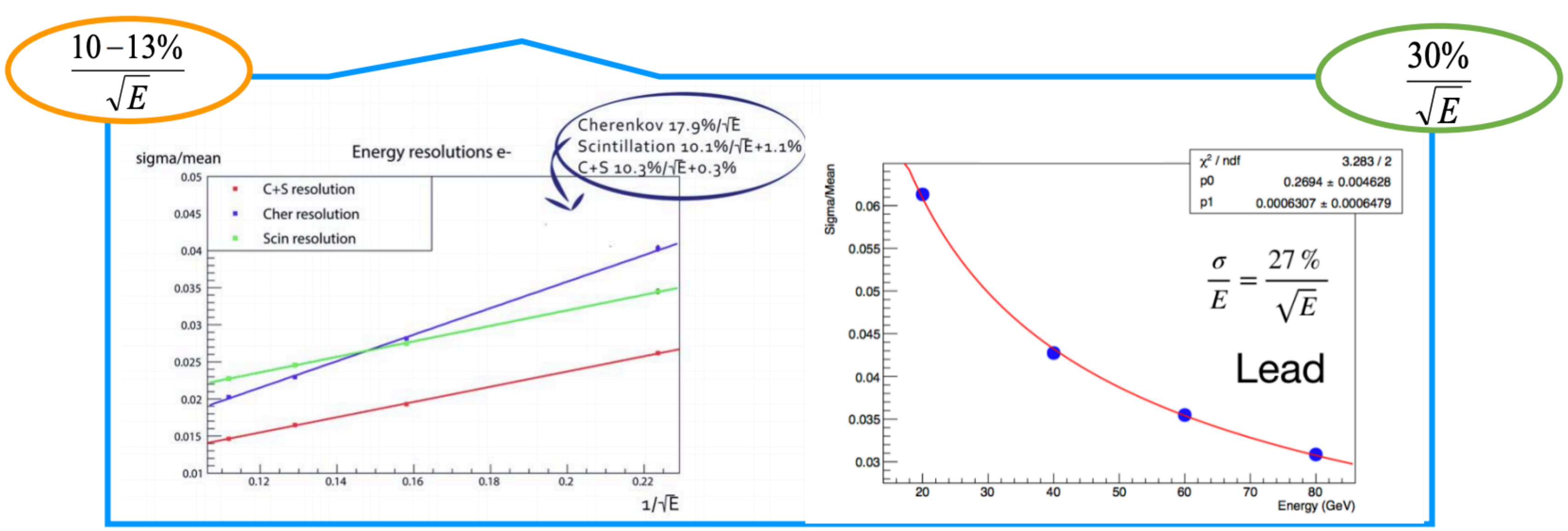
IDEA - (DR) Calorimetry

DELPHES fast simulation

```
#############
# Calorimeter
#############
module DualReadoutCalorimeter Calorimeter {
  set ParticleInputArray ParticlePropagator/stableParticles
  set TrackInputArray TrackMerger/tracks
```

```
# Lists of the edges of each tower in eta and phi;
# each list starts with the lower edge of the first tower;
# the list ends with the higher edged of the last tower.
# Barrel:  deta=0.02 towers up to |eta| <= 0.88 ( up to 45°)
# Endcaps: deta=0.02 towers up to |eta| <= 3.0 (8.6° = 100 mrad)
# Cell size: about 6 cm x 6 cm
```

- If EM > 0 and E_{had} = 0 → σ(EM)
  - e.g. γ
- If E_{had} > 0 → σ(had)
  - e.g. π^+ or (γ, π^+)

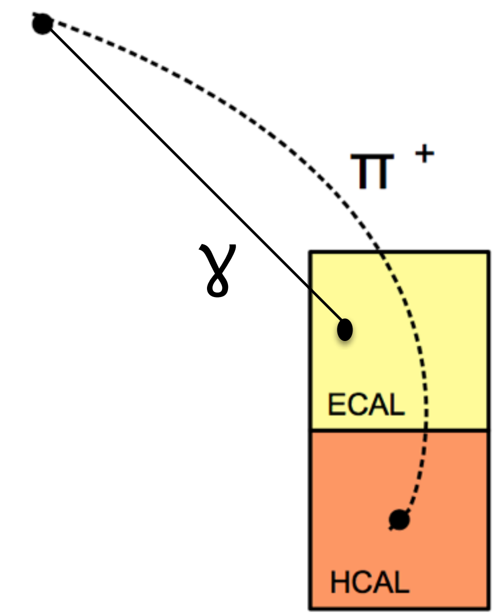


E. Fontanesi
L. Pezzotti
M. Antonello

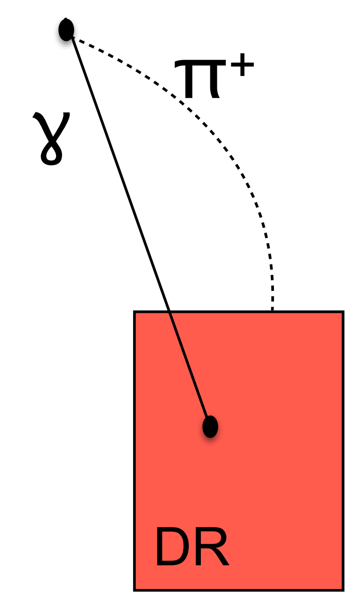
Dual Readout Particle-Flow

DELPHES
fast simulation

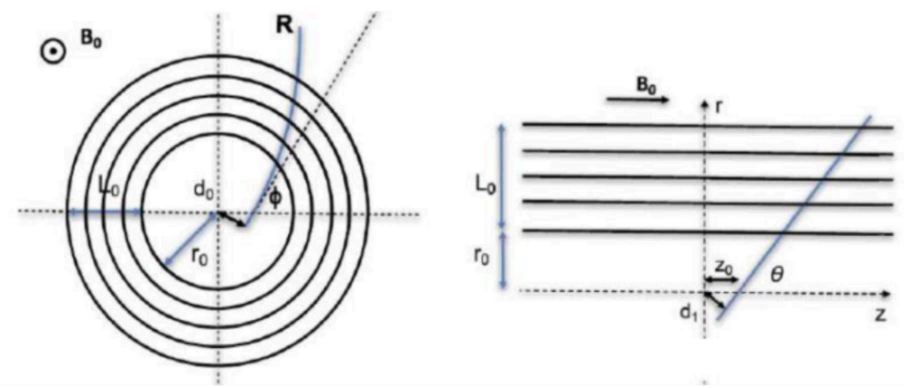
- Given **charged track** hitting **calorimeter cell**:
  - is deposit more compatible with **charged only** or **charged + neutral** hypothesis?
  - how to **assign momenta** to resulting components?
  - If **charged + neutral** how to associate particle ID to **charged** and **neutral** components, e.g **(γ,π^+)** or **(e^+,K_L)** ?

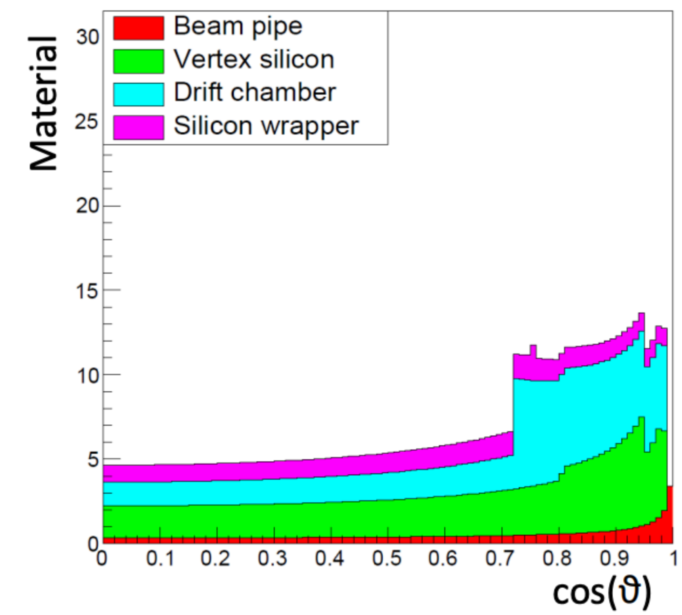


- **DualReadoutCalorimeter** module in Delphes assumes we can always disentangle these two cases
  - Probably ok at FCC-ee since probability of overlap not so large (except for taus?)
  - Impact of granularity on performance was studied extensively by Elisa Fontanesi (see here)



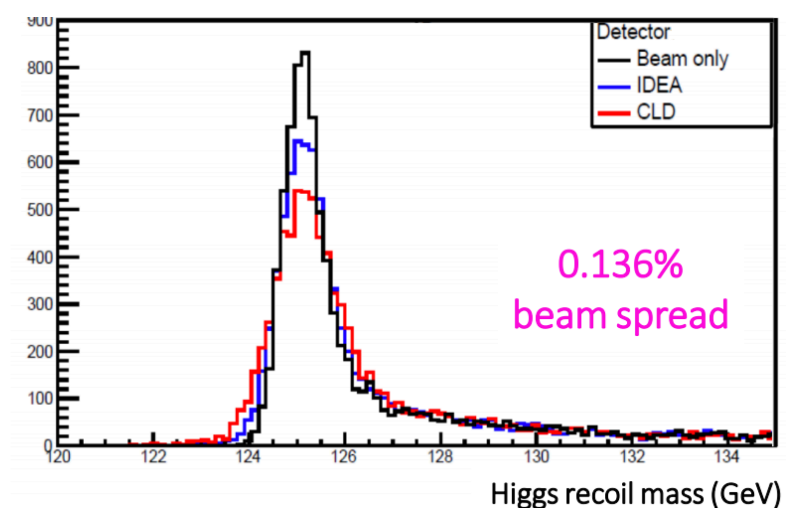
Fast Tracking Simulation





<u>Track Smearing</u>

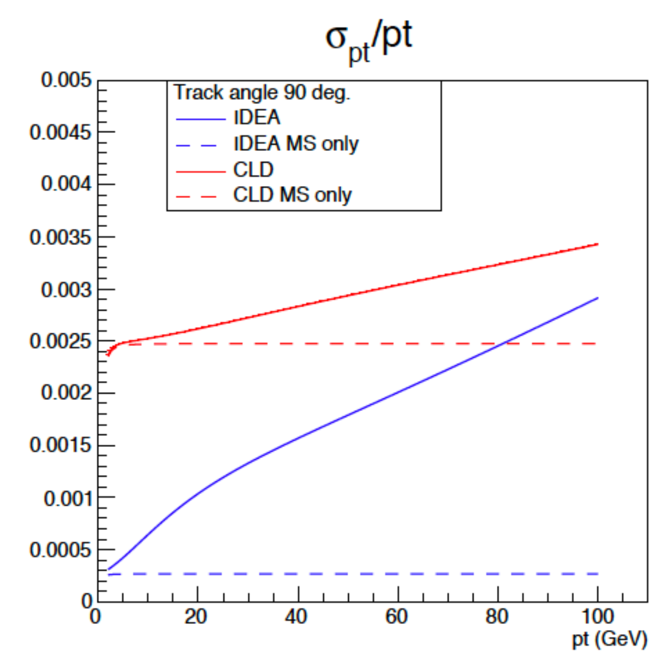
- Simple tracker geometry implementation, **including material**
- Computes **full covariance matrix** (in present Delphes we have “diagonal” smearing in the 5 tracking parameters)
- Can be used for studying impact of material and realistic **HF tagging** simulation

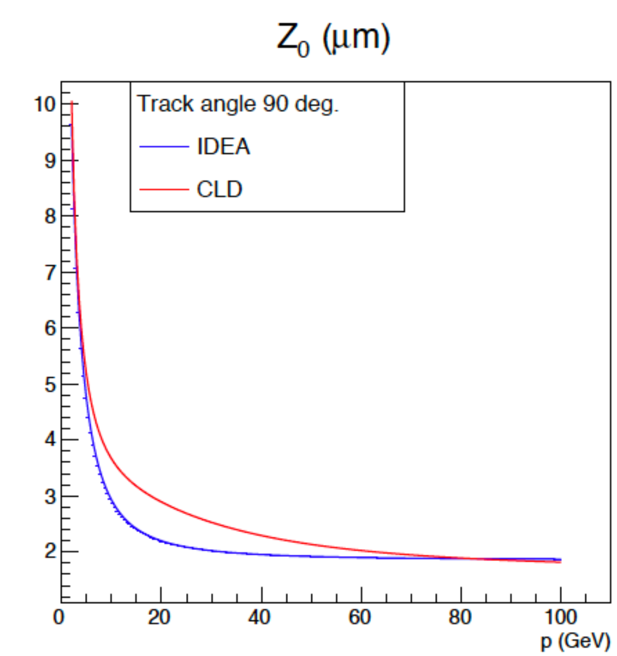


F. Bedeschi

FCC-ee TrackCovariance

```
########################################
# Smearing for charged tracks
########################################

module TrackCovariance TrackSmearing {

    set InputArray TrackMergerPre/tracks
    set OutputArray tracks

    ## minimum number of hits to accept a track
    set NMinHits 6

    ## magnetic field
    set Bz $B

    ## uses https://raw.githubusercontent.com/selvaggi/FastTrackCovariance/master/GeoIDEA_BASE.txt
    set DetectorGeometry {

  # barrel  name      zmin   zmax   r      w (m)     X0       n_meas  th_up (rad) th_down (rad)    reso_up (m)   reso_down (m)  flag

  1        PIPE      -100   100    0.015  0.001655  0.2805   0       0          0                0             0              0
  1        VTXLOW    -0.12  0.12   0.017  0.00028   0.0937   2       0          1.5708           3e-006        3e-006         1
  1        VTXLOW    -0.16  0.16   0.023  0.00028   0.0937   2       0          1.5708           3e-006        3e-006         1
  1        VTXLOW    -0.16  0.16   0.031  0.00028   0.0937   2       0          1.5708           3e-006        3e-006         1
  1        VTXHIGH   -1     1      0.32   0.00047   0.0937   2       0          1.5708           7e-006        7e-006         1
  1        VTXHIGH   -1.05  1.05   0.34   0.00047   0.0937   2       0          1.5708           7e-006        7e-006         1
```



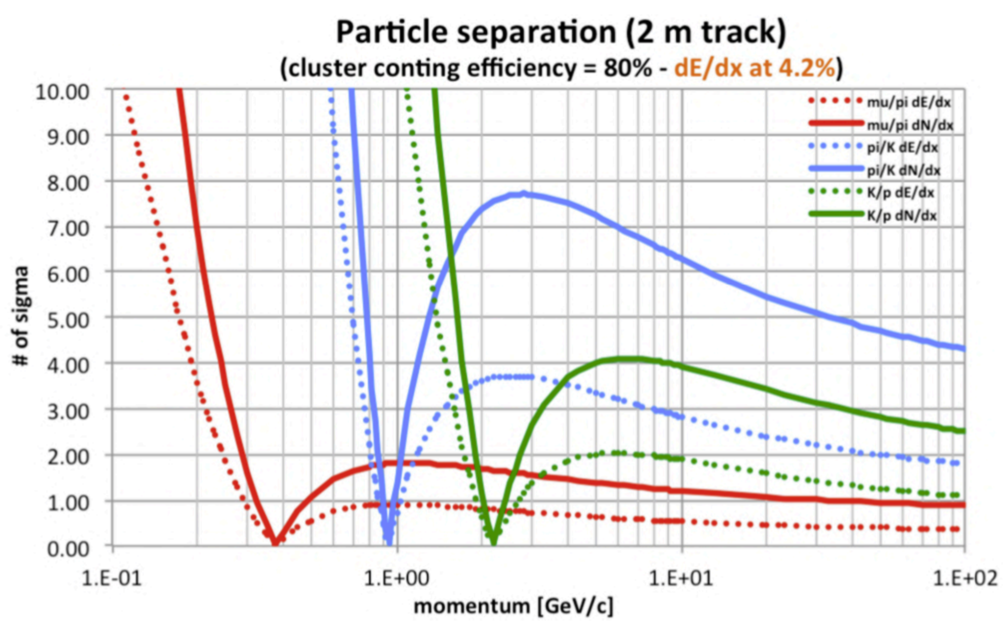


F. Bedeschi

TrackCovariance module

- **Requires:**
  - Geometry input (can be generated with Franco's code in standalone, or by hand)
  - Magnetic field

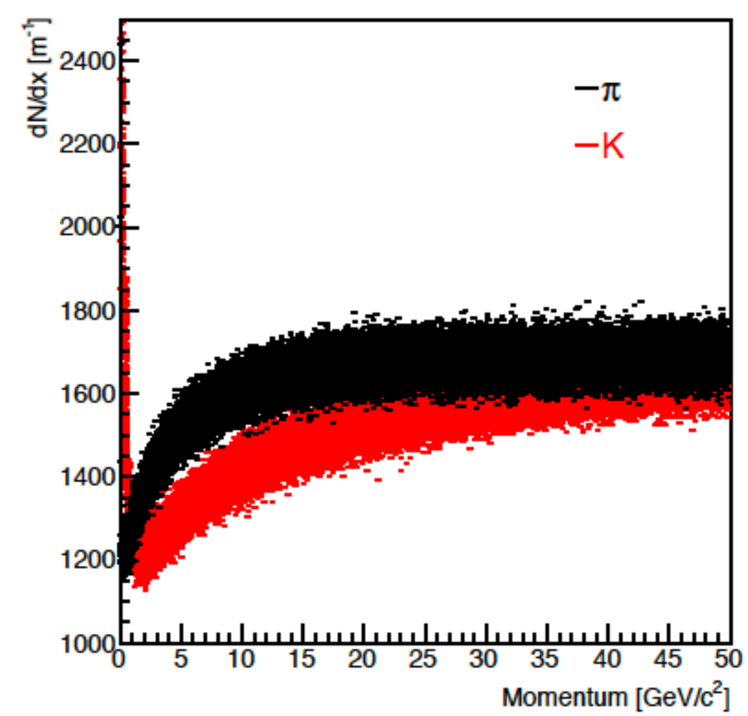
PID: ClusterCounting




```
###################
# Cluster Counting
###################

module ClusterCounting ClusterCounting {

  add InputArray TrackSmearing/tracks
  set OutputArray tracks

  set Bz $B

  ## check that these are consistent with DCHCANI/DCHNANO parameters in TrackCovariance module
  set Rmin $DCHRMIN
  set Rmax $DCHRMAX
  set Zmin $DCHZMIN
  set Zmax $DCHZMAX

  # gas mix option:
  # 0:  Helium 90% - Isobutane 10%
  # 1:  Helium 100%
  # 2:  Argon 50% - Ethane 50%
  # 3:  Argon 100%

  set GasOption 0

}
```

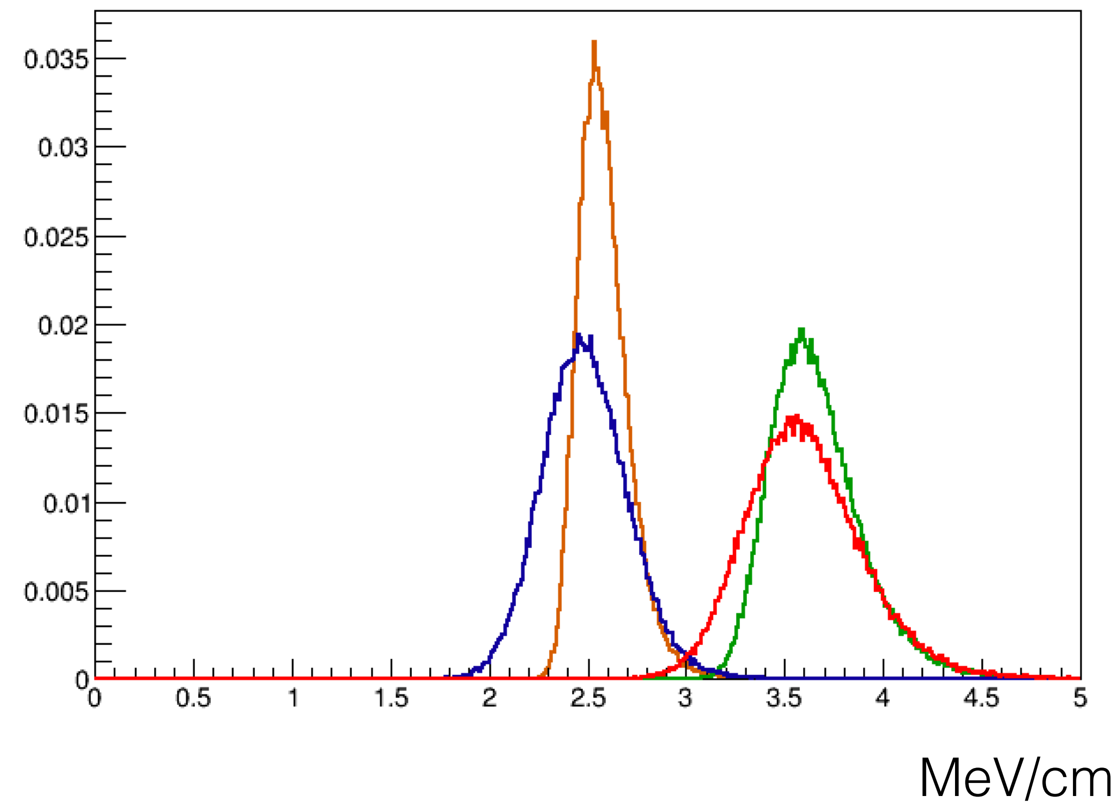
PID:

- Dndx method: implemented in same library that computes track parameters
- called by ClusterCounting module

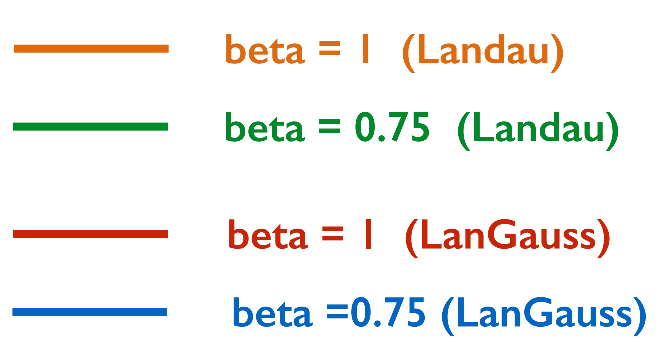


PID: dEdX

```
module EnergyLoss EnergyLoss {
  add InputArray ChargedHadronMomentumSmearing/chargedHadrons
  add InputArray ElectronMomentumSmearing/electrons
  add InputArray MuonMomentumSmearing/muons


  # absolute resolution per measurement (normalized in MeV/cm)
  # CMS pixel detector performance is reproduceable with r = 0.4
  # dedicated dEdX detector can achieve r = 0.0 or below (i.e better than Landau)

  #set Resolution 0.4
  set Resolution 0.2

  # fraction of measurements to ignore when computing truncated mean
  # suggested range [0.4-0.6]

  set TruncatedMeanFraction 0.5

  # detector properties (active fraction = nhits*thickness/L)
  set Thickness 100E-6
  set ActiveFraction 0.0006666

  # Silicon properties, for other materials:
  # cf. http://pdg.lbl.gov/2014/AtomicNuclearProperties/properties8.dat

  set Z 14.
  set A 28.0855
  set rho 2.329

  # material polarisation correction parameters
  set a 0.1492
  set m 3.2546
  set x0 0.2015
  set x1 2.8716
  set I 173.0
  set c0 4.4355
```



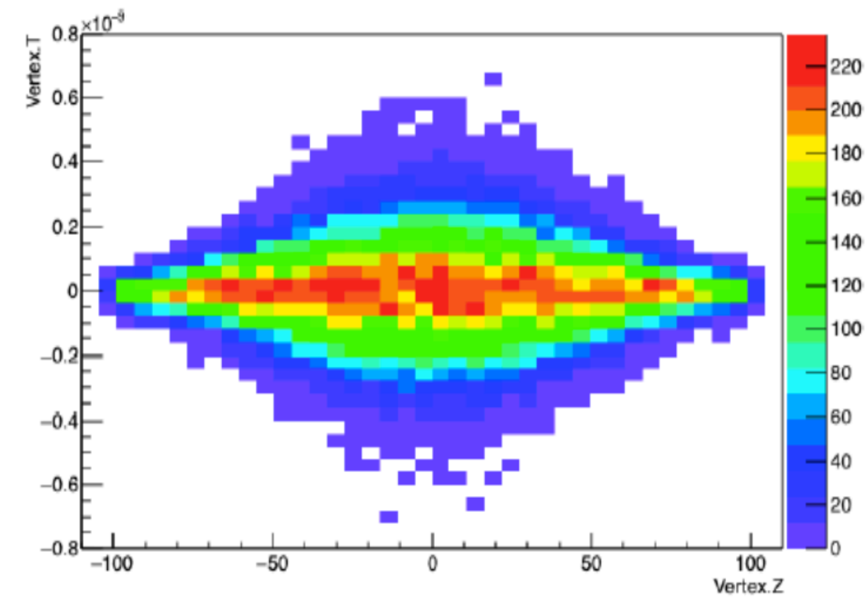
10 measurements x100 um silicon

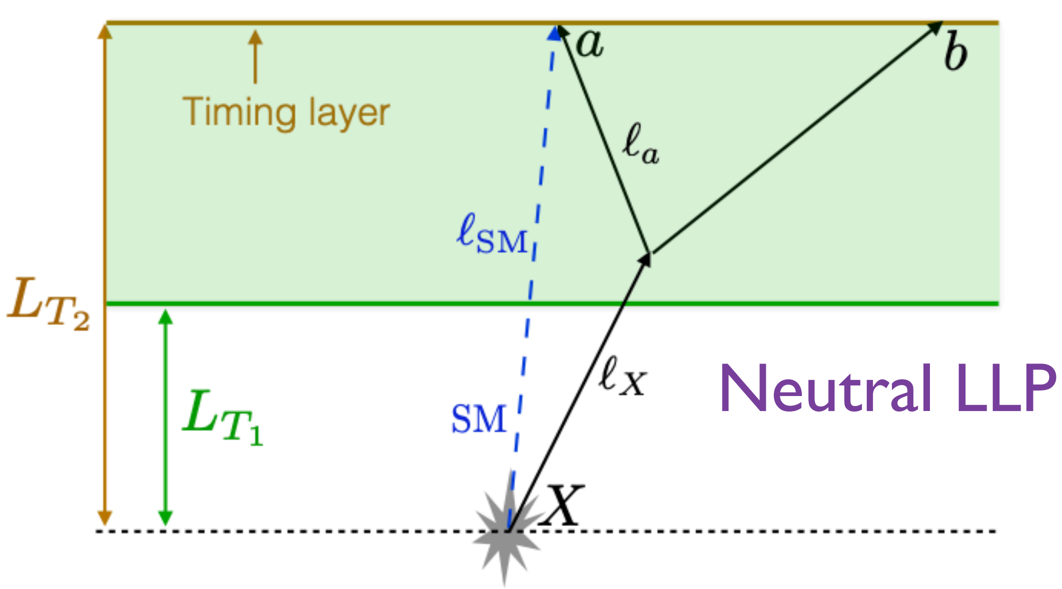


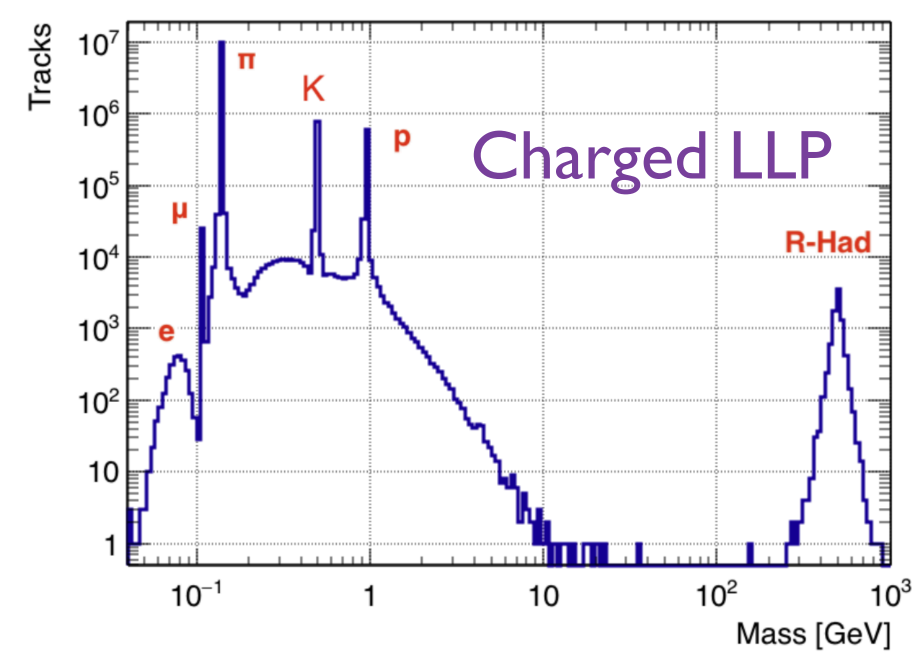
Timing detectors

- At the LHC, timing information can be used to disentangle hard vertex from pile-up, by vertexing in 4D
    - can this be used profitably in any way at FCC-ee?

- Timing can be used to measure TOF, and hence for particle ID (either SM or BSM long lived particles)







PID: Time of flight

- Time-of-flight PID computed by a sequence of modules:
  - ParticlePropagator computes path length and final time (L, t_F (MC))
  - TrackCovariance (Calorimeter) computes the momentum (energy)
  - TimeSmearing module computes measured time (with given time resolution)
  - TruthVertexFinder calculates position and time of MC truth vertices
  - TimeOfFlight module calculates initial time t_I:
    - $t_I = t_I$ (MC) — optimistic , default
    - $t_I = 0$ — pessimistic, for neutrals
    - $t_I = |x_{vtx}$ (MC)$| / \beta_{vtx}$ (MC) (naive attempt to realistically account for displacement)
- User has to compute $m_{t.o.f}$ on output tracks using :

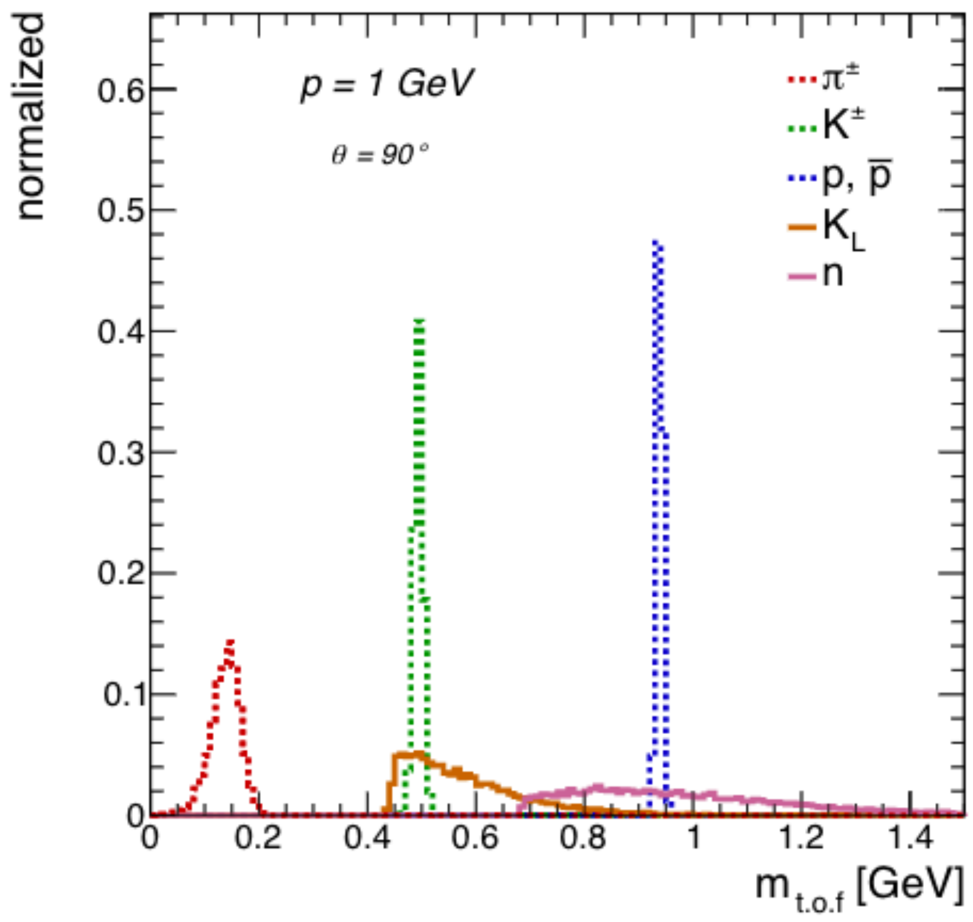
charged $m^{(c)}_{\mathrm{t.o.f.}} = p\sqrt{(\frac{t_{\mathrm{flight}}}{L})^2 - 1}$

$$t_{\mathrm{flight}} \equiv t_{\mathrm{F}} - t_{\mathrm{V}} = \frac{L}{\beta} = \frac{L\sqrt{p^2+m^2}}{p} = \frac{LE}{\sqrt{E^2 - m^2}}$$

neutral $m^{(n)}_{\mathrm{t.o.f.}} = E\sqrt{1 - (\frac{L}{t_{\mathrm{flight}}})^2}$

PID: Time of flight

- Implemented in the IDEA card for testing for both charged and neutrals:

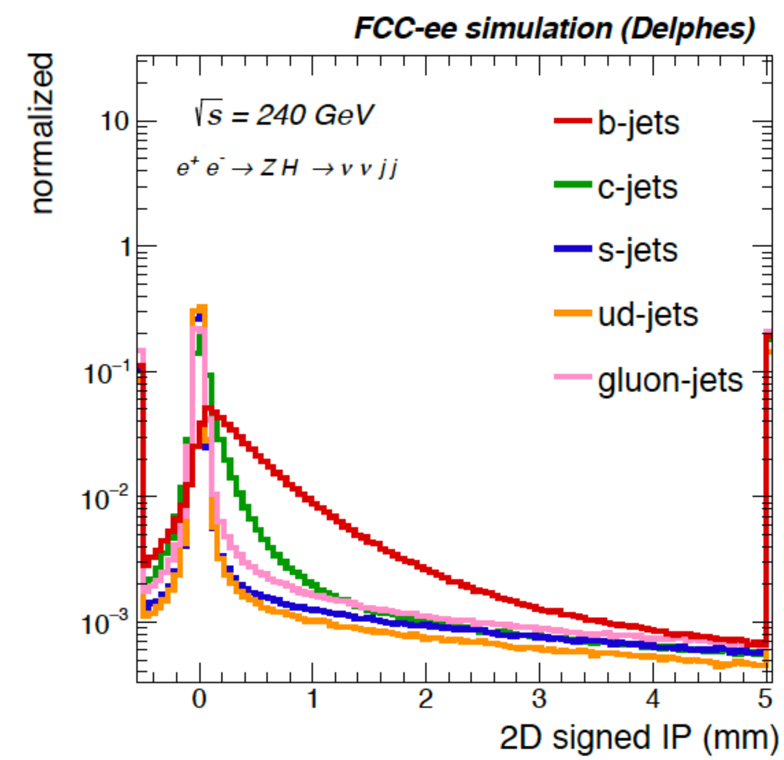



```
#######################################
#   Time Smearing MIP
#######################################

module TimeSmearing TimeSmearing {
  set InputArray ClusterCounting/tracks
  set OutputArray tracks

  # assume constant 30 ps resolution for now
  set TimeResolution {
                     (abs(eta) > 0.0 && abs(eta) <= 3.0)* 30E-12
                   }
}


#######################################
#   Time Of Flight Measurement
#######################################

module TimeOfFlight TimeOfFlight {
  set InputArray TimeSmearing/tracks
  set VertexInputArray TruthVertexFinder/vertices

  set OutputArray tracks

  # 0: assume vertex time tV from MC Truth (ideal case)
  # 1: assume vertex time tV = 0
  # 2: calculate vertex time as vertex TOF, assuming tPV=0

  set VertexTimeMode 0
}
```

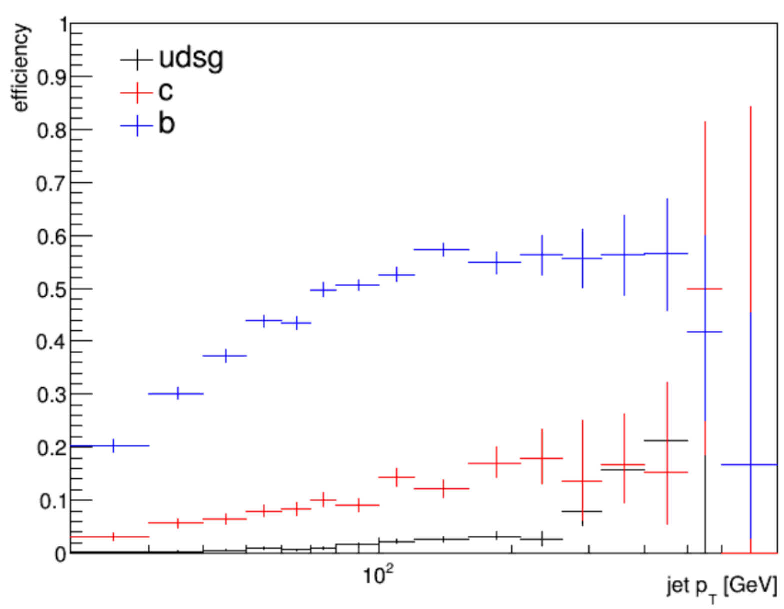
Heavy flavour Flavour Tagging

DELPHES fast simulation

- Track Counting B-Tagging:
  - parameterise **longitudinal** and **transverse impact parameter resolution (see previous slide)**
  - count number of tracks with significant displacement
  - no **secondary vertexing** is performed yet

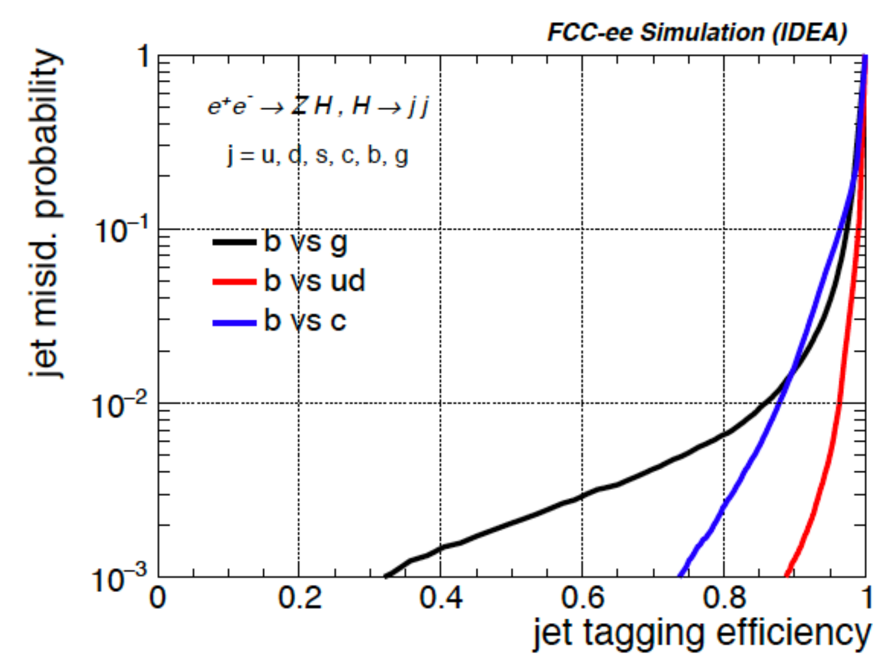


Can be used in conjunction with TrkCovariance module to build MVA HF jet tagger



Kevin Pedro

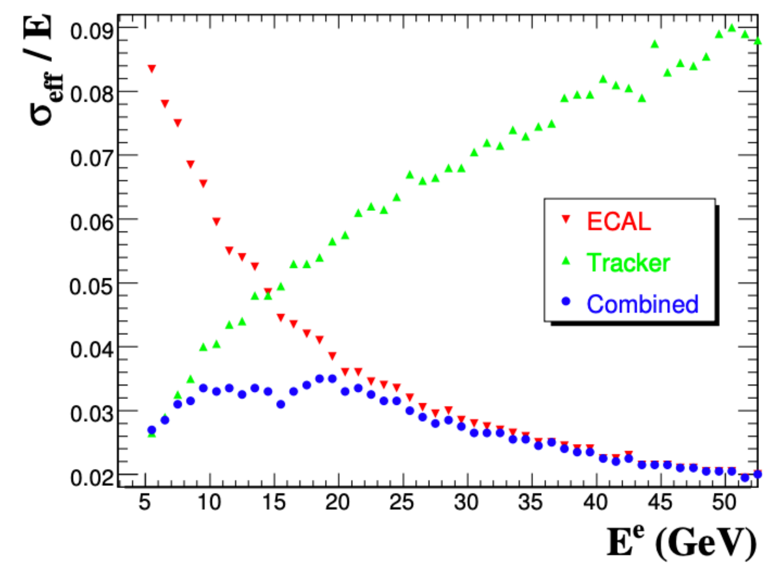
see L. Gouskos talk later



Electron momentum estimate

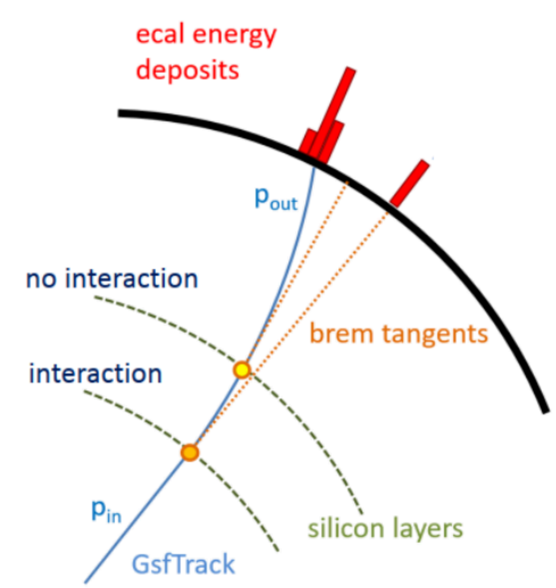
- Currently momentum/energy measurement of electrons is an error-weighted measurement from ECAL/tracking, as done for hadrons, basically resulting :
  - p_{track} if $p < p_0$
  - E_{ECAL} if $p > p_0$
    - **at FCC-ee == p_{track} (low energy)**

$$E^{\text{reco}}_{\text{combined}} = \frac{E_{\text{ECAL}}/\sigma_E^2 + p_{\text{tracker}}/\sigma_p^2}{1/\sigma_E^2 + 1/\sigma_p^2}$$



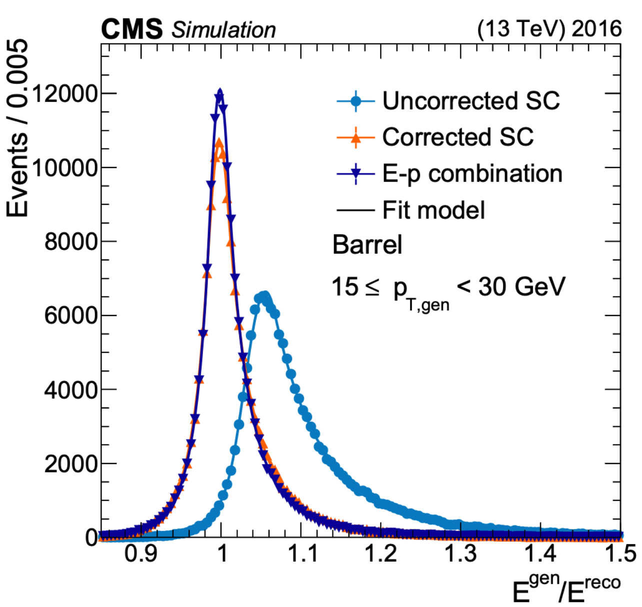
- Brem not taken into account in Delphes currently:
  - modelling energy loss directly in Delphes could be done but probably overkill
    - brem is recovered in real conditions
      - summing ECAL cluster in phi
      - p_{in} - p_{out}
      - and regressions using advanced ML algos exploiting more low-level observables
    - adds (non-gaussian) fluctuations to energy measurement, plus inefficiencies ..

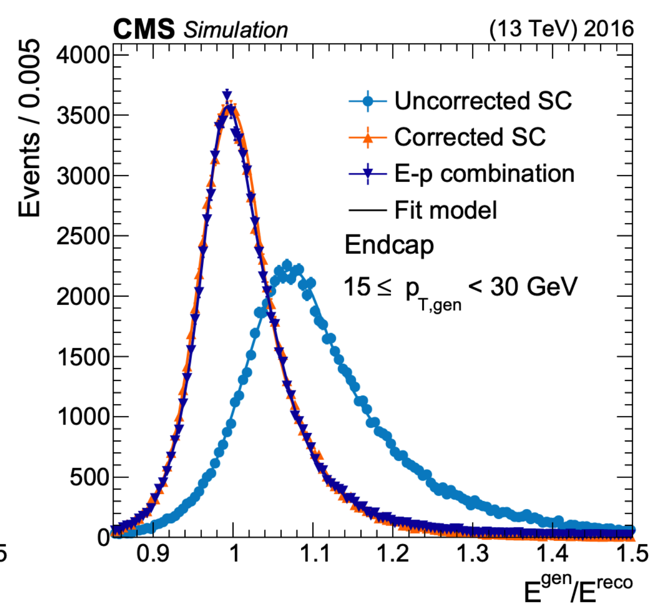
Bremsstrahlung

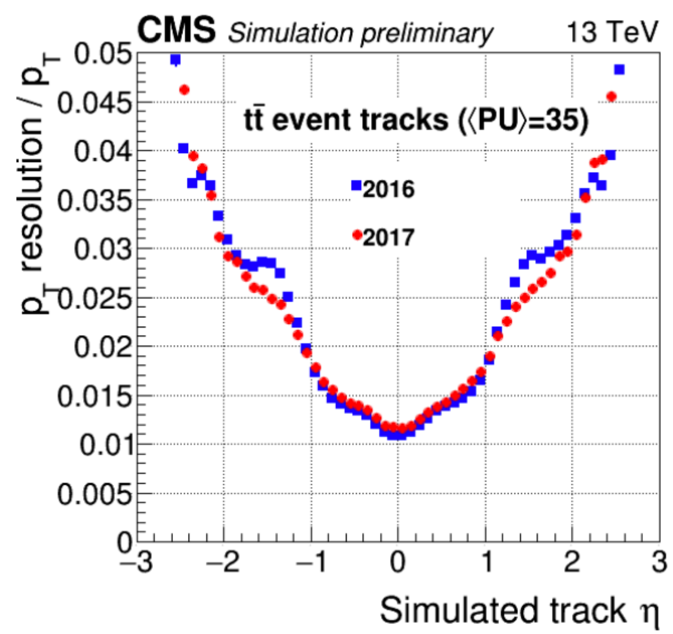


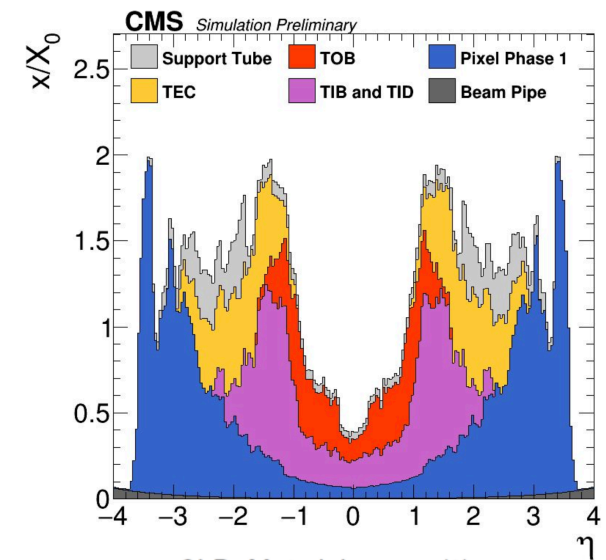
Electron momentum estimate

Delphes fast simulation

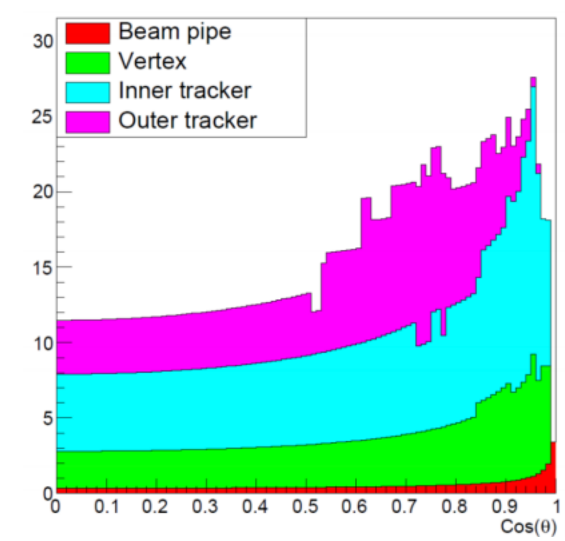




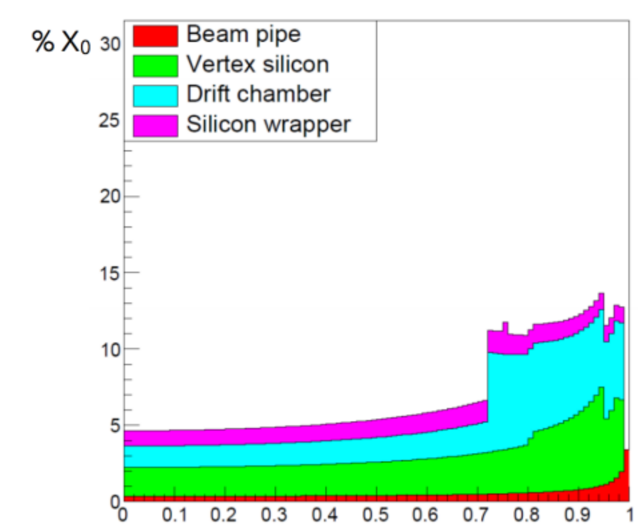




CLD: Material vs. cos(θ)



IDEA: Material vs. cos(θ)

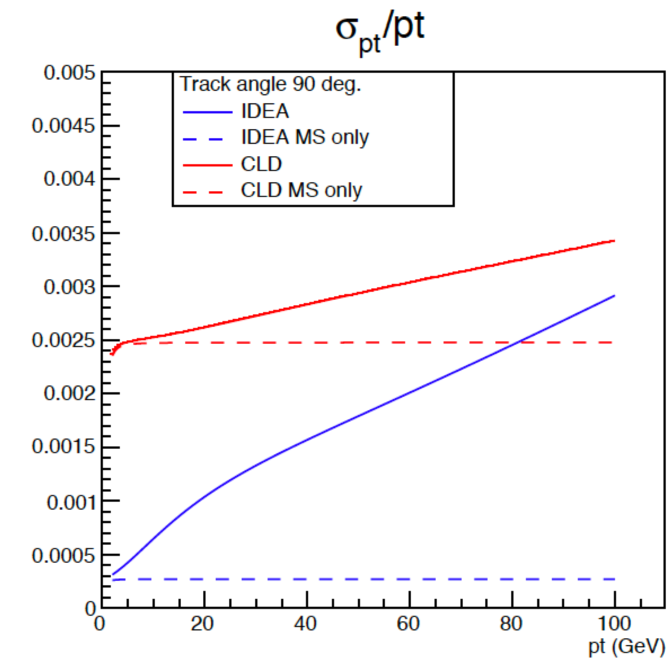


- In CMS, at low p_T :
  - σ/E (barrel) ~ 1-2 % (non-gaussian)
  - σ/E (endcap) ~ 2.5-5 % (non-gaussian)
    - not far from nominal tracking resolution!!
- Seems to suggest that σ_{barrel} / σ_{endcap} ~ $X0_{barrel}$ / $X0_{endcaps}$

X/X0	Barrel (X/X0)	Endcap (X/X0)
CMS	0.6	1.5
CLD	0.12	0.25
IDEA	0.05	0.10

Electron momentum estimate

σ/E	Barrel	Endcap
CMS	0.01-0.02	0.025 - 0.05
CLD	0.002-0.004	0.005-0.10
IDEA	0.001-0.002	0.0025-0.005



- Simple proposal:
  - take tracking resolution x2 worse than nominal tracking for electrons to account for brehm
    - note: still always get optimal resolution from tracking vs. Calo (possible exceptions forward electron)
  - ID Efficiency: 99% → 95%

Bug fixes: low momentum tracks

- Inefficiencies were observed for very low momentum tracks:
  - Delphes Efficiency module parameterises efficiency vs (p_T, η) using **position** vector (at calorimeter) as default
    - low p_T (loopers tracks) that should pass cuts were vetoed
      - using **momentum** vector instead fixes this
    - IDEA acceptance has also been correctly updated
  - Final track collection is produced by PF algorithm (i.e. by the Calorimeter module in Delphes)
    - Calorimeter vetoes tracks that fall outside acceptance, needed to be added back to final track collection

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

    set EfficiencyFormula {
        (abs(eta) > 2.56)                           * (0.000) +
        (pt < 0.1) * (abs(eta) <= 2.56)             * (0.000) +
        (pt >= 0.1) * (abs(eta) <= 2.56)            * (1.000)
    }
}
```

```
###########################
# Energy flow track merger
###########################

module Merger EFlowTrackMerger {
# add InputArray InputArray
  add InputArray Calorimeter/eflowTracks
  add InputArray ForwardLooperTracks/tracks
  set OutputArray eflowTracks
}
```

```
add Branch EFlowTrackMerger/eflowTracks EFlowTrack Track
```

Bug fixes: electrons/photons

- (DualReadout) Calorimeter module in Delphes performs particle-flow
  - when neutral excess is *significant* neutral particles are created (photons or neutral hadrons)
  - Large amounts of photons are created if parameters are not set correctly (can result in isolation mis-calculation)
  - thresholds have been adjusted (increased) to prevent large amounts of fake photons for being created
    - more thorough tuning of parameters accounting for isolation needed
    - lepton "dressing" is an option as well

- Minor bug in the DR calorimeter geometry (tower η = [2.98; 3.00] missing) fixed

```
    set ECalEnergyMin 0.5                      473      set ECalEnergyMin 0.5
    set HCalEnergyMin 0.5                      474      set HCalEnergyMin 0.5
-   set ECalEnergySignificanceMin 1.0          475  +   set ECalEnergySignificanceMin 3.0
-   set HCalEnergySignificanceMin 1.0          476  +   set HCalEnergySignificanceMin 3.0
```

Conclusion

- A Delphes Detector configuration of the **FCC-ee IDEA detector** exists
- The IDEA Delphes configuration includes correct description of:
  - **tracking** performance
  - **dual readout** calorimeter
  - PID tools (dNdx/timing)

- Can/should already be used for both **object reconstruction performance** and **physics studies / optimise detector design**

Backup

Detector Simulation

- Full simulation (GEANT):
  - **simulates** all **particle-detector interaction** (e.m/hadron showers, nuclear interaction, brem, conversions)

10^2-10^3 s/ev

- Experiment Fast Simulation (ATLAS, CMS ..)
  - **simplify** geometry, smear at the level of detector hits, frozen showers

10-10^2 s/ev

- Parametric simulation (**Delphes**, PGS):
  - **parameterise detector response** at the particle level(efficiency, resolution on tracks, calorimeter objects)
  - reconstruct **complex objects** and observables(use particle-flow, jets, missing ET, pile-up ..)

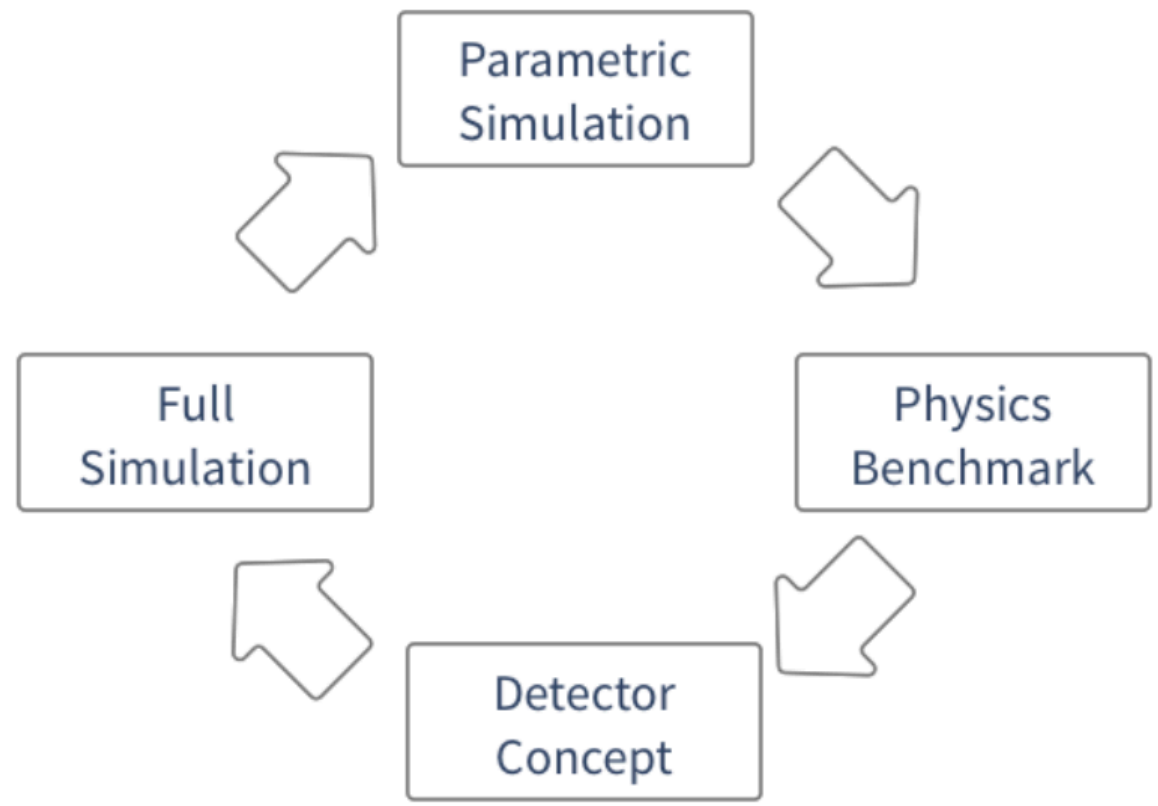
10^{-2} - 10^{-1} s/ev

- Ultra Fast (ATOM, TurboSim):
  - from parton to detector object (smearing/lookup tables)

Parametric simulation paradigm

Why fast parametric detector simulation?

- Easily scan detector parameters
- Reverse engineer detector that maximises performance
- Preliminary sensitivity studies for key physics benchmarks



→ (usual) paradigm adopted in the context of FCC studies