Particle Id at Z/Higgs factories

Michele Selvaggi

(in collaboration with F.Bedeschi and L. Gouskos)
CERN

November 11, 2021 - CEPC workshop

Why Particle Id?

- Particle Identification is crucial for virtually all final state:
  - Exclusive physics:
    - B, D, τ physics (Z pole)
      - E.g : CP violation in $B_s \to D_s K$:
        - $D_s \to \varphi(KK)\, \pi^+$ (charged), $D_s \to \varphi(KK)\, \rho^+(\pi^0\pi^+)$ (neutral)
    - Rare H decays ($\gamma\gamma, \tau\tau, \mu\mu, Z\gamma$)
  - Inclusive hadronic decays (flavour tagging)
    - $H \to jj$ $Z \to jj$ (Z crucial for calibration)
      - j=u,d,s,c,b,(g)

Challenge: cover wide variety of final states in a wide energy range (1-50 GeV)

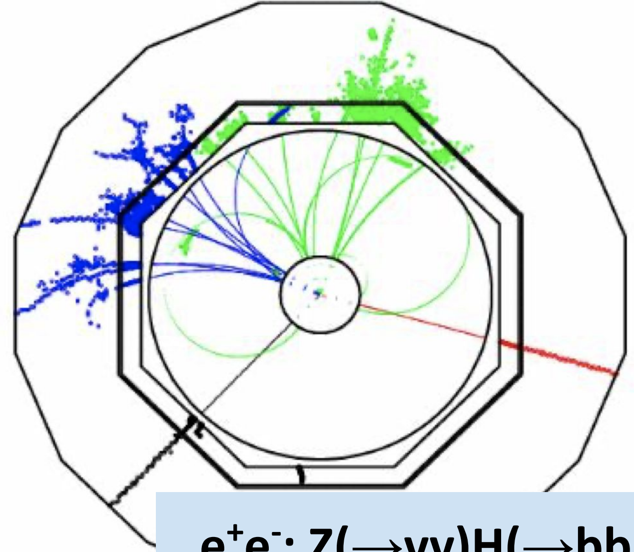
How Particle Id?

- Lepton, photon ID:
  - e/ɣ separation: tracking , particle-flow
  - γ/π^0: Calorimeter granularity (see next talk)
  - e/pi: tracker / calorimetry / particle-flow
  - mu/pi: detector design (pi containment), pointing muon detector ?

- π/K/p separation:
  - dE/dx (dN/dx) method: p ~ 5-30 GeV
  - Time of flight: p ~ 1-5 GeV
- K_L/n:
  - Calorimetry + time of flight ?
- Indirect particle Id:
  - Secondary vertex reconstruction
    - $b \to c \to s$ decay chains
    - $K_s \to \pi\,\pi$

will focus mainly on jet flavour tagging in this talk

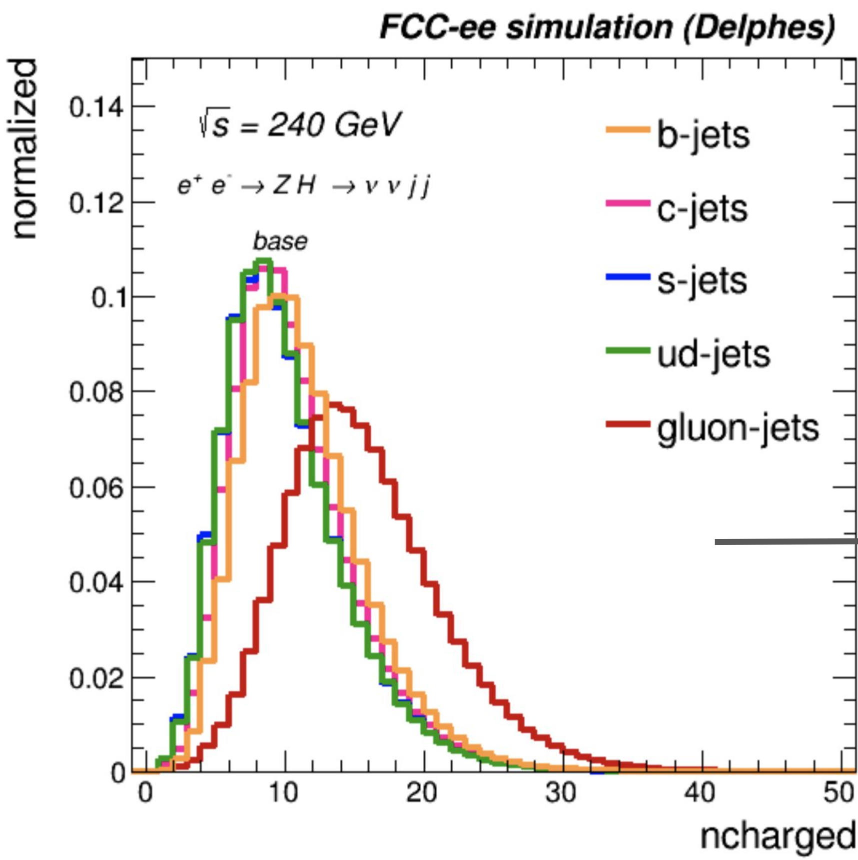
Flavour tagging

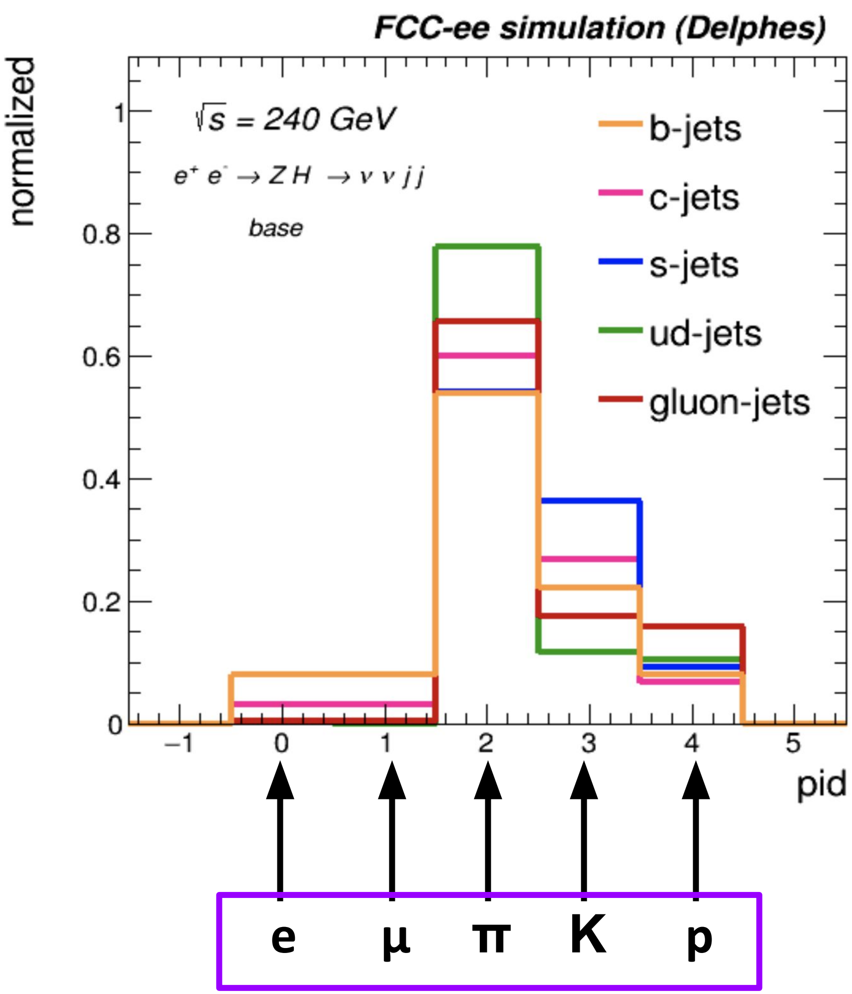


e^+e^-: $Z(\rightarrow \nu\nu)H(\rightarrow bb)$

- **Goals:**
  - Develop a versatile jet flavour tagger for the FCC-ee:
    - Identify with high purity light / strange / charm / beauty jets
      - multi-class classifier
  - Understand the detector requirements/optimize design
    - vertexing and **PID capabilities** of the FCCee detector

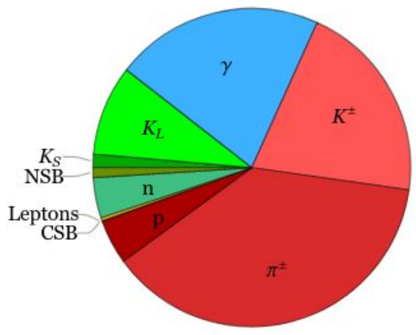
Particle content in jets



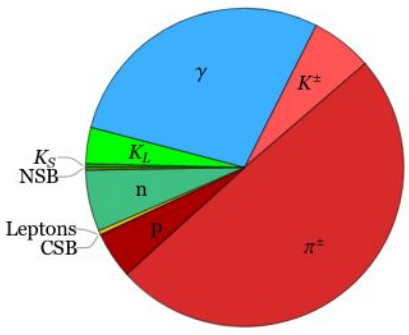


Basics of flavour tagging (strange)

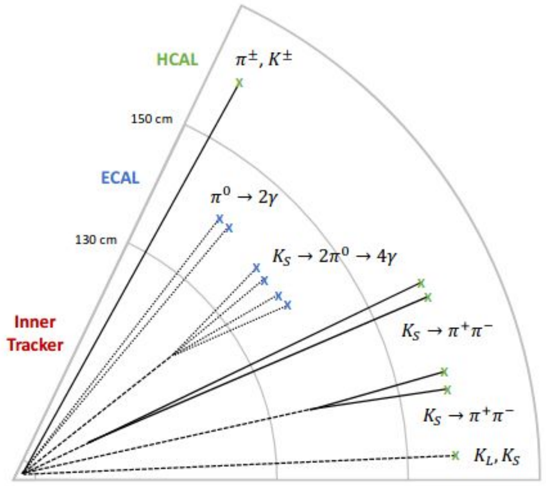
[2003.09517] Momentum weighted fraction:



Strange $p_T = 45\,\mathrm{GeV}$



Down $p_T = 45\,\mathrm{GeV}$



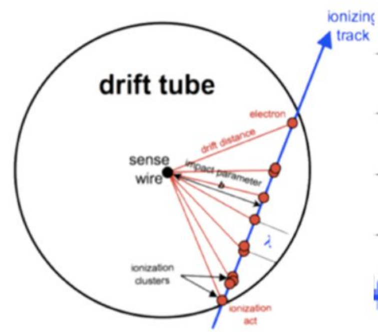
- Large Kaon content
  - Charged Kaon as track:
    - K/pi separation
      - TOF
      - dEdx/dNdx
  - Neutral Kaons:
    - $K_S \to \pi\pi$
      - Displaced 2 track vertex
      - 4 photons
    - K_L
      - TOF vs n ?

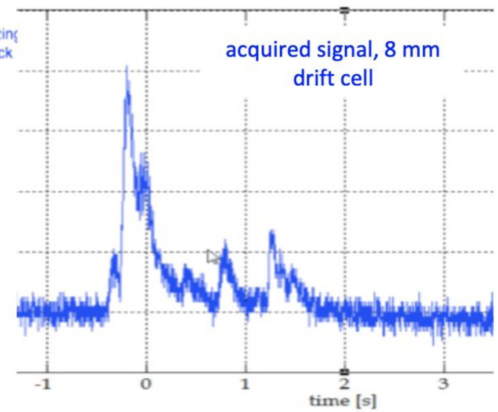
Detector constraints:

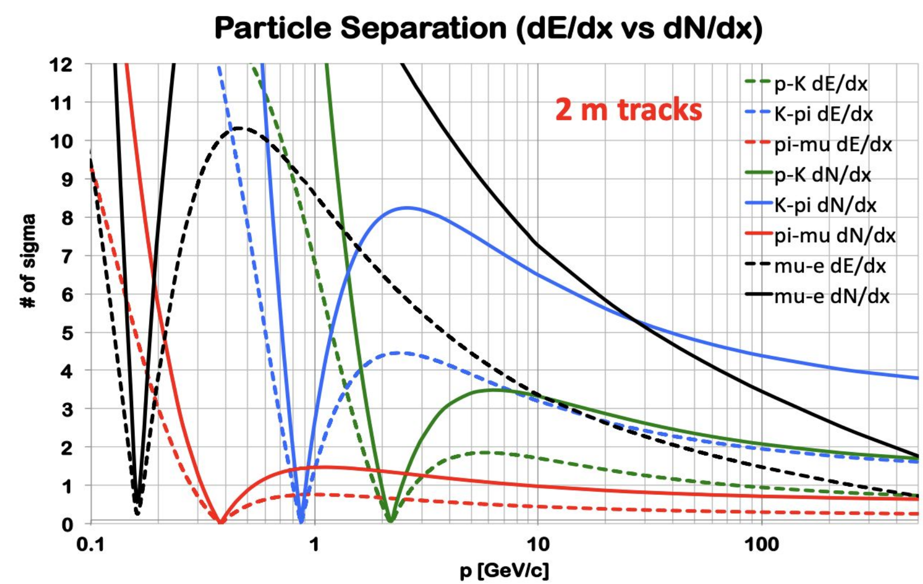
Need power pixel/tracking detectors
- good spatial resolution
- timing detectors
- charged energy loss (gas/silicon)

Cluster counting dN/dx

- Signals from primary ionisations can be separated by few ns
- Count number of **primary ionisation** clusters along track path dN_{cl}/dx → **poisson distributed**
- Avoids large landau flukes that one gets with traditional dEdx method





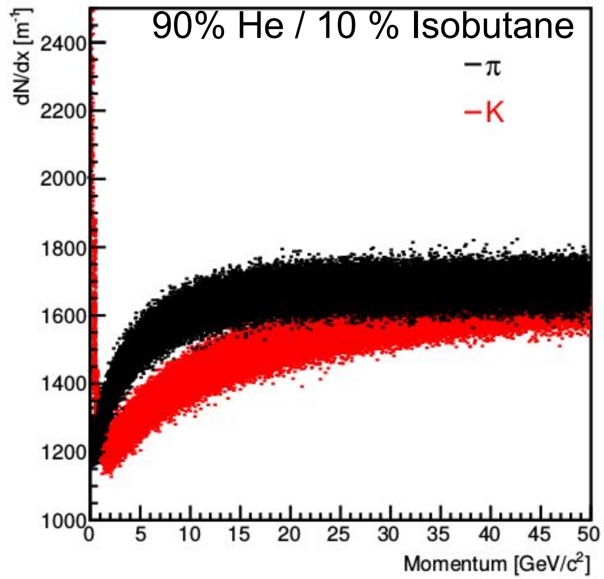


Cluster counting dN/dx

- Count number of **primary ionisation** clusters along track path
- Module added in Delphes

```
###################
# 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

}
```

IDEA detector:



covers wide momentum range (~2-30) GeV

Time-of-flight

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

- Ingredients
  - t_F : final time (with MTD/ calorimeter)
  - L: path length
  - Charged:
    - t_V: vertex time
      - σ_t (beamspot) ~ 12 ps (lower bound, can be further constrained using all tracks)
        - can assume $t_v = t_v$(MC)
    - p : tracking
  - Neutral:
    - $t_V = 0$
    - E = calorimeter

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

for displaced (e.g. K_s), can always use constraint $t_V = r_V / \beta_V$

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

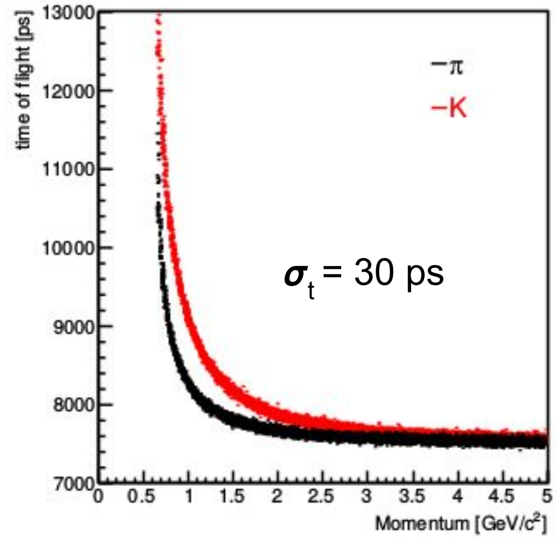
note: E resolution poor at low energy for neutrals, will dominate m_{tof}

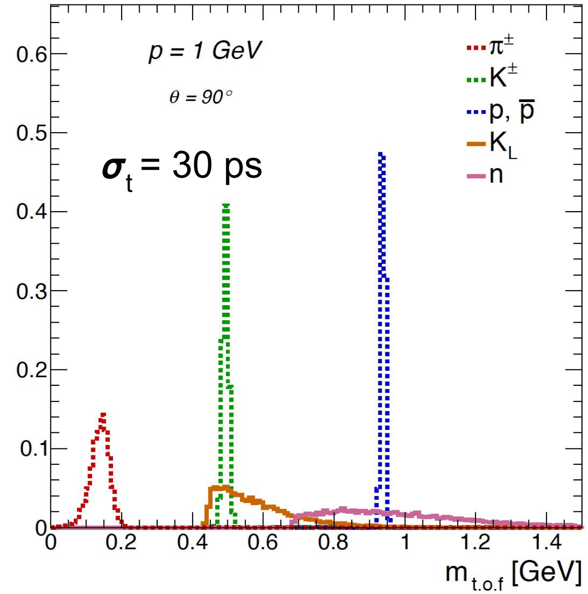
Time-of-flight

- Charged: Allows for good K/pi separation at low momenta:
- Time smearing/ TOF modules implemented in Delphes

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

module TimeOfFlight TimeOfFlight {
  set TrackInputArray 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 2
}
```

$$t_V = \frac{r_V}{\beta_V}$$



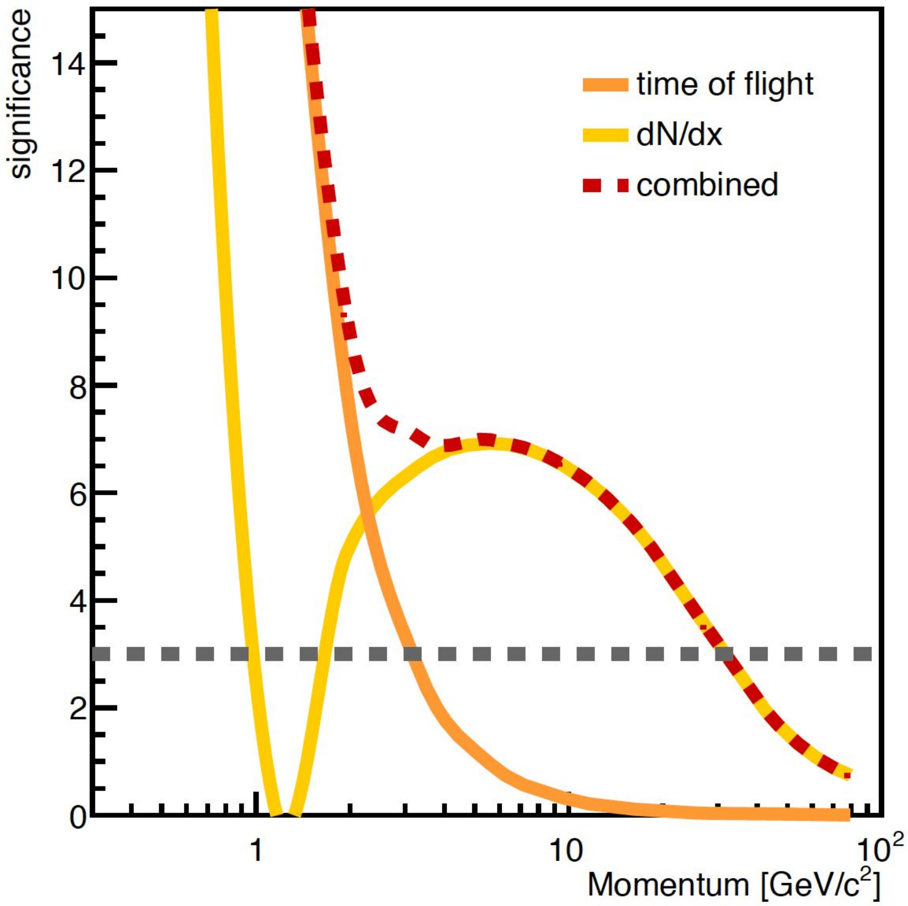


As expected charged (π/K/p) separation > (K_L/n)

Combined PID for charged particles

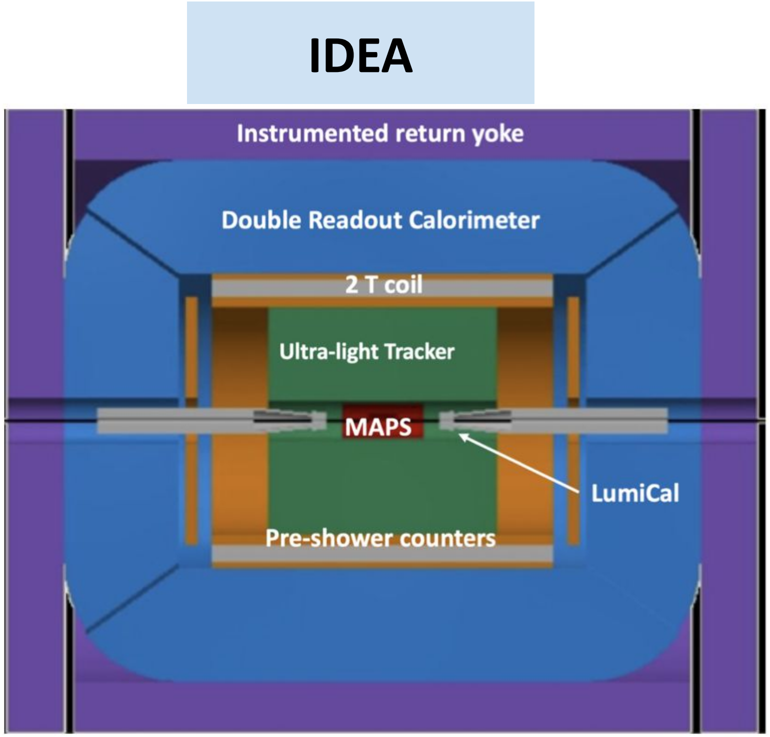
3 std deviation K/pi separation for tracks with p < 30 GeV

σ_t = 30 ps



FCCee detector

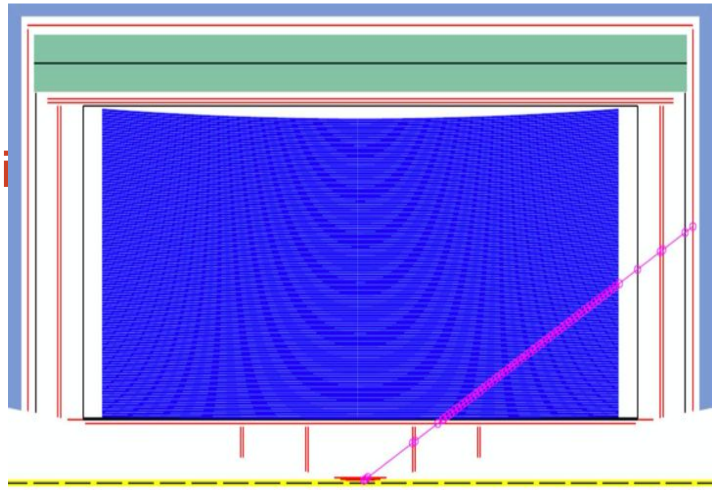
- Ideal for flavour identification [hence: measure Higgs couplings]
  - Impact parameter resolution
    - Low material budget tracker (minimise multiple scattering)
    - Small beam-pipe 1.5 cm -- investigating 1 cm
  - PID capabilities
    - dEdx (Si tracker) -- Cluster counting (Drift)
    - Time of flight -- timing layer



Simulation

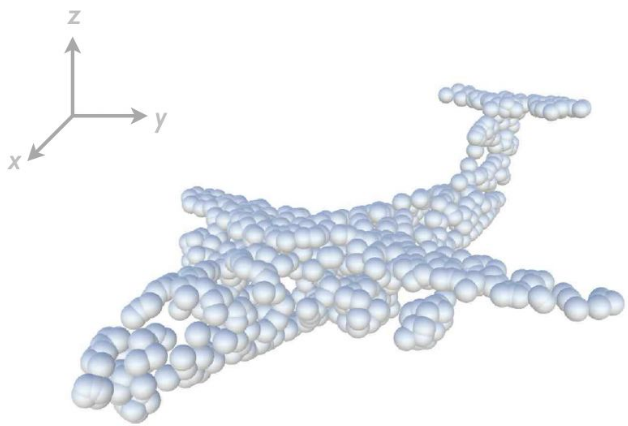
- MC Samples:
  - MG5+Pythia8 used to generate:
    - ee→ ZH→vvXX events (X: g, ud, s, c, b)
    -
- Detector response based on Delphes:
  - Including FastTrackCovariance (F. Bedeschi)
  - Computes:
    - full track covariance matrix (5x5)
      - Including MS
    - smeared track using the off-diagonal terms
    - path length and dN/dx for various gas mixes
    - Time of flight for charged and neutral
    - Allows fast turn-around when trying different detector options
- Jets clusters with the generalized-kT algorithm using *R*=1.5
  - Similar to the anti-kT algorithm [IRC safe]

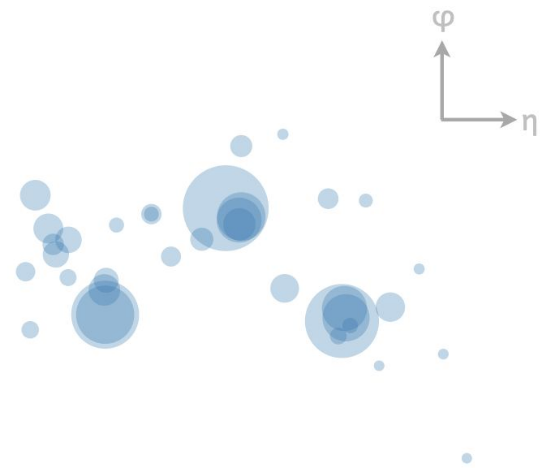
IDEA



Jet tagging as “particle” cloud

check Huilin Qu presentation tomorrow for more details!





Point cloud:

- points (un-ordered)
- input features: (x,y,z) 3D coordinates

Learn “local” structure, move to more “global” features

Particle cloud:

- particles (un-ordered)
- input features:
  - 2D coordinates (eta x phi)
  - momentum
  - charge/ID
  - displacement …

Graph Neural networks:

Generalizing Convolutional neural network for un-ordered/sparse images

DGCNN: [arXiv:1801.07829]

ParticleNet: [arXiv:1902.08570]

Input Features

Table 1. Set of input variables

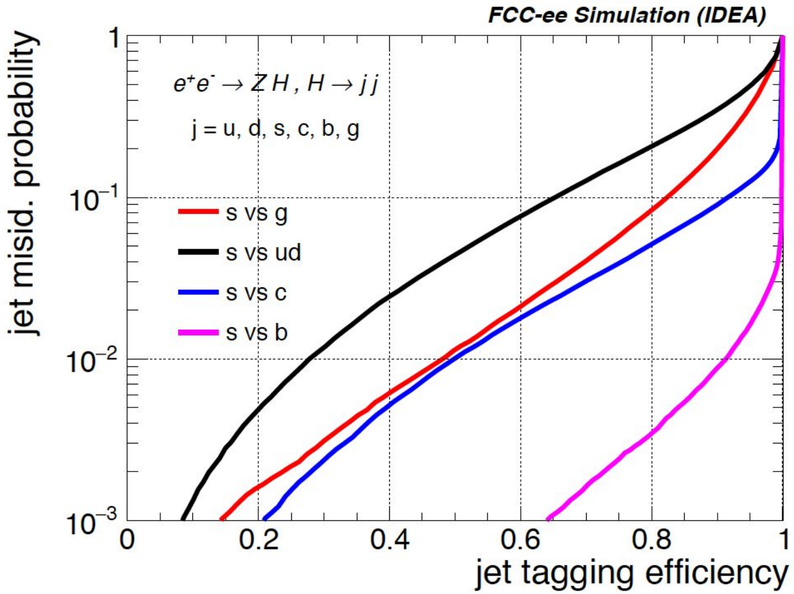
Variable	Description
	Kinematics
$E_{\mathrm{const}}/E_{\mathrm{jet}}$	energy of the jet constituent divided by the jet energy
$\sin(\theta_{\mathrm{jet,const}})$	sin of the angle between the constituent momentum and the jet momentum
$\cos(\theta_{\mathrm{jet,const}})$	cos of the angle between the constituent momentum the jet momentum
	Displacement
$\mathrm{SIP}_{2\mathrm{D}}$	signed 2D impact parameter of the track
$\mathrm{SIP}_{2\mathrm{D}}/\sigma_{2\mathrm{D}}$	signed 2D impact parameter significance of the track
$\mathrm{SIP}_{3\mathrm{D}}$	signed 3D impact parameter of the track
$\mathrm{SIP}_{3\mathrm{D}}/\sigma_{3\mathrm{D}}$	signed 3D impact parameter significance of the track
$d_{3\mathrm{D}}$	jet track distance at their point of closest approach
$d_{3\mathrm{D}}/\sigma_{3\mathrm{D}}$	jet track distance significance at their point of closest approach
	PID
q	electric charge of the particle
$m_{\mathrm{t.o.f.}}$	mass calculated from time-of-flight
dN/dx	number of primary ionisation clusters along track
`isMuon`	if the particle is identified as a muon
`isElectron`	if the particle is identified as an electron
`isPhoton`	if the particle is identified as a photon
`isChargedHadron`	if the particle is identified as a charged hadron
`isNeutralHadron`	if the particle is identified as a neutral hadron

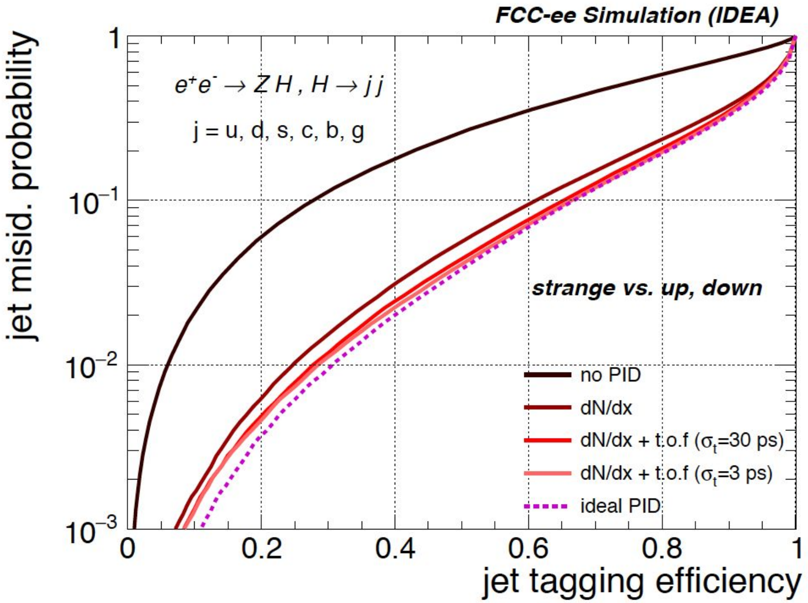
Inputs:
75 particles/jet

Training details:

- 1M jets split equally between classes
- 30 epochs
- Still room for improving the training details

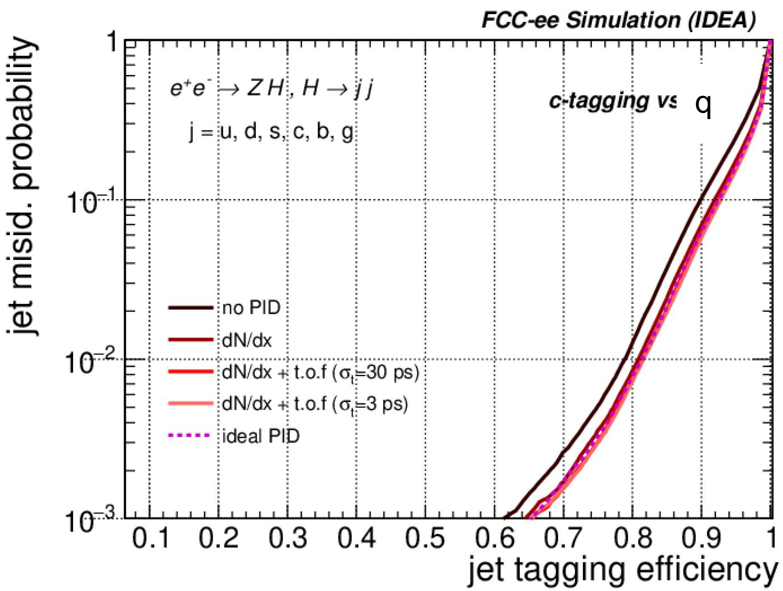
Strange tagging

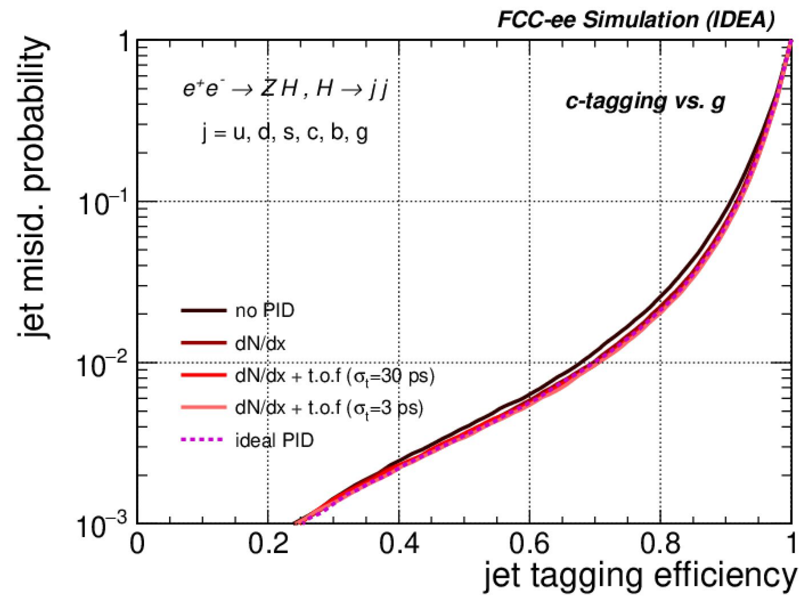




- Small room for improvement on the PID, in particular for strange tagging
  - TOF/dNdx methods complementary
    - 3 ps vs 30 ps resolution does not seem to make a large difference
    - dNdx + 30 ps timing seems to offer close to optimal separation power

charm/bottom tagging





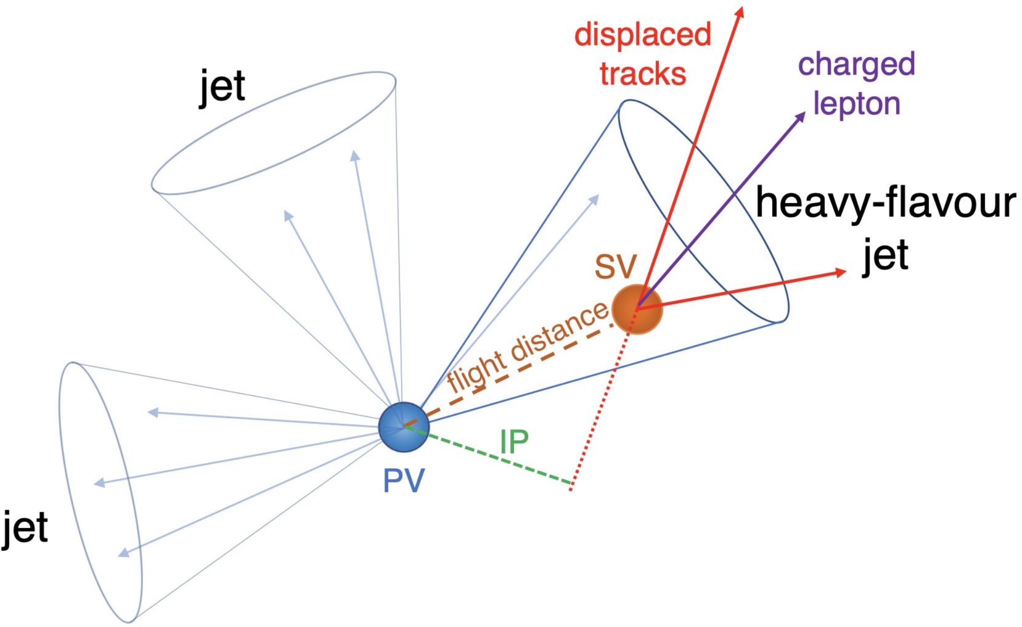
- Expect smaller improvement from PID vs strange tagging
- Small room for improvement on the PID, in particular for strange tagging
  - TOF/dNdx methods complementary
    - 3 ps vs 30 ps resolution does not seem to make a large difference
    - dNdx + 30 ps timing seems to offer close to optimal separation power

Summary & outlook

- A first version of a jet identification algorithm based on **PF candidates** and **PID** and **advanced ML** in place
    - Multi-class classifier b/c/s/ud/g
        - Results promising, in particular for charm and strange tagging
- PRELIMINARY conclusions:
    - Current PID using dNdx with 30 ps timing resolution seems to be close to optimal
        - Usual caveats of Delphes simulation (e/pi/gamma separation optimal)
- Next [short-term] steps:
    - propagate detector design choice to final sensitivity (studying H→ ss sensitivity for example)
    - address tagger calibration (at the Z pole)
    - provide framework for training/testing

Backup

Basics of flavour tagging (b/c)



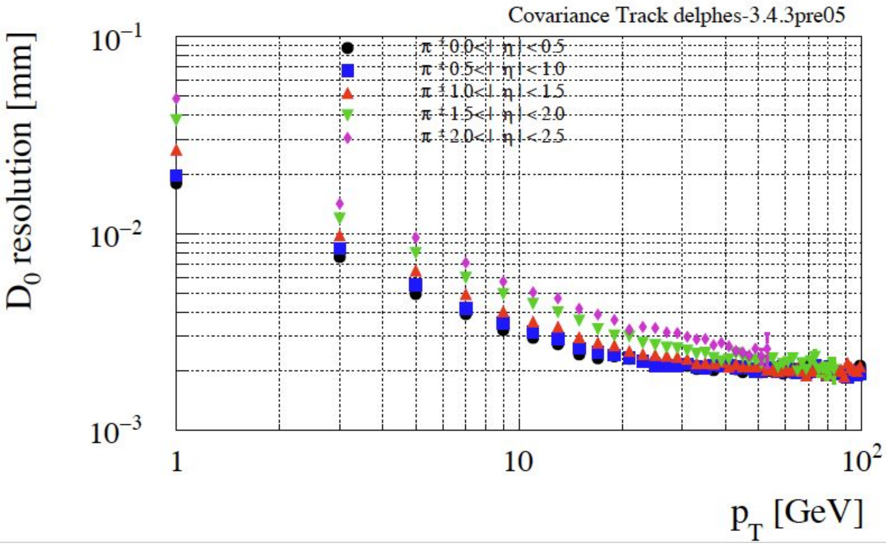
Detector constraints:
Need power pixel/tracking detectors
- Good spatial resolution
- As little material as possible
- Precise track alignment

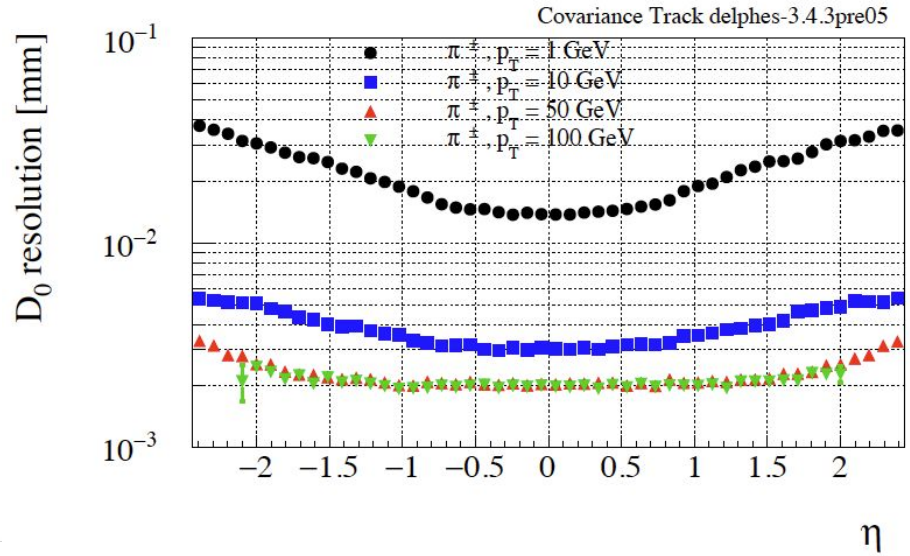
- Large lifetime
  - b (c) lifetime ~ps (~0.1ps)
  - b (c) decay length: ~5 (2-3) mm for ~50 GeV boost
- Displaced vertices/tracks
  - Large impact parameters
  - Tertiary vertices when B hadron decays to C hadron
- Large track multiplicity
  - ~5 (~2) charged tracks/decay
- Non-isolated e/μ
  - ~20 (10)% in B (C) decays

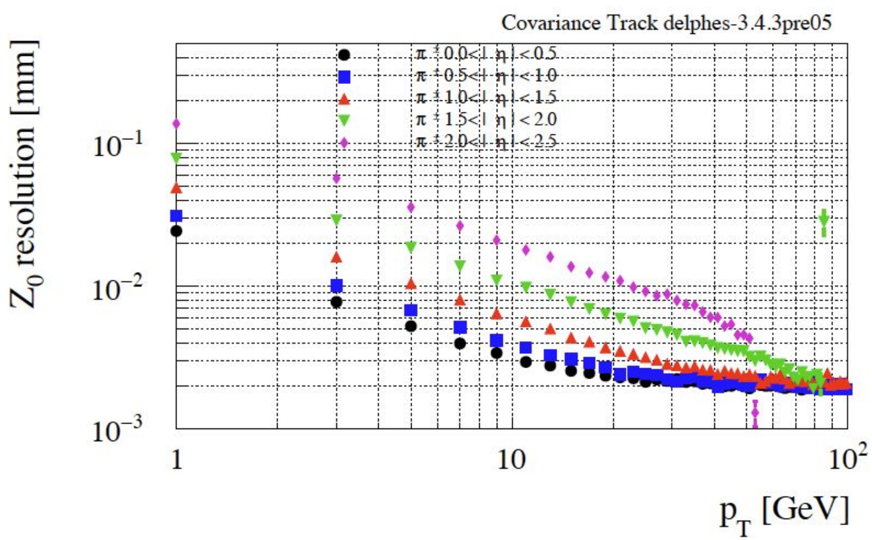
Impact parameter performance

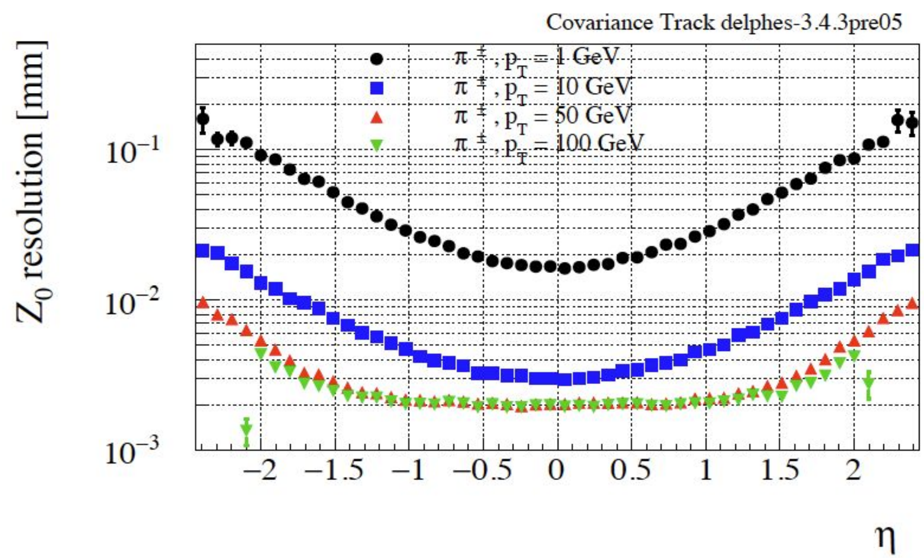
Credits to Sylvie Braibant

IDEA detector:







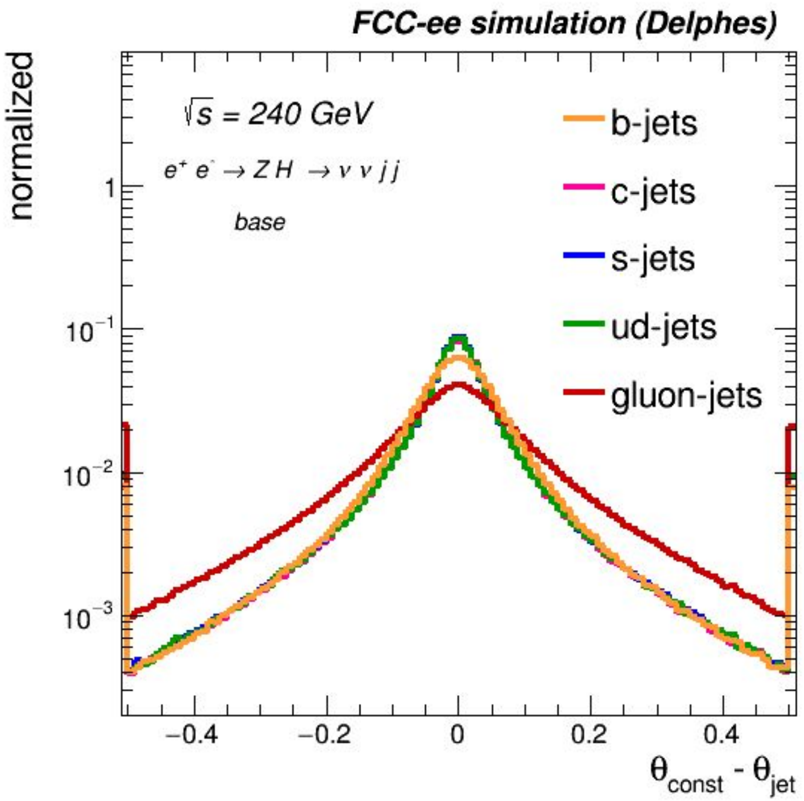


2μm IP resolution at high-p_T

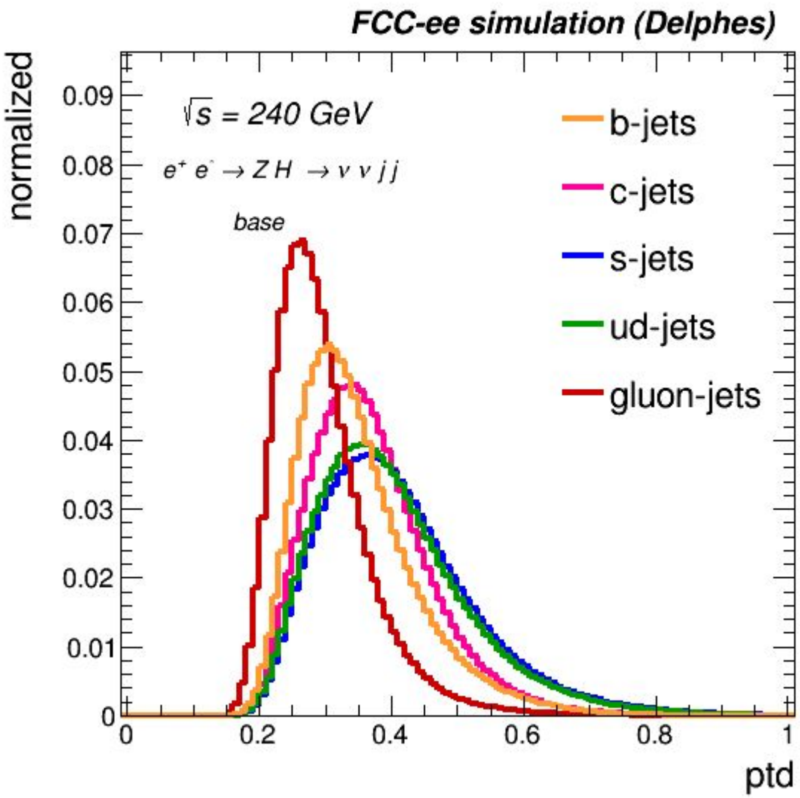
Input variables

- Comparison of input distributions for different jet flavors

Projection || to jet axis

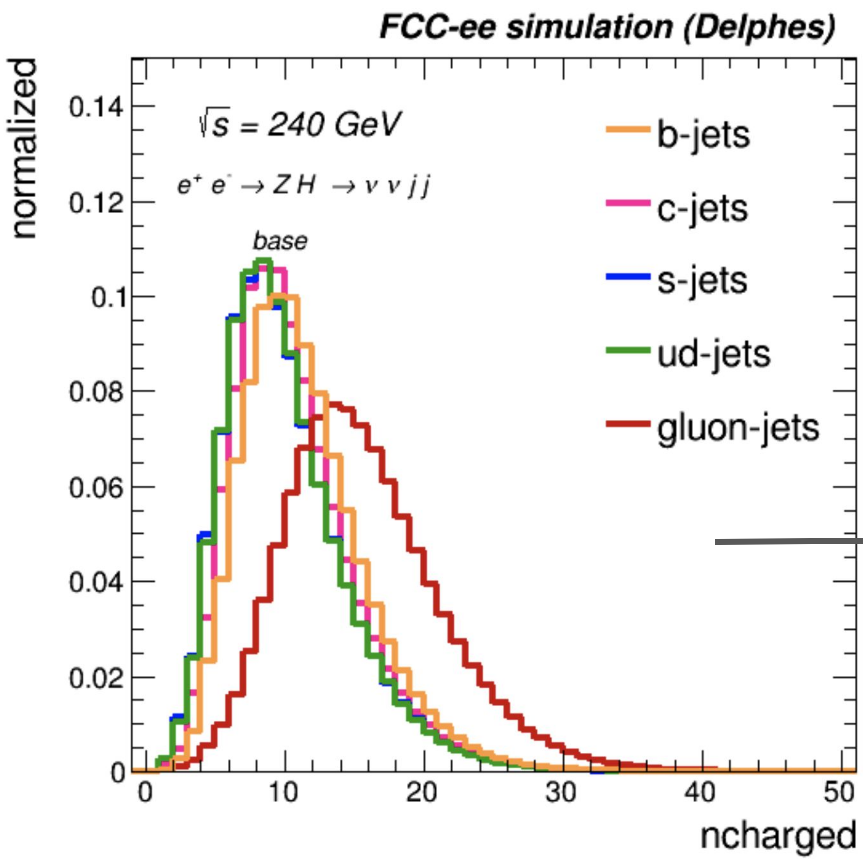


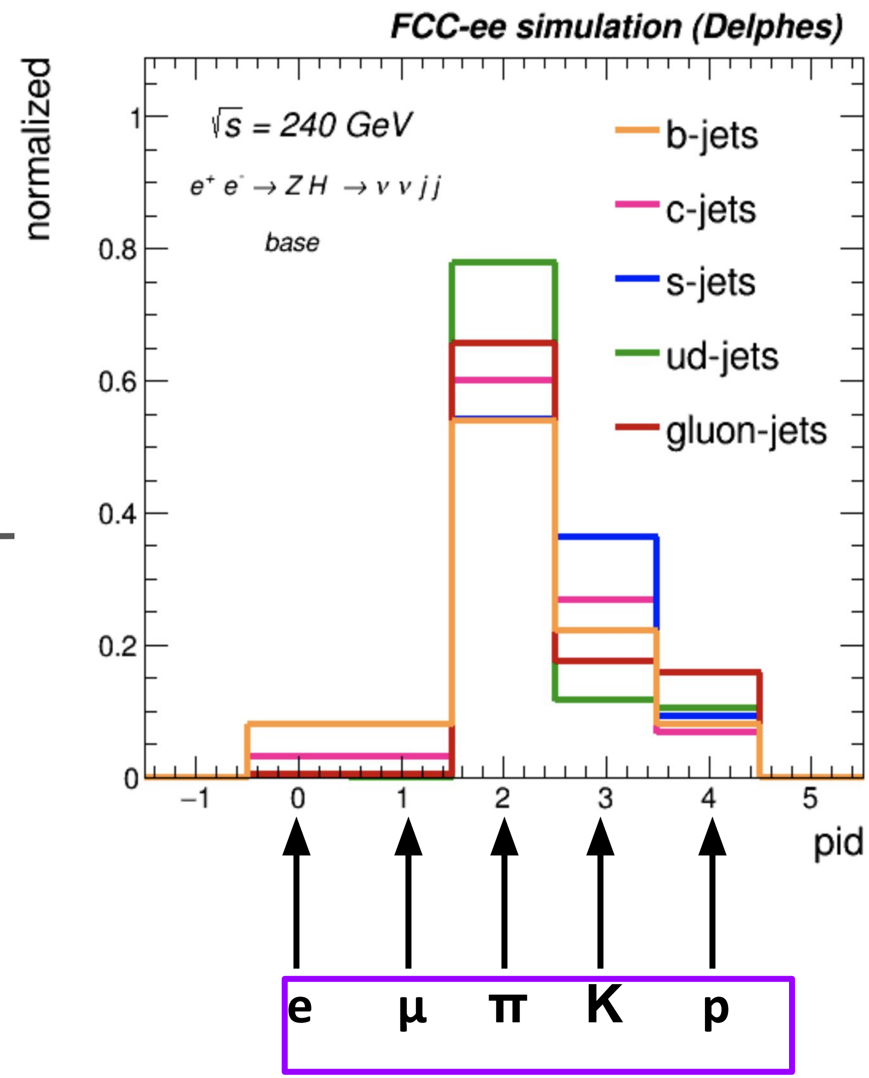
$p_T D$



- More comparisons:
  https://selvaggi.web.cern.ch/selvaggi/FCC/FCCee/FlavourTagging/

Performance w/ PID





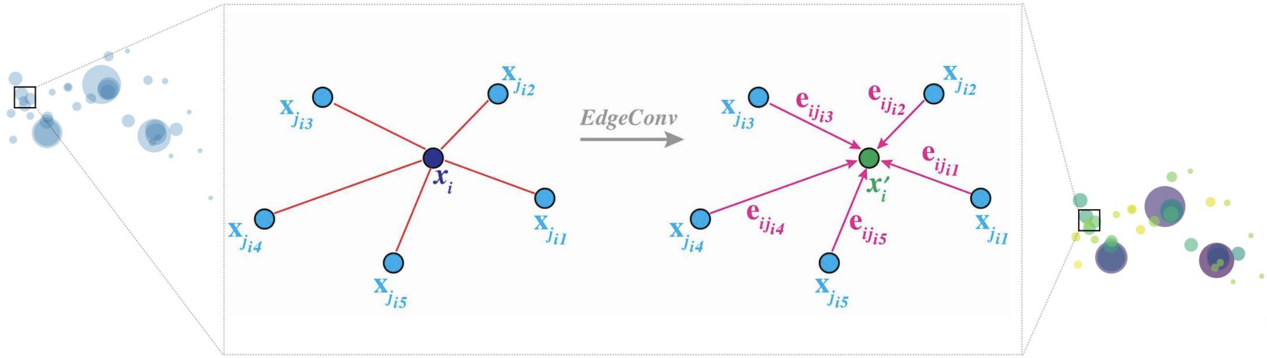
no PID: only charge

realistic: e, μ, mtof, dNdx

perfect PID: e, μ, + π, K, p from MC truth

Convolution on point cloud: EdgeConv

DGCNN: [arXiv:1801.07829]



<u>EdgeConv:</u> convolution on a graph

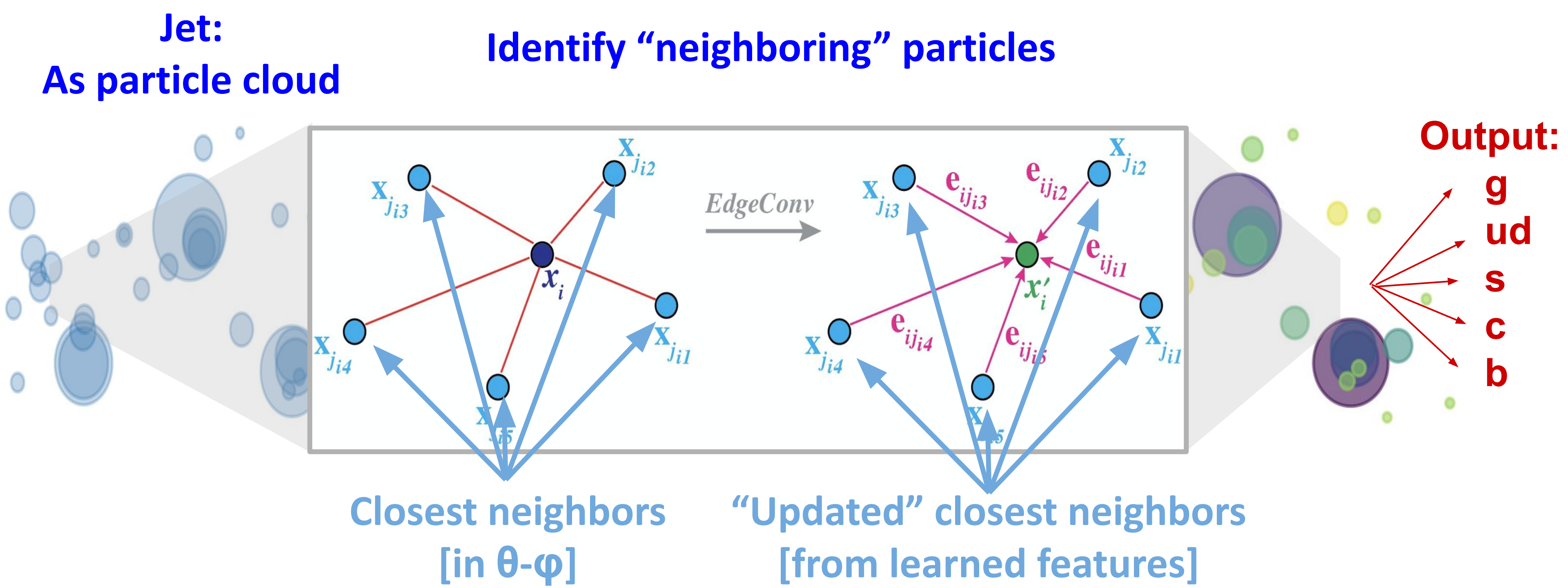
- **point cloud** is treated as **graph**, where each point is a **vertex**
- **local patch** defined by finding k-nearest neighbours
- **convolution** function:
    - define "edge feature" for each center-neighbour pair
        - $e_{ij} = h(x_i, x_j)$
    - aggregate all the features **symmetrically**:
        - $x'_i = \text{mean}_j\, e_{ij}$

<u>Key point:</u>

Generalizing CNN for
un-ordered/sparse images

Flavour tagging using ParticleNet

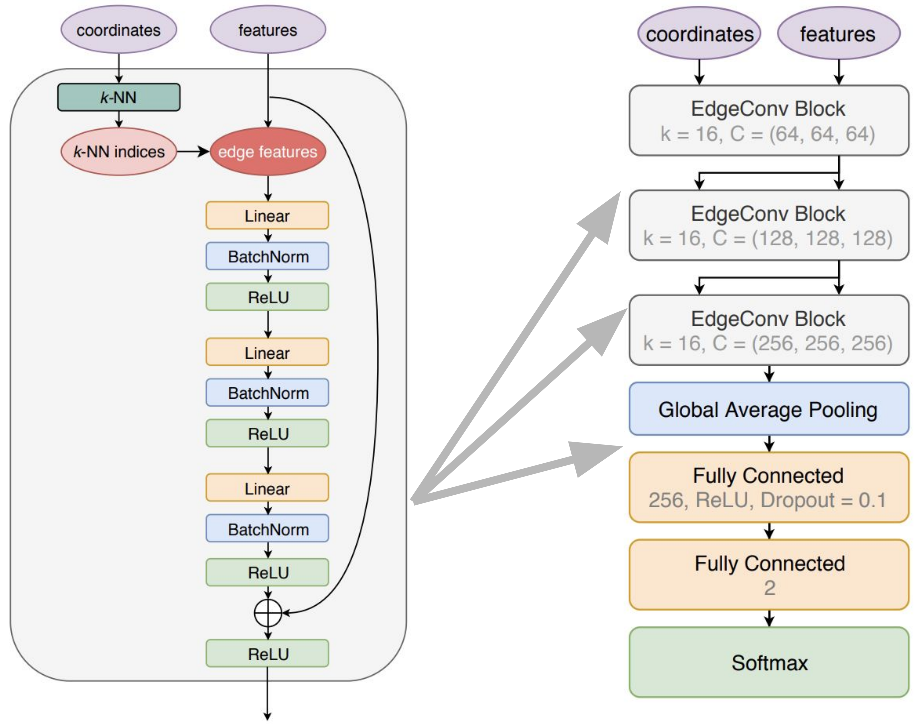
- Developing a flavour tagging algorithm based on ParticleNet
  - Jet is represented as a “particle cloud”
- Follow a hierarchical learning approach:
  - **First:** Learn “local” structures; **Then:** move to more “global” features
  - Treat the particle cloud as a graph
    - **Particles** are the **vertices** of the graph
    - **Relationships** between the particles are the **edges** of the graph



ParticleNet

ParticleNet: [arXiv:1902.08570]

- **local neighborhood** information automatically incorporated
- **EdgeConv** layers can be **stacked** (as CNNs), and learn **local** (shallow layers) and **global** features (deep layers)
- **new features** provide new coordinates (in some abstract latent space) to compute "local patch" in new iteration



EdgeConv block *ParticleNet architecture*

Designing a jet flavour tagging algorithm

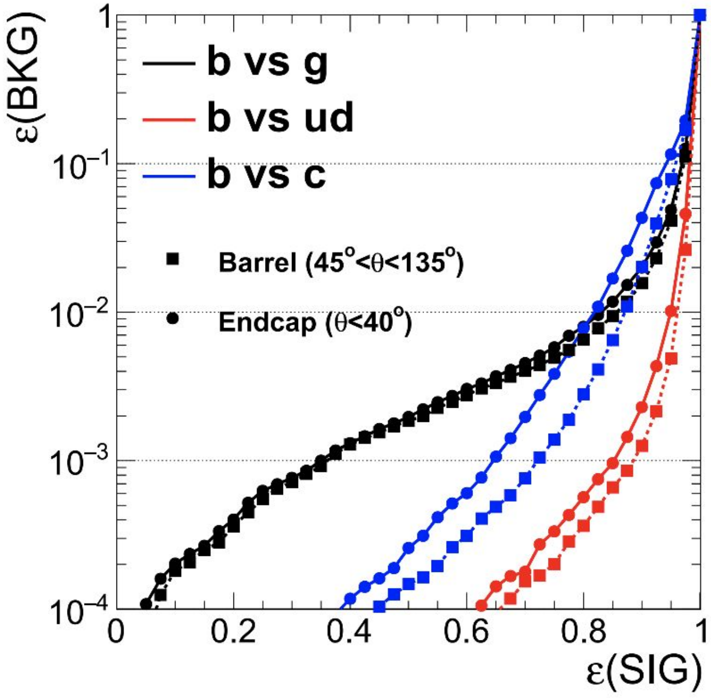
- How to represent a jet is one of the key aspects of algorithms for jet tagging
  - Improve performance → extend physics reach
  - Lead to fresh insight into jets → deepen our understanding of jet physics

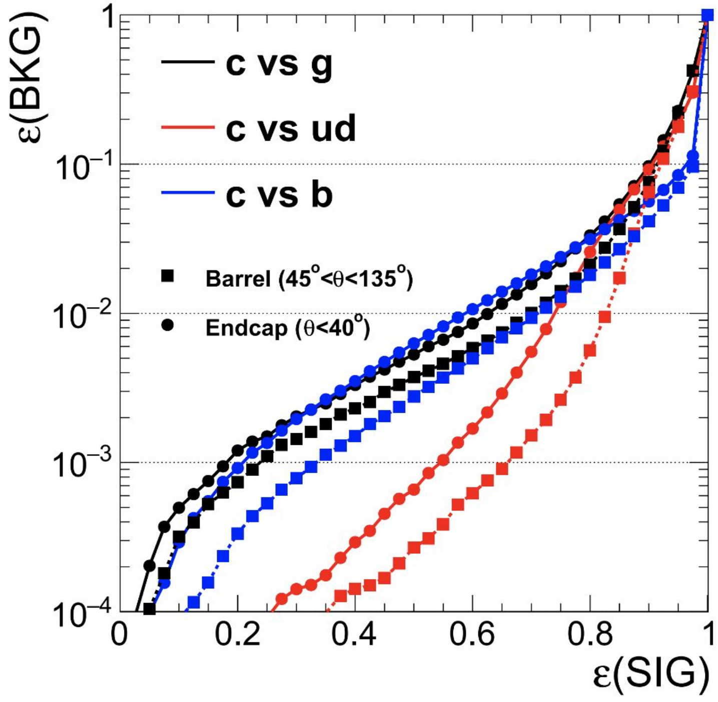
- Particles [associated to each jet] are intrinsically unordered
  - i.e., ordering by p_T(particle) or displacement from PV: suboptimal
  - Primary information: 2D coordinates in theta-phi space
  - Include additional features / particle: energy, displacement, charge, track quality, PID ...

Performance vs theta (b/c)

b-tagging | **c-tagging**

PRELIMINARY !! (LOW STATS TRAINING)





Tracking in Delphes

pattern recognition
track fitting

Full Simulation

-most accurate
-least flexible
-does not allow for change of geometry

parameterisation
(d_0, d_z, p, ctg θ, φ)

Delphes

TkLayout

-standalone tool
-predicts tracking performance for a given geometry
-allows for quick turnaroud
- no further implementation needed in Delphes
-requires intermediate step

FastTrackCovariance

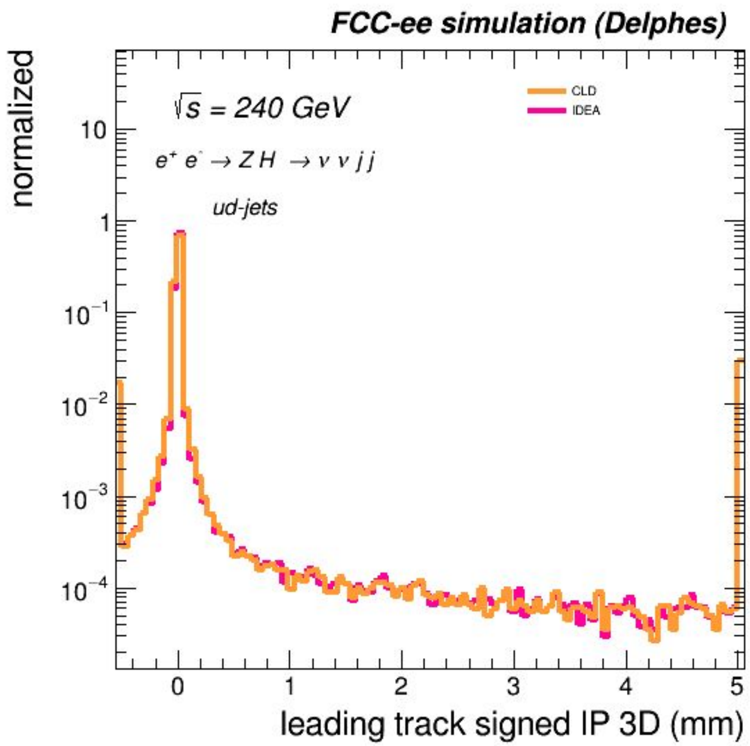
-(for now) standalone tool
-can be plugged into Delphes
-predicts performance given geometry
-allows for quick turnaroud
-not implemented in Delphes yet

23

Comparison: IDEA vs. CLD

- No big differences between in input variables between IDEA & CLD
  - small difference in material budget observed on light jets since dxy ~ 0
    - expect slightly better performance for IDEA detector for discrimination vs light

ud-jets



c-jets