Study of dE/dx with new software and new samples

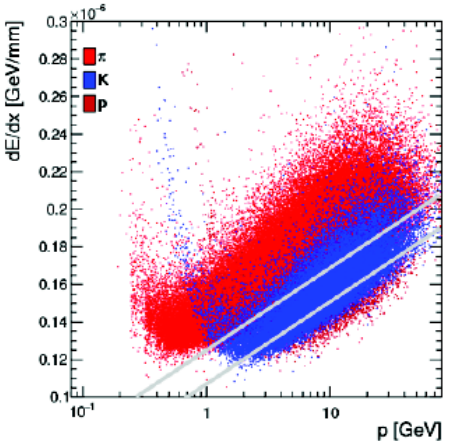
A. Irles, *28th Nov 2018, ILD Software/Analysis meeting*

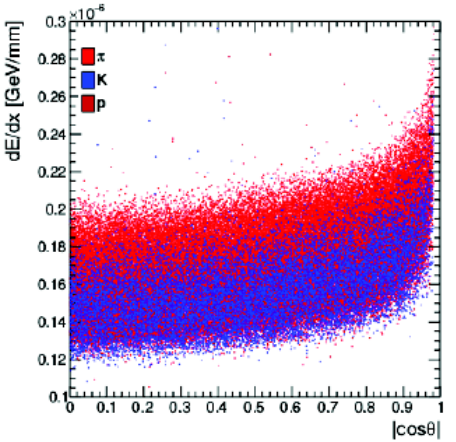
Reminder: dEdx for DBD e+e-→bb (250 GeV)

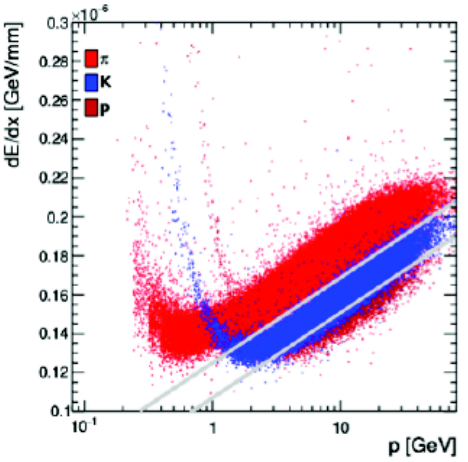
- ParticleTagger processor developped during S. Bilokin thesis
- Angular correction: $\frac{dE}{dx} \rightarrow \frac{dE}{dx}\theta^{0.15}$,
- Simple parametrization of dEdx vs log(p).
- Plots from S.B. thesis

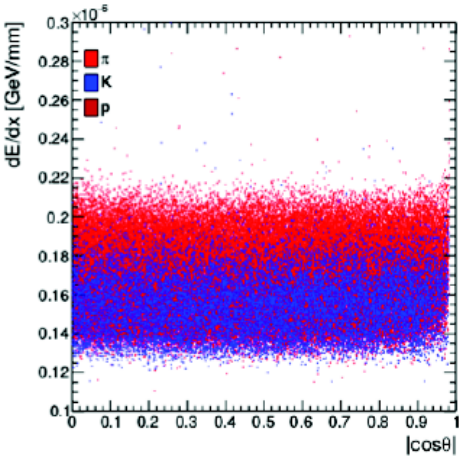
Generated \ Reconstructed	π	K	p
p	223	252	1249
K	1342	14693	713
π	97911	215	22

- Values calculated using 250GeV bb REC samples
- p>3 GeV









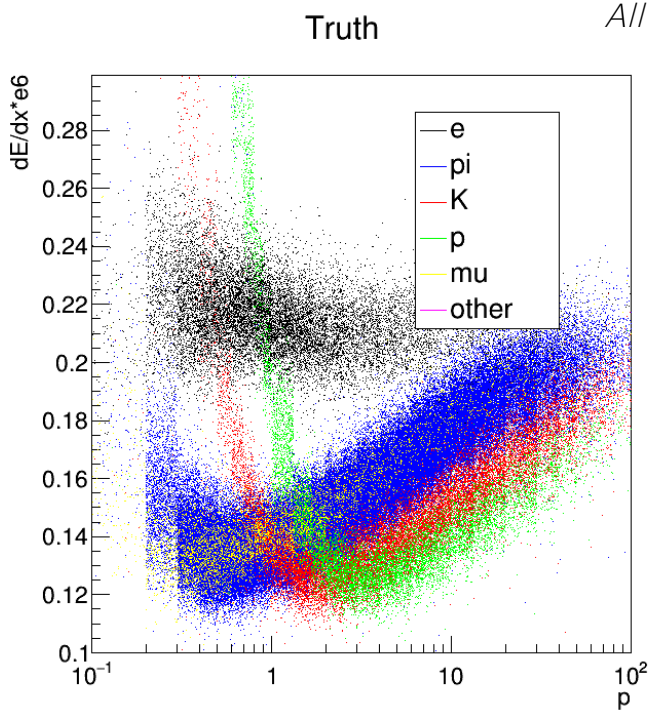
Kaon Identification using dEdx

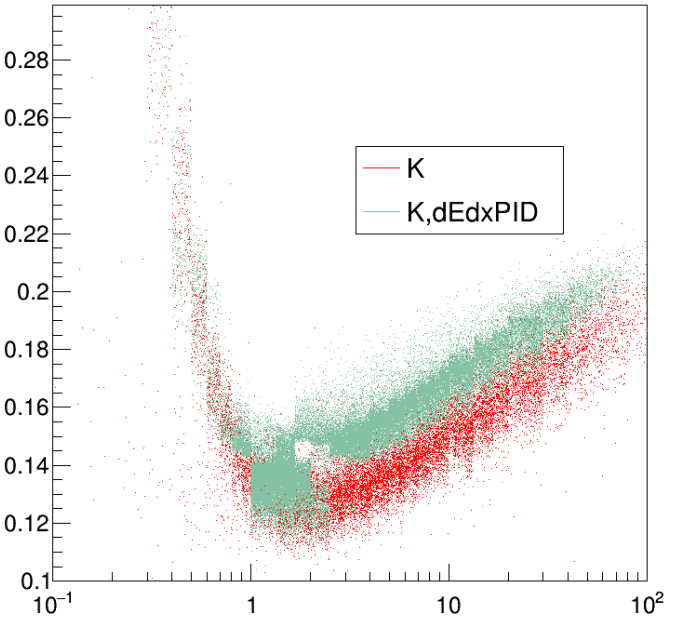
- ParticleTagger is not used in the 250GeV analysis, but its results are used for “cheating”
  - dEdx information is not available in the DBD DST files
  - We use the purity and efficiency values to select kaons from through secondary vertexes.
- dEdx info is now available in the DST samples for the IDR. Also, the angular correction seems to be by default applied. Even more, we have now access to PID restulst *(PIDHandler → see backup)*
  - DEdxPID → motivated by S.B’s studies but including low p ID.
  - LikelihoodPID → including info from dEdxPID
  - How do they perform for kaon identification?

```
collection name : PandoraPFOs
parameters:

-------------- print out of ReconstructedParticle collection --------------

 flag:  0x0
parameter PIDAlgorithmTypeID [int]: 0, 3, 4, 2, 5, 6, 7, 1,
parameter PIDAlgorithmTypeName [string]: BasicVariablePID, LikelihoodPID, LowMomMuID, ShowerShapesPID, TOFEstimators0ps, TOFEstimators10ps, TOFEstimators50ps, dEdxPID,
parameter ParameterNames_BasicVariablePID [string]: electronLikelihood, muonLikelihood, pionLikelihood, kaonLikelihood, protonLikelihood, hadronLikelihood, MVAOutput_mupiSeparation, electronProbability,
uonProbability, pionProbability, kaonProbability, protonProbability, hadronProbability, electron_dEdxdistance, muon_dEdxdistance, pion_dEdxdistance, kaon_dEdxdistance, proton_dEdxdistance,
parameter ParameterNames_LikelihoodPID [string]: electronLikelihood, muonLikelihood, pionLikelihood, kaonLikelihood, protonLikelihood, hadronLikelihood, MVAOutput_mupiSeparation, electronProbability, muo
nProbability, pionProbability, kaonProbability, protonProbability, hadronProbability, electron_dEdxdistance, muon_dEdxdistance, pion_dEdxdistance, kaon_dEdxdistance, proton_dEdxdistance,
parameter ParameterNames_LowMomMuID [string]: electronLikelihood, muonLikelihood, pionLikelihood, kaonLikelihood, protonLikelihood, hadronLikelihood, MVAOutput_mupiSeparation, electronProbability, muonPr
obability, pionProbability, kaonProbability, protonProbability, hadronProbability, electron_dEdxdistance, muon_dEdxdistance, pion_dEdxdistance, kaon_dEdxdistance, proton_dEdxdistance,
parameter ParameterNames_ShowerShapesPID [string]: electronLikelihood, muonLikelihood, pionLikelihood, kaonLikelihood, protonLikelihood, hadronLikelihood, MVAOutput_mupiSeparation, electronProbability, m
onProbability, pionProbability, kaonProbability, protonProbability, hadronProbability, electron_dEdxdistance, muon_dEdxdistance, pion_dEdxdistance, kaon_dEdxdistance, proton_dEdxdistance,
parameter ParameterNames_TOFEstimators0ps [string]: TOFFirstHit, TOFClosestHits, TOFClosestHitsError, TOFFlightLength, TOFLastTrkHit, TOFLastTrkHitFlightLength,
parameter ParameterNames_TOFEstimators10ps [string]: TOFFirstHit, TOFClosestHits, TOFClosestHitsError, TOFFlightLength, TOFLastTrkHit, TOFLastTrkHitFlightLength,
parameter ParameterNames_TOFEstimators50ps [string]: TOFFirstHit, TOFClosestHits, TOFClosestHitsError, TOFFlightLength, TOFLastTrkHit, TOFLastTrkHitFlightLength,
parameter ParameterNames_dEdxPID [string]: electronLikelihood, muonLikelihood, pionLikelihood, kaonLikelihood, protonLikelihood, hadronLikelihood, MVAOutput_mupiSeparation, electronProbability, muonProba
bility, pionProbability, kaonProbability, protonProbability, hadronProbability, electron_dEdxdistance, muon_dEdxdistance, pion_dEdxdistance, kaon_dEdxdistance, proton_dEdxdistance,
```

dE/dx (bb, 500GeV, large model)

All tracks, including primary vertexes



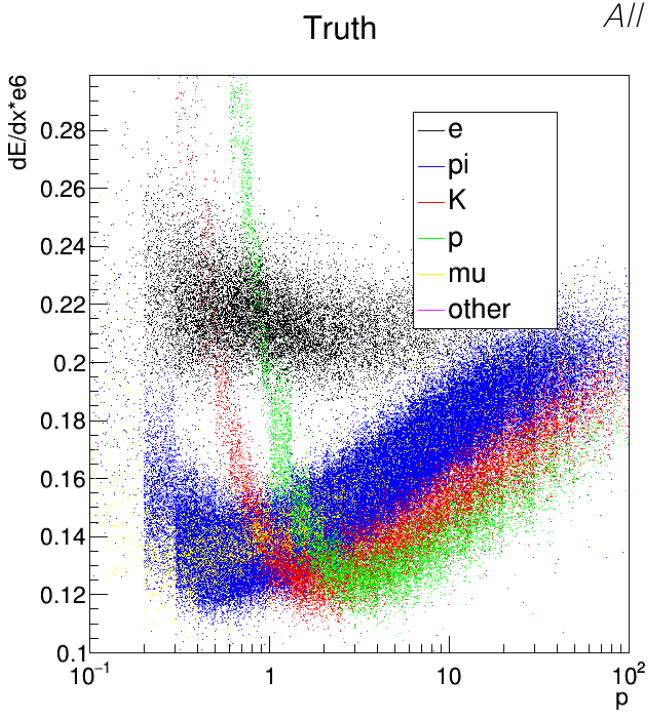


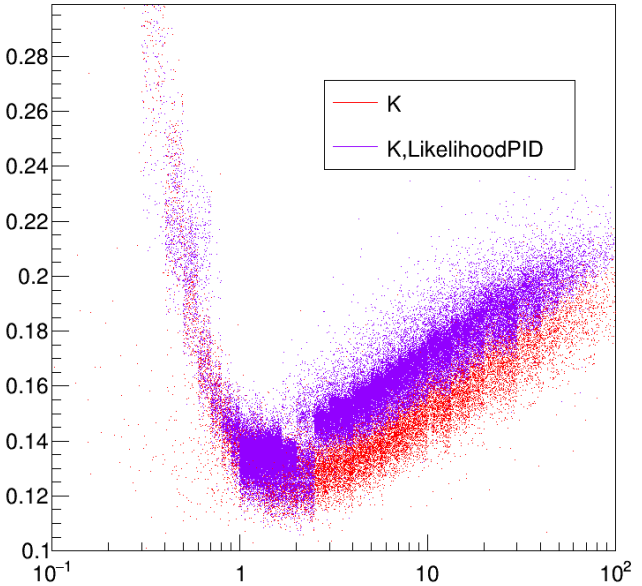
In red, truth kaons (measured dEdx)

- The dEdxPID mistakes kaons and pions at hight momentum. It looks like a simple bug or issue of the parametrization.

dE/dx (bb, 500GeV, large model)

All tracks, including primary vertexes



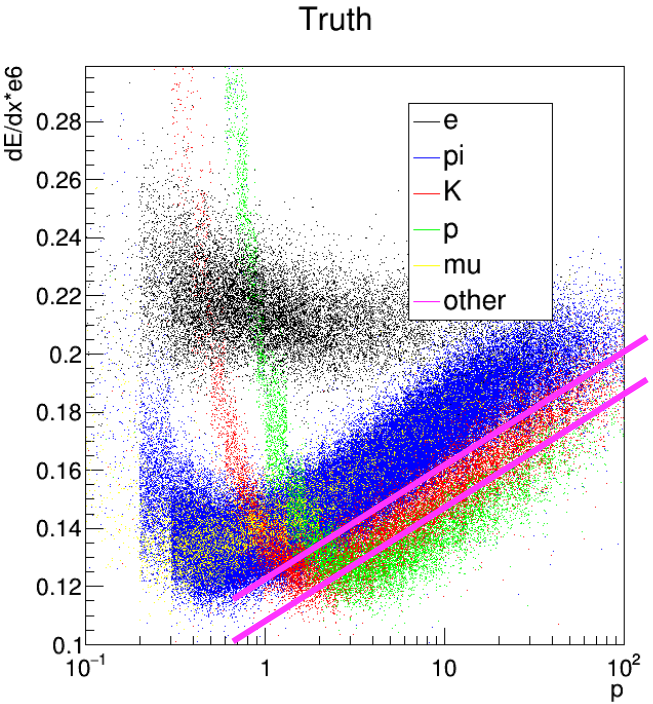


In red, truth kaons (measured dEdx)

- The dEdxPID mistakes kaons and pions at hight momentum. It looks like a simple bug or issue of the parametrization.
- Same for LikelihoodPID since it relies on same algorithm for high momentum.
- For low momentum, both perform better than the simple parametrization. Specially the LikelihoodPID.

dE/dx (bb, 500GeV, large model)

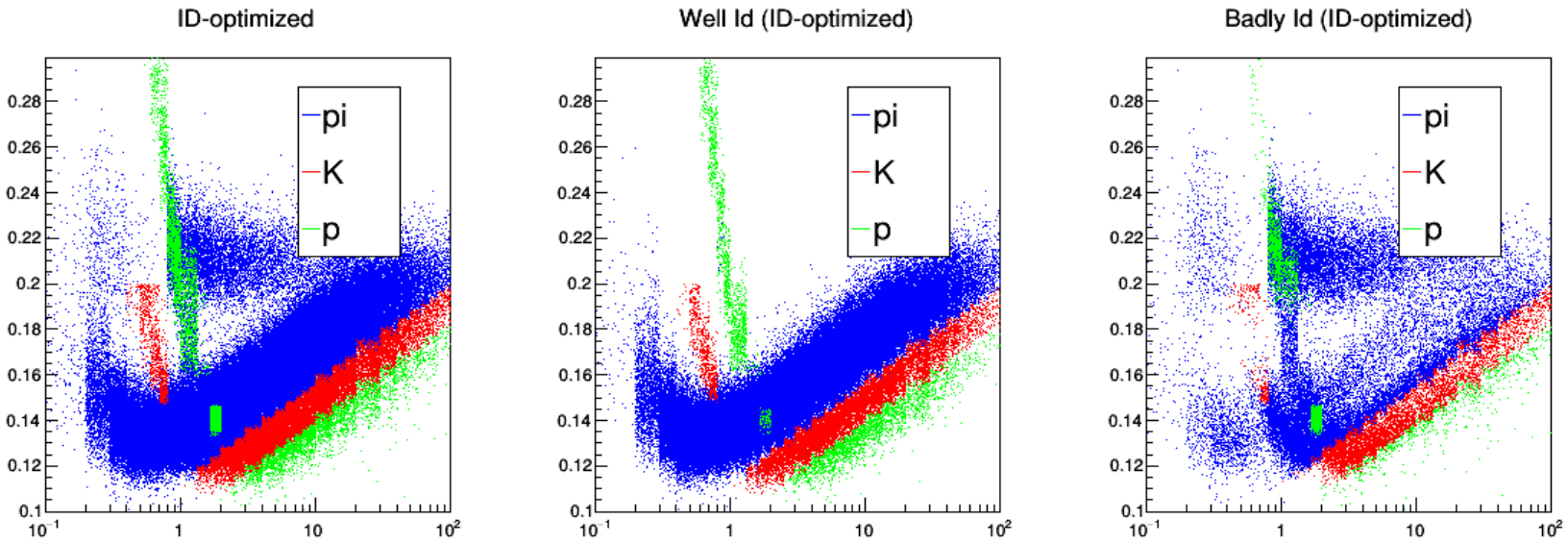
- We want to recover the good kaon identification for high momentum → Use the ParticleTagger processor in our analysis as a patch.
- Basic idea: parametrize the area where the kaon density is larger and use it for particle separation.
- A **new PIDalgorithm** is created and saved in the PIDHandler of the **PandoraPFOs**
  - KaonTagger
  - Without likelihood or probability value.
- Steering file and info about the processor and the PIDHandler in the backup



Using the new KaonTagger PID and the official PIDs

- Strategy: run the ParticleTagger algorithm over the PandoraPFOs and save the KaonTagger PID. In the QQbarProcessor (ttbar or bbbar branch) we use this information and the LikelihoodPID information to select kaons.
- First, I reject particles that:
  - Have low momentum (<2 GeV) and LikelihoodPID says that it is a proton
  - Have low momentum (<0.8 GeV) and LikelihoodPID says that it is a pion
  - Has dEdx>0.20 & LikelihoodPID says that it is a pion or an electron
- If the particle is not rejected, I identify it as a kaon if:
  - Has low momentum (<2 GeV) and LikelihoodPID says that it is a kaon
  - Has large momentum and it is in the area covered by the magenta lines.

Using the new and the official PIDs (bb, 500GeV, large model)

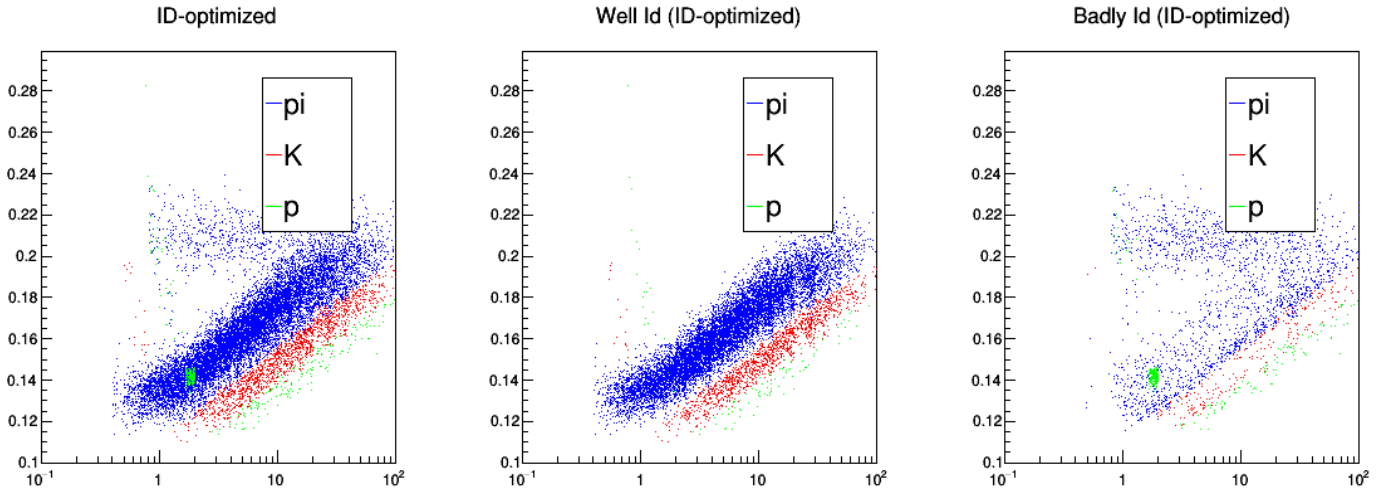


- This plots are not good to measure purities and efficiencies… Let's calculate them

```
  ################ new ID ################
Simulated kaons: 24071, pions: 201012, protons: 13111, electrons: 20019, muons: 6191, other: 20019
Reco, kaons: 16338, pions: 197604, protons: 10003
Well Reco, kaons: 10743, pions: 169682, protons: 5313
Contamination in kaons id, pions:: 1137, protons: 4206, electrons: 159, muons: 39
  ------------------------------------------------------------------
KAON ID: eff =0.678742  purity=0.657547
  ------------------------------------------------------------------
```

Using the new and the official PIDs (bb, 500GeV, large model)

- Only secondary vertexes, with |cos(theta)|<0.95.




```
  ----------------------------------------------------------------------

  ################ new ID ################
Simulated kaons: 2236, pions: 11734, protons: 299, electrons: 867, muons: 405, other: 867
Reco, kaons: 1633, pions: 13315, protons: 414
Well Reco, kaons: 1380, pions: 11345, protons: 105
Contamination in kaons id, pions:: 95, protons: 140, electrons: 14, muons: 4
  ----------------------------------------------------------------------
KAON ID: eff =0.730322  purity=0.84507
  ----------------------------------------------------------------------
```

- Note: adding the low momentum kaons increase the efficiency by ~2% with no loss of purity.

Summary and next steps

- The kaon identification is possible now with the new DST samples.
- The official PID in the DST files have a mistake in the kaon identification for low momentum, easily "patchable" for the ongoing analysis.
- For low momentum, they are well optimized.

- To do (short term)
  - Comparison large vs small model. Same parametrization? What about the angular correction?
  - Same for the b jets from tttbar decay (more similar topologies to bbbar @ 250GeV)

Back-up

How to use the ParticleTagger patch

- Add it to the processor chain, at the beginning or at least before QQbarProcessor.
- Example of steering file is in:
  https://github.com/QQbarAnalysis/ParticleTagger/tree/QQbarAnalysisBranch2018/scripts
- It saves the new PIDalgorithm in the PIDHandler of the PandoraPFOs → this has to be used in your analysis!
  Optimized for kaons.
- It also produces a root file. Example of analysis of this root file is in
  https://github.com/QQbarAnalysis/ParticleTagger/tree/QQbarAnalysisBranch2018/analysis

```
<processor name="MyParticleTagger" type="ParticleTagger">
  <parameter name="ROOTFileName" type="string" > pid_l5.root </parameter>
  <parameter name="NewPID" type="string" > KaonTagger </parameter>
  <parameter name="PFOCollection" type="string" > PandoraPFOs </parameter>
  <parameter name="TrackRelCollection" type="string" > MarlinTrkTracksMCTruthLink </parameter>
  <parameter name="slope_kaon_selection" type="float" > 0.019 </parameter>
  <parameter name="upper_limit_kaon_selection" type="float" > 0.1115 </parameter>
  <parameter name="lower_limit_kaon_selection" type="float" > 0.096 </parameter>
</processor>
```

- How to use the PIDHandler? See example in src/ParticleTagger.cc

```
PIDHandler pidh(pfoCol); //pfoCol is the name of the PandoraPFOs collection
for(int i=0; i<pfoCol→getNumberOfElements(); i++) {
  ReconstructedParticle * recoparticle = static_cast< ReconstructedParticle * > (pfoCol->getElementAt(i));
  int pid = pidh.getAlgorithmID(“LikelihoodPID”);
  int pdg = pidh.getParticleID(recoparticle, pid).getPDG();//this is the PDG obtained by LikelihoodPID
}
```

- https://github.com/QQbarAnalysis/ParticleTagger Branch QQbarAnalysisBranch2018