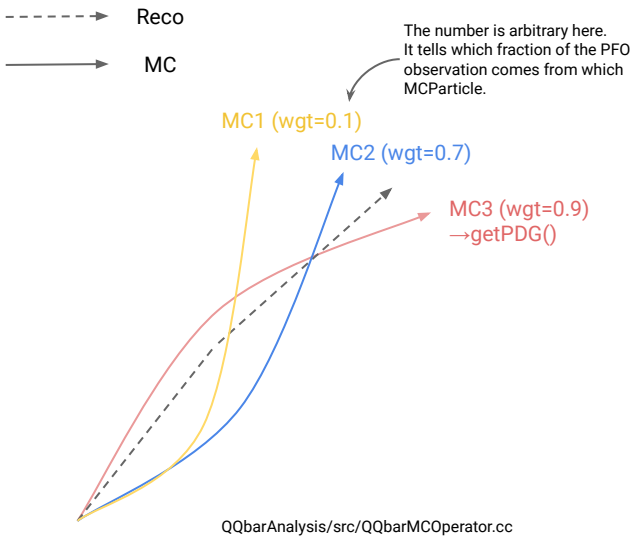
ILD Analysis/Software Meeting

Yuichi Okugawa
Oct 6th, 2021

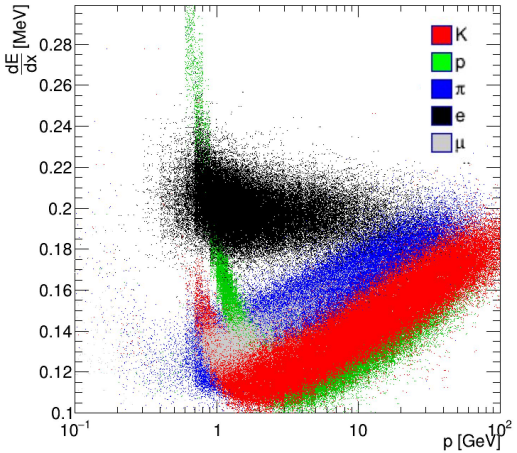
PID methods

- PDG Cheat



QQbarAnalysis/src/QQbarMCOperator.cc

- Non-cheat method

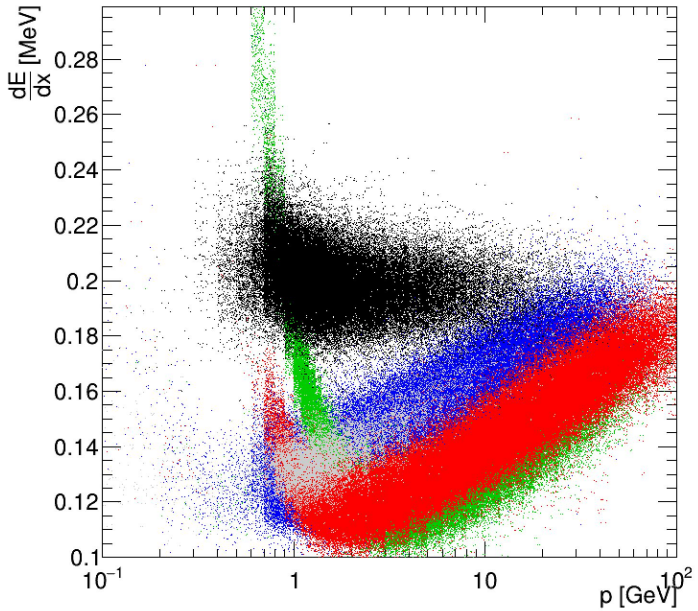


ssbar dEdx v. momentum distribution

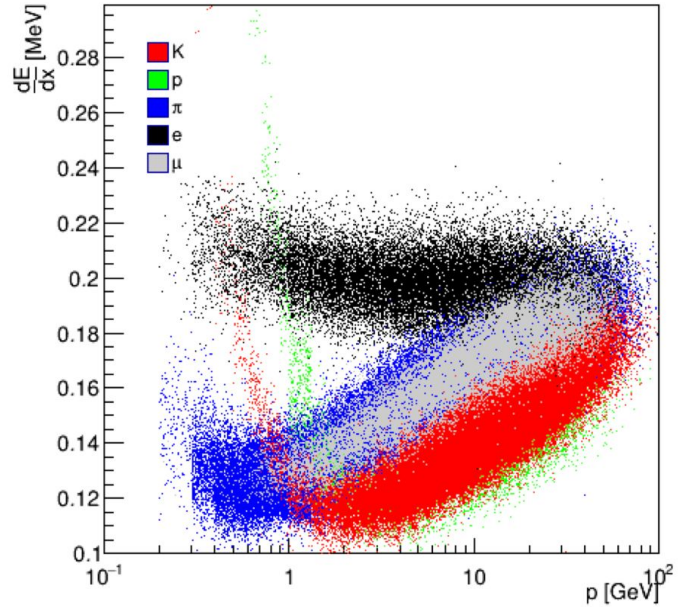
- Using dEdx information one can extract the region where it's dominated by one particle.
  → **dEdx PID**
- Two methods:
  - **dEdx projection**
  - **dEdx distance**

PID methods

ssbar



bbbar



Selection

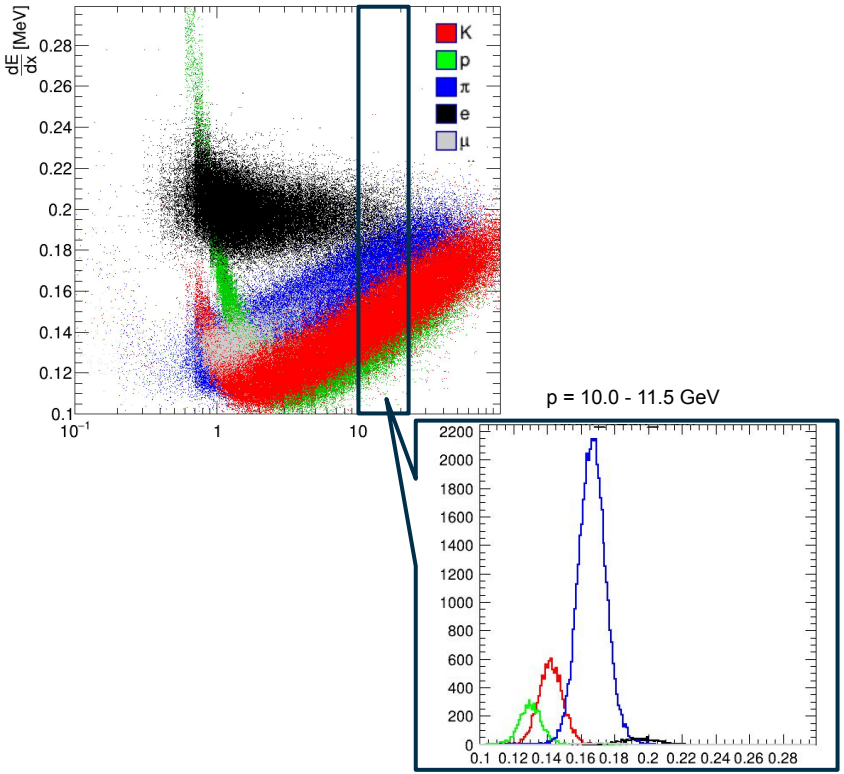
ISR/Gamma radiation rejection

- $\cos\theta_{\mathrm{QQsep}} > 0.95$ (back-to-back)
- $120 < p_q, p_{\bar{q}} < 127$
- Number of TPC hits:
  - $\cos\theta_{\mathrm{PFO}} < 0.75 \rightarrow \mathrm{TPC\ Hits} > 210$
  - $\cos\theta_{\mathrm{PFO}} > 0.9 \rightarrow \mathrm{TPC\ Hits} > 50$
  - In between: TPC Hits > 210 + (210 - 50) * (cosθ - 0.75) / (0.75 - 0.9)

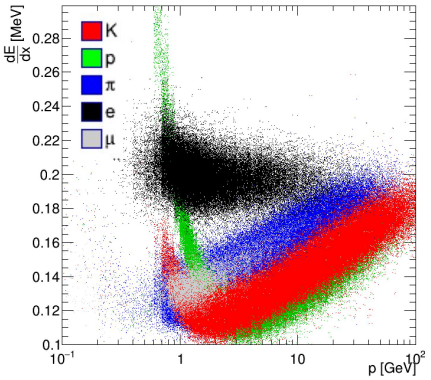
dEdx Projection

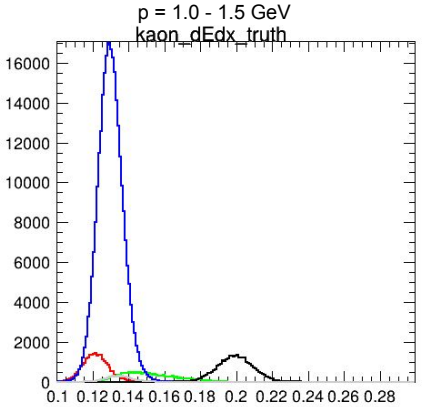
dEdx Projection

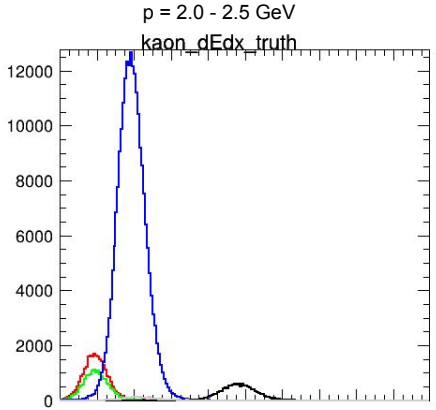
- dEdx Projection
  - Uses projection of dEdx between certain momentum region.
  - Finds the region dominate by a particle (e.g. kaon)
  - In real data, one can use this region to estimate where the particle is, with well defined uncertainty.

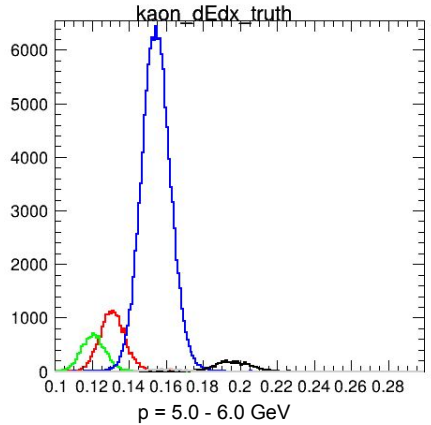


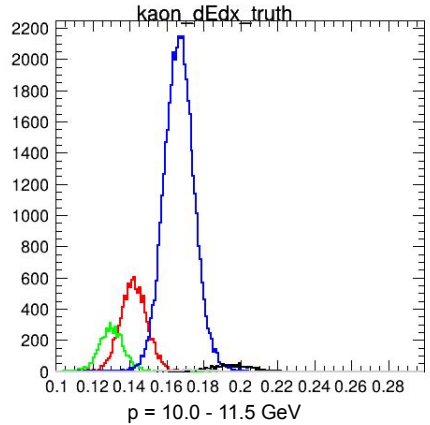
dEdx Projection

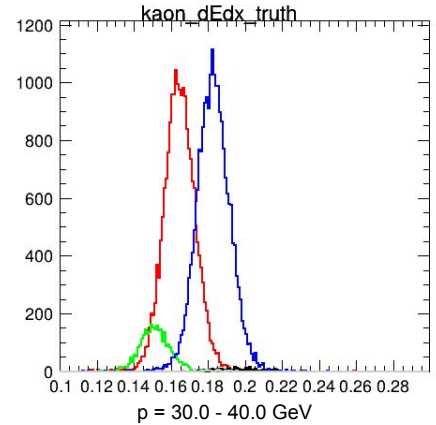






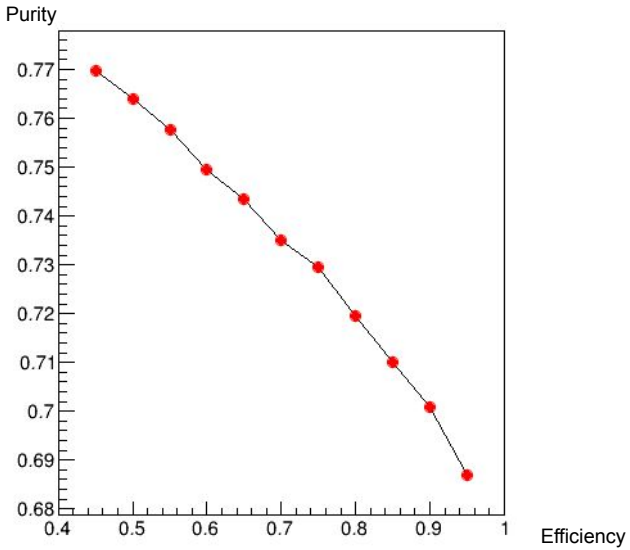


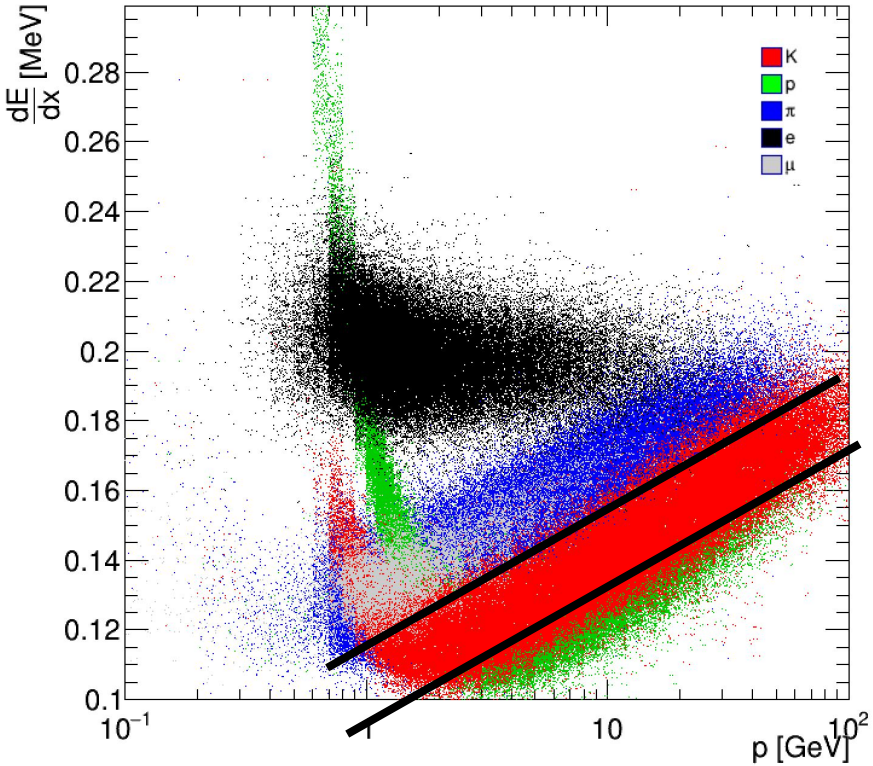




dEdx Projection

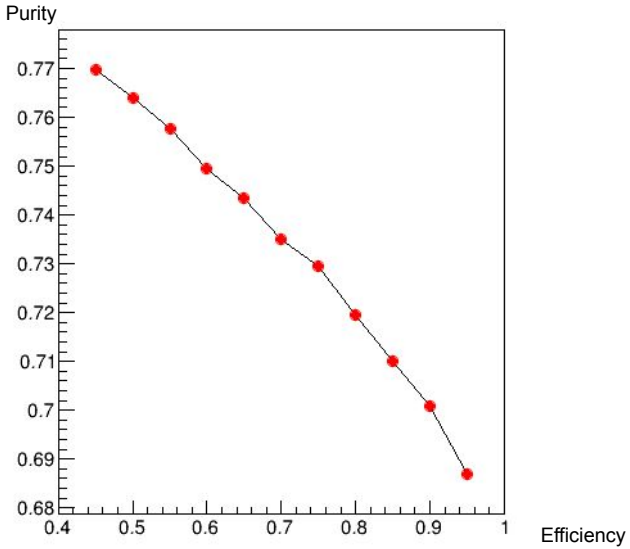
- Purity vs. Efficiency

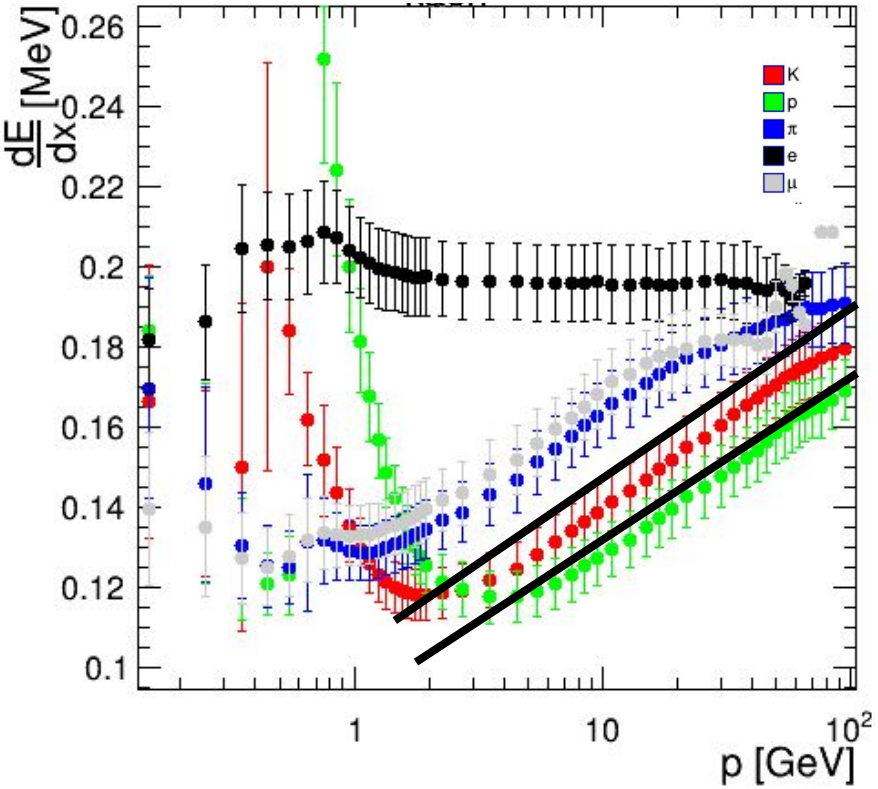




dEdx Projection

- Purity vs. Efficiency





dEdx Distance

dEdx Distance

- dEdx Distance
  - Uses Bethe-Bloch function corresponding to each particles.

$$-\frac{dE}{dx} = \frac{4\pi e^4 z^2 NZ}{(4\pi\varepsilon_0)^2 M_e v^2}\left[\ln\left(\frac{2M_e v^2}{I}\right) - \ln(1-\beta^2) - \beta^2\right]$$

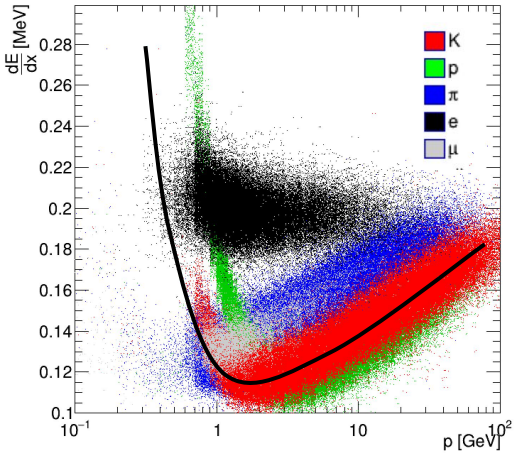
  - Calculate dEdx distances defined as following:

$$\text{dEdx distance} = \text{signed}\left[\frac{dEdx - dEdx_{\text{exp-kaon}}}{\Delta_{dEdx}}\right]^2$$

    where:

$$dEdx_{\text{exp-kaon}}\text{: Bethe-Bloch dEdx value}$$

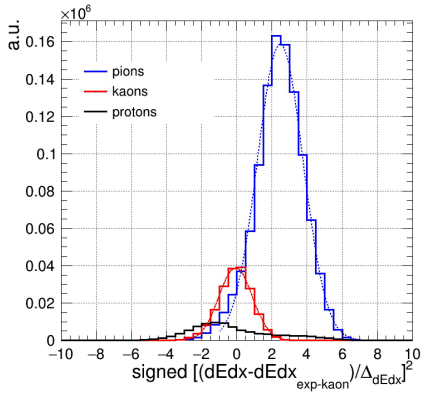
  - Minimize this value to get the PID.



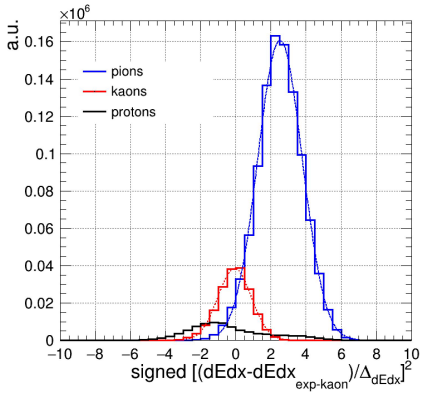
dEdx Distance

Pure **ssbar** generated

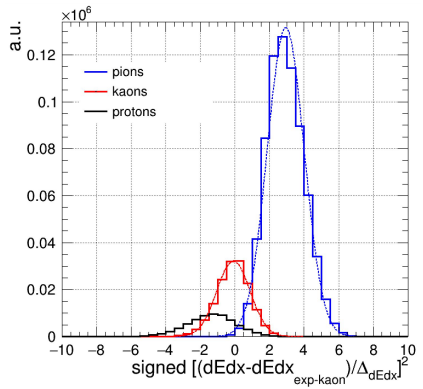
pcut = 0 GeV

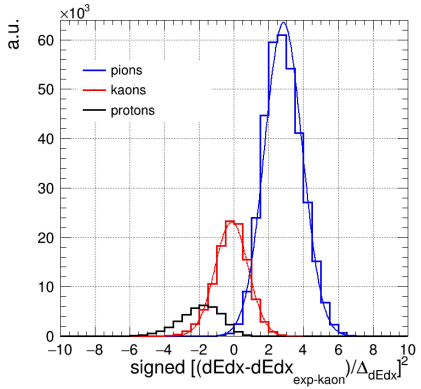


pcut = 1 GeV

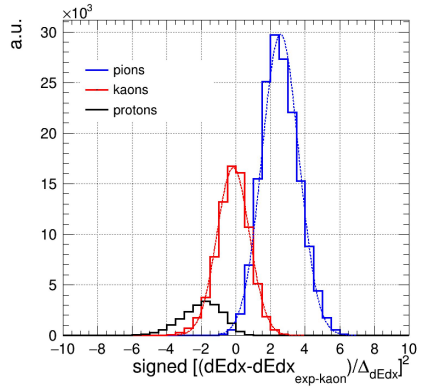


pcut = 2 GeV

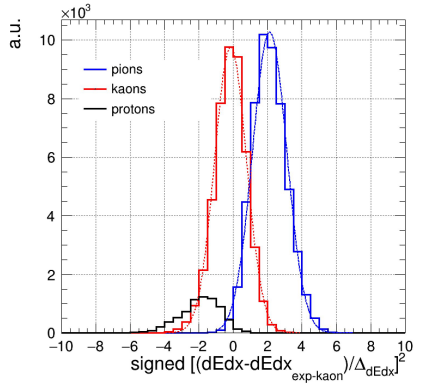




pcut = 5 GeV



pcut = 10 GeV

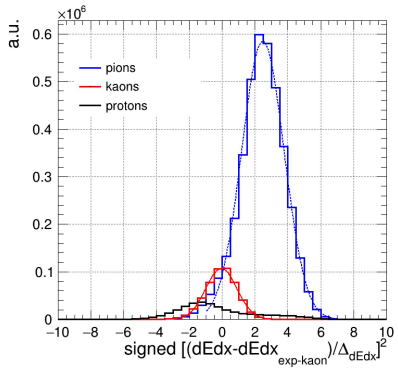


pcut = 20 GeV

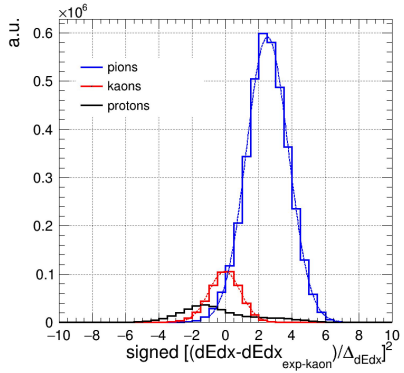
dEdx Distance

uds sample

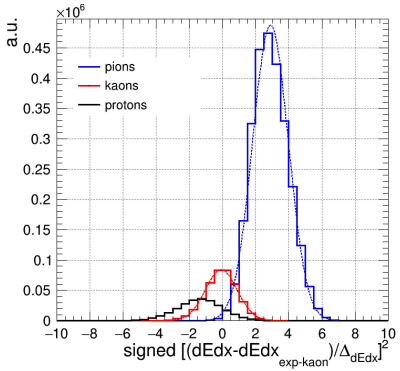
pcut = 0 GeV

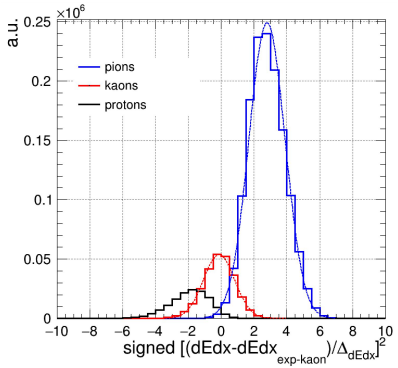


pcut = 1 GeV

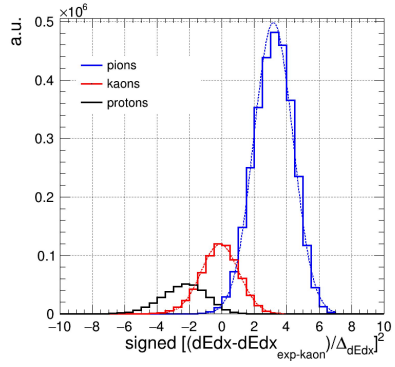


pcut = 2 GeV

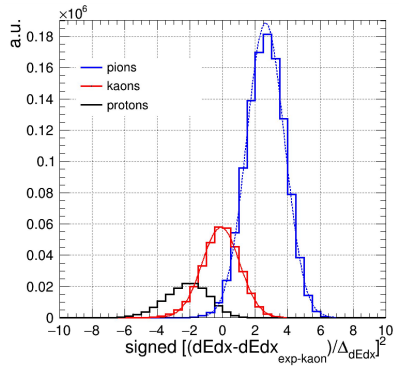




pcut = 5 GeV

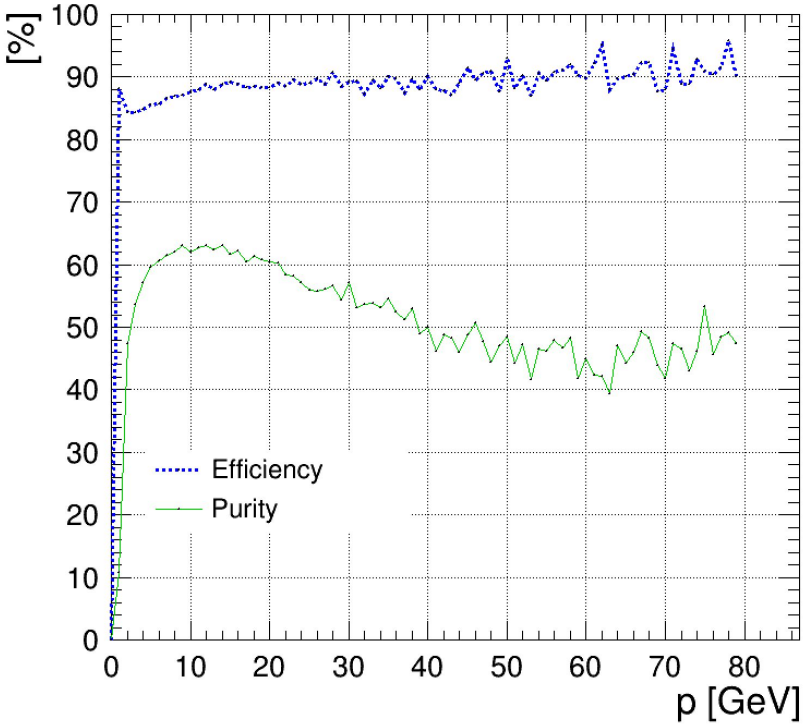


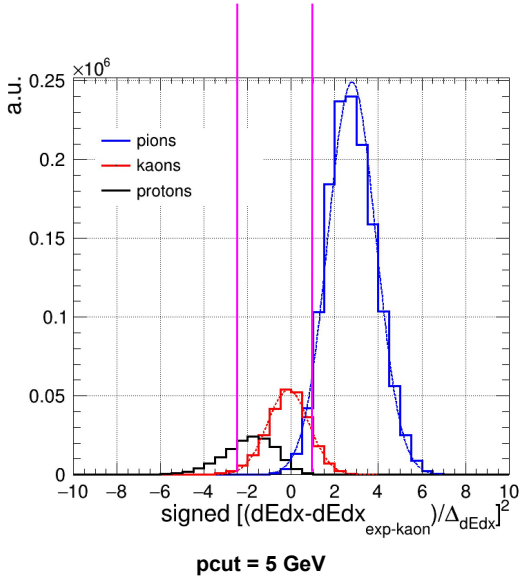
pcut = 10 GeV



pcut = 20 GeV

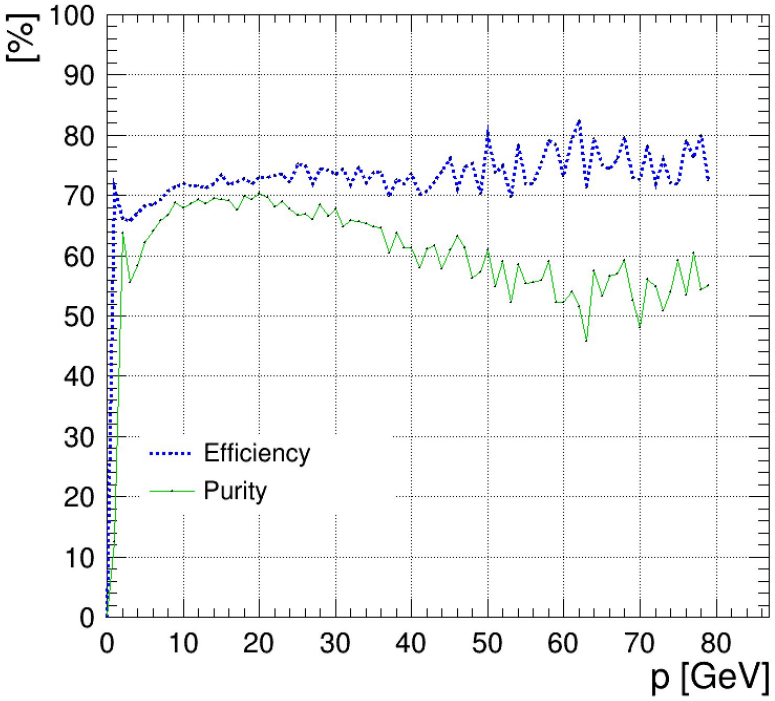
Efficiency and Purity

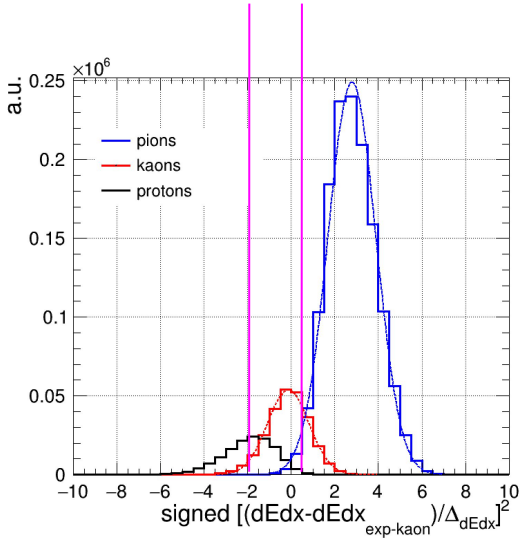




pcut = 5 GeV

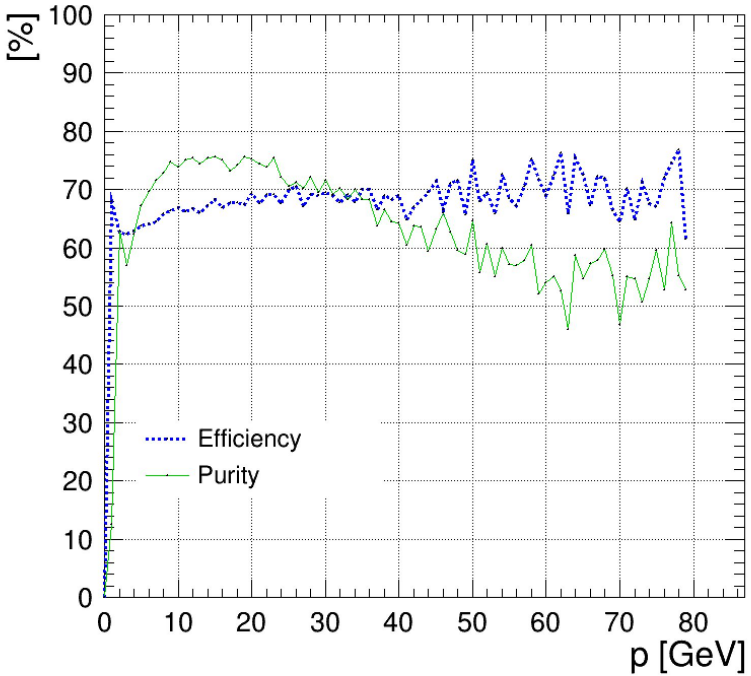
Efficiency and Purity

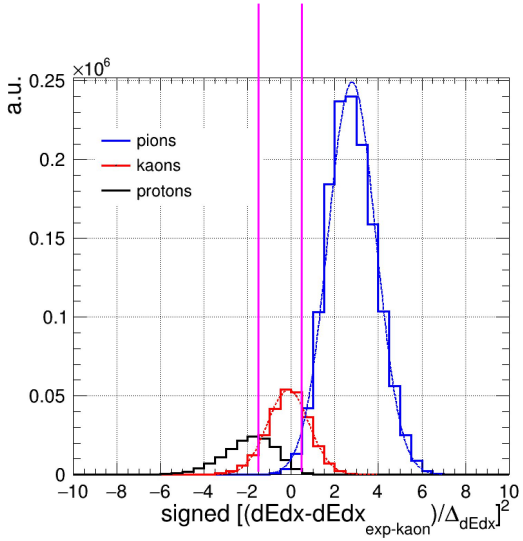




pcut = 5 GeV

Efficiency and Purity



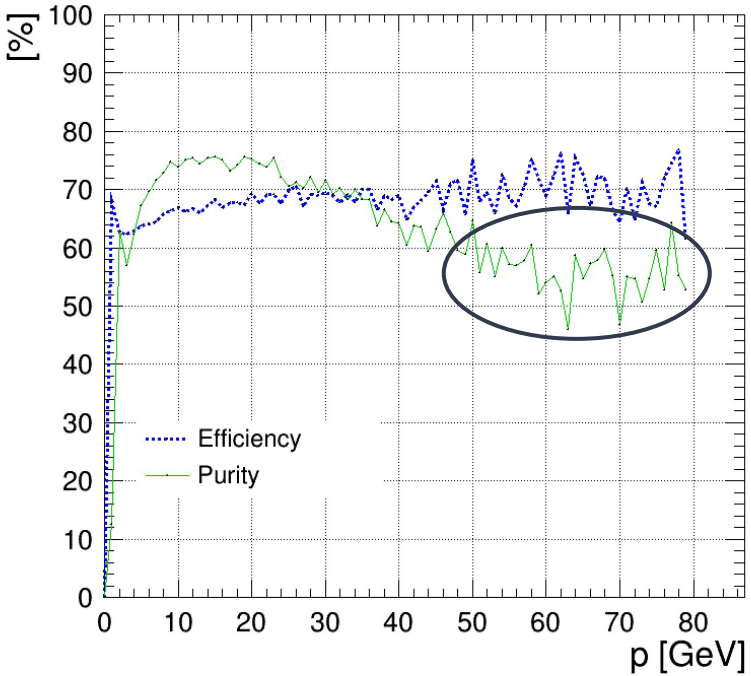


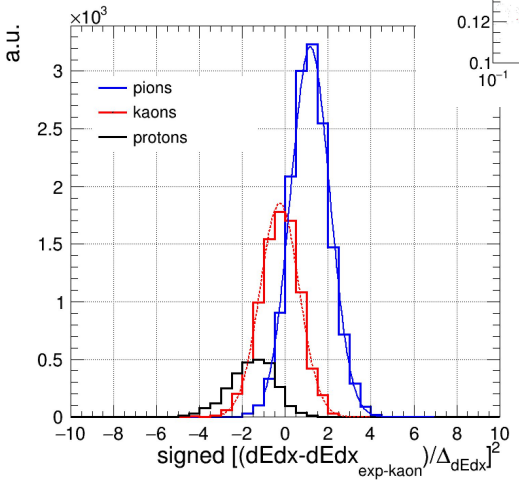
pcut = 5 GeV

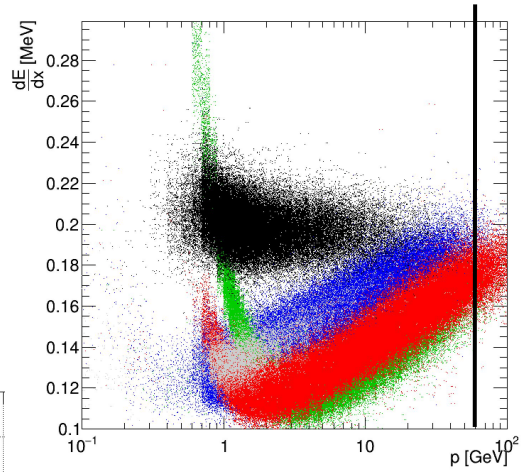
Backgrounds

- Pions
- Protons

Pions







pcut = 60 GeV

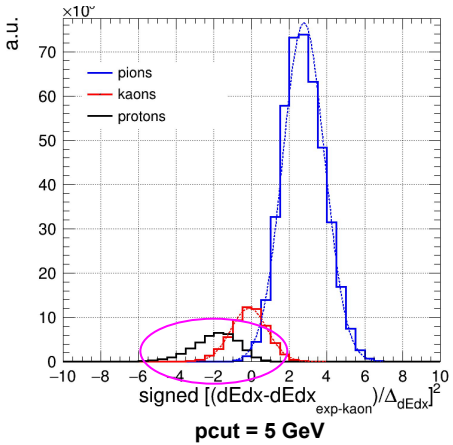
Protons

- Proton Contamination
  - Source?
    - Possible lambda decay

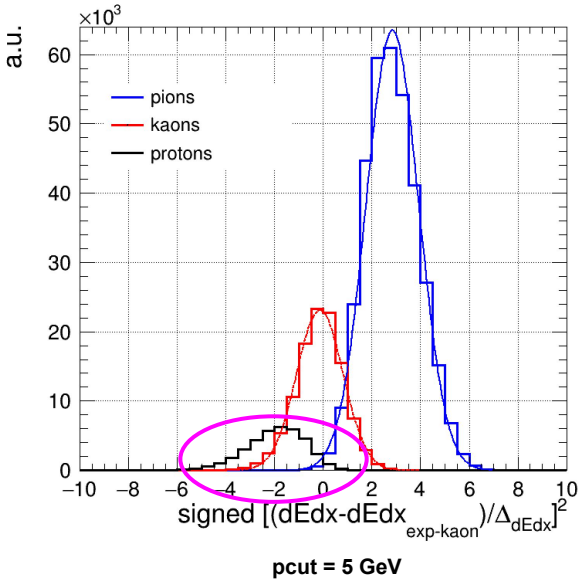
$$\Lambda^0(uds) \rightarrow p^+\pi^-$$

    - Also present in other process (ddbar)
    - If it's the case, can be eliminated by cutting on the offset.
      (lifetime ~ 2.6E-10s)

Pure **ddbar** generated



Pure **ssbar** generated



Summary

First look on PID using reconstructed information

- Attempted with 2 methods
  - **dEdx Projection**
  - **dEdx Distance**
    - Need further optimization for ssbar analysis
      - Proton rejection -> Lambda with an offset?
  - Choose leading PFOs in the analysis.

Backup Slides

Some discussions

- Requirements for # of hits (what are those?)

cos > 0.75 && cos < 0.9

```
bool nhits_bool = false;
if (fabs(costheta) < 0.75 && pfo_tpc_hits[ipfo] > 210) nhits_bool = true;
if (fabs(costheta) > 0.75 && pfo_tpc_hits[ipfo] > (210 + (210 - 50) * (fabs(costheta) - 0.75) / (0.75 - 0.9))) nhits_bool = true;
if (fabs(costheta) > 0.9 && pfo_tpc_hits[ipfo] > 50) nhits_bool = true;
```

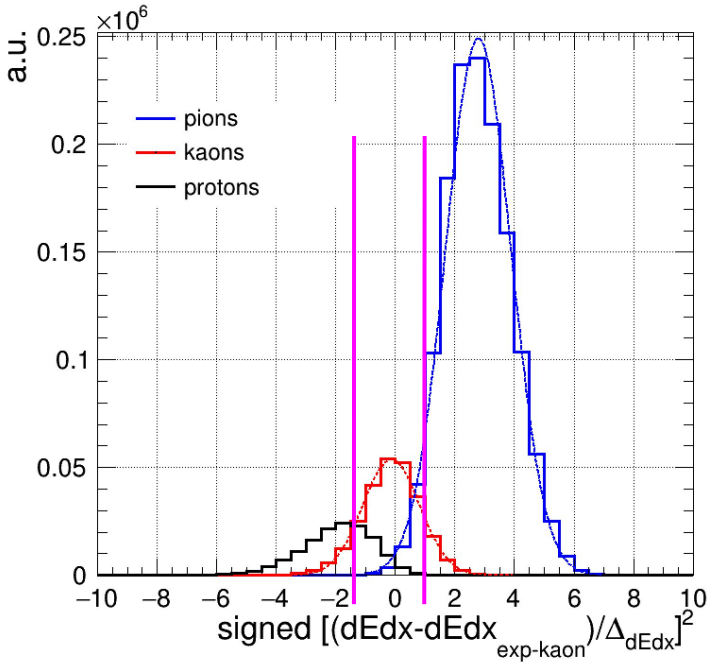
Q: There were some charged pfo's without tpc hit info. How are they processed in PFA?

- Cut can be applied to dEdx dist to separate kaons from pions.
  - Change with momentum?
- Calculation of eff. + purity will be done.

Definition of Purity

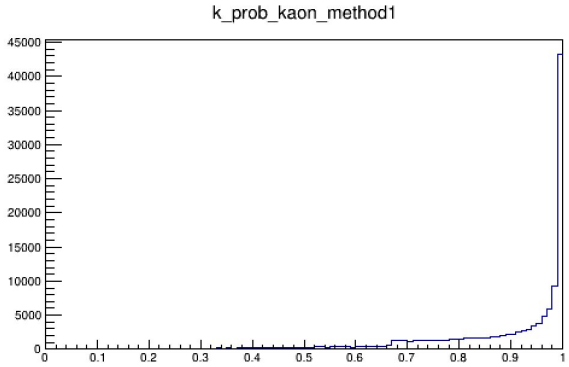
Eff & Purity Def.

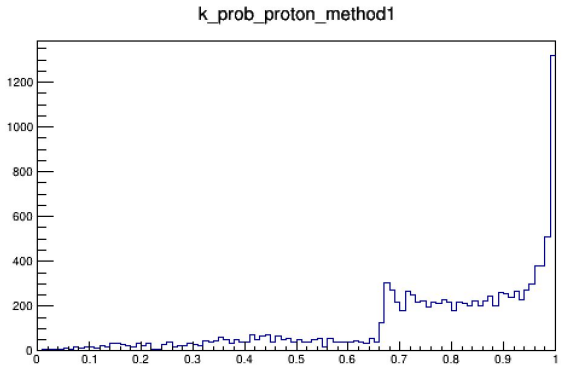
```
for(int i=0;i<80; i++) {
  int iproton=16;
  int ipion=22;
  // int iproton=18;
  // int ipion=21;
  // if(i>16) ipion=22-(i-16);
  float n_kaons=  kdEdxdist_kaon->Integral(i, i,iproton,ipion);
  float n_pions=  kdEdxdist_pion->Integral(i, i,iproton,ipion);
  float n_protons=  kdEdxdist_proton->Integral(i, i,iproton,ipion);
  float n_muons=  kdEdxdist_muon->Integral(i, i,iproton,ipion);
  float n_electrons=  kdEdxdist_electron->Integral(i, i,iproton,ipion);
  float nkaons=  kdEdxdist_kaon->Integral(i, i);
  if(nkaons==0) nkaons=10000000;
  x[i]=i;
  eff[i]=100.*(n_kaons)/nkaons;
  //  eff2[i-1]=100.*(n_kaons+n_pions+n_protons+n_muons+n_electrons)/nkaons;
  pur[i]=100.*n_kaons/(n_kaons+n_pions+n_protons+n_muons+n_electrons);
  n++;
}
```

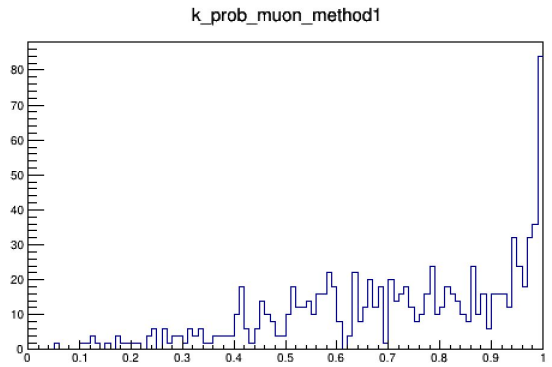


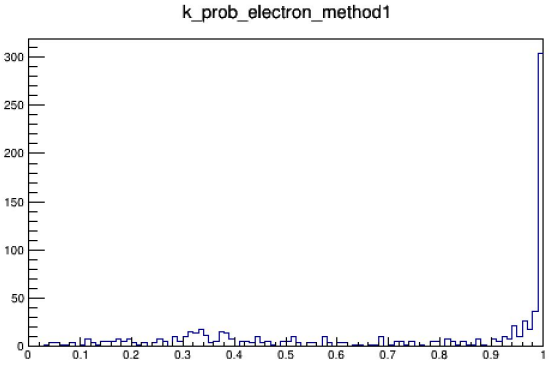
pcut = 5 GeV

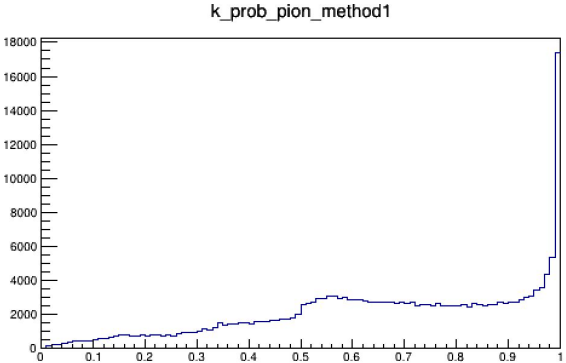
PID methods


k_prob_kaon_method1


k_prob_proton_method1


k_prob_muon_method1


k_prob_electron_method1


k_prob_pion_method1