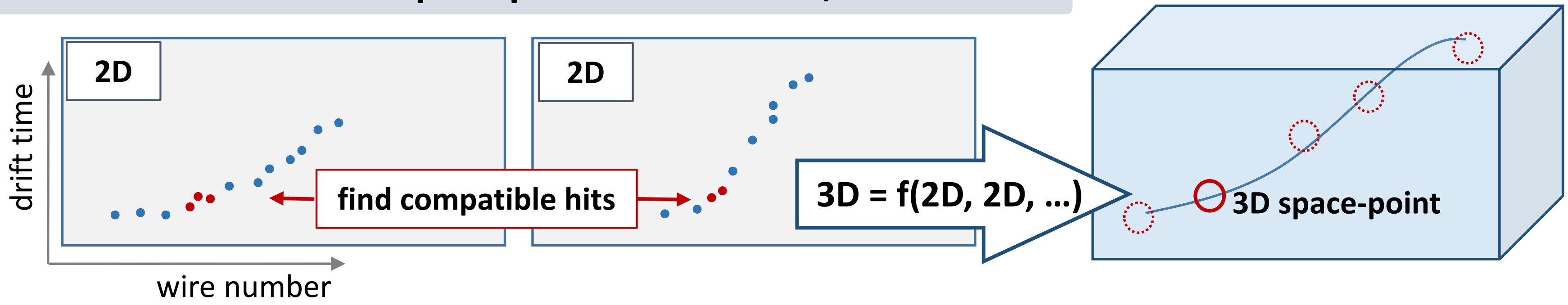
Projection Matching Algorithm for track 3D reconstruction
- LArSoft implementation

D. Stefan, R. Sulej
NCNR Warsaw

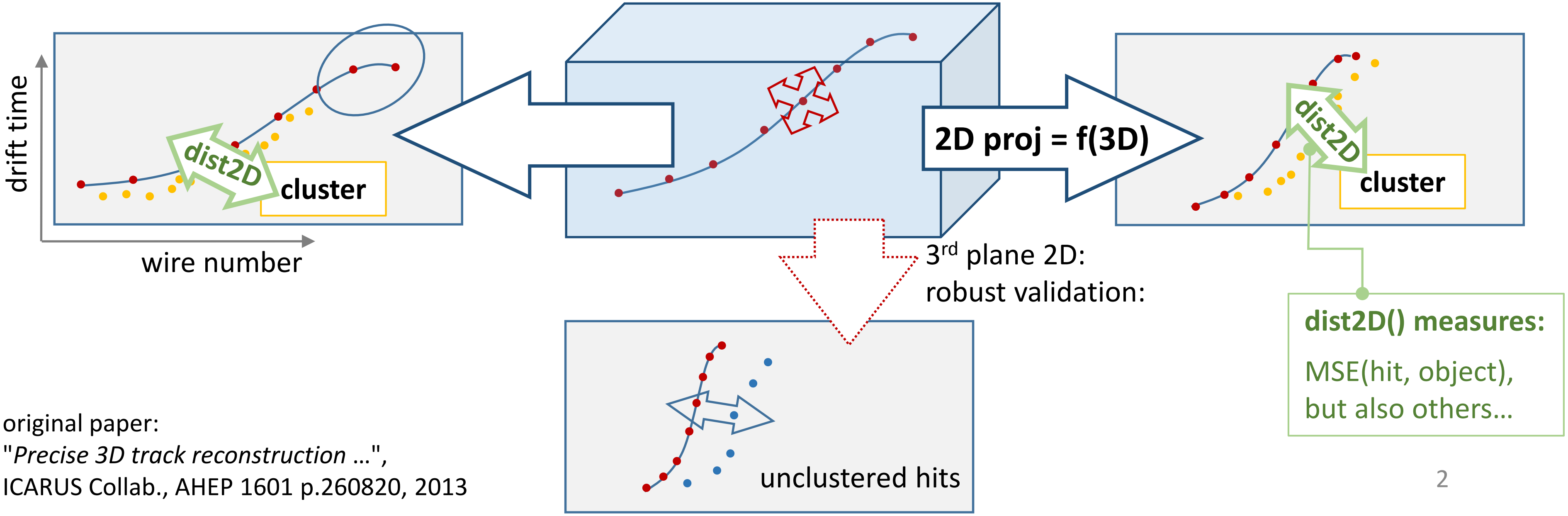
Another approach to build 3D things in TPC

usual: ***n*** **2D hits -> 3D space point -> 3D tracks, etc.**



up side down: ***Projection Matching Algorithm***

work in 3D (on *single tracks* or *full track structures*) to match 2D projections to hits



original paper:
"*Precise 3D track reconstruction …*",
ICARUS Collab., AHEP 1601 p.260820, 2013

Algorithm features

- **no explicit hit-to-hit associations between 2D planes**
- **simultaneous use of information from all planes**
- **3D objects driven by 2D parts, not only isolated points**
  - individual 2D planes can have some missing information (due to dificult track orientation, hit/cluster inefficiency, hardware, …)
- 3D optimization can take into account also 3D points: vertices, feature points, …, available from other algorithms

Typical use

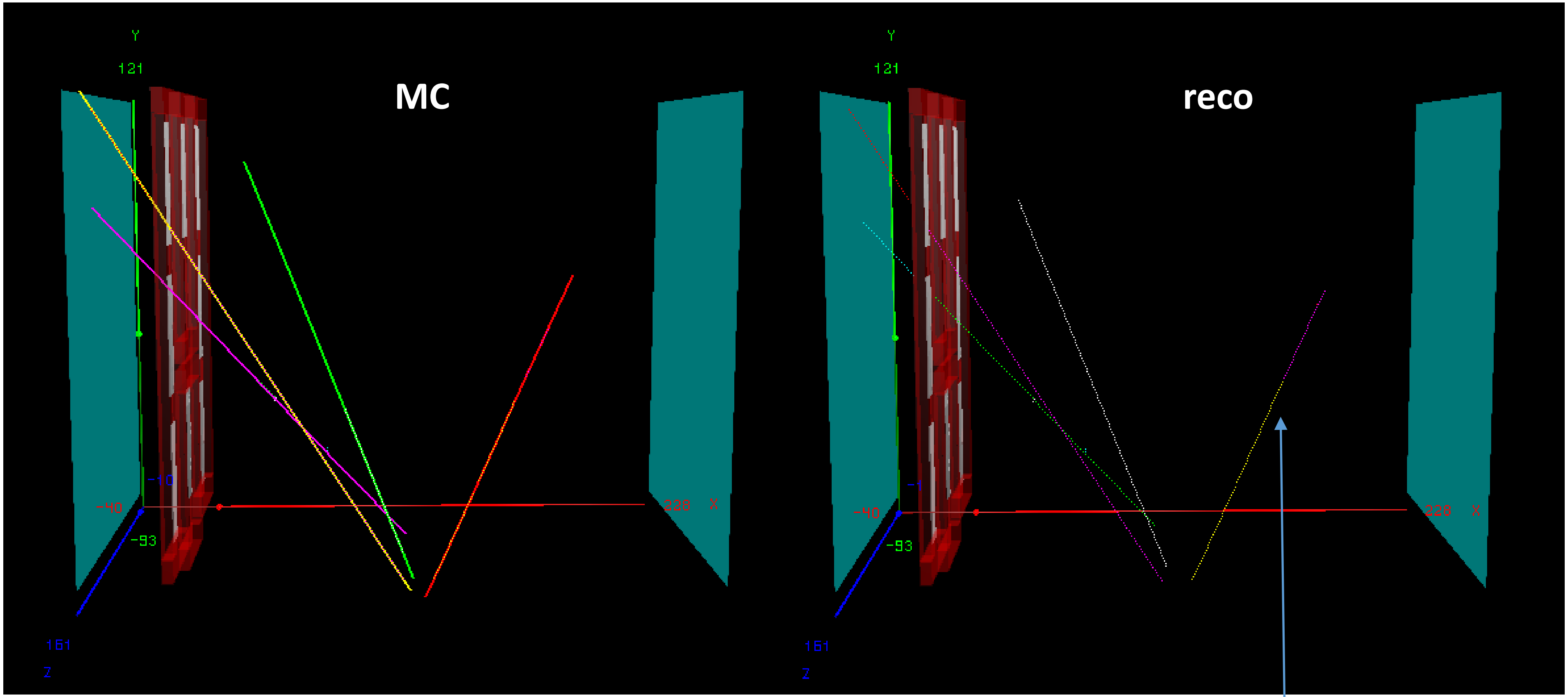
- associate two clusters -> optimize 3D -> check obj. function valuevalidate in the 3rd 2D view (initial clusters pair do not need to be precisely corresponding)
- grow / complete the track by adding compatible clusters
- **stitch tracks, find and conect full 3D structures -> reoptimize 3D**
- do analysis: initial directions, track dE/dx, PID, energy, …

PMA in LArSoft

- **PMA engine in: larreco/RecoAlg/PMAlg/***
- **Algorithm interface class:** larreco/RecoAlg/**ProjectionMatchingAlg**.h&cxx
    - few basic functions to create, extend and validate tracks
    - few basic parameters to controll algorithm
    - more to be added (to expose settings used in pma::Track3D)
        - weights used to combine information from different planes
        - weights assigned to 3D points from other algorithms
        - functionality for freezing track nodes (shower reco needs this)
        - …
- **Module to create tracks from clusters:**
  larreco/TrackFinder/PMAlgTrackMaker_module.cc
    - very basic logic to loop over clusters, first quick example and test of the algorithm implementation
    - loop starts from the largest cluster (any plane), finds best matching cluster by drift time span (any other plane), validates track (if 3rd plane available)
    - **many other logics possible** – we'll try, and we encourage others as well

First attempts (standard linecuster used as input)

Long, high energy muons crossing the detector, 5mu/event, large sample:

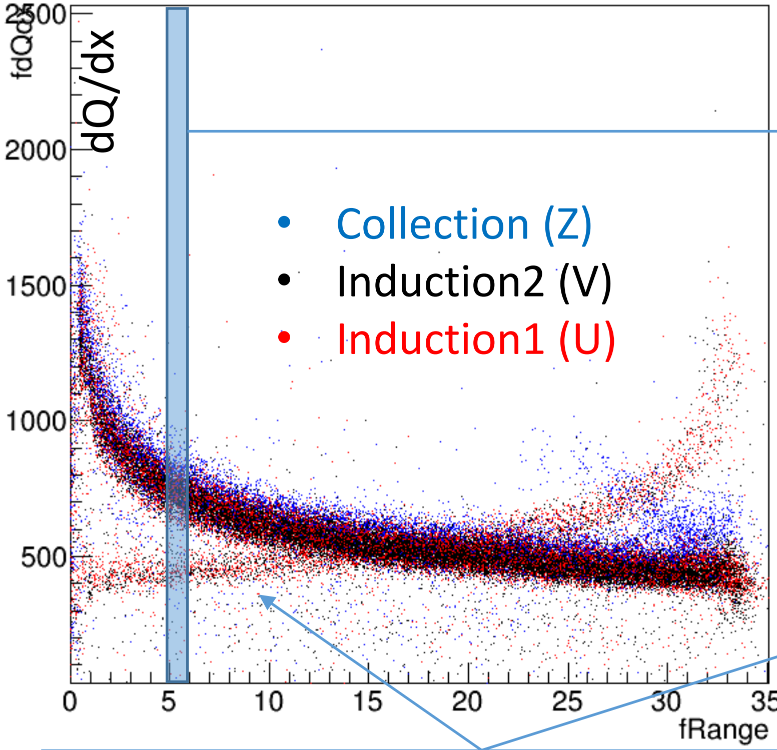


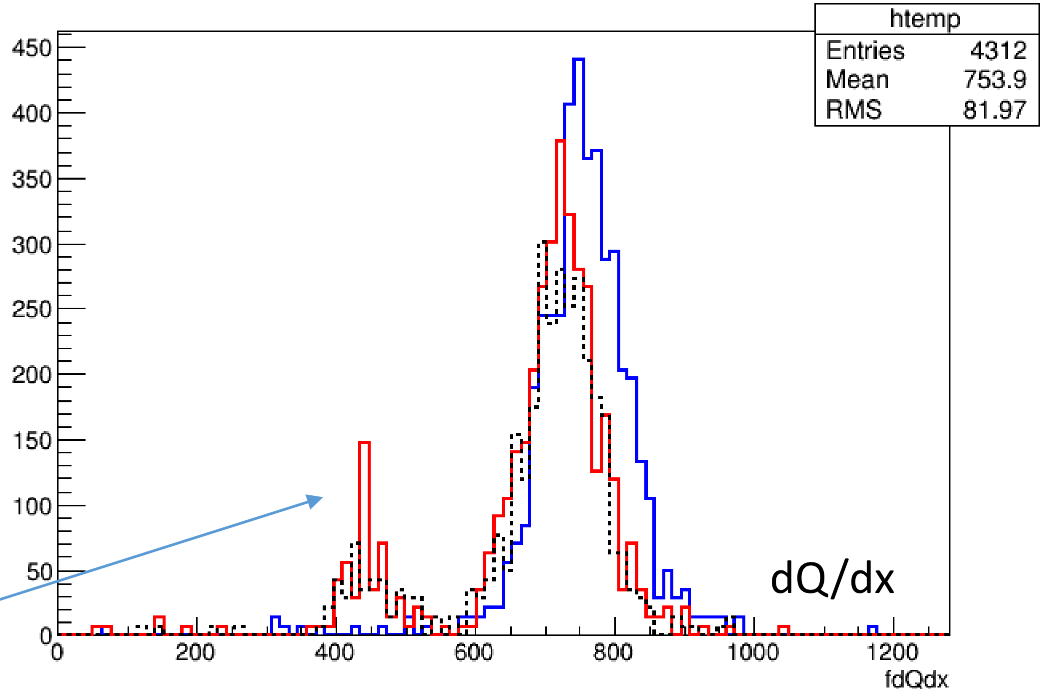
no stitching yet

Systematic efficiency measure needed, of course.

First attempts (standard linecuster used as input)

Single, low energy protons (700MeV/c, ~30cm), dQ/dx reconstruction:





Automatic flip of the track direction failed for a few protons with short projection in Collection, corrected today with Induction views.

dQ/dx in Induction planes lower than in Collection (Tingjun says it is known behaviour).

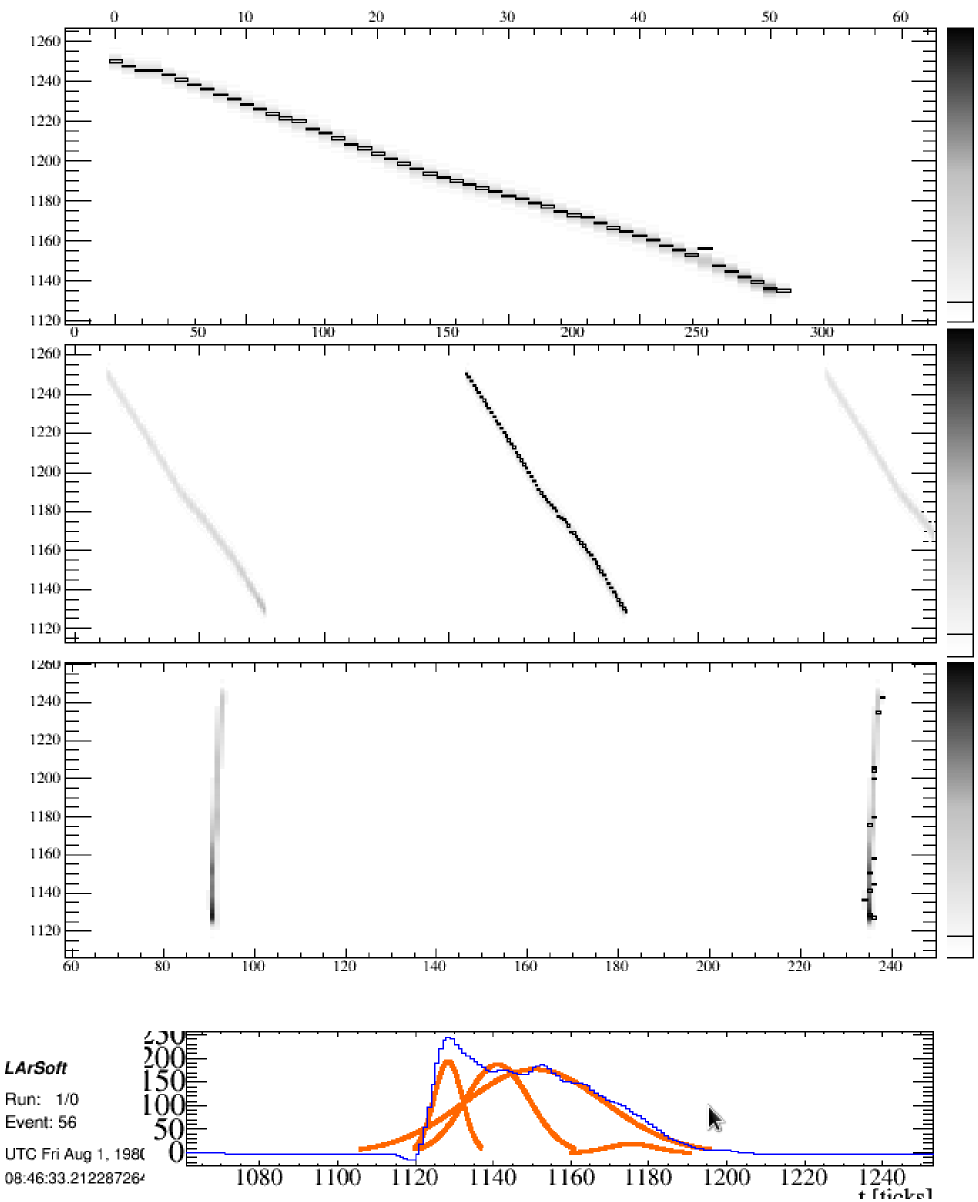
~5mm wire pitch: less data points / length than in T600 (3mm)

⇒ narrower dQ/dx distribution

⇒ lower spatial resolution (endpoint location, decay prod. separation, …)

⇒ **PID may perform diferently than we were used to** – interesting to check

First attempts (standard linecuster used as input)

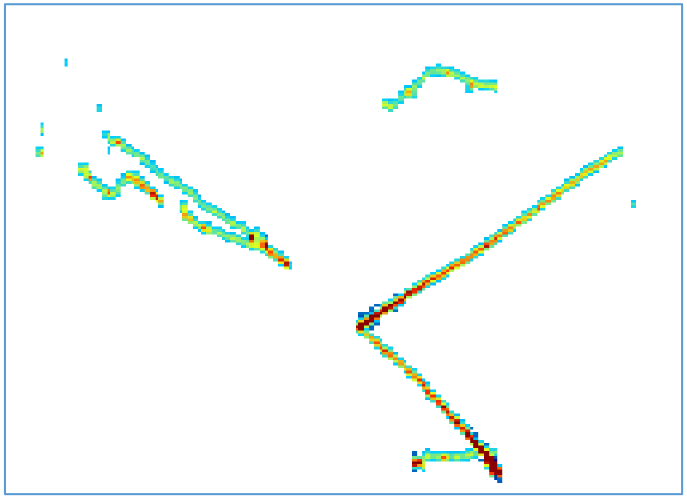


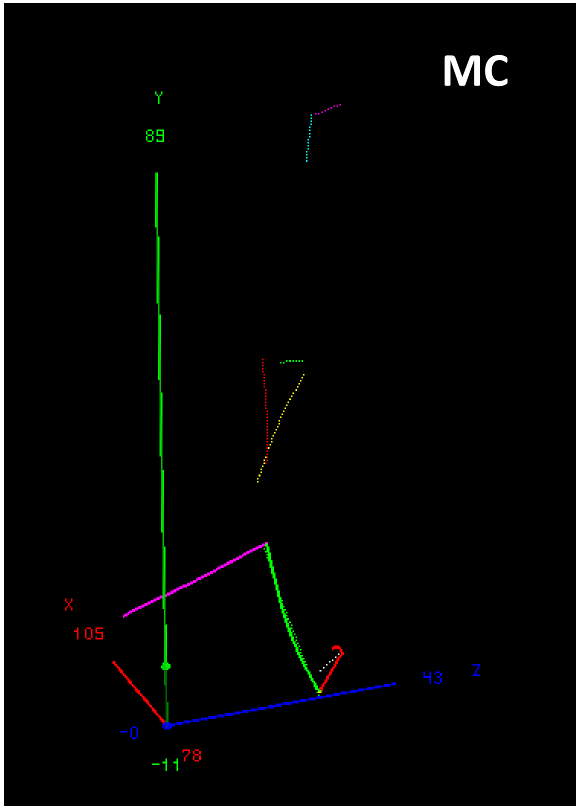
Hit reconstruction to be tuned:

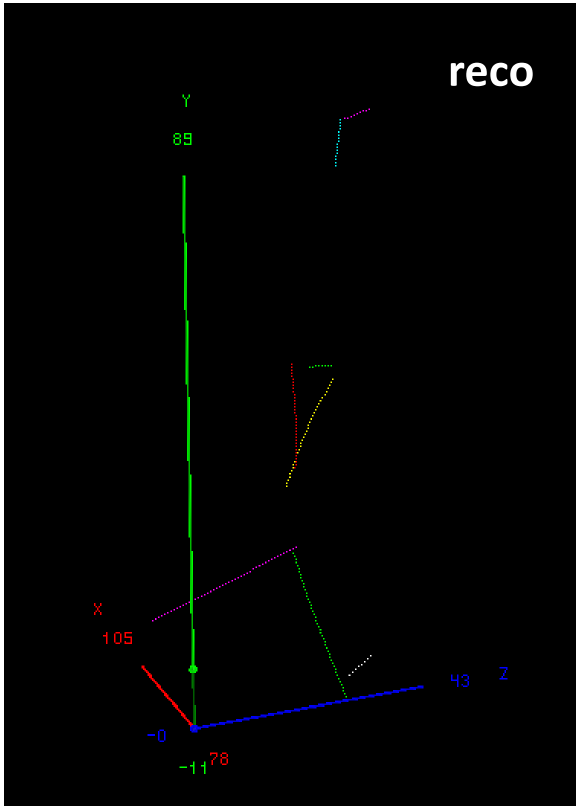
- Hit peak time delays due to deconvolution to be optimized
- Try to improve params of hit reconstruction for tracks ~parallel to the drift direction.

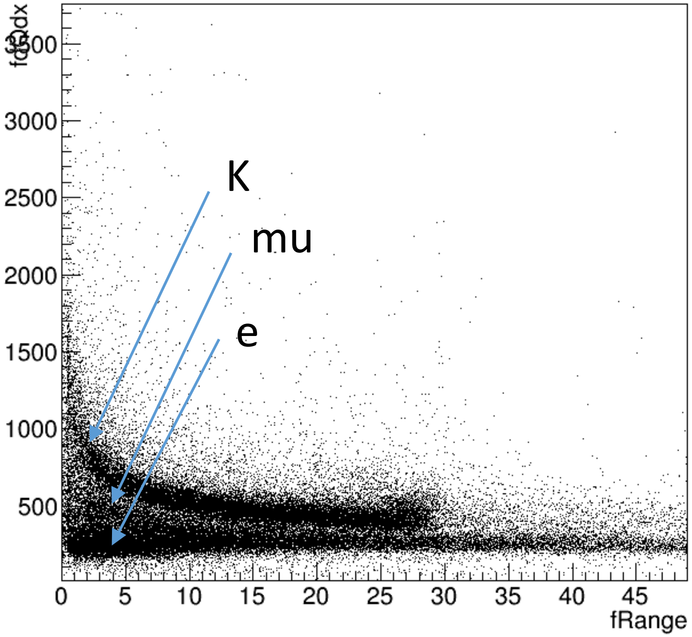
First attempts (standard linecuster used as input)

Decaying Kaon, just example of a few-track event:









Summary

- **many thanks to Tingjun!!!**
- **it is a pleasure to work in LArSoft environment**

- **basic algorithm is up and running**
- **To do:**
  - **some of well known special cases and obvious functionality** (handle tracks paralel to wire planes, merging tracks)
  - verify loop over clusters: seems that still not all compatible clusters are found (maybe due to hit peak time shifts)
- **efficiency measures to be applied**
- our first goal is shower initial direction -> functionality for this purpose is the next thing to add (a function in the algorithm class tuned to build short segments, optionally with one endpoint fixed)

- validation of 3D in the 3rd plane gives potential to apply the algorithm without disambiguation in „wrapped" planes

- any comments and suggestions are very welcome, we would be also glad to help those interested in using the algorithm

Thank you