GArTPC track to ECAL matching part 2

1) *Our story to date*
2) *Fixes and upgrades*
3) *New algorithm in some detail*
4) *Performance – Nope!*

9 Aug 2021

MPD Meeting

Leo Bellantoni

Our story to date

- The initial algorithm for matching tracks to ECAL clusters dates from June of 2019 was based on straight line extrapolation from the end of the track
- Actually not *so* bad for e.g. ν_e or μ from CC, but not *so* good for lower-momentum tracks
- Bigger problem: made no allowance for track drift in the 10μs gate time
- In the absence of a functioning BackTracker, efficiency and purity of the matching could not be determined either
- This time, we'll look at full GENIE default events, using 12 side ECAL, SPY v3 & MuID rather than just gas-interaction CC coh. No overlays, but event spill time is allowed for so we have stitched tracks.
- But first, a few slides summarizing upgrades & fixes

Fixes and upgrades

- `TrackPropagator::PropagateToCylinder` would extrapolate a helix (i.e. a track) to a cylinder along the x axis (i.e. a depth in the ECAL barrel).
    - There are either 0, 2 or ∞ solutions. Previously, output of this code was either an error code or 1 solution, chosen a bit arbitrarily.
    - Now has 2 outputs. The 1st is the one that it gave before. Calls in other code (event display modules as I recall) fixed for new calling sequence.
- New functions, `TrackPropagator::DirectionX(x)` and `TrackPropagator::DirectionPhi(phi)` will construct unit direction vectors, with direction determined by increasing ϕ along the tracks' helix.
- `TrackPropagator::PropagateToCylinder` also had a buglet… the track parameter for curvature can have a negative sign and in some places an `abs(…)` was needed
- The BackTracker algorithm would segment-fault and kill your job dead dead dead if two tracks had the exact same sum of `edep` ionization energies.

Fixes and upgrades

- **Two new BackTracker methods added:**

```
bool ClusterCreatedMCParticle(simb::MCParticle* const p,
                              rec::Cluster* const c);
bool MCParticleCreatedCluster(simb::MCParticle* const p,
                              rec::Cluster* const c);
```

These match the cluster back to the underlying energy deposits & then deposits to the GEANT particles; then search up or down the MC particle tree in search of `p`. If `p` actually contributed to the cluster directly, both routines will return `true`.

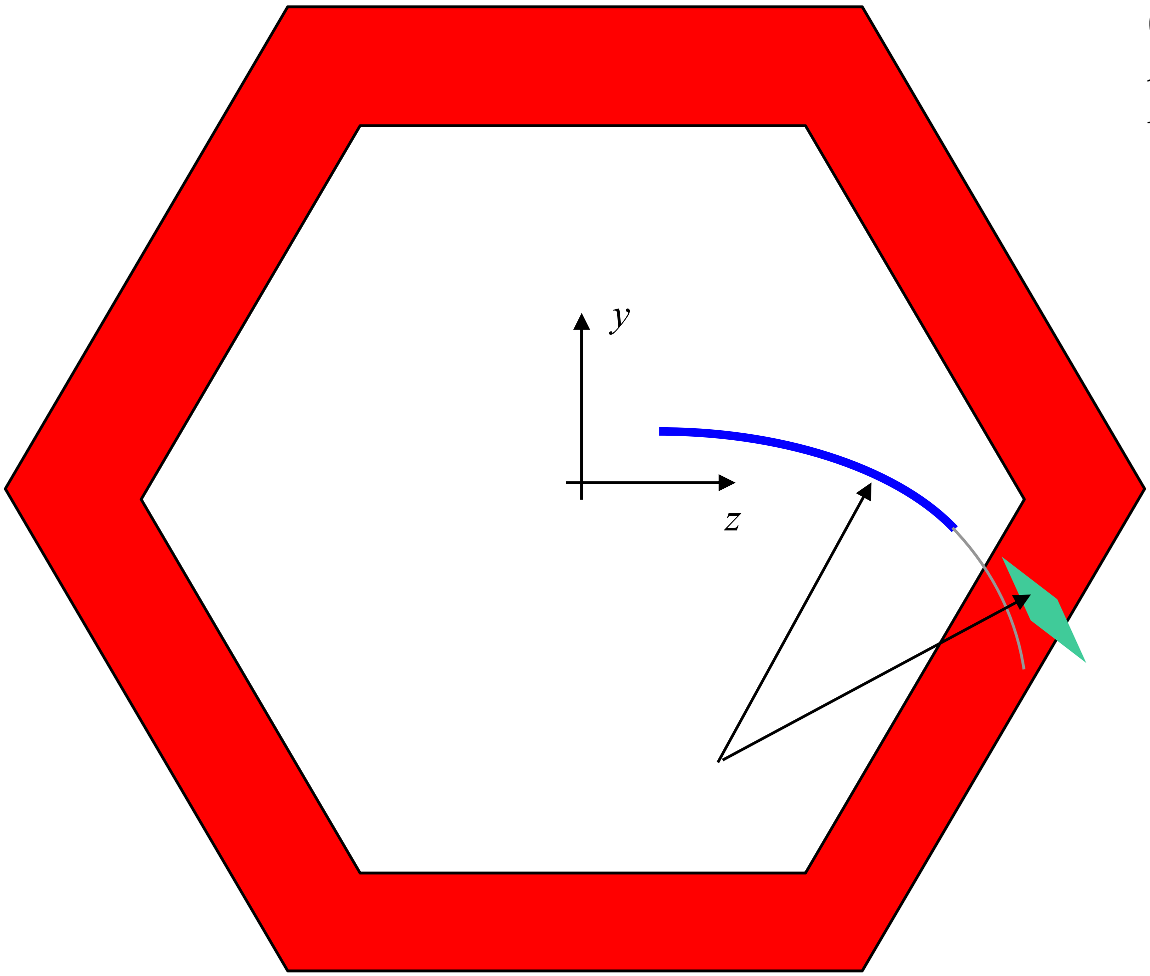
Fixes and upgrades

- **Things added to the (flat, and no doubt soon obsolete) `anatree`:**
  - Center of the MPD. Many algorithms make computations relative to this point but as we try different geometries, that point changes. The same number appears in each event, but ROOT's file-compression means that this only takes up ~3kbyte of the file (seems to be the smallest unit of space allocation)
  - `MCParticle`, as determined by the BackTracker, for each reconstructed track and cluster. For the cluster, the "TPCeve" is found, i.e. the ancestry tree is searched until it reaches an `MCParticle` that originates in the TPC gas or the primary interaction.
  - N.B. I had hesitated to add the BackTracker info earlier, as some physics study of some sort might be a good idea. Right now, I just take the "best match" and its corresponding ionization fraction as returned from the BackTracker… no idea what is in the other matches, or how good a match one should require.

- **Thanks also to Eldwan for ECAL strip-splitting bug fix**

- **All pushed to `develop` branch, over a period of months**

New algorithm – cut I



Track in blue

ECAL cluster in green

(Power Point requires a hexagonal ECAL)

For each track end outside cylinder

$\rho = 230$ cm $|x| = 215$ cm,

compare r_{TRACK} to distance of cluster from track center in transverse plane. Require

$$| r_{\mathrm{CLUS}} - r_{\mathrm{TRACK}} | < 8 \text{ cm}$$

(8 cm is a `fcl` parameter, as are the other cuts)

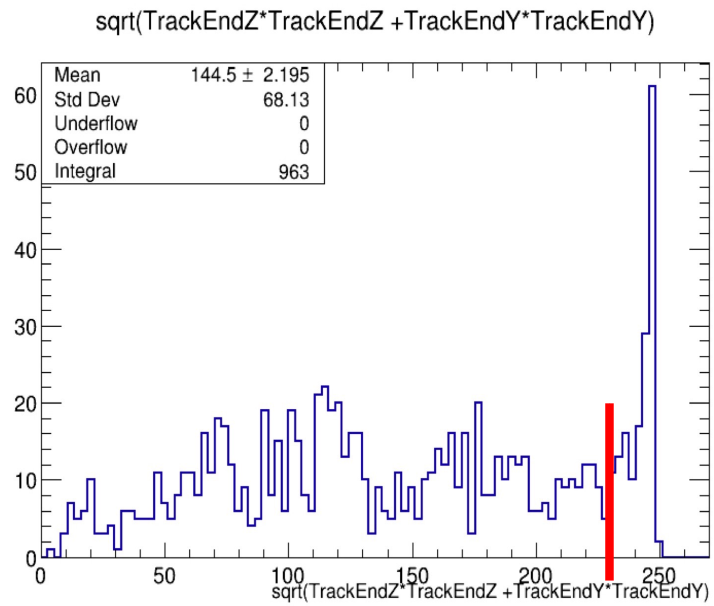
DEEP UNDERGROUND NEUTRINO EXPERIMENT

New algorithm – cut I

The cuts at ρ = 230 cm, $|x|$ = 215 cm cuts are from the following consideration:

Based on 10 full default GENIE spills (earlier geometry), the distribution of $\rho = \sqrt{z^2 + y^2}$ shows two populations – One is clearly tracks that leave the TPC radially and those are the ones we are looking for

If a particle hard-scatters in the center of the TPC and reco breaks it into two tracks, it's the one that ends near the ECAL that will get calorimeter clusters matched to it.

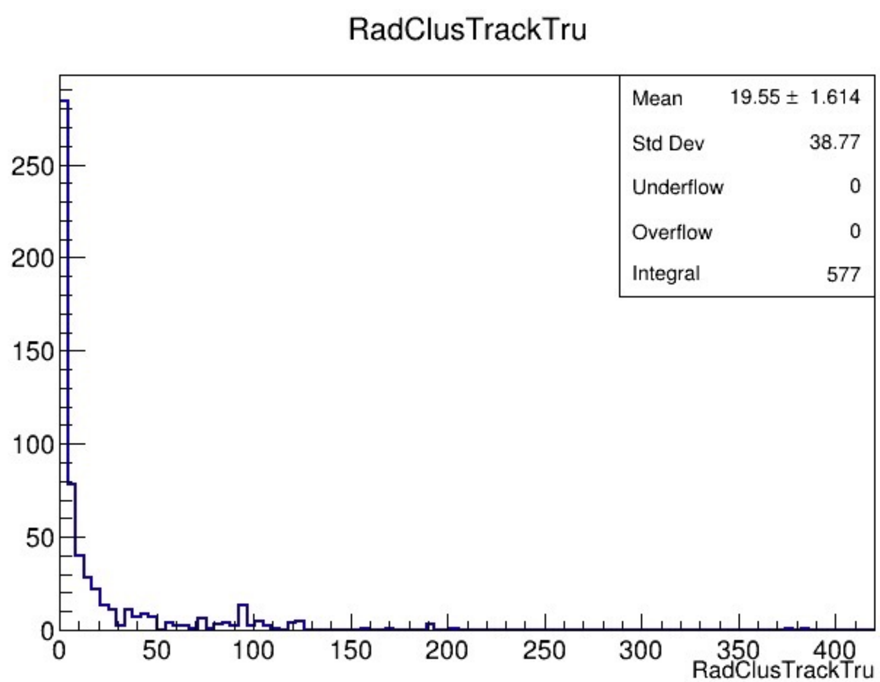


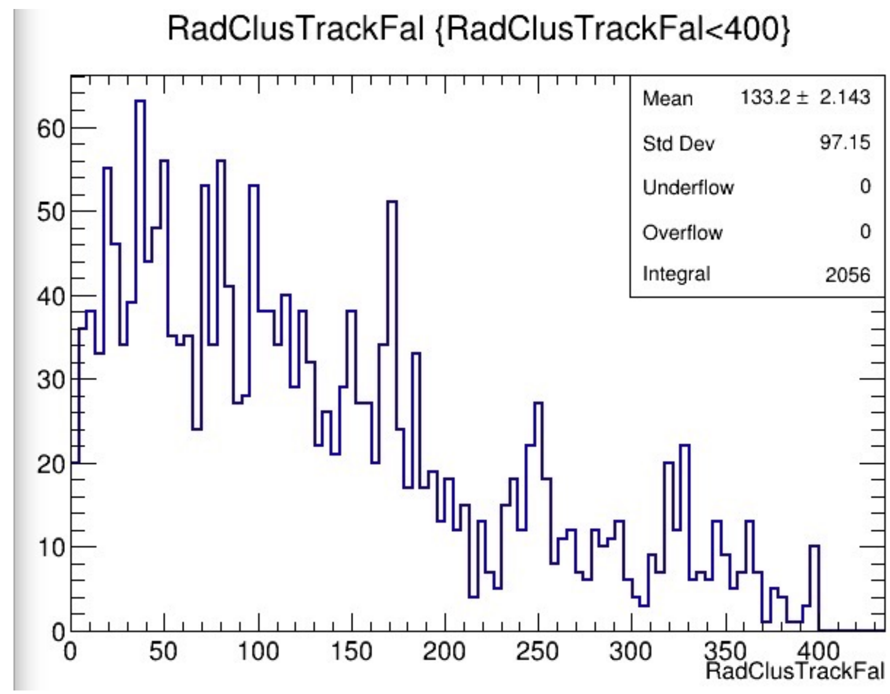
This end-region is 20 cm wide. In the x direction, in addition to this 20 cm, add 30 cm for drift and subtract 5 cm because resolution in x is different than in $r\phi$. Not that we've really got the resolution in x perfect in our simulation yet ☺️

New algorithm – cut I

Using the BackTracker, in a set of GENIE 3.0.6 events, identify cases where the both track & cluster are certainly from the same particle (left) vs cases where they aren't (right)

Tracks must have pVal>0, only 1 MCParticle contributing to it,
And the fraction of the ionization from that MCParticle must be > 1/2

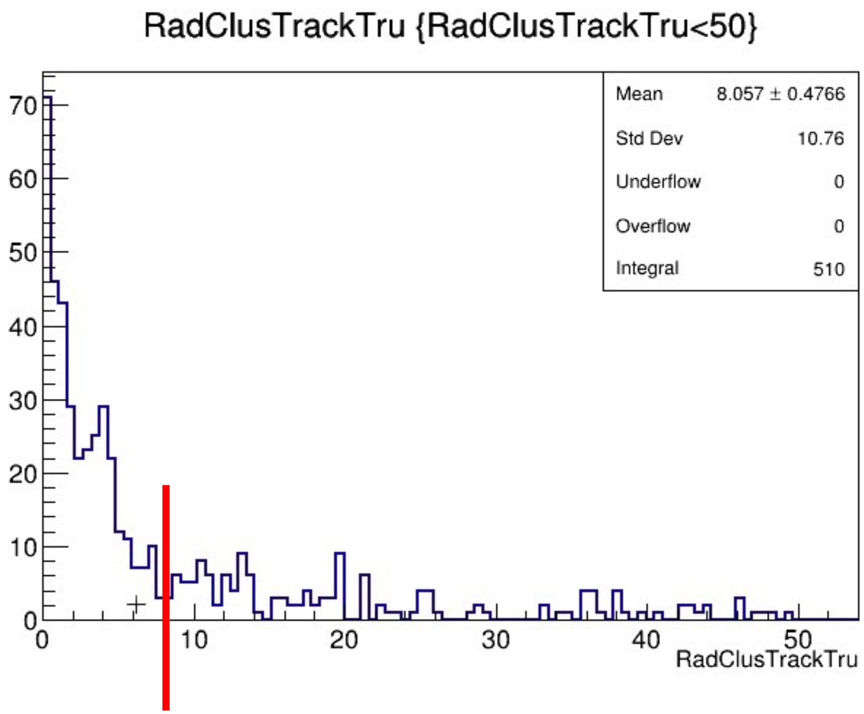


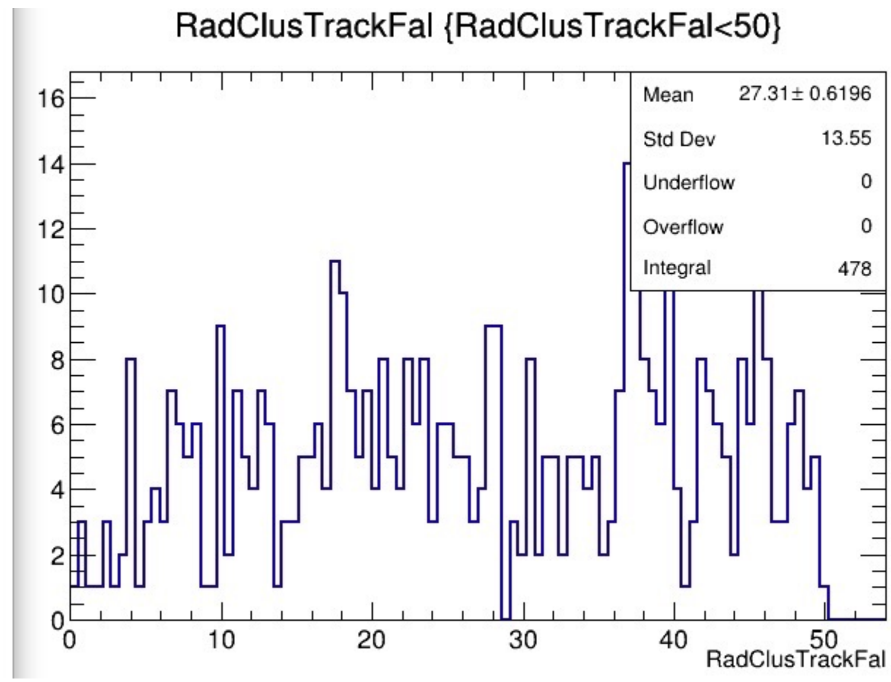


New algorithm – cut I

Using the BackTracker, in a set of GENIE 3.0.6 events, identify cases where the both track & cluster are certainly from the same particle (left) vs cases where they aren’t (right)

Tracks must have pVal>0, only 1 MCParticle contributing to it,
And the fraction of the ionization from that MCParticle must be > 1/2





New algorithm – cut II

The second cut is where the real problem raises its ugly head. We have to match the track to the cluster *without* timing information because the track time comes *from* the cluster.

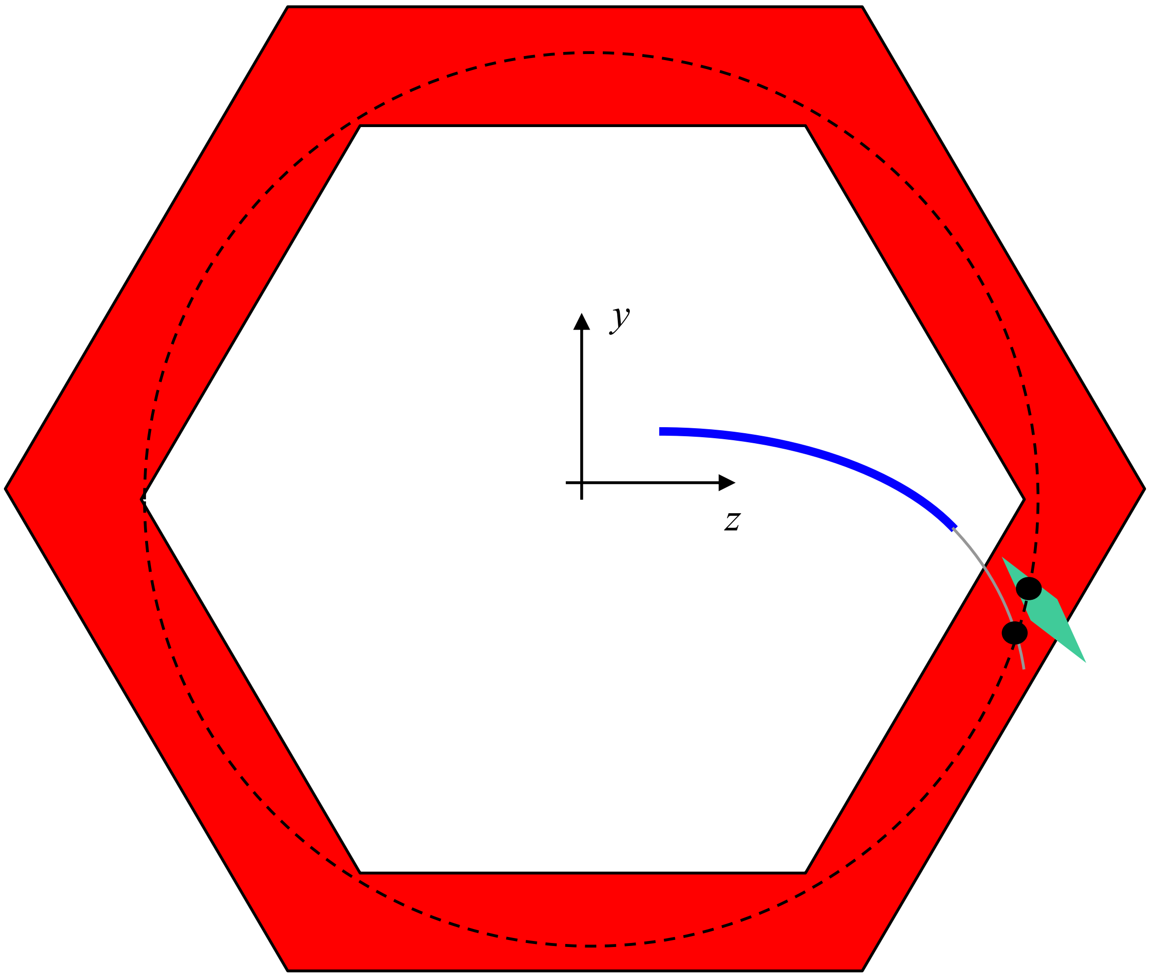
GENIE & GEANT tag a time for the interaction to occur, T_{I}, inside the $10\ \mu\mathrm{s}$ spill. Then `readoutsim` does the right thing, and computes a time for the hit to appear, T_{H}, which is

$$T_{\mathrm{H}} = (\text{drift distance})/(\text{drift speed}) + T_{\mathrm{I}}.$$

Then `reco` can only do the wrong thing, and places the track at $T_{\mathrm{H}} \times (\text{drift speed})$ from the endplate, which is too far from the endplate by $T_{\mathrm{I}} \times (\text{drift speed})$.

Result is that the extrapolated track can be closer to the cathode by some amount between 0 and $T_{\mathrm{I}} \times v_{\mathrm{DRIFT}} = 30.1$ cm.

New algorithm – cut II



1) Extrapolate the track out to a cylinder centered on the x axis of the detector and of radius given by the cluster

2) Consider the plot of

$$x_{\mathrm{EXTRAPOLATED}} - x_{\mathrm{CLUSTER}}$$

i.e. the distance between the 2 black dots in the axis perpendicular to the paper.

If on the near side of the cathode,

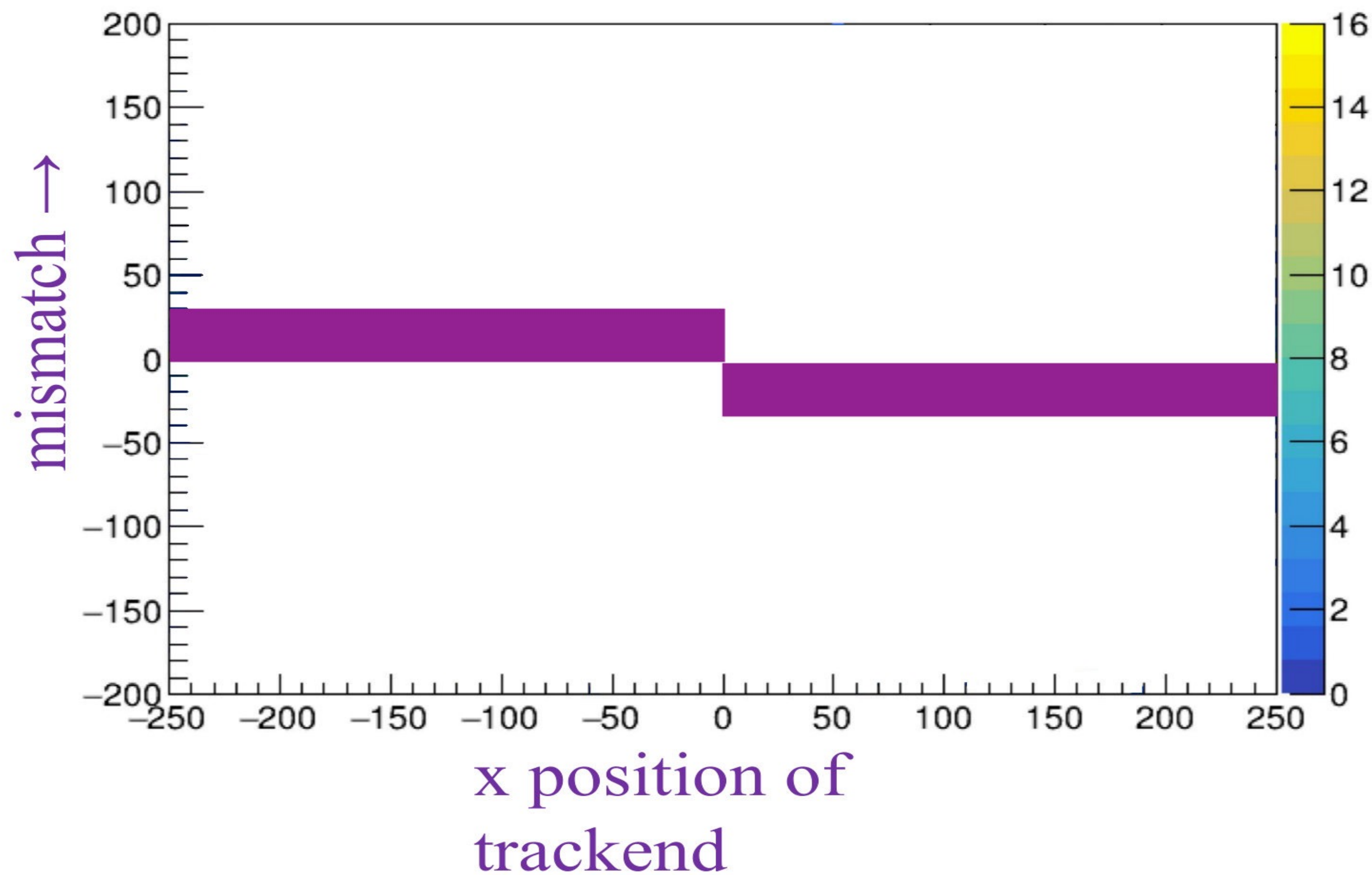
$$x_{\mathrm{EXTRAPOLATED}} < x_{\mathrm{CLUSTER}}$$

($x = 0$ at the cathode) and if on the far side,

$$x_{\mathrm{EXTRAPOLATED}} \geq x_{\mathrm{CLUSTER}}$$

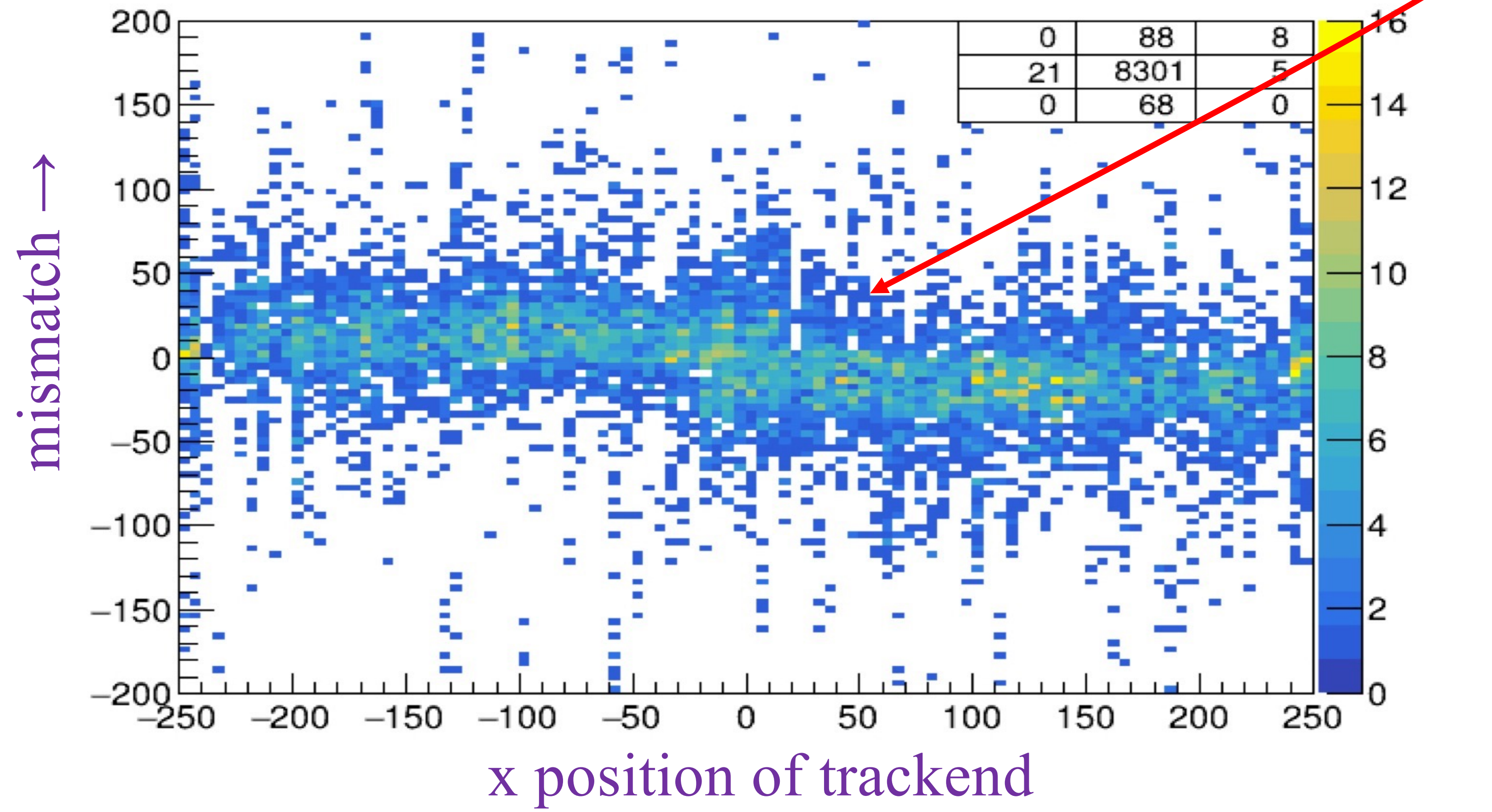
New algorithm – cut II

So a plot of $x_{\mathrm{EXTRAPOLATED}} - x_{\mathrm{CLUSTER}}$ vs $x_{\mathrm{EXTRAPOLATED}}$ in the case of no multiple scattering, perfect track and cluster fitting should look like this:



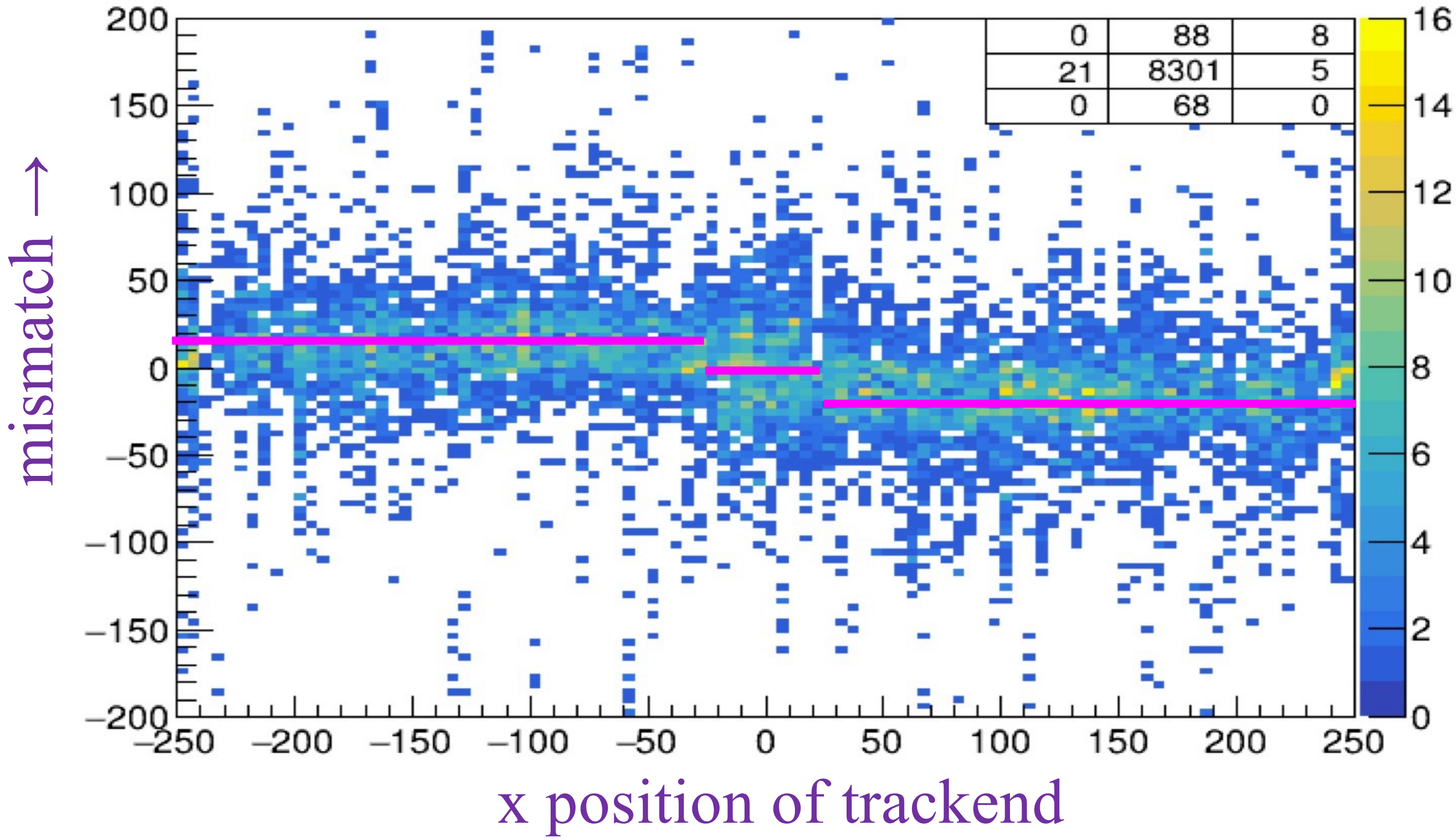
New algorithm – cut II

Actually…
Tracks that cross the cathode are "stitched" and can't have this offset



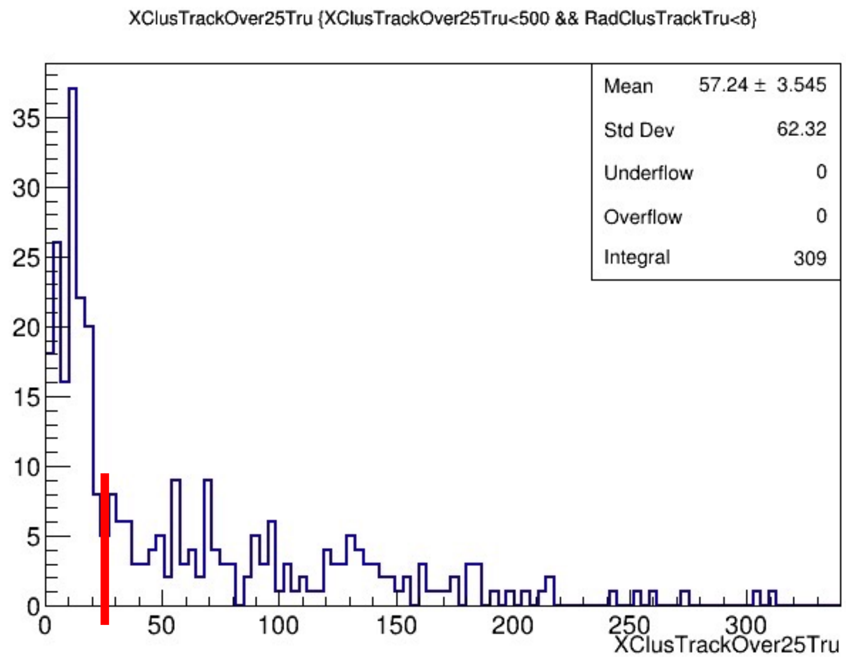
New algorithm – cut II

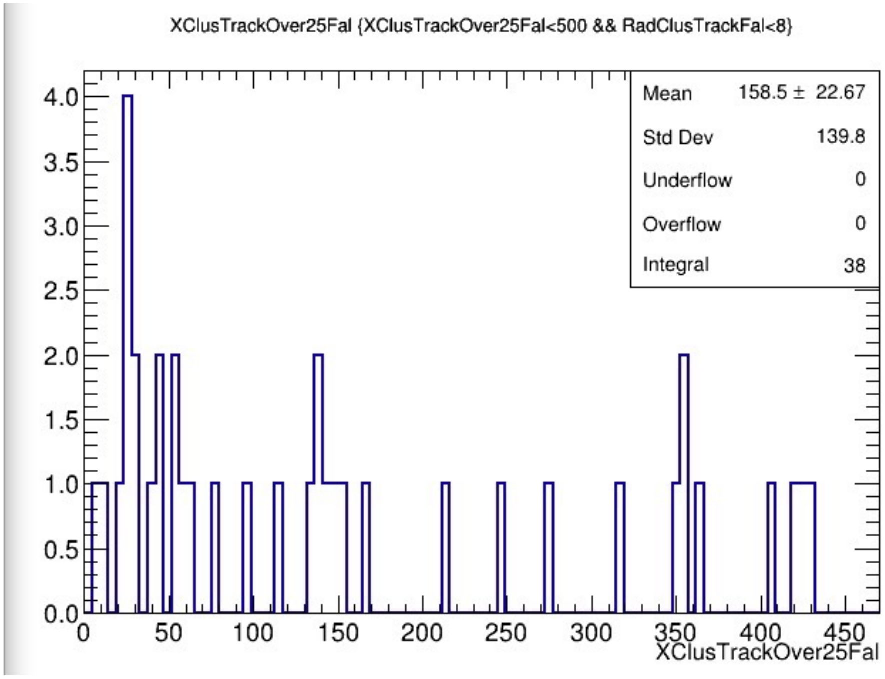
Cut on difference between the mismatch $x_{\mathrm{EXTRAPOLATED}} - x_{\mathrm{CLUSTER}}$ and the centerline of the idealized boxes



New algorithm – cut II

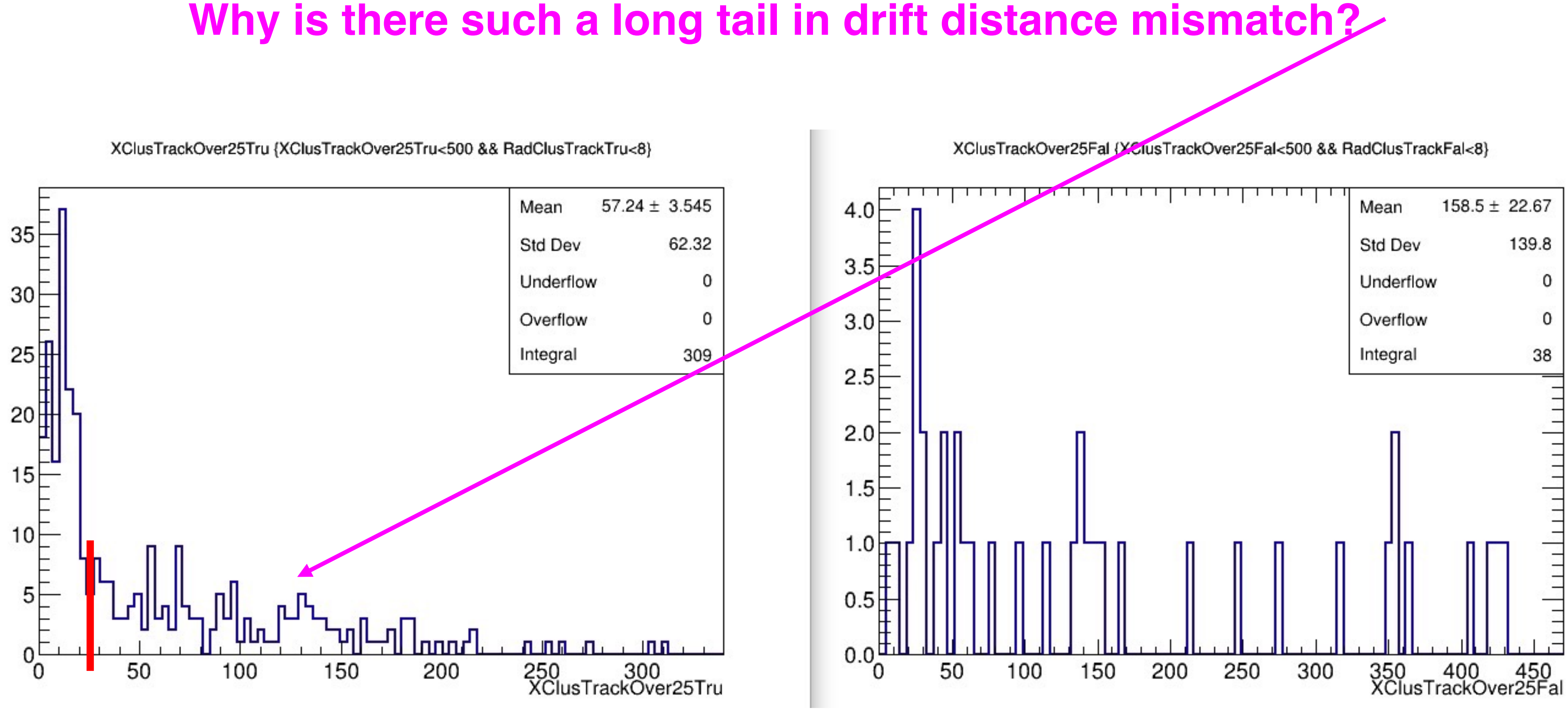
Cut at (drift distance in $10\mu s$)/2 + 10 cm = 25 cm





Whaaaaa?

Why is there such a long tail in drift distance mismatch?



Whaaaaa?

Why is there such a long tail in drift distance mismatch?

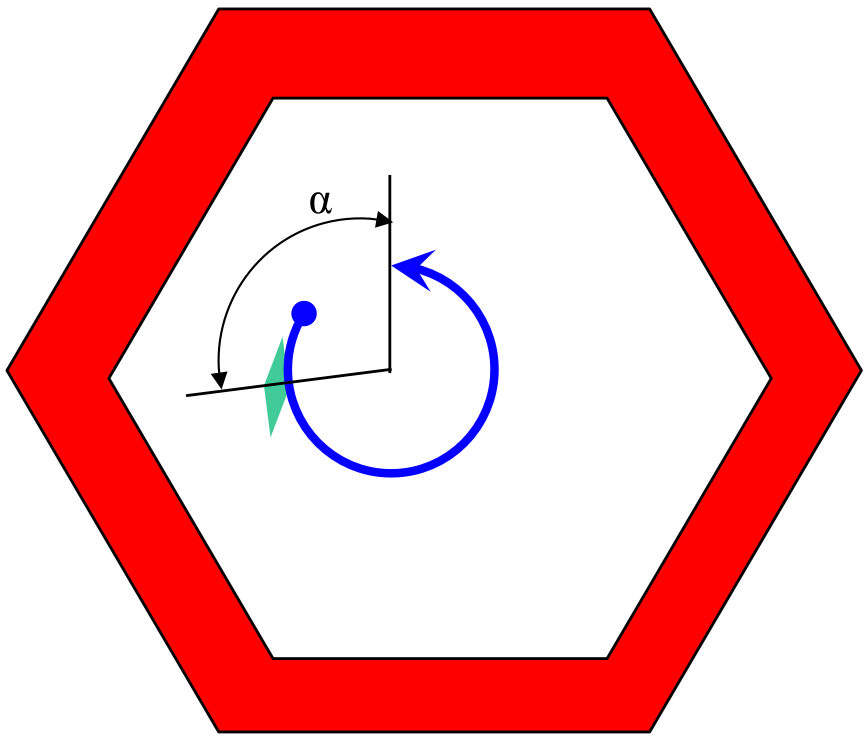
I looked at some events in *excruciating* detail

1. Despite the track quality cut of pVal>0, some tracks are still badly reconstructed and their extrapolation is invalid
2. The BackTracker assigns the "TPCeve" to the cluster based on the edeps in the cluster. A neutron can travel some distance from a DIS or Quasi-elastic in the ECAL to some other place in the ECAL
3. I have an event where the `CaloDeposits` from the GEANT stage do not **(?)** match the `CaloHits` which go into clustering – maybe related to some of Vivek's strange plots?
4. Surely there must be cases where a low-*p* track scattered through a wide angle

While the appropriate cuts are evident from the plots, we cannot get meaningful purity/efficiency numbers for ECAL/track matching

New algorithm – cut III

Cut III is similar to Cut II but for the endcap



After cut I, endcap clusters and the track are on the same circle in the (*z*,*y*) plane

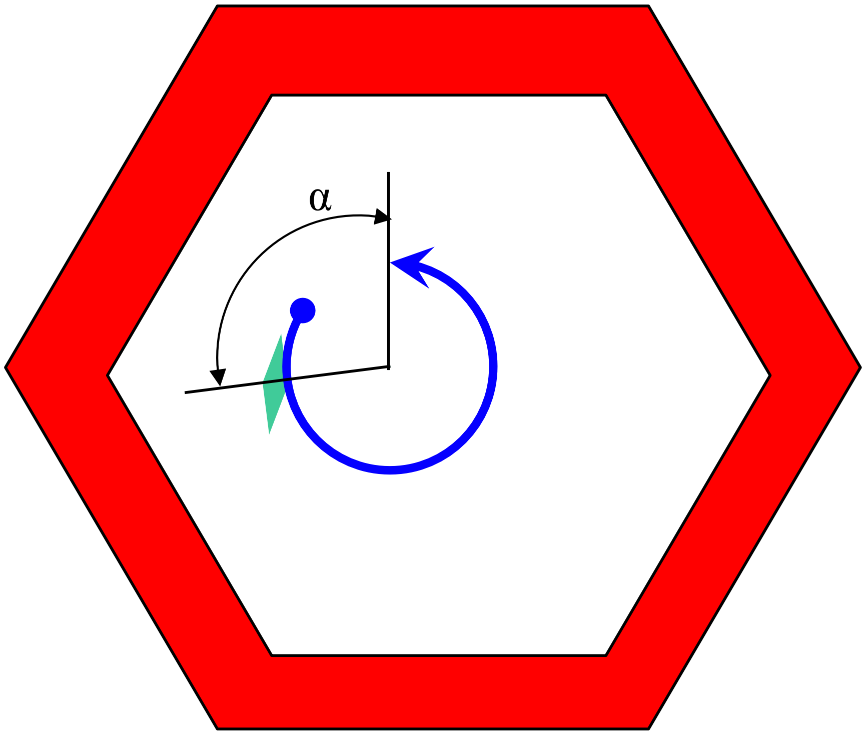
Question is, do they have consistent values of track azimuthal angle φ (actually, $r_{SPIRAL}\varphi$)?

Angle α is the advance in φ from TPC to cluster. The "correct" value depends on the distance from the extrapolating trackend to the ECAL

Which we don't know to within 30 cm because the drift velocity and spill time

New algorithm – cut III

Cut III is similar to Cut II but for the endcap



If the track is too "flat" i.e. is too close to parallel to the (*z*,*y*) plane, there is no ability to tell what α really should be.

So if

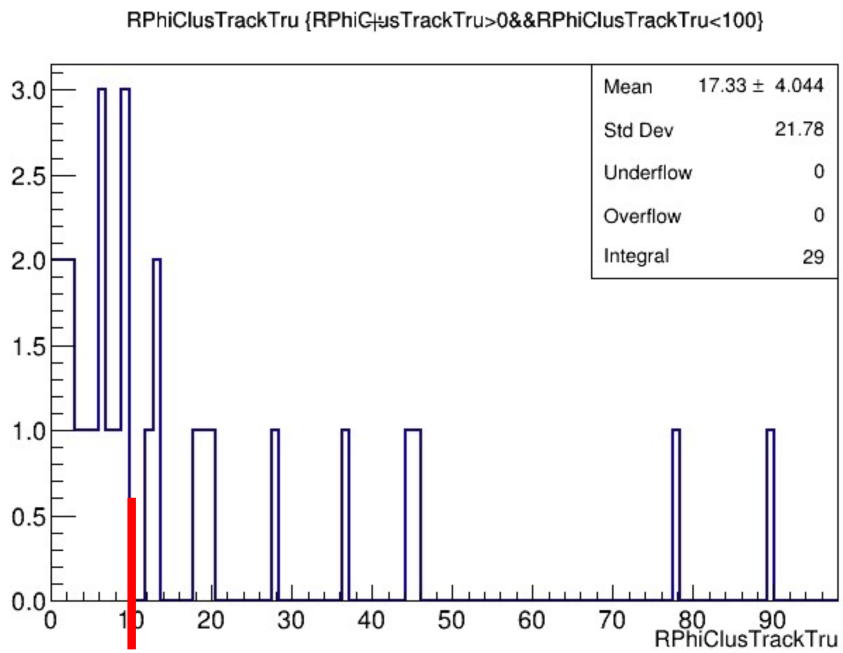
$$\left|\frac{\rho\, v_{DRIFT}\, T_{SPILL}}{\tan\lambda}\right| \geq 2\pi$$

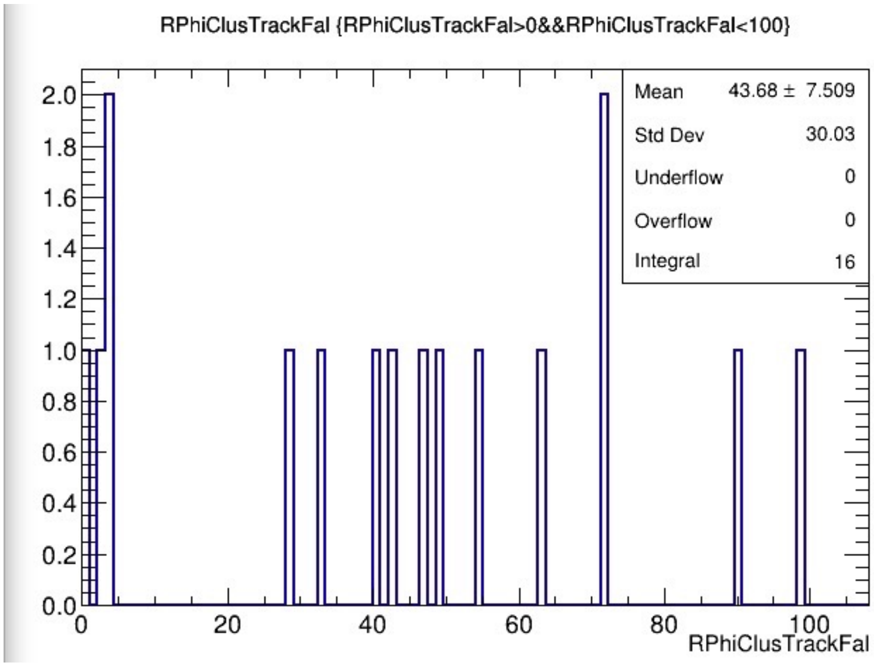
Just assume the cluster matches the track

Otherwise cut on $(\alpha)(r_{SPIRAL})$

New algorithm – cut III

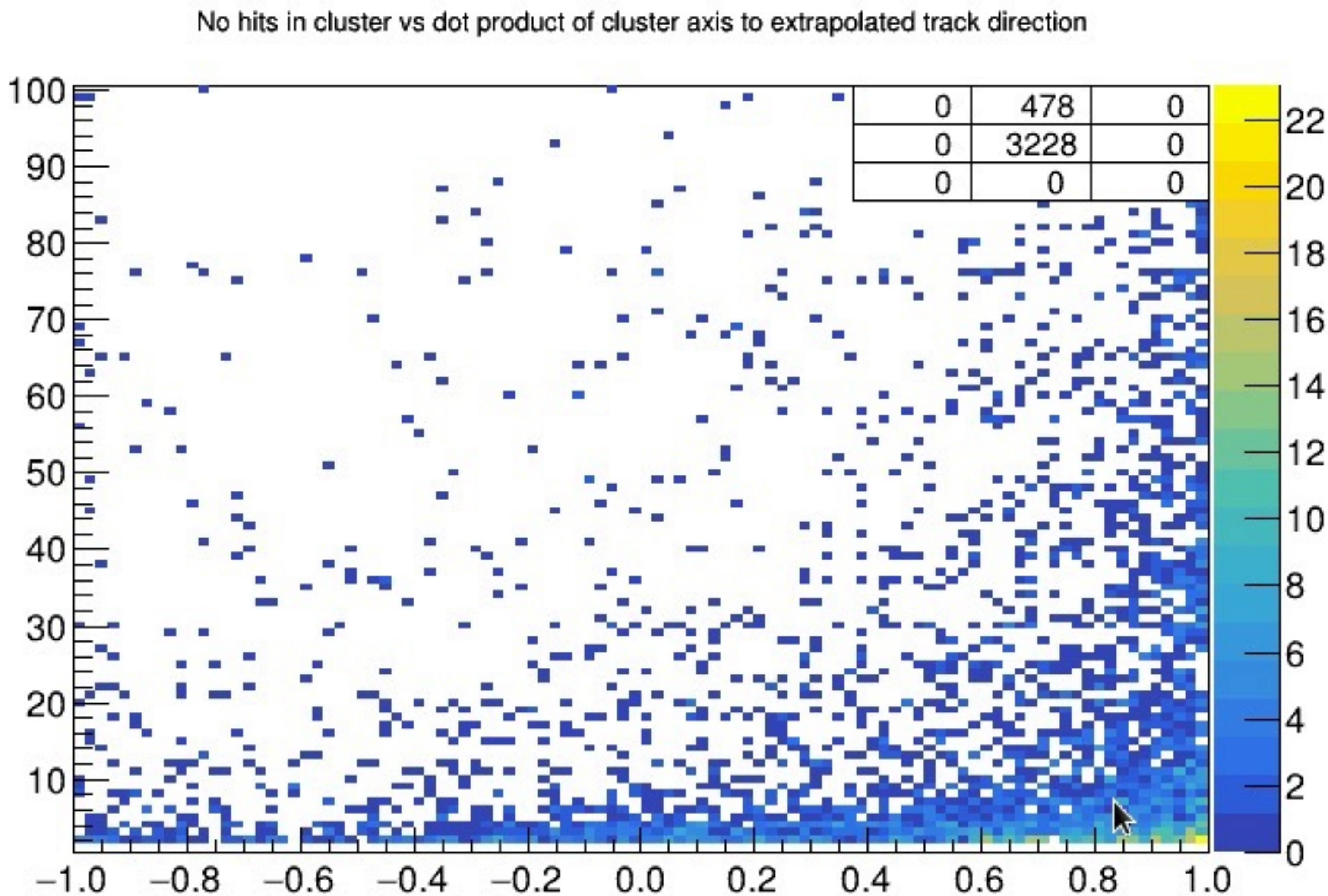
Decide to cut at 10 cm





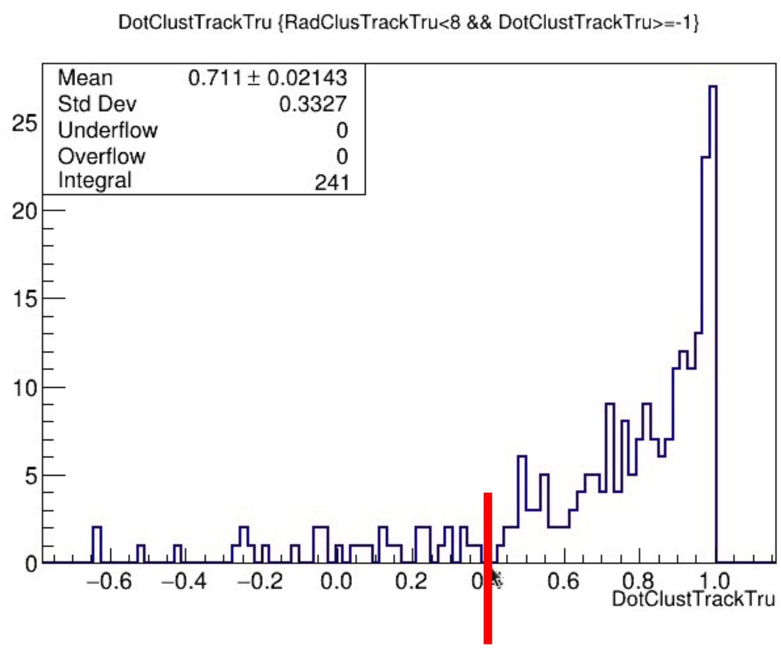
New algorithm – cut IV

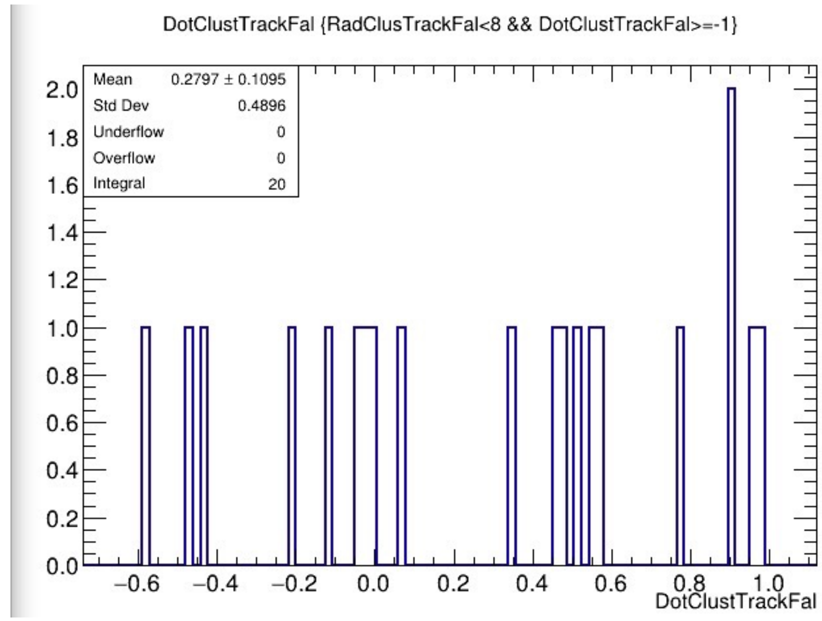
We also have some pointing ability in the ECAL



New algorithm – cut IV

After requiring 5 or more calorimeter hits in the cluster, cut on the dot product of the cluster direction re the extrapolated track direction at 0.4





Summary

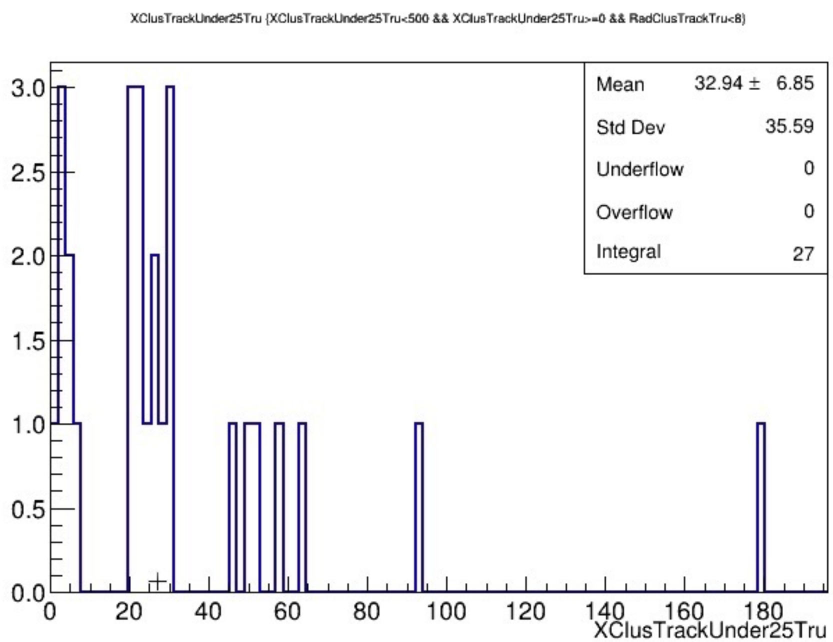
- ECAL – TPC matching about as good as it can be at this point
- Can't really quantify purity & efficiency yet
- p_{TRK} dependent matching cuts a la DØ also not easy now
- Lots of bug fixes & upgrades along the way
- What next?
    - Try to track down that ECAL hit-simulation bug
    - Put MuID into the BackTracker
    - Try again at dE/dx
    - Try again at ν_e – what if there is no upstream or side ECAL/MuID? (Trying to generate a bunch of these events now)
    - More work on this matching:
        - Tighten track quality cuts
        - (Optionally) not count neutron induced activity as part of shower
    - Some other totally different thing

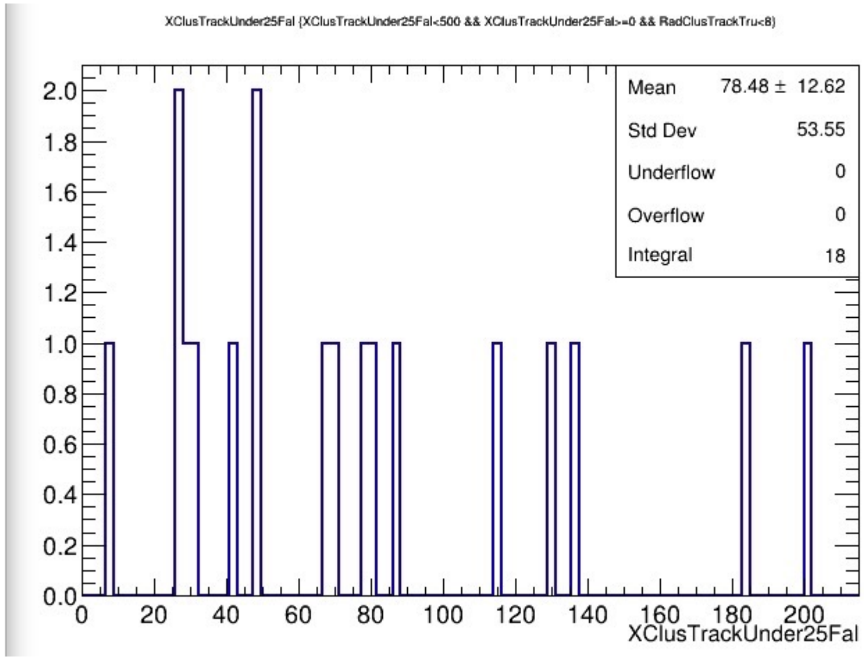
Priorities after discussion

- **Try to track down that ECAL hit-simulation bug**
- **Generate a bunch of ν_e and ask how many e– go into upstream or side ECAL**
- **(Optionally) not count neutron induced activity as part of shower**
- **Tracking: Get some quantitative assessment of what are the common failure modes, maybe look into quality cuts**
- **Put MuID into the BackTracker**
- **Try again at dE/dx**
- **ν_e as an analysis channel**

New algorithm – cut II

Here’s what the x direction mismatch looks like in the central 25cm





New algorithm – cut II

Here's what the x direction mismatch looks like in the central 25cm