TrajCluster Updates - Feb 2020

Bruce Baller
Feb 11, 2020

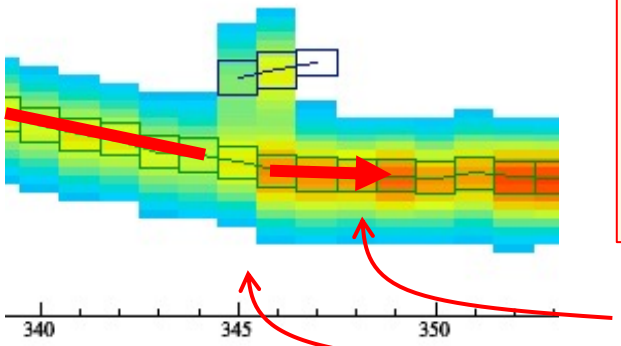
Motivation

- Tingjun informed me of some visually-obvious failures reconstructing protons with secondary interactions in ProtoDUNE - pandora and trajcluster
- Investigation led to significant improvements in TrajCluster
  - Kink detection
  - 2D vertex fitting
  - Identification of overlapping 2D trajectories
  - 3D reconstruction using dE/dx
- Which led to a re-write of the internal performance metric
- Which led to moving the performance metric to a separate module

- Some jargon: TP denotes a 2D Trajectory Point ~ a single hit (usually) dressed with local information (trajectory position & direction, environment, etc)

TrajCluster Kink Detection

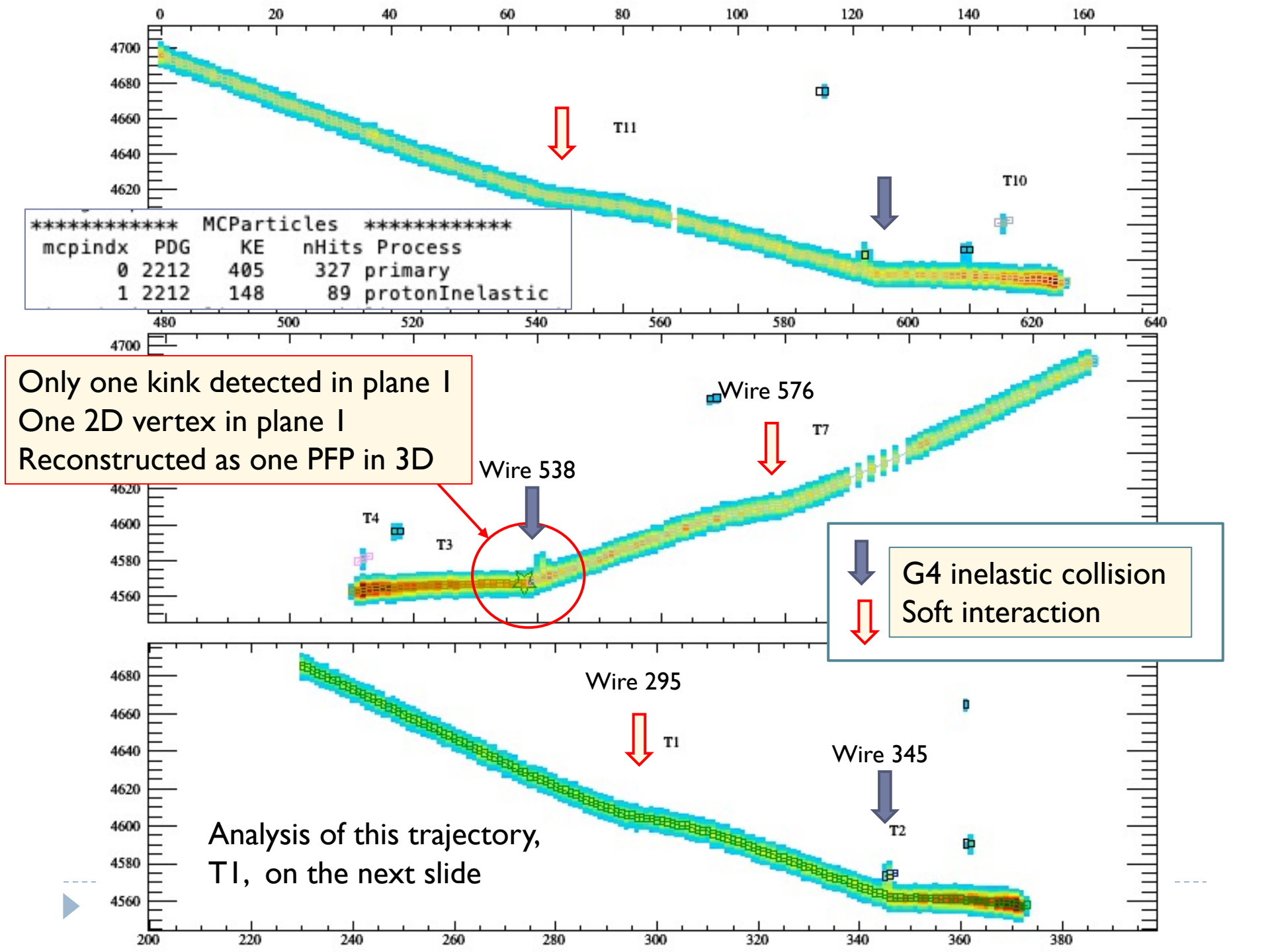
- GottaKink is an old algorithm that fits the last nPtsFit TPs added to a trajectory under construction (nPtsFit ~ 3) to a line
  - It looks for large angle differences before - after the fit point
- A kink is declared when the angle significance exceeds a cut

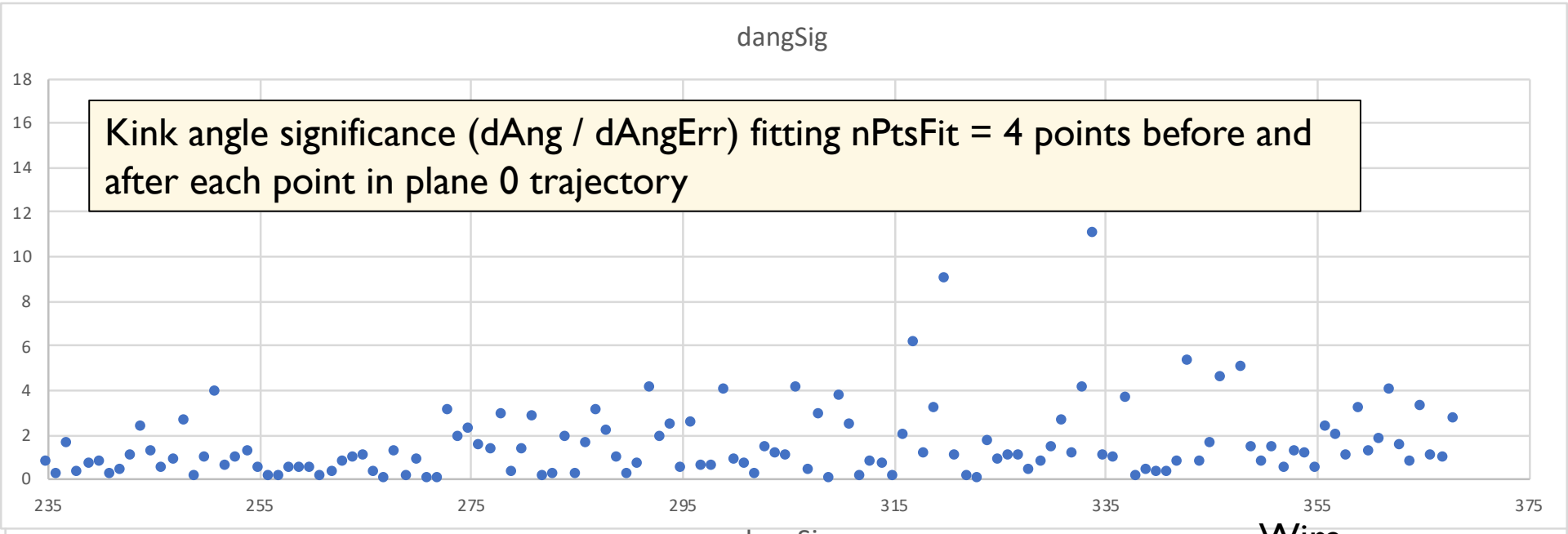


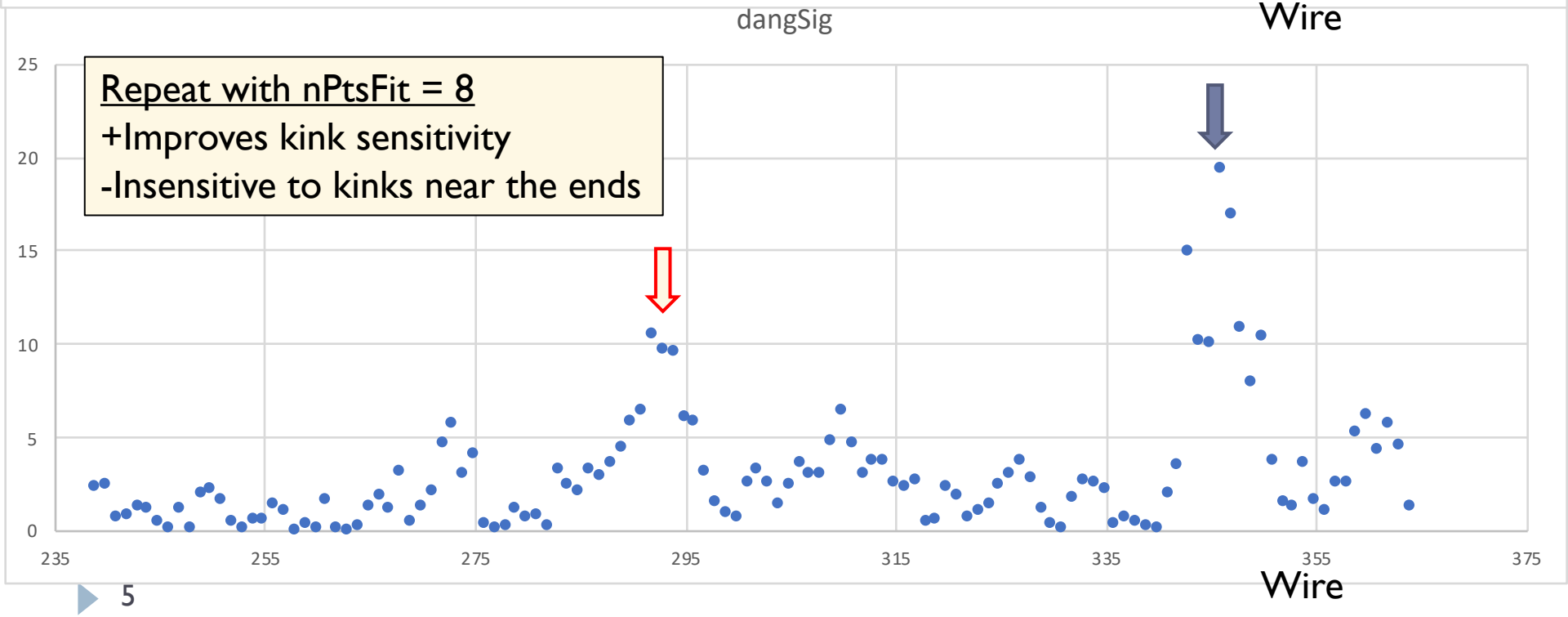
This fails for small angle kinks → misses 2D vertices that don't match in 3D → misses 3D vertices.
May not get the kink point correct.

Last point added
Presumed kink point

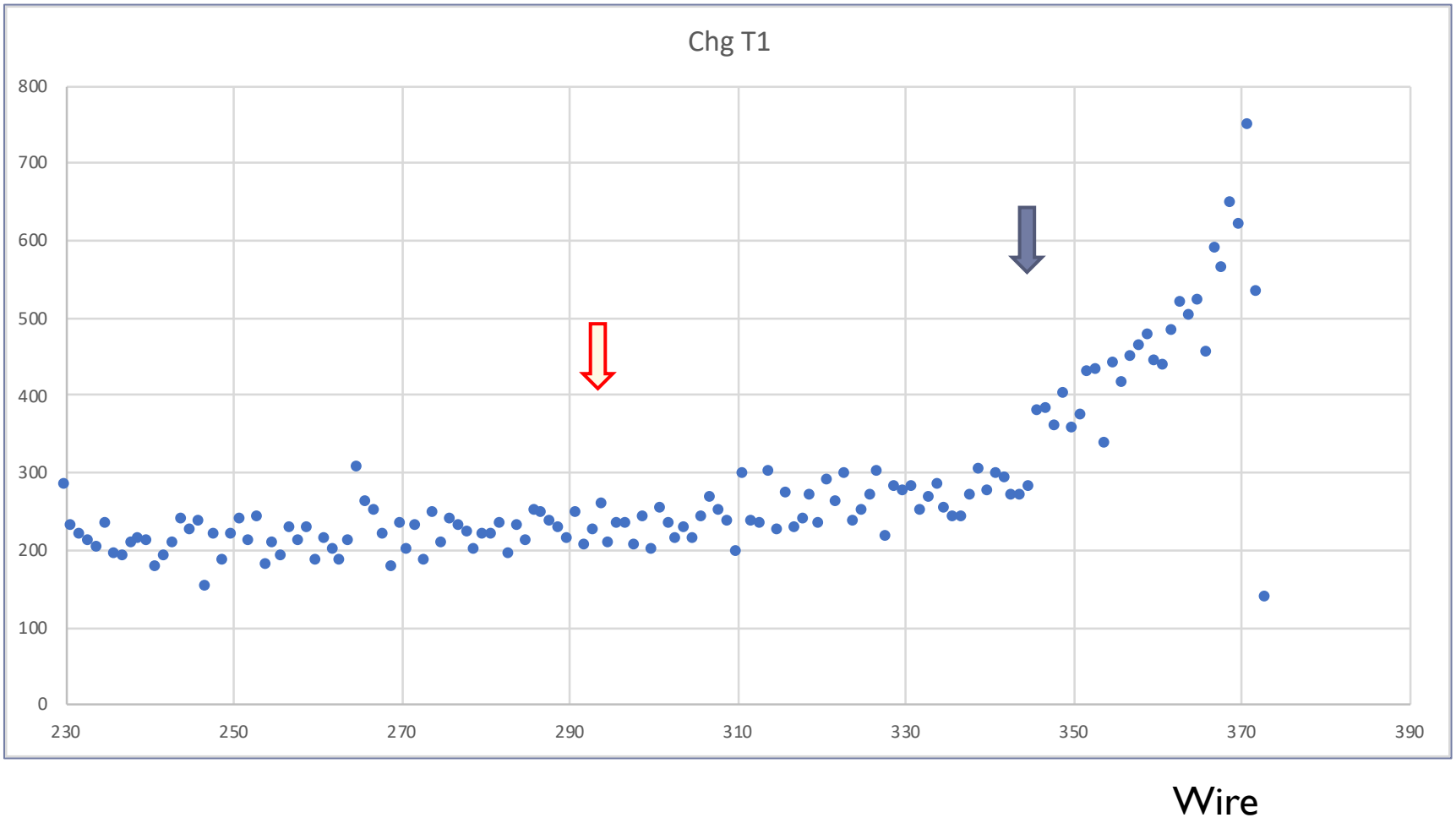
- Example of a failure on the next slide


T11
T10
********** MCParticles **********
mcpindx PDG KE nHits Process
0 2212 405 327 primary
1 2212 148 89 protonInelastic
Only one kink detected in plane 1
One 2D vertex in plane 1
Reconstructed as one PFP in 3D
Wire 576
T7
Wire 538
T4
T3
G4 inelastic collision
Soft interaction
Wire 295
T1
Wire 345
T2
Analysis of this trajectory,
T1, on the next slide


dangSig
Kink angle significance (dAng / dAngErr) fitting nPtsFit = 4 points before and after each point in plane 0 trajectory
18
16
14
12
10
8
6
4
2
0
235
255
275
295
315
335
355
375
Wire

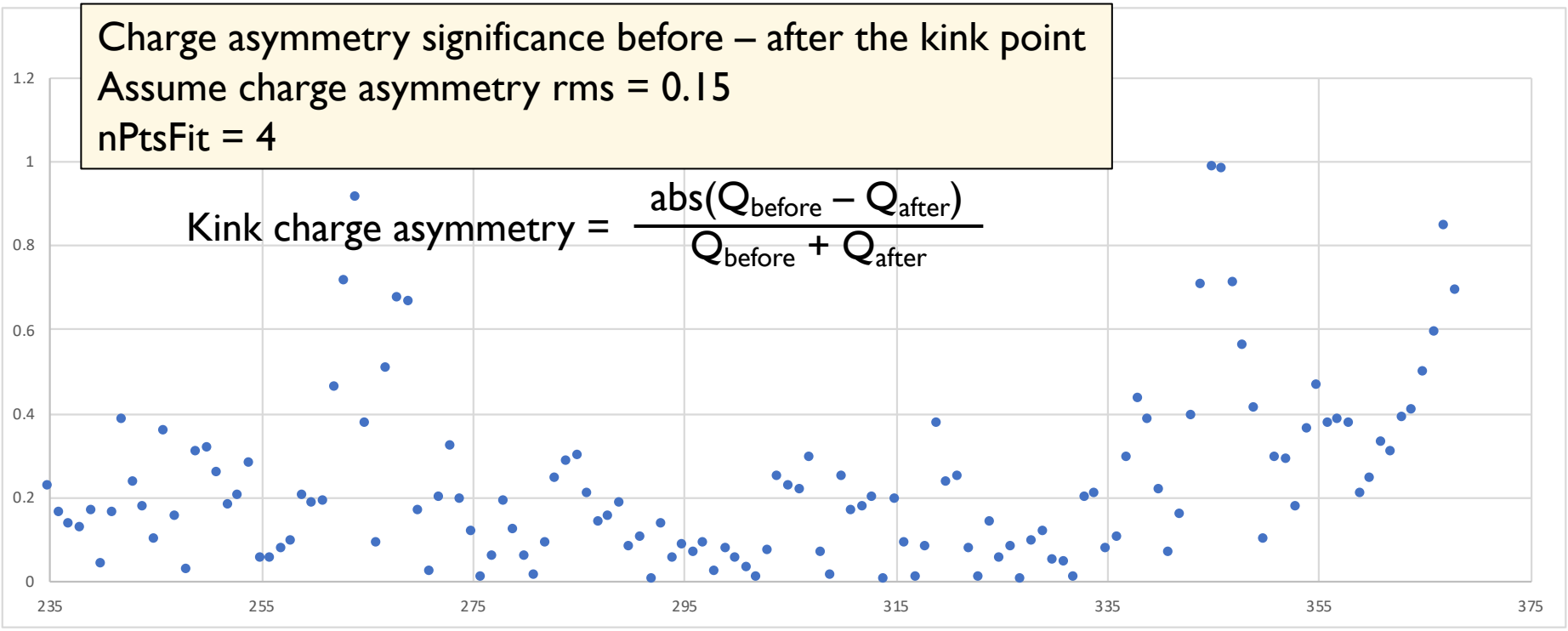

dangSig
Repeat with nPtsFit = 8
+Improves kink sensitivity
-Insensitive to kinks near the ends
25
20
15
10
5
0
235
255
275
295
315
335
355
375
Wire

TP Charge

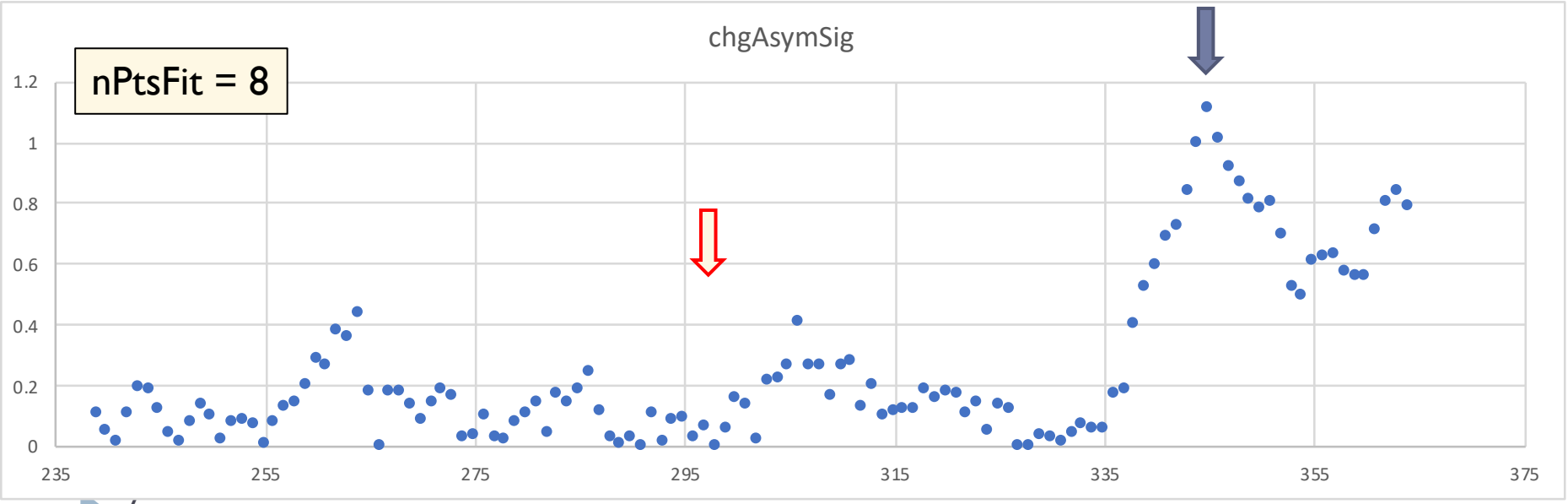

Chg T1
TP Charge
800
700
600
500
400
300
200
100
0
230
250
270
290
310
330
350
370
390
Wire

Charge asymmetry significance before – after the kink point
Assume charge asymmetry rms = 0.15
nPtsFit = 4

$$\text{Kink charge asymmetry} = \frac{\text{abs}(Q_{before} - Q_{after})}{Q_{before} + Q_{after}}$$

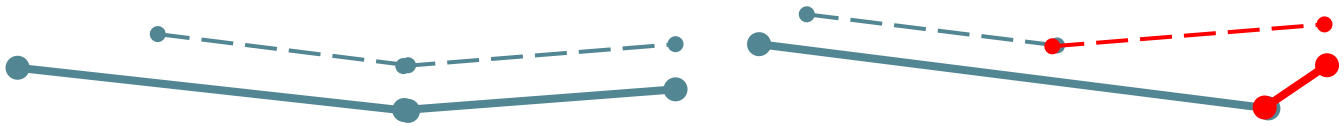


nPtsFit = 8



Observations

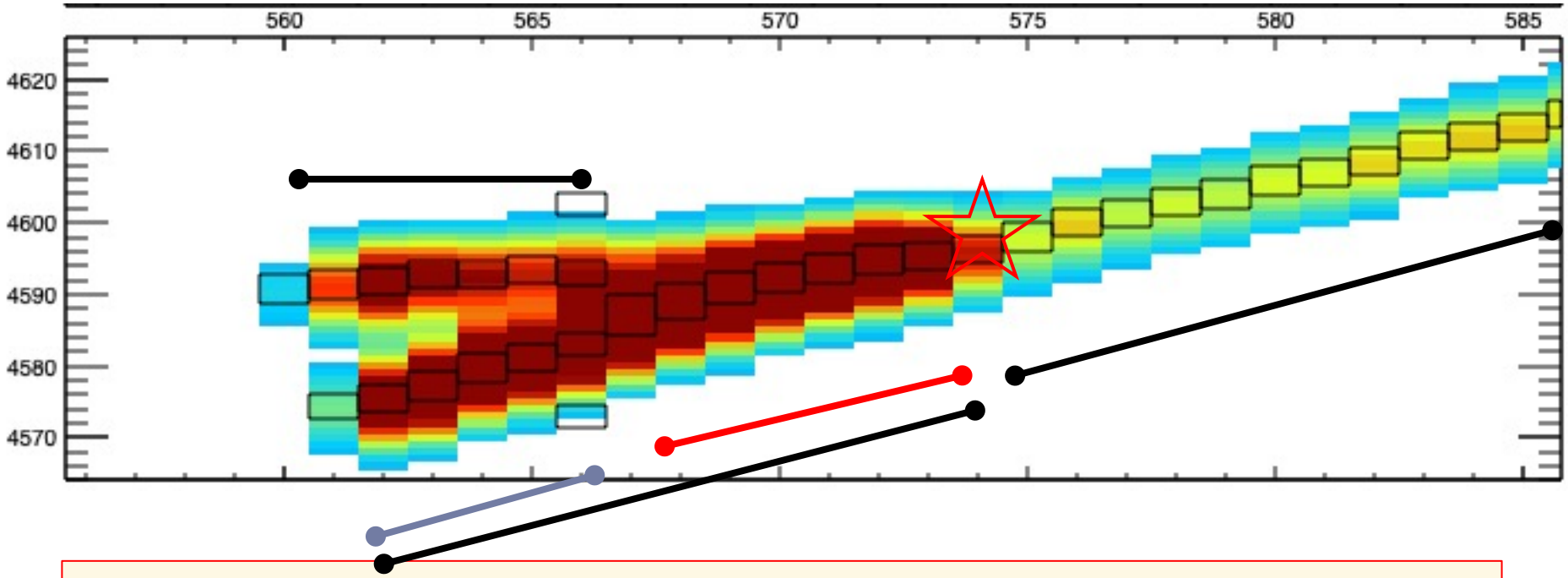
- The fit angle error using nPtsFit = 3 in a single plane is inadequate for small-angle kinks
  - Equivalent nPtsFit for human-eye kink detection using the event display is 10+
- Using a simple stop-tracking-when-above-threshold cut gets the kink point wrong for small-angle kinks
  - Need to wait until the kink is characterized before deciding
- Increasing nPtsFit increases the significance of small-angle kinks on long trajectories but decreases the significance for large-angle decay kinks at the end



Kink Updates in 2D

- Recommend users set nPtsFit to 8
  - fcl configuration
- Added a new kink significance variable to the Trajectory Point object
  - Kink angle significance * (1 + kink charge asymmetry)
- Significant changes to the kink finding algorithm
  - KinkCuts fcl configuration vector change
- Added an algorithm to check for large-angle kinks at the end of trajectories

2D Vertex Fit – Overlapping Trajectories



The fitted position of a reconstructed 2D vertex (open star) using the black Tjs will be biased by the TPs in the overlapping region (red bar) if used in the vertex fit. Also, hits in the overlap region should not be used when calculating dE/dx.

New: Identify trajectories that have overlapping TPs near 2D vertices then re-fit the vertex position using TPs outside the overlap region. Flag overlapping TPs and ignore them when calculating dE/dx in 3D – No mechanism exists to pass this information to downstream modules

2D Vertex Fit

- An ancient algorithm with untrustworthy errors and Chisq/DOF
- Improved using ROOT matrix methods

```
TMatrixD A(npts, 2);
TVectorD b(npts);
for(unsigned short itj = 0; itj < vxTPs.size(); ++itj) {
  auto& tp = vxTPs[itj];
  double dtdw = tp.Dir[1] / tp.Dir[0];
  double wt = 1 / (tp.AngErr * tp.AngErr);
  A(itj, 0) = -dtdw * wt;
  A(itj, 1) = 1. * wt;
  b(itj) = (tp.Pos[1] - tp.Pos[0] * dtdw) * wt;
} // itj
```

TPs used in the vertex fit

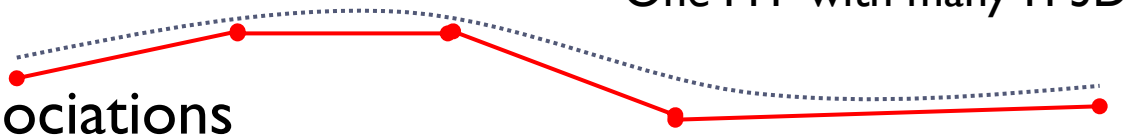
Can't get errors if TDecompSVD solve() is used
Errors from the covariance matrix →

```
TMatrixD AT(2, npts);
AT.Transpose(A);
TMatrixD ATA = AT * A;
double *det = 0;
ATA.Invert(det);
if(det == NULL) return false;
TVectorD vxPos = ATA * AT * b;
vx.PosErr[0] = sqrt(ATA[0][0]);
vx.PosErr[1] = sqrt(ATA[1][1]);
vx.Pos[0] = vxPos[0];
vx.Pos[1] = vxPos[1];
```

3D Updates
but first some TrajCluster conventions…

- A PFP is a 3D trajectory composed of a vector of 3D trajectory points, TP3Ds, sorted by path length with each assigned to a "section." 3D line fits are done in each section

One PFP with many TP3Ds in 4 sections



- Associations
  - 2D
    - Trajectory owns TPs
    - Optional Trajectory [end] → 2D vertex
  - 3D
    - 3D vertex → 2D vertices in several planes
    - First: PFP → 3D-matched 2D trajectories
    - Second: PFP owns TP3Ds each having a one-to-one TP3D → TP assn
      - 2D TP ownership ignored until the final set of Tjs (clusters) are constructed

PFP Construction Updates

- Use SpacePoints to match trajectories in 3D (optional)
  - Assns: SpacePoint → 3 Hits → 3 TPs → 3 trajectories
  - Count matches for all combinations of Tjs then sort by decreasing TP match count
  - Much faster than TrajCluster 3D matching (~6x in ProtoDUNE)
  - Not used for PFP construction – efficiency ~ 0.9
- Improved the algorithm for associating TPs (2D) to PFPs (3D)
  - Consider overlapping 2D trajectories (ref Slide 10)
  - Consider dE/dx when adding/removing TPs to PFPs
- New 3D kink finding algorithm
  - Uses 3D kink angle difference * (2D) TP kink significance
- New algorithms to reconcile 2D – 3D vertex assn conflicts

Monitoring Performance - New ClusterAnaV2 Module

- Moved the internal TrajCluster metrics Efficiency, Purity and Efficiency * Purity (EP) to a new module
  - A hit is considered MC-matched if the IDE energyFrac > 0.5
  - A set of "cluster" hits is matched to a MCParticle if the hits in each TPC and each plane have the most MC-matched hits to that MCParticle
    - Note that low-purity clusters may be matched to more than one MCParticle (see backup slides)
- Metrics calculated separately for electrons, muons, pions, kaons and protons and then averaged
  - Option to ignore user-selected PDG codes, e.g. electrons
- All MCParticles are weighted equally
  - Missing correct MC-matched hits (low Eff) or adding incorrect hits (low Pur) has a more significant effect on the average EP for short clusters than long clusters
- User supplies either a Cluster module label OR a Track module label
  - Hits are filtered for each TPC and plane if clusters/tracks span several TPCs
- User options to print increasing levels of detail for all events (PrintLevel) or those that that are poorly reconstructed (BadEP)

Monitoring Performance - New ClusterAnaV2 Module

- Motivation: A few well-understood numbers (0 ... 1)
  - Goal is to make a code change or fcl configuration change and evaluate the change quickly (~< 30 minutes)
    - ~100 iterations for the improvements described here
  - Compare performance of different reconstruction modules
  - Compare reconstruction performance in 2D and 3D
    - See a slight loss in 3D vs 2D for TrajCluster and Pandora
  - Enables doing a fcl configuration grid search to optimize cuts for specific analyses or experiments without a module expert
    - A separate talk…

Monitoring Performance

- Use 350 ProtoDUNE MCC12 1 GeV proton events
  - *PDSPProd2_protoDUNE_sp_reco_35ms_sce_datadriven…*
  - Beam statistics: 387 muons, 213 pions, 638 protons
  - Sample size is enough to measure 1% differences
- Compare trajcluster 2D reconstruction with pandora
  - Trajcluster produces clusters and a refined hit collection

```
physics.analyzers.clusterana.HitModuleLabel: "trajcluster"
physics.analyzers.clusterana.ClusterModuleLabel: "trajcluster"
physics.analyzers.clusterana.TrackModuleLabel: "NA"
physics.analyzers.clusterana.TruthOrigin: 4 # 0 (anything) 1(nu), 2(cosmics), 3(SN nu), 4(SingleParticle)
physics.analyzers.clusterana.SkipPDGCodes: [ 11 ] # ignore MCParticles with these PDG codes
physics.analyzers.clusterana.PrintLevel: 0
physics.analyzers.clusterana.BadEP: 0.5 # print cluster/track info if Eff*Pur < BadEP
```

  - Pandora clusters reference the hitpdune hit collection = gaushit + disambiguation for wrapped wires

```
physics.analyzers.clusterana.HitModuleLabel: "hitpdune"
physics.analyzers.clusterana.ClusterModuleLabel: "pandora"
physics.analyzers.clusterana.TrackModuleLabel: "NA"
physics.analyzers.clusterana.TruthOrigin: 4 # 0 (anything) 1(nu), 2(cosmics), 3(SN nu), 4(SingleParticle)
physics.analyzers.clusterana.SkipPDGCodes: [ 11 ]
physics.analyzers.clusterana.PrintLevel: 0
physics.analyzers.clusterana.BadEP: 0.5
```

First Results – 2D Reconstruction

- TrajCluster

```
ClusterAnaV2 summary results for 350 events using ClusterModuleLabel: trajcluster Origin: 4
Efficiency (Eff), Purity (Pur) and Eff * Pur (EP) by selected truth particle types
particle     Eff     Pur     EP
------------------------------
   Mu      0.910   0.875   0.853
   Pi      0.905   0.828   0.753
   P       0.892   0.788   0.720
Averages for all selected truth particles
 Ave Eff 0.900 Ave Pur 0.823 Ave EP 0.769
 Cnts Mu 415 Pi 213 P  638
```

Trajcluster has significantly higher EP comparing TrajCluster today vs Pandora circa MCC12 production

- Pandora

```
ClusterAnaV2 summary results for 350 events using ClusterModuleLabel: pandora Origin: 4
Efficiency (Eff), Purity (Pur) and Eff * Pur (EP) by selected truth particle types
particle     Eff     Pur     EP
------------------------------
   Mu      0.964   0.877   0.865
   Pi      0.912   0.692   0.658
   P       0.800   0.487   0.472
Averages for all selected truth particles
 Ave Eff 0.872 Ave Pur 0.649 Ave EP 0.632
 Cnts Mu 415 Pi 213 P  638
```

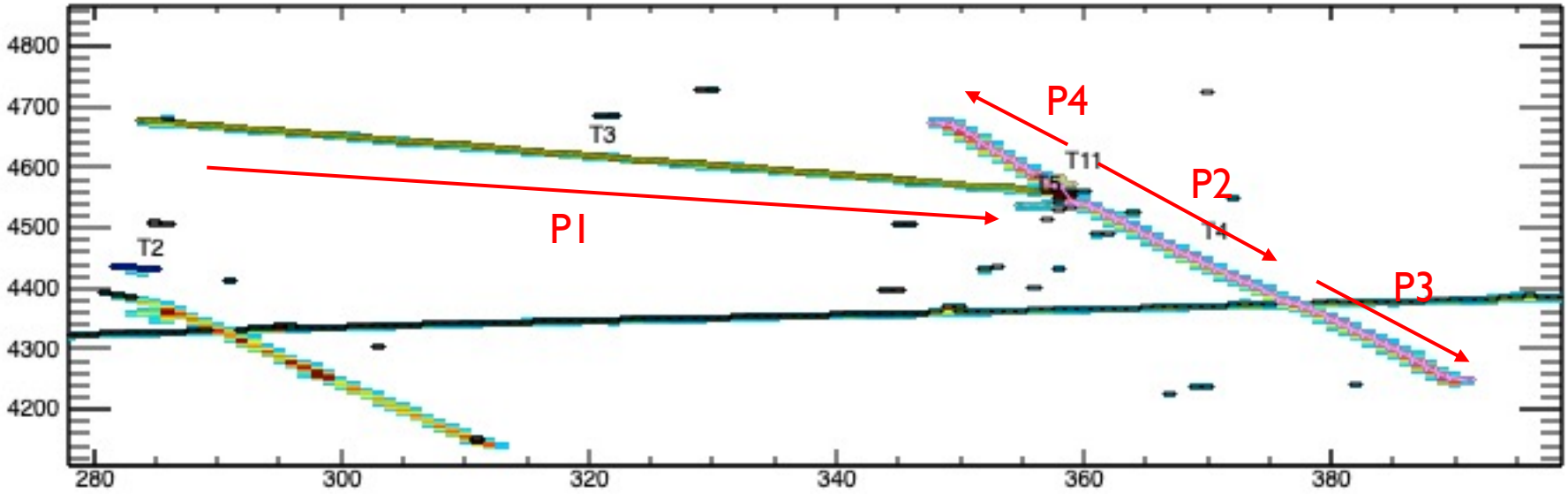
Feb 10 Update

TrajCluster EP degraded after code clean-up and updating production fcl files.
No change in the Pandora EP

Summary

- Significant improvements to TrajCluster
- Internal performance metrics moved to ClusterAnaV2
  - Includes a mechanism for tracing reconstruction problems
    - Not described here
  - Comparison of ClusterAnaV2 and Pandora performance metrics underway
  - Propose re-naming ClusterAnaV2 → ClusterAna
- Code lives on feature branch larreco/bb_TCWork2
  - Code cleanup requested by Kyle completed
  - But found a change in TrajCluster EP yesterday after "final" testing…
    - Suspect that clusteralgorithms.fcl wasn't updated correctly

Backup Slide – EP Metric



T3 matched to 900 MeV proton (P1). Efficiency = 0.99, Purity = 1.0
T4 matched to 266 MeV proton (P2). Efficiency = 1.0, Purity = 0.61
T4 matched to 122 MeV proton (P3). Efficiency = 0.79, Purity = 0.25
T4 matched to 76 MeV proton (P4). Efficiency = 1.0, Purity = 0.14

Metric averages based on 4 MCParticles and 2 trajectories (clusters)