TrajCluster Updates - Jan 2020

Bruce Baller
January 15, 2020

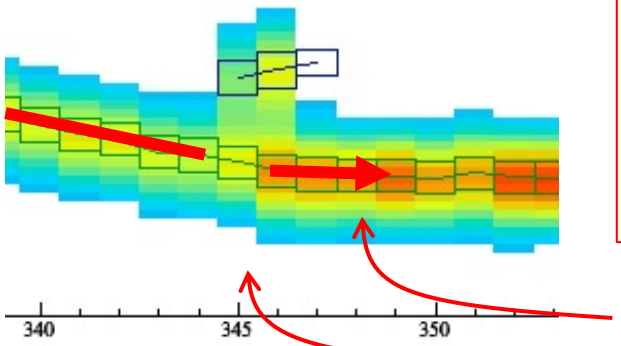
Motivation

- Tingjun informed me of some visually-obvious failures reconstructing protons with secondary interactions in ProtoDUNE - pandora and trajcluster
- Investigation and mitigation 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 a re-write of the ClusterAna reconstruction performance module

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

Kink Detection

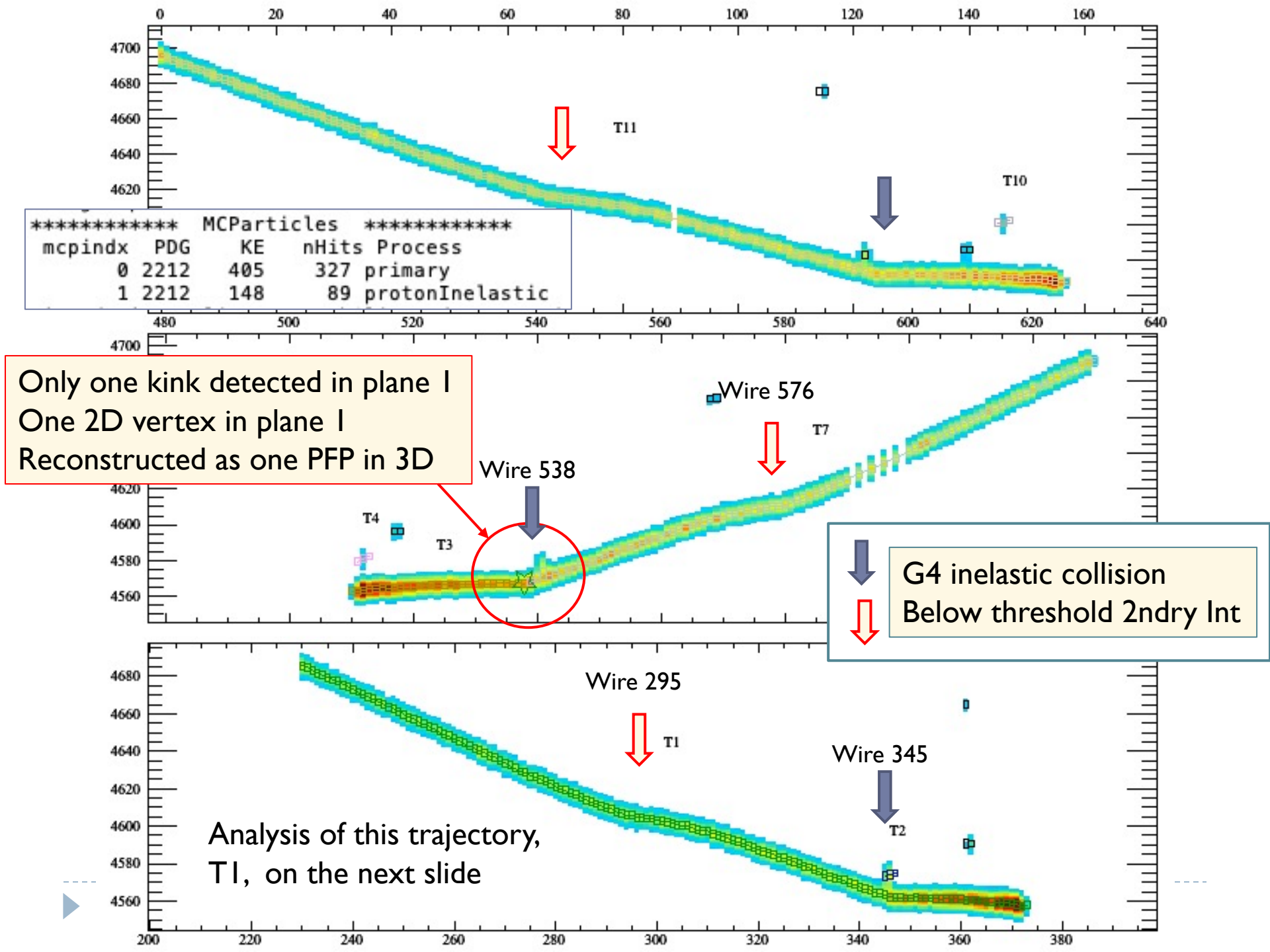
- GottaKink is an old algorithm that fits the last nPtsFit TPs added to a trajectory under construction (nPtsFit ~ 3) to a line
  - It looks for large angle differences before - after the fit point
- A kink is declared when the angle significance exceeds a cut

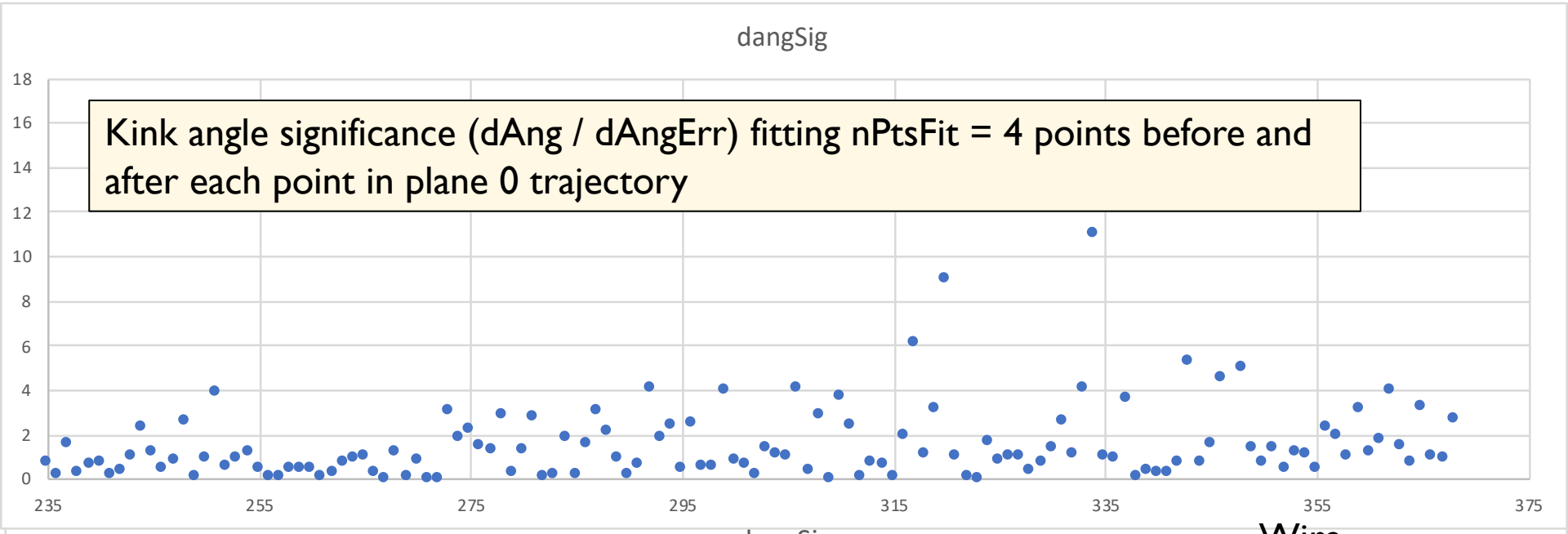


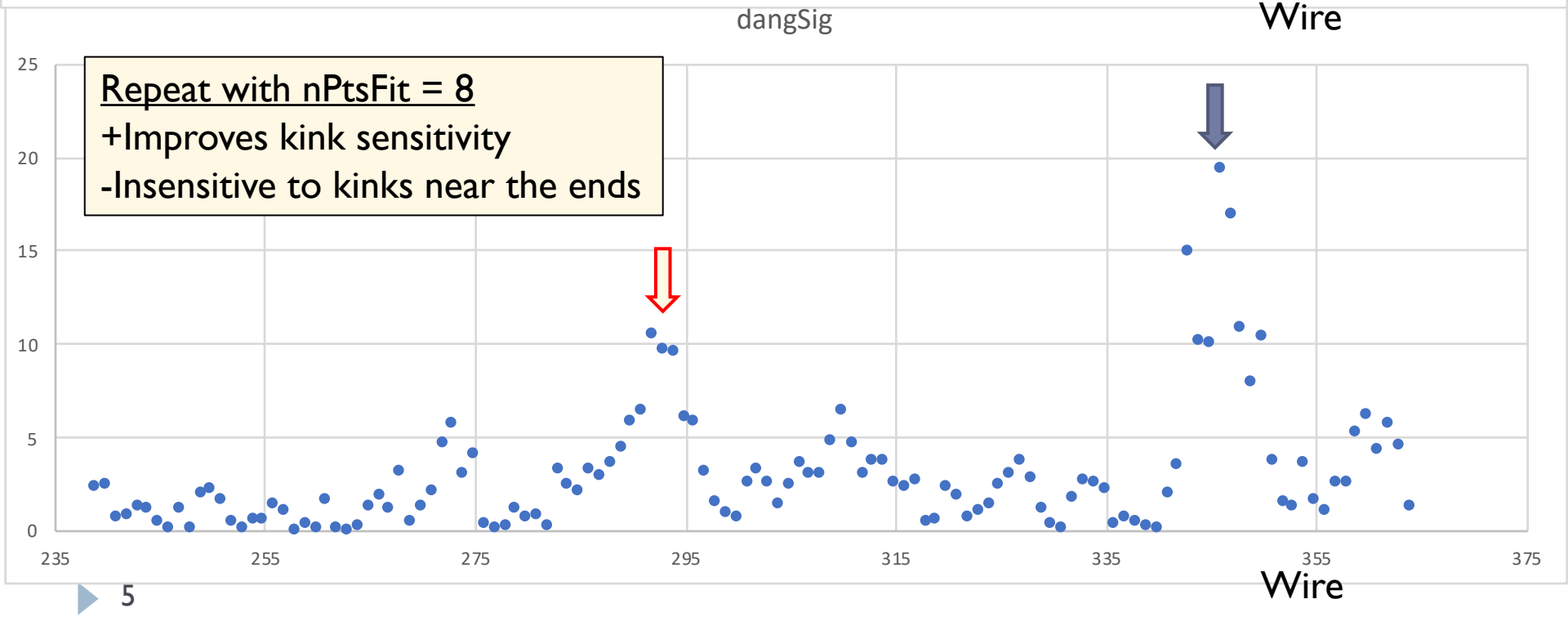
This fails for small angle kinks → misses 2D vertices that don't match in 3D → misses 3D vertices.
May not get the kink point correct.

Last point added
Presumed kink point

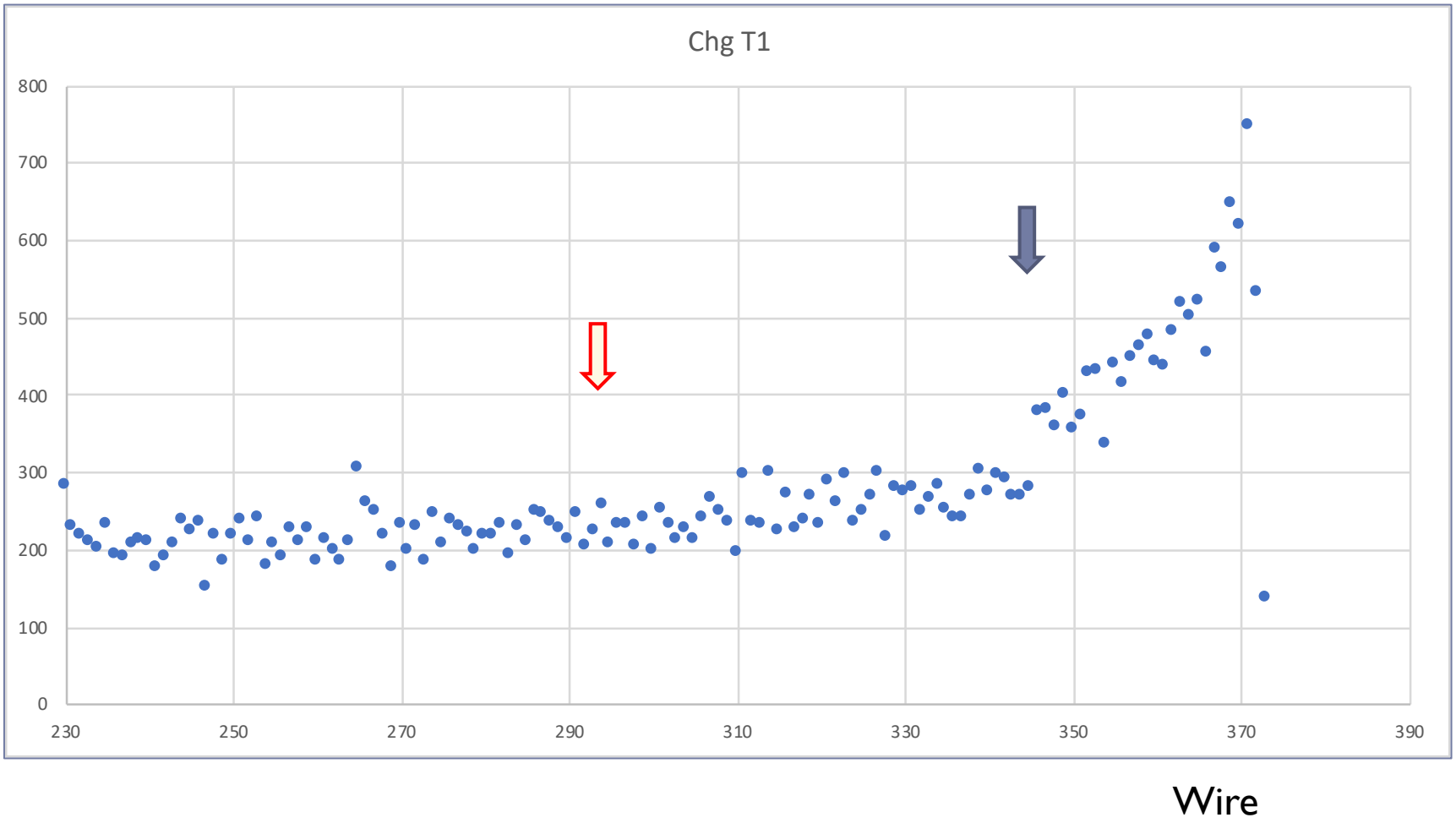
- Example of a failure on the next slide


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
Below threshold 2ndry Int
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

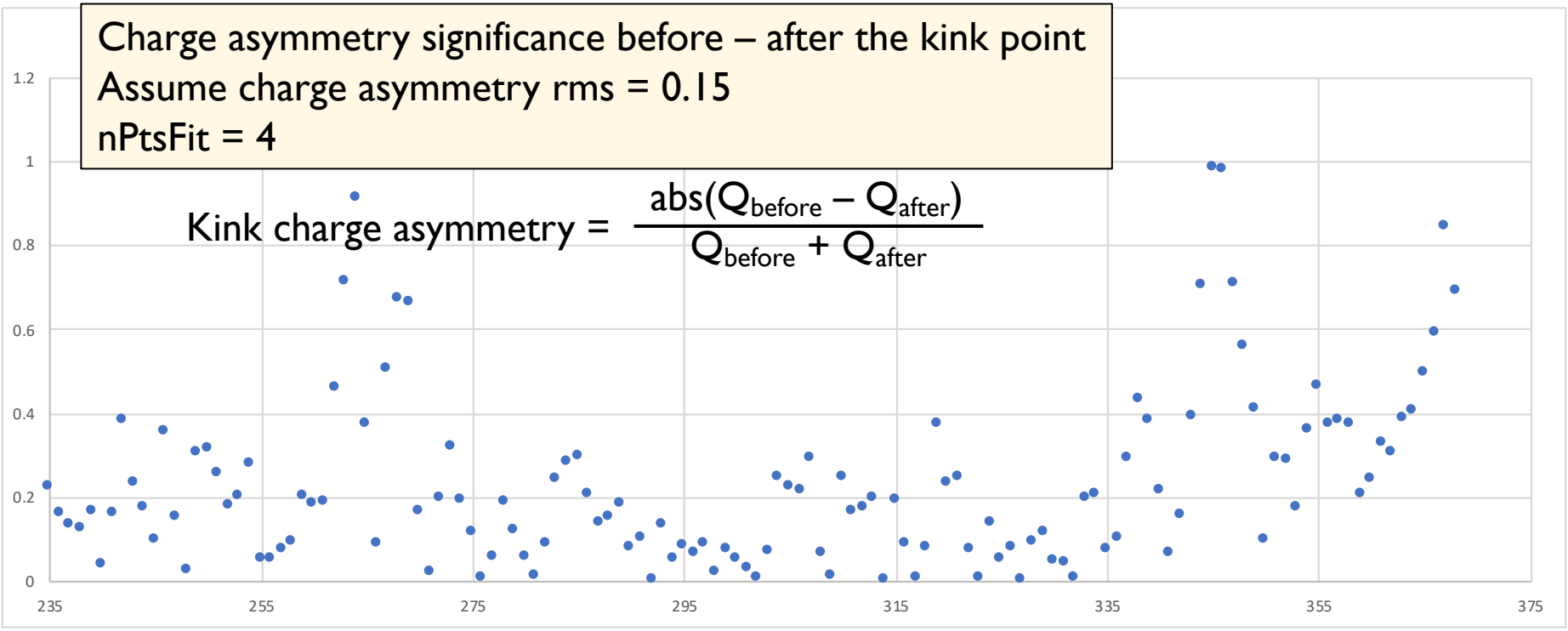

dangSig
Repeat with nPtsFit = 8
+Improves kink sensitivity
-Insensitive to kinks near the ends
25
20
15
10
5
0
235
255
275
295
315
335
355
375
Wire

TP Charge

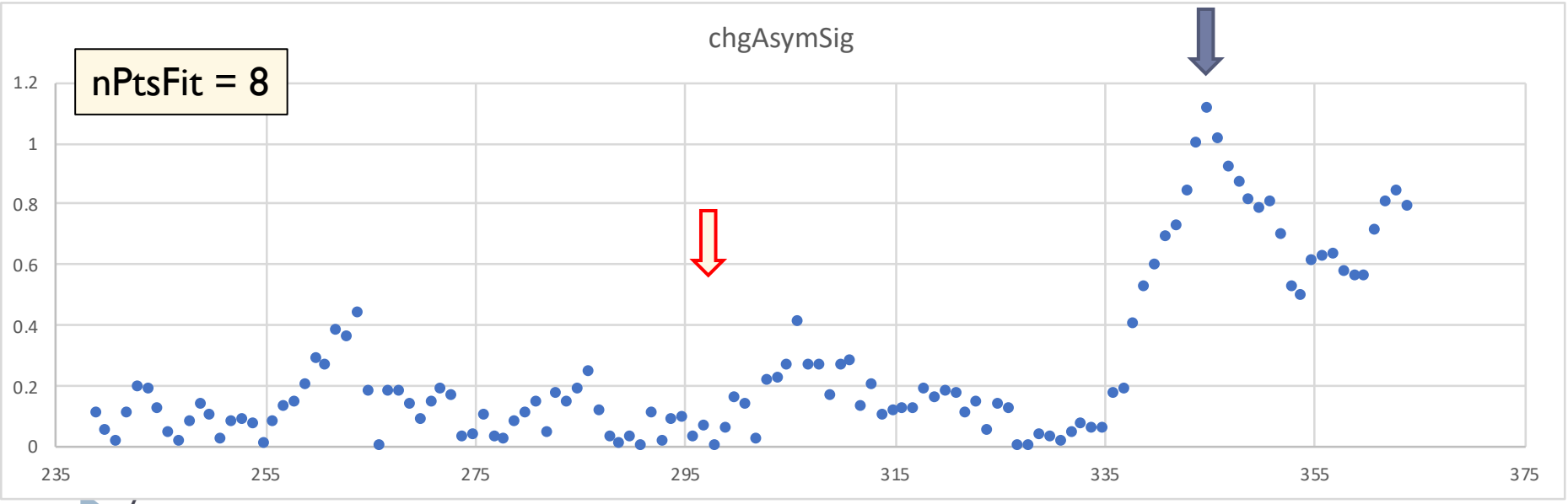

Chg T1
TP Charge
800
700
600
500
400
300
200
100
0
230
250
270
290
310
330
350
370
390
Wire

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

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

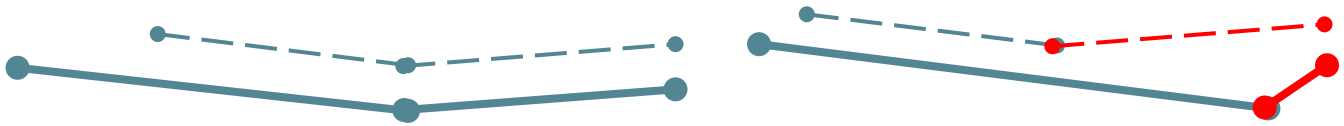


nPtsFit = 8



Observations

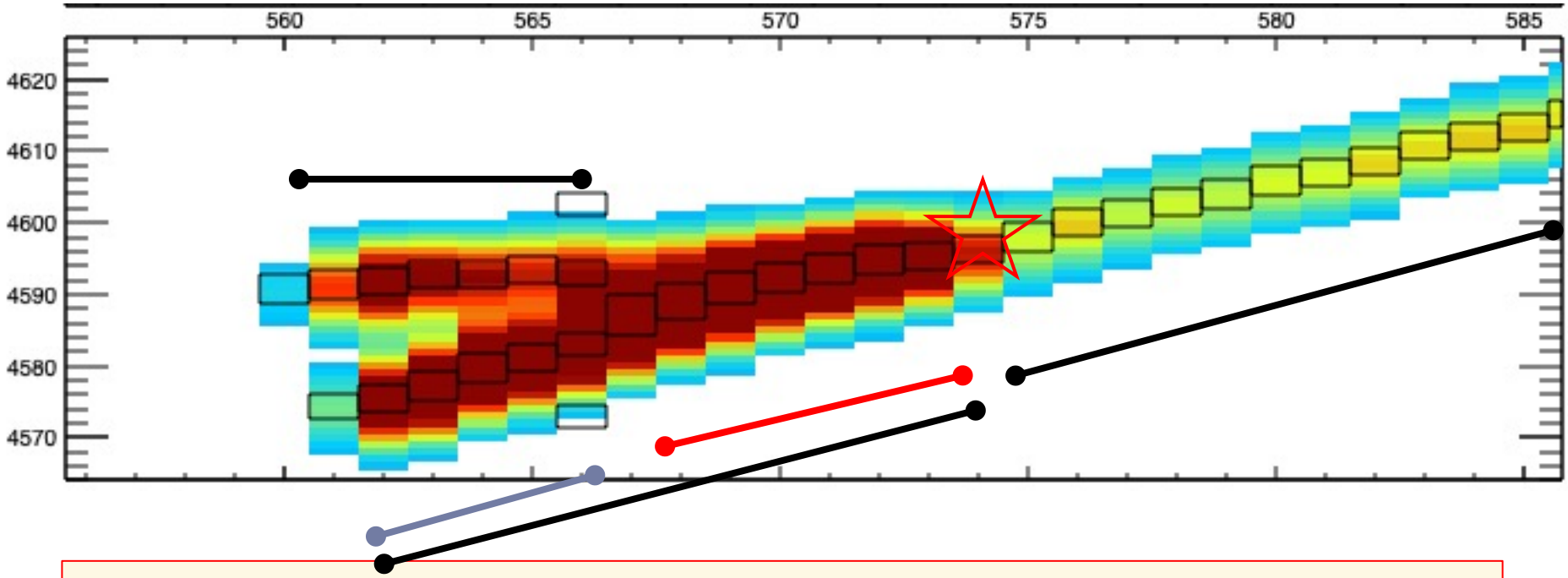
- The fit angle error using nPtsFit = 3 in a single plane is inadequate for small-angle kinks
  - Equivalent nPtsFit for human-eye kink detection using the event display is 10+
- Using a simple stop-tracking-when-above-threshold cut gets the kink point wrong for small-angle kinks
  - Need to wait until 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 setting nPtsFit to ~8
  - fcl configuration
- Added a new kink significance variable to the TrajectoryPoint struct, KinkSig, constructed using angle and charge asymmetry
  - Kink angle significance * charge asymmetry
- Significant changes to the kink finding algorithm
  - KinkCuts fcl configuration vector change
    - See backup slides - TBD
- 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 if any of the TPs in the overlapping region (red bar) are 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. Re-fit the vertex position using TPs outside the overlap region). Don't use overlapping TPs when calculating dE/dx in 3D.

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

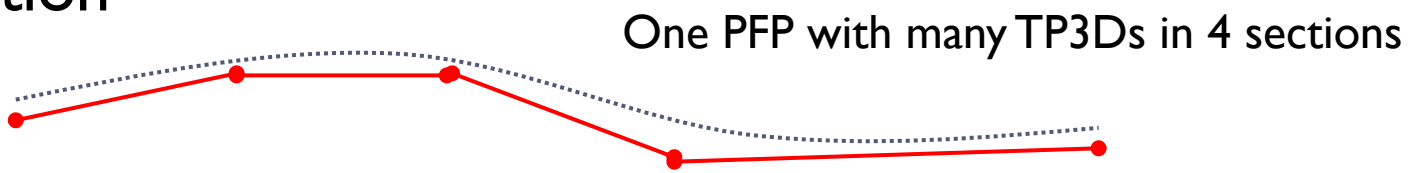
Can't get errors if TDecompSVD solve() is used

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



- Associations
  - 2D
    - Trajectory → TPs and 2D vertices
  - 3D
    - 3D vertex → 2D vertices
    - PFP → 3D-matched trajectories reconstructed in 2D (first phase)
    - PFP owns TP3Ds each with a 2D TP assn (second phase)

PFP Construction Updates

- Optionally use SpacePoints to match trajectories in 3D
  - Assns: SpacePoint → 3 Hits → 3 TPs → 3 trajectories
  - Much faster than TrajCluster 3D matching (~6x)
  - Not used for PFP construction - not high efficiency
- Improved the algorithm for associating TPs to PFPs
  - Consider overlapping 2D trajectories (ref Slide 10)
  - Require compatible 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

- Uses the metrics Efficiency, Purity and Efficiency * Purity (EP)
  - Efficiency = Completeness
  - Evaluated for each MCParticle when there are > 2 MC-matched hits in each TPC and each plane
  - A hit is considered MC-matched if the IDE energyFrac > 0.5
  - The cluster having the most MC-matched hits to a MCParticle is matched to it
    - Note that low-purity clusters may be matched to multiple MCParticles
- Metrics calculated for each particle type and averaged
- All MCParticles have equal weight → short ptcls have fewer hits and therefore higher weight
  - Missing correct MC-matched hits (or adding incorrect hits) has a more significant effect on EP on clusters that have fewer hits
- Option to ignore user-selected PDG codes, e.g. electrons
- Code debugging just completed…

Monitoring Performance

- Use 350 ProtoDUNE MCC12 1 GeV proton events
  - *PDSPProd2_protoDUNE_sp_reco_35ms_sce_datadriven…*
  - Use special TrajCluster mode to only reconstruct hits that are MC-matched to single particle origin = kSingleParticle = 4
    - Statistics: ~400 muons, ~200 pions, ~530 protons
  - Sample size is small but is enough to measure 1% differences
- Compare trajcluster 2D reconstruction with pandora
  - Trajcluster clusters reference a refined trajcluster hit collection

```
physics.analyzers.clusterana.HitModuleLabel: "trajcluster"
physics.analyzers.clusterana.ClusterModuleLabel: "trajcluster"
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
```

  - Pandora clusters reference the hitpdune hit collection

```
physics.analyzers.clusterana.HitModuleLabel: "hitpdune"
physics.analyzers.clusterana.ClusterModuleLabel: "pandora"
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
```

Very Preliminary Results

```
ClusterAnaV2 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      0.975   0.938   0.914
   Pi      0.905   0.828   0.753
   P       0.900   0.796   0.727
Averages for all selected truth particles
 Ave EP 0.790 Ave Eff 0.925 Ave Pur 0.846
 Cnts Mu 387 Pi 213 P  632
```

Trajcluster has significantly higher EP with lower Efficiency but much higher Purity. Is this correct?

```
ClusterAnaV2 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      0.987   0.443   0.436
   Pi      0.952   0.710   0.675
   P       0.961   0.583   0.565
Averages for all selected truth particles
 Ave EP 0.539 Ave Eff 0.969 Ave Pur 0.556
 Cnts Mu 401 Pi 203 P  529
```

Questions:
Are pandora clusters final or interim data products?
Are Cnts different due to lower pandora Purity or is there a bug?

Summary

- Using 3 hits in a 2D kink fit is not enough make a good kink decision
- Revised the 2D kink finder to use error weighted angle difference and charge asymmetry
  - This 2D information is used to identify kinks in 3D
  - Breaking change to fcl configuration – sort of
- 2D vertex fitting code re-written to give more reliable errors
- 3D reconstruction improvements
  - Significant CPU reduction using SpacePointSolver SpacePoints for pattern recognition - not reconstruction
  - Use dE/dx when constructing PFParticles
- Developed a new ana module, ClusterAnaV2, to assess relative performance of different reconstruction modules in 2D
  - Code just completed
  - Plan to add tools to help trace reconstruction failures ala TrajCluster