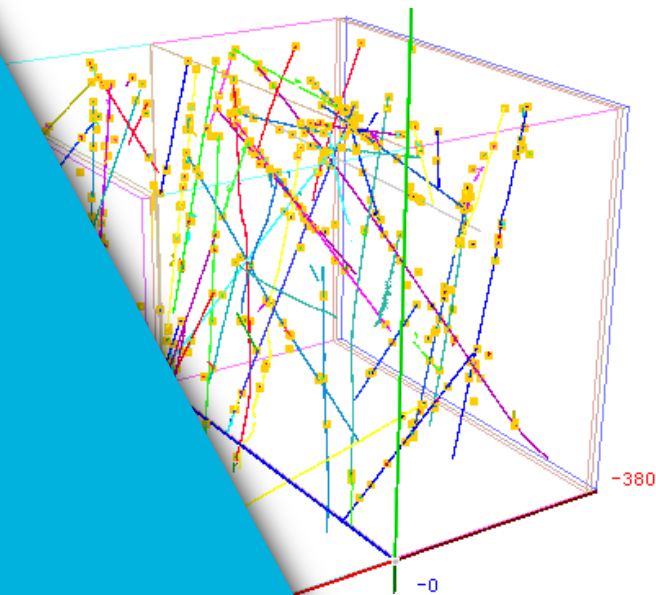# Reconstruction algorithms
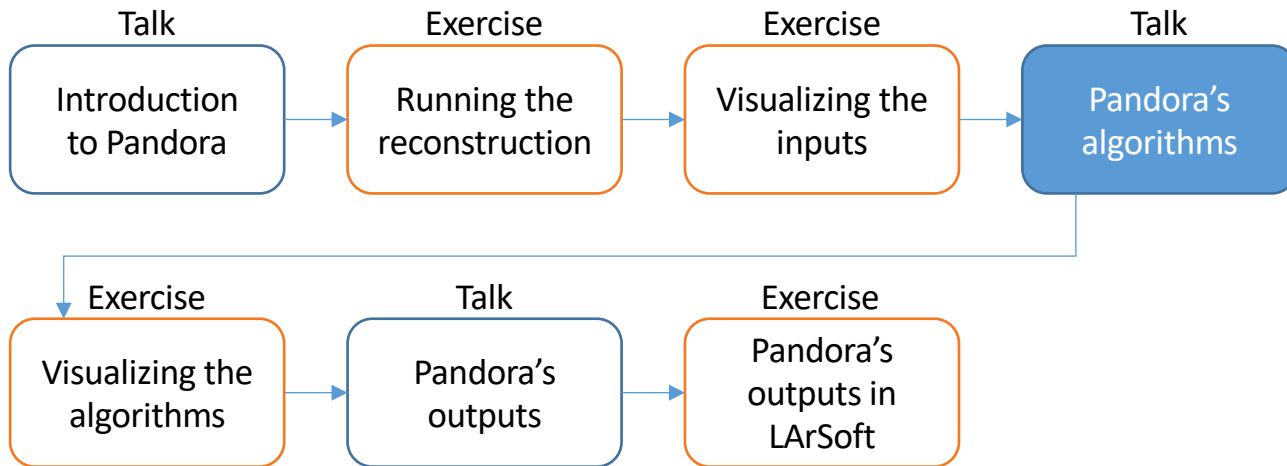
Maria Brigida Brunetti for the Pandora team

09/11/2022   7th UK LArTPC Software and Analysis Workshop
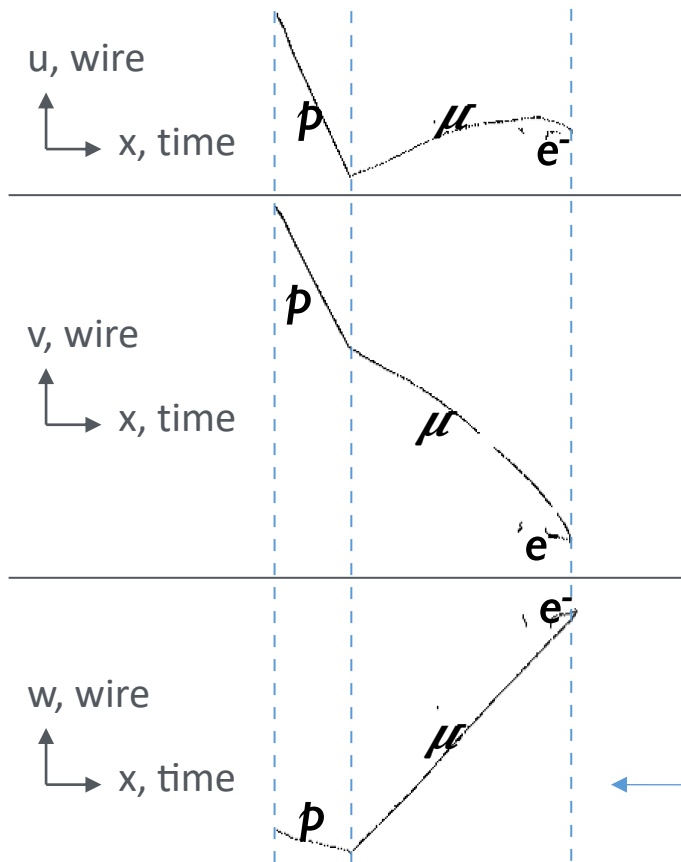
−380

−0

Reconstruction session

Talk	Exercise	Exercise	Talk
Introduction to Pandora	Running the reconstruction	Visualizing the inputs	Pandora's algorithms

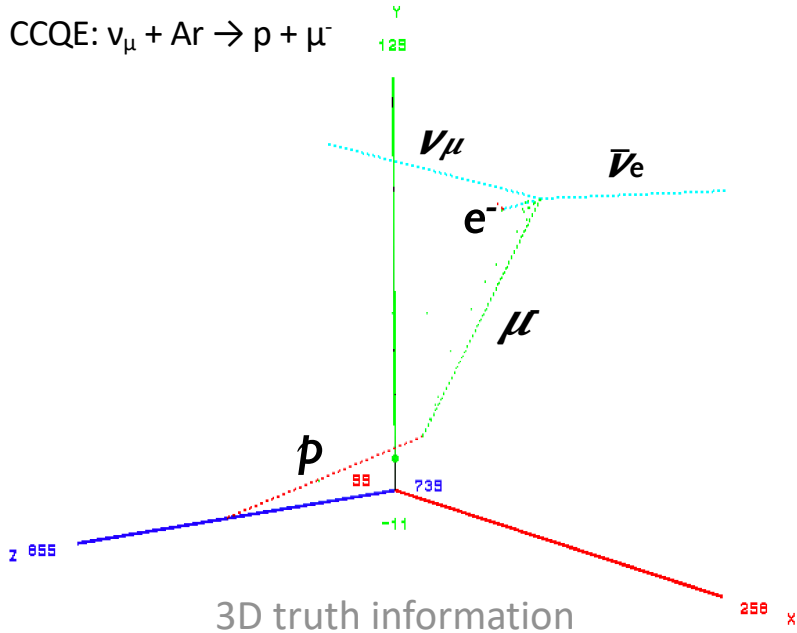|		
Exercise	Talk	Exercise
Visualizing the algorithms	Pandora's outputs	Pandora's outputs in LArSoft

Credit: These slides are based on previous LArSoft workshop slides by John Marshall

Key references: Pandora ProtoDUNE paper
Pandora MicroBooNE paper

Inputs to Pandora

u, wire

x, time

p μ e^-

v, wire

x, time

p μ e^-

w, wire

x, time

e^- μ p

CCQE: ν_μ + Ar → p + μ^-

ν_μ $\bar{\nu}_e$
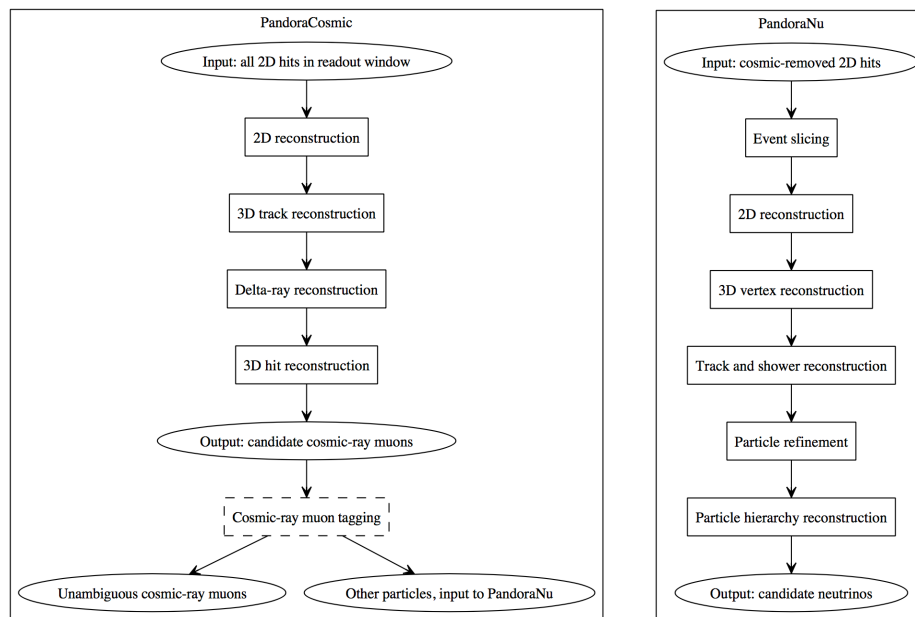
e^-

μ

p

3D truth information

Three 2D representations with common
x coordinate, derived from drift time

Algorithm chains

- Two Pandora algorithm chains created for LArTPC use, with many algs in con
  - PandoraCosmic: strongly track-oriented; showers assumed to be delta rays, added as daughters of primary muons; muon vertices at track high-y coordinate.
  - PandoraNu: finds neutrino interaction vertex and protects all particles emerging from vertex position. Careful treatment to address track/shower tensions.

Initially use a two-pass approach:
Input to PandoraNu excludes hits
from unambiguous cosmic rays.

PandoraCosmic

Input: all 2D hits in readout window

2D reconstruction

3D track reconstruction

Delta-ray reconstruction

3D hit reconstruction

Output: candidate cosmic-ray muons

Cosmic-ray muon tagging

Unambiguous cosmic-ray muons   Other particles, input to PandoraNu

PandoraNu

Input: cosmic-removed 2D hits

Event slicing

2D reconstruction

3D vertex reconstruction

Track and shower reconstruction

Particle refinement

Particle hierarchy reconstruction

Output: candidate neutrinos

Cosmic-Ray Muon Reconstruction - 2D

- For each plane, produce list of 2D clusters that represent continuous, unambiguous lines of hits
- Clusters refined by series of 15 cluster-merging and cluster-splitting algs that use topological info.

Example: Crossing cosmic-ray muons

Simulated unresponsive channels →

w, wire position

x, drift position
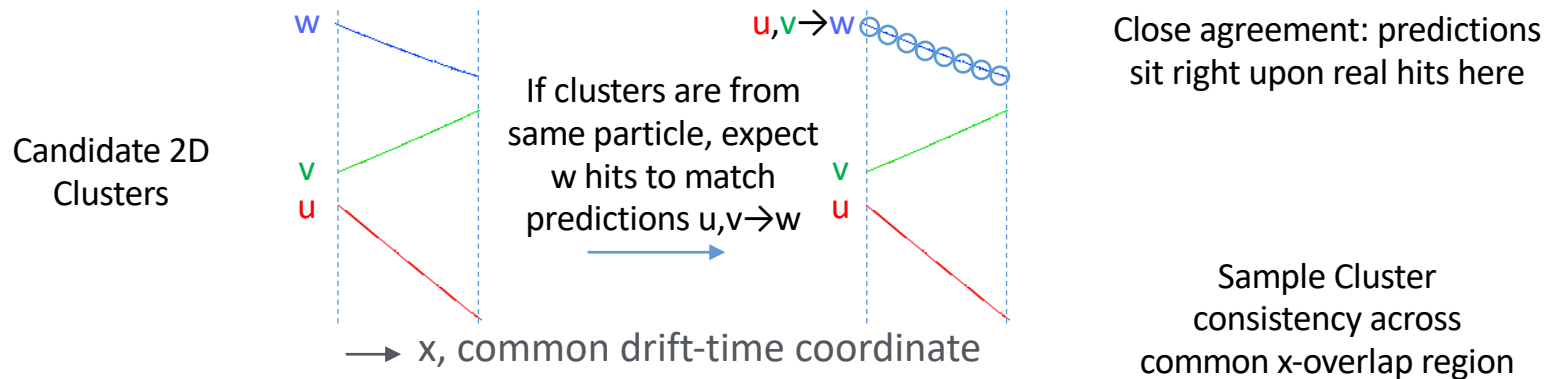
(a)   (b)
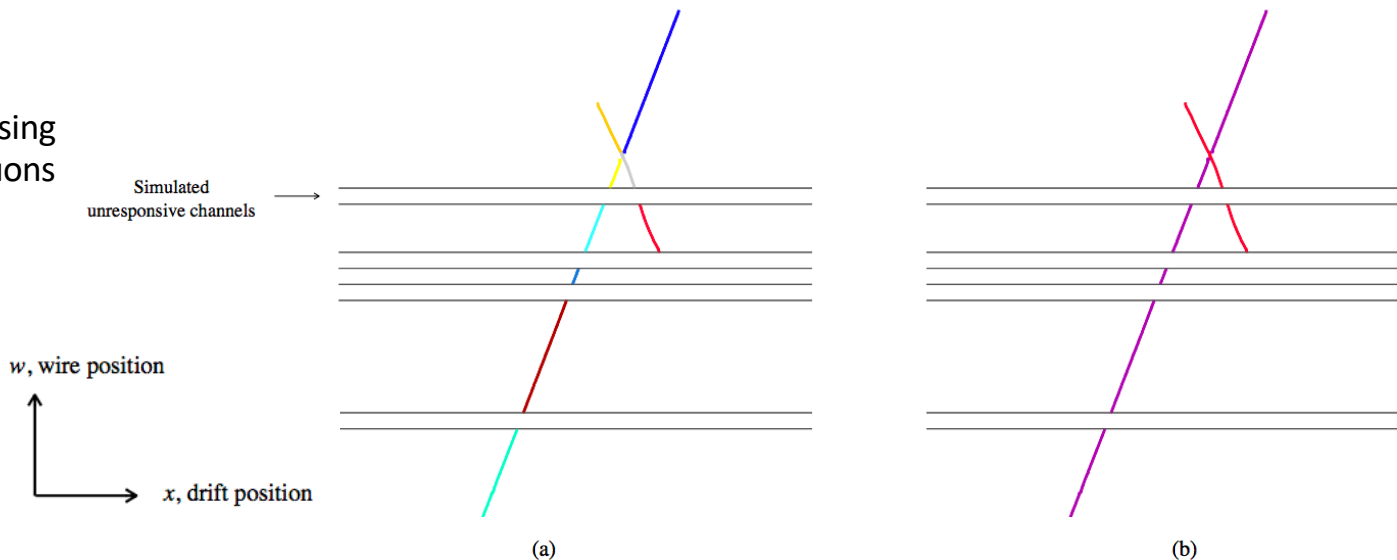
Topological Association - 2D

- Cluster-merging algorithms identify associations between multiple 2D clusters and look to grow the clusters to improve completeness, without compromising purity.
    - The challenge for the algorithms is to make cluster-merging decisions in the context of the entire event, rather than just considering individual pairs of clusters in isolation.
    - Typically need to provide a definition of association (for a given pair of clusters), then navigate forwards and backwards to identify chains of associated clusters that can be safely merged.
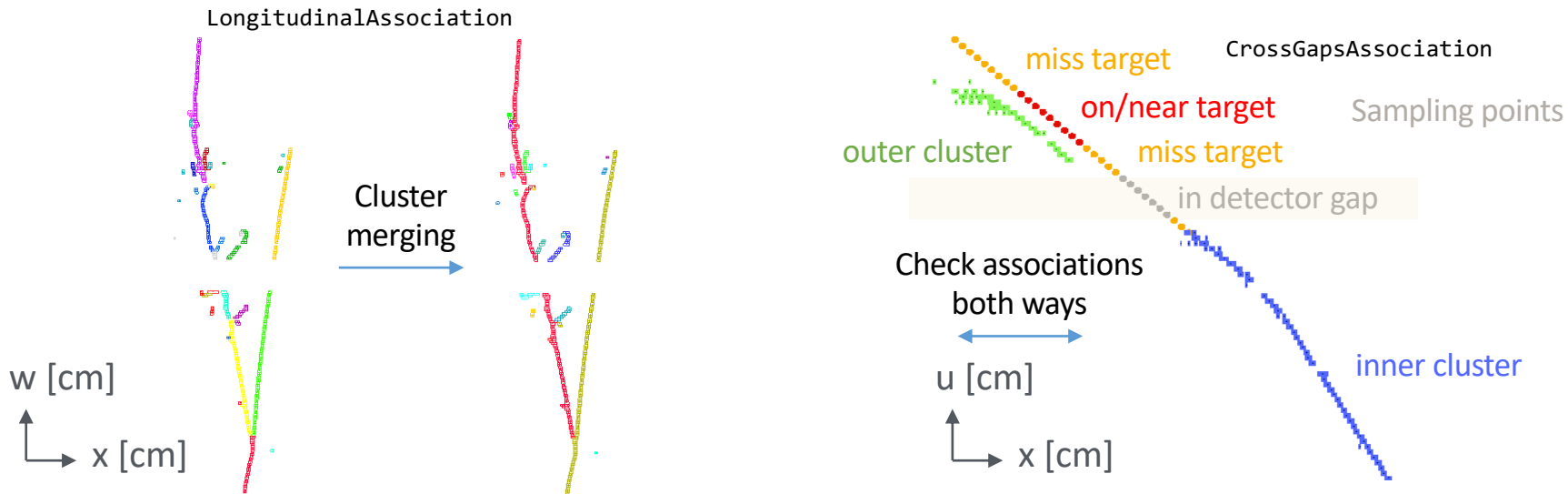
Track Pattern Recognition - 3D

- Our original input was 3x2D images of charged particles in the detector.

- Should now have reconstructed three separate 2D clusters for each particle:

  - Compare 2D clusters from u, v, w planes to find the clusters representing same particle.
  - Exploit common drift-time coordinate and our understanding of wire plane geometry.
  - At given x, compare predictions {u,v→w; v,w→u; w,u→v} with cluster positions, calculating χ^2

w

u,v→w

Close agreement: predictions sit right upon real hits here

Candidate 2D Clusters

If clusters are from same particle, expect w hits to match predictions u,v→w

v

u

v

u

Sample Cluster consistency across common x-overlap region

→ x, common drift-time coordinate

Store all results in a "tensor", recording x-overlap span, no. of sampling points, no. of "matched" sampling points and χ2. Documents all 2D cluster-matching ambiguities.
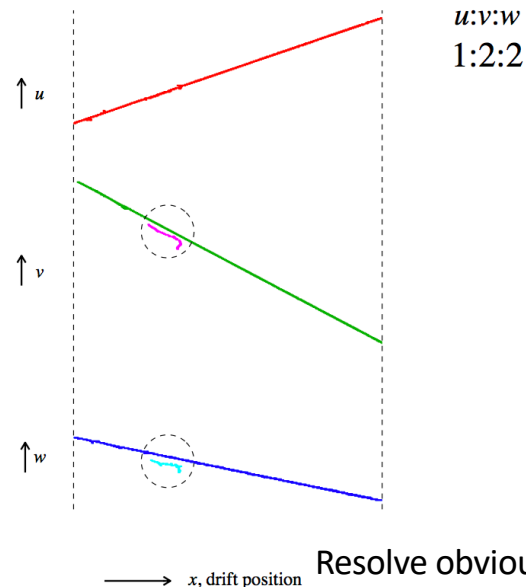
Track Pattern Recognition - 3D

Tensor stores overlap details for trios of 2D clusters. Tools make 2D reco changes to resolve any ambiguities. If a tool makes a change (e.g. splits a cluster), all tools run again.

`ClearTracksTool`

$u:v:w$
1:1:1

No. of associated 2D Clusters u:v: w
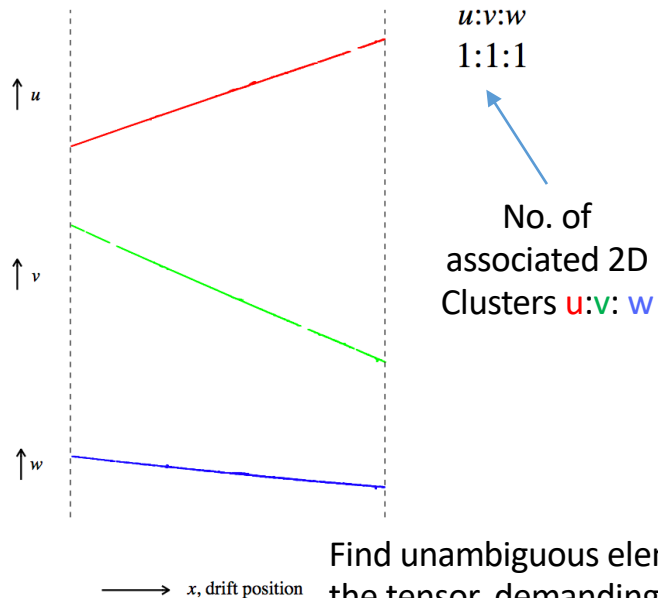
Find unambiguous elements in the tensor, demanding that the common x-overlap is 90% of the x-span for all three clusters.

`LongTracksTool`

$u:v:w$
1:2:2
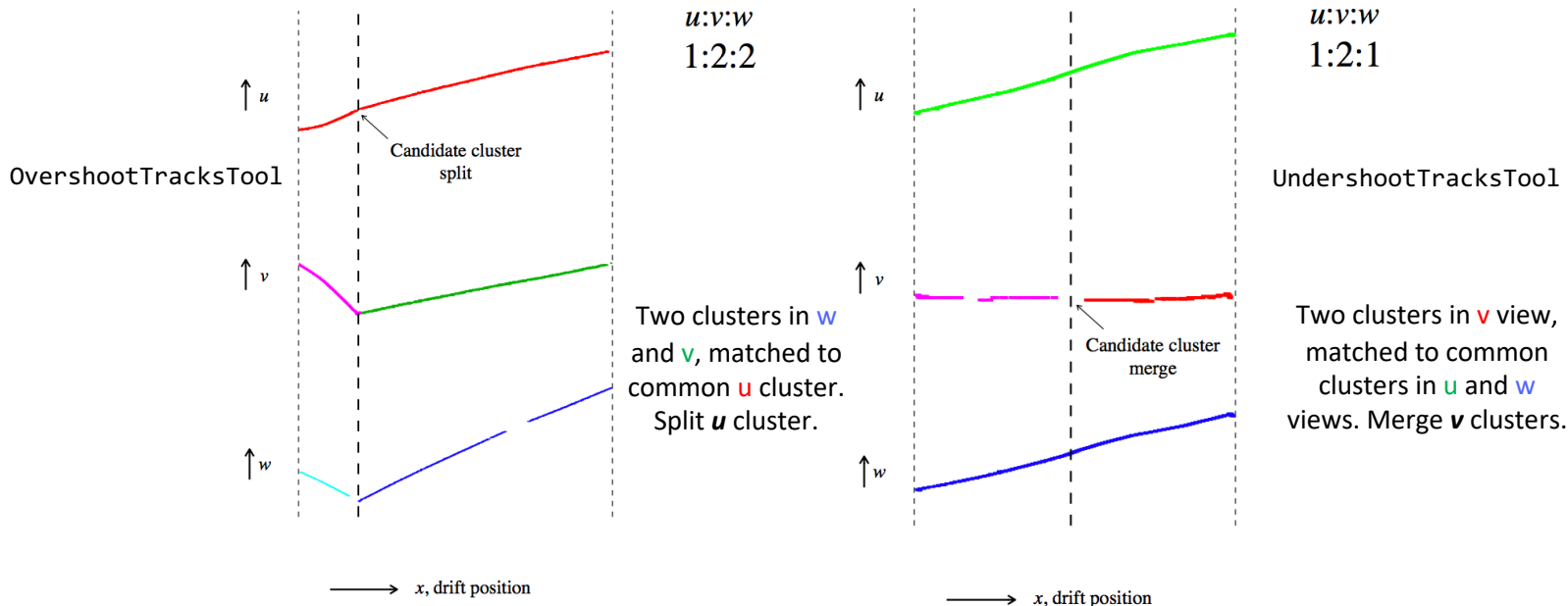
Resolve obvious ambiguities: clusters are matched in multiple configurations, but one tensor element is much "better" than others.

# Track Pattern Recognition - 3D

$u{:}v{:}w$
1:2:2

OvershootTracksTool

Candidate cluster split

Two clusters in w and v, matched to common u cluster. Split **u** cluster.

$u{:}v{:}w$
1:2:1

UndershootTracksTool

Candidate cluster merge

Two clusters in v view, matched to common clusters in u and w views. Merge **v** clusters.

x, drift position

x, drift position

- Use all connected clusters to assess whether this is a true 3D kink topology.
- Modify 2D clusters as appropriate (i.e. merge or split) and update cluster-matching tensor.
- Initial ClearTracks tool then able to identify unambiguous groupings of clusters and form particles.
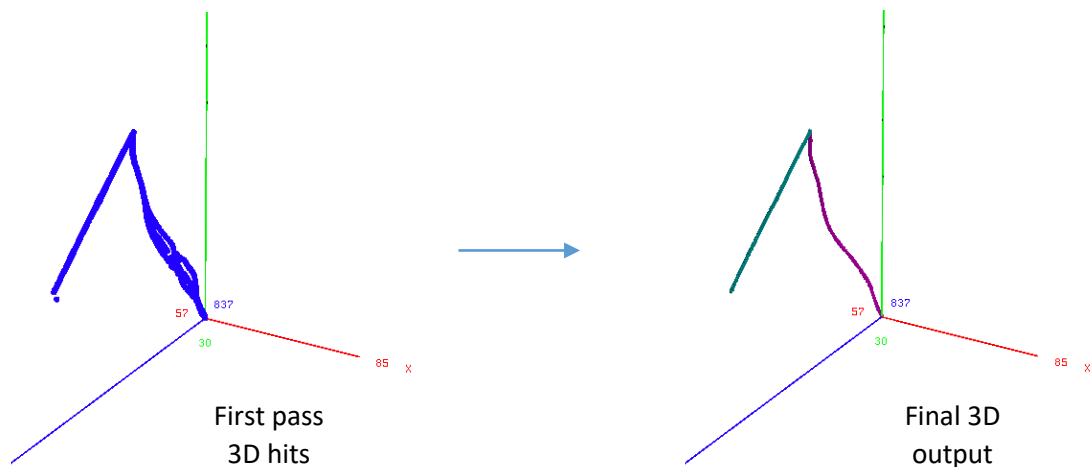
Delta-Ray Reconstruction - 2D, 3D

- Assume any 2D clusters not in a track particle are from delta-ray showers:
  - Simple proximity-based reclustering of hits, then topological association algs.
  - Delta-ray clusters matched between views, creating delta-ray shower particles.
  - Parent muon particles identified and delta-ray particles added as children.
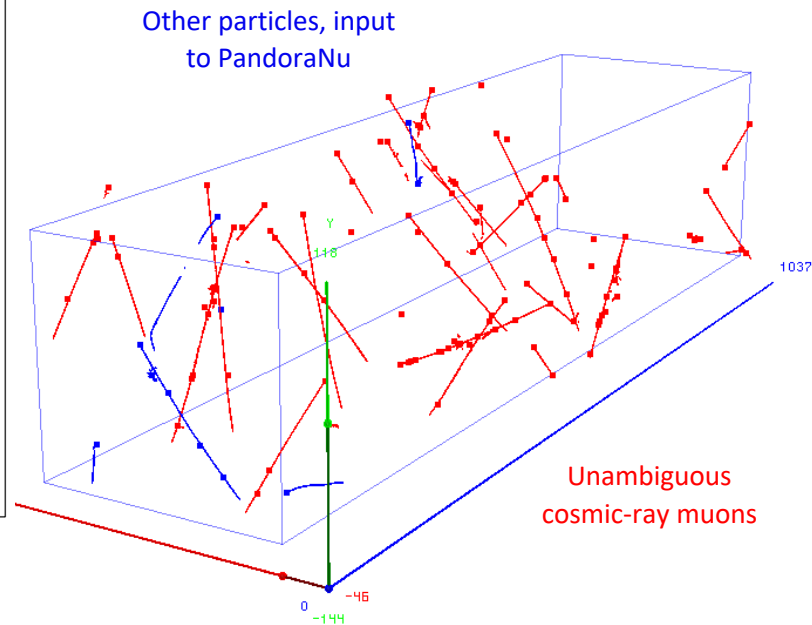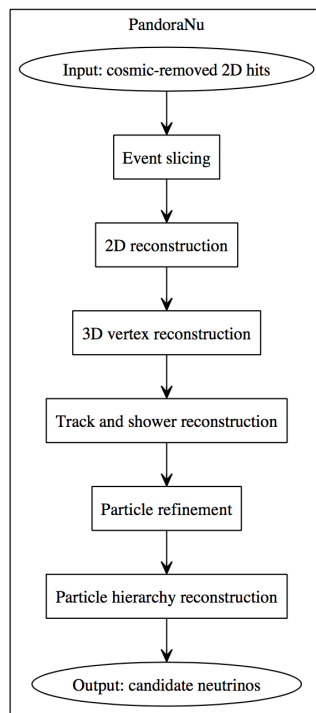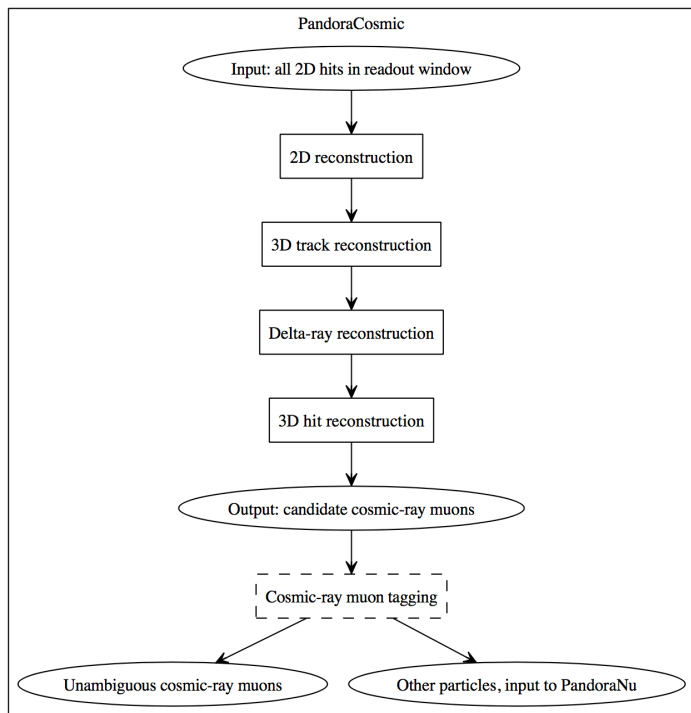
Child delta ray (shower) particles

Parent muon (track) particle
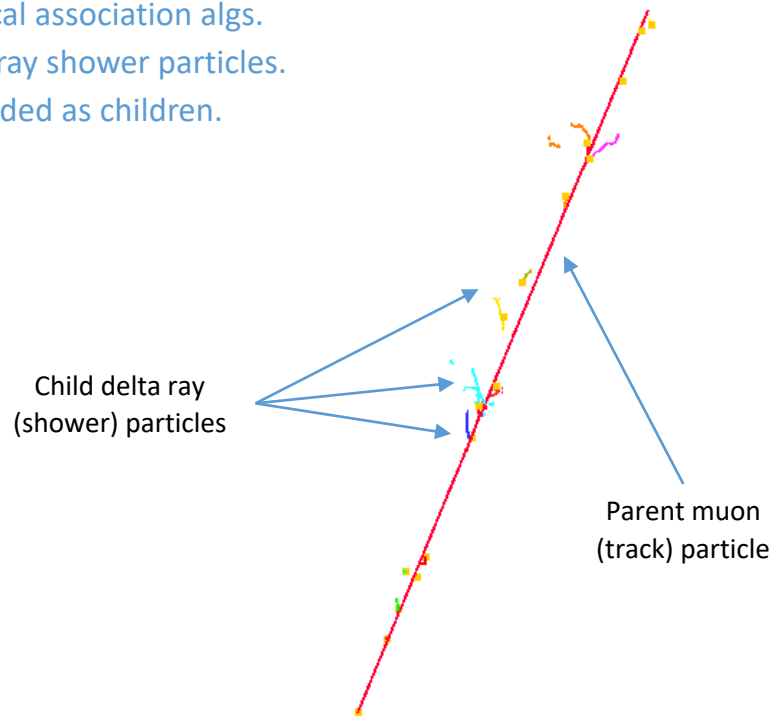
3D Hit/Cluster Reconstruction

- For each 2D Hit, sample clusters in other views at same x, to provide u_{in}, v_{in} and w_{in} values

- Provided u_{in}, v_{in} and w_{in} values don't necessarily correspond to a specific point in 3D space

- Analytic expression to find 3D space point that is most consistent with given u_{in}, v_{in} and w_{in}
    - $\chi^2 = (u_{out} - u_{in})^2 / \sigma_u^2 + (v_{out} - v_{in})^2 / \sigma_v^2 + (w_{out} - w_{in})^2 / \sigma_w^2$
    - Write in terms of unknown y and z, differentiate wrt y, z and solve
   - Can iterate, using fit to current 3D hits (extra terms in χ^2) to produce smooth trajectory

First pass
3D hits

Final 3D
output

PandoraCosmic → PandoraNu

PandoraCosmic

- Input: all 2D hits in readout window
- 2D reconstruction
- 3D track reconstruction
- Delta-ray reconstruction
- 3D hit reconstruction
- Output: candidate cosmic-ray muons
- Cosmic-ray muon tagging
- Unambiguous cosmic-ray muons
- Other particles, input to PandoraNu

PandoraNu

- Input: cosmic-removed 2D hits
- Event slicing
- 2D reconstruction
- 3D vertex reconstruction
- Track and shower reconstruction
- Particle refinement
- Particle hierarchy reconstruction
- Output: candidate neutrinos

Other particles, input to PandoraNu

Unambiguous cosmic-ray muons

Neutrino Reconstruction

- Must be able to deal with presence of any cosmic-ray muon remnants.
  - Run fast version of reconstruction, up to 3D hit creation
  - "Slice" 3D hits into separate interactions, processing each slice in isolation.
  - Each slice ⇒ candidate neutrino particle.

- Neutrino pass reuses track-oriented clustering and topological association.
  - Topological association algs must handle rather more complex topologies.
  - Specific effort to reconstruct neutrino interaction vertex.
  - More sophisticated efforts to reconstruct showers.

Neutrino event, amongst cosmic-ray remnants

w [cm]

x [cm]
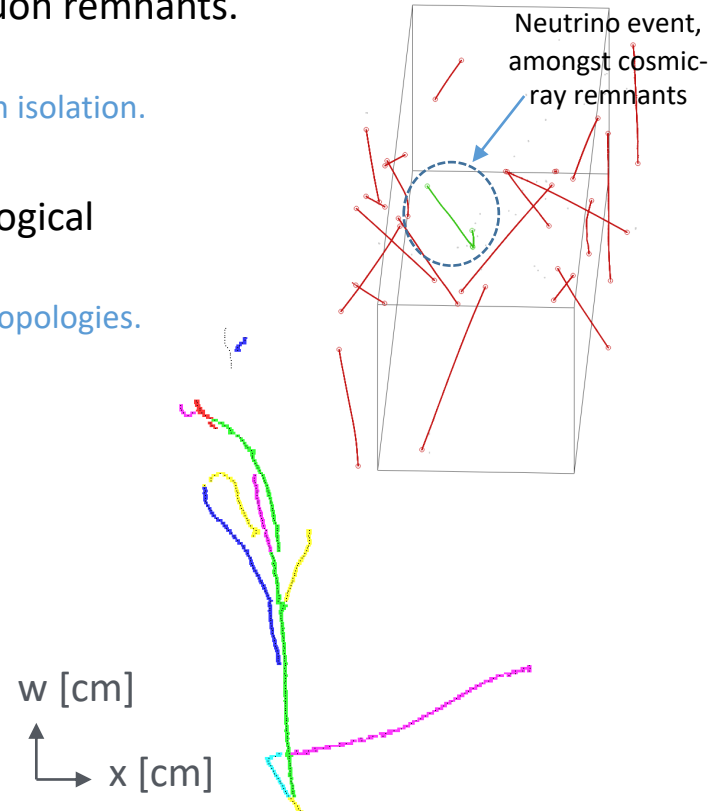
Neutrino Reconstruction

- Search for neutrino interaction vertex:
  - Use pairs of 2D clusters to produce list of possible 3D vertex candidates.
 - Examine candidates, calculate a score for each and select the best.

High ET sum:
⇒ suppress candidate

w [cm]

x [cm]

2D projection of 3D vertex candidate

Vertex candidate A

B

C

D

Selected candidate E

w, wire position

x, drift position

Candidate	S	$S_{energy\ kick}$	$S_{asymmetry}$	$S_{beam\ deweight}$
A	4.9E-07	3.5E-06	1.00	0.14
B	1.3E-02	3.1E-02	0.99	0.42
C	1.1E-03	2.4E-03	0.95	0.46
D	5.7E-10	1.1E-09	1.00	0.52
E	9.0E-01	9.0E-01	1.00	0.99

Scores for labelled candidates, with breakdown into component parts:

- Downstream usage:
  - Split 2D clusters at projected vertex position.
  - Use vertex to protect primary particles when growing showers.
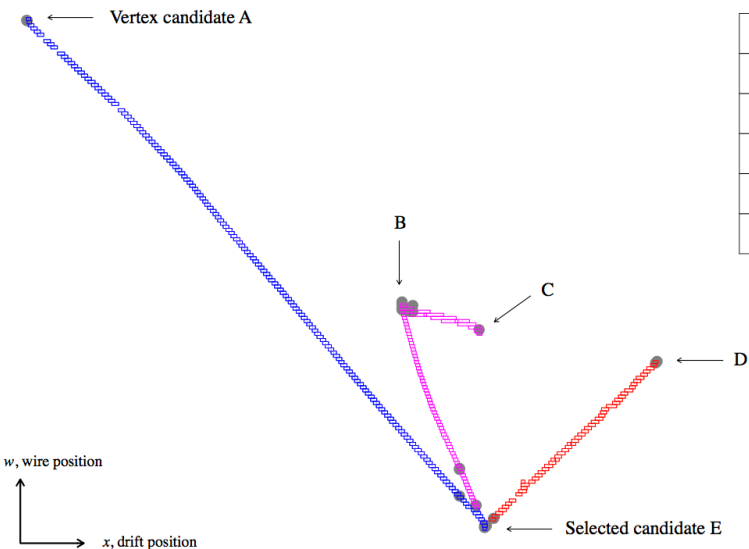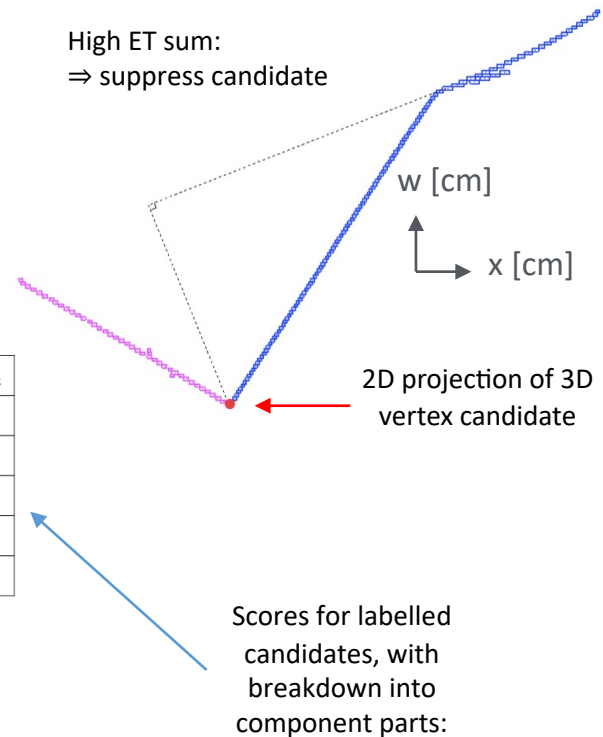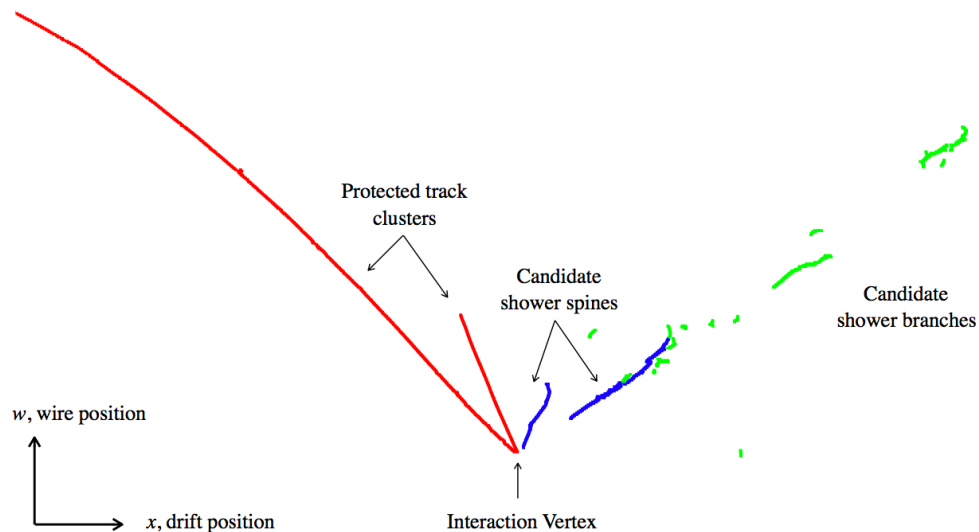
Vertex Reconstruction – 3D

- Interaction vertex is an important feature point in our LArTPC images:
  - Continued development, ever-more sophisticated approaches to finding 3D vertex position
  - Boosted Decision Trees (BDTs) or Support Vector Machines (SVMs) to select best candidate
  - Exploit Convolutional Neural Networks (CNNs) ⇒ Deep Learning talk/tutorial
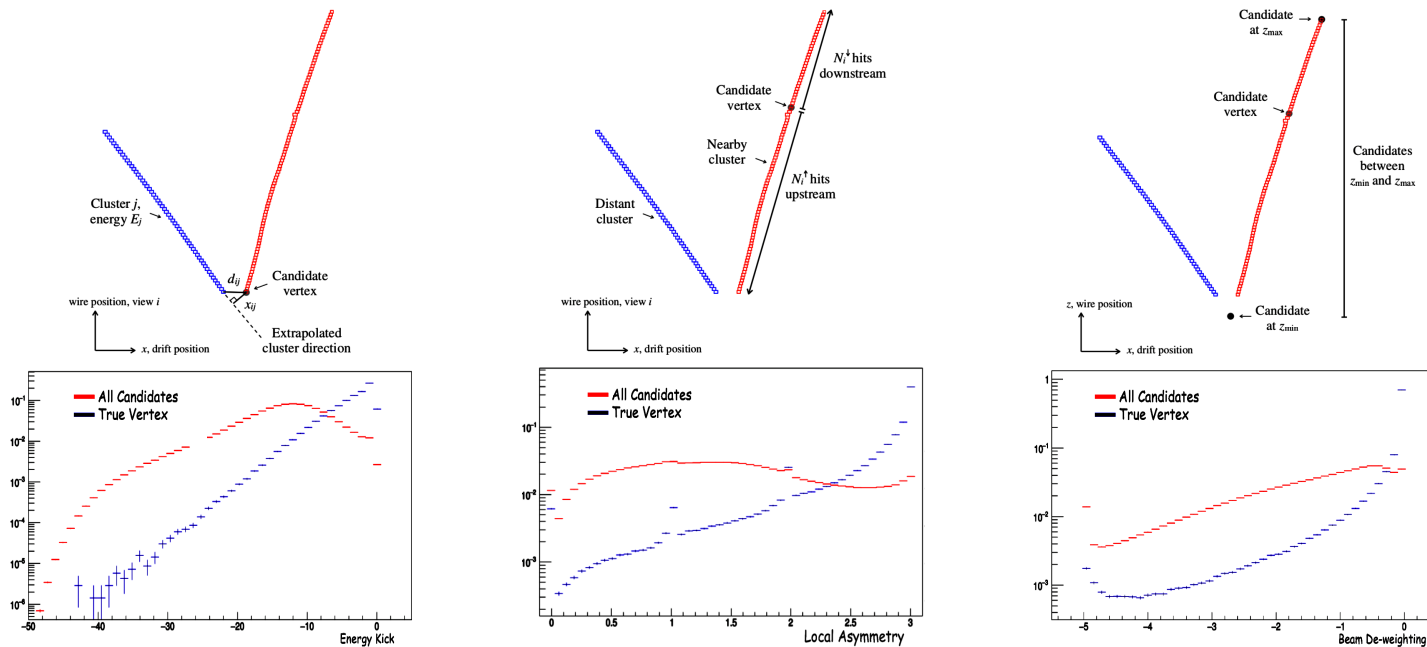
E.g BDT/SVM "features"

Shower Reconstruction - 2D

- Track reconstruction exactly as in PandoraCosmic, but now also attempt to reconstruct primary electromagnetic showers, from electrons and photons:
  - Characterise 2D clusters as track-like or shower-like and use topological properties to identify clusters that might represent shower spines.
  - Add shower-like branch clusters to shower-like spine clusters. Recursively identify branches on the top-level spine candidate, then branches on branches, etc.
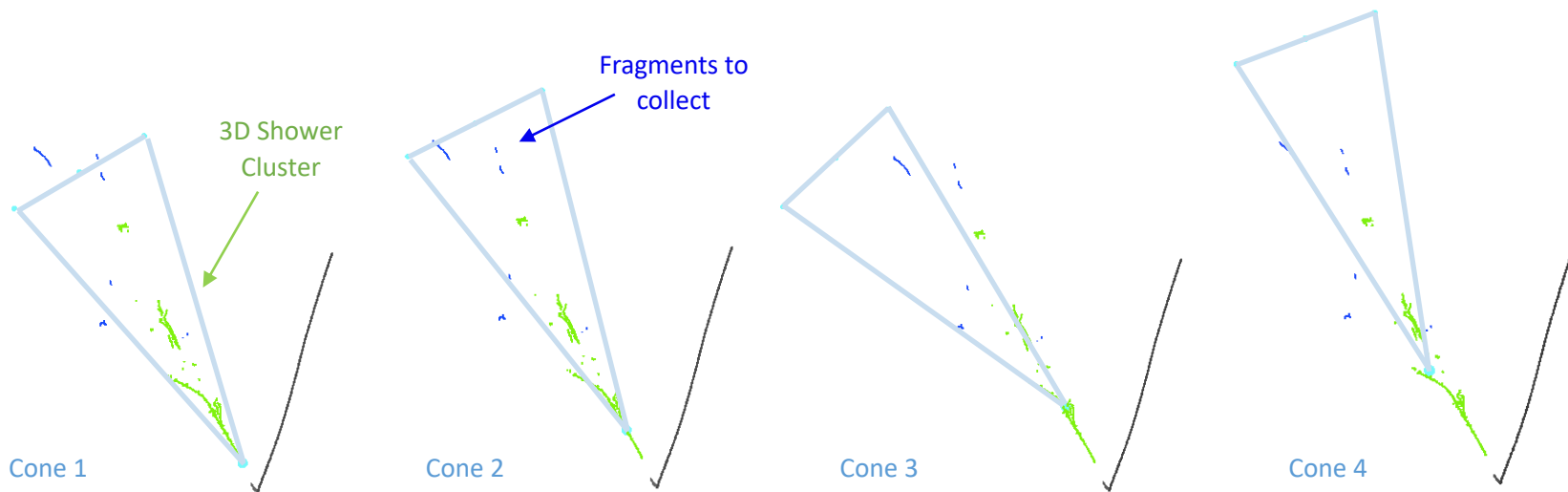
Shower Reconstruction - 3D

- Reuse ideas from track reco to match 2D shower clusters between views:
  - Build a tensor to store cluster overlap and relationship information.
  - Overlap information collected by fitting shower envelope to each 2D cluster.
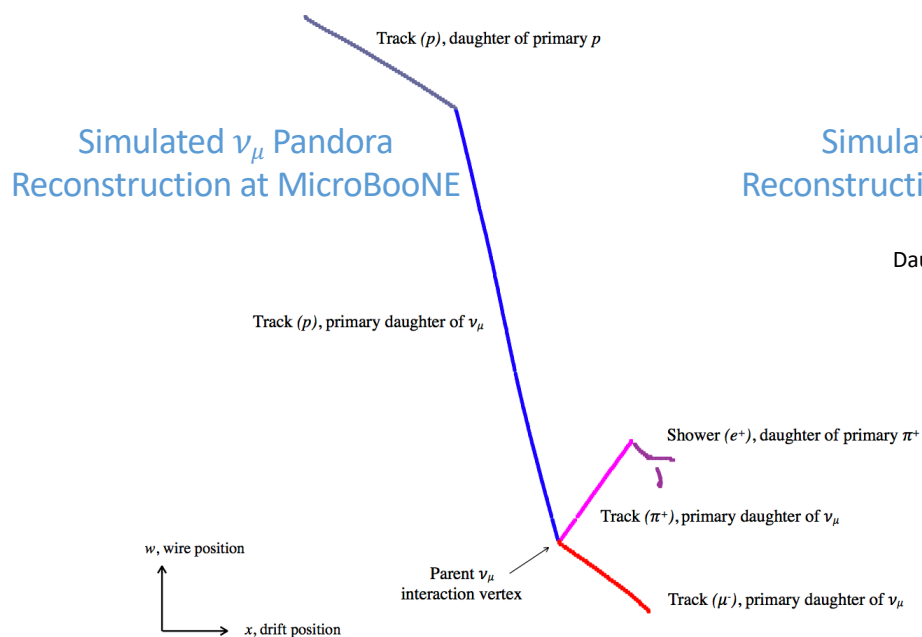 - Shower edges from two clusters used to predict envelope for third cluster.

Fitted shower envelopes for v and w clusters

Predicted shower envelope for u cluster

x, drift position
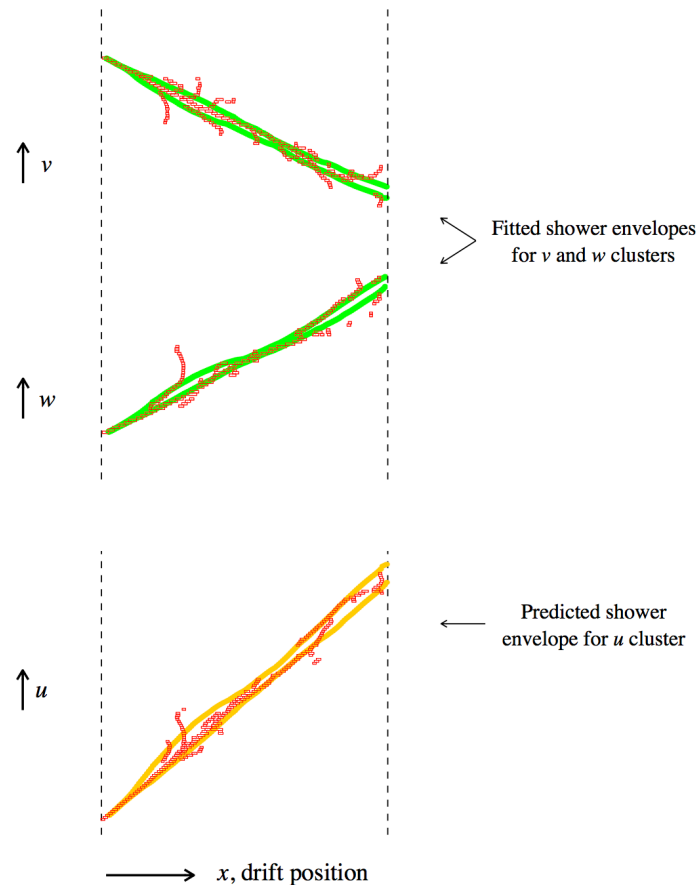
Particle Refinement - 2D, 3D

- Series of algs deal with remnants to improve particle completeness (esp. sparse showers):
  - Pick up small, unassociated clusters bounded by the 2D envelopes of shower-like particles.
  - Use sliding linear fits to 3D shower clusters to define cones for merging small downstream shower particles or picking up additional unassociated clusters.
  - If anything left at end, dissolve clusters and assign hits to nearest shower particles in range.
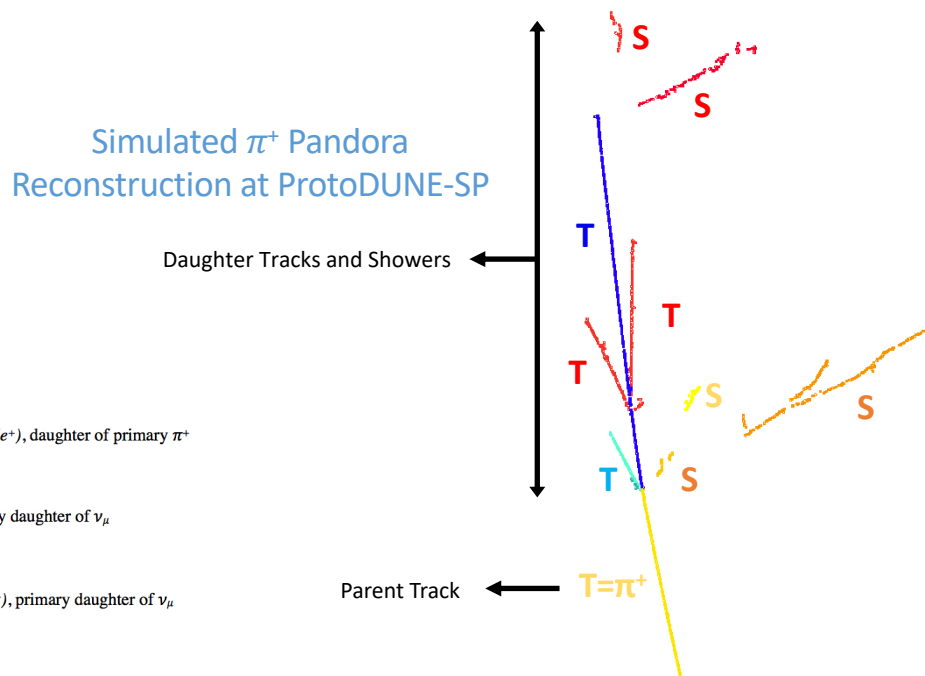

Fragments to collect

3D Shower Cluster

Cone 1   Cone 2   Cone 3   Cone 4

Particle Hierarchy Reconstruction - 3D

- Use 3D clusters to organise particles into a hierarchy, working outwards from interaction vtx:

Simulated ν_μ Pandora
Reconstruction at MicroBooNE

Simulated π^+ Pandora
Reconstruction at ProtoDUNE-SP

Track (p), daughter of primary p

Track (p), primary daughter of ν_μ

Shower (e^+), daughter of primary π^+

Track (π^+), primary daughter of ν_μ

Parent ν_μ interaction vertex

Track (μ^-), primary daughter of ν_μ

w, wire position

x, drift position

Daughter Tracks and Showers

Parent Track

S

S

T

T

T

S

S

T

S

T=π^+

EPJC (2018) 78:82

We will now try visualizing actions of individual algorithms