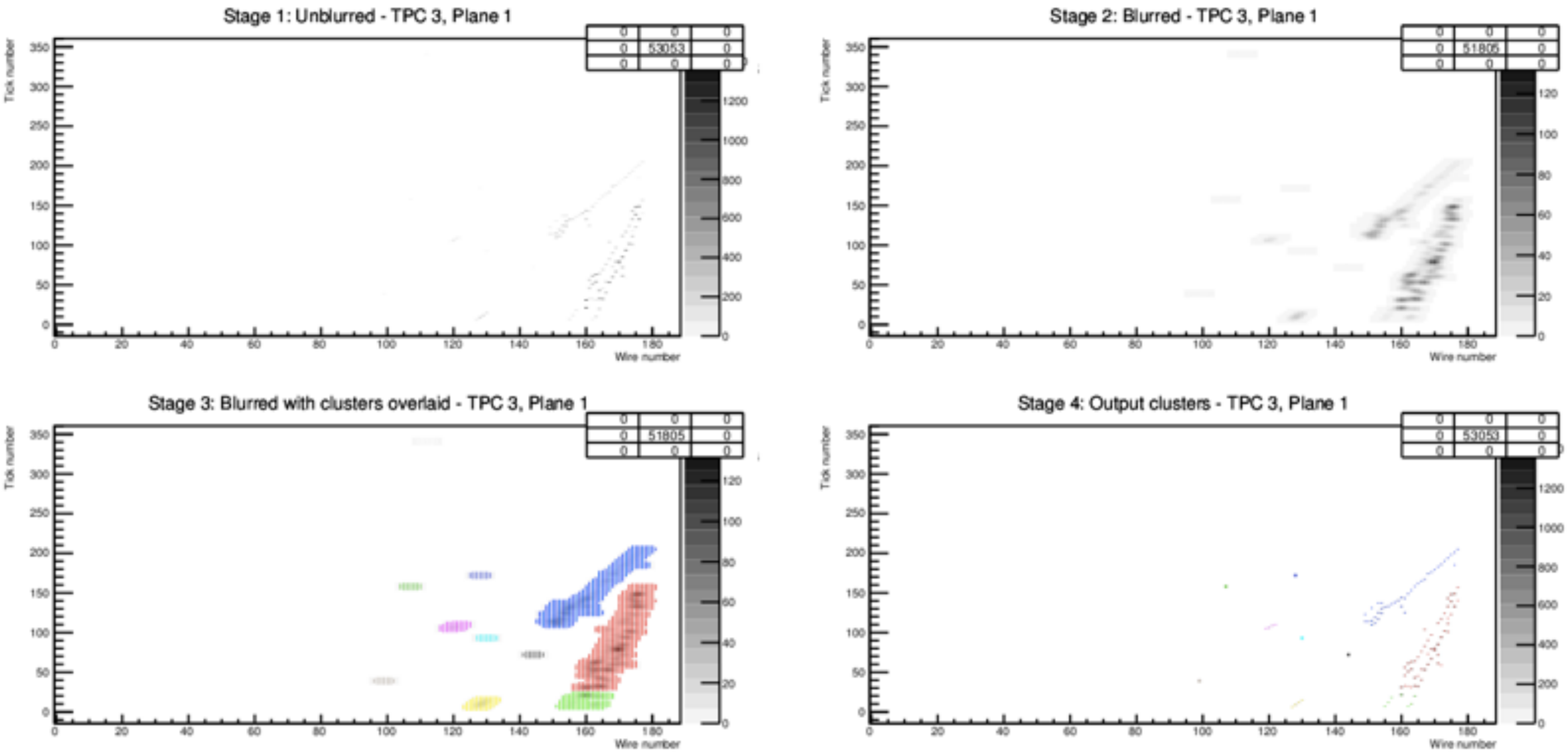
Blurred Clustering: Improved Dynamic Blurring

Mike Wallbank
University of Sheffield

14/7/2015

The Usual Slide

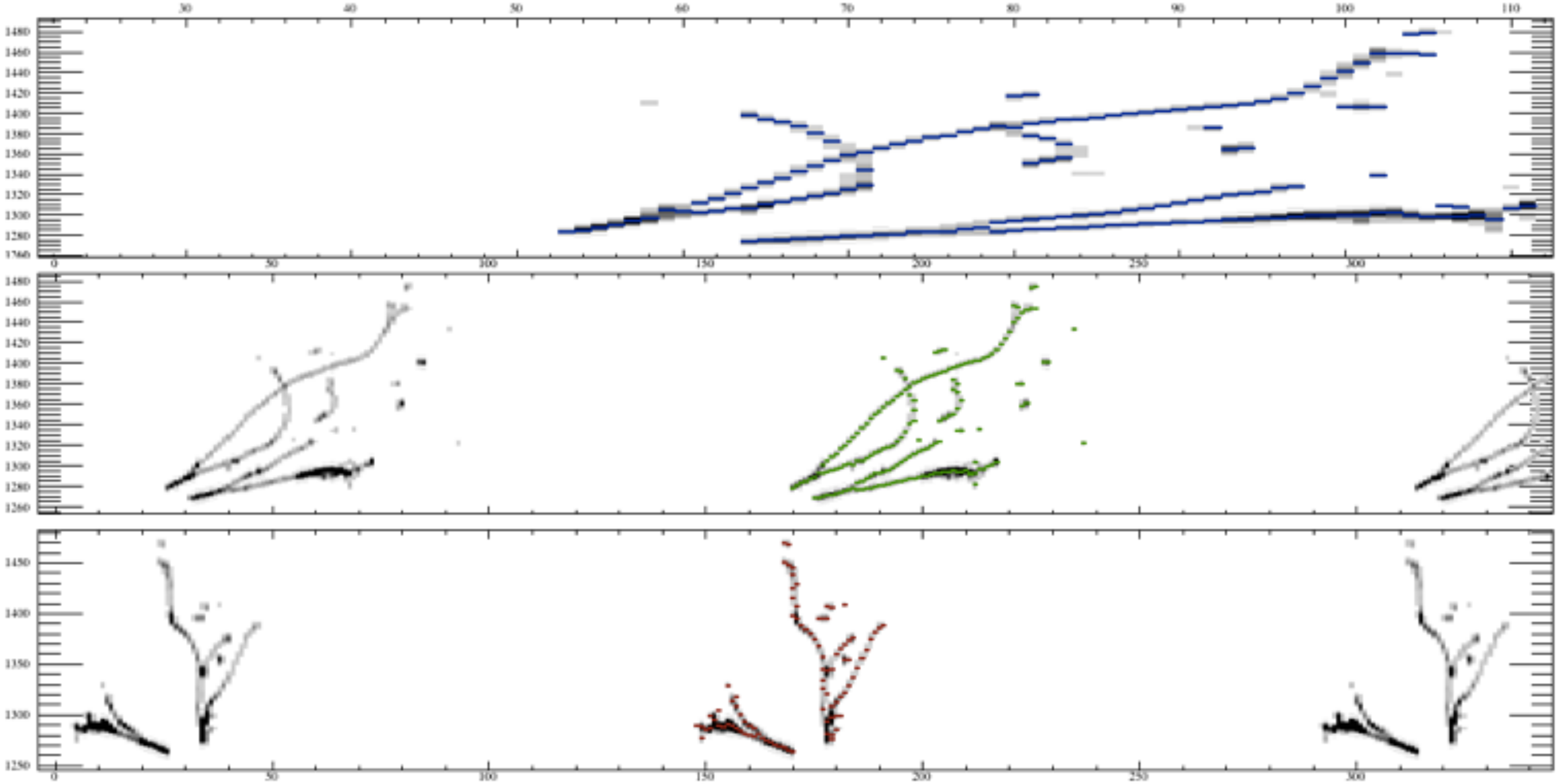
- Clustering technique which uses a Gaussian smearing to produce more full and complete clusters.



- Blurs the hit map and then clusters neighbouring hits before removing the ‘fake hits’.

Dynamic Blurring

- Last update (24 June, https://indico.fnal.gov/conferenceDisplay.py?confId=10081), I had identified a major problem with the blurring method:



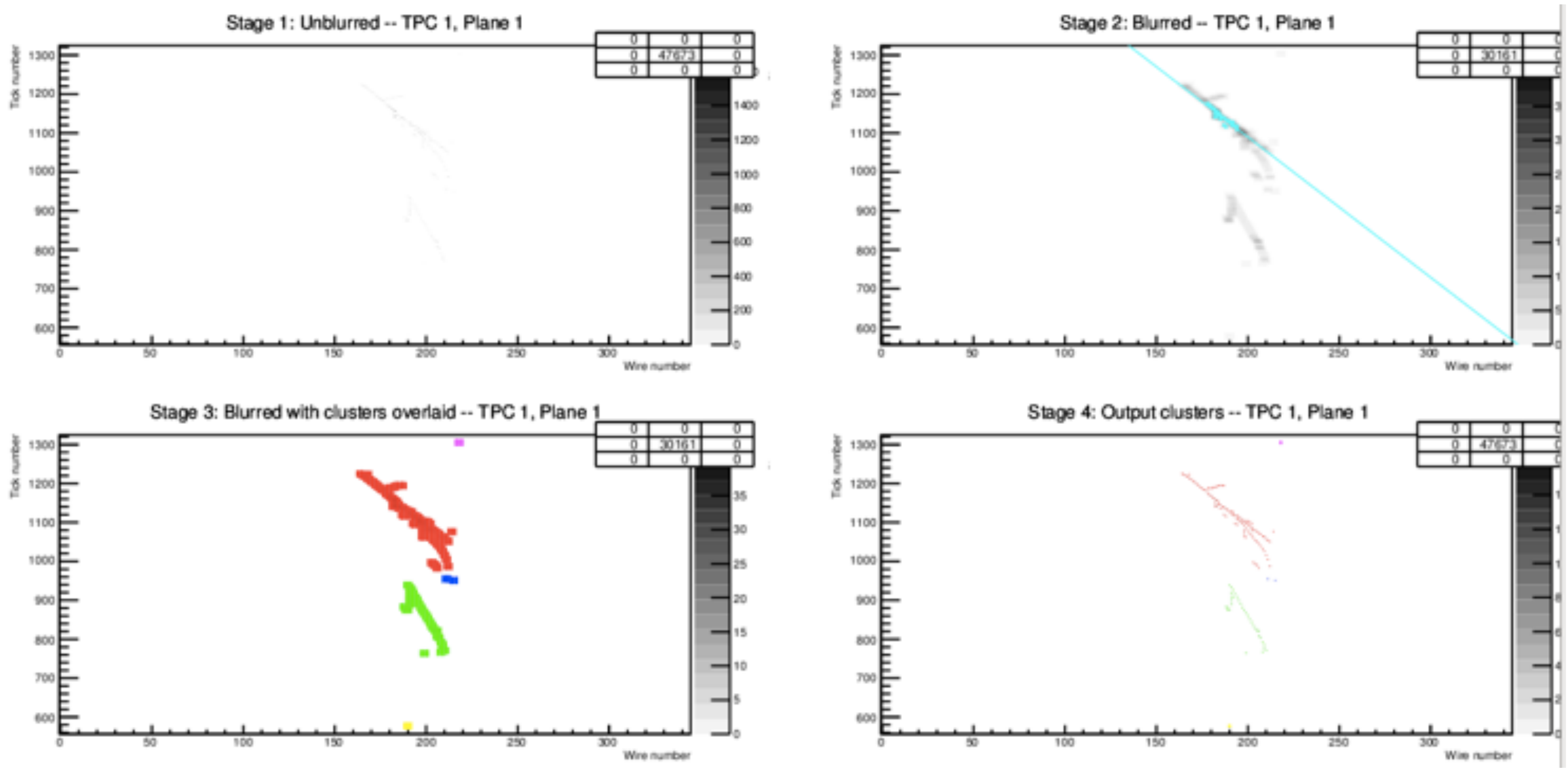
Tracks tend to travel in the similar direction and so are easily blurred and clustered together as one object

Dynamic Blurring

- I started investigating a possible solution to this problem: Dynamic Blurring.
- Idea:
  - Get some idea of the direction a track/shower is going in (in the plane/wire space) before blurring or clustering
  - Use this information to allocate the most appropriate blurring radii so the blurring can follow the particle as closely as possible
  - Clustering then proceeds over a smaller distance since the blurring encompasses the track/shower
- Assumes tracks are vaguely parallel (good assumption I think!)

What I Showed Last Time…

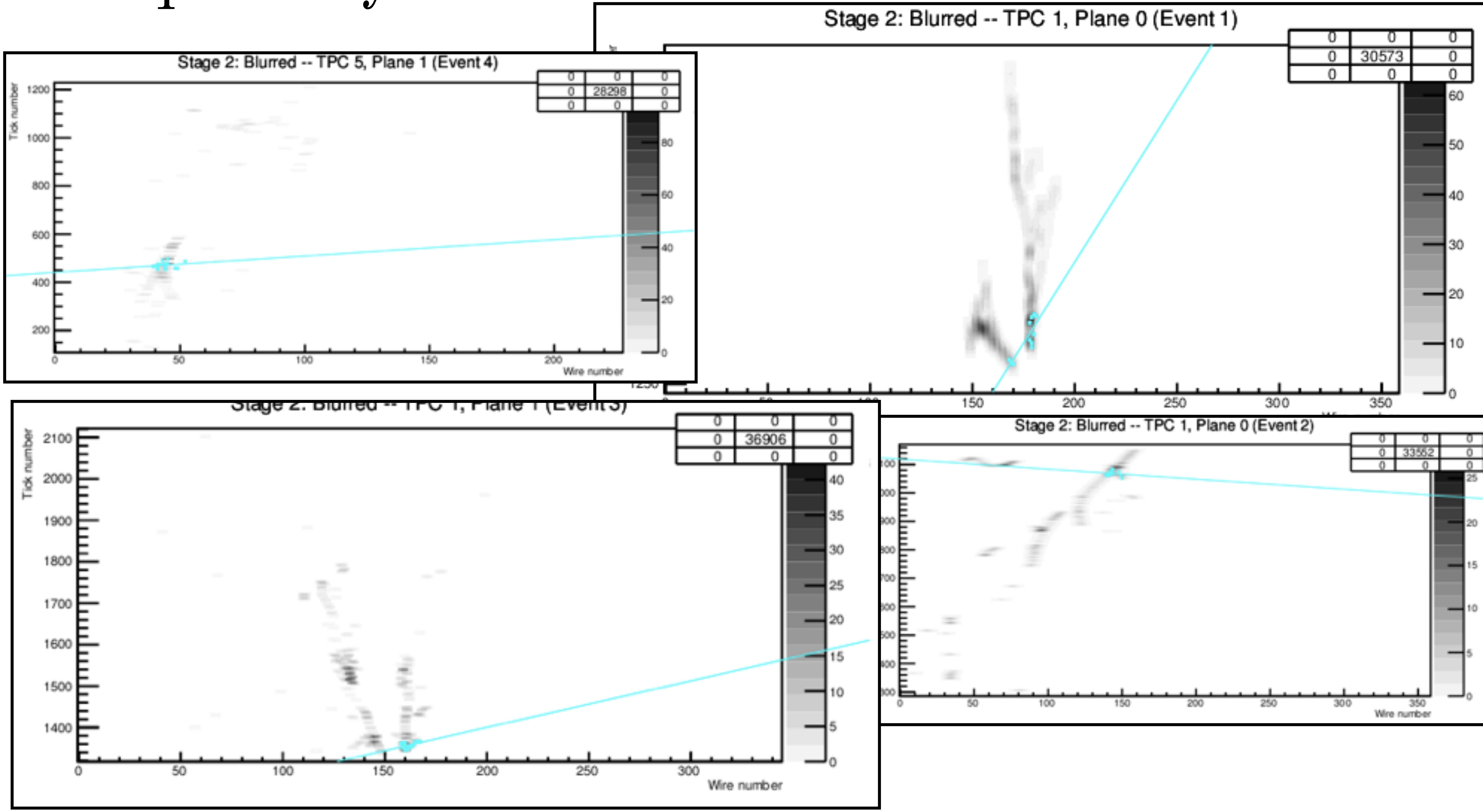
- I implemented this originally by using a gradient through a select number of points to hypothesise the direction…



- Great when it worked! However…

What I Didn’t Show Last Time…

- … It ~~quite~~ very often failed!

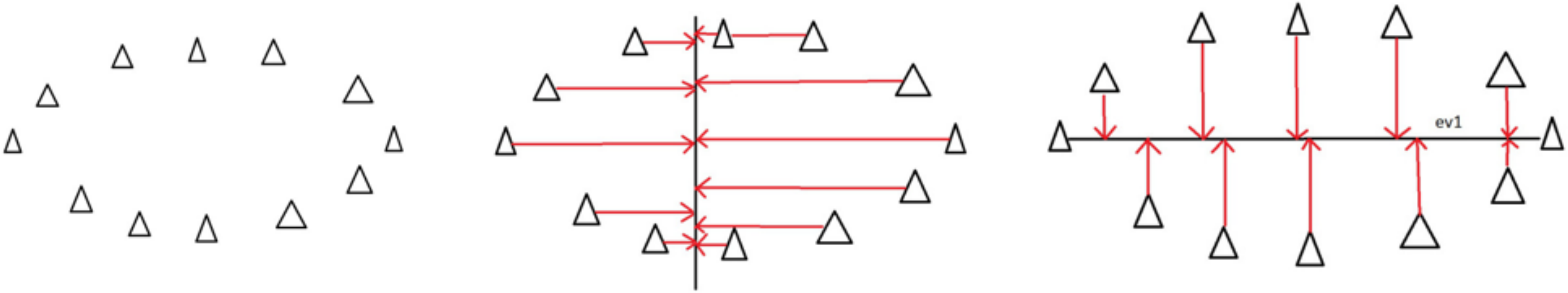


Using a PCA

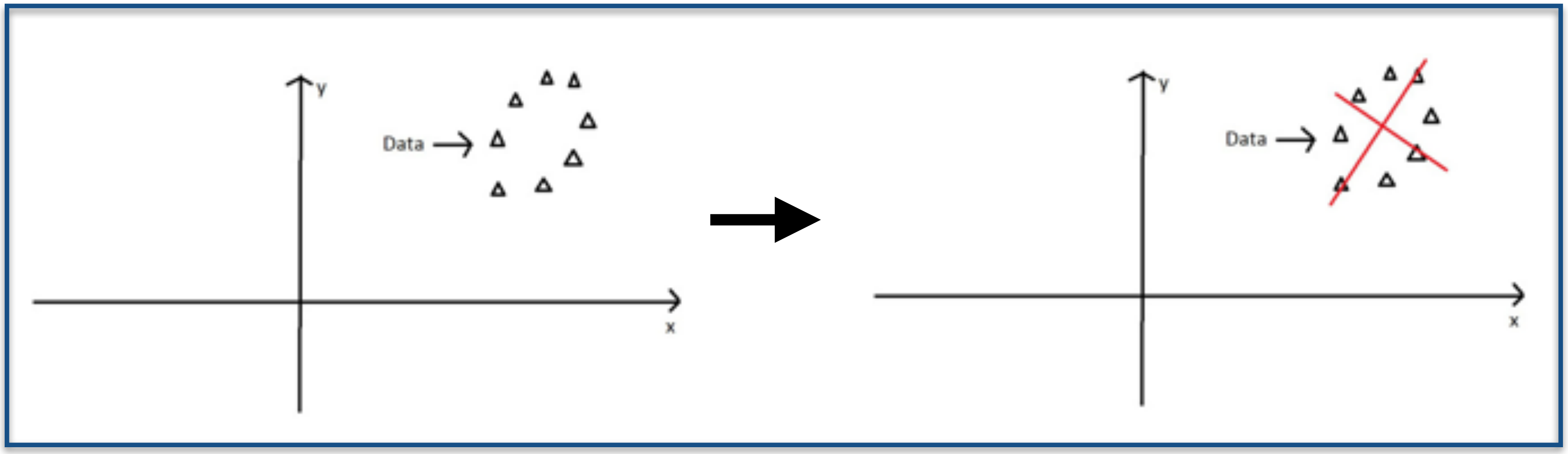
- It appeared that if I got the direction right, the clustering would work very well…
- I started experimenting using a Principal Component Analysis (PCA) to find the rough directionality of the clusters.
- HUGE thanks to Dom Brailsford (Lancaster) for suggesting this at the previous meeting when I presented my initial attempts!

Principal Component Analysis

- Finds the principal component of a set of data points…
- I learnt about them last week from this blog:



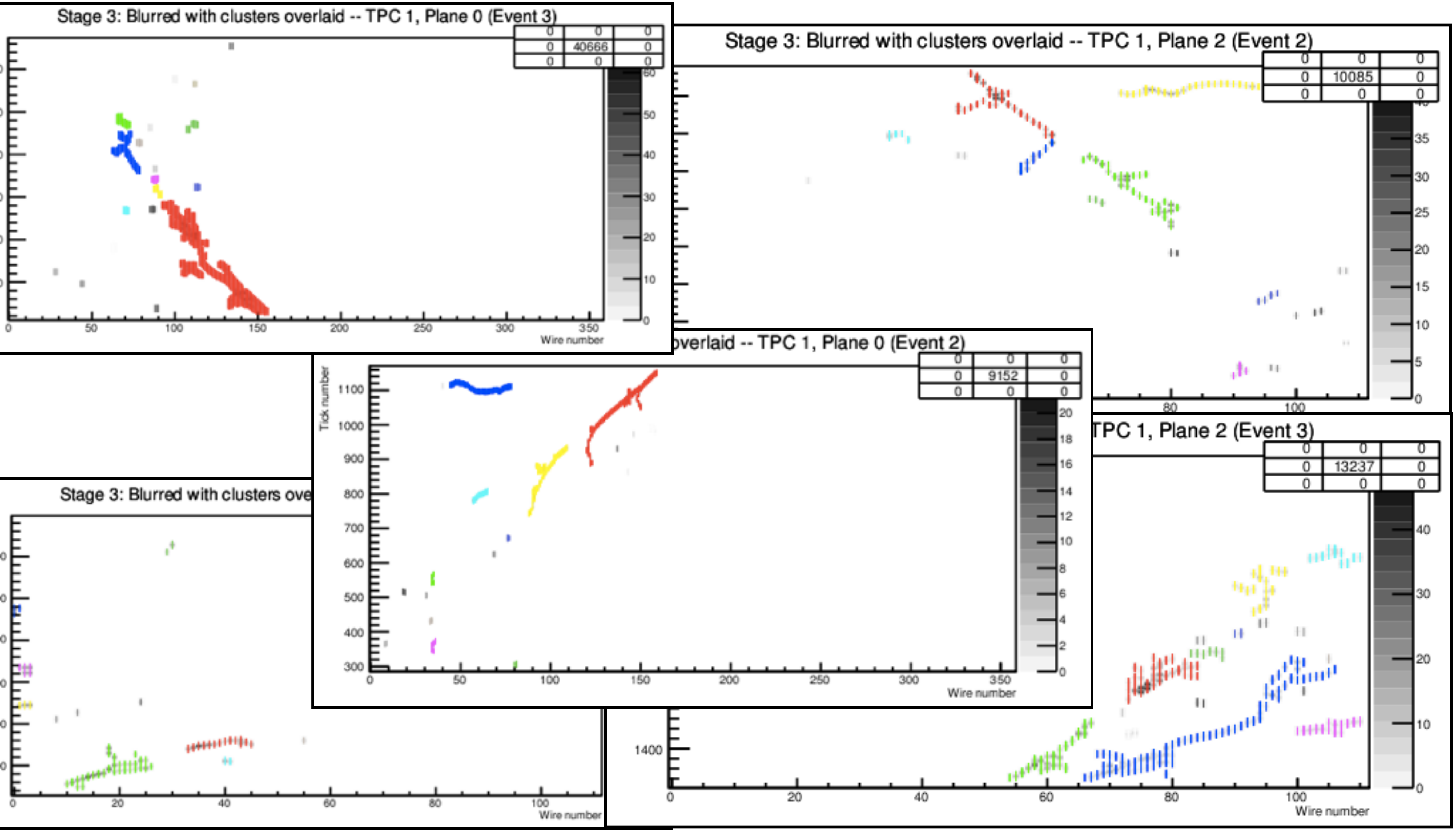
More variance — principal component



Improved Dynamic Blurring

- Using a PCA, the principal axis is now found for each TPC/plane requiring clustering, and the appropriate blurring radii are taken from this.
- The blurring thus follows the path of the particle much more accurately and yields much better reconstruction.
- Will show some completeness/cleanliness plots later on…

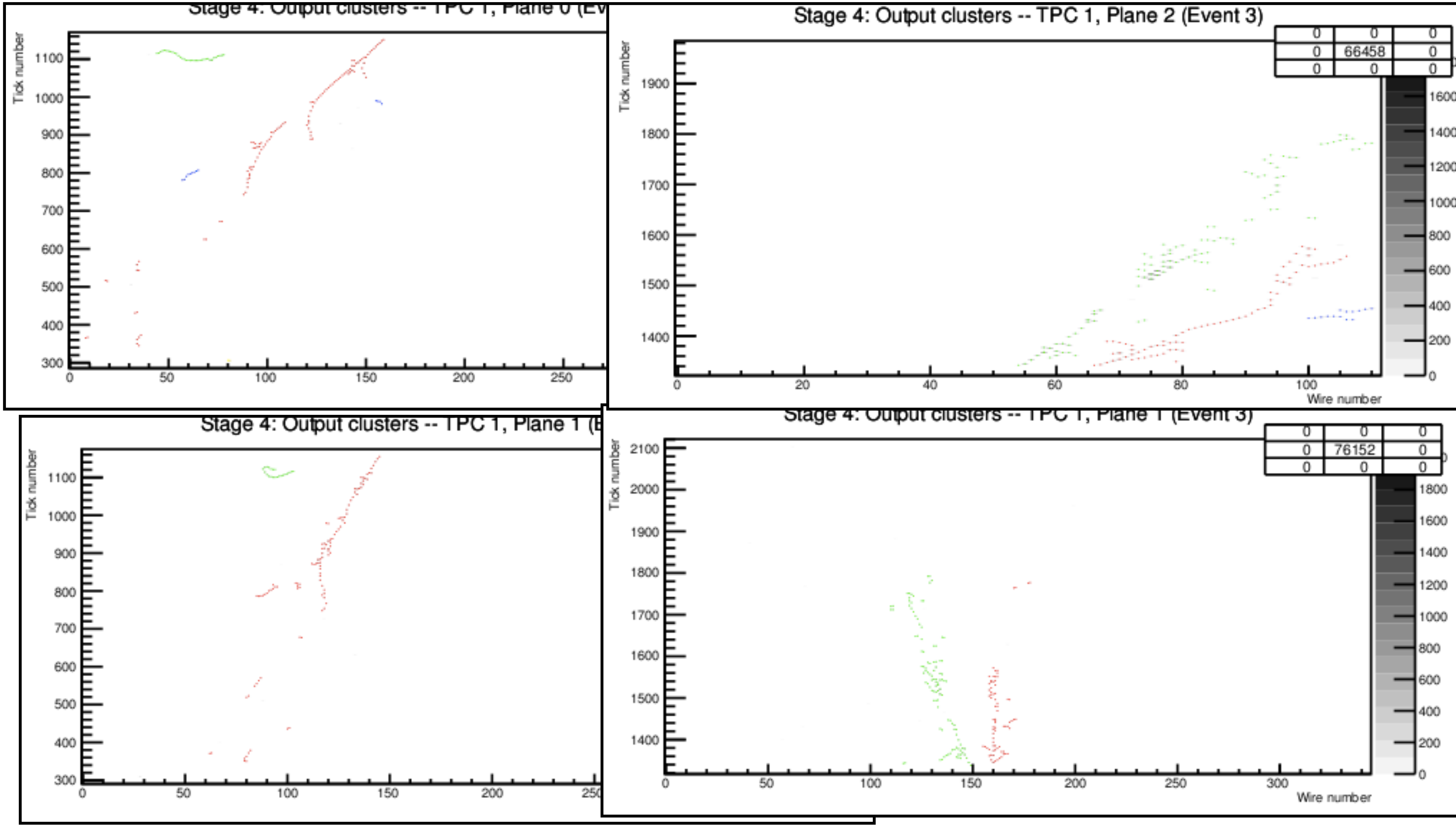
Final? Problem



PCA To The Rescue!

- The clustering works well after the blurring follows the particles as much as possible. However, there are cases where a track/shower is obviously split into multiple fragments…
- After the initial success of PCAs, decided to try and make use of them again!
- Added a merging algorithm:
    - Runs at the end of the clustering algorithm
    - Considers all possible matches of cluster recursively and calculates the PC for each
    - If the component has a sufficiently high eigenvalue (indicating a very straight line), the clusters are merged.
- Now…

More Complete Clusters



The Merging Algorithm

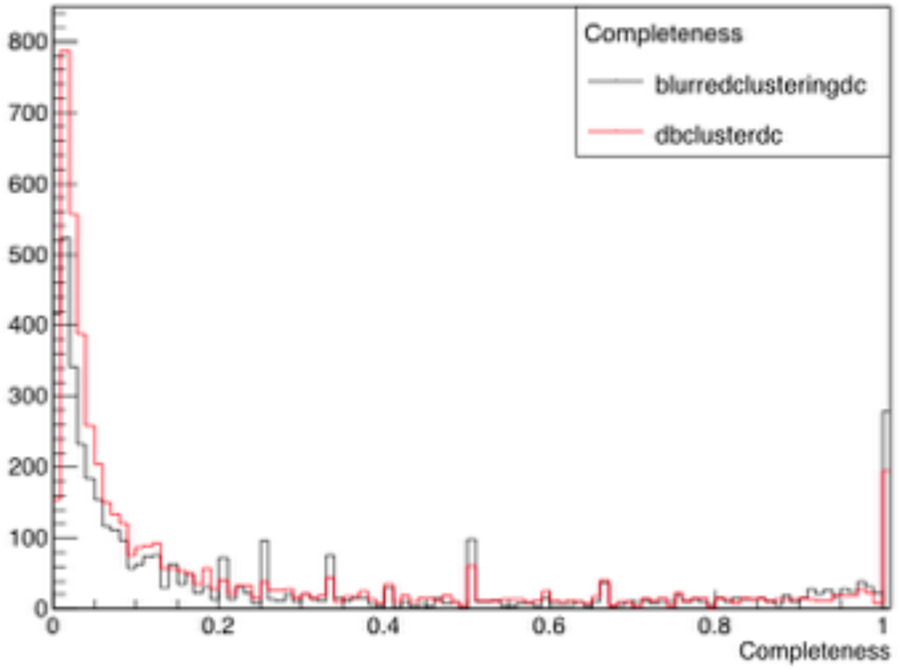
- Written very generically and designed to run over the final output clusters from any clustering algorithm
  - i.e. runs over std::vector<art::PtrVector<recob::Hit> > s
- From looking at many, many, many event displays recently, I see dbcluster has the same problem.
- Will probably be useful for other algorithms too, so I'm happy to write it as a separate module instead as a method of the Blurred Clustering algorithm.
- Two free parameters: minimum size of cluster to merge and merging threshold (minimum eigenvalue needed to merge).

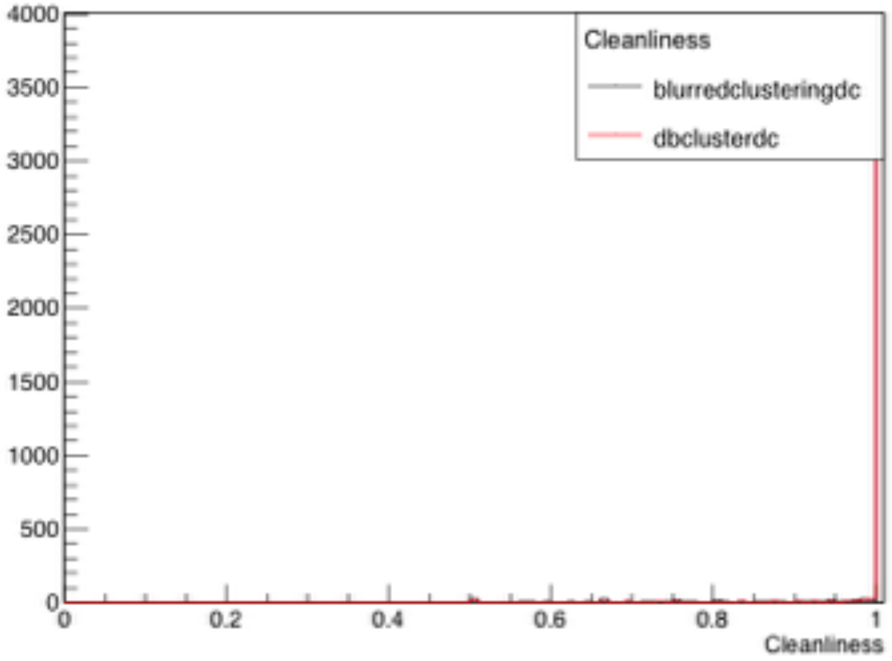
Characterising The Clustering

- I have now implemented almost all the possible improvements I have thought of, so this is as close to the best clustering I feel is possible!
- It will be instructive to characterise and again compare to dbcluster.
- Use the completeness, cleanliness, efficiency metrics defined in many previous talks:
  - Completeness: hits clustered/hits left by particle
  - Cleanliness: hits associated with particle in cluster/hits in cluster
  - Efficiency: fraction of all events which pass cut (2 clusters, each >=50% complete)

Weighted Histograms

- Prior to this week, the distributions were populated mainly with high cleanliness, low completeness clusters (e.g.



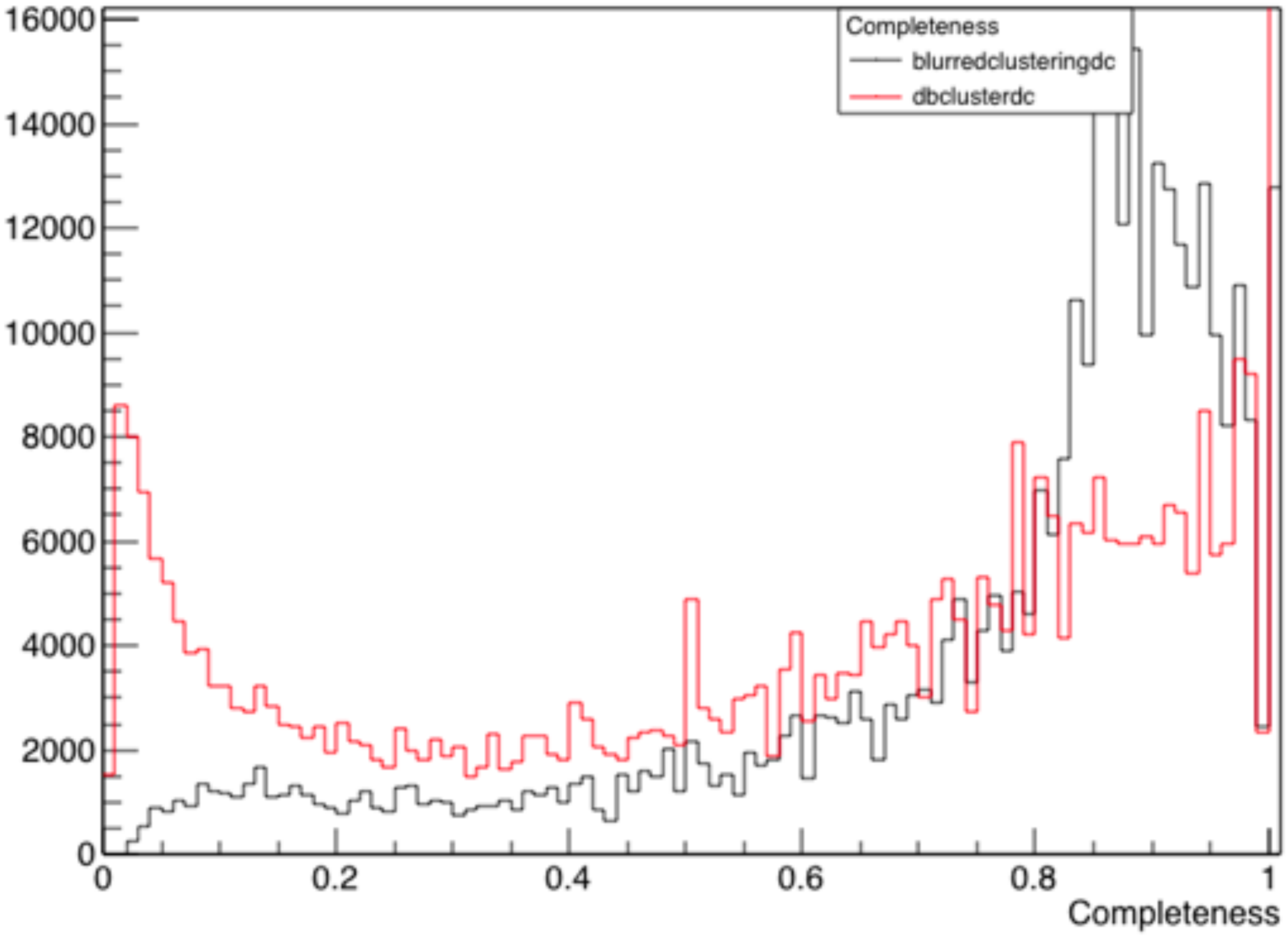


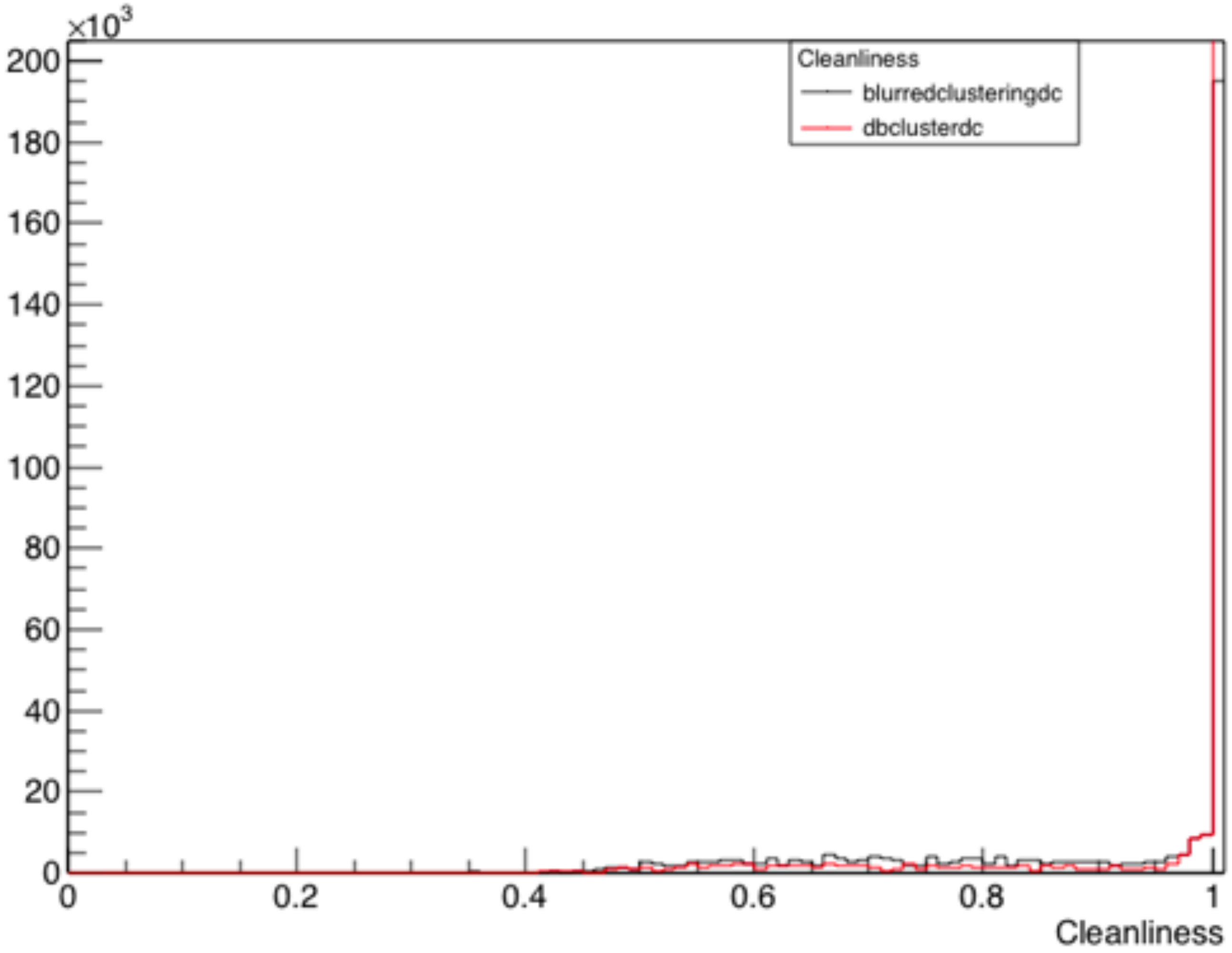
)

These are all small clusters (<10 hits) which are very clean but very fragmented and skew the effect of the histograms massively.

- They are now weighted by cluster size (number of hits).

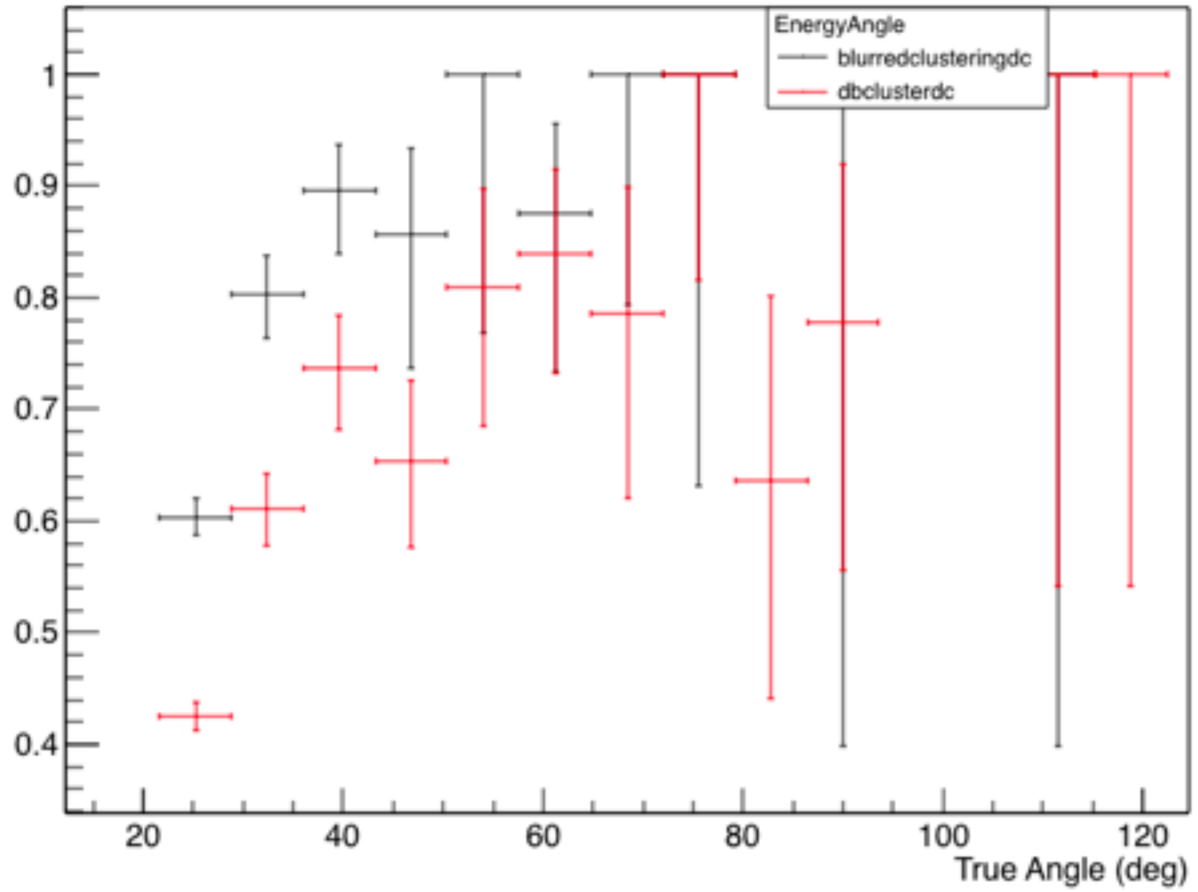
Cleanliness / Completeness



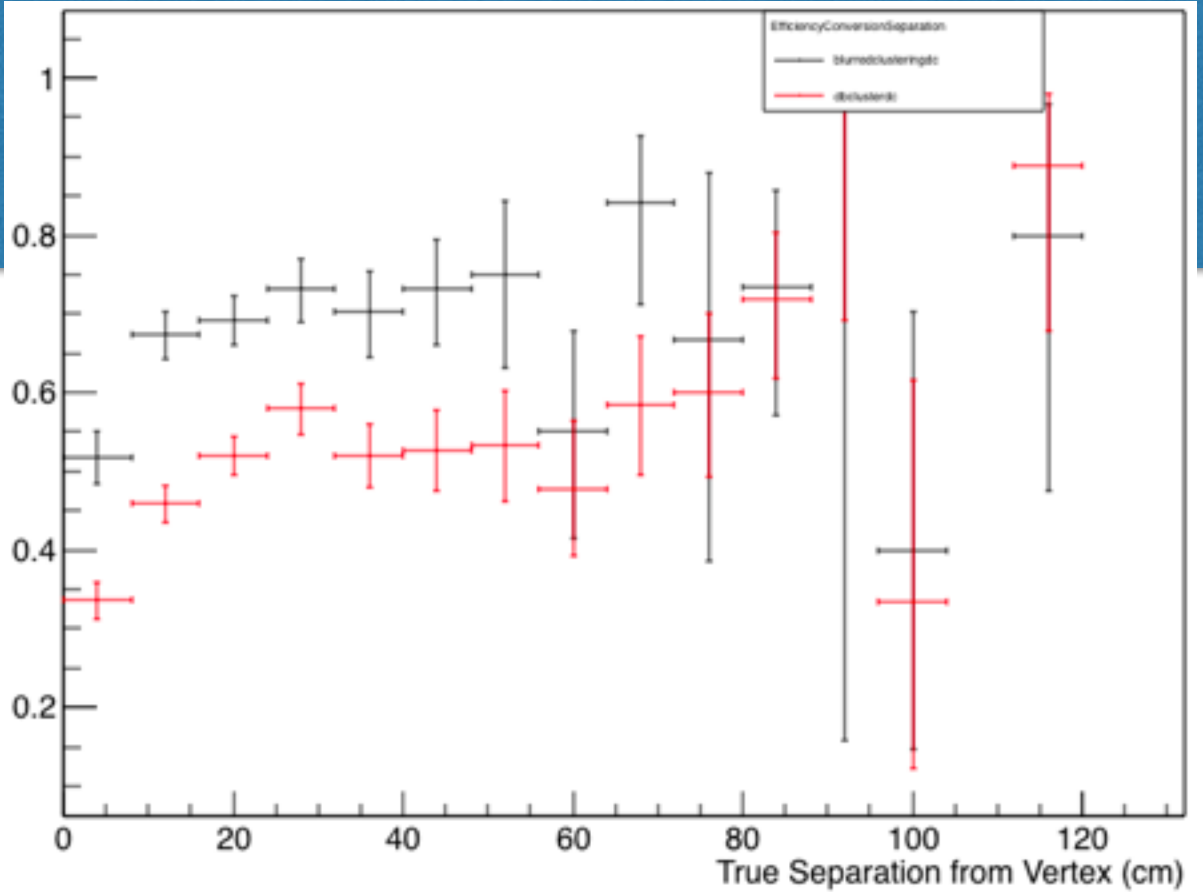


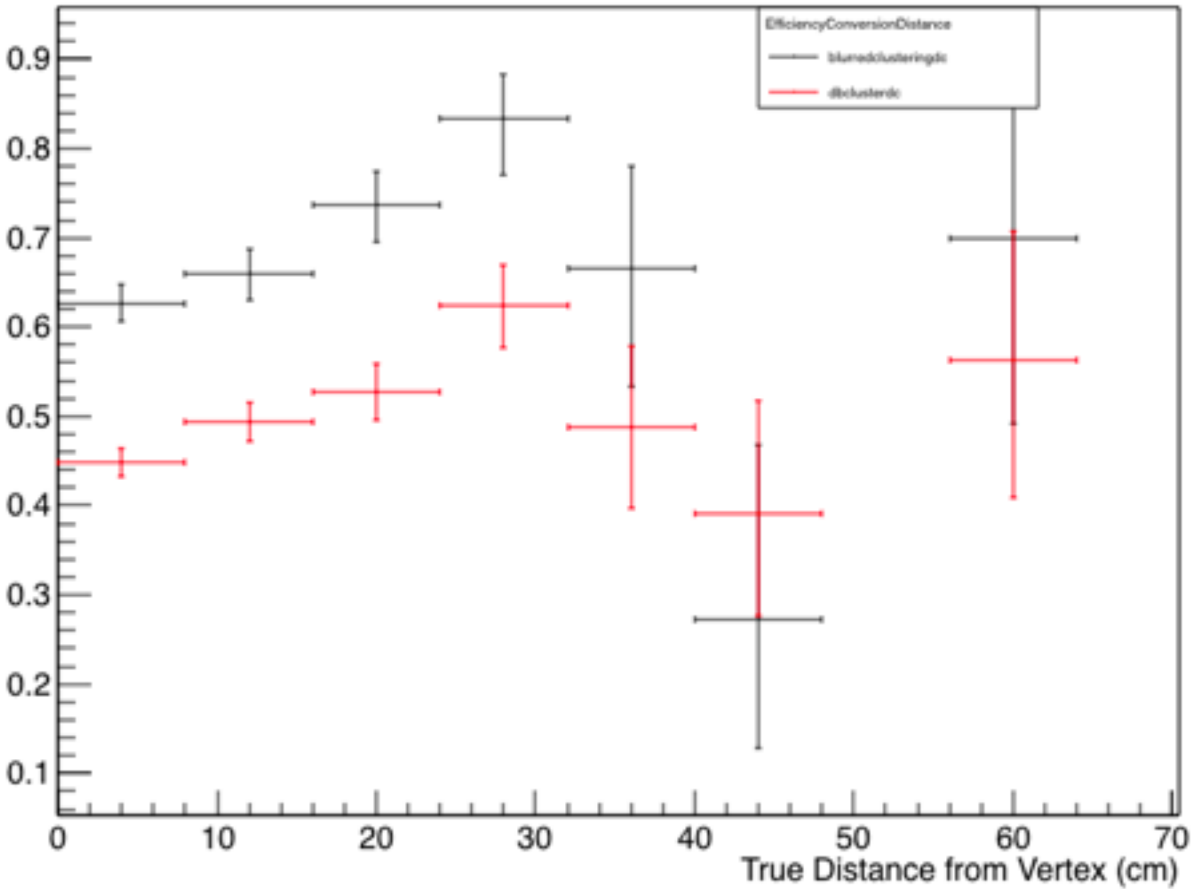
- 500 events.
- Blurred Clustering significantly better than dbclsuter now.

Efficiencies

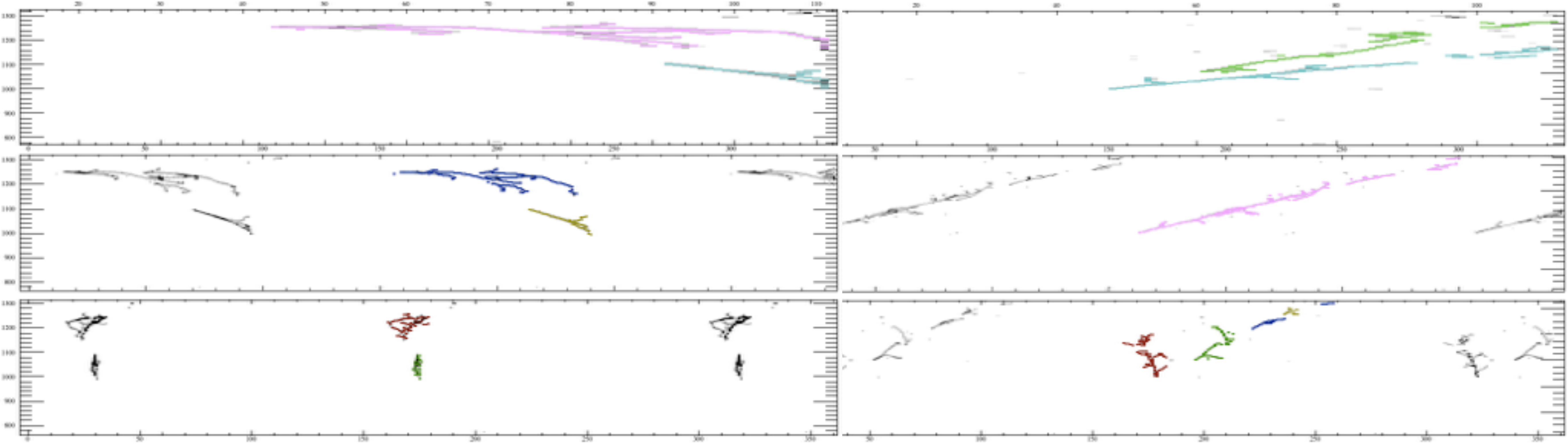


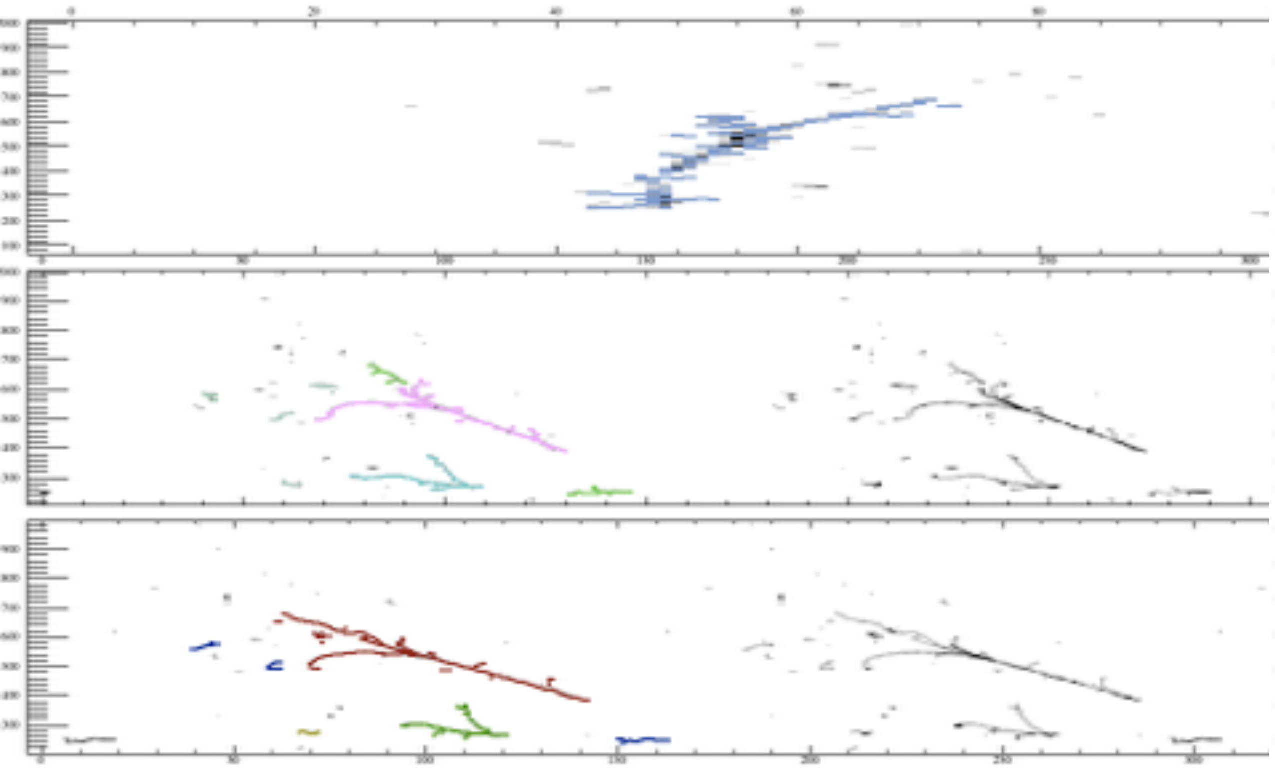
- Decay angle (above)
- Conversion separation (top right)
- Conversion distance (bottom right)

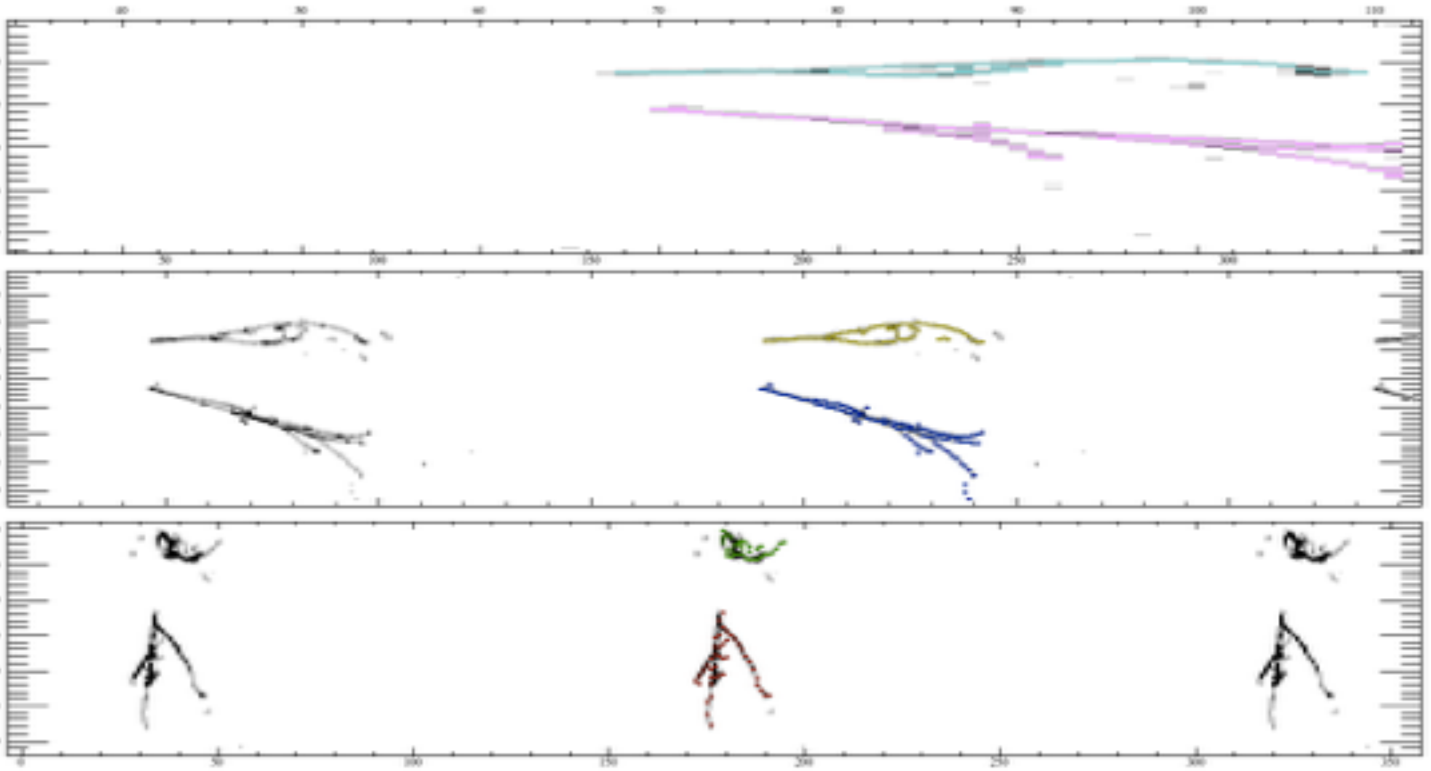




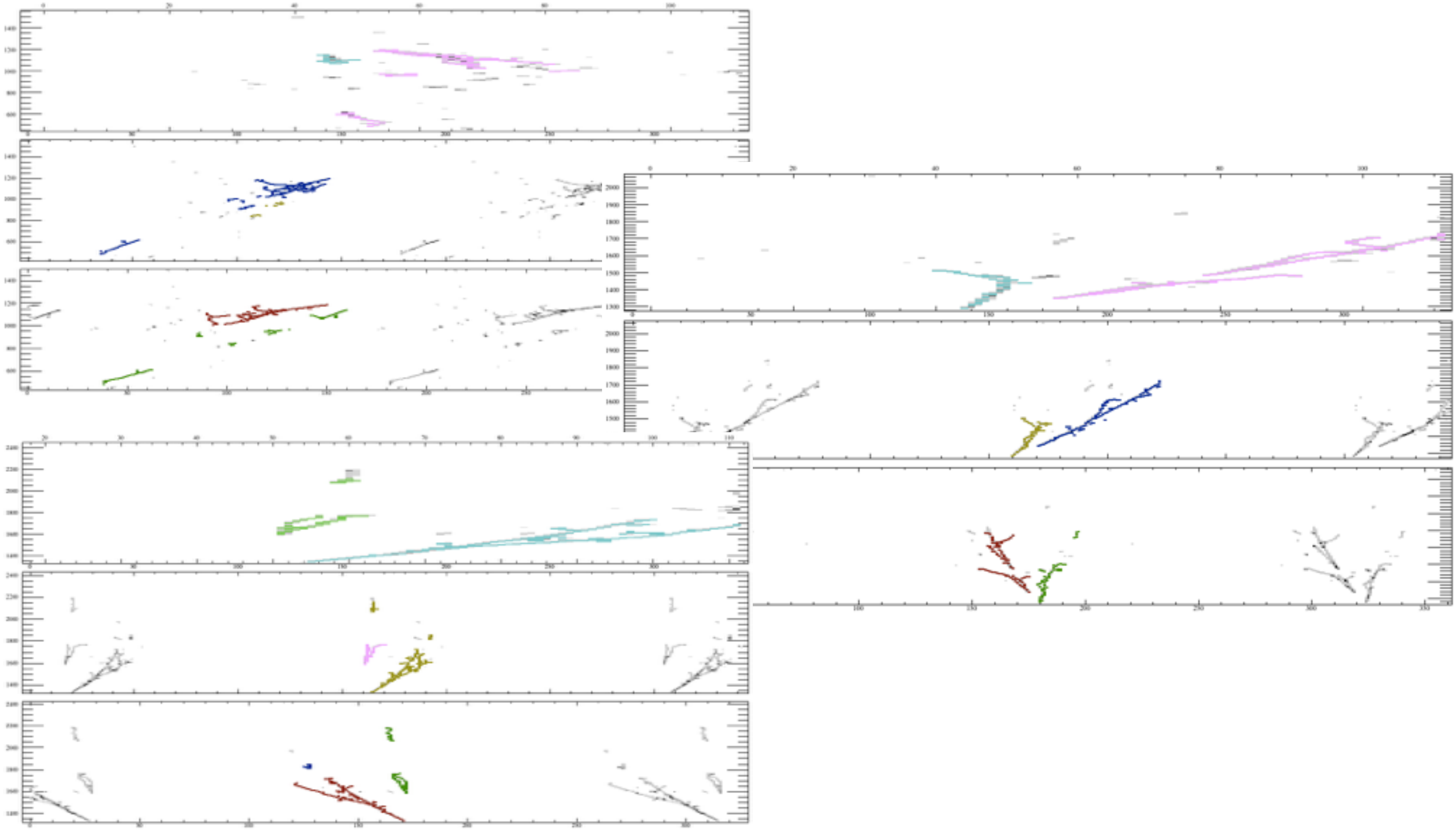
Examples...





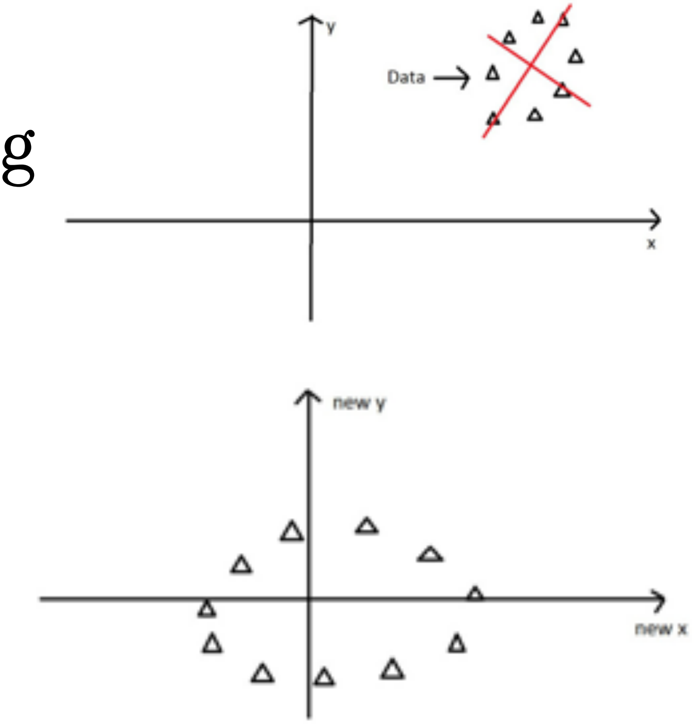


Examples...



Improvements

- I’m happy with how the clustering looks now and don’t have many huge improvements I can think of…
- Couple of ideas:
  - Dynamic Sigma: determine the Gaussian sigma dynamically (analogous to the radii) for different blurring if considering two close tracks or a spread shower.
    - Not sure if sigma has too much of an effect so will probably leave this for the moment.
  - Cluster in PC/SC space: instead of blurring and clustering in the wire/tick space, it is more intuitive to do this in the space defined by the two components found by the PCA:
    - May improve things but will be a lot of work! Considering it…



Summary

- Blurred Clustering is tuned and gives very nice clusters for the pi0 sample.

- It is a flexible algorithm (many, many parameters!) and so can be tuned to provide many different types of clustering.

- It is probably as good as it can be right now so I am going to move on and use it for shower reconstruction etc.

- Will update it whenever necessary!