Vertexing and 2D clustering in Pandora

Andy Chappell

13th January | FD Sim/Reco Meeting

DEEP UNDERGROUND NEUTRINO EXPERIMENT

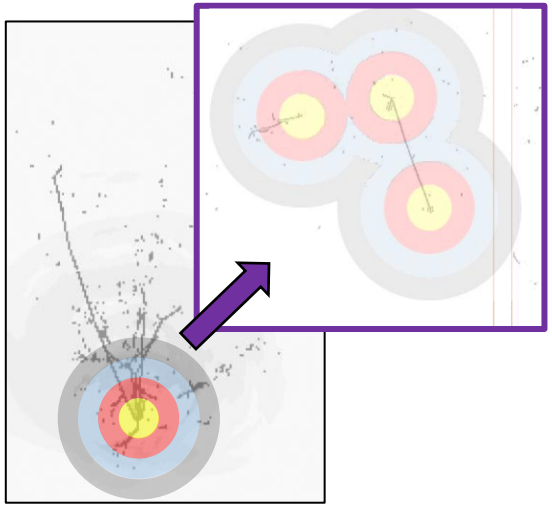
WARWICK

Status of the primary vertexing paper

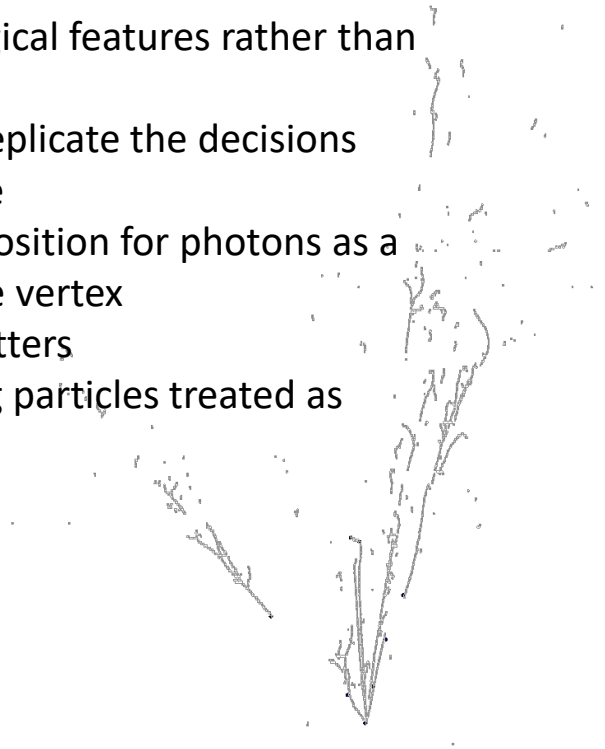
- DUNE paper on primary vertexing with deep learning in Pandora
  - Responded to comments from first round of Collaboration Review
  - Second round of Collaboration Review will start soon

Secondary vertices

Learn distance from hits to closest, rather than primary vertex



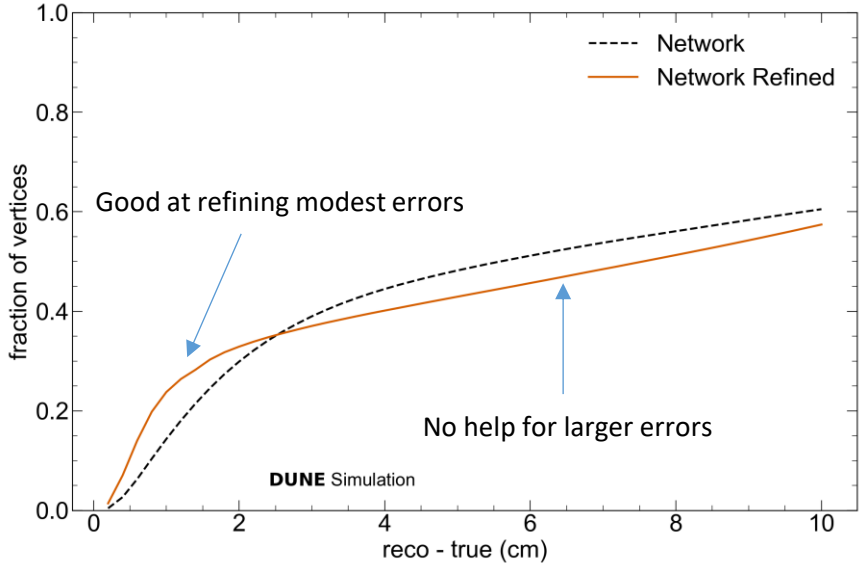
- Attempt to capture topological features rather than exactly match simulation
  - Want the network to replicate the decisions you would make by eye
  - Tag start of charge deposition for photons as a vertex, rather than true vertex
  - Ignore small elastic scatters
  - Track endpoints/exiting particles treated as vertices



Refining reconstruction using higher order vertices

Work in progress

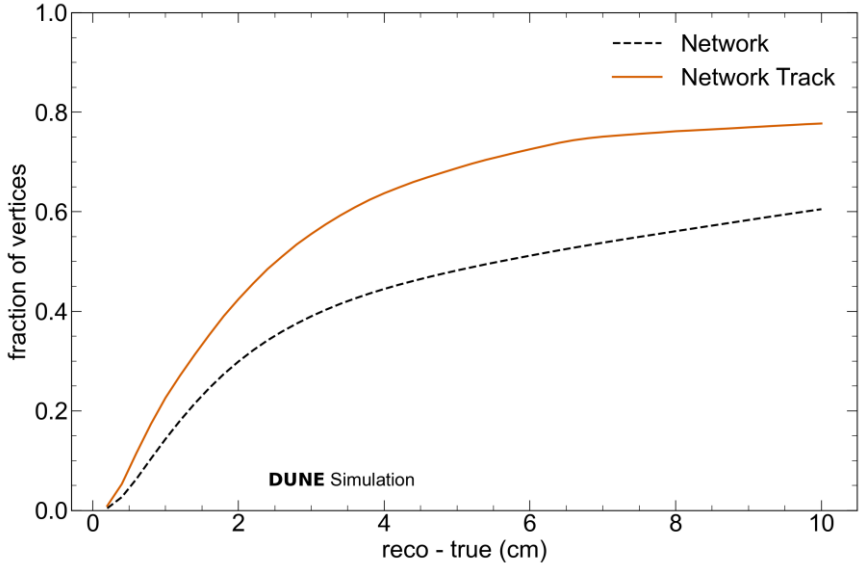
- Network region finding quite good, but location imprecise
- Re-run at high resolution too computationally expensive
- Refine the vertices by considering how cleanly hits align with angular bins around the estimated vertex region

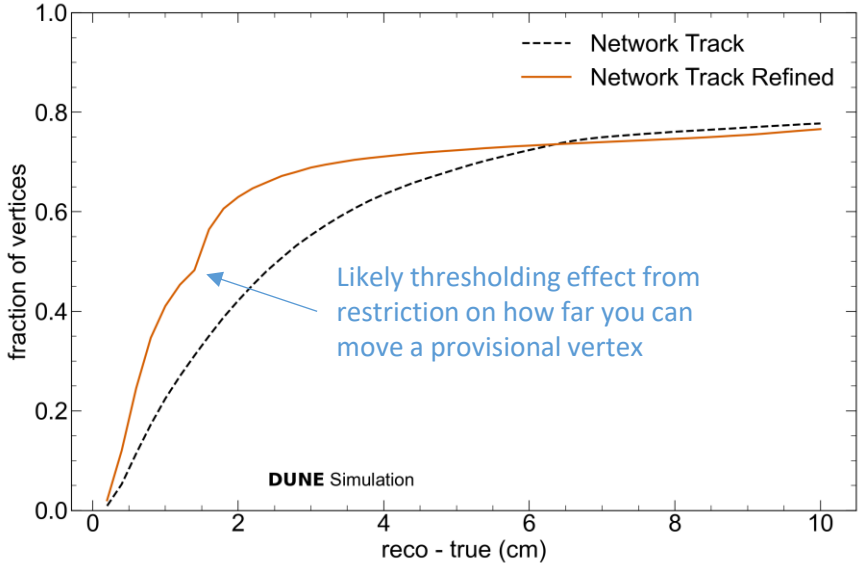


Restricting topologies

- Anecdotally, the shower regions are the most challenging
- What if we ignore them?
- Only consider vertices with track-like topologies
- Track/Shower tagging network may allow more precise identification of vertices by filtering shower hits

Work in progress

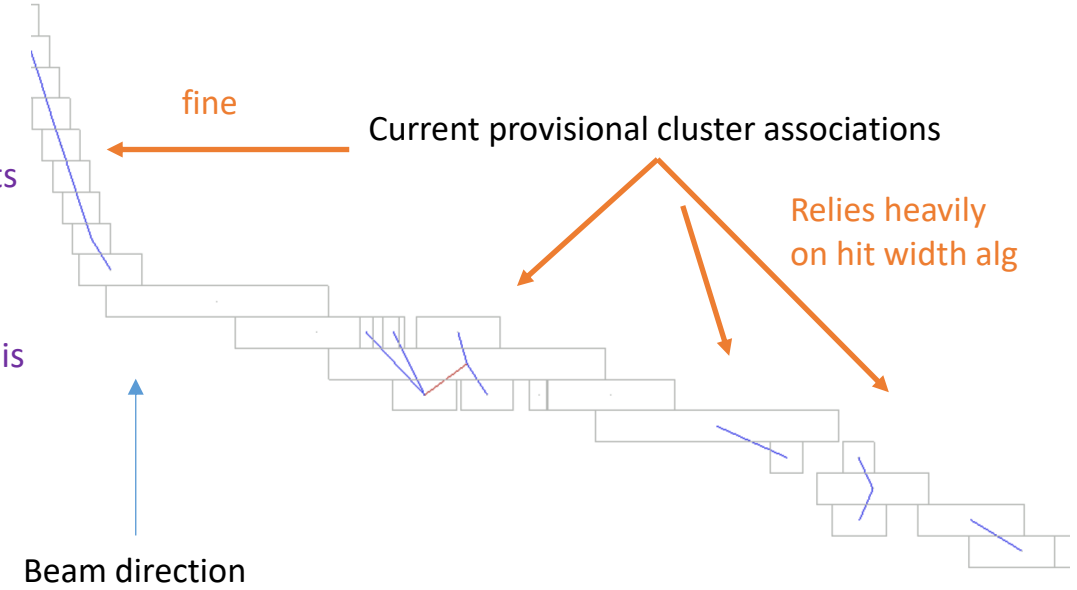




Improving provisional 2D re-clustering

- In addition to fixing problems after they've occurred, there is an ongoing effort to improve the quality of the initial clustering
- The very first clustering algorithm that runs in Pandora is very beam-centric
- To the right is an example of a collection of candidate associations between hits
- Note the beam-aligned hits have good associations, but the wide, transverse hits have only sparse connections between hits on adjacent wires

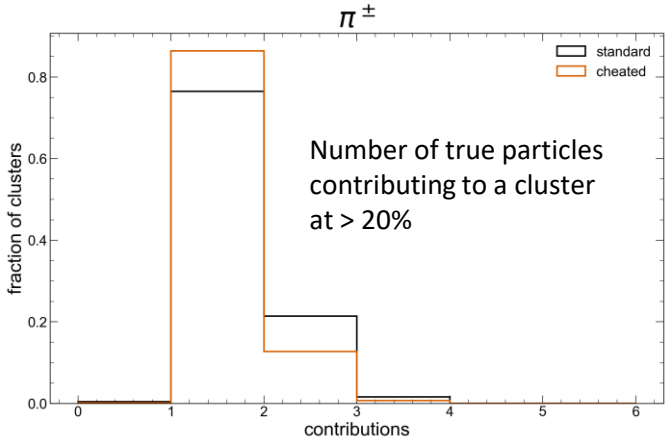
- Not ideal, even in an LBL scenario
- Isotropic samples even more subject to this

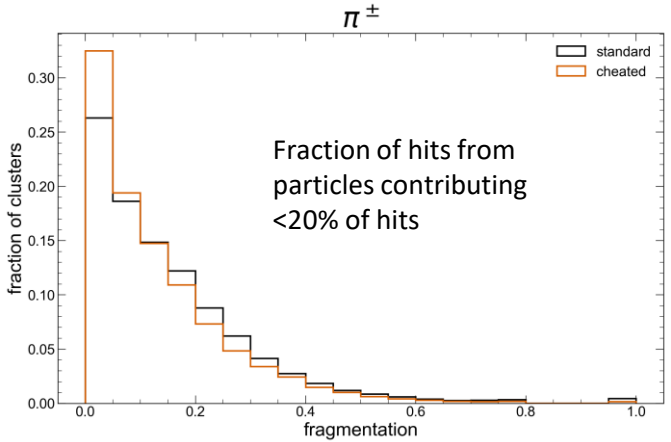


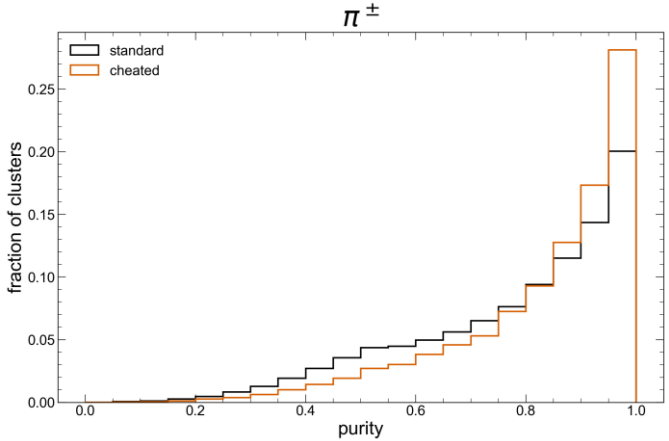
Cheating workflow to test alternative approach

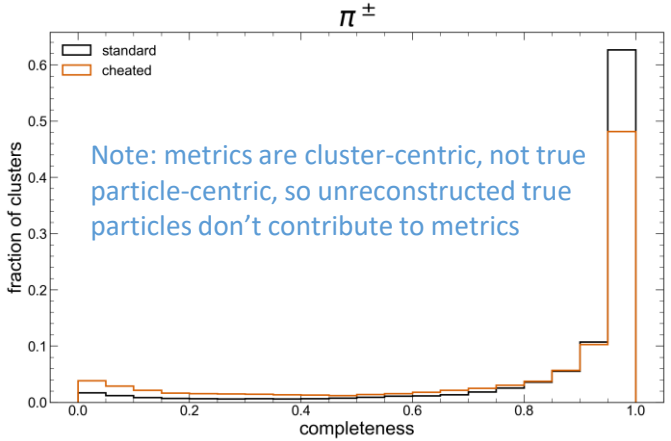
- Cheating 2D clustering is quite tricky
  - Simply clustering hits based on true parent particle leads to perfect 2D clusters
  - This makes downstream steps trivial
  - End up with an unrealistically good reconstruction
  - Learn nothing about the likely effect of a real algorithm
- Developed two cheating algorithms
  1. Cheated cluster splitting: Let the 2D reco proceed in the usual way for one view, then identify **substantial** contributions to clusters from additional true particles, and split them based on truth (i.e. let small errors remain)
  2. View matching: Use the clusters in one view to match to hits in the other views, and create the corresponding clusters in those views – improving the inter-view coherence
- The idea here is to see if a Kalman filter (or similar), that considers view correspondence, can produce better provisional clusters
  - By considering the views together, we have fewer ambiguities in 3D than in 2D (more complete provisional clusters while maintaining purity, easier 2D->3D matching)
  - By using a Kalman filter approach we can remove the dependence on a preferred direction

Performance of cheating workflow – charged pions









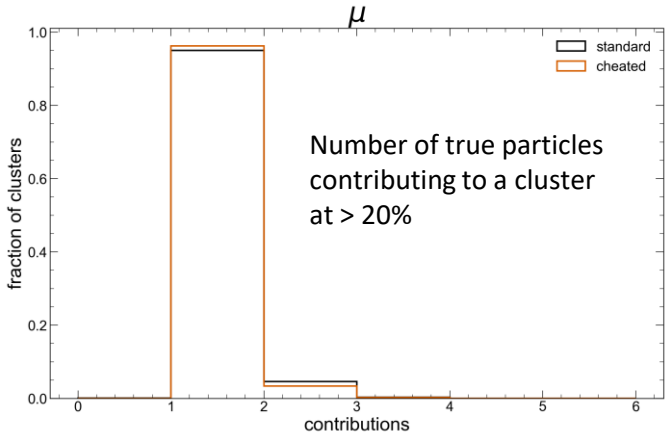
See backup for other track-like particles

Summary and next steps

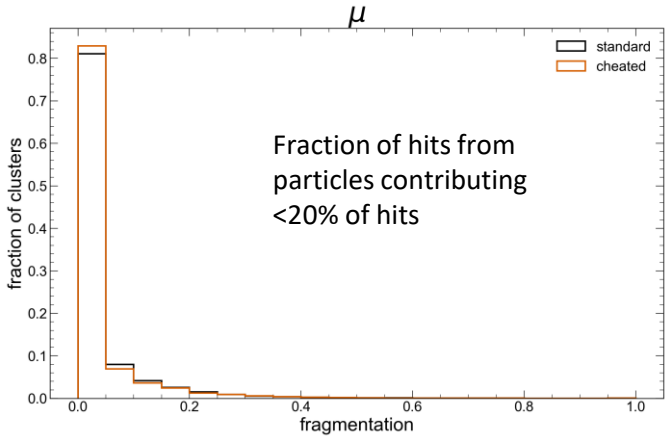
- Significant gains in secondary vertexing performance if we can filter out shower-like contributions to network inputs
- Cheating studies indicate improvements to the very first 2D clustering algorithm can have large downstream effects

Backup

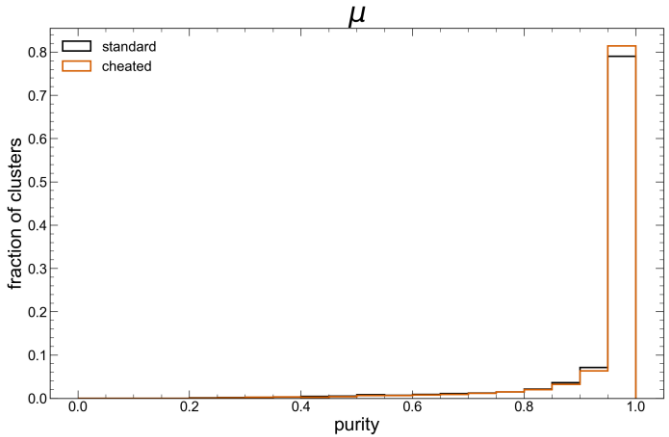
Performance of cheating workflow - muons

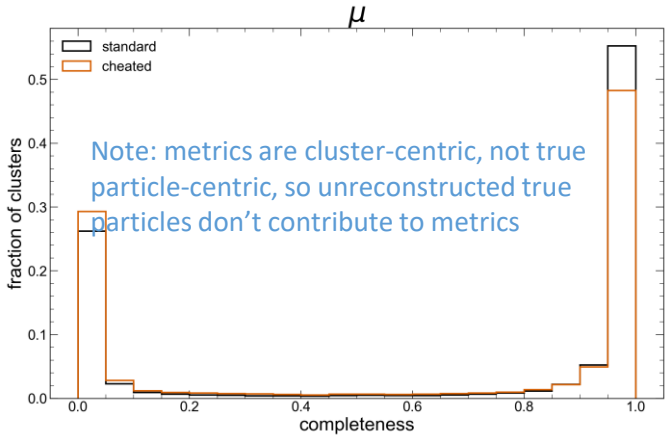


Number of true particles contributing to a cluster at > 20%



Fraction of hits from particles contributing <20% of hits

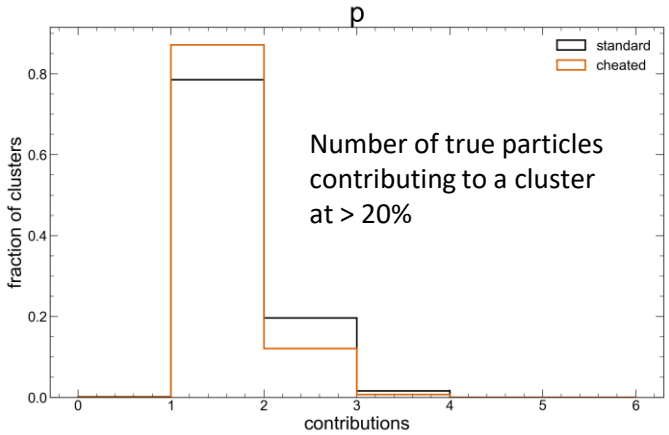


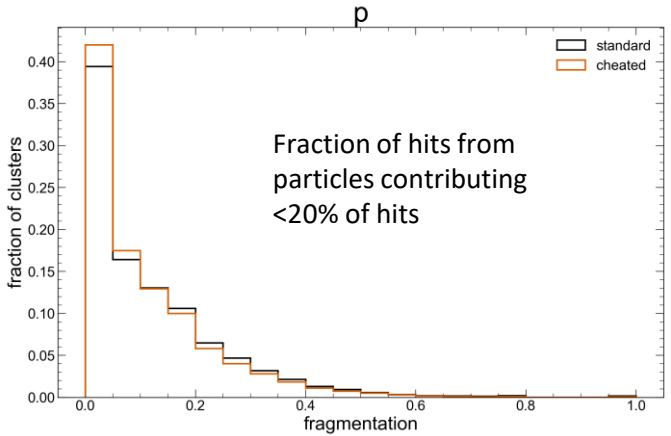


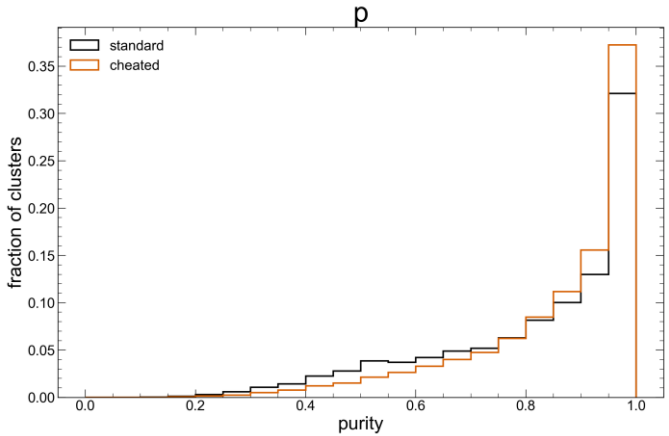
Note: metrics are cluster-centric, not true particle-centric, so unreconstructed true particles don't contribute to metrics

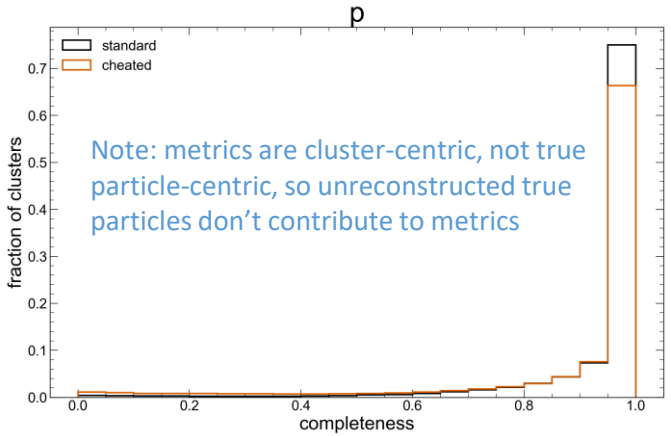
See backup for other track-like particles

Performance of cheating workflow - protons



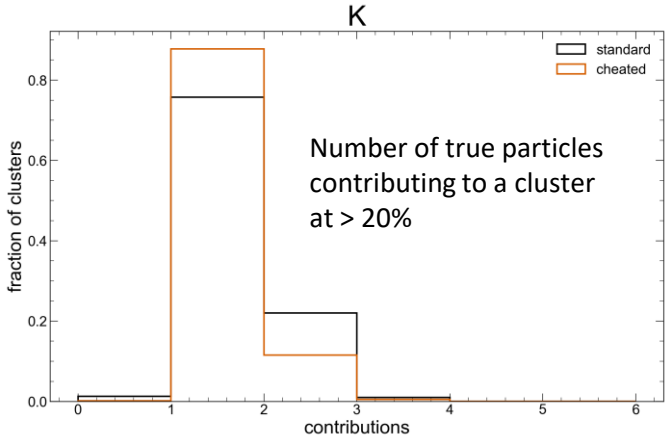


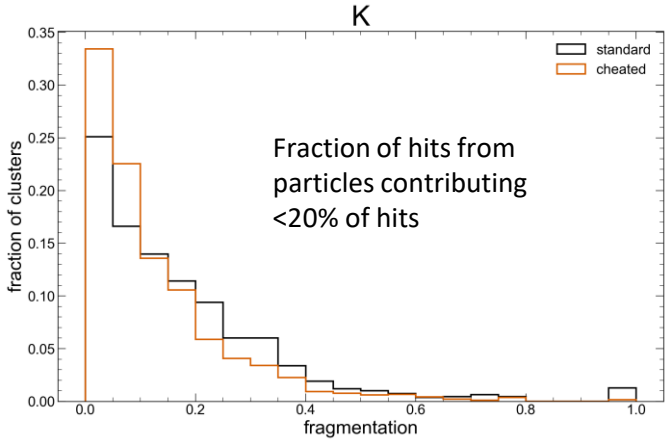


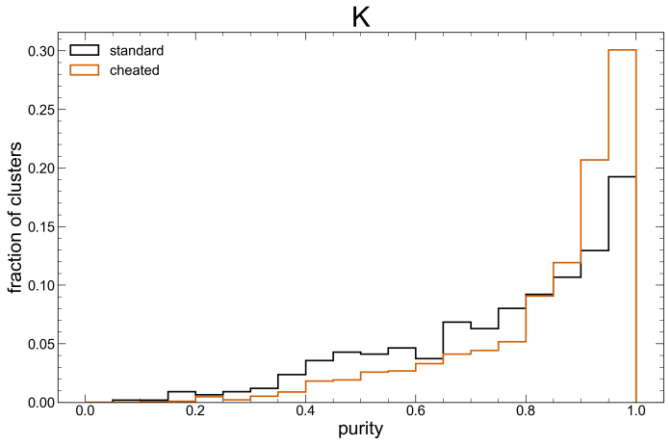


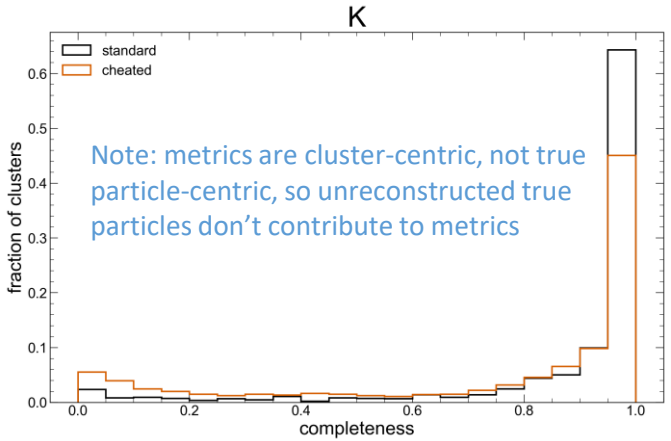
See backup for other track-like particles

Performance of cheating workflow - kaons









See backup for other track-like particles