# Track Creation/ Selection in Pandora

High Level Reconstruction Workshop

Boruo Xu - University of Cambridge

# Notes

- Comments from John Marshall and Mark Thomson:
- One needs to be extremely careful regarding track selection. A very long time has been spent on understanding the details of tracking failures.
- Re-evaluating this would probably be a good idea. It would, however, probably be a substantial task.
- Minor note:
- All numbers in the following slides are configurable. I put the number there instead of the variable for clarity.

# Notes II

- Only proper reference for this is the C++ implementation and then provide some links to the viewvc pages for:
- The MarlinPandora -> TrackCreator
- The TrackPreparationAlgorithm
- The PfoConstructionAlgorithm
- The TrackCreator is responsible for deciding which tracks are passed to Pandora (all, unless they fail cuts on the number of track hits) and setting the track usage flags.

# Basic logic

- if ( nTrackHits < **minTrackHits** OR nTrackHits > 5000):
  - Discard the track. It is not in the Pandora Track list
- Otherwise: track is in the **Pandora Track list**
- In **Pandora Track List**:
- A track object has boolean flags **canFormPfo** and **canFormClusterlessPFO**
- A track would ONLY become part of PFO if:
  - **canFormClusterlessPFO** = True, OR
  - **canFormPfo** = True AND the track has associate cluster(s)

# Set track flags

- If track doesn't reach calorimeter OR a track is a parent track(has daughters), don't set flags.
- The track is still in the track list, but won't be used to form PFO
- This is because they are subsequently handled by the two algorithms in Pandora. These use tricky recursive functions to navigate the track relationships.
- The idea is that a good quality daughter track, reaching the calorimeter, will pick-up an associated cluster. The final Pfo will include both the daughter and parent tracks, plus the cluster. There is then a set of rules for extracting the Pfo properties from these constituents.
- Otherwise…

# Set track flags II

- If track doesn't pass **QualityCuts:**
  - Set **canFormPfo** if track is a daughter track OR track is a V0
- Otherwise, set **canFormPfo** under conditions:
  - if (**d0** < 50 AND **z0** < 50 AND **rInner** < TPCinnerRadius + 50), OR
  - If **passRzQualityCuts** AND UseNonVertexTracks[True], OR
  - If track is a daughter track OR track is a V0
- **D0** = absolute value of the 2D impact parameter w.r.t (0,0)
- **Z0** = absolute value of the z coordinate at the 2D distance of closest approach
- **rInner** = minimum radius in x-y plane of track hits, sqrt(x^2+y^2)

# Set track flags II

- If track doesn't pass **QualityCuts**…
- Otherwise, set **canFormPfo** under conditions…
- At same time, set **canFormClusterlessPfo** under conditions:
  - If UseUnmatchedVertexTracks[True] AND trackEnergy<5:
    - if (**d0** < 5 AND **z0** < 5 AND **rInner** < TPCinnerRadius + 50), OR
    - If **passRzQualityCuts** AND UseNonVertexTracks[True] AND UseUnmatchedNonVertexTracks[False], OR
    - If track is a daughter track OR track is a V0
- **D0** = absolute value of the 2D impact parameter w.r.t (0,0)
- **Z0** = absolute value of the z coordinate at the 2D distance of closest approach
- **rInner** = minimum radius in x-y plane of track hits, sqrt(x^2+y^2)

# End of track creation

- Set other track parameters
- End of track creation
- Following slides are definitions of variables used previously.

# Calculate minTrackHits

- If (track_tan(Lambda)) <= **tan(Lambda)FTD:**
  - **minTrackHits** = 5
- Otherwise:
  - If (track_tan(Lambda)) > **tan(Lambda)FTD:**
    - Count layers [**ExpectedFTDHits**] which track_tan(Lambda) in between tan(Lambda) FTD for inner radius and outer radius
  - **minTrackHits** = max(5, **ExpectedFTDHits** )
- **tan(Lambda)FTD** = FTD Z Position / FTD outer Radius

# Terminology

- **D0** = absolute value of the 2D impact parameter w.r.t (0,0)
- **Z0** = absolute value of the z coordinate at the 2D distance of closest approach
- **rInner** = minimum radius in x-y plane for all track hits, sqrt(x^2+y^2)
- **zMin** = absolute value of minimum Z distance for all track hits

# passRzQualityCuts

- **passRzQualityCuts** = True, if:
  - **zMin** < **zCutForNonVertexTracks**, AND
  - **rInner** < TPCinnerRadius + 50
- **zCutForNonVertexTracks** = TPCinnerRadius * |**pZ** / **pT**| + 50
- **pT** = momentum at the distance of closet approach in x-y plane
- **pZ** = momentum at the distance of closet approach in z direction

# PassesQualityCuts

- False if:
  - If the distance of trackState at calorimeter to IP < 100, OR
  - If track_Omega = 0 (Omega is the signed curvature of the track in [1/mm].The sign is that of the particle's charge.), OR
  - If **sigmaPOverP** > 0.15
- True if:
  - |Momentum| at distance of closet approach <= 1
- If |Momentum| at distance of closet approach > 1
  - False if:
    - If **pT** = 0 OR pZ = 0 OR RadiusOfInnermostHit = TPCouterRadius, OR
    - if **nTpcHits** < **minTpcHits** AND **nFtdHits** < 2
- Otherwise, if it is not set to False, return True

# Terminology II

- **sigmaPOverP** = sqrt(track->getCovMatrix()[5] / |track_Omega|)
  - getCovMatrix() Covariance matrix of the track parameters.Stored as lower triangle matrix where the order of parameters is: d0, phi, omega, z0, tan(lambda). So we have cov(d0,d0), cov( phi, d0 ), cov( phi, phi), ...
- **nTpcHits** = track->getSubdetectorHitNumbers, get TPC part
- **nFtdHits** = track->getSubdetectorHitNumbers, get FTD part
- **minTpcHits** = **nExpectedTpcHits***0.2

# nExpectedTpcHits

- **nExpectedTpcHits** = tpcMaxRowNumber * **frac**, if:
  - **pZ** <  ( tpcZmax / tpcOuterRadius * **pT** )
    - **Frac** = (tpcOuterRadius - **innerExpectedHitRadius**) / (tpcOuterRadius - tpcInnerRadius)
    - **innerExpectedHitRadius** = max(tpcInnerRadius, RadiusOfInnermostHit)

# nExpectedTpcHits

- [Override above] **nExpectedTpcHits =** tpcMaxRowNumber * **frac2**, if:
  - **pZ** <= (tpcZmax / tpcInnerRadius * **pT**) AND **pZ** >= (tpcZmax / tpcOuterRadius * **pT**)
    - **Frac2** = (tpcZmax * **pT** / **pZ** - **innerExpectedHitRadius**) / (tpcOuterRadius – **innerExpectedHitRadius**
- [Override above] **nExpectedTpcHits** = 0, if:
  - |**pZ**| / |momentumAtDca| < tpcMembraneMaxZ / tpcInnerRadius