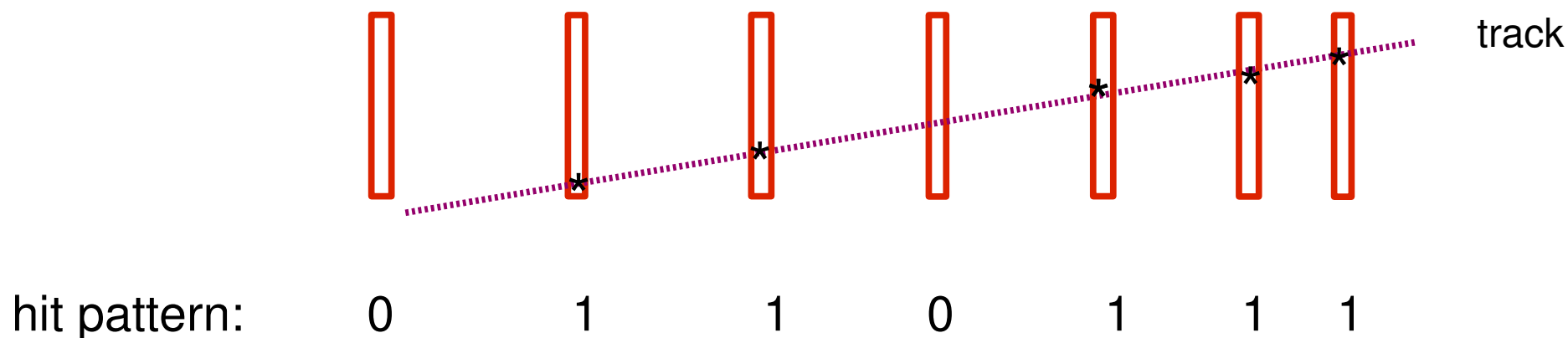


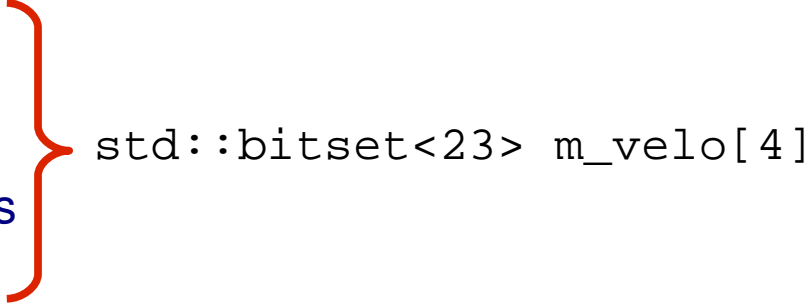
'hit-pattern'

- hit pattern: bit pattern that tells which layers are hit



- can be constructed from LHCbIDs
 - extract subdetector and 'layer' from LHCbID
 - only input from geometry: number of bits == number of layers (hardcoded)
- my main purpose: select overlap tracks for alignment trigger
 - but can also be used for ghost rejection

LHCbLernel/Kernel/HitPattern

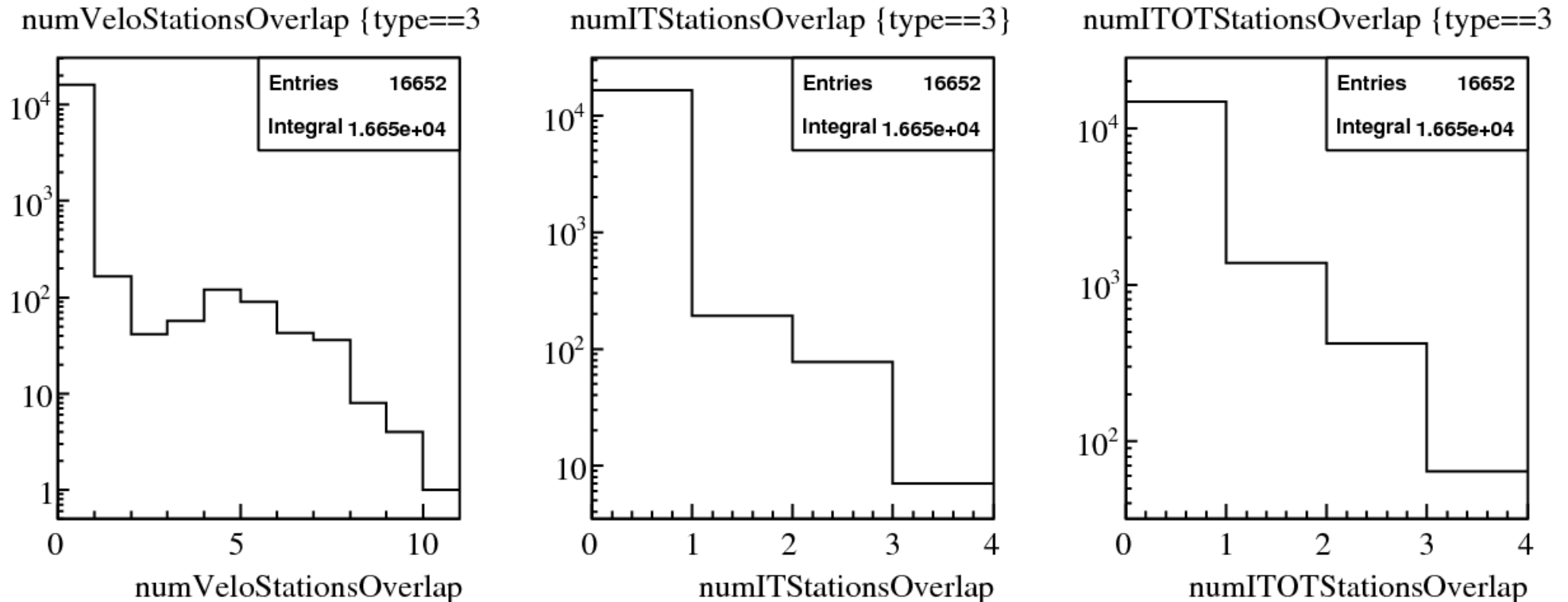
- new class in LHCb kernel, uses 'std::bitset' as bit patterns
 - constucted from std::vector<LHCbID>
 - contains hit patterns for all tracking detectors
 - does not know anything about real geometry, besides the number of layers
- example for VELO
 - separate R and phi → can count clusters
 - separate A side and C-side → can count overlaps
 - 21 layers + 2 pile uplayers

```
std::bitset<23> m_velo[4]
```
- other tracking detectors
 - TT: 1 pattern with 4 bits
 - IT: 4 patterns (top,bottom,A-side,C-side) with 12 bits
 - OT: 4 patterns (Q0...Q3) with 12 bits (e.g. only one bit per double layer)
 - MUON: 1 pattern of 5 bits
- besides the usual logical operations, std::bitset also has a 'count' function

overlap regions

- example: counting the number of overlap stations in the velo:

```
N = (( m_velo[VelORA] | m_velo[VelPhiA] ) &  
      ( m_velo[VelORC] | m_velo[VelPhiC] ) ).count() ;
```

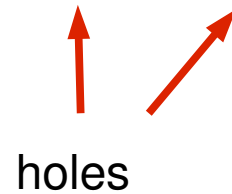


- about 1% of long tracks has overlap in ≥ 5 velo stations
- about 1% of long tracks goes through 2 boxes in at least one IT station
- about 10% of long tracks goes through OT and IT in at least one station

holes

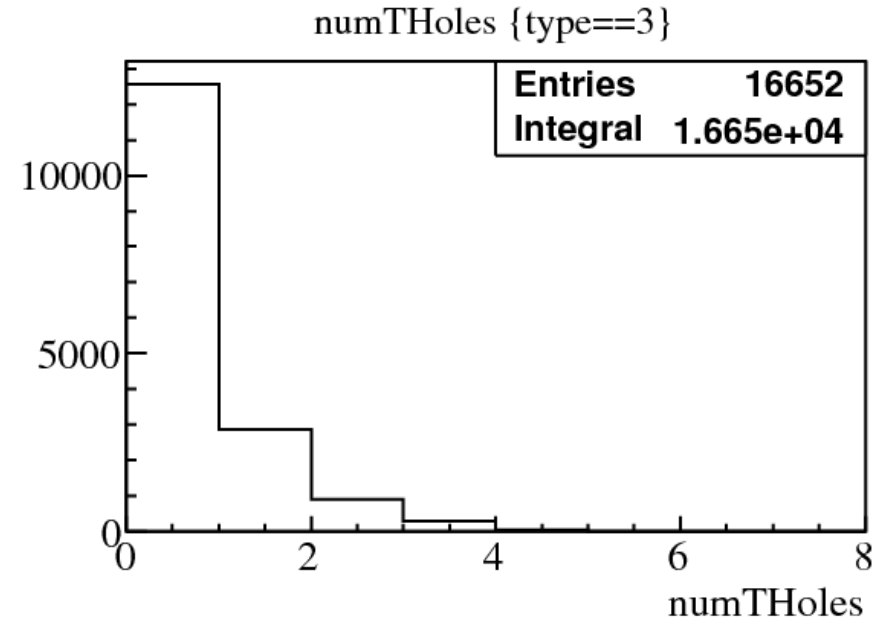
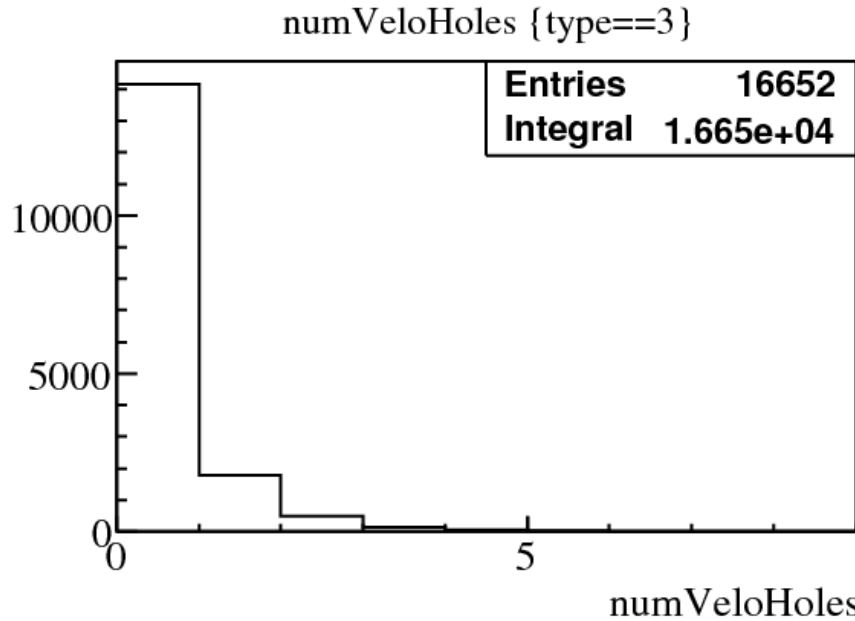
- hit pattern also useful for identifying 'holes' in the track

0001101111010

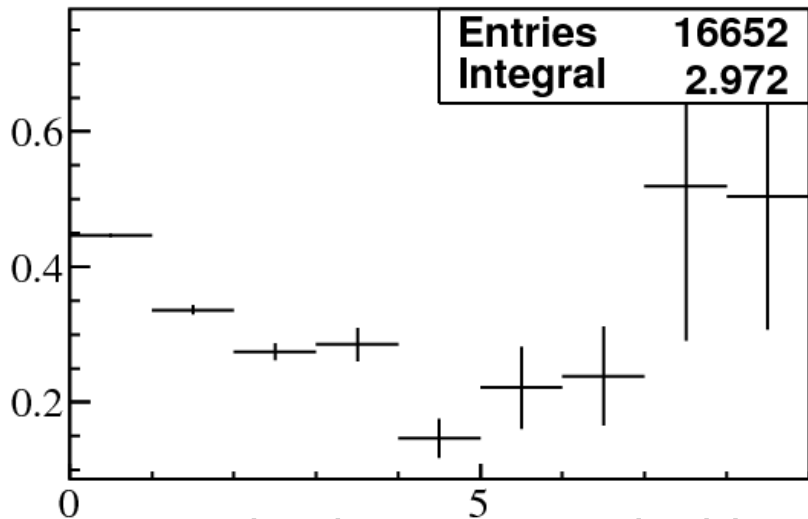


- could be useful for track selection
- note: not as reliable as a real 'ExpectedHits' tool
 - hit pattern doesn't know about real holes in detector
 - cannot identify missing hits at begin or end of segment
 - e.g. not very usefull if you want to find out if track is secondary or from V0
- best of both: fill hit 'expected' pattern from ExpectedHitTool (to be done)

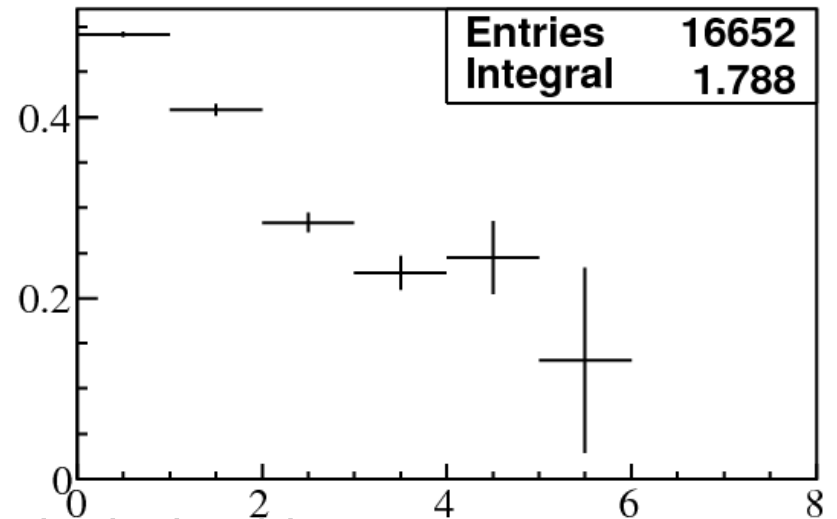
holes and chisquare



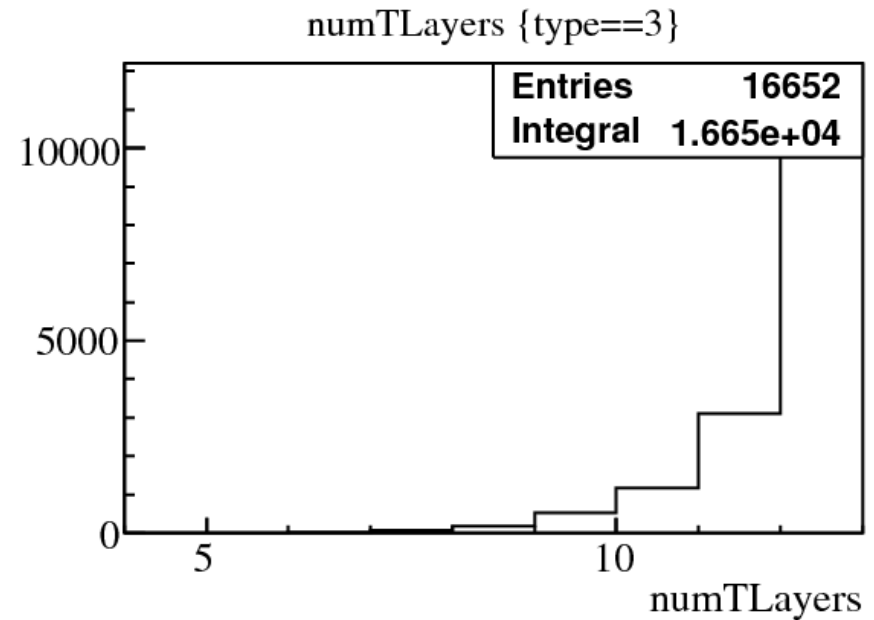
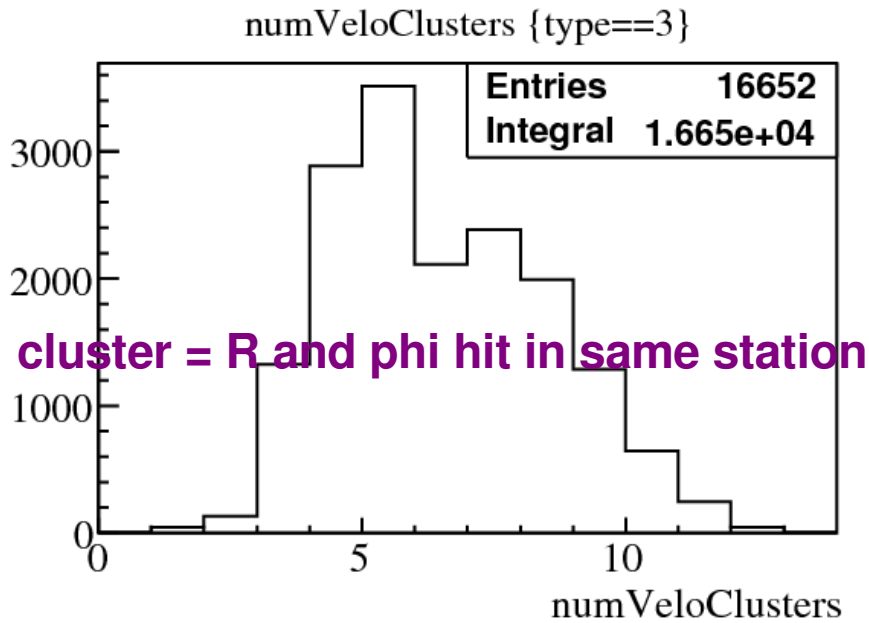
TMath::Prob(veloChi2,veloN dof):numVeloHoles {type=



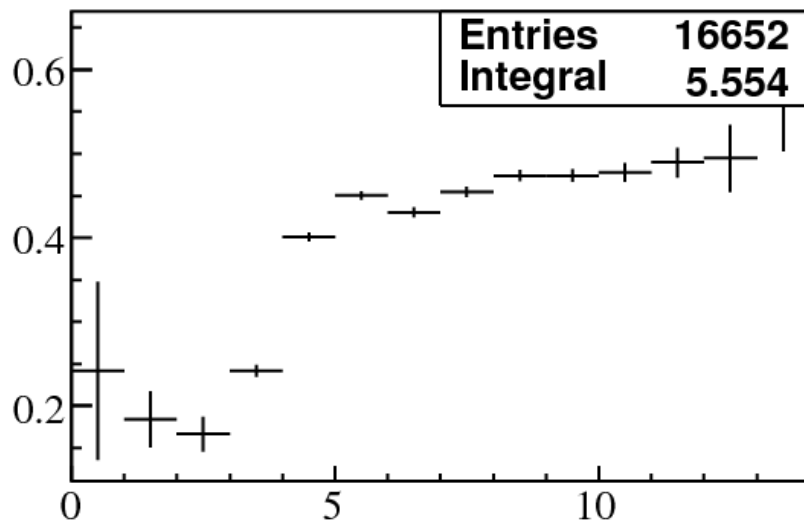
TMath::Prob(TChi2,TN dof):numTHoles {type==3}



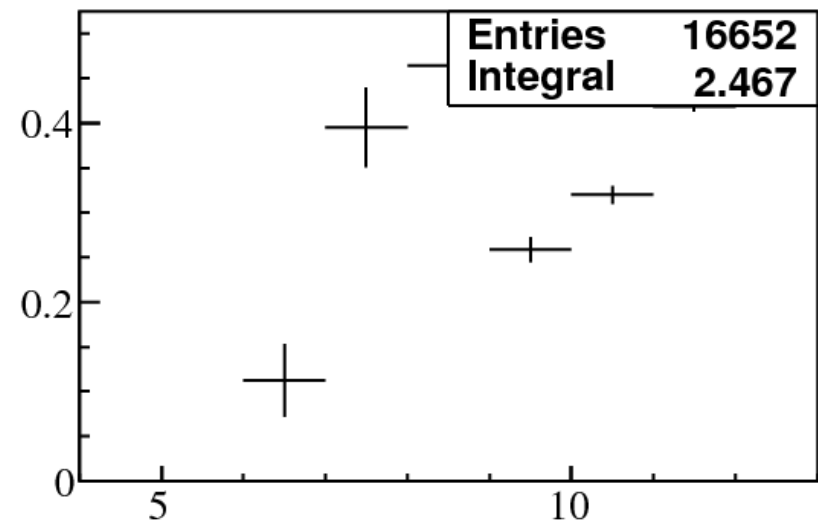
- large correlation between track chisquare and missing hits
- hypothesis: tracks with holes have high probability to be ghosts



TMath::Prob(veloChi2,veloNdof):numVeloClusters {type

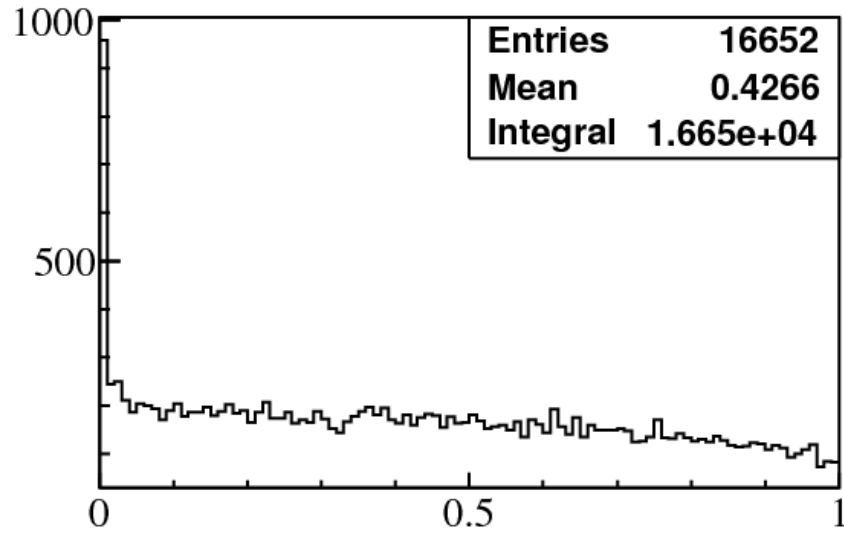


TMath::Prob(TChi2,TNdof):numTLayers {type==3}

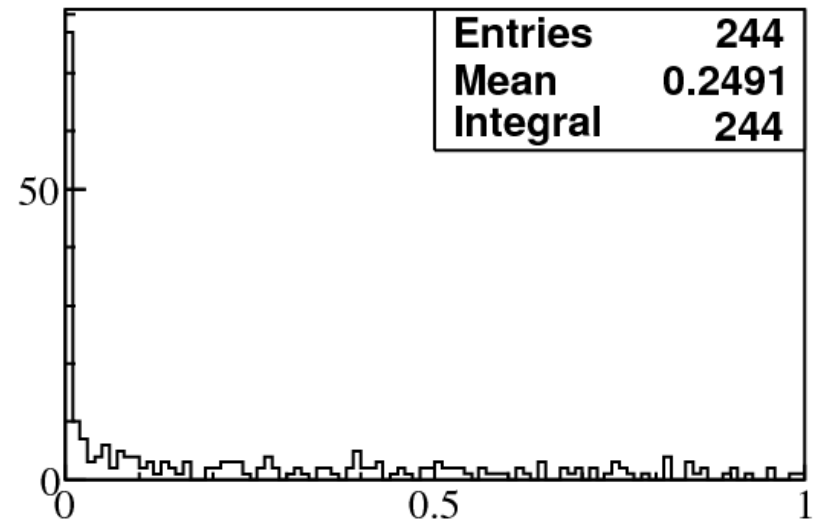


- but then ... also large correlation between number of VELO hits and chisquare

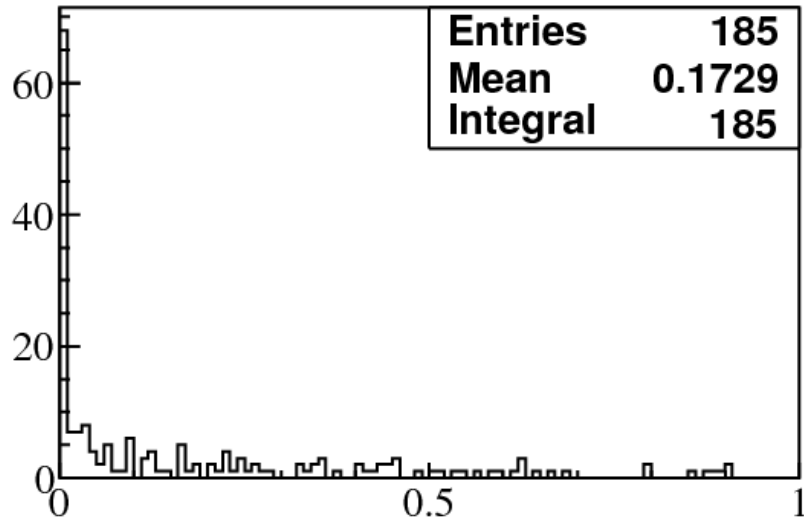
velo chisquare prob



≥ 2 velo holes



≤ 2 velo clusters



≤ 2 velo clusters or ≥ 3 velo holes

