

Pixel Tracking and Vertexing

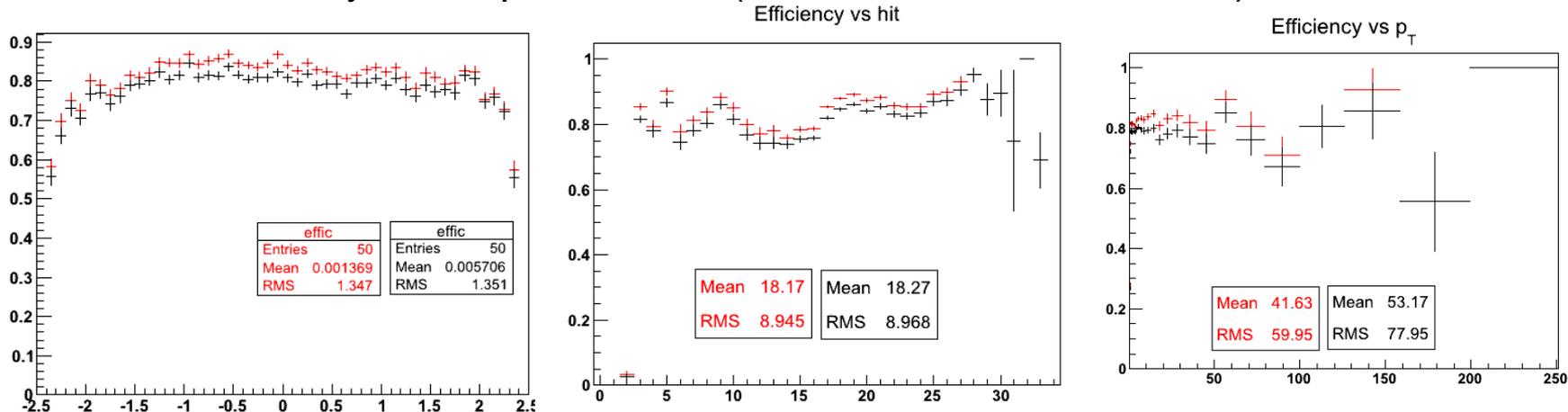
Suvadeep Bose

University of Nebraska Lincoln

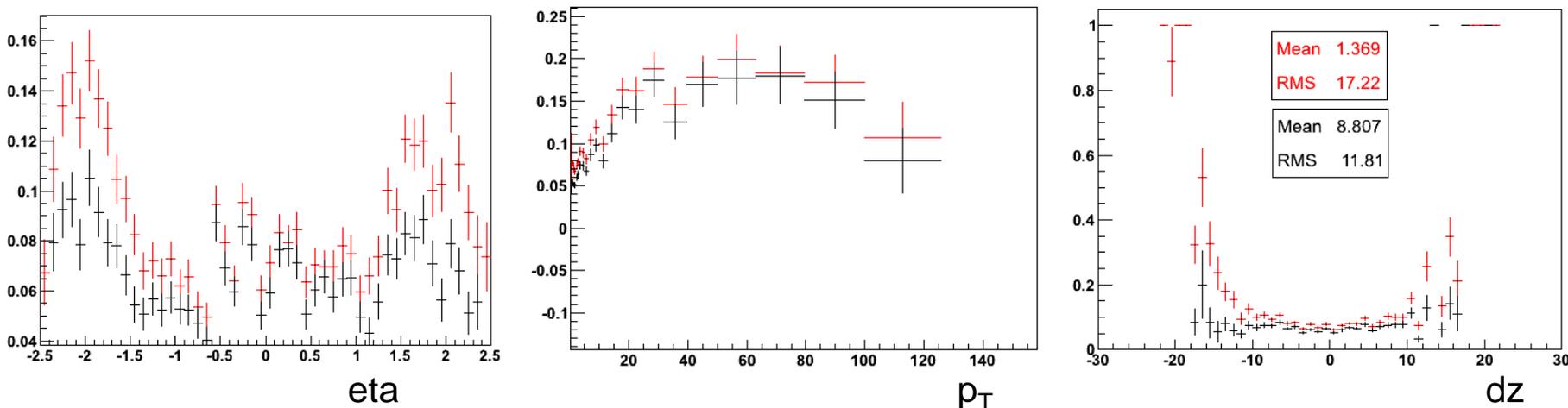
- ❑ Pixel stand-alone tracking and vertexing used both at **HLT and Offline**
- ❑ Pixel tracks are made from PixelTriplets and are used at **HLT** at level 2.5 to make **1D pixel vertices** that gives z-position of the vertex
- ❑ 1D pixel vertices are made using **Divisive algorithm**
- ❑ Aaron and I worked on improving that to 3D vertexing using Kalman Fitter – this though shown to have better timing performance is still not part of the cms machinery (resolution issue)
- ❑ **Offline**: general tracking pixel tracks used for seeding, and the same algorithm that is used to make primary vertices in general tracking (**Adaptive vertex fitter**) is used to make **3D pixel vertices** out of pixel tracks which are used both in the seeding and in the track selector

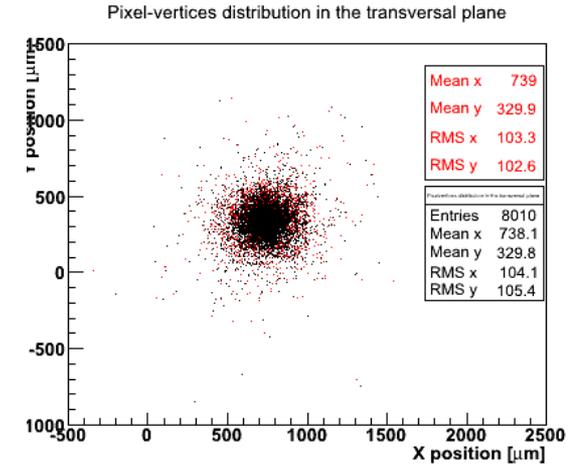
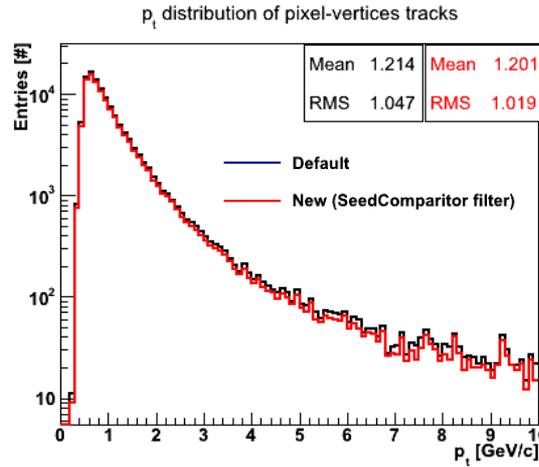
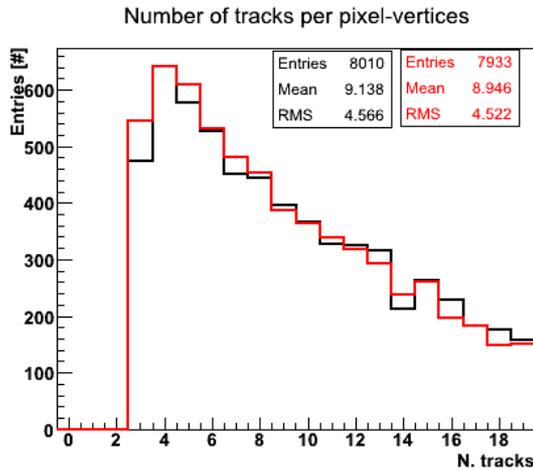
- Pixel tracking:
 - Some basic kinematic characteristics of selected tracks (multiplicity, p_T)
 - Resolution, Purity (Fake rates), Efficiency of pixel tracks (as a fn of p_T , η)
 - Perhaps – no of tracks vs. no of pile up interactions (eg. A. Venturi plots)
- Pixel Vertexing:
 - 3D pixel vertices (as used in Offline)
 - 1D pixel vertices (HLT) ?
 - Look at resolution (x,y,z), d_0 , dz + as a function of p_T , η
 - Pulls (d_0 , dz) as a function of η , p_T , no of tracks
- Latest tracking improvements:
 - Too many triplets -> SeedComparator filter (A. Venturi)
 - Pixel error parametrization (M. Konecki, SB)
- Timing performances ?
- The final plots should include the latest error estimates (to correct for pulls)
- 2011 Collision data to be compared with MC-Reco and MC-Truth (Simtrack)

Efficiency of the pixel tracks (reco matched to MCTruth)

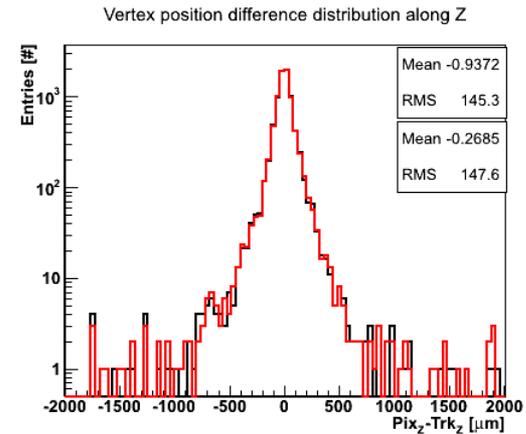
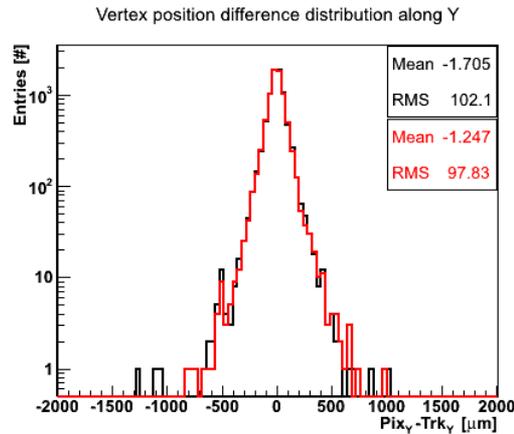
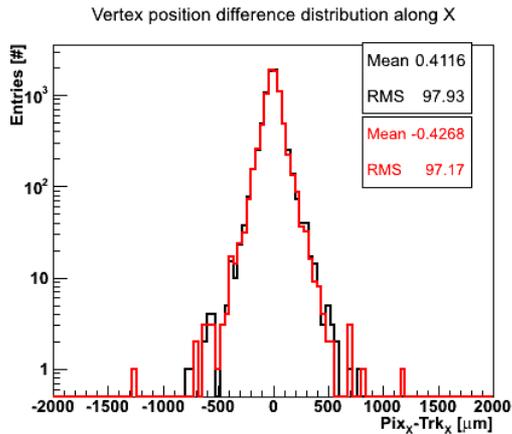


Fake rates as a function of p_T , eta, dz





3D Pixel vertices resolution: Data



- ❑ For the TRK POG paper things (that I worked on) were scattered. I compiled them together and currently running everything on the latest JSON with the full 2011 (at least the 2011A dataset) - 2.2 fb^{-1}
- ❑ Have to wind up the Pixel error parameterization work and run on 2011 data to obtain the new Pull distribution with the improvised errors.
- ❑ Can produce a 1st draft shortly but that may need update with the latest and greatest error estimations (+ other tracking improvement filters - decision pending?)



Back up

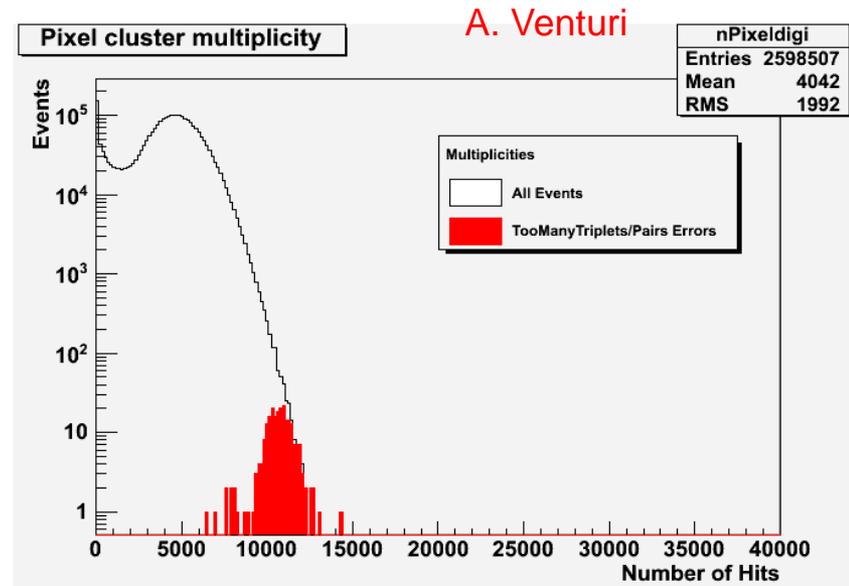


- The track reconstruction software is protected against too large multiplicity events to avoid crashes due to an excessive memory usage. There are three level of protections:

TooManyClusters, TooManyTriplets/TooManyPairs, TooManySeeds

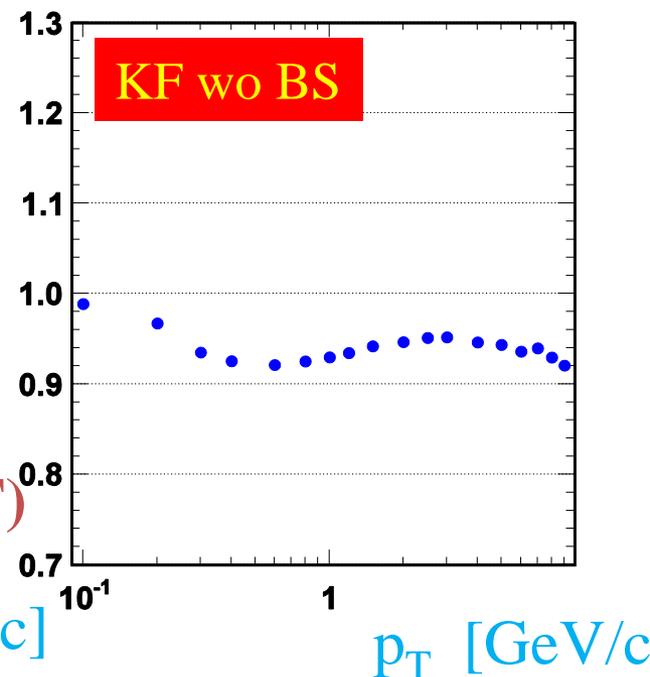
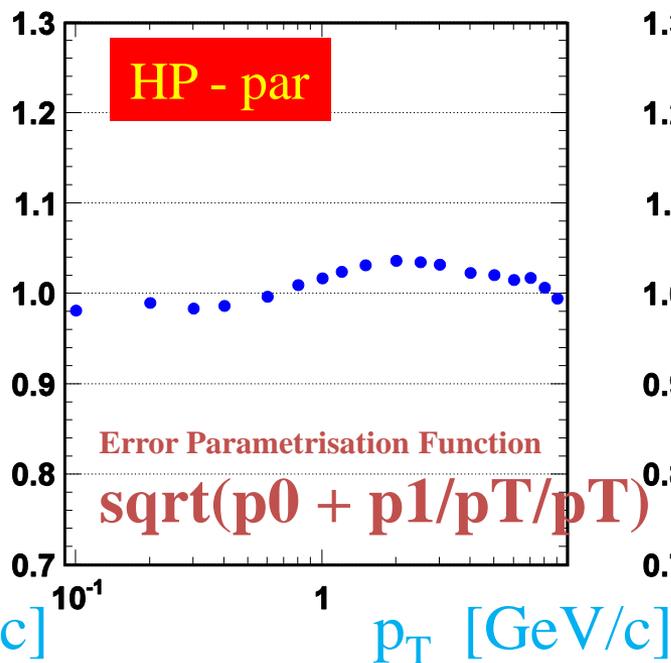
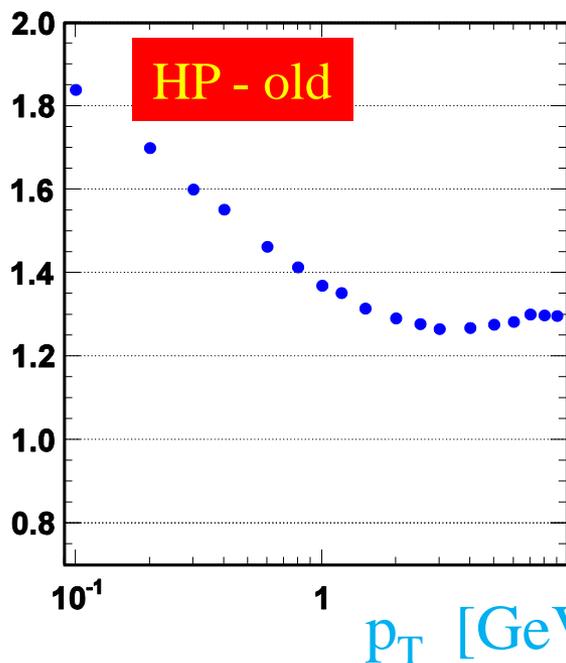
- TooManyTriplets/TooManyPairs: the seeding steps have an internal protection when the number of triplets or pairs used to produce the seeds exceed a threshold (presently 100k). When it happens the collection of triplets or pairs is reset and the seeding continues. Usually these errors affects each iteration independently since the number of triplets and pairs is very dependent on the seeding configuration.

- **SeedComparator filter:**
 - 1) to filter the incompatible triplets before the threshold on the total number of triplets is applied. This will remove all the "TooManyTriplets" errors
 - 2) to apply the same filter also in the reconstruction of the pixel tracks
 - tag currently is available in CMSSW_4_4_0_pre4 onwards

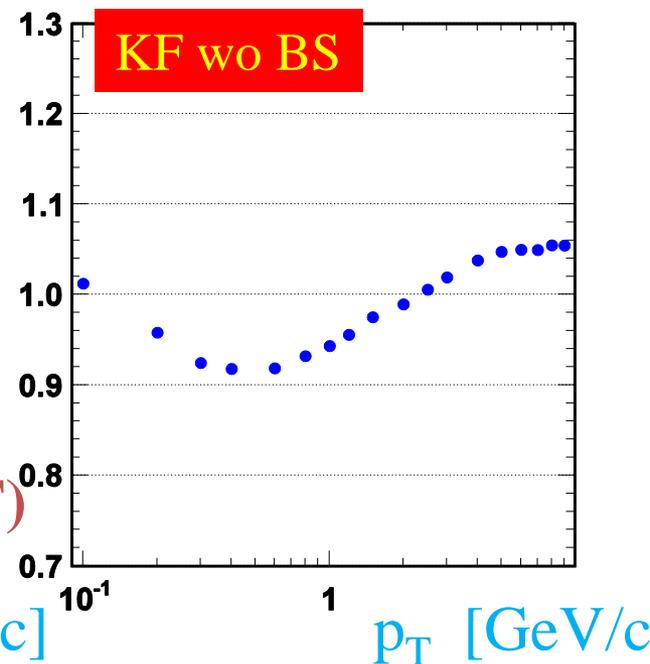
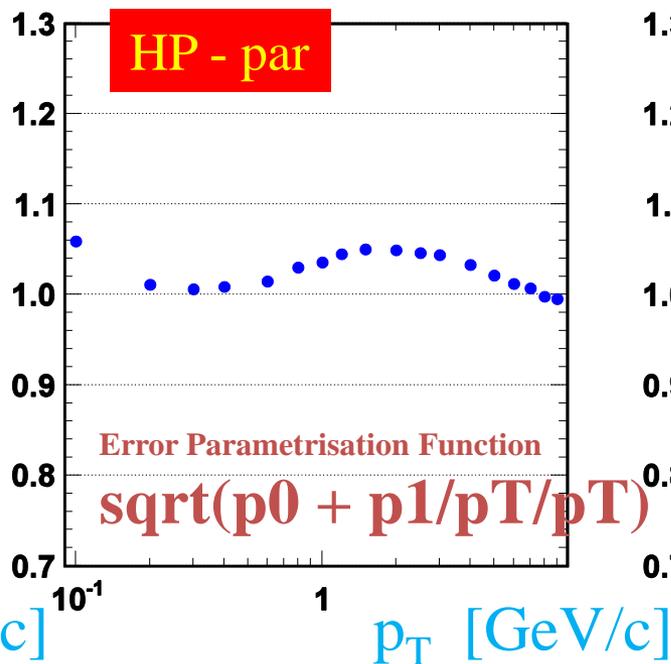
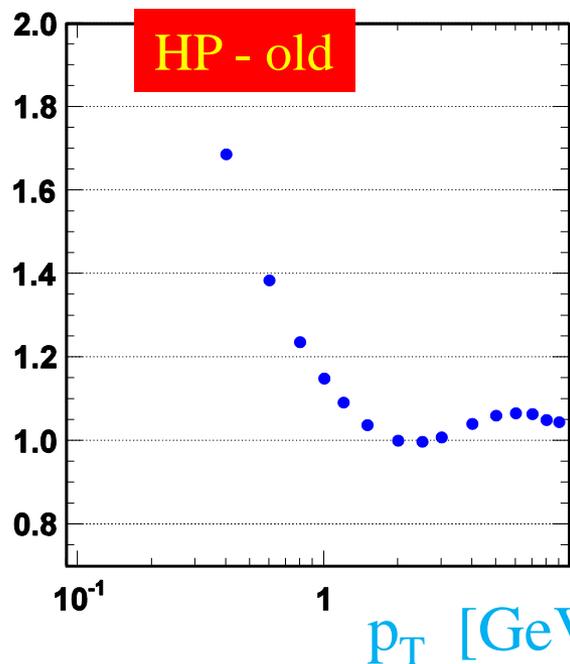


Pulls - TIP

Marcin's plot



Marcin's plot



- The configuration with **SeedComparator** filter being used for generating pixelTracks:

- CPU time (s)
- TimeReport 0.009 pixelTracks
- TimeReport 0.057 pixelVertices

- The **default** configuration:

- CPU time (s)
- TimeReport 0.008 pixelTracks
- TimeReport 0.058 pixelVertices

- **So the filter does not change the timing by a significant amount.**
- **This conclusion is based on running on data for 5000 events being run on same**
- **computing node.**