Track Extrapolation to the ECAL

Update

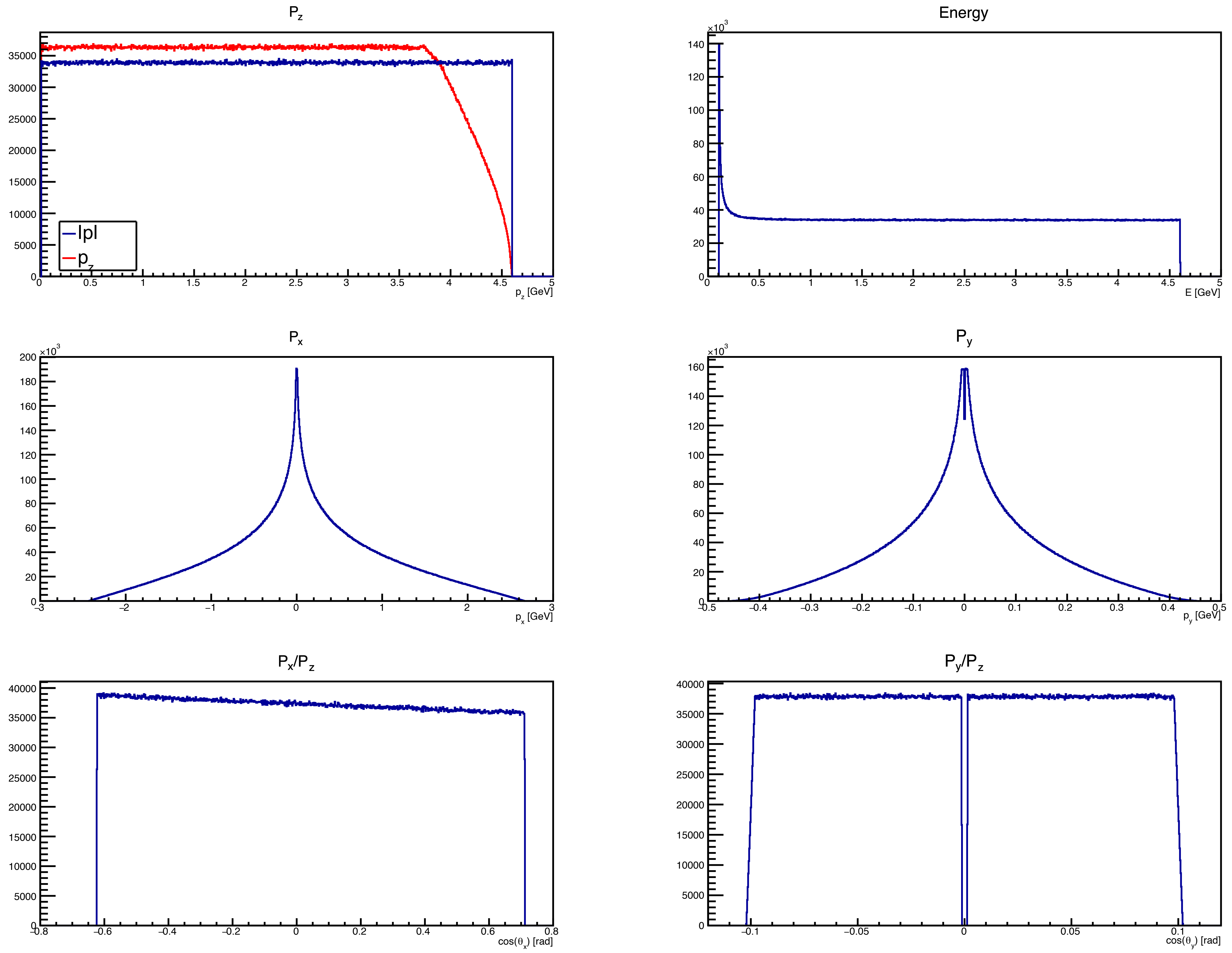
Maurik Holtrop & Lewis Wolf

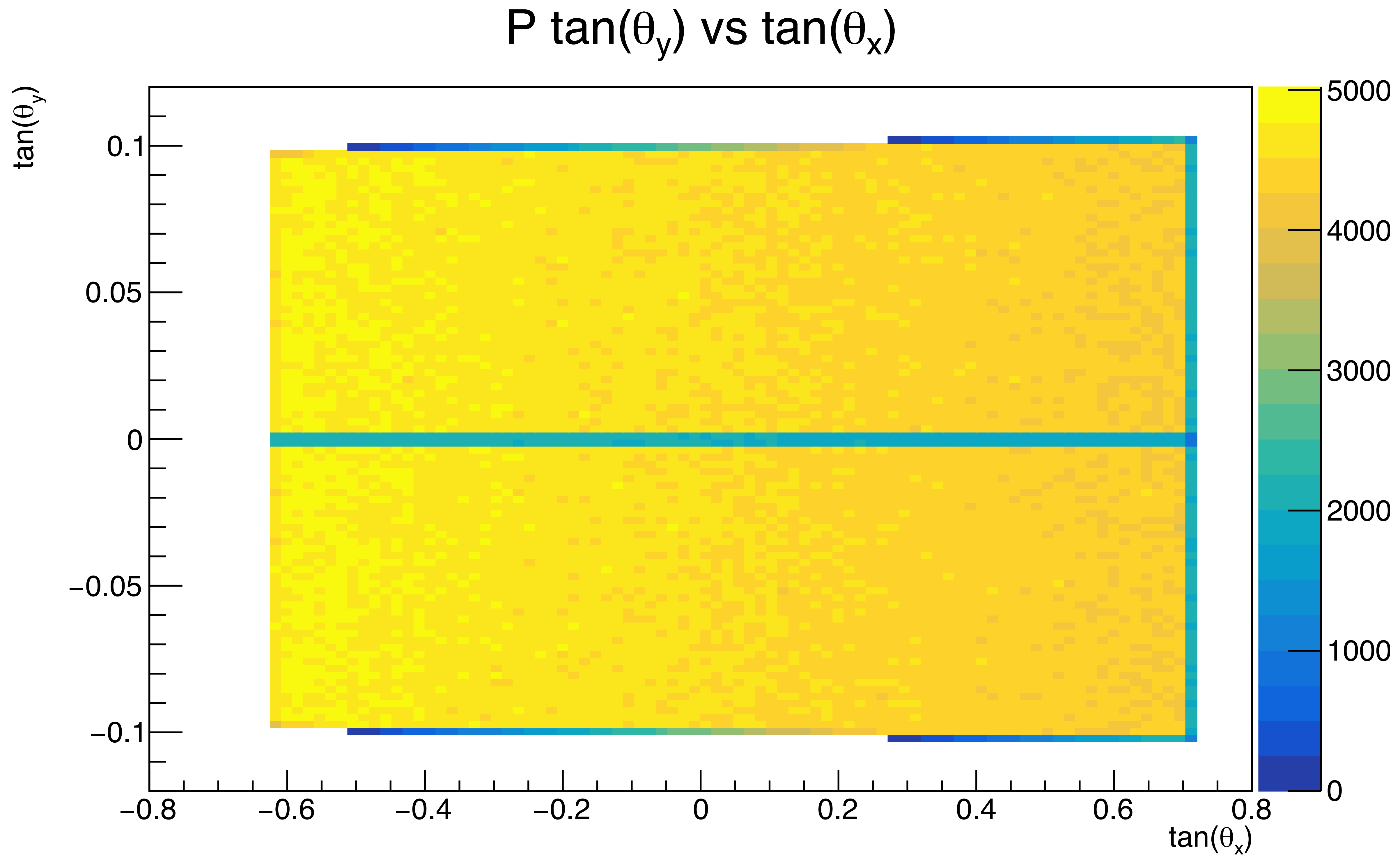
Introduction

- Lewis has already made several presentations on this subject, following presentation by Norman, and Alic (Weekly 4/21/2021)
- We observe a consistent discrepancy between the track extrapolated to the ECal and the hit in the ECal, even if the hit in the ECal is corrected for the shower depth, or if the comparison is with the scoring plane in front of the ECal (which has no shower depth.)
- Code that produced these plots are on GitHub:
    - https://github.com/mholtrop/Analysis/blob/develop/ECAL/ECAL_Muons.ipynb

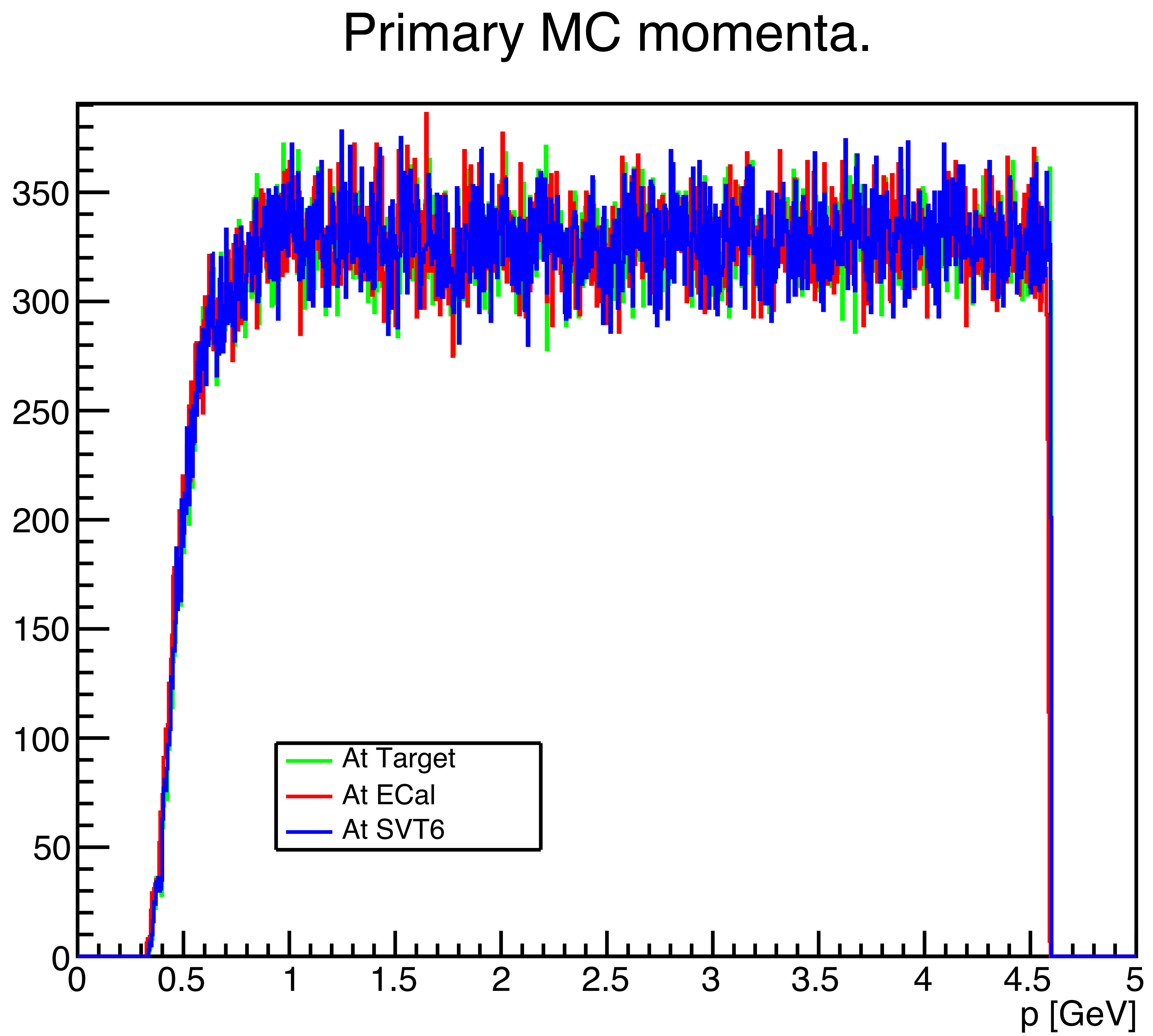
Input distribution

- As shown before by Lewis: a single negative muon is thrown per event by flat distribution in momentum and tan(θ_x), tan(θ_y), 5 M events were simulated.

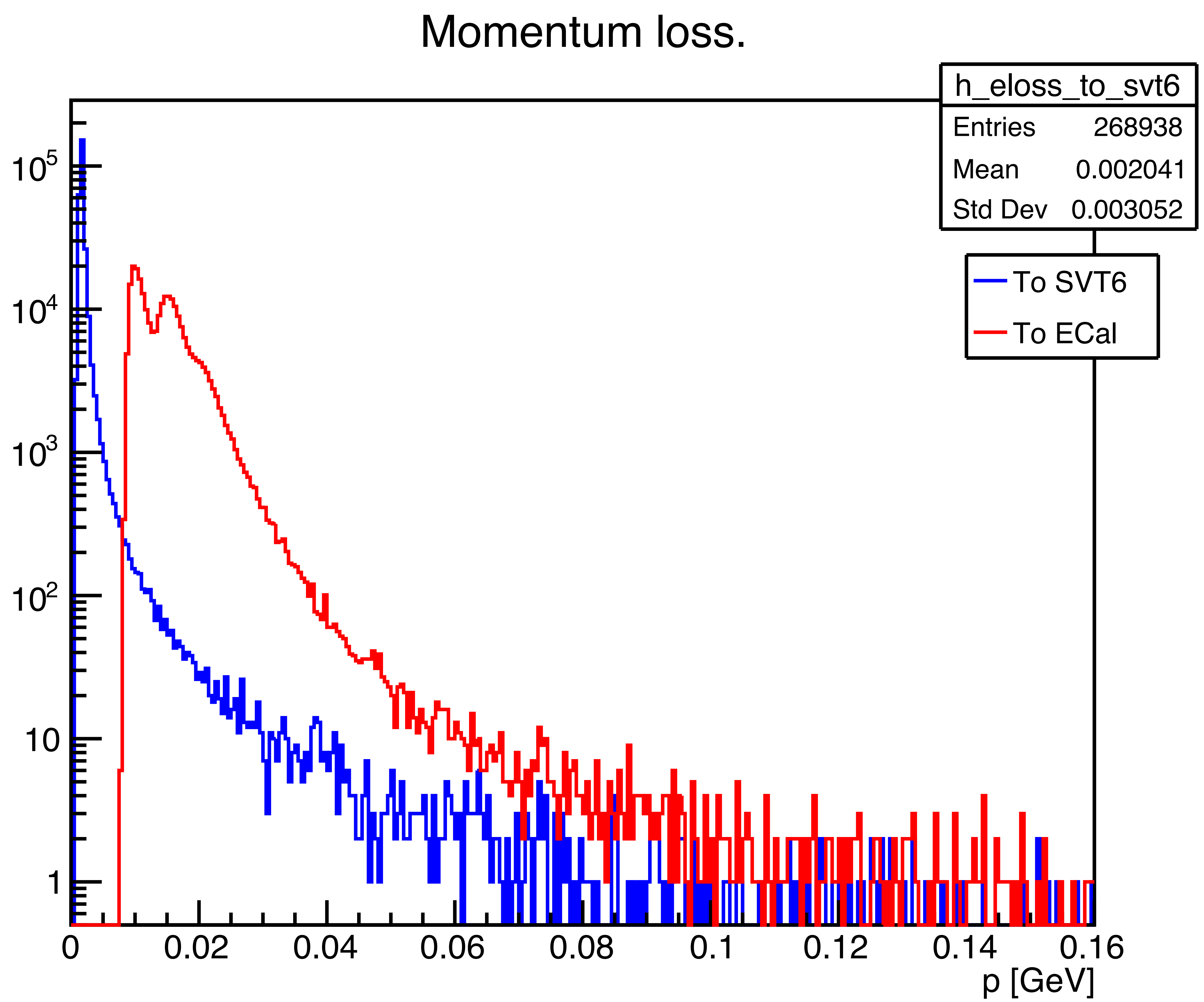




Muon distributions



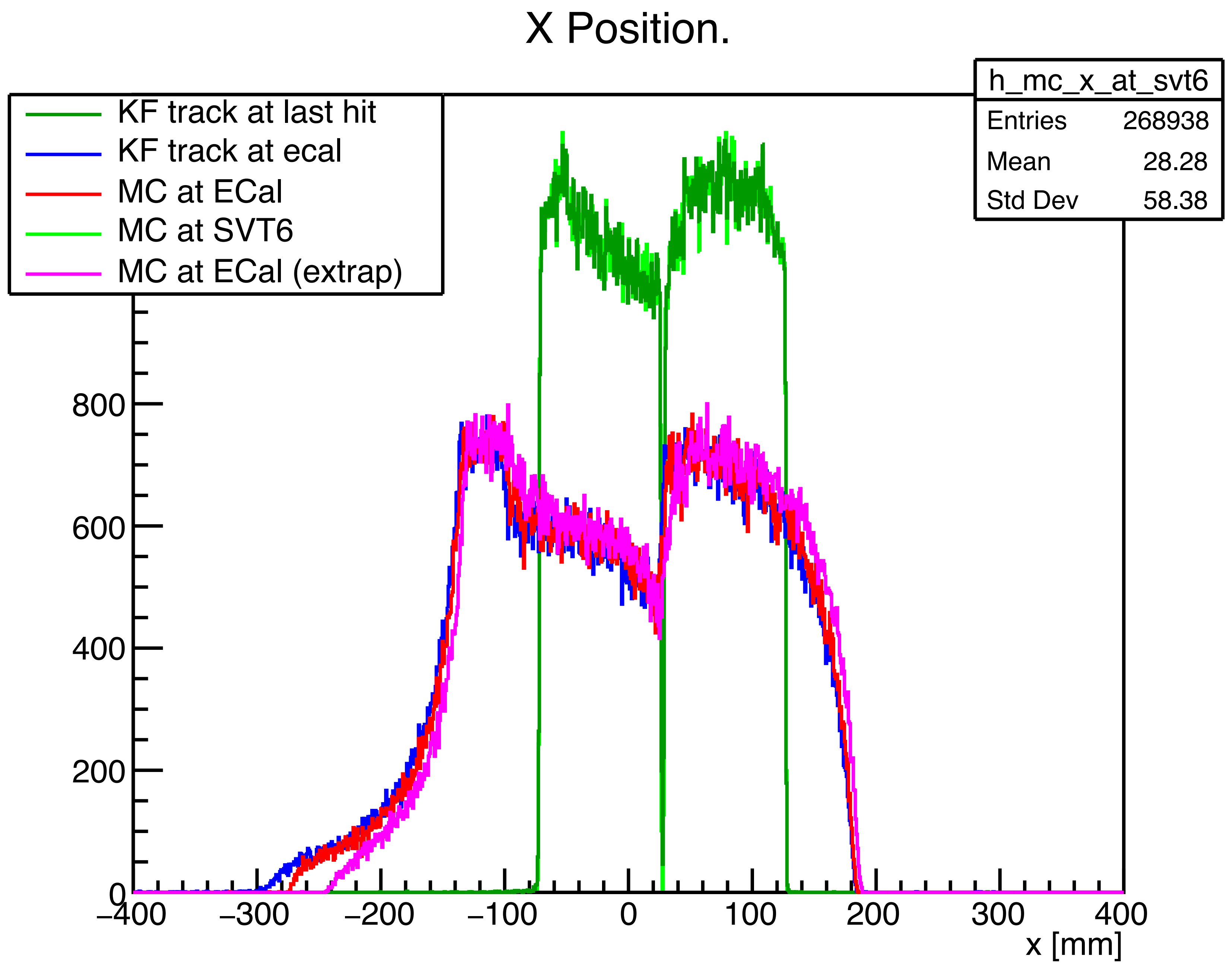
A fairly flat distribution in momentum of the muon all the way to the ECAL



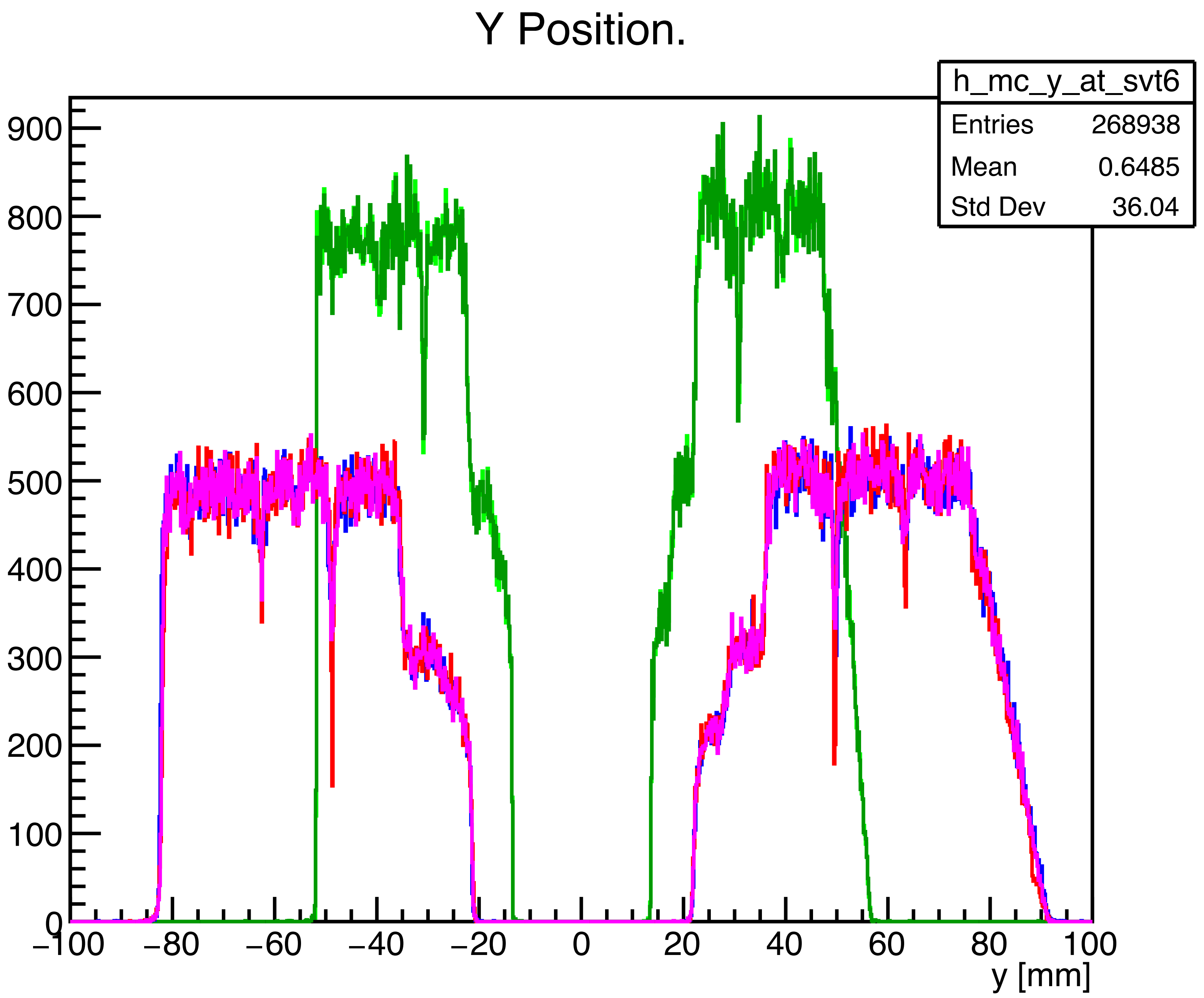
Not much momentum lost due to multiple scattering.
From last layer of the SVT to the ECal the vacuum exit plate is the main contribution to energy loss, and the production of secondaries.

Track at ECal

Plot the x and y positions of the track at the last tracker (green) and the ECal (blue)
Also plot the MC scoring plane position at the last tracker (light green) and ECal (red)
Extrapolate from last tracker scoring plane to ECal scoring plane, *using the MC scoring information only.* (magenta).



X Position.



Y Position.

NOTE:

There are strict cuts on this data:

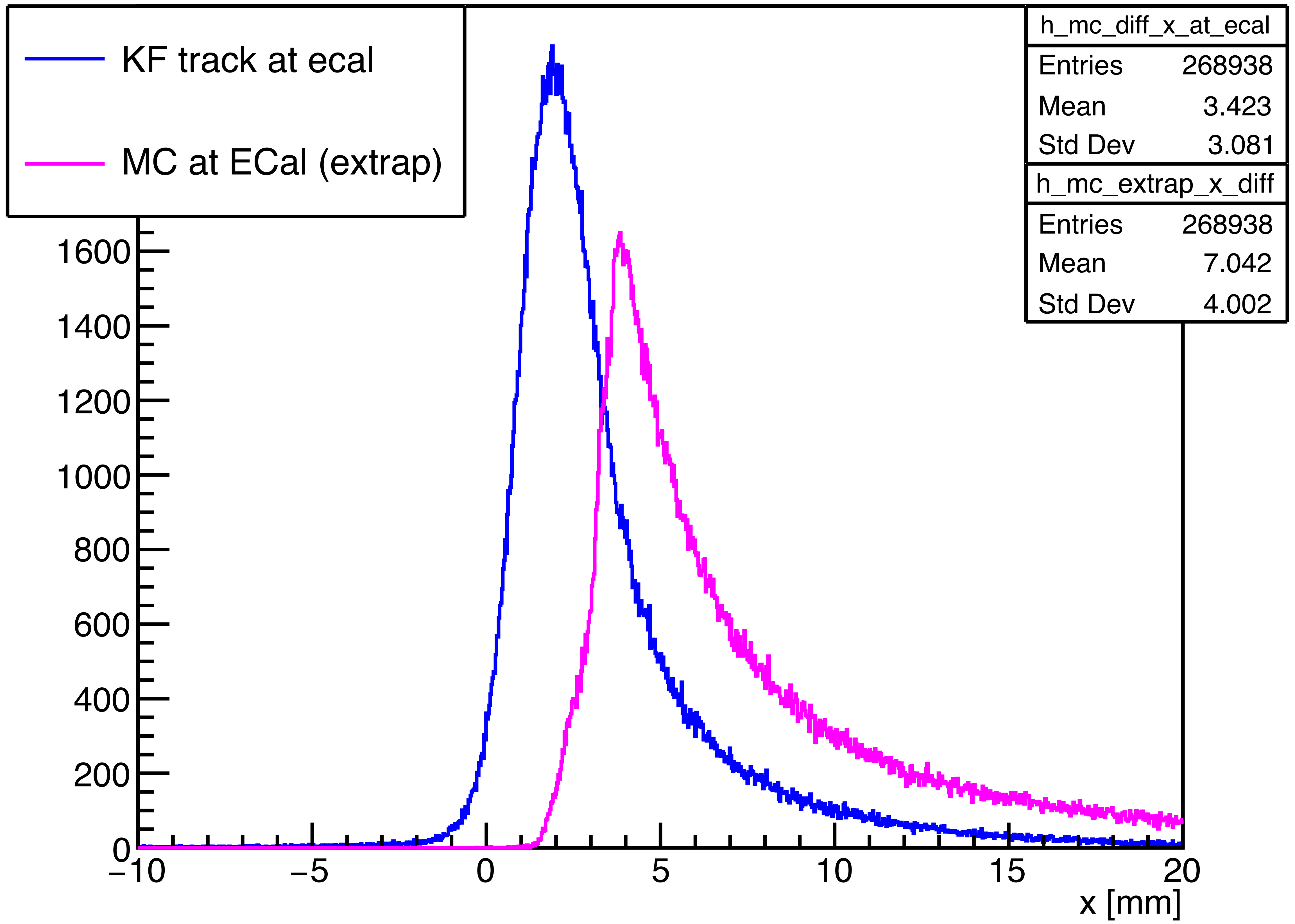
```
Select only 1 Primary MC particle: pass=2560892    all=4999900    -- eff=51.22 % cumulative eff=51.22 %
Select only 1 KF or GBL track.:    pass=1017929    all=2560892    -- eff=39.75 % cumulative eff=20.36 %
Select only 1 MC particle at ECal: pass=858624     all=1017929    -- eff=84.35 % cumulative eff=17.17 %
Select only 1 ECal cluster:        pass=441478     all=858624     -- eff=51.42 % cumulative eff=8.83 %
Select only 1 MC particle at SVT6: pass=268938     all=441478     -- eff=60.92 % cumulative eff=5.38 %
```

This causes subtle structure in the position distributions.

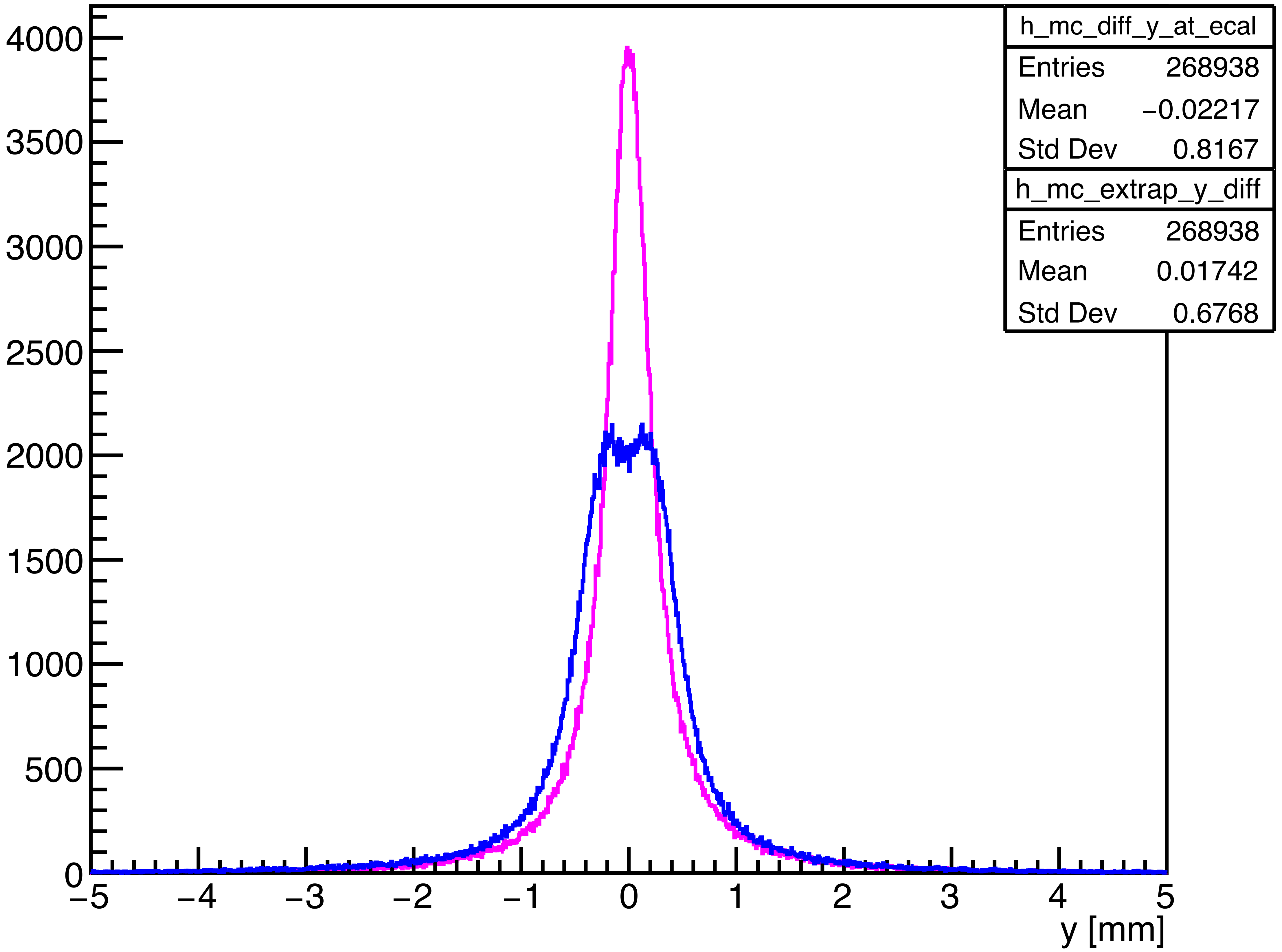
Compare position at ECal

Track at the ECal compared to MC scoring plane at ECal scoring plane (blue).
Extrapolation from last tracker scoring plane to ECal scoring plane (magenta)
The extrapolation is simply a straight line in the direction of the MC truth momentum to the ECal scoring plane.

X difference.

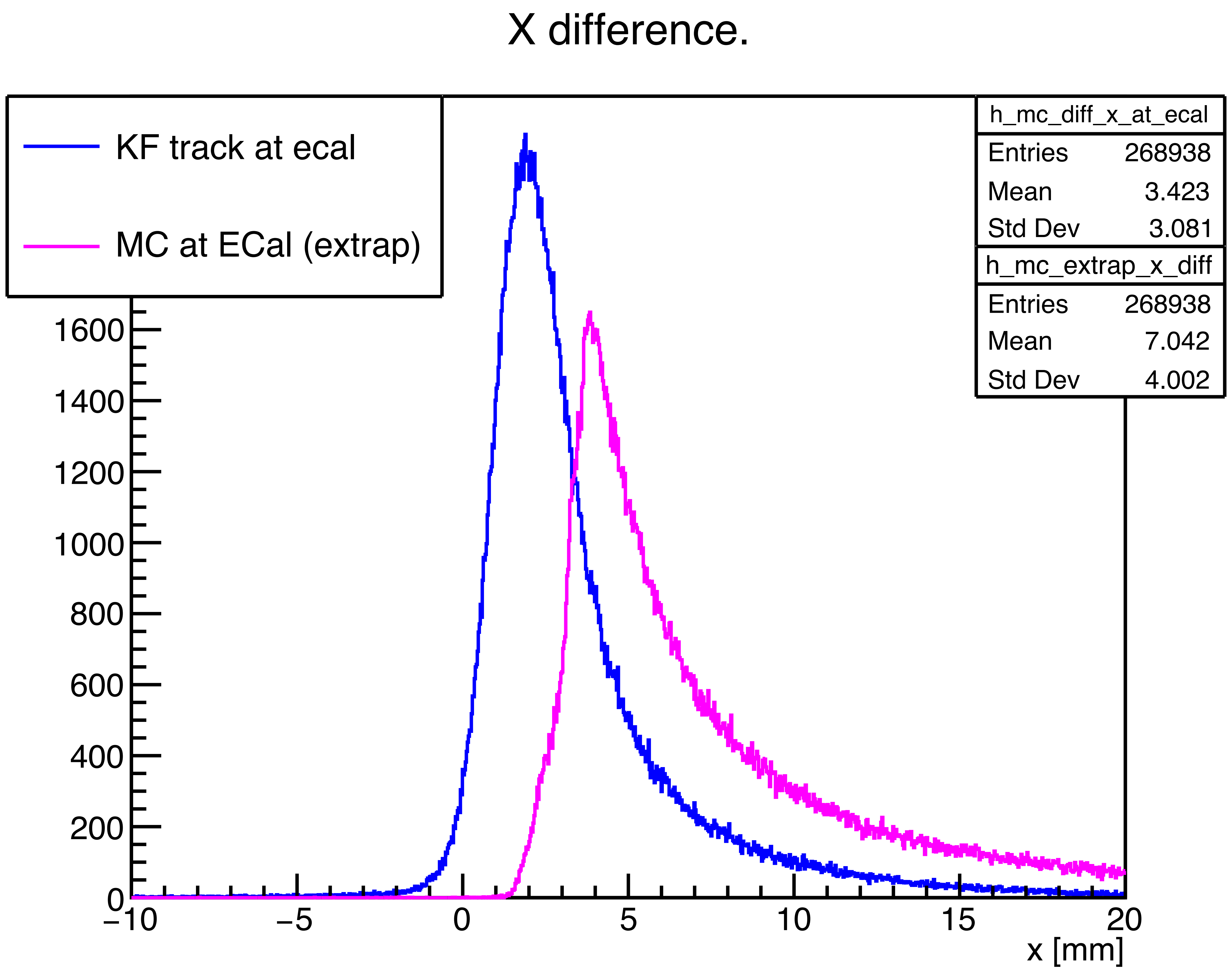


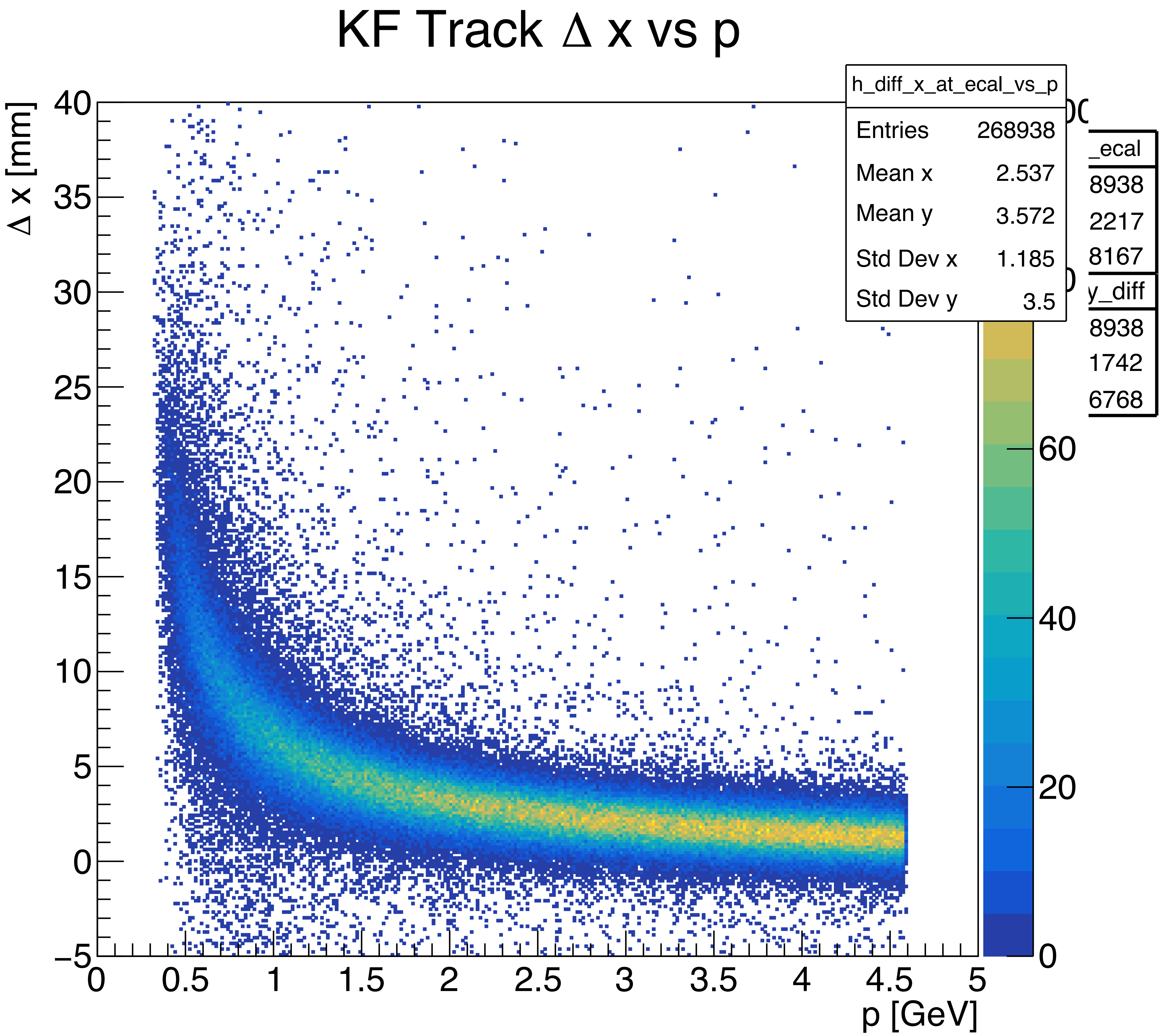
Y difference.



Compare position at ECal

Track at the ECal compared to MC scoring plane at ECal scoring plane (blue).
Extrapolation from last tracker scoring plane to ECal scoring plane (magenta)
The extrapolation is simply a straight line in the direction of the MC truth momentum to the ECal scoring plane.



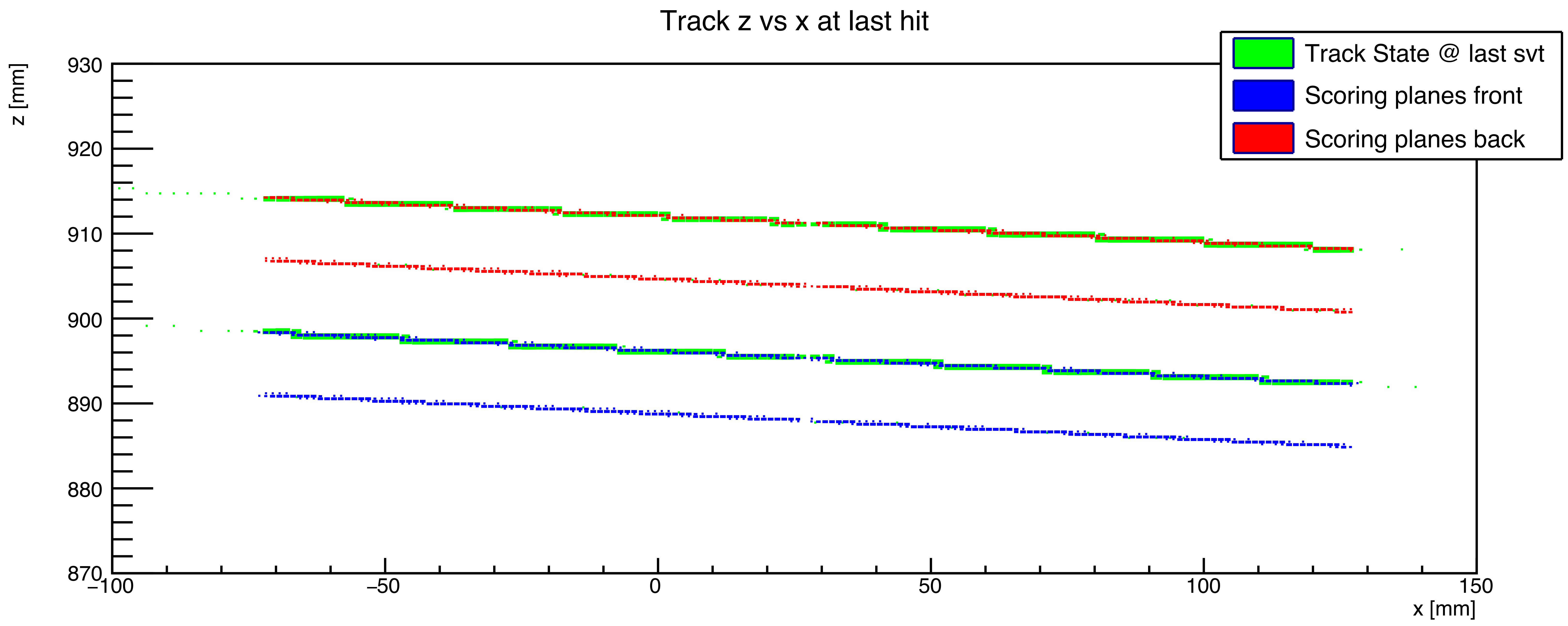


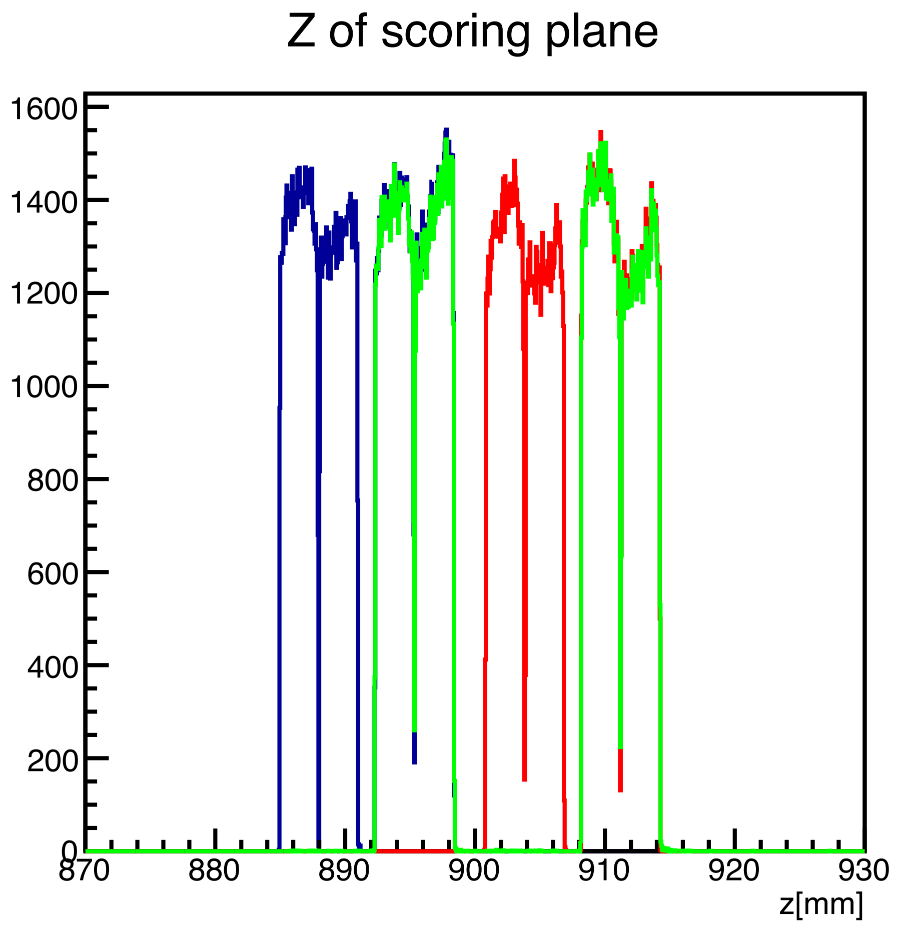
Track at last SVT

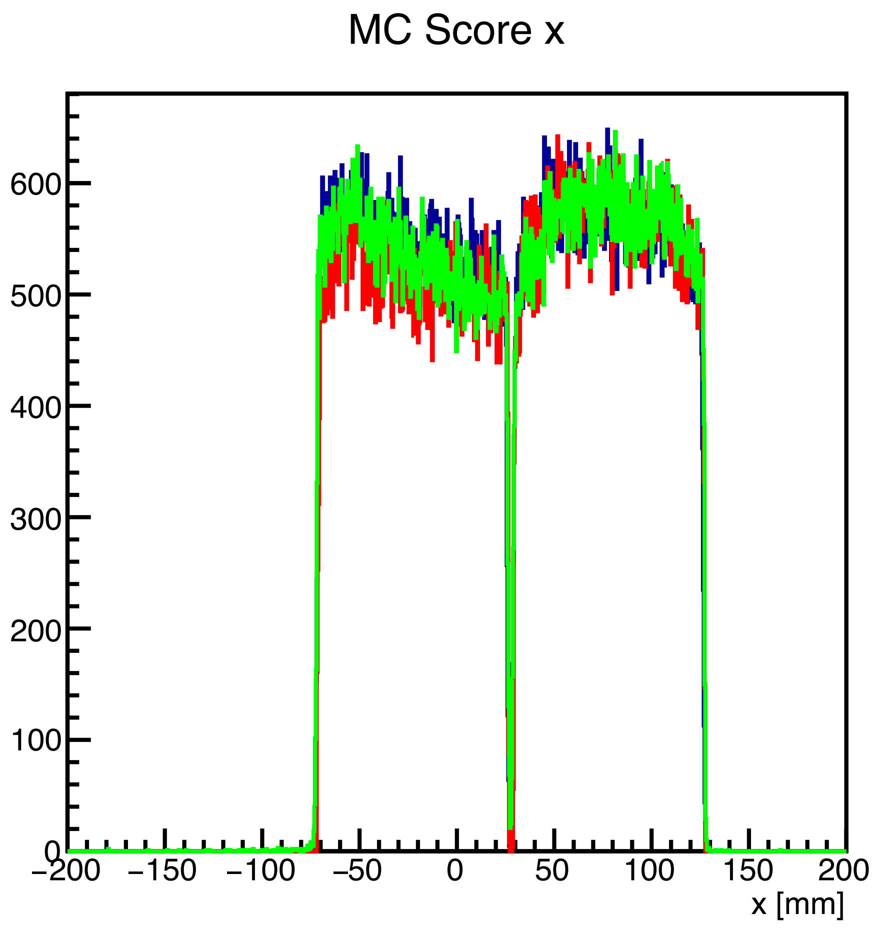
At the last tracker, there are 4 scoring planes, 2 for the top tracker, and 2 for the bottom tracker. The tracker, and the scoring planes, are rotated around the y-axis by the rotation of the tracking volume.

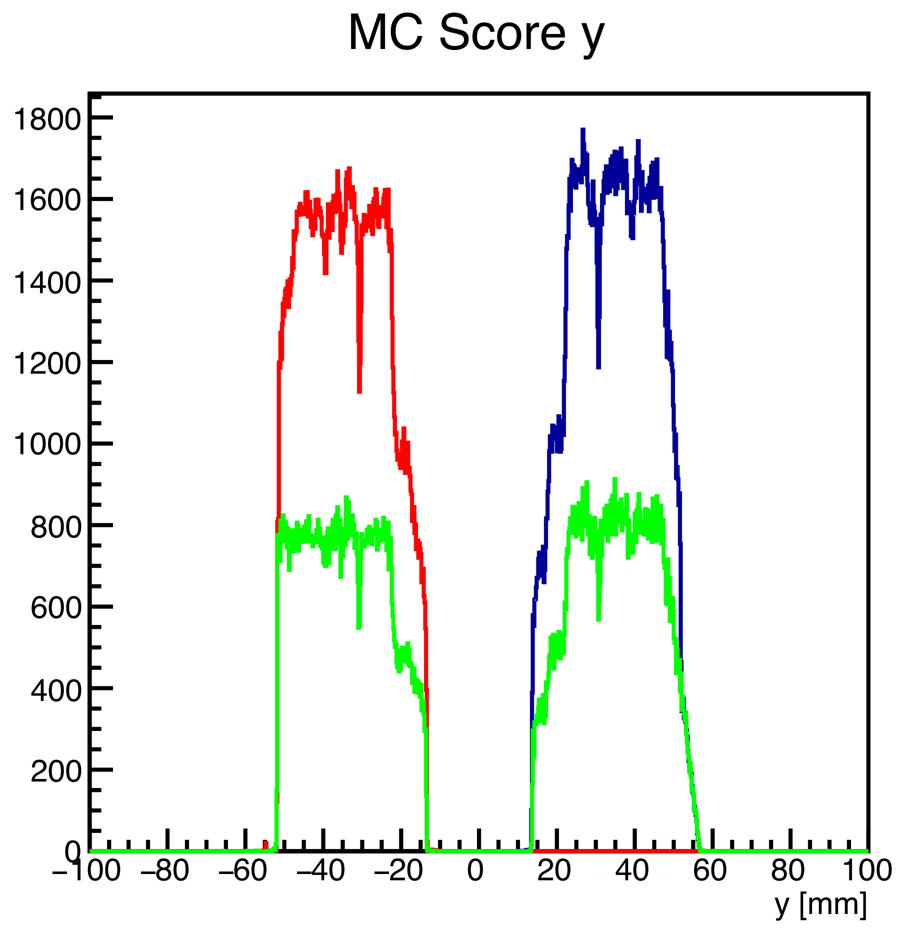
This arrangement is shown on the right.

The scoring plane toward the back is at the same location as the track state "at last hit", which is the track extrapolated to the last tracking plane.









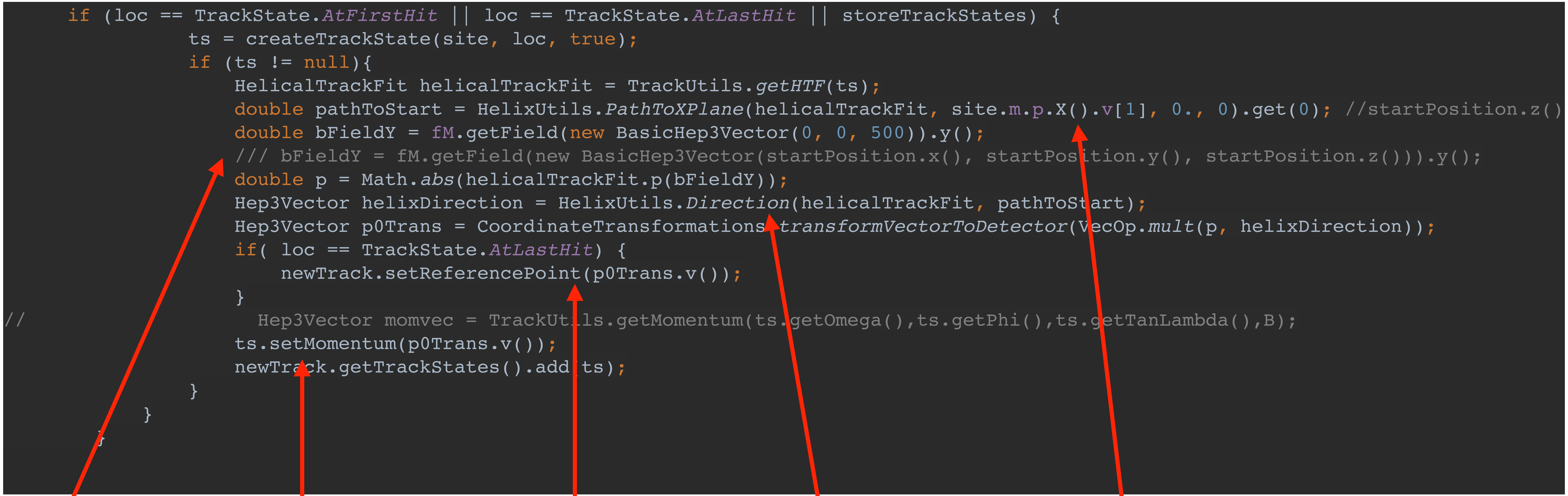
Decoding Track States

Results of a long debugging session with PF:

Correctly decoding the track states computed in hps-java is *very* confusing:

- The Kallman Filter code *always* uses the magnetic field strength at the center of the detector to convert (px, py, pz) to track state omega.
- The Kallman Filter code translates the reference point for each track segment to the origin (ask PF if this is correct phrasing). So the reported phi and tan_lambda are with respect to the track at the origin (so *not* at the point of reporting the track state.)
- Correctly interpreting a track state can only really be done in hps-java code.
- The track state has a momentum (px, py, pz) class variable, but unfortunately *this is not written out to the slcio file.*
- The (x, y, z) position of the track at the track state location is written correctly.
- The only recourse to get the track state momentum is to hack hps-java.

Hack hps-java



```
   if (loc == TrackState.AtFirstHit || loc == TrackState.AtLastHit || storeTrackStates) {
        ts = createTrackState(site, loc, true);
        if (ts != null){
          HelicalTrackFit helicalTrackFit = TrackUtils.getHTF(ts);
          double pathToStart = HelixUtils.PathToXPlane(helicalTrackFit, site.m.p.X().v[1], 0., 0).get(0); //startPosition.z()
          double bFieldY = fM.getField(new BasicHep3Vector(0, 0, 500)).y();
          /// bFieldY = fM.getField(new BasicHep3Vector(startPosition.x(), startPosition.y(), startPosition.z())).y();
          double p = Math.abs(helicalTrackFit.p(bFieldY));
          Hep3Vector helixDirection = HelixUtils.Direction(helicalTrackFit, pathToStart);
          Hep3Vector p0Trans = CoordinateTransformations.transformVectorToDetector(VecOp.mult(p, helixDirection));
          if( loc == TrackState.AtLastHit) {
            newTrack.setReferencePoint(p0Trans.v());
          }
//         Hep3Vector momvec = TrackUtils.getMomentum(ts.getOmega(),ts.getPhi(),ts.getTanLambda(),B);
          ts.setMomentum(p0Trans.v());
          newTrack.getTrackStates().add(ts);
        }
      }
    }
```

B field at
center of magnet

Hacked version
of BaseTrackState
allows directly setting
the momentum vector
useful for debugging

Put the momentum
at the last tracking plane
in the *reference point* of the track
just to get it into the lcio file.
reference point is otherwise
always [0,0,0]

Requires hack of lcio

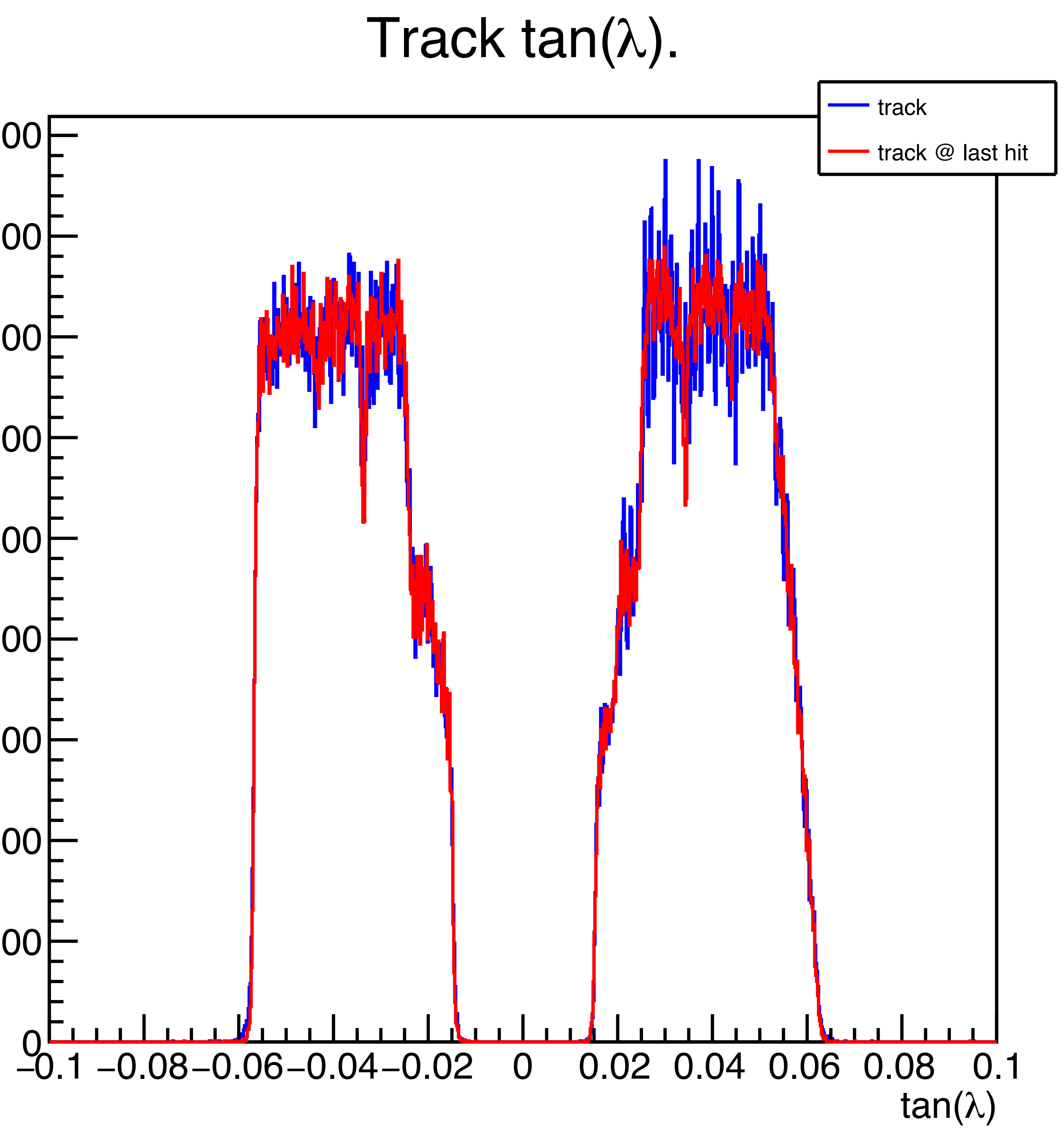
Get the *direction of the helix*
at the point of the track state

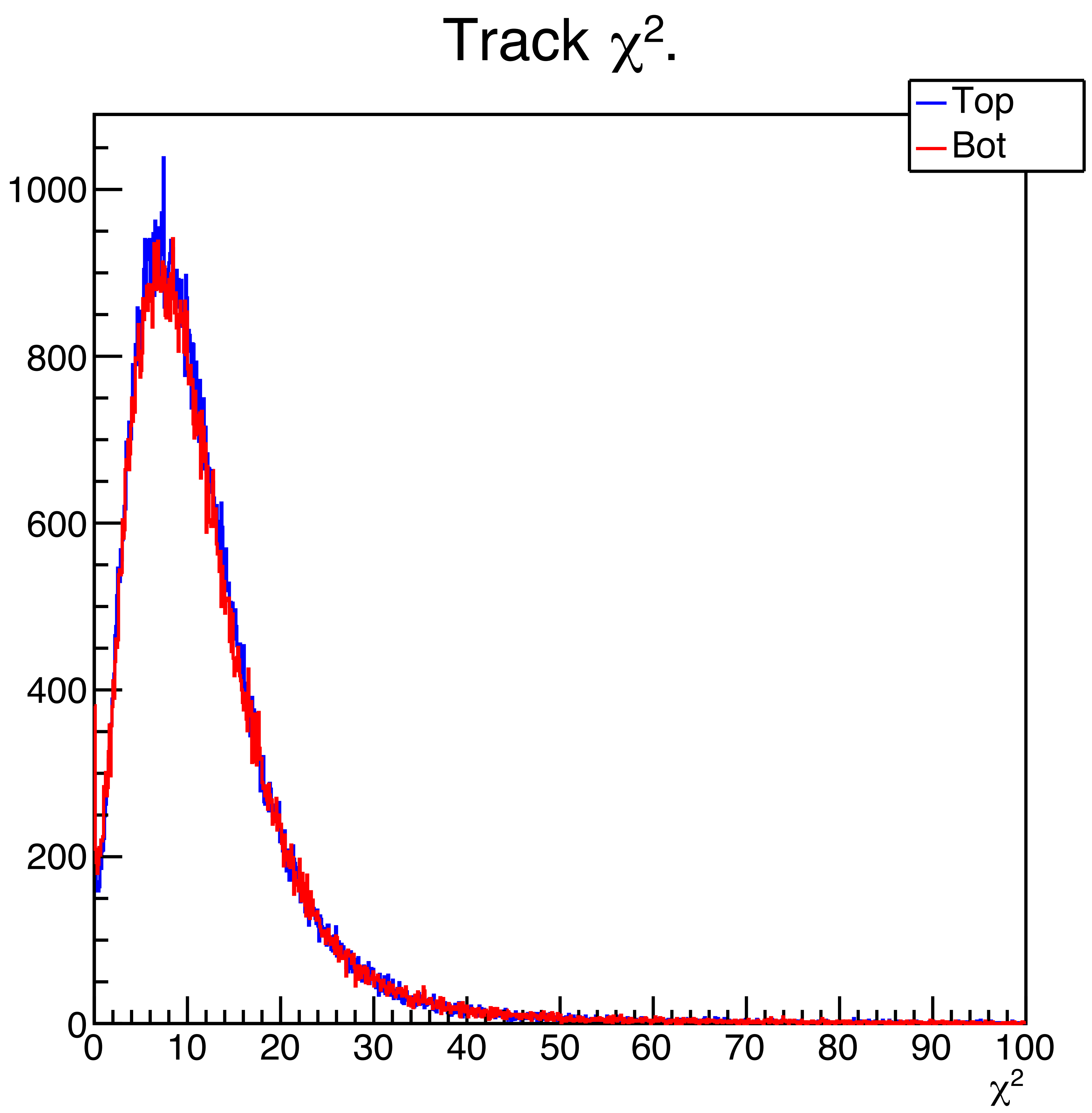
That is the z position
of the track state.

In hps-java git branch: iss2017_mom_with_KF_track_state

Compare Track and Scoring Plane

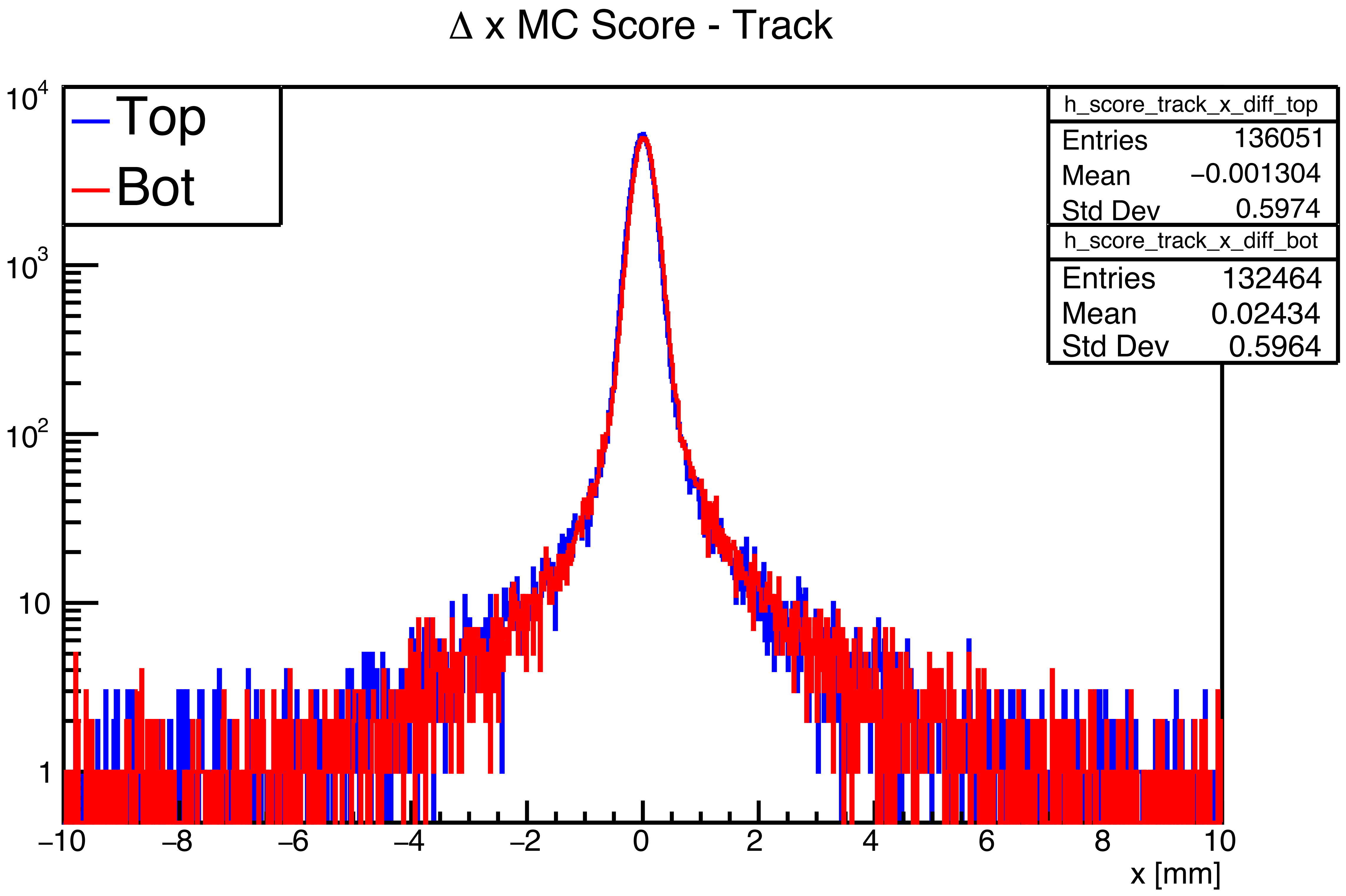
All the filtering made of very clean tracks.

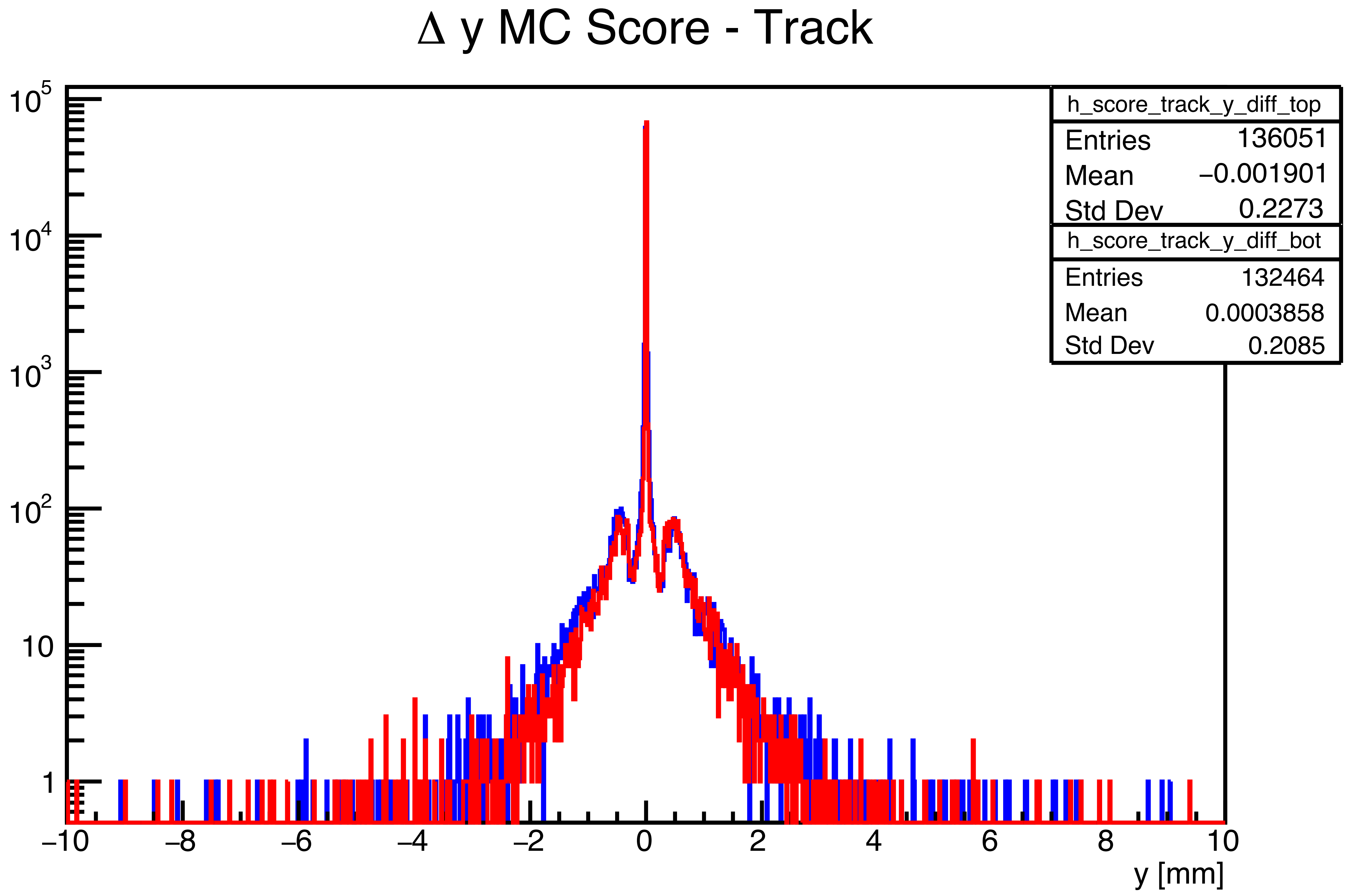

Track tan(λ).


Track χ^2.

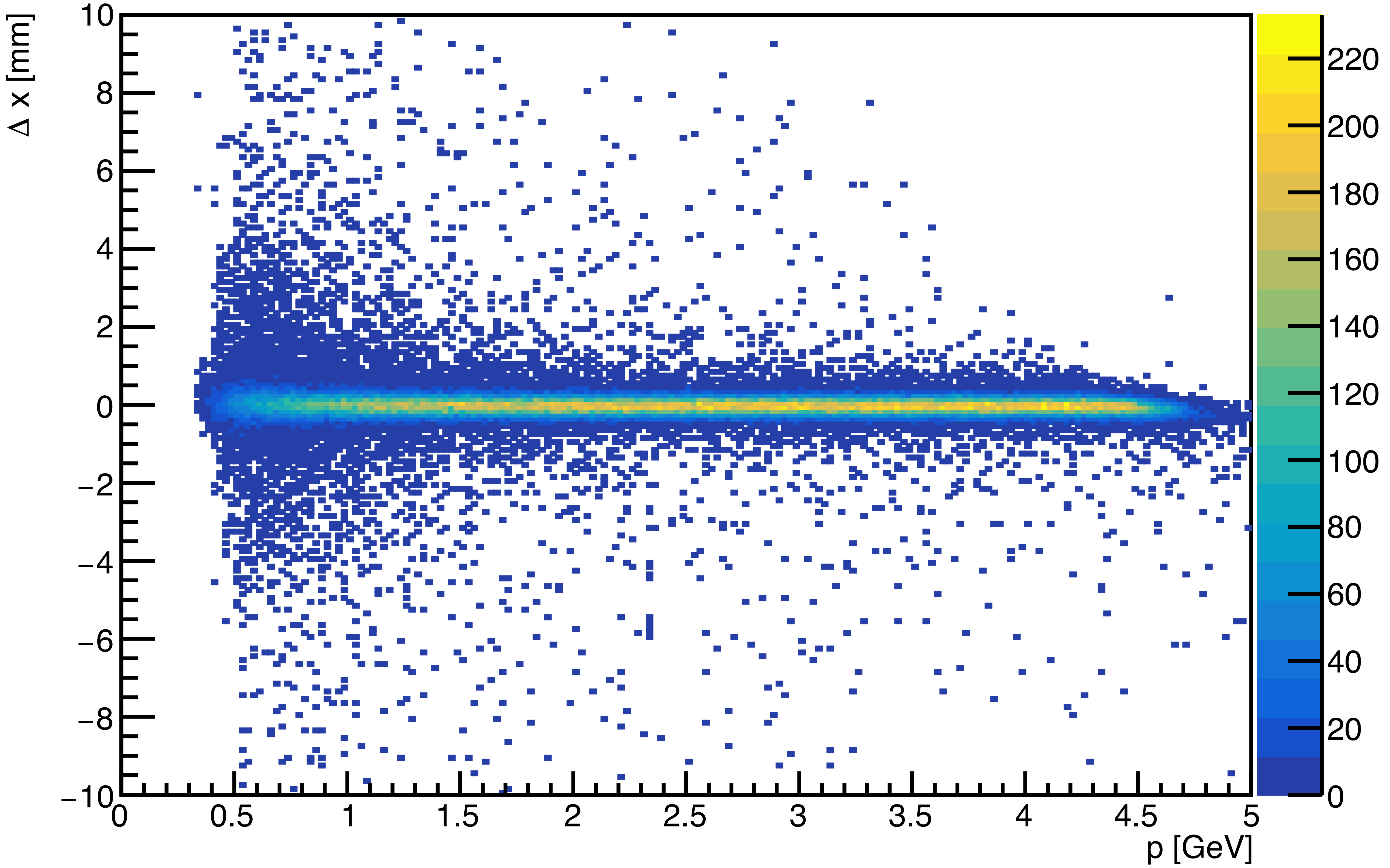
Compare Track and Scoring Plane

Positions match well. (Already shown by Lewis)

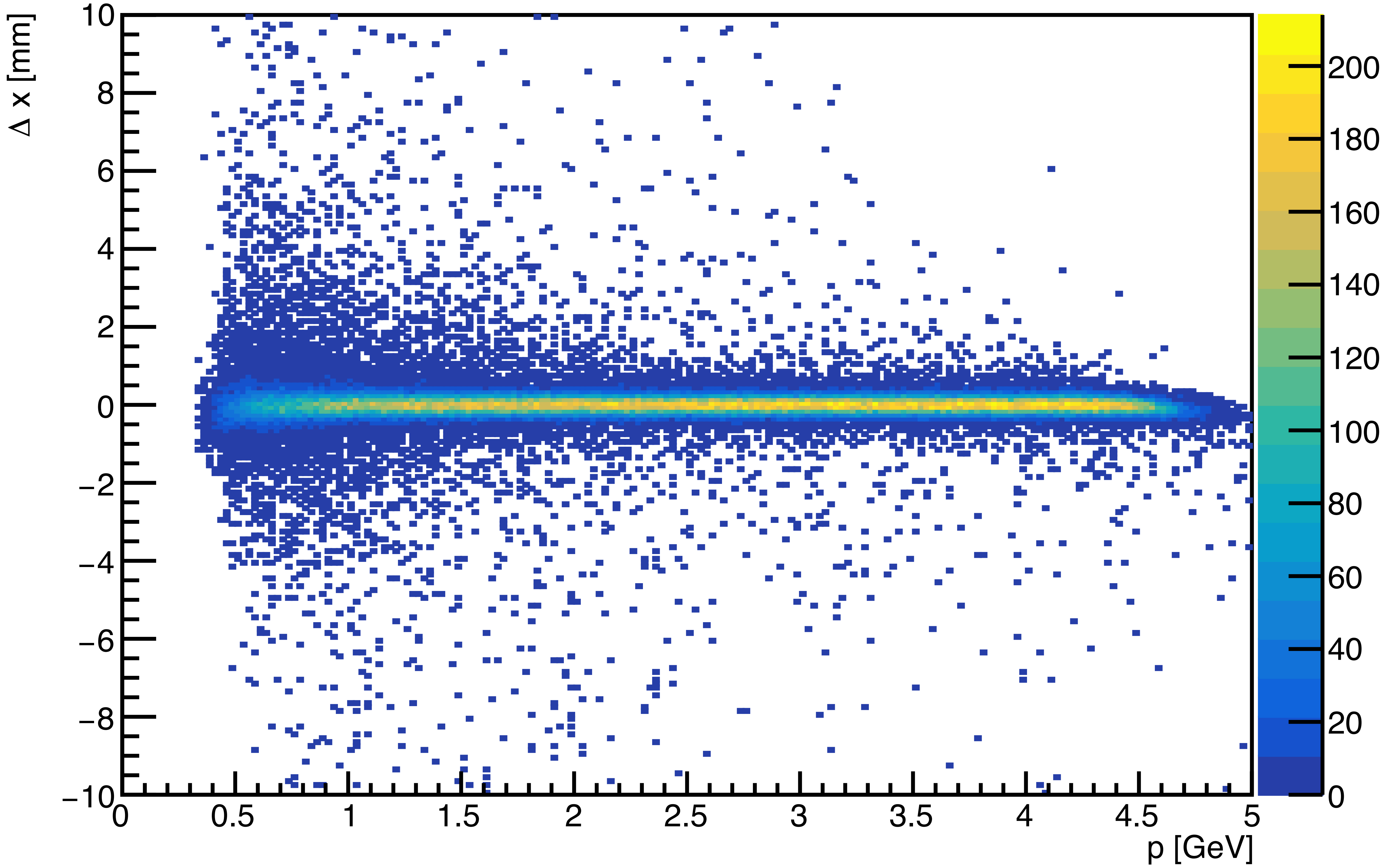




Compare Track and Scoring Plane



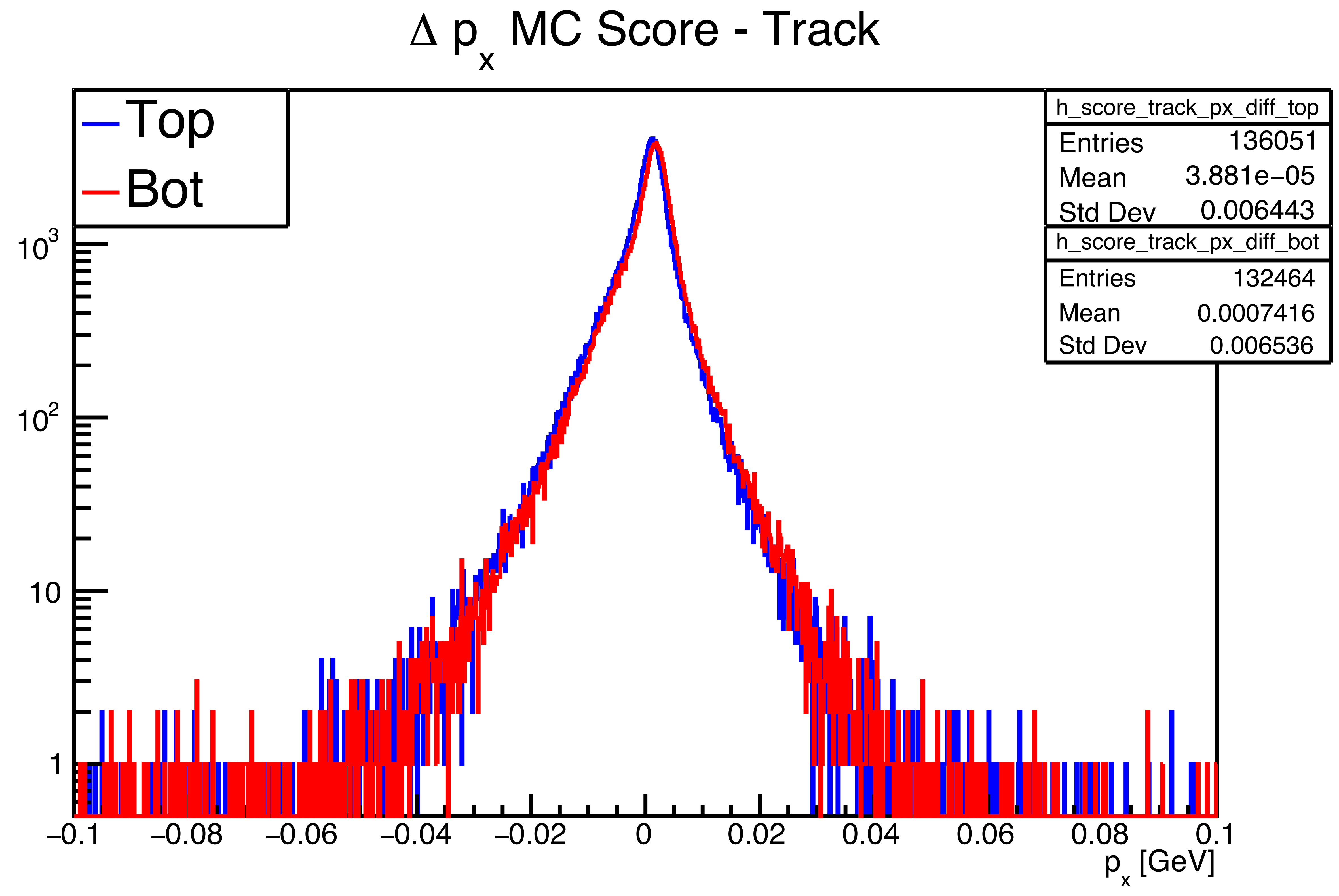
Track Δ x vs p top

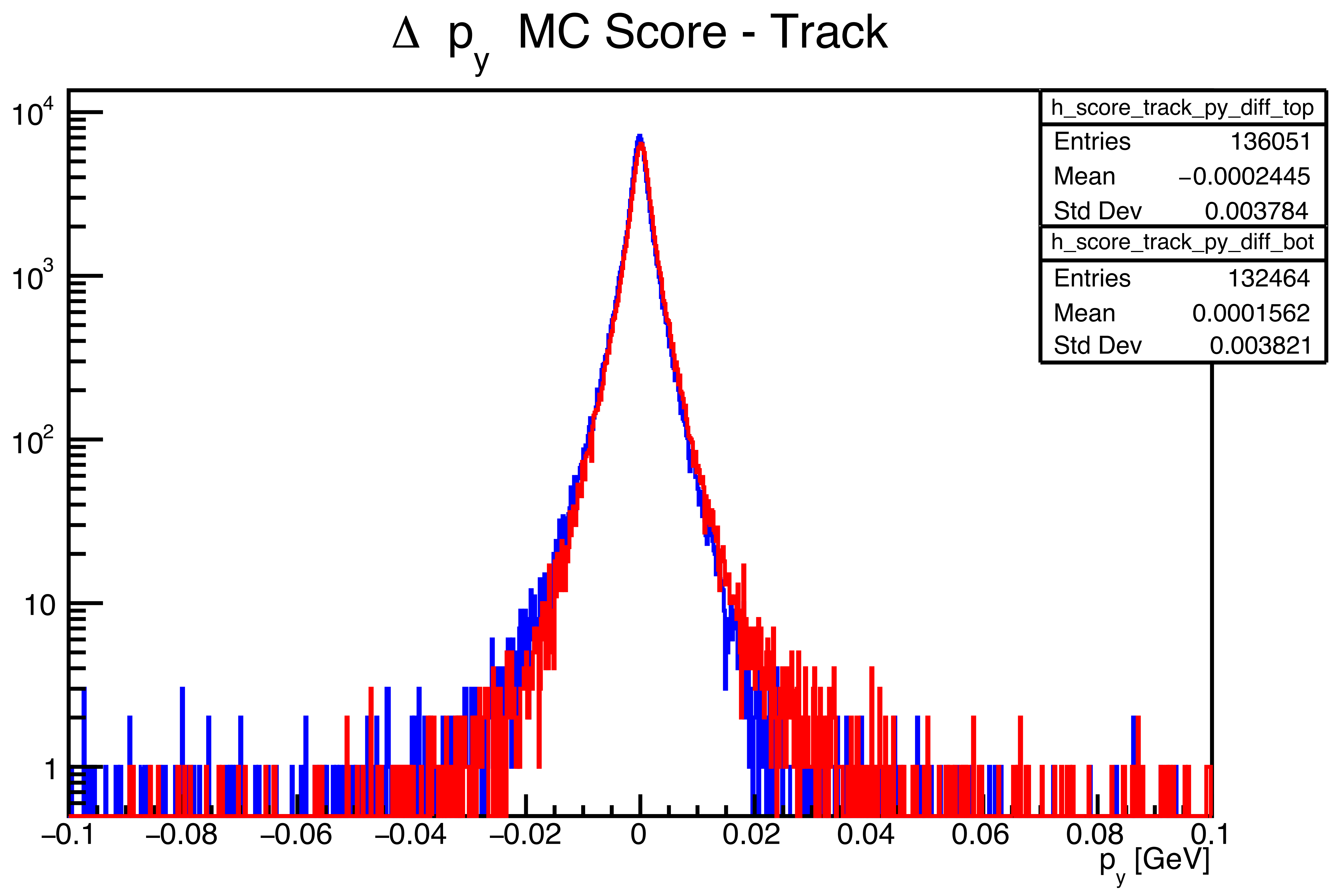


Track Δ x vs p bot

Compare Track and Scoring Plane

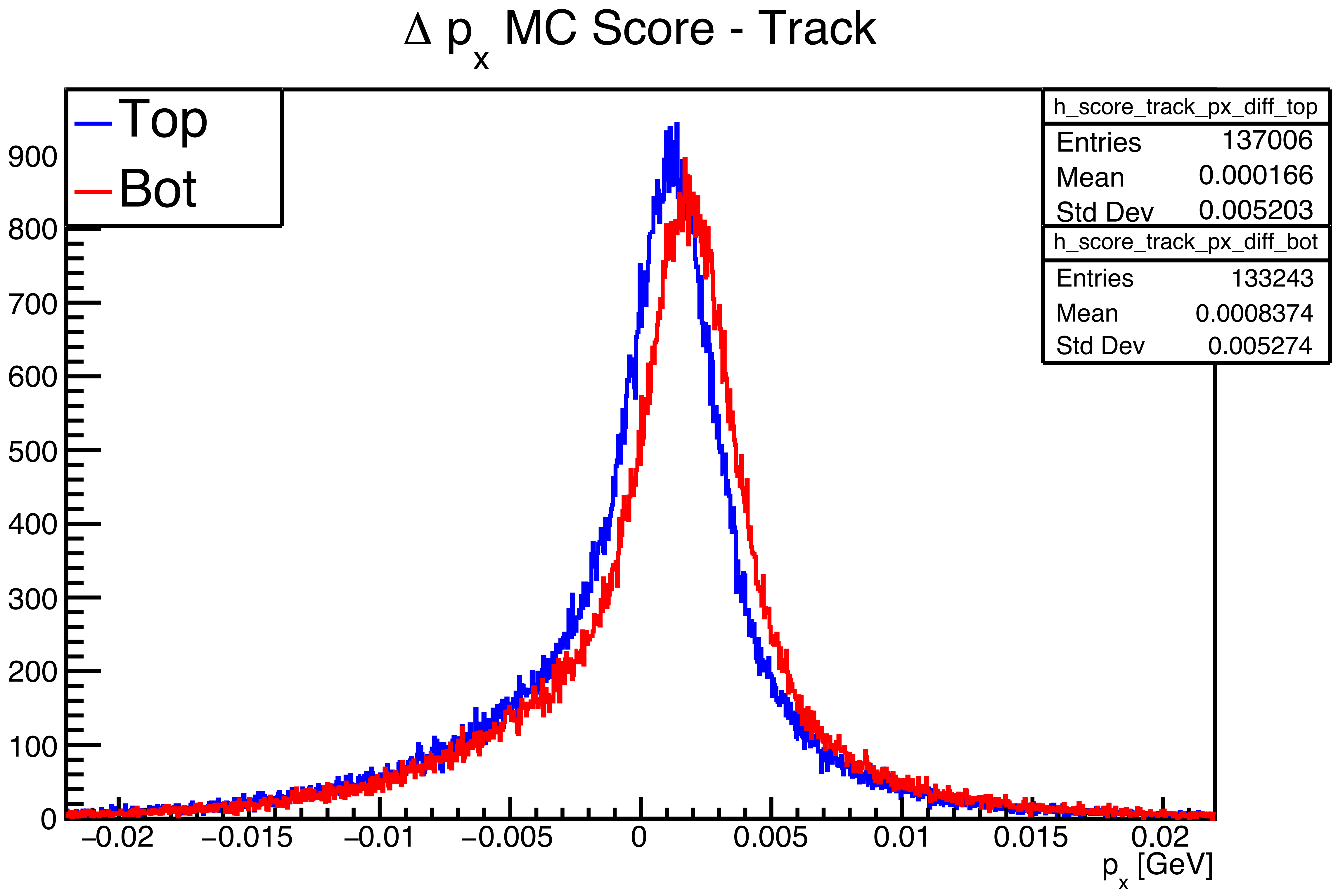
Momentum shows slightly more spread.





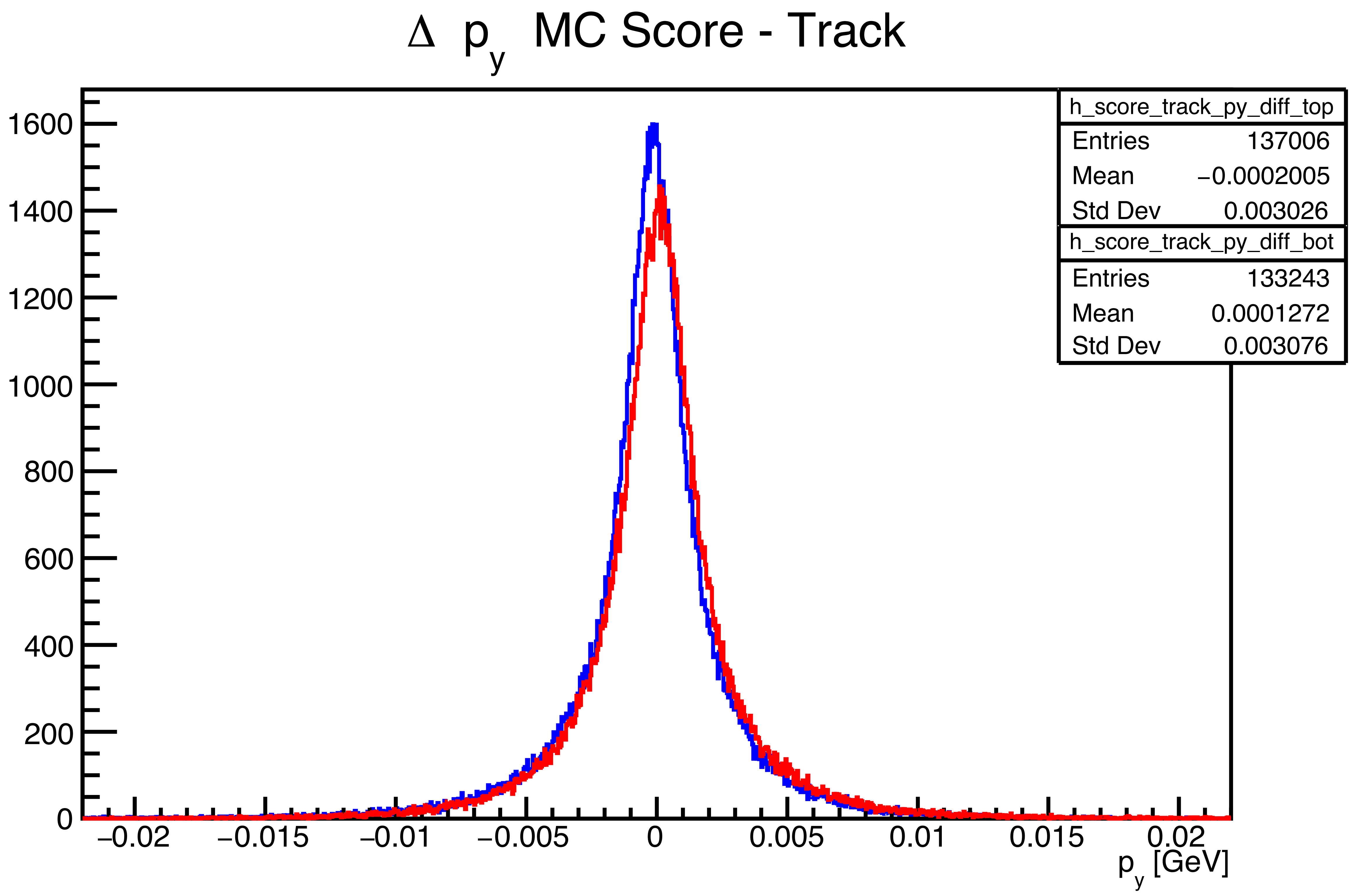
Compare Track and Scoring Plane

Momentum shows slightly more spread, linear scale.



h_score_track_px_diff_top	
Entries	137006
Mean	0.000166
Std Dev	0.005203

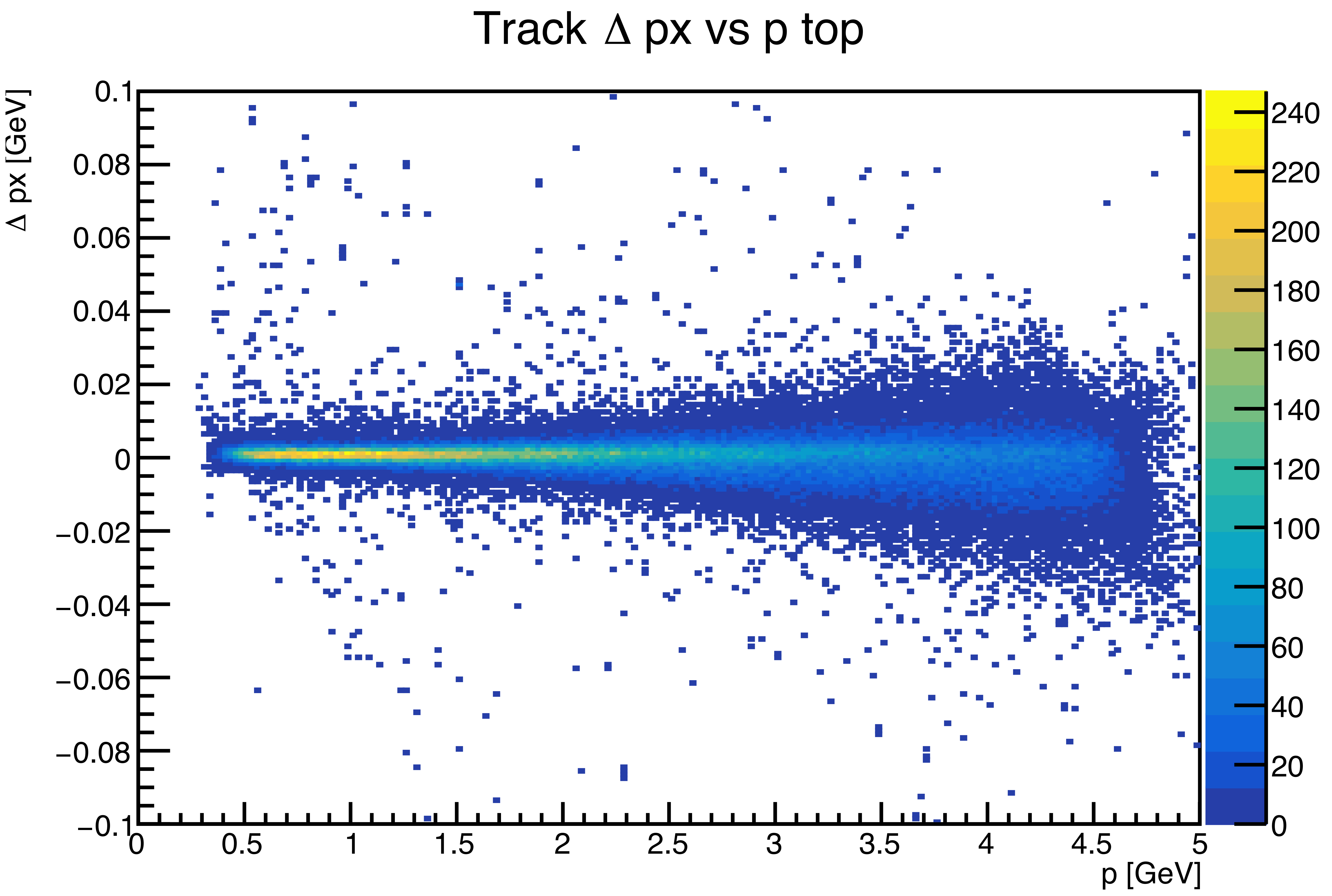
h_score_track_px_diff_bot	
Entries	133243
Mean	0.0008374
Std Dev	0.005274

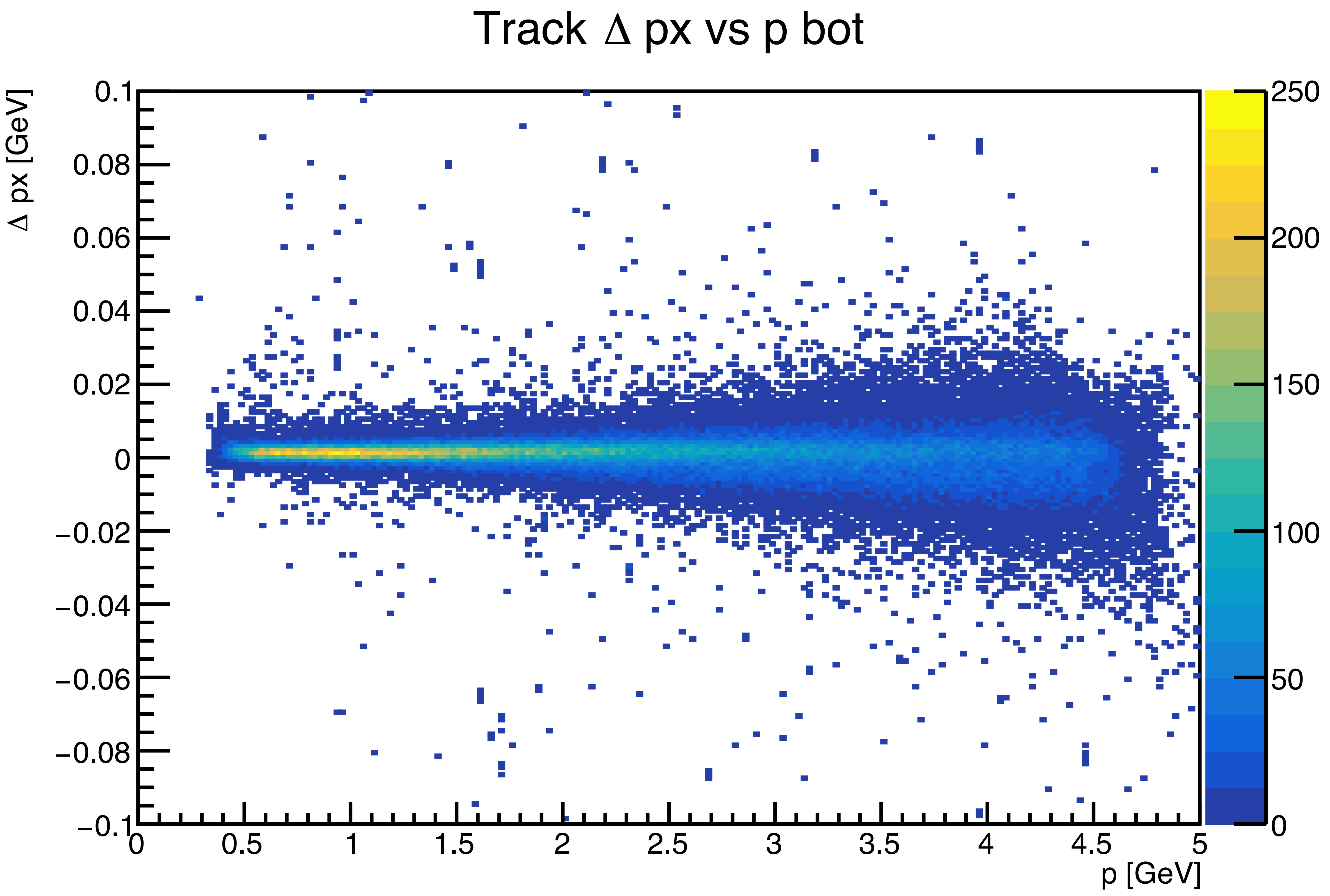


h_score_track_py_diff_top	
Entries	137006
Mean	−0.0002005
Std Dev	0.003026

h_score_track_py_diff_bot	
Entries	133243
Mean	0.0001272
Std Dev	0.003076

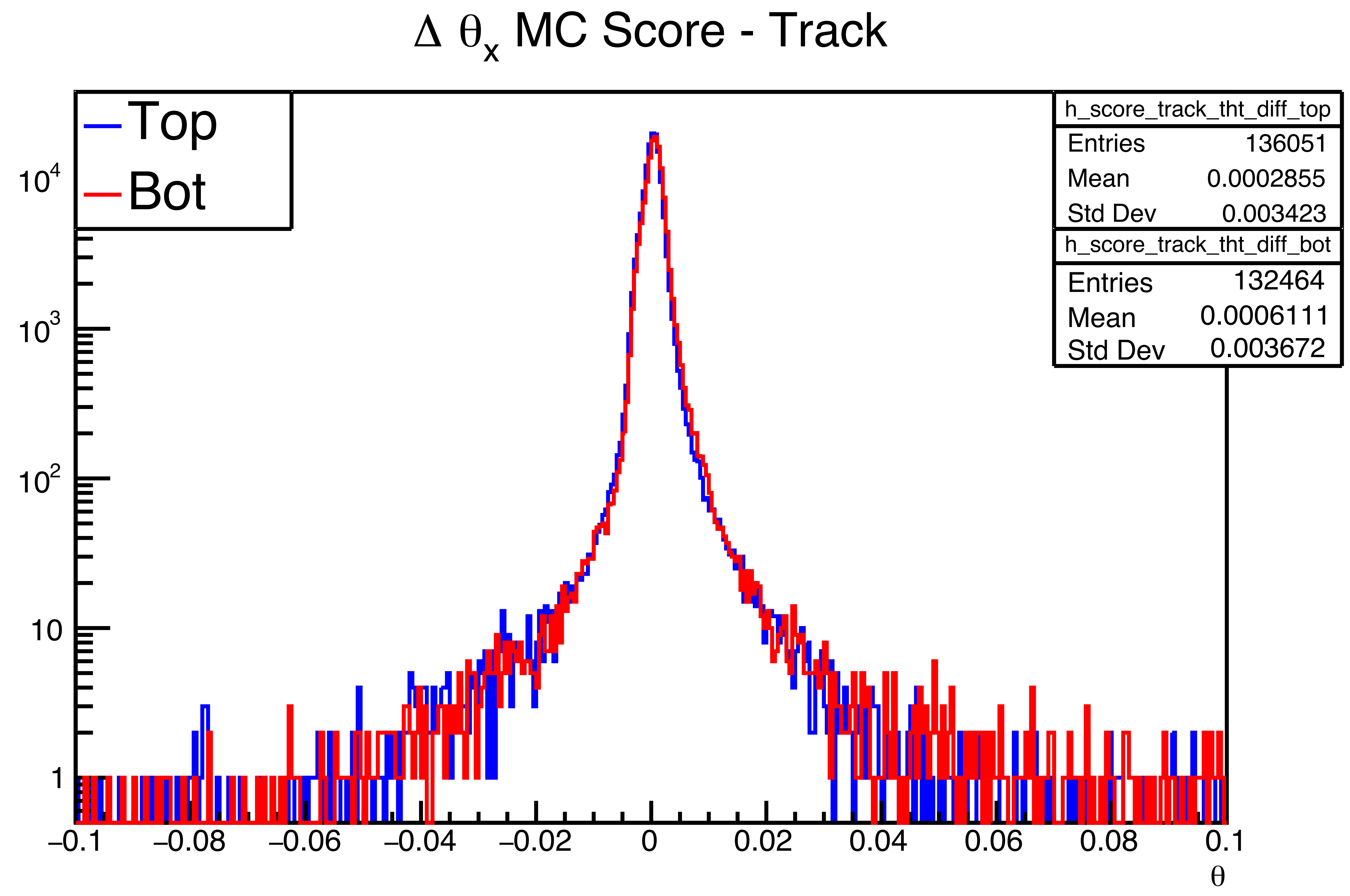
Compare Track and Scoring Plane

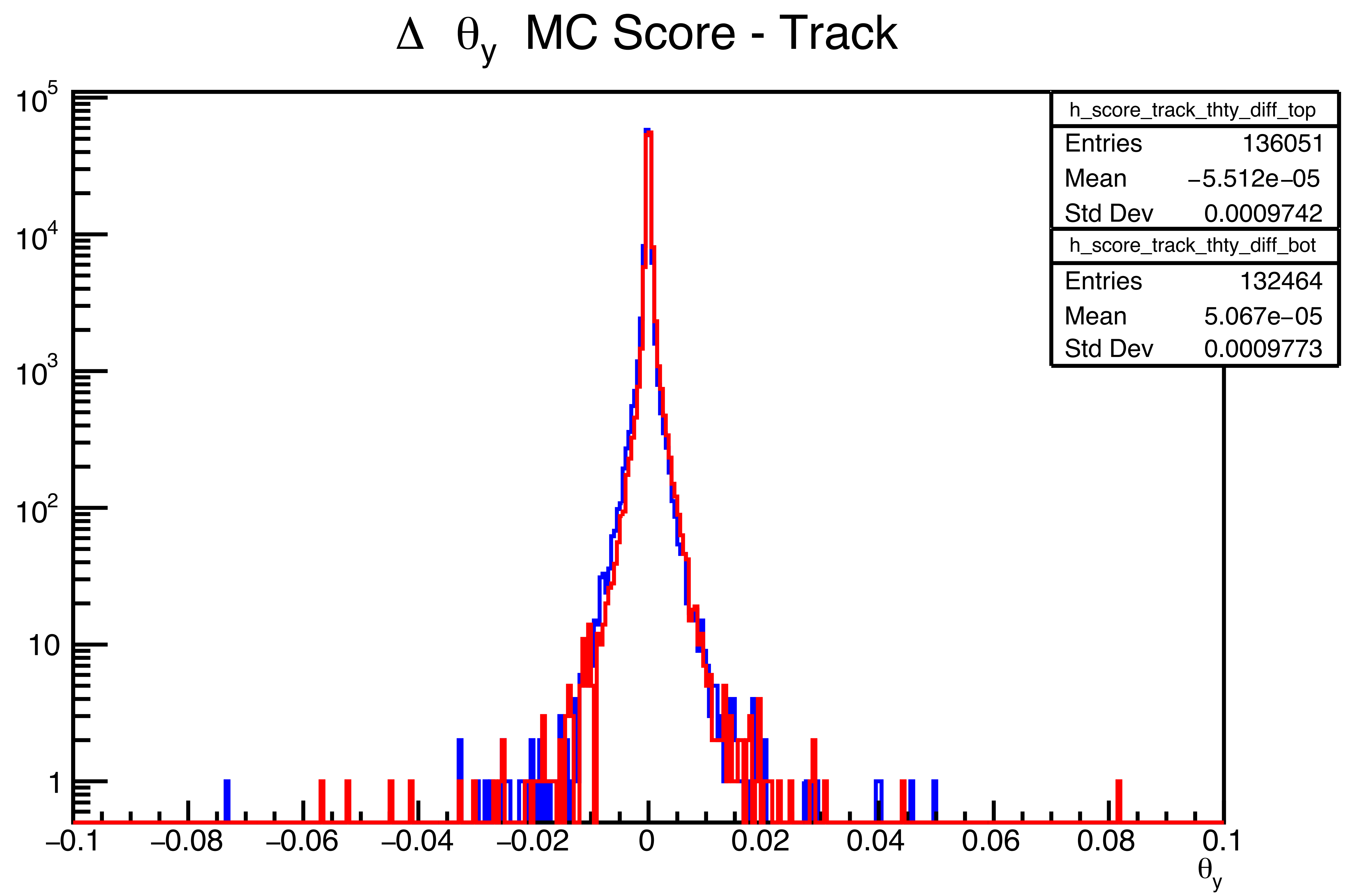




Compare Track and Scoring Plane

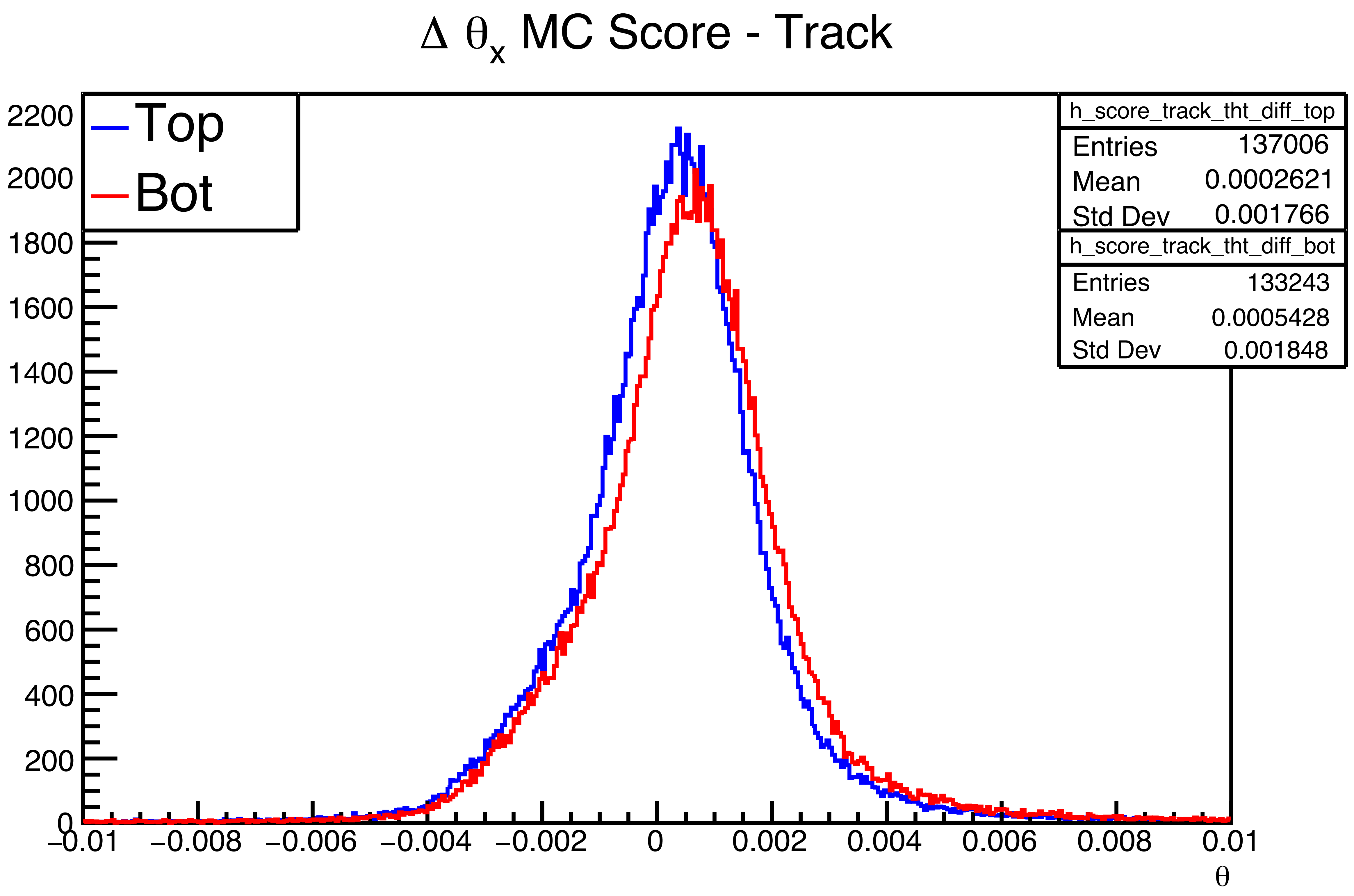
Angles match well.

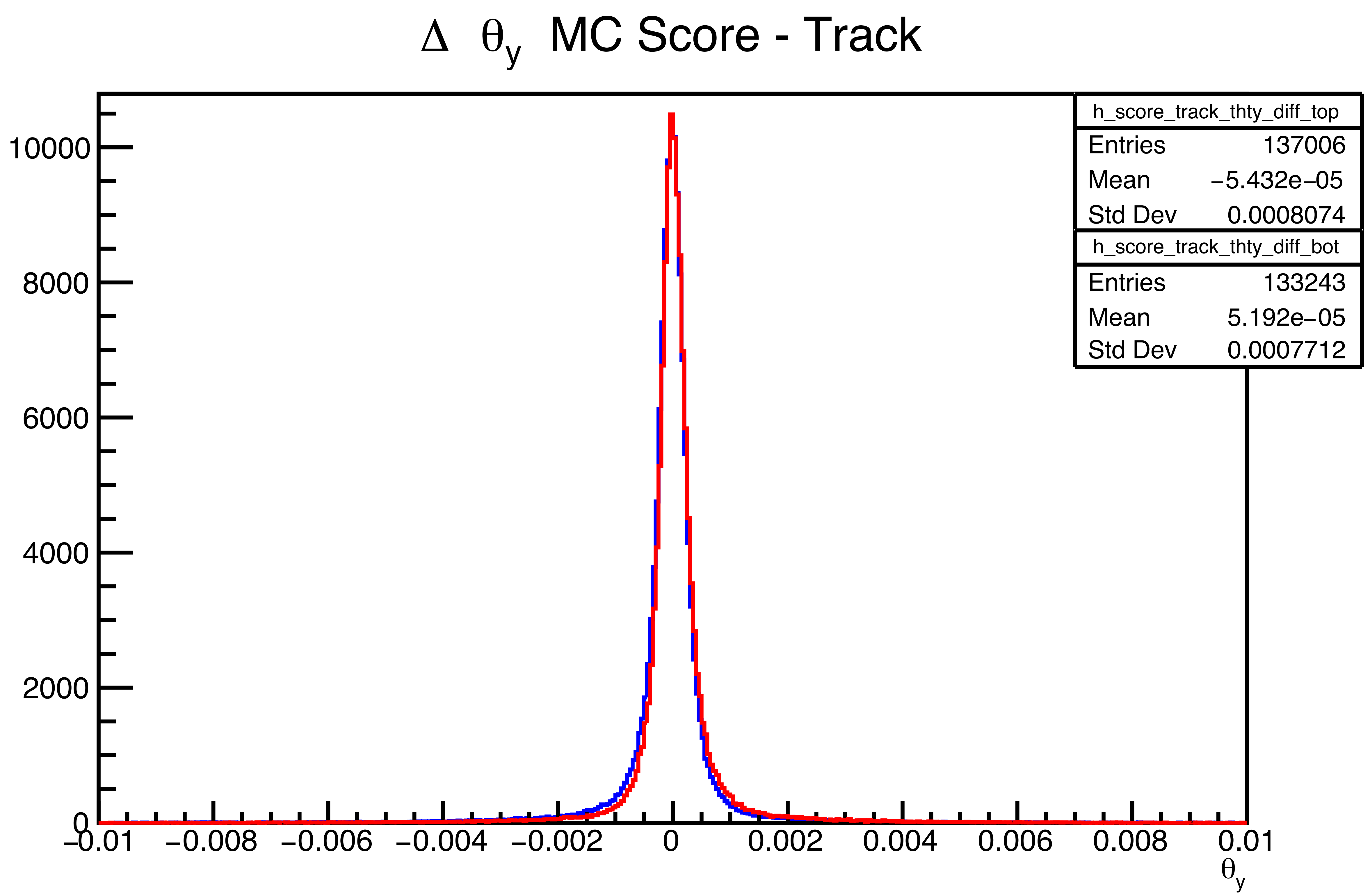




Compare Track and Scoring Plane

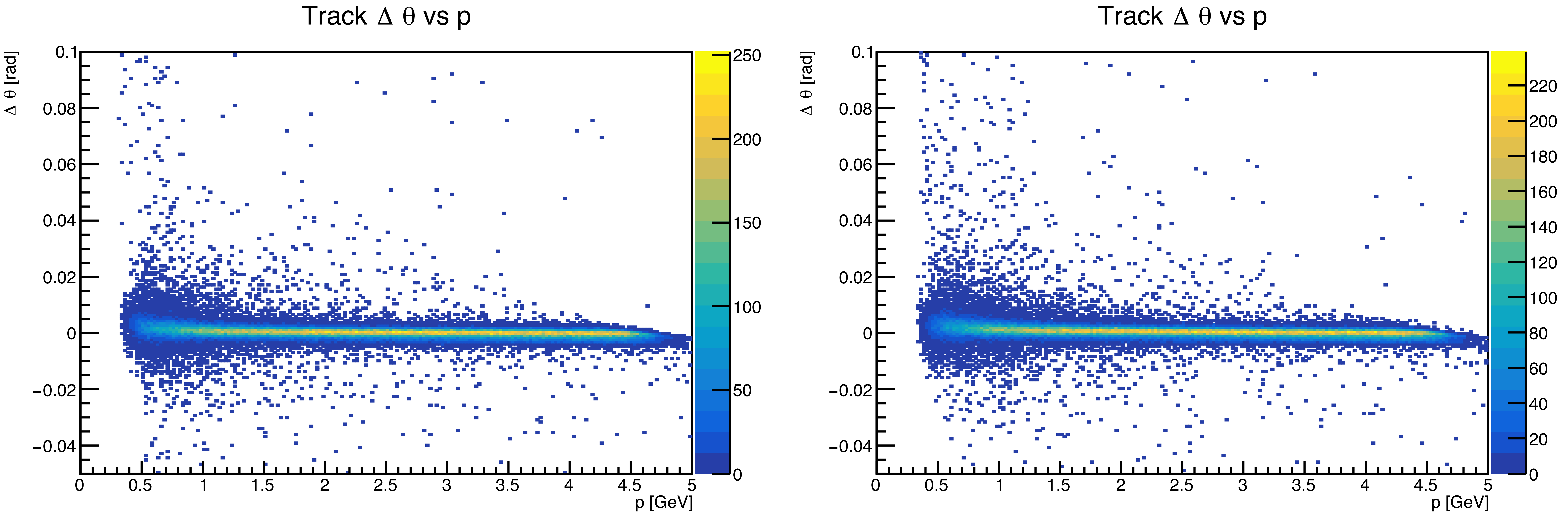
Angles match well on linear scale too 😁.



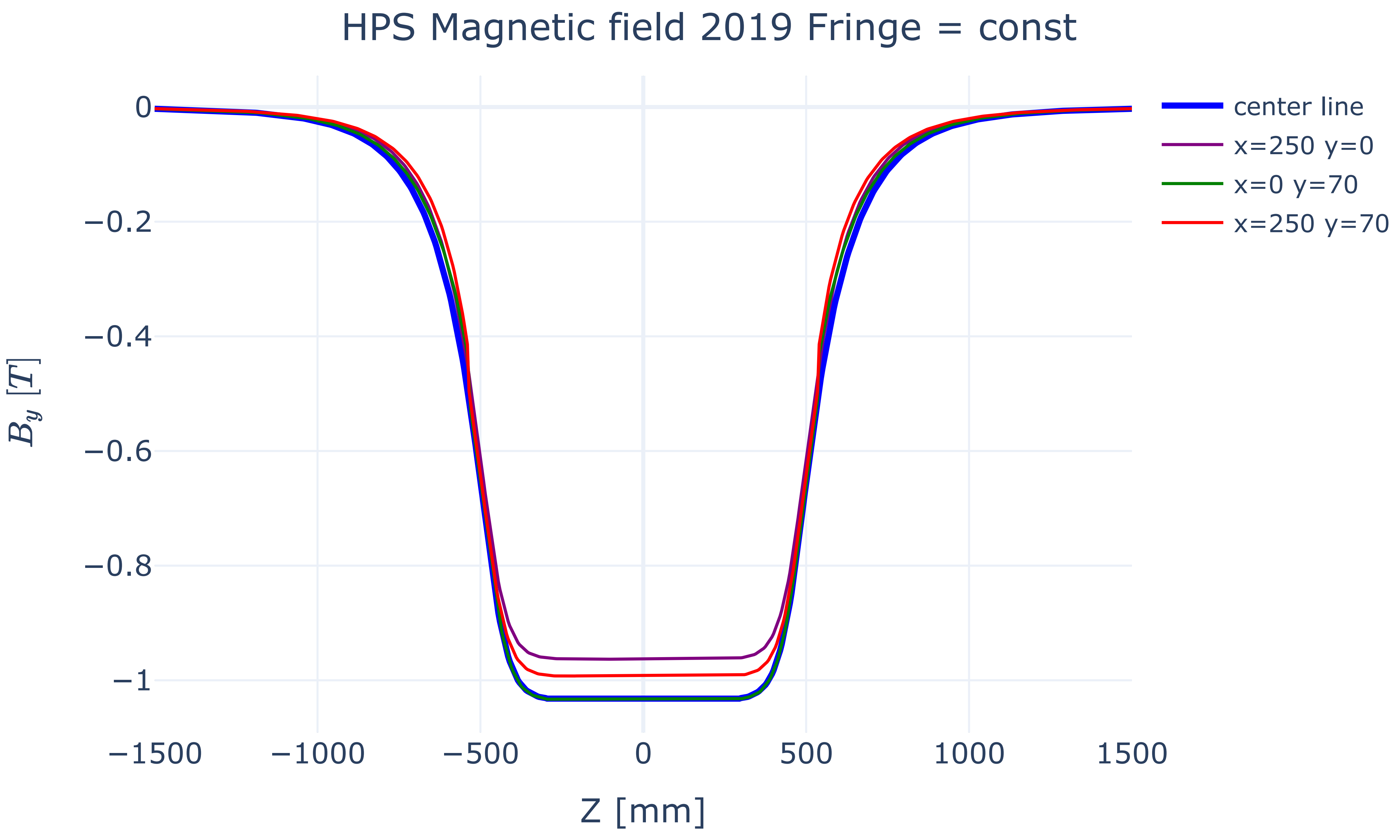


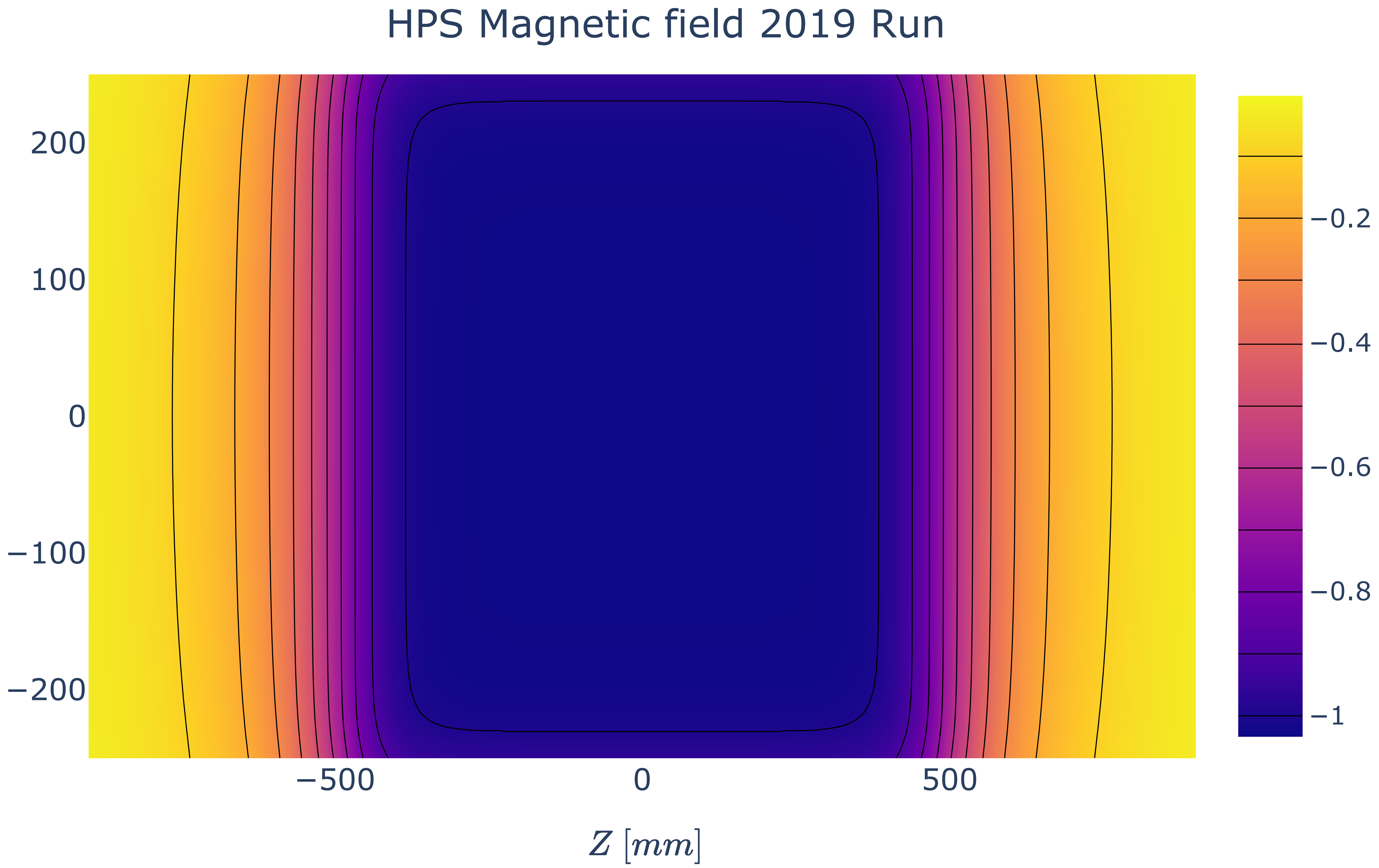
Compare Track and Scoring Plane

Angles match well.



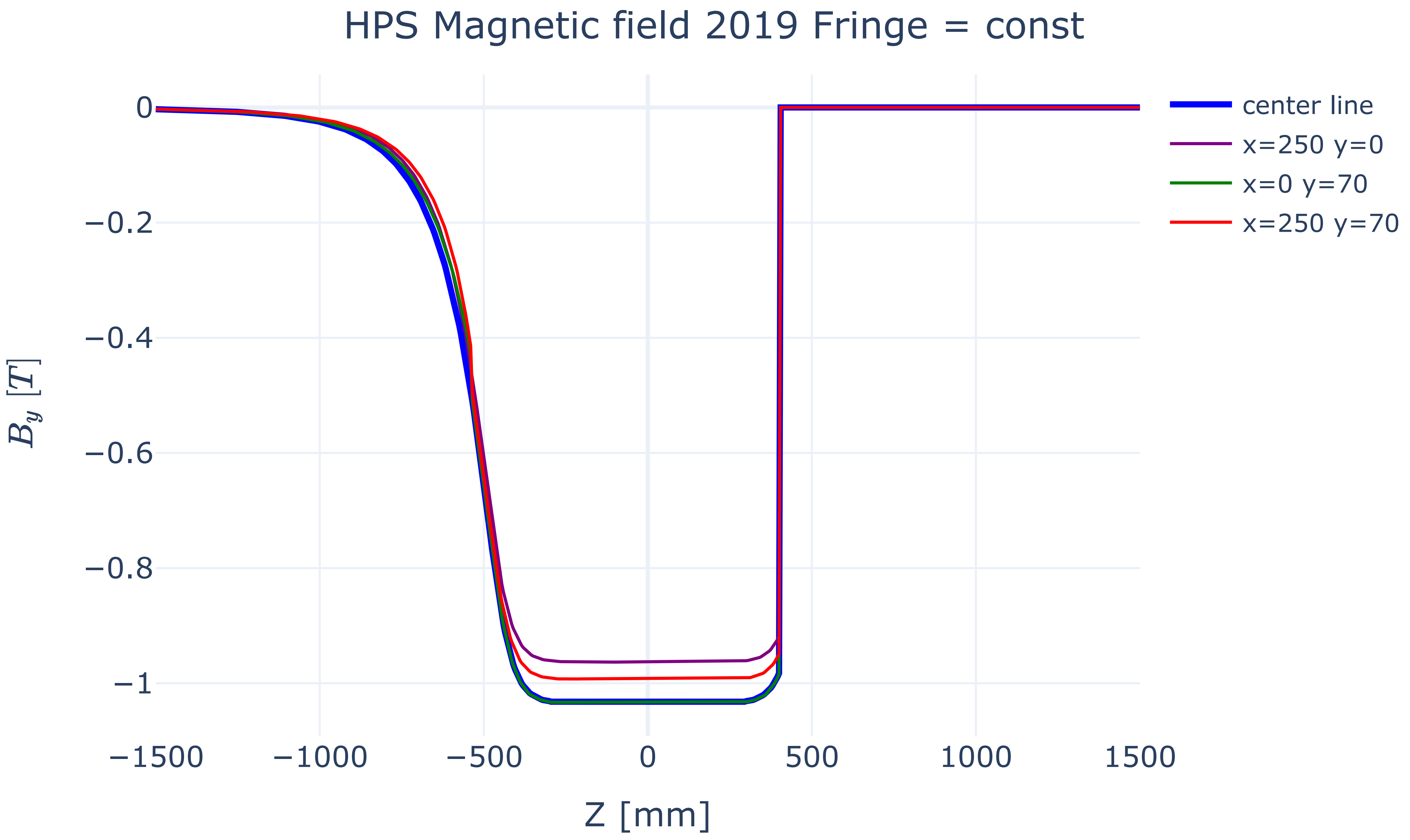
Normal Fringe Field

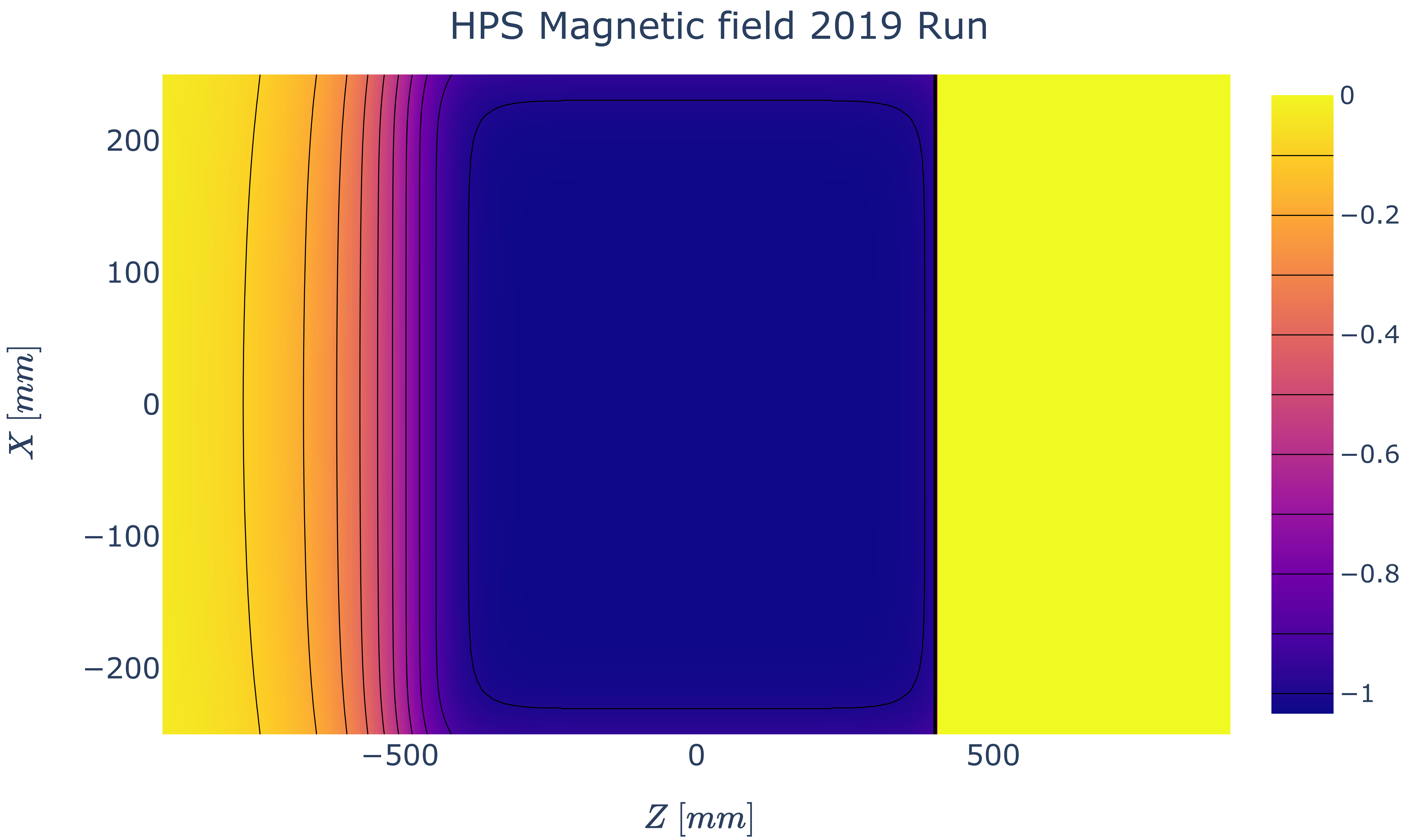




Fringe field = zero

- To test the behavior of the code in the fringe field, we set the fringe field to zero at the last tracker. So By = 0 when z>400 (magnet coordinates, 872 in detector coordinates). This means the bottom last tracker is in zero field, while the top last tracker is in the full field.

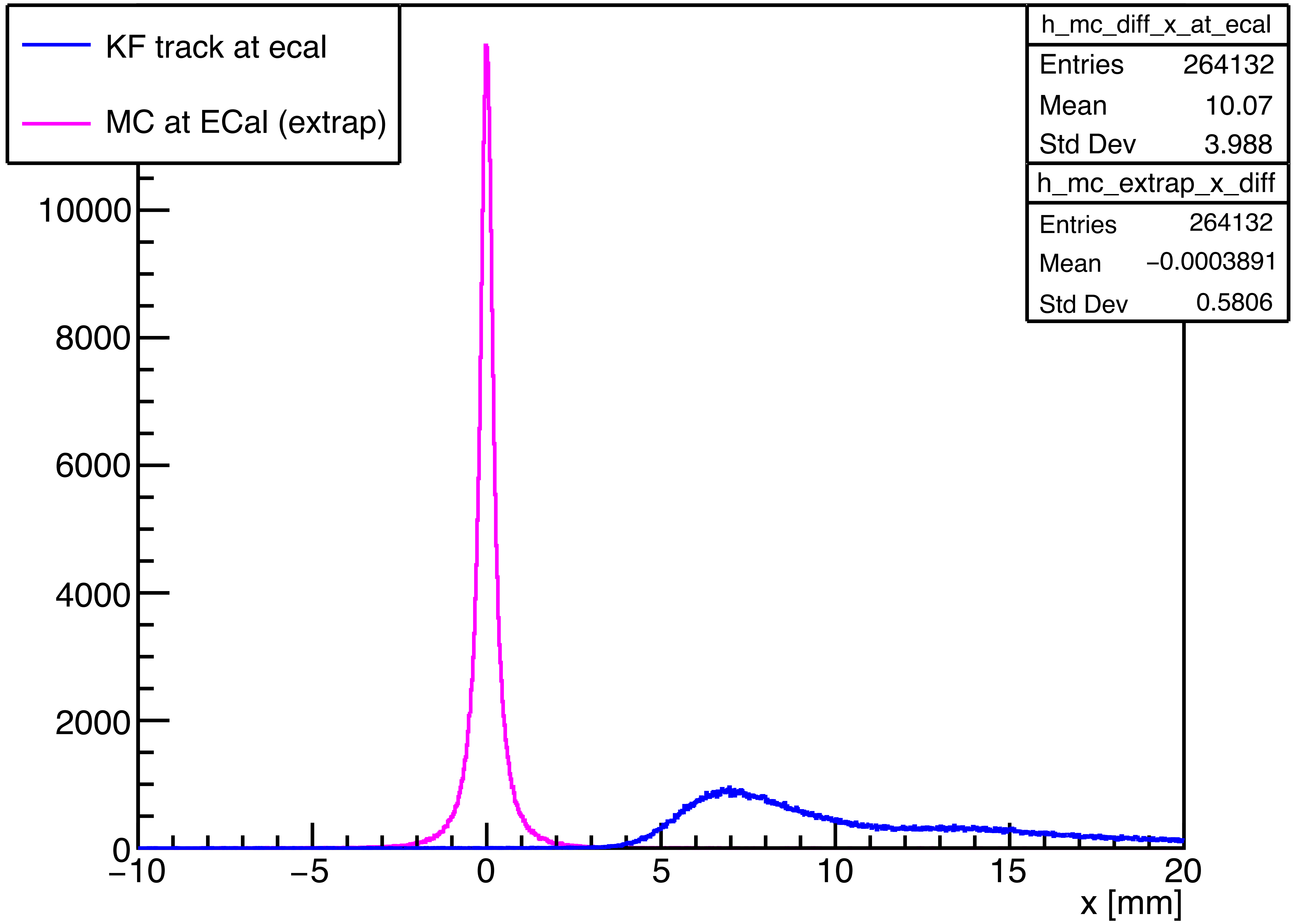




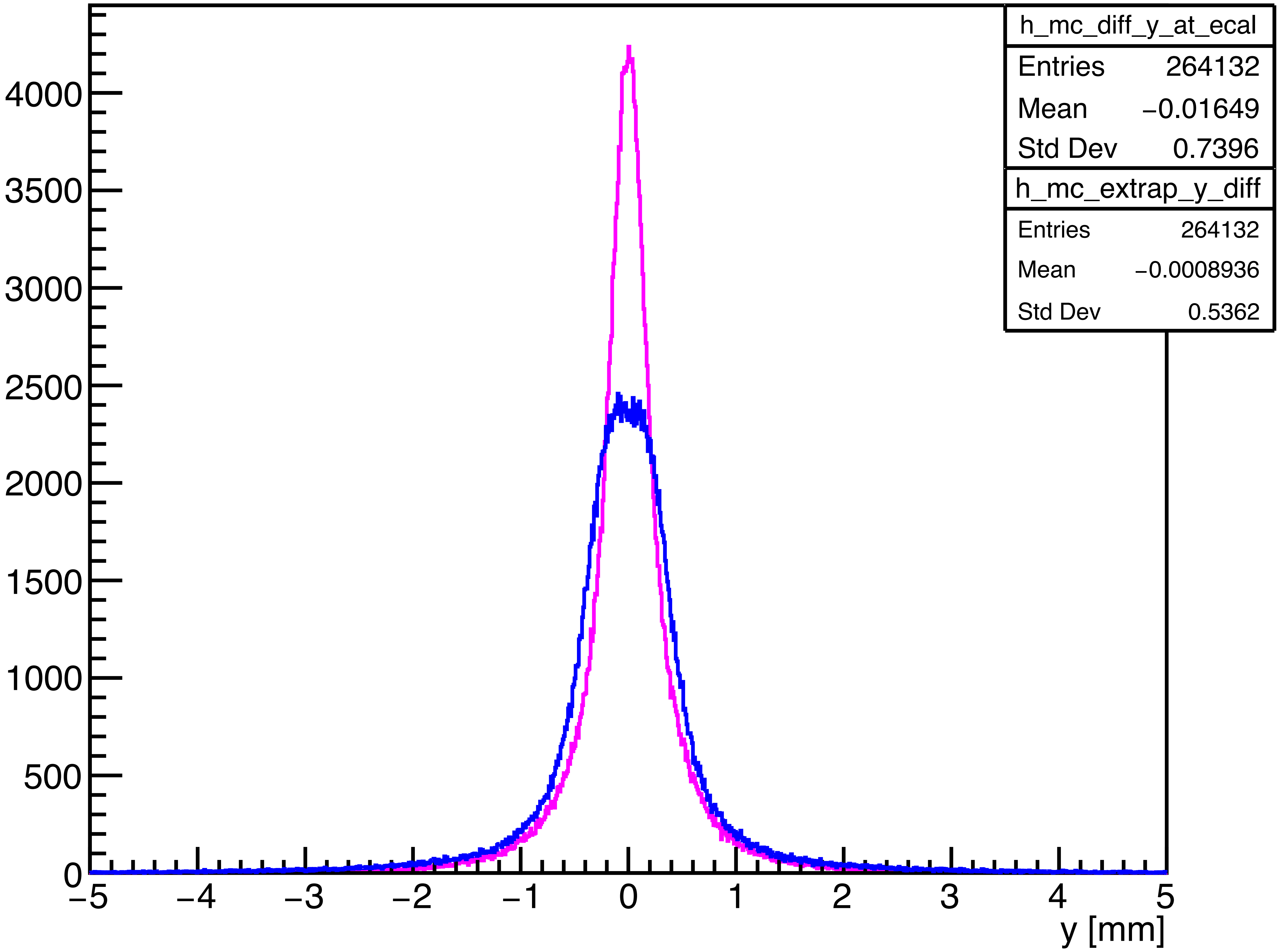
Zero Fringe: Compare position at ECal

The KF track extrapolated to the ECal now is much worse, while the simple extrapolation now gives the correct result, since there is no field. Lewis also showed this. This new simulation has the B field actually set to zero, so this is not likely due to interpolation of the field map by hps-java.

Extrapolated x difference

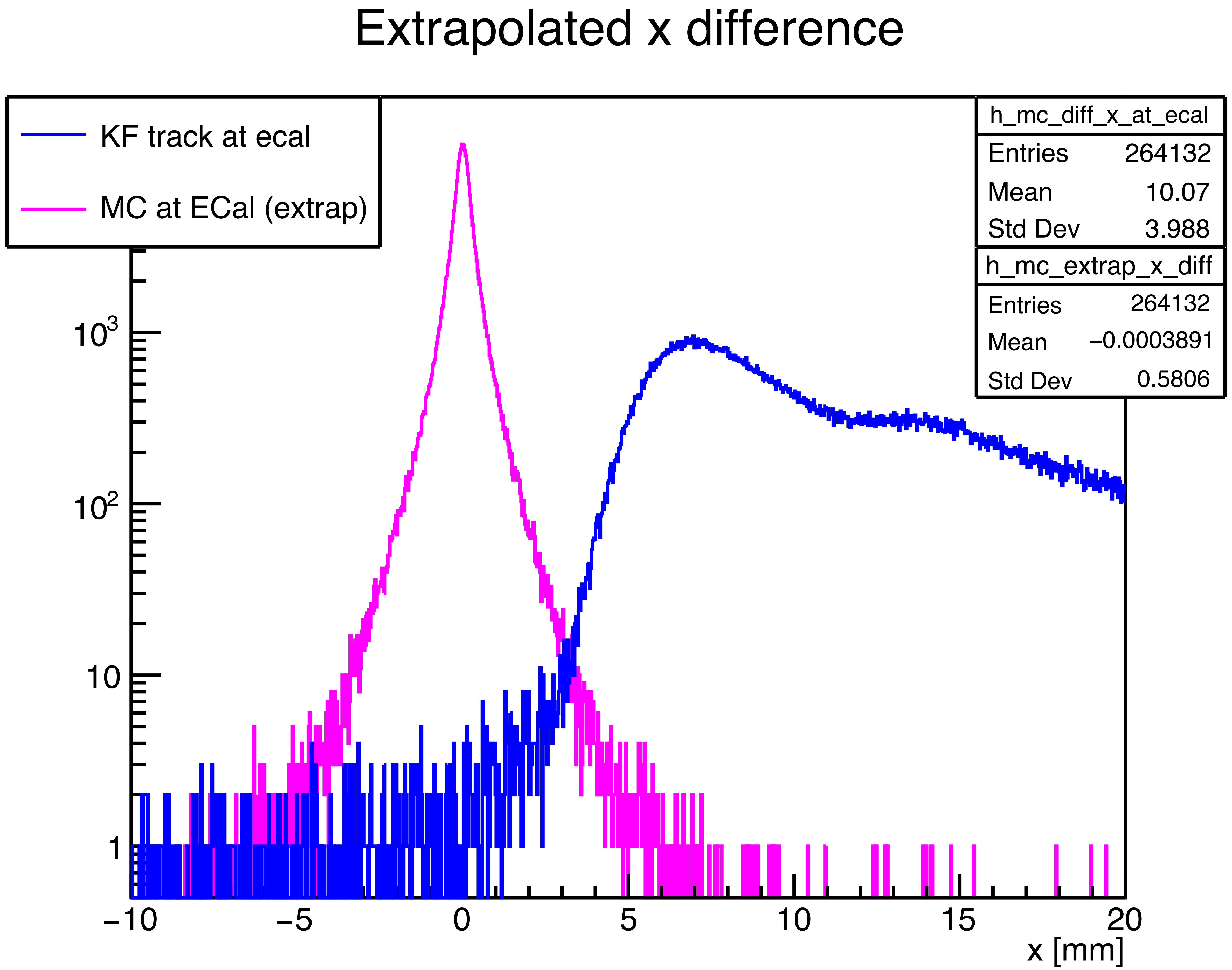


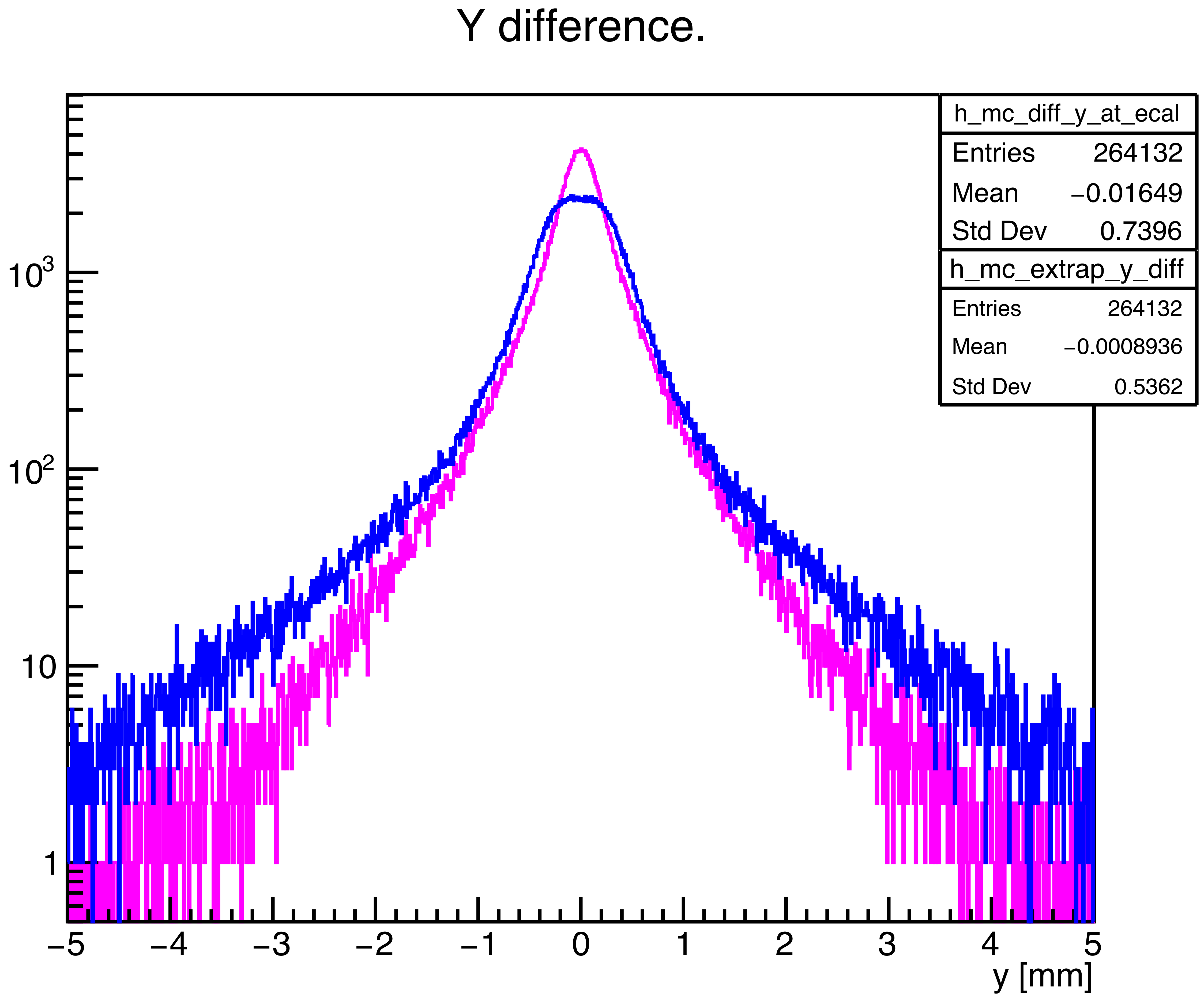
Y difference.



Zero Fringe: Compare position at ECal

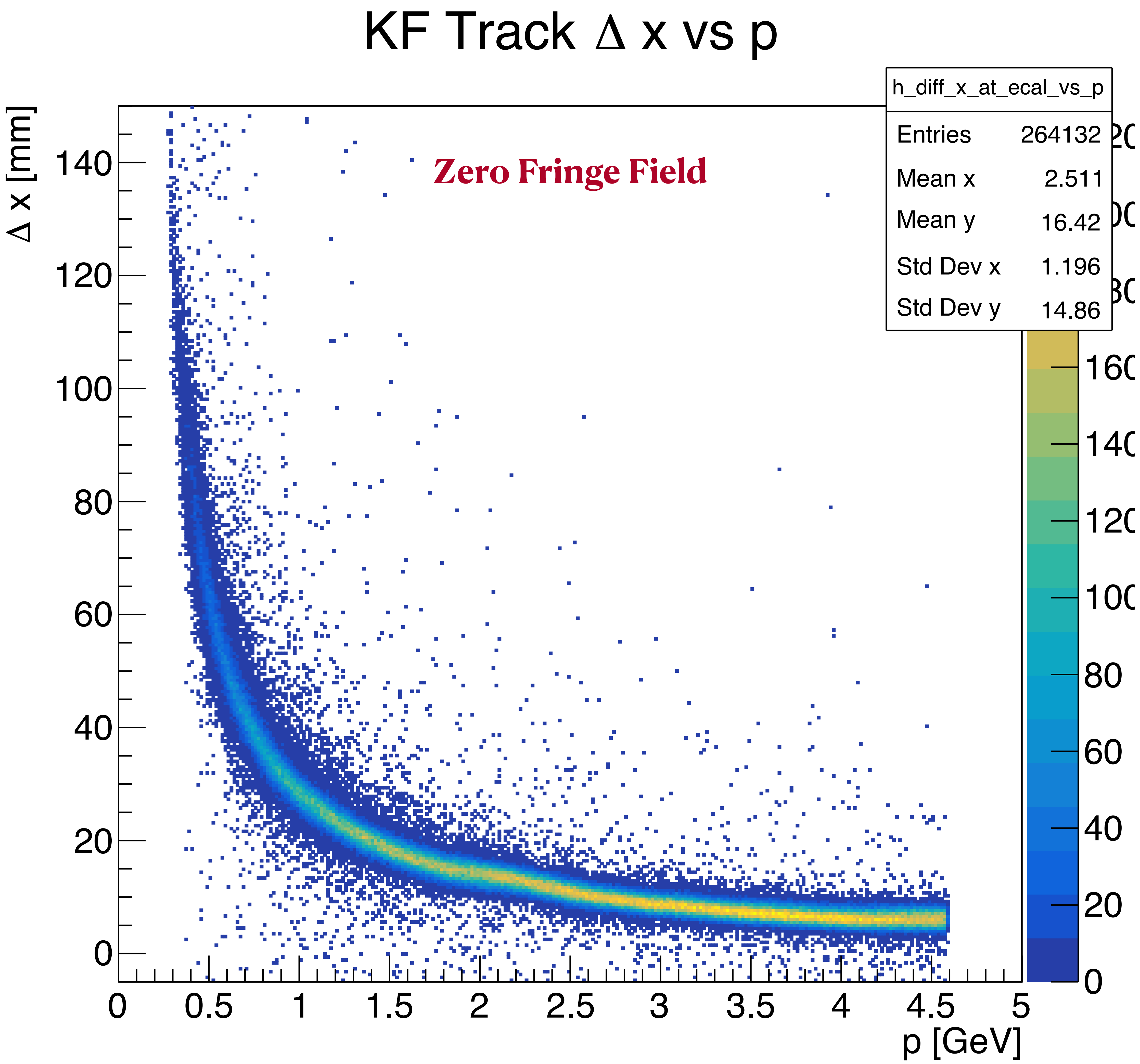
The KF track extrapolated to the ECal now is much worse, while the simple extrapolation now gives the correct result, since there is no field. Lewis also showed this. This new simulation has the B field actually set to zero, so this is not likely due to interpolation of the field map by hps-java.



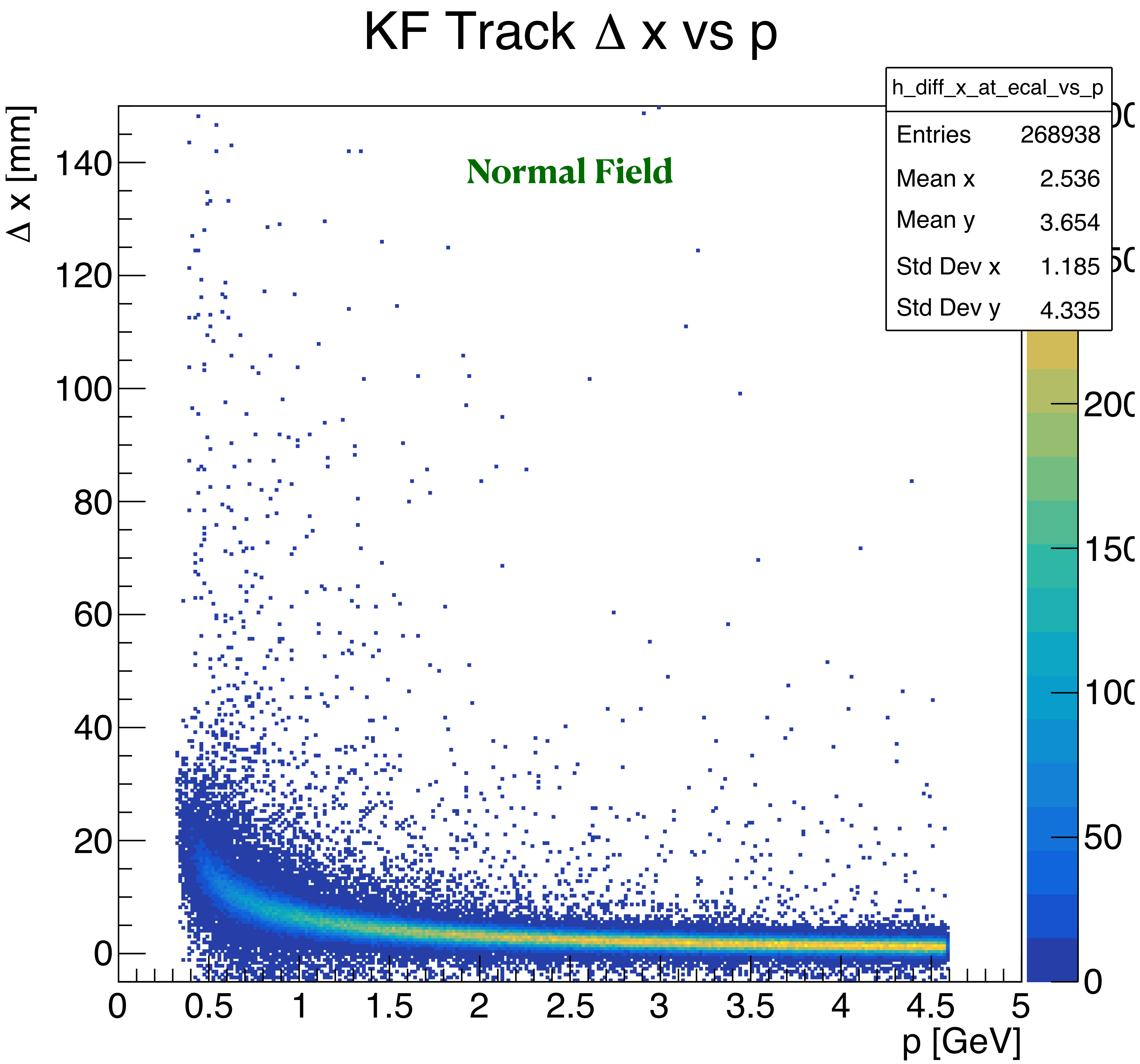


Zero Fringe: Compare position at ECal

The KF track extrapolated to the ECal now is much worse, while the simple extrapolation now gives the correct result, since there is no field. Lewis also showed this. This new simulation has the B field actually set to zero, so this is not likely due to interpolation of the field map by hps-java.



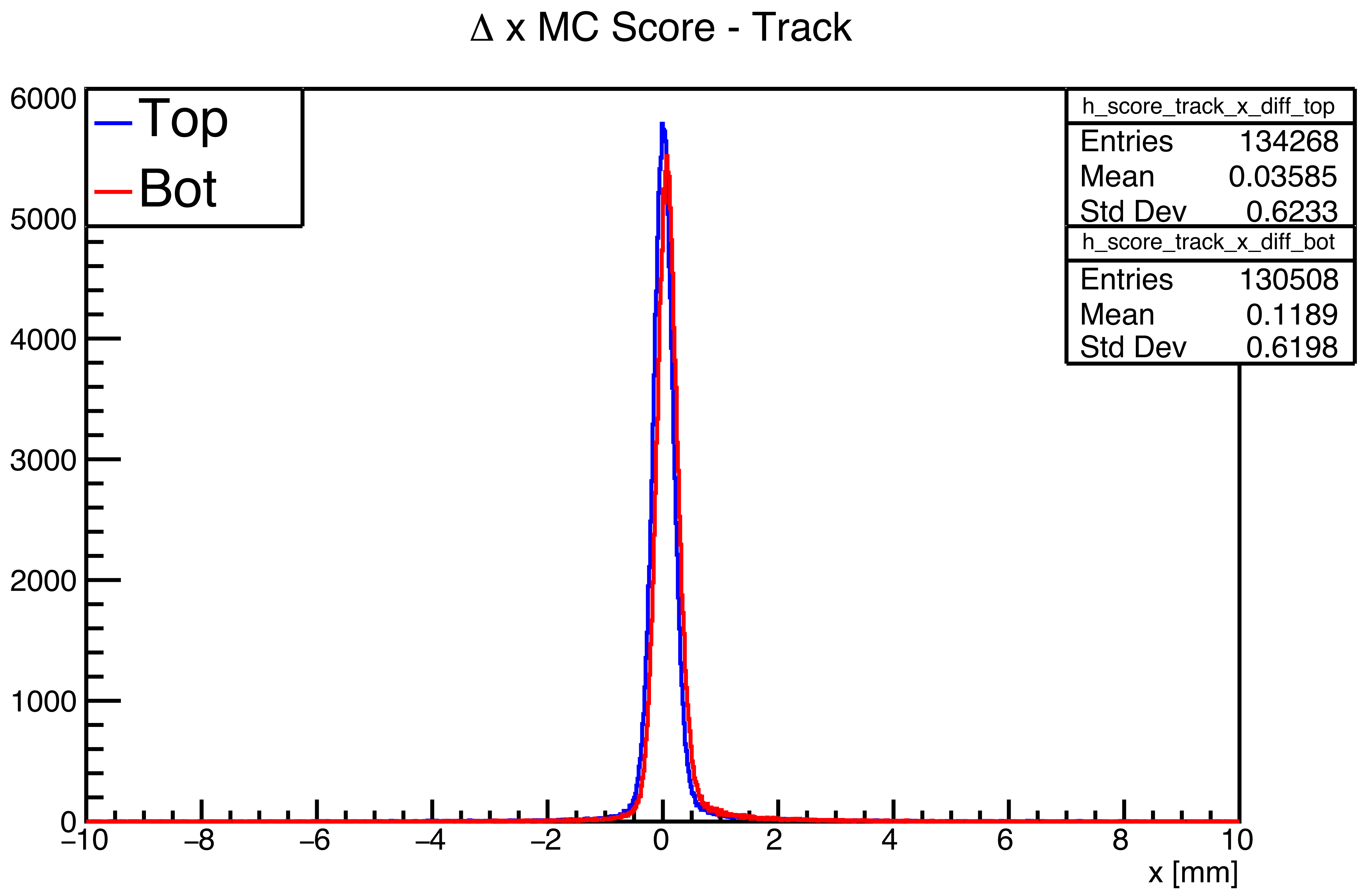
h_diff_x_at_ecal_vs_p	
Entries	264132
Mean x	2.511
Mean y	16.42
Std Dev x	1.196
Std Dev y	14.86

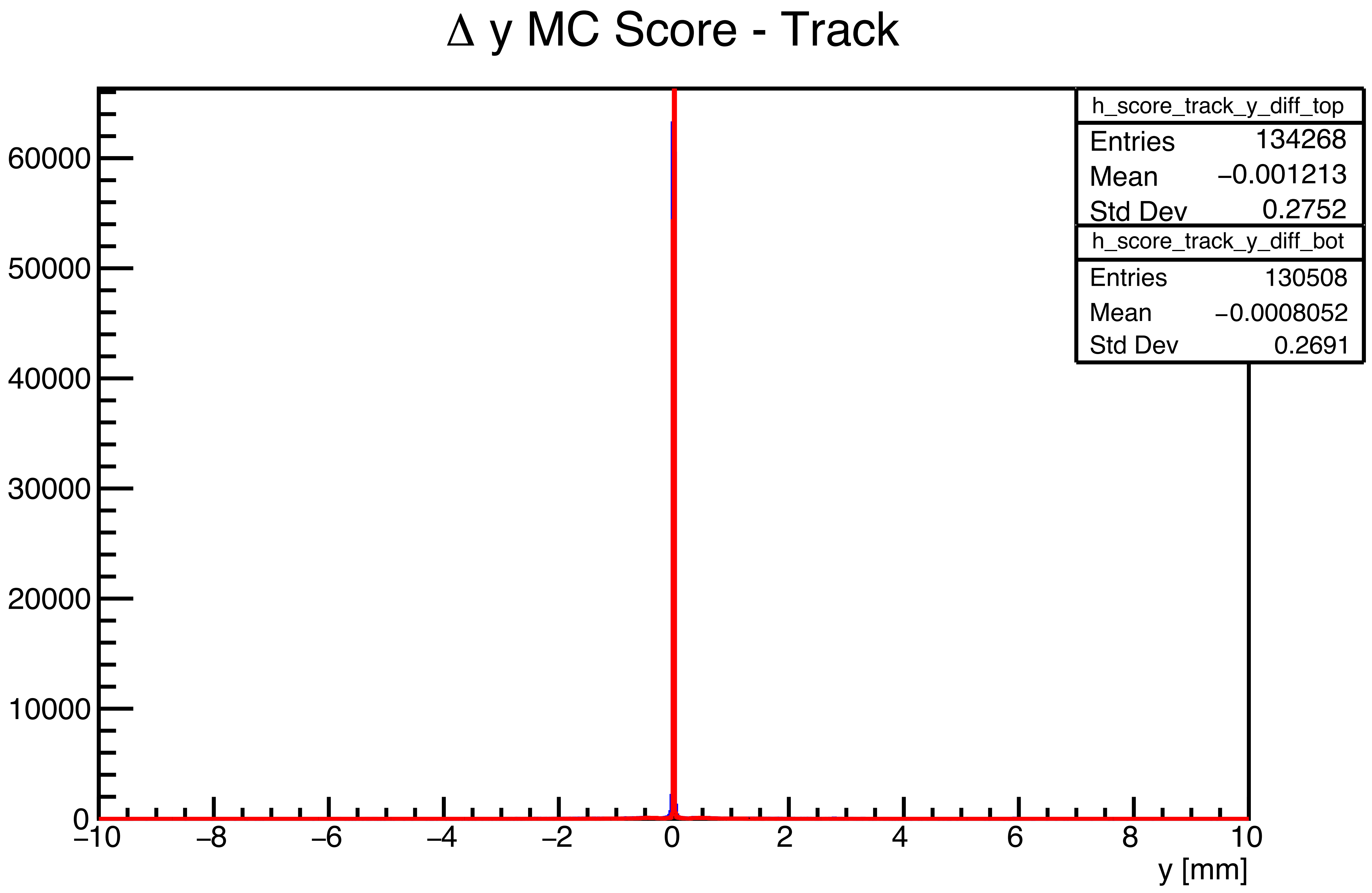


h_diff_x_at_ecal_vs_p	
Entries	268938
Mean x	2.536
Mean y	3.654
Std Dev x	1.185
Std Dev y	4.335

Zero Fringe: Compare Track and Scoring Plane

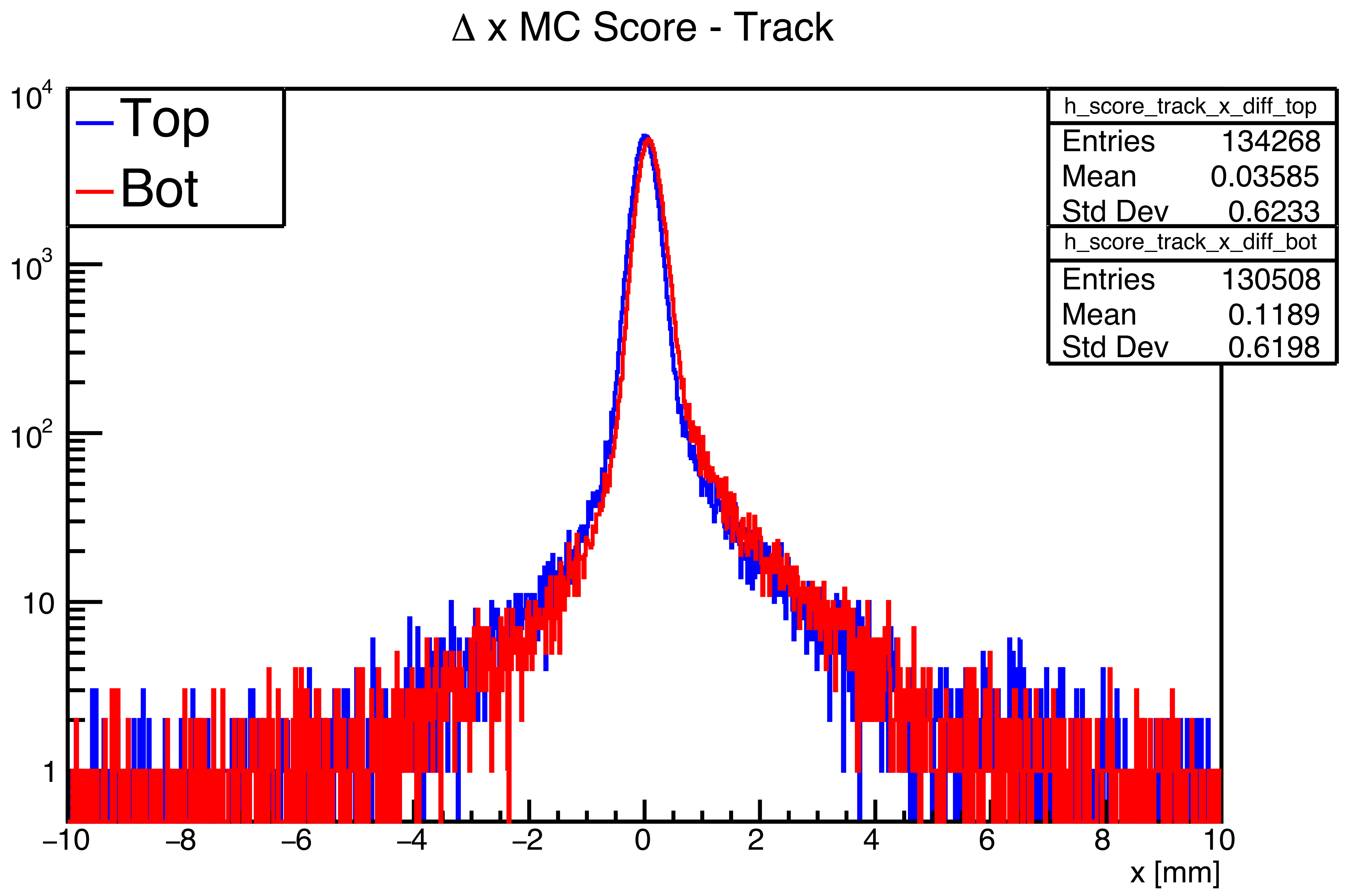
Positions still match well.





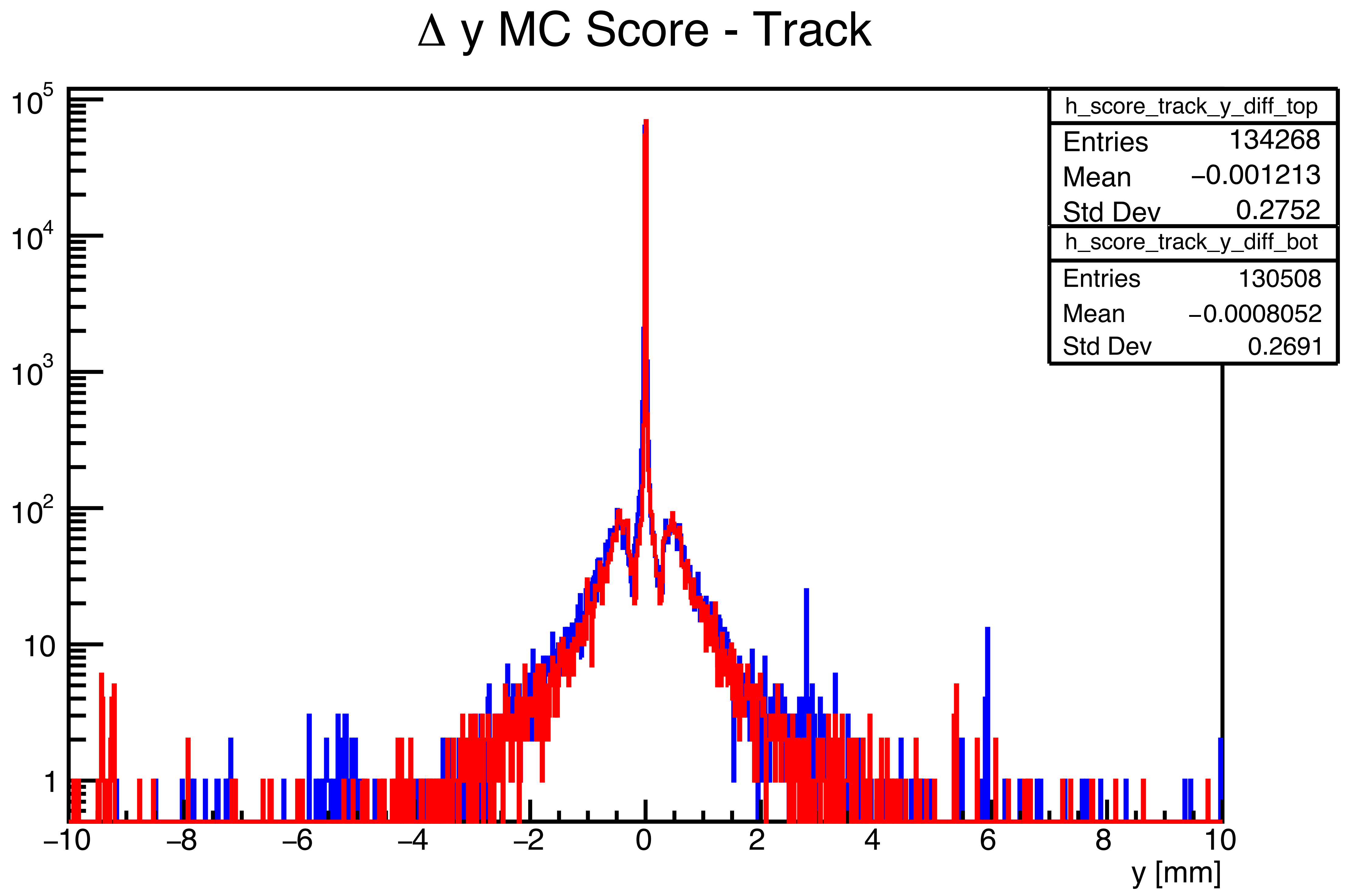
Zero Fringe: Compare Track and Scoring Plane

Positions still match well.



h_score_track_x_diff_top	
Entries	134268
Mean	0.03585
Std Dev	0.6233

h_score_track_x_diff_bot	
Entries	130508
Mean	0.1189
Std Dev	0.6198

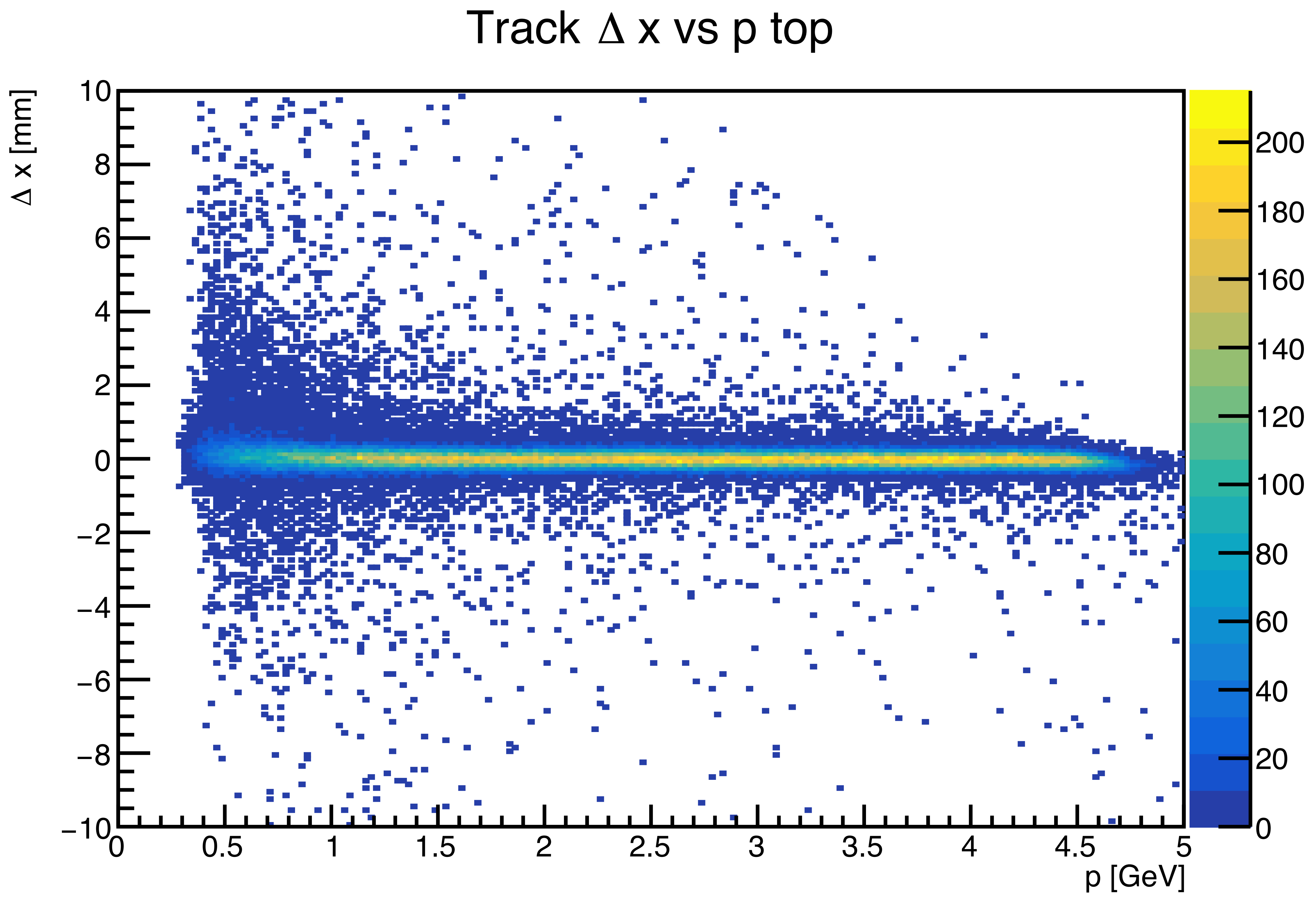


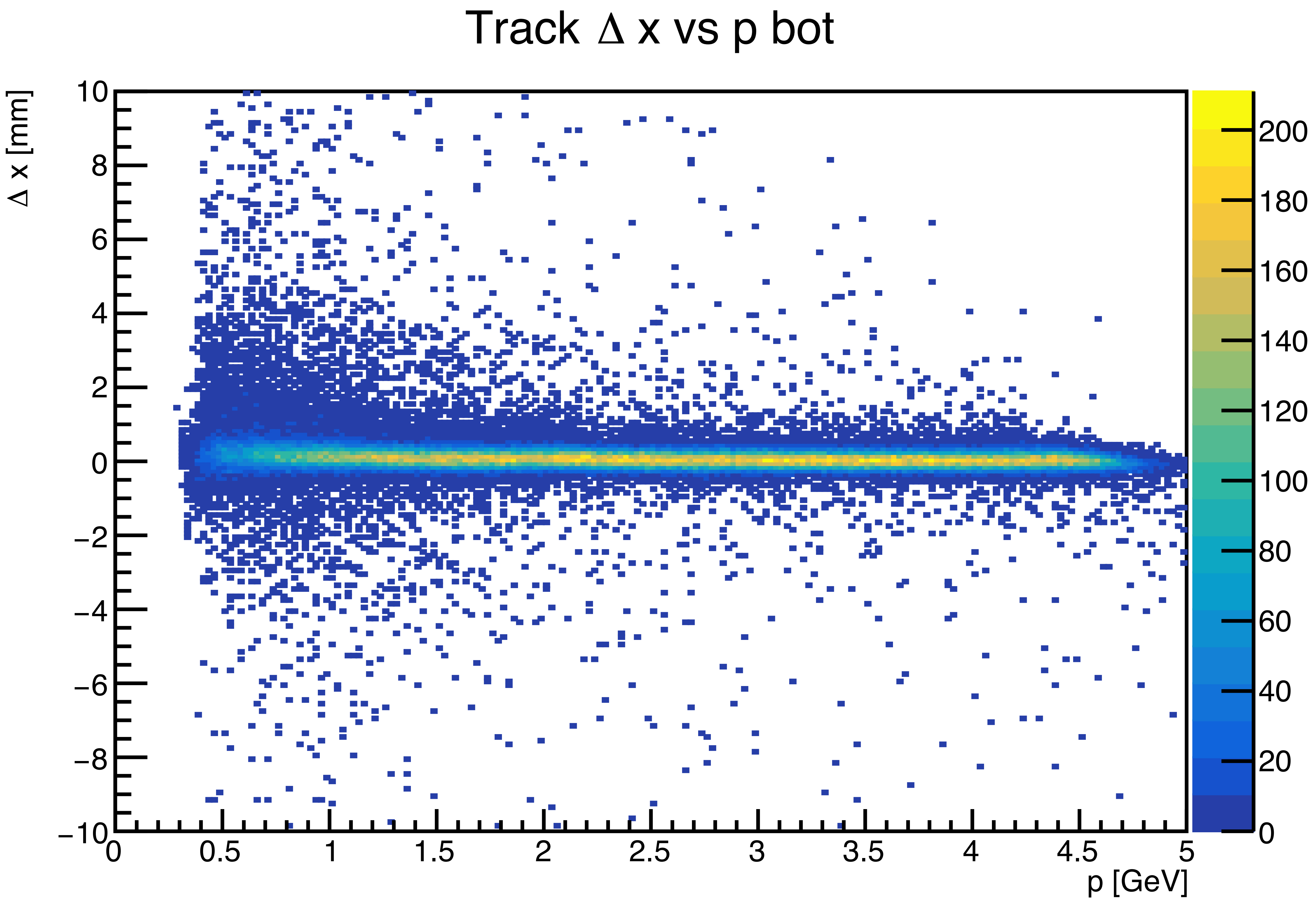
h_score_track_y_diff_top	
Entries	134268
Mean	–0.001213
Std Dev	0.2752

h_score_track_y_diff_bot	
Entries	130508
Mean	–0.0008052
Std Dev	0.2691

Zero Fringe: Compare Track and Scoring Plane

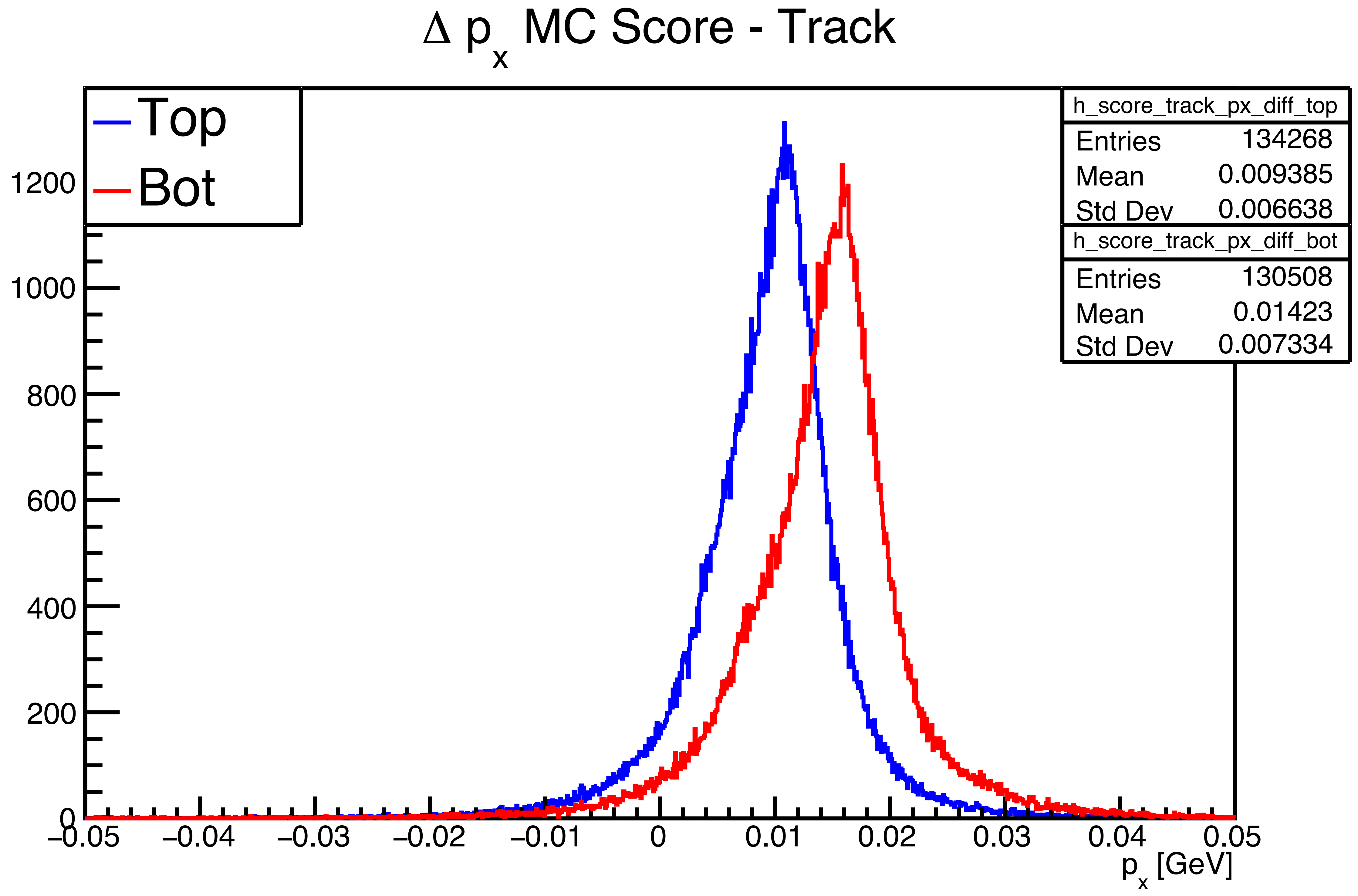
Positions still match well.

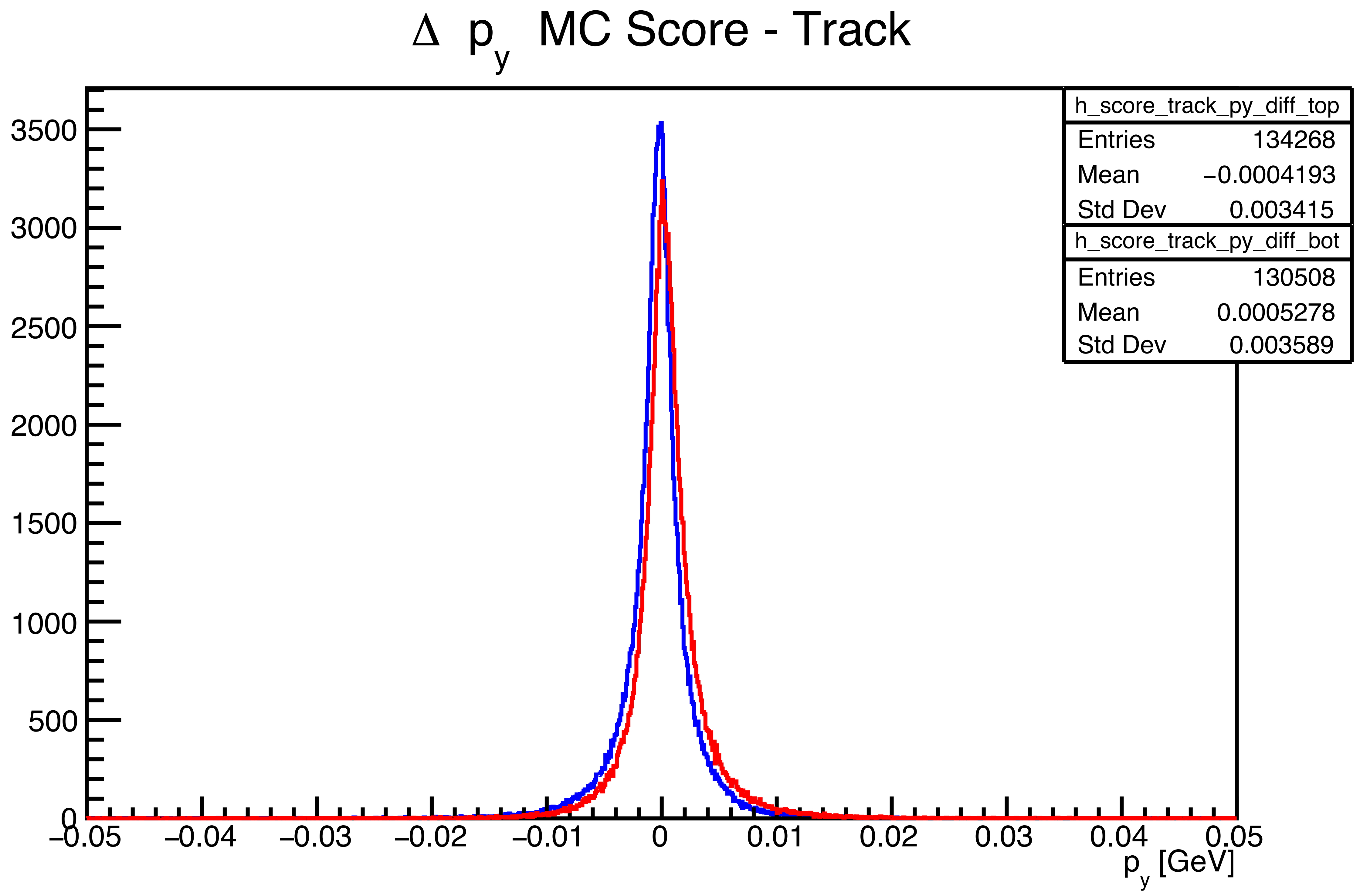

Track Δ x vs p top


Track Δ x vs p bot

Zero Fringe: Compare Track and Scoring Plane

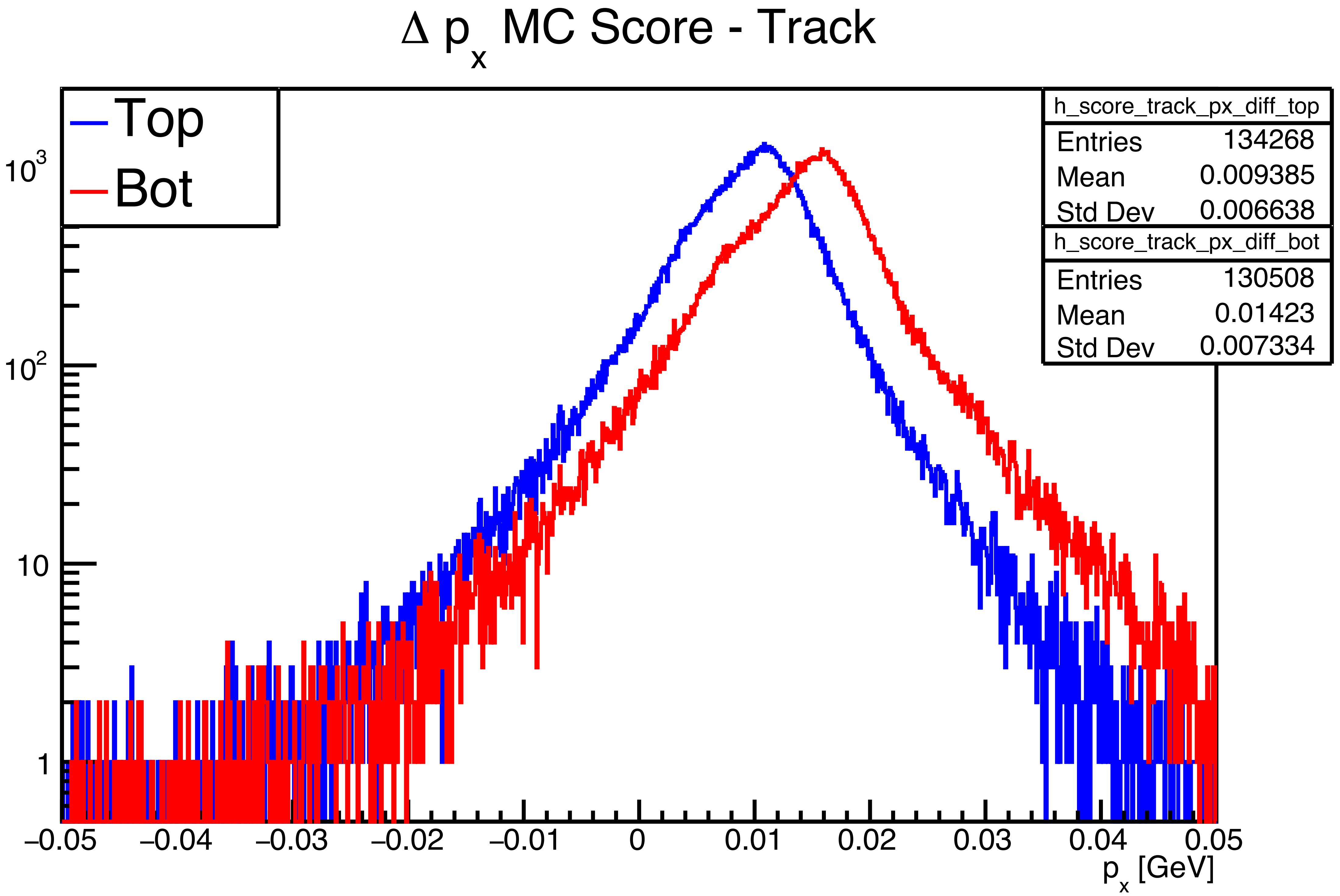
Momentum plot, show some more distortion.

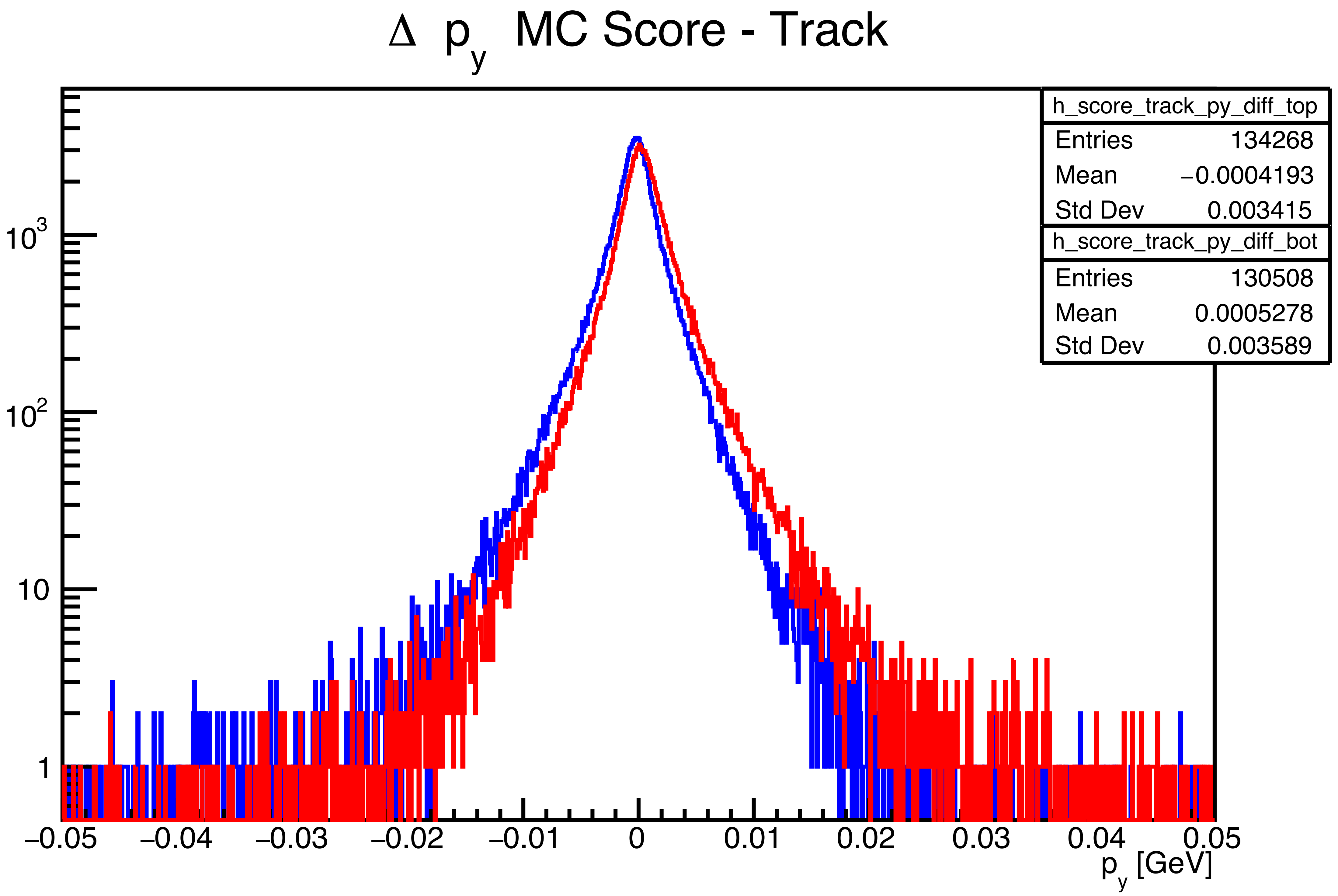




Zero Fringe: Compare Track and Scoring Plane

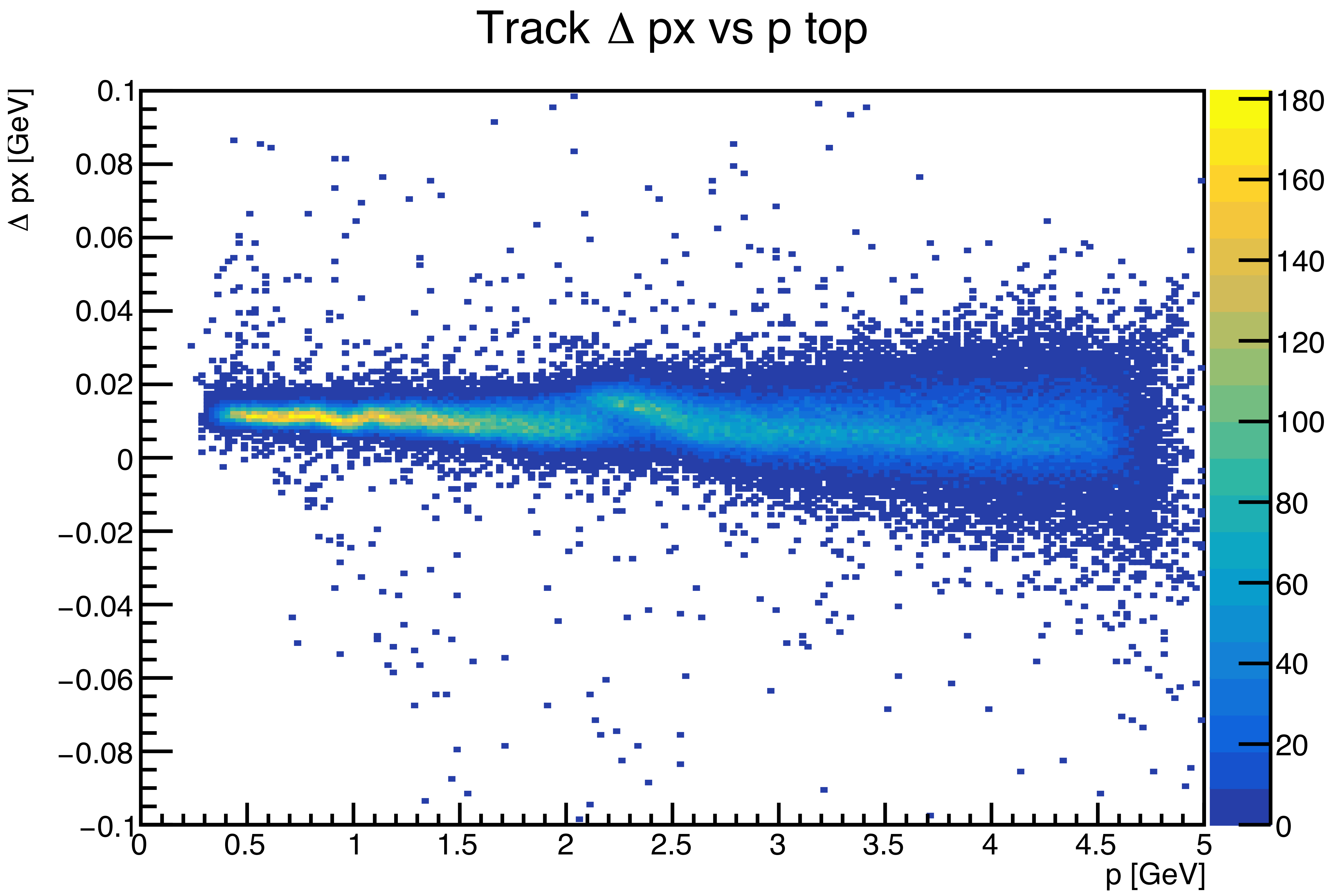
Momentum plot, show some more distortion.



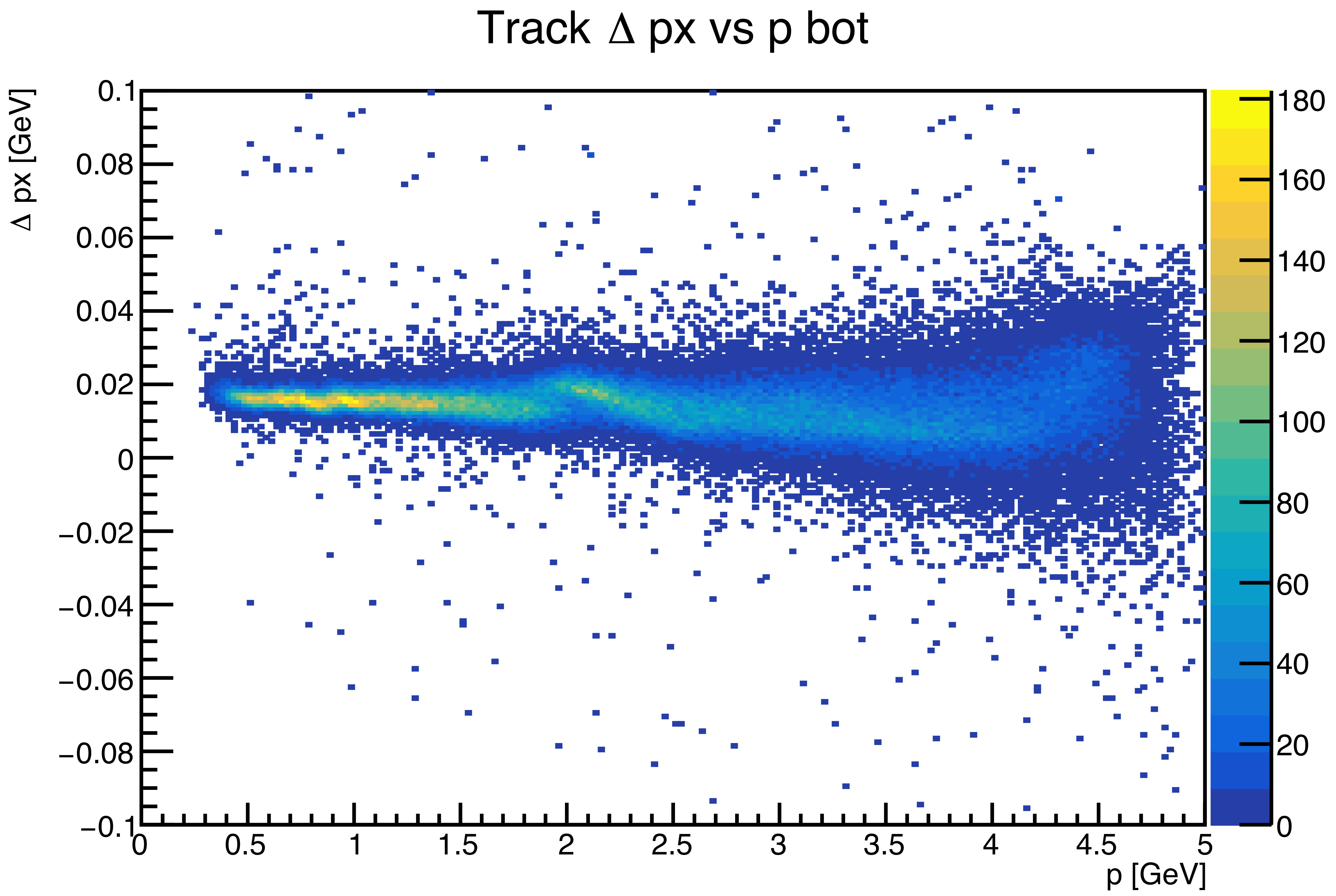


Zero Fringe: Compare Track and Scoring Plane

Momentum plot, show some more distortion.



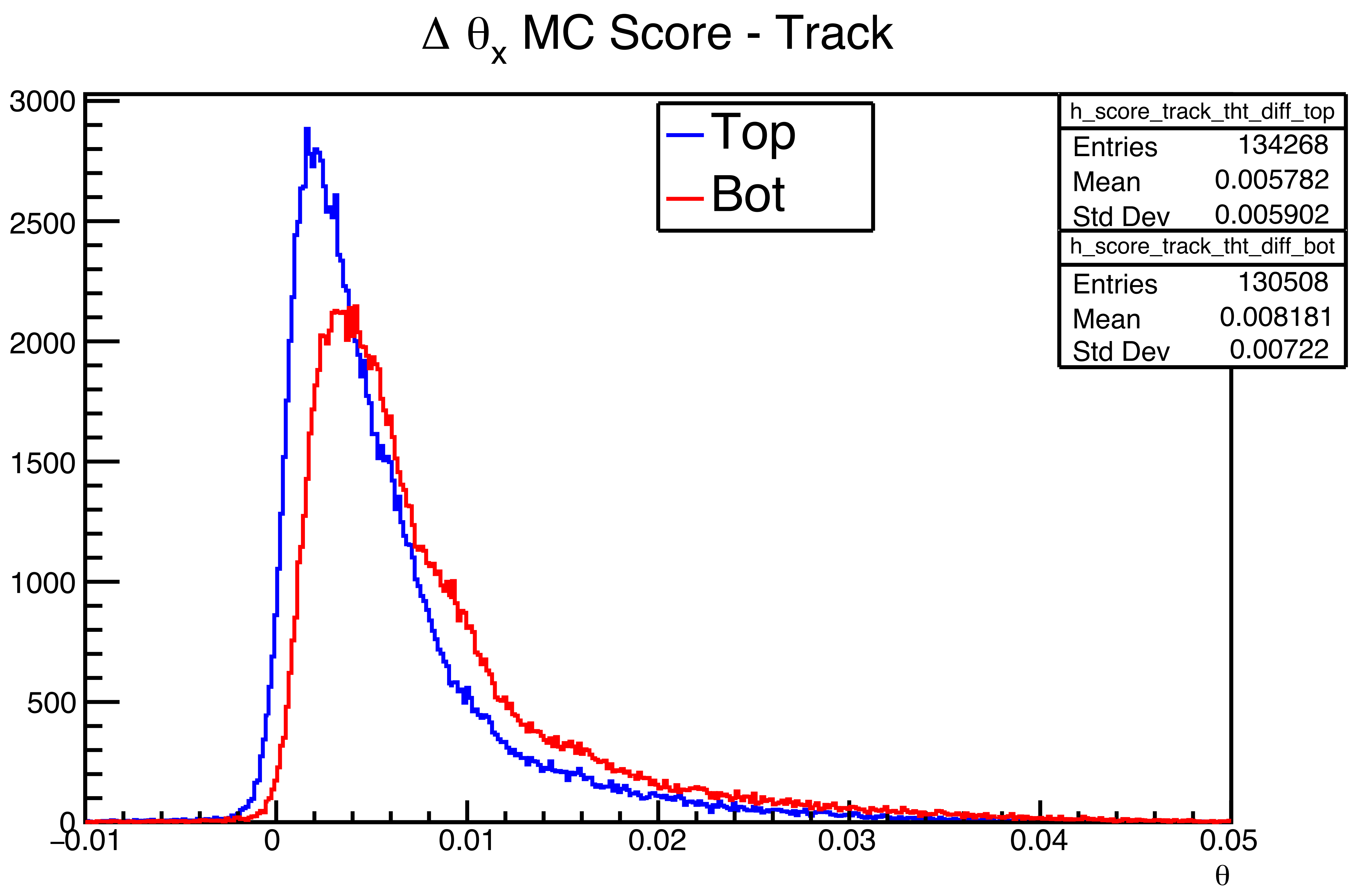
Track Δ px vs p top

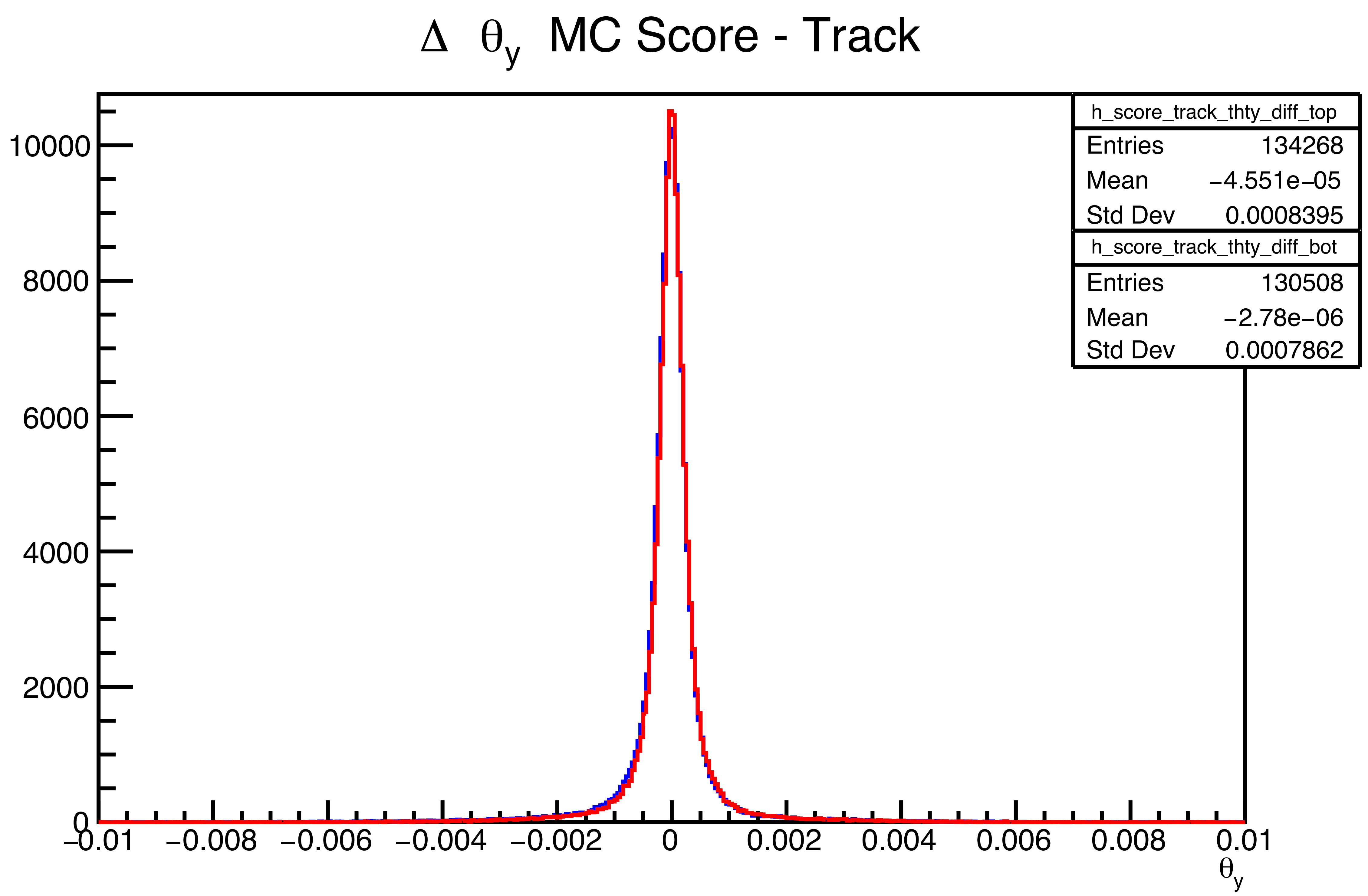


Track Δ px vs p bot

Zero Fringe: Compare Track and Scoring Plane

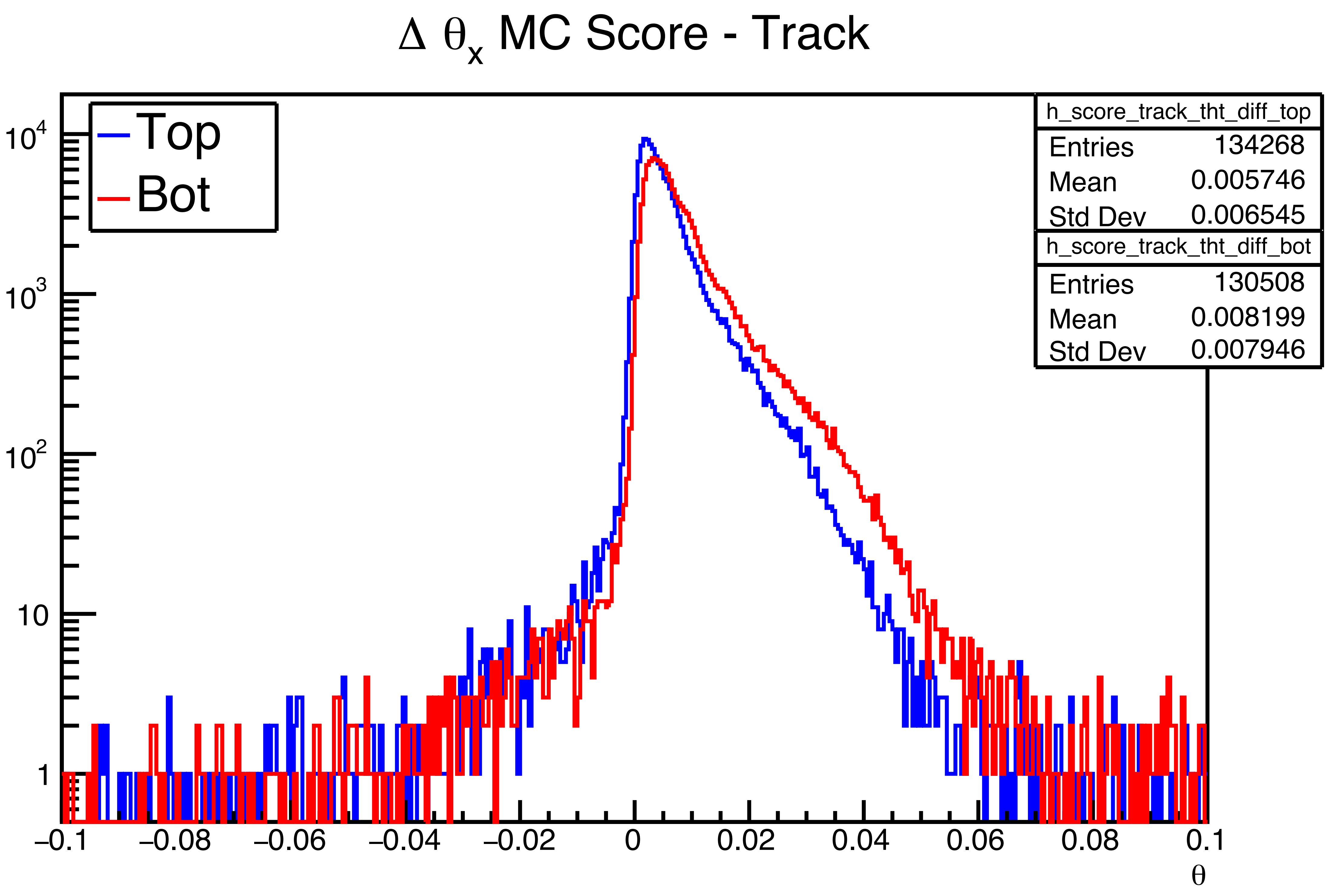
Angles, linear plot, show some distortion.

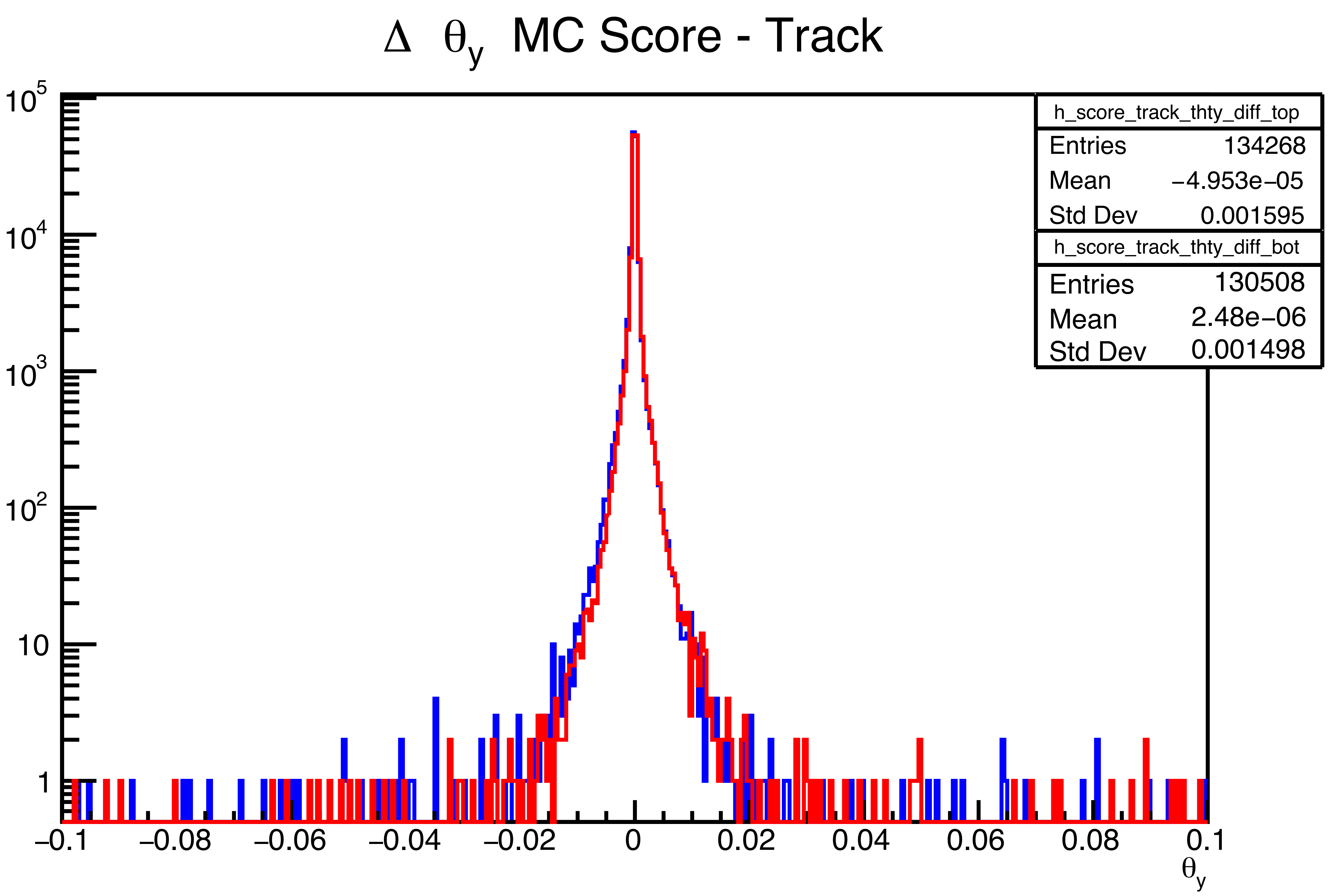




Zero Fringe: Compare Track and Scoring Plane

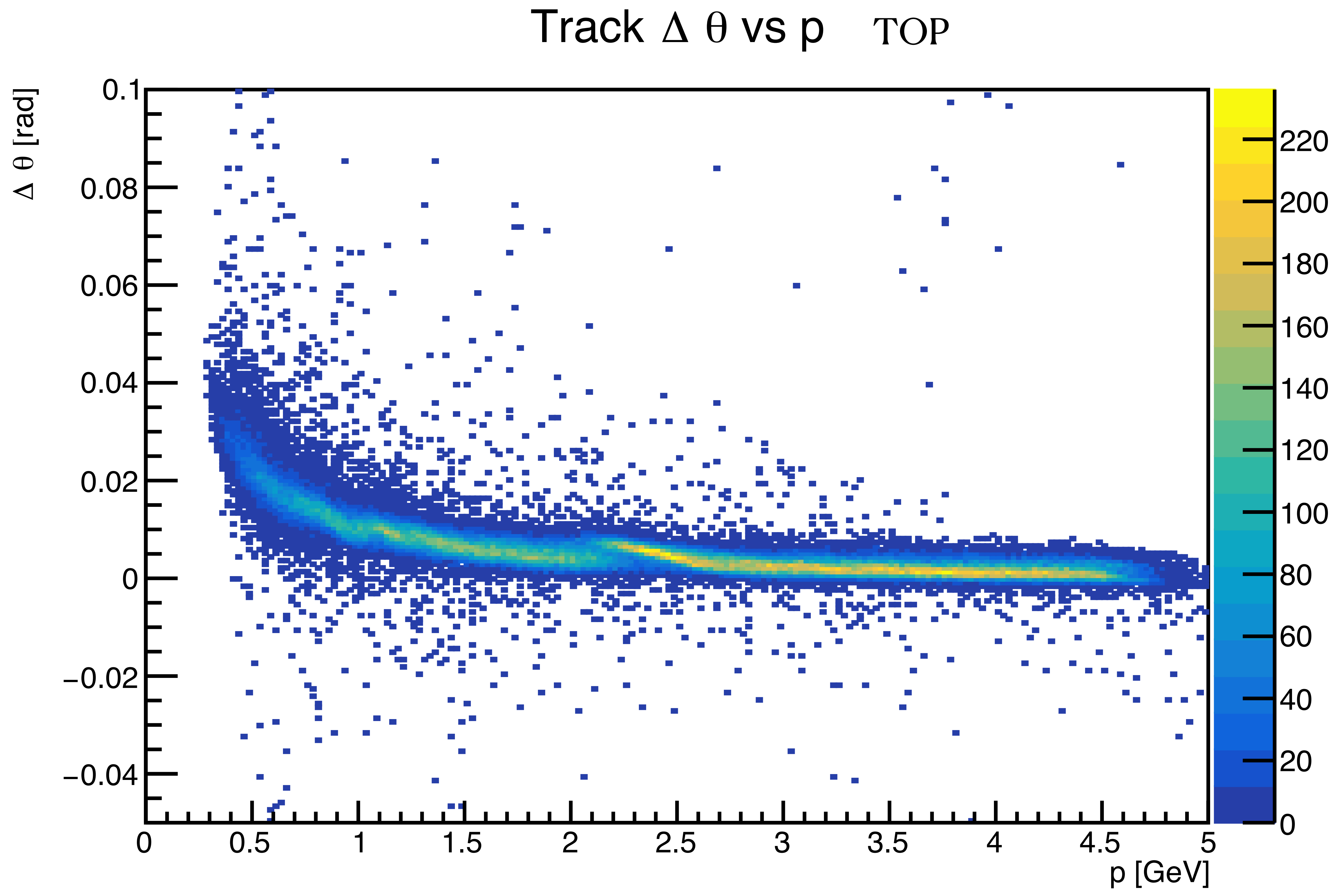
Angles, linear plot, show some distortion.

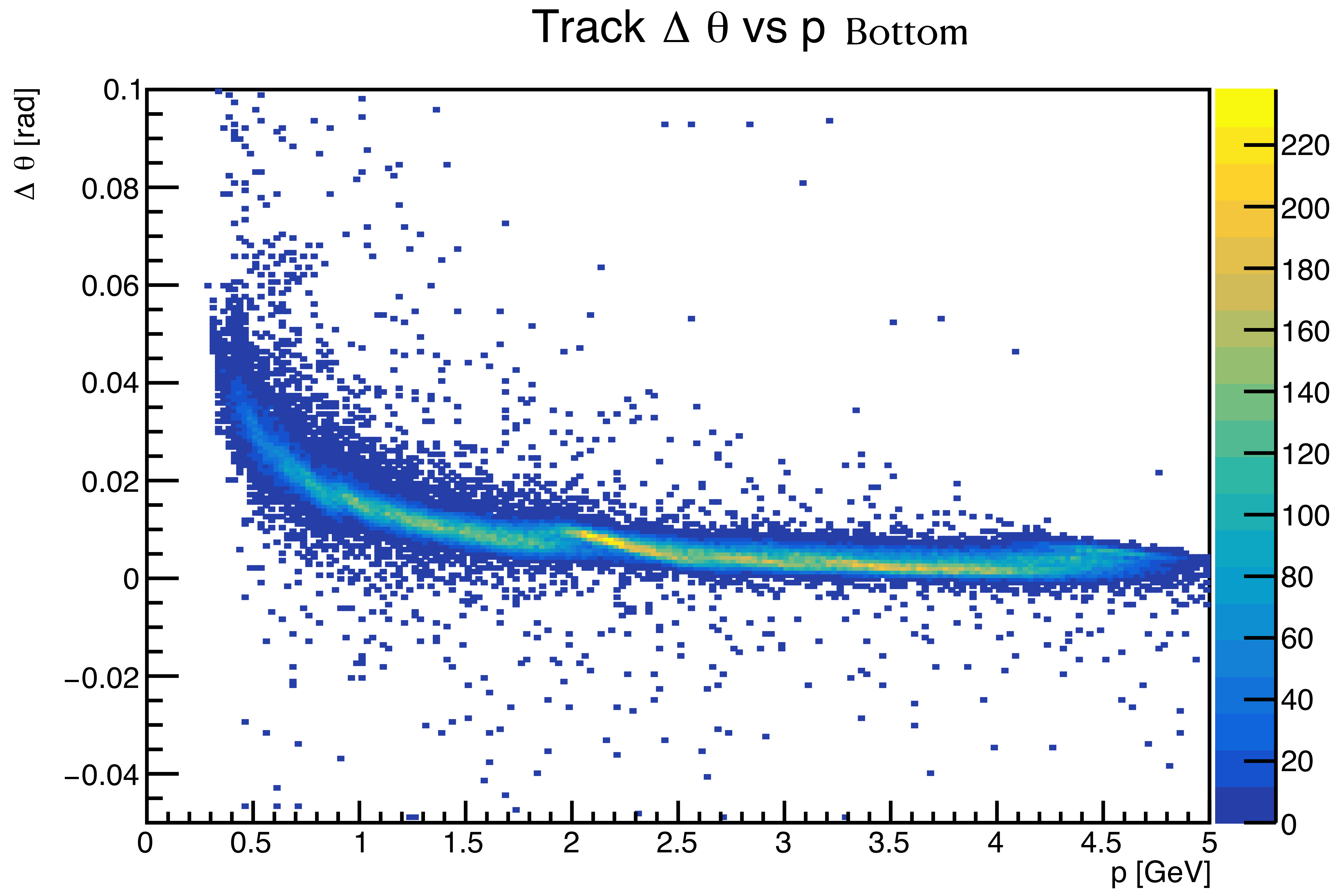




Zero Fringe: Compare Track and Scoring Plane

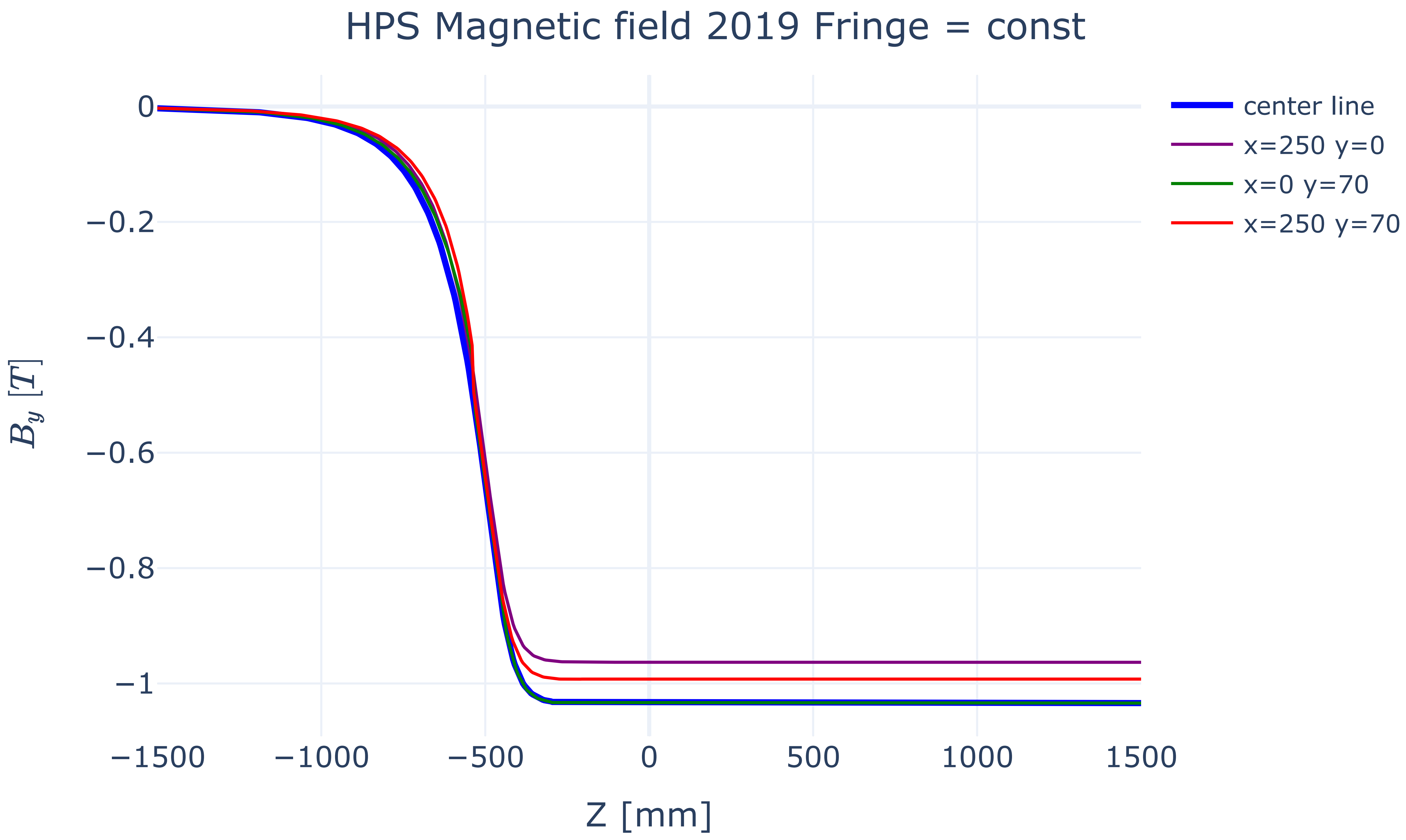
Angles, linear plot, show some distortion.

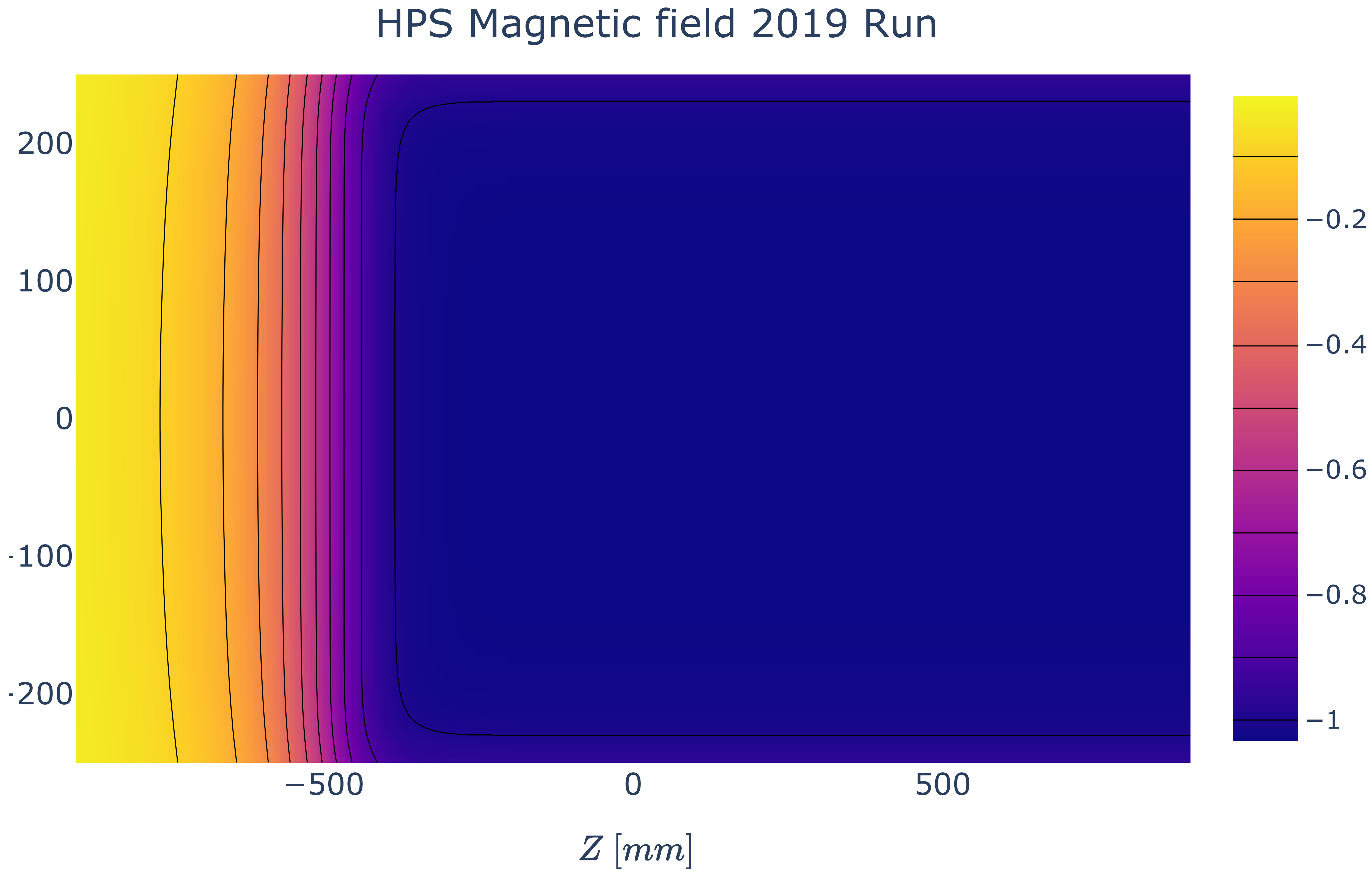




Fringe field = const

- To test the behavior of the code without a fringe field, we set the fringe field to central value after the last tracker. So By = -1.034 when z> 500 (detector coordinates). This means the last tracker all the way to ECal is in a constant field.

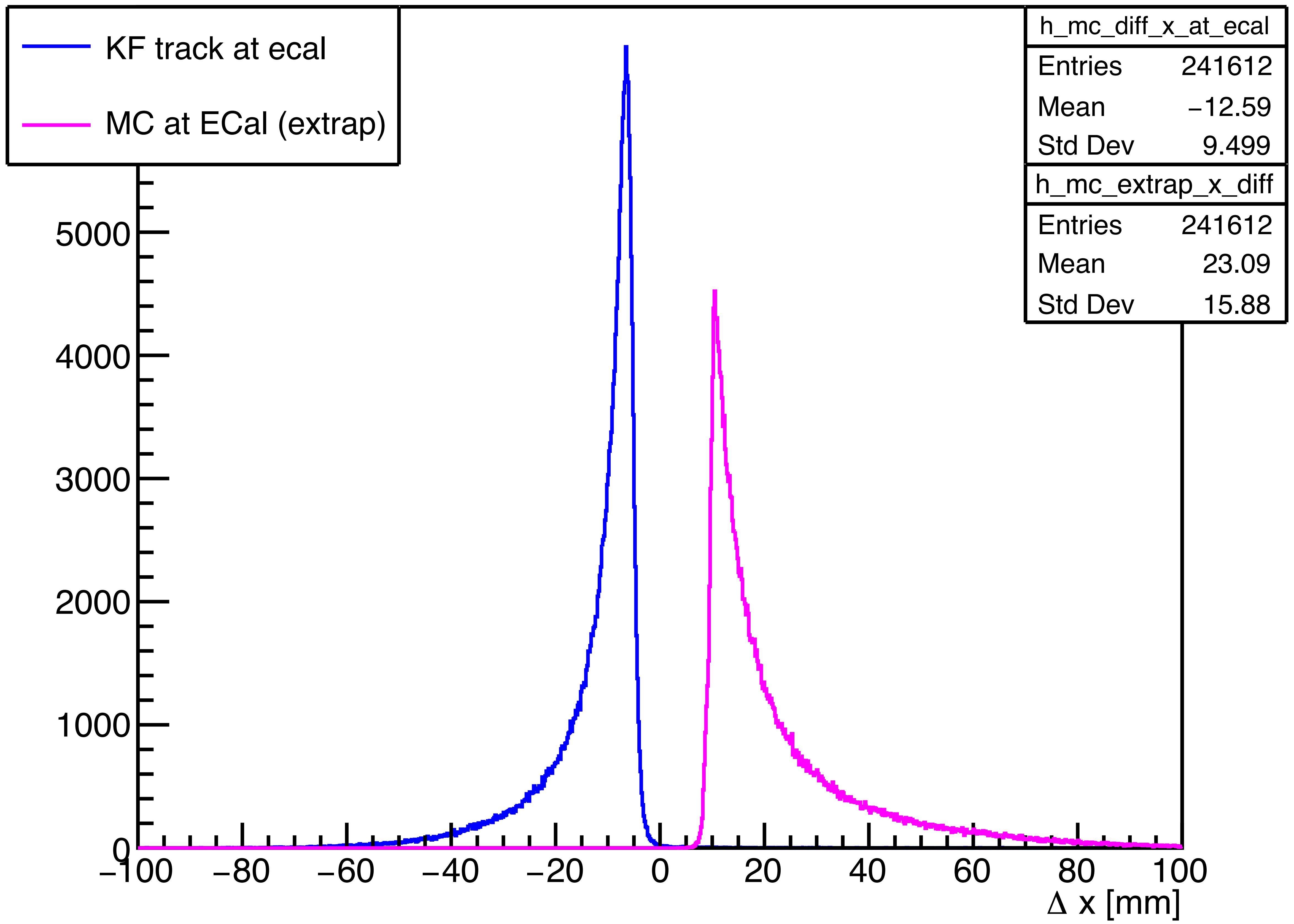

HPS Magnetic field 2019 Fringe = const


HPS Magnetic field 2019 Run

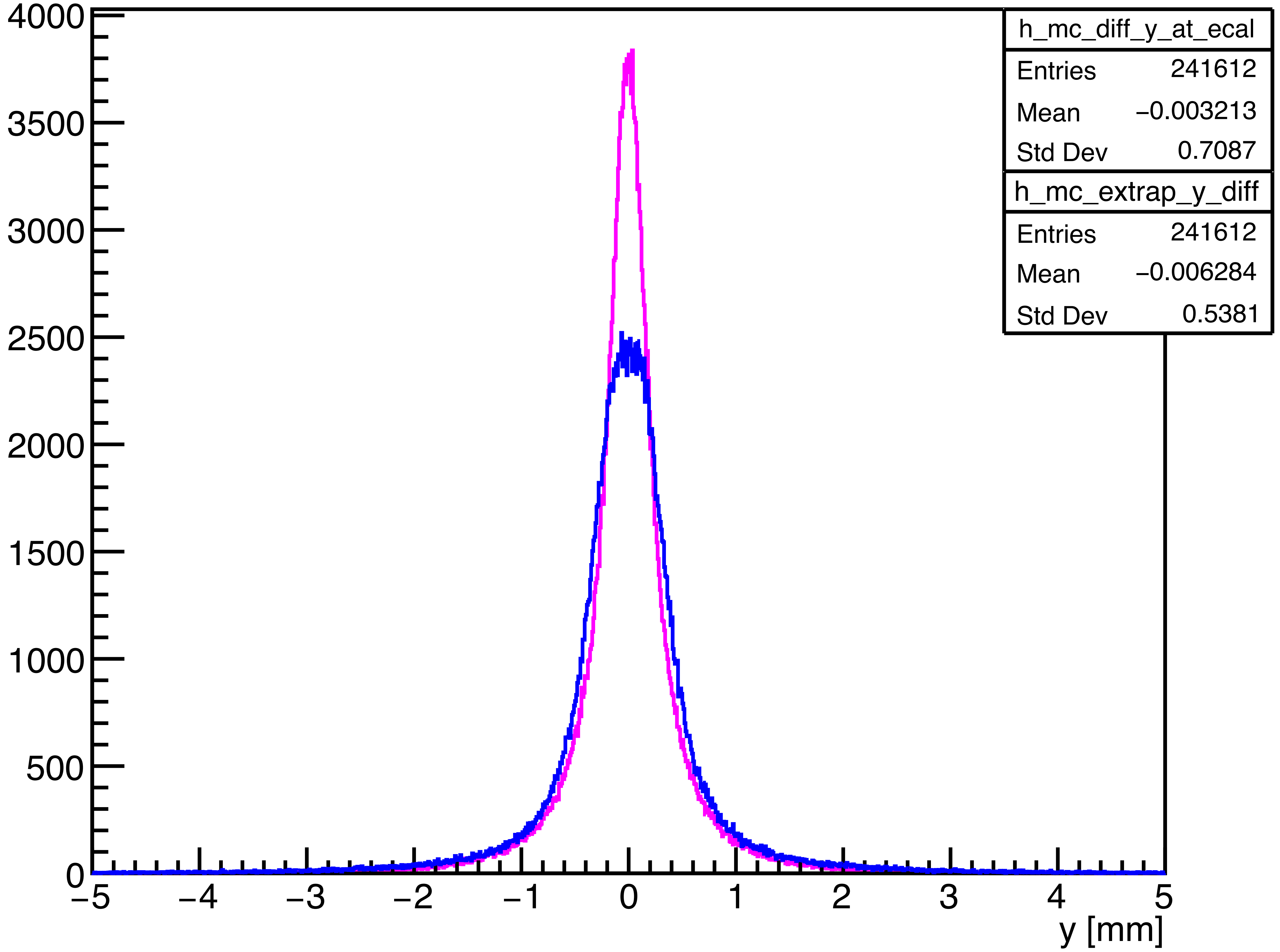
Constant Fringe: Compare position at ECal

The KF track extrapolated to the ECal now goes the other way, while the simple extrapolation still gives the wrong result as previous. This was unexpected.

X difference.

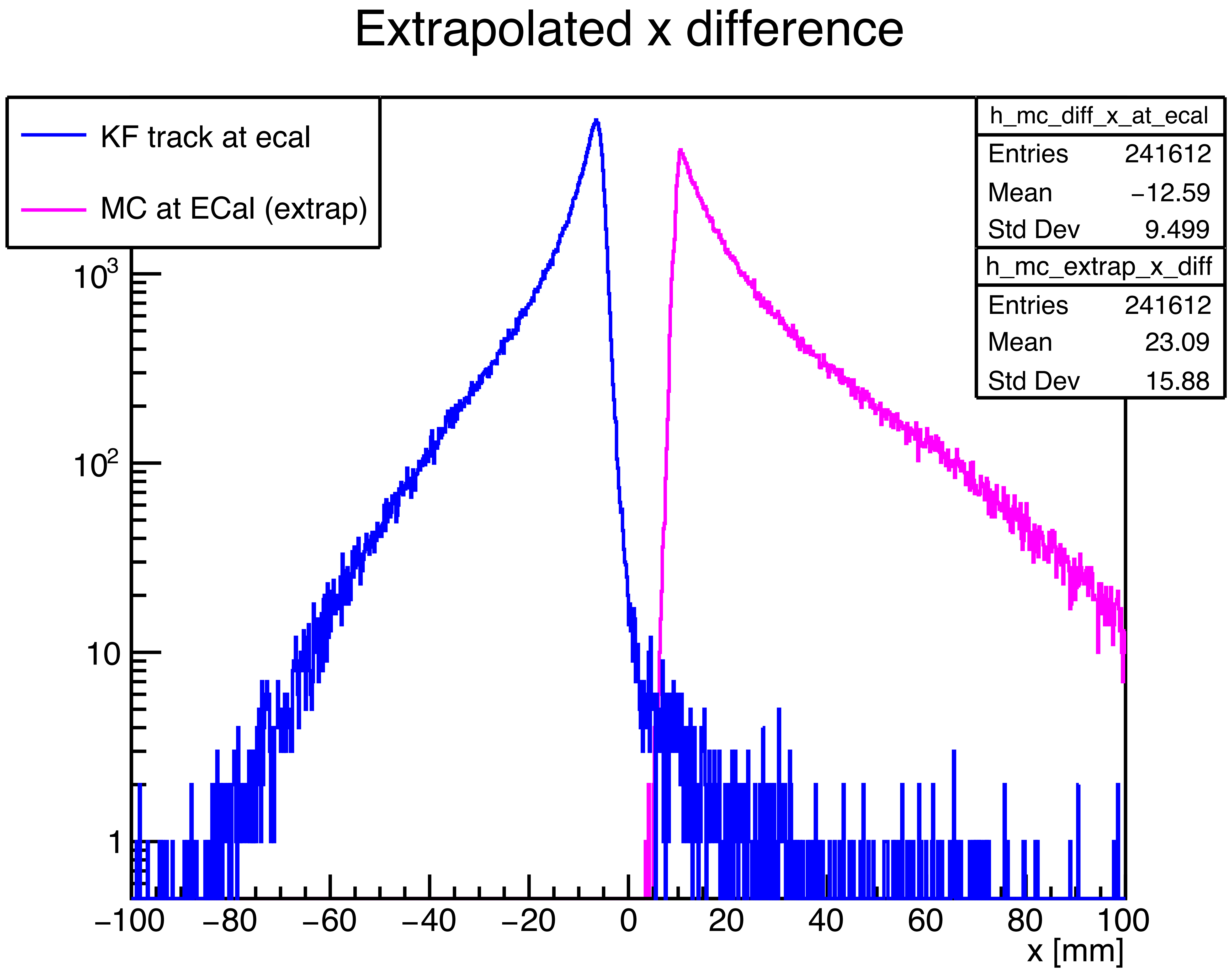


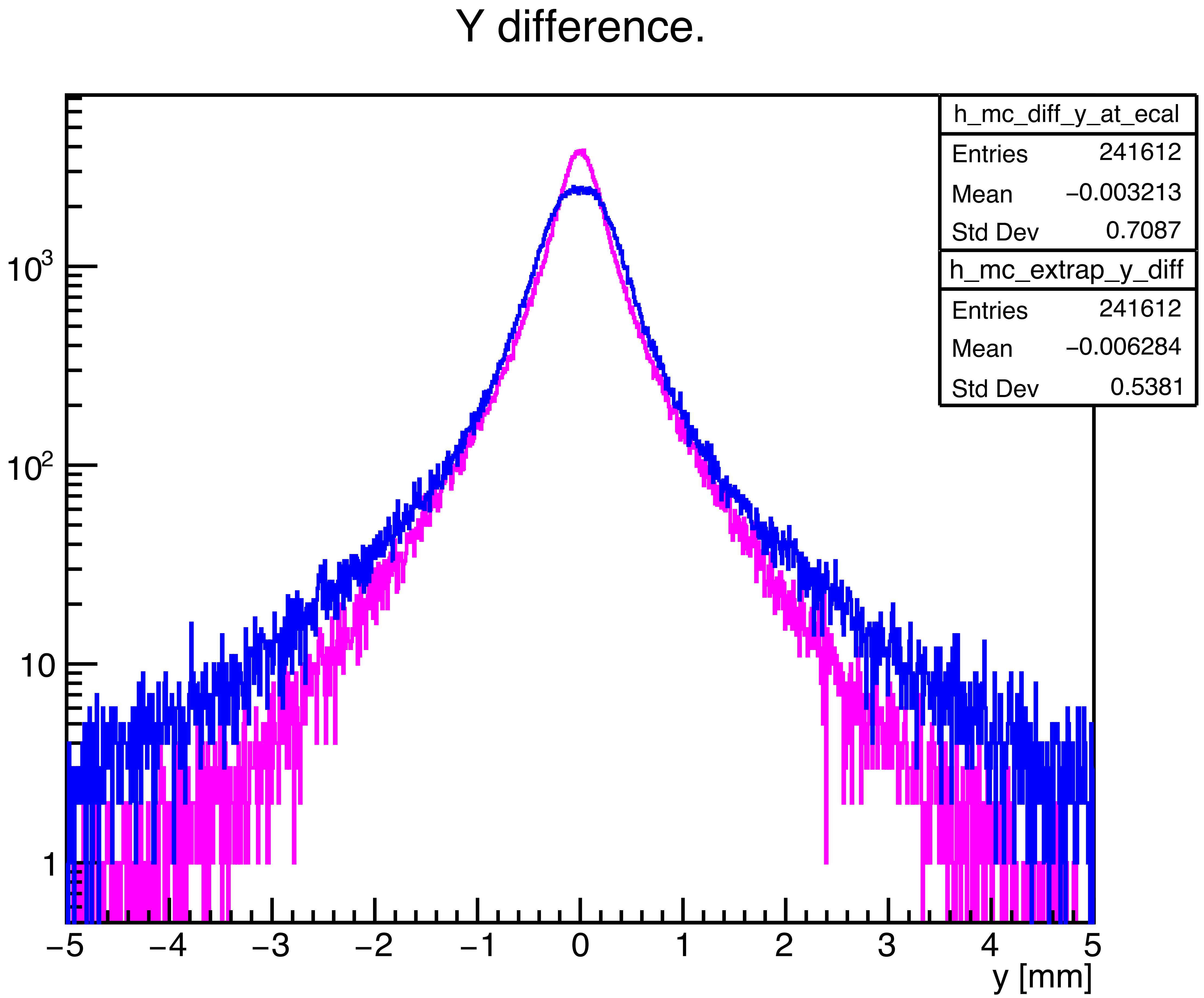
Y difference.



Constant Fringe: Compare position at ECal

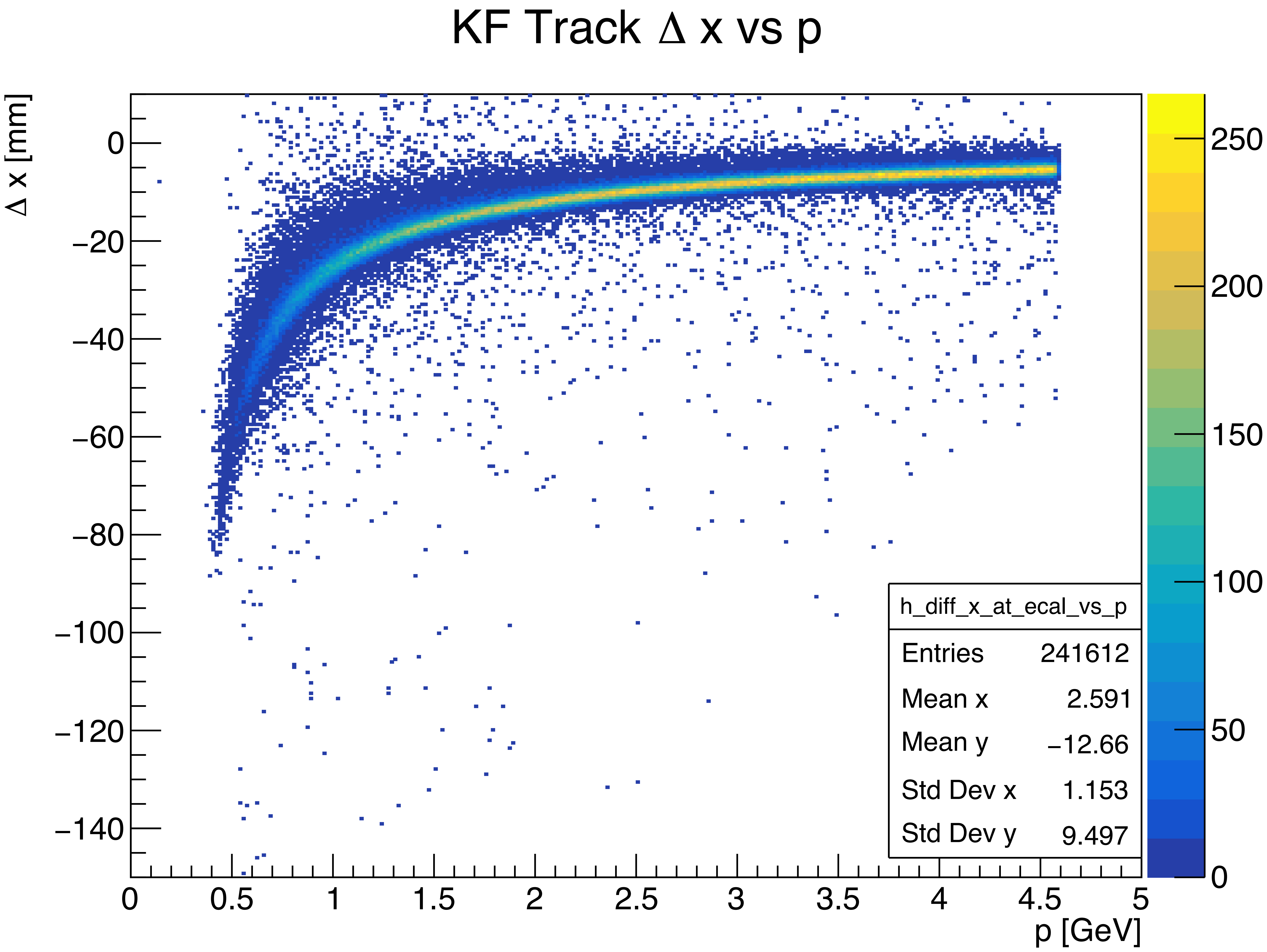
The KF track extrapolated to the ECal now goes the other way, while the simple extrapolation still gives the wrong result as previous. This was unexpected.





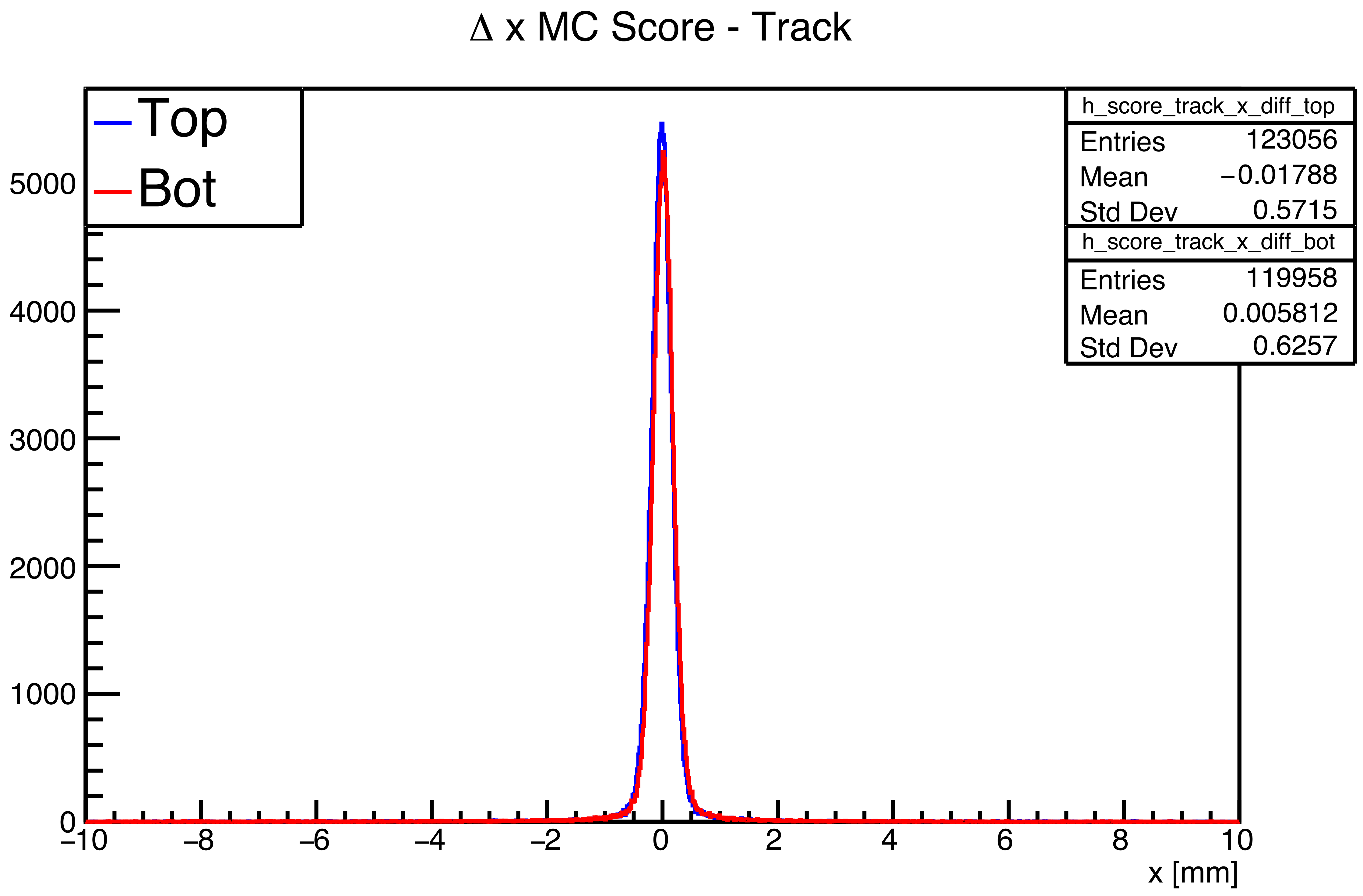
Constant Fringe: Compare position at ECal

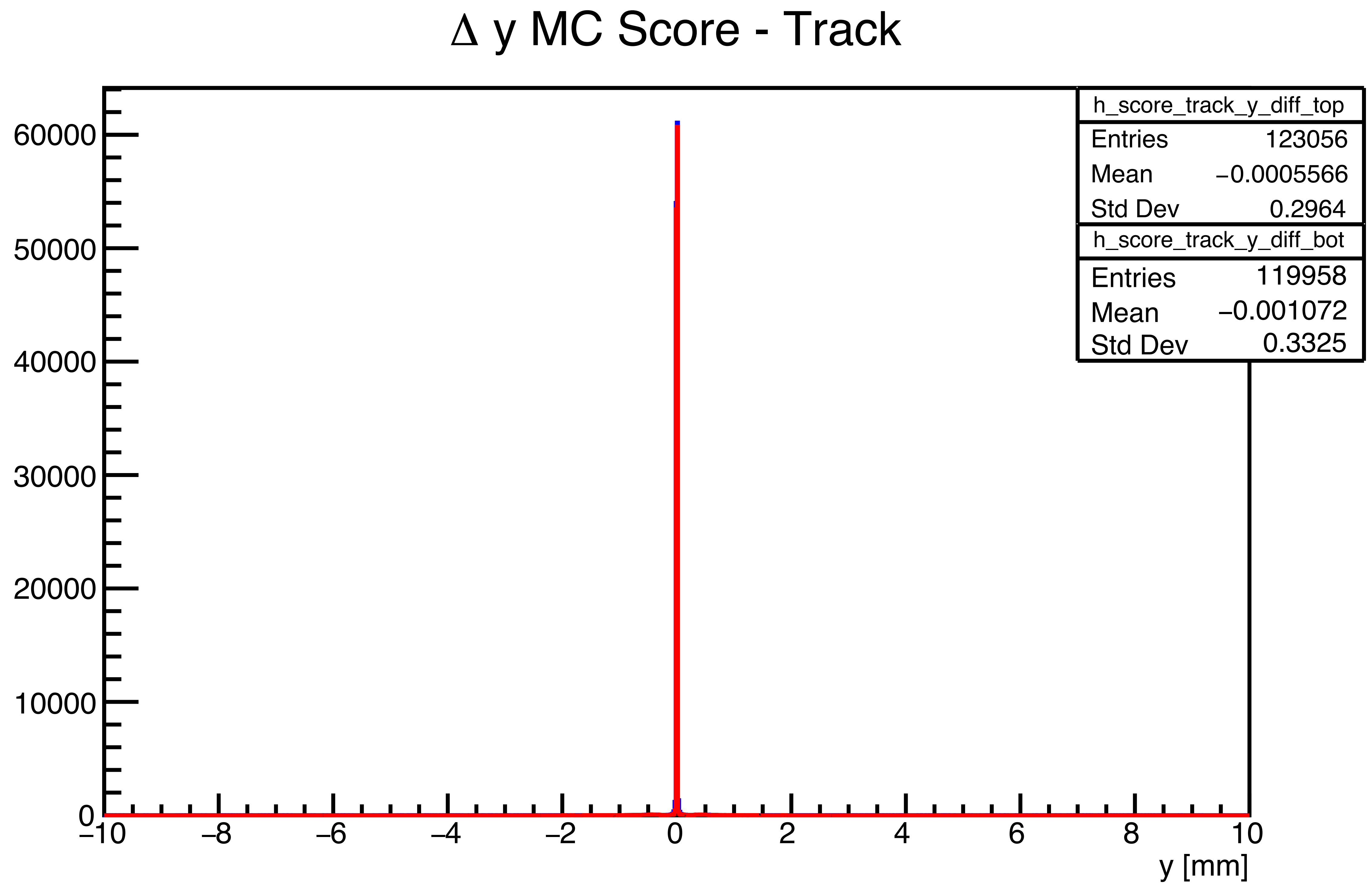
The KF track extrapolated to the ECal now goes the other way, while the simple extrapolation still gives the wrong result as previous. This was unexpected.



Const Fringe: Compare Track and Scoring Plane

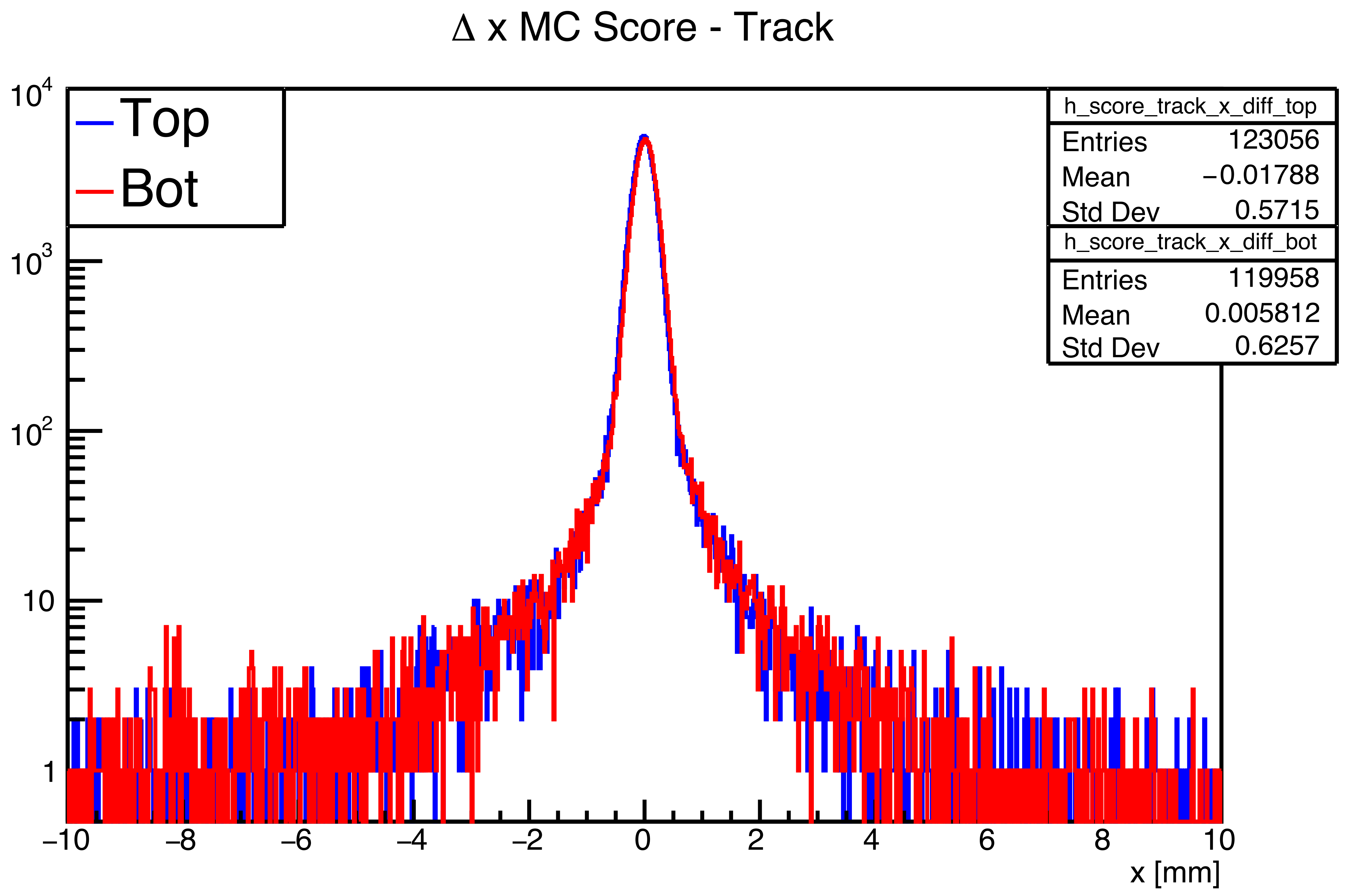
Positions still match well.

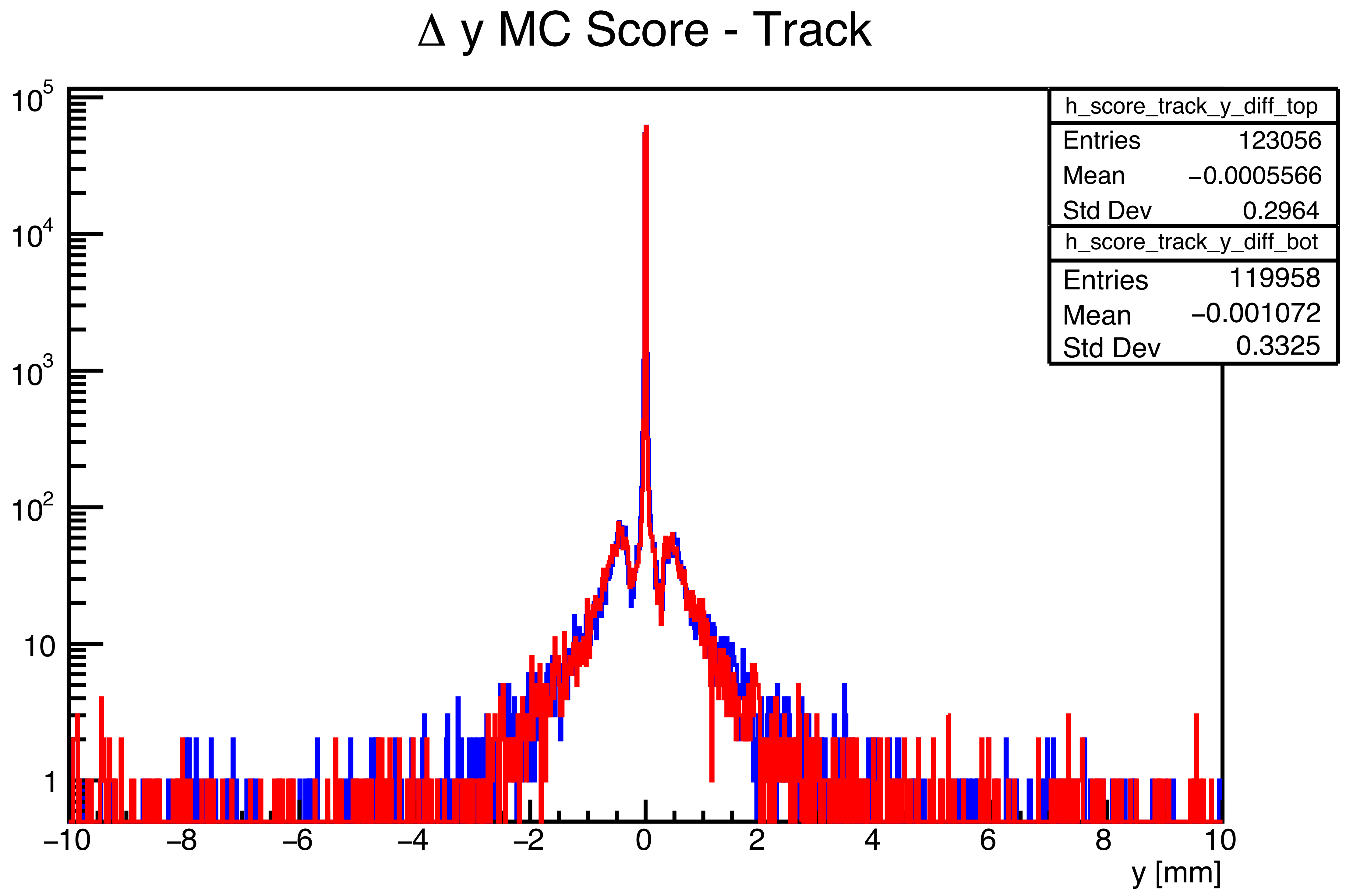




Const Fringe: Compare Track and Scoring Plane

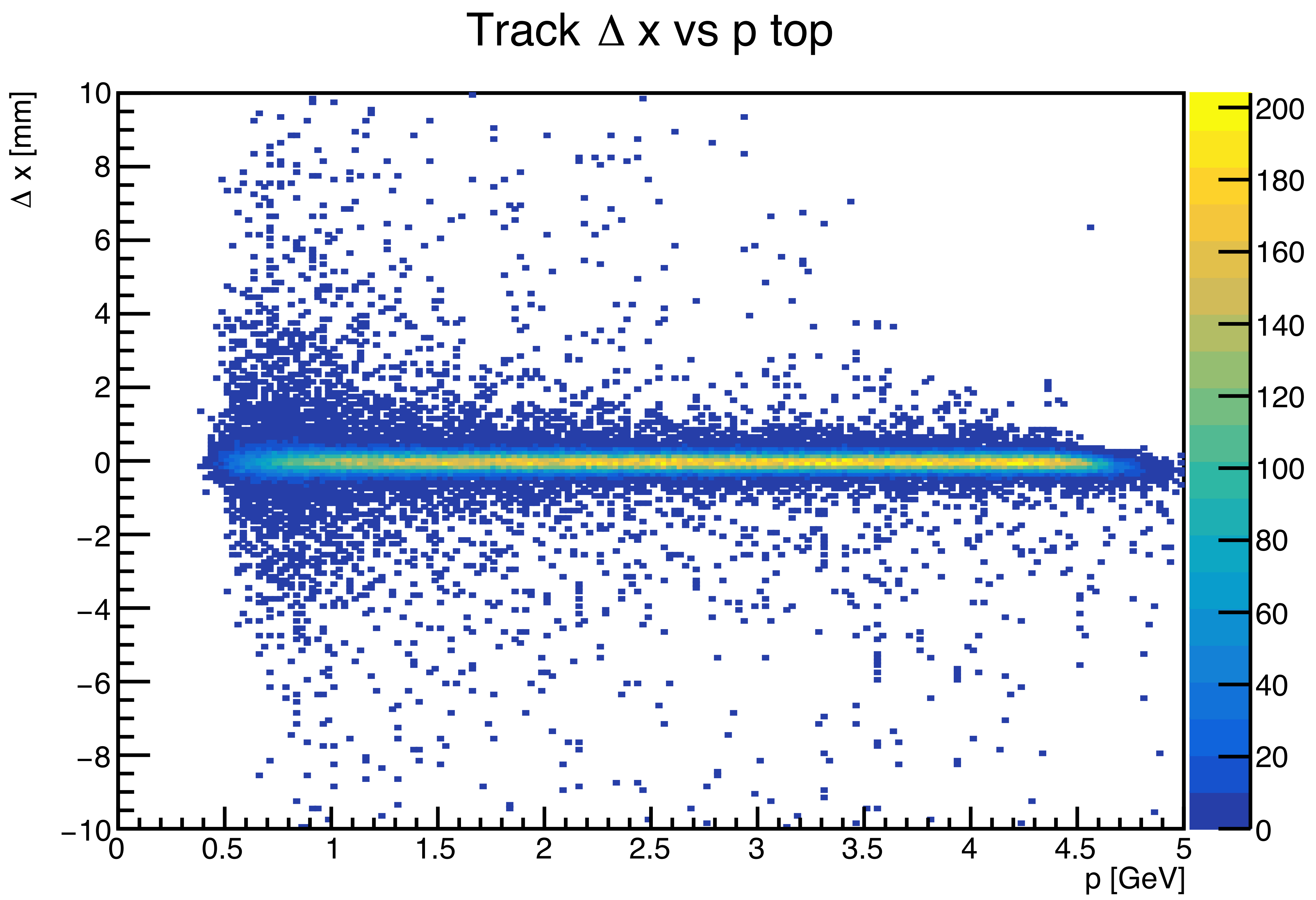
Positions still match well.

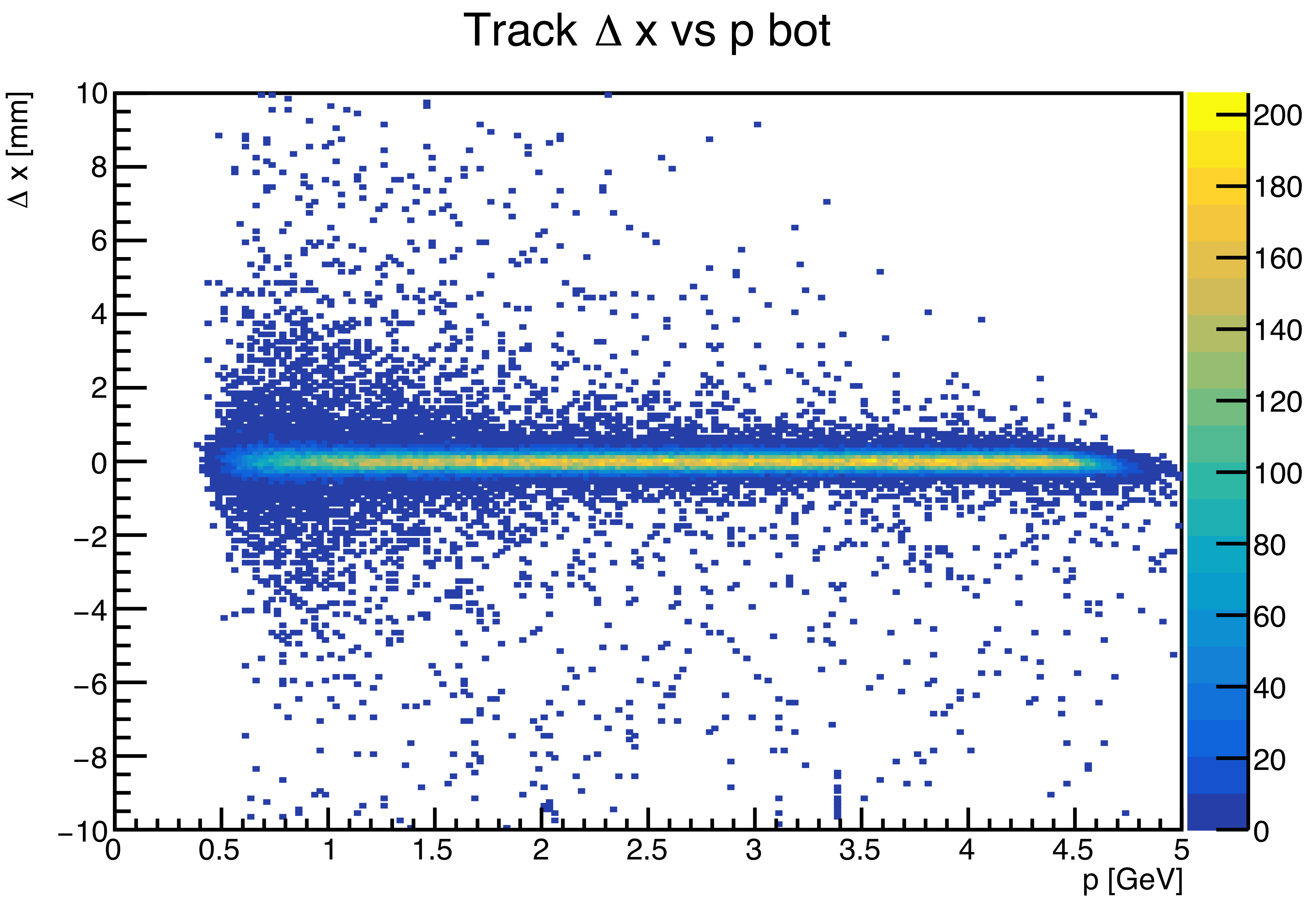




Const Fringe: Compare Track and Scoring Plane

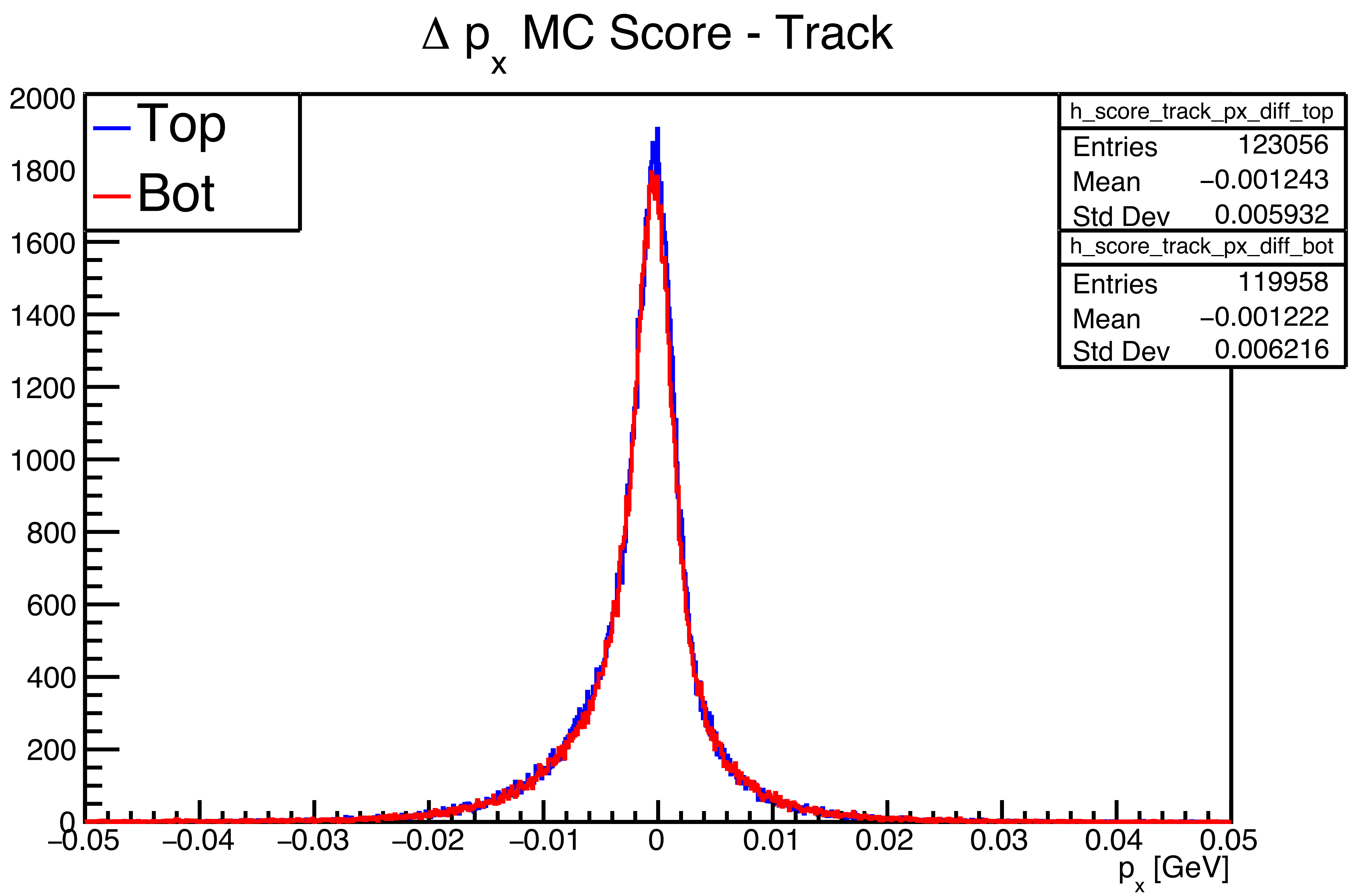
Positions still match well.

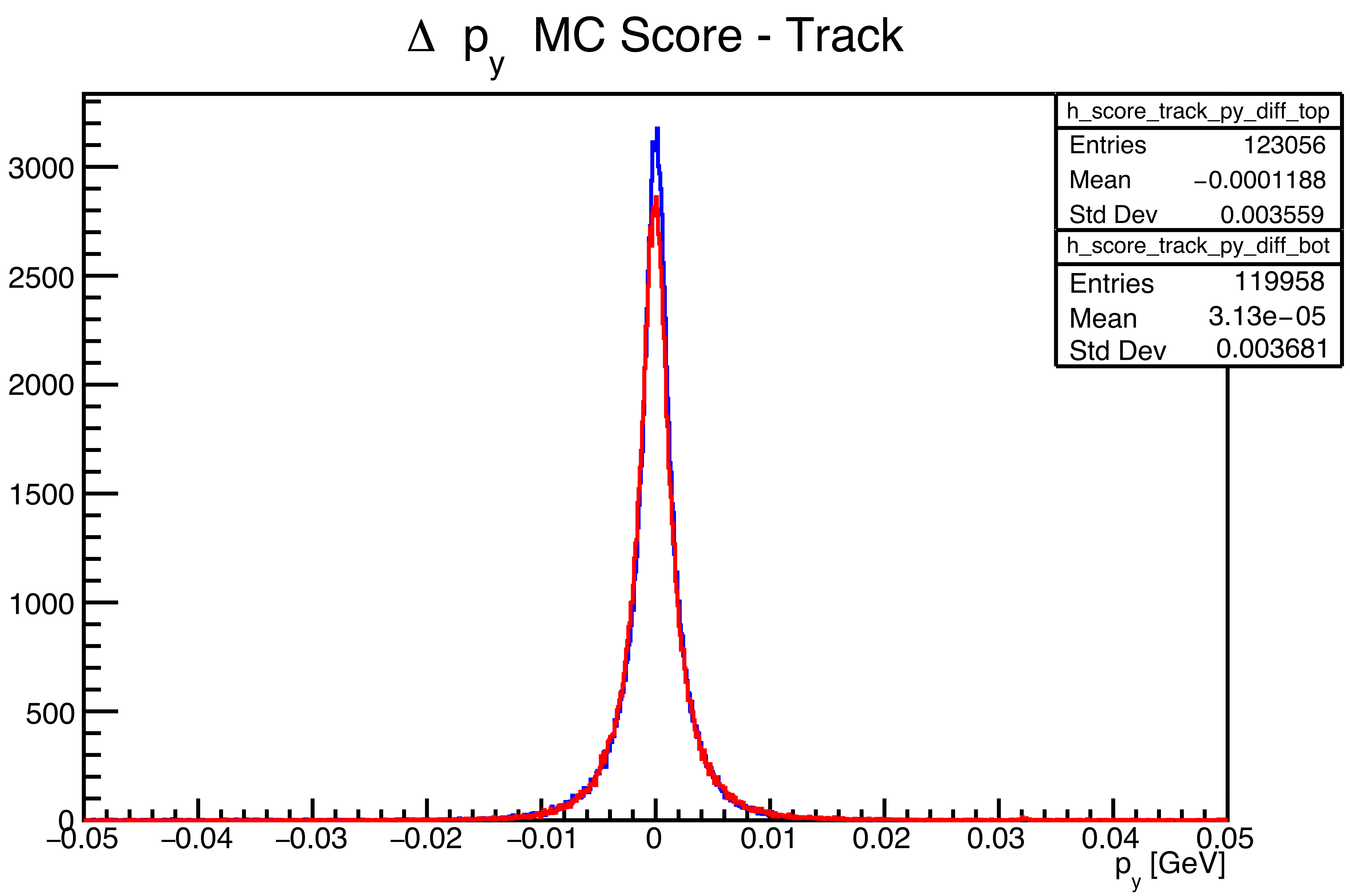

Track Δ x vs p top


Track Δ x vs p bot

Const Fringe: Compare Track and Scoring Plane

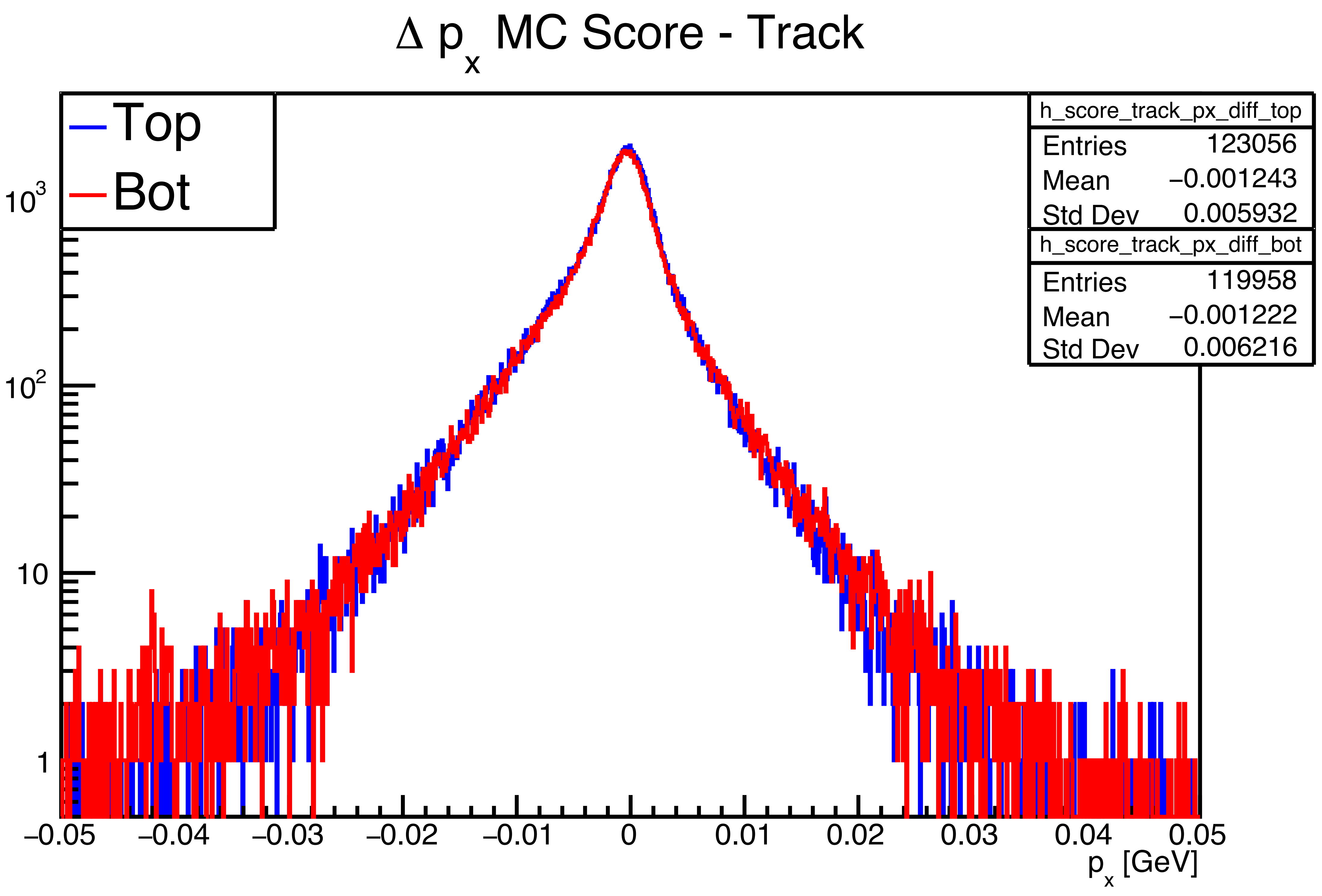
Momentum also still match well.

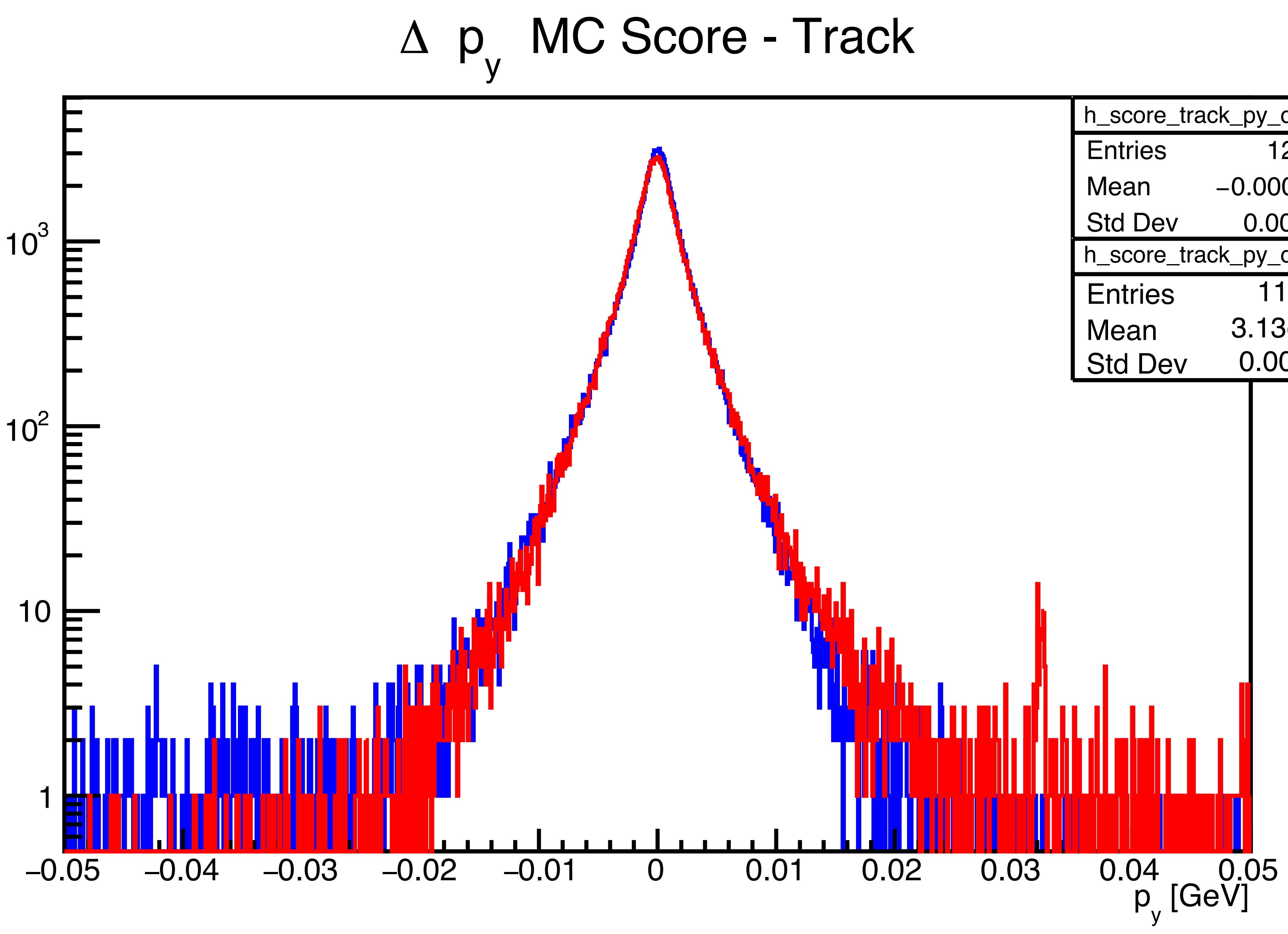




Const Fringe: Compare Track and Scoring Plane

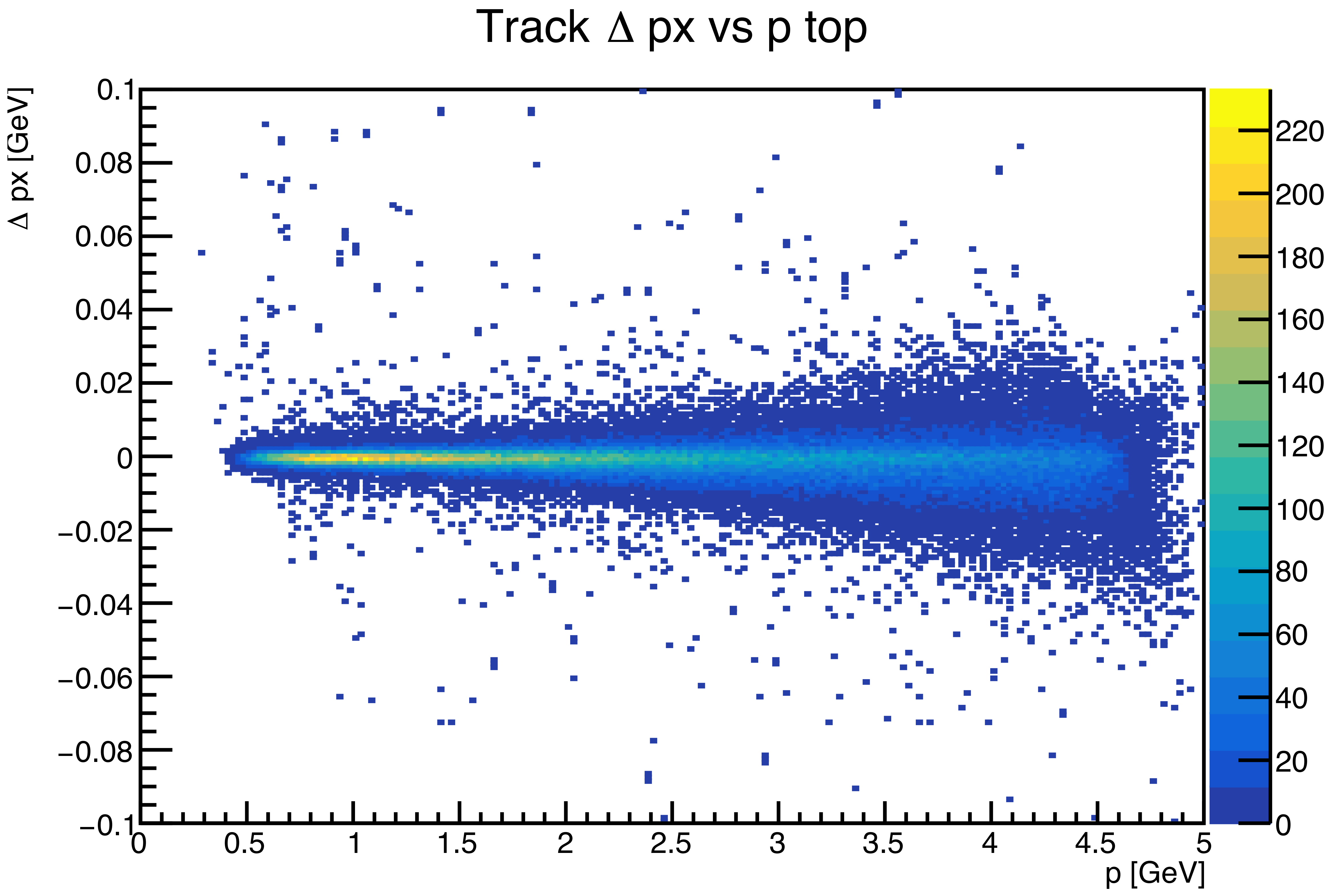
Momentum also still match well.

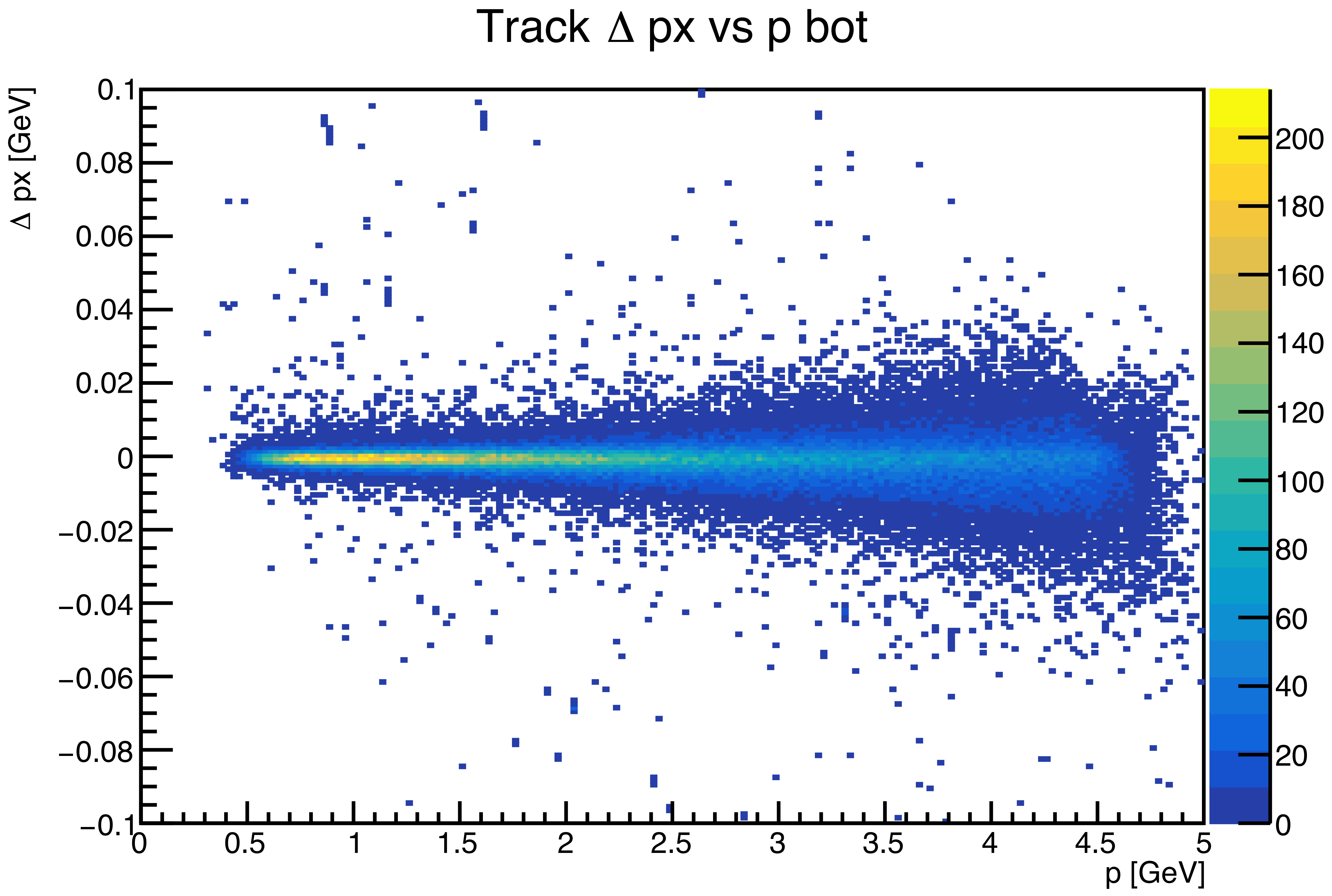




Const Fringe: Compare Track and Scoring Plane

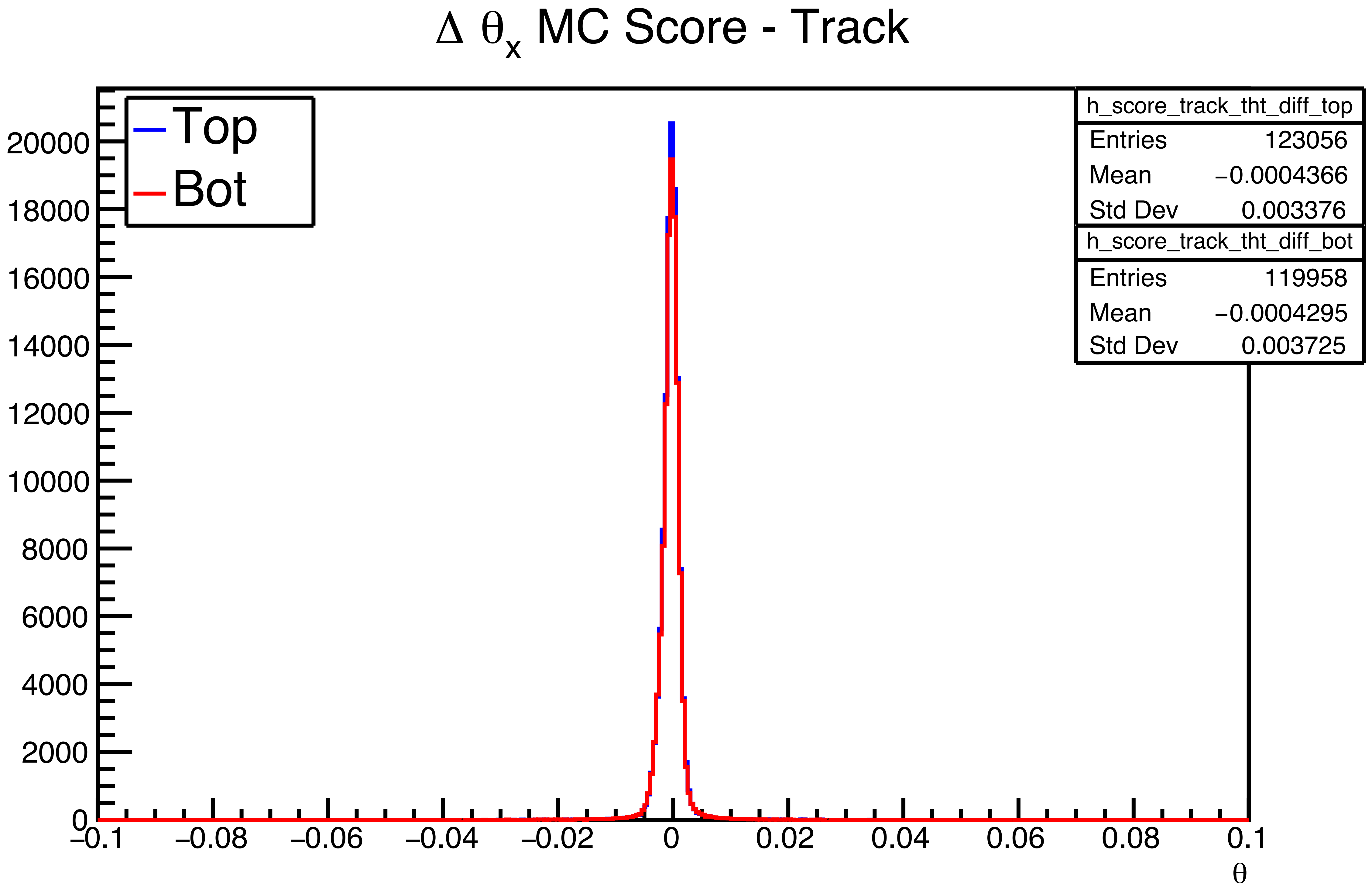
Momentum also still match well.

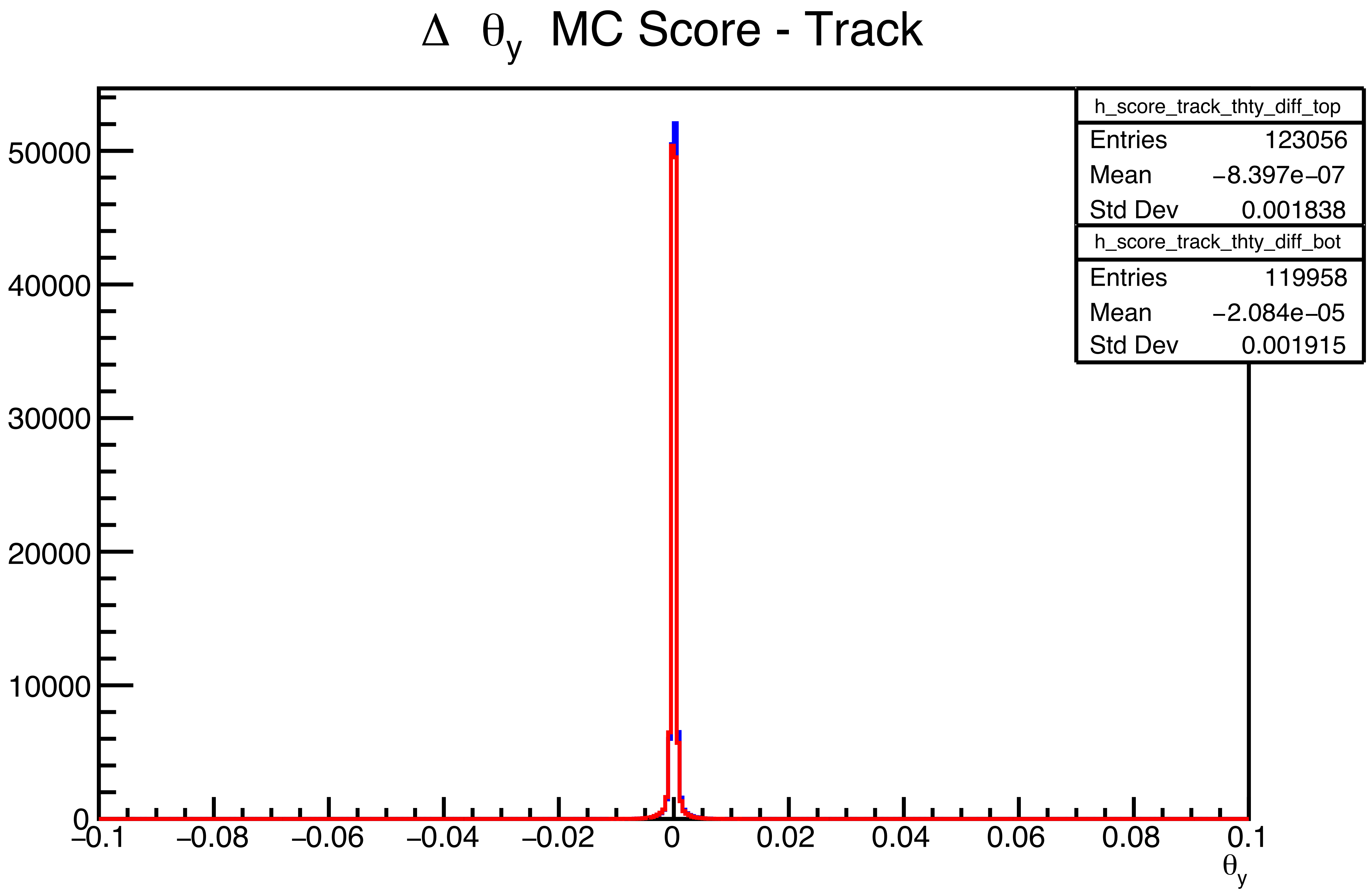




Const Fringe: Compare Track and Scoring Plane

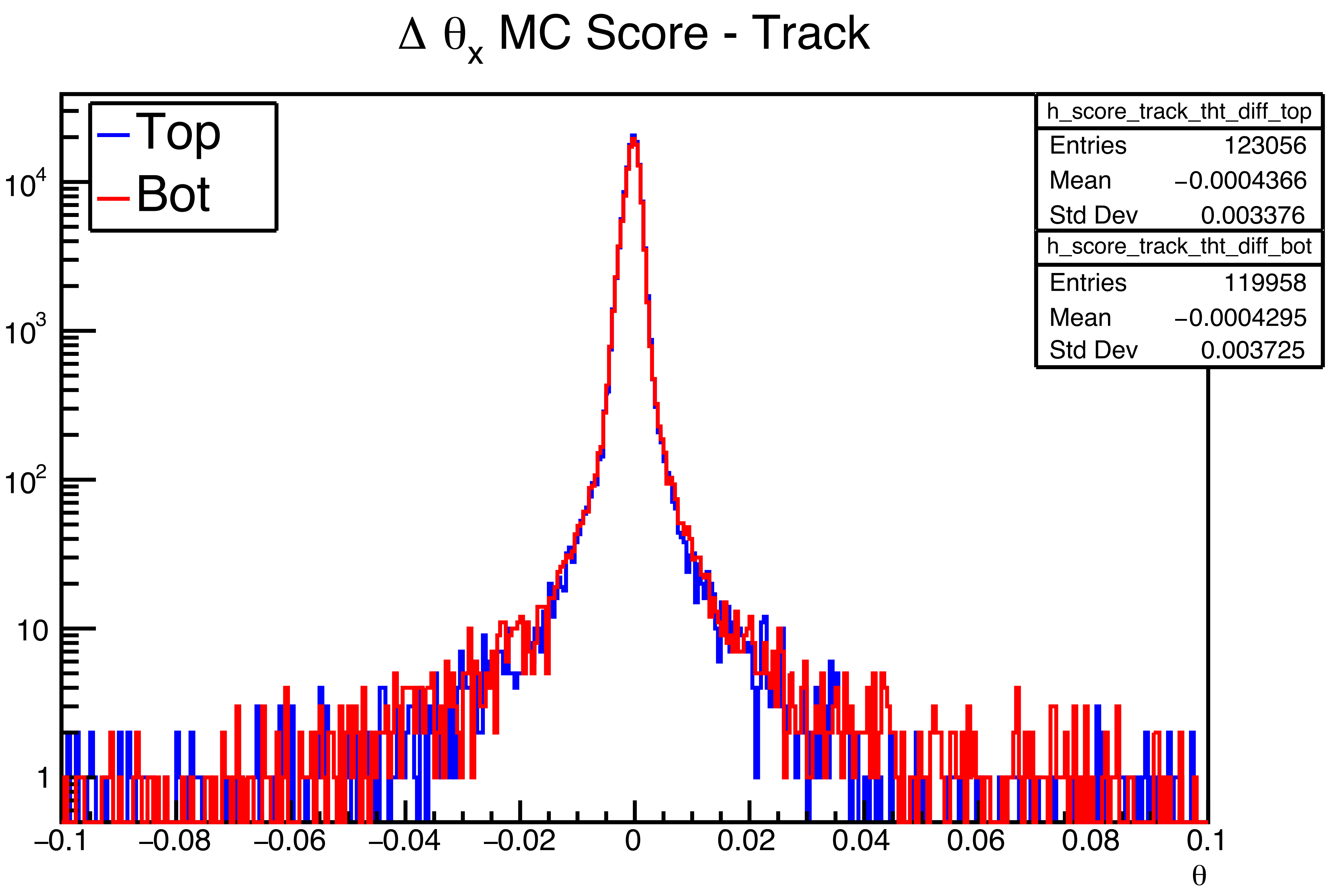
Angles also still match well.

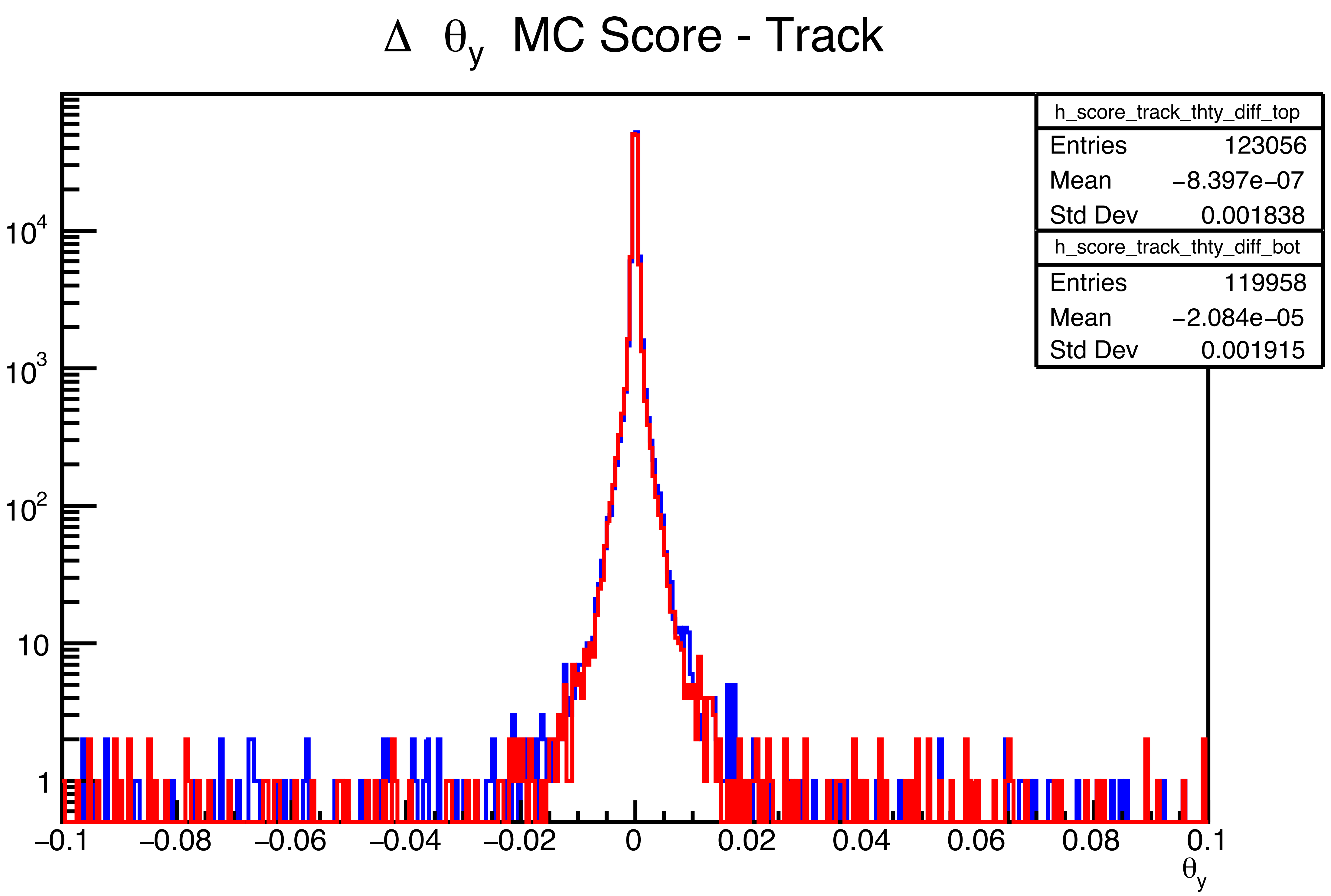




Const Fringe: Compare Track and Scoring Plane

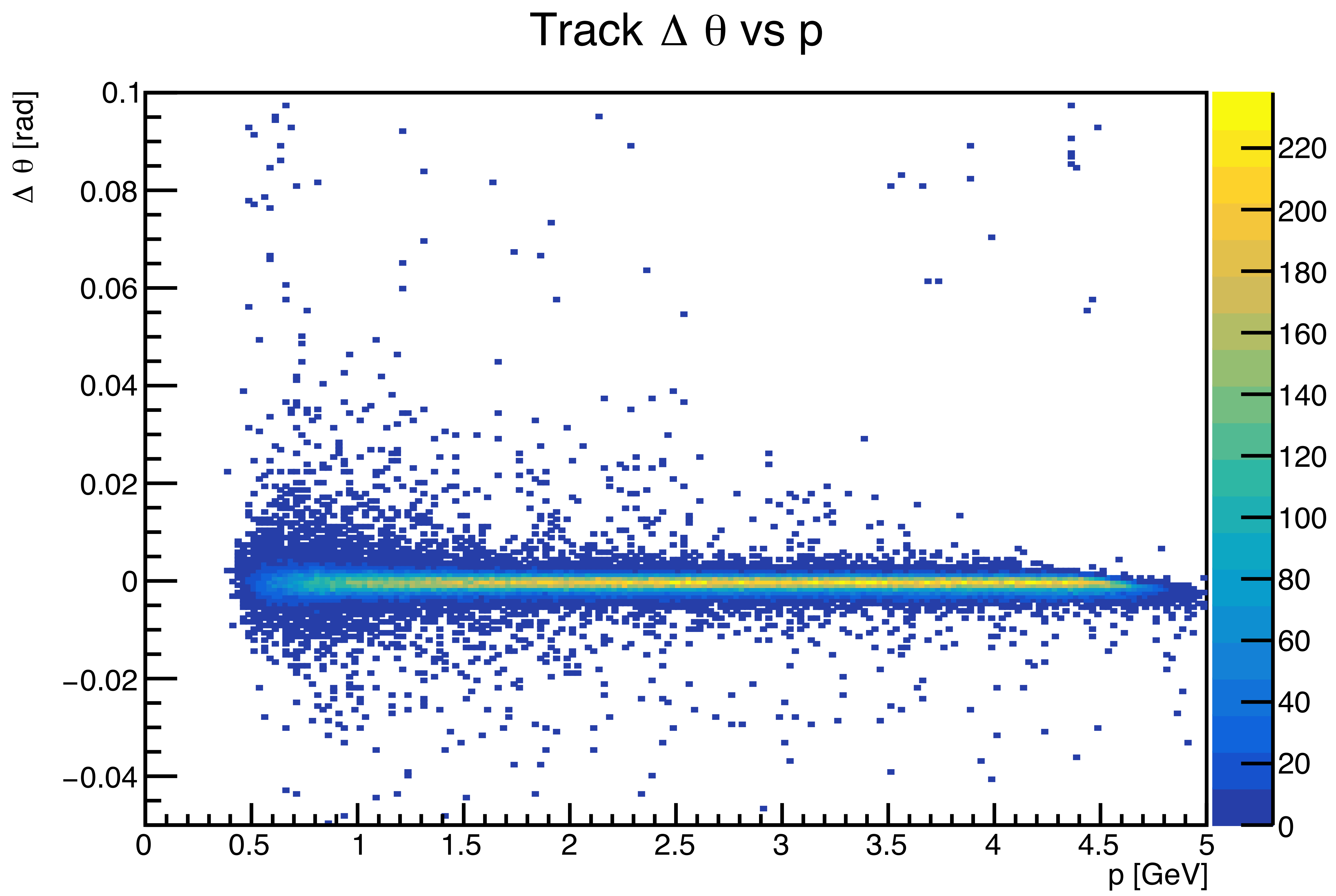
Angles also still match well.

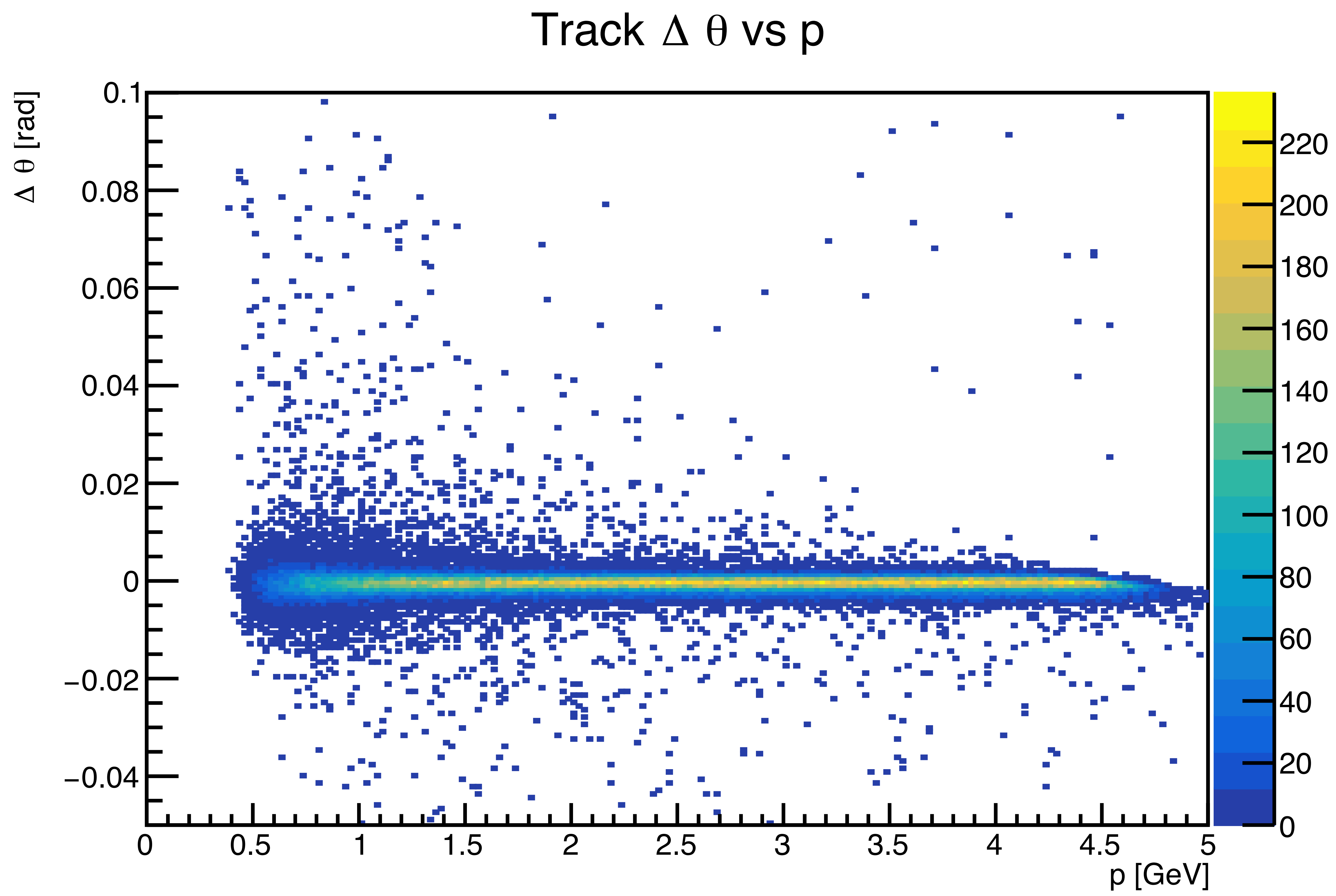




Const Fringe: Compare Track and Scoring Plane

Angles also still match well.



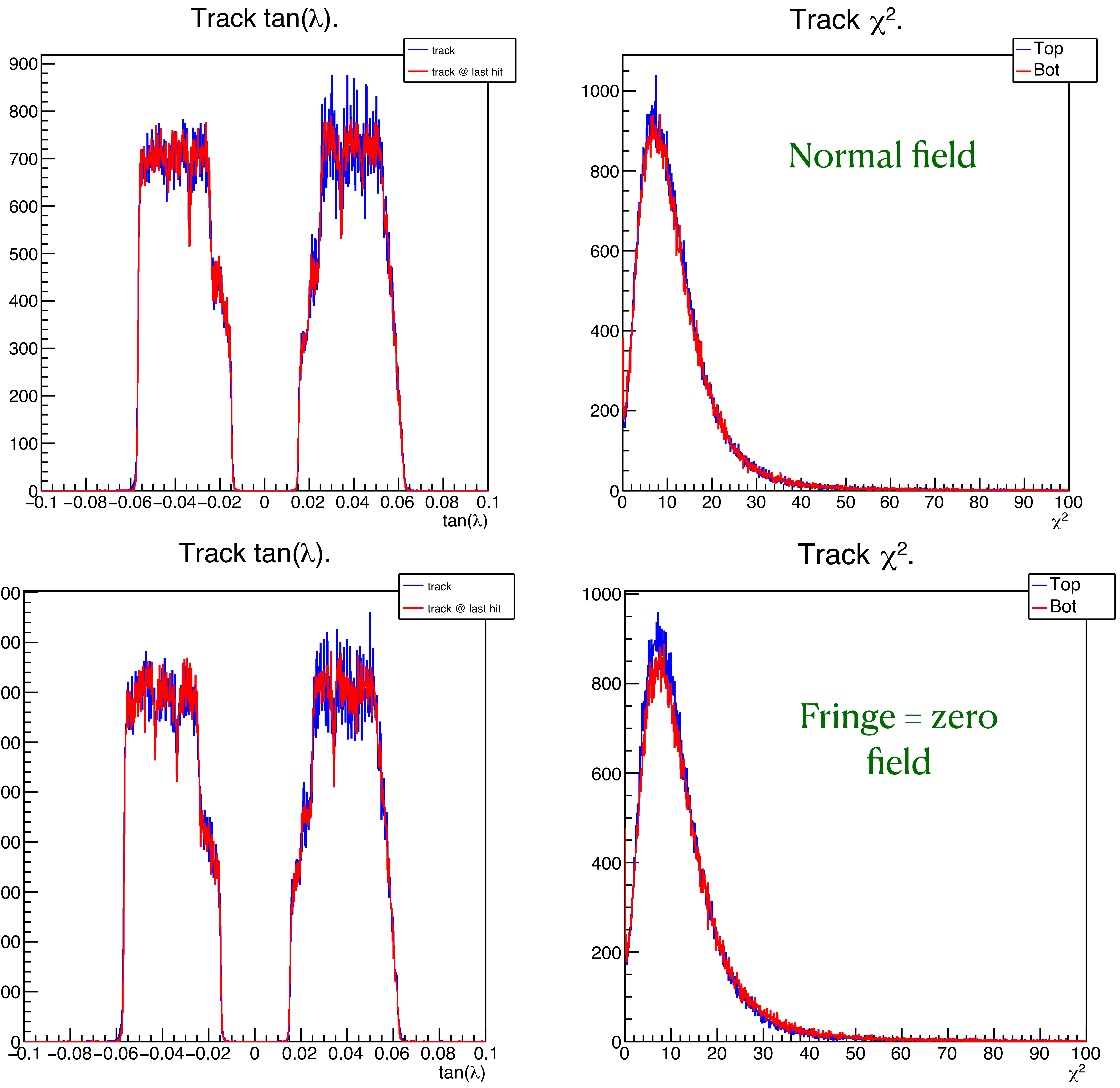


Conclusions

- It seems that KF tracking is working well for the MC geometry with the standard field, up to the last tracking plane. Variable x, px and θx are well predicted.
    - Turning a "track state" into a momentum 3-vector is not the trivial step it should be. This would be much easier if LCIO was able to store and write to file the 3-vector in a logical manner.
- Extrapolation to the ECal does not work well for the tracks.
    - This gets *worse* when the field is set to zero. It may be that the interpolation for the field map is using a different algorithm than that used by GEANT4, which could cause big differences when the field changes rapidly.
- The next, time consuming and technically challenging step would be to compare GEANT4 stepping through the field map against hps-java stepping through the same field map.

Extra Slides

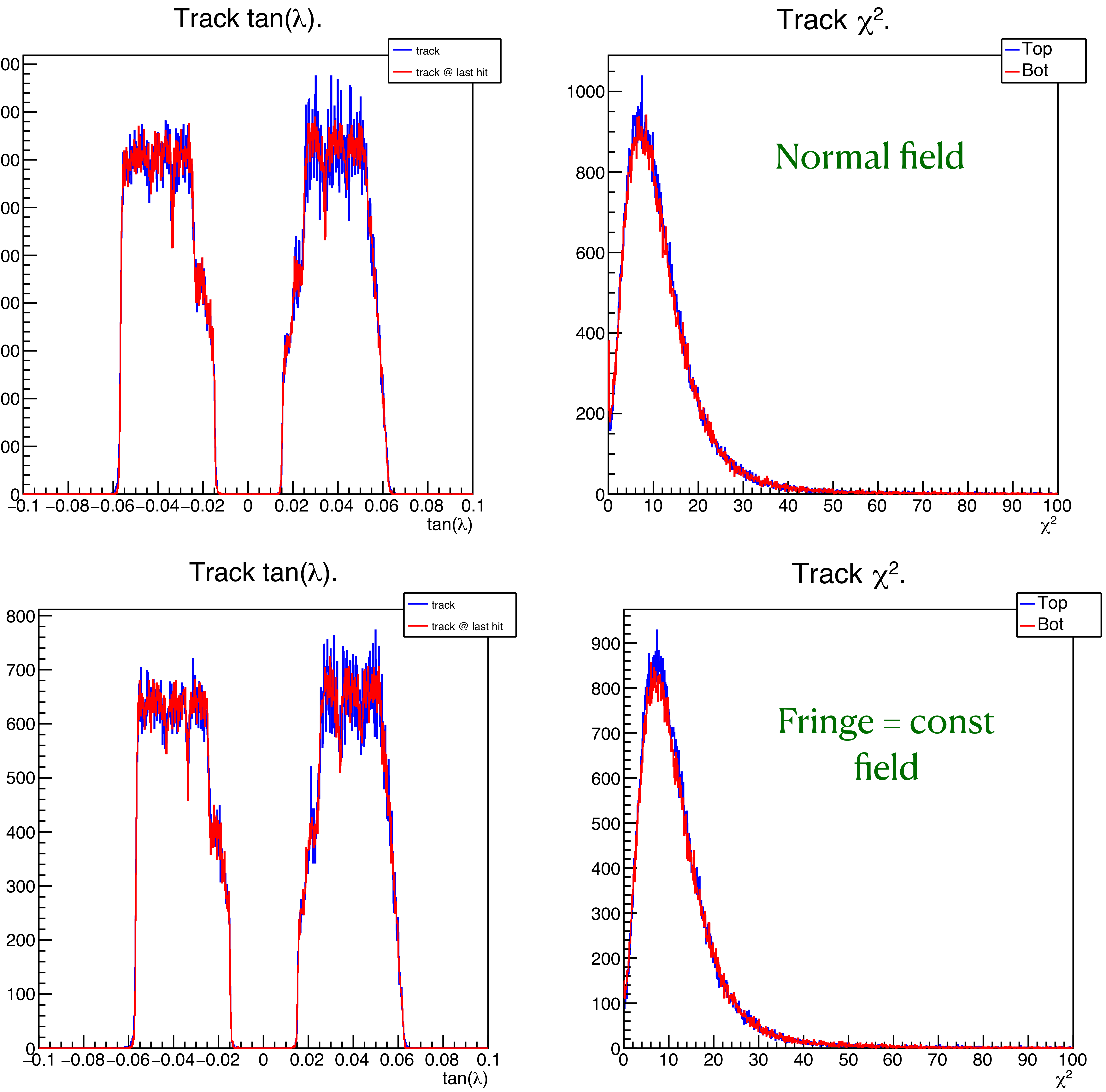
Track Quality Comparison



Normal field

Fringe = zero field

Track Quality Comparison


Track tan(λ).
track
track @ last hit
tan(λ)
Track χ².
Top
Bot
χ²
Normal field
Track tan(λ).
track
track @ last hit
tan(λ)
Track χ².
Top
Bot
χ²
Fringe = const
field