Kalman Filter Pattern Recognition and Fitting Status Update

Robert Johnson

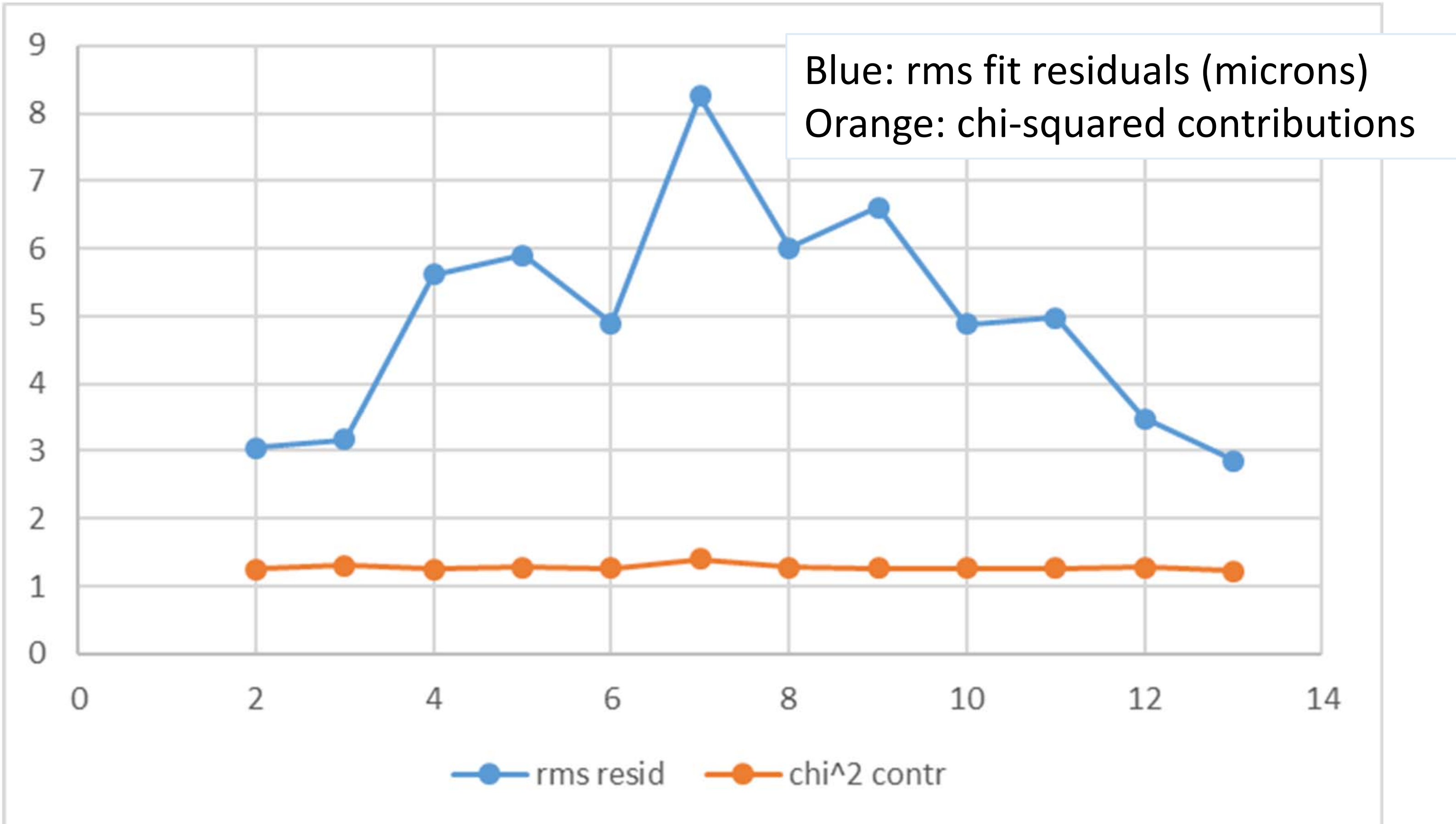
U.C. Santa Cruz

April 7, 2020

Progress since the last collaboration Mtg.

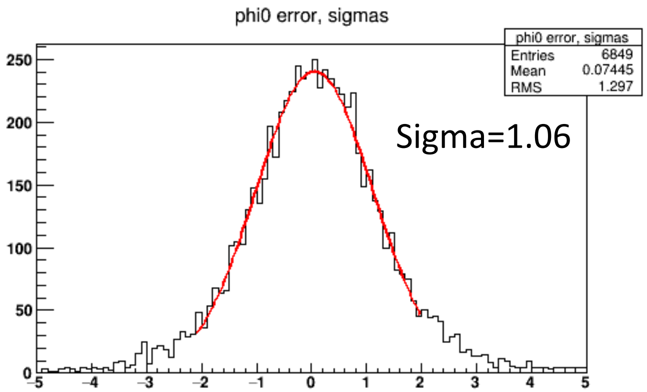
- Problems with coordinate transformations were understood, so the fit works properly in HPS java.
  - The Kalman fit quality is reasonable for single-particle events. The chi-squared mean, unlike in the toy MC case, is 20% to 30% high, but that is probably due to
    - Non-Gaussian scattering contributions
    - Tails in the hit error distributions, as seen in MC truth.
- The pattern recognition code was completed and tested on 2016 HPS MC (tri-trig-beam) as well as the toy MC.
  - Based on testing with realistic events, this involved fairly major re-writes since the November 2019 status.
  - Timing cuts were added to the pattern recognition.
- Integration into the hps-java framework and event reconstruction was (nearly) completed, thanks to a lot of contributions from PF.

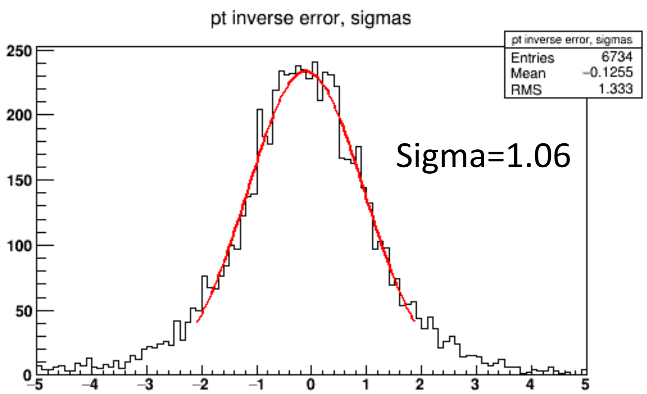
2 GeV Single MC Electrons, 2016 detector

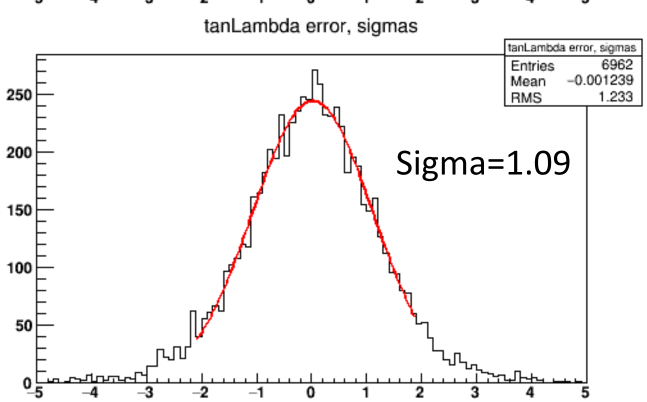


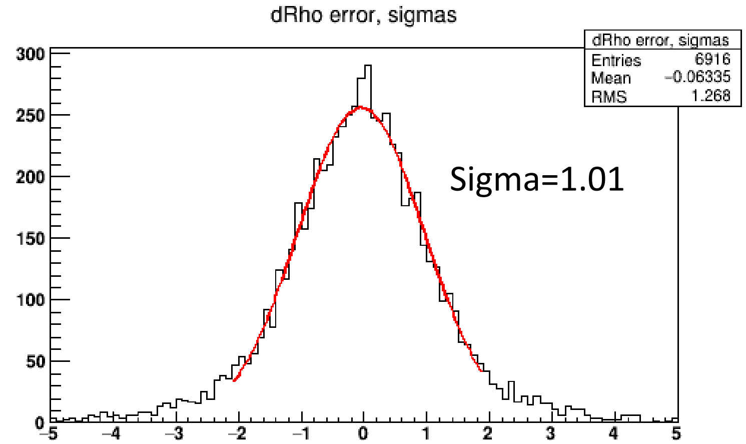
In all cases the chi-squared contributions are about the same from all 12 layers, as they should be, but are 20% to 30% higher than unity.

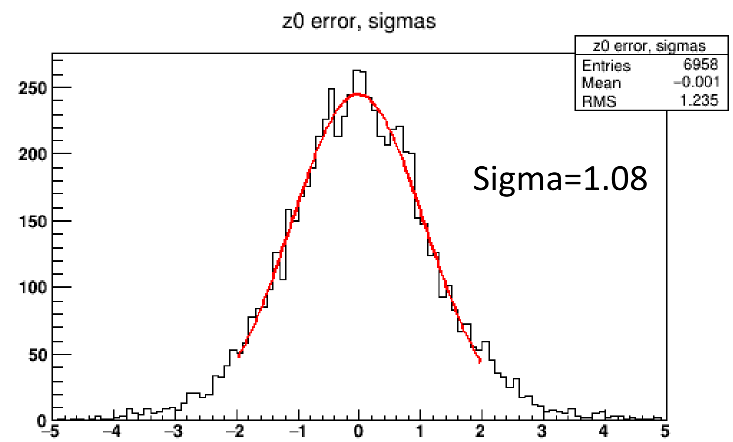
Helix Parameter Pull Distributions





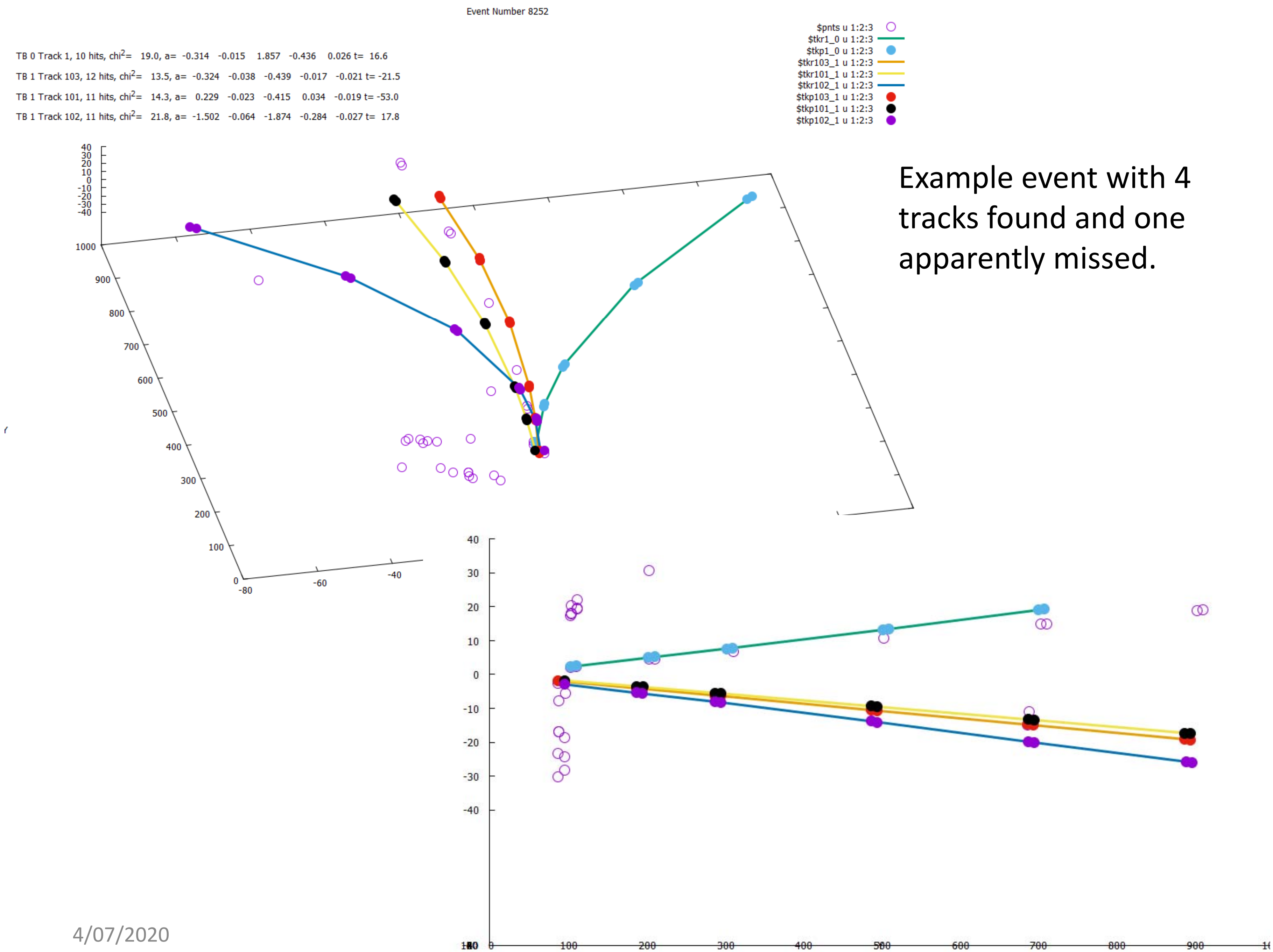




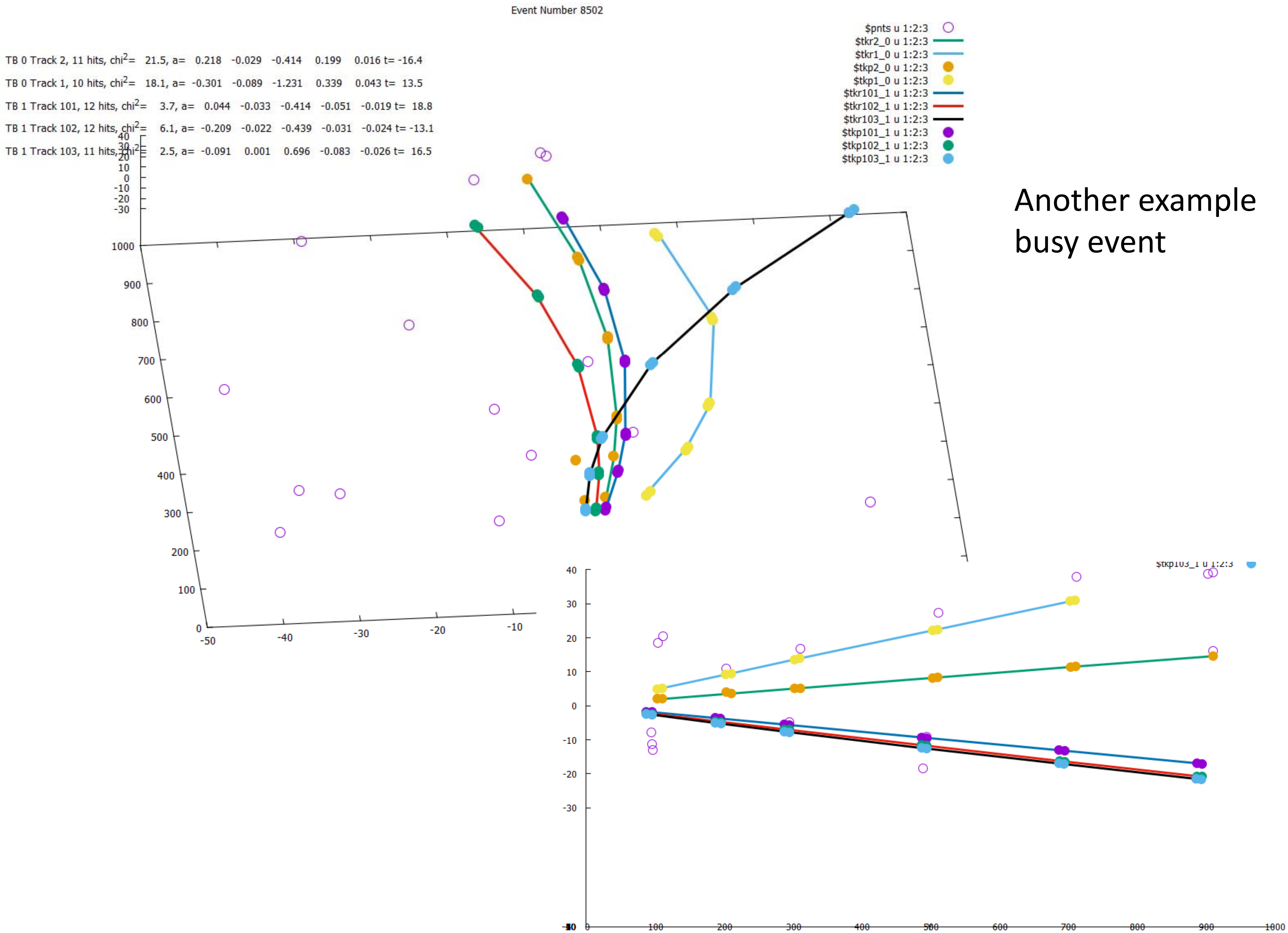


Pattern Recognition Strategy (simplified)

- Seed the Kalman Filter with a linear 5-parameter fit.
  - Loop over seed strategies, each a set of 3 stereo and 2 axial layers
    - Zero d.o.f. fit when only 5 hits are input, as is done here.
    - 3-D hits are never formed or used!
  - Do the linear fit on all combinations of hits in the 5 layers (this quickly gets out of hand if there is a huge number of garbage hits)
  - Reject seeds that don't look interesting (e.g. don't extrapolate near to the vertex). Some speedup can probably be had by improving this.
- Sort the seeds and input each into the Kalman Filter to follow the track and pick up more hits.
  - The resulting track candidates are sorted, and poorer candidates that share too many hits with better ones are discarded.
  - A shared hit is allowed only if it fits very well to both tracks.
- Remaining candidates are re-filtered and smoothed, including some bad hit removal.
- Extrapolation to the origin (or target).

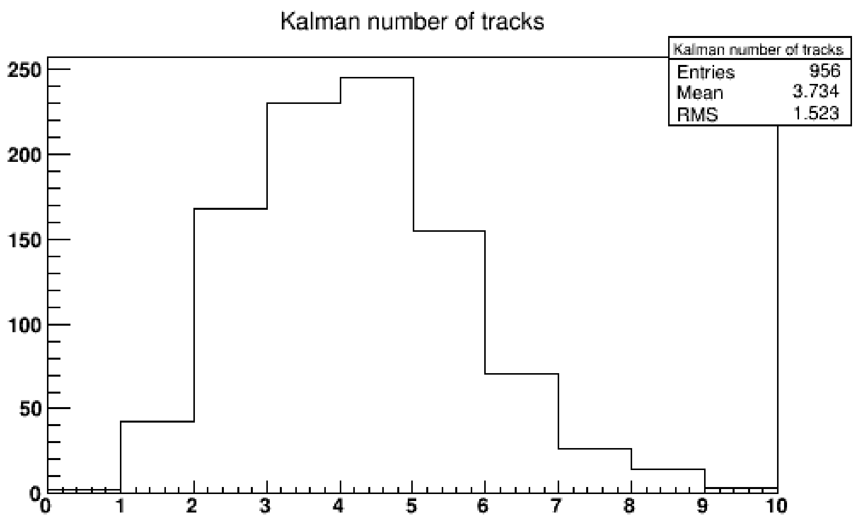


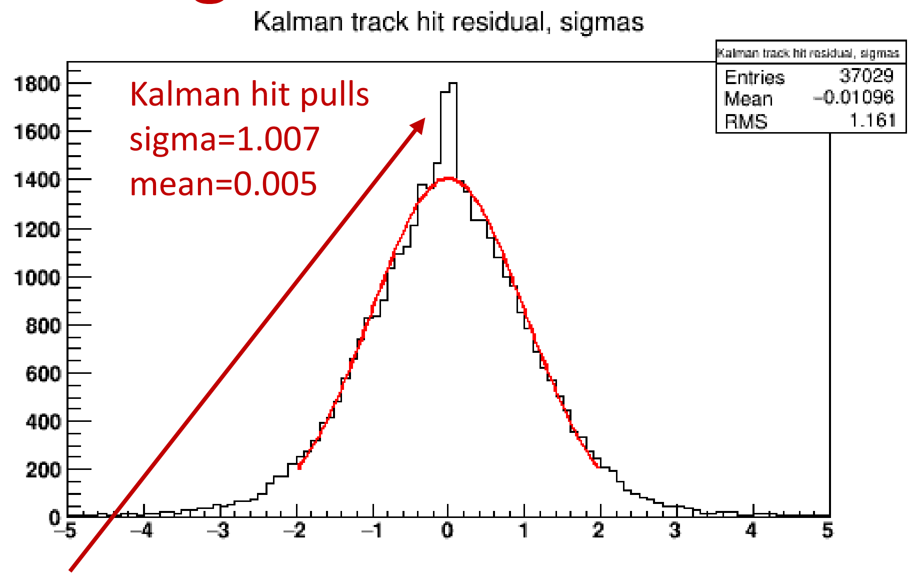
Example event with 4 tracks found and one apparently missed.



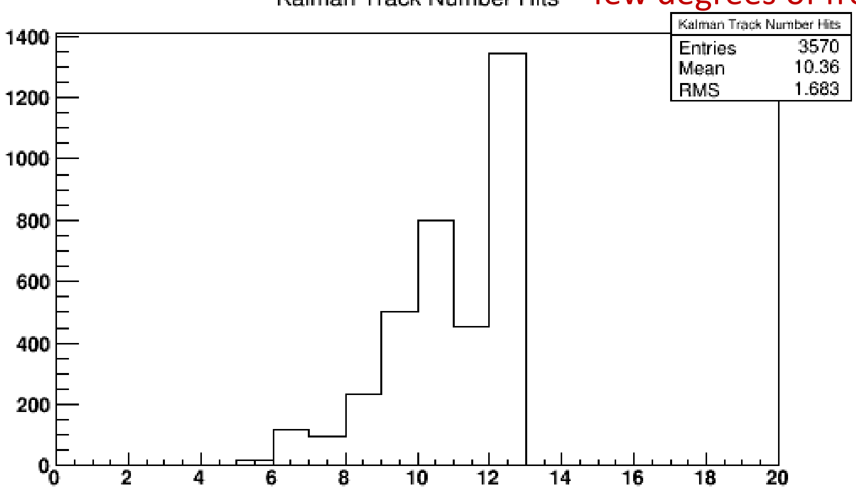
Another example busy event

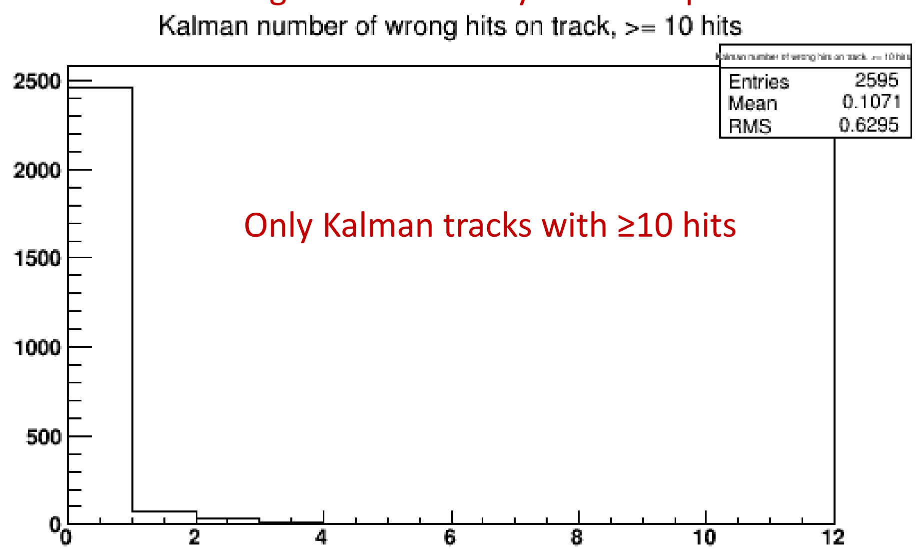
Some Results from 956 Tri-Trig-Beam Events





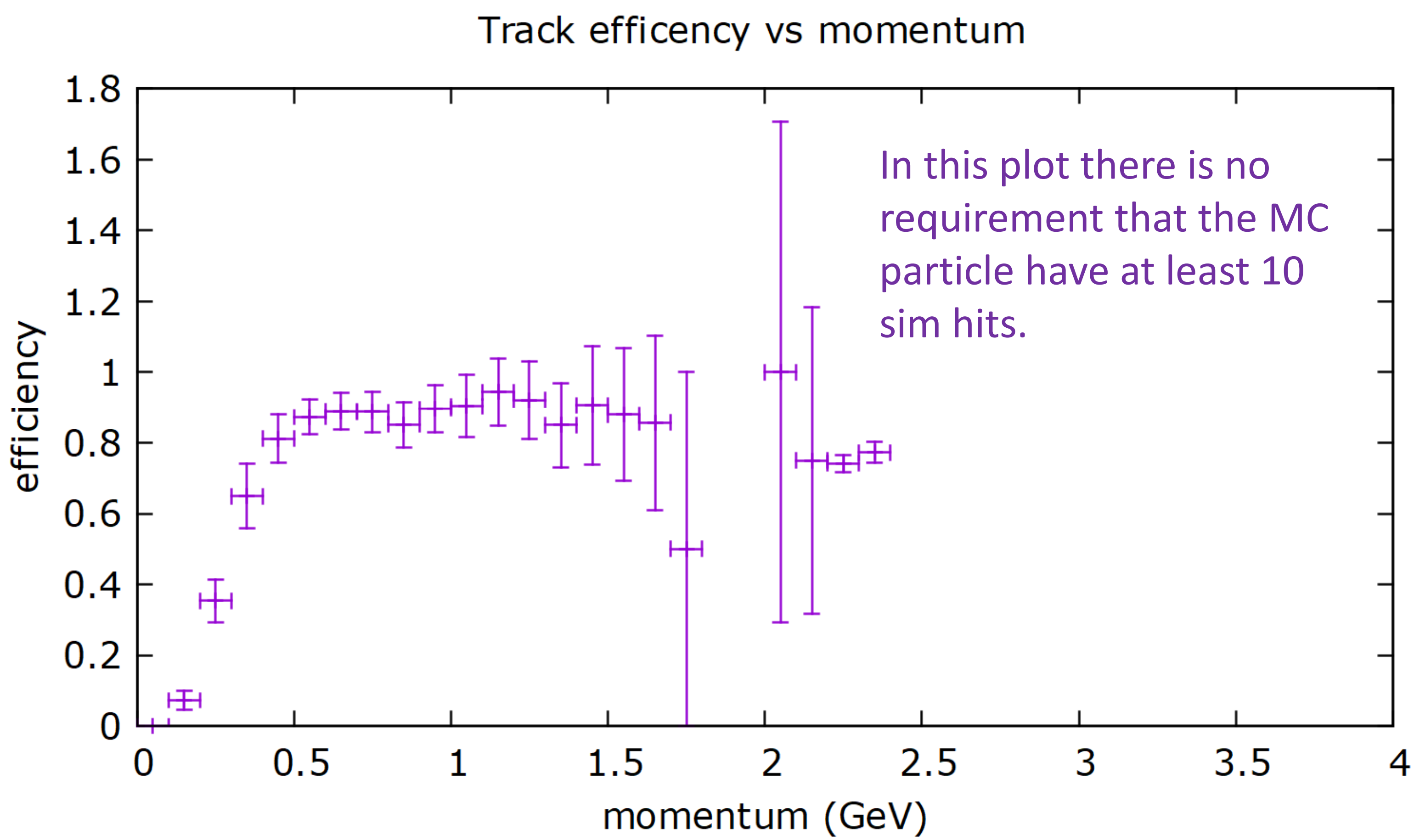
The too narrow peak at the center is from tracks with few hits and very few degrees of freedom. Such tracks give anomalously low chi-squared.





- I tried to analyze the tracking efficiency by defining success to be when at least 6 hits match back to a single MC particle and not more than 2 hits belong to a different MC particle.

- In that case, the overall efficiency above 0.7 GeV for MC particles associated with at least 10 sim hits is 93%.

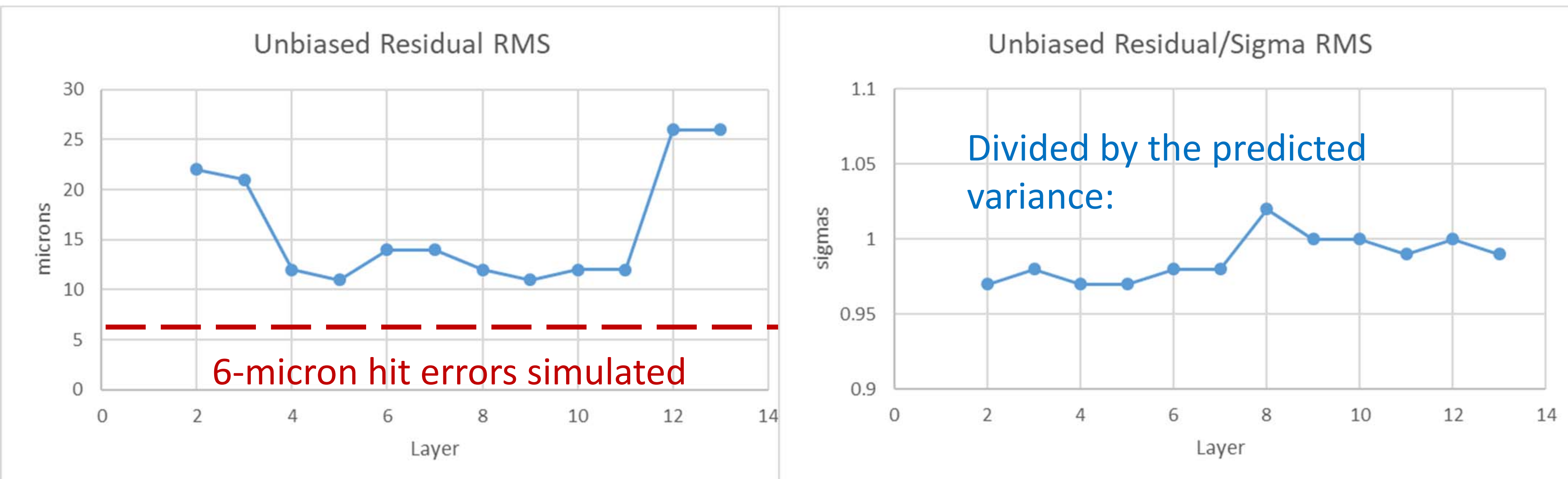


Work in Progress

- Generalize the helix extrapolation in the non-uniform field (e.g. to the target and to the ECAL).
  - 4th order Runge-Kutta integration works well for the helix, but not for its covariance.
  - Stepping the helix through a series of ~25 mm steps, each time making a pivot transform using the local field, should work well for the covariance.
    - It also gives the helix extrapolation with a result that I cannot distinguish to >6 decimal places from the Runge-Kutta result.
    - Scattering contributions to the covariance get added in whenever a silicon plane is crossed (e.g. if extrapolating from the second layer to the target for a track that has no hit on the first layer).
  - This is implemented, but I'm not yet certain that the covariance extrapolation is correct. I'm trying to work out a solid way to check the result numerically (by MC).

Work in Progress

- Unbiased Residuals
  - The Kalman-Filter framework provides a convenient way to calculated an unbiased residual at each layer, using the so-called Inverse Filter, which removes a given hit from the fit while retaining all of the input from the other hits.
  - This is implemented and works nearly perfectly for 12-hit tracks in the toy Monte Carlo, see below.
  - Still trying to understand some negative predicted variances that arise in more complicated environments.



Why use the Kalman P.R. for 2019 data?

- It appears to be faster than the existing tracking code, and with some efforts it could probably be made faster yet.
- It never uses the 3-D hit artifice, which allows it to handle naturally double-layers with a missing axial or stereo hit.
- It uses the full HPS field map.
  - This seems not to be an important point, however, based on my testing, comparing fits with uniform vs non-uniform field.
- It's performance on MC events already seems to be at least as good (see PF's following talk).