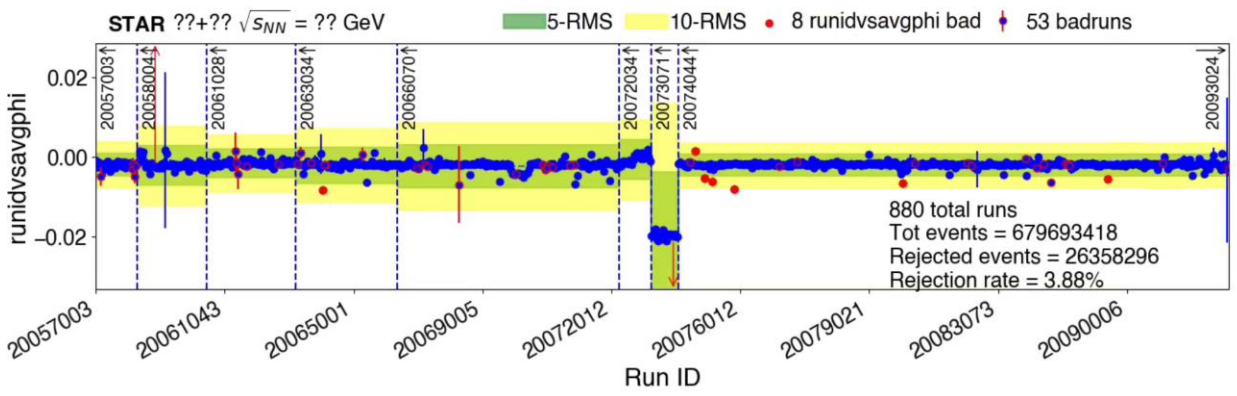
Why does run-by-run v2 and v3 give different bad run list.

Recap what run-by-run QA does

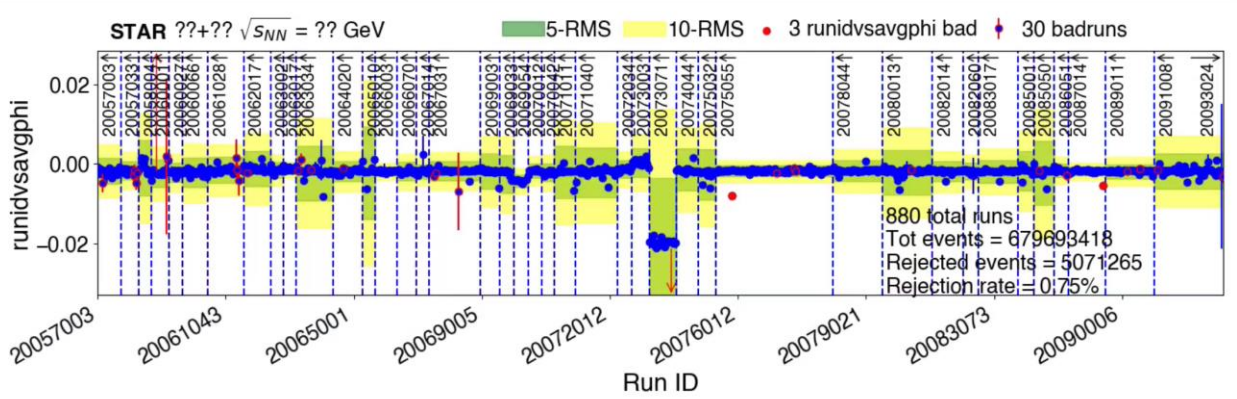
1. Breaks runs into segments.
2. Calculate mean and standard deviation (Std) of each segment.
3. Reject runs that falls outside of mean +/- 5Std.
4. Repeat step 1 – 3 until no more runs are rejected.

Parameter used by segmentation code: gamma. Smaller the value, the more segment there is.

Gamma = 5



Gamma = 0.5



Run repeatedly for different gamma values. Use gamma value that returns the most numbers of bad runs.

Change #1: Different gamma range

V2

- Gamma = [0.5, 1, 2, 3, 5, 9]
- They are supposed to be the same.
- Different due to typos.

v3

- Gamma=[0.5, 1., 2., 5., 9.]

Change #2: Different rejection criteria

V2

- Reject if (data – mean)^2 > 5*5*(uncertainty square + segment variance)

v3

- Reject if |data – mean| > 5*segment variance + uncertainty

Change #3: v3 don’t merge segment manually anymore

v2

- After segmentation, consecutive segments whose means + std ranges overlaps are manually merged into one.

v3

- This step is not performed

Change #4: v3 uses parallel rejection

- Parallel rejection:
  - Calculate the mean and variance of all observables (e.g. dedx, avgphi, mult) simultaneously.
  - Reject runs if any observables lies outside 5 Std.
- Sequential rejection:
  - Calculate the mean and variance of the first observable.
  - Reject runs from that observable.
  - Calculate the mean and variance of the second observable with rejected runs excluded.
  - Continue sequentially for all observables.
  - Depend on the order of observables being loaded.

Change #4: v3 uses parallel rejection

v2

- Uses parallel rejection only on the first iteration.
- Uses sequential rejection on second to last iterations.

v3

- Uses parallel rejection on all iterations.

Changes #5: v3 doesn't renormalize data on every iteration

v2

- Data being fed to segmentation routine need to be recentered and rescaled.
  - (x – global mean(x))/global std(x)
- Normalization is done on every iteration, excluding runs being rejected from previous iterations.

v3

- Normalization is only done on the first iteration, then I reuse the same global mean/std for all iterations.

Changes #6: Different weight on runs

v2

- Weighted by 1/err^2

v3

- Weighted by counts

Changes #7: Different ways to handle invalid runs

v2

- In some runs, I get 0 in tofmult, but reasonable values in avgphi for example.
- Calculation of mean and Std is done without that run for tofmult, but that run is included in the calculation of avgphi.

v3

- If I get 0 in any observable for a run, that run is tossed out, not included in the calculation of mean and std of any observables.

Changes #8: v2 has lower floating-point accuracy

v2

- I read ROOT files with a ROOT script that print all observables with std::cout, pipe the standard ouput to txt file from which the run-by-run v2 will read.
- Std::cout only has 6 significant figures in its default settings.

v3

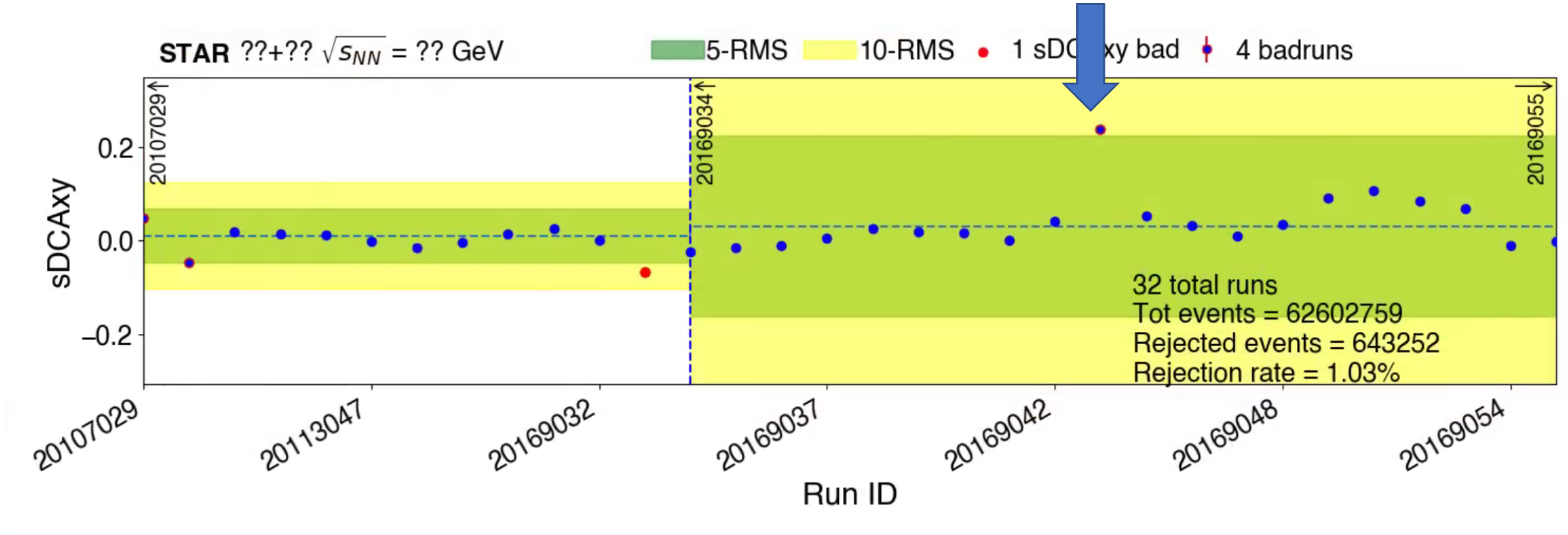
- ROOT files are being read natively with full 17 digits accuracy.
- Yes, it does affect the segmentation sometimes. I’m not splitting hair.

Emulate the behavior of v2 with legacy mode

- Python3 QA.py -i <QA hist.root> -v <QA variable list.txt> -o <bad run list.txt> ***--legacy***
- Mahammad is testing if legacy emulate the behavior of v2 accurately.
- If so, I have found out all the differences between the two versions.

Curious behavior

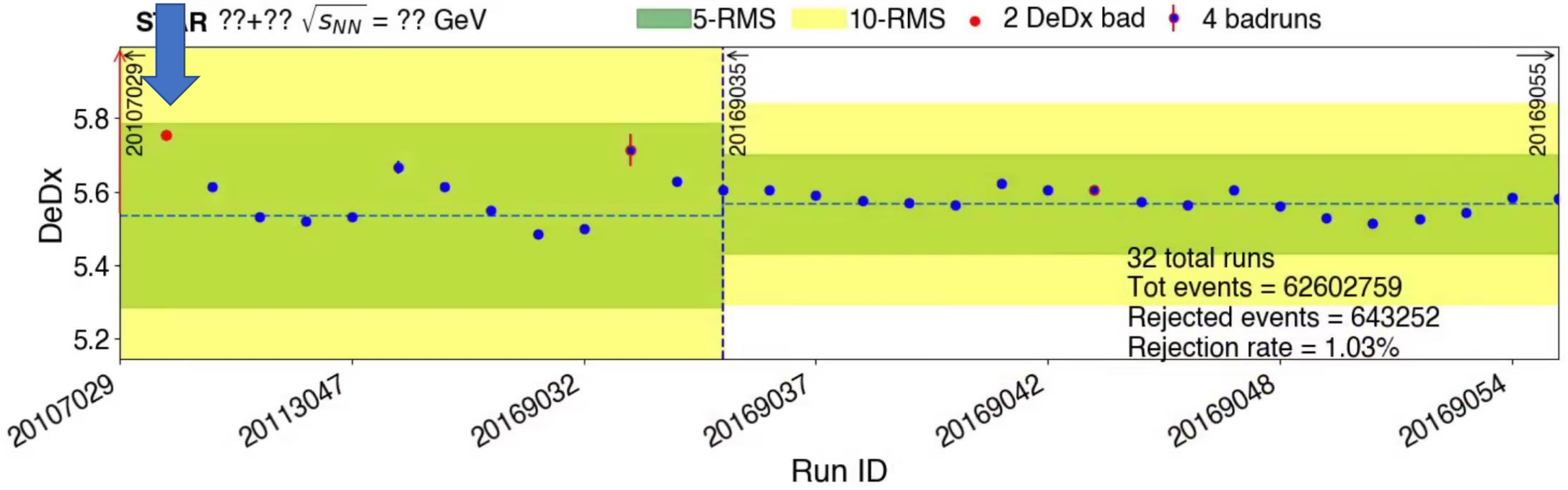
sDCAxy not recognized as reason for rejection because it was rejected in the first iteration where the segment boundary is somewhere else sDCAxy width is larger.



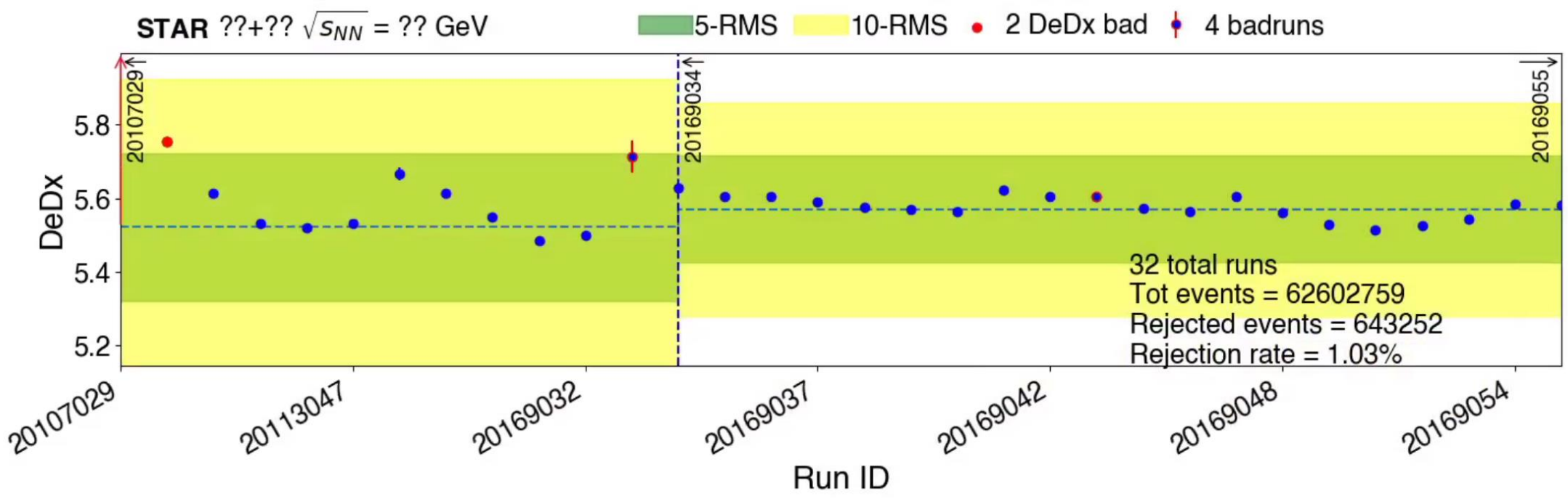
Rejected despite point within 5 Std

A bug in the plotting routine

N.O. runs rejected = 0 at the last step, I plotted segments, mean and std from that last step. Should've used second to last where rejection happens.



After bug fix



New development: global rejection

- Added a ‘-g’ flag for global rejection
- Disabled by default.