INFN PISA
Istituto Nazionale di Fisica Nucleare
Sezione di Pisa

Extreme Energy Events
Science inside Schools

Helium-based gas mixtures: status of the analysis review

E.Bossini
(with the irreplaceable help M. Abbrescia, S. Boi, F. Noferini,
M.P. Panetta & C. Ripoli)

Analysis summary

GOAL:

- review the results obtained with the PISA-01 and REND-01 telescopes, fluxed with He+R1234ze.
- Investigate (& solve) eventual critical aspects.
- Proceed to publication

DATA: acquired by the two stations in the period September 2021 – June 2022. To be compared with std (R134a+SF6) gas mixture.

ANALYSIS OUTPUT:

- Efficiency
- Cluster multiplicity
- Streamer fraction

} Test chamber with He mixture, triggering chambers with std mixture
Dedicated reco.

- Angular distribution, speed, TOF of the reconstructed tracks
- Stability in time (rate, %reco,...)

} All chambers with He mixtures
data from Standard reconstruction.

- Time resolution without TW correction?

Analysis steps

Extreme Energy Events
Science inside Schools

☑ Review of efficiency code(s) :

- ➢ CNAF official reco code (from binary to “debug tree”)
- ➢ Dedicated efficiency code by S.Boi.

2 main critical aspects :

- Efficiency, streamer fraction and cluster multiplicity are not obtained in a consistent way (different codes and/or different cuts). -> Difference in efficiency between CNAF eff. code and dedicated code (tuned for streamer % computation) ~10%
- Streamer are not <<1%, but of the order of 10% (50/50 mixture)

☑ Debug/improve actual code. Main changes:

- ➢ New clusterization algorithm
- ➢ New calibration procedure (simultaneous time/space calibration)
- ➢ New selection cuts

At present the code can extrapolate streamer and efficiency simultaneously. Difference in efficiency between CNAF eff. code and dedicated code below 2%

❑ Further optimization/automatization of the code, target discrepancy below 1% (Autom./optim. to be refined)

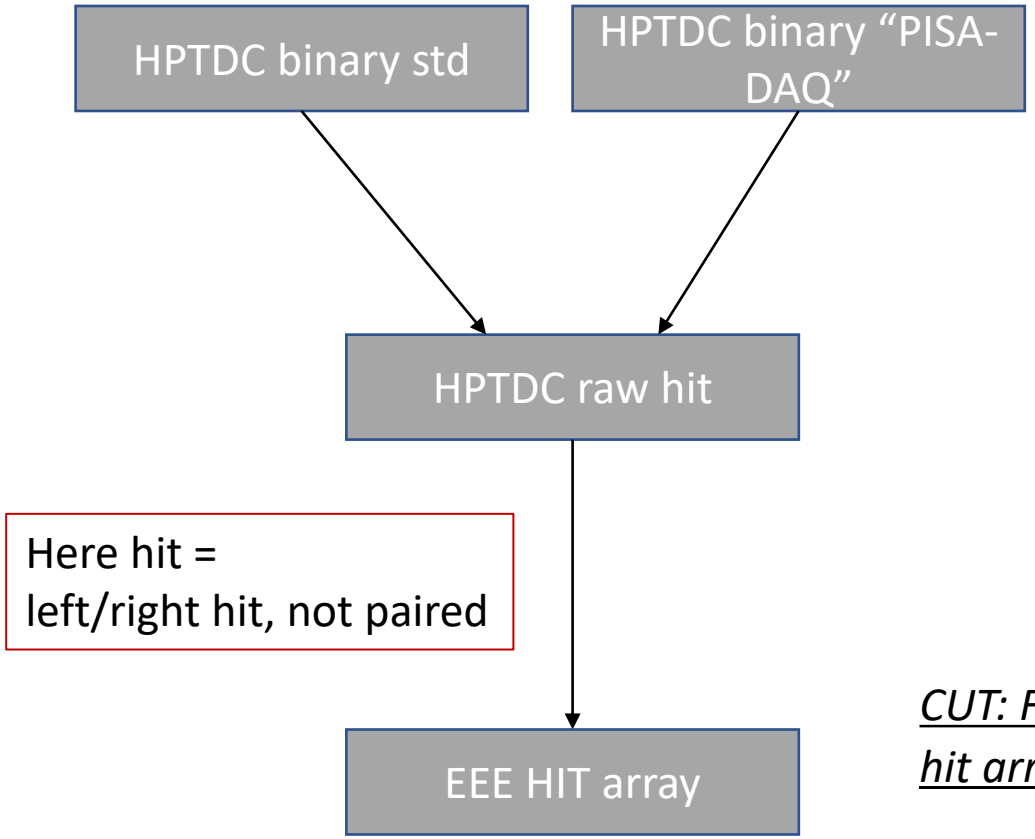
❑ Validation on a larger set of runs (at present I’m using a PISA run with 50/50 mixture @ eff. plateau, worst condition in terms of reconstruction).

❑ Recompute efficiency for the selected efficiency scans (2 telescopes, ~4 mixtures)

❑ Re-reco of PISA data after fix of DST producer -> New plots of parameter distributions (beta, Theta, ToF,...) (ONGOING)

Code review: CNAF

Several codes used for the efficiency analysis, I decided to base the analysis review on the code developed by S.Boi. It takes as input the DST files generated by the CNAF reconstruction code, using a low level TTRee.



1. *multi-hit on the same channel possible*
2. *Trailing edges without prior leading edge are discarded*
3. *leading edges without a trailing edge are registered with TOT=0*

CUT: hit is discarded is the time of arrival is outside the limits taken by the configuration file «if (timeHit >= fCalib->GetTbLowLimitRight() && timeHit < fCalib->GetTbHighLimitRight())»

CUT: For each channel a maximum of 6 hit are passed to the hit array

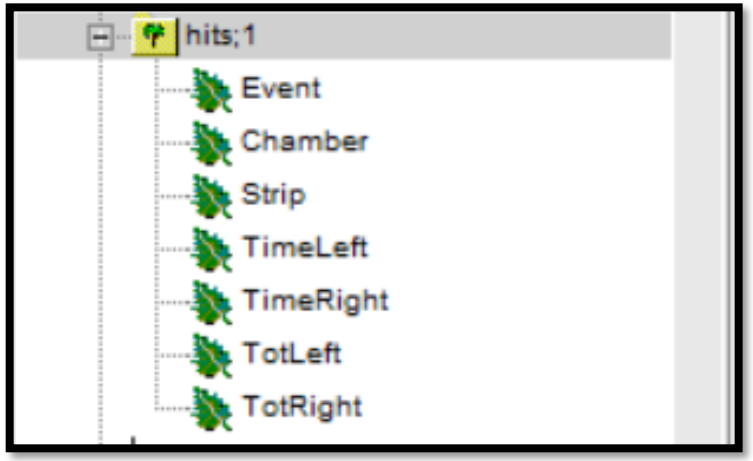
Code review: CNAF

EEE HIT array → EEE Debug tree

Note: If a strip has no hit on one side, the time on that channel is set to 0 and the TOT=-1

CUT: ONLY the first hit per channel is transmitted, other are discarded (more on this later)

DST file content



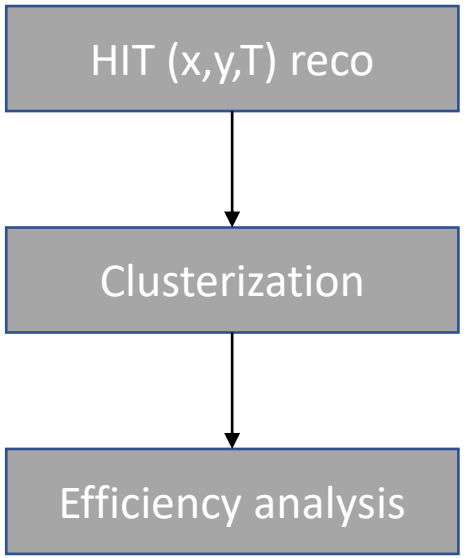
- Data from "PISA" DAQ are reconstructed with hardcoded values:
  - of the geometry (in particular distance between chambers, wrongly set)
  - of the architecture (NINO version, correctly set)

This does not affect the efficiency but should be fixed for the future data

DST distribution to be recomputed with correct values (ONGOING)

Code review: efficiency extraction

The workflow of Stefano was used in several plots already presented but was not the only one used.



It turned out that:

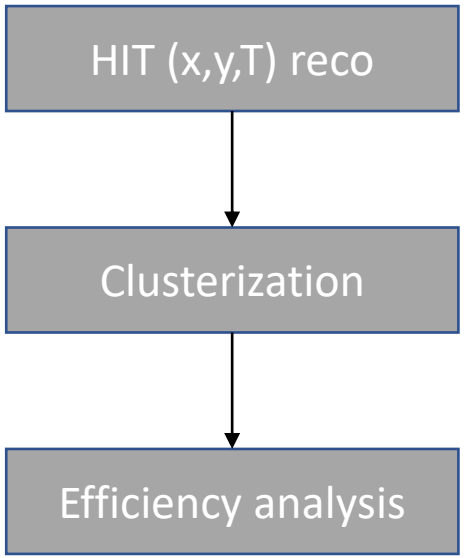
- Efficiency, streamer fraction and cluster multiplicity are not obtained in a consistent way (different codes and/or different cuts).
- Difference in efficiency between CNAF eff. code and dedicated code (tuned for streamer % computation) ~10%
- Streamer are not <<1%, but of the order of 10% (50/50 mixture)

Keeping the same code infrastructure (well done and with an event display!), the following section were changed:

- New calibration procedure (simultaneous time/space calibration)
- New clusterization algorithm
- Upgraded selection cuts
- Minor fixes (not discussed here) and more control histograms

Code review: efficiency extraction

The workflow of Stefano was used in several plots already presented but was not the only one used.



TEST File:
Stations: PISA-01
Gas: R1234ze+He (50-50%)
Test chamber : Bottom
HV: ~19KV

It turned out that:

- Efficiency, streamer fraction and cluster multiplicity are not obtained in a consistent way (different codes and/or different cuts).
- Difference in efficiency between CNAF eff. code and dedicated code (tuned for streamer % computation) ~10%
- Streamer are not <<1%, but of the order of 10% (50/50 mixture)

Keeping the same code infrastructure (well done and with an event display!), the following section were changed:

- New calibration procedure (simultaneous time/space calibration)
- New clusterization algorithm
- New selection cuts
- Minor fixes (not discussed here) and more control histograms

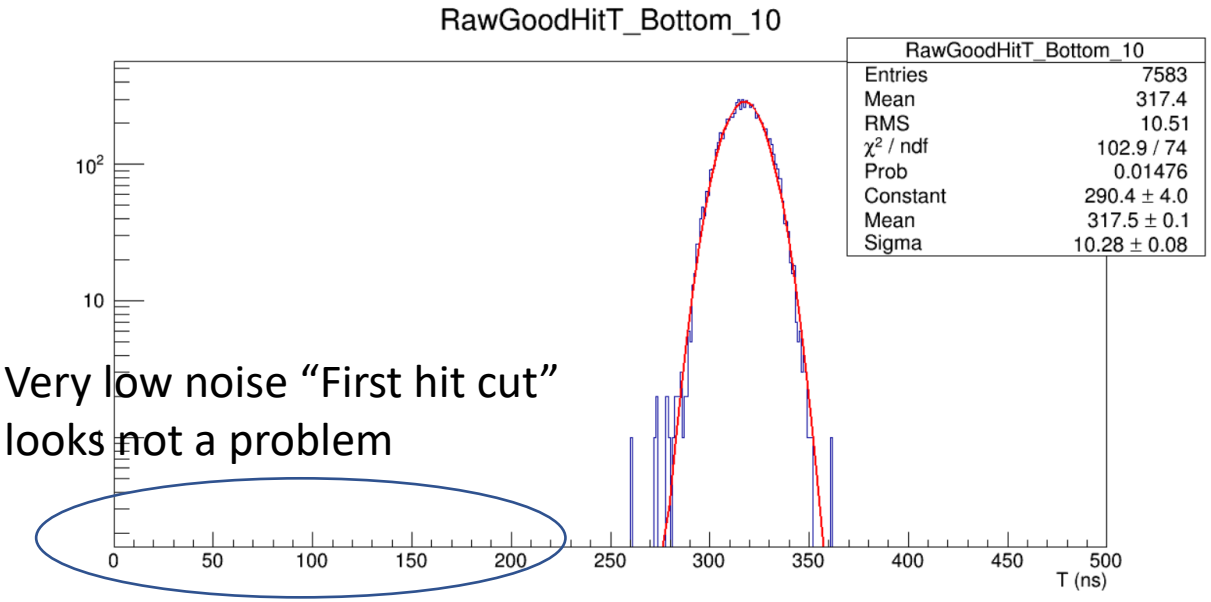
Calibration

Old calibration was performing only a spatial calibration on the x (longitudinal) coordinate.
Calibration was applied on the reconstructed position of the hit, after pairing.

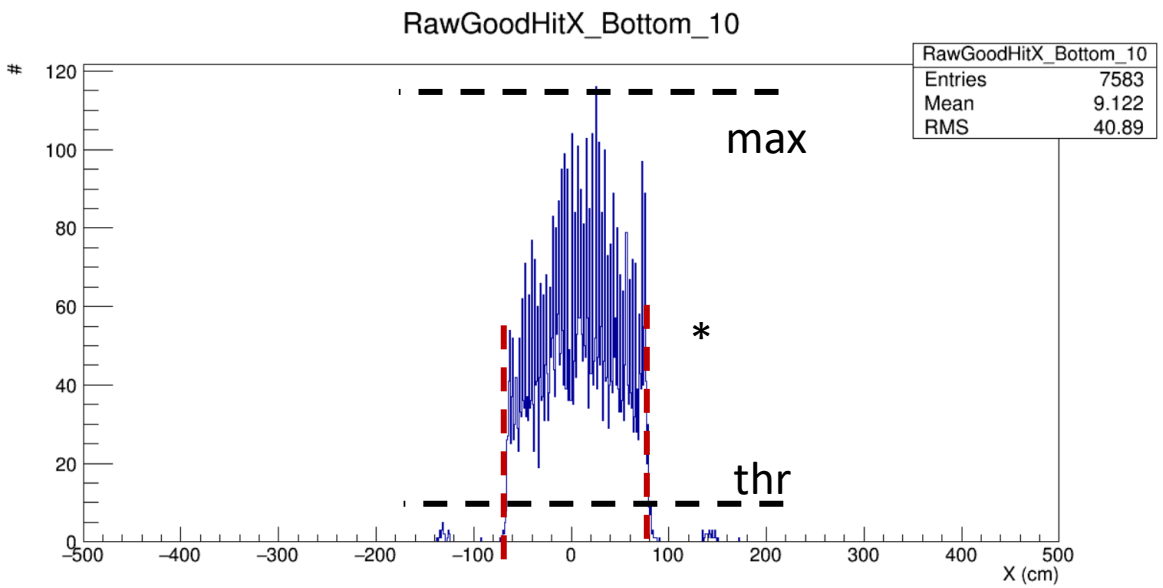
New calibration performs a simultaneous calibration of space and time.
It also calibrate the average middle time to be centered w.r.t. the outer chambers.

1. For each strip compute
   - average x coordinate
   - average hit T (Tl+Tr/2) distribution

2. For each chamber:
   - Average hit T distribution



Very low noise “First hit cut”
looks not a problem



Simple average affected by large calib. error for non uniform hit distributions

Calibration

3. Space/ correction are independently applied

```
if(!external_calib)
{
    std::cout<<"Computing corrections..."<< std::endl;
    double delta_time,strip_offset, Avg_T_adjust[3],x_offset;

    Avg_T_adjust[0]=.0;
    Avg_T_adjust[1]= ch_time[1] - ((ch_time[2]+ch_time[0])/2);
    Avg_T_adjust[2] = .0;

    std::cout << "Mean hit time adjust for chamber 0: " << Avg_T_adjust[0] << std::endl;
    std::cout << "Mean hit time adjust for chamber 1: " << Avg_T_adjust[1] << std::endl;
    std::cout << "Mean hit time adjust for chamber 2: " << Avg_T_adjust[2] << std::endl;

    for (int chamb_idx = 0; chamb_idx < 3; chamb_idx++) //chambers loop
    {
        //adjust strip offset
        for (int str_idx=0;str_idx<24;str_idx++) // secondo loop sulle strip
        {
            //offset tra la media della strip e la media della camera
            if (MeanT_map[make_pair(chamb_idx,str_idx)]>0)
                delta_time=((ch_time[chamb_idx] - Avg_T_adjust[chamb_idx]) -MeanT_map[make_pair(chamb_idx,str_idx)]);  //ns
            else delta_time=0.0;
            Calib_map[make_pair(chamb_idx,str_idx)].first=delta_time;   //hit time = (Tleft+Tright)/2
            Calib_map[make_pair(chamb_idx,str_idx)].second=delta_time;

            //std::cout<<"strip "<<strIdx<<" time correction: "<< delta_time<<std::endl;
            //correzione spaziale


            x_offset = MeanX_map[make_pair(chamb_idx,str_idx)];
            std::cout<<"strip "<<str_idx<<" x cm offset: "<< x_offset<<std::endl;
            strip_offset=x_offset*2.0/EEEHit::SpeedOfPropagation; //mean ns strip offset
            //std::cout<<"strip "<<strIdx<<" y ps offset: "<< strip_ps_offset<<std::endl;

            Calib_map[make_pair(chamb_idx,str_idx)].first+=strip_offset/2.0;   //negative side
            Calib_map[make_pair(chamb_idx,str_idx)].second-=strip_offset/2.0;   //positive side
            std::cout << "Chamber " << Chamber_name[chamb_idx] << ", strip " << str_idx << " left:" << Calib_map[make_pair(chamb_idx,str_idx)].first << s
            std::cout << "Chamber " << Chamber_name[chamb_idx] << ", strip " << str_idx << " right:" << Calib_map[make_pair(chamb_idx,str_idx)].second <<
        }

    }
```

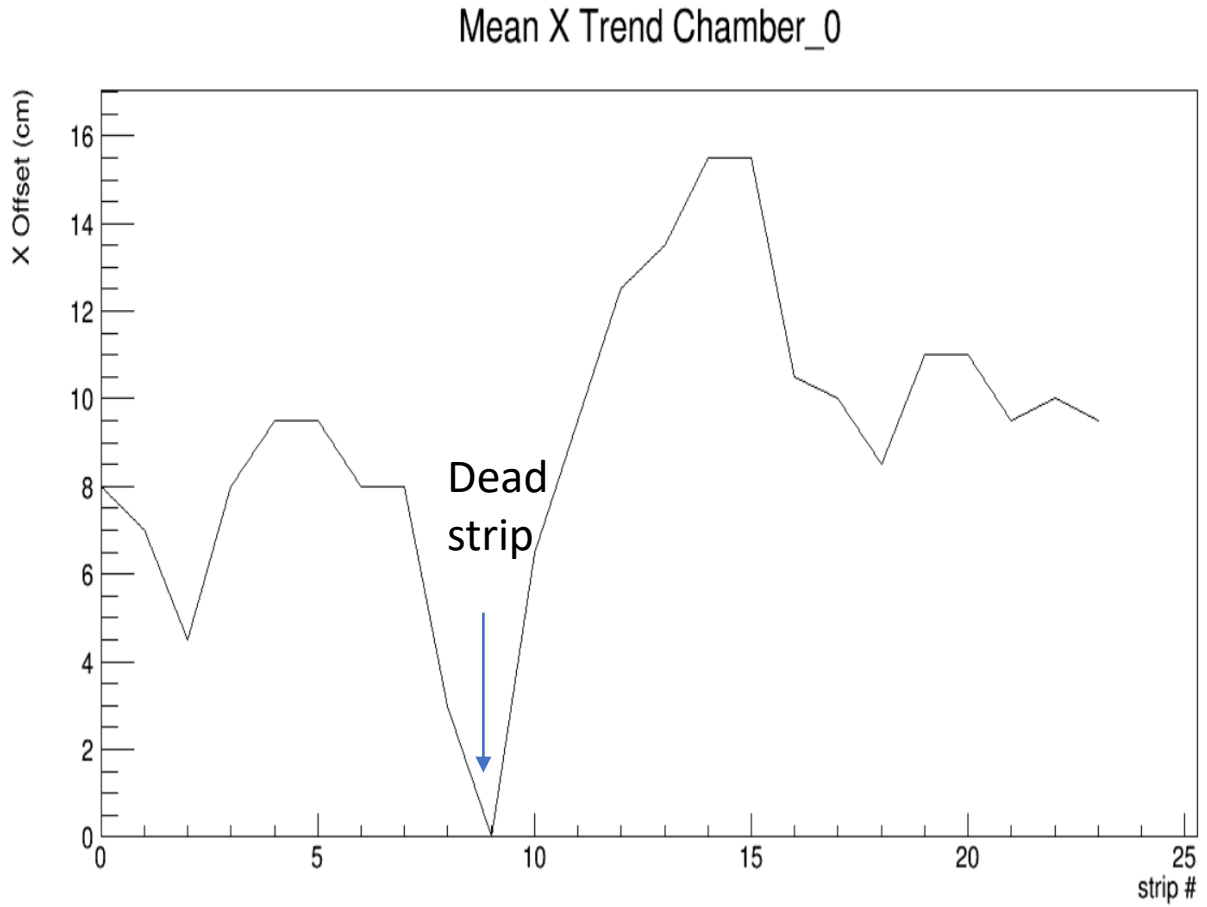
Average chamber correction

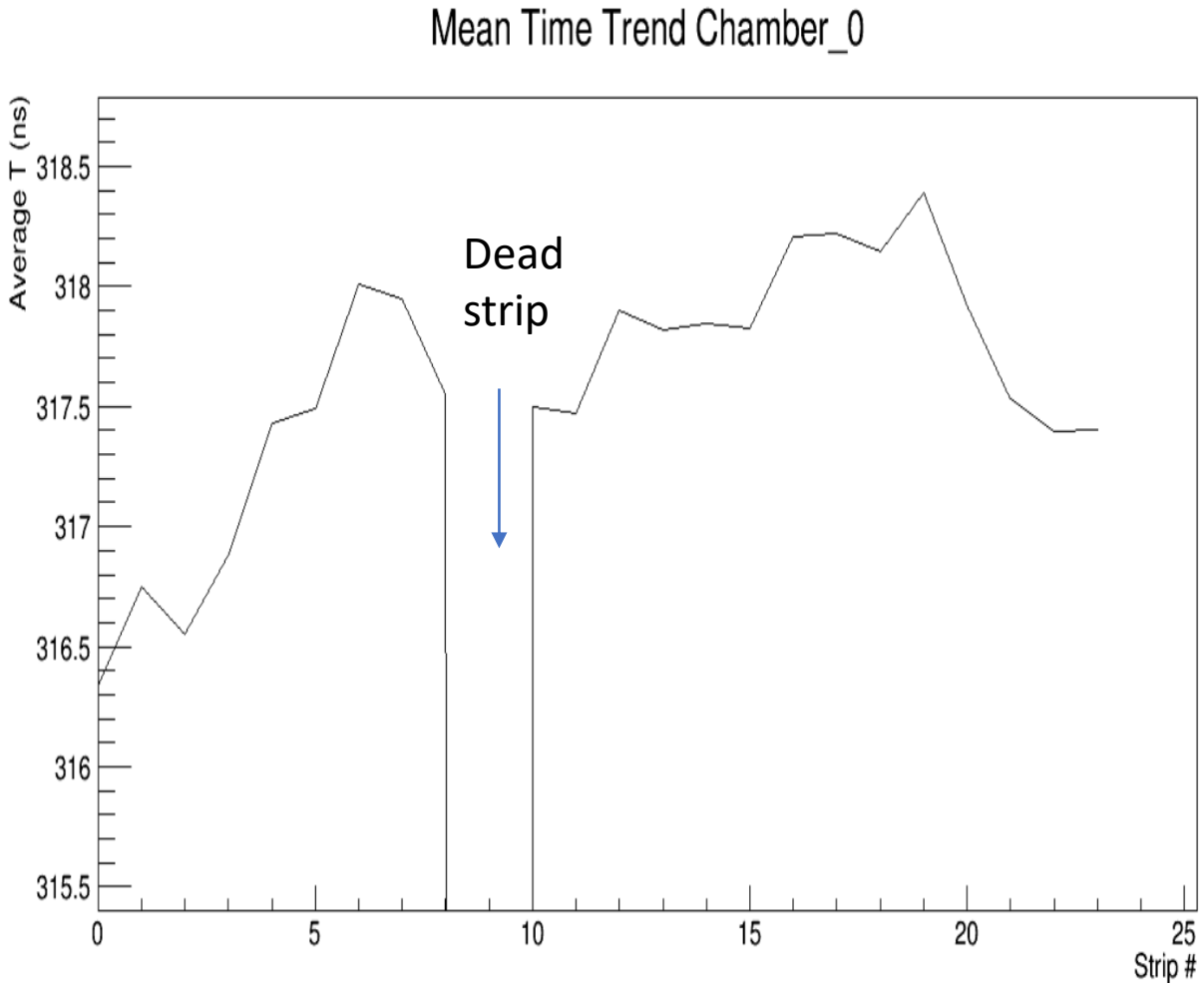
Time calibration

Space calibration

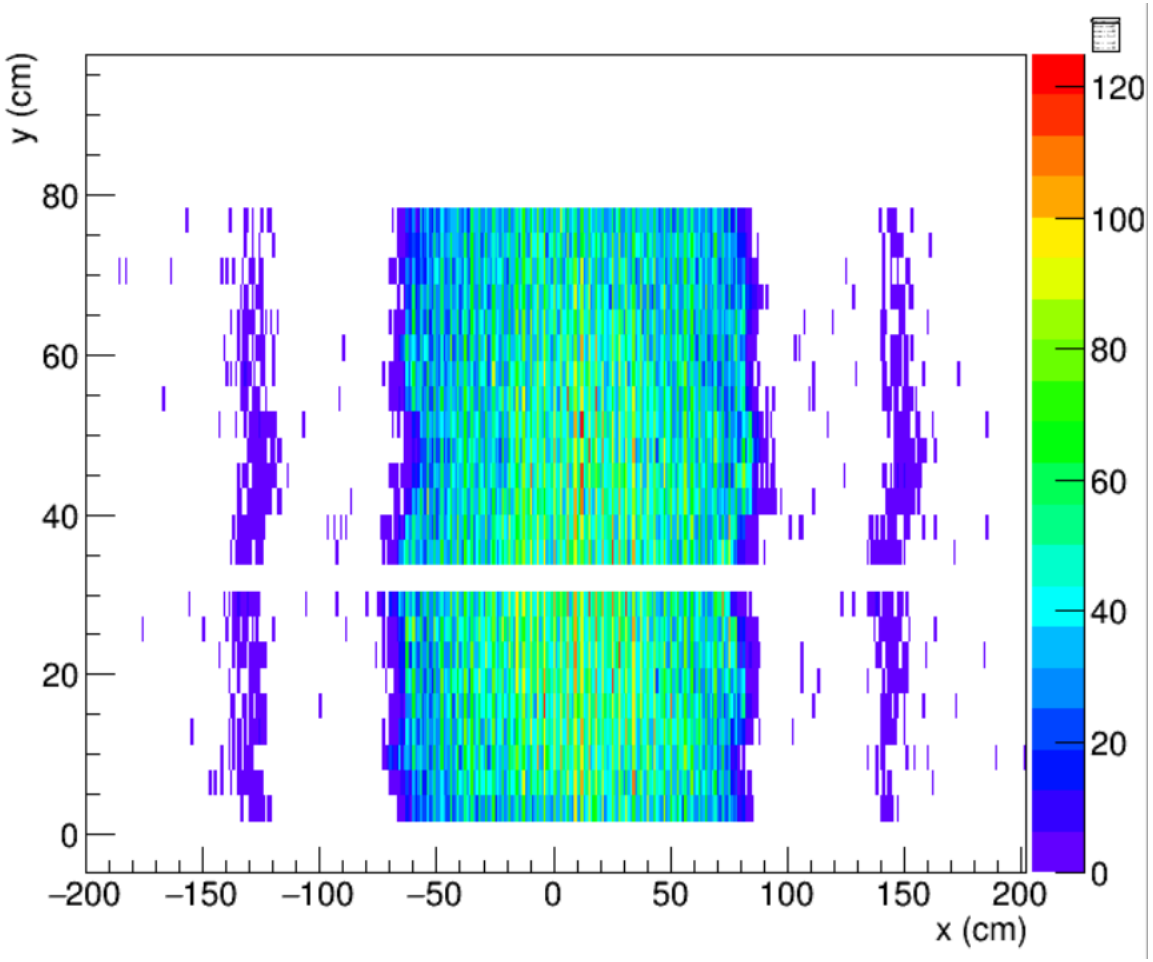
Calibration can be saved/retrieved

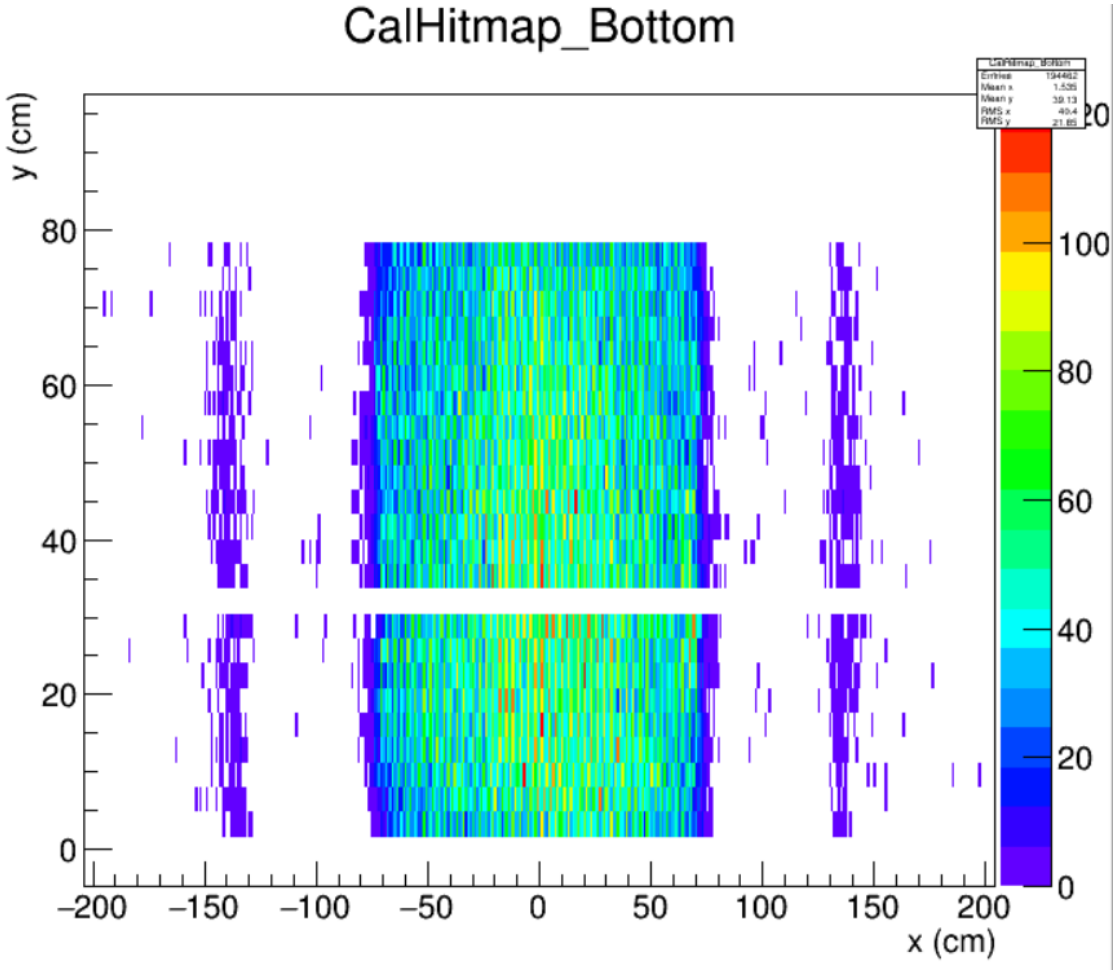
Raw data offset



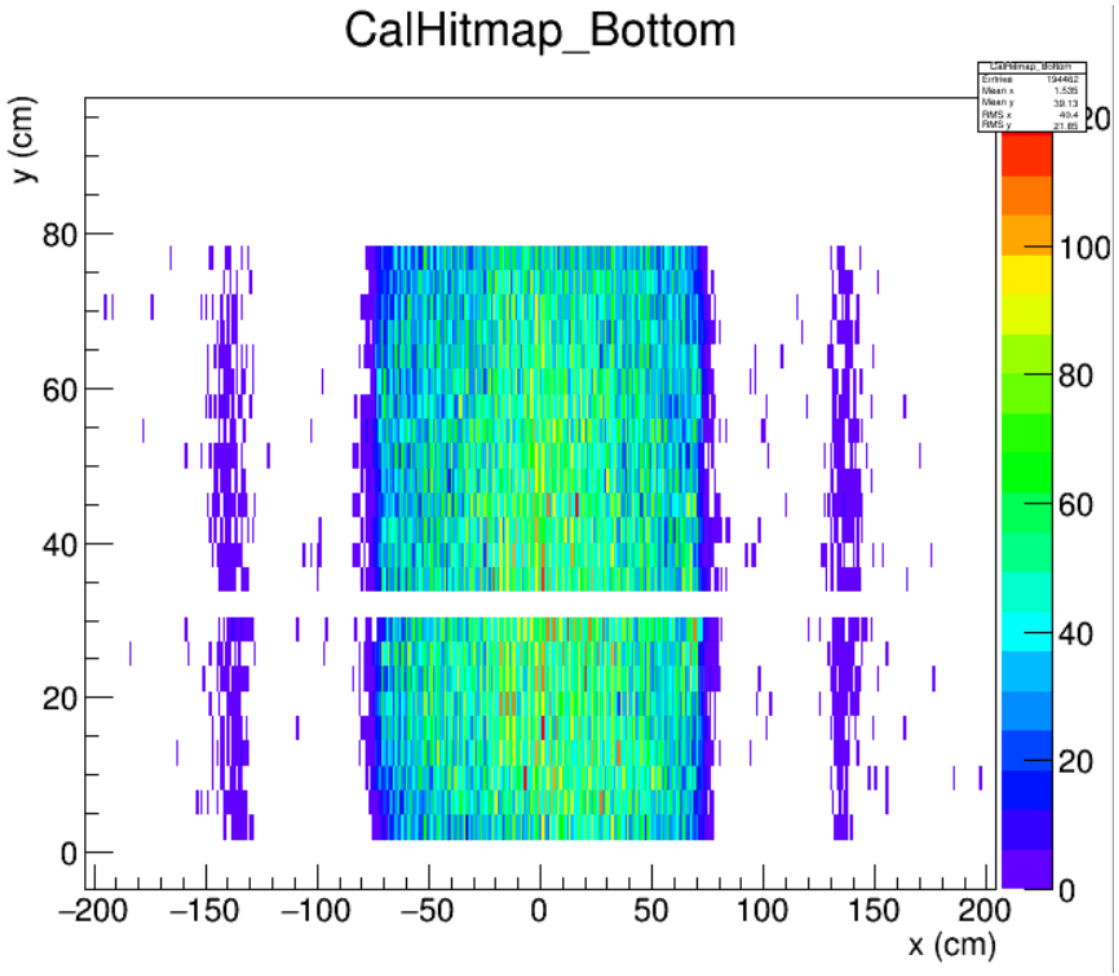


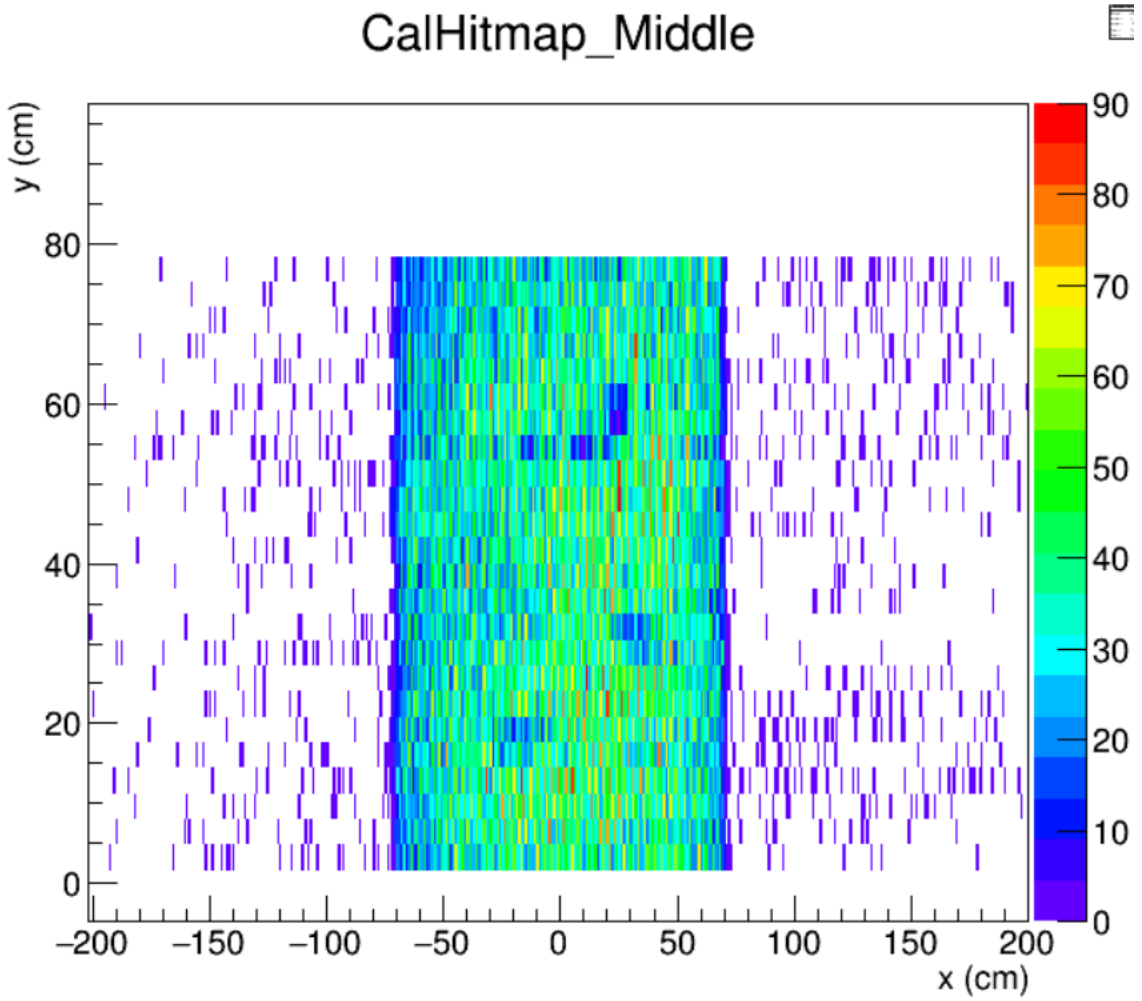
Calibration impact





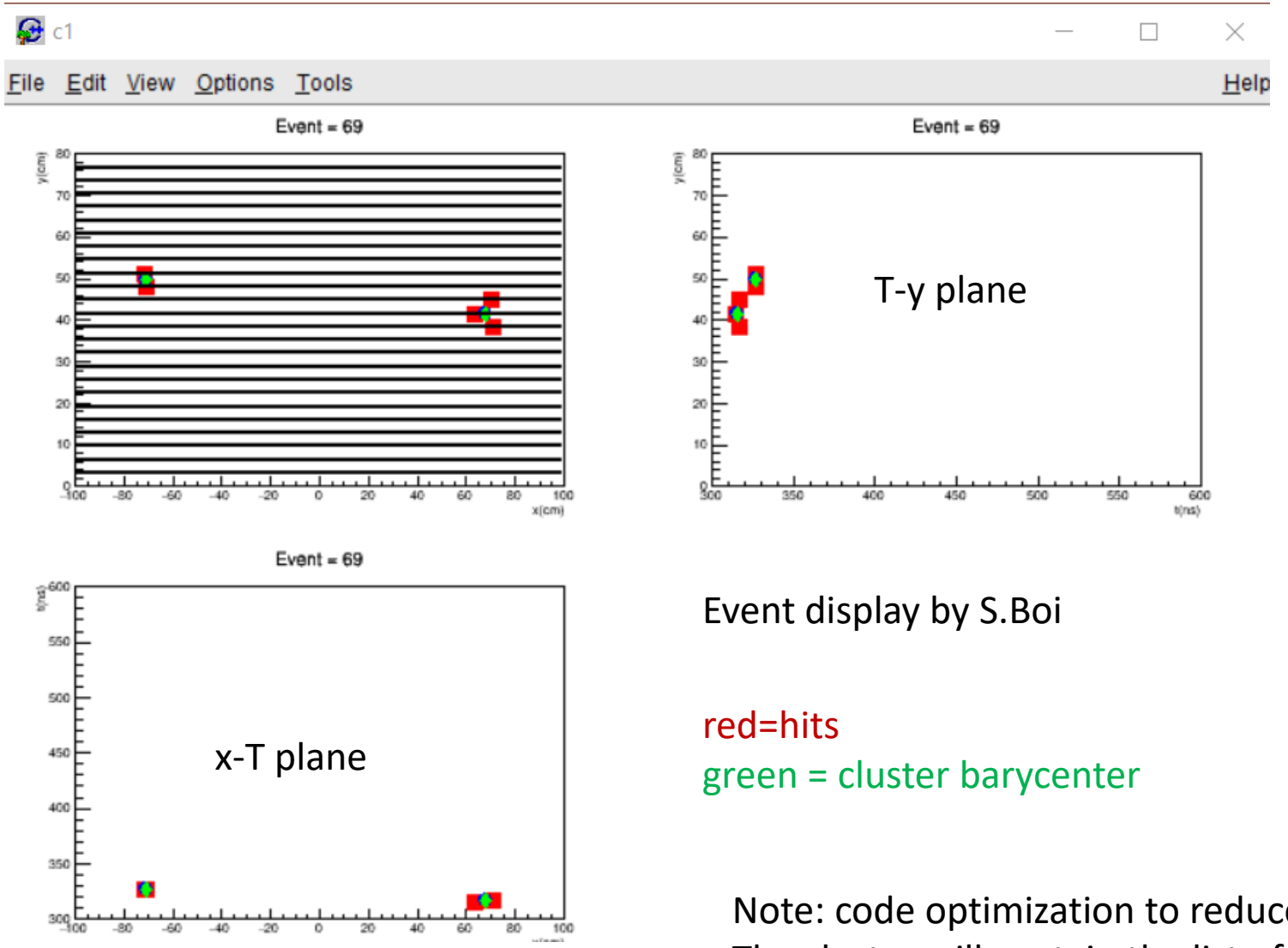
Hits outside chamber edges





In test chamber, hits reconstructed outside active region of the chamber are accumulated in a well-defined area.
Reason still not clear, they represents the ~0.06% of the total.

Clusters



Event display by S.Boi

red=hits

green = cluster barycenter

The algorithm in short:

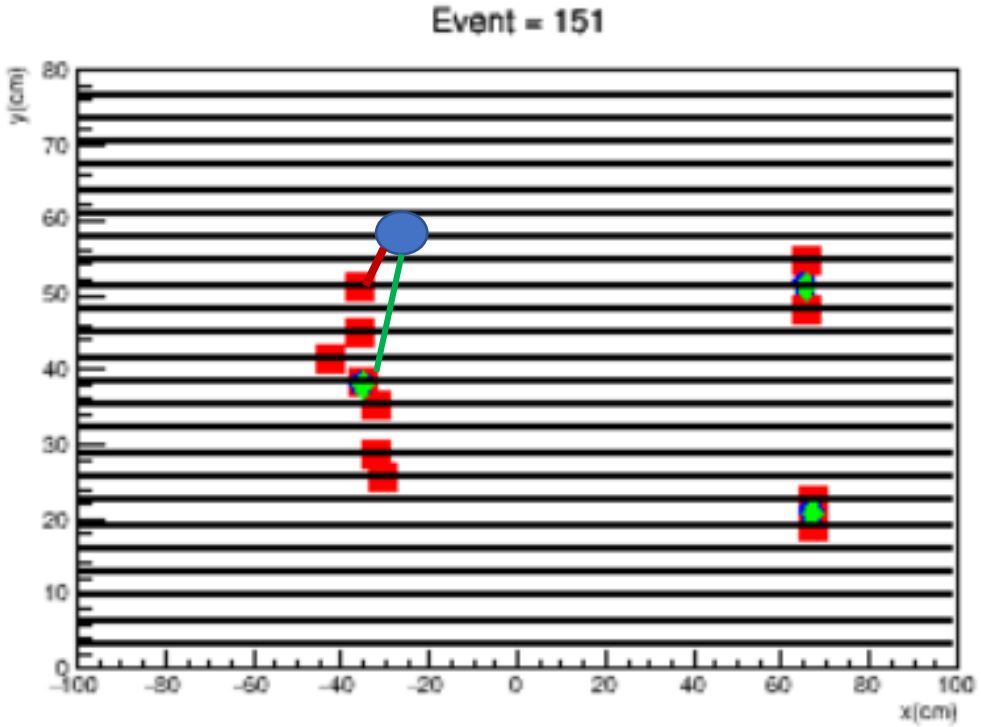
1. First hit is promoted to cluster and removed from hit array.
2. Scan over the hit array to find the first hit with XY distance below 10cm. Metrics: minimum distance between the hit and all the hits already part of the cluster
3. If some hit is added to cluster, remove it form the hit array and go back to point 2.

4. When no more hits can be added to the present cluster, compute cluster parameter (baricenter, T, average ToT)
5. if the hit array is not empty, create a new cluster with first hit and go back to point 2.

Note: code optimization to reduce clusterization step to few seconds (100K events)
The cluster will contain the list of hits -> useful in the last stage of the analysis

Efficiency

Event selection as in the past but new metrics to check the “distance” between a cluster and the projected hit



Old metrics: Distance to barycenter

New metrics: distance to closest hit of the cluster

Cluster multiplicity and streamer are computed from the multiplicity of best-matching cluster.

Selection cut on the triggered events (some tuning still needed):

- Extrapolated hit within test chamber acceptance (5<y<75 cm, -60<x<60 xm)
- zdir > 0.9
- particle inverse beta within 0.2 from the average inverse beta peak

Criteria for efficiency:

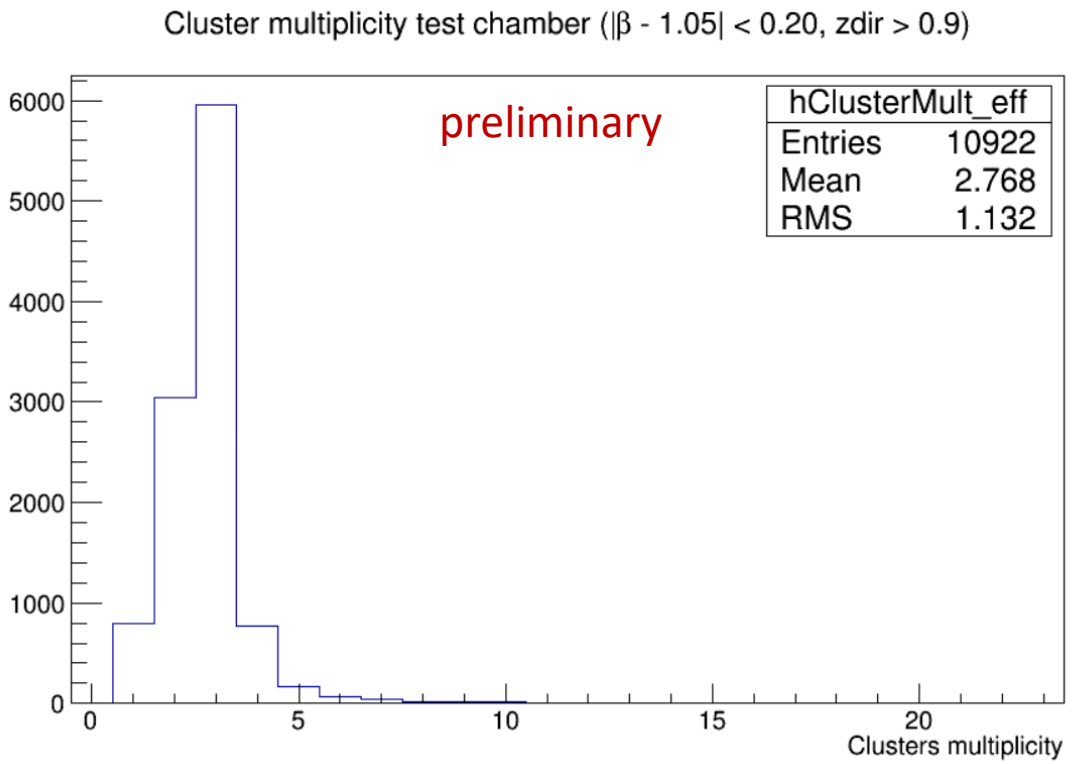
- distance between the extrapolated hit and the closest cluster below 15 cm
- Time difference between the extrapolated hit and the closest cluster below 4 ns

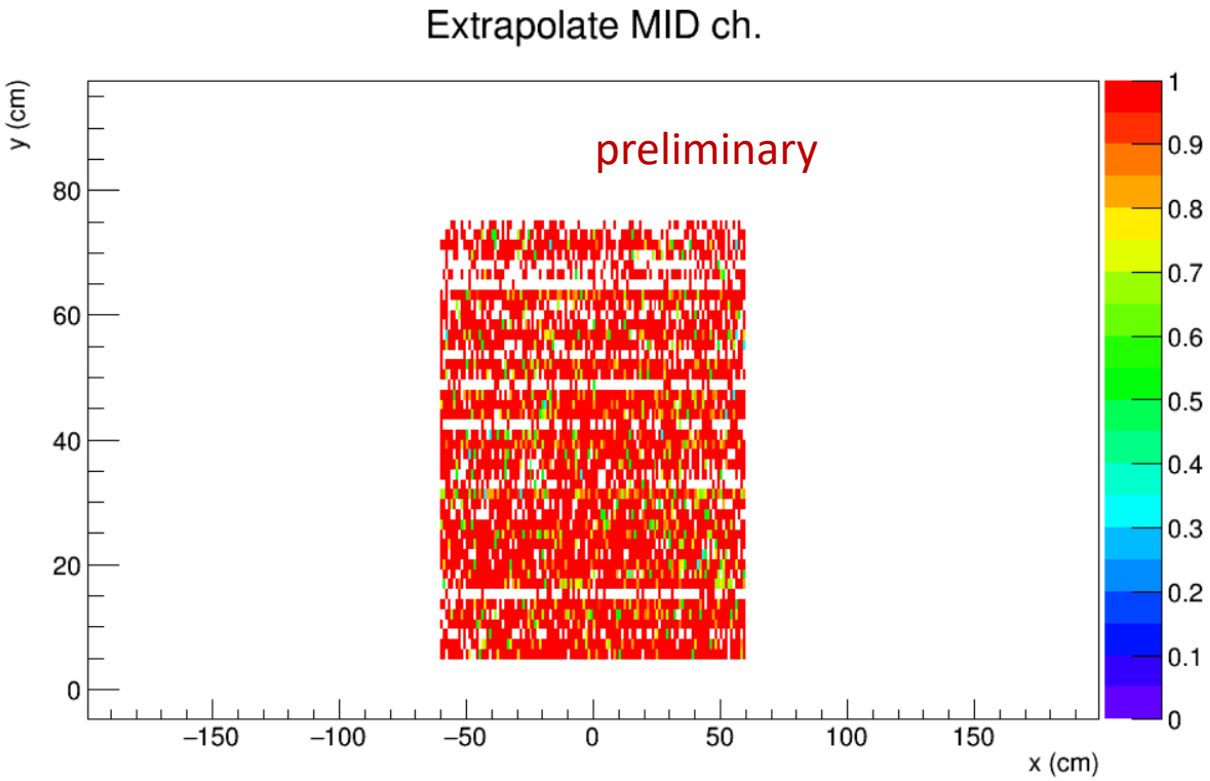
Option to remove residual background based on the inverse beta distribution available in the original code: check ongoing

Test file results

Results with test file:

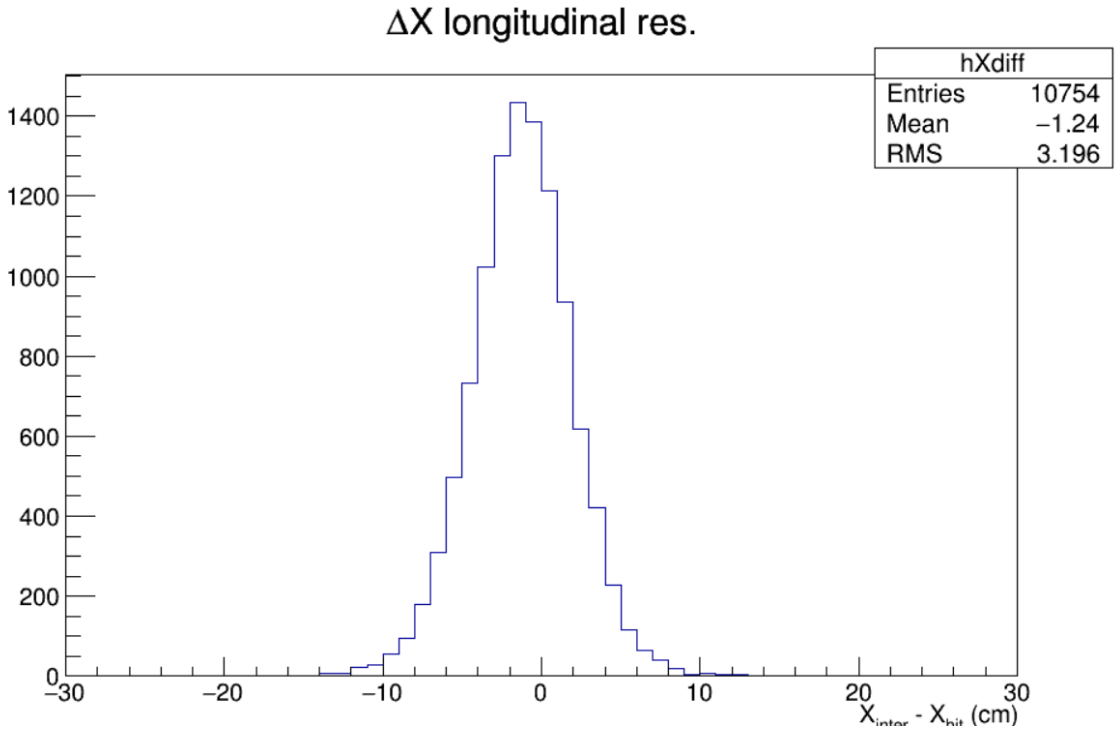
Global efficiency = 94% (previously 86% with same selection/cuts)
Streamer fraction = 0.095
Efficiency close to the one computed with CNAF efficiency code (96%)

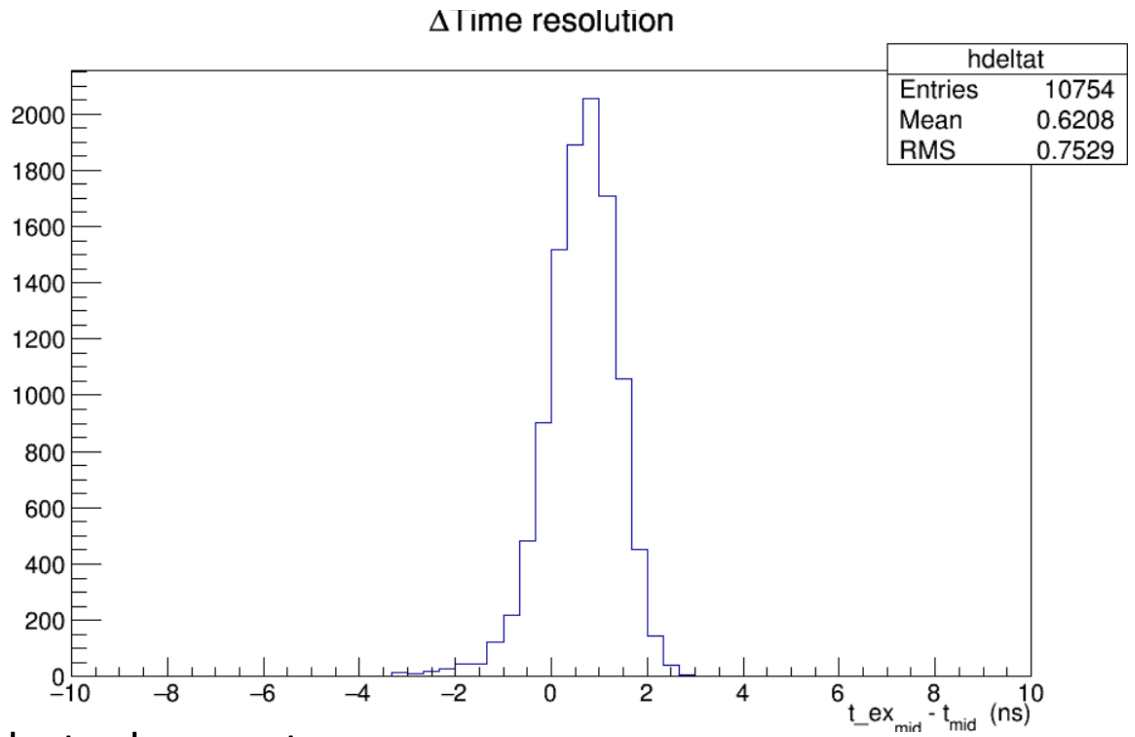




Test file results

Results with test file:
Global efficiency = 0.94 (previously 86% with same selection/cuts)
Streamer fraction = 0.095
Efficiency close to the one computed with CNAF efficiency code (0.96%)





Residuals computed w.r.t. cluster barycenter

Offset on X residuals due to x calibration method (binning width)
Offset on T residuals under investigation (probably due to non-weighted average of the hits)

Conclusion

Extreme Energy Events
Science inside Schools

- A Review of efficiency code(s) have been performed
- Critical aspects have been identified and corrected
- Code has been validated with a test run (R1234ze-He mixture in the chamber under test, plateau of efficiency)
- Further optimization/automatization of the code is ongoing (not much left...)

- After a validation on a larger set of runs we will be ready to proceed with the new efficiency plots
- In parallel re-reco of PISA data after fix of DST producer can be performed -> New plots of parameter distributions (beta, Theta, ToF,...)