6-3-2019

SFGD beamtest analysis

Systematic Unpacking, MC development, ghost hits and 3D charge reconstruction

C. Jesús-Valls

cjesus@ifae.es

UNPACKING PROCEDURE: (to unpack 1 run)

1. Extract .daq FEB files from each mini-crate's .daq file.
2. Convert .daq to .root
3. Apply calibration to .root
4. Convert calibrated .root to the new data structure.

NEW SCRIPTS: (for all runs)

Allow to send each of the 4 steps as a collection of jobs in LXPLUS. All data can be easily converted at once.

You can find it at:

/eos/experiment/neutplatform/t2knd280/Super_FGD/sFGD_beamtests_AugSep2018/ROOT_DATA

And the code can be found at:

https://gitlab.com/cesarjesusvalls/sfgd_unpacking_scripts.git

Is as easy to use as:

source unzip_all.sh ../run_list.txt

```
[cjesus@lxplus014 ROOT_DATA]$ ls
1September_1   1September_20  25August_16  26August_19  27August_16  28August_5   29August_18   2September_16  2September_28  30August_17  31August_18   3September_13  4September_1   5September_2
1September_10  1September_3   25August_17  26August_25  27August_3   28August_6   29August_19   2September_17  2September_29  30August_2   31August_19   3September_14  4September_12  5September_3
1September_11  1September_5   25August_18  26August_4   27August_5   28August_8   29August_5    2September_18  2September_9   30August_20  31August_20   3September_15  4September_14  5September_4
1September_12  1September_6   25August_8   26August_5   27August_9   28August_9   29August_6    2September_19  30August_1    30August_4   31August_21   3September_16  4September_17
1September_13  1September_8   26August_1   26August_6   28August_1   29August_1   29August_7    2September_21  30August_10   30August_5   31August_22   3September_3   4September_18
1September_14  1September_9   26August_10  26August_7   28August_12  29August_10  29August_8    2September_23  30August_11   30August_6   31August_6    3September_4   4September_4
1September_16  24August_13    26August_11  26August_9   28August_13  29August_11  29August_9    2September_24  30August_12   30August_8   31August_7    3September_5   4September_5
1September_18  25August_10    26August_12  27August_10  28August_17  29August_12  2September_10 2September_25  30August_13   31August_12  31August_8    3September_6   4September_7
1September_19  25August_11    26August_13  27August_14  28August_18  29August_13  2September_11 2September_26  30August_14   31August_15  3September_10 3September_7   4September_8
1September_2   25August_12    26August_16  27August_15  28August_4   29August_15  2September_15 2September_27  30August_16   31August_17  3September_11 3September_8   5September_1
```

The run_list.txt contains:
by Dana

```
23August_2 1 0 0 0
24August_1 2 0 0 0
24August_2 3 0 0 0
24August_4 4 0 0 0
24August_5 5 0 0 0
24August_6 6 0 0 0
24August_7 7 0 0 0
24August_8 8 0 0 0
24August_9 9 0 0 0
24August_10 10 0 0 0
24August_13 13 1 55 55
25August_1 14 0 55 55
25August_2 15 0 55 60
25August_3 16 0 55 60
25August_4 17 0 55 60
25August_5 18 0 55 60
25August_6 19 0 55 50
25August_7 20 0 55 50
25August_8 21 1 55 50
25August_9 22 0 55 50
25August_10 23 1 50 55
```

Currently

1st column: Name of the run

2nd column: Run number

3rd column: Status (0 bad // 1 good)

4th column: LG

5th column: HG

Future?

6th column: Beam Kind (0 hadron // 1 muon // … etc)

7th column: Beam momentum

8th column: Magnetic Field

…

From a detailed run_list.txt would be easy to find runs with same conditions to enlarge the statistics of our analysis.

TChain can be used to analyze sets of files stored in lists: Currently implemented in stopping_protons.C

Motivation:

To have a comparison for the real data analysis, debug algorithms with true information, etc, a realistic MC would be helpful.

Update:

- A 'realistic' MC has been developed, simulating data inputs using t2k-nd280-up GEANT4 framework.
- It provides data with 'realistic' crosstalk, and energy deposit in #p.e.
- It stores data in the dataStructure.root from Dana, namely in Event and Hit classes.

Some modifications have been done in the classes to store True information.

- The MC output and the sfgd_reconstruction (let's call it sfgd_analysis) work together: The analysis package includes options to deal with true information.

Remove/Include true crossTalk.

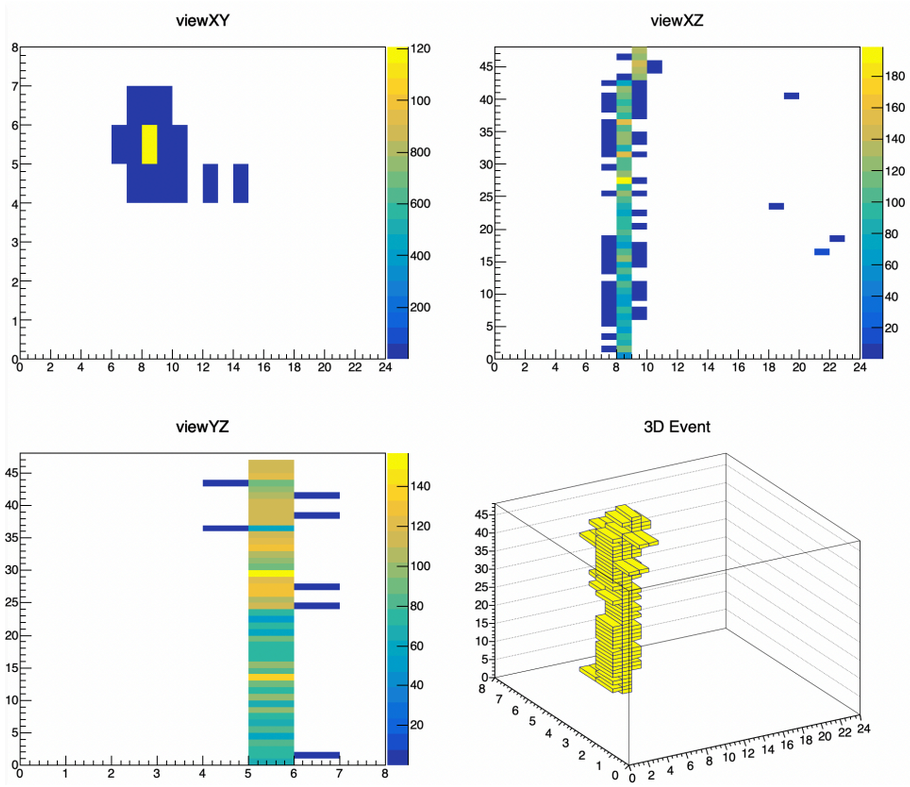
If you need it, please ask:

- To avoid you spending extra time learning to work with: 't2k-nd280-up' I can provide you the MC files that you require for your analysis in new DataStructure.root format.
- Also can give more details if someone is interested in the MC.

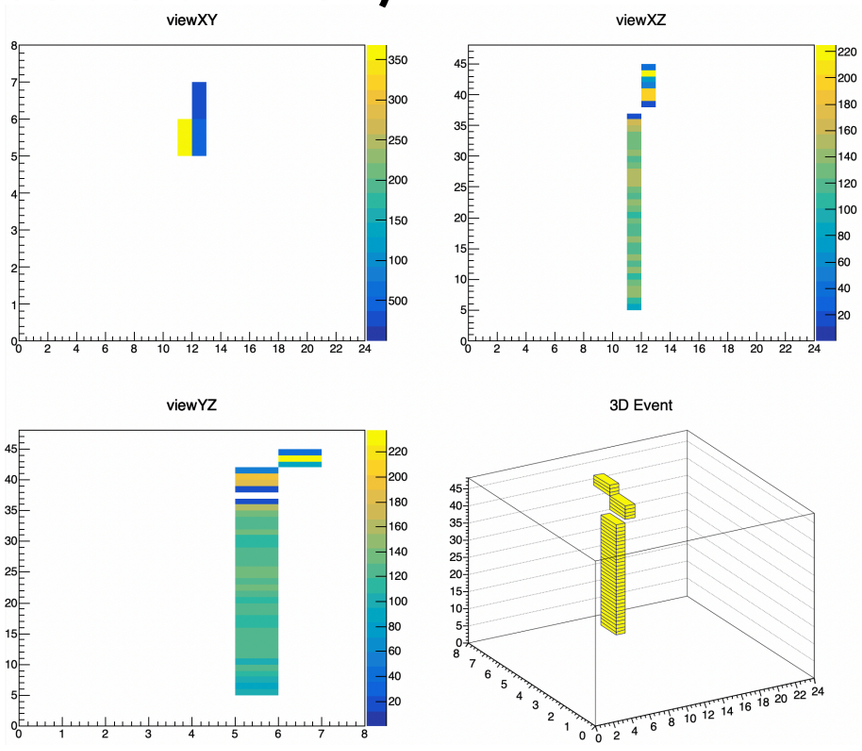
MC DEVELOPMENT :: CROSSTALK

- The results look quite similar concerning shape, crosstalk and #p.e.
- Still preliminar version.
- Time effects are not included.

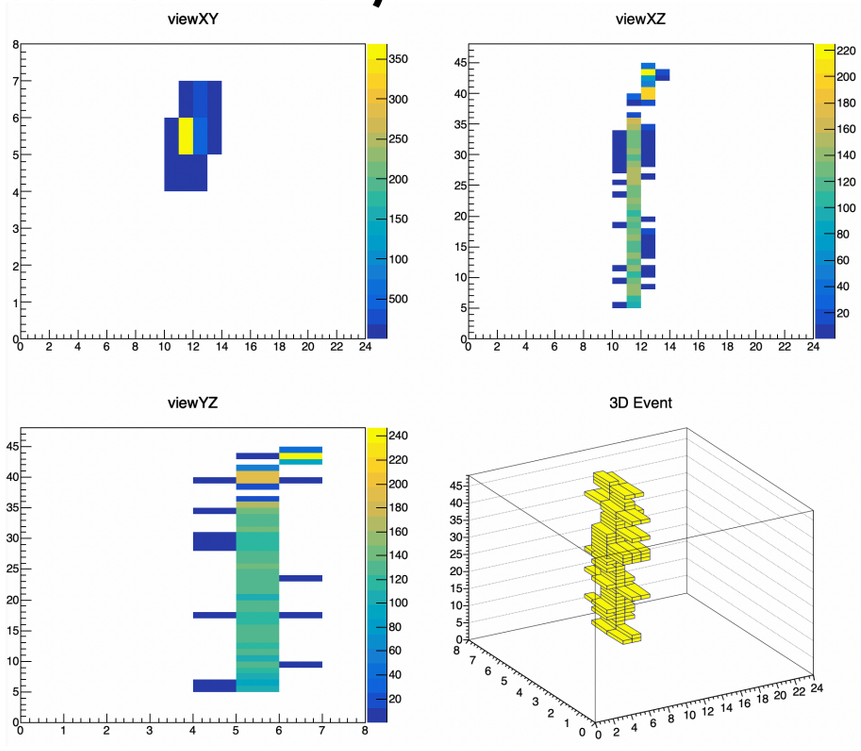
REAL EVENT:



MC (true crossTalk off)



MC (true crossTalk on)



For each true 3D cube with energy deposit, look to its X and Y coordinates.

Lateral crosstalk:
For X: Add a crosstalk hit at X+1 or X-1 with 45% probability (each).
For Y: Add a crosstalk hit at Y+1 or Y-1 with 2% probability (each).
Charge on the crosstalk cube (TotQ): 5% of energy deposit from its associated core cube.
Charge on each fiber of the crosstalk cube: Poisson distribution of TotQ/3.

Diagonal crosstalk:
X+1 & Y-1 // X-1 & Y-1 // X+1 & Y+1// X-1 & Y+1
Correlated to X and Y crosstalk probability. (if the Random–>Uniform associated to X+1 is small adds some difficulty to have crosstalk in X+1 & Y+1 // X+1 & Y-1)

Corner crosstalk (current not simulated)
X+1 & Y+1 & Z+1 // X+1 & Y+1 & Z-1 // X-1 & Y+1 & Z+1 // X-1 & Y+1 & Z-1 // X+1 & Y-1 & Z+1 // X+1 & Y-1 & Z-1 // X-1 & Y-1 & Z+1 // X-1 & Y-1 & Z-1

FUTURE:
Precise values depend on aside studies (all input are more than welcome!) I want to implement dependence of crosstalk in the probability.

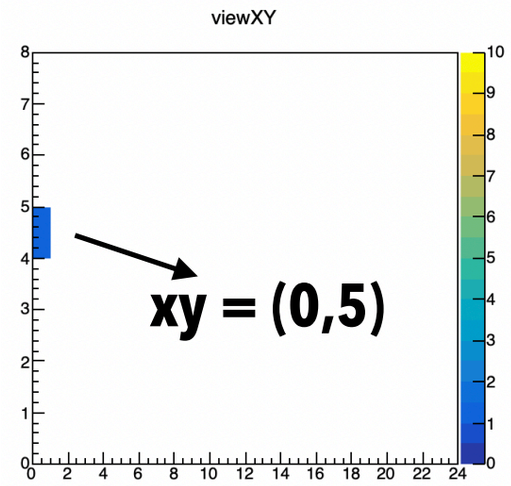
Current matching algorithm:

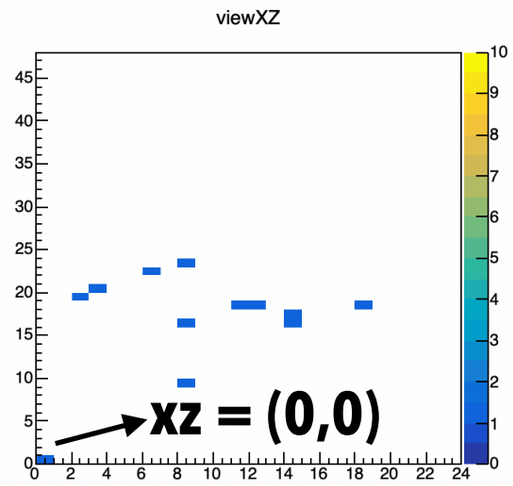
if hits in viewXY and viewXZ share X, add X,Y,Z and hits (2) to vX
if hits in viewXY and viewYZ share Y, add X,Y,Z and hits (2) to vY
if hits in viewXZ and viewYZ share Z, add X,Y,Z and hits (2) to vZ

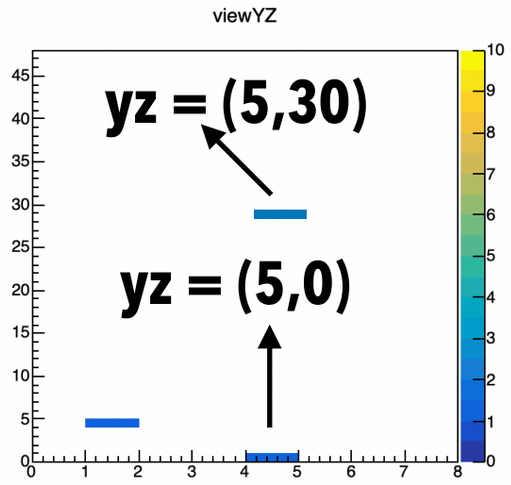
if vX and vY share X,Y,Z, add X,Y,Z and hits (3) to vXY
if vX and vZ share X,Y,Z, add X,Y,Z and hits (3) to vXZ
if vY and vZ share X,Y,Z, add X,Y,Z and hits (3) to vYZ

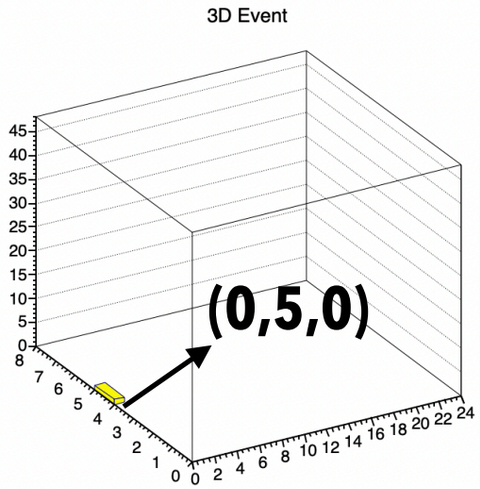
select **all** different combinations of X,Y,Z, and hits(3).

The blue conditions ensure that all voxels
are constructed from 3 fibers with at least 1 hit.









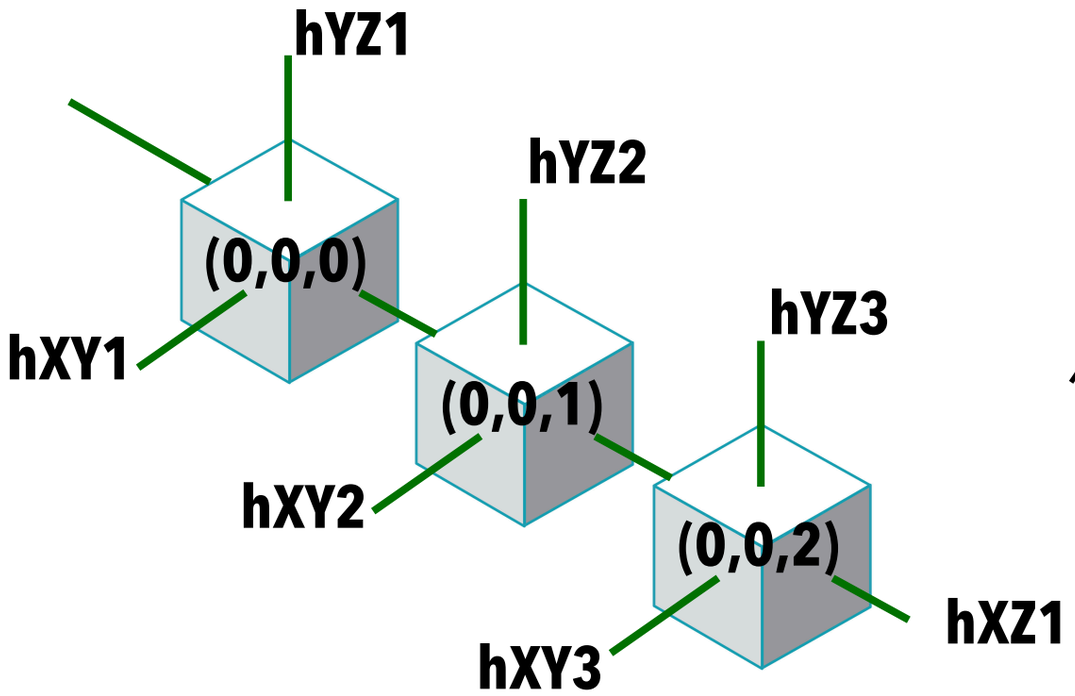
What happens if MPPC not records a hit? ⟶ With current algorithm **voxel is lost.**

What happens if we remove green conditions? ⟶ **'Ghost hits' appear.** ⟶ **Is it possible to have ghost hits anyway?**

Using xy = (0,5) and noise hit yz = (5,30) we construct the irreal voxel xyz = (0,5,35).
To be real I should see xz = (0,35)

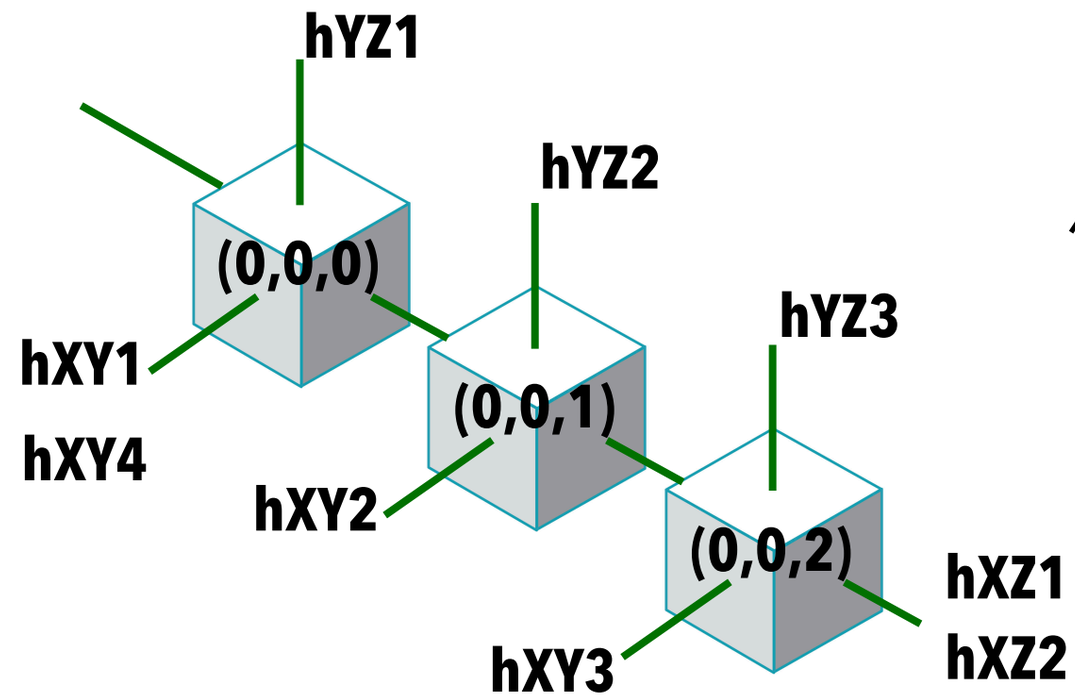
select **all** different combinations of X,Y,Z, and hits(3).

Consider the following situations



Reconstructed:

(0,0,0): hXY1 hYZ1 hXZ1
(0,1,0): hXY2 hYZ2 hXZ1
(0,2,0): hXY3 hYZ3 hXZ1



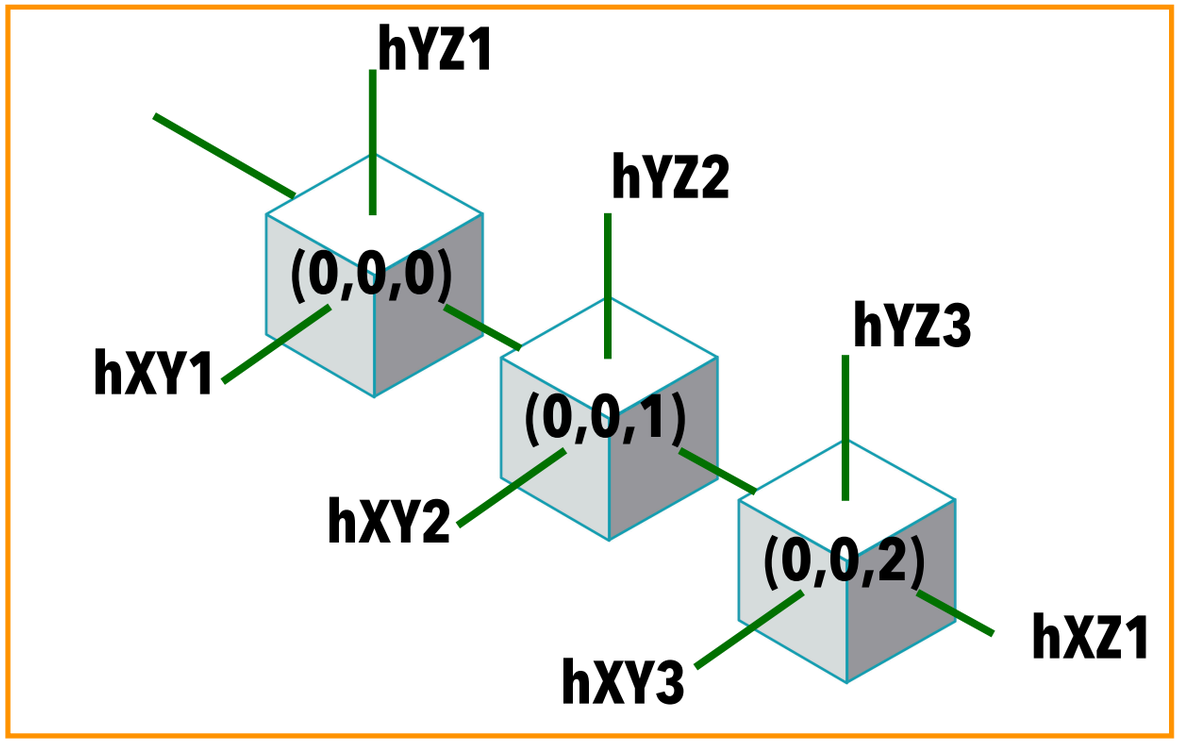
Reconstructed:

(0,0,0): hXY1 hYZ1 hXZ1
hXY4 hYZ1 hXZ1
hXY1 hYZ1 hXZ2
hXY4 hYZ1 hXZ2
(0,1,0): hXY2 hYZ2 hXZ1
hXY2 hYZ2 hXZ2
(0,2,0): hXY3 hYZ3 hXZ1
hXY3 hYZ3 hXZ2

hXY4 is used to reconstruct 2 voxels, it has multiplicity 2
(0,0,0) is reconstructed with 4 hits combinations, this voxel has multiplicity 4. ⟶ This is was complete caos.

NOW: merge **all** hits sharing a fiber.

Only 1 hit per fiber: Indistinguishable from time arguments!! (at least in the prototype)



Reconstructed:

(0,0,0): hXY1 hYZ1 hXZ1
(0,1,0): hXY2 hYZ2 hXZ1
(0,2,0): hXY3 hYZ3 hXZ1

No multiplicity in Voxels (unique XYZ).
Multiplicity in hits. E.g: hXZ1 has multiplicity 3.

How do we reconstruct charge?

Right now not using XY fiber (using beamtest like pgun)

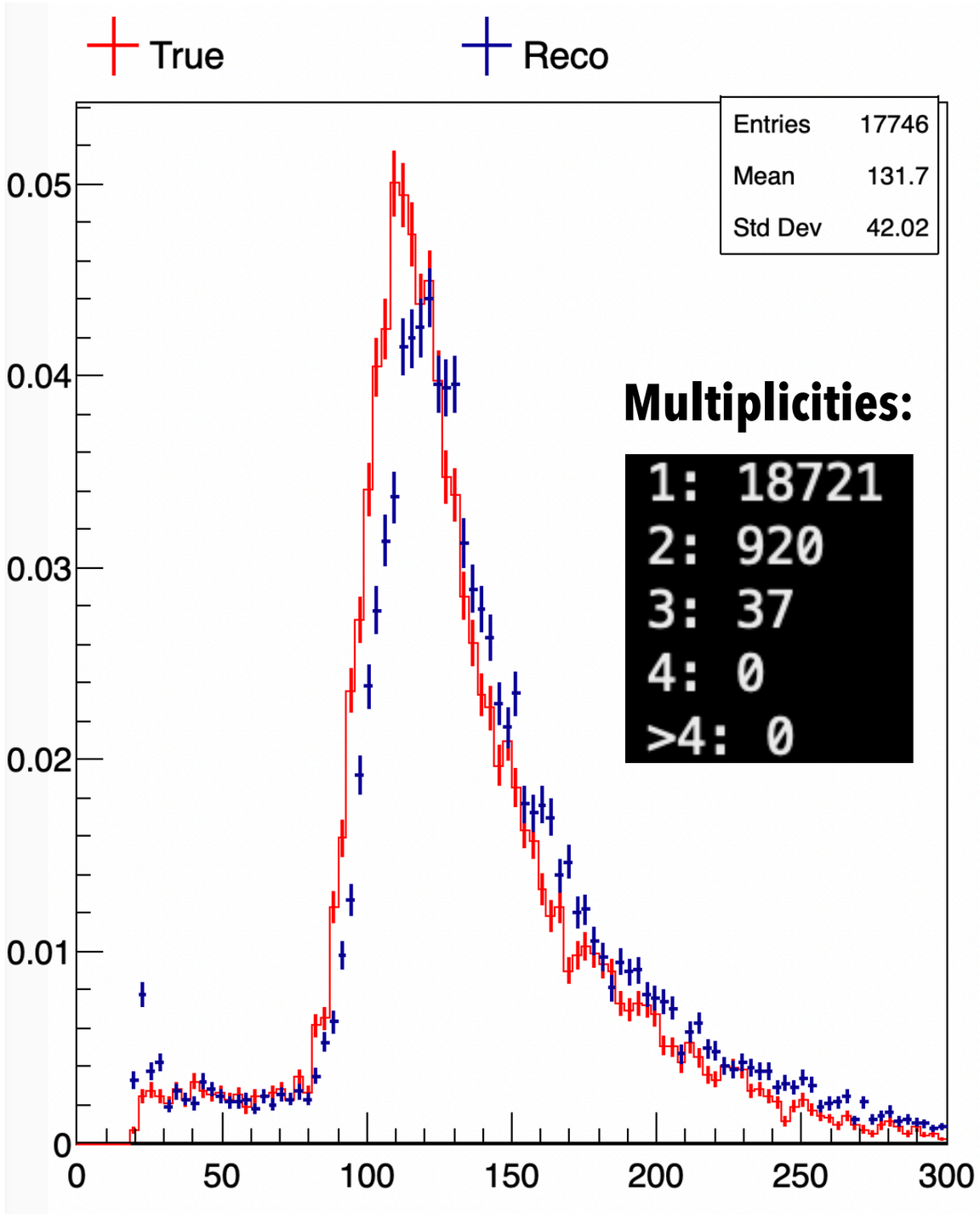
What happens with fibers with several core cubes? → **Ignore highly degenerated information.**

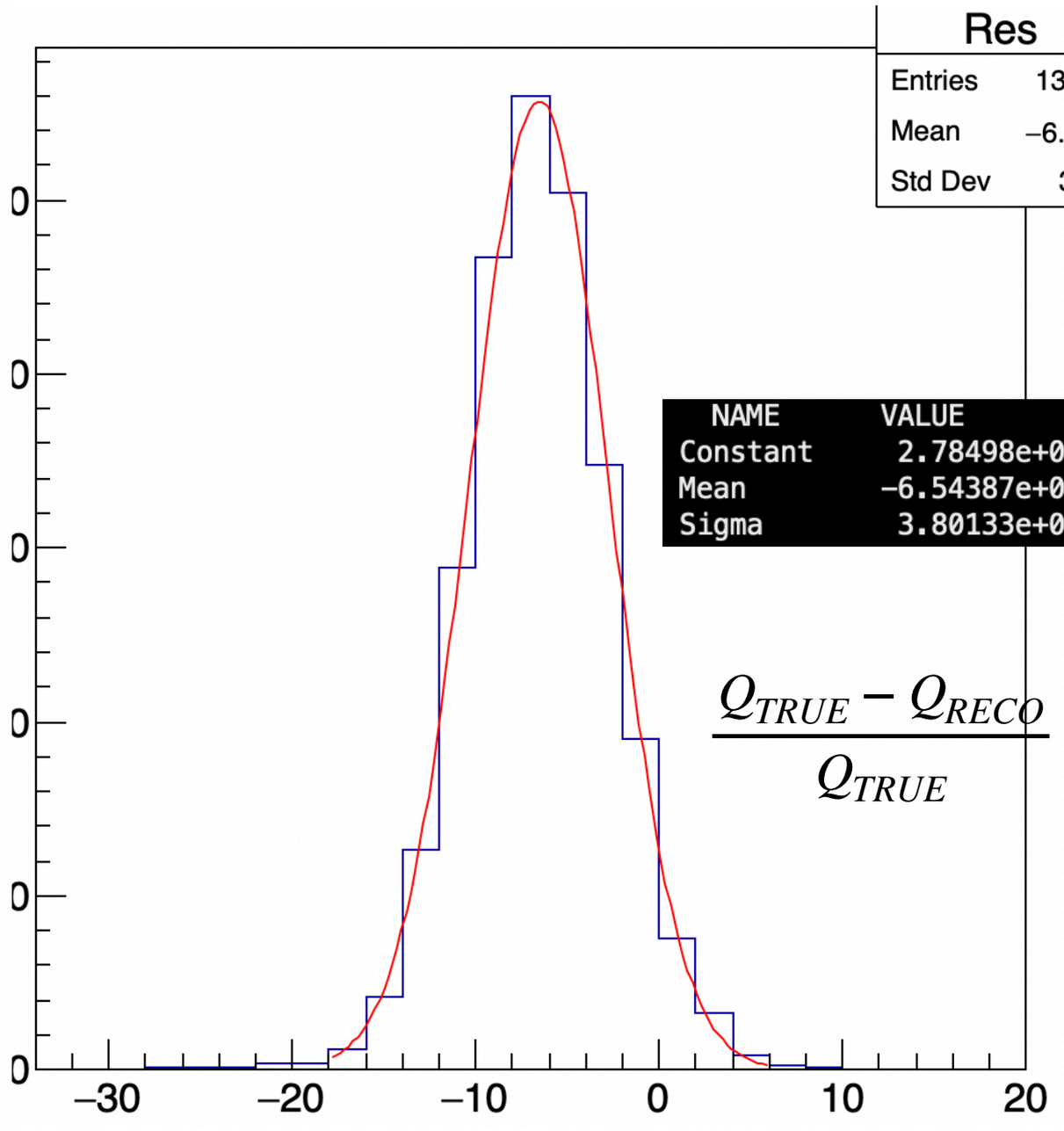
Iterations $N_0 = 1$

There is at least 1 hit with multiplicity N? → **Yes** → **1** → Q_i
Yes → **2** → $(Q_i+Q_j)/2$
No → **3** → $(Q_i+Q_j+Q_k)/3$
No → **N++**

Cut: Ignore hits with <10p.e

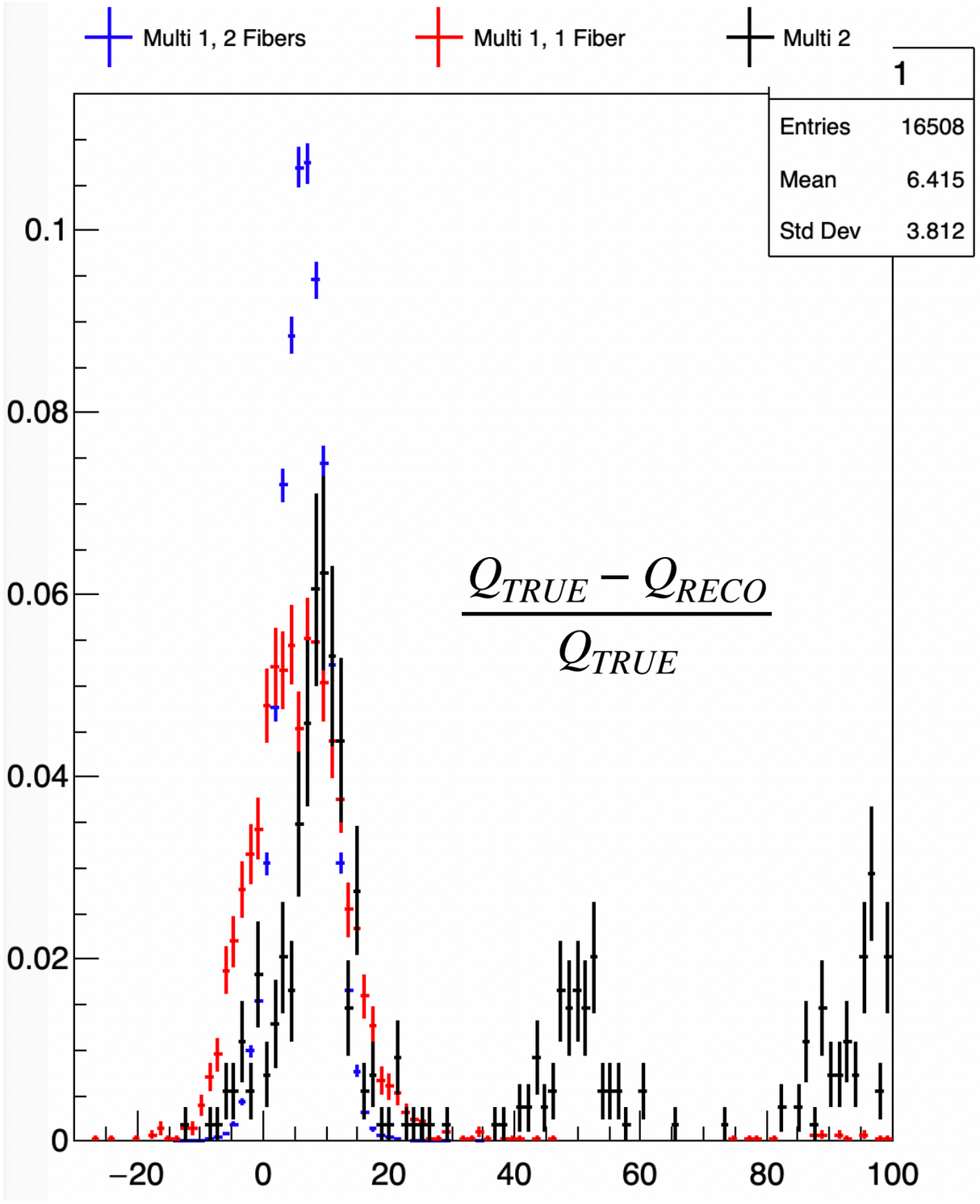
The MC stores 3MPPC hits per cube –––> **True Charge :** (ChargeXY + ChargeXZ + ChargeYZ) / 3





How does it look depending on the multiplicity?

Cut: Ignore hits with <10p.e

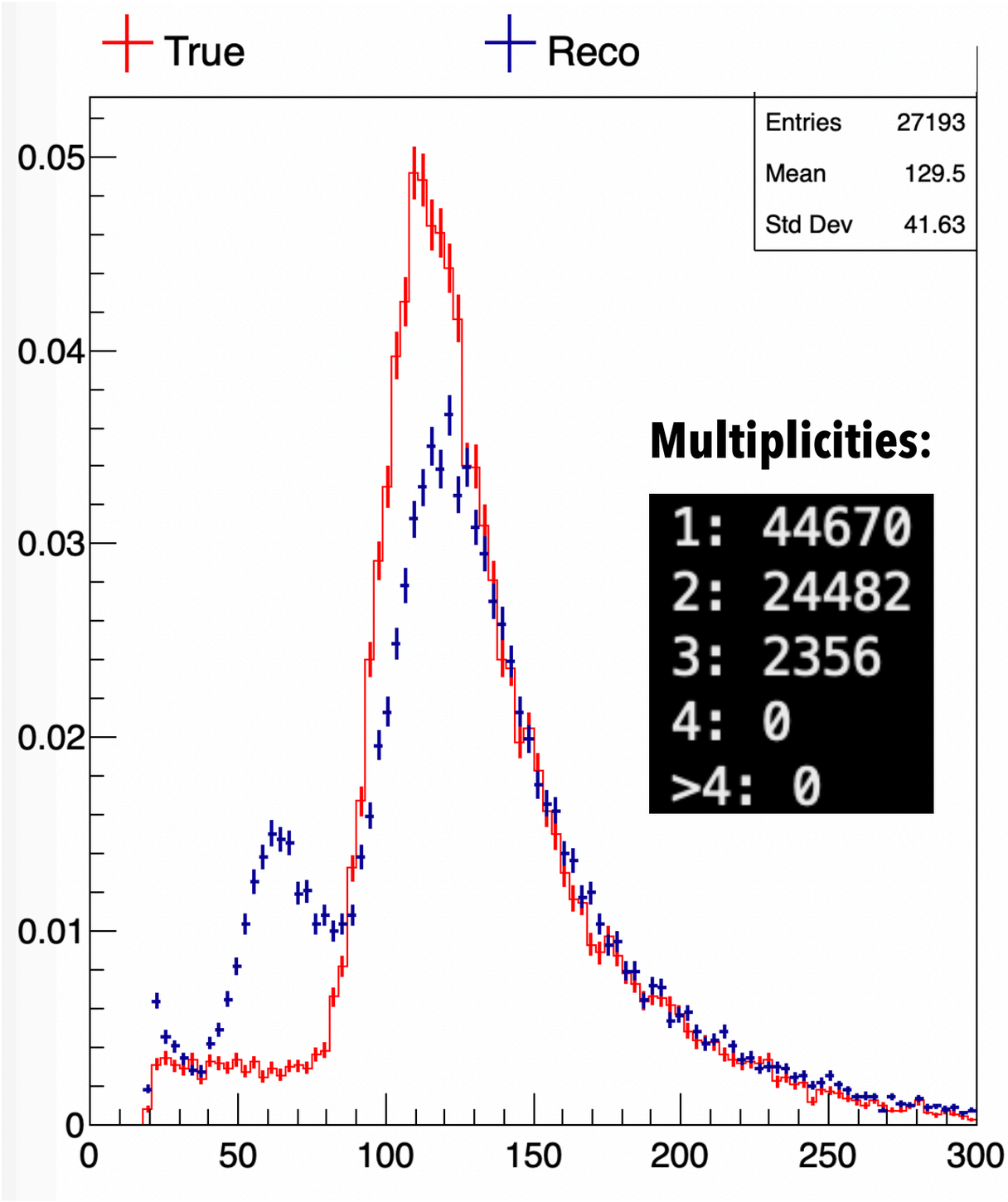


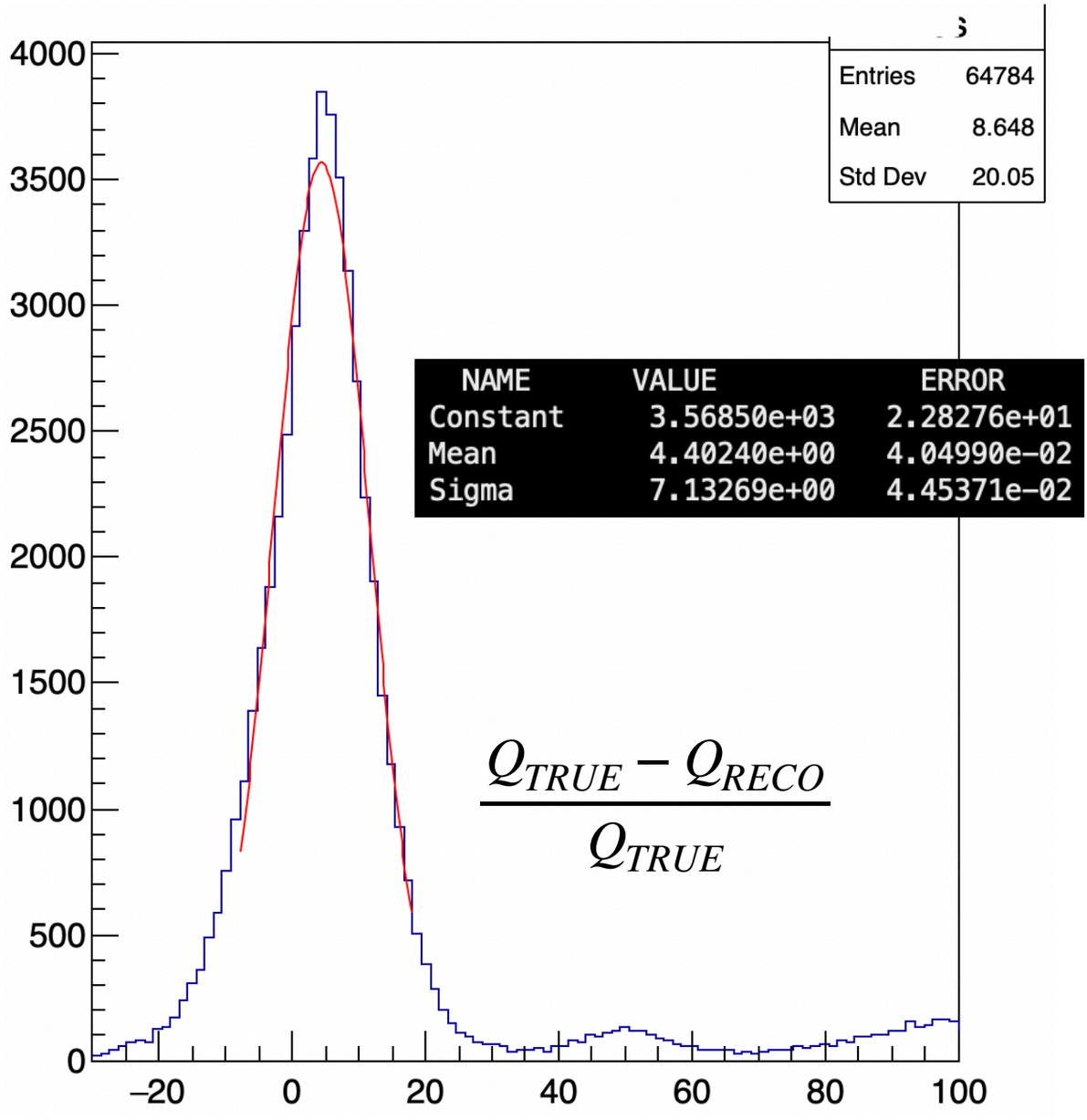
Multiplicity 2 starts being dangerous.

If this is how it looks with cut 'on'.
Multiplicity 1 is 95% of the voxels!

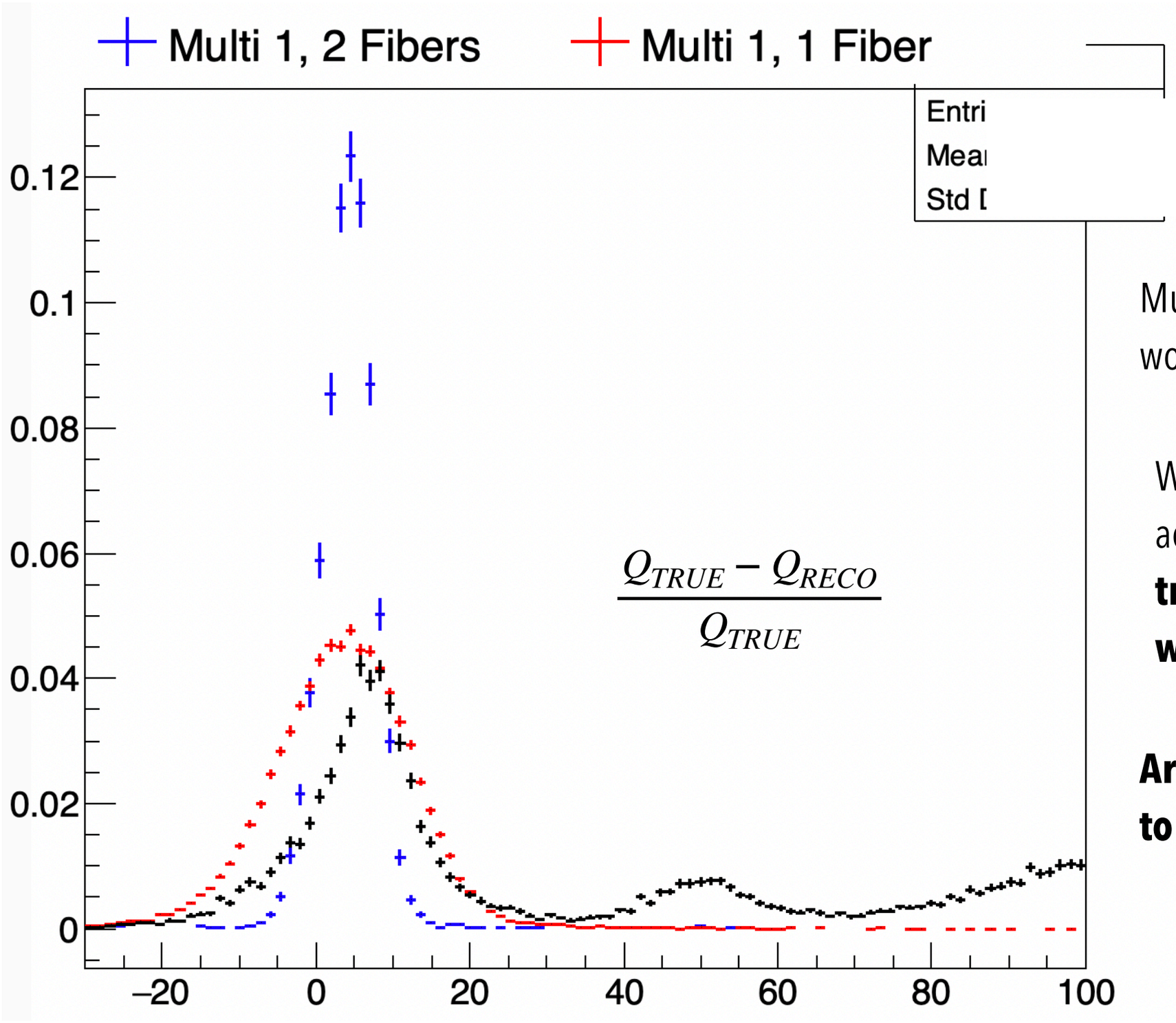
How does it look when we set the cut off?

3D CHARGE: WITHOUT CUTS





How does it look depending on the multiplicity?



Multiplicity 3 not plot, but has to be worse than Multiplicity 2…

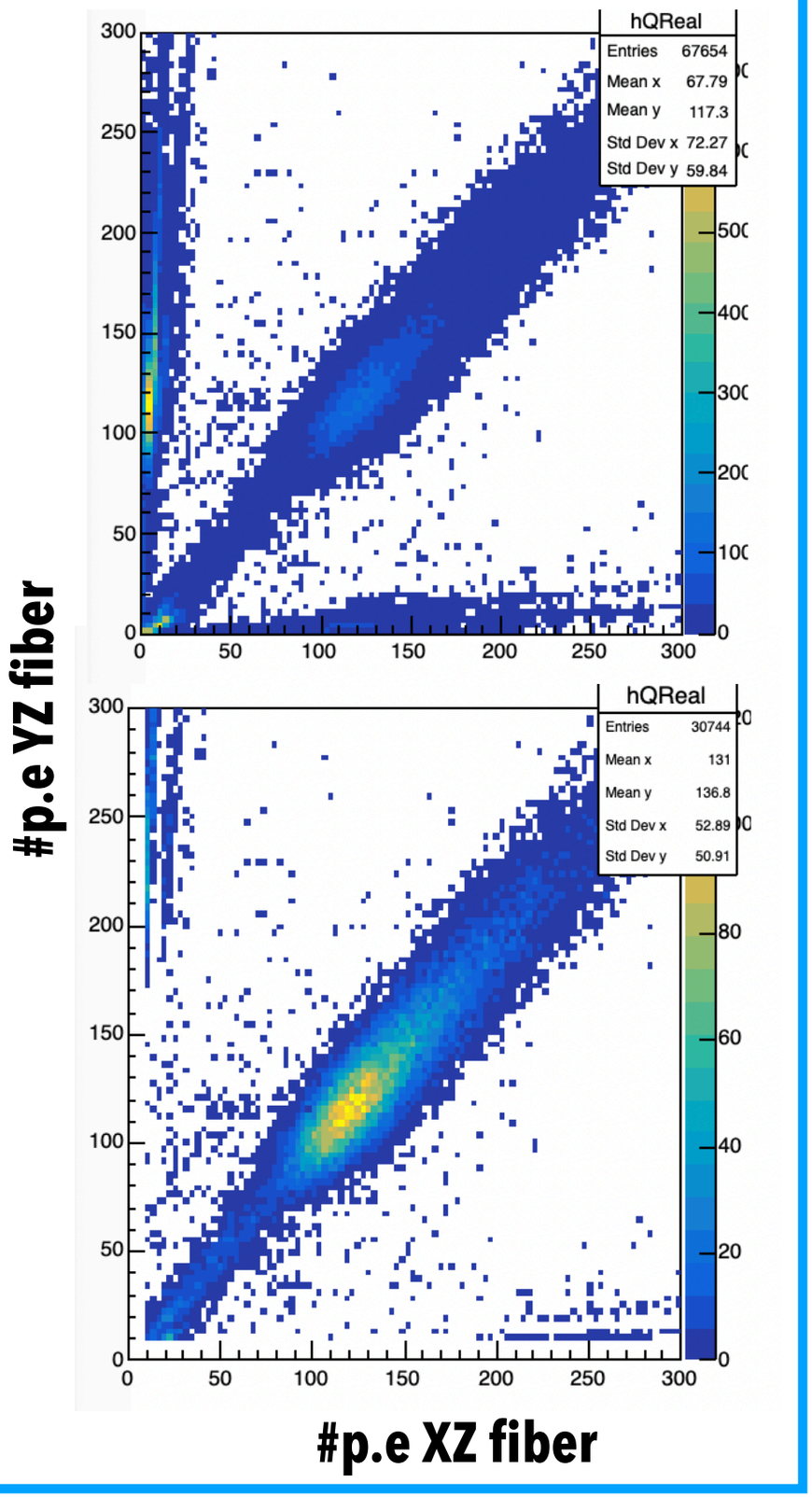
We have clear motivation to activate the cut but… **how much true core voxels do we lose with it?**

Are there any other motivations to apply the cut?

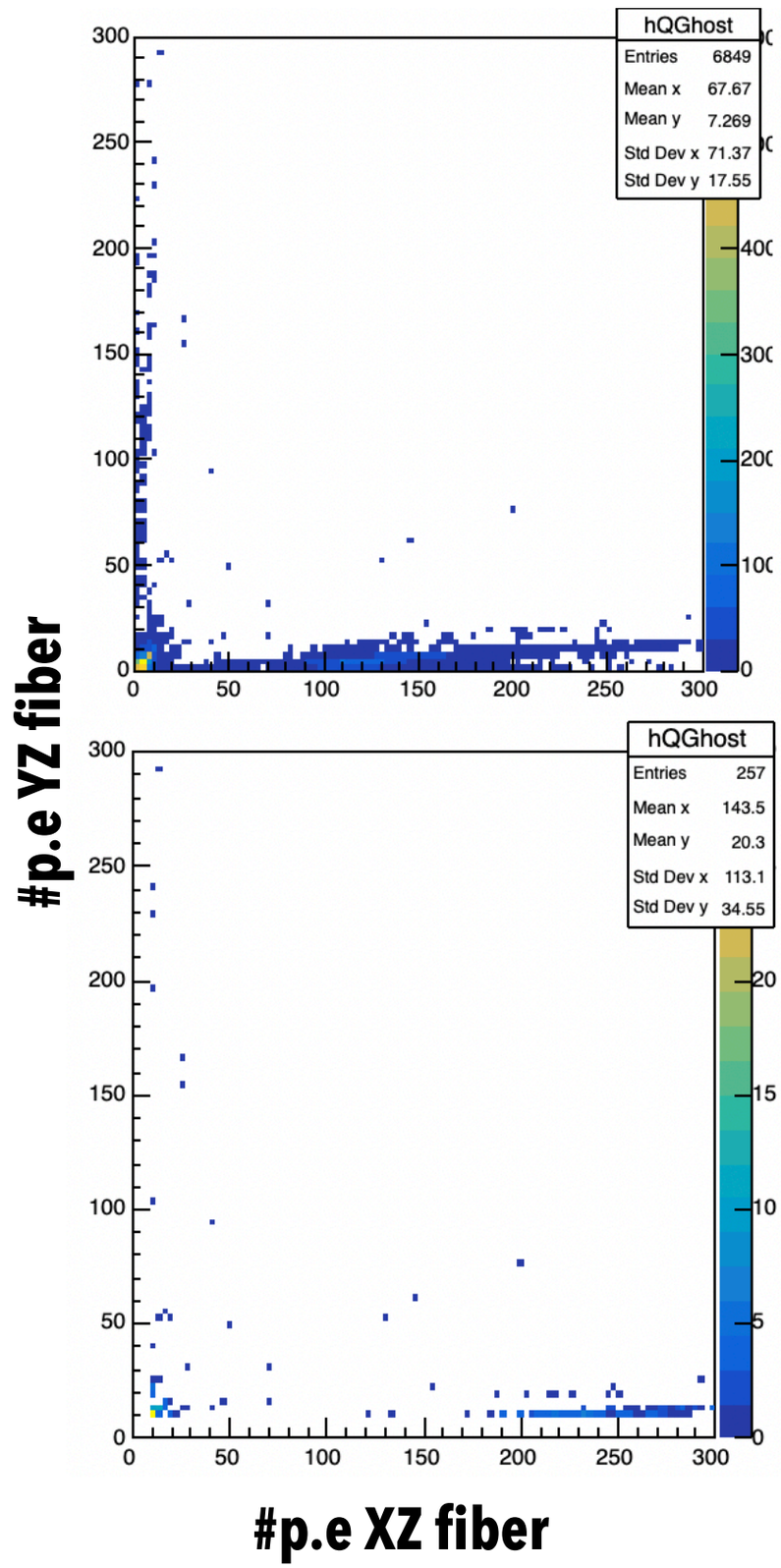
Spoiler: Ghost hits

3D CHARGE, GHOST HITS & CUTS

Distribution of Real hits



Distribution of ghost hits

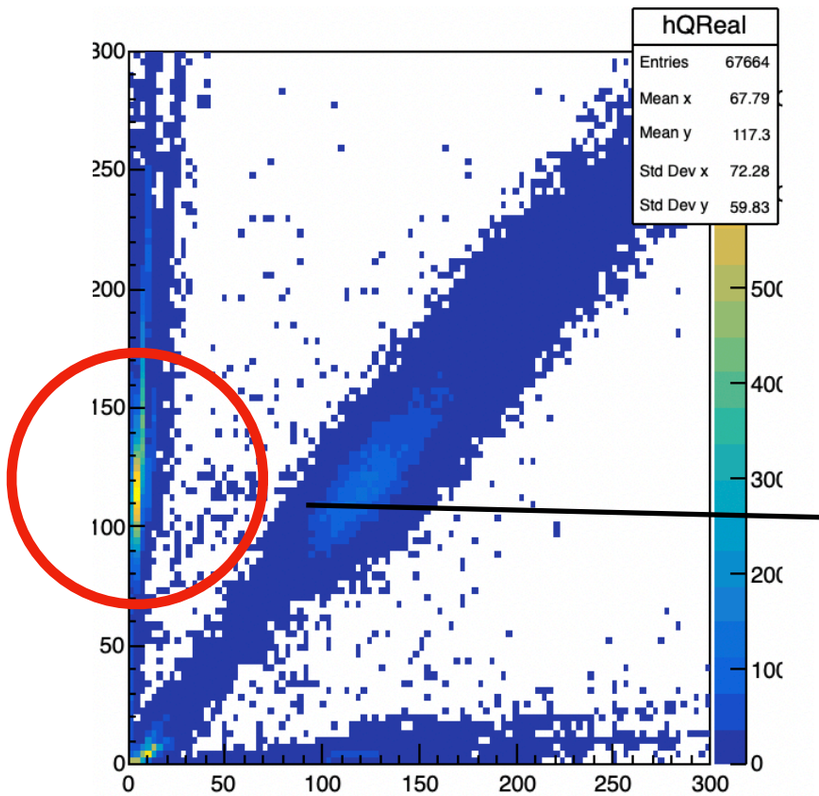


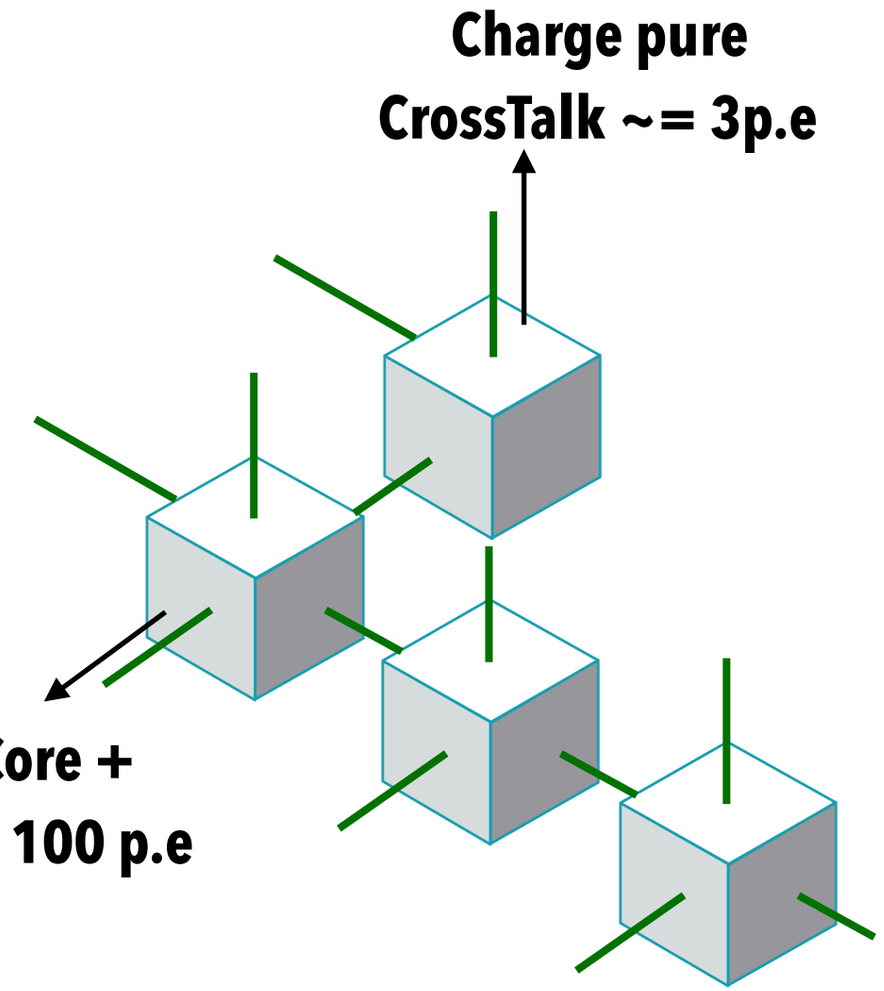
A **real** (**ghost**) hit is a cube **with** (**without**) a real energy deposit and that is intersected by 3 MPPC hits.

- Ghost hits are almost imposible to distinguish from real hits using geometric arguments.
- They are a dangerous nuisance for the 3D charge reconstruction.

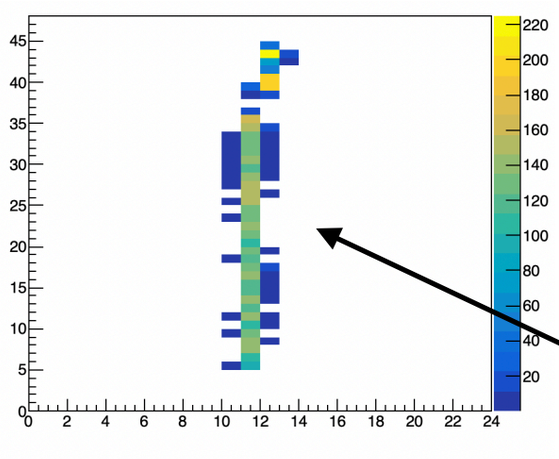
- The cut in charge (<10p.e) kills 96% of ghost hits.
- 0.9% of real Voxels lost with and average Q of 5p.e.

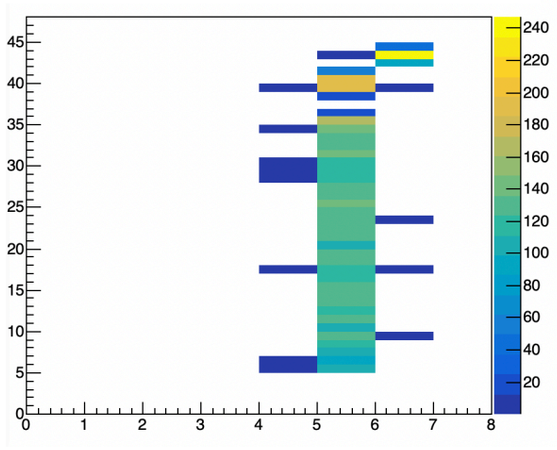
ABOUT THE ASYMMETRIC DISTRIBUTION





Charge is Core +
CrossTalk ~= 100 p.e

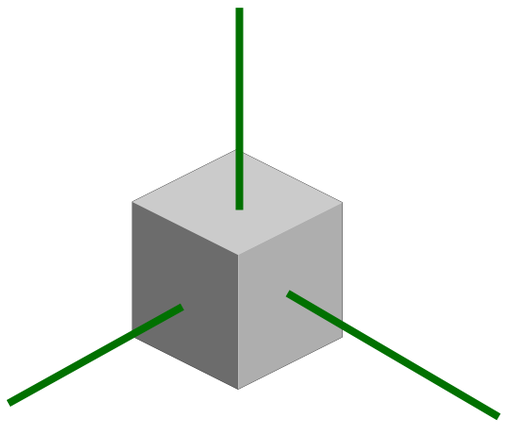




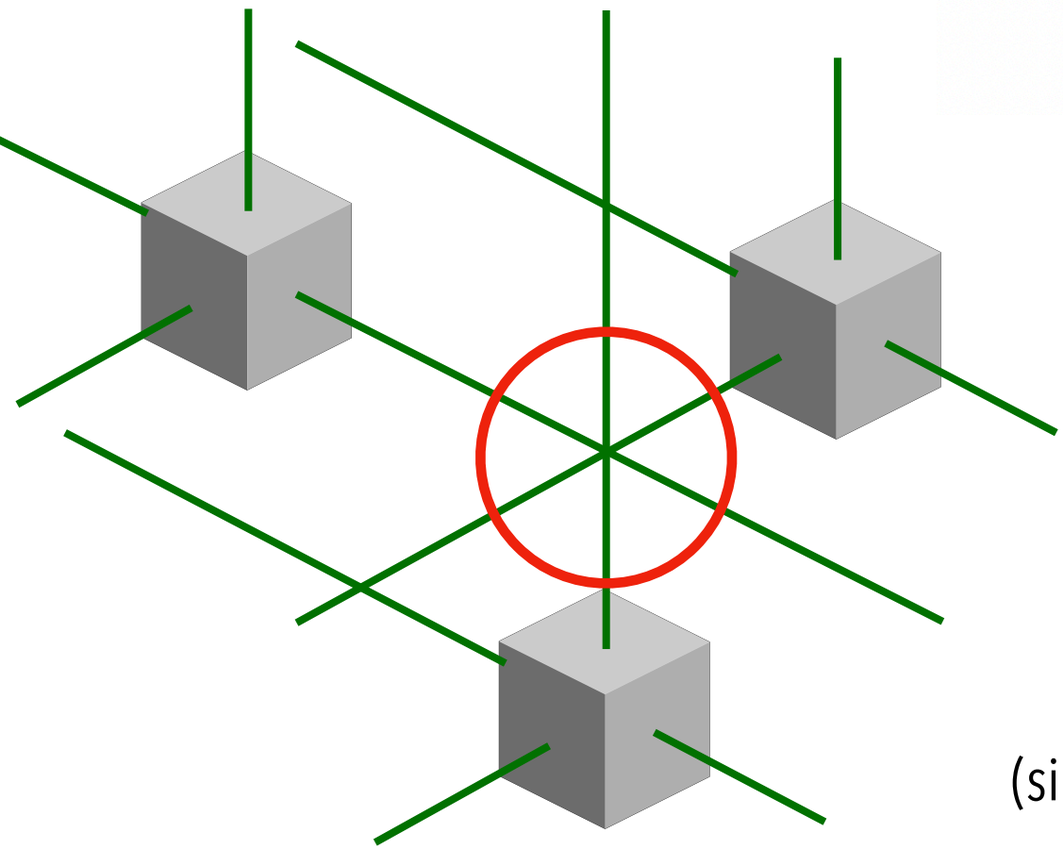
The plots is asymmetric because of the asymmetric crosstalk.

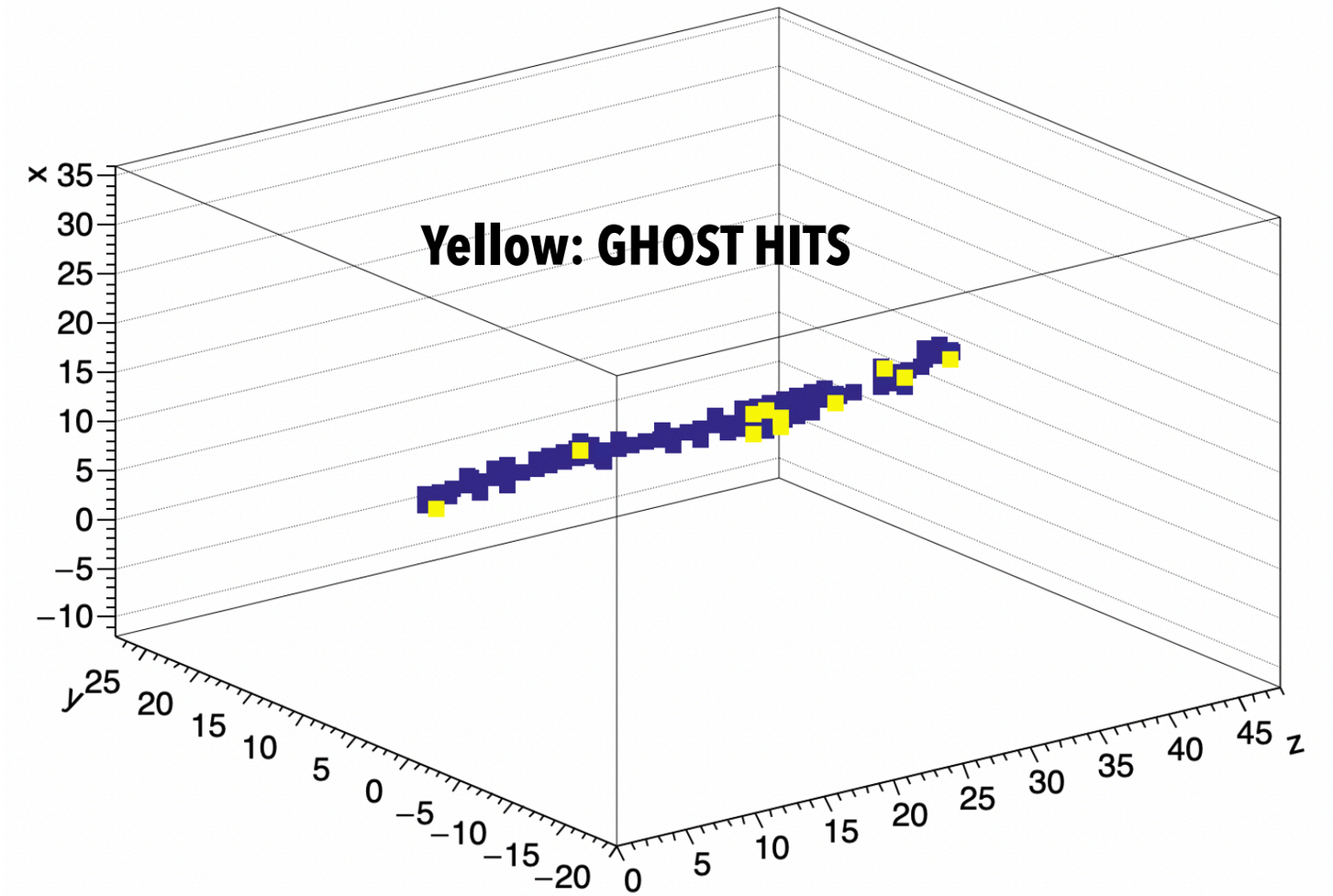
GHOST HITS

Real:



Ghost:





The algorithm match intersecting fibers with hit in the MPPC. Sometimes ghost hits appear.

→ This happens much more often when there is crosstalk. (Otherwise very straight track)

Future check ghosts with antiproton particle gun: (similar to neutrino interaction)