Bi207 - Analysis Framework & Charge Equalisation

Jordi Capó, on behalf of the Bi207 working group.

IFIC
INSTITUT DE FÍSICA CORPUSCULAR

DUNE

PyROOT framework

- The analysis framework is a **pyROOT** based framework designed in an **event-by-event** basis loop.
- **A ROOT conversion happens once** on a given set of dates/files, where event data (waveforms) are transcript into a TFile:
  - *Saves large amount of memory*.
  - *Speeds-up significantly the event loop* by opening just one file, instead of 4k json files.
- The content of the events is organised in a **hierarchical** manner, so one can access the **low-level data** from any point of the chain one is on.

ROOT converter

- It is a very basic script that takes all events from the json files and writes them into the ROOTfile.
- In order to be more efficient, all data are int() or float(), so dates and times have been transformed into int() accordingly.
  - 11:08:23 -> 110823 -> 6 characters -> Split by 2 and recover original timestamp.
  - 09:34:21 -> 93421 -> 5 characters -> Transform into string, add a 0 at the beginning -> Split by 2 and recover original timestamp.
- **It is not ready to accept ".bin" files as input** -> Still to be decided.

Event Handling: software workflow

Event()

The event gets each entry of the converted ROOTfile and creates de following objects according to the needs of each of the analyses. One can skip any of the classes if needed, in order to save computation time.

Cluster()

A cluster is a collection of spatially connected voxels. From the coordinates, using DBSCAN algorithm clusters are found. From clusters, one can obtain track angles, total track charge deposition, noise identification, centroid position …

Voxel()

A voxel is a real charge deposition in a certain region (determined by the pitches) of the TPC.

Information about total the charge deposition coordinates (from Hit() and Strip()) and charge deposition (from Hit())

Strip()

Each channel has a different equation that is used to find the intersection between any given pair.

The equations are defined according to the collection and channel pitches, imposing appropriate boundary conditions

Hit()

A hit is a physical hit on a strip.

Charge, real channel, hit time, channel type…

The idea is to use this class to add any needed information that the Peak() should not be aware of

Peak()

The information of a peak found in a waveform: peak position, integral value, peak height, SNR, FWHM…

Waveform()

RAW waveform information

Critical points: Peak Finder Algorithm

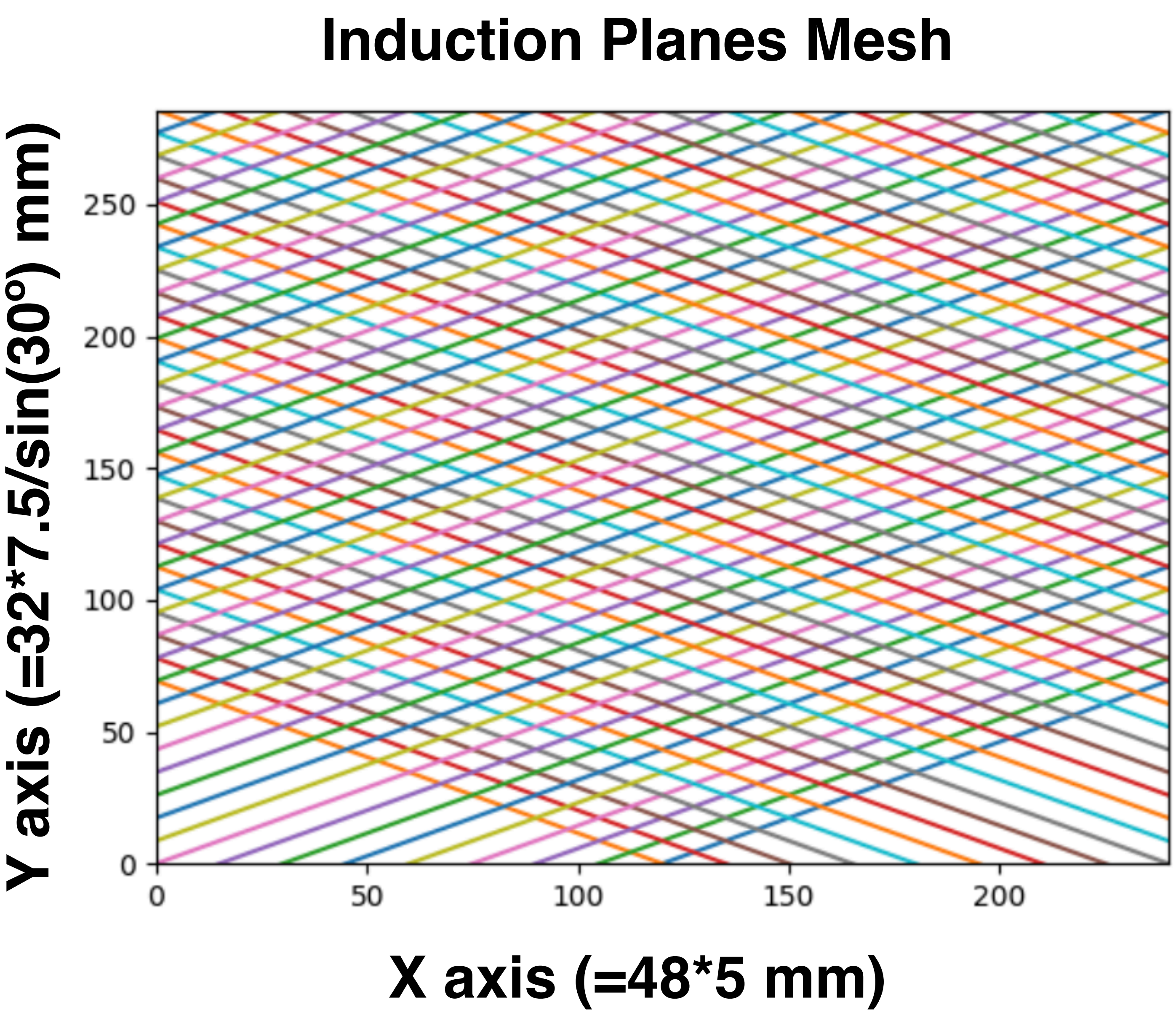
- **Peak()** : the peak finder algorithm is designed to only distinguish **peaks that are separated more than 20 time ticks** (this can be easily adjusted).
  - If two peaks are within 20 time ticks, then it is considered as a single Peak().
  - ***Collection Strips***: The startTime and endTime of the peak is computed from the peak maximum until 1.05xBaseline value is reached.
  - ***Induction Strips***: The startTime and endTime are calculated considering the bipolar shape of the signal. The peak time returned is the one corresponding to the intersection with the horizontal axis.

Critical points: Voxelisation

- **Hit times** for each of the planes are **compared with other planes** and grouped according to **best time match**. *Hits are grouped together according to minimum time difference coincidence.*
- A voxel can still be created if there is NOT Collection strip hit.
- To find the coordinates of a voxel, at least **2 hits in two different planes** are required. It is designed to manage the four possible situations, depending on the amount of hits: **(C,I1,X), (C,X,I2), (X,I1,I2), (C,I1,I2)**.
- For the (C,I1,I2) it **may happen** that *the intersection of a pair of hits does not match with the intersection of the other possible pairs*.
- If there is NOT triple coincidence, the voxel is not created -> This ambiguities may be managed by applying different weights to the different planes?
- Another important key point here is to have the **TPC dimensions well defined** so the reconstructed coordinates correspond to the real ones.

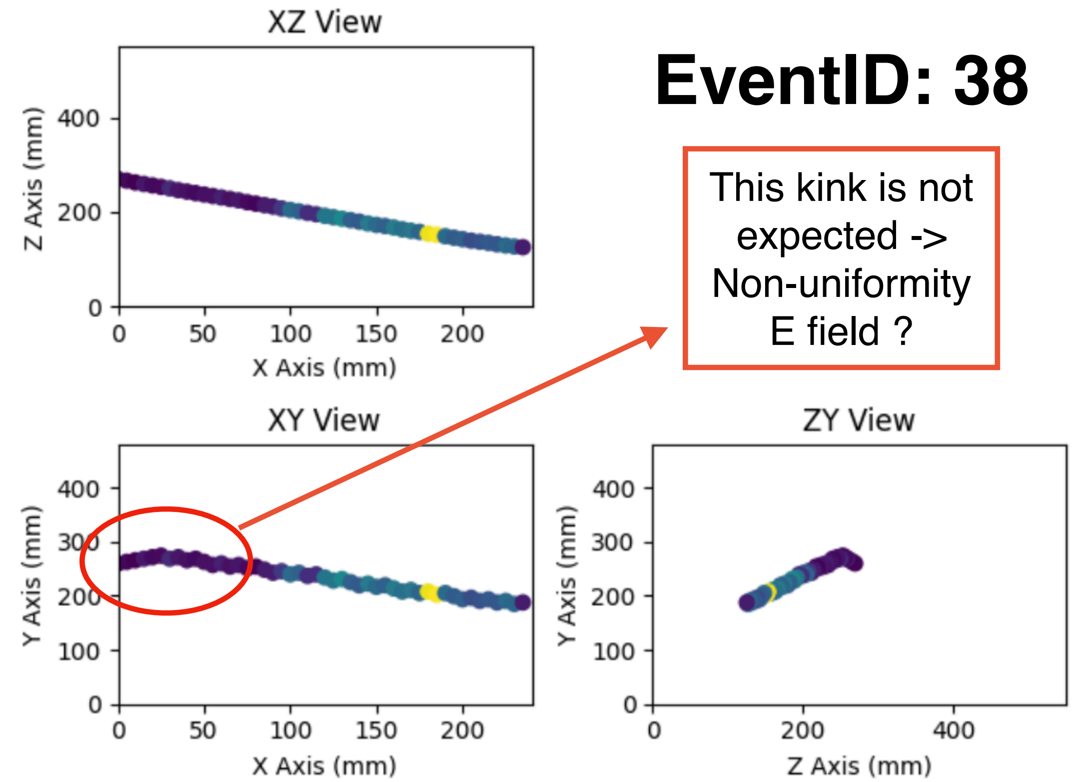
Equation Definition of Strips: Mesh()

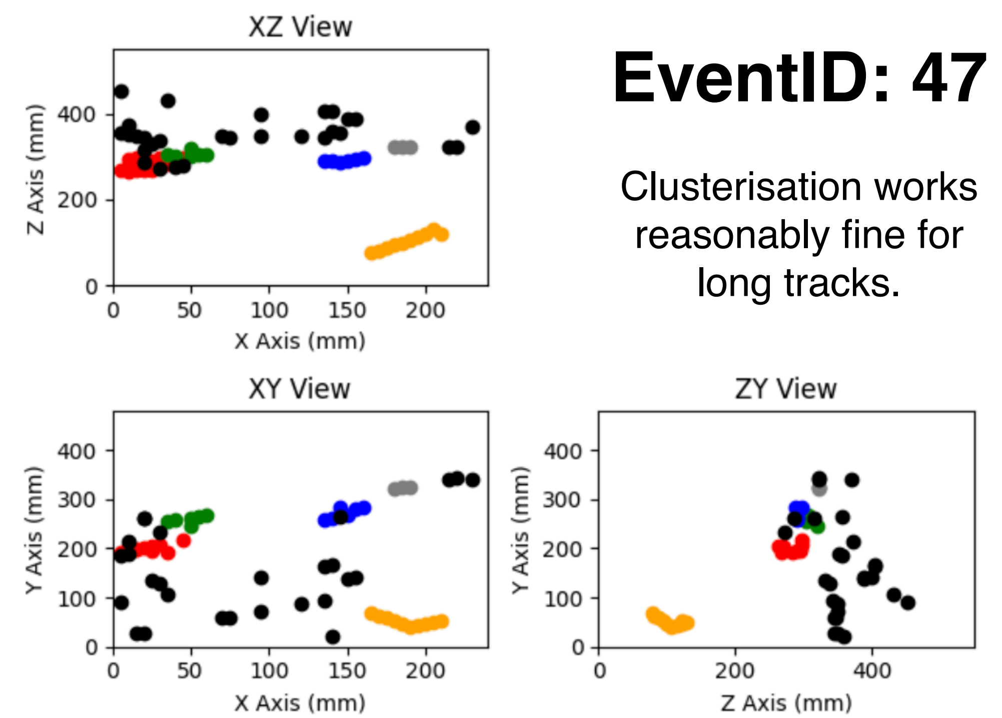
- There are two regions where the induction planes do not cross themselves.
- The collection plane is not represented. It is composed by 48 vertical lines spaced 5mm.
- X axis=240 mm; Y axis=480mm; Z axis=500mm (not really important)



Critical points: Clusterisation

- **DBSCAN** is a powerful AI algorithm that allows to find clusters given two simple parameters (at first approximation):
  - **EPS**: “max distance between adjacent voxels”
  - **N_voxels**: “min number of voxels to create a cluster”
- It returns a dictionary classifying the clusters as NOISE=“-1” or SIGNAL=“natural number”.
- It can only be designed in order to find single-voxel clusters.





Charge Equalisation Correction

- Select almost fully contained muons crossing the entire TPC. **Selection criteria**:
  - *NOISE clusters removed*
  - *Number of voxels per cluster > 45*
  - *Straight tracks only*
- **Effective track pitch**:
  - *It can be in principle calculated using track angle information, and projecting it on the collection strips.*
  - It is actually calculated using the formula:

$$\Delta L = \sqrt{(\Delta Z)^2 + \frac{4}{3}[(\Delta C)^2 + (\Delta I)^2 - \Delta I \Delta C]}$$

$$L_{projected} = \frac{\Delta L}{N_{Collection\ strips}}$$

$$\Delta Z = N_{time\ ticks} \cdot \delta t \cdot v_d$$

$$\Delta C = N_{Collection\ strips} \cdot \delta_C$$

$$\Delta I = N_{Induction\ strips} \cdot \delta I$$

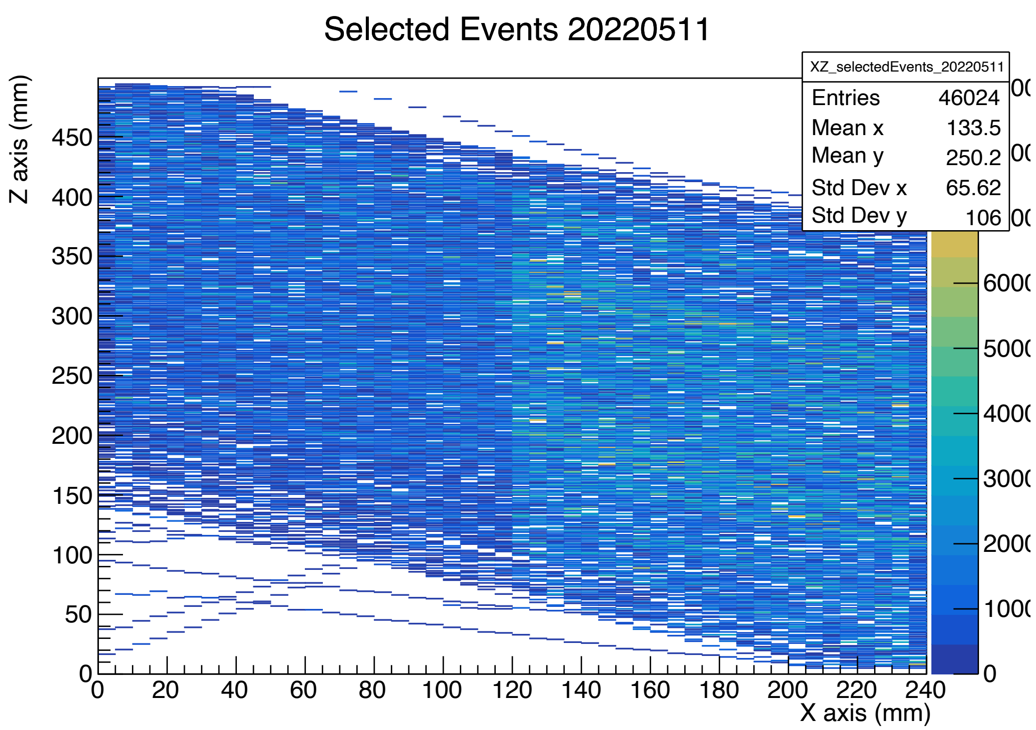
$$v_d = 1.55\ mm/\mu s$$

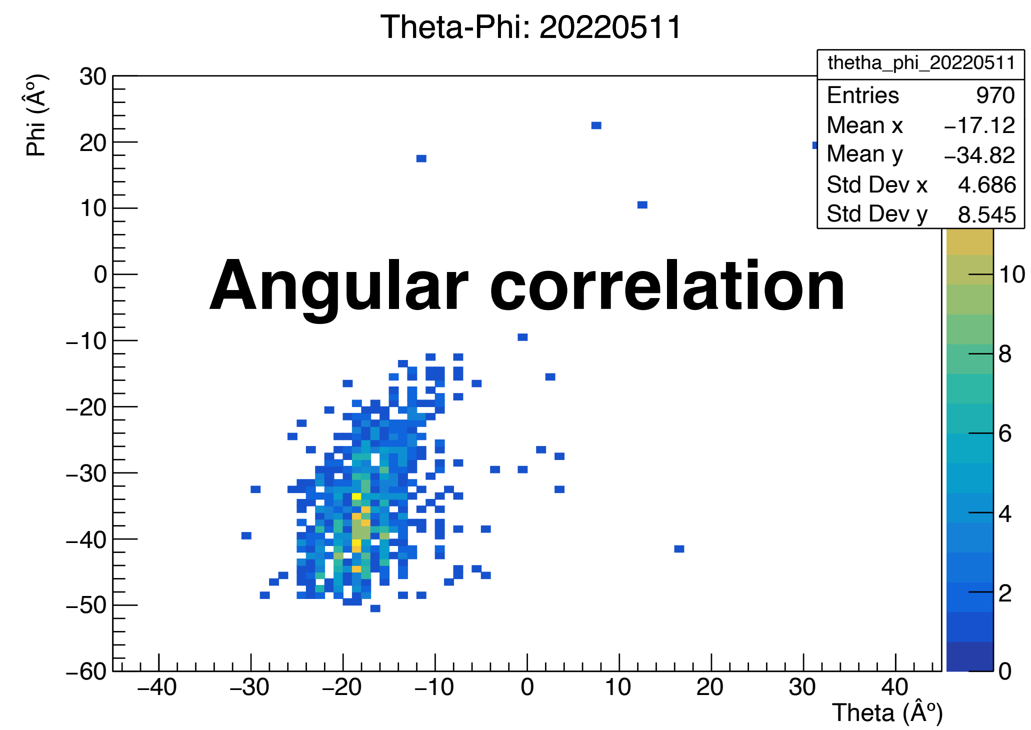
$$\delta_C = 5\ mm$$

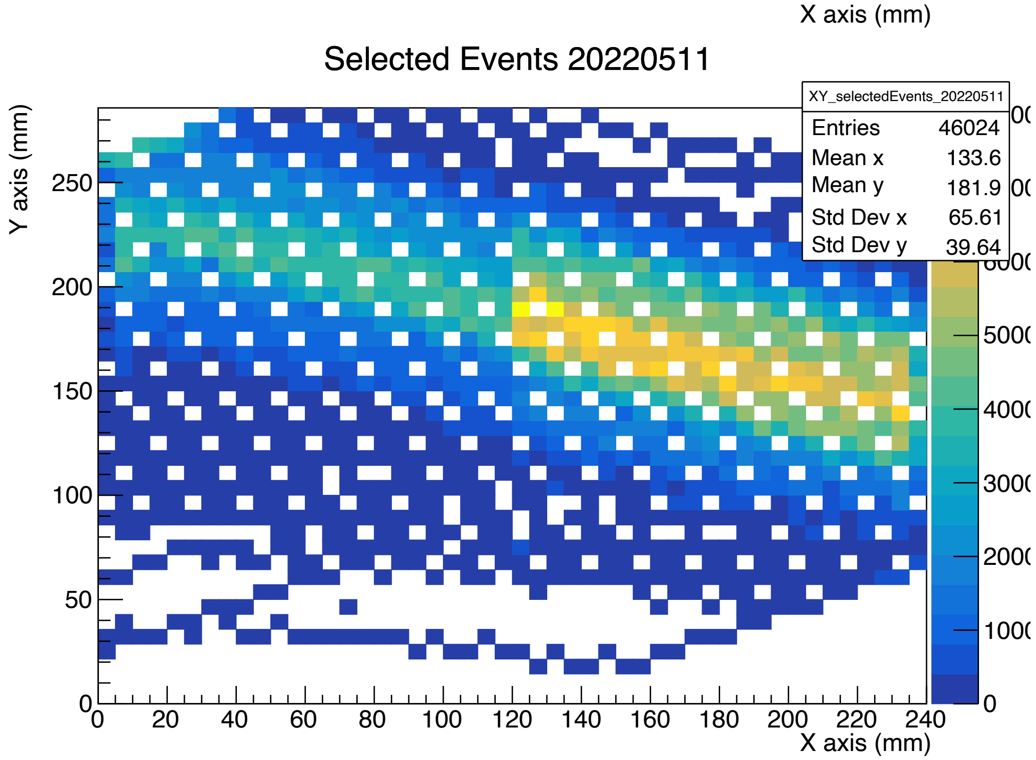
$$\delta_I = 7.5\ mm$$

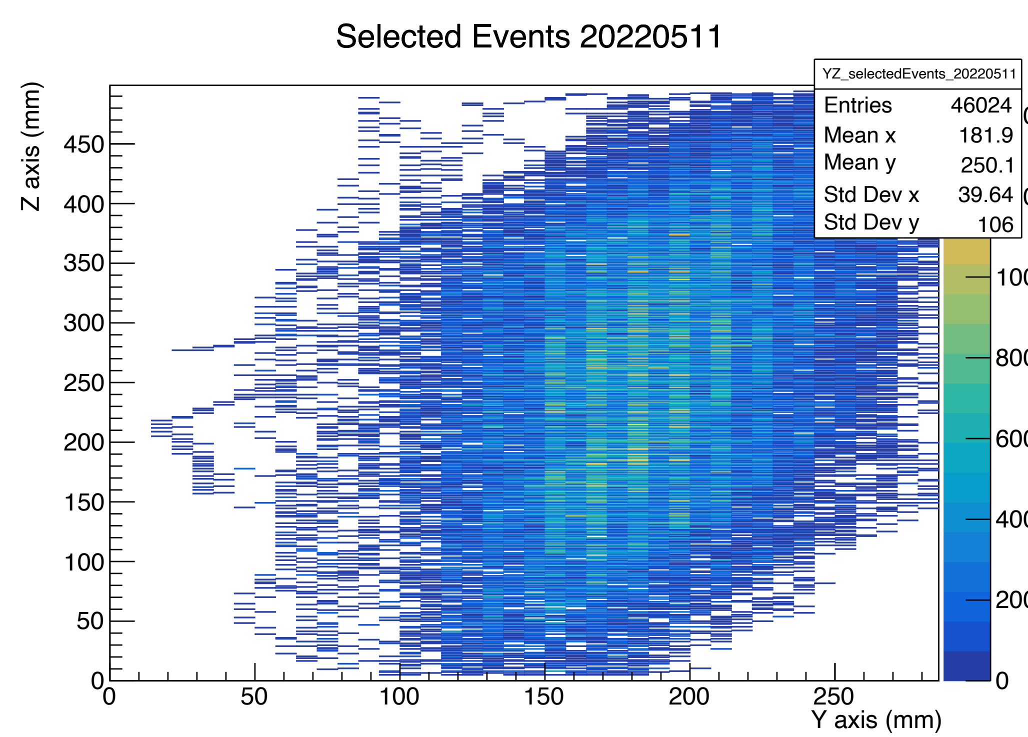
$$\delta_t = 0.5\ \mu s$$

Selected Events: overview



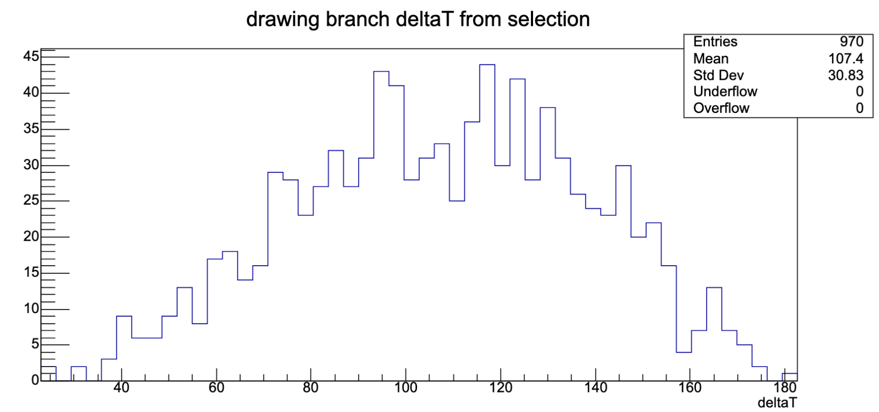


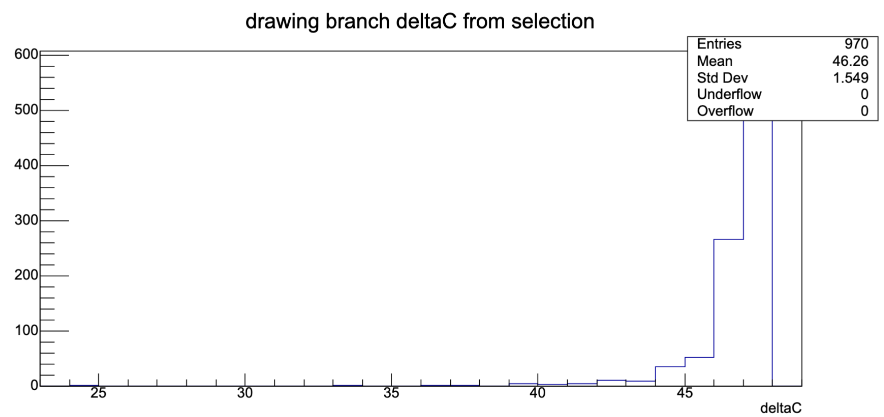


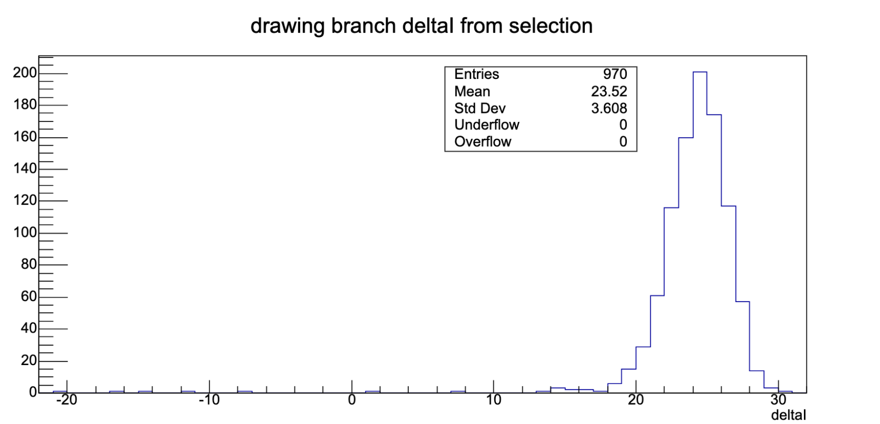


Selection Variables

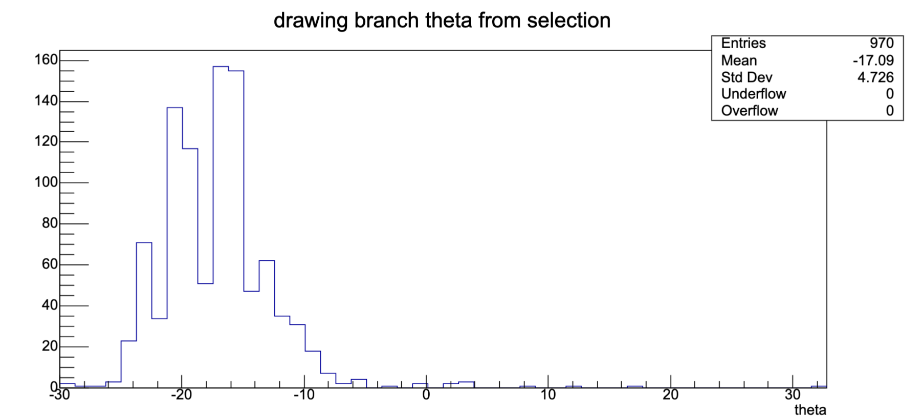
Non-reconstruction Variables

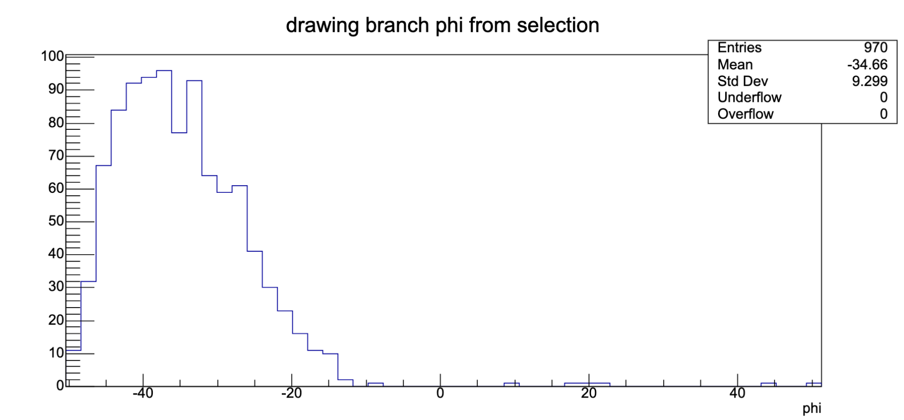


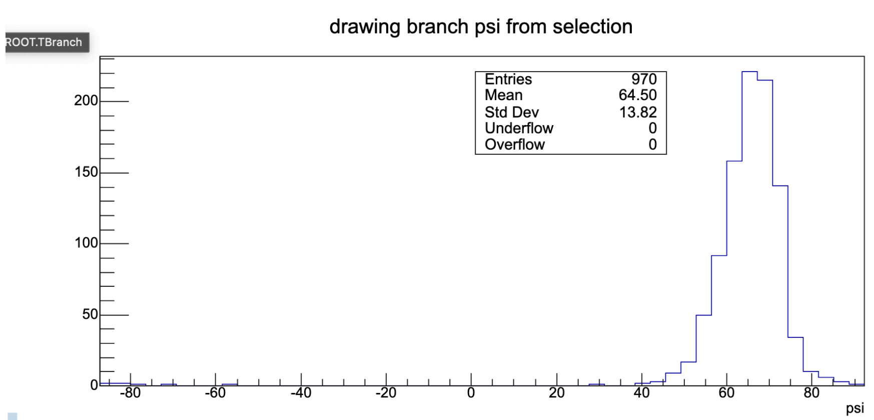




Reconstruction Variables

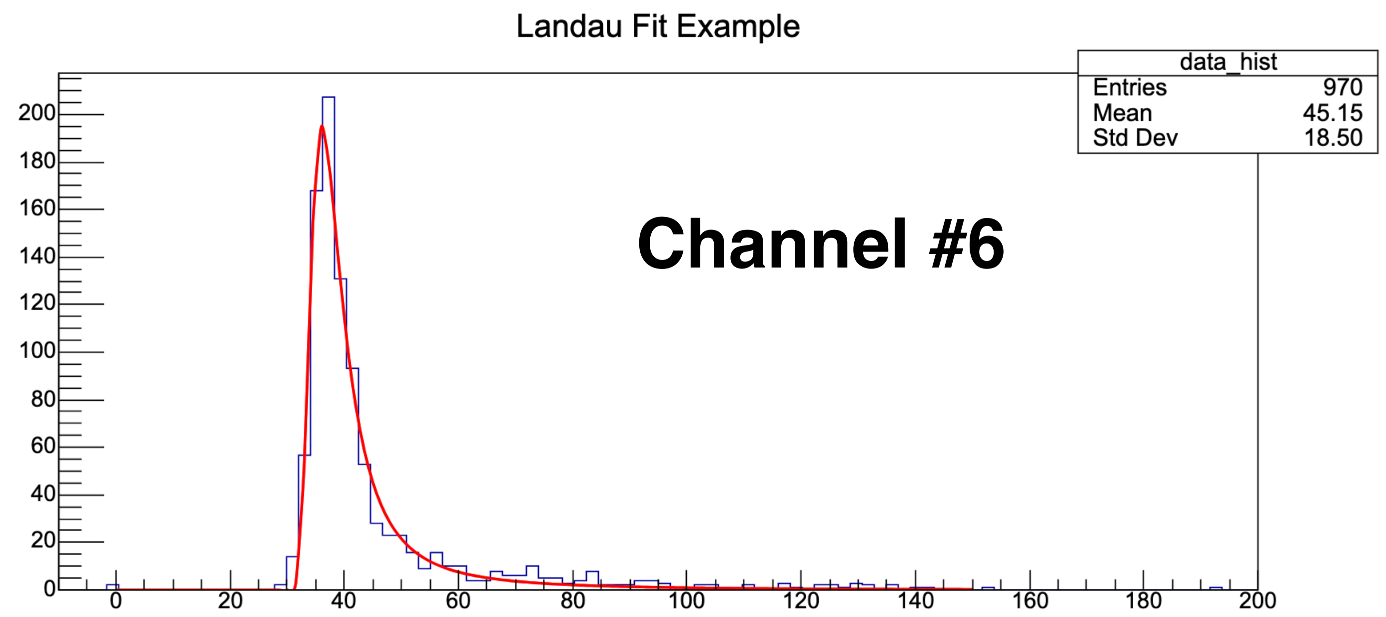


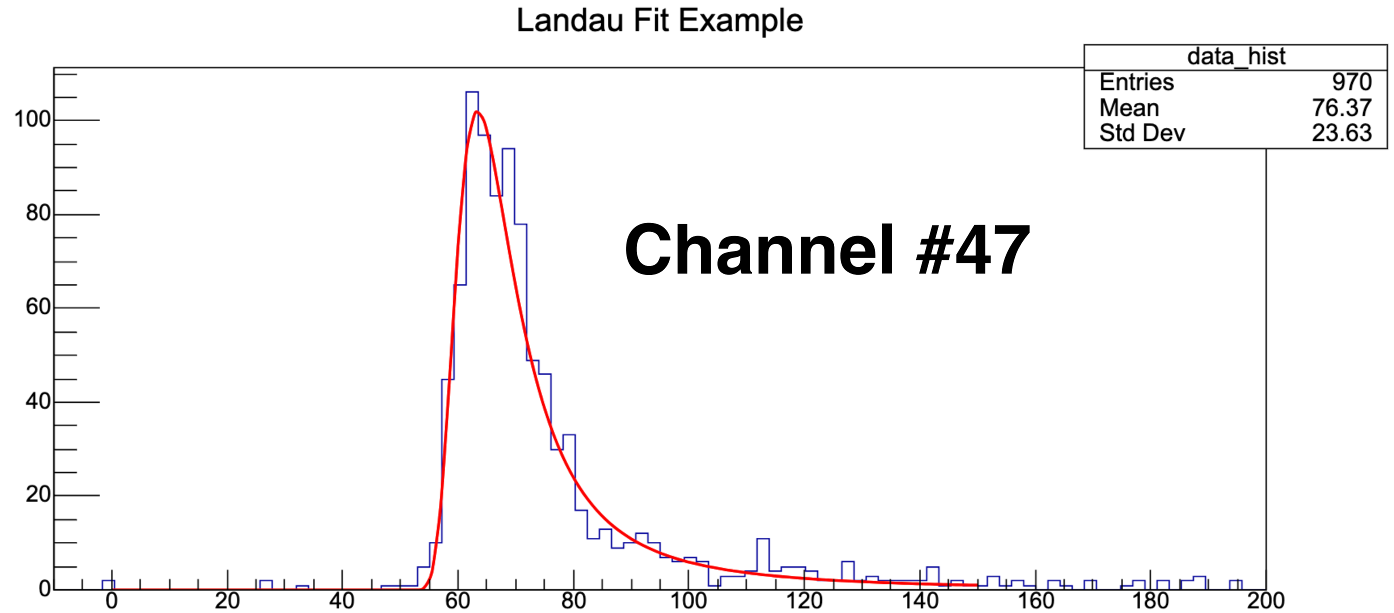




Landau Fits

- Charge from all hits on the collection plane is **stored independently on every channel**.
- **Landau Fitting** is performed using ***iMinuit***, from pyROOT.
- Histograms are fitted to **ONLY Landau distribution.** To add convolutions with gaussians/ exponentials does not improve really significantly the performance (*keep trying*).
- Starting/Final channels show different distributions, in terms of mean charge collection, than central ones.

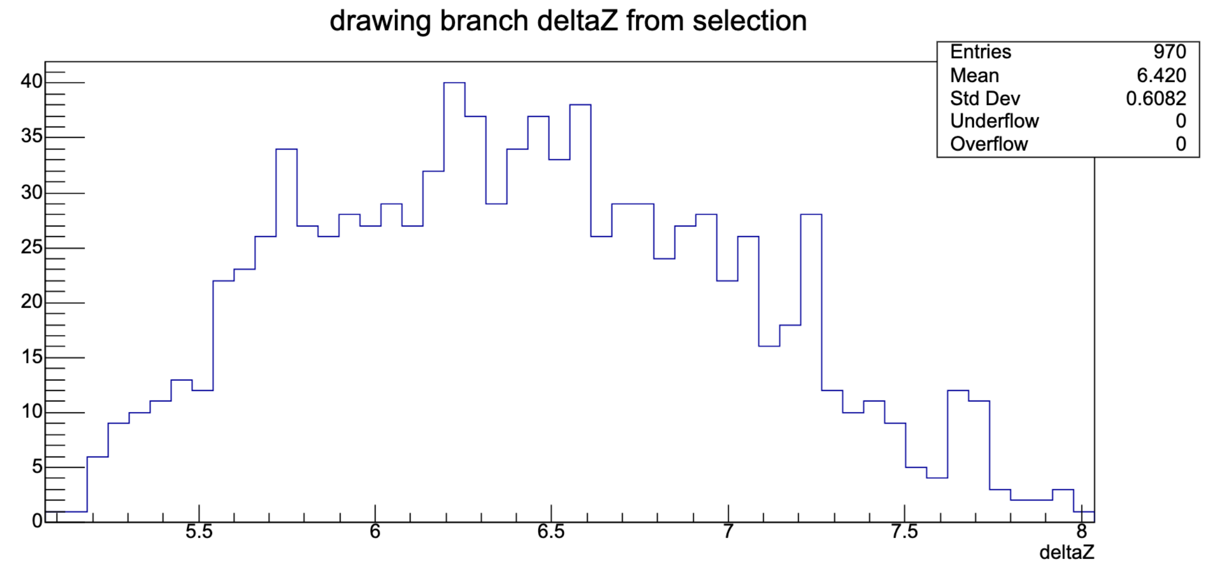




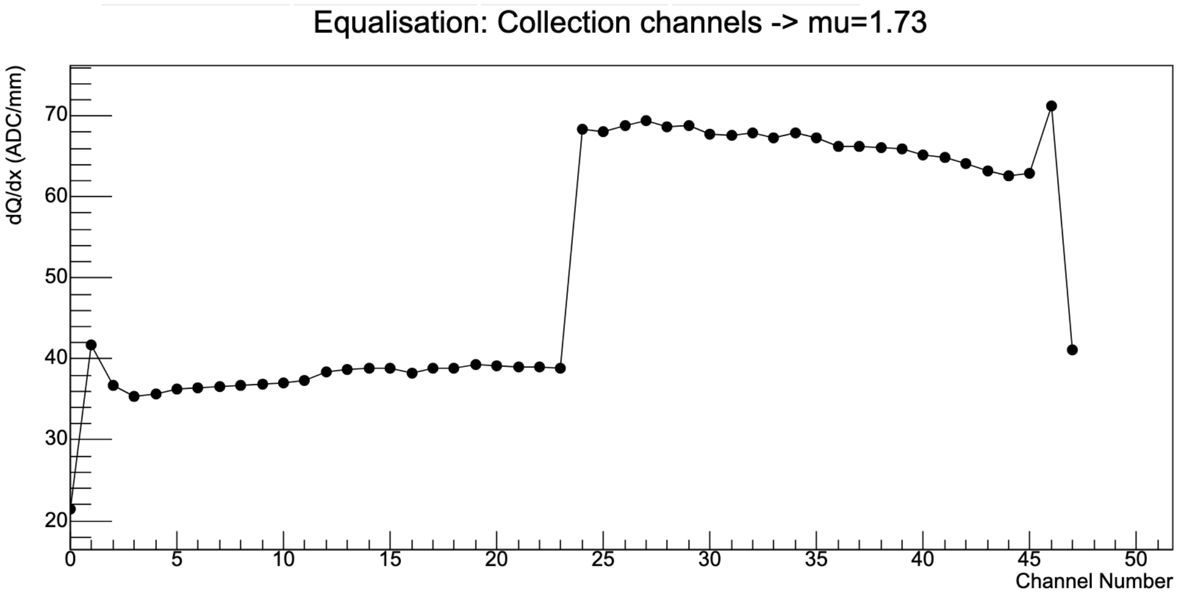
Equalisation Results

- The effective track pitch length **agrees** with the previously presented values (*Furkan Dolek's talk*).
- There might be **non-uniformities** of the **electric field** close to the border of the TPC that *causes charge that should deposit on first/ last channel* to be deposited to the *neighbour* one, or to *escape the readout*. (?)
- This is specially significant for the day (**20220511**). If data from other days is added, this effect relaxes indicating it may be a single-day issue.

Effective Track Pitch



Equalisation

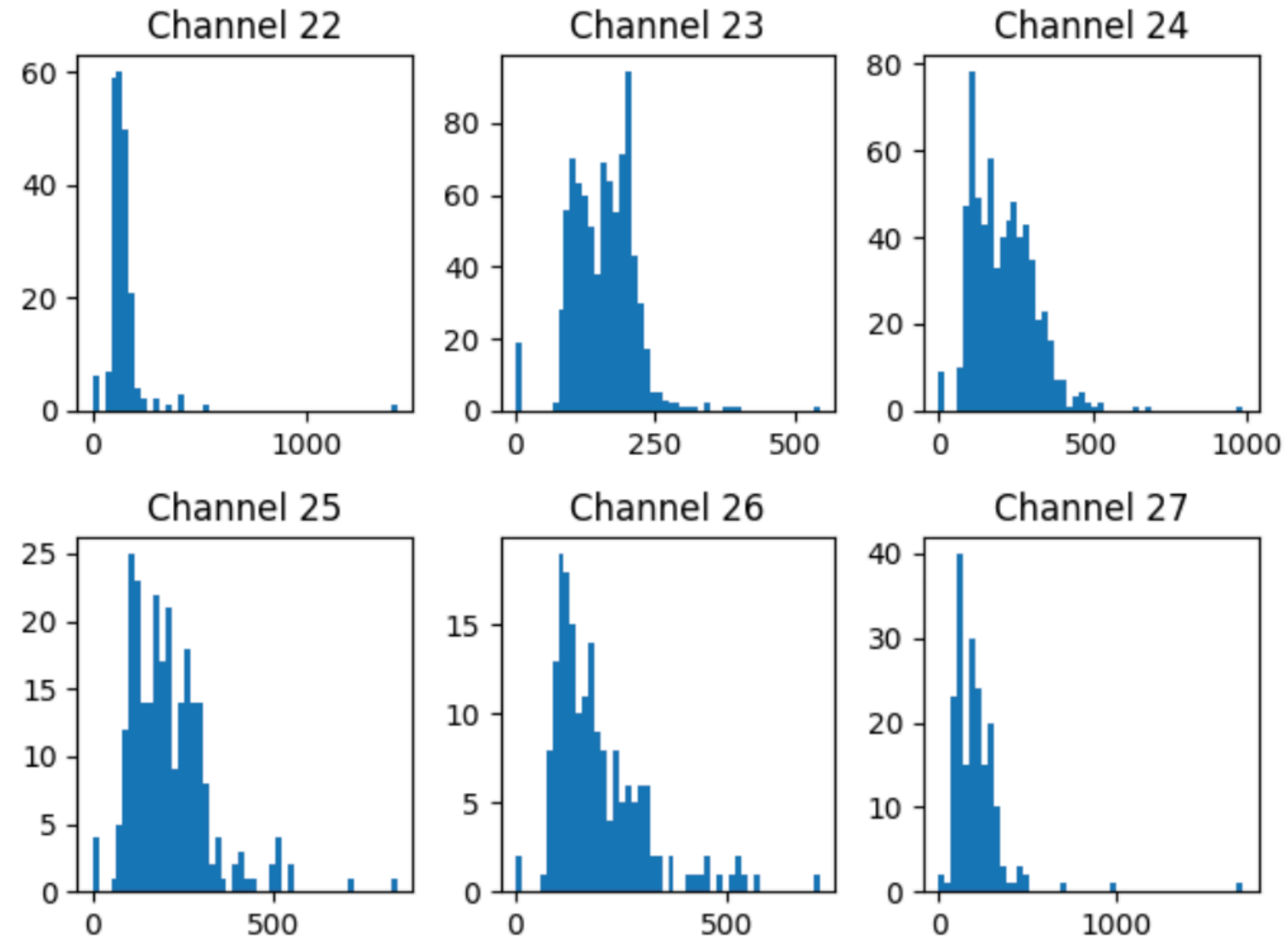


Source Events Selection

- It is straightforward to just select clusters that are **located at the expected coordinates**, i.e. where the sources are.
- One can filter by the **cluster size,** go through the desired coordinate region and perform the desired operation.
- These results show a single-voxel clusters selection, independently of their position, and collected charge stored for each channel.

PRELIMINARY

Performance on Source Selection



Next Steps

- Understand where the **small kink** in the reconstruction of the first channels comes from.
- Present the **equalisation results** with more days and improve the data fitting.
- Once done, **apply the equalisation correction** at the Hit() level.
- Calculate the **effective track pitch** by using the reconstructed length of the track, and projecting it onto the collection plane -> Compare the two methods.
- The same can be done for the other channels, although the selection criteria would be different as comics are highly parallel to the "induction2" plane.
- …