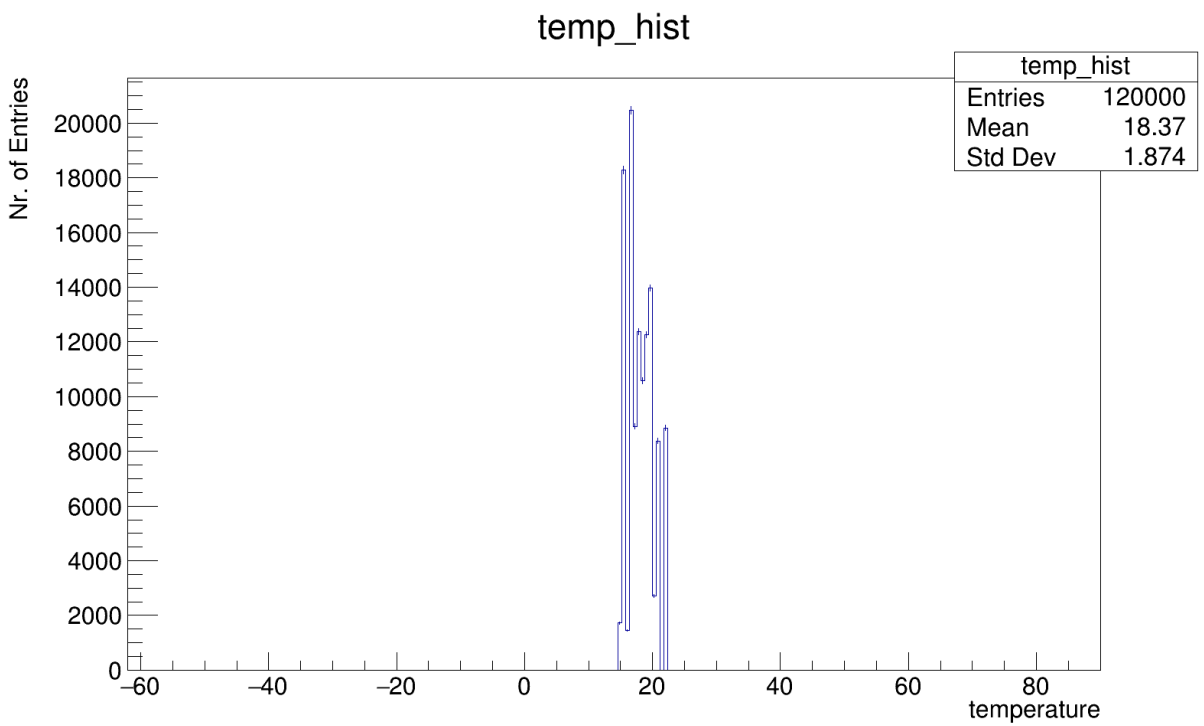
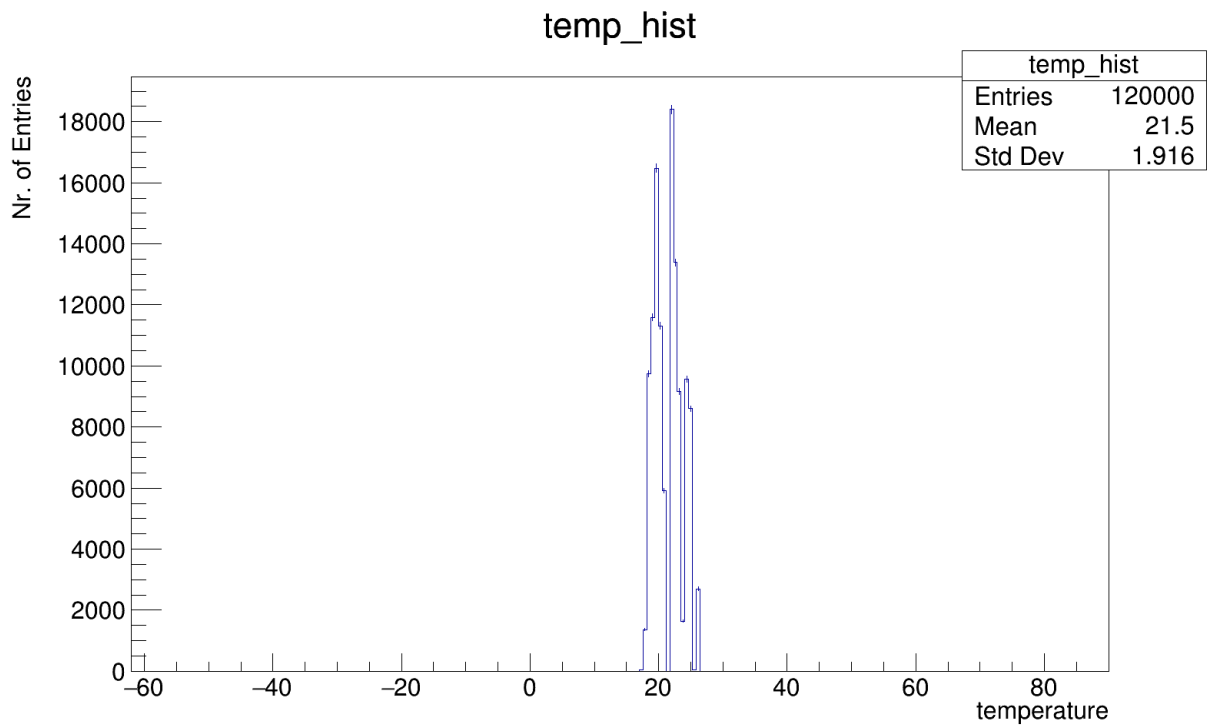
Temperature data

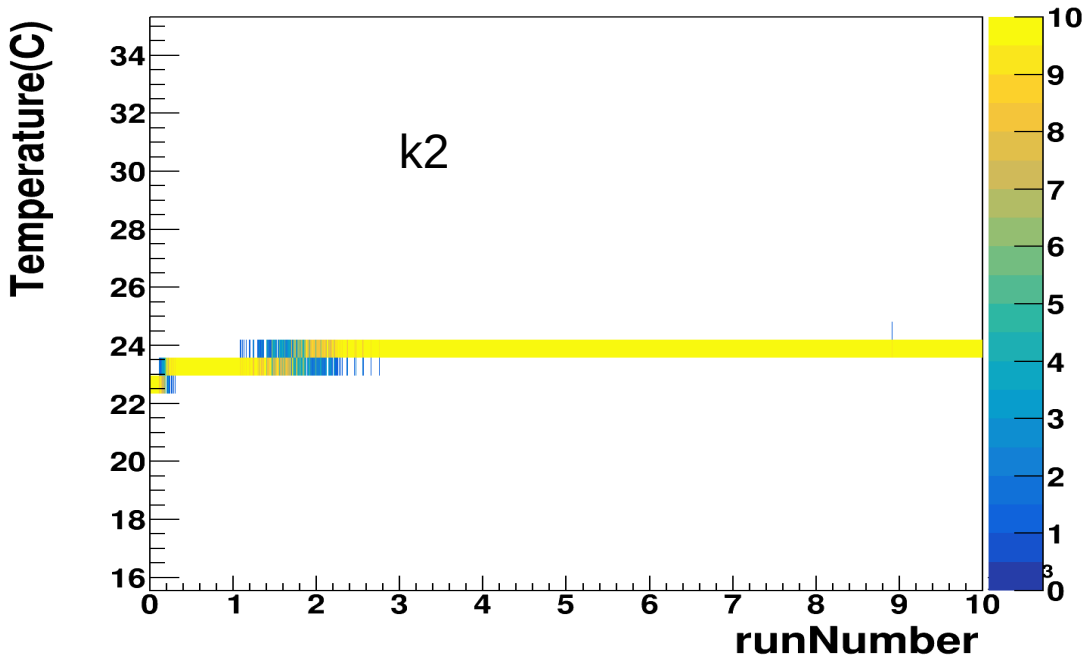
- Below is the temperature data from twelve KpiX from two different data sets (from different days).
- Overall there are little changes in temperature between acquisitions. A few degrees at most.
- These measurements were done with software acquisition with no run limit i.e. active most of the time.
- This variation appears to be *mostly* KpiX related

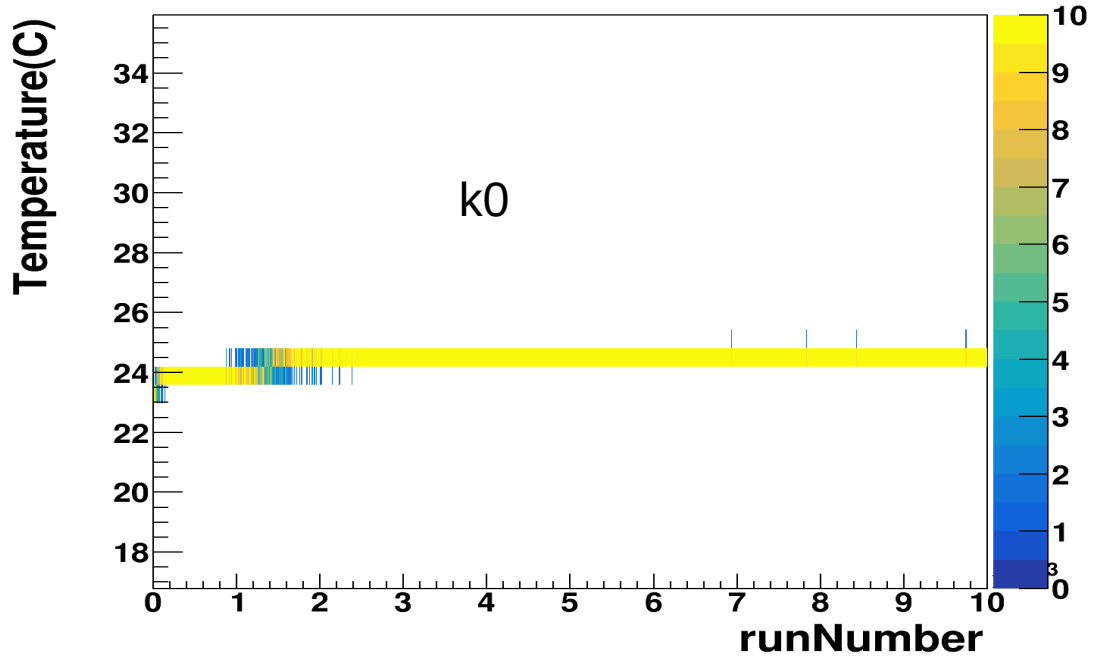




Temperature data

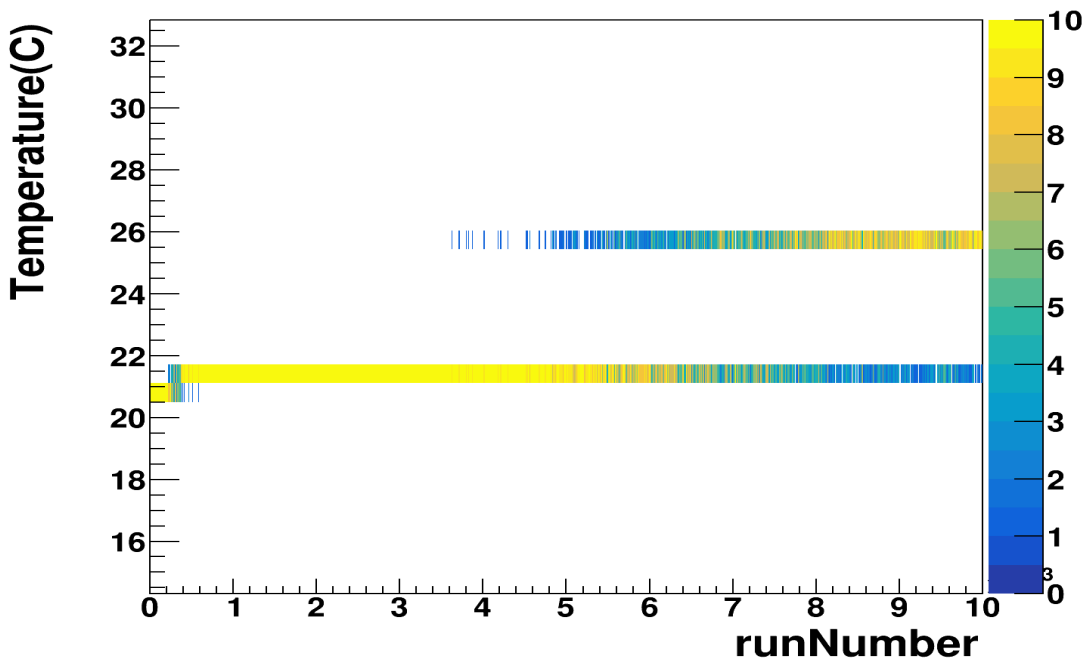
- When checking the temperature versus the runNumber (equivalent to time since start of run) we do see a slight heating up of the KpiX.
- Unfortunately as temperature data is taken at the start (I think) of the entire acquisition I cannot tell if there is some heating up during the acquisition itself.

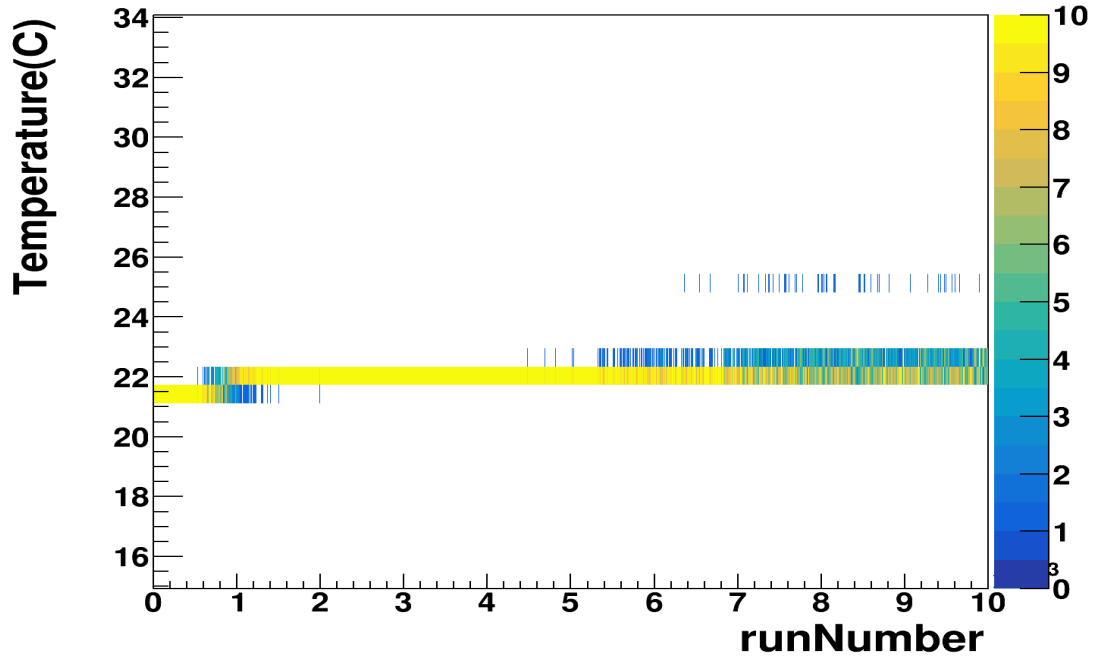




Temperature data (an outlier)

- Almost all KpiX show a similar behavior (see backup)
- K9 and K10 have some weird switch in the temperature
- To me seems mostly related to an involuntary bit flip as otherwise I cannot explain this jump in temperature.



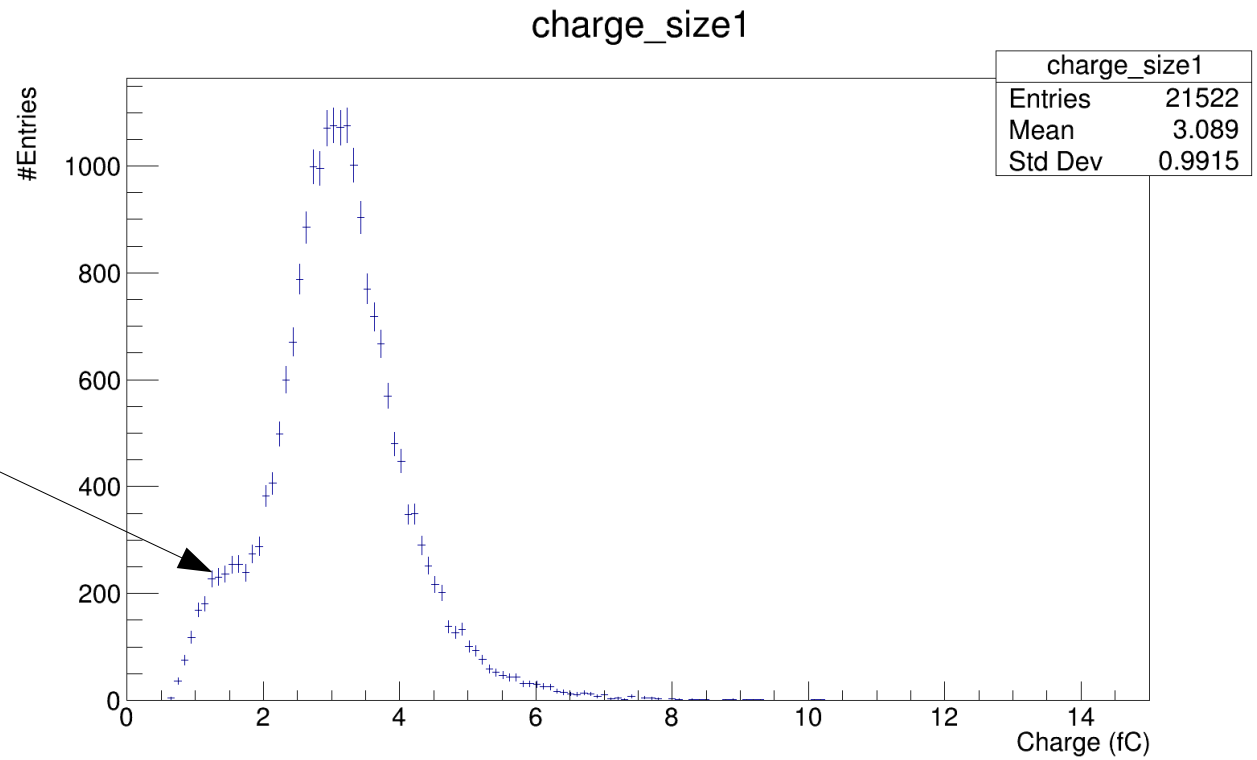


Investigating charge sharing and loss to backplane

- We previously determined whether something is a floating hit or a readout hit by the cluster size.
- While roughly true this is severely influenced by the clustering algorithm and cut values can result in 2 hit clusters being identified as 1 hit clusters and vice versa.

- Weird bump at low values.
- Either:
  - 2 strip clusters where one strip was lost to the S/N cut
- Or:
  - Incorrect cluster splitting splits charge between adjecent clusters incorrectly

(explanation at the end)



Investigating charge sharing and loss to backplane

- We previously determined whether something is a floating hit or a readout hit by the cluster size.
- While roughly true this is severely influenced by the clustering algorithm and cut values can result in 2 hit clusters being identified as 1 hit clusters and vice versa.
- Partially inspired by Dieter I decided to determine the hit position not by cluster size but by the projected track position.
- We decided to use what I call **golden hits**. Namely that tracks were generated with the Azalea telescope (6 pixel planes) and searched for hits on the Lycoris planes that matches these tracks.
  - Unbiased from any internal configurations.
  - Higher position precision than using Lycoris (2 to 3 micron)

Investigating charge sharing and loss to backplane

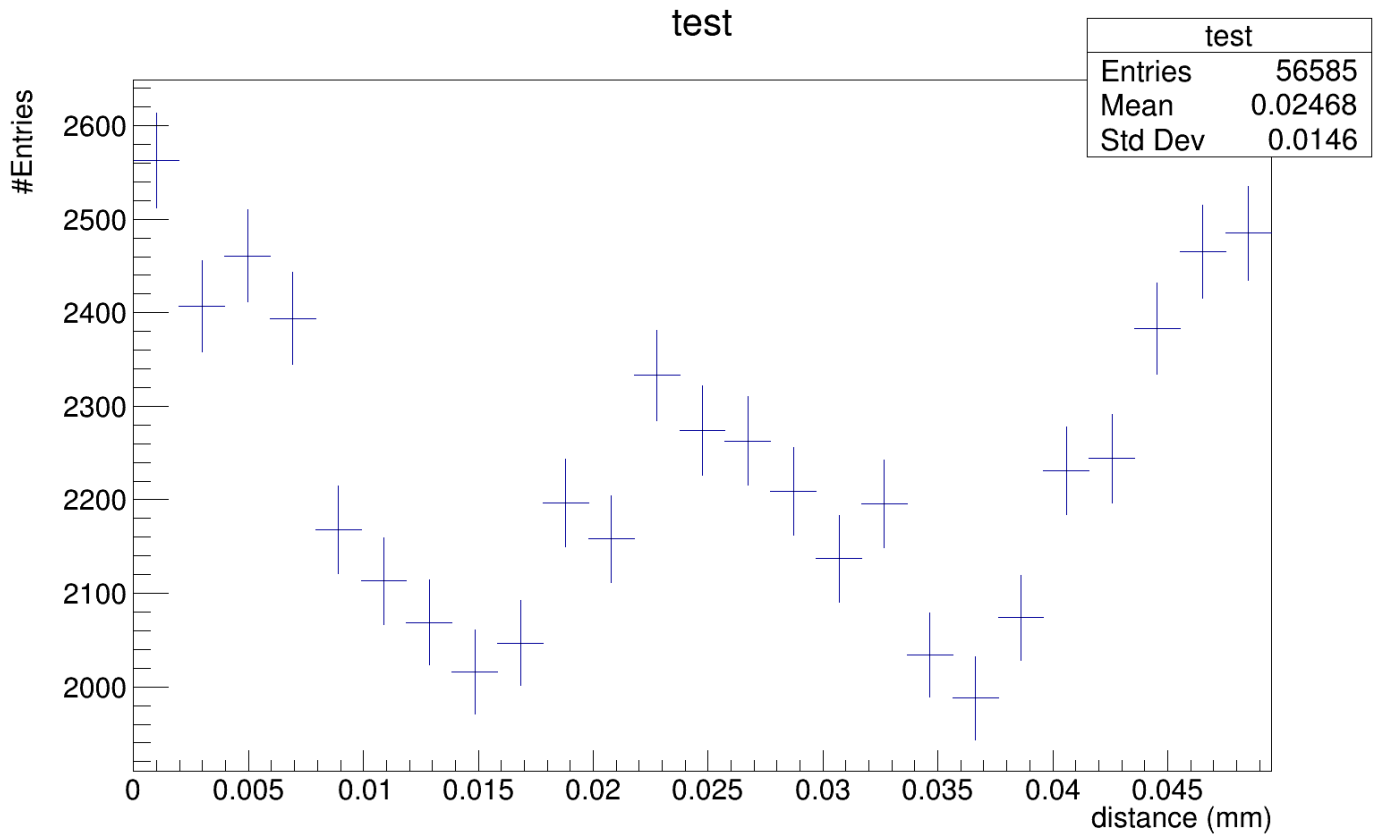
- First checked whether the previous assumption of:
  - 2 strip cluster = floating hit
  - 1 strip cluster = readout hit

  is accurate
- For this I took the track hit position. Transformed it into the local coordinate system and use
  - Projected_Position modulo 50 micron
- I would expect that this gives me floating strip hits when the values are at around 25 micron and readout hits when the values are around 0 and 50 micron.

Investigating charge sharing and loss to backplane

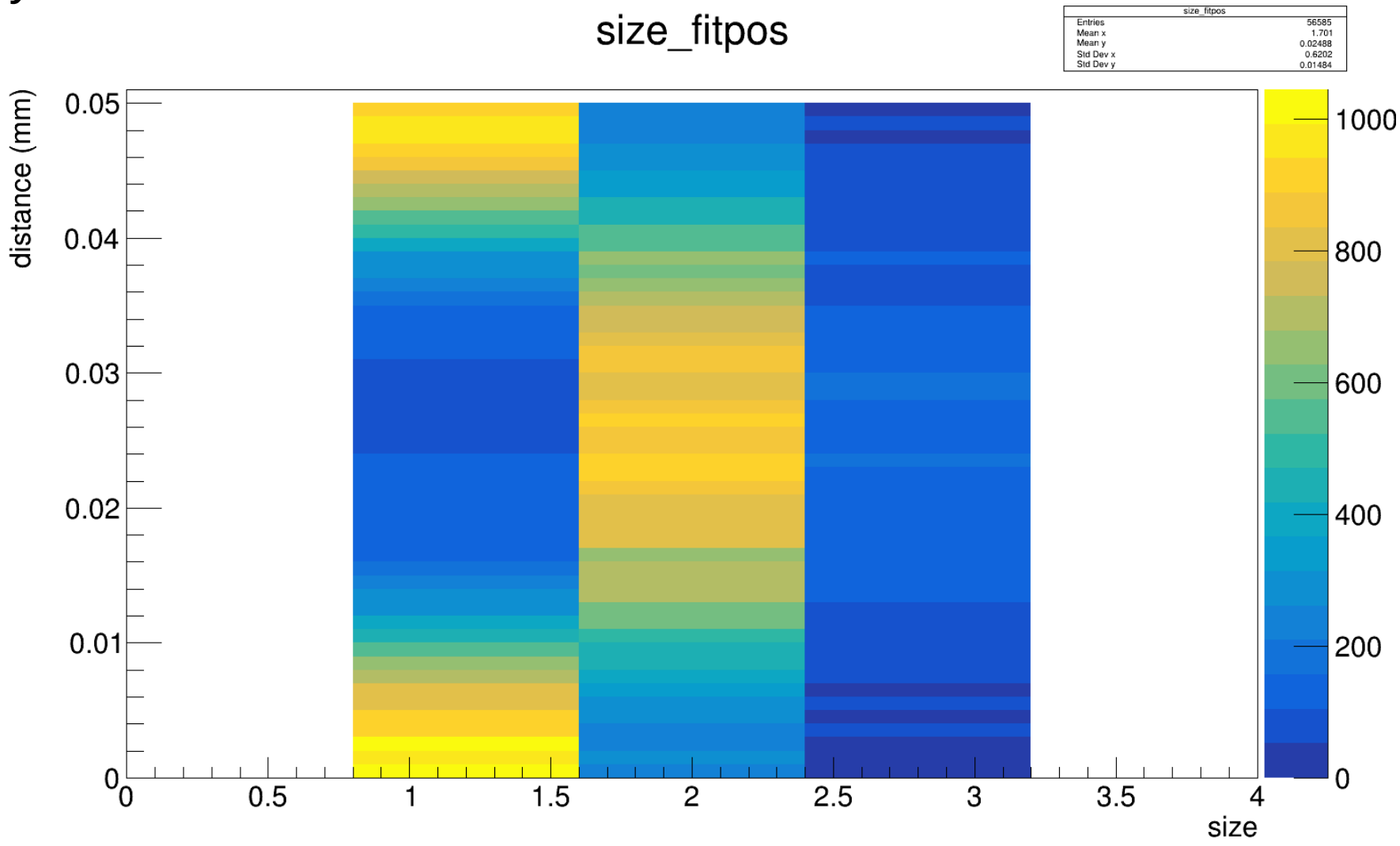
- First checked whether the previous assumption of:
  - 2 strip cluster = floating hit
  - 1 strip cluster = readout hit

  is accurate
- For this I took the track hit position. Transformed it into the local coordinate system and use
  - Projected_Position modulo 50 micron
- I would expect that this gives me floating strip hits when the values are at around 25 micron and readout hits when the values are around 0 and 50 micron.



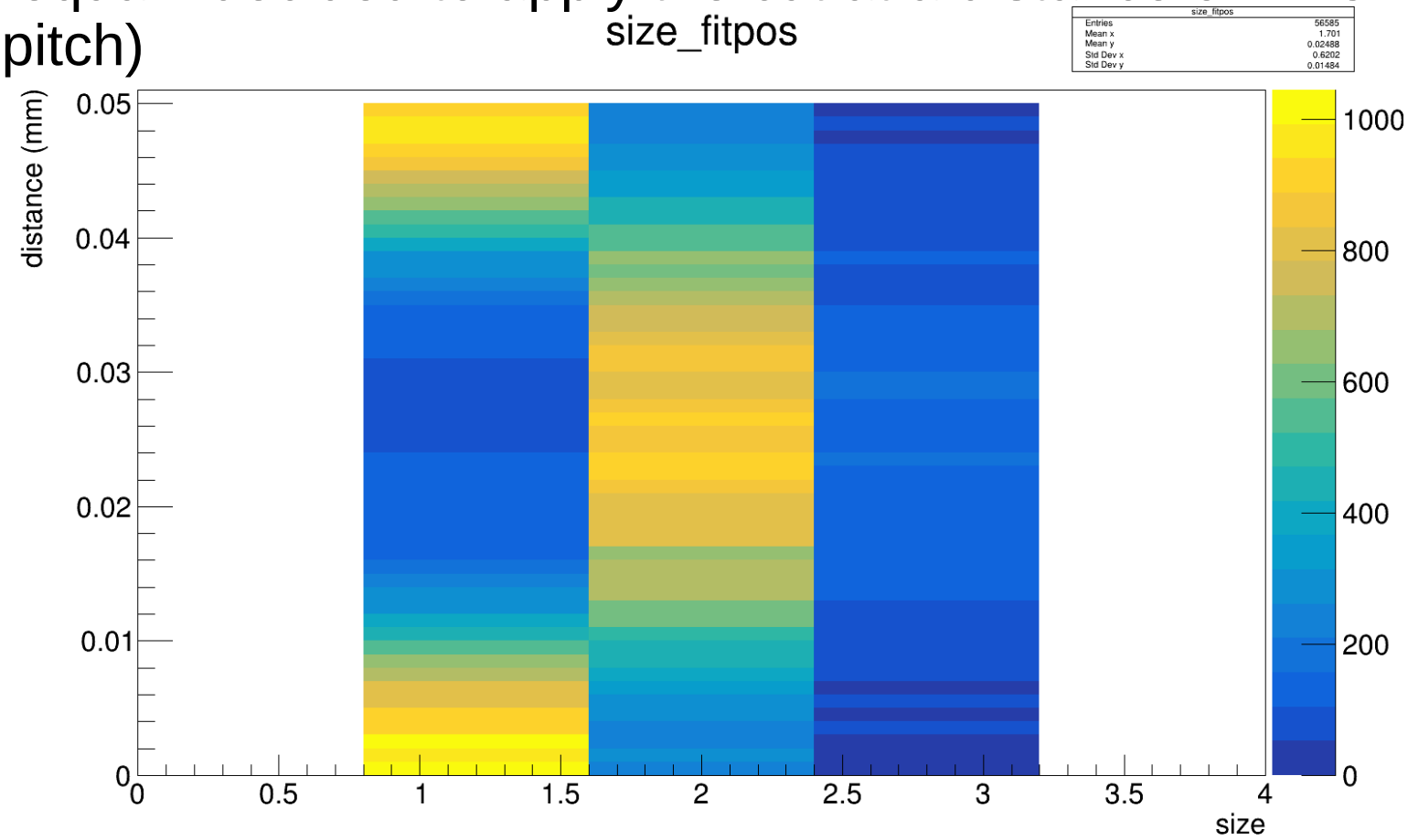
Investigating charge sharing and loss to backplane

- To first order it is true that single strip clusters are mostly at the edge and multi strip clusters mostly in the center.



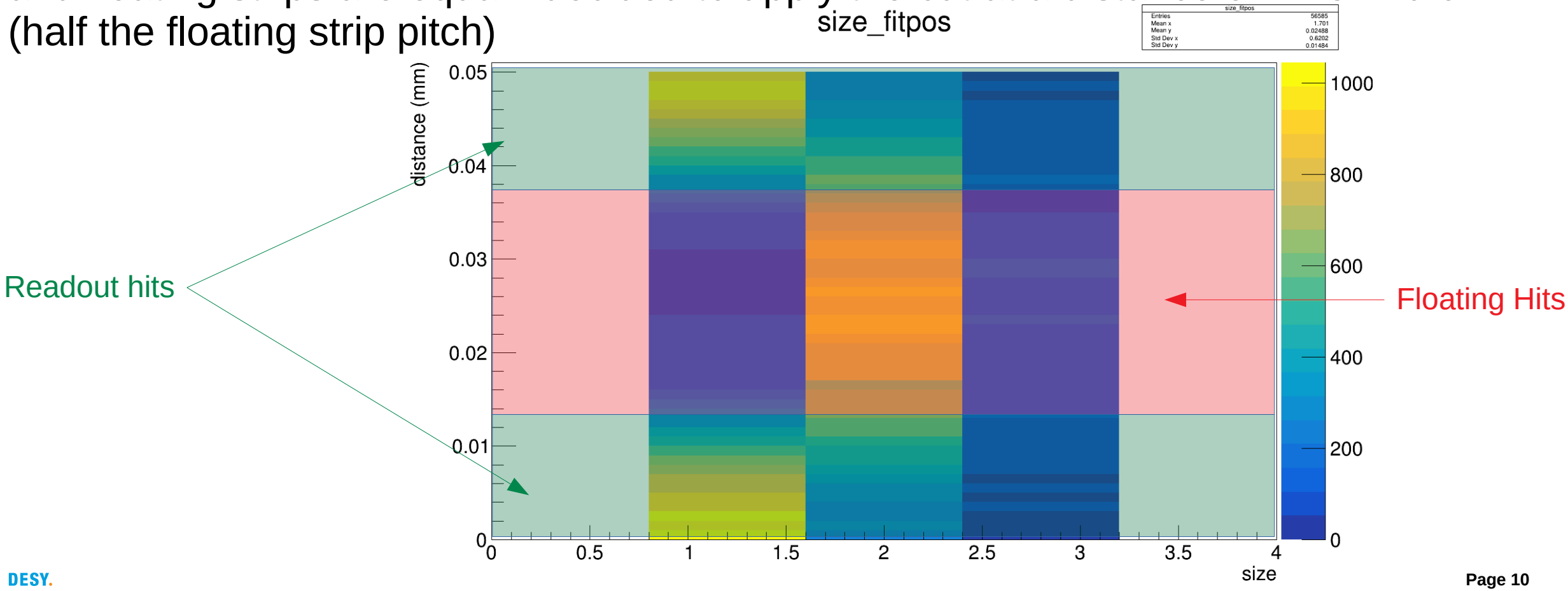
Investigating charge sharing and loss to backplane

- The question was then where to perform a cut on this distance from readout strip parameter.
- After some thought, since electrically for charge drift within the sensor both readout and floating strips are equal I decided to apply the cut at a distance of 12.5 micron (half the floating strip pitch)



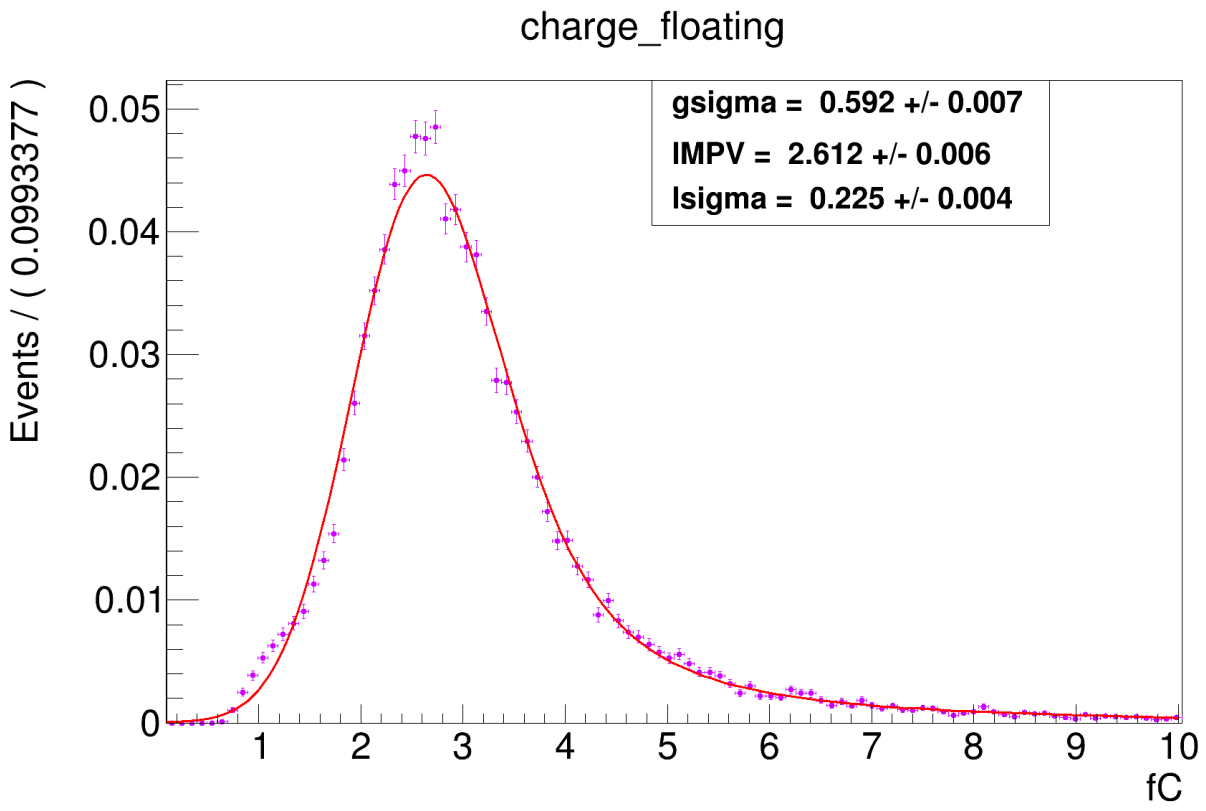
Investigating charge sharing and loss to backplane

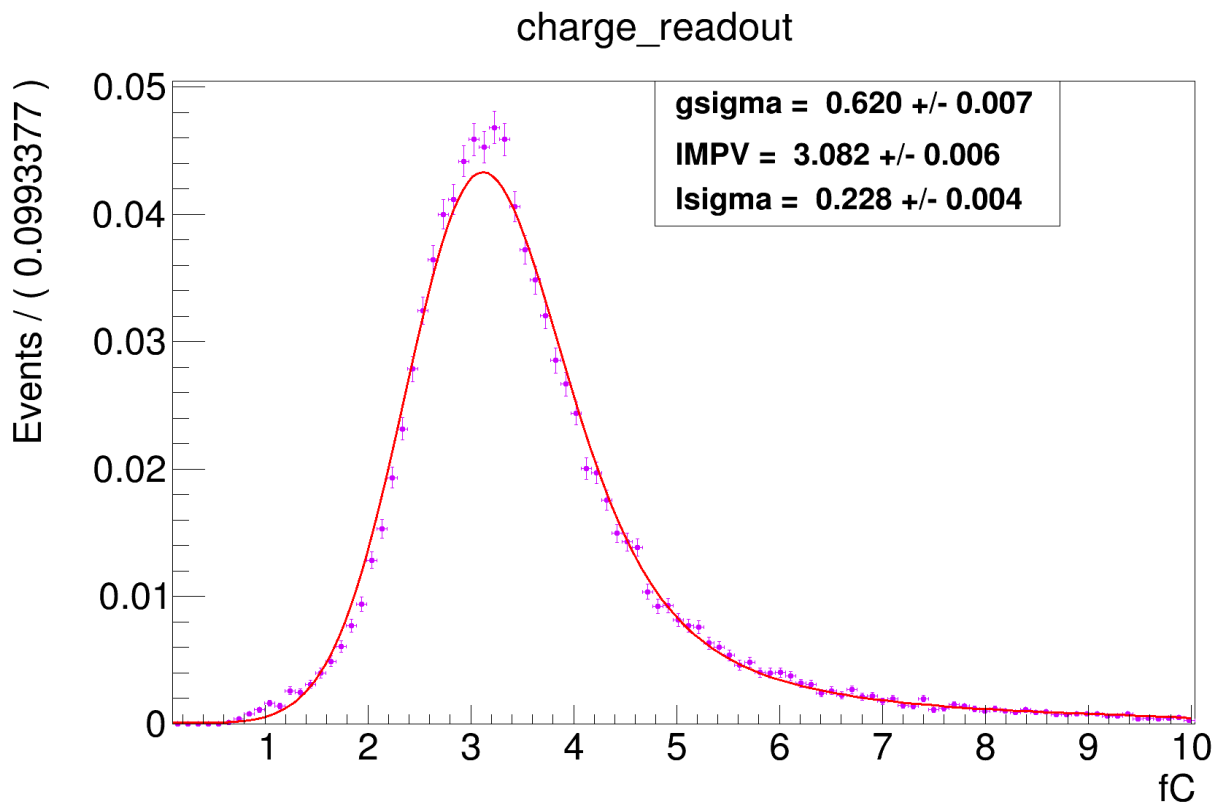
- The question was then where to perform a cut on this distance from readout strip parameter.
- After some thought, since electrically for charge drift within the sensor both readout and floating strips are equal I decided to apply the cut at a distance of 12.5 micron (half the floating strip pitch)



Investigating charge sharing and loss to backplane

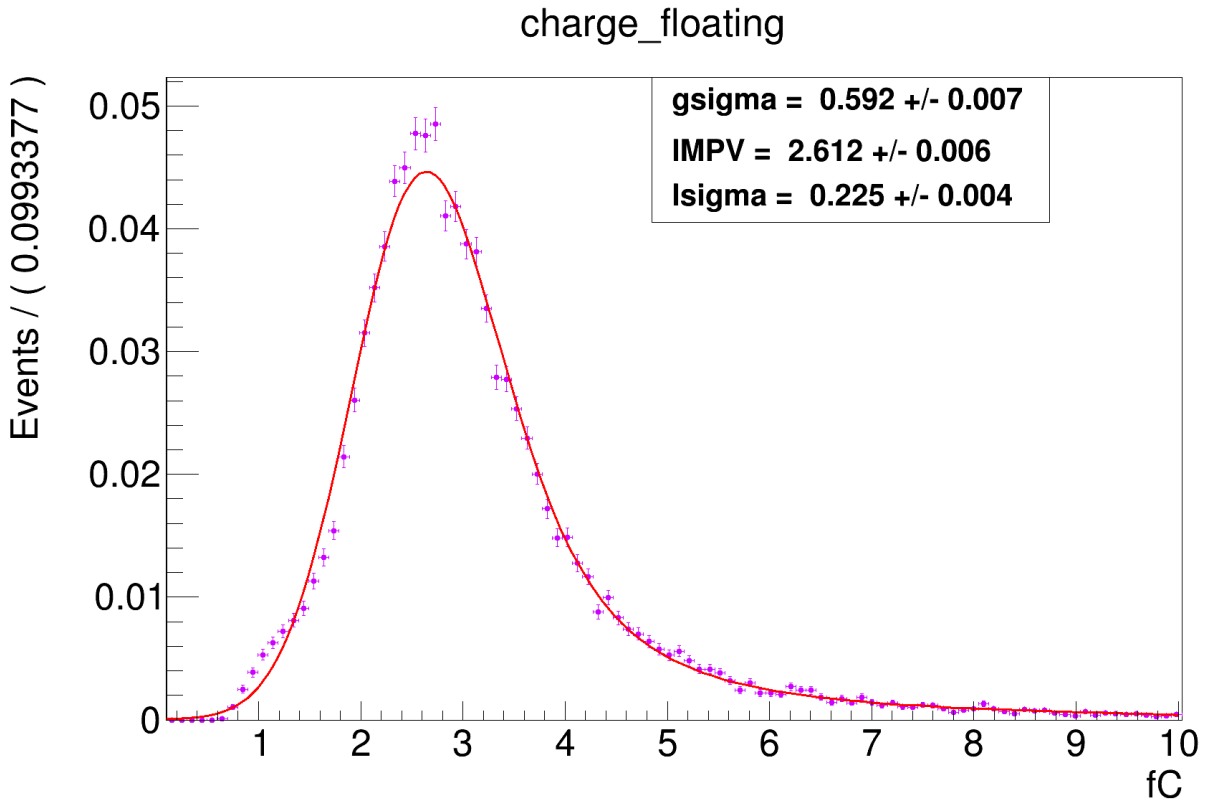
- With this split I decided to check the charge of these clusters and fit a landau gauss convolution to each of them.
- In general the bump is now present in both but more equally split between readout and floating hits.

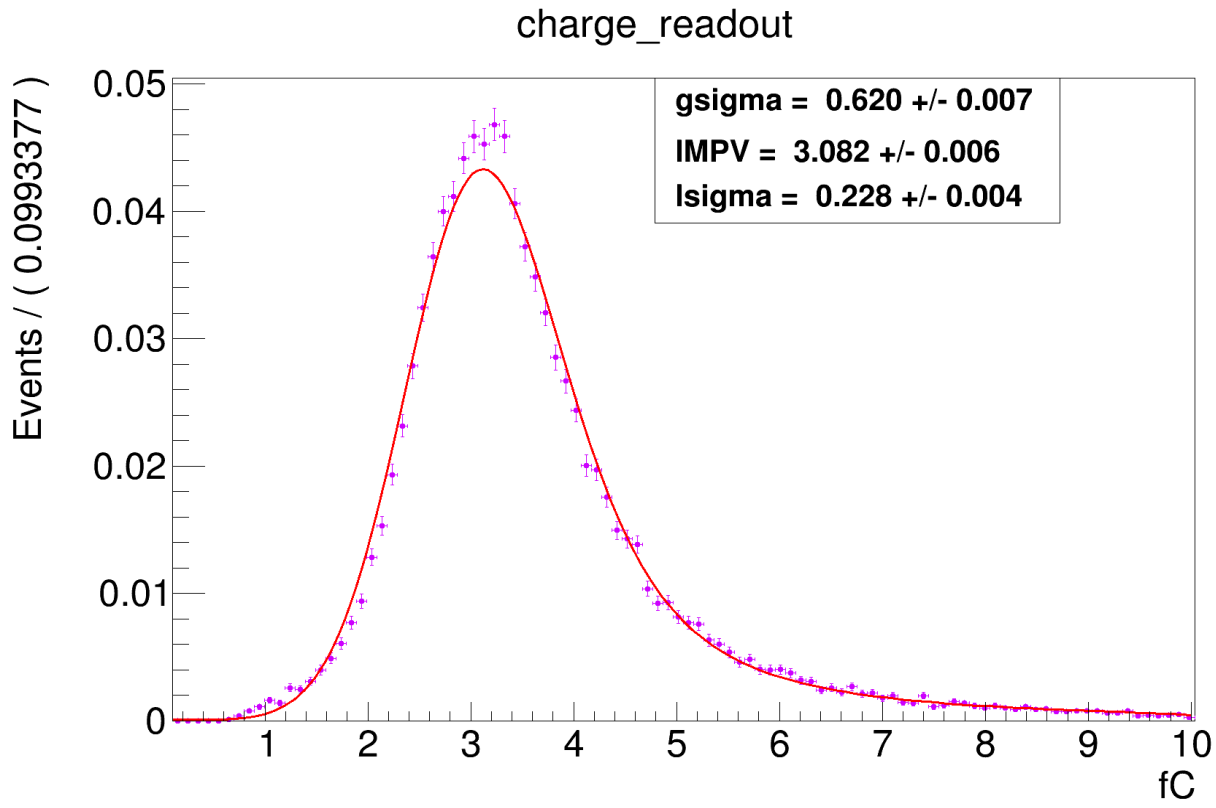




Investigating charge sharing and loss to backplane

- Using their MPV I can extract the loss to the backplane that floating strip this get.
- 2.612 fC / 3.082 fC ~= 0.85
- So we would see a loss of 15% to the backside which would give us a ratio of C_b/C_{ss} of about 0.088.

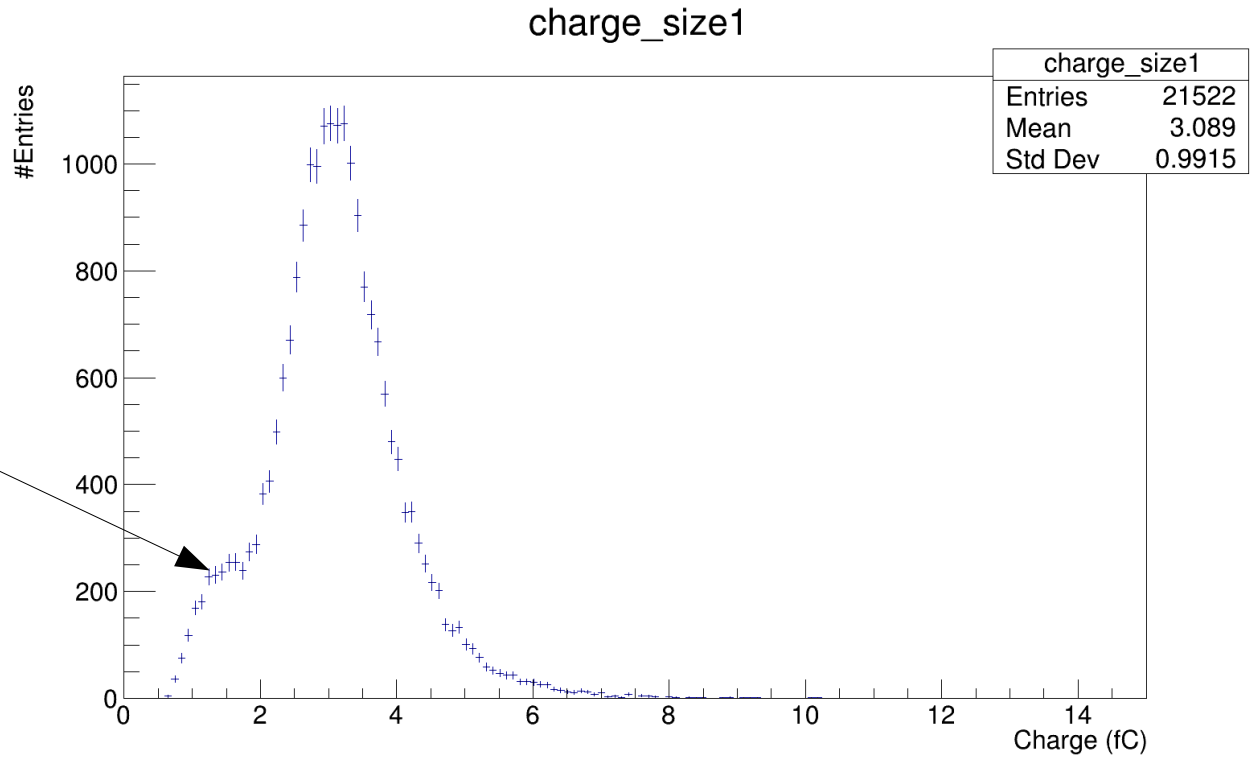




Potential reasons for the low charge bump

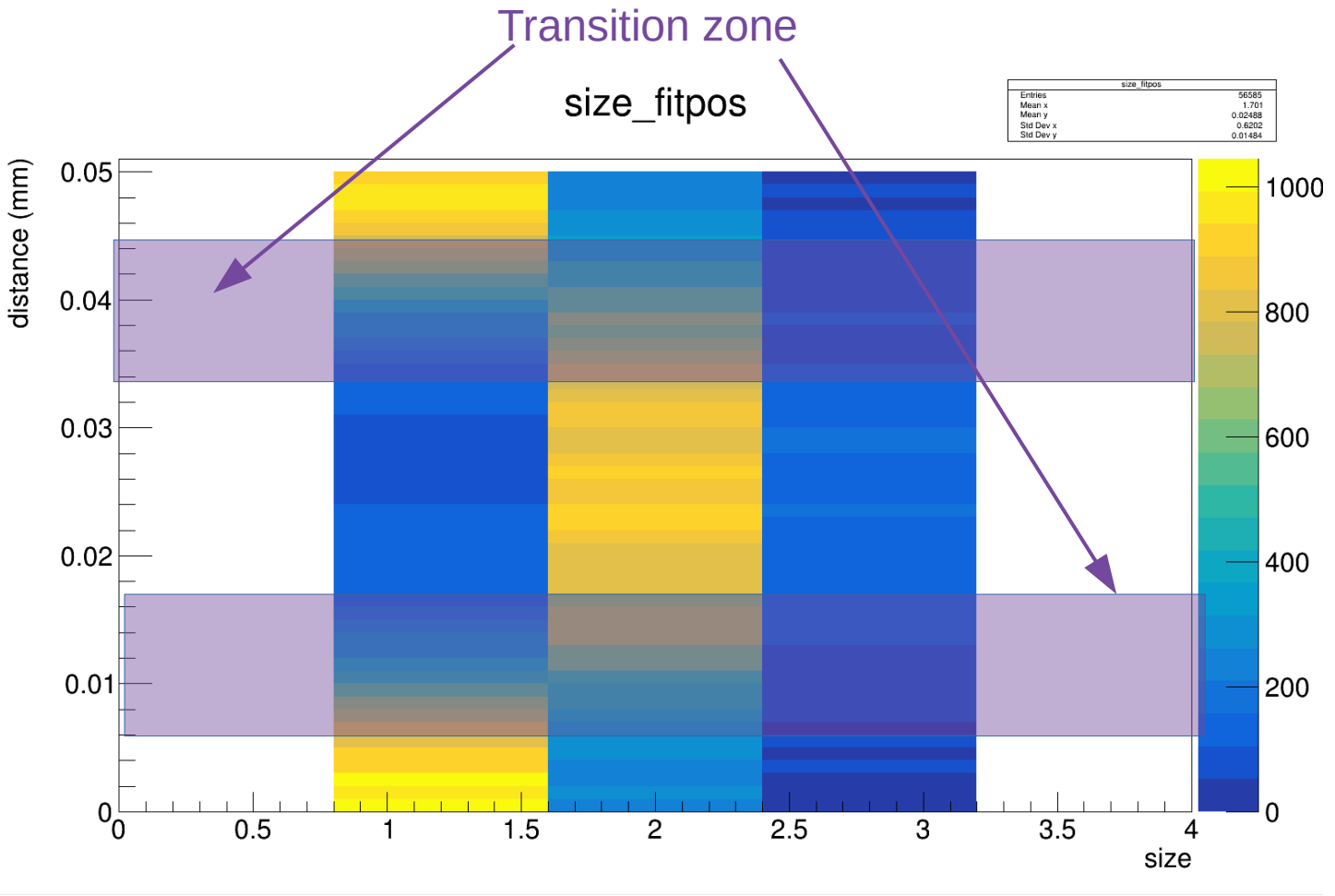
- Weird bump at low values.
- Either:
  - 2 strip clusters where one strip was lost to the S/N cut
- Or:
  - Incorrect cluster splitting splits charge between adjecent clusters incorrectly

(explanation at the end)



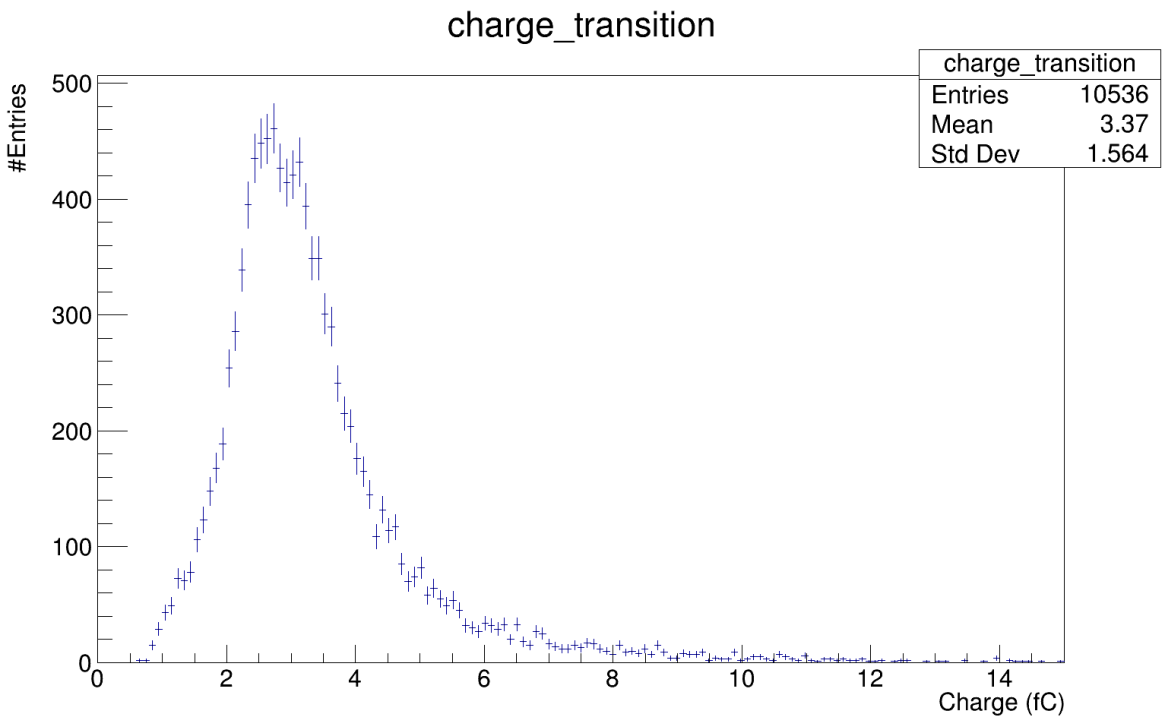
Potential reasons for the low charge bump

- Theory 1:
  - These are simply hits in the transition zone between floating and readout strip. This results in one strip receiving ~¾ of the charge and the other receiving ~¼ (ignoring loss to the backside)
  - If the ¼ charge is below my cut value of S/N > 2 then this charge is also lost artificially reducing the charge of the single hit cluster.
- Test: Look only at the charge distribution in the transition zone between floating and readout strip



Potential reasons for the low charge bump

- There is no clearly visible bump in the transition zone.



Potential reasons for the low charge bump

- Theory 2:
  - Artifact of the clustering algorithm.
  - Example: If we have 10 strips as a sub sample. With the number being their S/N value and | being a split where one cluster stops and another begins

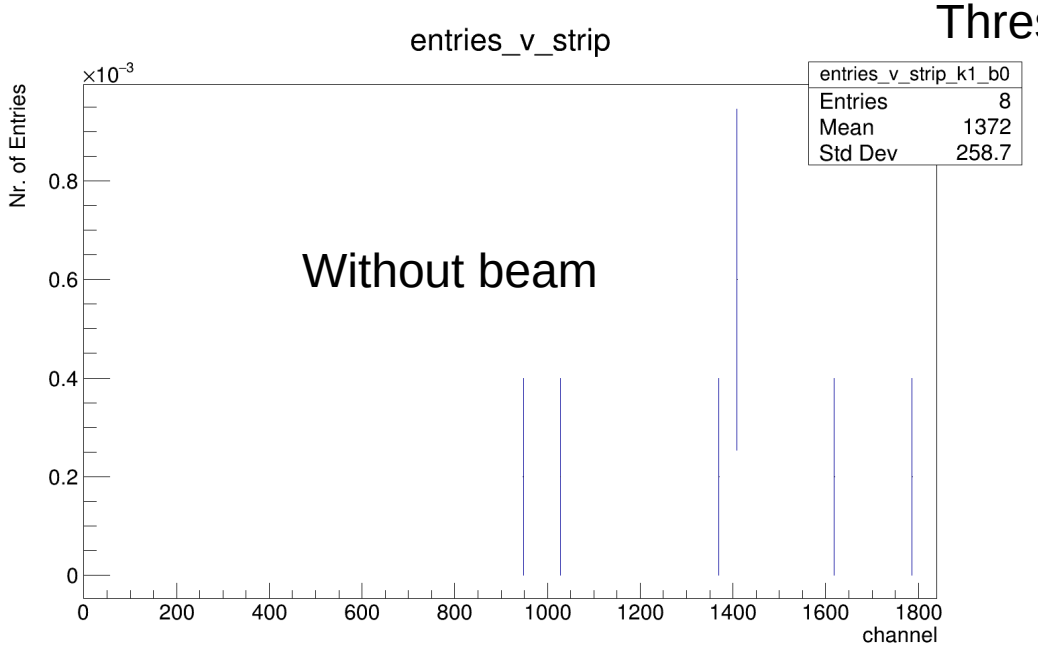
0 1 | 2 6 4 | 5 | 1 | 4 2 | 1

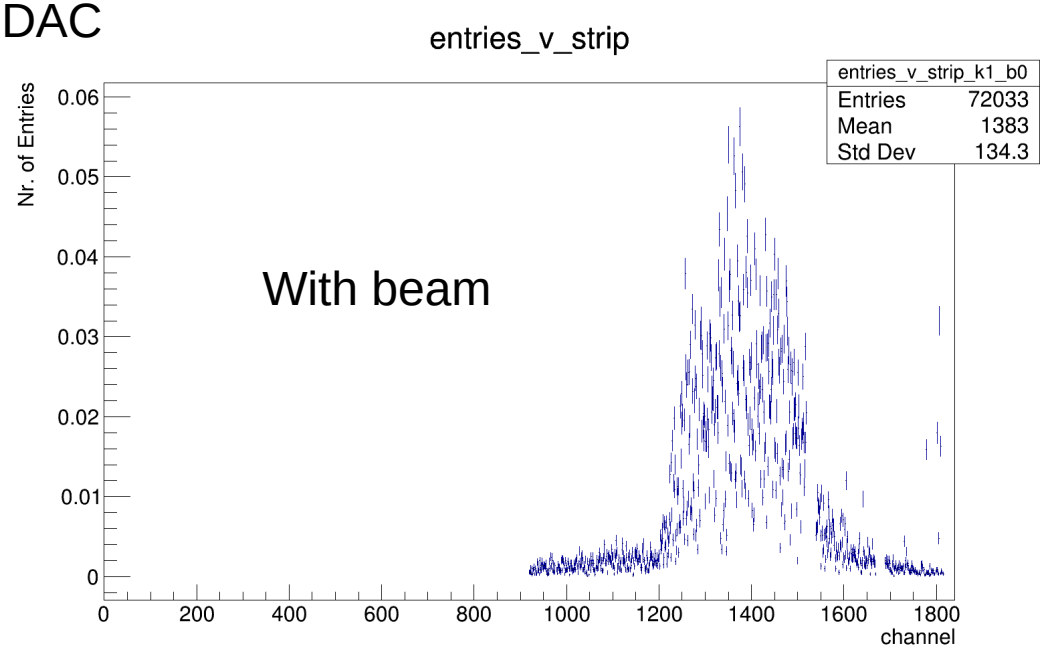
  - The algorithm starts with the highest S/N strip as seed and continues to add up all adjacent strip until one of two cut criteria is fulfilled
    1. If no adjacent strips are above my S/N cut (2 in this case) then the clustering stops
    2. If the next adjacent strip has higher charge then the previously added strip the clustering stops. This is because I expect the charge to drop monotonously with increased distance to the center and therefore a higher charge can only be explained by a new cluster seed.
- The clusters are the numbers in red. The issue is that nothing in principle guarantees me that the full charge of the strip between 6 and 5 should be completely assigned to the left cluster and not partially or completely assigned to the right cluster.
- Test: I have some idea to, after clustering, perform a reevaluation where ambiguous strip are split according to the charge of the seed strip and the distance.
  - This would require some reworking and lots of tests concerning the stability and functionality and I currently have many things on my plate already. Not speaking of whether this is even the correct choice of action

Side Note: Self triggering operation

- I mentioned that I wanted to take self triggering data during test beam. If just to prove that it does not work. I never bothered looking deeply into the data. What I show below is not any deep analysis just two little plots I found noteworthy that I apparently never reported.

Threshold of 230 DAC

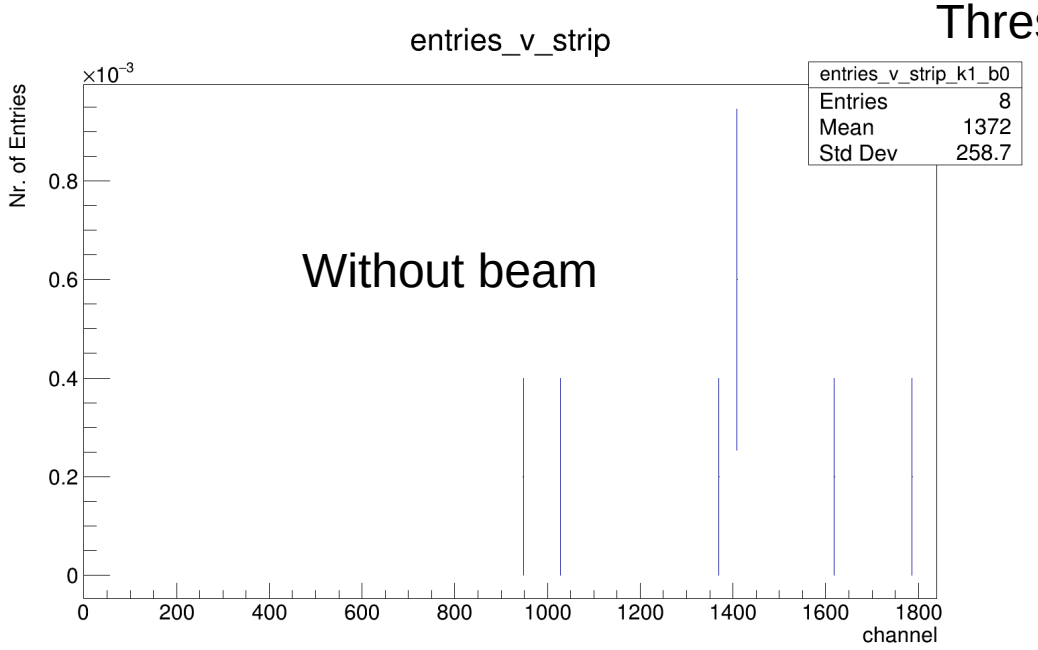


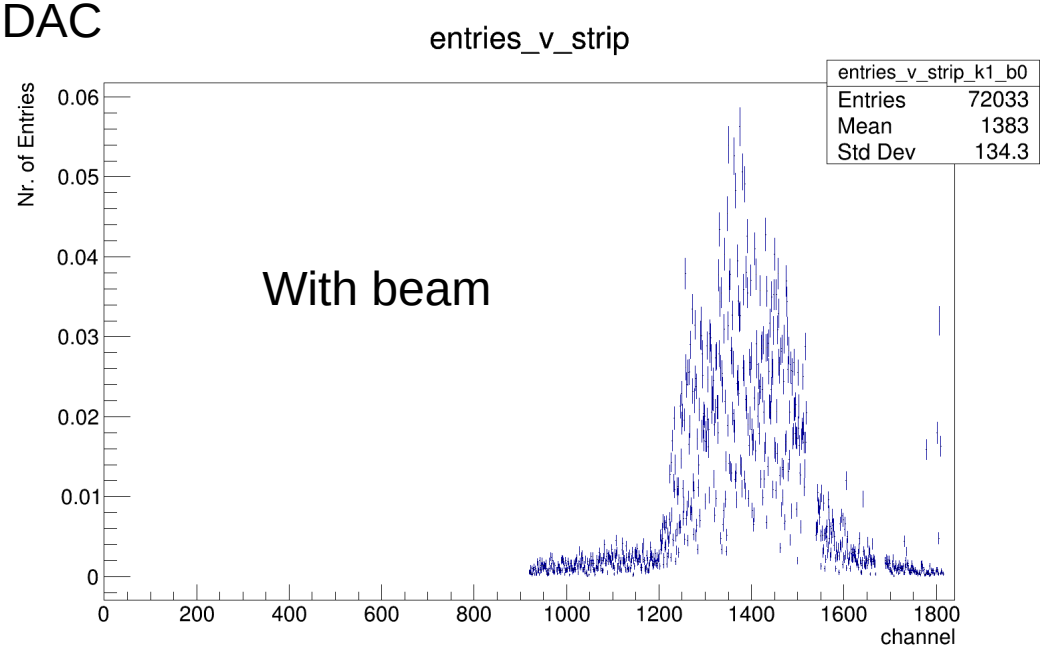


Side Note: Self triggering operation

- I mentioned that I wanted to take self triggering data during test beam. If just to prove that it does not work. I never bothered looking deeply into the data. What I show below is not any deep analysis just two little plots I found noteworthy that I apparently never reported.

Threshold of 230 DAC

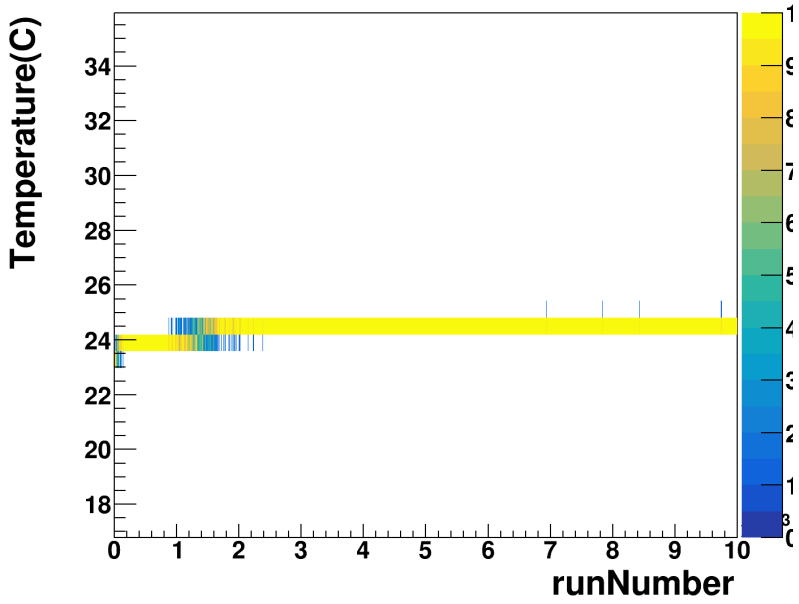


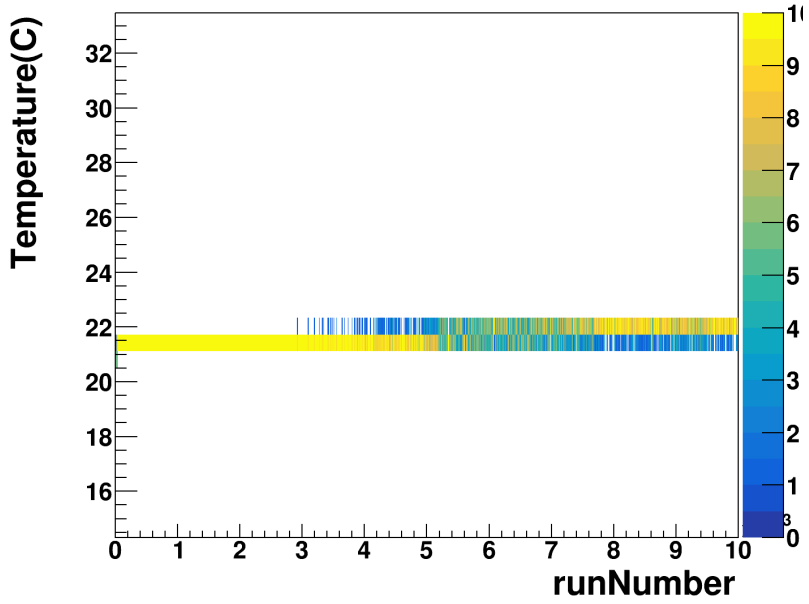


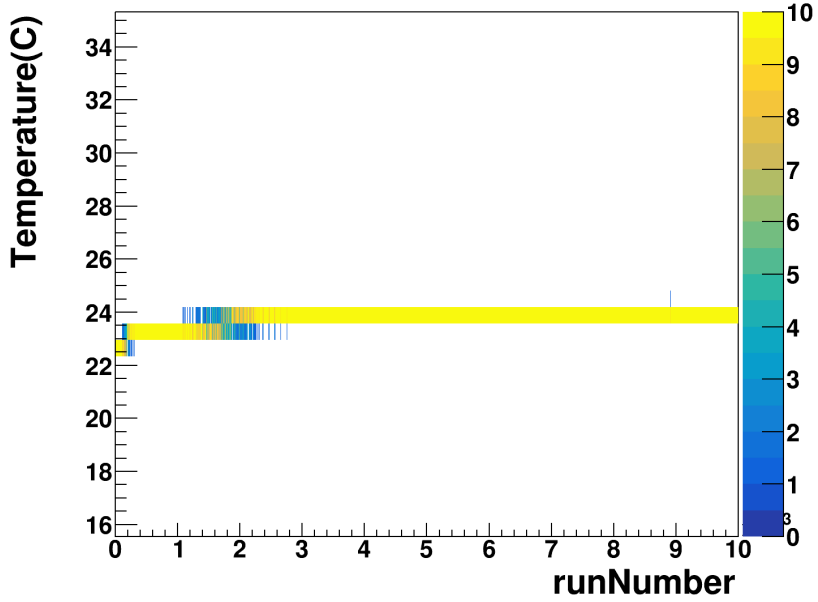
- To get to this result required significant effort.
  - I started at a threshold of 200 DAC. And slowly moved down to a DAC value of 230. I only enabled half of the KpIX (one side of a sensor) and had to remove roughly 30 to 40% of all channels to ensure no monster events happen.
  - I started at 200 and removed channels that trigger often, then moved to 205 and repeated this procedure until I arrived at 230.
  - Going further in DAC threshold would require disabling roughly 70% of all channels so I decided to stop

Backup

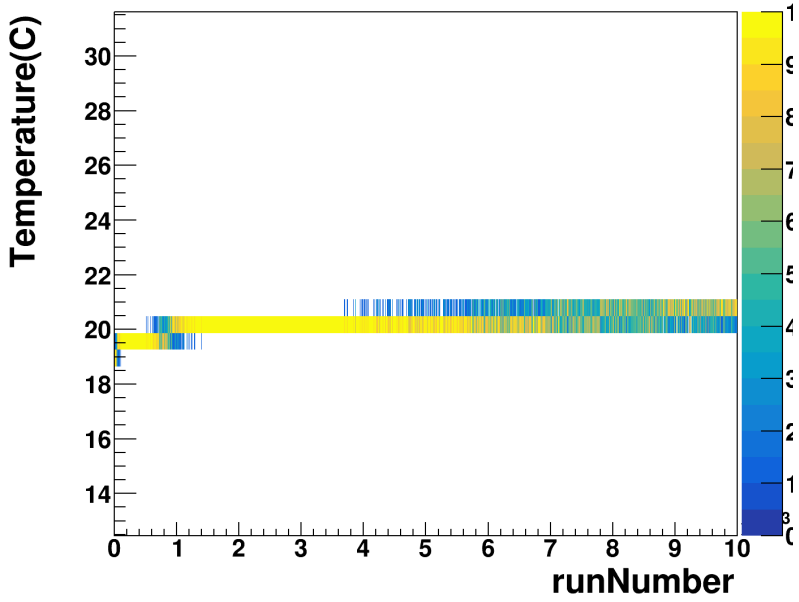
Kpix Temp (0+1+2)

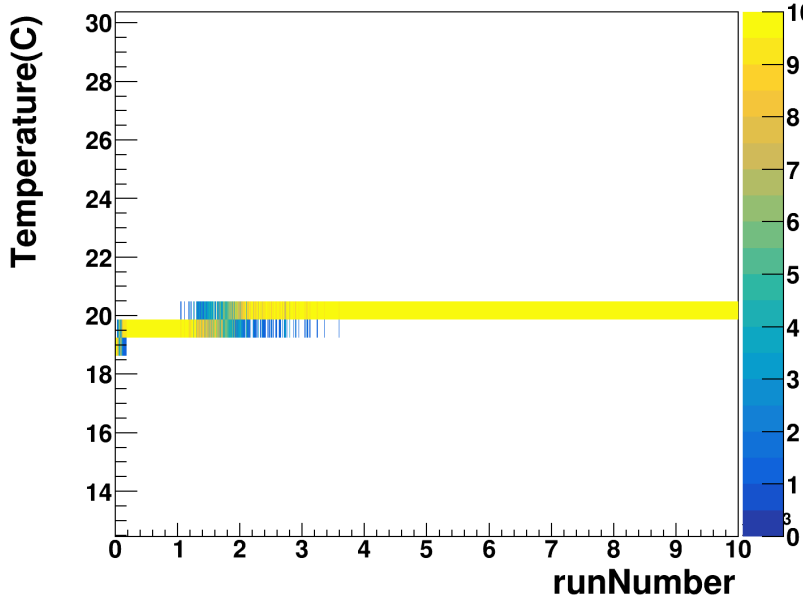


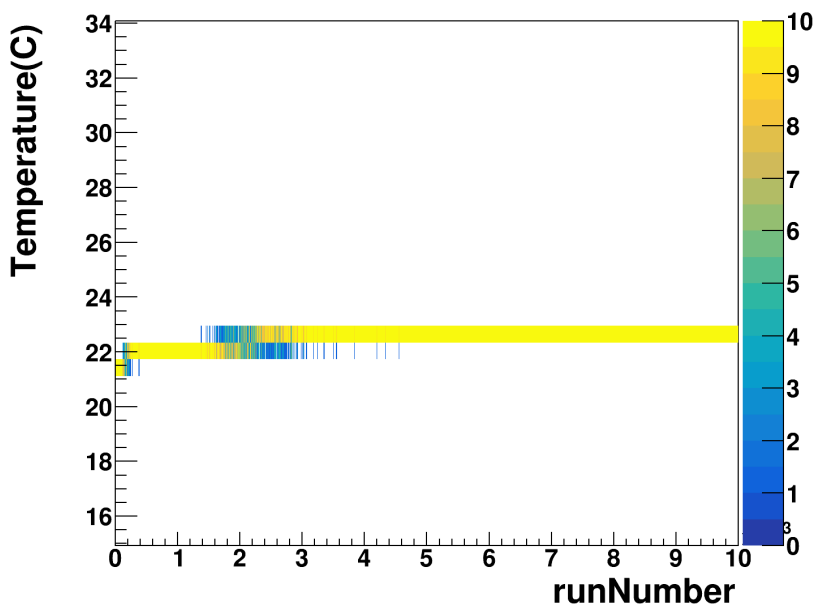




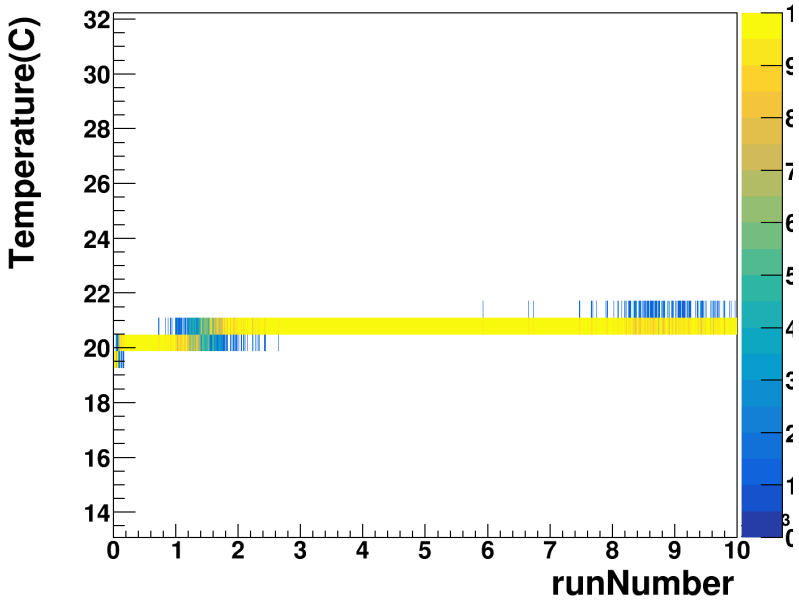
Kpix Temp (3+4+5)

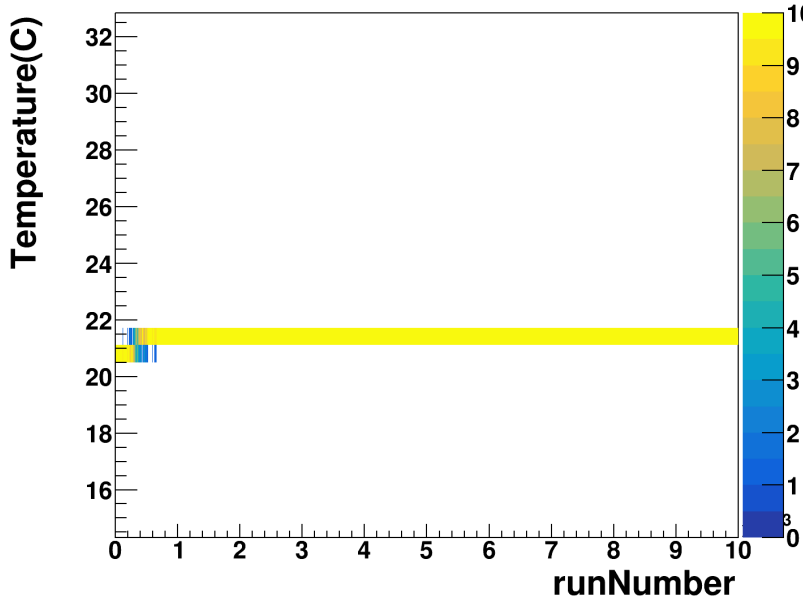


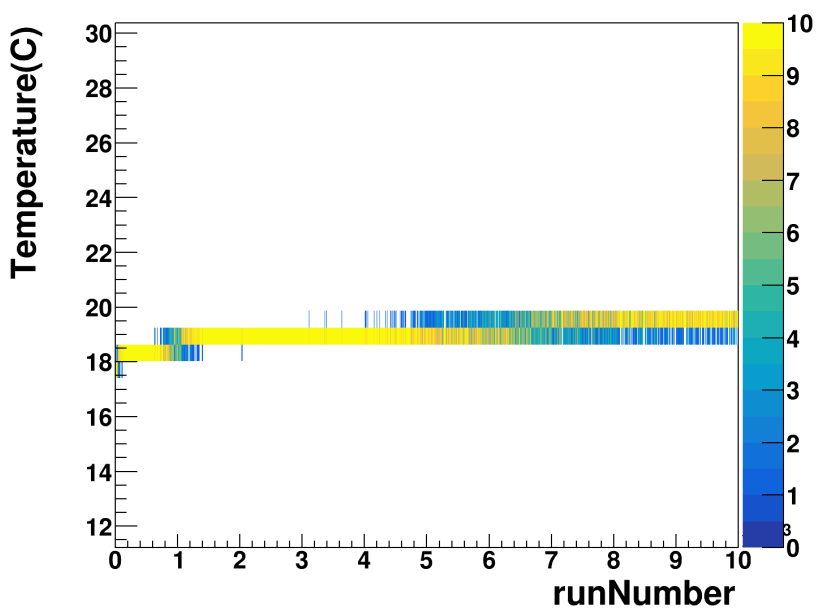




Kpix Temp (6+7+8)







Kpix Temp (9+10+11)