UNIVERSIDADE FEDERAL DE JUIZ DE FORA

Design and development of trigger-less trigger for CYGNO Data Acquisition System

Preliminary results

Igor Pains
Rafael Nóbrega and Giovanni Mazzitelli

1.
Summary

Summary

- Introduction
  - Proposal
  - Datasets
- Development
- Results
  - False alarm
  - Signal detection efficiency
- Conclusions
  - Conclusions
  - Next steps

2.
Introduction

Proposal

- Develop two algorithms to be tested as online trigger to decide whether to save or not images taken by the detector
  - **Simple algorithm based on subimage metrics: mean, std**
    - We are starting from the simplest possible case to then:
      - gradually increase complexity following up the performance evolution (efficiency, false alarm and response time) and
      - optimize computing processing issues
  - **Convolutional Neural Network**
    - on training stage (not tested yet)

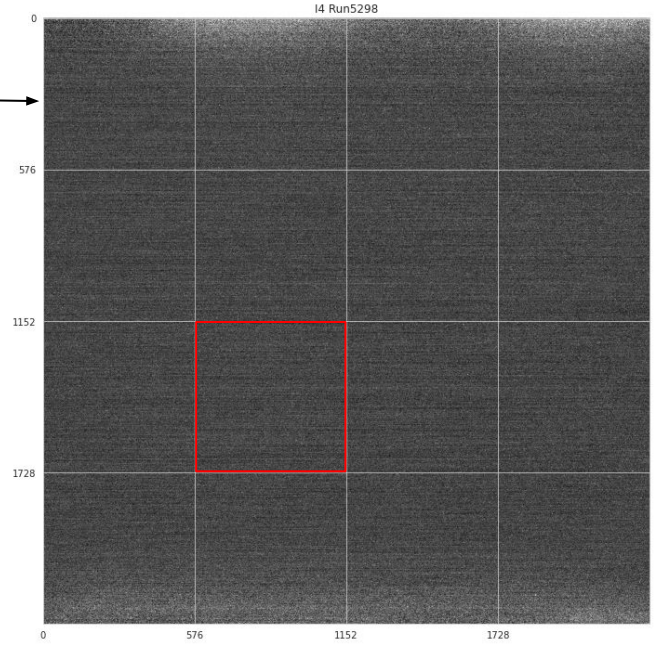
Subimage method

Division of the image

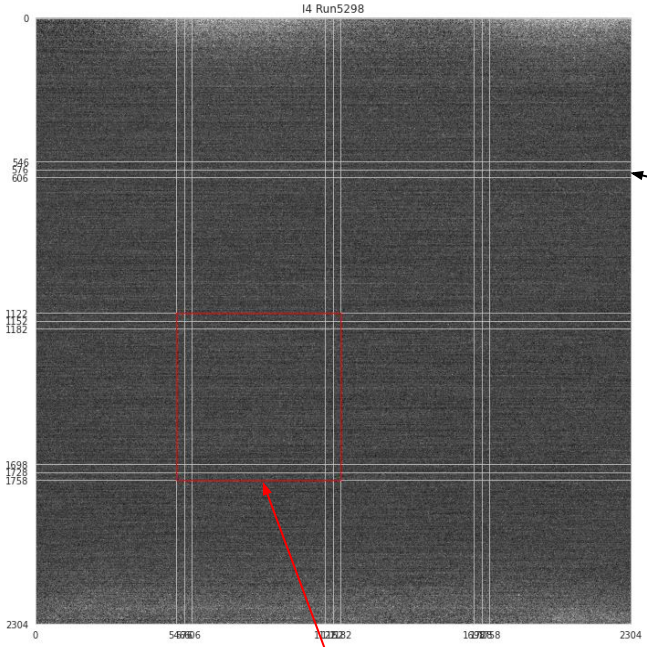
Store the mean and std of the pixels

Decide a threshold based on the distributions.

Original idea



Overlapping implementation



Overlapping region between two subimages

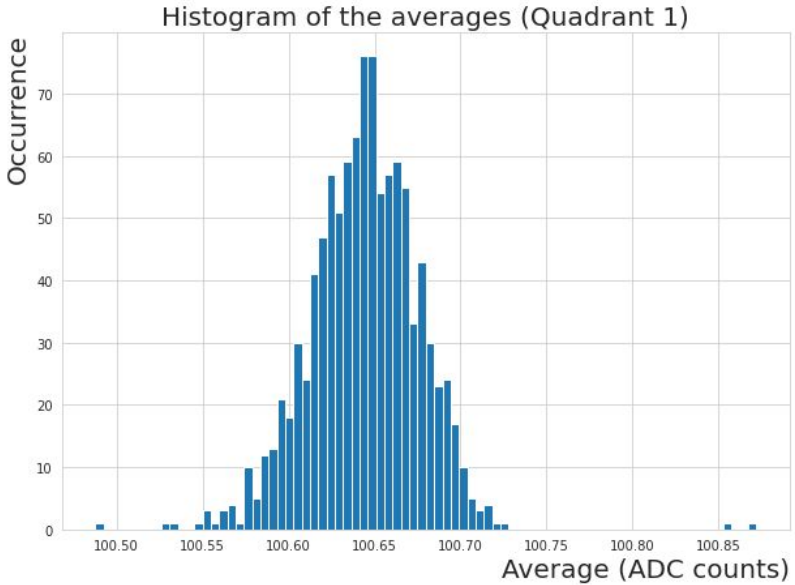
Subimage considering the overlapping region

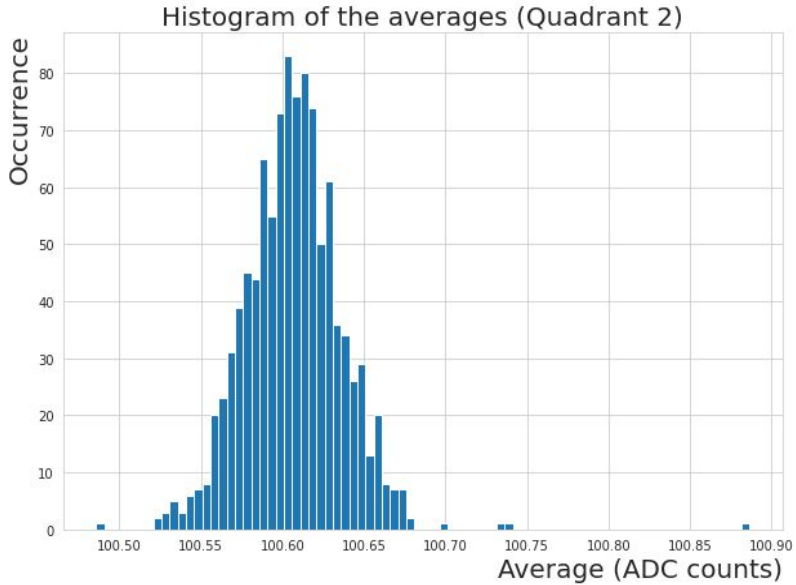
Datasets

- Three datasets were selected to develop this algorithm:
  - Train: 10 pedestal runs taken underground (1040 images).
  - Test (background): 10 pedestal runs taken underground (1033 images).
  - Test (signal): ^{55}**Fe** simulation combined with pedestal runs taken underground (300 images).

3.
Development

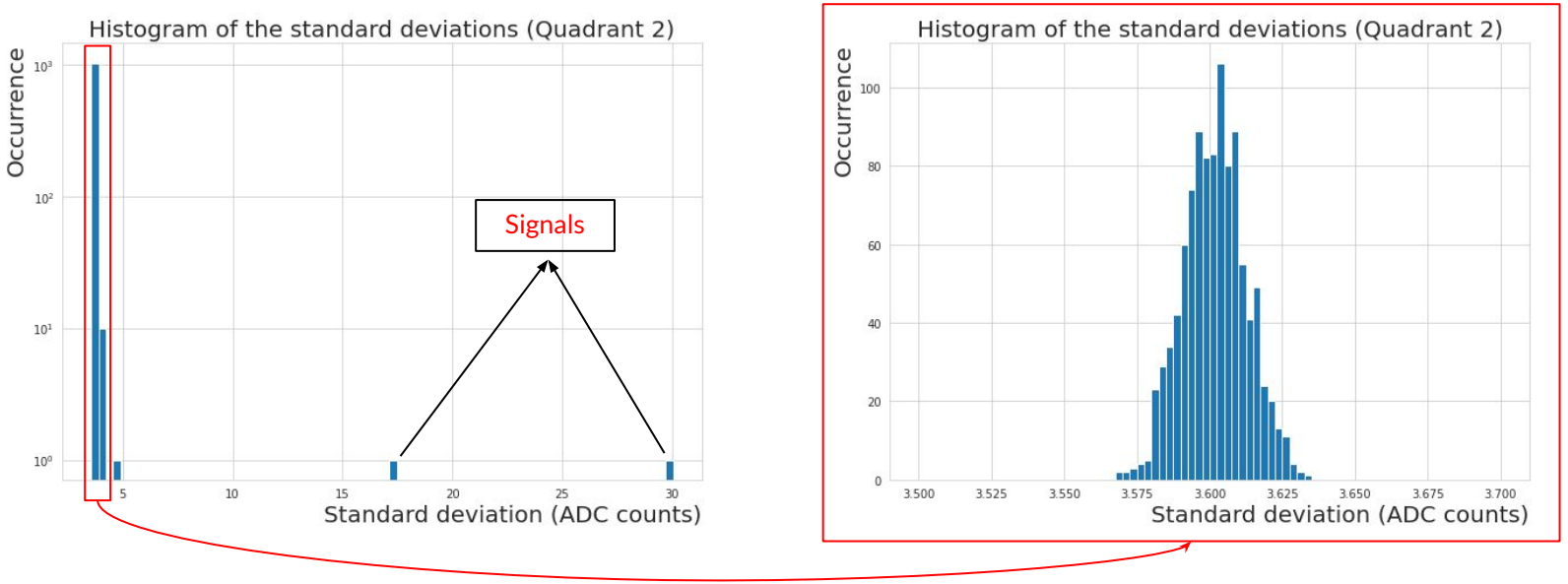
Average distributions





- After taking a first look at the average distributions, they have a ~ gaussian shape.
- Thus, the initial idea was to find a threshold based on the mean and standard deviation measures.

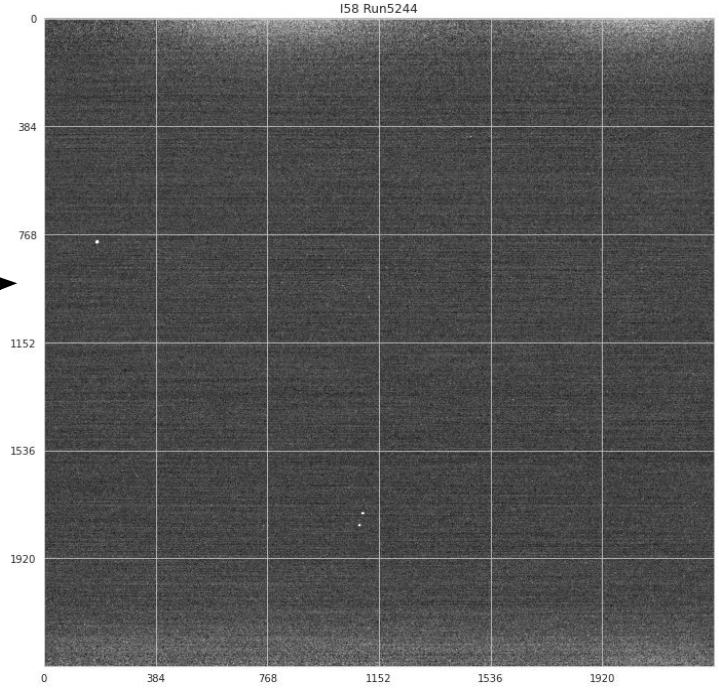
Standard deviation distributions



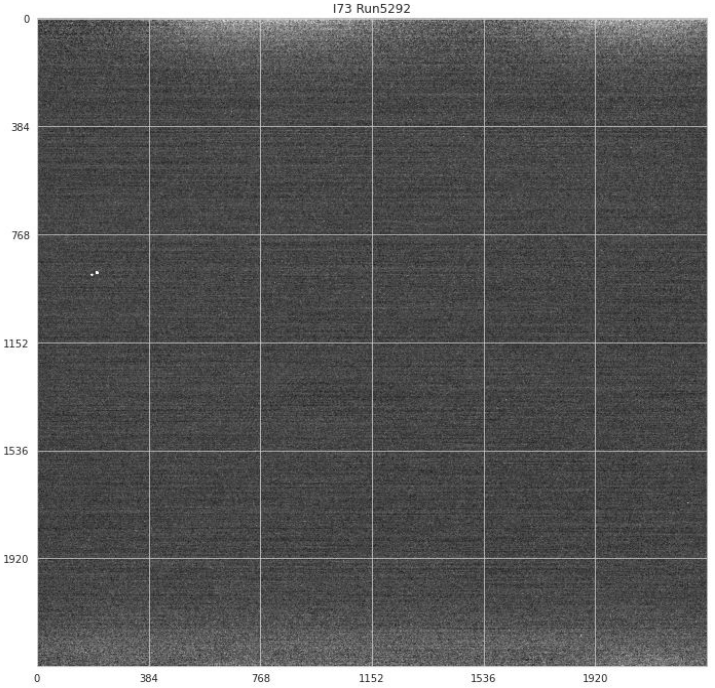
- Giving a a first look at the standard deviation distributions, unexpected high values were observed.
- The train dataset contained images with some kind of '**dots**' not related with electronic noise.
- Taking a look at the core parts of the distributions, they have a ~ gaussian shape.

Signals

One of the **'dots'** detected on quadrant 2



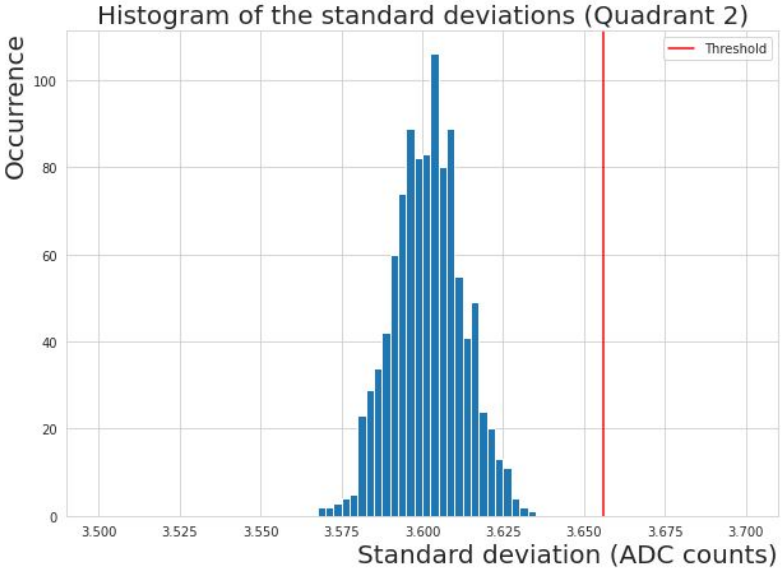
One of the **'dots'** detected on quadrant 2

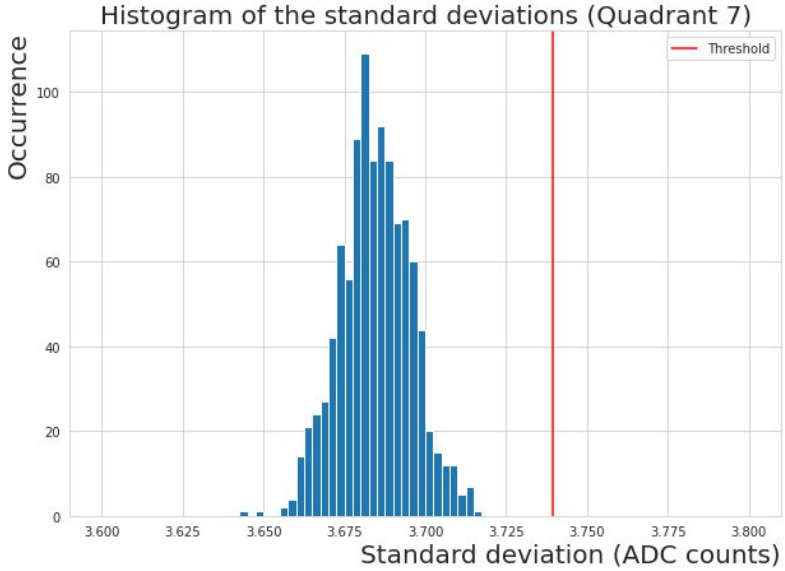


Threshold selection

- The main idea is to process the distributions to get rid of possible outliers *(usually caused by signals)*.
- To use a gaussian model to estimate the mean and standard deviation to then define a threshold based on them.
- The threshold used for the first tests was:
  - Mean + 5*(Standard Deviation).

Threshold selection





- Example of a threshold selected on the basis of the standard deviation distribution.

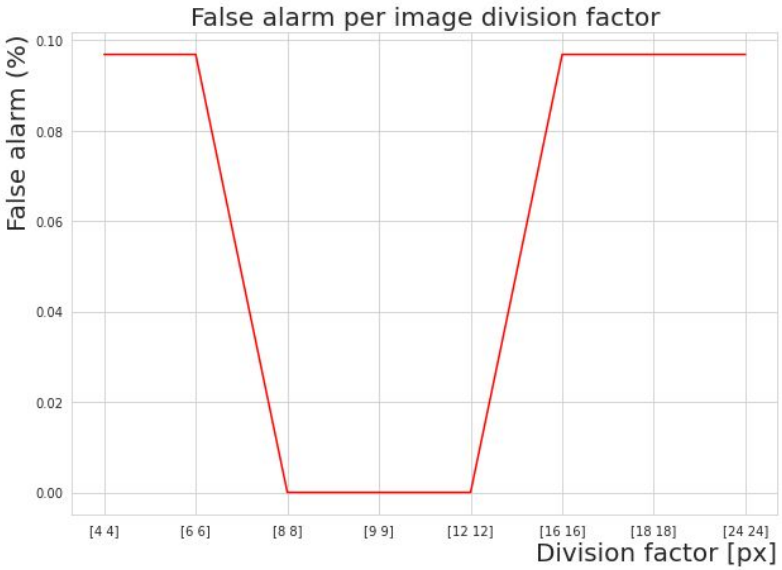
4.
Results

Number of divisions scan

- Following the threshold selection methodology, a scan on the number of divisions in each axis was done.
  - Symmetric division factor from 4 to 24 (each subimage shape from 576x576 to 96x96 plus the overlapping region).
  - Each division factor has an array of thresholds that will be used to in each quadrant to detect signal or sparks.
  - The time used for the analysis considering each division factor was also stored.

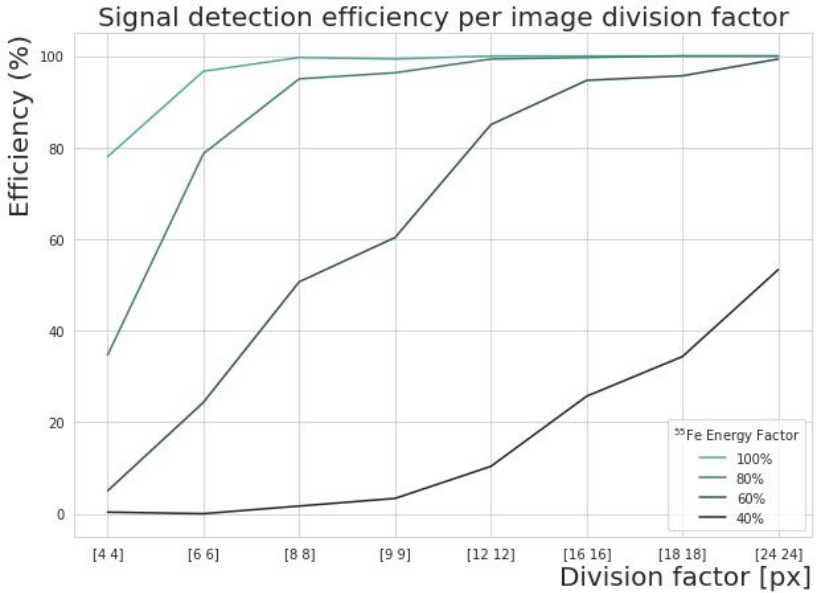
False alarm performance

- The thresholds calculated on the training dataset were used on the test dataset.
- The raw images were used, with no preprocessing.
- A total of 63 signals were detected on the test dataset.
- A total of 2 events passed the threshold without containing a signal.

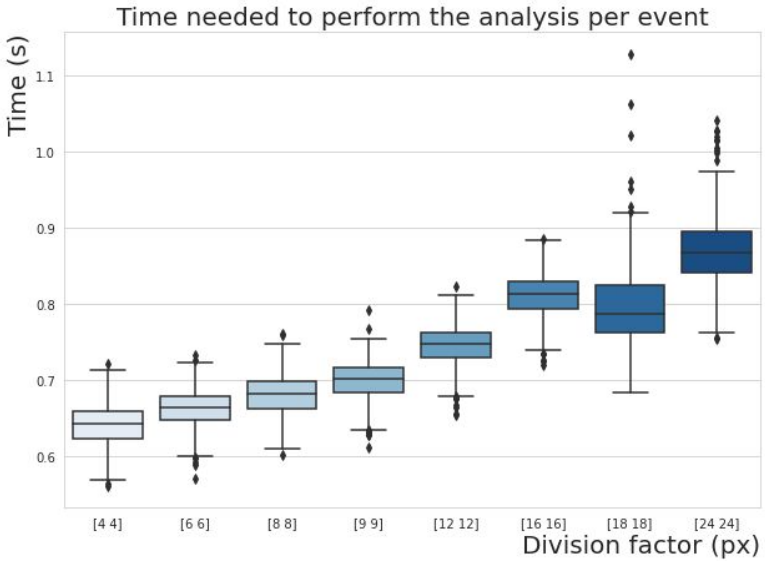


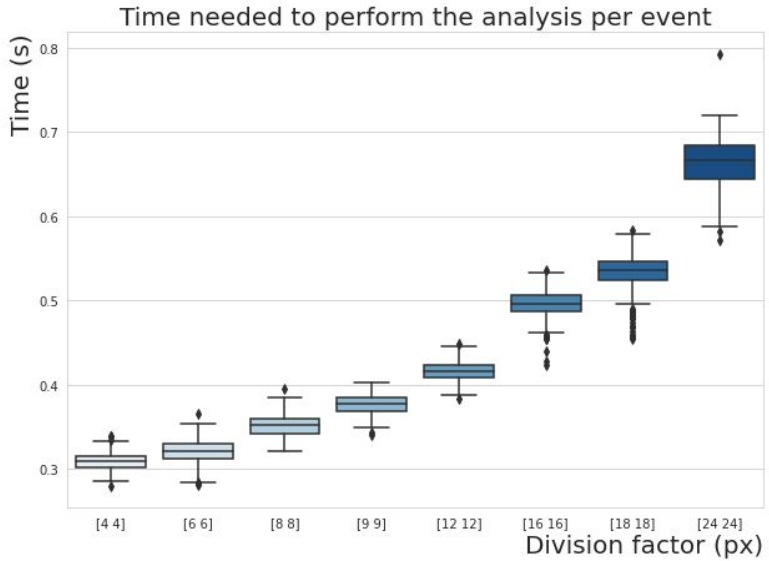
Signal detection efficiency

- The simulated events used in this scan had signal centered in a random position in each image.
- In order to simulate signal with energy smaller than 5.9 keV, each ADC of the signal was multiplied by a factor.
- It is clear that the higher the number of divisions, the more sensitive to signal the algorithm is.



Time efficiency





- On the left is the boxplot of time used per event to load the image, divide it and store the features on the training dataset.
- On the right is the same analysis, but not taking into account the time to load the event, on the test dataset.
- Both times were measured on the same machine (cloud) and IDE (jupyter notebook).

5.
Conclusions

Conclusions

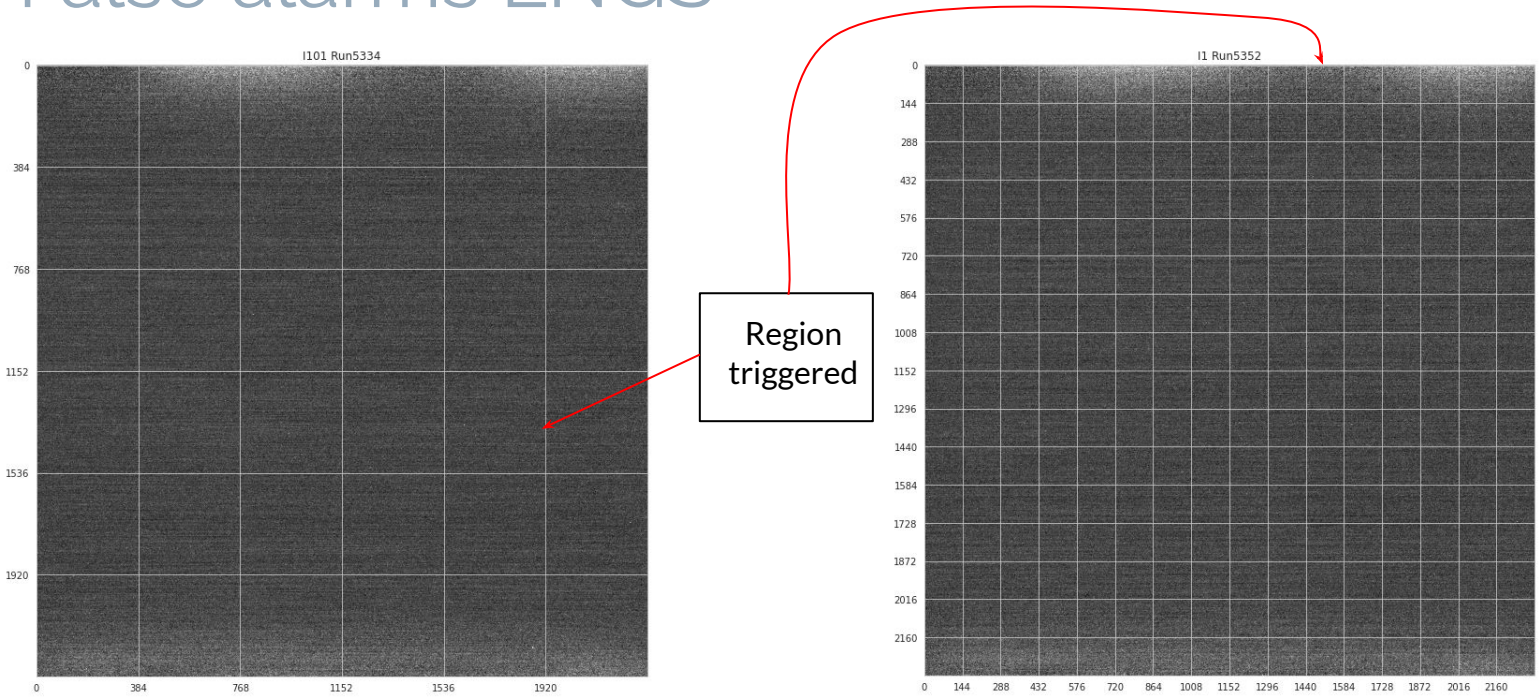
- Two algorithms are being proposed and in development: one which splits the image into several blocks and one based on a CNN
  - a simple version of the first case is being tested at LNF and it will be used as a starting point to gradually increase its complexity to arrive to a more elaborated and efficient proposal.
  - for the second case, the CNN design is being evaluated.
- The tested algorithm presented a decent performance considering its simplicity on both background rejection and signal efficiency *(for 5.9 keV)*.
- It struggles to detect signals with smaller energy *(for events with the same energy profile of a ^{55}Fe signal but factored).*

Next steps

- Explore other promising features to increase the signal detection and background rejection.
- Speed up the algorithm: use the available GPU on the trigger machine on LNF, use multiple cores of the processor or do an implementation with cython.
- Compare these results with a trained CNN.

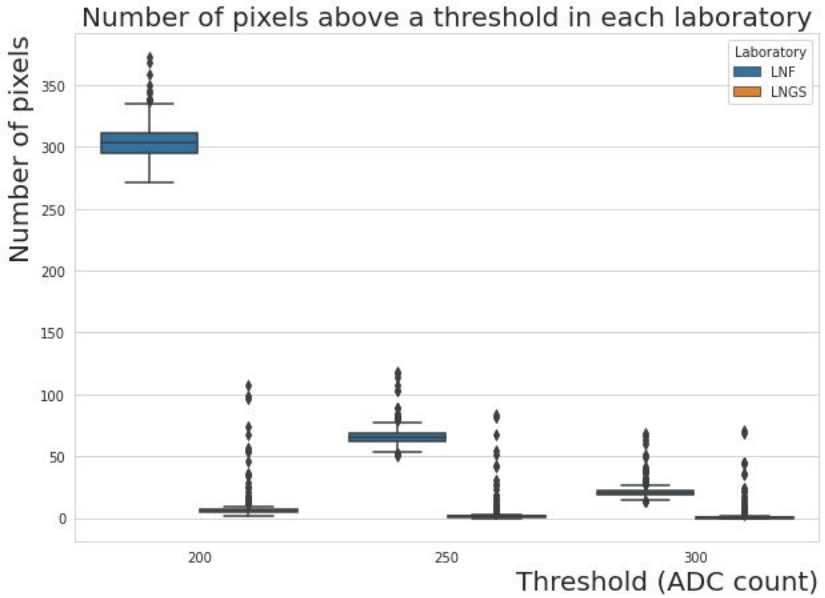
6.
Backup slides

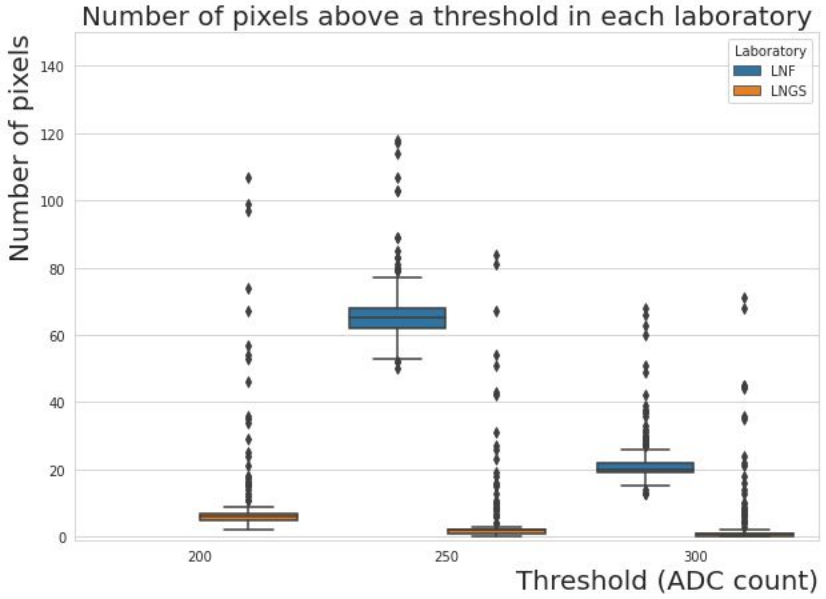
False alarms LNGS



- One of the false alarms occurred because of a hot pixel whereas the other on the border of the image.

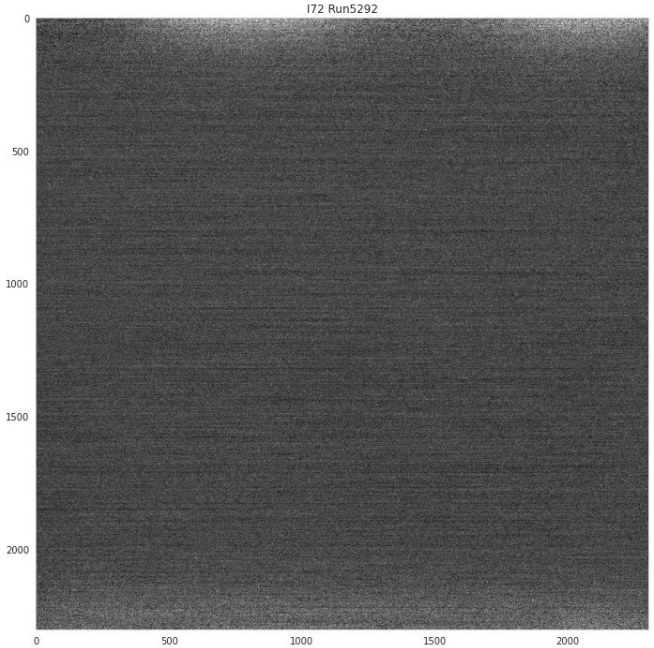
Pixels above threshold LNF vs LNGS

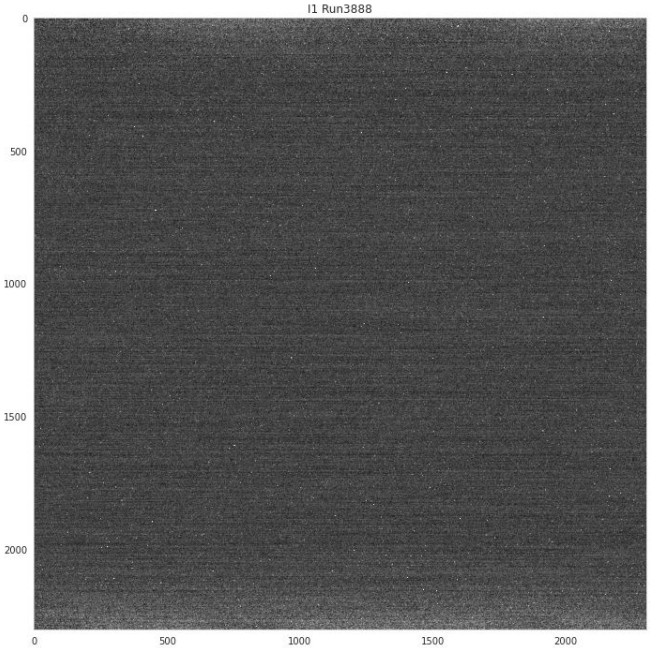




- The LNF images contain more sparks and are noisier than the LNGS ones.
- This behavior makes an algorithm based on the changes on quadrant's pixels struggle to ignore the background, so a preprocessing is needed.

Image from LNGS vs LNF



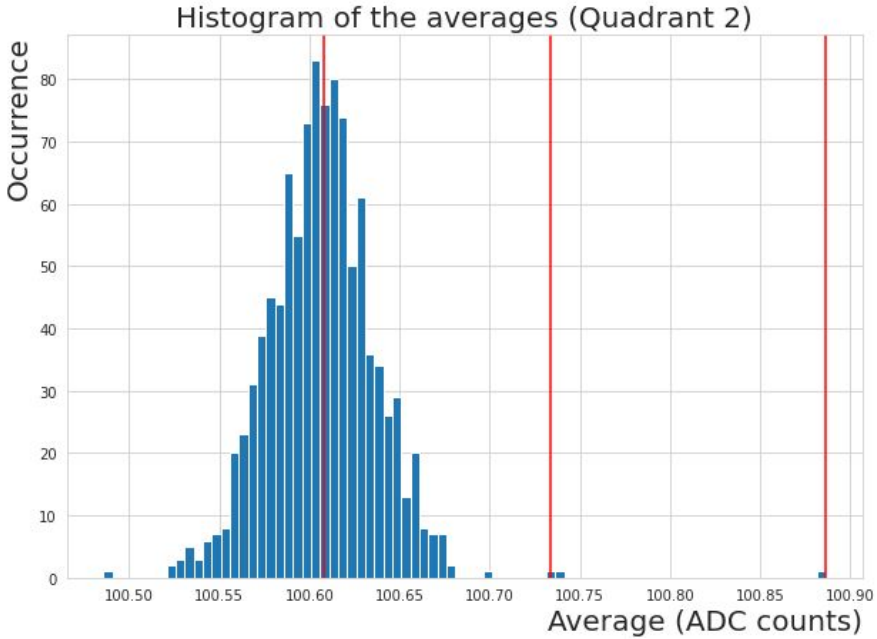


- The LNGS image (mean 100.93) contains far less sparks or hot pixels than the LNF (mean 100.64) one.
- It contains 5 pixels above the threshold of 200 ADC counts, whereas the LNF one contains 296.

LNF preliminary results

- A preprocessing was implemented to get rid of the sparks present in the images before the algorithm is implemented.
- Concerning the false alarm ratio, a total of 7 events were triggered without containing a signal present in the image.
- 88 events containing signal were detected with the algorithm.
- The signal detection efficiency has not been done yet.

Average distributions



- The average distributions cannot discriminate the signals as the standard deviations, as can be seen in this example, where the red lines represent the average of the subimage containing signal.