# Building the EUDAQ2 online monitor for LYCORIS Telescope

Lorenzo Cotrozzi
Università degli studi di Pisa
Italy

Supervisor: Dr. Mengqing Wu
DESY

September 4, 2019

**Abstract**

A six-plane beam telescope based on silicon strip sensors, namely LYCORIS, was designed and tested at the DESY II test beam facility. The EUDAQ2 data acquisition framework provides a monitor that generates and displays correlation plots, i.e. 2-dimensional histograms that select two different planes and match their strip coordinates that were hit for every event.
This report is to present the current status of the monitor for LYCORIS, both for its offline and online performance. More specifically, the newly implemented algorithms for the analysis of external trigger data were validated by an accurate study of LYCORIS's channels behaviour under different experimental conditions, and they are henceforth explained in this report. The correlation plots produced with data from the test beam facility in July/August 2019 now show a clear correlation between adjacent planes of the telescope.

*Keywords:* Silicon Strip Sensor, EUDAQ2 Monitor, Correlation plots, External trigger

# Contents

# 1 Introduction

## 1.1 LYCORIS Telescope

**LYCORIS** (**L**arge area x-**Y** **CO**verage **R**eadout **I**ntegrated **S**trip) is a new six-layer beam telescope, constructed as an improvement for the DESY test beam infrastucture within the EU AIDA-2020 project [1]. The telescope addresses the user demands for momentum measurement in a 1 T solenoid magnet, namely PCMAG, installed in area T24/1 of the test beam facility at DESY [2], providing a spatial resolution of $\sim 7.2\,\mu m$ along the bending direction. In figure 1.1 the telescope is shown inside PCMAG during a test beam setup: the six sensor planes are hosted by two cassette-like supports, one placed before the DUT inside the magnets (*upstream*) and one after the DUT (*downstream*).

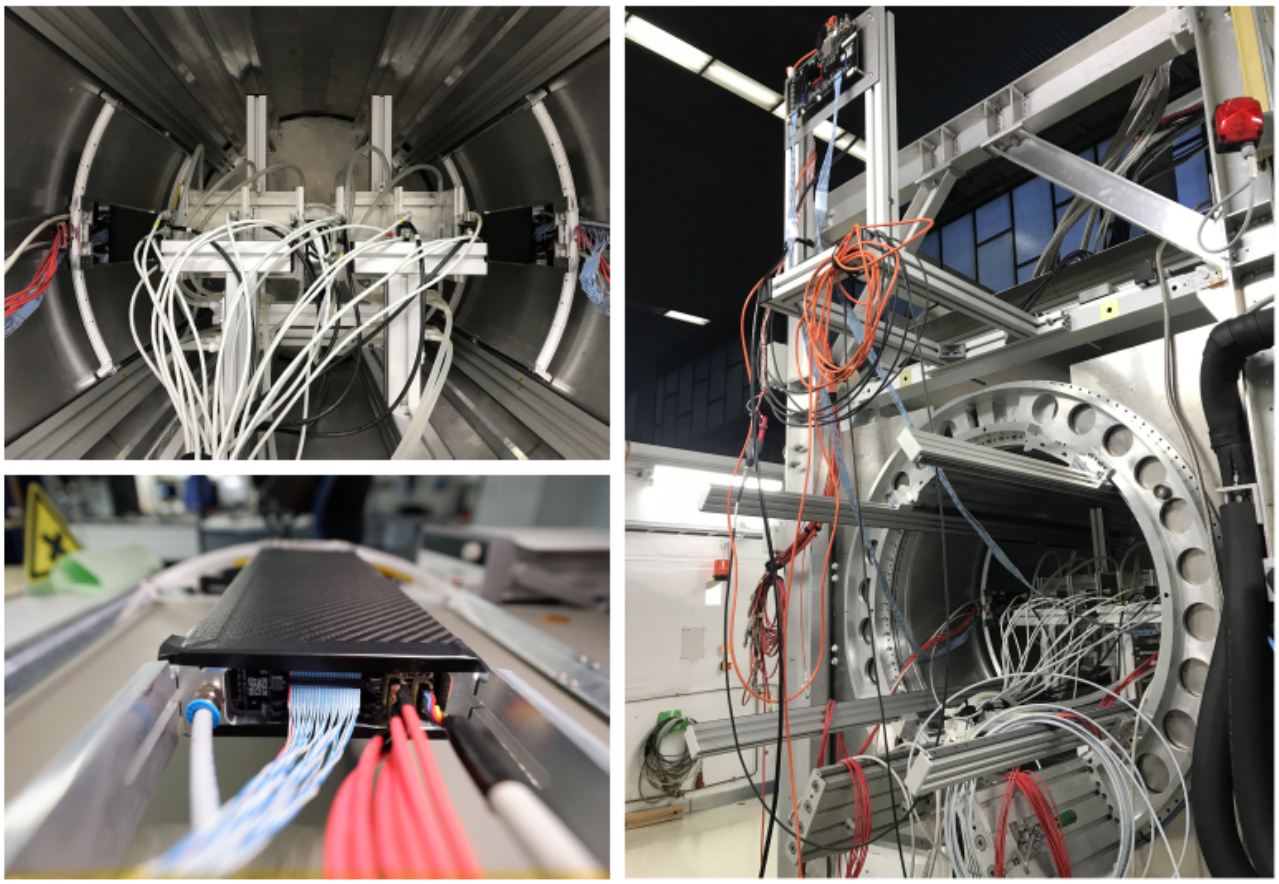

**Figure 1.1:** LYCORIS in the test beam facility. Upper left: upstream and downstream cassettes positioned inside the PCMAG, with MIMOSA telescopes in the between. Lower left: close view of one of the cassettes, already connected to DAQ and mounted on rail support. Right: overview of experimental setup in room T24/1.

The test beam facility at DESY provides an electron beam to Test beam area T24/1: energies go from 1 to 6 GeV; the DESY II cycle has a sinusoidal energy curve that repeats every 80 ms.

Each plane is 320 µm thick, $10 \times 10\,cm^2$ large; the sensor module has 25 µm sense pitch, and is readout by two bump-bonded 1024-channel KPiX [3] developed by SLAC. Not all 1024 channels are connected to readout strips: each sensor has 1840 readout strips, connected to two KPiX chips, i.e. each KPiX has 920 channels connected to strips.

Figure 1.2 shows the orientation of the x-y-z axes as defined in the textbeam facility. Notice the DAQ flow, and the TLU module that collects trigger events from the photomultipier. The ±2° angles are stereo angles needed for spatial resolution along the x-axis, they will be explained in more detail in section 2.

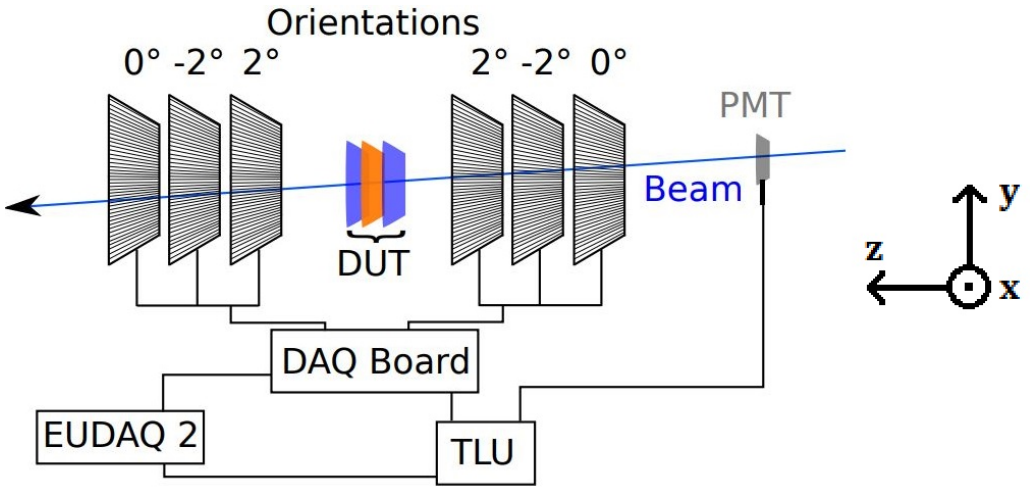


**Figure 1.2:** The definitions of the x, y and z axes is given: z-axis is the one along the beam direction; y-axis is along the strip coordinates of each sensor; the x-axis is the one coming out of the figure, it is the direction along which strips are positioned.

## 1.2 EUDAQ2 software architecture

EUDAQ2 is a data acquisition framework, written in C++, designed to be modulable and portable [5]: "A core library is utilized to manage the readout, data collection and steering. The core only depends on standard C++ functionalities, allowing for platform independent developments. Hardware specific code, a `Producer`, is linked to the core and can utilize more specific external libraries. Figure 1.3 shows the architecture split into different processes.
The `Run Control` is the central controller that manages all the processes, taking the ini/config files as an input; `Producers` are processes that communicate with the hardware, read out the data and send it to Data Collectors; `Data Collectors` are processes that collect the data flows from individual Producers and merge them into a single data stream, writing it to file stored in Disk. The file format is .raw, which means that, for our particular monitor, namely StdEventMonitor, a converter is needed to turn the binary raw event to a standard event format: then the monitor is able to read it and display plots out of it; the `Log Collector` displays log messages from all other processes in one unified logging window.

### 1.2.1 Online monitor for Telescopes

The `Monitor` reads the data file and generates online-monitoring plots for display. It can be run in online or offline mode: offline mode is for data which was collected beforehand and stored in disk; in online mode, instead, the monitor can connect to the Run Control, so it will know when new runs are started, it will automatically open when a new file is create and it will automatically update during the acquisition. For the purpose of this report, the most important plots produced by the monitor for LYCORIS are the correlation plots: these are 2-dimensional histograms where the strip coordinates of a sensor are displaced on the x-axis (which therefore ranges from 0 to 1839, as explained in section 1.1), the ones of another sensor

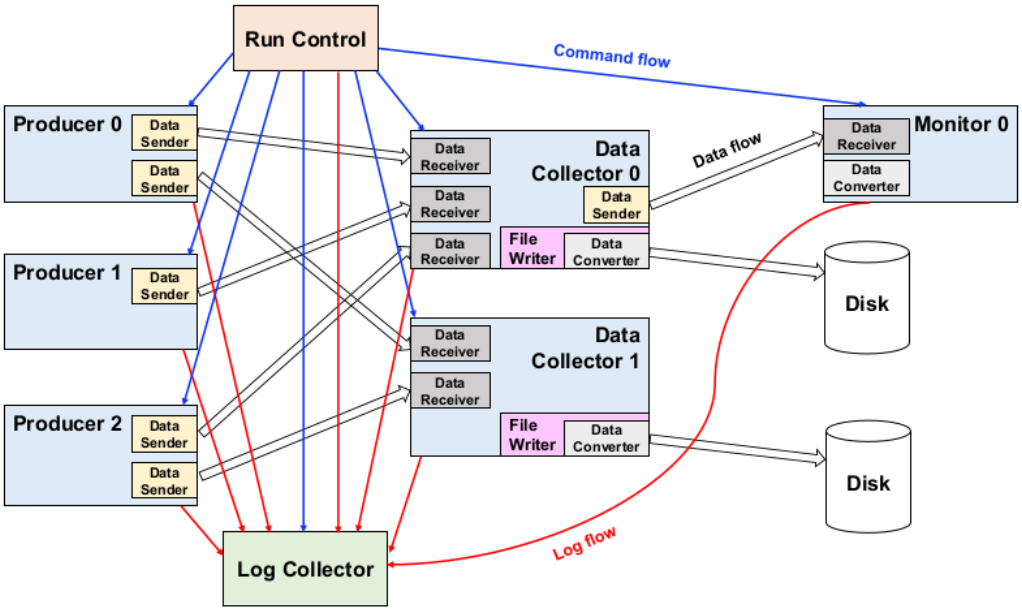


**Figure 1.3:** Schematic view of the EUDAQ2 architecture. Red arrows represent data flow. Run Control and Log Collector processes are omitted in this figure.

are instead on the y-axis, and, whenever there is signal during the data acquisition, the monitor matches the strips that were hit on both sensors and creates a new entry. Examples can be seen in figure 1.4 for MIMOSA telescopes.

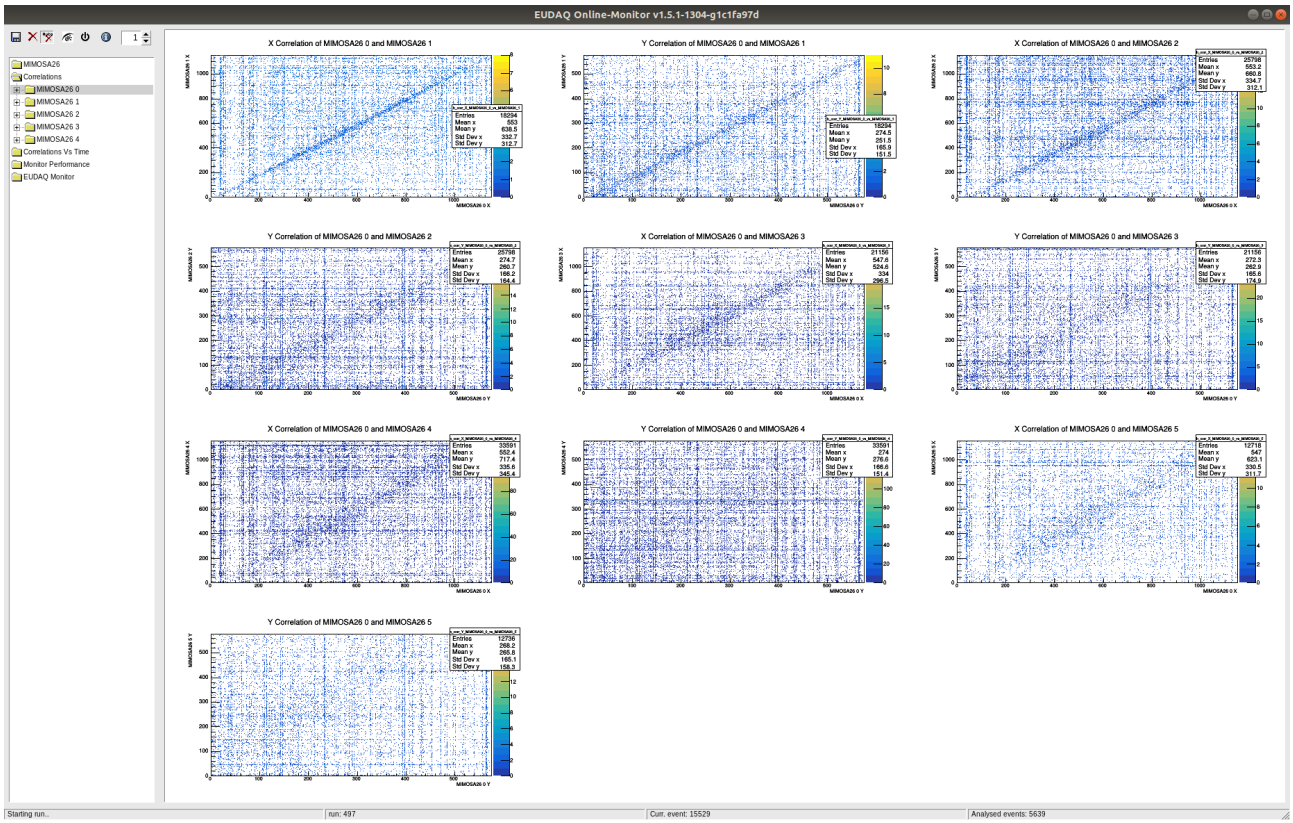


**Figure 1.4:** Screenshot of the online monitor for MIMOSA pixel telescopes. For all histograms: the labels on each axis are the strip coordinates of different sensors. These are defined as the "X" or "Y" coordinates.

Correlation plots are very important because of the underlying physics on which they are based. If beam particles travel perpendicularly to LYCORIS's planes, we will expect to see many events on the diagonal that stretches from the lower left corner to the upper right corner: signal hits should involve the same strip coordinates on different sensor planes, i.e. the particles are supposed to travel through LYCORIS's cassettes always at the same height, if the magnet is turned off and there is therefore no bending. In reality, the trajectory of electrons will not be in a straight line, since multiple scattering can occur inside planes, so the distribution of events near the diagonal line in correlation plots will have a wider spread. If, instead, the magnetic field is turned on and the particles of same energy are bent with the same curvature, we still expect to see a diagonal in correlation plots, parallel to the one without any bending, but with a different offset, i.e. not passing through the zero coordinates of both sensors.

Diagonals are seen quite clearly in figure 1.4, but at the beginning of the summer school they could not be seen for LYCORIS with the online monitor: this is because of the type of trigger mode used for acquisition, which will now be explained in more detail. As the KPiX was designed for ILC environment, the chip performs power pulsing, it is therefore not active all the time but just for a short period, $\sim 20\,\text{ms}$, for each acquisition cycle. As stated in section 1.1, the DESY II cycle follows a periodic, sinusoidal energy curve: the telescope's acquisition cycle can be synchronized to this curve, because DESY II sends a minimum energy signal that works as external start-up signal [6]. Then there are two trigger modes:

- Self (or internal) triggering: the charge collected in a certain channel is collected only if it exceeds a user-defined threshold;
- External triggering: all charges in all channels are collected, as soon as an external signal - which is supposed to indicate the passage of a particle - is fed to KPiX.

The immediate consequence of external triggering is that the charge collected by most of the strips will be just noise, whereas only a few strips will have recorded a proper signal. The chase for Landau distribution, expected for MIP particles, will require algorithm for signal/noise selection, and will be the study in sections 3 and 4.

### 1.2.2 StdEventMonitor architecture schematic view

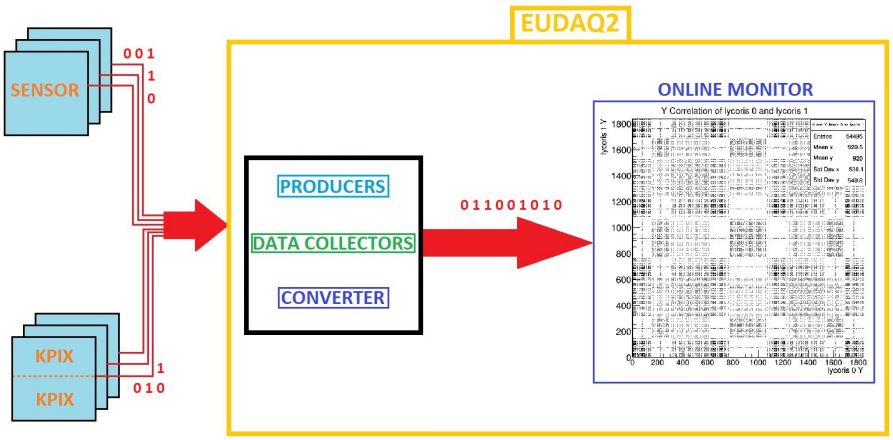


**Figure 1.5:** EUDAQ2 flowchart for LYCORIS.

# 2 Converter Source Code

As stated in section 1.2, data in .raw format cannot be read by our StdEventMonitor, but has to be converted into StdEvent first.
The source code for our converter is "kpixRawEvent2StdEventConverter.cc", written by Dr. Mengqing Wu on the 20th of June 2019, based on the new KPiX DAQ for LYCORIS telescope. The code includes some ROOT header files to be able to work with histograms, trees and files. It also includes "kpix_left_and_right.h", which is simply a lookup table so that the conversion from channel ID to strip ID for every sensor is possible.

The converter first receives the .raw file as an input and checks if it is empty or not. It is supposed to recognize if it is data from external trigger or self trigger runs, but for the moment this option has to be hardcoded with a bool variable. The other two inputs are a configuration file and the file where the StdEvent is supposed to be saved.
The converter then defines the StandardPlane as if it were a pixel plane with 1840 pixels in y-axis and only 1 pixel in x-axis, so basically it reproduces a strip sensor. After this, the converter calls the "parseFrame" function: a Frame is defined as 1 acquisition cycle, containing multiple data Samples; samples contain data collected by channels, they can be stored in 4 different buckets (i.e. memory cells) but only the bucket 0 is used by the converter.

We will now explain what parseFrame does for self trigger runs; the algorithm for external trigger is different, it will be explained in section 3 and its implementation with a temporarily workaround will be motivated and validated in section 4. For self triggering, the function basically loops over all samples inside the frame, to look for hits in each plane and push them to the plane already defined by the converter. During this loop, the function calls "parseSample", which is a simple function that looks inside a sample and return the following things:

- the KPiX number, which goes from 0 to 11;
- the channel number, which goes from 0 to 1023;
- the strip number, which for each KPiX goes from 0 to 919 and is obtained by the channel number by means of the lookup table "kpix_left_and_right.h";
- the charge collected by the strip.

As regards the charge: the KPiX readout is not binary, so ADC values have to be converted to fC; ADC/fC slopes are different for every channel, so a calibration file is needed and has to be included as well, to work as a lookup table. It should be stated that data acquisition can be performed in two different gain modes: normal and high gain. This results in different slope values for each channels, typical values for ADC/fC slopes in normal gain are $2 \sim 5\,\mathrm{fC}$, whereas for high gain they are $10 \sim 20\,\mathrm{fC}$.

So now parseFrame knows, for every iteration inside the loop, the number ID of the sensor, given the KPiX number. The conversion from KPiX number to PlaneID is possible if the configuration is known. The converter code uses the configuration in figure 2.1 as a reference.

Finally, the converter considers the orientation of sensors inside the cassettes: as seen in figure 2.1, for each cassette there are two sensor planes oriented in one direction and one in the opposite direction. There are mechanical reasons why this happens: a stereo angle of 2° is needed for some planes, so that not only the strip coordinate (y-axis) for every hit is recorded, but also the position along the x-axis, which means along the strips, can be derived with a spatial

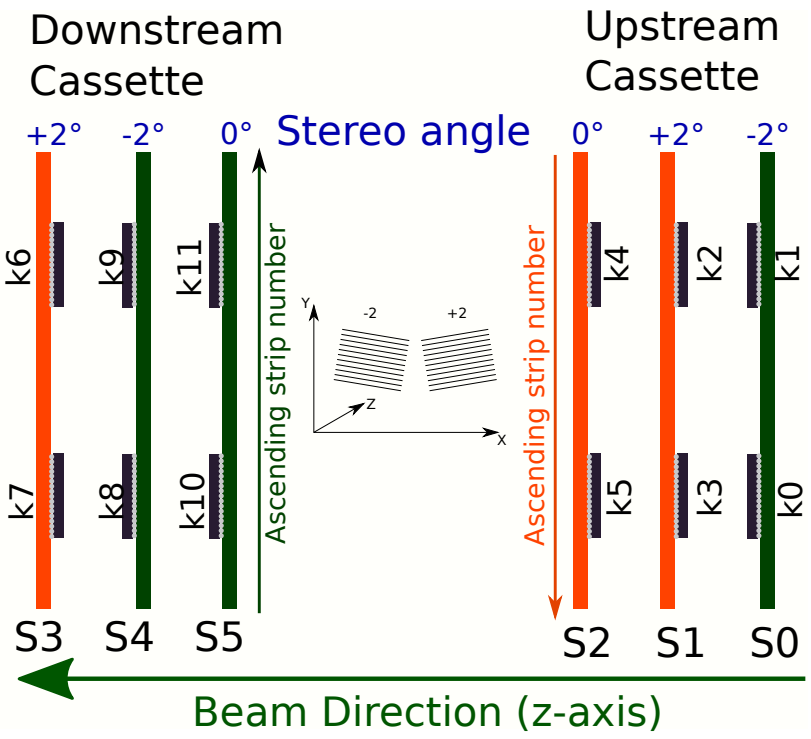


**Figure 2.1:** Test beam parametrization as of July 2019. S# stands for the number of sensor plane, ranging from 0 to 5; k# stands for KPiX number, ranging from 0 to 11 [9].

resolution of $\lesssim 1\,\text{mm}$. Each sensor plane is inside its own support: this can be placed in the cassette with a 0° or a 2° stereo angle, depending on which hole of the support is chosen. To obtain a $-2°$ stereo angle, the whole sensor has to be reverted upside down, and the 2°-hole has to be chosen. The converter takes this into account and chooses to revert the strip coordinates of all sensors marked in red in figure 2.1: in this way, at the end, it is as if all sensors had the same y-axis definition, with strip coordinates starting from 0 at the bottom and finishing at 1839 at the top, like the green ones in figure 2.1 already have.

At this point, the loop inside parseFrame is ready to push the coordinate of the strip that was hit for each sample. For self trigger runs, it is also possible to match only events which occur within a specified amount of time. For external trigger runs, instead, the parseFrame loop has to do even more, because it has to take into account the fact that most charges collected are noise and not signal, as already stated. The algorithm will be fully explained in section 3.

# 3 Offline data analysis algorithm

This section explains how the offline external trigger data analysis algorithm works.

## 3.1 Pedestal subtraction

When performing an external trigger run, as soon as an external signal is sent by the TLU, all charges collected on all channels are stored. This means that, together with the signal, most of the data collected will be just noise. So an algorithm to select signal has to be implemented, and this requires a series of different steps. The very first step comes from looking at the charge distribution on a single channel (for example, figure 3.1) and realizing that it is not centered in 0 fC, but typically around $40 \sim 60$ fC. This is due to the pedestal level, that is an intrinsic noise level that depends on the electronics; so it is expected to be different from channel to channel because of the different circuitry.

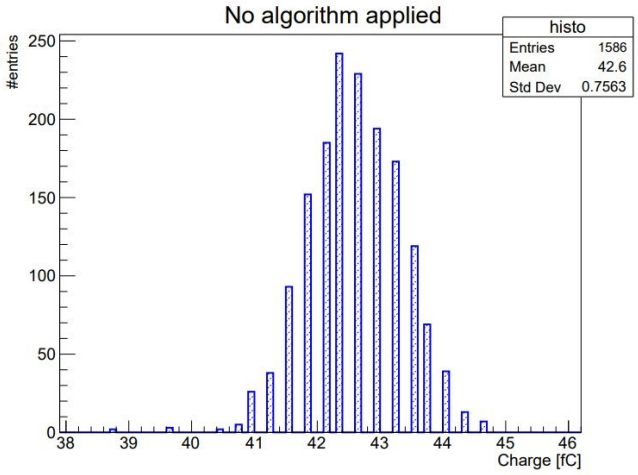


**Figure 3.1:** Charge distribution of KPiX 4 - Channel 20 before any algorithm is applied.

So, firstly, the charge distribution has to be corrected by subtracting the pedestal level, which results in the noise distribution centered at 0 as expected. The strategy for pedestal level determination is the following:

1. For each channel of each KPiX, a std::vector<double> is created - for a total of $12 \times 1024 = 12288$ vectors. Each is meant to be the vector of all charges in fC collected by a particular channel, which is why it is defined to contain double-type values;
2. While looping over all cycles of the .raw data file, the values of charges collected by each channel are pushed in the vectors. At the end, all vectors have the same length;
3. After the loop, a median of the charges is evaluated for each vector. This value is therefore called "pedestal median" and has to be subtracted from each entry of the vector;
4. Also the Median Absolute Deviation, or MAD, is evaluated. MAD is simply the median of the distribution of the absolute difference between values in the vector and the vector median.

The MAD gives therefore an indication of how spread the charge distribution is. Typical values for MAD for LYCORIS are around $0.3 \sim 0.6$ fC, but it is known that some channels have MAD

equal to zero, indicating that the measurement is not reliable (because at least some electric noise would be expected) so the algorithm for analysis is asked to cut on such channels. Finally, median and MAD are more robust estimators than mean and standard deviation, because they are less dependent on outliers.

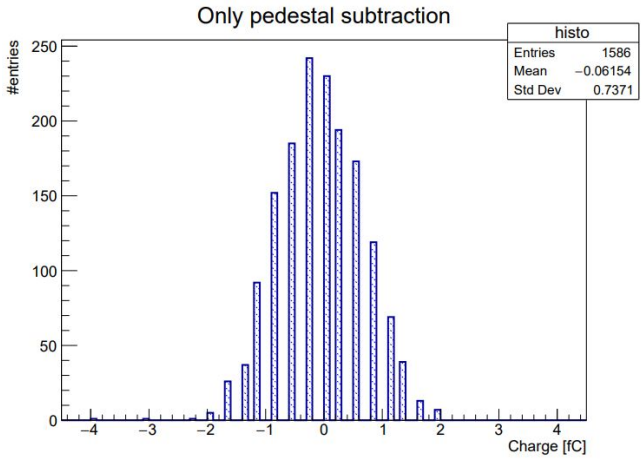


**Figure 3.2:** Charge distribution of KPiX 4 - Channel 20 after pedestal subtraction.

Figure 3.2 shows an example of the charge distribution on a single channel after pedestal subtraction: it is clear from the x-axis range that the distribution was shifted much closer to zero than it was before (figure 3.1). But a further correction is still needed, since the so-called "common mode noise" affects the distribution. The source of this additional, intrinsic noise level is the power cycling of the sensors: all channels of a KPiX share the same front end electronics, so this type of noise coherently affects their charge readout [8]. The algorithm has to consider, for each KPiX and for each different cycle, the charge distribution - already subtracted by the pedestal level - of all 1024 channels, take a median out of it and subtract it from the charges collected.

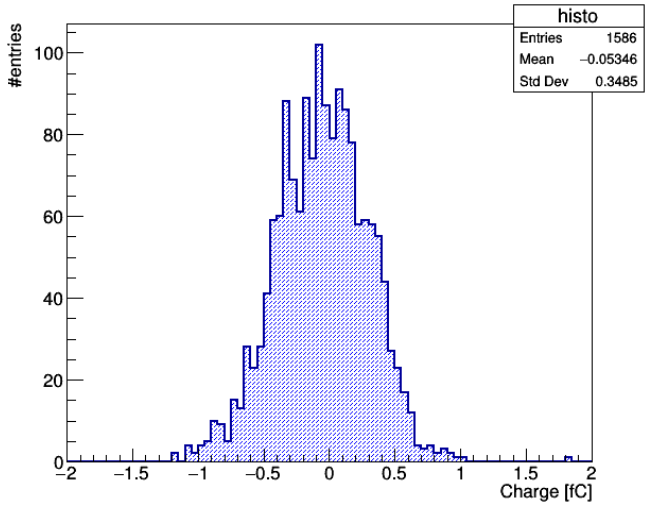


**Figure 3.3:** Charge distribution of KPiX 4 - Channel 20 after pedestal and common mode noise subtraction.

Figure 3.3 shows an example of the charge distribution on a single channel after both pedestal and common mode noise subtraction: now the noise distribution is centered at 0 fC, which opens

the possibility to evaluate the significance of hits, with the algorithm explained in section 3.2.

## 3.2 Signal significance selection

First of all, figure 3.4 shows how it is possible to make a gaussian fit on the charge distribution of a single channel over all cycles. As most of the charge is noise, the noise level is defined as the sigma of the gaussian distribution: $\sigma_{noise}$.

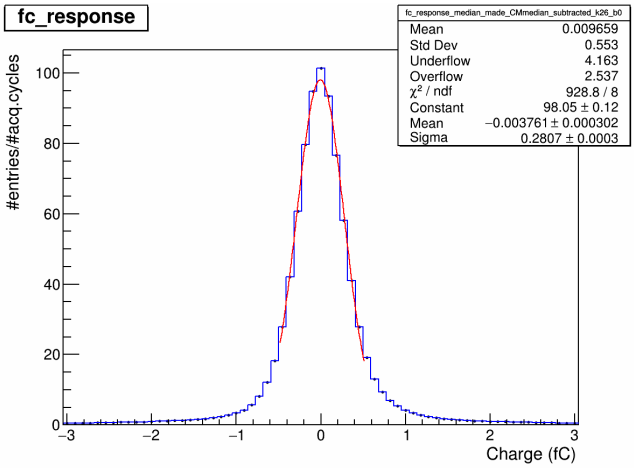


**Figure 3.4:** Noise distribution, gaussian fit.

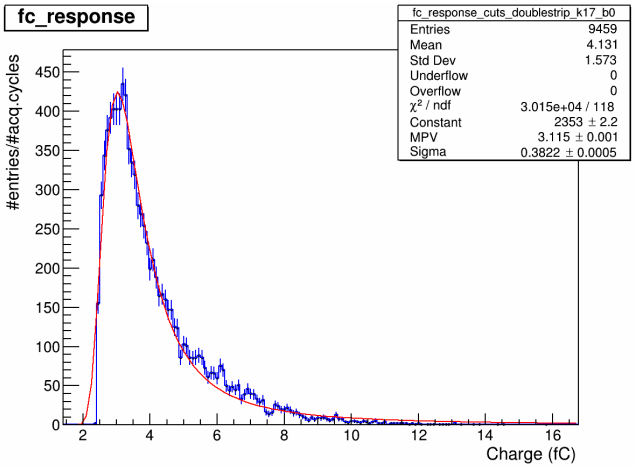


**Figure 3.5:** Signal distribution, landau fit.

The algorithm used for offline analysis does not need to perform any fit procedure, because, for the gaussian distribution, a robust estimator for $\sigma_{noise}$ comes from the newly obtained charge distribution MAD, according to equation (3.1) [7]:

$$\sigma_{noise} = 1.4826 \times \text{MAD} \tag{3.1}$$

At this point it is possible to evaluate the significance of every hit, which is the ratio between the charge and $\sigma_{noise}$. Then "good strips" are found based on a simple hypothesis test. Figure 3.5: according to previous study on this project, one can get an expected Landau distribution out of channels inside beam area. The selection for such events against noise events is S/N ratio above 3.

# 4 Online data analysis algorithm

The algorithm explained in section 3 was only suitable for offline data analysis, where it is possible to loop over all data samples and take a look at the overall charge distribution, subtract the pedestal and common mode noise level and perform cuts on noise, based on the evaluated $\sigma_{noise}$; however, it is not possible to implement the same algorithm in online data analysis, because the pedestal median and noise level would have to be updated on the fly, for every new cycle, but currently the converter only processes 10 events at a time, and can not store the previous event information nor update the plotted events. For the time being, a workaround was thought of. The idea is to proceed as follows:

1. At the beginning of data acquisition, a preliminary run should be taken;
2. The data from the first run should then be analyzed offline: for each channel, pedestal levels and noise levels (from pedestal MAD) should be stored in a .root tree file;
3. For the following runs, subtract from the charge distribution the pedestal median calculated from the preliminary analysis. Then subtract the common mode noise - this can easily be done on the fly, because it depends uniquely on the charge distribution of each sensor for the current cycle;
4. Perform the signal/noise selection based on $\sigma_{noise}$ evaluated from the preliminary run. Only parse "good hits" to the monitor.

This procedure can be implemented in the converter, but it clearly requires the strong assumption that pedestal median and noise levels do not change between runs. The purpose of this section is to prove that, even under different experimental conditions, within certain limits, said assumption holds for most of the channels and it is reasonable to implement this temporarily solution.

## 4.1 Pedestal database

In this section I will present my studies on pedestal median: how it changes over different runs; how the behaviour of a single channel can be classified as good or bad; most importantly, I will prove that the preliminary run workaround can be safely applied.

During July test beam campaign, many external trigger data files were taken, in many different configuration. In this study I selected a small, random sample of eight different channels from the telescope; for each one, I plotted the distribution of the pedestal median over 13 different runs, in which all experimental conditions were the same except for the bias voltage applied to the silicon sensors (that varied in the range between 50 V and 150 V). The results are shown in figure 4.1.

In figure 4.1, the x-axis is always the same for all channels, ranging from 0 to 100 fC. The y-axis is the number of runs in which a certain pedestal level was observed for that channel. Different histograms have differently shaped distributions of pedestal level: some channels have a pedestal level that hardly changes between runs, whereas for some other channels the pedestal level varies significantly - which is why their histograms are highlighted in red.

For each distribution of pedestal level, a box is provided with the following information:

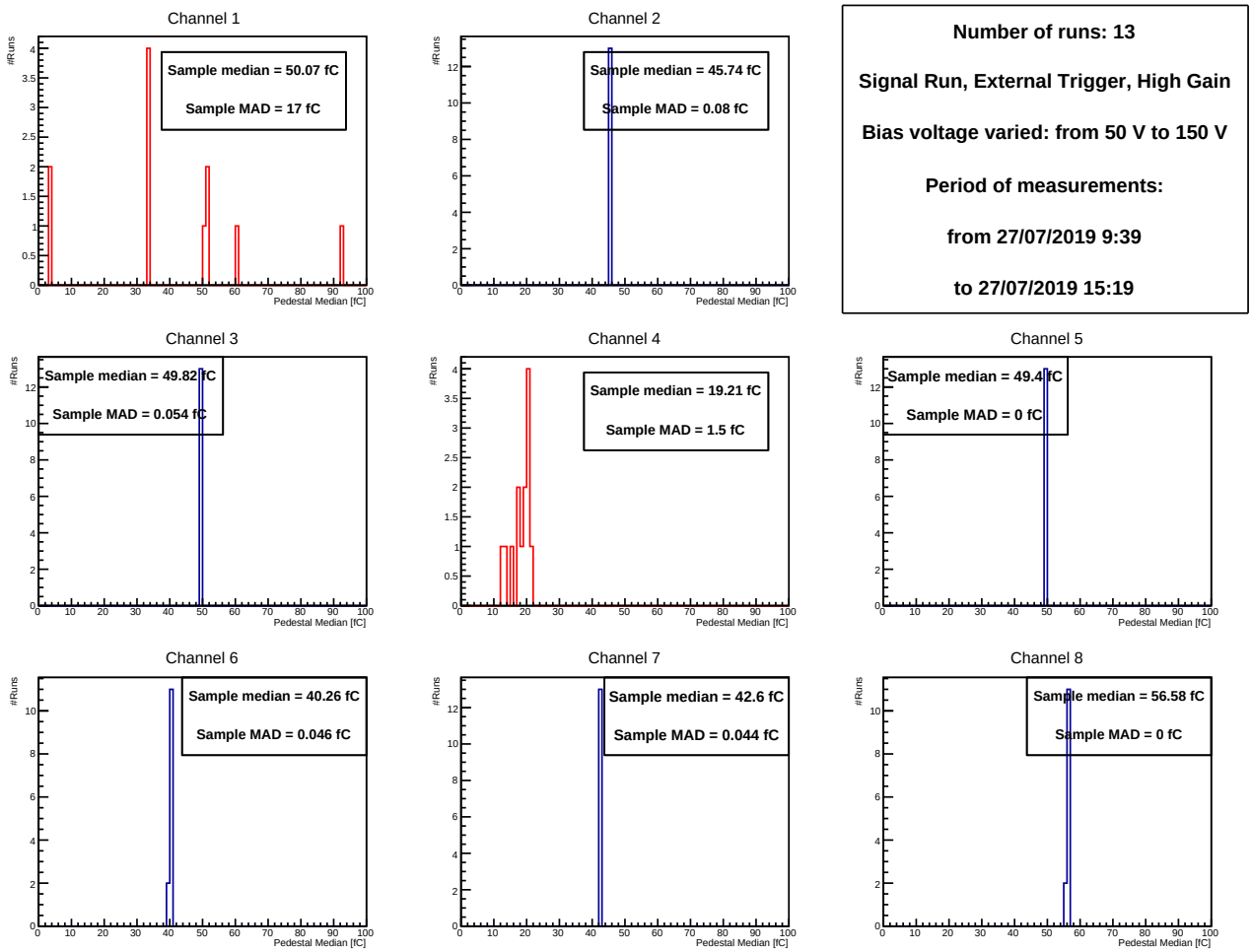


**Figure 4.1:** Pedestal median distribution for eight different random channels.

- Sample median: a median is evaluated out of the distribution of pedestal level over different runs. The reason why the median was chosen rather than the mean is, again, to avoid the influence of outliers. If, out of 13 runs, a channel presents the same pedestal level in the majority of cases, and only has significantly different pedestal levels in very few cases, the sample median does not penalize it too much;

- Sample MAD: the MAD of the distribution is also evaluated. It is, again, an indication of how spread the distribution of pedestal level is for each channel.

What clearly distinguishes blue channels from red channels is their "Sample MAD": we can notice that blue histograms have a sample MAD far less than 1 fC, whereas red histograms have a sample MAD that can go from $\sim 1$ fC up to $\sim 20$ fC or, presumably, even more. So a quick way of deciding how many channels are well-behaving, as opposed to the ones who significantly change their pedestal level between runs, is - rather than checking one by one the 12288 possible channels - to plot histograms of the sample MAD distribution over all channels. The result of this study is summarized in figure 4.2. For this plot, a random cut at 0.5 fC was chosen, but it is evident that any cut between 0.2 fC and 0.5 fC would not produce much difference.

Figure 4.2 shows that only $\sim 3\%$ of the channels have a bad behaviour under different run conditions (here, when applying different bias voltages). This means that the temporarily solution for pedestal level determination is actually suitable for our goals.

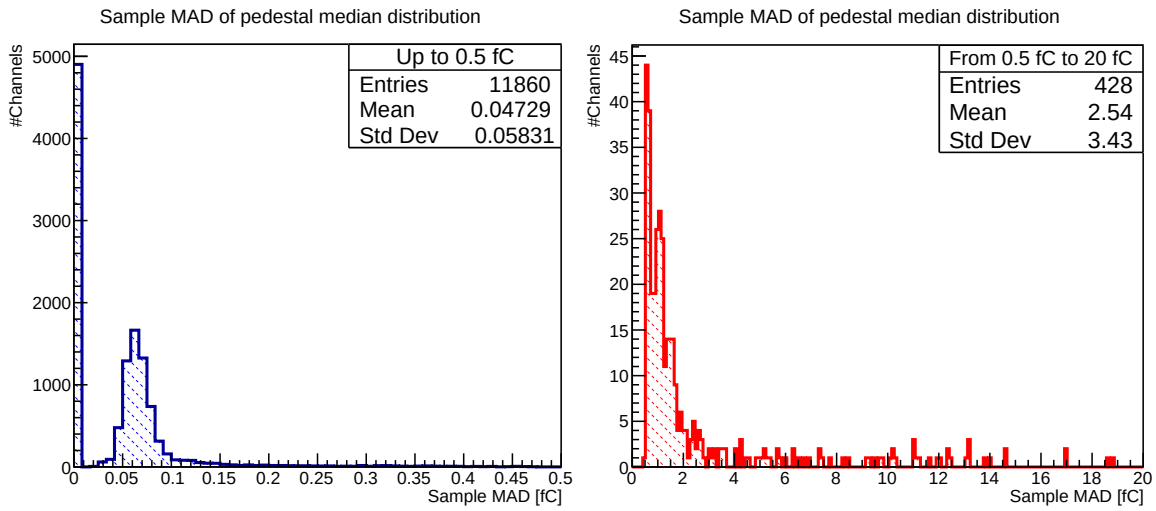


**Figure 4.2:** For each of the 12288 channels, the sample MAD of pedestal level distribution over 13 runs is calculated and fills the histogram.

## 4.2 Noise level determination database

The next step is to check whether the noise level, $\sigma_{noise}$, evaluated after common mode noise subtraction, changes significantly between runs. The same random sample of eight different channels from the telescope was considered; for each one, the distribution of noise level over runs was plotted. The results are shown in figure 4.3.

In figure 4.3, the x-axis is always the same for all channels, ranging from 0 to 3.5 fC. The y-axis is the number of runs in which a certain noise level was observed for that channel. Again, channels which have a bad behaviour are highlighted in red. For this study, two different variables can be used to discriminate between blue and red histograms: firstly, the sample MAD of blue channels are mostly below 0.01 fC; secondly, the sample median of blue channels is less than 1 fC, whereas for the red channel the sample median is significantly higher. It is therefore interesting to see the overall distribution of sample MAD and sample median for noise level over different runs, in a 2-dimensional histogram. Figure 4.4 shows the result of this study, with a logarithmic scale on z-axis to have a better overview of the distribution.

The result of this study is satisfactory: $\gtrsim 90\%$ of the channels are "good", which means that they belong to the region where their sample MAD is less than 0.5 fC and the sample median is less than 1 fC.

Combining together the results of this section, we can safely apply the pedestal database and noise database from a preliminary sample to new runs.

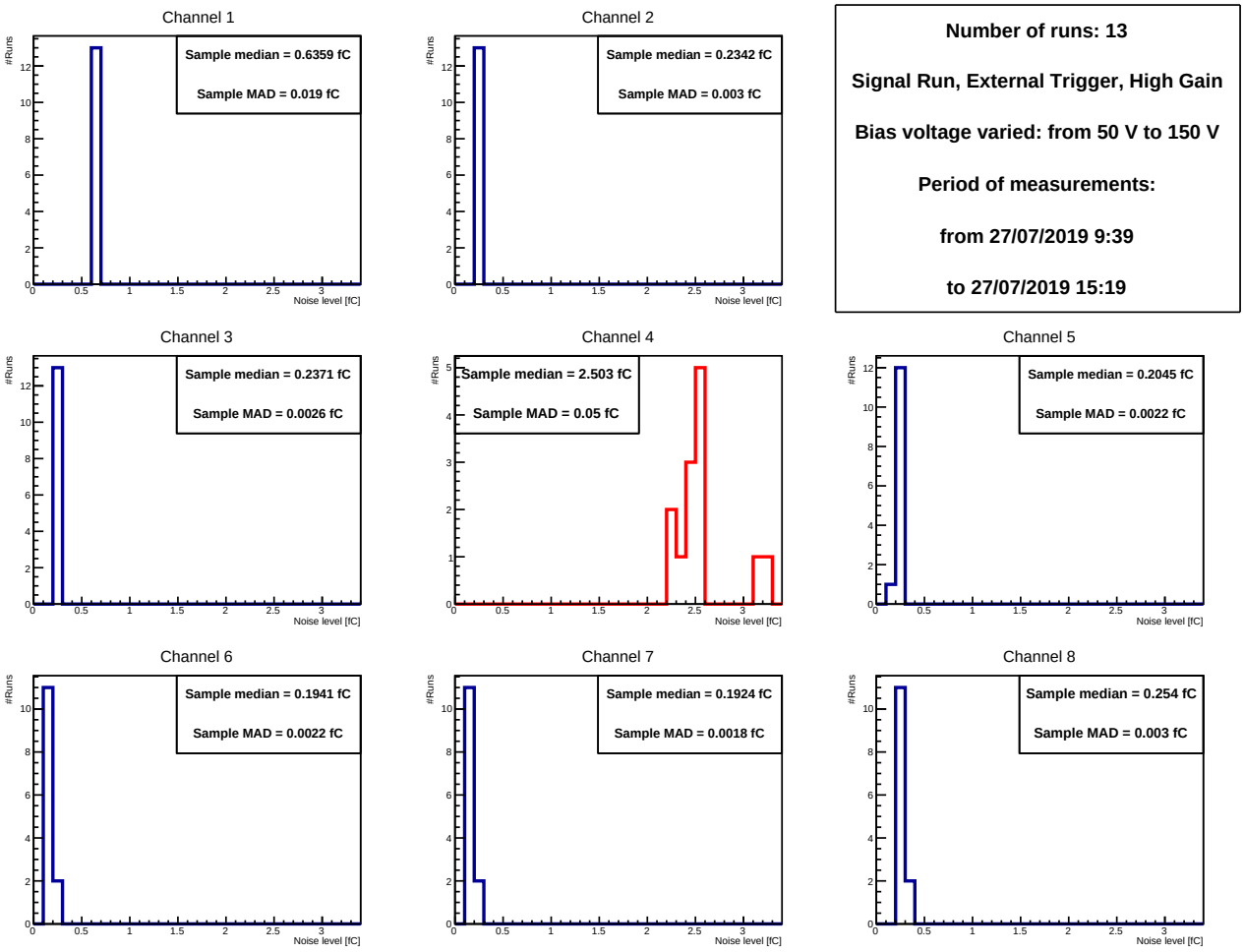


**Figure 4.3:** Noise level distribution for eight different random channels.

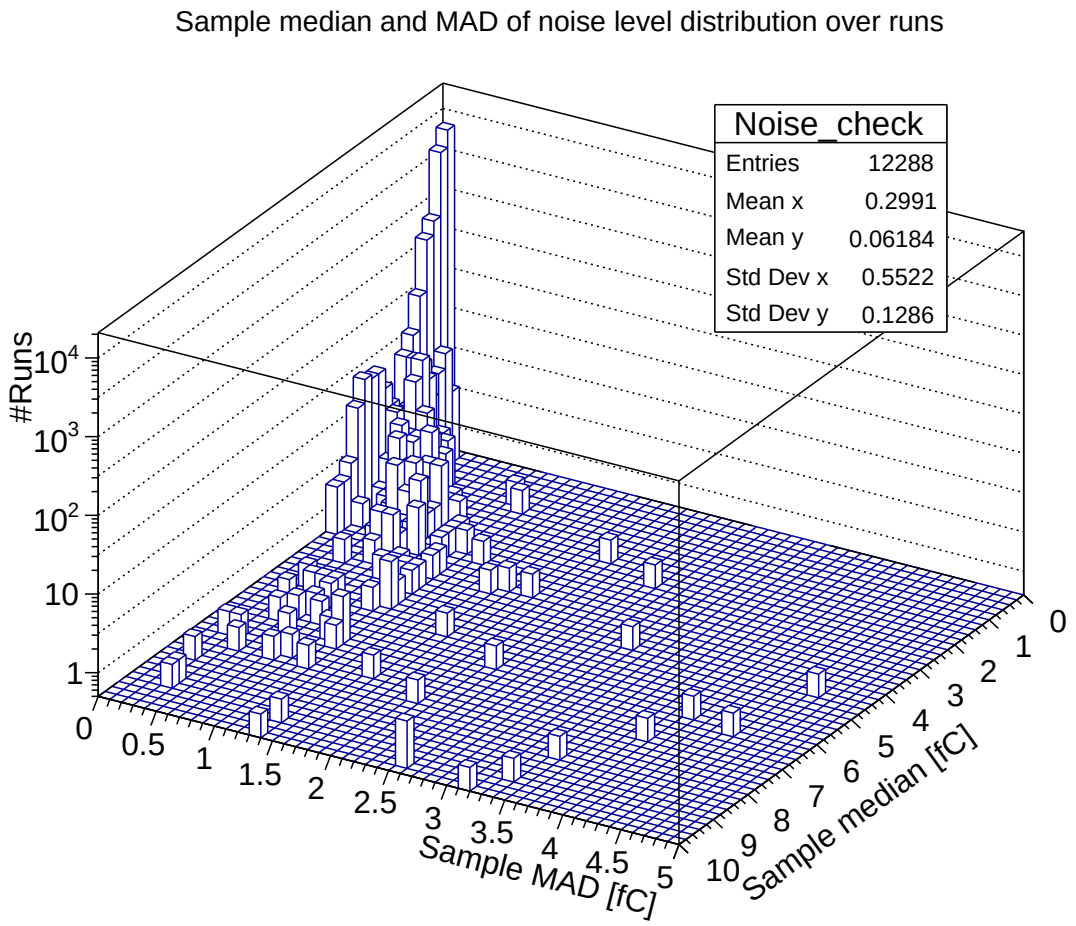


**Figure 4.4:** For each of the 12288 channels, the sample MAD and sample median of noise level distribution over 13 runs is calculated and fills the histogram. The result is presented as a lego plot with logarithmic scale on the z-axis.

# 5 Results and discussion

## 5.1 July 2019 test beam campaign

Since results from section 4 proved that the online monitor now can perform pedestal subtraction and signal/noise selection on the fly, thanks to a database, I will now present the current status of the monitor when ran over external trigger data from test beam runs in July 2019. This campaign aimed to test LYCORIS performance inside the PCMAG, with the aid of MIMOSA telescopes placed between the upstream and downstream cassettes, as was shown in figure 1.1. The displacement of the six sensor planes is shown again in figure 5.1.

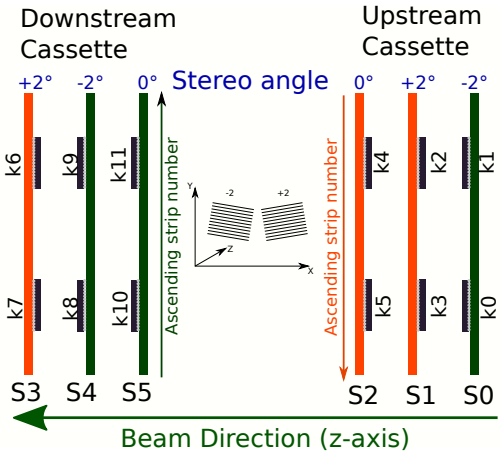


**Figure 5.1:** Test beam parametrization as of July 2019. S# stands for the number of sensor plane, ranging from 0 to 5; k# stands for KPiX number, ranging from 0 to 11 [9].

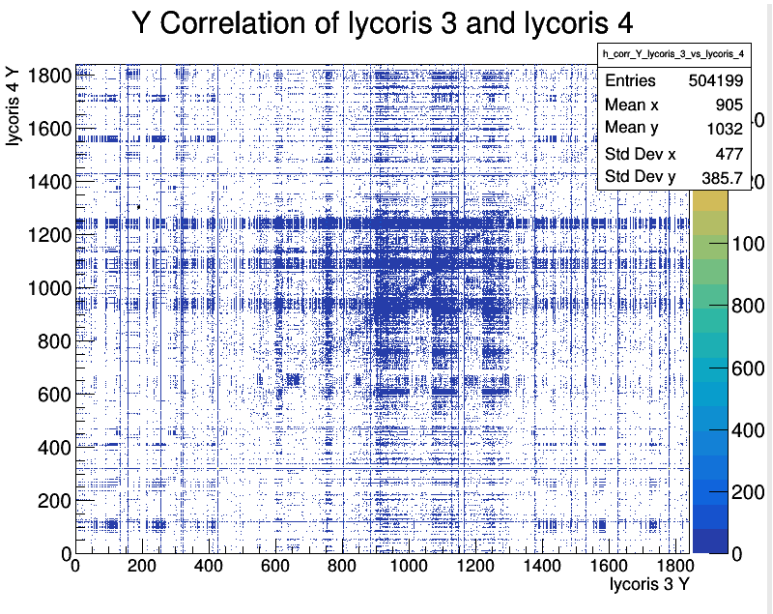


**Figure 5.2:** Current status of the online monitor. A clear correlation is seen between sensor planes when a cut of S/N ratio above 3 is applied.

Figure 5.2 shows the results of the current version of correlation plots produced by online monitor for external trigger runs: the expected diagonal that stretches from the lower left corner to the upper right corner is clearly seen.

## 5.2 Summary

I have worked on delivering a telescope online monitor prototype which plots correlations between sensor planes. From this work, I was able to adapt the offline data analysis algorithm for the online monitor software framework. I have demonstrated that the pedestal value and the noise level of the sensor do not vary too much between different runs with similar experiment conditions.

### 5.2.1 Real data taking experience

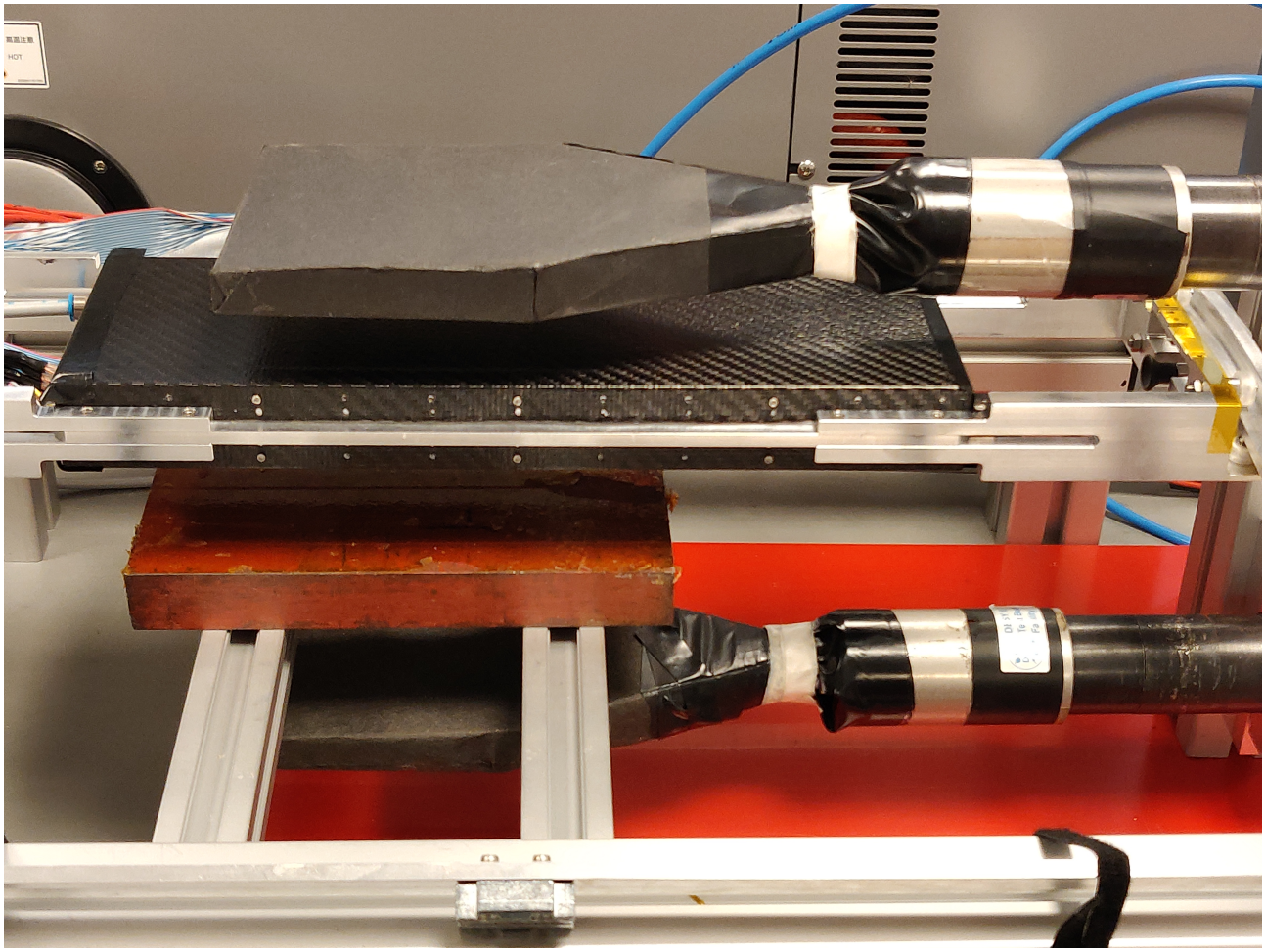

**Figure 5.3:** Setup in the eLab. Scintillators are used to provide external trigger, lead block is placed to select hard muons.

In terms of hardware experience, I have participated to the July test beam campaign and helped to setup a cosmic ray run setup in the FH E-lab. Here I will add two photos to show what the setup looks like: figures 5.3 and 5.4.

## 5.3 Outlook

It is clear that, even though our prototype has undergone many improvements, several improvements are still needed. In this section, the most important ones - that will presumably be implemented before future tests in online mode with test beam - are discussed.

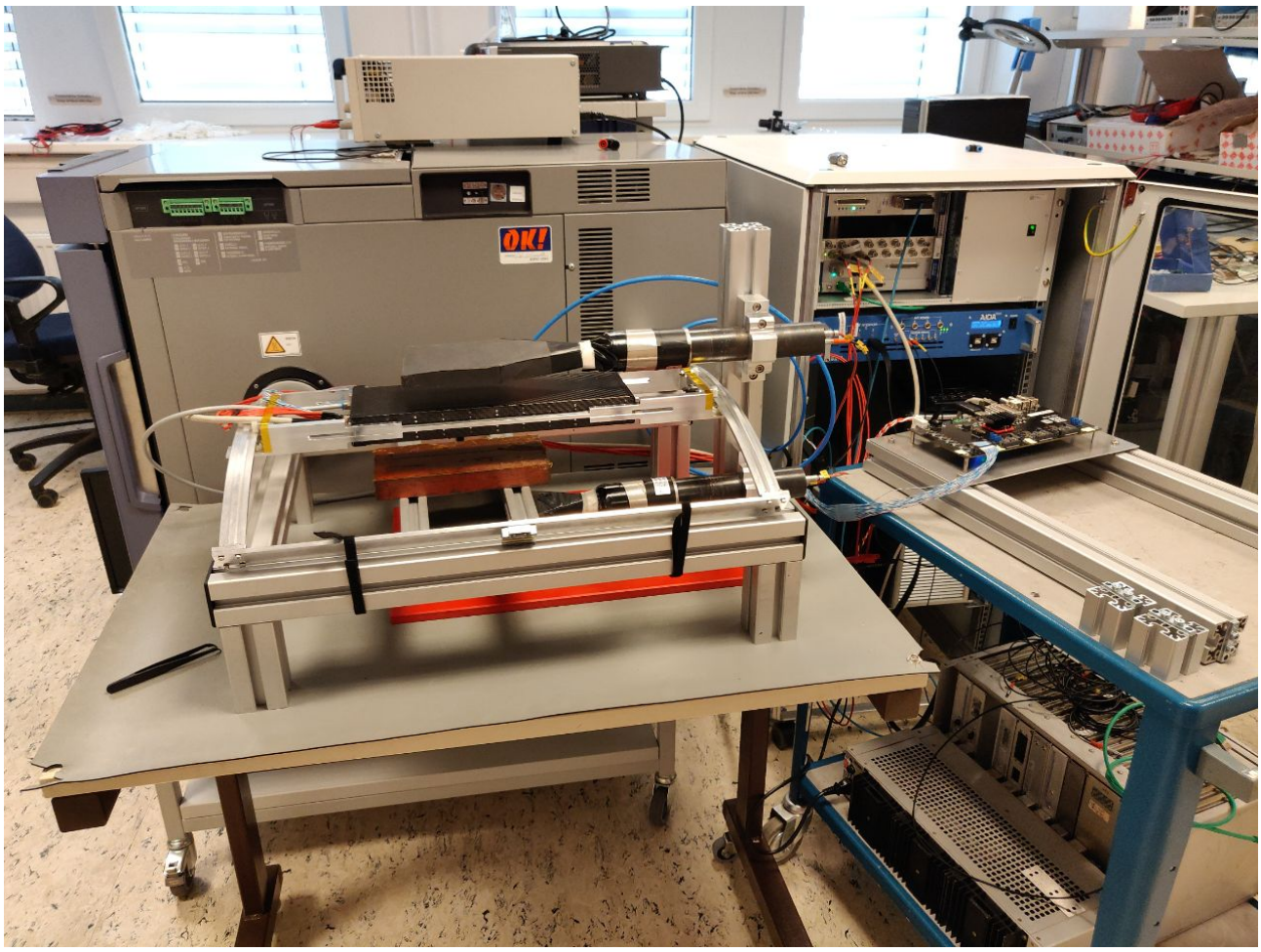


**Figure 5.4:** Setup in the eLab. The telescope and a lead block are placed between two scintillators; photomultipliers are powered with 1150 V; the DAQ board can be seen on the table on the right, and the Trigger Logic Unit, or TLU, is the blue module on the right.

## +++ Online Data Analysis +++

In section 4, a workaround for the implementation of the online analysis algorithm was presented, motivated and validated. It has been shown that the temporarily solution could work, but it would be better and it would save much time if the monitor were able to perform the pedestal subtraction and noise level determination properly on the fly, i.e. being able to handle a running median instead of using a database.

If, however, the temporarily solution were to be adopted in the long term, the studies on channel behaviour should definitely be improved. For instance, referring to section 4:

- Plots such as the one in figure 4.2 could be improved: a different cut could be applied, perhaps lowering it to 0.2 fC could make sense, since this value is comparable with typical noise level for "good" channels - that is, the ones in blue in figure 4.3;
- Channels with pedestal MAD equal to 0, i.e. those who do not even seem to collect electric noise in their charge distribution, should be excluded from the studies on pedestal and noise level databases, since those channels would be excluded by the algorithm anyways. It should also be checked how many of them contribute to the large zero bin in figure 4.2;

- Throughout all studies for pedestal and noise level databases, the only experimental condition that was varied between the 13 chosen runs was the bias voltage applied to the sensors. Of course, other situations were quickly tested and the results are shown in the Appendix. It could be interesting, for example, to compare results between magnet on and magnet off data.

### +++ Hit Clustering +++

Currently, the StdEventMonitor does not have any algorithm for Hit clustering, but we plan to implement it in the future. The following process should be done for each different acquisition cycle, for each sensor:

1. Within an acquisition cycle, the collected charges of all 1840 channels of each sensor plane are considered. The charges have already been corrected with pedestal and common mode noise subtraction, and both KPiX of each sensor are being considered at the same time. Hit candidates are to be individuated based on the signal/noise selection;

2. Cluster seeds have to be chosen amongst all hit candidates: the ones with highest significance are chosen, but a cut is applied on charge value, because we want to avoid hits from accidental spikes;

3. a loop over all the seeds is performed, to group neighboring strips starting from the cluster seeds and ending when no more "good hits" with significance lower than the seed are found.

4. The position of the center of gravity has to be obtained by weighting the position of each strip with its charge value; this center of gravity will then be the position of the cluster.

### +++ New control information +++

The Online Monitor could be configured to include some slow control information, such as Humidity, Temperature and Bias Current of the sensor planes. These variables are already present in .raw data files but not currently used.

# Appendix

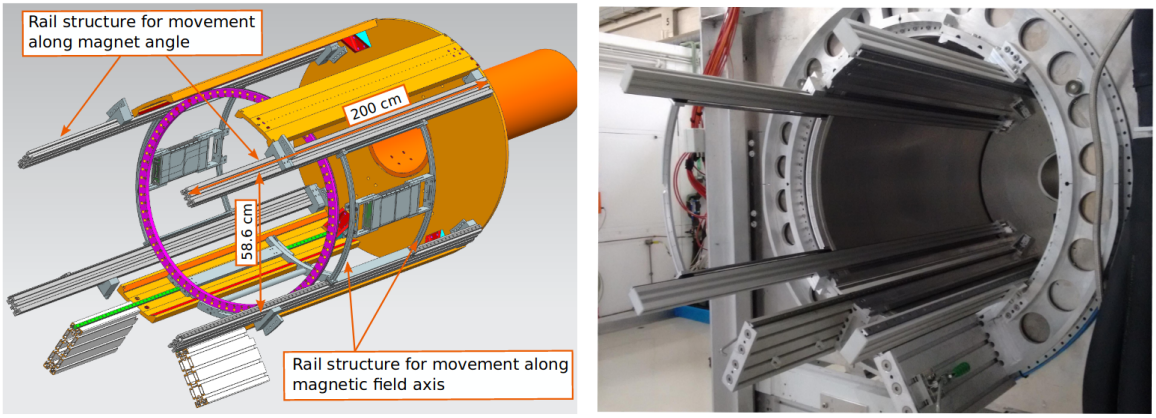


**Figure 5.5:** Rail structure for movement along magnet angle is relevant to our further studies.

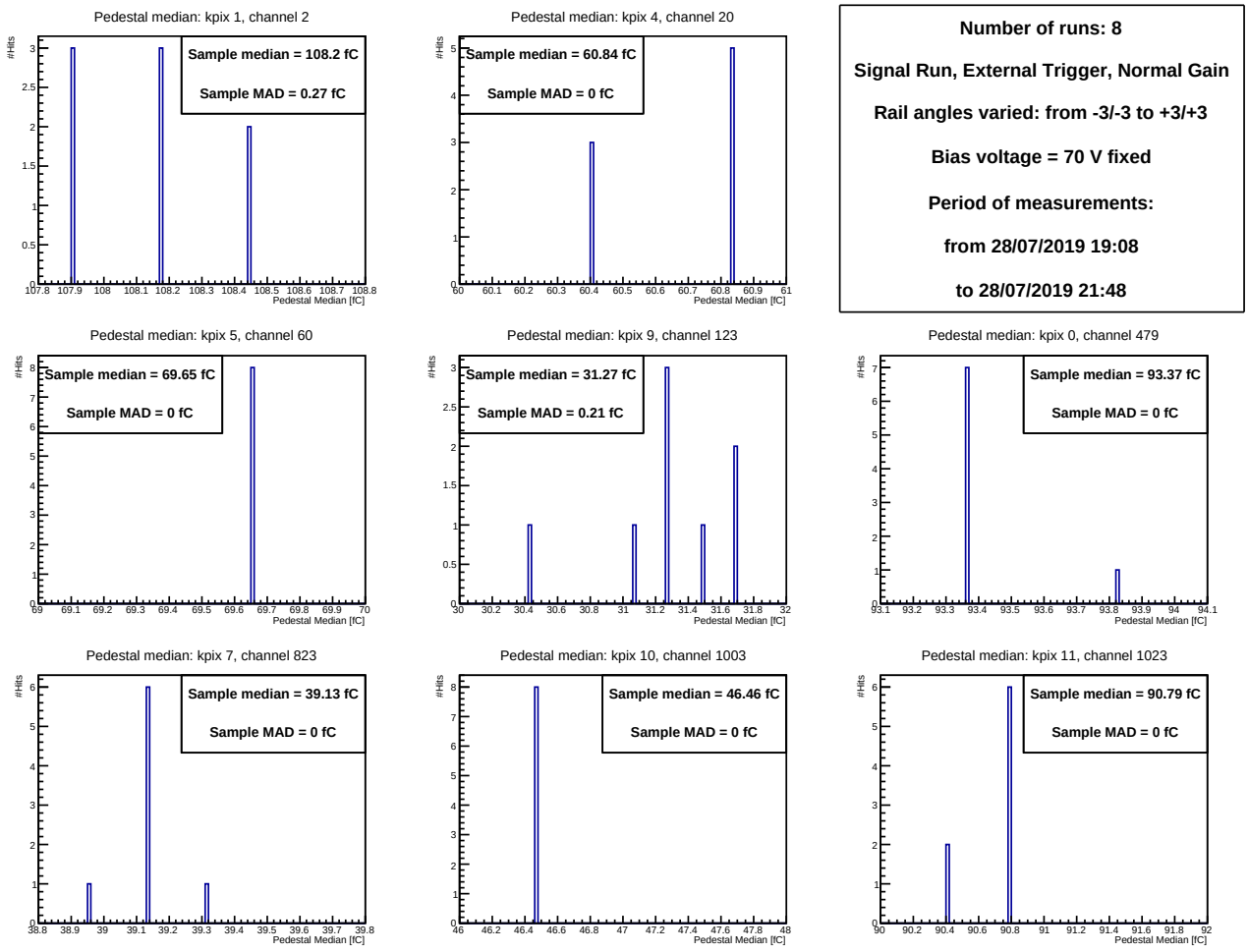


**Figure 5.6:** Pedestal median distribution for eight different random channels, when rail angles were changed and PCMAG was off.

Further studies on pedestal and noise level databases are presented. In section 4 the 13 different external trigger runs only differed for bias voltage applied; here, bias voltage is set to 70 V, but

the position of the cassettes along the rail supports is changed and the magnet can be turned on or off.

Figure 5.5 shows the rail support for the cassettes. For figures from 5.6 to 5.9, rail angles are said to change between -3/-3 and +3/+3: from a mechanical point of view, this means that cassettes were moved along their rail supports, and an angle switch of "1" resulted in a 5° change in orientation. The x-axis ranges significantly vary between histograms, and bad channels are not highlighted in red.

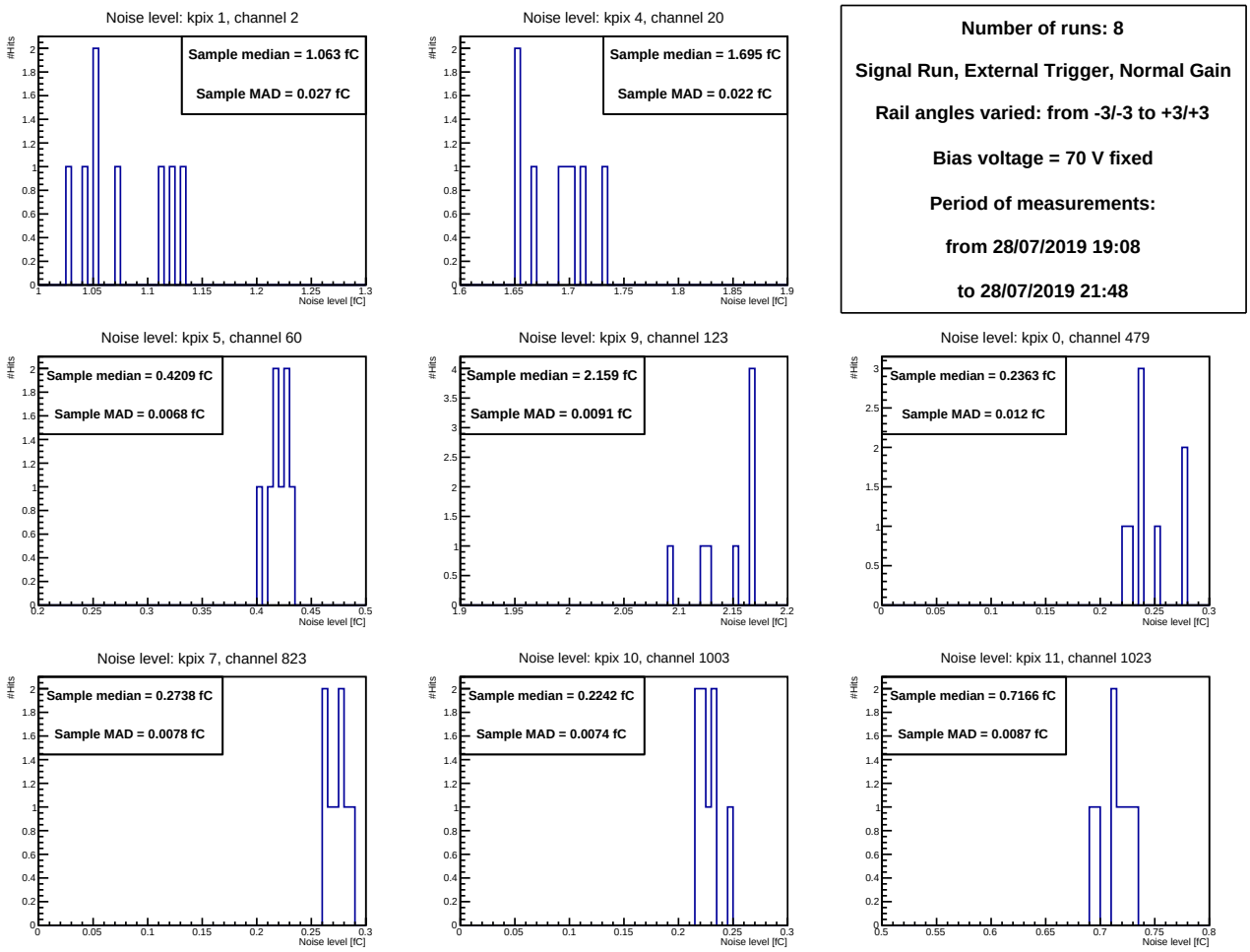


**Figure 5.7:** Noise level distribution for eight different random channels, when rail angles were changed and PCMAG was off.

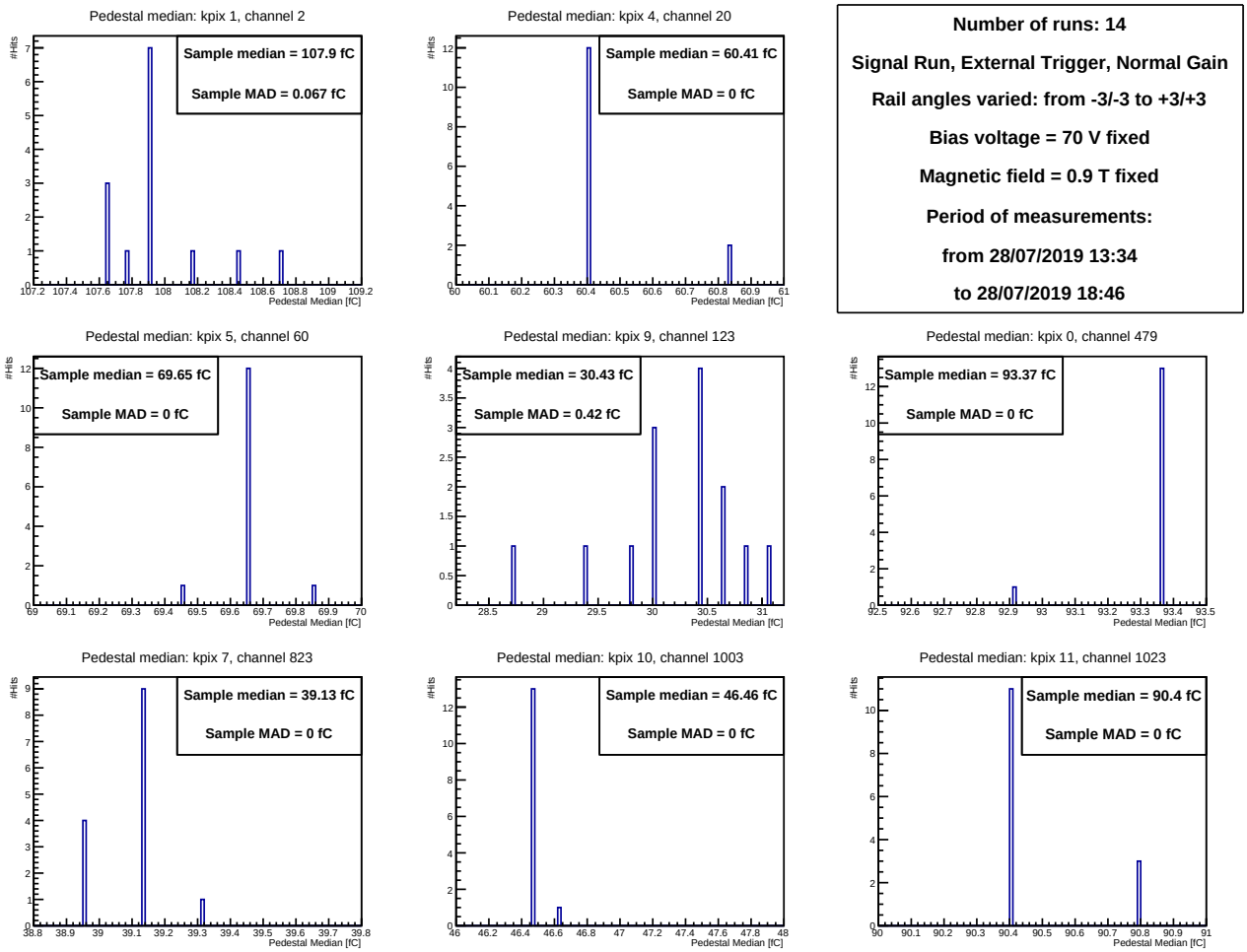


**Figure 5.8:** Pedestal median distribution for eight different random channels, when rail angles were changed and there was a magnetic field of 0.9 T.

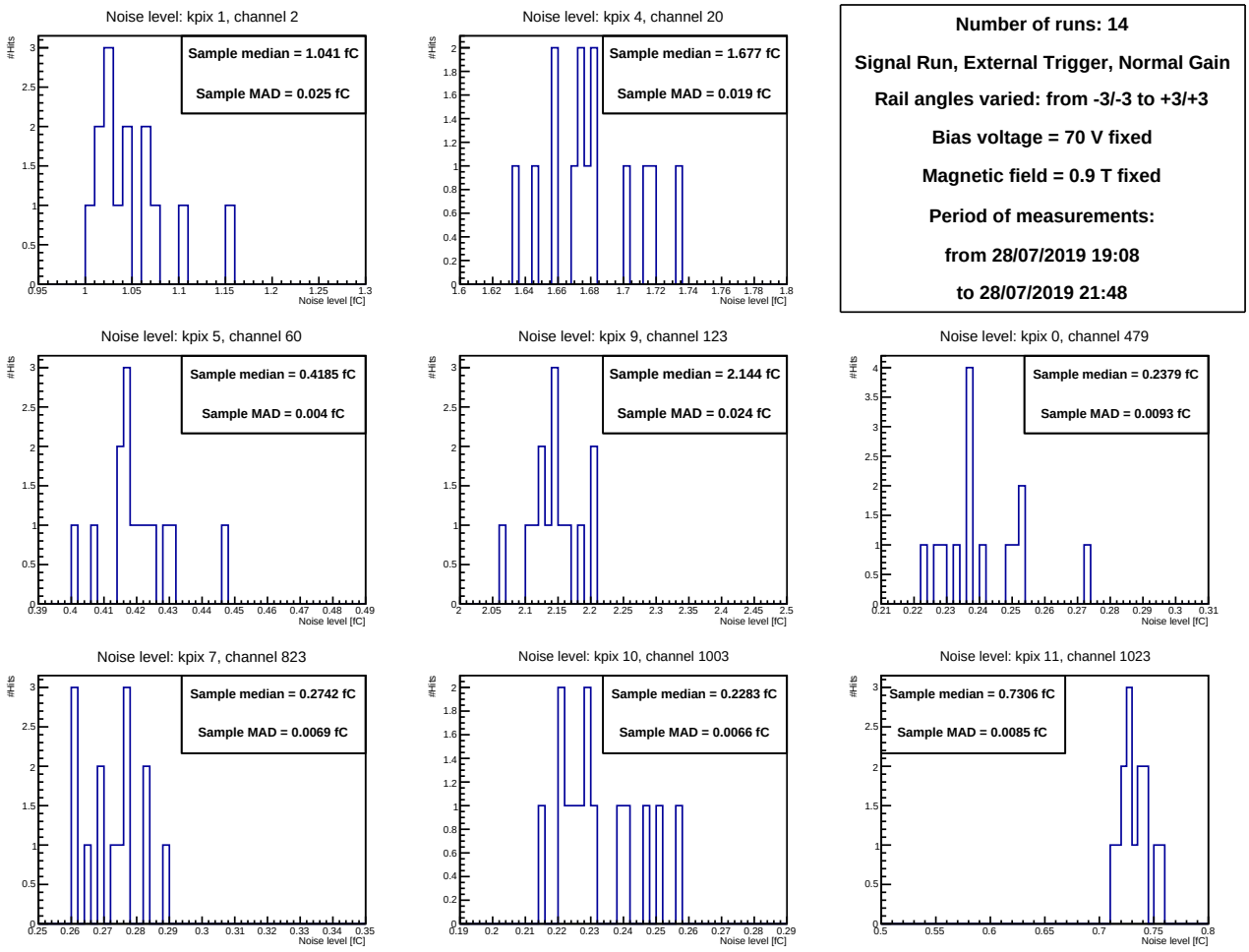


**Figure 5.9:** Noise level distribution for eight different random channels, when rail angles were changed and there was a magnetic field of 0.9 T.

# Acknowledgements

I would like to thank my supervisor, Dr. Mengqing Wu, for her attentive guidance, her constant support and her kind patience when answering all of my questions throughout the whole period of my project.

I would also like to thank my summer student colleague Andreas Löschcke Centeno and his supervisor Uwe Krämer, who both gave a significant contribution to my work; and of course the other summer students - Matthew Koster, Sukeerthi Dharani and Peter McKeown - with whom a strong bond was formed.

It was an honour to get to know and share coffee with all the wonderful people working here at DESY, especially the organizing team, the colleagues from FLC and those who took part in the test beam meetings, where they gave us the chance to present our results weekly and were willing to give us many suggestions.

Finally, I have some special thanks to all my summer student friends. Particularly, those who were always keen on organizing fun projects on the weekends; those I had pleasant chats with in the hostel kitchens, and with whom I baked cakes together; the Italian friends I'll have the chance to hang out with after this summer school, and those I hope will visit me soon in Pisa, so that I can make them feel as warmly welcomed as I felt thanks to them.

# References


[1] Advanced European Infrastructures for Detectors at Accelerators AIDA-2020 Project Home Page, http://aida2020.web.cern.ch, Accessed on: Jan. 09, 2019

[2] M. Wu *et al.*, Development of a large active area beam telescope based on the SiD microstrip sensor, Feb 2019, AIDA-2020-POSTER-2019-001, http://cds.cern.ch/record/2666439

[3] J. Brau *et al.*, KPiX - a 1,024 channel readout asic for the ilc, in: 2012 IEEE Nuclear Science Symposium and Medical Imaging Conference Record (NSS/MIC), 2012, pp. 1857-1860. doi:10.1109/NSMIC.2012.65551433

[4] R. Diener *et al.*, The DESY II Test Beam Facility, Nucl. Instrum. Meth. A922 (2019) 265–286. arXiv:1807.09328, doi:10.1016/j.nima.2018.11.133

[5] Y. Liu *et al.*, EUDAQ2 – A Flexible Data Acquisition Software Framework for Common Test Beams. Prepared for submission to JINST as of September 2019

[6] U. Krämer *et al.*, LYCORIS - A Large Area Strip Telescope, 25 Jan 2018, Talk presented at the International Workshop on Future Linear Colliders (LCWS2017), Strasbourg, France, 23-27 October 2017. C17-10-23.2. arXiv:1801.08505v1 [physics.ins-det]

[7] P. J. Rousseeuw & C. Croux (1993) Alternatives to the Median Absolute Deviation, Journal of the American Statistical Association, 88:424, 1273-1283 doi: 10.1080/01621459.1993.10476408

[8] http://rnc.lbl.gov/ jhthomas/public/SSD/Electronics/Common_mode_GC.pdf

[9] U. Krämer, FLC, Deutsches Elektronen-Synchrotron (DESY). Personal communication, July 2019