FACULTY OF PHYSICS
AND APPLIED INFORMATICS
University of Lodz

# Low-level analysis

J. Sitarek

1st CTAO School, 2024.06.24, La Palma

Original image credit: https://merovia.obsidianportal.com/wikis/shadow-sorcerer

## Outline

- Data levels – low-level analysis
- Preprocessing and readout features
- Baseline and signal extraction
- F-factor method
- Time calibration
- Image cleaning
- Hillas parametrization and other image parameters
- Beyond Hillas: model analysis, CNNs, …
- Stereoscopic reconstruction
- MCs
- gamma/hadron separation and energy and direction reconstruction
- IRFs

# Data levels in CTA

- R0 – raw data
- DL0 – first levels of calibration
- DL1 – image parametrization
- DL2 – event reconstruction and classification
- DL3 – lists of event after cuts with corresponding IRFs
- DL4 – science-binned data (bins of time, arrival direction)
- DL5 – science products (skymaps, light curves, spectra, flux/upper limits maps, …)
- DL6 – catalogs, surveys, ...

This talk  (low-level analysis and some “mid-level” as well)

This will be covered in the high-level analysis part (Atreyee, Fabio & Fabio)

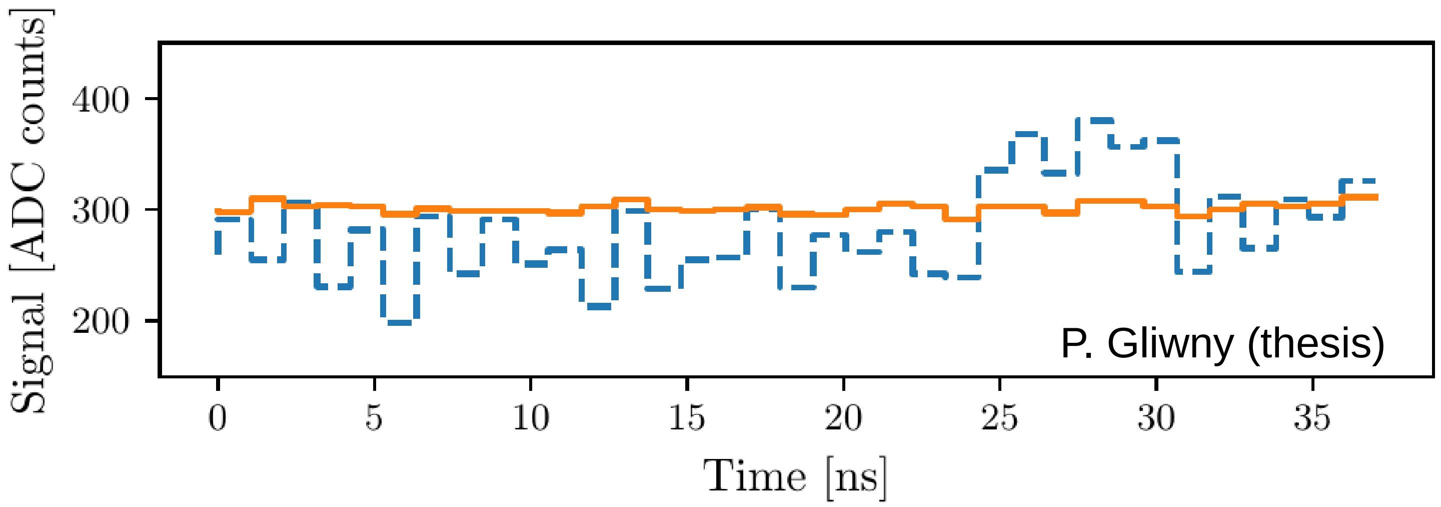
## General idea for this presentation

- There is no ctapipe 1.0 yet, and the software is from time to time introducing major changes, also breaking API.
- Individual CTA teams (including LST) are introducing their own pipelines based on ctapipe (however there is an ongoing process of standardization using ctapipe “stages”)
- LST-1, the first telescopes that will be part of CTAO, is still a single telescope, and mono analysis differs from stereo
- Therefore this is not a hands-on, or ctapipe-only talk, because it would get outdated soon, instead I will explain the general methods and point to their (current) implementation in ctapipe
- I will use LST for most of the examples (but there are also some old plots from MAGIC)

# Why should I care about low-lever analysis?

- CTAO is expected to provide analysis tools and first stages of the analysis (so you do not have to repeat boring stuff and deal with enormous data set)
- But:
  - You should always understand what you are doing, and how the data that you are using were processed
  - It will take time for CTAO to understand what is “good data” and sometimes you need to go back to earlier stages of data processing
  - Scientists are curious people, and want to improve things – the standard (even low-level) analysis can be improved in time with something with higher performance
  - For peculiar scientific use cases (extremely large showers, earth skimming events, direct Cherenkov emission, …) the standard processing for sure will not be the most efficient

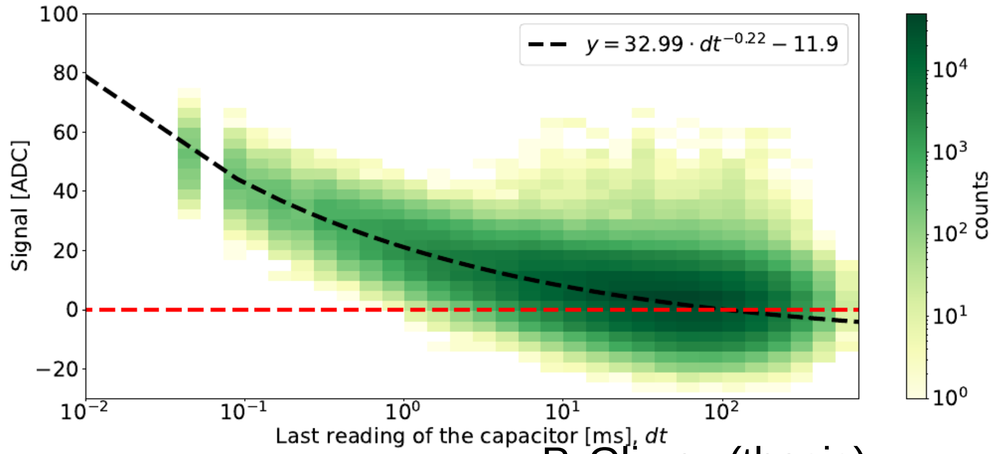
# Pre-processing



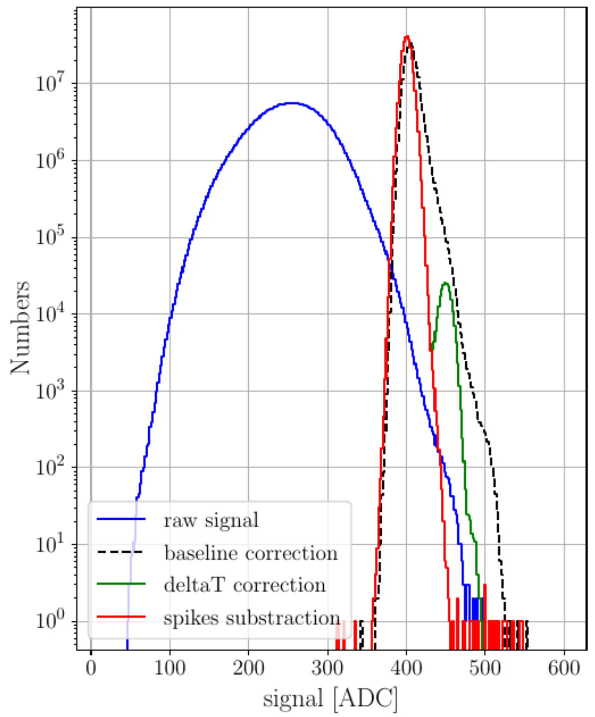
- **The analysis of the data starts even before they are written to disk!**
- The readout system employed in IACTs often require some correction procedures to improve its performance
- As an example DRS4 chip (used e.g. in LST) needs to have pedestal subtracted independently for each capacitor.

# Preprocessing

- Another example from DRS4: the pedestal value increases if the same capacitors are read out in short succession
- There are also (deterministic) spike-like features
- All those features if not corrected for would increase the noise considerably

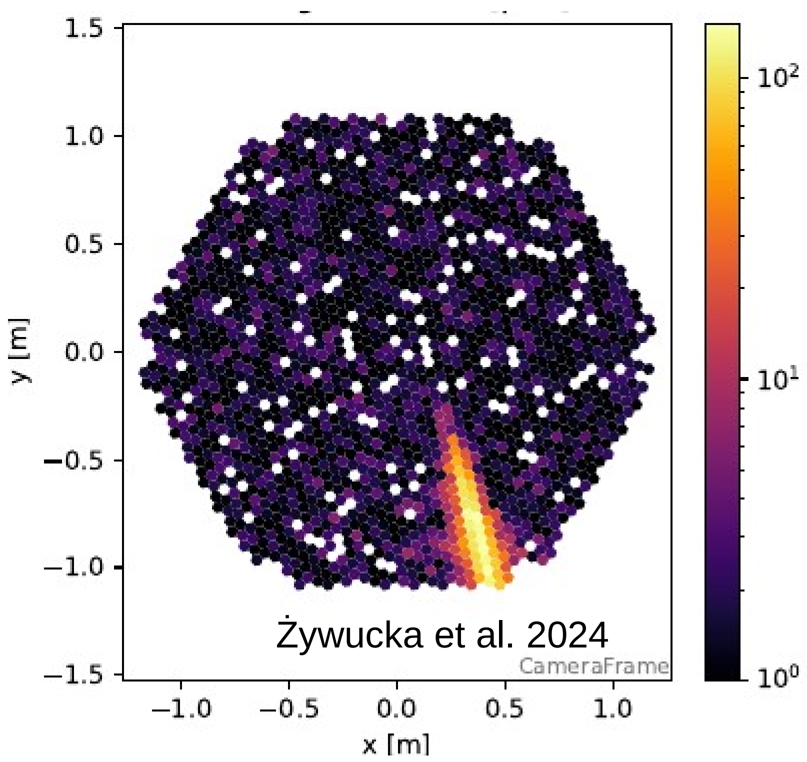


P. Gliwny (thesis)



# Why preprocessing?

- The lowest level corrections need to be done at some point: online or offline
- If they are done offline, they possibly would have to be repeated in RTA.
- Raw IACT data take a lot of space, while the shower information is nearly exclusively in a few tens of pixels. In case of CTA the raw data need to be reduced considerably on the fly
- Algorithms to select the gain and preselect the signal pixels are more reliable if precalibration of waveforms is applied



## Two gains

- In order to increase the dynamic range LST is using two gains: HG and LG, which differ by a factor of ~15.
- For small signals it is important to use HG to fish them out of electronic noise (NSB noise is also applified with the same gain)
- For big signals (>~100 p.e.), HG will start to saturate and LG should be used instead
- There is a large overlap region when both gains can be used (and cross-calibrated)
- To select proper gain (and decrease the data size by a factor of 2) we need roughly pedestal-subtracted waveform.
- Note that the same event can have pixels with LG (most) and HG

## Preprocessing in ctapipe/lstchain

- Even if the corrections are done online in EVB for LST, they need some input that is created offline.
- For example: lstchain/tools/lstchain_create_drs4_pedestal_file.py creates the pedestal files (including also spike heights)

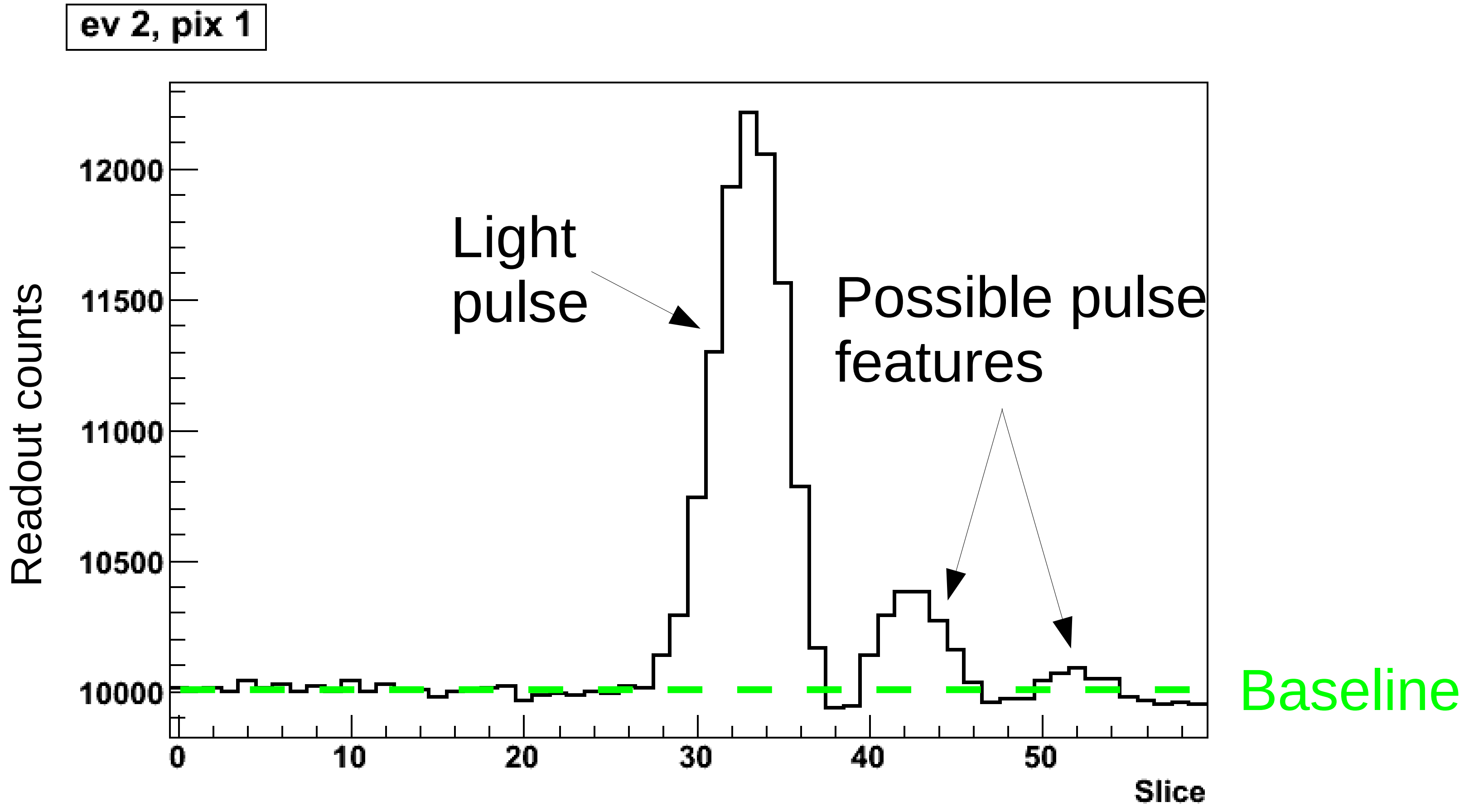
## Calibration of the waveforms

- The purpose of the calibration of the data is to obtain two pieces of information for each pixel in each event:
  - Charge (measured in photoelectrons)
  - Time at which this signal arrived
- This is a huge reduction of the data set – from a few tens of number per pixel-event to just 2.
- The calibration to p.e. can be reapplied to the waveforms – then we can integrate signals in different times.

## Special events

- To perform calibration of real data we need some special events:
  - Flat-field (calibration) events: rapid shots of laser (lambda = 355 nm for LST) with a particular intensity (controlled by filters)
  - Pedestal events: random noise events (without signal)
- These events can be taken as:
  - Separate runs (also with closed camera for pedestal)
  - Interleaved events during the data taking (to update drifts etc.)

# Typical waveform of light pulse event

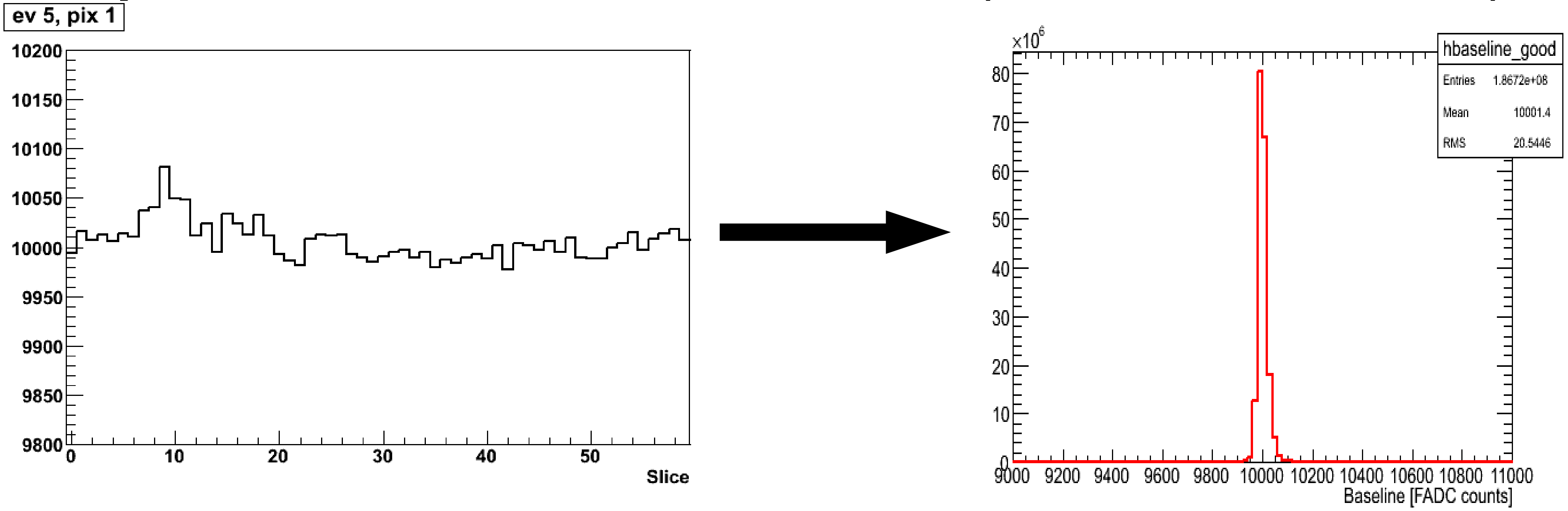


Old example from MAGIC

## How to estimate baseline

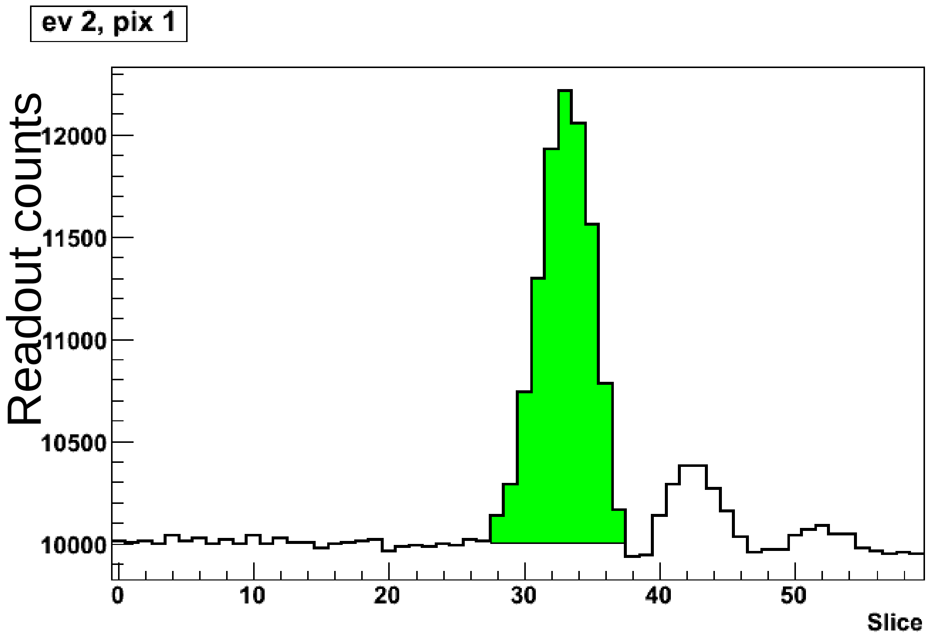
(assuming that preprocessing is done)

- Take many “pedestal” events
- Bin the signal from each time slice in a single histogram and take a mean/median/fit a Gaussian.
- Note that e.g. in LST the preprocessing already corrects the pedestal (for each capacitor), so you do not have to do it, unless you want to correct for the drifts in time (with interleaved events).



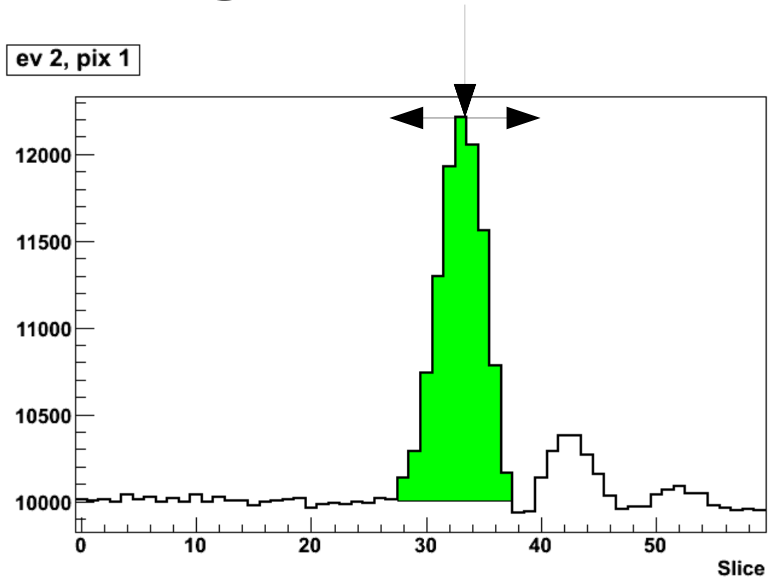
# How to extract the signal ?

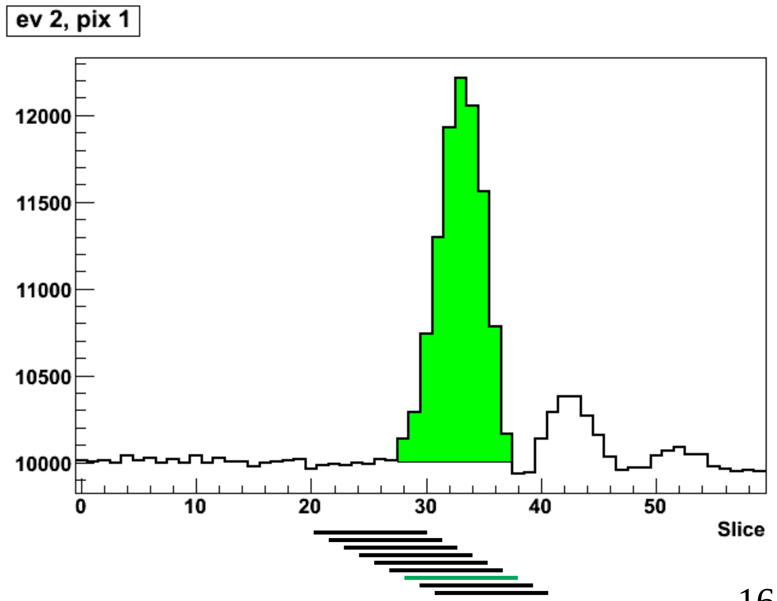
- One needs to integrate the counts above the baseline in the region of the pulse
- The extractor can be peak-search (which even in the case of lack of signal finds the highest fluctuation), or unbiased (with average of 0)
- There are multiple methods how to do it. A bunch of extractors is implemented in ctapipe: `ctapipe/image/extractor.py`



# How to extract the signal ?

- LocalPeakWindowSum (main LST extractor):
  - Find the peak
  - Integrate a fixed number of slices before and after the peak
- Sliding window (used e.g. by MAGIC)
  - Integrate N consecutive slices and take the highest sum



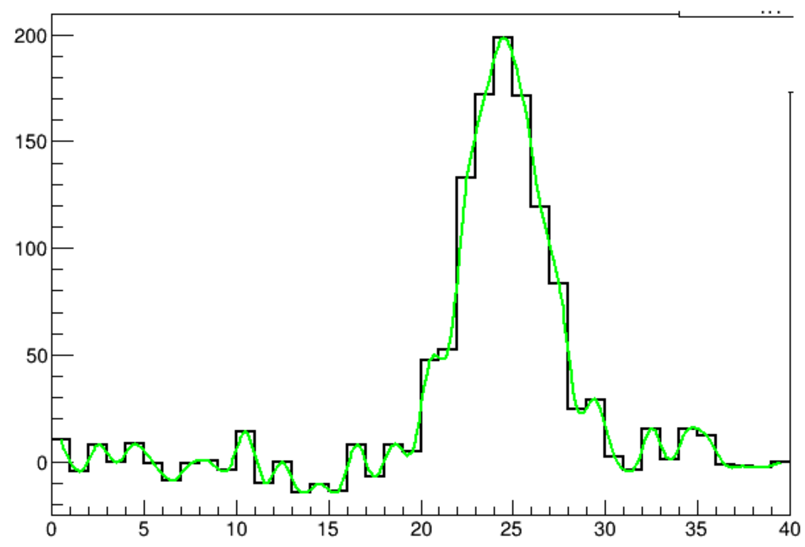


## Unbiased extractors

- GlobalPeakWindowSum:
  - Stack up waveforms (should be done after time flat-fielding!) of all (or just the brightest) pixels and find one global peak
  - Integrate a fixed number of slices before and after that peak
  - Fluctuations from many pixels average out and the resulting peak position is nearly not biased.
  - In LST this extractor is used for muons – many pixels with small signal

# More fancy extractors – example of spline

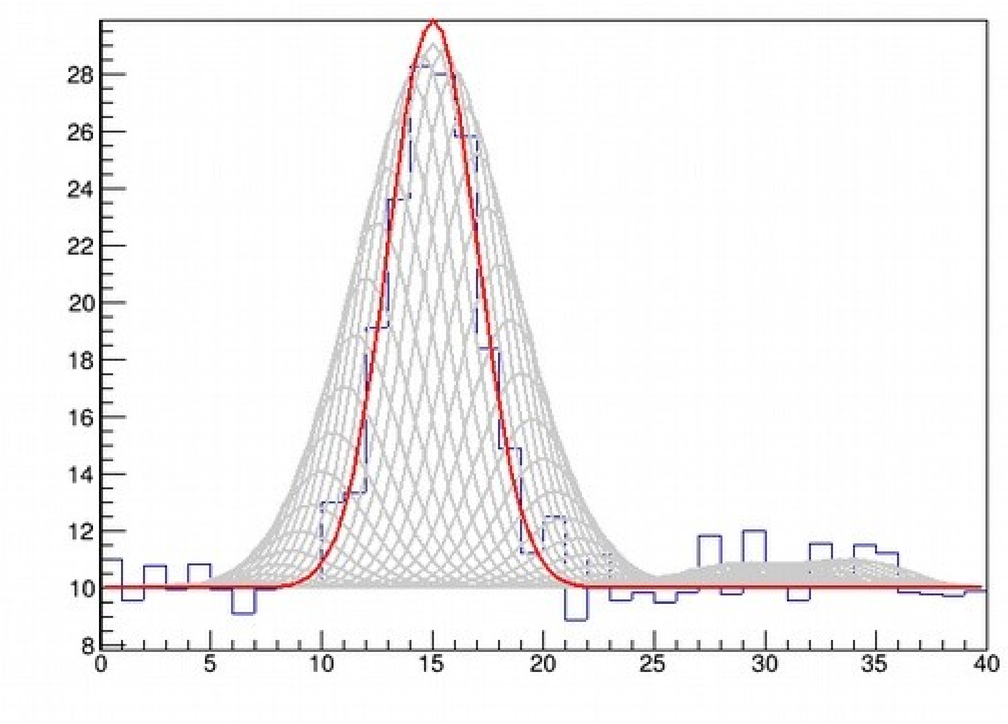
- Works like peak extractor/sliding window, but interpolating between samples with a spline
- Somewhat slow, but can help if sampling is course



Not (yet?) in ctapipe

# More fancy extractors – example of digital filter

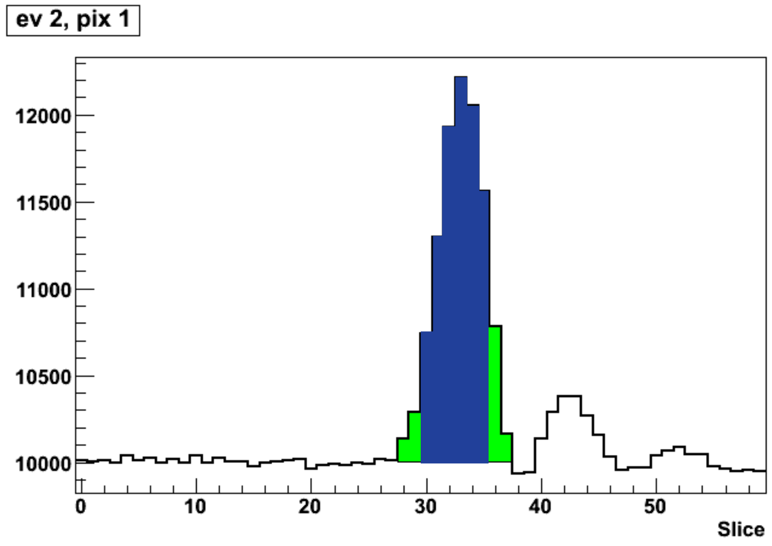
- Extractors can be also more complicated, e.g. a digital filter is a kind of scalar product between (assumed!) pulse shape, and the real data
- You can achieve better performance (in particular if the data are noisy), but at the price of slower processing and possibly higher systematics



Not (yet?) in ctapipe

## Correction for not integrated signal

- Depending on the extractor you will sum up more or less of the signal.
- Note that the extractor can be applied already on the calibrated waveform to integrate p.e.
- In this case you either have to correct for the missing charge or make sure that you reproduce it exactly in MCs



In ctapipe extractors this option is called apply_integration_correction lstchain is not using it (setting to False)

## Calibration to p.e.

- The waveforms are taken in units of **readout counts.**
- We want to express the signals (either integrated, or the waveforms) in p.e., i.e. we need to know the conversion factor from readout counts to p.e.
- The conversion factor will be different for each pixel, it can also evolve in time

# F-factor (excess noise) method

- Assume that 1 readout count corresponds to *C* phe
- A pulse with *N* phe will create on average a signal with:

$$MEAN = N/C \text{ counts}$$

- Assuming Poisson statistics the fluctuations of this signal should be:

$$\sigma = \sqrt{N}/C \text{ counts}$$

- Thus:

$$\left(\frac{MEAN}{\sigma}\right)^2 = N \qquad C = \frac{N}{MEAN}$$

**This way we can calibrate without knowing the calibration light intensity**

# F-factor (excess noise) method

- In reality one has to take into account a correction for non-Poissonian response of PMT ($F^2$, of the order of 1.2), and for the fluctuations of the pedestal

$$N = F^2 \frac{MEAN^2}{\sigma^2_{signal} - \sigma^2_{pedestal}}$$

$$C = \frac{N}{MEAN} \qquad g = \frac{MEAN}{N}$$ Gain

# Instabilities

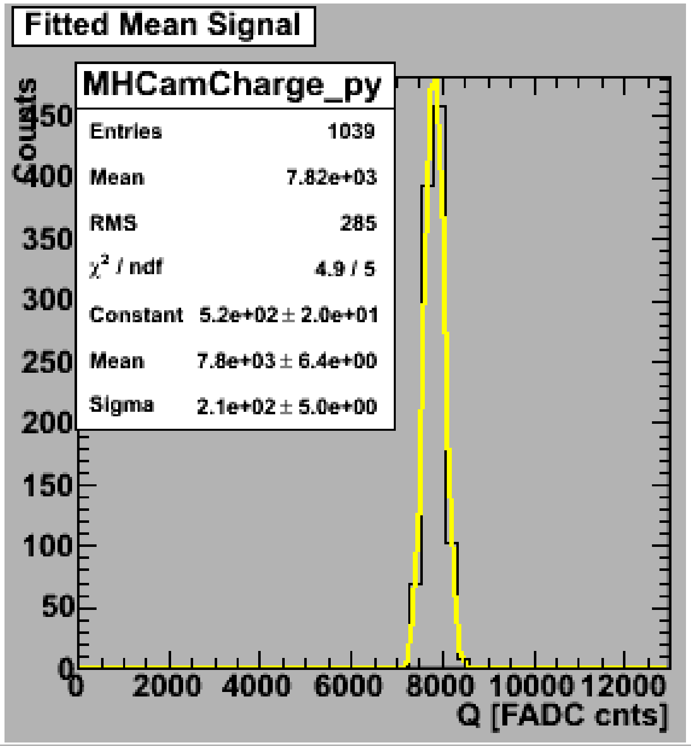
- F-factor method is heavily based on analytical reproducing of fluctuations of the signal.
- If there is some instability (varying laser light intensity, unstability of the readout integrated signal, …) it will increase the signal sigma and overestimate the gain

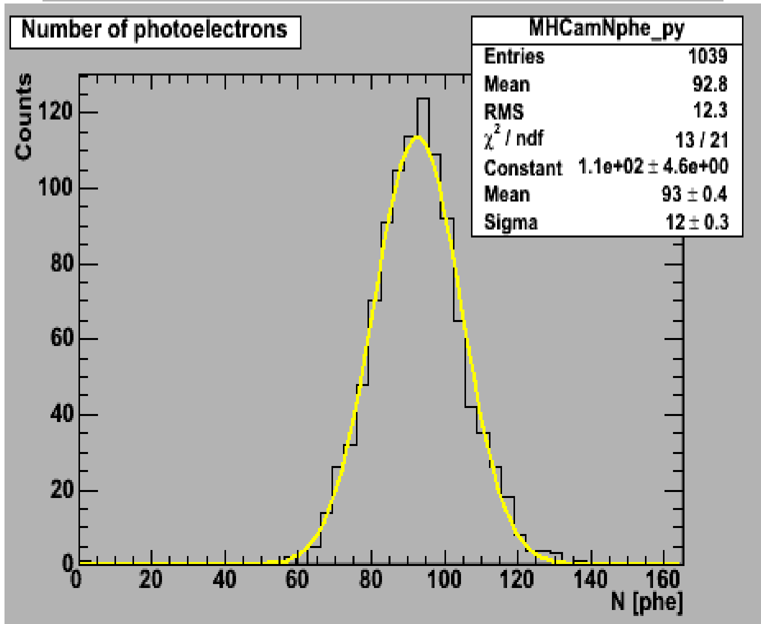
$$g = \frac{\sigma_Q^2 - \sigma_{ped}^2}{F^2(\mathrm{Q} - ped)} - \frac{B^2}{F^2}(\mathrm{Q} - ped)$$

For example in LST laser contributes to ~0.7% instability, and readout not uniform sampling to 2.5%

# Flatfielding and equivalent phe

- HV of pixels is adjusted to have the same signal in all the pixels from a homogeneous illumination of the camera, thus the distribution of mean signal for all the pixels is very narrow
- However PMTs differ from one to another (QE curve), so even for homogeneous illumination there is an extra spread of ~10% in the number of p.e. from such pulses.





## Flatfielding and equivalent p.e.

- Since for the shower images we want to have the same signal all over the camera for the same light intensity we do not calibrate to real p.e., but to “equivalent p.e.”:

$$C_{pixel} = \frac{N_{camera\,average}}{MEAN_{pixel}}$$

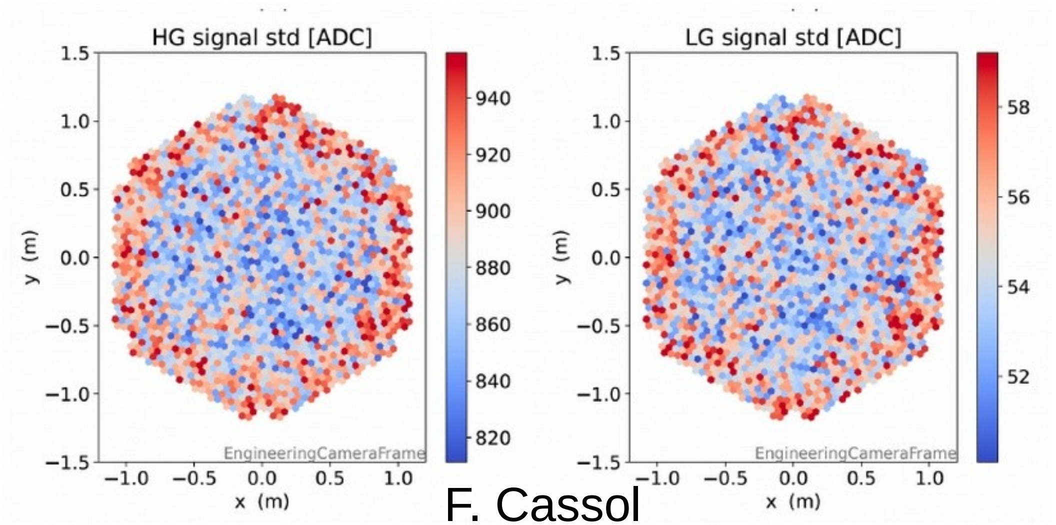
- No broken/outlayer pixels should be used for the averaging
- LST is using median instead of mean.

# What are the requirements for F-factor method to work correctly ?

- System should behave linearly (or you need a different conversion factor for each signal range)
- It is better if light intensity is significantly larger than the noise (especially if the noise is not Gaussian)
- If light intensity is large $\sigma_{\text{signal}}$ / $\text{MEAN}_{\text{signal}}$ is small, and you are more and more dependent on doing instability correction right.

# Example from LST

- Different values for each pixel and each gain
- The radial spread is a real effect (selection of PMTs according to QE curves)

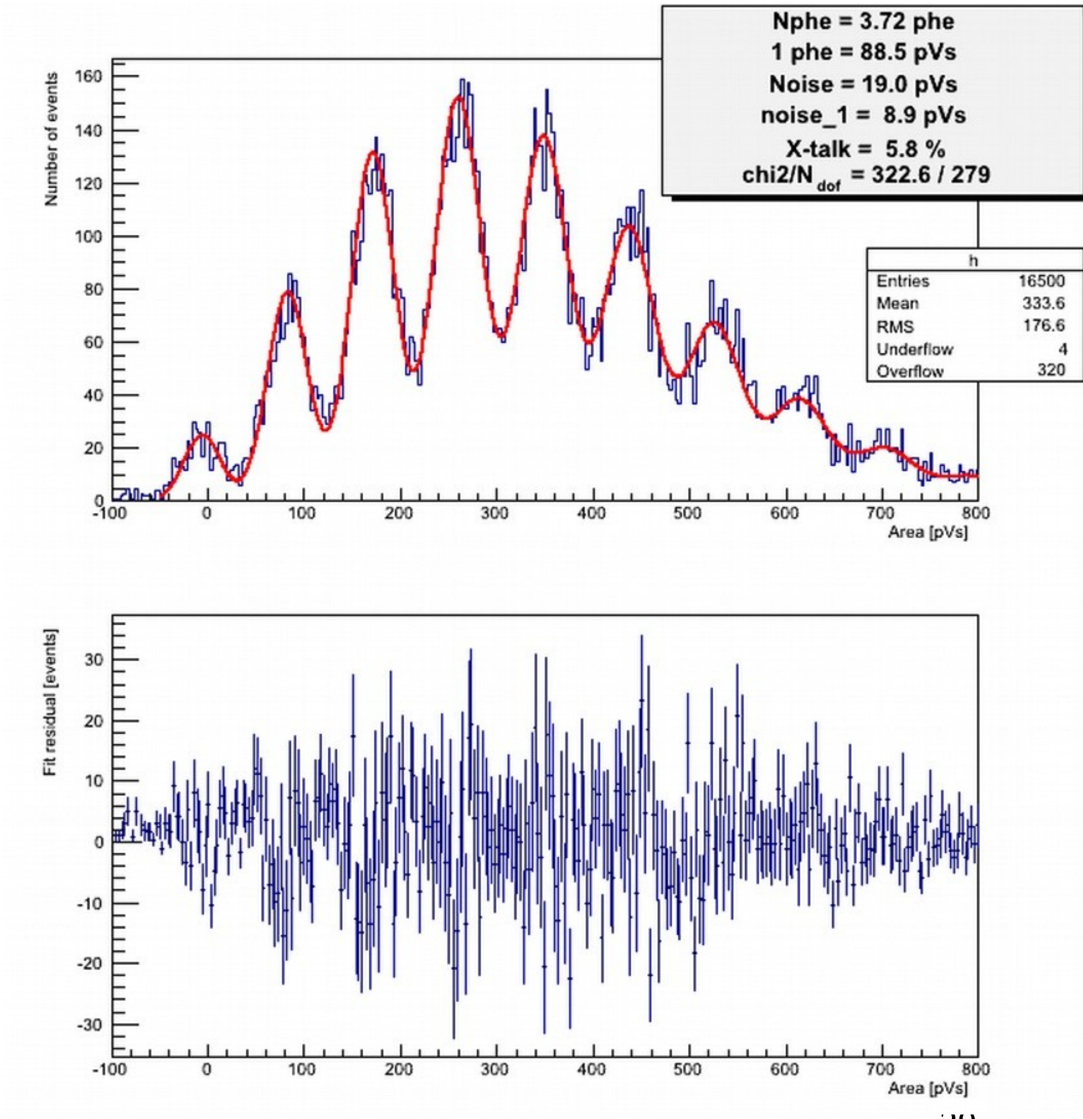


F. Cassol

Photo-electrons/pixels of FF events
(the pattern reflects the PMTs' QE)

# Calibration of SiPMs

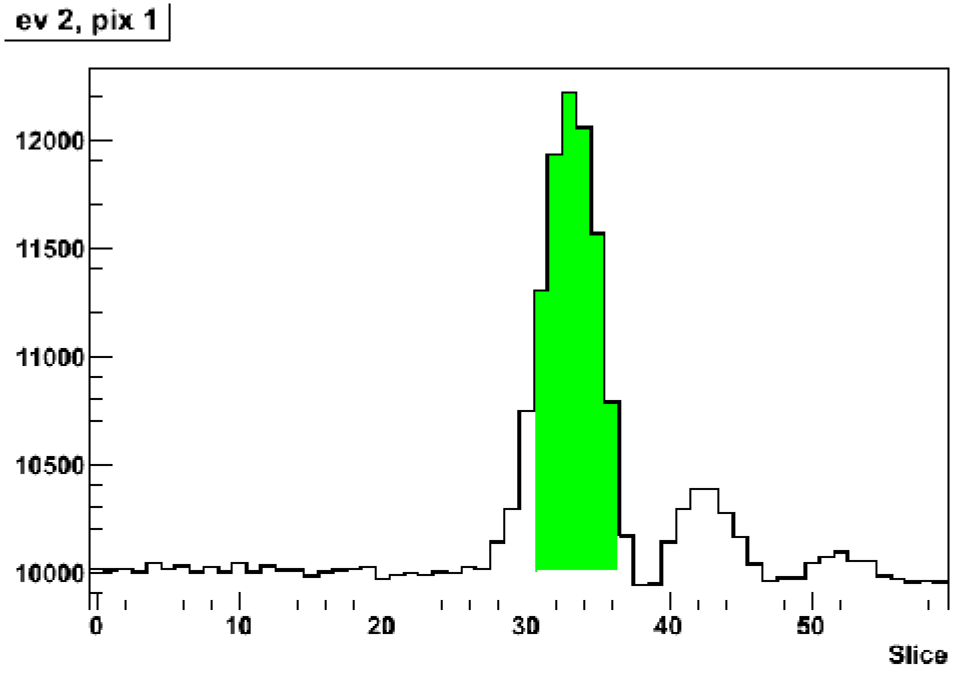
- F-factor method is essential for PMTs where in normal conditions we cannot distinguish single p.e. signals.
- SiPMs have very sharp single p.e. peaks, so one can derive the gain from the peak-to-peak distance
- … but one needs to watch out for the extra p.e. from the optical crosstalk



# Arrival time of pulses

- Ctapipe extractors estimate the arrival time of the pulse as the average time slices of the integrated window weighted with the signal in them

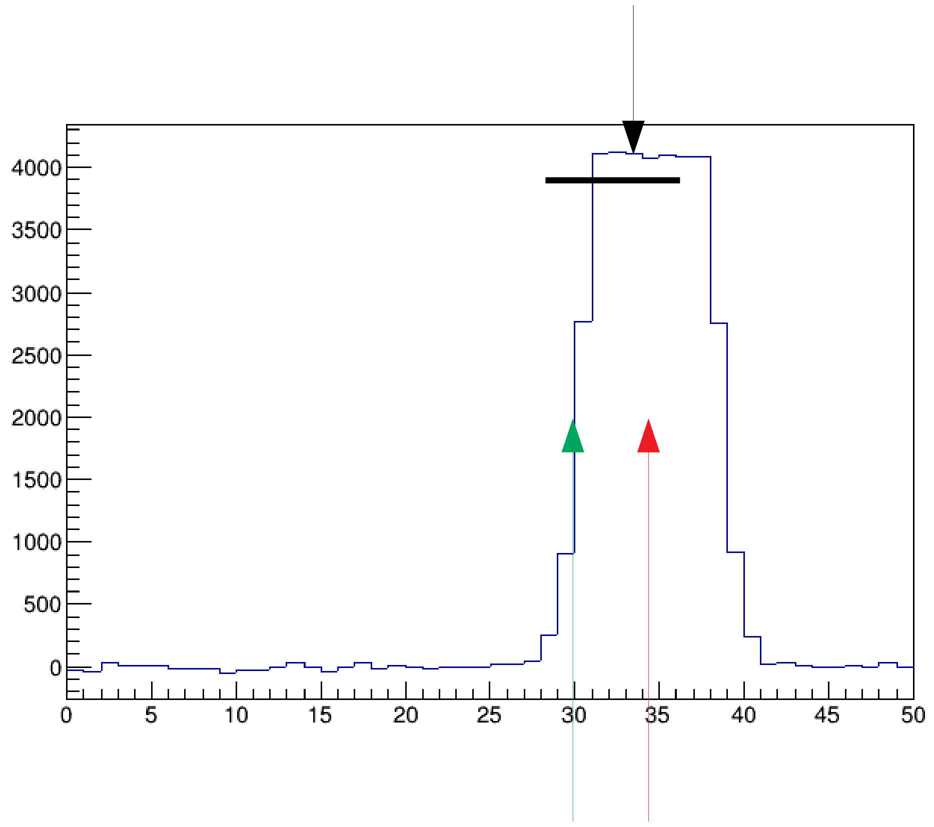
$$t_{arr} = \frac{\sum i\, s_i}{\sum s_i}$$



- $i$ – time slice number
  $s_i$ – signal in slice $i$
  summing is performed over the pulse integration window

# Saturated pulses

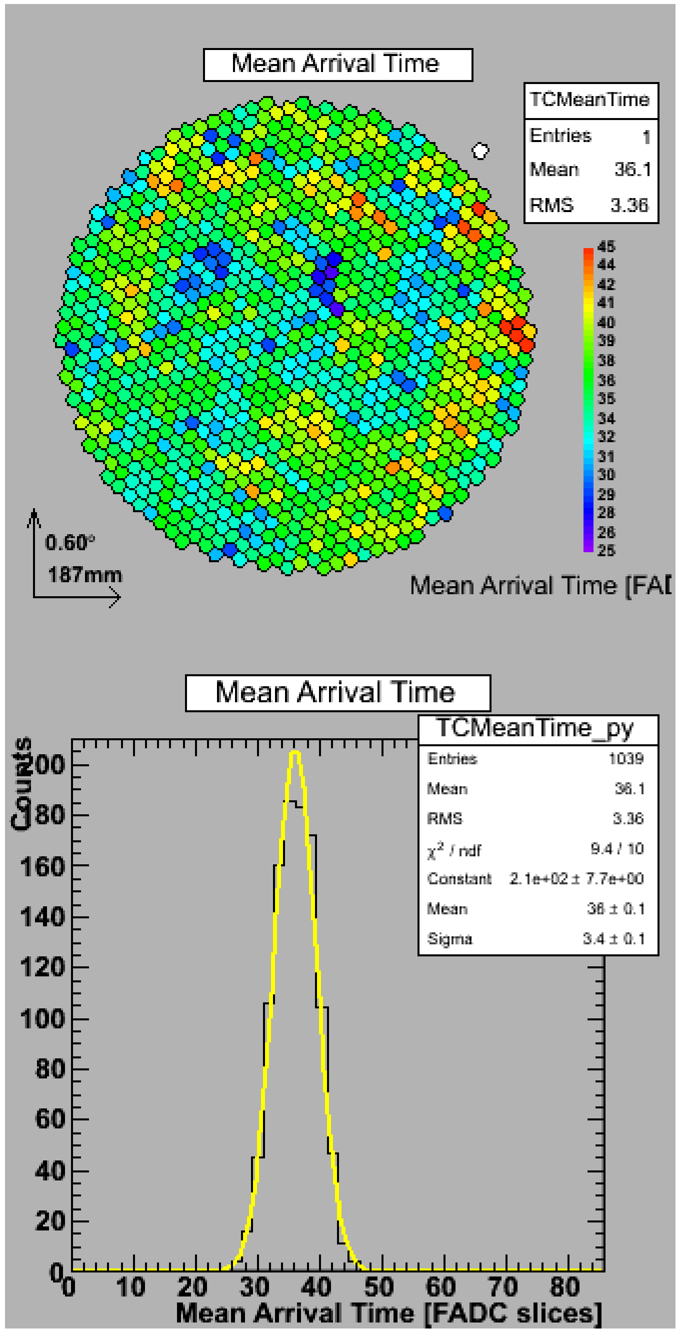
- In case of heavily saturated pulses the method from the previous slide might not be efficient
- Thankfully LST is resilient to saturation up to very high signals due to usage of two gains



Alternative method commonly used is to take the raising edge of the pulse, but this is less precise to compute (and cannot be mixed with the weighted slice method).

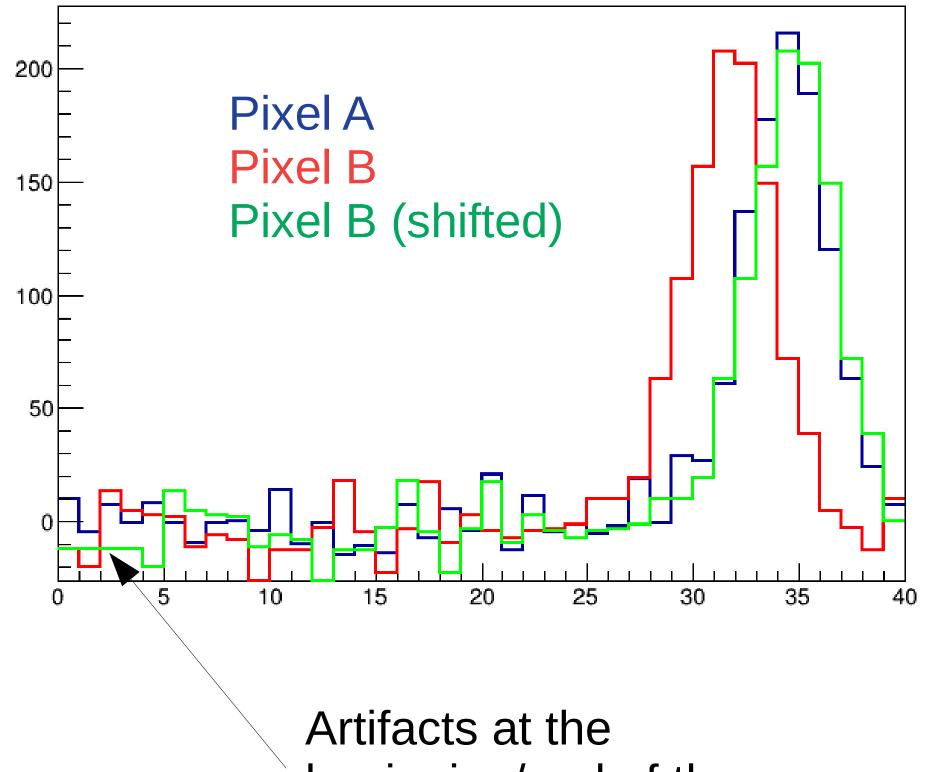
# Time calibration

- Due to small differences in the length of the optical fibers, electronic paths and the transit times of the electrons inside the PMT (different HV settings) a synchronous light pulse illuminating the whole camera will not be recorded at the same readout sample for all readout channels.
- The most basic correction to calibrate those time differences is to calculate the mean arrival time in calibration pulse and then subtract it



# Shifting waveform vs calibrating time

- Extracting the arrival time from the pulse we can correct it for the average arrival time from calibration
- Alternatively we can shift the waveform – this allows more flexibility (e.g. for Global Peak extractors, but has caveats:
  - The shift can be done easily only by an integer number of slices
  - Artifacts at the beginning/end of ROI

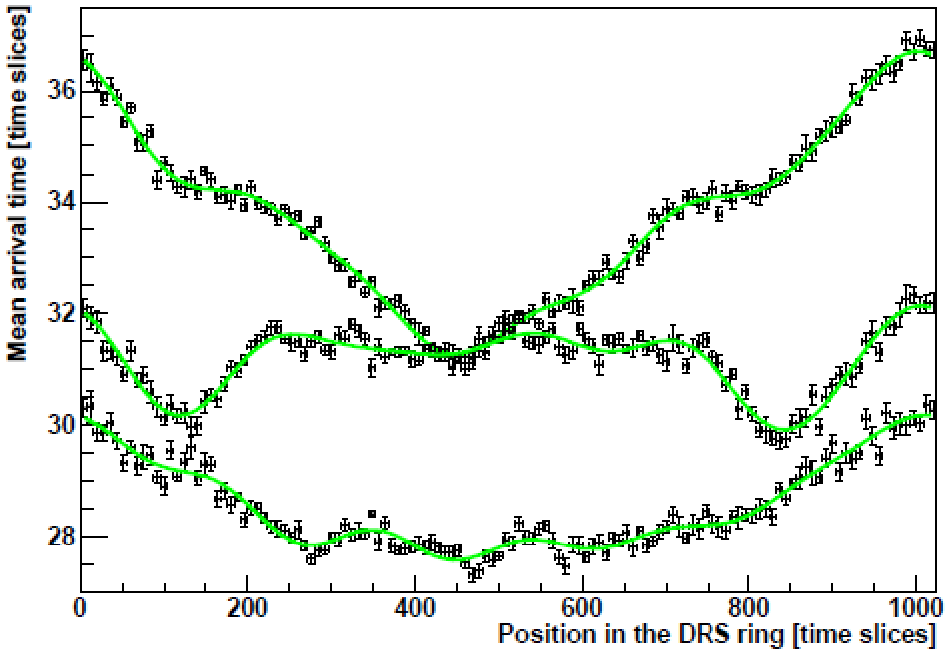


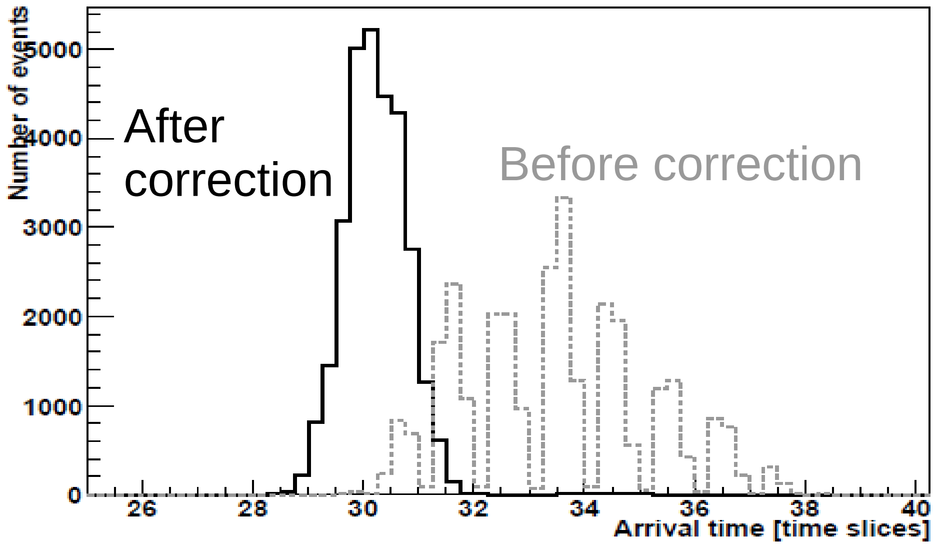
Artifacts at the beginning/end of the ROI due to missing information

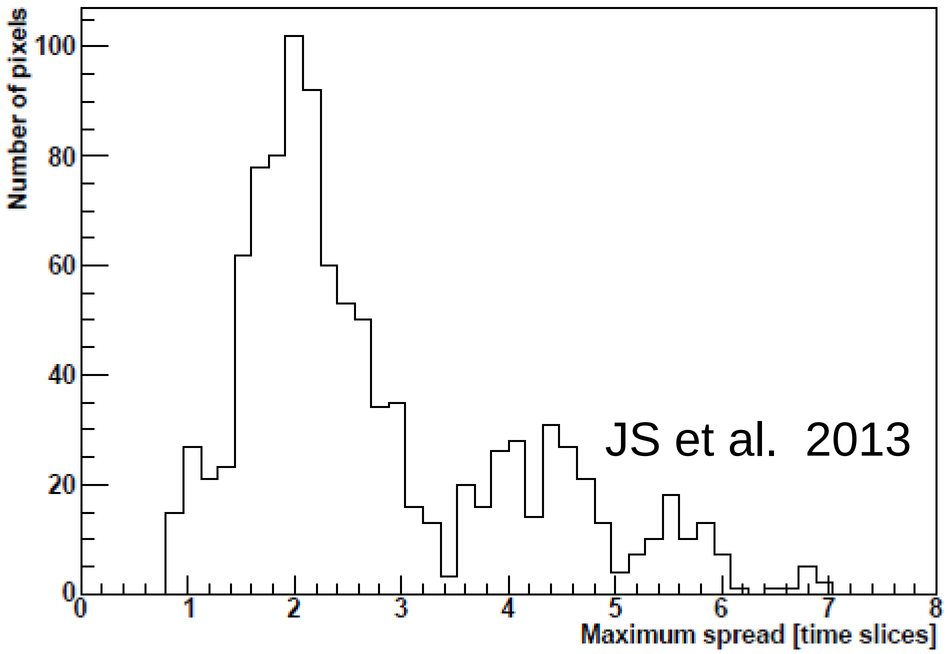
Extractor option in ctapipe to shift the waveform:  apply_waveform_time_shift

# DRS arrival time calibration

- Due to uneven sampling in DRS there are additional delays up to a few ns dependent on the position of the signal in the ring
- The correction is different for different DRS chips







JS et al. 2013

## Additional calibration information: pedestal bias and RMS

- Using dedicated pedestal runs we can calculate how big noise do we have in our data – this is essential for optimizing the cleaning
- The simplest way to do it is trying to search for a signal in runs without showers
- Two important number:
  - Bias (average signal extracted with peak search from empty field)
  - Standard deviation (noise on top of empty field or on top of signal, the two are slightly different)
- Moonlight and brighter Galactic fields will increase the noise in all the pixels
- Stars and technical problems might increase the noise in individual pixels

## Data reduction

- CTAO has strickt requirement on only a small fraction of pixel waveforms being saved (to save space)
- While the shower information is indeed located mostly in a small fraction of pixels, the problem is that apriori you do not know in which ones, so first you need to do image cleaning (we will come back to it), and based on it come back to waveforms and keep only the important ones (the ones that survived the cleaning plus their neighbours)

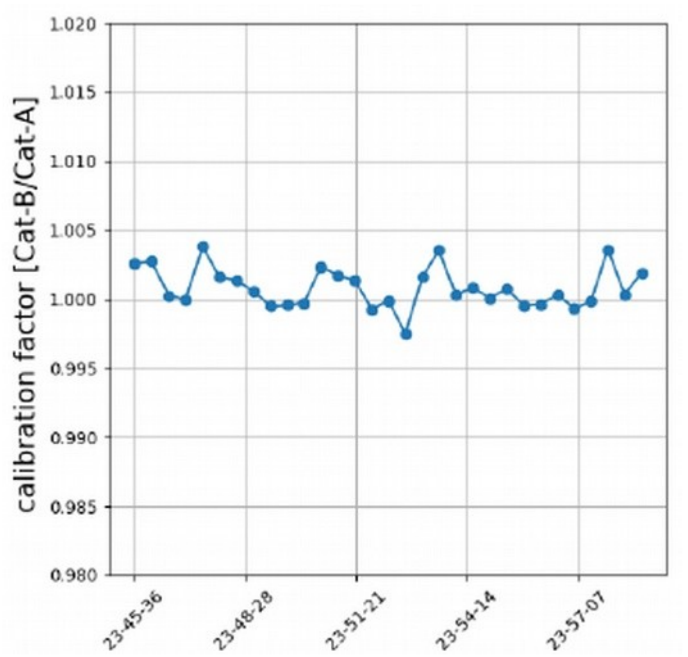
## DL0 level

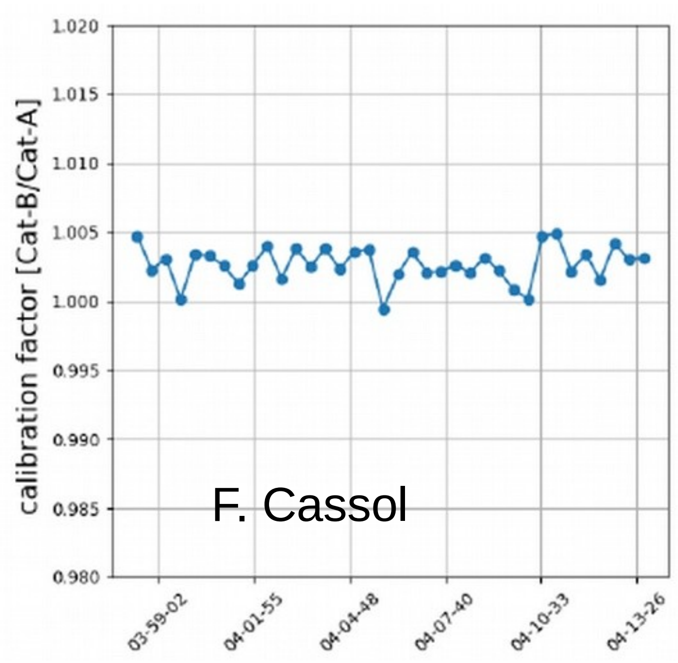
At this point we should be at DL0 level:
We still have waveforms, but they are calibrated in units of p.e. and the timing of pixels is also calibrated

## Cat A / Cat B calibration

- Level A calibration are the basic calibration procedures (all the listed before) that need to be available already in real time analysis
- Further improvement of the calibration can be done with offline Level B correction using **interleaved pedestal/calibration events**
  - Update of gains (changing HV due to stars, drift, ...)
  - Tagging of unusable pixels
  - Detect and discard periods with too many unusable pixels
  - ...

# Example of Cat B calibration for LST-1 gain evolution



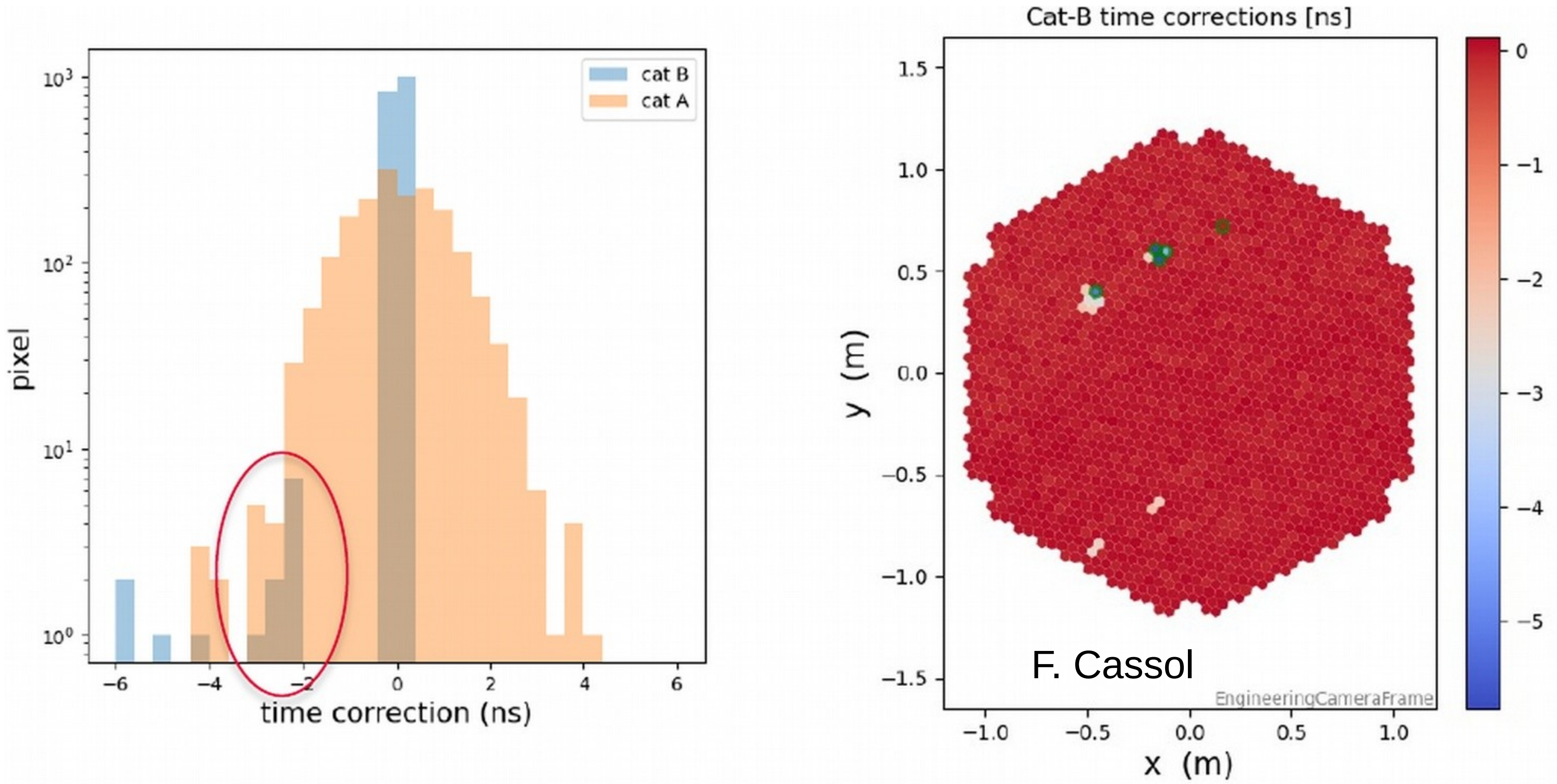


F. Cassol

~1% effect

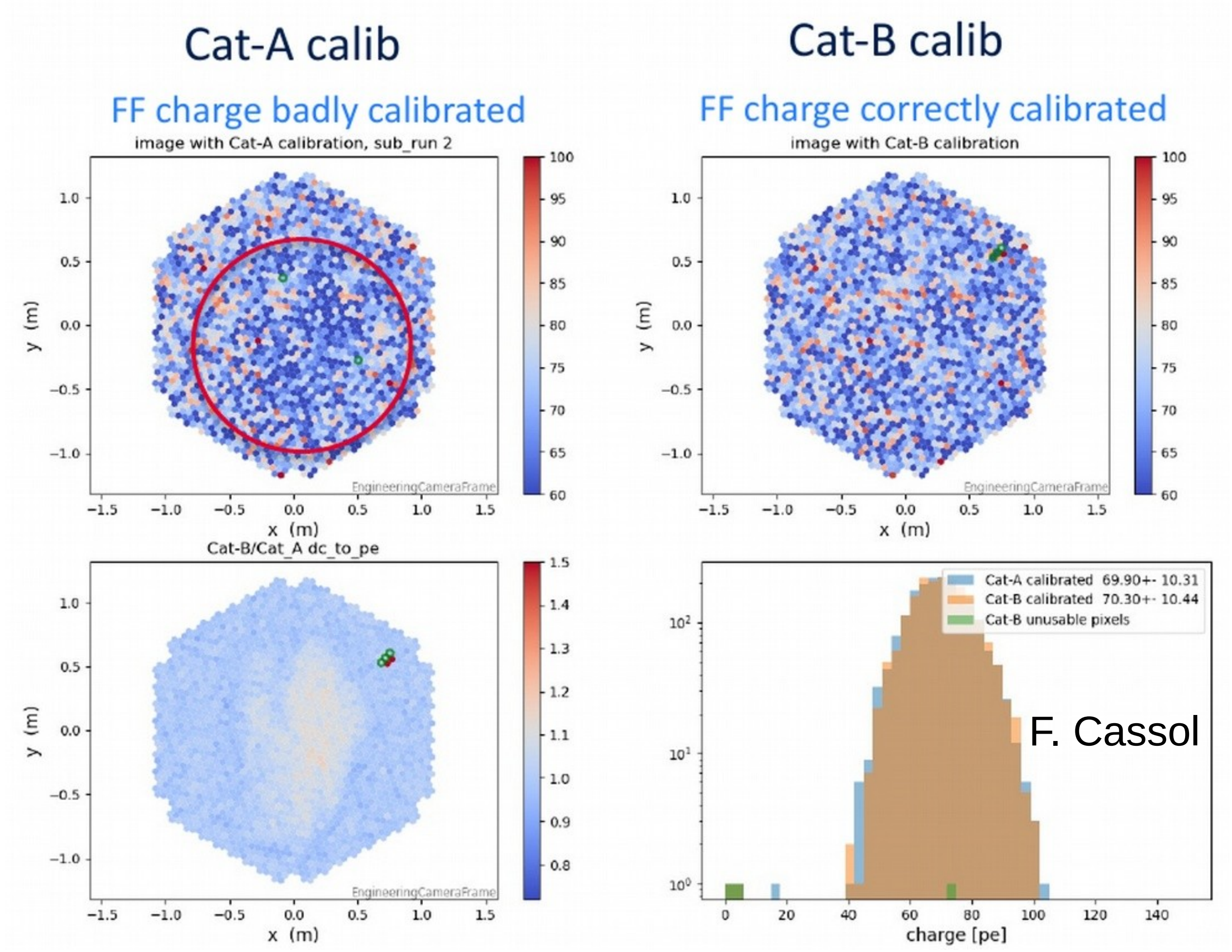
Additional 0.3% constant shift

# Example of Cat B calibration for LST-1 time calibration evolution



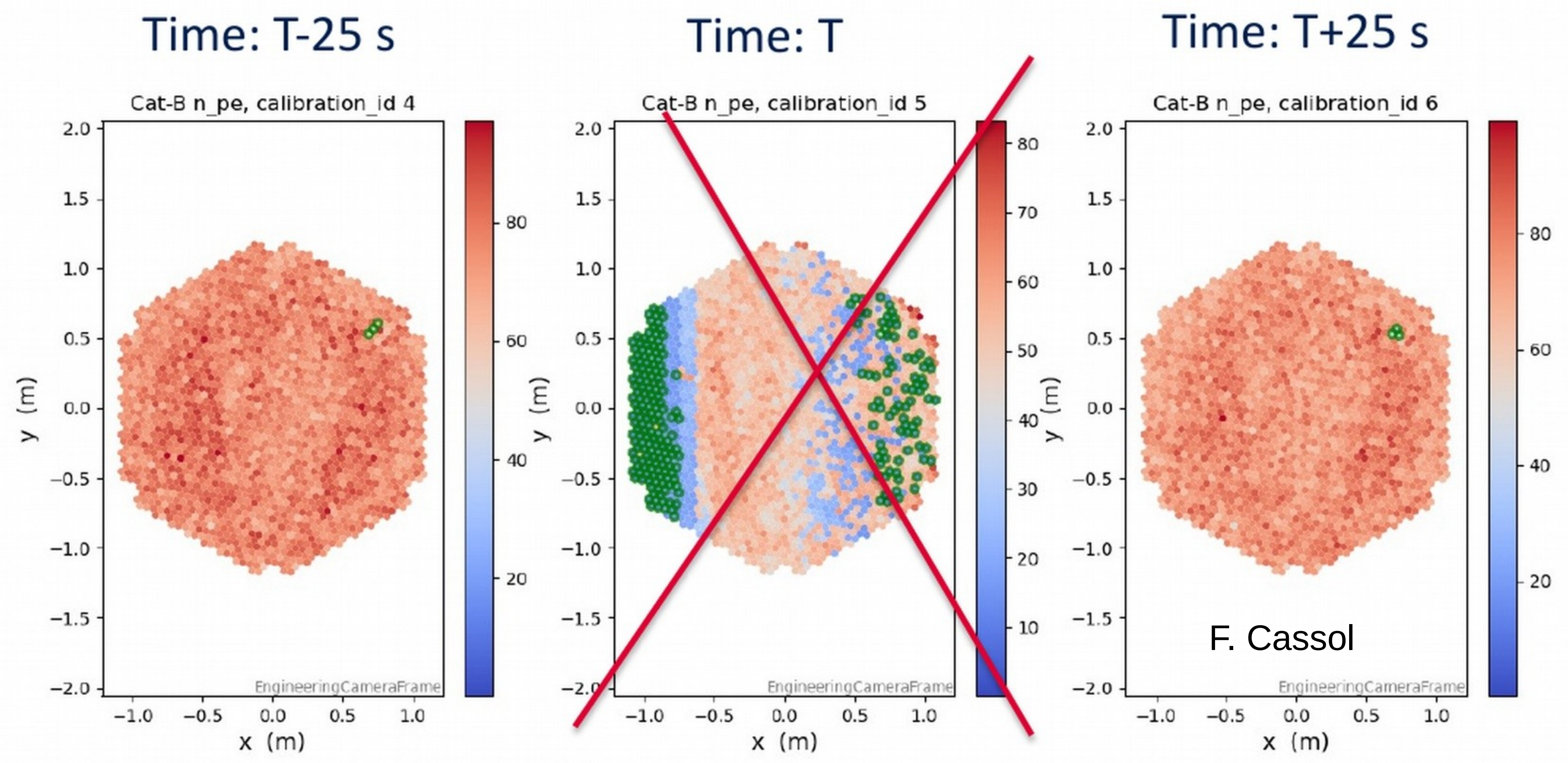
Bright stars (in green) cause completely dead pixels, but pixels around might still have lowered HV that will delay the arrival time in them

# Problems during the night: example of a humidity spot in the camera



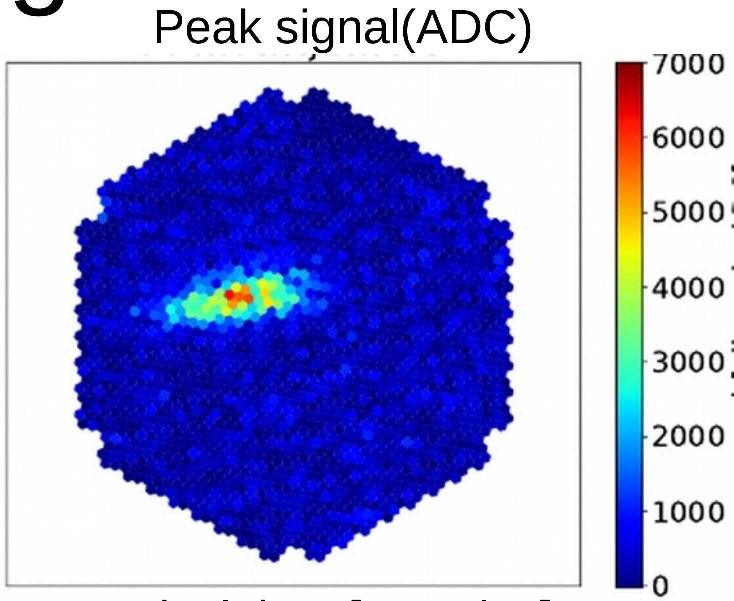
F. Cassol

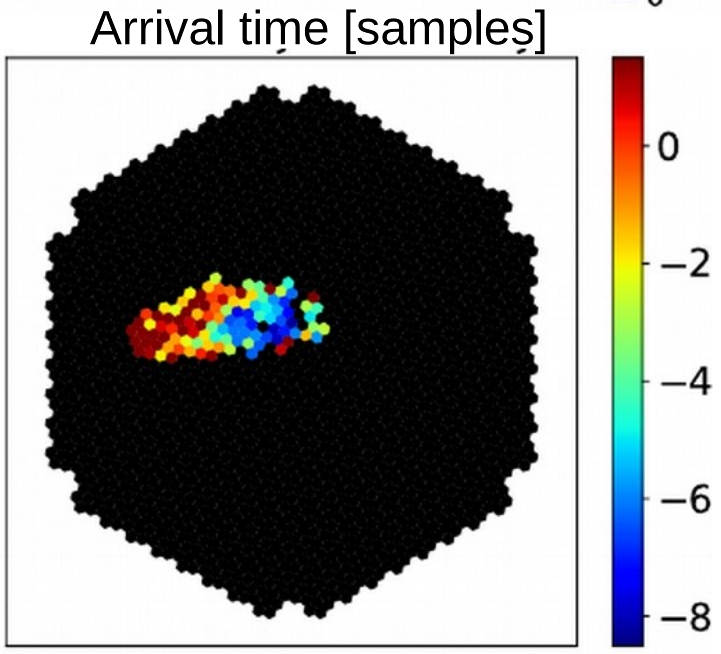
# Problems during the night: car flash



# Image cleaning

- IACT cameras have O(1000) pixels, but in most of the images shower is contained in a dozen – few tens.
- Rest of the pixels have a mixture of electronic noise (small in LST) and NSB/afterpulses
- Cleaning algorithm profits from:
    - Shower signals stick out from NSB (more of higher energies)
    - Shower signals are bundled in neighbouring pixels (but stars can also mimic this)
    - Shower signals have time structure

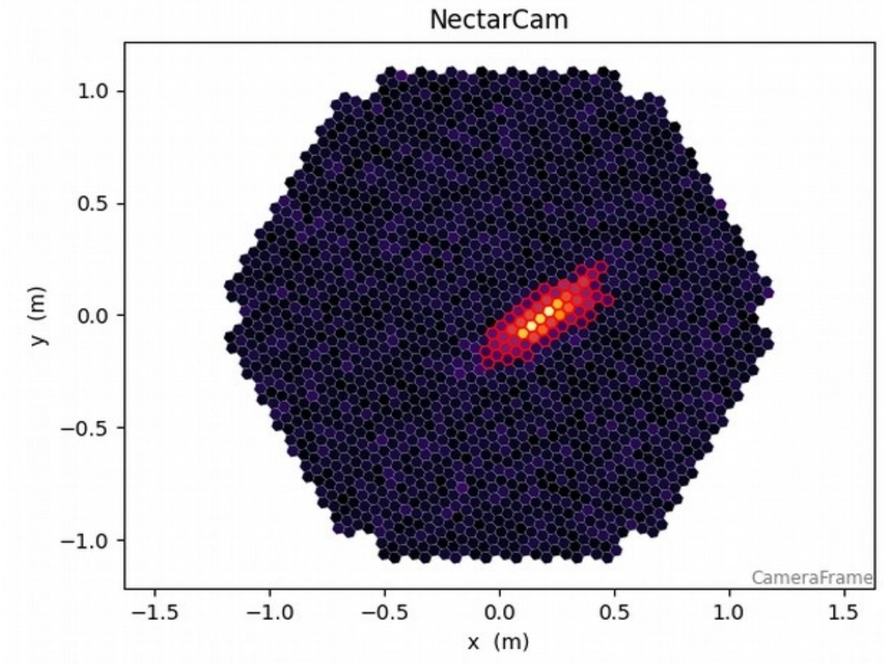




P. Gliwny

## Tailcuts cleaning

- Tailcuts is a classical 2-threshold method
- First core of the image is found with a higher threshold.
- Then additional boundary pixels are added with lower threshold, but next to a core pixel
- Available in ctapipe under image/cleaning: tailcuts_clean



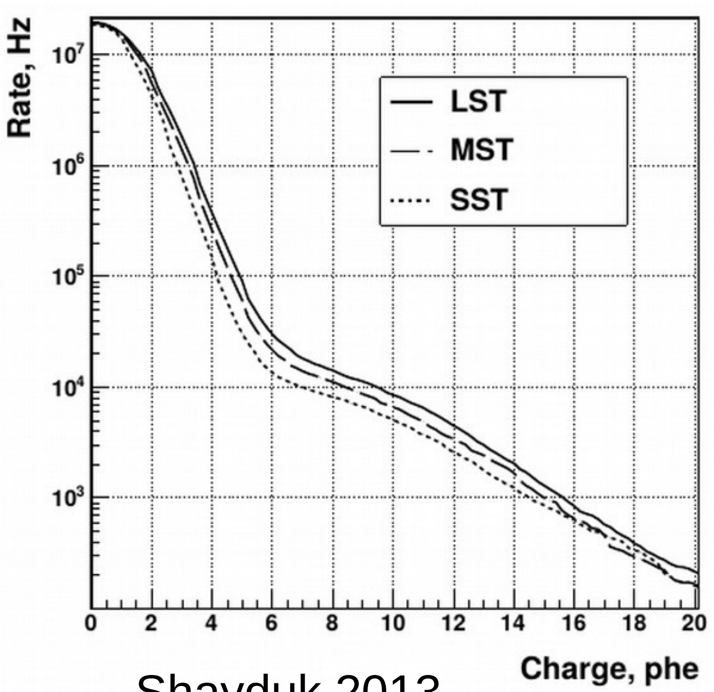
ctapipe docs

## Usage of arrival time

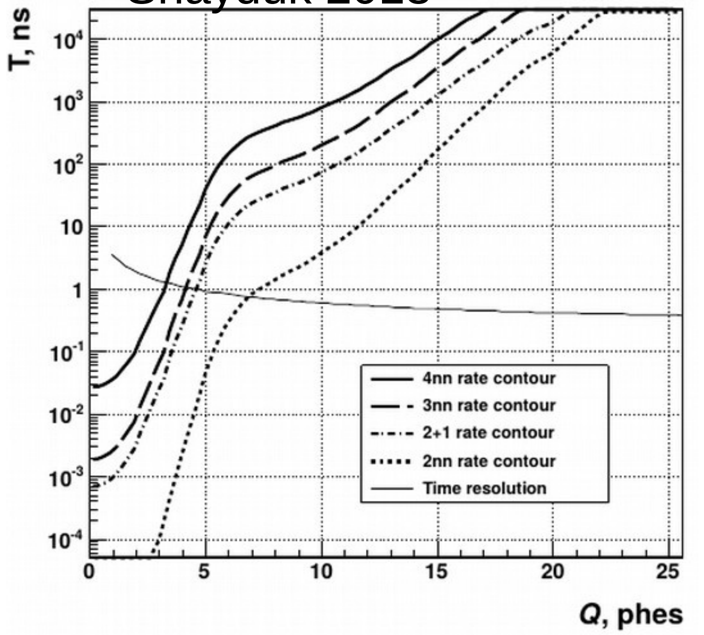
- To lower the energy threshold one needs to lower the charge thresholds in the cleaning (to reconstruct also dim images), but this gets quickly limited by the noise.
- Arrival time is an independent information from charge, and thus by adding time thresholds we can lower the charge thresholds and end up with the same noise
- We first make regular tailcuts clening, but then exclude pixels that do not have a minimal number of pixel neighbours with signals close in time
- In ctapipe/image: apply_time_delta_cleaning

# Next neighbour cleaning

- Time and charge information can be combined in a more efficient way
- Next-neighbour cleaning is using pedestal charge distribution and change-dependent time resolution to compute optimal charge and time thresholds for 2NN, 3NN, 4NN combinations
- Not used in ctapipe, but a similar algorithm is used in MAGIC (and magicctapipe)

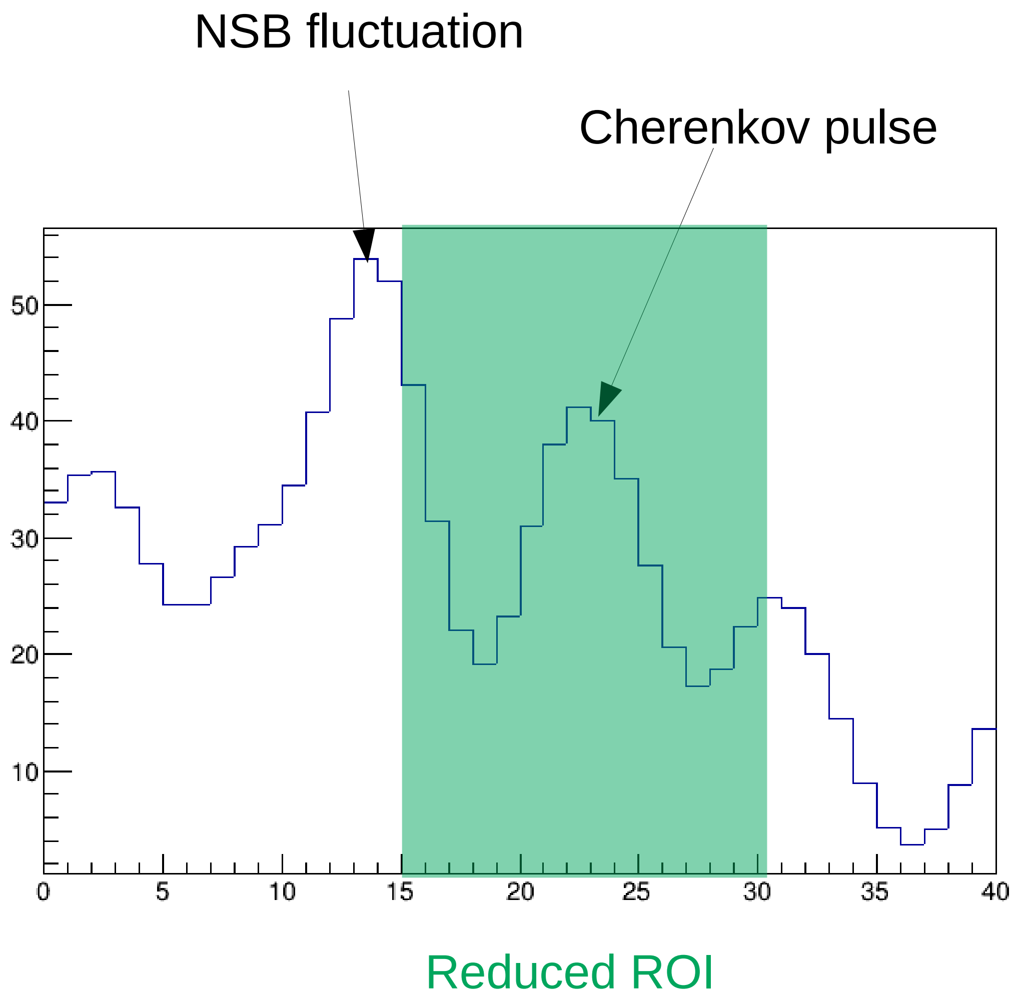


Shayduk 2013



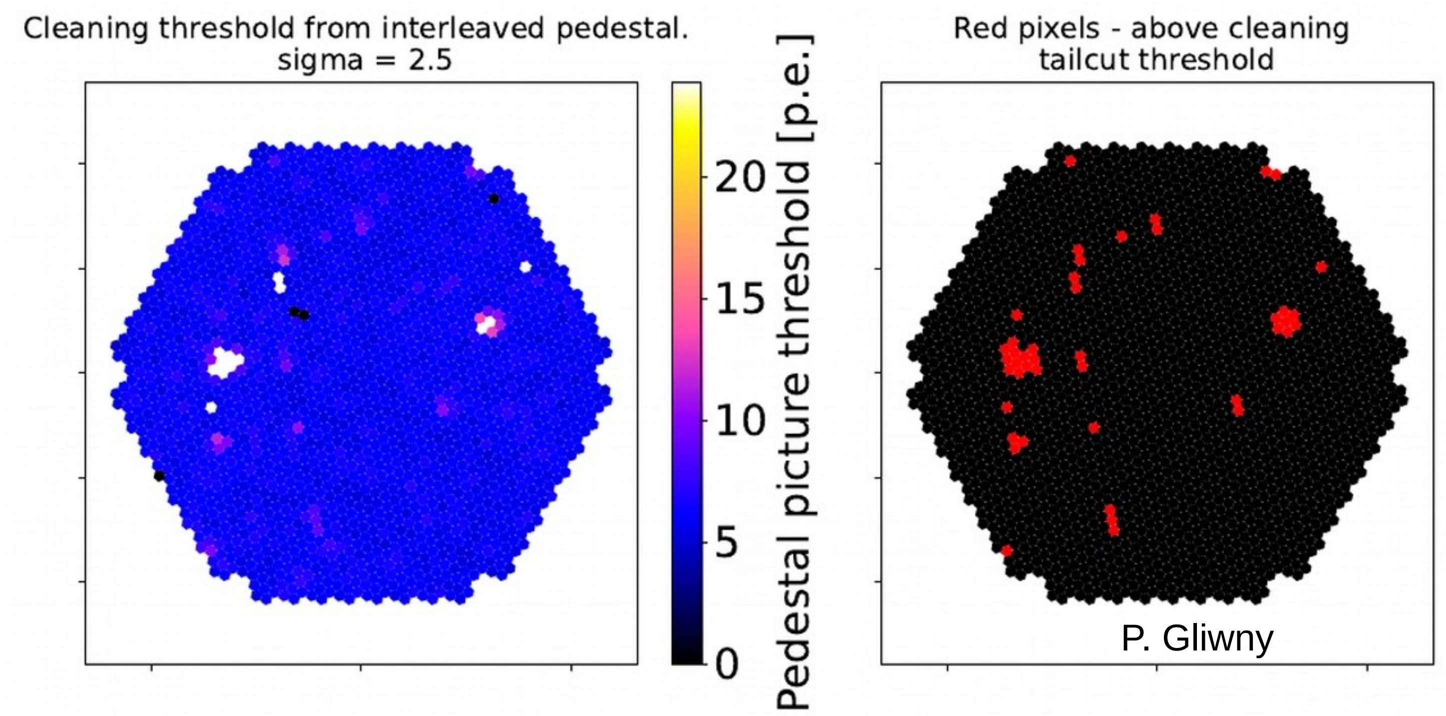
# Two-pass cleaning

- In tailcuts cleaning the charge threshold is reduced for boundary pixels, but the signal might be still missed due to a larger NSB signal somewhere else in ROI
- The idea of the two pass cleaning is to extract first the image with higher threshold, fit the time slope to estimate the arrival time in each pixel, then reextract the signals with reduced ROI
- Not implemented in ctapipe (yet?), but was there in the "Chimp" CTA analysis



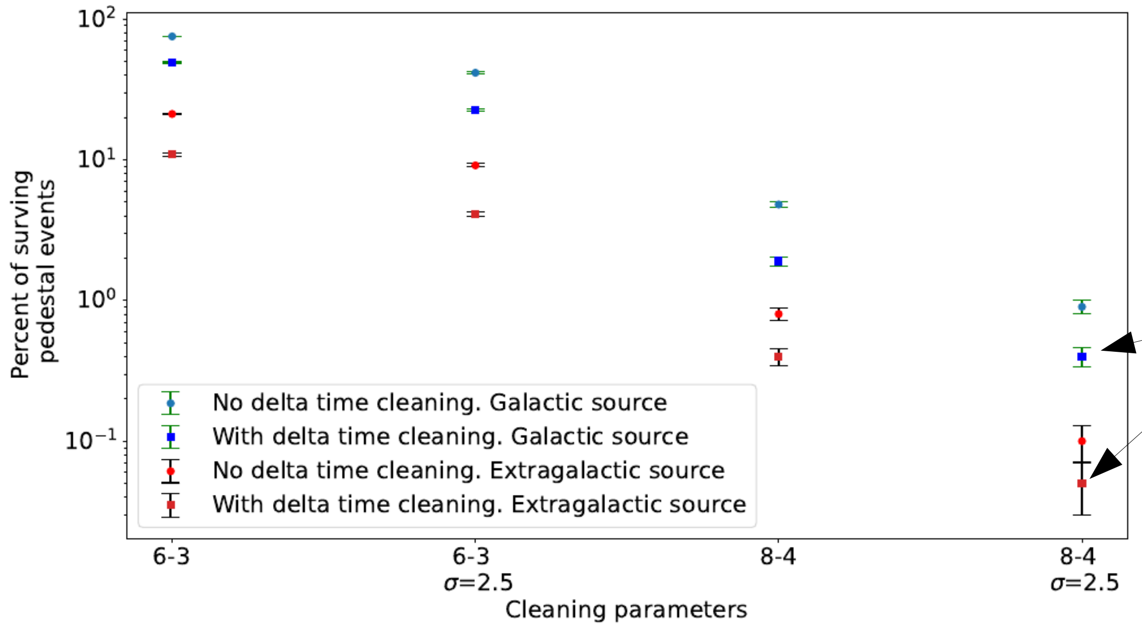
## Stars in cleaning

- Stars normally affect a few closeby pixels and as such can mimic small showers (however with a randomized time of pulse arrival).
- We can increase the threshold of the affected pixels
- Too many pixels with increased threshold would cause data/MC mismatches!



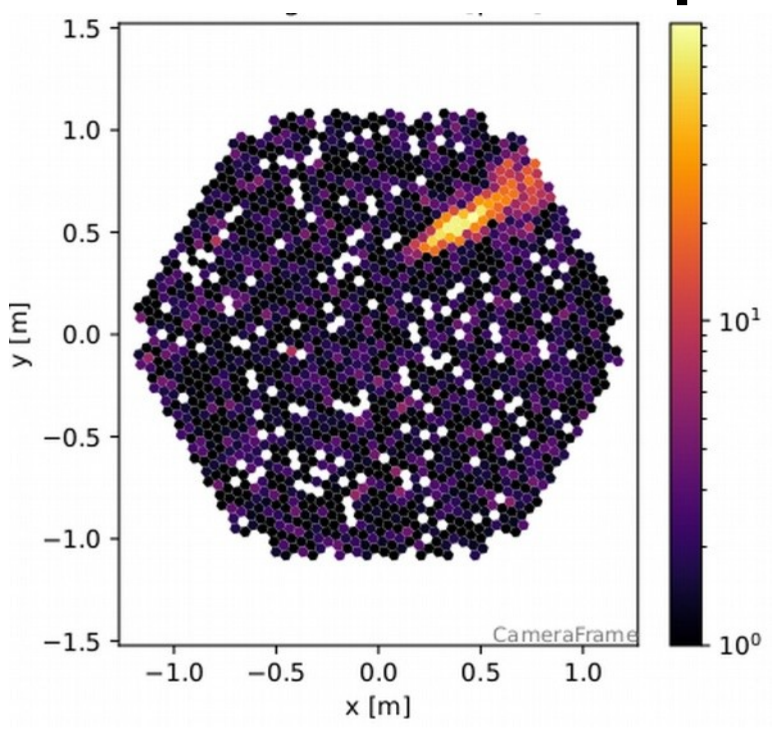
# Judging if the cleaning is fine

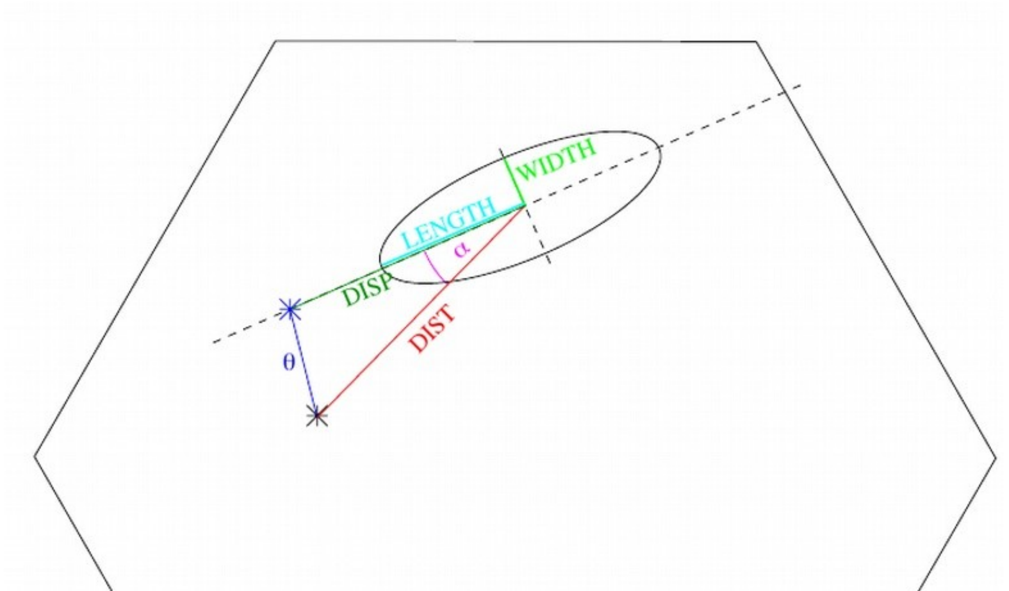
- To check the core thresholds one can apply the same cleaning to empty pedestal events and see how many survive – those would be fake “islands” in the true showers
- For checking boundary conditions it is more tricky and needs a clever MC study
- Optimal thresholds are different for extragalactic, galactic and moon time observations!



LST defaults

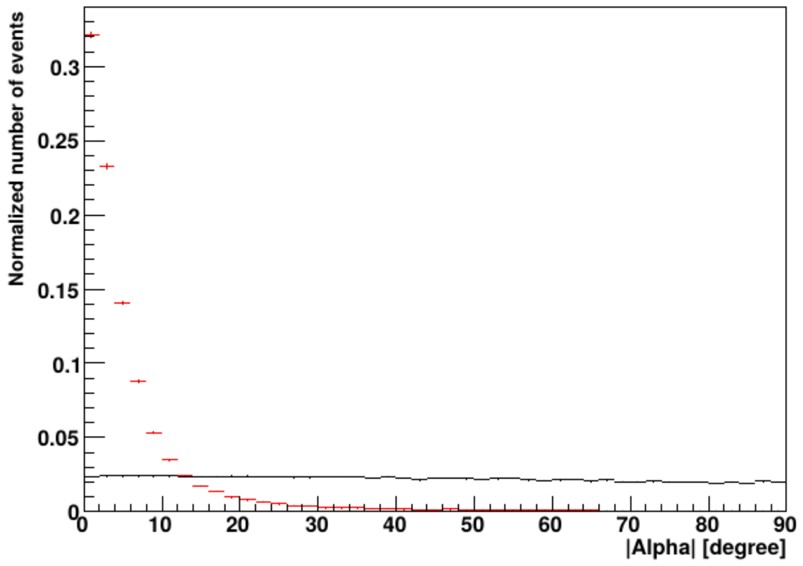
# Hillas parametrization

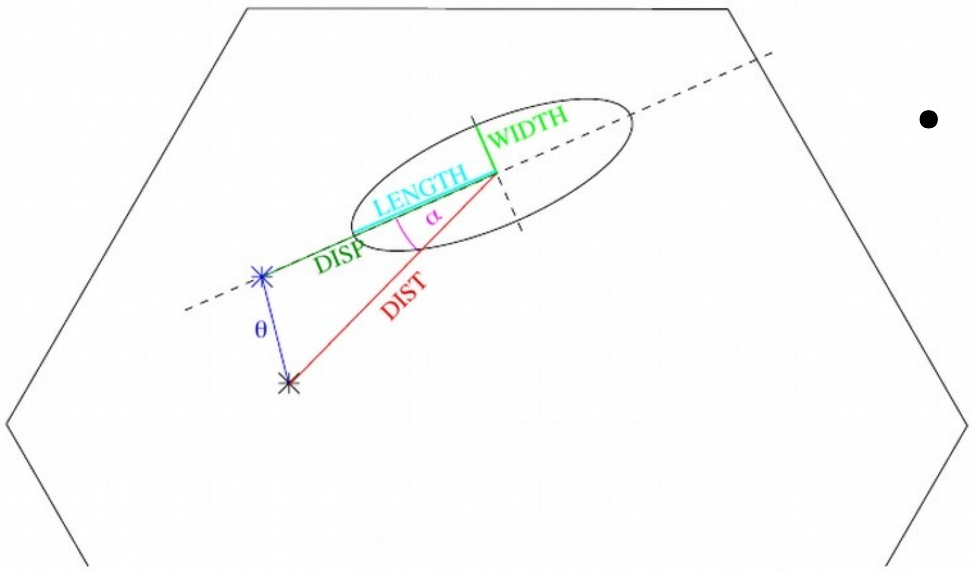




- Images of gamma rays are normally regular and can be roughly described by an ellipse
- Original Hillas parameters are analytical representation of the image as a 2D Gaussian, with its total intensity (0th moment), center (1st moment), half-axis (2nd central moments) and orientation
- In ctapipe/image: hillas_parameters

# Additional parameters (src-dep)





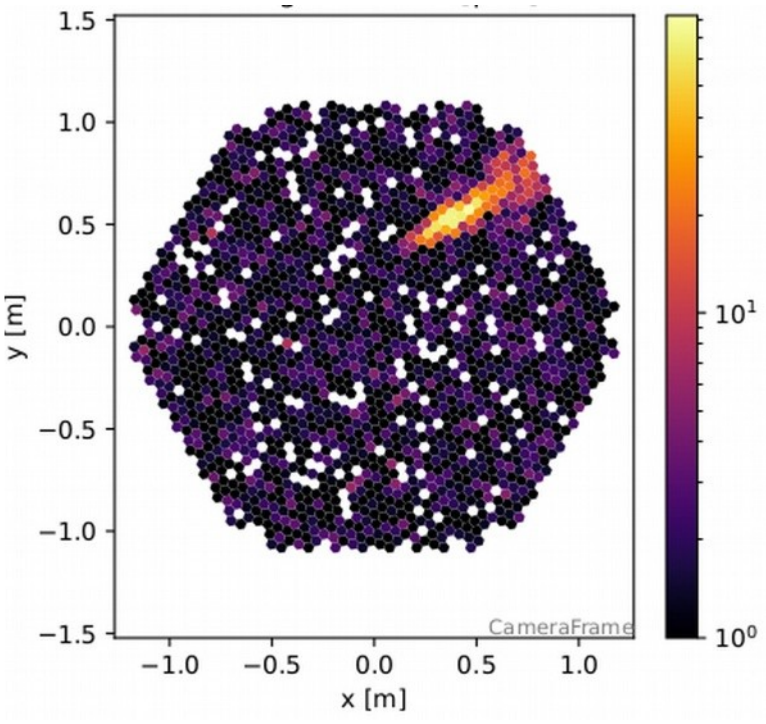
- Source-dependent parameters – depend on the assumed source position
  - Dist: distance of the image centroid from source position
  - Alpha: angle between the main axis of the image and direction to the source
- Since the parameters are source-dependent they are also pointing-dependent, and IACT pointing also requires calibration (e.g. with starguider)

# Additional parameters (higher moments)

- Head-tail asymmetry parameters (to know which end of the ellipse corresponds to top and which to bottom of the shower):
  - Skewness (3th central moment / std dev$^3$ )
- Higher moments of the image:
  - Kurtosis (4th central moment / std dev$^4$ )

## Additional parameters: leakage

- Some of the images are clipped by the edge of the camera. The effect of this is measured by the leakage:
  - pixel_width_[12] – number of pixels in the last 1-2 rings of the camera
  - intensity_width_[12] – corresponding sum of signals in the image boarder pixels

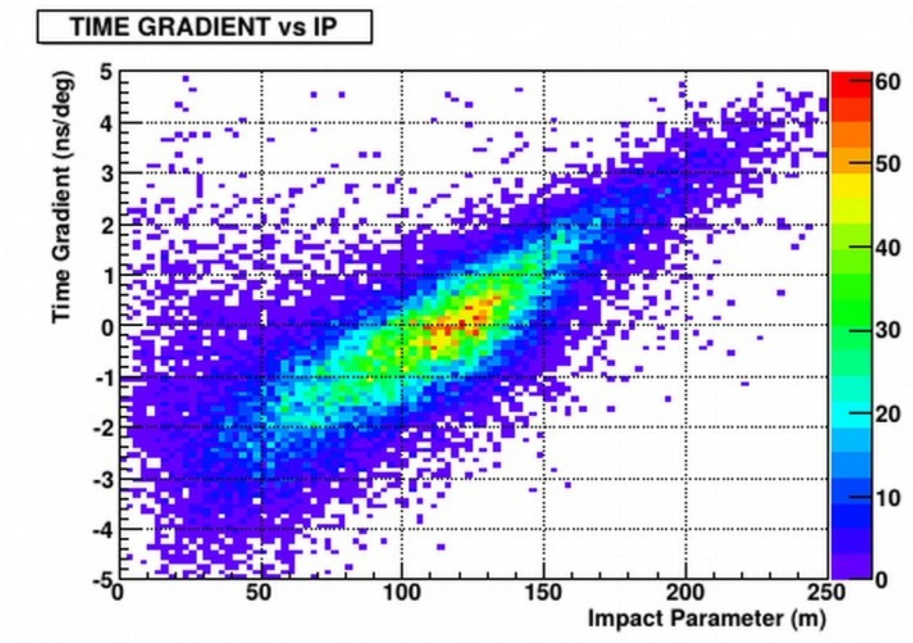


## Additional parameters: concentration

- Gamma ray images have a certain level of “peakiness” - that can be used in the reconstruction.
- Hadrons are more messy and thus usually less concentrated
- Extremely concentrated events can also mean hardware issues
- Concentration parameters:
  - Cog: fraction of p.e. within one pixel diameter of the cog
  - Core: fraction of p.e. within Hillas ellipse
  - Pixel: fraction of p.e. in the brightest pixel
- The definition makes it possible to have some unnatural values of concentration (<0 or >1), in particular if the COG is not within the image (e.g. there are two islands)

# Additional parameters: timing

- Timing not only helps with image cleaning, but also adds the third dimension to the image.
- Time gradient (slope) measured along the main axis of the image is strongly correlated with the impact parameter.



Aliu et al. 2008

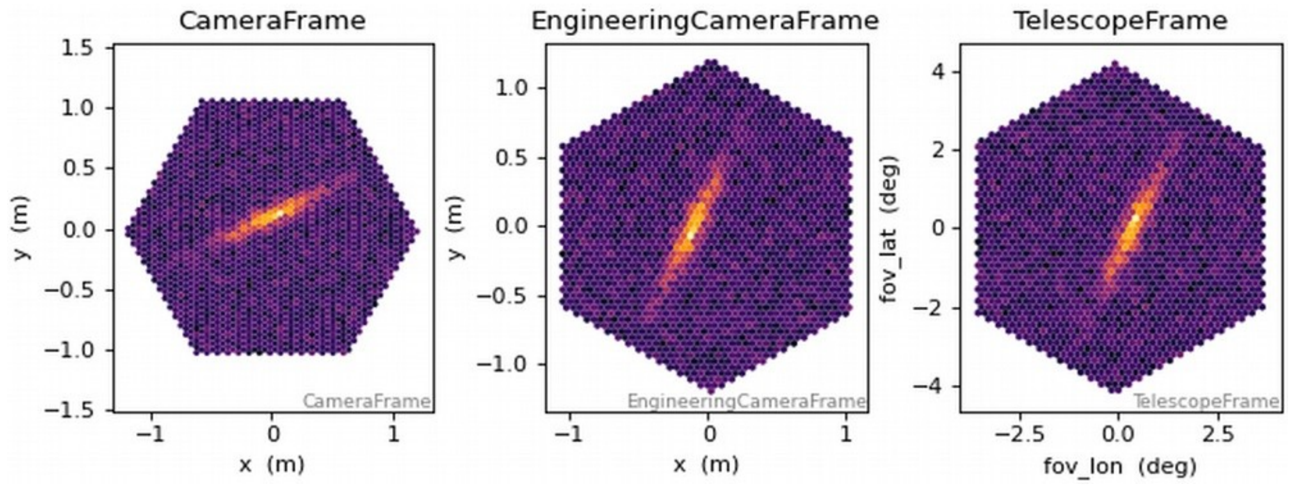
## Additional parameters: image morphology

- Hadronic showers are more messy, therefore it is also worth to describe the morphology of the image:
  - Number of pixels
  - Number of islands (isolated groups of pixels)

## DL1 files

At this point we are at DL1 level: we have images (or drop them to save space) and they are cleaned and parametrized

# Reference frames



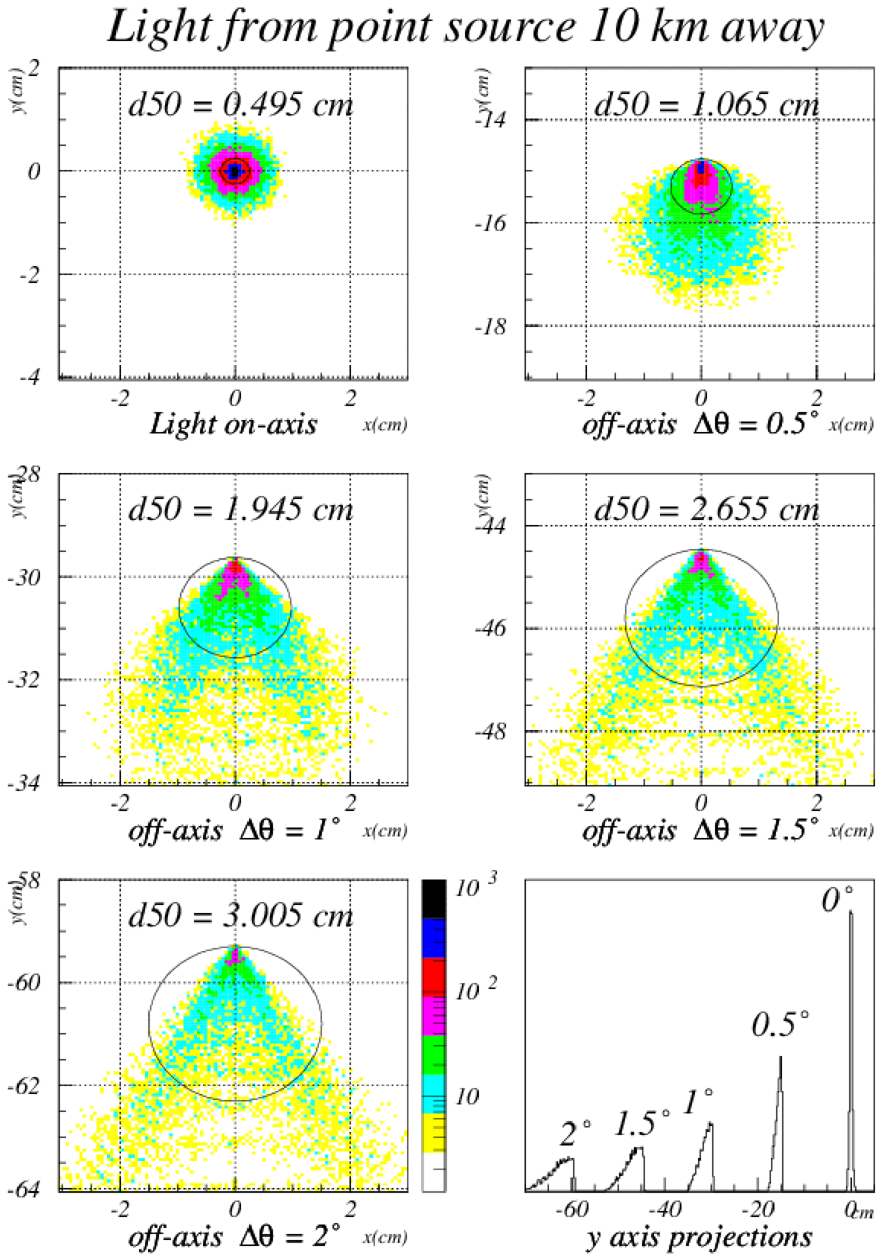
Ctapipe docs

https://ctapipe.readthedocs.io/en/stable/api-reference/coordinates/index.html

- Most of the image parameters will depend on the reference frame
- CameraFrame **[m]** is ctapipe version of sim_telarray standard reference frame
- EngineeringCameraFrame **[m]** is MAGIC-like “natural-view” camera frame with a transformation of coordinates x’=-y, y’=-x
- TelescopeFrame shows angular coordinates on the sky

# Using reference frames

- When dealing with single telescope parameters, e.g. for gamma/hadron separation, you can use them in any frame you like (just **do not mix them with different frames**!) - you can recognize the reference frame by the container name (e.g. HillasParametersContainer vs CameraHillasParametersContainer) or by units.
- When you start the reconstruction of the arrival direction, you need to move to sky coordinates, and for the conversion focal length of the telescope is used
- In addition to the nominal focal length there is also “effective” one that corrects for the outward distortion of images due to coma aberration (~4% effect for LST) – **do not mix us telescope frames with different focal lengths**



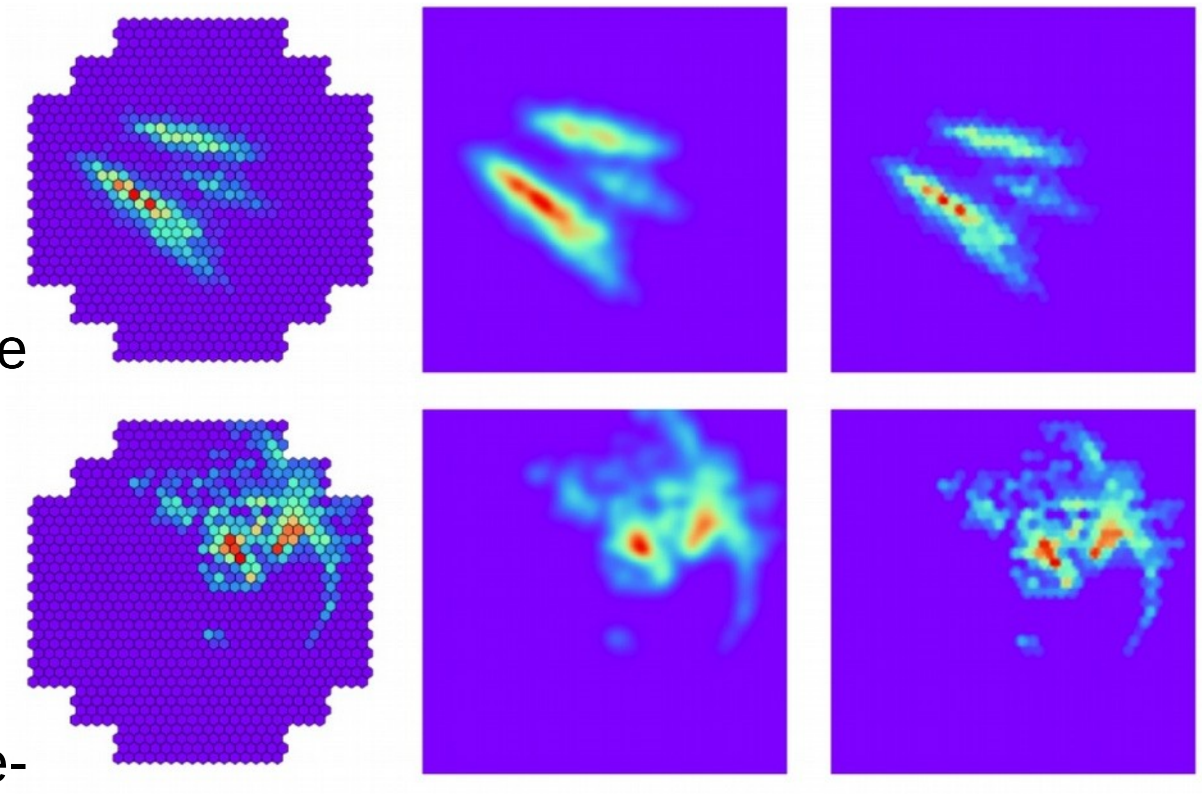
A. Moralejo (MAGIC reflector manual)

## Going beyond Hillas

- Hillas parametrization is 40 years old, and in addition to simply adding more parameters there are also completely different approaches:
  - CNNs – using full information of the image in deep learning approach – many different solutions in CTLearn package
  - Model analysis – using templates (MC-based or semi-analytic) to fit image (LHfit in lstchain is one of such analysis chains). Some of them provide directly shower physics parameter (e.g. depth of the first interaction)

# Beyond Hillas - CNNs

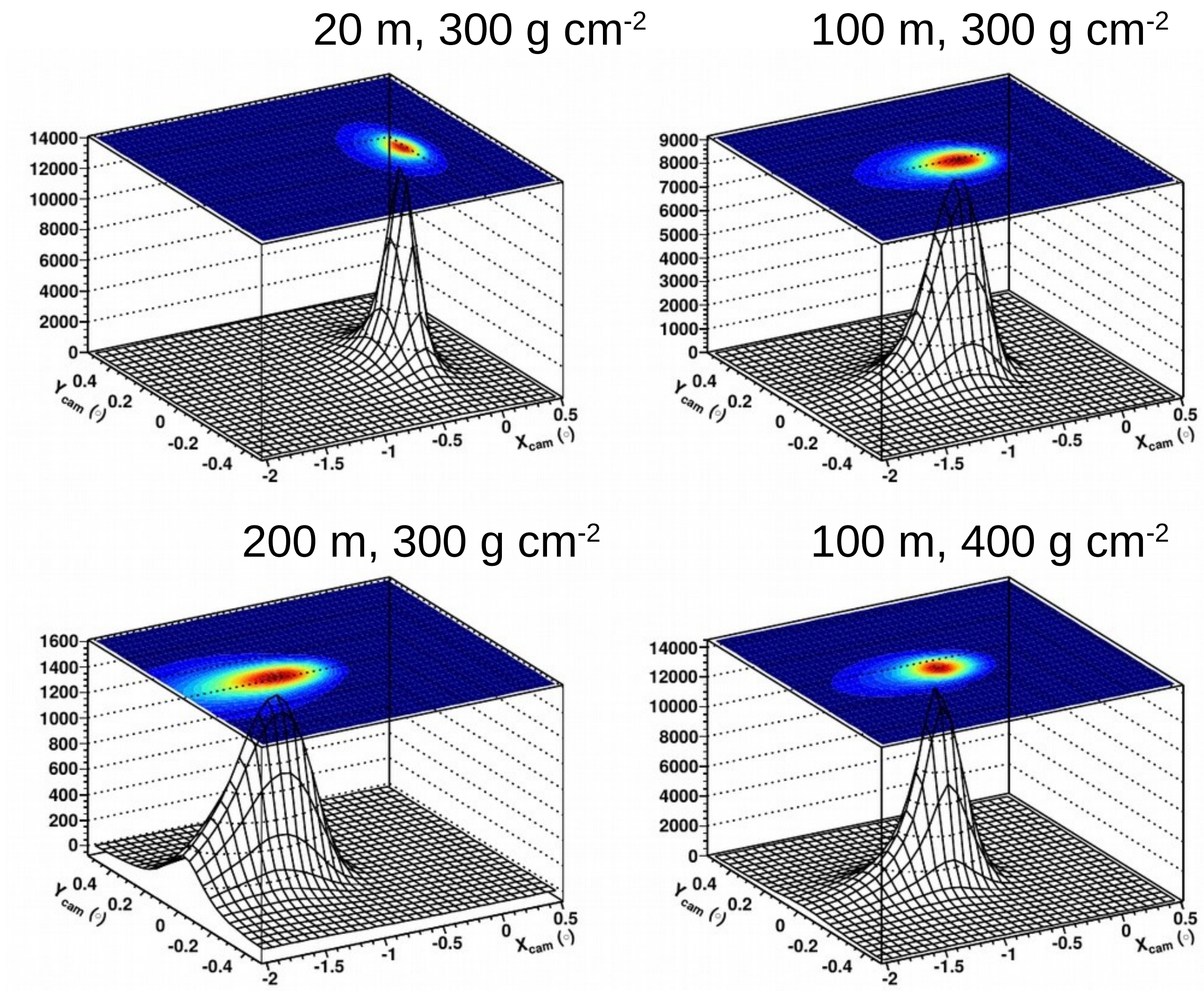
- Typical problem of finding known patterns in images
- Allows to use the full information about the event
- But:
  - More then one image: multiple input layers with different telescopes, charge and time
  - Most of CNN methods use square pixelization, while IACTs often use hexagonal pixels, conversion is resource-hungry and can produce artifacts. **There are also methods tuned for hexagonal pixels.**



Shilon et al. 2019

# Beyond Hillas – model analysis

- Two-dimensional image templates as a function of basic shower parameters (zenith, azimuth, energy, impact, first interaction or height of the shower maximum)



Parsons & Hinton 2014

## Going beyond Hillas

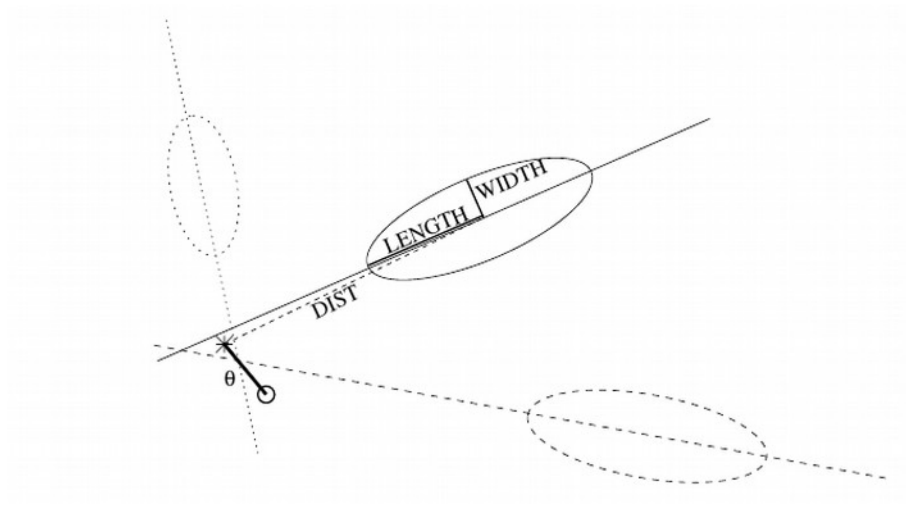
- Beyond-Hillas methods have a very high potential and in some cases their performance was proved to surpass standard Hillas analysis, but:
  - They are much more resources-hungry (slow, sometimes require GPUs, …) - this is an issue due to large LST rates (~10 kHz)
  - Most of the information about the shower is concentrated in a small fraction of image pixels, while most of the pixels only carry NSB – even small error in the description of noise in MCs can produce large data/MC mismatches
- Always apply with caution!

## Stereoscopic reconstruction

- With more then one telescope (or one telescope and some tricks) we can move from “flat” camera images to full 3D information about the shower. This works like our eyes
- The objective is:
  - (First) estimation of the arrival direction of the shower
  - Estimation of the shower impact point (distance from the telescopes) and height of the shower maximum.

# Axis crossing method: on the camera

- If we stack up images from all the telescopes that saw the same event we can get the first estimate of the arrival direction of the primary gamma ray
- Important to take into account the relative pointing corrections of all the telescopes!

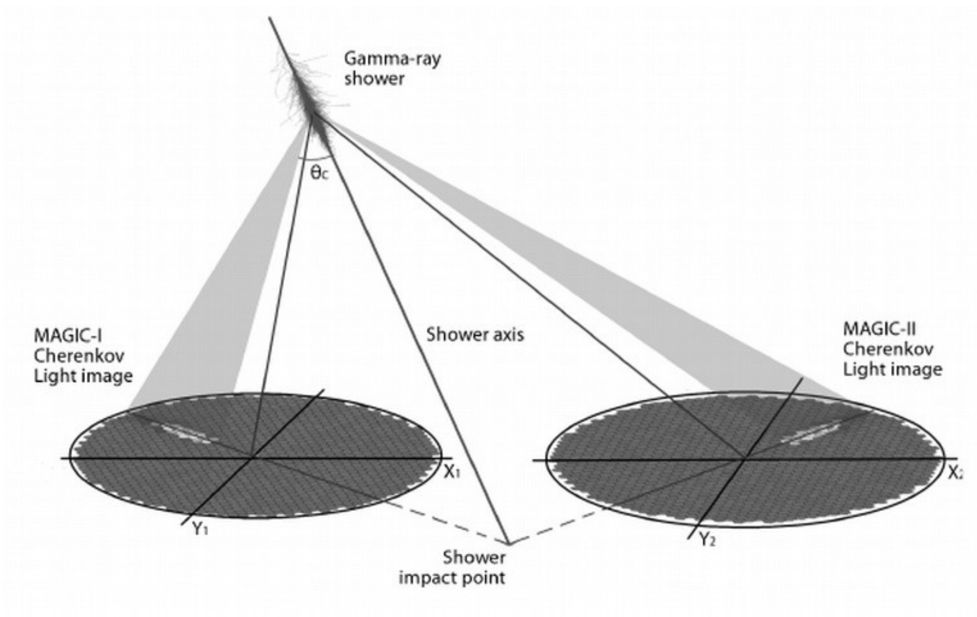


JS & G. Giavitto 2024

In ctapipe/reco:
HillasReconstructor

# Axis crossing method: on the ground

- Each pixel in the camera corresponds to one direction, so each line to a plane
- Crossing of the planes corresponding to the main axis of the images gives the estimation of the shower axis
- Crossing shower axis with the ground gives the impact point

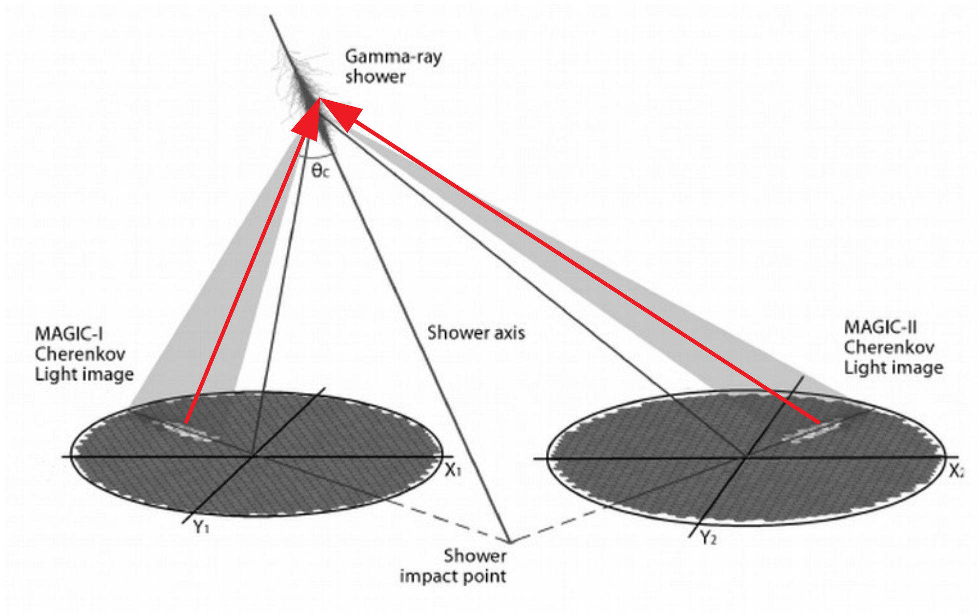


JS & G. Giavitto 2024

In ctapipe/reco:
HillasReconstructor

# Height of the shower maximum

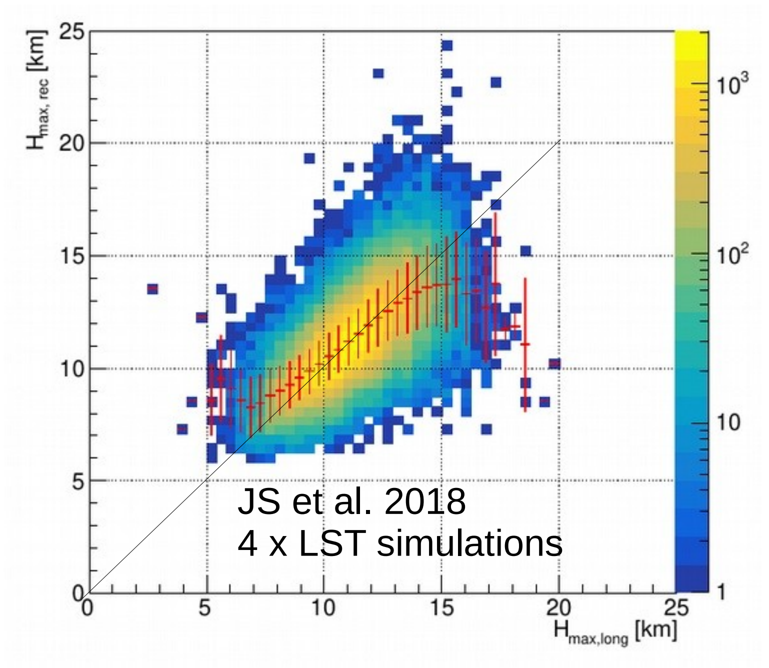
- Having the geometry of the shower axis reconstructed, it is easy to get the height of the shower maximum ($H_{max}$) from the directions in the sky corresponding to COG in different images



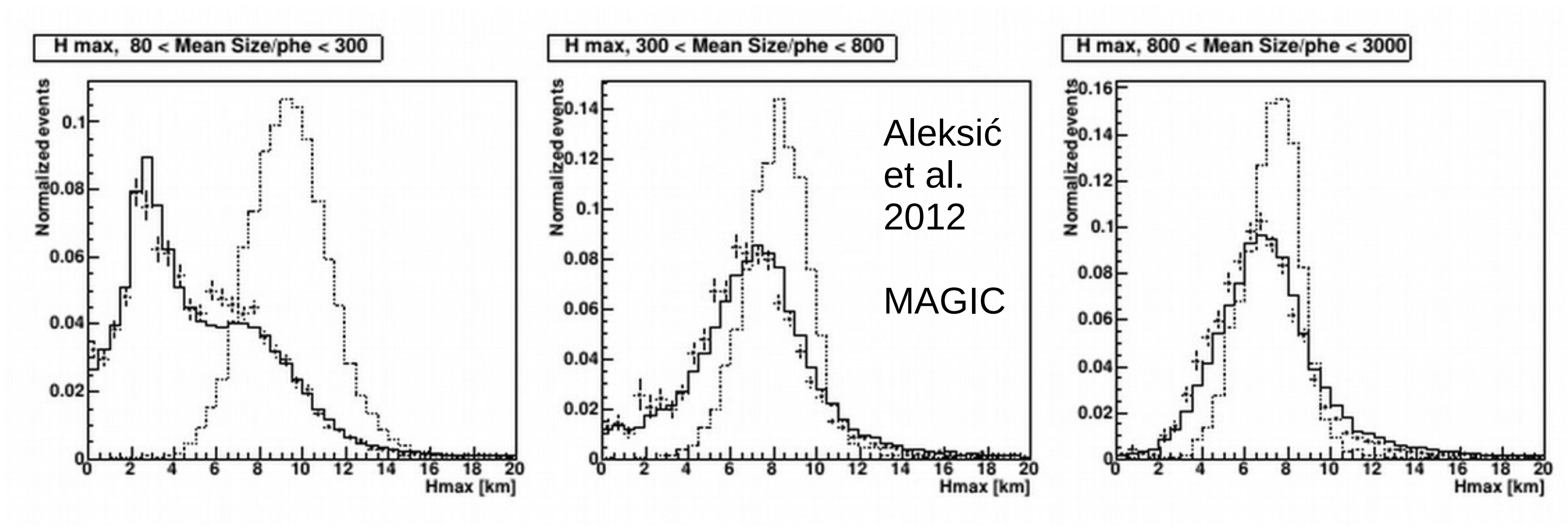
JS & G. Giavitto 2024

# Height of the shower maximum

- In shower physics the height of the shower maximum is the peak of the longitudinal distribution of the shower particles
- In IACTs we see the emitted Cherenkov light (which is roughly proportional to the number of charged particles above a given threshold) and in addition some of this light is absorbed
- Still, the IACT reconstructed $H_{max}$ turns out pretty close to the Corsika definition.



# Why $H_{max}$ is important?



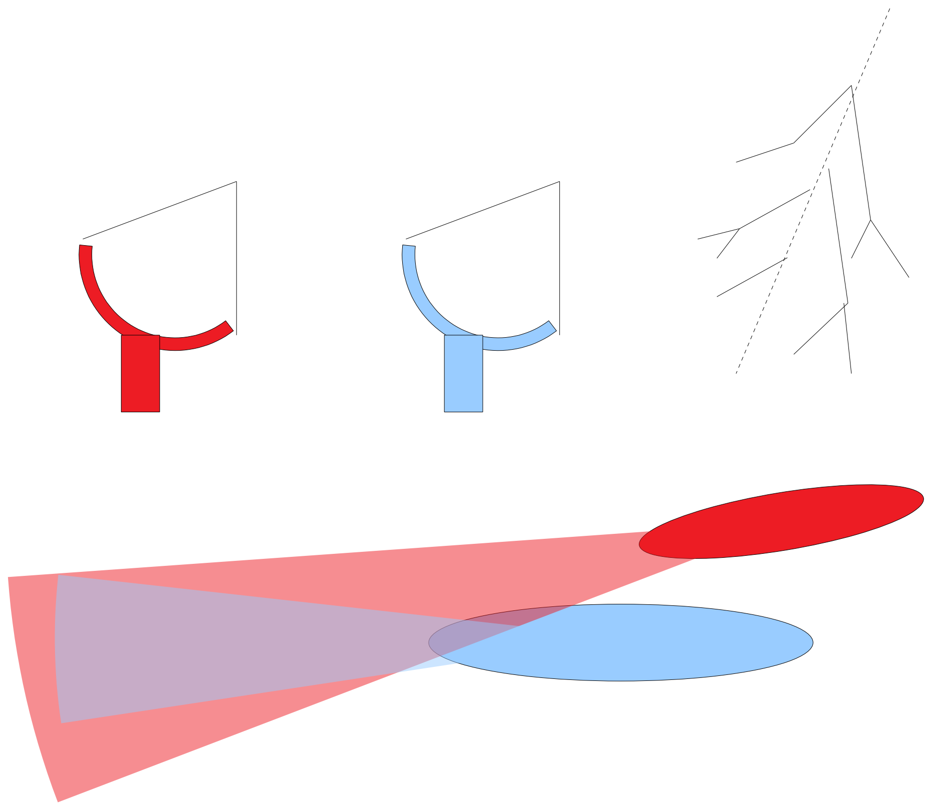
- $H_{max}$ is a powerful gamma/background separation parameter:
  - Mean free path for first interaction for protons is larger then for gammas, and in addition to $\pi^{+/-} \Longrightarrow e^{+/-} + \ldots$ you also get $\pi^0 \Longrightarrow \gamma\gamma$ that still need to interact to produce charged particles – proton showers develop deeper in the atmosphere
  - Distant (~100 m) single muons have elliptic images, that look like low energy gamma rays, but their light is reconstructed at ~2 km above the telescope, and thus such events are easily removable
- How deep the shower develops in the atmosphere also influences the amount of light that reach the ground – important for energy estimation

## Two vs more telescopes

- If there are only two telescopes there is a unique solution for arrival direction and impact point
- If there are more telescopes, all of the lines (planes) will not cross in one point (line), but should still be closeby.
- Typically we apply all the possible crossings, and make a weighted average (HillasReconstructor applies weights in direction reconstruction, but not in the core position reconstruction)
- The weights normally favour images that are easier to reconstruct: with higher intensity, length/width ratio, etc.

# Parallel images

- If only two telescopes see the event and the impact point of the shower is aligned with the line connecting the telescopes the resulting images will be nearly parallel
- This makes a big uncertainty in the derivation of the crossing point and consequently in the calculation of the stereo parameters

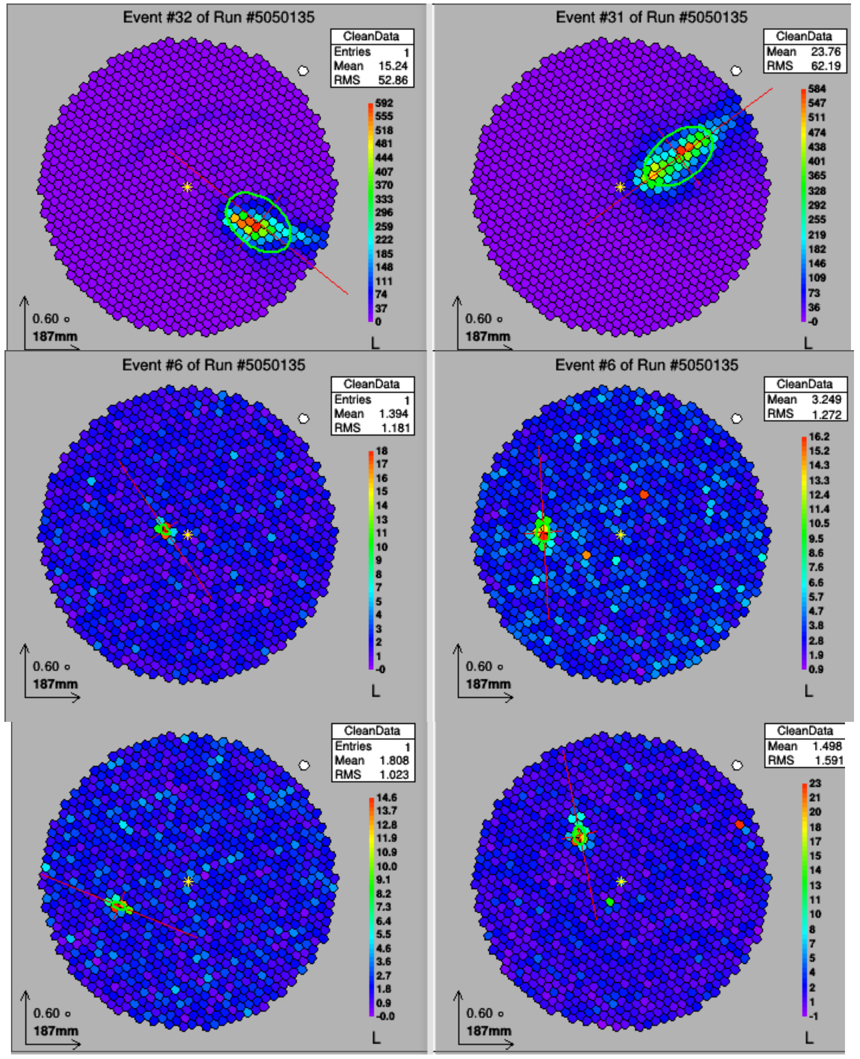


## Reconstruction of the event

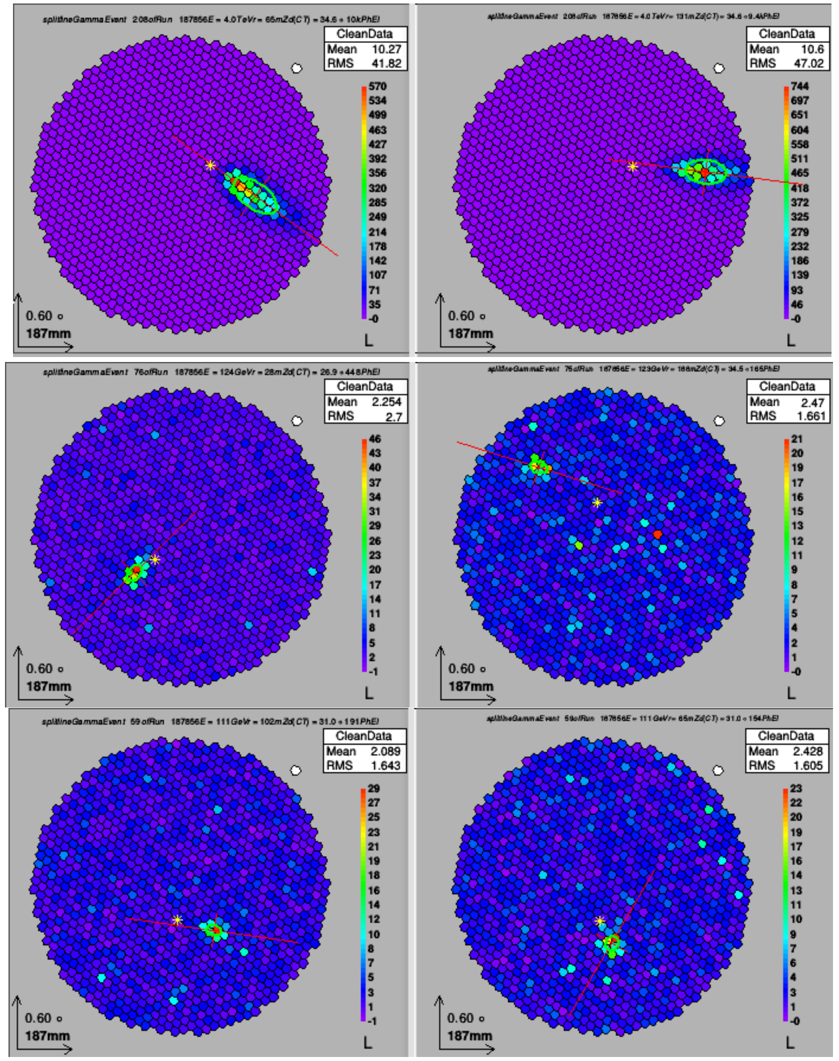
- The task is like figuring out the type of the car, its direction and speed from the observation of its lights...
- At this step we have all the images parametrized, and tentative reconstruction of the shower geometry
- We want to reconstruct:
  - Particle type: gamma or background
  - (Improved) arrival direction
  - Energy estimation

# Gamma/background separation

- For each gamma ray we have 1000 protons – **needle in a haystack**
- For large showers you could do it even by eye!
- For small showers we can only set a likelihood of the event being one type or another



Background events



Gamma ray

# Monte Carlo comes to the rescue

- Shower physics is already complicated by itself and can be expressed analytically only with coarse assumptions and simplifications
- Moreover, what the telescopes see is further folded by the emission and propagation of the Cherenkov light and by instrumental effects
- Monte Carlo simulations are essential for IACT analysis
- We simulate a large library of showers with different energies, distribution of impact points, etc. and use it as a **labeled dataset** for **supervised learning**.

# Processing Monte Carlos

- Monte Carlo simulations of a shower are done with Corsika, and the response of telescopes is simulated with sim_telarray – at its output you should have raw-data-like information (plus labels: energy, true direction, …)
- Ideally the MCs should carefully reproduce all the artefacts caused by the instrument, such that the same analysis is applied to it as to the data.
- In reality it is impossible to simulate everything and simplifications are made. In some aspected MCs are produced as “spoiled” to reproduce as closely as feasible the actual technical performance of the telescope (residual accuracy after all corrections)

# Example simplifications in MCs
## (on the example of LST and sim_telarray)

- No electronic noise correlation in neighbouring samples is simulated ==> effective (uncorrelated Gaussian, but with higher spread) noise is used
- No DRS4 pedestal and timing features are simulated ==> uncertainties in baseline and arrival time are added instead
- No F-factor method is applied to MC calibration ==> the calibration exploits the assumed single p.e. pulse shape and applies a fixed randomization to limit the expected accuracy

# MC simplification and MC/data mismatches vs analysis methods

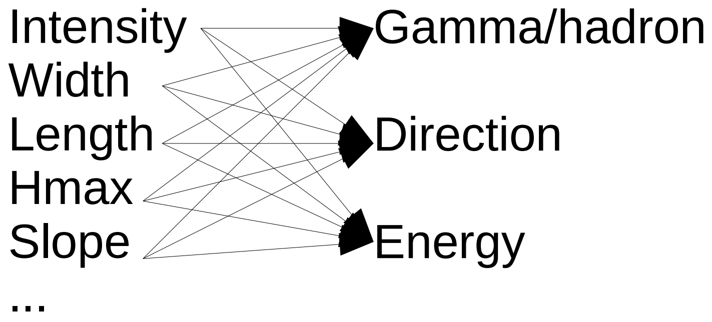
- When inventing/developing a new analysis method you always need to think if the MC/data mismatches will not make it not feasible.
- Example 1: Imagine that there are constant (but different in each telescope) differences between the average arrival time of the triggered events in data and MC
  - In standard analysis you do not care (as long as the signals fit in ROI
  - But if you introduce a new parameter: telescope to telescope time difference they will completely mismatch

# MC simplification and MC/data mismatches vs analysis methods

- Example 2: You have a data set with a rather strong star
  - In standard analysis (if you apply relative cleaning update to the image cleaning) the star pixels have higher threshold – you are only slightly affected (small showers on top of the star)
  - But if you work on a new CNN-based analysis using the full camera information: the star pixels will appear in most of the events faking a hadron-like island.
- **We normally develop first the analyses on MCs, but validation on real data is essential**

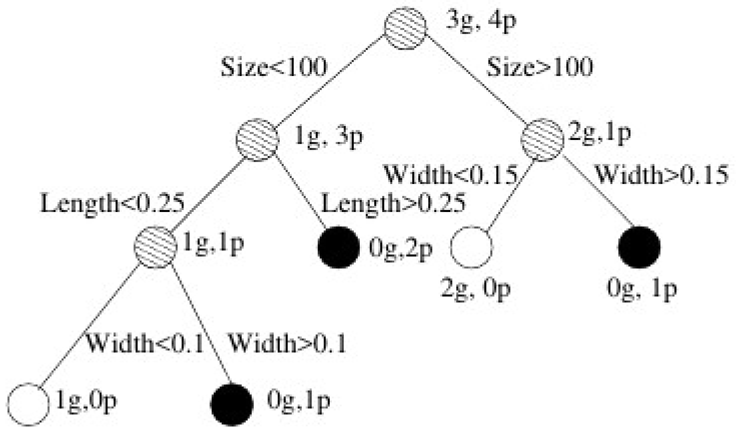
# Parameter-based estimators

- We have multiple parameters. Some of them are the basis of the estimation of a particular variable (intensity for energy, Width and Hmax for g/h separation, Length/Width for direction), but other parameters help as well.
- Classically IACTs used sets of cuts, LUT, fit functions, …  for estimation
- Nowadays the most common way is to use machine learning methods, in particular Random Forest (but also e.g. Boosted Decision Trees, ANN, …)



# Random Forest

- Constructed as a set of trees starting with randomly drawn input subsamples
- Individual (decision) trees are grown by selecting randomly a subset of parameters and selecting among them one that gives the best separation
- Cutting is continued until the end nodes are composed of single particle type or too small



JS & G. Giavitto 2024

# Random forest: regressor vs classifier

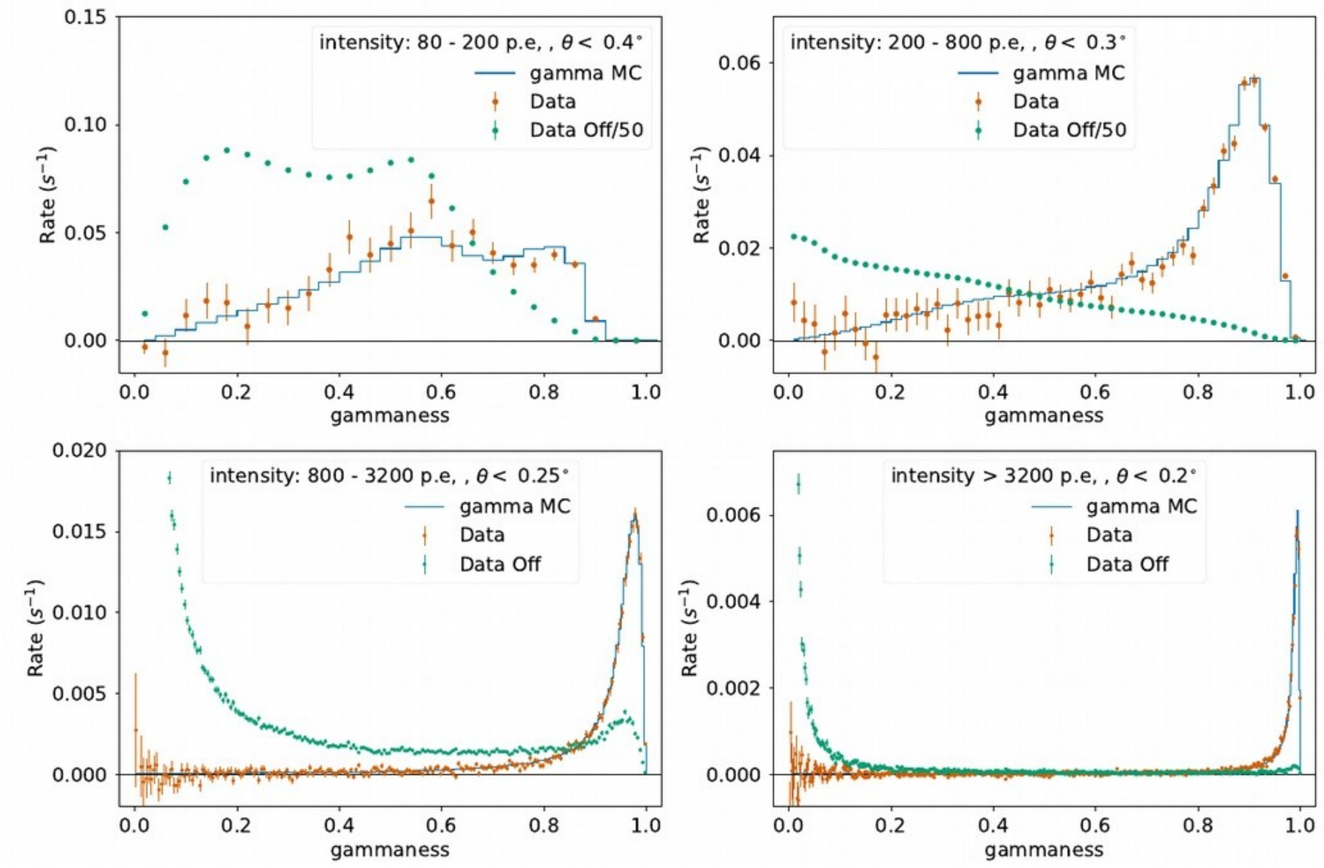
- cta-lstchain is using sklearn.ensemble RandomForestClassifier and RandomForestRegressor
- Classifier is used for gamma/hadron separation , input: (labeled) MC gammas and protons
  In LST-1 also used for head-tail discrimination
- Regressor is used for continuous variables: energy and DISP (for arrival direction), input MC gammas with labels of energy, DISP, etc.
- They differ in node split criterion (Gini index, squared error)
- A bunch of options can be selected: number of trees, how far should they grow, number of random parameters checked at each step, ...

# Energy estimation

- Based mainly on intensity and impact parameter
- Since it has a few orders of magnitude spread normally we train for log of it
- Training is typically done on gamma rays, so the resulting energy only works for them, background events will have it underestimated by a factor of a few (for standard analysis no problem, but if you want to work with proton energies, you need to train the energy estimation with protons and possibly other elements)
- It is often computed as the first one, because it can be used in estimations of other parameters

# Gammaness

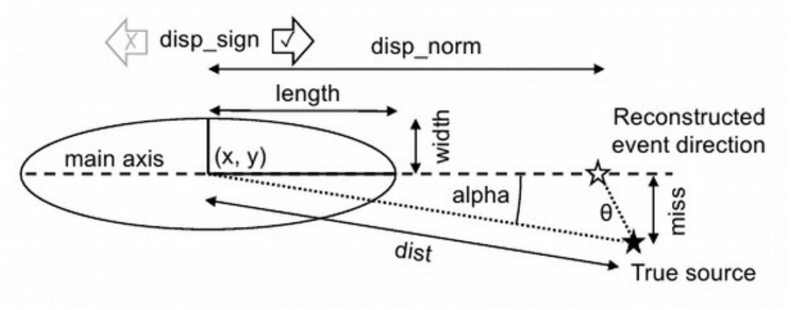
- Higher energies (larger images) separate better: a cut excludes most of protons while keeping most of gammas
- At low energies the distributions heavily overlap



Abe at al. 2023a

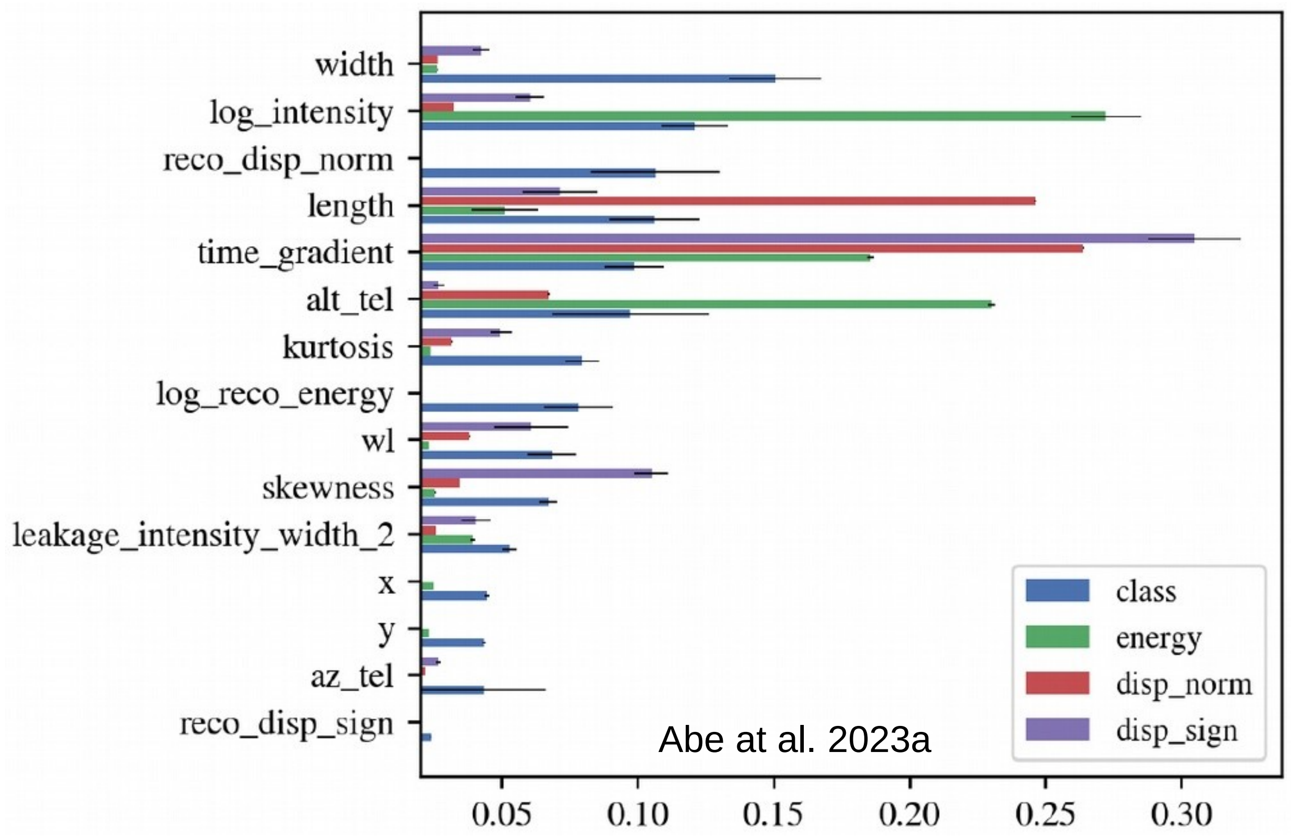
# Arrival direction

- On single telescope, with a “disp” regressor we can estimate the distance (measured along the main axis of the image) to the source
- Disp is mostly derived with length/width (more elongated images are farther from the source) and time gradient, also leakage is important
- Based mostly on the “signed” parameters: time gradient and skewness parameters we can estimate on which side of the image the source is.



Abe et al. 2023a

# Example from LST1 – importance of individual parameters



A bit simplified view, since:

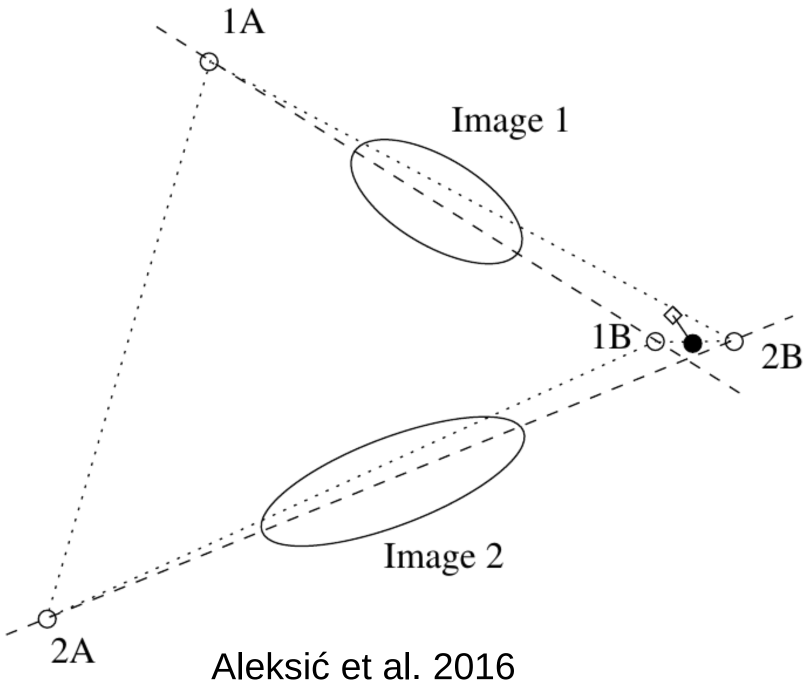
- The importance of parameters will vary with energy,
- Some of the parameters are correlated
- The importance also depends on observation conditions, e.g. zenith

## Averaging of the telescopes

- For CTA it is impossible to use the image parameters from different telescopes together, because different events will be triggered by different telescopes
- Estimation is done telescope-wise using image parameters from this telescope and common stereo parameters
- Then a second step of averaging (for gammaness and energy estimation) happens
- Nuances:
  - Weights needed for optimal performance
  - Energy should not be averaged linearly but in log scale

# Averaging of the arrival direction

- Occasionally the head/tail discrimination fails and then the source is estimated ~1 deg away!
- You cannot simply average the positions from all the telescopes, because even a single fail would largely spoil angular resolution
- Different approaches:
  - Stick to axis crossing
  - Check all the pairs and select the closest one (used in MAGIC and magicctapipe)



Aleksić et al. 2016

## DL2 files

At this stage we should have DL2 files: data with reconstruction of the intrinsic gamma-ray event parameters

## Next step

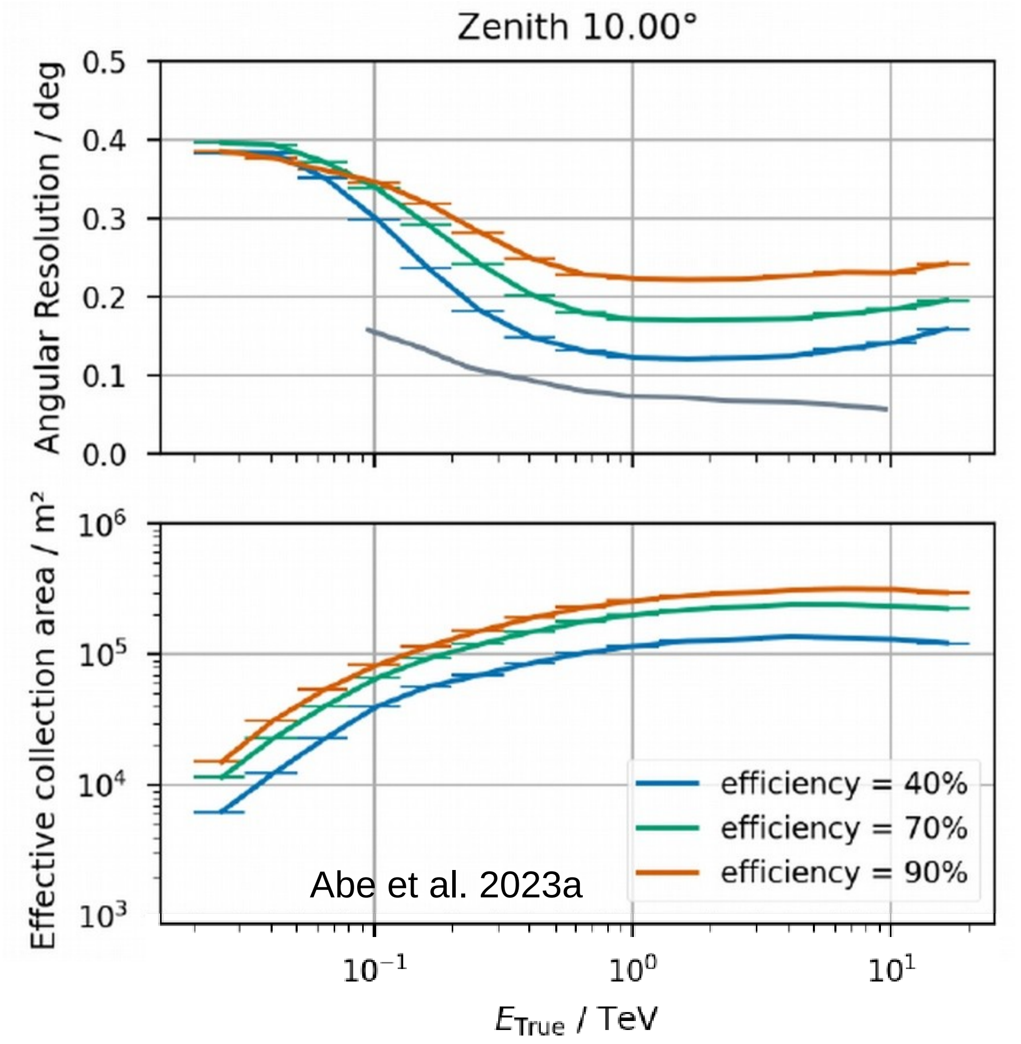
- In order to proceed later on with the high-level analysis we need to:
  - Select only gamma-like events
  - Evaluate the instrument response functions (IRFs) to know
    - How many gammas we catch
    - Bias and energy resolution
    - (Bias and) angular resolution
    - (possibly a background model)
- Since IRFs change along the observations, the corresponding ones need to be joint to the data

## Monte Carlo simulations (again)

- At this analysis stage we again need MC simulations to evaluate IRFs (and possibly to derive optimal cuts)
- Even while the “test” MC samples should be processed in the same way as the used before “train” samples, they cannot be the same files.
- **Evaluating IRFs on the same files as used for the training would bias them: they would be reported too good!**

# Gammaness cuts

- Gammaness by itself does not have any physical meaning – the values of gammaness will strongly depend on the training
- More "physical" is efficiency of the cut, i.e. a cut that keeps a certain percentage of the gamma-rays
- Efficiency cuts needs to be computed independently in each energy bin
- With stronger (more restrictive) cut:
  - You exclude more background
  - You might improve signal-to-background ratio
  - Significance or signal-to-noise ratio might increase or decrease
  - You decrease the number of gamma rays used in analysis
  - Usually you improve other performance parameters (angular&energy resolution)



## Strong cuts vs weak cuts

- Selection of optimal cuts normally depends on the exact use case:
  - Using MC simulations one can derive the cuts that maximize the sensitivity, but it will depend on the energy bin/range/spectrum
  - At the highest energies, or when looking for short timescale looser cuts are better: there is little background, but every photon count
  - Looser cuts are also less prone to systematic errors due to data/MC mismatches
  - At the lowest energies, and also for extended sources stronger cuts can work better: they will improve signal-to-background ratio (and thus decrease systematics)

# Cuts and trials

- If you want to detect a new source and hunt for 5$\sigma$ each set of cuts that you try counts as another “trial”
- Significance needs to be corrected for the number of trials (using integrated Gaussian distribution), e.g. 3$\sigma$ x 5 trials = 2.5$\sigma$,  5$\sigma$ x 1000 trials = 3.4$\sigma$.
- Trying many similar cuts would not increase the pretrial significance much because different trials are correlated with the common fraction of events, but taking this correlation into account is not trivial and considering them independent is more conservative
- **Always think in advance what kind of cuts you want to apply/test on your data set and bear the consequences!**

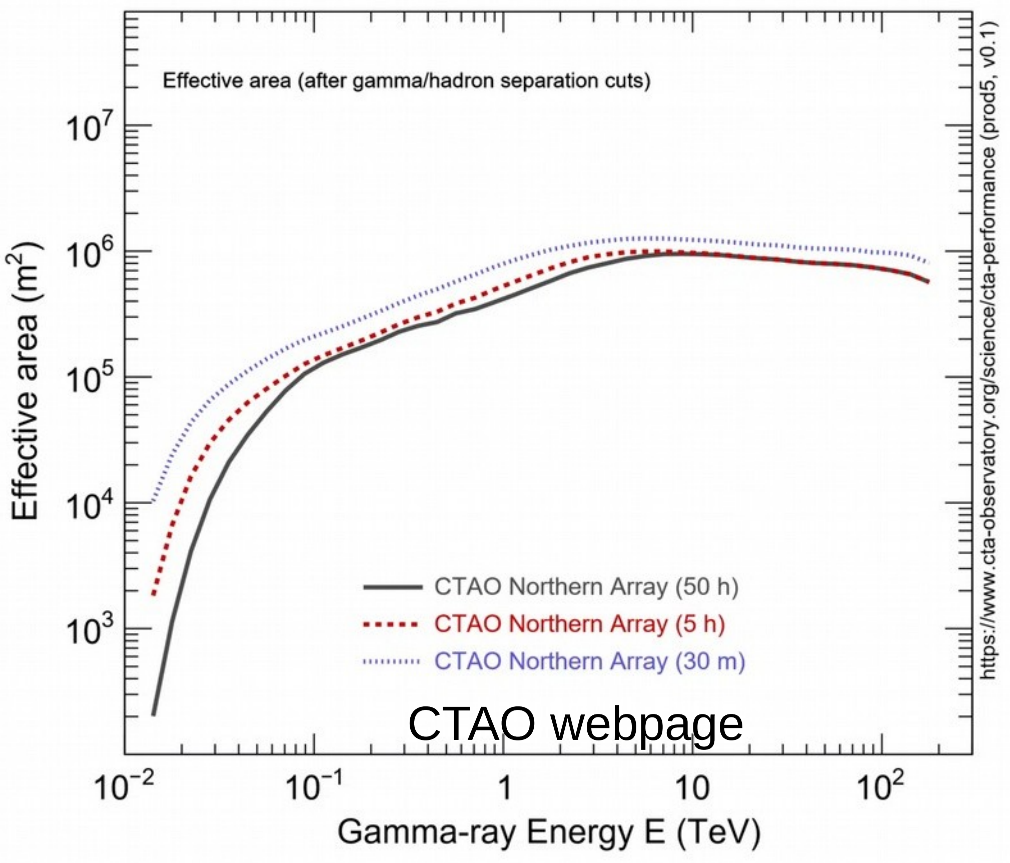
# IRFs of CTA

- There is a separate python module for dealing with IRFs: https://github.com/cta-observatory/pyirf
- The IRFs are provided in tables as a function of different variables:
  - Energy (always) “ENERG”
  - Offset angle from the camera center “THETA”

  and stored in fits format
- **All the IRFs depend on the cuts used to create them – the same cuts have to be applied to the data**

# Collection area

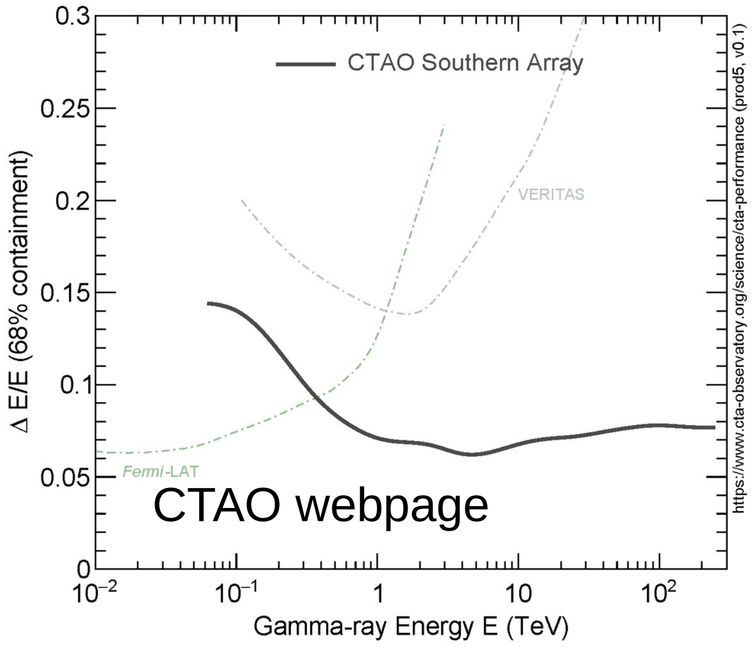
- A measure of how effective is IACT in catching, reconstructing and selecting gamma-rays
- Measured rate is effective time times effective area and flux (folded over energy)



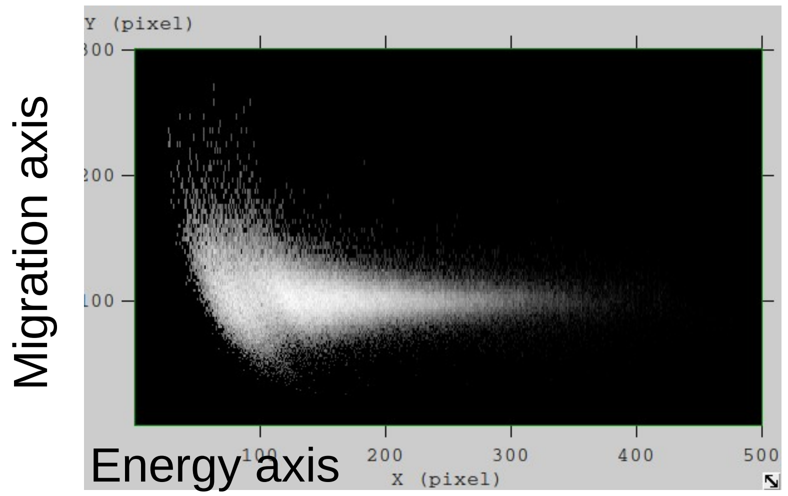
CTAO webpage

# Energy resolution/migration matrix

- Measure of the energy reconstruction bias and accuracy
- Expressed as a matrix with probability density of migration by a given factor for each energy and theta bin



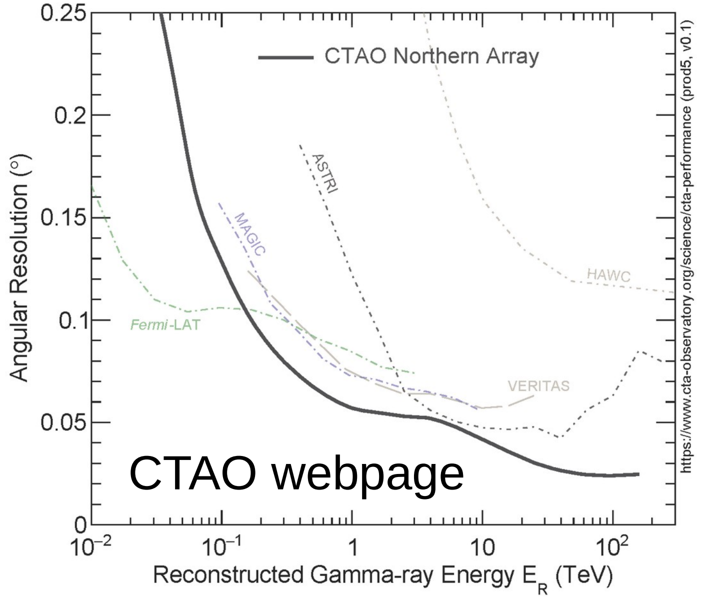
CTAO webpage



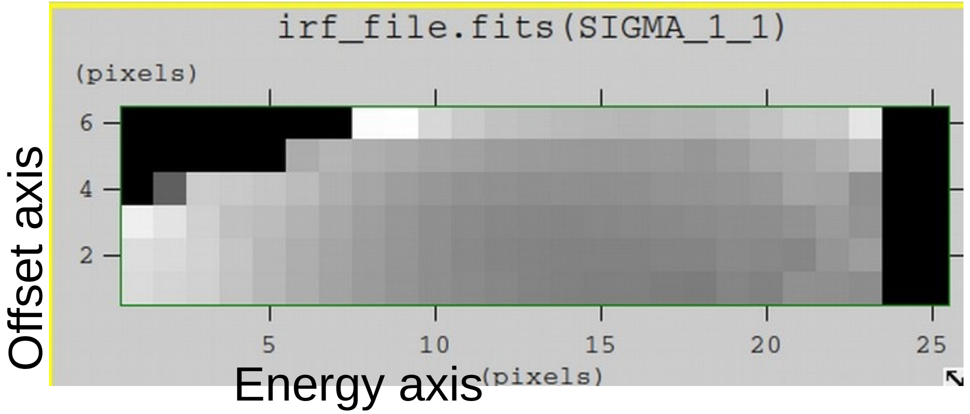
CTAO prod 3b IRFs

# Angular resolution/PSF

- Measures the spread of the reconstructed arrival directions from a point like source
- As a performance parameter is usually expressed as 68% containment
- For extended source analysis the full description of the PSF is needed:
  - With a fit (2D Gaussian, double 2D gaussian, King function, …)
  - “migration” +1D distribution
- For point-like analysis it is not needed – instead angular cut is incorporated in the collection area



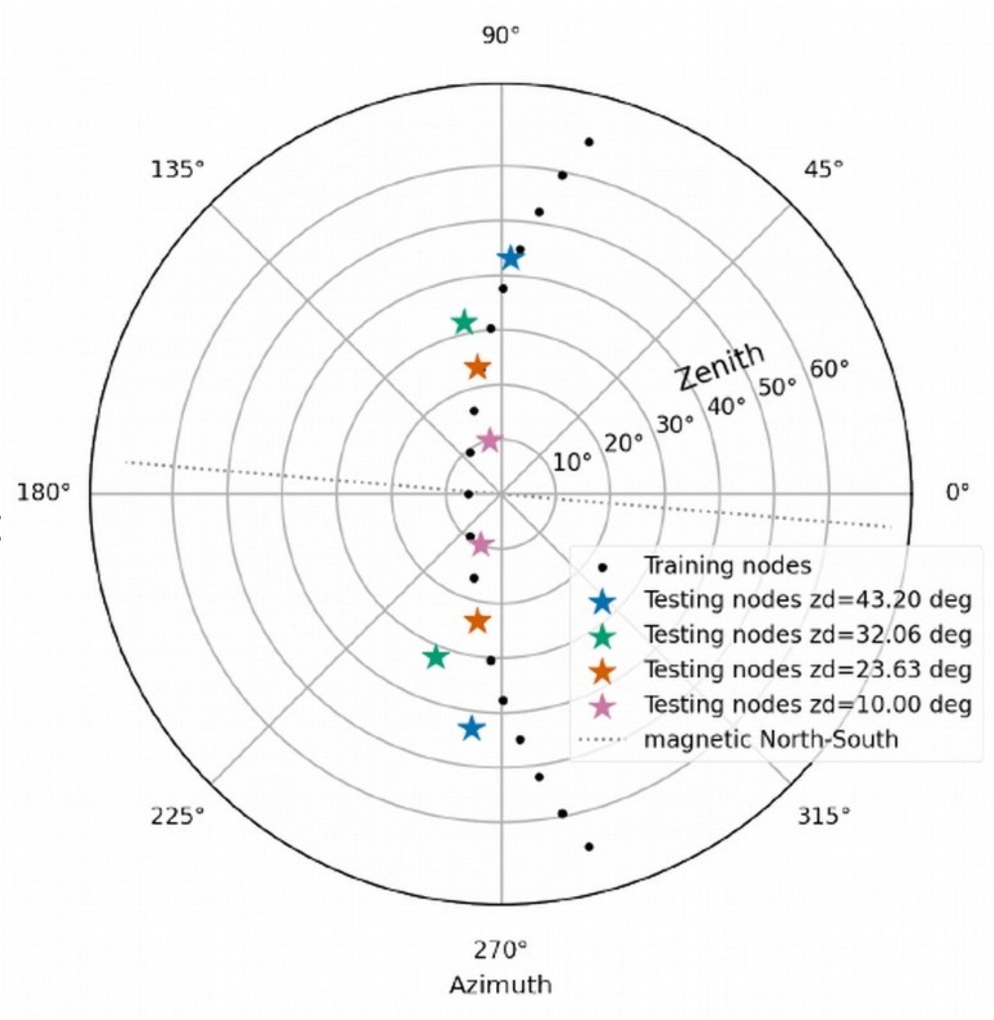
CTAO webpage



CTAO prod 3b IRFs

# Matching IRFs to the data

- IRFs are produced with specific MC samples, normally created for a set of fixed position of zenith/azimuth (nodes)
- The data files normally fall in between the different nodes – interpolation (or extrapolation!) is needed
- The interpolation is not trivial, because it needs to conserve the meaning of the IRFs (e.g. the probability distributions need to sum up to 1), and also should be physics-motivated to avoid large errors
- Pyirf has a number of interpolation tools for IRFs
- In addition along a single file the position also changes, in extreme cases it might not be anymore well described by the same IRFs as the beginning of the run



## DL3 files

Now you have DL3 files with event lists and corresponding IRFs and can apply high level analysis

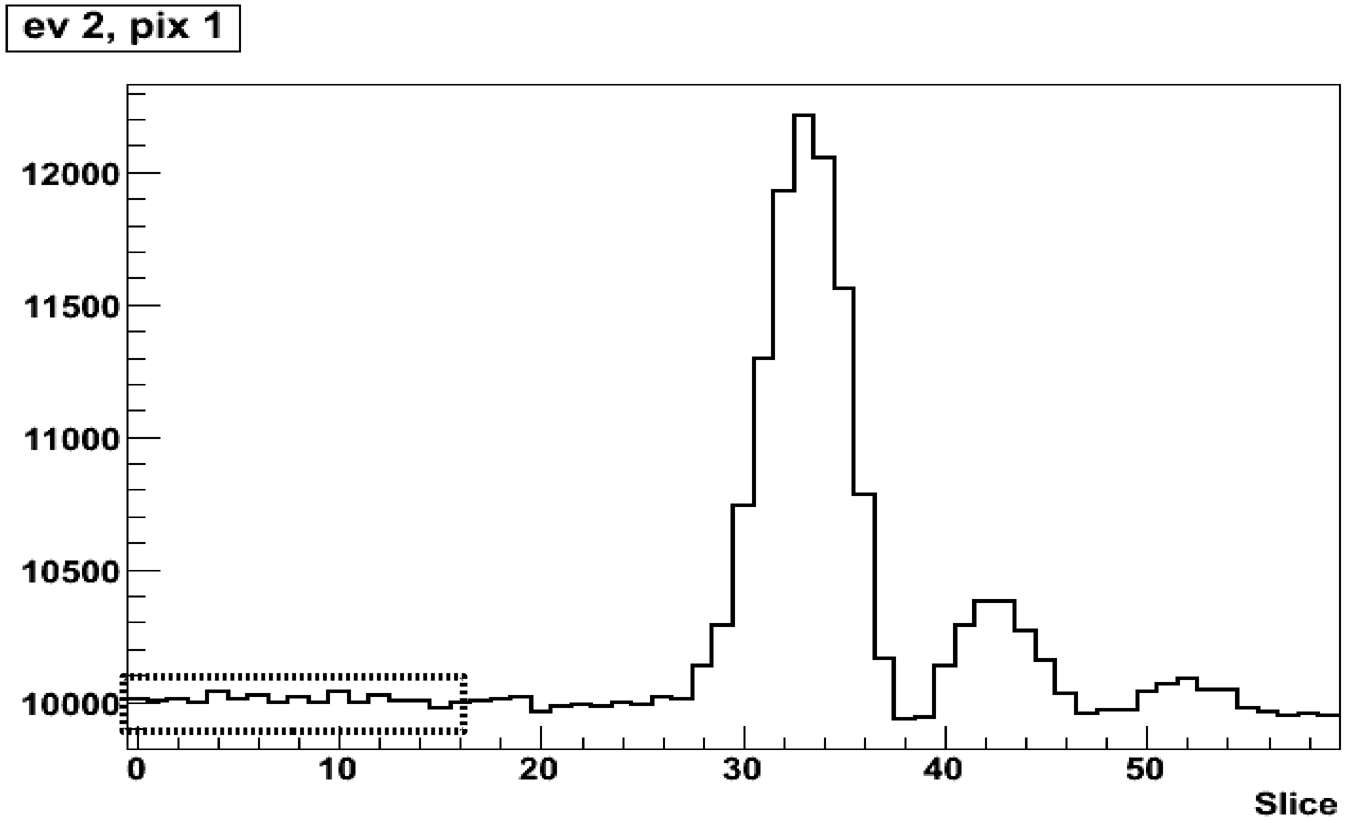
## Summary

- R0(DL0) ==> DL3 is a multistep process, that requires dealing with massive data – as much as possible the output files should be provided to the end users
- In some specific cases you may want to modify the standard analysis chain, that will require starting with the files at lower level than DL3
- Even if you start with automatically-generated DL3 files, you should still understand how the files are produced

## backup

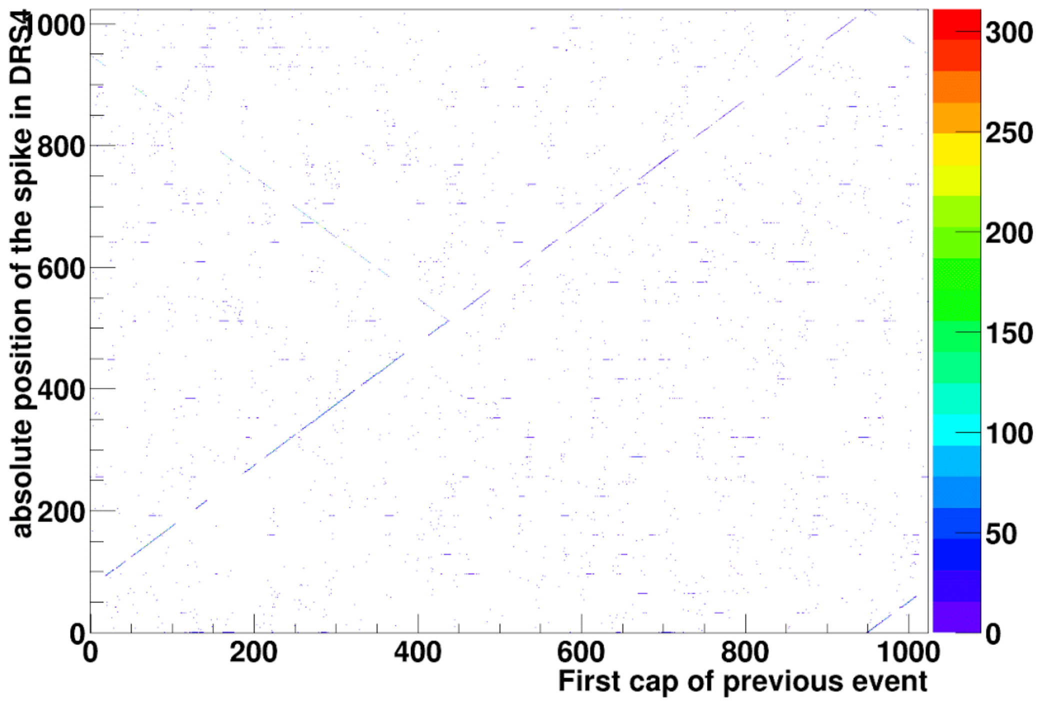
# How to estimate baseline: method 2

- For each event you take a number of first time slices and average it.
- You use different baseline for each event

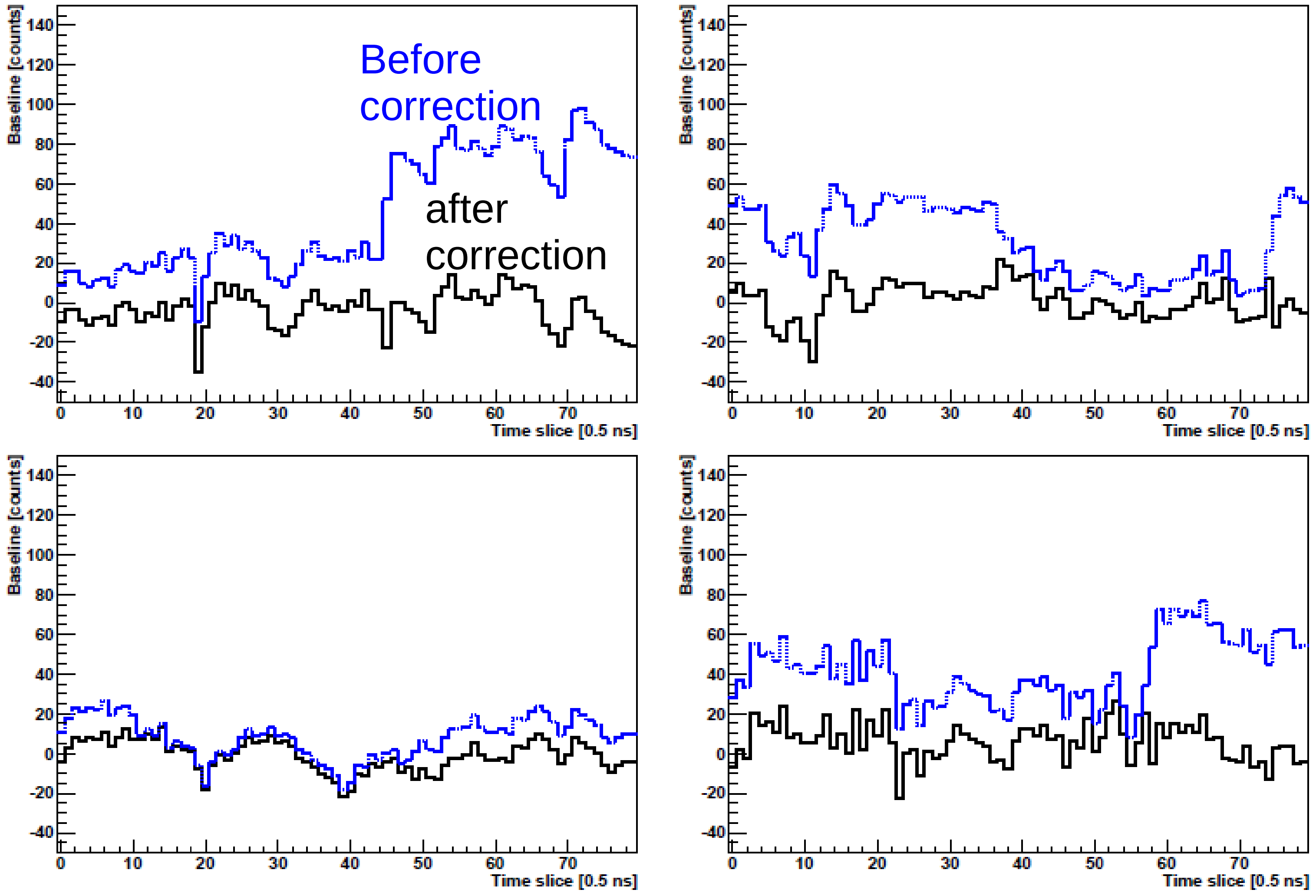


# Spike removal

- Special algorithm to predict where a DRS spike will happen.
- The height of the spike is evaluated from pedestal events and subtracted



# Time baseline correction


Before correction
after correction
Baseline [counts]
Time slice [0.5 ns]