Re-look at CZTI pipeline

Debdutta Paul

Abstract

An important aspect of the Level2 (L2) CZTI pipeline is to clean the L2 CZTI data from events that are definitely not associated with source photons. I take a re-look at all such steps that are currently implemented. I propose a new sequence of cleaning the data: *cztbunchclean* is modified; a new process, which I call 'DPHclean', is proposed after flagging gross noisy pixels from bunchcleaned data, followed by removing flickering pixels: a combination of these as a substitute to *cztpixclean*. While bunchclean is dependent on on-board bunch data and gross noisy pixels are removed as previously, DPHclean implements a new algorithm that identifies all events that cluster in DPHs binned at short timescales. The parameters involved in all the processes are outlined and the optimized values, chosen after thorough examination of data, are also tabulated with due reasoning. Currently the script is written in *Python*, which uses L2 FITS file and bunch file, and outputs cleaned data ready for subsequent analysis, as well as livetime corrected lightcurves (livetimes are calculated at the start from the L2 files and updated after each step).

It is pointed out that whereas the flagging of gross noisy pixels and flickering pixels removes electronically-induced noise, and bunchclean is understood to flag cosmic ray events, bunch data can be used to provide quantitative understanding of the effects of genuine photons on the electronics. This provides a qualitative explanation to the excess of Compton double events present in the data, as compared to expectations from simulations. Whereas some events flagged by DPHclean are produced by electronic effects in the detector, a large number of them are likely induced by high-energy particles, showing remarkable similarity to high energy cosmic-ray induced events seen by the PICsIT detector on board INTEGRAL. An in-depth analysis of such 'DPHstructures' is currently in progress.

1 Introduction

I have used L2 FITS files created by *cztscience2event*, and the corresponding **_bunch.fits** file. The proposed revised pipeline, currently a *Python* script, has been tested on a variety of data, which includes weak GRBs, strong GRBs, and March long background observations. The PHA column is not used, hence spectroscopically bad pixels are not flagged.

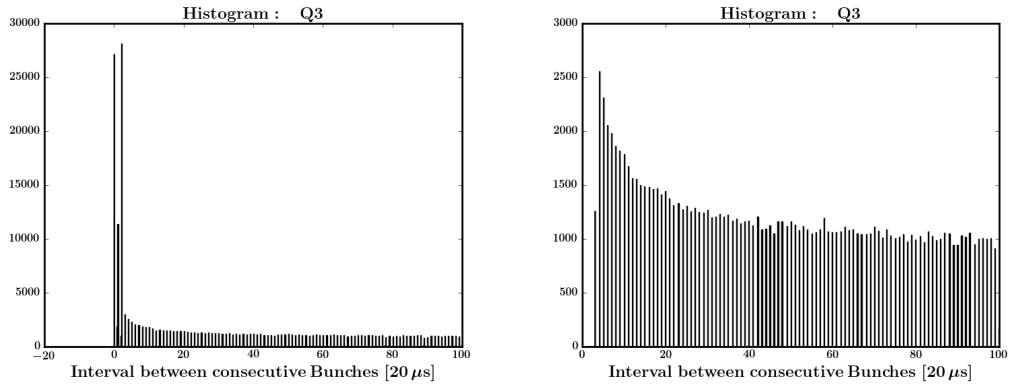

Fig. 1: ΔT is defined as the end of a bunch and the start of a next bunch. *Left*: Raw data, the histogram peaks at small ΔT clearly showing that the definition of bunches is incorrect. The parameter for bunch redefinition, such that the histogram becomes smooth is referred to as t_2. *Right*: With $t_2 = 60\,\mu$s, the histogram indeed becomes smooth. Although the above plots are taken from one orbit of March background data, this observation is true for all datasets examined (the first point in *Right* corresponds to the bunch redefinition timescale, and is an artifact created due to the limitation of the way division is carried out in the binary system: it persists whatever value of t_2 is used).

2 Re-look at *cztbunchclean*

2.1 Redefining bunches

Here I propose that in addition to the definition of bunches in the on-board software, we need to redefine bunches to validate the understanding behind bunches, which is that each bunch is created by one cosmic ray particle generating a series of electronic events within timescales shorter than the instrumental resolution of $20\,\mu$s. If such is the case, the bunches are independent of each other, and the interval between one bunch and the next is expected to decay gradually and not sharply. However, it is clearly seen from bunch data that such is not the case (see *Left* of Fig. 1). This leads one to assume that the electronic effects of a single charged particle lasts for more than the time-resolution of the instrument. Hence it is proposed to redefine bunches, such that if the interval between one bunch and the next is less than a certain threshold t_2, then these two bunches are understood to be created by the same cosmic ray particle and all the data within it are clubbed within a single bunch. Empirically, it is seen that the sharp spike is removed on choosing $t_2 = 60\,\mu$s. For $t_2 = 40\,\mu$s, the spike is still clearly visible, whereas for $t_2 = 80\,\mu$s, the spike is replaced by a dip, clearly showing that it is an overkill: $t_2 = 60\,\mu$s is optimal. It is observed that less than 10% bunches are redefined as 'super-bunches'. A comparison of bunches and super-bunches is currently in progress.

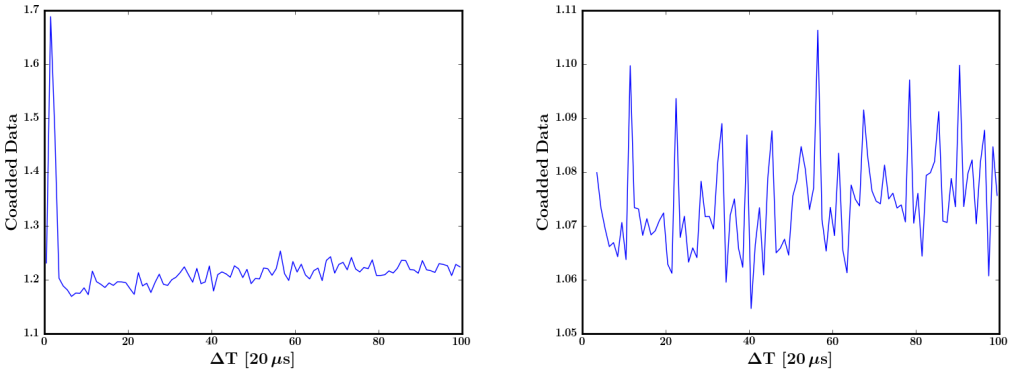

Fig. 2: *Left*: A sharp drop is seen at timescales lesser than 60 μs if co-added lightcurve, corrected for the exposure, is made from data post bunches, even after redefining bunches. This implies that post bunches, significant amount of electronic noise, created by the bunches, is persistent. Hence, data post bunches is removed up to the parameter t_3. *Right*: After flagging **all** data post bunches, with $t_3 = 60\,\mu$s, the co-added lightcurve looks flat (all the way up to 2 ms) as expected, validating the assumption.

2.2 Post-bunch cleaning

As noticed earlier, the electronic effects of cosmic-ray particles persist for some amount of time post-bunch, after initially triggering a series of events in the detector. As shown in Fig. 2, this timescale is again of the same order as that seen while redefining bunches. I parametrize this (similar to *skipT1*, *skipT2*, *skipT3* as defined in the current version) as t_3. I note that if $t_3 = 60\,\mu$s, the co-added lightcurve becomes flat. I have re-examined whether flagging only the Detector Modules with different *bunch_threshold*s also give the same feature, with long stretches of data: the result is same since the optimized value of t_3 is so small that most of the data successive to the bunches are mostly in the same modules. Instead of three parameters for post-bunch cleaning, only one is sufficient. The fact that t_2 and t_3 are similar, leads one to guess that the physical mechanism behind both the effects are same.

2.3 Using bunches to quantify electronic noise created by source photons

A preliminary examination of bunch lightcurves is carried out for a few datasets. Sharp spikes lasting for few seconds are seen in such lightcurves, corresponding to possible increase of cosmic ray induced events, which appear randomly in different datasets and Quadrants. Moreover, they are almost always due to bunches with total number of events, called bunch-length, equal to 3 (and not higher). Sometimes bunches of greater lengths also show increase from the continuum level, but they are not as sharp, moreover they appear uncorrelated to bunches of length 3 during such increase. However, longer timescale bunch-

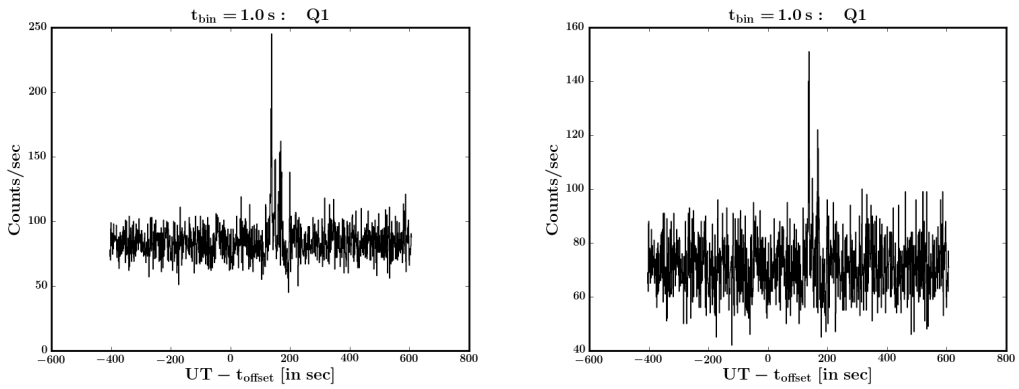

Fig. 3: Bunch lightcurves for bright GRB160821A, with the time axis offset to the known GRB trigger time (note that the trigger time in all Quadrants of CZT data as well as Veto for this GRB is offset by $\sim$ 150 s from the value reported by IceCube). Such an enhancement is not expected if all bunches are due to cosmic rays. *Left*: All bunches *Right*: Bunches with total number of events greater than 3 also show enhancement. Increasing this threshold does not suppress this effect, implying that GRB photons trigger electronic events mimicking as short as well as bunches.

spikes, lasting for $\gtrsim$ 10 s, and they always appear at the same times for bunches of different lengths. However, these are extremely rare ($\sim$ once in 10 orbits), and seem to affect Q3 the most, followed by Q2, although this statement is subject to low-count statistics. If it is true however, it can be understood to be caused by the fact that these Quadrants are in the open side of the satellite and is hence prone to high energetic charged particles: Q3 is open from two sides as compared to one for Q2, wheres Q0 and Q1 are closed by high-Z absorbers from all sides.

If bunches are all indeed created by cosmic ray photons, then they should not show any enhancement during GRBs. Although this is seen to be true in general, Fig. 3 demonstrates that bunches show enhancement during a few GRBs, albeit only few very bright ones. Chance coincidence of double events with single events cannot explain this enhancement, falling short by one order of magnitude. Hence, the identification of all bunches with cosmic rays is questionable. Moreover, the enhancement seems to correlate with the GRB flux, being $\sim$ 80 for GRB160623A, GRB160802A and $\sim$ 150 for GRB160821A, which is brighter than the former two by roughly the same factor. Moreover, any attempt to segregate this effect into bunches of different lengths fails: for GRB160821A, the enhancement is clearly seen for bunch lengths at least up to 6.

I make lightcurves of raw data (without bunchclean) and bunches for any dataset, binned at the same timescale and with the same boundaries, and plot them against each other, shown in Fig. 4. I see that there is a clear correlation, for data away from the GRB durations (neglecting the bunch spikes). However, the correlation vanishes on doing bunchclean. This is easily understood from the fact that the fraction of the total events in the raw data are bunches. I note that the average bunch-length is 6 (computed from on-board bunch data and available in **_bunch.fits** file), implying that the average event-rate

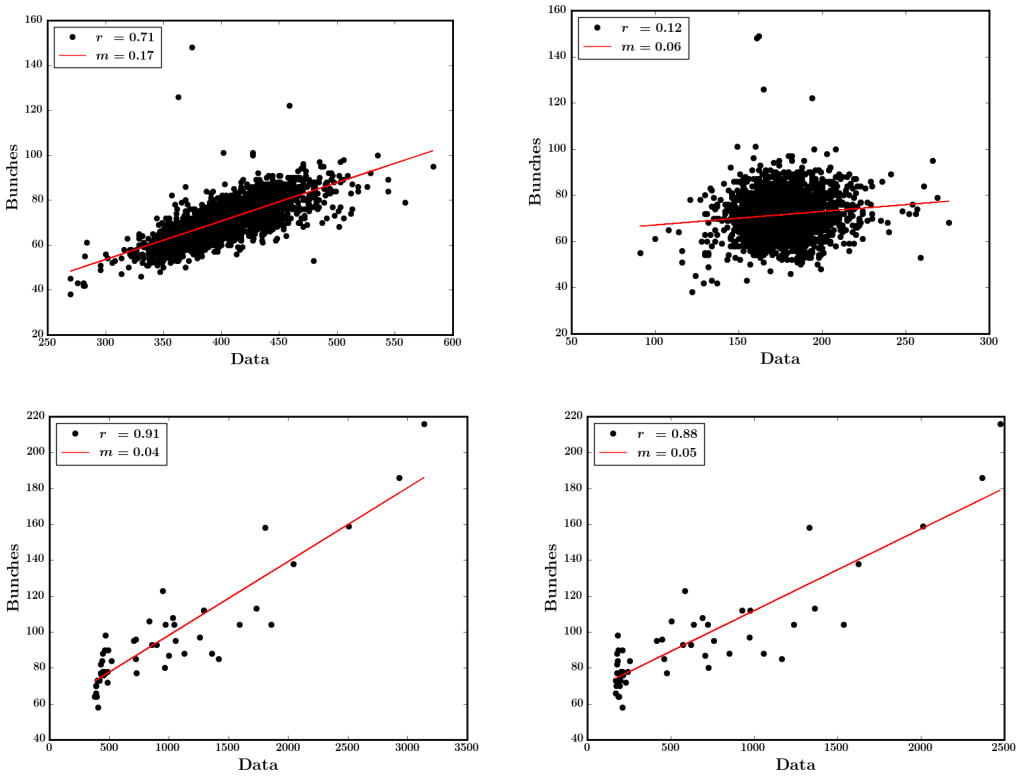

Fig. 4: *Top*: Bunch-rate versus event-rate (corrected for livetime) away from the duration of the very bright GRB160821A. r is the Pearson correlation coefficient, m is the slope of the fitted straight line. *Top-Left*: In the raw data including bunches, i.e. before bunchclean. *Top-Right*: After bunchclean: the correlation is gone, and the scale in the x-axis is reduced by half. The slope is reduced by a factor of ~ 3. *Bottom*: The bunch-rate versus total event-rate in the raw data during the duration of the bright GRB160802A. *Bottom-Left*: Before bunchclean. *Bottom-Right*: After bunchclean: the correlation is still present. It is primarily driven by the (albeit small number of) bins points corresponding to the GRB excess. The slope obtained is comparable for all the three bright GRBs examined, before as well as after bunchclean. This clearly proves that there is a driving mechanism of the bunch excess by the GRB excess, independent of datasets used or the duration or flux of the GRBs. The similarity in the slope with that of the continuum data post cleaning (*Top-Right*) is also indicative of the universality of this driving mechanism, implying that *all* source photons, including sky background, create such an electronic effect. Not all bunches can hence be thought to cosmic ray triggered. However, this gives an overall scaling in the number of bunches as well as double events. All bunches, whether induced by cosmic rays or this electronic mechanism, need to be removed anyway.

in the raw data due to the remnant bunches is ~ 400 (average bunch-rate is 70, although it varies with datasets). The event-rate in the L2 data is also of the same order (~ 400), and this is consistent with the fact that the total event-rate before on-board bunch-cleaning is roughly twice the average number of events before on-board bunchclean. This can be further illustrated from the slope of the best-fit straight line (~ 0.16) as follows,

$$\begin{aligned}\text{slope} = \frac{\text{avg bunch rate}}{\text{avg data rate}} &\implies \text{avg bunch rate} = 0.16 \times \text{avg data rate} \\ \therefore \text{avg bunch length} \times \text{avg bunch rate} &= 6 \times 0.16 \times \text{avg data rate} \\ &\implies \text{avg event rate due to bunches} = \text{avg data rate},\end{aligned}$$

which is observed.

Next, I plot the same around the GRB, obtained from bunches against that obtained from the data, as shown in Fig. 4, *Bottom*. I again see a clear correlation, irrespective of whether bunches are removed or not. However, the slope is reduced by a factor of ~ 3 as compared to the slope from the correlation in the continuum. Moreover, this slope is consistent between all the three GRBs for which bunch-excess is seen. The excess for weaker GRBs is less than 500, also explaining why significant bunch excesses are not seen during such GRBs. This implies that the inherent cause of these excesses are similar for all GRBs, and the effect is linear in the rate of incident photons. The slope can be used to calculate the probability of bunches being created from genuine photons, if it is assumed that *all* incident photons, whether they are from GRBs, background or created by cosmic rays, create additional electronic effects that mimic bunches. This is verified from the fact the slope obtained from the fit during the GRBs are of the same order as that obtained from the fit from the data away from the GRBs obtained after bunchclean, as illustrated in Fig. 4, *Top-Right*. This is also seen to be independent of the dataset used, pointing to an universality of the driving mechanism.

Since the slope is ~ 0.05, the average number of such bunches is $0.05 \times 400 = 20$, i.e. on an average, 20 out of 70 bunches are not induced by cosmic rays. However, the problem of distinction of cosmic-ray induced bunches with bunches created from sky photons, as well as the identification of the source photons from the artificial electronic events within the latter kind, means that it is always safe to flag bunches, whatever the mechanism that causes bunching. This is because the purpose of flagging is to clean the data from events that are known to be created by anything other than source (including sky background) photons.

Assuming that the number of events in the electronically-generated bunches is 4 (since bunch-excess for different bunch-lengths are seen during GRBs, but are less prominent for bunch-lengths very much greater than 3), the number of such events flagged during bunchclean $\sim 20 \times 4 = 80$. After all processes of cleaning, we are left with a total number of ~ 200 events (both single and double), which means out ~ 400 events that are flagged during bunchclean, $\sim 20\%$ are events due to **source photon + associated electronic noise** and the rest are events from genuine cosmic rays. If we extrapolate this idea to double events, we can say that 20% of the remaining double events are generated by electronics, and since these are also likely to be in adjacent pixels, they can mimic what we think are Compton-scattered double-events.

3 Re-look at flagging gross noisy pixels

I examine whether there is any better way of flagging gross noisy pixels than what is currently done, which is to iteratively identify and flag pixels showing greater than 5 sigma deviation in the total DPH from the entire observation. Two modifications are attempted: one is to correct for the effective areas of the pixels (from **CALDB** file), and the other is to make the DPHs made every t_{avg} during the observation, each DPH being scaled down by the total counts, taking care of the variation of the event-rate within the particular observation. This basically takes out the variation of the background counts with the satellite position. Both the modifications are attempted individually as well as together, in the latter case in both possible orders. It is seen that the effective area of spectroscopically bad pixels generally get over-corrected if **CALDB** data is used, and this leads to the identification of these pixels as gross noisy pixels. Flagging spectroscopically bad pixels from the data itself, however, does not lead to any change in the converged solutions, whether the lightcurve weighting is done or not, up to $t_{avg} = 2$ s, below which statistical uncertainties in the lightcurve actually leads to incomplete identification of the gross noisy pixels. I conclude that the solutions obtained by the current method is optimal and also the most efficient. This is true even if there are bright GRBs in the data, because they illuminate the entire Quadrant. Henceforth I parametrize this step with $gross cutoff$, which is the deviation (in units of σ) that is used to identify gross noisy pixels. The optimized value of 5 does a satisfactory job in the sense that the solutions always converge to the same gross noisy pixels, roughly 10 per Quadrant, and being independent of the duration of the observation used (unless it is too small).

It is noted from the lightcurves of gross noisy pixels thus identified exhibit spikes which are entirely uncorrelated with bunches, GRBs, Veto spikes or to each other. Occasionally they create loss of data-acquirement in other pixels because the number of events in these pixels themselves (each time a single pixel) can be greater than that allowed by on-board electronics. This effect is known, is taken care of while writing the GTI columns in the L2 file, and the livetime correction of lightcurves accounts for the data loss. However, even after livetime correction, sometimes lightcurves show a dip at these bins, albeit not statistically significant, being limited by the calculations of the GTI.

4 DPHclean

Strong spikes in lightcurves made from the data after removing gross noisy pixels are observed to correlate with events that cluster in DPHs. The timescale for detecting such clustering is examined, parametrized by t_{look}. Initially, such clustering was observed to be present for 5σ outliers in lightcurves binned at 100 ms, other than bright GRBs. The events that contribute to the clustering are spread over timescales less than 100 ms, and only very rarely do they spill into two consecutive bins. The identification of such clusters by an algorithm which is independent of the total number of events in the DPH is implemented, hence the only constraint on t_{look} is that it should be more than the duration of such events. 100 ms is optimized in this regard, catching such clustering as well as being an order of magnitude smaller than the T_{90} of short GRBs.

It is noted that some of these 'DPHstructures' includes pixels which register counts 4 or higher in 100 ms bins. This means that if 'flickering pixels' are removed based on pixel-wise

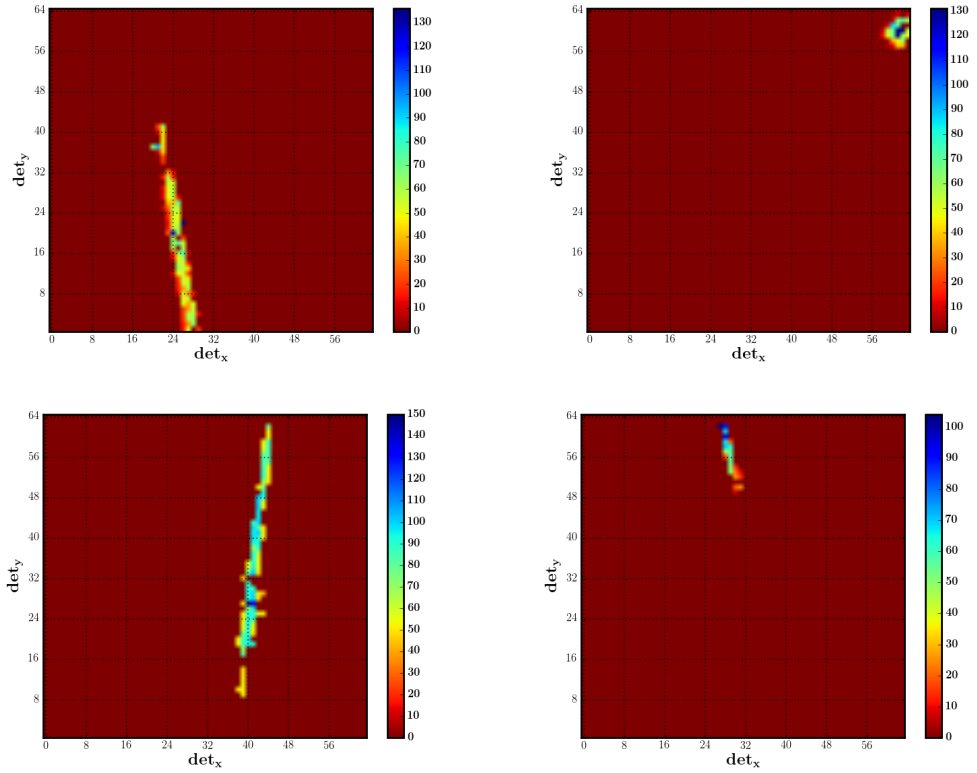

Fig. 5: Detector Delay Histograms: plotted in color are the delay of the particular event from the first event in the cluster, in milliseconds, as a function of the position in the detector plane. These examples last unusually long, covering two consecutive bins. The delay patterns in *Top* are remarkably similar to those seen in PICsIT on INTEGRAL due to phosphorescence-decays from hits from cosmic ray showers. Additionally seen are linear tracks, *Bottom*. In *Bottom-Right*, the delay increases along the track.

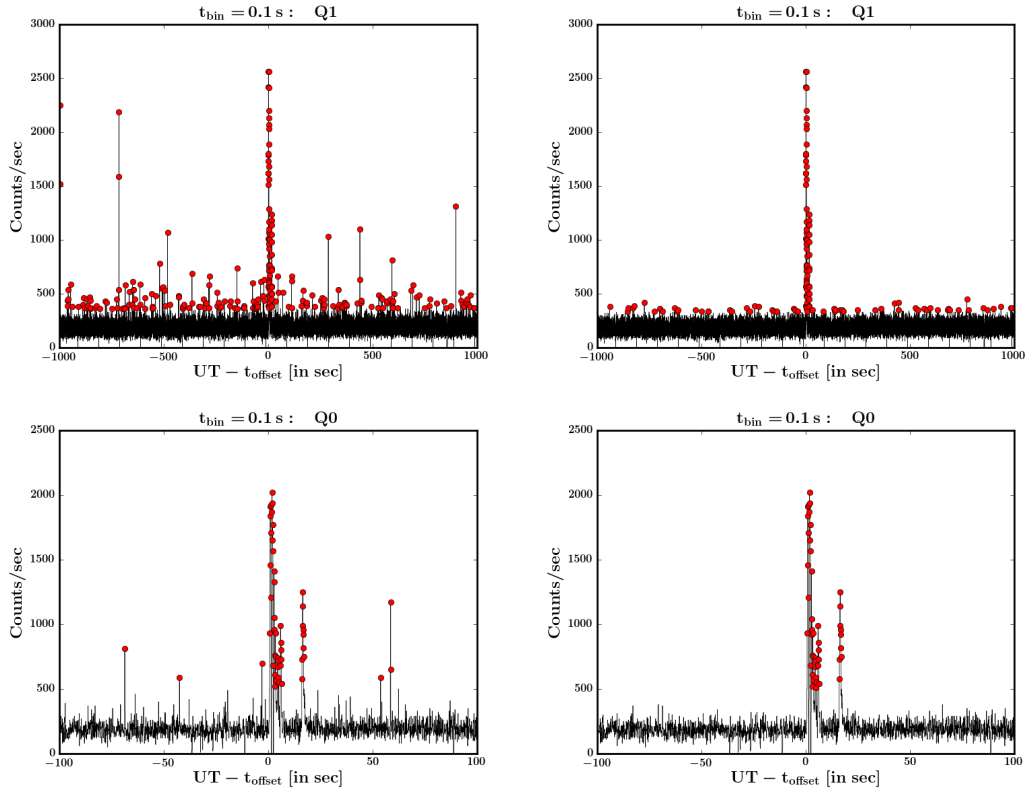

Fig. 6: Lightcurves binned at $t_{look} = 100$ ms before (*Left*) and after (*Right*) DPHclean near the bright GRB160802A, red points showing 2σ outliers: *Top*: longer stretch of Q1 and *Bottom*: zoomed around Q0 (note scale is different). It is noted that noisy spikes are removed but GRB photons are not flagged. The second spike within 20 seconds of the start of the prompt emission is also a part of the GRB, and is seen distinctly in the zoomed lightcurves of all Quadrant, with similar profile, so it is not be mistaken as noise.

lightcurves previous to carrying out this step, some events in these structures get partially removed, leading to non-detection of the clustering in some cases (since the pixels within the cluster registering 2 or 3 counts in such 100 ms bins do not always spatially correlate due to larger separation) and making it difficult to optimize the parameters used for robust cluster-detection. Hence the removal of flickering pixels is postponed after DPHclean.

The algorithm for detecting 'DPHstructures' is detailed in the Appendix. Here it is pointed out that in case a DPH shows clustering, only those events in the DPH that are responsible for the same are removed from the data. Hence, even in the presence of such clustered events during bright GRBs, the algorithm selectively picks out events in the cluster and removes them, the advantage evident. It is noticed that running this algorithm on *all* DPHs made from a given set of data, reduces the noise significantly more than selectively running it on outliers, the number of DPHs exhibiting clustering being $\sim 3\%$. To highlight the case that randomly distributed GRB photons are left unharmed by DPHclean, Fig. 6 compares the lightcurve binned at $t_{look} = 100\,\mathrm{ms}$ before and after implementing this step.

The number of DPHstructures is ~ 250 per orbit. A large fraction of these DPH-clustered events occupy regions only a few pixels wide: most likely, they are locally generated electronic noise or tails of cosmic ray bunches creating long timescale lingering effects. Although it is strictly difficult to distinguish, the DPHstructures that strongly correlate with strong outliers in lightcurves, are most likely caused by genuine ionization in the detectors and not electronic artifacts. This idea is validated by their detector delay histograms, some examples of which are given in Fig. A.4. Moreover, their rarity points to their origin being in very high energy charged particles, further validated by the monotonicity of the delay along linear tracks indicating a real trajectory (Fig. A.4, *Bottom*). Another class of such events show the general pattern of the delay being more in the inside of the cluster compared to the boundaries (Fig. A.4, *Top*). Similar patterns are also seen in PICsIT on board INTEGRAL (Segreto et al., 2003). They are created by cosmic ray showers, exciting phosphorescence states in the detector, the delay pattern tracing the density of the cosmic ray shower in the vicinity of the detector in the logarithmic scale. In the CZT detector, however, the origin is unclear. Preliminary investigation done in this regard has failed to identify any correlation with bunches, although the limited bunch data prevents this conclusion to be statistically significant. Moreover, DPHstructures are also much rarer than bunches. Lightcurves have been made from the events contributing to the DPHstructures, both total and pixel-wise, but they do not offer any clue to the causality of the events. The behaviour of all the Quadrants during a DPHstructure in any Quadrant, and the possibility of correlations with very long bunches, are being investigated. For the purpose of the pipeline flow, however, it is safe to remove all DPHstructure events.

5 Re-look at flickering pixels

Having flagged spatially correlated 'noise', the origins of which are not completely clear, it is next important to remove flickering pixels. For this purpose, pixel-wise lightcurves are made in the timescale $flick\,t_{bin}$. To obtain its optimal value, I vary it from 100 seconds, and decrease successively, each time by one order of magnitude. It is seen that the 'flickering' timescale, most likely a feature of the detector electronics, is 100 ms: for timescales going down to 1 sec, the histogram of events is broad, and the effect of short-lived electronic

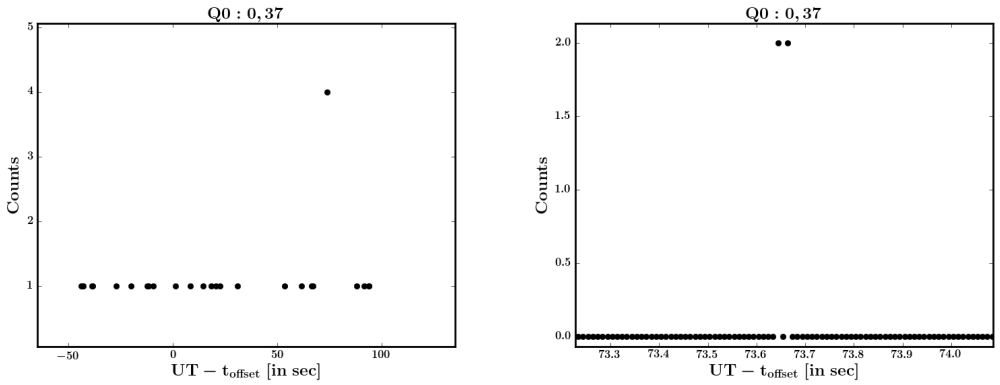

Fig. 7: *Left*: Lightcurve of a flickering pixel. One bin of 100 ms registers 4 counts. It shows normal behaviour ~ 1 s both before and after the bin in which it flickered, meaning it is not worth removing all counts from the pixel and also that the flickering timescale is less than 1 s. *Right*: Lightcurve of the same pixel, binned at 10 ms: the flickering bin at the left divides into two nearby bins separated roughly by 20 ms. This means that we cannot catch the flickering behaviour if the bin-size is ~ 10 ms.

flickering is averaged out. At $flick\, t_{bin} = 100$ ms, however, the tail of the integrated histogram of pixel-wise lightcurves is quite sharply divided from a smooth decay, much like how the histogram of counts in a DPH sharply segregate into normal and gross noisy pixels. At this timescale, some pixels randomly show counts going up to 10 and then exhibit normal behaviour in the next bin. This means that the decay timescale of the electronic effect is smaller than 100 ms, found to be ~ 50 ms maximum. The difference between successive events in a flickering pixel alone becomes comparable to the average difference between two successive events in the *entire* quadrant if such pixels are not considered, ~ 5 ms. This remarkable difference justifies the assumption that these pixels behave abnormally, most likely due to electronic effects, and hence need to be flagged. It is further demonstrated in Fig. 7.

Even for the brightest GRBs seen in the data, which is assumed to be have count-rate of 2000 during its peak, probability to have counts more than 3 counts even once during the peak of the GRB is

$$\left(2 \times \frac{200}{3900}\right)^3 \sim 10^{-3},$$

accounting for a 100% difference in the effective area distribution. For an average event-rate, this is an order of magnitude smaller. This means that it is highly unlikely for one pixel to register 3 counts for more than one instance in the duration of an orbit (probability is 10^{-7}), even accounting for the presence of such bright GRBs. Moreover, two neighbouring pixels exhibiting more than 2 counts at the same instance is also equally less probable. For any pixel to register 4 counts or larger in any 100 ms even once during an entire orbit is again improbable (probability is 10^{-8} for average count-rate and during a very bright GRB $\sim 10^{-5}$). Hence, the following scheme is proposed for identifying flickering pixels:

Parameter	Proposed optimal values
t_2	$60\,\mu$s
t_3	$60\,\mu$s
$gross\,cutoff$	5
t_{look}	100 ms
$threshold$	0.70
$allowable$	3
$flick\,t_{bin}$	100 ms
$flick\,threshold$	2

Tab. 1: Optimal values of chosen parameters in the proposed pipeline.

Firstly, those pixels are chosen which have counts in $flick\,t_{bin}$ (proposed 100 ms) that show more than $flick\,threshold$ counts in that bin. Amongst these, all these registering $> (flick\,threshold + 2)$ counts are deemed flickering. They are flagged for only the bins where they flicker, since the number of such instances hardly go above 2 per pixel during the entire observation, and also since these pixels' behaviour immediately after the flickering bins are found to be usual. For those pixels which exhibit $(flick\,threshold + 1)$ counts, only those exhibiting this only **once** during the whole orbit are *not* defined as 'flickering'; however, if there are any pixels amongst these which exhibit $(flick\,threshold + 1)$ counts in the same instance when an adjacent neighbourhood is also flickering, both pixels are understood to be due to neighbourhood electronics, hence deemed 'flickering' and flagged during the identified times. It is seen that the flickering nature of neighbouring pixels last for $\sim$ 10 ms, justifying the choice of $flick\,t_{bin}$ as well as why flagging data during the flickering intervals is enough.

It is observed that in the remaining pixels, the fraction of occurrence of two nearby pixels with 2 counts each at the same time-bin, is negligible, being consistent with chance-coincidence, preferably occurring near GRBs. In most cases electronically created events like these also excite other neighbouring pixels and hence get removed by DPHclean.

It is observed that the number of flickering pixels is ~ 100 per quadrant, and the total number of such flickering instances ~ 500 for data consisting of one *AstroSat* orbit. The proposed scheme, like DPHclean, leaves GRB photons intact (see Fig. 8), reducing the noise in the data around GRBs, thus enhancing the Signal to Noise Ratio (SNR) of all known GRBs (compared to post DPHclean) significantly more than other alternatives.

6 Conclusions

The proposed optimized values of the parameters in the revised pipeline are listed in the Table.

Although I have provided a qualitative and semi-quantitative understanding of the effects of source photons on the detector electronics, I have not examined the cause of these events. The consistency between the timescales t_2 and $t_3 \sim 60\,\mu$s, as compared to $\sim$ few ms for flickering pixels, points to the mechanism to be same for bunches whether the effect is a redefinition of bunches or flagging data, whereas the mechanism is entirely different for the flickering in some pixels.

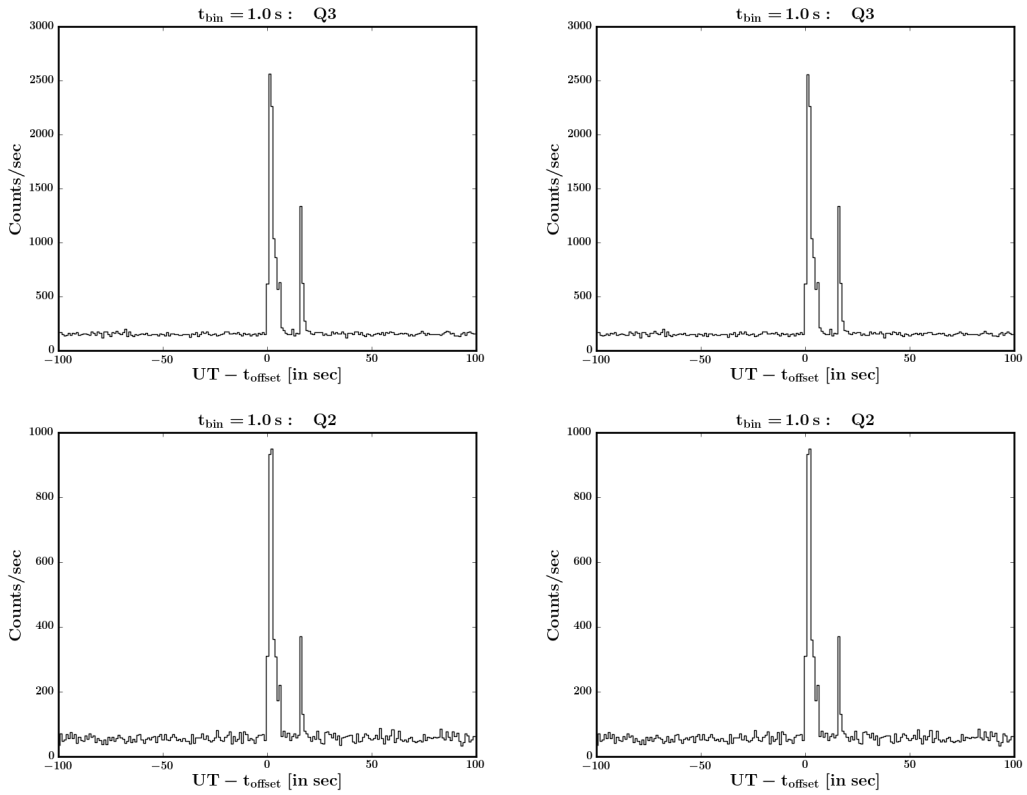

Fig. 8: Zoomed lightcurves of single events in Q3 (*Top*) and double events in Q2 (*Bottom*) : before (*Left*) and after (*Right*) removing flickering instances with the current scheme. The GRB profiles have not changed after removing flickering instances, unlike the pixclean in the current version of the pipeline.

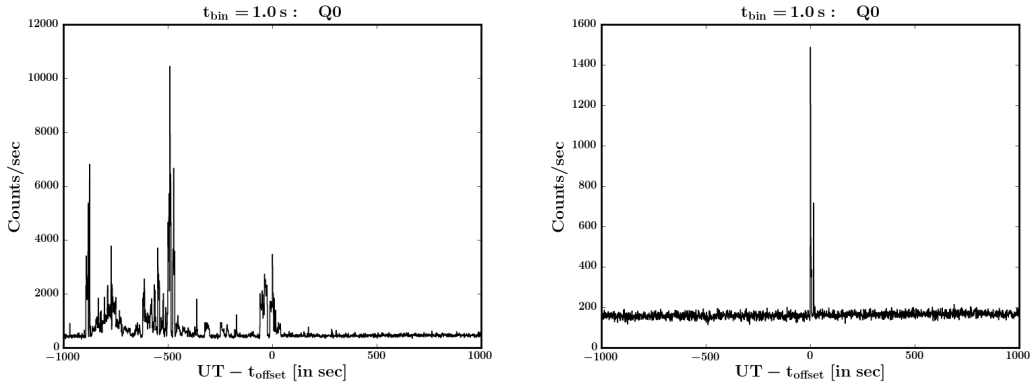

Fig. 9: *Left*: All events at start. *Right*: All single events post cleaning.

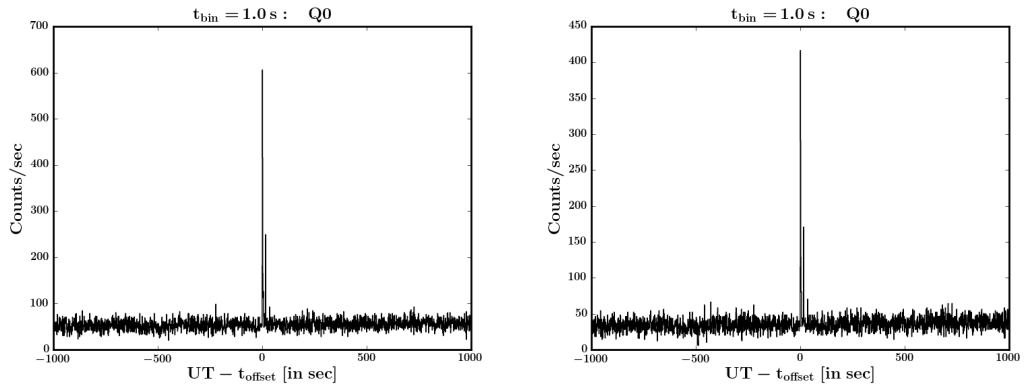

Fig. 10: Double events post cleaning (includes all energies). *Left*: All double events. *Right*: Only those exhibiting simple Compton criterion, i.e. those that are in neighbouring pixels only. The average of such rough Compton events is 30 compared to 50 for all double events, however the enhancement during the GRB appears comparable. The small discrepancy can be due to electronic effects of incident photons, as discussed in Section 2.3.

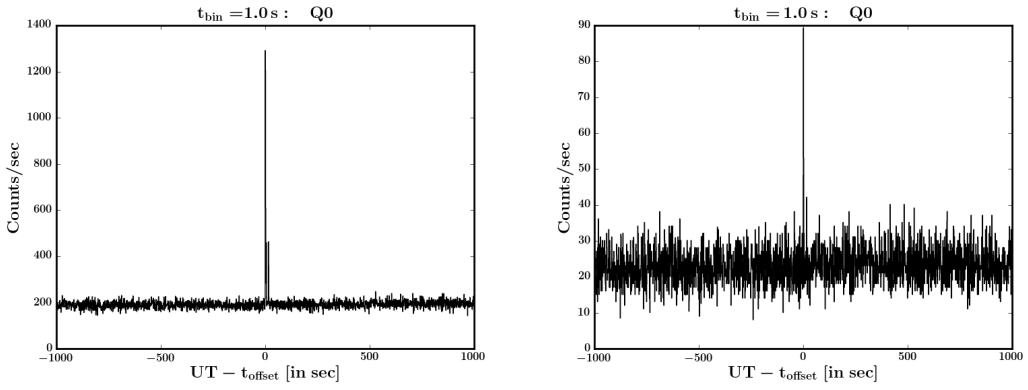

Fig. 11: *Left*: Veto events (includes all energies). *Right*: All Veto-tagged CZT events.

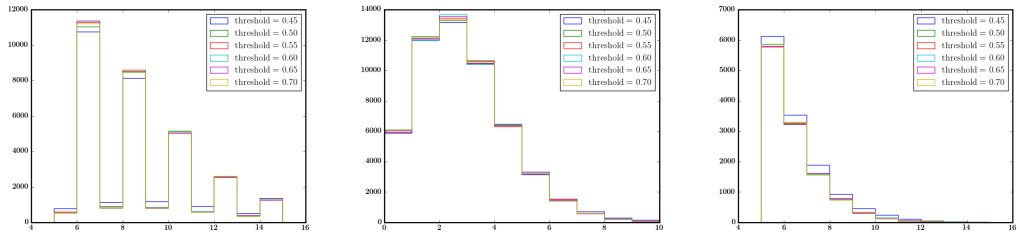

Fig. A.1: Histograms of *Left*: $N_{\rm points}$, *Middle*: $M_{\rm sum}$ and *Right*: $N_{\rm pairs}$ with different values of *threshold*. Since most of the pairs are non-identical, $N_{\rm points}$ is more likely to be even than odd, hence showing the regular dips at odd numbers. It is observed that these parameters are insensitive to the value of *threshold* chosen, hence it is most reasonable to fix it at its most conservative upper limit.

DPHstructures are being more thoroughly examined. The timescale associated with these events cannot be currently explained.

Finally, to make lightcurves, livetime corrections are implemented, calculated initially from L2 GTI and then updated every step. The SNR of GRBs from both single and double events are seen to increase after each step after flagging gross noisy pixels (comparison before flagging gross noisy pixels is meaningless due to the presence of a lot of junk events sometimes also during the GRBs). Sample lightcurves for a stretch of data before and after the bright GRB160802A at the start and after all steps of flagging are shown in Figs 9, 10 and 11.

The Python script is well-documented and can be shared, along with dependencies, for purposes of testing. On the other hand, cleaned data can be also be output in standard files for further processing. It is hoped that the proposed pipeline gets tested and eventually implemented, with modifications wherever necessary.

Appendix: Algorithm for detecting 'DPHstructures'

The aim of such an algorithm is to consider a DPH, and numerically decide whether the DPH shows any clustering or not. It is to output a flag 0 if clustering is detected or 1 if it is not. A requirement of such an algorithm is that it should be independent of the total number of events in the DPH, since it is to be run on DPHs made during average count-rates as well as during GRBs. The algorithm is detailed below:

- Consider only those pixels in the DPH which register non-zero counts. If there are n such pixels, there are ${}^nC_2$ pairs. For each pair, calculate a measure of 'hotness',

$$m_{ij} = \frac{c_i \times c_j}{D_{ij}},$$

where c_i is the counts in the i^{th} pixel and D_{ij} is the distance between the pixels (in units of detx/dety, which is unity). This quantity is large if count in either pixel is large, and/or if the distance between the constituents making the pair is small.

- If
$$m_{ij} > threshold,$$
where *threshold* is a parameter as discussed below, I call it a 'hot pair'. It is to be noted that the maximum allowable value for threshold is
$$threshold_{\max} = \frac{1 \times 1}{\sqrt{2}} \simeq 0.707,$$
which is the case for two diagonally-located neighbouring pixels registering 1 count each (we will miss these hot pairs if *threshold* is larger than this).

- Construct the set of all pixels which contribute to any such hot pair.
- A modification to the choice of hot pairs is made: if a pair is such that it consists of two neighbouring pixels only, each registering one count, and there is no hot pixel in its immediate neighbourhood, then such a pair is not constructed for the following steps. This is to ensure that actual double events are not considered whereas neighbouring pixels with one count each in the neighbourhood of a cluster are retained.
- Calculate the 'gross' parameters:
  1. total number of *non-identical* points contributing to the identified hot pairs: N_{points} ;
  2. the sum of the measures of the hotness for each such hot pairs:
$$M_{\text{sum}} = \sum_{\{\text{all pairs}\}} m_{ij} \, ;$$
  3. the number of hot pairs detected (note that even if one pixel contributes to two/more hot pairs, all these pairs are counted): N_{pairs} .
- Construct a parameter based on these gross parameters as a proxy for the randomness in the DPH. When the value of this proxy exceeds a certain cutoff, parametrized by *allowable*, then the DPH is flagged, i.e. deemed to show clustering, otherwise not.
- If the DPH shows clustering, identify only those in it that contribute to this flagging. Particularly, remove any lingering isolated single or double event, that may have correlated with a pixel registering many counts, randomly, due to its proximity.

To optimize the values of *threshold* and *allowable*, I resort to simulations of random DPHs, with mean count-rate of single and double events as inputs. The mean count-rate is typically 90 for single and 60 for double events, so in a 100 ms timescale, they are 9 and 6 respectively. First the number of single and double events to be chosen for a particular DPH to be simulated are drawn from Poisson distributions with the given means. Then, these many values of detx and dety are drawn from a uniform random distribution of all possible detx and dety values (0 to 63). For double events, one of the neighbouring events is

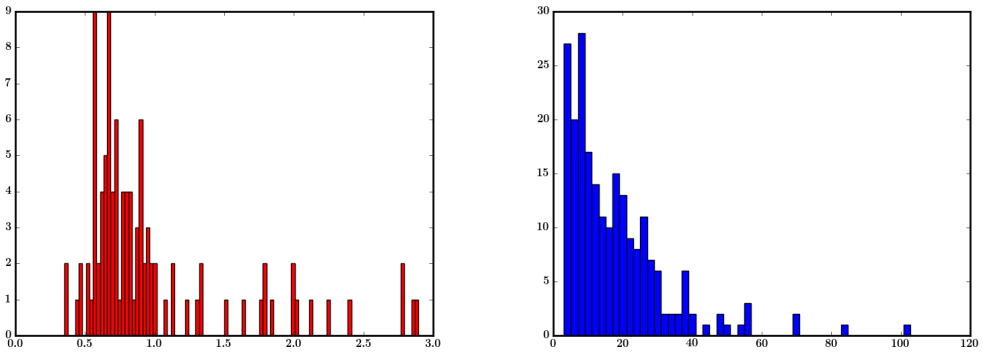

Fig. A.2: *Left*: Histogram of the parameter M_{sum}/N_{points} for the DPHs which are *not* flagged. It rarely goes close to 3, which is the value of *allowable* used for flagging. *Right*: Same for the ones that *are* flagged. Note that although the total number of flagged DPHs in a typical dataset is much smaller than the number of DPHs that are flagged, most of them do not have any hot pairs, hence both M_{sum} and N_{points} are zero. *Left* includes only those within finite values of M_{sum} and N_{points}, explaining why the total number is smaller than in *Right*. The sharp increase at values close to 4 in *Right*, compared to the rarity of those in *Left* below 3, demonstrates that the distinction is real. The reality of this distinction is also verified by manual examination of each DPH for long stretches of data which preferentially include weak as well as bright GRBs, examples of which are given in Fig. A.4.

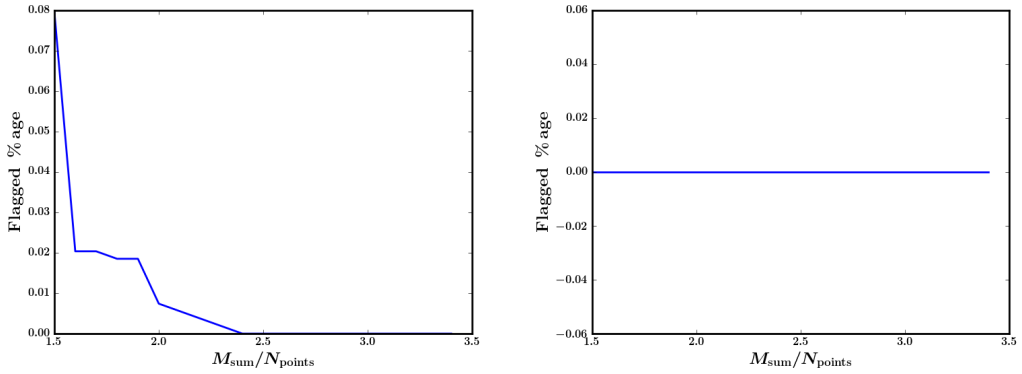

Fig. A.3: Random DPHs are simulated with the event-rate as input. The parameter *allowable* is allowed to vary, and flagging is carried out on the random DPHs based on these variable values. The plots show the resulting number of DPHs flagged as a percentage of the total number of DPHs simulated (5400), as a function of the variable. *Left*: Zoomed for average count-rate of 150 per second. *Right*: Zoomed, for count-rate 1500 per second. The clustering algorithm is seen to be more robust when the count-rate is *more*, implying DPHs during bright GRBs have ~ 0 probability of getting flagged.

first chosen randomly and the other is drawn randomly from the neighbouring coordinates, taking due care of corners and edges. For the case of GRBs, the mean count-rates input into the simulation process are increased, as discussed below (see Fig. A.3).

For each such simulated DPH, the gross parameters $N_{\rm points}$, $M_{\rm sum}$ and $N_{\rm pairs}$ are calculated, and this is done on multiple DPHs (typically 5400 for one full orbit) with different inputs to the parameter *threshold*. The identification of hot pairs based on *threshold* is insensitive to the value of this parameter, as demonstrated in Fig. A.1. Hence it is safe to keep it fixed at its most conservative maximum value, i.e. 0.70, which will detect diagonally-placed neighbouring pixels each registering a count.

Next, I experiment the construction of *allowable* based on the three gross parameters, and flag random DPHs based on the different experimental values of these parameters. It turns out that both $allowable = M_{\rm sum} = 8$ and $allowable = N_{\rm pairs} = 8$ flag less than 1% random DPHs, but this conclusion is seen to break down in the presence of bright GRBs like GRB160802A, since the number of photons in the DPH are greater than the usual by ~ 10 and random pixels get paired and marked as hot pairs. Normalizing any of the parameters by the total number of photons does not help because extremely bright DPHstructures has total number of counts comparable to the total counts in random DPHs during GRBs, simply because the clustering illuminates its neighbourhood very brightly. Hence I define $allowable = M_{\rm sum}/N_{\rm points}$, normalizing for the additional $M_{\rm sum}$ contribution from the pairs that are created due to chance co-incidence of a larger number of random events from GRBs. This simple modification fantastically tells clustered DPHs from random ones. The reason is that, although the total number of counts in a clustered DPH is large, the clustering is spread over a few pixels, and the same pixels register many events; on the

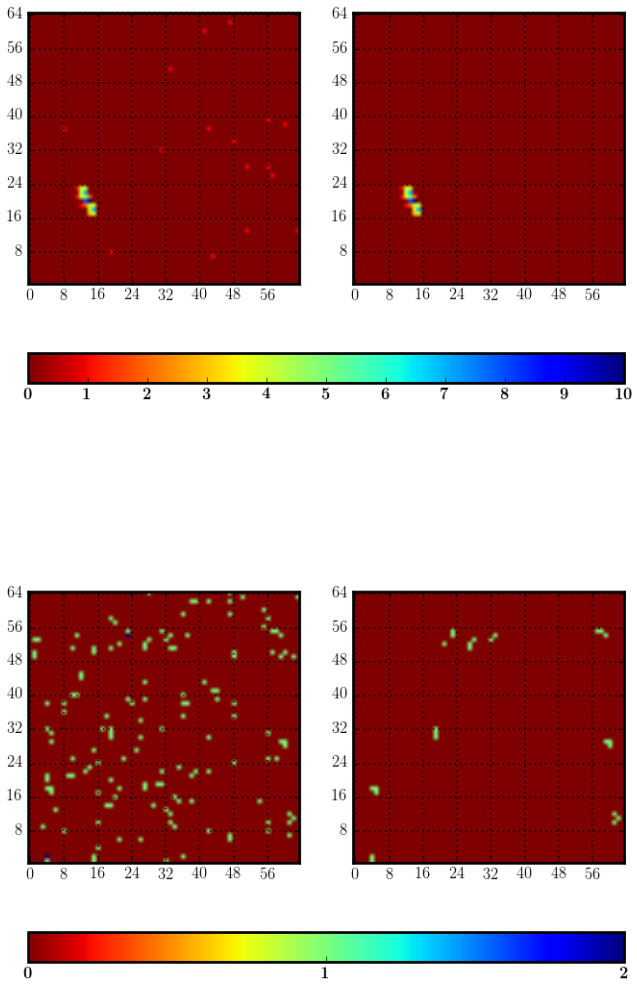

Fig. A.4: *Top*: On the *left* is a DPH which shows clustering (the color-bar is of counts), with the identified clustered events shown in the *right*. *Bottom*: On the *left* is a DPH that does not show clustering. The pairs that are used to test the clustering are explicitly shown in the *right* to demonstrate that the presence of such random pairs are not enough to flag this DPH.

other hand, random DPHs with increased total counts, where the $M_{\rm sum}$ is increased by coincidental pairing of random events, have many such pairs which are themselves randomly distributed over the entire Quadrant. In comparison, *allowable* $= M_{\rm sum}/N_{\rm pairs}$ does not do a better job because the small number of neighbouring pixels in a cluster tend to pair up with most of the other pixels in the cluster.

Random DPHs from GRBs and during average count-rates are examined along with DPHs that show clustering: it is seen that *allowable* $= M_{\rm sum}/N_{\rm points} = 3$ distinctly separates clustered DPHs from random ones, whether they are during a GRB or otherwise. This is verified first visually by looking at a significant number of DPHs by eye, and also demonstrated in Figs A.2 and A.3. Examples of detected DPHstructures and also DPHs with non-detections are shown in Fig. A.4.

References

Segreto, A., Labanti, C., Bazzano, A., et al. 2003, A&A, 411, L215