Here are what I use for this tutorial:

Source: MCXC J0106.8+0103
Observation ID: 0762870601
SAS Version: 19.0.0

To download this tutorial:
git clone https://github.com/poonh/poonh.github.io
cd poonh/poonh.github.io
then you can begin

Products produced using this tutorial is in this main folder. There are several sub folders.

1) src – folders for programs
2) esas_products – products I pre-produced using ESAS commands. If you want to check whether you produce the products correctly, you can compare with this folder.
3) ED_products – products I pre-produced using this tutorial.

I mainly use mos1 as an example in this tutorial.

Filtering for Soft Protons

Run the following commands to get the esas products in poonh.github.io/. If you cannot produce any of them, get them from esas_products/. When you run any esas command, the terminal shows up a lot of stuff which tells you every step that esas command does, and whether a file is successfully produced or not. Any esas command contains many sub-commands. If there is anything wrong in running that sub-command, you can tell the reason from the output in the terminal. Therefore, I suggest putting the output in a log file so that you can easily check if there is a problem.

```
mkdir log
cifbuild  >& log/cifbuild.log
```

ccf.cif is produced.

```
odfingest >& log/odfingest.log
```

2849_0762870601_SCX00000SUM.SAS is produced

```
epchain >& log/epchain.log
epchain withoutoftime=true  >& log/epchainoot.log
emchain >& log/emchain.log
```

epchain and emchain produce an unfiltered event lists, namely mos1S001-ori.fits, mos2S002-ori.fits,pnS003-oot.fits and pnS003-ori.fits. They will be used for further filtering to exclude soft protons.

```
pn-filter >& log/pn-filter.log
mos-filter >& log/mos-filter.log
```

pn-filter and mos-filter use the unfiltered event lists produced from the above step to create the final clean event list – mos1S001-clean.fits, mos2S002-clean.fits, pnS003-clean.fits
 and pnS003-clean-oot.fits. These files will be used for all subsequent analysis. For mos1 and mos2, mos-filter also checks for anomalies of each ccd of them, i.e. ccd 2 to ccd 7, by comparing the lightcuves in high and low energies extracted from the corner, which does not contain any events from the FOV (hence, any anomalies would mean intrinsic). Sometimes they have high counts and they have to be excluded from analysis, but their events are still in mos1S001-clean.fits and mos2S002-clean.fits.

When mos-filter is done, you find a file named command.csh. I renamed it to mosproblemccd.txt in esas_products/. Note if you use different SAS version, a slightly different result would be produced. command.csh is a temporary product and it would be overwritten afterwards, so remember to rename it.

Open it and you see the following:

```
# Limit=1.0 CCD=2 Hardness=4.472 Uncertainty=0.825
# Limit=1.0 CCD=3 Hardness=0.000 Uncertainty=0.000 ****
# Limit=3.4 CCD=4 Hardness=2.625 Uncertainty=0.630 ####
# Limit=3 CCD=5 Hardness=4.957 Uncertainty=0.801
# Limit=1.0 CCD=6 Hardness=0.000 Uncertainty=0.000 ****
# Limit=1.0 CCD=7 Hardness=6.846 Uncertainty=1.437
# Limit=1.0 CCD=2 Hardness=3.295 Uncertainty=0.482
# Limit=1.0 CCD=3 Hardness=4.182 Uncertainty=0.810
# Limit=1.0 CCD=4 Hardness=4.795 Uncertainty=0.844
# Limit=3.5 CCD=5 Hardness=0.784 Uncertainty=0.073 ####
# Limit=1.0 CCD=6 Hardness=5.708 Uncertainty=1.263
# Limit=1.0 CCD=7 Hardness=4.267 Uncertainty=0.707
```

This is the hardness ratio of the lightcuves of mos1 and mos2, if the ratio is too low that means ccd analomies are detected and that ccd has to be excluded in subsequent analysis. In this example, mos1 ccd4 and mos2 ccd5 are anamolous. Mos1 ccd3 and ccd6 are missing.

You also find a file called mos1S001-hist.qpd. It is a diagnostic plot showing you the selection. I will soon show you how to produce it yourself.

qdp mos1S001-hist.qdp
PGPLOT file/type: */xw*

It looks like this:

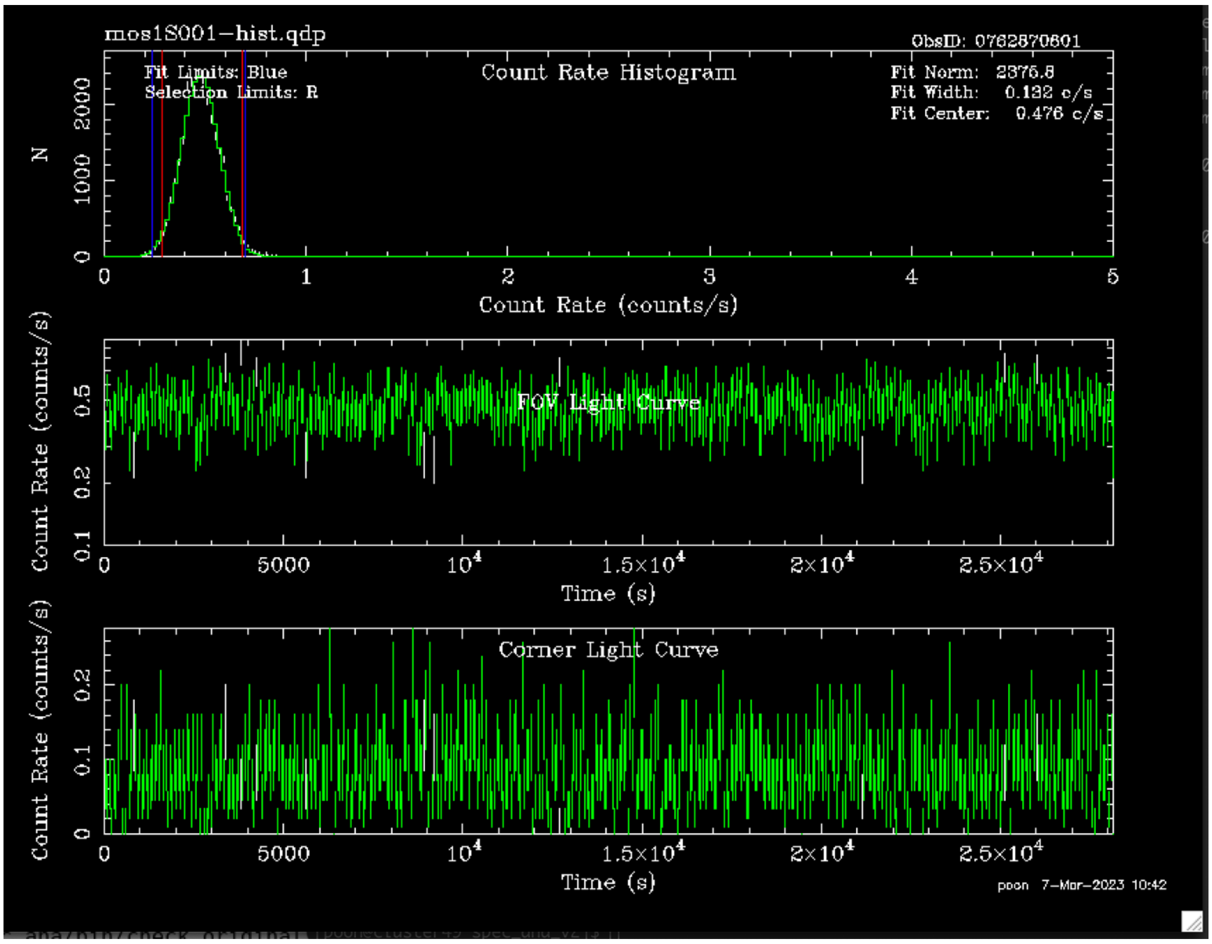


Fig.1 mos1S001-hist.qdp – a plot showing the extracted events in mos1S001-clean.fits

Also, you see mos1S001-gti.txt. It is the file containing the good time interval, i.e. events within the red boundary in the first panel of Fig.1. This file is used to extract the clean event list (mos1S001-clean.fits) from the raw event list (mos1S001-ori.fits) . Open it and take a look

```
>cat mos1S001-git.txt
```

```
552003584.76082599     552004364.76082599     +
552004424.76082599     552006944.76082599     +
552007004.76082599     552007364.76082599     +
552007424.76082599     552007784.76082599     +
552007844.76082599     552009164.76082599     +
552009224.76082599     552012464.76082599     +
552012524.76082599     552012764.76082599     +
552012824.76082599     552016244.76082599     +
552016304.76082599     552024704.76082599     +
552024764.76082599     552028664.76082599     +
552028724.76082599     552029564.76082599     +
552029624.76082599     552031737.76082599     +
```

Of course finally, you have the most important final product – mos1S001-clean.fits, which contain both FOV and corner events.

Here is the physical idea of the above processes.

1) Create lightcurves in a certain energy range, i.e. 2.5-12 keV (used by ESAS or a range of your choice) using mos1S001-ori.fits (the event list which contain all unfiltered events). Energies > 2.5 keV well avoid cosmic X-ray background and solar wind contaminations. After 7 keV, the cluster emission is small, so high count rates are likely due to soft proton flares. The lightcurves are in units of count rates per second.
2) Bin the lightcurves, the default is 60s.
3) Plot a histogram to see the distribution of count rates per second, then make a gaussian fit.
4) Set criteria for filter, e.g. within 3sigma of the best fit. A text file and a fits file containing the good time interval fulfilling the selection criteria is written in mos1S001-gti.txt and mos1S001-gti.fits
5) Use the good time interval file (mos1S001-gti.fits) to write the final product for all analysis - mos1S001-clean.fits

Now you run the following command to do the physical process yourself using python. I put the pre-finished products in ED_products/ already. The default is mos1, if you want to test with mos2, just edit the program by changing the keyword “ccd” at the beginning. Feel free to change other parameters as well, like energy range(elow,ehigh), binszie, fit_limit or sig.
Check for main/tmp_command.csh if any process goes wrong.

```
python src/LC_filter.py
```

If you run successfully, you see the following in the terminal.

```
The LCs for the FOV and corner are produced:  mos1S001-LC-2.5-12.0.fits and
mos1S001-LC-corn-2.5-12.0.fits
best fit parameters (x0 and sigma):  0.48 0.10
```

The filtered clean event list is created: mos1S001-clean_ED.fits

I am now breaking down each step. This input file for this program is mos1S001-ori.fits, the unfiltered event list.

1.Lightcuves in the energy range [2.5 – 12] keV are produced for the FOV and corner, namely mos1S001-LC-2.5-12.0.fits and mos1S001-LC-corn-2.5-12.0.fits, using the following commands.

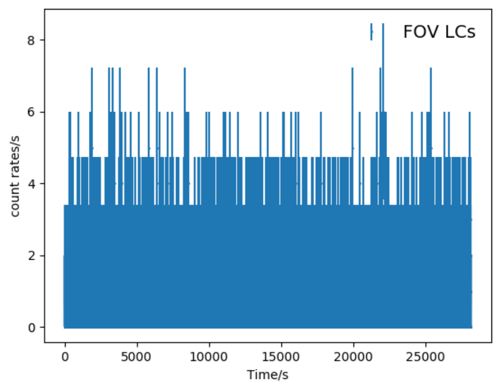
FOV

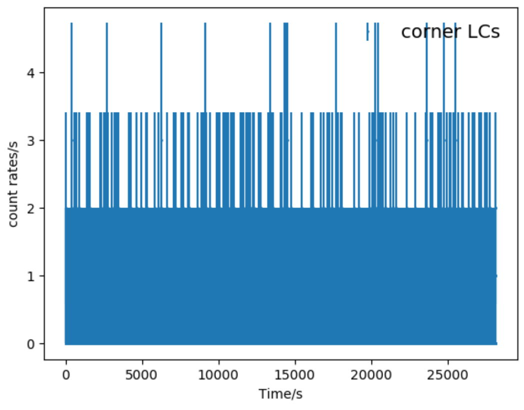
```
evselect table=mos1S001-ori.fits expression='(PATTERN<=12)&&(PI in [2500:12000])&&((FLAG & 0xfb0000) == 0)&&!((DETX,DETY) in BOX(10167,13005,3011,6575,0))' filtertype=expression rateset=mos1S001_LC_2.5-12.0.fits timecolumn=TIME timebinsize=1 maketimecolumn=yes makeratecolumn=yes withrateset=yes
```

corner

```
evselect table=mos1S001-ori.fits withfilteredset=yes expression='(PATTERN<=12)&&(PI in [2500:12000])&&(((FLAG & 0x766a0f63) == 0)||((FLAG & 0x766a0763) == 0))&&!((DETX,DETY) in BOX(13280,-306,6610,6599,0))&&!((DETX,DETY) in BOX(-13169,-105,6599,6599,0))&&((FLAG & 0x766a0f63) == 0)&&!(((DETX,DETY) in CIRCLE(100,-200,17700))||((DETX,DETY) in CIRCLE(834,135,17100))||((DETX,DETY) in CIRCLE(770,-803,17100))||((DETX,DETY) in BOX(-20,-17000,6500,500,0))||((DETX,DETY) in BOX(5880,-20500,7500,1500,10))||((DETX,DETY) in BOX(-5920,-20500,7500,1500,350))||((DETX,DETY) in BOX(-20,-20000,5500,500,0)))' filtertype=expression rateset=mos1S001_LC_corn_2.5-12.0.fits timecolumn=TIME timebinsize=1 maketimecolumn=yes makeratecolumn=yes withrateset=yes
```

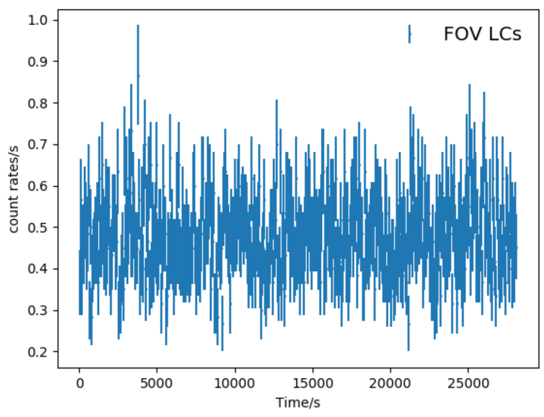
Here are the unbinned lightcurves.

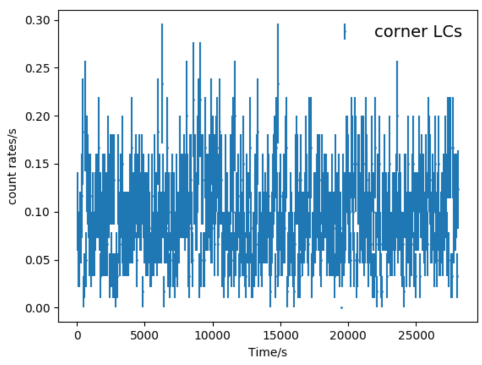




FOV_rawLC.png and corner_rawLC.png

2.Then the lightcurves binned in 60s.





FOV_rawLC.png and corner_rawLC.png

and they are displayed in the second and third panel of mos1S001_2.5_12.0_gti_ED.png. Then the histogram of the FOV lightcuves is fit with a gaussain profile (first panel) using the criteria set in the program to filter out unwanted events.

3 and 4. From the binned FOV LCs, a histogram is plotted and a gaussian fit is made within the fitting range. I set the fitting range to be the [highest histogrambin/1.4,highest histogrambin*1.4]. In *espfilt* (SAS command), it is decided by "rangescale", but I don't know the exact algorithm. As for the selection criteria, I set it to be 2 sigma.

Here is the histogram, FOV LCs and corner LCs with selection indicated from this program .

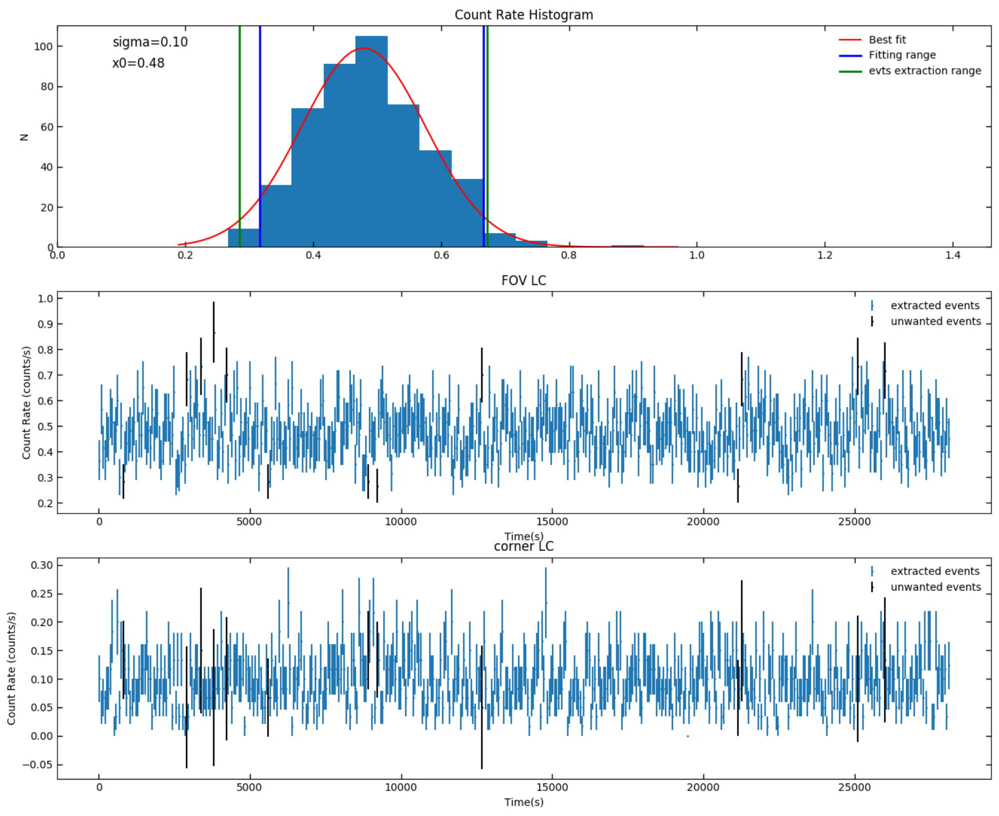


Fig2. mos1S001_2.5_12.0_gti_ED.png. In the first panel, the blue boundary indicates the fitting range. The range is set to cut the high counts at the high end due to soft proton flares (they don't always appear, of course). Within the green boundary are those events wanted. For the second and third panel, those unwanted events are indicated in black

5. From Fig 2. mos1S001_gti_ED.txt is produced. It is the good time interval files containing the filtered event times. If you compare this with the product from ESAS – mos1S001-gti.txt, you find some different. This comes from the different binning of the histogram, different fitting range and different selection significance. To create the final filtered event list, i.e. mos1S001-clean.fits, you need to convert mos1S001_gti_ED.txt to fits format:

ftcreate colname.lis mos1S001_gti.txt mos1S001_gti.fits extname = "STDGTI" clobber=yes

(Ref: https://heasarc.gsfc.nasa.gov/lheasoft/ftools/headas/ftcreate.html) colname.lis contains information of the column name, which has to be created in advance. It is just the following;

```
START D s
STOP D s
```

Now the fits good time interval is called mos1S001_gti_ED.fits

6.Finally, to produce the final product - mosS001_clean_ED.fits, we use the following command, with the good time interval file as a selection criteria:

```
evselect table=mos1S001-ori.fits filteredset=tmp.fit
expression='(PATTERN<=12)&&GTI(mos1S001_gti_ED.fits,TIME)&&(((FLAG &
0x766a0f63)==0)||((FLAG & 0x766a0763) == 0))' filtertype=expression
```

Checking for problem ccd

Remember mos-filter returns you a file which contains analomous ccd information. If is it still not overwritten, it is command.csh, or you check esas_products/mosproblemccd.txt. Now we product this file manually from

```
python bin/problem_ccd_check.py
```

The input file for this is mos1S001-corn.fits, the clean event list from the corner, extracted from mos1S001-clean.fits. The corner events do not have anything from the FOV and reflect intrinsic anormalies of the ccds. If a ccd is normal, the (count rates in the high energy range/count rates in the low energy range) should have a certain ratio. If it is too low, that means the ccd exhibits higher-than-usual count rates and has to be discarded for further analysis. We will see this issue again when studying spectrum.

Now we extract two filtered event list in each corner in the following ranges:

[0.5 – 0.8] keV
[2.5 – 5.0] keV

These ranges can be slightly different in reality.
And 12 lists would be produced, 2 for each ccd. Remember we do ccd 2-7, 1 is not included. The product names are mos1S001_Xccd_0.5_0.8.fits and mos1S001_Xccd_0.5_0.8.fits, X stands for the ccd number.
Here is the command for extracting the filtered event list:

```
evselect table=mos1S001-corn.fits withfilteredset=yes expression='
(PATTERN<=12)&&!((DETX,DETY) in BOX(10167,13005,3011,6575,0))&&(((FLAG
& 0x766a0f63)==0)||((FLAG & 0x766a0763) == 0))&&(CCDNR==7)&&(PI in
[2500.0:5000.0]) ' filtertype=expression filteredset=mos1S001_7ccd_2.5_5.0.fits
```

This example is mos1 ccd7. Just change CCDNR, PI, and filteredset for a different ccd, energy range and filtered event list name.

After you produce all filtered event list, you just count the number of events in each list and do the hardness ratio. Of course, you can do it by hand, for mos1S001_2ccd_0.5_0.8.fits and mos1S001_2ccd_2.5_5.0.fits,

```
>fv mos1S001_2ccd_0.5_0.8.fits
>fv mos1S001_2ccd_2.5_5.0.fits
```

For the extension EVENTS, the cols is the name of each quantity and rows indicate the number of events. It is 36 rows and 161 rows for the low and high energy file, so the hardness ratio is 161/36 = 4.472 for mos1 ccd2. If you check esas_products/mosproblemccd.txt, you find the same value.

If you run the program successfully, the result is in the terminal and written in mos1S001_problemccd.txt.