CAB - Solar Flares

N-Sigma

Part 1) Smoothing a lightcurve:

From attempting the previous task, you might have noticed that the lightcurves you get from XSM aren't exactly smooth and have a lot of local variation. An automated method of identifying flares would be difficult to implement in such a case as a lot of start and end points may occur for a given cutoff due to noise. Hence, we need to smooth out the lightcurve before applying any detection algorithm on it.

> 1.1) Read up about moving averages and convolutions. We will essentially be performing a moving average over the lightcurve to smoothen it
>
> 1.2) Astropy has a function for implementing box-car averaging. Look into documentation/ search online and implement a box car average over the lightcurves you plotted.
>
> 1.3) Overplot the original and smoothed lightcurve and notice the difference.

Part 2) Binning:

As you might have observed from the previous task, XSM gives count rates with a spacing of 1 second. While this is good for finding transients like gamma ray bursts, it is not particularly helpful for flares as they tend to have a longer duration.

> 2.1) To resolve this, implement a method to rebin the histogram with a different timestep. Experiment on different time bins from a range of ~20 to ~200. Pick a few values (say 5) in this range and plot the lightcurve

Part 3) N-sigma:

Building off the bonus task, we can implement a better version of the n sigma algorithm

> 3.1) Calculate the mean and standard deviation of the count rate for the entire binned smoothened lightcurve and plot the mean + std dev as a horizontal line in the lightcurve
>
> 3.2) Is it correctly picking out flares from the lightcurve? How are the start and stop time you get here (you can measure this from matplotlib, at the points where the horizontal lines intersects the lightcurve/flare)

3.3) Mask all the points above this threshold and set them to the value of the threshold, i.e set all points that have a count rate above this cutoff to the cutoff value. Plot this modified lightcurve

3.4) Repeat step one on this new lightcurve thrice. Plot the final cutoff. Did the flare detection improve?

3.5) Tune the above method by replacing mean + std dev by mean + n*std dev. Pick a value of n that identifies flares well from all the lightcurves i.e one value of n that works best for all files together.

(Hint: check out `sigma_clipped_stats` from astropy docs)

Part 4) Background:

4.1)Mask all the flares you got in the last part. The remaining part of the lightcurve is the background! Attempt a curve fit of this background with a straight line (linear fit)

4.2) What is the mean background rate? How does it compare to the average mean value of a flare in a lightcurve?