Surface Brightness

Carter Rhea

May 7, 2019

1 Introduction

X-ray astronomy is known for a phenomenon aptly-monikered *photon-starvation*; this is amplified in certain cases when we are looking back at incredibly distant objects. Photon-starvation poses as serious issue when astronomers wish to determine whether or not a galactic cluster is a Cool-Core Cluster or not. With a strong signal-to-noise, one can "simply" use Weighted Voronoi Tesselations to construct an appropriate bin-map and then use **XSPEC** to develop a temperature map of the region. Unfortunately, for cases in which there is a dissapointingly low signal-to-noise, astronomers must resort to other measures: enter Surface Brightness Concentration Value (SBCV). Using this tool, we can comfortably determine whether or not an object is a CCC even with exceptionally low signal-to-noise. I will not delve into the details regarding the virtues of the SBCV but rather encourage any interested party to read the following papers: [1] [2] [3]. However, it is necessary to define this parameter:

$$SBCV = \frac{F(R < 40kpc)}{F(R < 400kpc)} \tag{1}$$

where F is the X-ray Flux and R represents the radius of the annulus.

According to [3], we are interested in the following regimes:

$$\begin{cases} \texttt{Non Cool Core} SBCV < 0.075 \\ \texttt{Moderate Cool Core} 0.075 < SBCV < 0.155 \\ \texttt{Strong Cool Core} SBCV > 0.155 \end{cases} \tag{2}$$

It is critical to note that this approach assumes a **Poisson Distribution** (i.e. the count frequency is not highly variable with time) and thus **CANNOT** be used with merged observations where the time between observations is significantly different!!

2 Algorithm

The algorithm itself is not particularly complicated, but there are several crucial steps which makes it worth detailing.

Data: Reprocessed Event File

Result: Surface Brightness Concentration Value with Bounds

Step 1: Use Astomety tool (*ASCalc.py*) – located in the *Astrometry* directory – to calculate the angular seperation for 40kpc and 400 kpc;

Step 2: Use this value (in arcseconds) in *ds9* to generate *.reg* files for 40/400kpc;

Step 3: Run *Precursor.py* – located in the *GeneralUse* directory – to generate *.arf* files for 40/400kpc;

Step 4: Calculate Monochromatic Energy AND Surface Brightness Coefficients;

Step 5: Calculate SBCV and bounds using *CSB.py* which is located in the *SurfaceBrightness* directory;

Algorithm 1: Surface Brightness Concentration Value

I have intentionally neglected to include the associated python file in Step 4 because I have two primary methods of calculating Monochromatic Energy and Energy Flux (i.e.

Surface Brightness). For basic use, please employ *SurfBright.py* which is located in the *SurfaceBrightness* directory. However, if you are interested in the details for calculating each step, see *SBCalc_complete.py*. While breaking down the *SBCalc_complete.py* algorithm, I would like to explain how the surface brightness (once all other requisite quantites are calculed) is determined along with error bounds.

2.1 *SBCalc_complete.py*

To begin, we simply need to define the event file, energy range (in eV), the region of interest, the background, and finally a boolean dictating whether or not we will use exposure maps The program truly begins with a calculation of source counts, n, source area, A_s, background counts, m, and background area, A_b, using the `dmextract`[1] command. We then assume a ideal PSF (Point Spread Function), though subsequent versions of the code may include options for more complicated PSFs; thus we set the PSF fraction in source aperature, $\alpha = 1$, and the PSF fraction in the background aperature, $\beta = 0$. Additionally, we quickly grab the exposure time from the event file's header ($TSTOP - TSTART$). At this junction we have two options: exposure map or no exposure map...

Well, we are going to calculate an exposure map either way because we need it to calculate the average effecetive exposures, E_s and E_b.

If we choose to set Exposure == `False`, then we must calculate average photon energies in source and background aperature, eng_a and eng_b, in units of $ergs$ using `dmtcalc`. This option calculates the net energy flux from average photon energies. In choosing to determine net energy flux in this manner, we set the average of photon energy per effective exposure in the source and background, $flux_s$ and $flux_b$, to 1.

However, if we choose to set Exposure == `True`, then we set $eng_s = eng_b = 1$ and calculate $flux_s$ and $flux_b$ using `eff2evt`. According to the documentation on computing net counts, flux, etc.[2], this method is preffered because it wieghts the high energy photons in such a manner so that they do not erroneously dominate the energy flux.

And finally, we supply all of this information (after selecting a confidence contour – usually %90) to `aprates`[3] which calculates the fluxes/counts of interest:

- Net Counts
- Net Count Rates
- Net Photon Flux
- Net Energy Flux
  - using average photon energies
  - using photon energy per effective exposure

[1] http://cxc.harvard.edu/ciao/ahelp/dmextract.html
[2] http://cxc.harvard.edu/ciao/threads/aprates/
[3] http://cxc.harvard.edu/ciao/ahelp/aprates.html

2.2 Calculation of Flux and Confidence Intervals

2.2.1 Flux

The flux calculations[4] comes down to solving the following equations simultaneously:

$$n = fs + cb \tag{3}$$

$$m = gs + rcb \tag{4}$$

for the source and background quantities of interest, s and b, resp. All other variables are selected based on the quantity of interest.

Net Counts

$$f = \alpha$$
$$g = \beta$$
$$c = A_s$$
$$r = \frac{A_b}{A_s}$$

Net Count Rates

$$f = \alpha T_s$$
$$g = \beta T_b$$
$$c = A_s T_s$$
$$r = \frac{A_b T_b}{A_s T_s}$$

Net Photon Flux

$$f = \alpha E_s$$
$$g = \beta E_b$$
$$c = A_s E_s$$
$$r = \frac{A_b E_b}{A_s E_s}$$

Net Energy Flux: option A

$$f = \alpha \frac{E_s}{eng_s}$$
$$g = \beta \frac{E_b}{eng_b}$$
$$c = A_s \frac{E_s}{eng_s}$$
$$r = \frac{A_b E_b eng_s}{A_s E_s eng_b}$$

Net Energy Flux: option B

$$f = \alpha \frac{E_s}{flux_s}$$

$$g = \beta \frac{E_b}{flux_b}$$

$$c = A_s \frac{E_s}{flux_s}$$

$$r = \frac{A_b E_b flux_s}{A_s E_s flux_b}$$

If eng_s or eng_b equal zero, we then set the zero-value equal to the midpoint energy of the given band.

2.2.2 Confidence Intervals

Although not discussed in detail here, we calculate an *upper bound* for a given confidence region using a posterior probability distribution function detailed in [5].

It is, however, crucial to note how the *lower and upper bounds* is calculated for Net Energy Flux. The `srcflux` command calculates the lower and upper bounds in the following manner:

1. Calculate the upper confidence value using the posterior PDF for `net_rate`
2. scale the `energy_flux` value by the upper and lower `net_rate` values to obtain upper and lower limits for `energy_flux`

As evident, the lower and confidence values are not truly calculated, but rather proxy values.

2.3 Dealing with Merged Observations

The primary concern with calculating the surface brightness concentration for merged observations stems from the inherent difficulties in extracting spectrum from merged images. Since we cannot extract spectra from merged observations, we are unable to make ARF files (Also its worth noting that ARF files are high dependent on the individual observation). The ARF file is necessary to calculate the monochromatic energy in low-count observations which is in turn used to competently calculate the effective energy map (exposure map). A natural question to ask is: why don't we simply approximate the monochromatic energy for the energy band of interest by the peak value while avoiding any cuts[4]? Unfortunately, this method neglects the major effect that the monochromatic energy has on the exposure map (i.e. a small error in the monochromatic energy propogates heavily through the exposure map). Therefore, we need to determine a method by which we can calculate the monochromatic energy safetly and in turn use it to calculate the exposure map for a merged image.

[4] http://cxc.harvard.edu/ciao/why/monochromatic_energy.html

2.3.1 High Number of Counts

If we have a high number of counts than the mean energy is a decent-enough approximation of the monochromatic energy. In this case, we can simply use the following command:

```
dmstat "evt2.fits[sky=region(region.reg),energy=energy_band][cols energy]"
```

2.3.2 Low Number of Counts

Thankfully, `ciao` contains a tool that allows us to combine arf files! To merge two observations' arf files, simply run the following command:

```
addresp infile="" arffile="obsid1.arf,obsid2.arf"
phafile="obsid1.pi,obsid2.pi" outfile="" outarf=combined.arf
```

After running this, we can calculate the monochromatic flux using this arf file followed by the creation of an exposure map by running *merged_obs infile=... outroot=... bands=energy_range:MonoEn*. The event file and arf file will naturally have the same root – a fact that will be capitalized on during the flux calculations. If they do not have the same root, then please edit the source code for *flux_calc*...

2.4 Test: A1413

In order to verify that we are getting results consistent with literature, we tested our algorithm again *A1413*. In [1], the c_{sb} is quoted at 0.095; however, it isn't specified what indicator is used.... Thus, we decided to use both the counts per second and ergs/s/cm^2 parameters. Using `obsid:537` and `srcflux` we find the following values: 0.119 and 0.117. These are relatively consistent since [1] neglected to mention the observation ids used... We then used our program and found 0.12 and 0.112. We conclude that our calculations are accurate.

References

[1] J. S. Santos, P. Tozzi, P. Rosati, The evolution of cool-core clusters (2005) (2018) 1–14. arXiv:arXiv:1008.0754v1.

[2] J. S. Santos, P. Rosati, P. Tozzi, B. Hans, S. Ettori, A. Bignamini, Searching for Cool Core Clusters at High redshift (cc) (2018) 1–14. arXiv:arXiv:0802.1445v1.

[3] D. R. Semler, R. Suhada, K. A. Aird, M. L. N. Ashby, M. Bautz, M. Bayliss, G. Bazin, S. Bocquet, B. A. Benson, L. E. Bleem, M. Brodwin, J. E. Carlstrom, C. L. Chang, H. M. Cho, A. Clocchiatti, T. M. Crawford, A. T. Crites, T. D. Haan, S. Desai, M. A. Dobbs, J. P. Dudley, R. J. Foley, E. M. George, M. D. Gladders, A. H. Gonzalez, N. W. Halverson, N. L. Harrington, F. W. High, G. P. Holder, W. L. Holzapfel, S. Hoover, J. D. Hrubes, C. Jones, M. Joy, R. Keisler, L. Knox, A. T. Lee, E. M. Leitch, J. Liu, M. Lueker, A. Mantz, D. P. Marrone, M. Mcdonald, J. J. Mcmahon, J. Mehl, S. S. Meyer, L. Mocanu, J. J. Mohr, T. E. Montroy, S. S. Murray, T. Natoli, S. Padin, T. Plagge, C. Pryke, C. L. Reichardt, A. Rest, J. Ruel, J. E. Ruhl, B. R. Saliwanchik, A. Saro, J. T. Sayre, K. K. Schaffer, L. Shaw, E. Shirokoff, J. Song, H. G. Spieler, B. Stalder, Z. Staniszewski, A. A. Stark, K. Story, C. W. Stubbs, A. V. Engelen, K. Vanderlinde, J. D. Vieira, A. Vikhlinin, R. Williamson, O. Zahn, A. Zenteno, High-redshift cool-core galaxy clusters detected via the sunyaevzel'dovich effect in the south pole telescope survey 1arXiv:arXiv:1208.3368v3.

[4] F. Primini, Revised Specifications for Computing Aperture Photometry Quantities (2007).

[5] V. L. Kashyap, Background marginlaized X-ray source intensity.