Temperature Map Pipeline

Carter Rhea

December 21, 2018

1 Temperature Maps

Before we can get to the temperature map (I promise I'm getting there...), we first have to actually use the data we got from the Weighted Voronoi Tessellation Algorithm; we need the pixels binning information.

1.1 Intermediate Step: Binning the Data

As this subsection's title suggests (look up), we must first go through the process of binning the data (our fits file) based of the recommendation of our WVT algorithm.

Instead of describing this in great detail, I will simply include an algorithm outlining the process and point the reader toward the python file (*Bin_data.py*) for more in depth analysis.

Data: WVT Bins and Fits File
Result: Binned PHA files
initialization $--> $ read in WVT bin information;
for *bin* **do**
  for *pixel in bin* **do**
    Create fits file for pixel;
    Generate PI/PHA file;
  end
  Combine Pixel's PI/PHA files;
end

Algorithm 1: Binning Algorithm

Clearly the implementation is a little more complicated and uses several *CIAO* tools such as *specextract* and *dmextract*. For more information on these wonderful tools, check out the *CIAO* website:

http://cxc.harvard.edu/ciao/

1.2 Generating Temperature Maps

We can now (FINALLY) use our combined pixels to generate a temperature map assuming you have already created a binning of some sort – I would suggest my Weighted Voronoi Tesselations algorithm[1]. We will be employing *XSPEC*[2] for our modeling needs.

There are two main types of models:

- Addative: Source of emission i.e. producer of photons
- Multiplicative: Modifies spectrum i.e. acts on photons

Let's examine the major componenets of our model...

[1] https://github.com/crhea93/WVT
[2] https://heasarc.gsfc.nasa.gov/xanadu/xspec/

1.2.1 zpow

`zpow` is a simple photon power law taking into account redshift. This is just classic blackbody goodness...

$$A(E) = K[E(1+z)]^{-\alpha} \tag{1}$$

where α is the dimensionless photon index of the power law, z is our objects redshift, and K is a normalization factor at $1keV$ in units of $photons/keV/cm^2$. Photon indices of AGN are typically between $1.5 - 2.5$; this value needs to be determined from the literature.

Its worth noting that `zpow` is an addative model.

1.2.2 zphabs

`zphabs` is a model for hydrogen column density. Hydrogen column density is the "number of units of matter observed along a line of sight that has an area of observation"[3]. Basically, it takes into account all the stuff in between us and the object we are looking at (though it could be missing things such as a warm absorber like a galactic cluster which would require another model component – wabs in this instance). zphabs has a very simple equation:

$$M(E) = e^{-n_H * \sigma(E(1+z))} \tag{2}$$

where n_H is the equivalent column density in 10^{22} $atoms$ cm^{-2}, $\sigma(E)$ is the electron cross section – not Thompson – where z is the redshift.

All we specify are n_H and z.

`zphabs` is a multiplicative model

1.2.3 apec

`apec` is a wonderful model to handle the emission spectrum of a collisionally-drive optically-thin plasma (think AGN). Hence we use it to constrain the temperature of the object. For real details please check out the following website:

http://www.atomdb.org/physics.php

The most important thing to know is that it does in fact model thermal bremmstrahlung. We need only specify T (KeV), as a guess, and z which is our redshift.

`Apec` is a multiplicative model

1.2.4 χ^2 statistic

The χ^2 is defined as the following:

$$\chi^2 = \sum_{i=1}^{N_{el}} \left(\frac{x_i - \mu_i}{\sigma_i} \right)^2 \tag{3}$$

[3] https://en.wikiversity.org/wiki/Column_densities

Where x_i is the data value at point i, σ_i is the corresponding error, and μ_i is the model value.

By calculating the χ^2 we can get a decent guess on how well our fit is. Optimally, our χ^2 will equal the number of **degrees of freedom**. We call this ratio the **Reduced Chi-Squared Statistic**. Hence we can very easily calculate the reduced chi-square if we know the number of degrees of freedom for a model:

$$\chi^2_{red} = \frac{\chi^2}{N_{dof}} \tag{4}$$

So an optimal fit using a reduced χ^2-statistic means $\chi^2_{red} = 1$. Hence for our algorithm, we will try to hit a $\chi^2_{red} = 1$ ... or as close as we can get ...

1.2.5 C-statistic

For low counts (in my opinion most X-ray observations are relatively starved in the photon department), we should instead use the *Cash-statistic*, *c-stat*, beacuse at low counts the variance approaches 0 and thus the χ^2-*statistic* fails. We define the *c-stat* in the following manner:

$$c = 2\sum_{i=1}^{n} s_i - N_i + N_i ln(N_i/s_i) \tag{5}$$

where s is the number of counts according to the model, N_i is the number of observed counts, and n is the number of bins. In addition, we are able to calculate a "goodness of fit" for this statistic, C_R:

$$C_R = \frac{C}{n-p} \tag{6}$$

where p represents the number of free parameters.

In `XSPEC` we simply need to set `fit.statistic = 'cstat'`.

2 The Pipeline

Data: Binning Map
Result: Temperature Map and Graphic
Step 1: Create binned spectra using *binned_spectra.py.* ;
Step 2: Run XSPEC Model on binned data by "simply"[a] running *Temperature_Fits.py.* ;
Step 3: Plot the temperature map with *Temperature_Plot.py.*;
Algorithm 2: Temperature Map Pipeline

[a]Like fitting a model is simple..... But seriously this is where a mistake could be made resulting in incorrect Temperature Profiles so play around with the model on a *single* observation to ensure it is a decent fit...

The algorithm itself is fairly succinct because all of the mechanics are intentionally under to hood. For instance, we actually create what I deem as super-pixels in the creation of the spectra for each bin since `specextract` runs faster with less regions. The super-pixel algorithm groups as many pixels together to create a box-region to reduce computational time[4]

2.1 Binning Spectra

2.2 Fitting Spectra

[4]The mechanics for doing this are well commented in the *binned_spectra.py* file.

3 Dealing with Merged Data Set

How do we create temperature maps for merged data sets? As documented on the CIAO site [5], we are not able to create spectra from merged observations. Thus we have to be very careful in the creation of temperature maps for merged data sets.

The basic steps are as follows:

1. Merge Observations (Don't worry! We are just going to use the event file and exposure map).
2. Create a bin map using your favorite binning algorithm; I suggest using Weighted Voronoi Tesselations.
3. For each individual observation do the following:
   (a) Create individual event files for each bin
   (b) Extract spectrum for each bin
4. Run fitting software on each bin by fitting all datasets simultaneously

In following these steps, we are able to reconstruct a temperauture map for a merged observation while avoiding the inherent drawbacks of merging data.

[5] http://cxc.harvard.edu/ciao/caveats/merged_events.html

4 NGC 4636

For an example, I will continue with the cluster `NGC 4636` which I used to demonstrate the efficacy of my Weighted Voronoi Tessellations algorithm except this time we will be using the entire CCD on which the cluster is located and wont be doing any background/exposure subtraction.

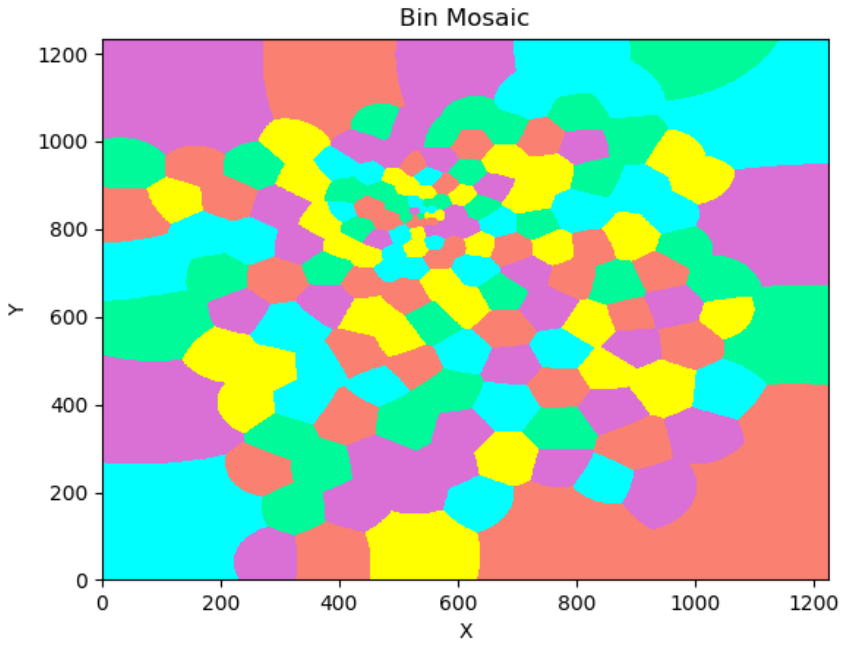


Figure 1: The final WVT map for `NGC 4636` with 177 bins in image units.

The next step involves running the creation of spectra for each bin. I have included the input parameters in case anyone wants to reproduce the results :)

4.1 *binned_spectra.py*

```
filename = '/home/user/Documents/NGC4636/Merged_unbinned/WVT_data.txt'
base_directory = '/home/carterrhea/Documents/NGC4636'
OBSIDS = ['323/repro','324/repro']
source_file = 'center'
background = 'bkg'
output_dir = 'binned/'
```

4.2 *Temperature_Fits.py*

Finally we are able to fit the data! I opted to simply used an absorbed apec model from *Xspec* through *Sherpa*. The model was fit simultaneously to both observation Ids (323 and

324) with:

$$set\_source(1, xsphabs.abs1 * (xsapec.apec1))$$

$$set\_source(2, abs1 * (xsapec.apec2))$$

and the column density nH of the absorber was set to $1.91e^{20}cm^3$ which was taken from the NASA HEASARC NH Tool[6]. Again, I am including the inputs...

```
base_directory = '/home/user/Documents/NGC4636/'
fold_ext = 'repro/binned'
dir = [base_directory+'323/'+fold_ext,base_directory+'324/'+fold_ext]
file_name = 'center'
output_file = 'Temp_bin'
num_files = 177
redshift = 0.003129
n_H = 1.91-2
Temp_guess = 1
energy_min = 0.5
energy_max = 8.0
grouping = 10
statistic = 'chi2gehrels'
plot_dir = base_directory+'FitPlots/'
```

4.3 *Temperature_Plot.py*

And now we get to plot our temperature map!!

```
base_dir = '/home/user/Documents/NGC4636/'
bin_file = 'Merged_unbinned/WVT_data.txt'
temp_file = 'Temp_bin.txt'
output_dir = base_dir
output_name = 'NGC4636_temperature'
color_map = 'jet'
```

In comparing this temperature map with that from Diehl and Statler 2005[7], we can see that the map is well-recreated[8]!

[6] https://heasarc.gsfc.nasa.gov/cgi-bin/Tools/w3nh/w3nh.pl

[7] https://arxiv.org/abs/astro-ph/0512074

[8] Small caveat: At the time of writing this bins 61 and 132 were artificially set to 1 keV because their calculated temperatures were horrendously constrained!!

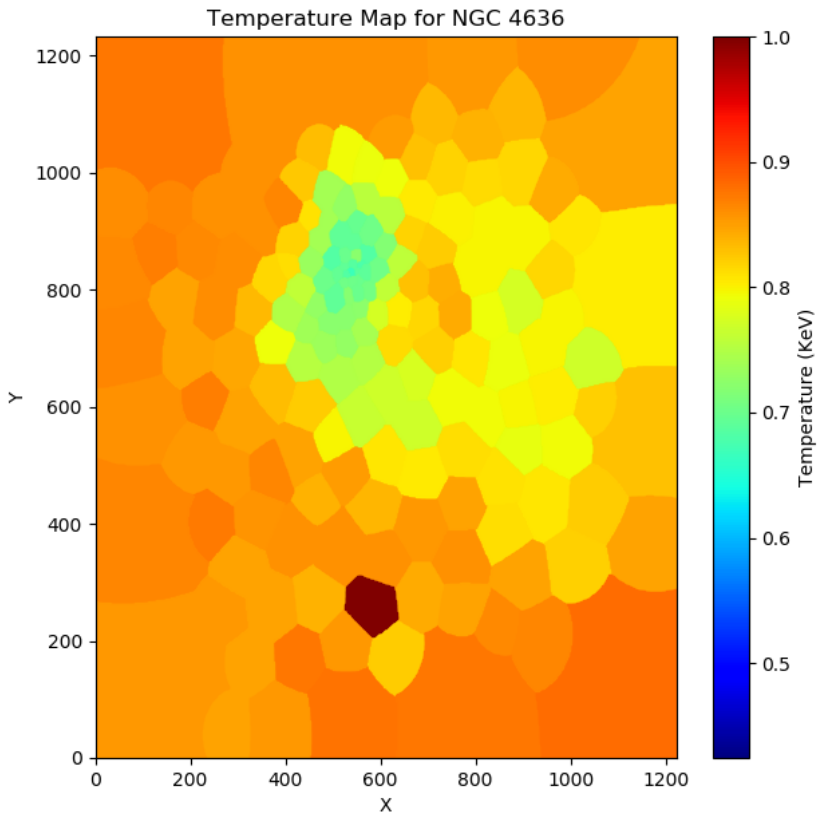
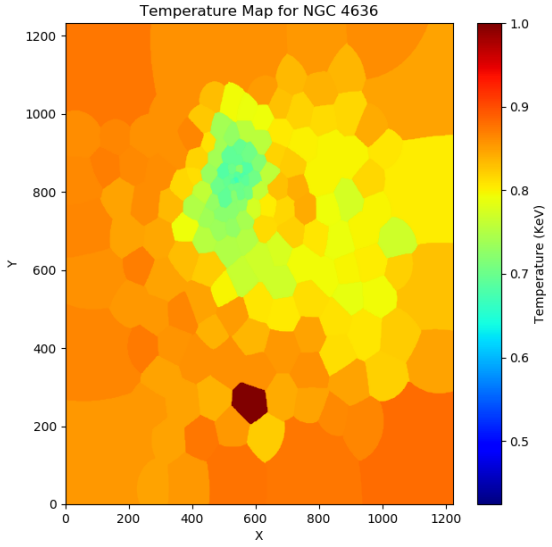
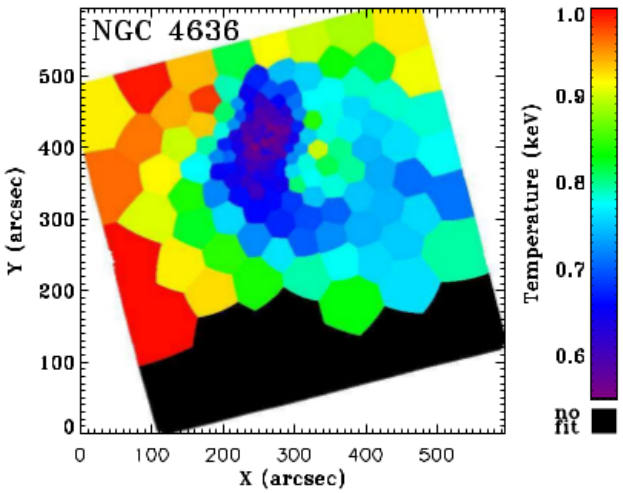


Figure 2: Final Temperature map for NGC4636 again in image units.



(a) New Temperature Map



(b) Diehl and Statler 2005 Temperature Map

Figure 3: Comparison of the two temperature maps