SHOCKS & FRONTS IN THE MERGING X-RAY BRIGHT CLUSTER ABELL 2219

A THESIS IN
Physics

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment of
the requirements for the degree

MASTER OF SCIENCE

by
JOE HUBER

B.S., Missouri State University, 2019

Kansas City, Missouri
2025

SHOCKS & FRONTS IN THE MERGING X-RAY BRIGHT CLUSTER ABELL 2219

Joe Huber, Candidate for the Master of Science Degree
University of Missouri–Kansas City, 2025

ABSTRACT

We present a deep 470 ks Chandra observation of Abell 2219, a very hot and X-ray luminous cluster experiencing a major merger event.

Abell 2219, at a redshift of z = 0.225, is only the second galaxy cluster merger where both the forward and reverse shock fronts are identified with X-ray temperature and density measurements (Russell et al., 2012, the other is Abell 2146) and one of only a handful with any shock fronts unambiguously detected as both temperature and density discontinuities. The reason for this rarity is the requirement of a near plane-of-sky merger, to mitigate the effects of projection, and also the inherently low X-ray surface brightness of shocked regions in the outskirts of clusters. Nonetheless, these sharp discontinuities, along with cluster cold fronts, have the potential to illuminate the micro-scale transport processes occurring in the hot intracluster medium.

Abell 2219 is also one of the hottest and most X-ray luminous galaxy clusters known, with a system temperature of 12 keV, and unusually, a hot yet dense core, suggesting evidence for ongoing shock activity at the core. It hosts a bright radio halo and three strong radio galaxies. Previous *Chandra* observations have revealed this system is in the early throes of a violent merger. In this work we present the development of data reduction and analysis pipelines to process our high spatial-resolution, deeper X-ray data, allowing us to confirm the presence of of both shocks and cold fronts within the cluster merger.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the School of Science & Engineering, have examined a thesis titled "Shocks & Fronts in the Merging X-ray Bright Cluster Abell 2219," presented by Joe Huber, candidate for the Master of Science degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Dr. Paul Rulis, Committee Chair
School of Science & Engineering

Dr. Leila Borvayeh
School of Science & Engineering

Dr. Da-Ming Zhu
School of Science & Engineering

CONTENTS

Chapter

APPENDIX

LIST OF ILLUSTRATIONS

LIST OF TABLES

Table Page

ACKNOWLEDGMENTS

I would like to personally acknowledge and thank the many people that have helped me and given me support during my pursuit of a Master's degree.

I extend my deepest gratitude to my research advisor, Dr. Becky Canning, for her generosity, support, and continued patience throughout this project. When I just needed someone to give me a chance, she went out of her way to ensure that I had that chance and the guidance that I needed. She is an excellent mentor and even better person, and my appreciation cannot be overstated. I am equally thankful to Dr. Paul Rulis, one of the few constant influences throughout my time at UMKC. His commitment to supporting not only me but all students within the program speaks to the kind of person he is, and we are all better for it. I would also like to acknowledge Dr. Ben Floyd, who has been a co-mentor and friend to me throughout my time spent at UMKC. I am quite sure I would not be writing this if not for his continued support and kindness, and his belief in me and my potential as a researcher. I would also like to thank the rest of the faculty who have helped in many ways during this project, and shaped me into the person that I am today.

I want to thank my peers and friends for our time together these last few years and for the support they have given me, and that I hope I have given in return. Thank you to Rachel, Jay, Alysse, Al, Aubry, Deo, Marie, Europa, and many others for being there through everything this time has thrown at us.

This research has made use of data from the Chandra X-ray Observatory and software provided by the Chandra X-ray Center (CXC). The *Chandra* data described in this

work are available in the *Chandra* data archive (https://cxc.harvard.edu/cda/). This research has made use of NASA's Astrophysics Data System as well as adstex (https://github.com/yymao/adstex). The HEASoft software library (https://ascl.net/1408.004 has also been used. This work made use of Astropy:http://www.astropy.org a community-developed core Python package and an ecosystem of tools and resources for astronomy (Astropy Collaboration et al., 2013, 2018, 2022). The python library NumPy was also used (Harris et al., 2020). Processed data products detailed in this paper will be made available on reasonable request to the author.

CHAPTER 1

INTRODUCTION

Astronomy is one of the oldest natural sciences, with the earliest written records produced by the ancient Babylonians around 1600 B.C. Driven by more than curiosity alone, the charting of the sky represented the inherent drive in our species to understand the world around us and the Universe we live in. Our understanding of the Universe has changed drastically over the last few centuries, with questions that were previously ridiculed as impossible now being considered mundane and foundational. We have come a long way from the star charts of the Babylonians, having extensively mapped not only the milky way, but many hundreds of millions of galaxies outside our galaxy have been cataloged. The cataloging of these galaxies have led us to the discovery that structure in the Universe from hierarchically.

In the current standard model of cosmology, structure forms hierarchically through gravitational instabilities that originate from small perturbations in a homogeneous and isotropic Universe. Observations of the cosmic microwave background (CMB) have shown that the Universe has a flat geometry (de Bernardis et al., 2000), and observations of Type Ia supernovae have shown that the expansion rate of the Universe is accelerating (Perlmutter et al., 1998; Riess et al., 1998), requiring the addition of a cosmological constant Λ that we refer to as dark energy (Efstathiou et al., 1990). These parameters combined are referred to as the Λ Cold Dark Matter (ΛCDM) concordance model of cosmology and represent the current standard for structure formation.

Galaxy clusters are the the last phase of hierarchical growth. In the ΛCDM model, baryons (i.e. visible matter) fall into the gravitational potential wells of dark matter halos that formed from the initial gravitational instabilities mentioned above, forming the stars and galaxies we see today. As time passes, baryons continue to fall into these potential wells, merging with the existing structures to form galaxy groups and clusters. We can use these clusters to study the cosmology of the Universe as well as laboratories for galaxy evolution, giving us insights on the microphysical transport processes and related physics of the hot-intracluster gas.

This thesis will investigate a merging galaxy cluster, Abell 2219. Below I introduce galaxy clusters, galaxy cluster mergers, and Abell 2219 in more detail. In Chapter 2 I will present the data reduction process for the *Chandra* X-ray data used in this analysis and in Chapter 3 the thermodynamic analysis of the hot gas in this major cluster merger.

1.1 Galaxy Clusters

The tendency for galaxies to cluster has been known since the late 1700's as Charles Messier and William Herschel were compiling their first catalogs of these objects, and this clustering became more apparent into the nineteenth and twentieth centuries as more and more catalogs were published. Galaxy clusters are the largest known objects that are held together by gravity, and can contain many hundreds of galaxies or more. Many of the bulk properties of clusters are thought to be determined solely by the initial conditions, dissipationless dark matter (DM) that dominates the cluster mass budget, and gravity (Kravtsov & Borgani, 2012).

While there are hundreds of galaxies in a typical cluster, these galaxies only account for about 1% of the mass fraction of the cluster. The gas within the intracluster medium (ICM) accounts for only another 9% of the mass fraction, with dark matter accounting for the remaining 90% of this fraction. Typical clusters have a mass of $10^{15} M_{\odot}$ and volumes of several hundred cubic megaparsecs.

Forming at redshifts $z < 2$, galaxy clusters represent the final phase of hierarchical growth. As mentioned above, galaxy clusters form as baryons fall into the gravitational potential wells of dark matter halos. As a cluster forms, the infalling gas and dust is compressed and subsequently heated by the shocks created by the accretion itself, eventually coming to thermal equilibrium at what is referred to as the virial temperature, determined by the total mass and virial radius of the cluster. The virial temperature is typically in the range of $1 - 15\,\mathrm{keV}$ where $1\,\mathrm{keV} \approx 1.16 \times 10^{7}\,\mathrm{K}$. At these high temperatures, the diffuse plasma seen in typical x-ray images of clusters is not actually associated with individual galaxies, but instead is the ICM. The ICM contains the majority of the normal baryonic matter in massive clusters, and although it is not directly associated with the galaxies, their properties are correlated.

1.2 Cluster Merger History

The merger of galaxy clusters are the most energetic events since the Big Bang, releasing vast quantities of thermal energy into the ICM (see Markevitch & Vikhlinin, 2007, for a review). When galaxy clusters merge, the galaxies themselves are collisionless but the gas between the clusters is collisional. A great example of this is the Bullet cluster (Markevitch et al., 2002), shown in Figure 1. In the top image the hot X-ray emitting ICM is shown in purple while below it is shown in pink. The clusters are dominated by massive red spheroidal galaxies which contain little ongoing star-formation, while bluer spiral galaxies are located towards the outskirts and in the field. In the relaxed cluster the gas and galaxies share the same center of mass, dominated by the dark matter. However, in the merging cluster the gas is offset from the main mass distribution and sharp edges can be observed, indicating shock and cold fronts. In this merging cluster image the purple indicates the majority of the mass traced by weak lensing.

When galaxy clusters collide, shocks are generated that dissipate energy from the merger into the ICM through turbulence and the acceleration of particles. Given this impact on the ICM and the subsequent gas-dynamics, these shocks likely play a key role in the evolution of large-scale structure in the Universe. These shocks are driven by the infalling subclusters, and have been found to be accompanied by “cold fronts,” or sharp

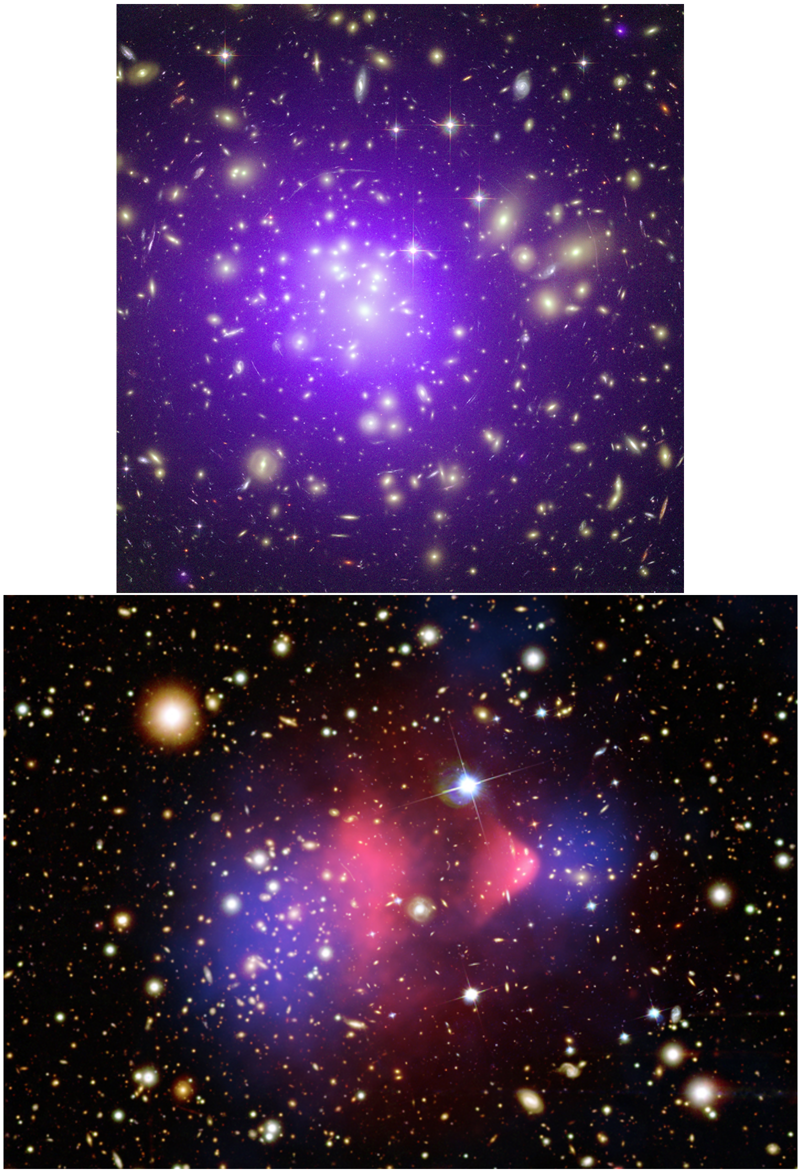

Figure 1: This figure shows a ‘relaxed’ (non-merging) galaxy cluster, Abell 1689 (above), and a merging galaxy cluster, the ‘bullet’ cluster (below). In both cases galaxies are shown in an RGB color scale. Credit: NASA/CXC

contact discontinuities between regions of gas with different entropies (Markevitch et al., 2000). Both shock fronts and cold fronts provide us with tools to study the ICM on a wide variety of scales, where the dark matter gravity, thermal pressure, magnetic fields, and ultra-relativistic particles are all at play.

Currently, there are thought to be three stages to merging clusters: a pre-merging stage where the galaxy clusters have started to fall towards each other and the ICM begins to interact, a merging stage in which the cores of each cluster pass each other, and a post-merging stage where the ICM relaxes as the shocks and turbulent gas dissipate. The timescales involved in each stage are not yet well constrained, but due to their large size and diameters, these collisions can take about a billion years to complete (White & Rees, 1978).

During a galaxy merger, we can observe these shocks and cold fronts as discontinuities in the X-ray gas, allowing us to produce temperature, pressure, and entropy maps of the merging cluster. The gas dynamics and eventual energy dissipation are poorly understood, and studying these maps allows us to constrain the gas dynamics and transport processes, giving us insights into these systems.

1.3 Observational signatures of galaxy clusters

Galaxy clusters can be observed through identification of their galaxies — typically dominated by large red galaxies which sit on a red sequence — but also by the ICM between galaxies. As mentioned, the ICM is the largest baryonic matter component of galaxy clusters. Since clusters are such massive objects with deep gravitational potential wells the gas that fills the ICM is very hot, resulting in thermal bremsstrahlung emission in the X-ray. Given the high temperature of the ICM, most if not all elements are highly ionized, resulting in a very hot plasma. Within this plasma, the electrons pass close to an ion causing the accelerating charge to emit radiation. This is the radiation we observe in the X-ray. Since the electron also loses energy in this process, it slows down—hence the name bremsstrahlung, or braking, radiation. As the electron slows down, the plasma itself also loses energy, resulting in the cooling of the gas. As a whole, the amount of emission depends on the density of the gas in the ICM. The brighter the emission, the denser the gas in the cluster. At these high temperatures the emissivity is approximated by the emissivity for *free-free* emission, given by equation 1.1.

$$\varepsilon^{ff} \approx 3.0 \times 10^{-29} erg\, cm^{-3}\, s^{-1} (\frac{T}{10^8\, K})^{\frac{1}{2}} (\frac{n_H}{10^{-3}\, cm^{-3}})^2 \tag{1.1}$$

Given that the ICM is heated to such high temperatures, X-ray spectroscopy is an important tool we can use to gain information about the structure and physics of clusters.

Once we can understand and model the X-ray spectra, the model can be used to study the thermal structure of the ICM, as well as a tool to determine cluster mass. It can also be used to probe the central regions of clusters, and the observed spectral lines can be used to perform a chemical analysis on the ICM and cluster as a whole. In a typical cluster spectra, the increasing dominance of bremsstrahlung with increasing temperature can be clearly seen. As the temperature increases, fewer ions retain electrons and the plasma is subsequently almost completely ionized at higher temperatures. By extracting spectra and fitting the features we can directly measure temperature, density, and metallicity. From these we can also derive the pressure (equation 1.2) and entropy (equation 1.3) of the gas from well known physics.

$$P_e = n_e \, k_b \, T \tag{1.2}$$

$$K = \frac{k_b \, T}{n \, e^{\frac{2}{3}}} \tag{1.3}$$

To observe these clusters in the X-ray and extract spectra from them, Earth's atmosphere dictates that you must do this from space. There are several X-ray telescopes currently in orbit, the most prominent of which are NASA's *Chandra* and Swift observatories, ESA's XMM-Newton observatory, and the JAXA XRISM telescope. A unique requirement of X-ray telescopes is that the optical systems require the use of grazing-incidence optics. Given the high energies of X-ray photons, the usual normal-incidence optics do not work as the photons simply pass through most materials, so the photons are reflected by multiple mirrors at very shallow grazing angles and focused on the detector. While this approach allows us to capture the photons, it comes with its own set of challenges and nuances explained below.

In order to be able to measure the morphologies of features such as shocks and cold fronts, very high spatial resolution is needed. The *Chandra* observatory is the only available option with a high enough resolution to see these features. At $z < 0.05$, *Chandra's* angular resolution of 1" corresponds to a linear resolution of less than a kiloparsec. The *Chandra* telescope is shown in Figure 2 along with the focal plane of the Advanced CCD Imaging Spectrometer (ACIS). X-ray photons are individually detected, allowing for the recording of their position, energy, and time of arrival, resulting in the high resolution images needed for this analysis. The four ACIS-I chips are front-illuminated, and when used together allow the observer to obtain the largest field of view. The ACIS-S chips can be used without a grating to get moderate resolution spectra, with the back-illuminated S3 chip offering the best spectral resolution for this purpose.

A point of note is that due to the nature of grazing-incidence optics, the point spread function (PSF)—which describes the shape and size of an image produced by a point source—have unique characteristics that must be understood in order to get the most out of the high spatial resolution capabilities of *Chandra*. The shape and size of the PSF varies significantly with the location and spectral energy distribution (SED) of the source. The appearance of the observed PSF also varies with source counts, especially at

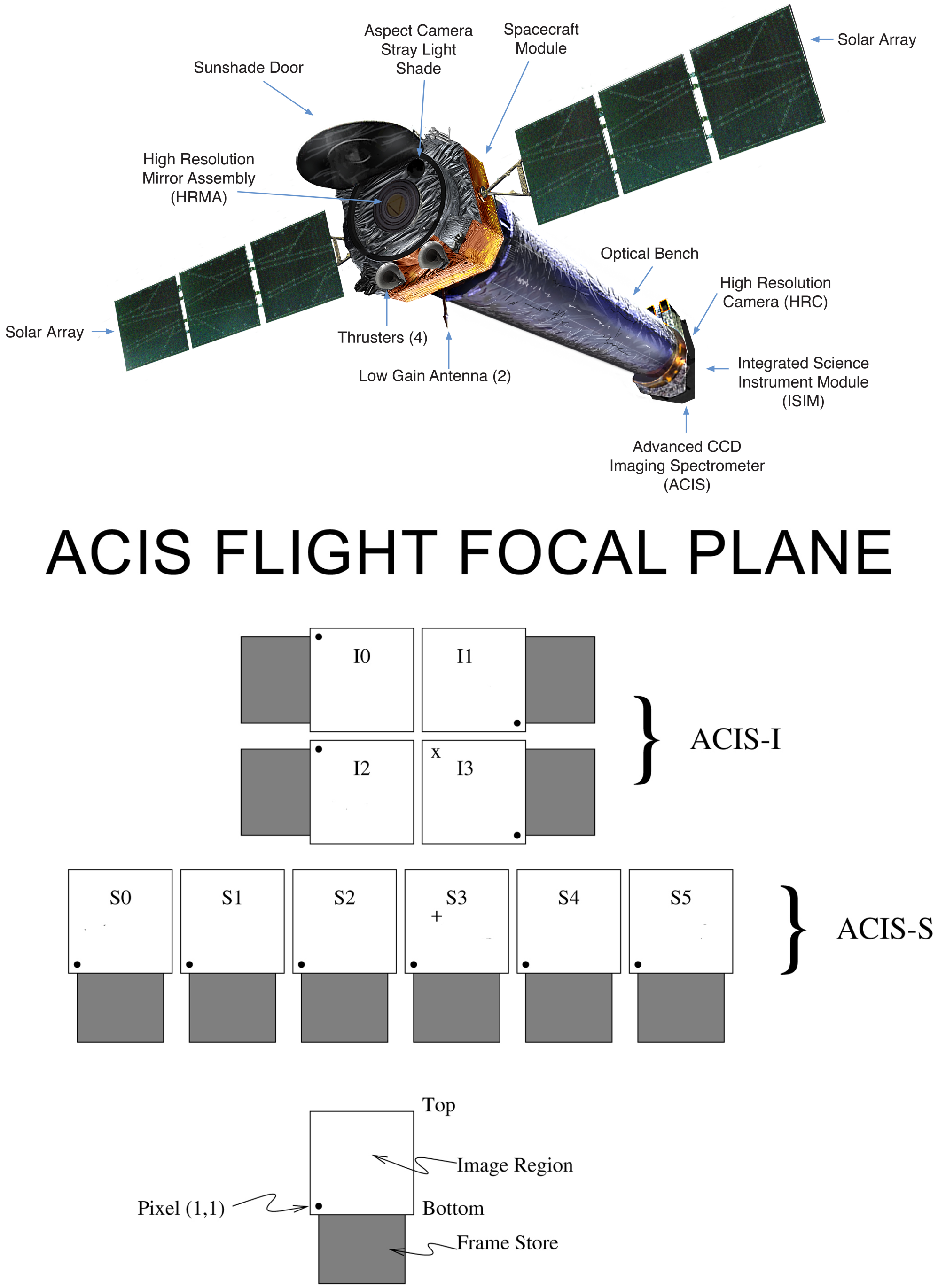

Figure 2: Top: The Chandra X-ray Space Telescope. Bottom: The Advanced CCD Imaging Spectrometer (ACIS) focal plane on board the Chandra telescope. Credit for both images: NASA/CXC

large off-axis angles that impact the apparent morphology of the object (see Figure 3).

1.4 Abell 2219

For this thesis, we study the merging cluster Abell 2219 (see figure 4). As one of the hottest and most luminous X-ray clusters known, Abell 2219 is more than twice as bright as the Bullet cluster (see Figure 1). Previous observations show clear evidence of disturbed morphology, suggesting a complex merger history with current infall of several clumps. Using *Chandra*, Million & Allen (2009) previously detected shock fronts, confirmed along with an apparent cold front by Canning et al. (2017).

1.5 Previous Work

This work builds off previous analysis as described in Canning et al. (2017) of Abell 2219, but with a deeper dataset. Direct comparisons to this work will be made where relevant.

The following chapters of this thesis focus specifically on the reduction and analysis of Abell 2219 data. Chapter 2 details the steps required to take the raw data from *Chandra* and process it as required for scientific analysis. Chapter 3 covers the subsequent analysis on the dataset, and Chapter 4 explains the results and describes planned follow-up studies. Source code for all steps is included in Appendices A & B.

In this work, we assume a concordance cosmology with $H_0 = 70$ km s^{-1} Mpc^{-1},

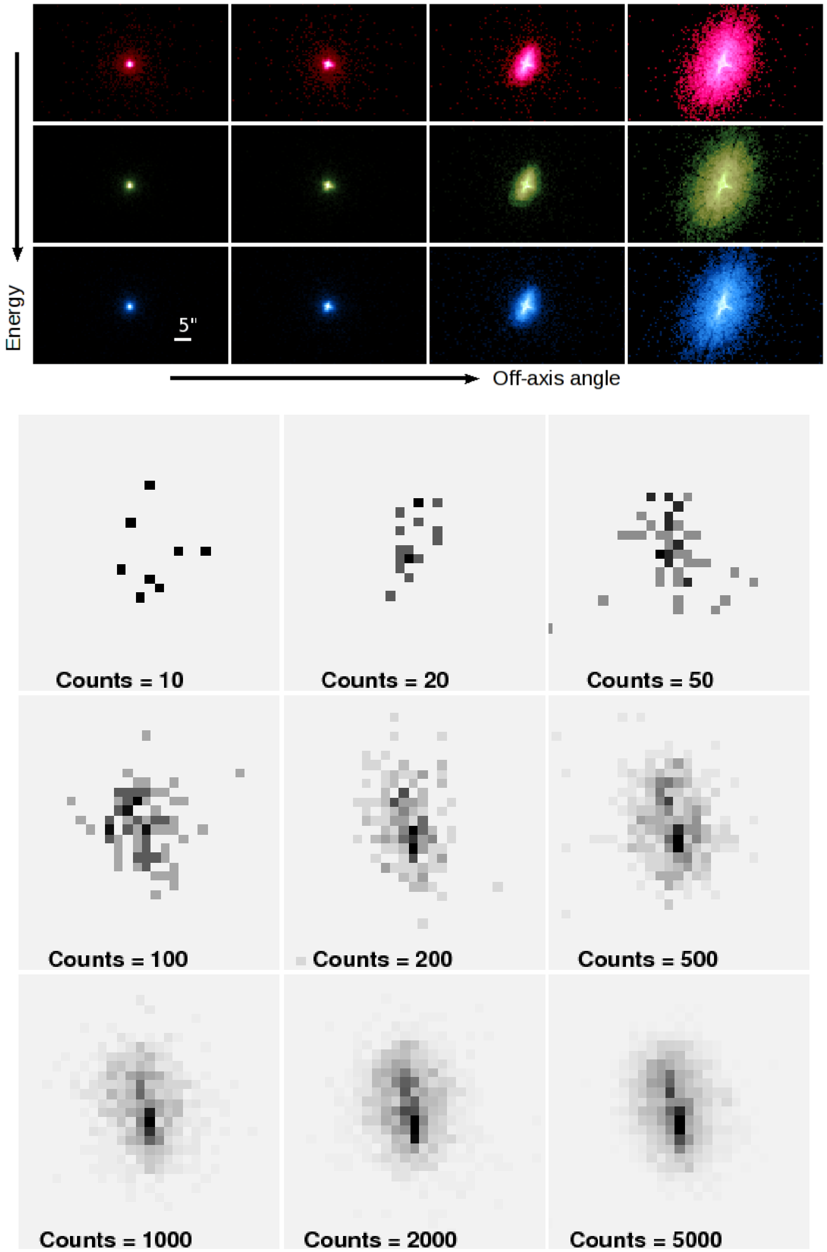

Figure 3: Top: Plot of simulated PSFs at a set of off-axis angles (0 arcmin, 2.4 arcmin, 4.7 arcmin, and 9.6 arcmin) and mono-chromatic energies (0.92 keV, 1.56 keV, and 3.8 keV) from the CSC soft, medium, and hard bands. Bottom: A simulated 1.49 keV point source, 5 arcmin off-axis. By varying the number of source counts, the apparent morphology is strongly affected. Credit for both images: CXC

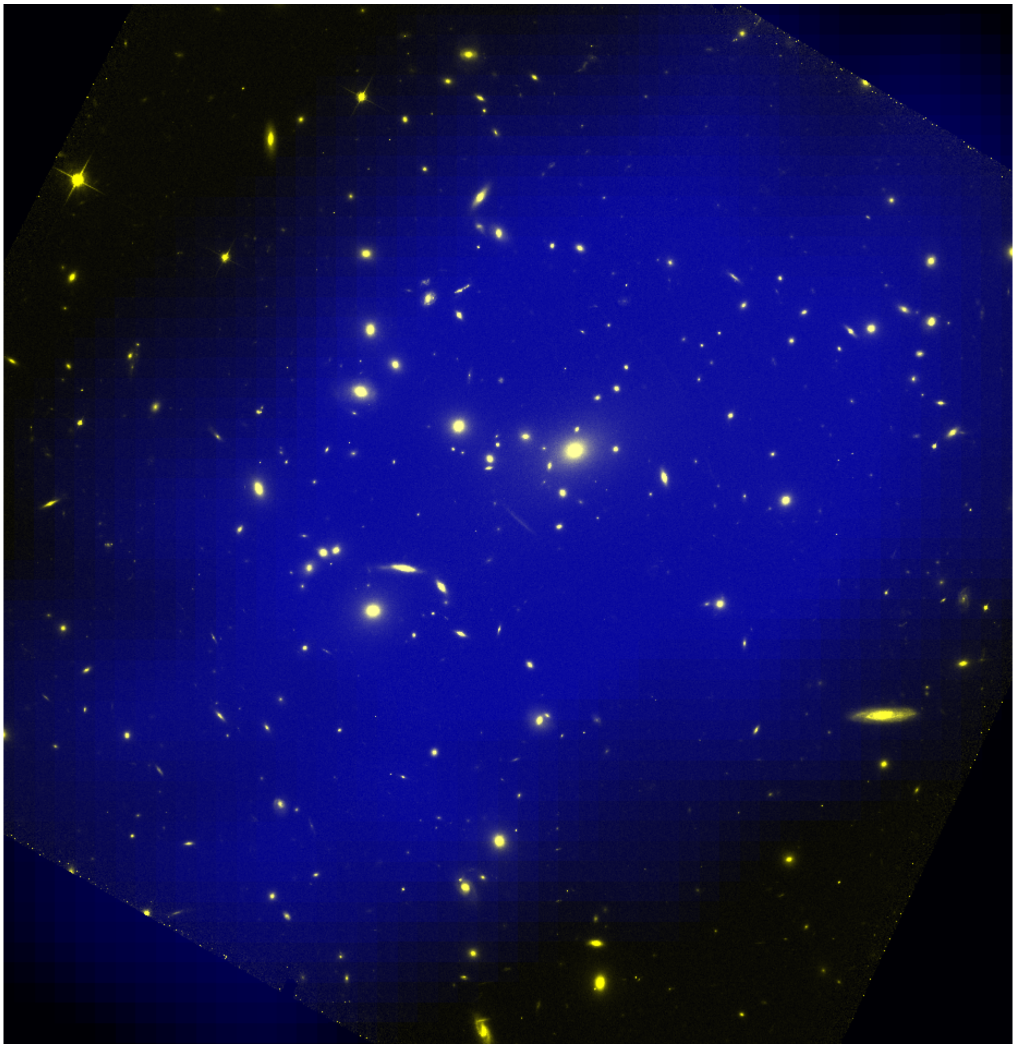

Figure 4: Abell 2219 HST/ACS F850LP optical image with Chandra broad-band x-ray overlay.

$\Omega_m = 0.3$, and $\Omega_\Lambda = 0.7$ unless explicitly stated. For this cosmology, at the redshift of Abell 2219, an angular size of 1 arcsec corresponds to a distance of 3.588 kpc.

CHAPTER 2

CHANDRA DATA PREPARATION

This chapter contains an overview of the steps taken to prepare the *Chandra* data for analysis, with more detailed descriptions for each step in the following sections. While data reduction pipelines already exist for many telescopes and wavelength bands (including some for *Chandra*), these pipelines do not always suit the needs of the project. There were several aspects of the reduction and analysis that required more control than was possible with the automated software available, leading to the development of my own pipeline. An added motivation was to develop the pipeline in such a way as to automate the reduction and analysis, and for it to be easily used on any dataset for which a similar analysis was desired.

When public data is downloaded from the *Chandra* archive, it has undergone "standard data processing" by the *Chandra* X-Ray Center (CXC). In general, there are several stages to processing data: level 0 data is the raw data received from the telescope—effectively a count of photons that have registered on each pixel of each CCD, per observation. This data is converted into the FITS file format to facilitate further processing. Level 1 data represents the initial refinement of the raw level 0 data and includes basic calibrations and instrument-dependent corrections, such as the aspect solution or setting the gain for each CCD. Level 1 outputs are reversible, but refinements are generally only needed if the user wishes to manually reprocess the data to better suit their specific science needs. Level 2 products are the intermediate data products, taking the level 1 outputs and applying standard (but irreversible) corrections. These corrections include cosmic ray rejections, position transformation to celestial coordinates, and filtering of the event file. This work uses the CIAO software package (see Section 2.1) to take and refine these data product and perform several analyses on them.

Several *Chandra* observations of Abell 2219 were made between May 26 2012 and January 3 2020. The observations were made with the Advanced CCD Imaging Spectrometer (ACIS) using CCDs 0-3 and 6. The data was reprocessed using tools from CIAO version 4.17 and CalDB version 4.12.0 (Fruscione et al., 2006). The total cleaned exposure time of the observations after the reduction process is 430 ks. This is a 250% improvement over the previous analysis in Canning et al. (2017). A description of the included observations is given in Table 1.

2.1 Data Reduction Pipeline

The following sections contain a description of the data reduction pipeline (in plain text) and detail the process of taking the raw data from the *Chandra* observations and reprocessing it to be suitable for this analysis. The data reduction and subsequent

OBSID	Date	Instrumental Setup	Cleaned Exposure (s)
13988	2012-05-26	ACIS-I, 01236	8670
14355	2012-06-28	ACIS-I, 01236	26100
14356	2012-10-15	ACIS-I, 01236	41450
14431	2012-05-27	ACIS-I, 01236	34700
14451	2012-06-26	ACIS-I, 01236	17660
20588	2018-08-18	ACIS-I, 01236	31600
20589	2018-09-25	ACIS-I, 01236	62500
20785	2018-11-23	ACIS-I, 01236	25000
20951	2018-09-28	ACIS-I, 01236	63900
20952	2018-10-02	ACIS-I, 01236	44800
21966	2018-11-20	ACIS-I, 01236	20100
21967	2018-11-24	ACIS-I, 01236	29700
21968	2019-01-03	ACIS-I, 01236	22800

Table 1: A list and description of *Chandra* observations used in this analysis.

analysis make heavy usage of the CIAO software package, developed by the CXC to analyze data from *Chandra*. The newest version of the software has integrated several useful packages such as DS9 (general image processing) and XSPEC (spectra model fitting). Usage of these and other packages are explained below, with source code and accompanying documentation contained within Appendix A. A "wrapper" script was created, containing each piece of the data reduction process. This allowed all data reduction steps to be ran automatically and in sequential order, without additional input from the user.

There are many pre-made tools and scripts that have been created by the *Chandra* team that handle some of the individual processing steps, effectively automating the detailed sub-processes required by the full reduction process. Several of these tools are implemented within the pipeline and discussed in further detail where relevant.

2.2 Data Reprocessing

Once we have our data downloaded, we need to reprocess our dataset using the `chandra_repro` tool, which automates the recommended data processing steps presented in the CIAO analysis threads. The script reads data from the standard data distribution (detailed above) and creates a new bad pixel file, a new level 2 event file, and a new Level 2 Type II PHA file with the appropriate response files. The bad pixel file must be set by the user, as it is specific to each observation. It is important to use the most accurate list of bad pixels, as this file is used in many subsequent data processing steps.

The PHA file refers to the pulse height, an instrumental quantity related to the photon energy; the mapping from PHA to energy varies with detector position and sometimes time, and we also use the pulse invariant (PI) value, which is the PHA corrected to a standard energy scale. These details are necessary to ensure the accurate extraction of spectra (see Chapter 3). Parameters can be used if more control over certain reprocessing steps is required.

There are four main data reduction processes that require further intervention: correct alignment of the astrometry between the different observations, filtering of the photon arrival events to remove spurious flares, generation of appropriate backgrounds, and combining observations into a mosaic to increase signal-to-noise ratios and improve upon the spatial resolution. These steps are handled manually and detailed in the following sections.

2.3 Update World Coordinate System

The next step was to manually correct the absolute astrometry for each observation by updating the world coordinate system (WCS). This is done to ensure that all of the observations are aligned, so that the images can be stacked accurately later in the analysis process. To update the WCS, a reference for the transformation needs to be identified. The deepest dataset - i.e. the observation with the longest integration time (ObsID 20951 for our dataset) - is used for this purpose. This was achieved in two steps using CIAO tools: `wcs_match` and `wcs_update`. `wcs_match` compares two sets of source lists from the same sky region. If three or more sources are found to be a close match in position, `wcs_match` then calculates a transformation matrix. This matrix is then used with `wcs_update` to align the input set of source positions with the reference set of source positions (ObsID 20951). This is done by modifying the right ascension (RA) and declination (DEC) of the aspect solution (asol) file based on the transformation matrix generated in the previous step. This transformation file calculates four equivalent parameters that are used to update the output WCS; two translational, one rotational, and one scaling parameter.

This script also makes use of the `fluximage` and `wavdetect` tools from CIAO to create some of the main data products used in the rest of the reduction and analysis. `Fluximage` is used to create exposure-corrected images of an observation, and allows for the filtering of the image over one or more energy bands. The tool automatically creates matching exposure maps as well as additional instrument specific files. These output files can then be used with the `wavdetect` tool, which uses a Mexican-hat wavelet function of varying scales to identify and list bright sources. This tool is also used in the analysis to remove point sources from our mosaicked image (see Chapter 3.1).

2.4 Deflare

Once the images have been reprocessed and WCS-corrected, the next step is to "deflare" the observations. This involves removing unwanted data from flares that would interfere or washout the data and features that would otherwise be seen during the analysis. Here we define a flare to be cosmic rays, AGN flux variability, and any other event or phenomena that is not part of the merger process. To do this, the WCS-corrected event file for a given ObsID is filtered over a preferred energy range (0.5–9.0 keV). This energy range is chosen as it best represents the actual emission from the merging galaxy clusters and is optimized by the design of *Chandra's* mirrors and detectors.

This filtered image, along with new aspect solution, bad pixel, and mask files from the previous reprocessing step, are then used to create exposure-corrected images and exposure maps for each ObsID. Using the exposure-corrected image, the `wavdetect` tool is ran to identify any point sources—likely active galactic nuclei (AGN). This then generates a region file containing these point sources that is then used to mask them out and remove them from the image. Now that the image has been cleaned, a light curve—a time-series showing brightness over time—can be extracted (see Figure 5).

The light curve is then used to run the `deflare` tool, which is a simple iterative sigma-clipping routine that detects and remove flares from a light curve. This ultimately generates a good time-interval (GTI) file—a list of all the "clean" data over the integration

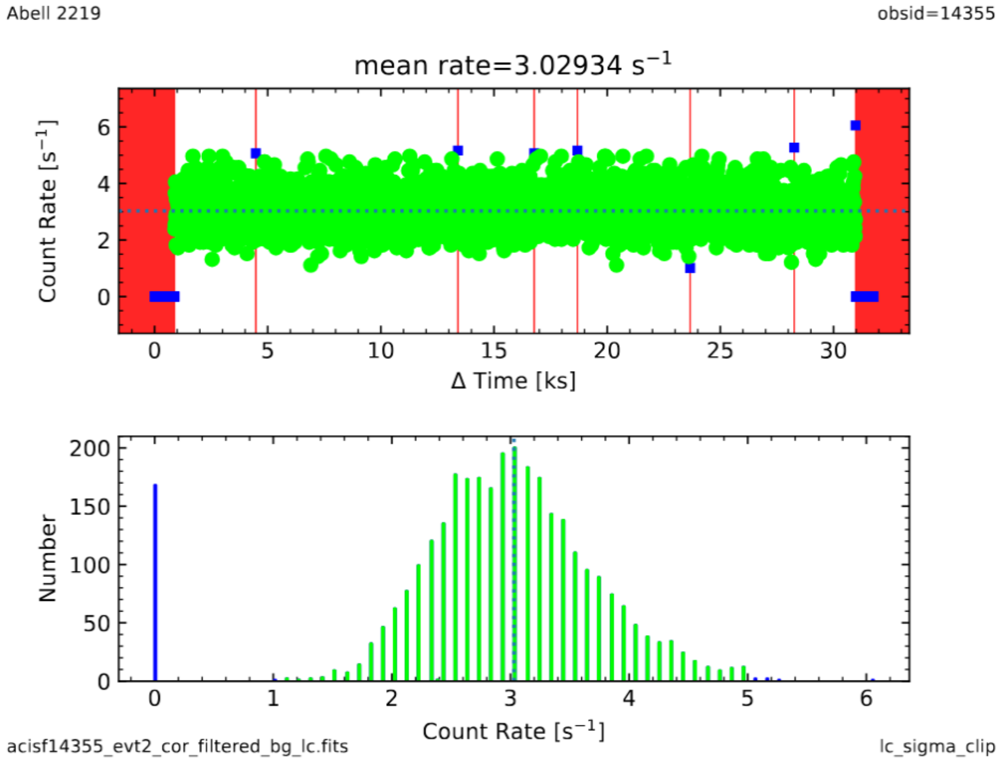

Figure 5: Light curve and histogram of ObsID 14355 after running the `deflare` routine, plotting counts per second as a lightcurve (top) and GTI (bottom). Red indicates excluded times; green indicates good times; blue indicates the excluded events.

time—of each observation. Once this GTI file has been generated, it can then be used to filter the observation, thus removing all flares, point sources, and other unwanted data from the time-dimension of each image.

2.5 Backgrounds

Now that the images have been deflared, background files that are compatible with each observation's deflared image need to be created. This is done using the CIAO tool

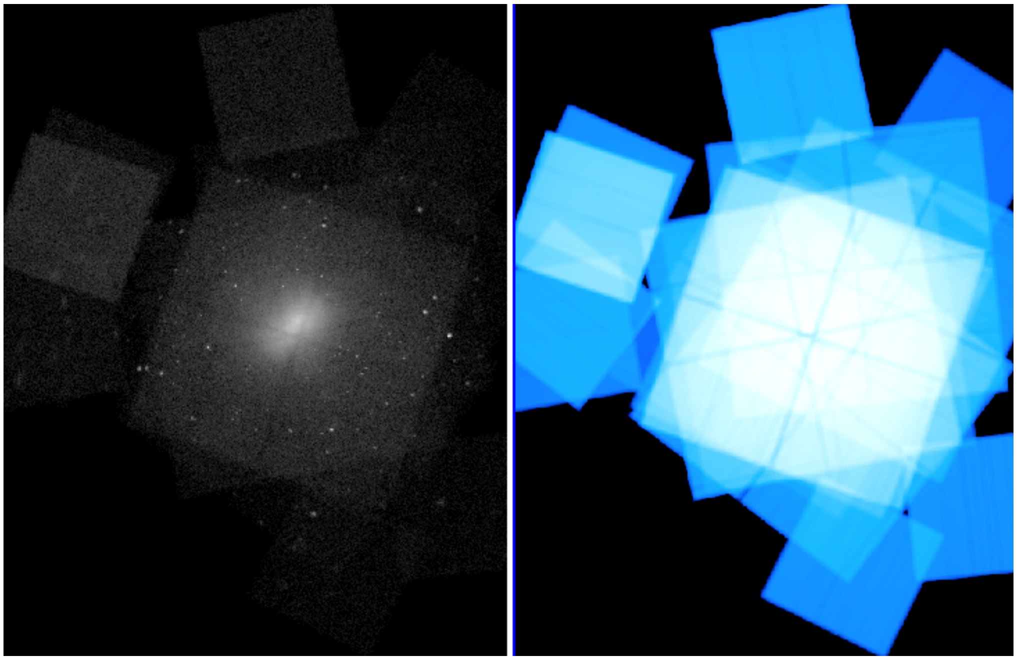

Figure 6: Final products from the reduction processes. **Left**: Mosaicked broad-band image. **Right**: Background exposure map.

`blanksky` which will generate this background file automatically, locating the correct background files and reprojecting them to match the input file. The background files `blanksky` looks for are specific to the ACIS-I instrument, created by co-adding sets of observations, taken from throughout the *Chandra* mission, of relatively empty regions. As these are multi-CCD images, the background files for each CCD are combined before it is reprojected to match each input deflared file (see Figure 6).

Band	Min Energy	Max Energy	Effective Energy
Broad	0.5	7	2.3
Soft	0.6	1.2	0.92
Medium	1.2	2.0	1.56
Hard	2.0	7.0	3.8

Table 2: Energy bands used in this project. Energies are given in keV and the effective energy is the monochromatic energy used to calculate the exposure map for each band.

2.6 Mosaic Images

Once the proper background files have been created for reach observation, these images and backgrounds can be stacked to create a single WCS-corrected, deflared, exposure-corrected, background-subtracted mosaic image of Abell 2219 (shown in Figure 7). To do this we use the CIAO tool `merge_obs` to automate much of the process, and also allowing the process to be automatically repeated for several different energy bands. For the purposes of this project, the bands used are analogous to using wavelength filters in optical wavelengths but due to the unique nature of X-ray photometry, the bands can be defined in a post-observation sense instead by filtering over a desired energy range (see Table 2).

After the initial mosaic has been created, the CIAO tool `blanksky_image` is then used to create a scaled background as well as a background subtracted image to go with our mosaic. Finally, `merge_obs` is run again to combine the reprojected background event files created earlier, effectively building a mosaicked background image for each of the energy bands specified during the first `merge_obs` call.

2.7 CCD Identification and Regions

The last step in the data reduction process was to identify which CCDs were used in each observation, as well as define which CCD would be considered the primary CCD. Chip 3 on the ACIS-I instrument (see Figure 2) contains the on-axis point where the psf is least distorted, (see section 1.3 for more detailed explanation), so we center our observations here, setting it as the primary CCD.

Once the relevant CCDs are identified, we needed to create and save region files for each, literally defining the region of the sky that each CCD covered. The coordinates for these regions were then converted into WCS, matching later products generated during the analysis phase. This was automated using a script, saving the basic information to a plain text file to be used during specific steps in the analysis.

2.8 Publication Quality Image

For the purposes of generating a publication-quality image of our data, we can mask out the point sources using `wavdetect` and then using `dmfilth` to infill the removed sources with appropriate data. To do this, annuli are created as regions in the area immediately surrounding a removed point source, using a sample distribution of pixel values from this annuli to infill the removed region. This is a very good approximation of

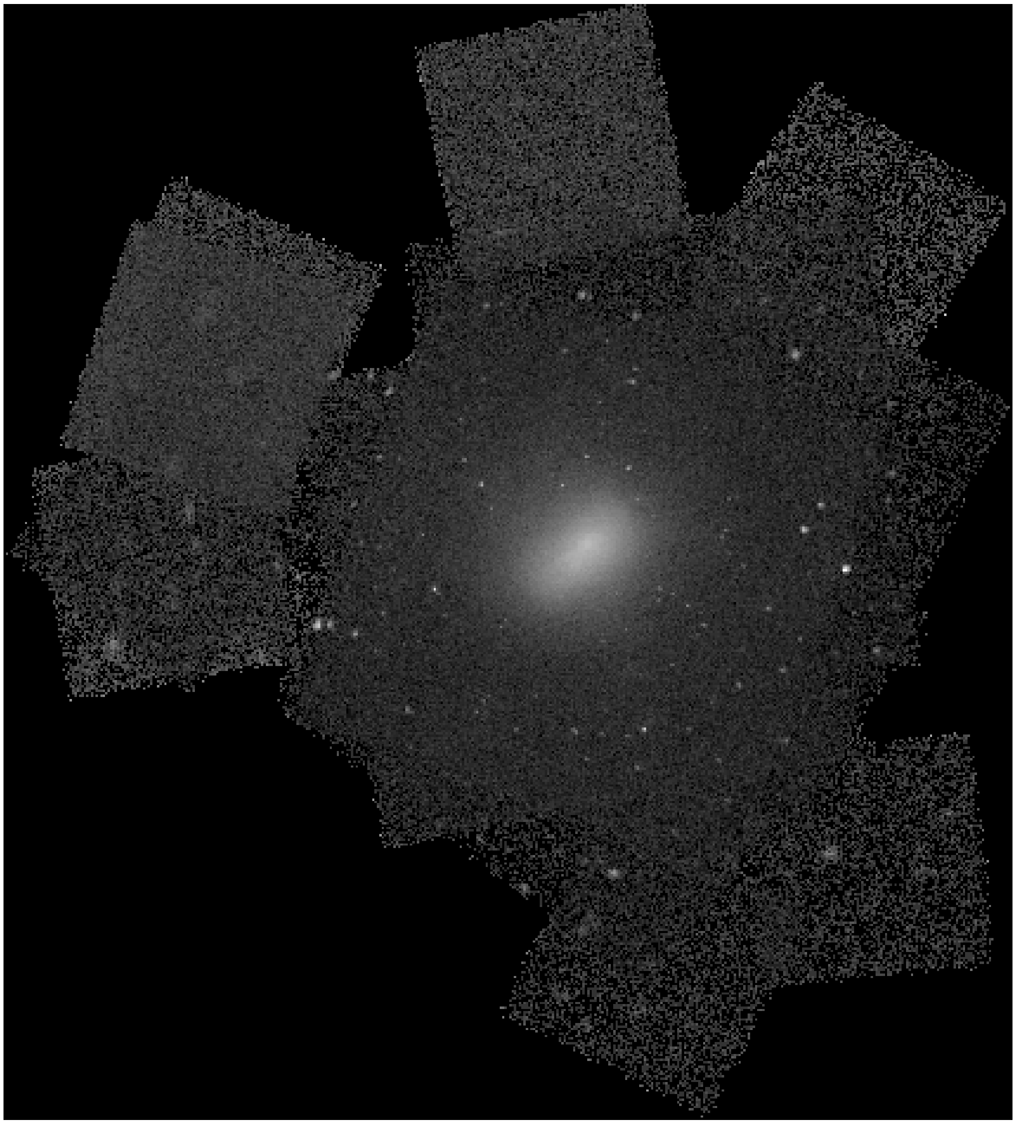

Figure 7: Mosaicked broad-band flux-image of Abell 2219

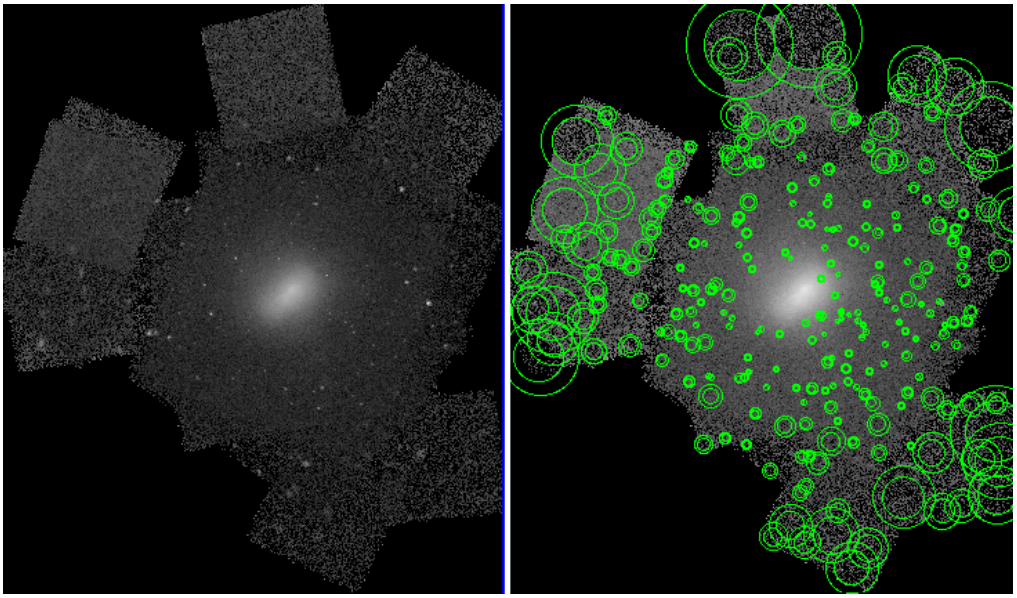

Figure 8: Example `dmfilth` output. **Left**: Mosaicked broad-band image. **Right**: Background annuli used to fill in removed point source data.

what the cleaned data should look like, and prevents the loss of features in the image (see Figure 8).

CHAPTER 3

ANALYSIS

Having detailed the completion of the Level 2 intermediate data products, this chapter discusses the creation of the Level 3 science data products and their use in the analysis. Detailed descriptions are included, and source code and accompanying documentation is included in Appendix B.

3.1 Contbin

The first step of the analysis was to "clean" our mosaicked image, removing point sources and flares. This is done in a similar way to deflaring step in the reduction pipeline, using the points sources found with `wavdetect` as a mask for the mosaic.

The `Contbin` software (Sanders 2006) was used to bin the masked-out images, filtered over a range of 0.6–7.0 keV. `Contbin` is a contour binning and accumulative smoothing software that allows us to bin the images by surface brightness. Binning by surface brightness is very important, especially as opposed to Voronoi tesselation or other binning choices. If you were to use Voronoi tesselation, you would be losing most if not all of the high resolution data gained by using *Chandra*. For example, if you bin by temperature using Voronoi tesselation across a region with temperature discontinuities, you would be getting the average of hotter and colder regions resulting in a "warm" region.

Binning by surface brightness allows us to maintain the resolution needed to see the discontinuities at the shock and cold fronts. We chose to bin with a signal-to-noise ratio of 30 and a smoothing factor of 15, maximizing the detail seen at the highest signal-to-noise. The software generates several files, the most important of which are the binned image and the binmap (see Figure 9).

3.2 Make Spectra

Using the `Contbin` software, we generate region files matching each bin from the binmap output in the previous script. Once generated, you need to determine which CCDs each region lies on (as each CCD has a different response profile and background scaling correction). If a region covers more than one CCD, we split the region into new regions according to the chip boundaries (e.g., xaf_1.reg becomes xaf_1_0.reg and xaf_1_3.reg). This step is performed by the `region_split` script.

3.3 Extract Spectra

Next the spectra are extracted per region using the `extract_spec` script. This involves creating several files that the `XSPEC` modeling software needs to extract the spectra, including: a background event file, a file containing a list of detector channels and weight map, a response-matrix-file (RMF), an auxilliary response file (ARF), and a file that groups the detector channels together to meet a given count threshold using the

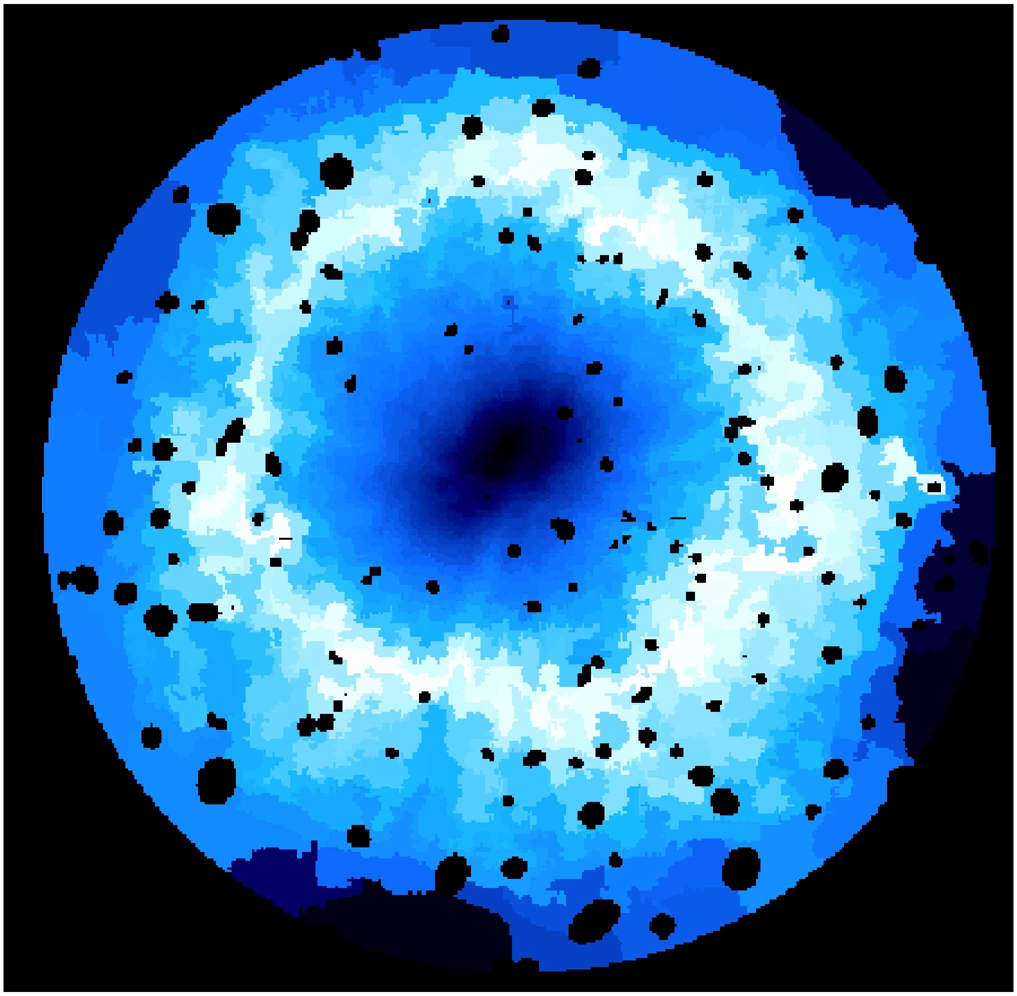

Figure 9: Example `Contbin` output. The binmap is a FITS image where each pixel is numbered according to which bin it is in. There are 876 bins in this image.

`grppha` tool from the `HEASoft ftools` library. The background event file is required as each spectra needs to have an appropriate background subtraction, specified by region and CCD. The `dmextract` tool is used to extract the source for each region and generate an accompanying weight map. The weight map allows the software to determine the appropriate weighting for calibrations which depend on detector position, and the .pi is the file type expected by `XSPEC`. The RMF contains the mapping between the physical properties of photons and their detected properties, for a specific detector. The ARF contains the combined effective area and the quantum efficiency (QE) of each region.

3.4 Spectra Model Fitting

Using the `XSPEC` software, an `APEC` thermal model is fit to the extracted spectra and results are saved as statistics per region. The initialization of `XSPEC` is handled by the `fit_contbin_phabsapec.tcl` file (see Appendix B). This file includes defined parameters for the modeling, as well as the statistics used and parameters for the Markov Chain Monte Carlo (MCMC) fitting method. The extracted statistics per region can then be used to plot thermodynamic plots, covered in the next section.

3.5 Thermodynamic Maps

Using the extracted spectra, a projected map of temperature (in units of keV) in Figure 11 of Abell 2219. We can see clearly that the previously identified shock and cold

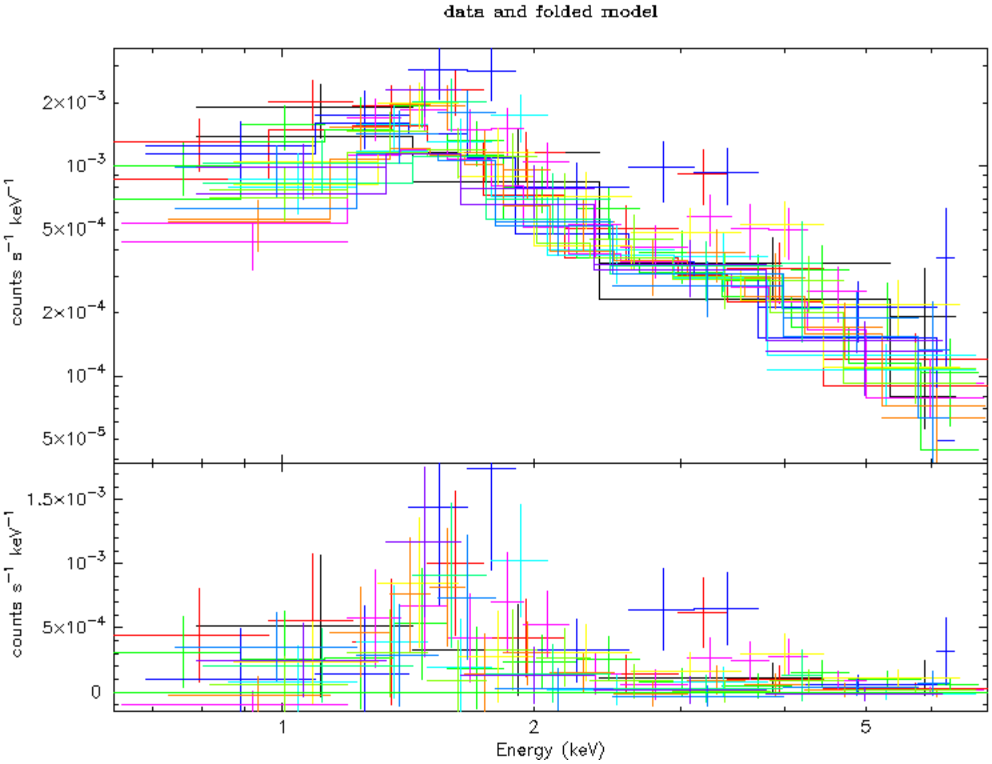

Figure 10: Example output of an extracted spectra and model fitting from our analysis. Spectra shown is for one region only.

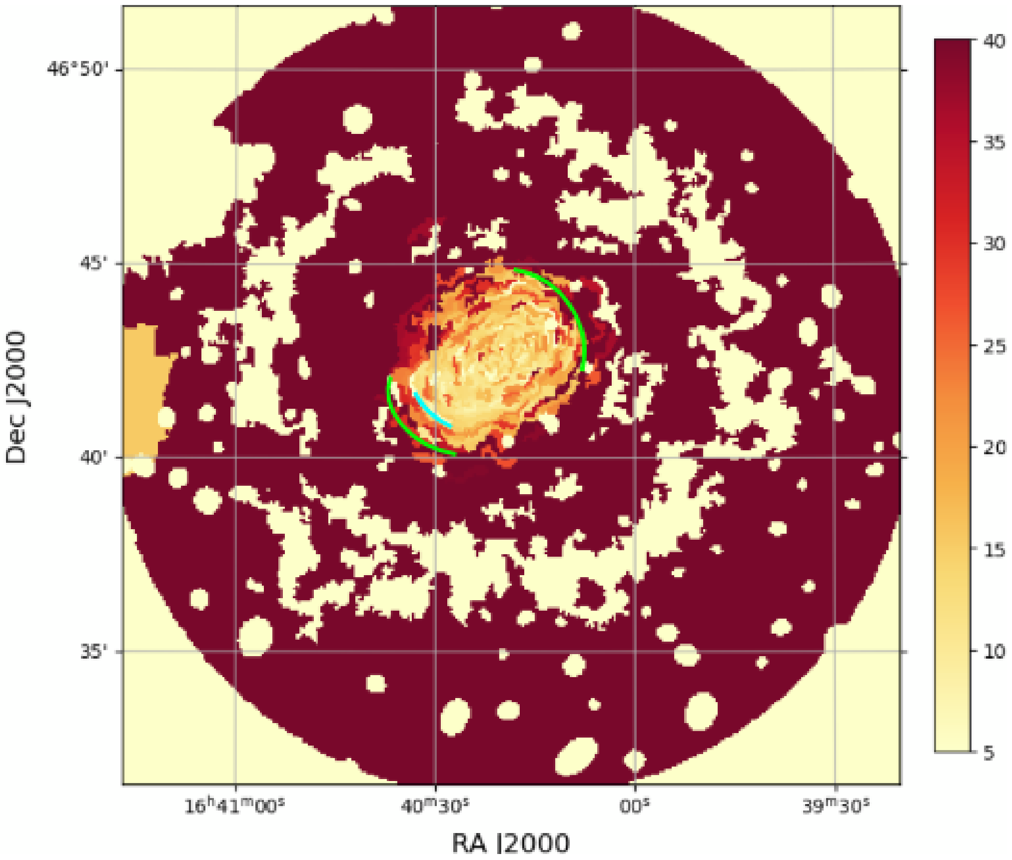

Figure 11: Temperature map of Abell 2219 [keV]. The shock fronts are represented by the green arcs, with the blue arc representing the cold front.

fronts are recovered and confirmed at our higher resolution. It should be noted that not all of the regions appear to be recovered, implying there is an issue either with the splitting of regions as detailed in section 3.2, or that the bins contain insufficient counts for spectra to be accurately and reliably extracted for those regions. This is an as of yet unsolved issue, and will be investigated as part of the planned future work for this project.

CHAPTER 4

CONCLUSIONS AND FUTURE PROJECTS

This chapter concludes the thesis and will provide context to the research at large. It also includes a discussion discuss future projects that can extend the research in the future, described in section 4.2.

4.1 Conclusions

In this thesis, we have recovered the previously identified shock and cold fronts with higher resolution and signal-to-noise. The initial result confirms the validity of this analysis, and gives good reason to be excited for the results of further analysis. Once completed, we will be able to put our results in context with similar studies carried out on other merging galaxy clusters.

4.2 Extensions and Follow-up Studies

There are several additional analyses planned to further this study and will be completed in the near-term. As mentioned in Section 3.5, there are aspects of the analysis that have errors or deficiencies that need to be investigated. A thorough debugging of the source code will be required, and should be completed prior to any other extensions or follow-ups. Once complete, we will generate the other thermodynamic maps we have data for, namely projected maps of density, pressure, and entropy. With these maps, surface brightness profiles can be extracted using radial binning. There is weak evidence (Canning et al., 2017) for temperature and density discontinuities in an arc roughly $\approx 20 - 30$ arcsec from the core, and it is expected that we can confirm this with the deeper data of this analysis.

The `contbin` software allows the user to choose what signal-to-noise and smoothing parameters you wish to use, allowing you to find a good balance between fine enough binning so as to isolate any features of interest, but not so fine that we lose our statistics or the ability to resolve these features. As such, it is planned to rerun the analysis pipeline, iterating over several combinations of signal-to-noise and smoothing to find the right balance. See figure 12 for a comparison.

A further follow-up study will be the investigation of the morphological connection between the radio halo and the shock fronts within Abell 2219. Radio halos are large-scale sources of diffuse radio emission found in the center of some galaxy clusters. When in a merger, these radio halos are thought to created by turbulence (Brunetti et al., 2001; Petrosian, 2001), while radio relics are generally located at the edge of clusters and associated with the acceleration of particles at shock fronts (Ensslin et al., 1998). Abell 2219 contains three strong radio galaxies (Owen et al., 1992), and combining the radio imaging data of these with the polarization data, we can investigate the previously

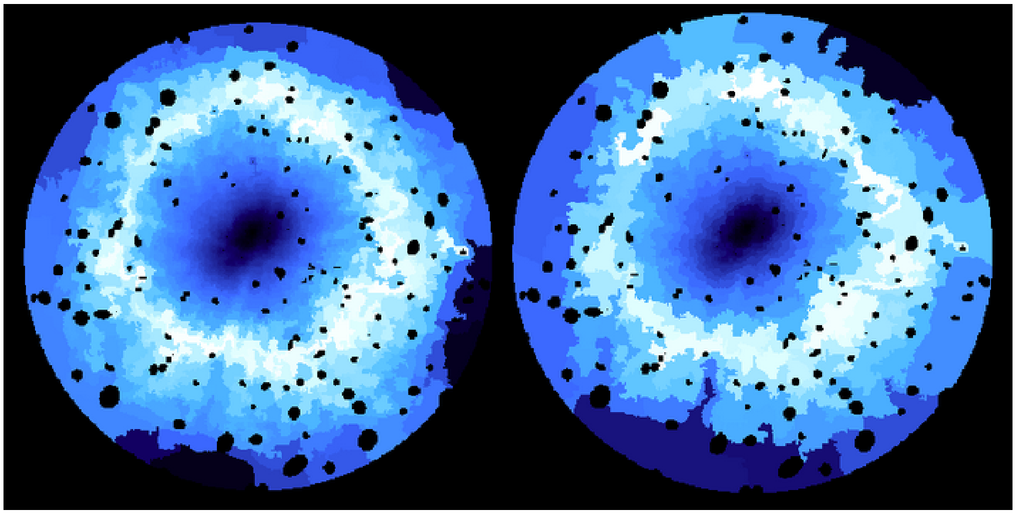

Figure 12: A comparison of changing parameters within `Contbin`.
Left: Signal-to-noise of 30 and smoothing factor of 15. Right: Signal-to-noise of 60 and smoothing factor of 60. The change in binning is apparent upon close inspection.

mentioned morphological connection. We will also be able to better determine and constrain the role that shocks and cold fronts are playing compressing the cluster fields and re-accelerating the electrons and ions within the ICM.

The completion of these follow up studies is planned for the near-term, with the intention of the publication of the results in peer-reviewed journals.

APPENDIX A

ABELL 2219 DATA REDUCTION PIPELINE

A.1 Description of Use

See Chapter 2 for an overview of the data reduction pipeline and the code contained therein. Appendix A includes all scripts required to prepare a given dataset for analysis, automated by a wrapper script (`dr_pipeline`). The pipeline consists mainly of bash scripts that call both python and compiled binaries to complete our tasks. One of the secondary objectives of this project was to create this pipeline (including the analysis in Appendix B) so that anyone could do a similar analysis on a chosen dataset with limited input. These scripts were written in such a way that this could be achieved, really only requiring the user to define which object they wish to download data for, and to edit the parent directory paths for the wrapper scripts.

A.2 Source Code

The beginning of each script contains a brief introduction/explanation in plain text of that script and its purpose.

A.2.1 Data Reduction Pipeline

```
#! /bin/bash

```

```
# This wrapper calls - in order - the scripts needed to
 ↪  complete all data reduction steps required before
 ↪  starting
# analysis on our dataset

# Define our directories structure
obsids_dir="/home/jhuber/Abell_2219/data"
code_dir="/home/jhuber/Abell_2219/code"

# 1) Find and download ObsIDs for a given source/object
 ↪  from the Chandra database
./download_data

# 2) Reprocess our data with recommended steps from CIAO
./chandra_reprocess

# 3) Update the world coordinate system (WCS) for our
 ↪  dataset
./update_wcs

# 4) Identify flares and remove them
/remove_flares

# 5) ACIS Backgrounds using blanksky command
./make_bkg_event

# 6) Mosiac evt files
./make_mosaic

```

```
# Find the ACIS CCDs used/on for our observations and create region files for the CCDs
./ccds_and_regions
```

A.2.2 Download Data

```
#! /bin/bash

# Download observation data from the Chandra database for a given source/object (e.g. Abell 2219) utilizing the
# download_chandra_obsid tool from CIAO (https://cxc.cfa.harvard.edu/ciao/ahelp/download_chandra_obsid.html)

# Move into directory where the ObsID data will be stored
cd ~/Abell_2219/data

# Get the relevant Obs IDs for your source/object; Define that we only want to process data taken with the ACIS-I
# instrument (ignoring ACIS-S) and print this list and relevant data to a file "chandra_obsids", where data includes
# ObsID, observation date, integration time, etc. This list will be used in most scripts
find_chandra_obsid A2219 > chandra_obsids
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

# Iterate over the chandra_obsids list, downloading the per-observation data and writing it to a directory matching
# the ObsID (e.g. /14355)
```

```
for OBSID in $A2219_OBSIDS; do
        download_chandra_obsid $OBSID
done
```

A.2.3 Chandra Reprocess

```
#! /bin/bash

# Reprocess a Chandra dataset using the chandra_repro script, which automates the recommended data processing steps
# presented in the CIAO analysis threads (https://cxc.cfa.harvard.edu/ciao/ahelp/chandra_repro.html)

# The script  reads data from the standard data distribution (e.g. /primary and /secondary directories) and creates a
# new bad pixel file, a new level=2 event file, and a new level=2 Type II PHA file with the appropriate response files.
# There are parameters to control certain reprocessing steps with a finer-grained approach; we do not need this control,
# and therefore choose to run the default reprocessing script.

# Define the directory where the ObsID data is stored
parent_dir='/home/jhuber/Abell_2219/data'

```

```
# Repeat the step from download_data to define that we only want to process data taken with the ACIS-I instrument
# (ignoring ACIS-S) and iterate over this list. See that script for more details.
cd ${parent_dir}
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

# For each observation, run the chandra_repro tool with output to the OBSID/repro folder
# We also define our directory structure with /working per OBSID and subdirectories relevant to our reduction pipeline
# where /base is the update_wcs script output, and /mosaic for our final mosaiced image
for OBSID in $A2219_OBSIDS; do
        chandra_repro indir=${OBSID} outdir=${OBSID}/repro cleanup=yes clobber=yes
  mkdir -p "${OBSID}"/working/{base,deflare,backgrounds,images,ccds,regions}
  mkdir -p "${parent_dir}"/mosaic
done
```

A.2.4 Update World Coordinate System

```
#! /bin/bash

# Update the world coordinate system (WCS) for our dataset (all ObsIDs) through primarily two steps using CIAO tools:
```

```
# 1) Run wcs_match, which compares two sets of source lists
 ↪  from the same sky region. If three or more sources are
 ↪  found
# to be a close match in position, wcs_match will calculate
 ↪  a transformation matrix which, when used with
 ↪  wcs_update,
# can be used to align the input (infile) set of source
 ↪  positions with the reference (refsrcfile) set of source
# positions. If only one source is found to be a match, a
 ↪  simplified, translation-only transformation matrix can
 ↪  be
# computed
# 2) Run wcs_update, which will modify the aspect solution
 ↪  (asol) file RA and DEC or infile WCS based on the
# transformation matrix generated in the previous step.
 ↪  wcs_update implements the transformation file data by
# calculating four equivalent parameters to use to update
 ↪  the outfile; two translational, one rotational, and one
# scaling parameter

# Define the directory where the ObsID data is stored
parent_dir='/home/jhuber/Abell_2219/data'
cd ${parent_dir}

# Define our reference dataset (we choose our deepest
 ↪  dataset, i.e. the ObsID with the longest integration
 ↪  time)
ref_OBSID=$(awk 'time<$5 || NR==2{ time=$5; obsid=$1 } END{
 ↪  print obsid }' chandra_obsids)
```

```

# Move into the relevant ObsID subdirectory where we wish
 ↪  to store the script data outputs
cd ${ref_OBSID}/working/base

# Create counts image and exposure map using the fluximage
 ↪  tool from CIAO
# The fluximage tool creates exposure-corrected images of
 ↪  an ACIS (or HRC) observation given an events file and
 ↪  one or
# more energy bands. It automates the creation of the
 ↪  aspect histogram, instrument, and exposure maps. The
 ↪  output image
# size can be automatically chosen to cover all the data in
 ↪  the observation, or it can be set to match an existing
# image. At this stage we are using our chandra_repro
 ↪  outputs as our input files
fluximage ../../repro/acisf${ref_OBSID}_repro_evt2.fits
 ↪  acisf${ref_OBSID} bin=1 band=broad psfecf=0.9
 ↪  clobber=yes

# Create psfmap; this step is redundant and can be removed
 ↪  if desired, as fluximage can also generate a psfmap
# If fluximage is used, the [psfecf] parameter can be used.
 ↪  This calculation is done by the mkpsfmap tool, and the
# energy is taken to be the same value used to create the
 ↪  exposure maps (specifically, the effective energy
 ↪  described in
```

```
# the bands parameter of the weight file, i.e. band=broad
 ↪  (0.5-7.0 keV, effective energy=2.3 keV)
#mkpsfmap acisf${ref_OBSID}_broad_thresh.img
 ↪  acisf${ref_OBSID}_broad_thresh.psfmap energy=2.3
 ↪  ecf=0.9 mode=h clob+

# Run the wavdetect ciao tool; wavdetect is a Mexican-Hat
 ↪  Wavelet source detection tool. It correlates the image
 ↪  with
# wavelets of different scales (selected by the user) and
 ↪  then searches the results for significant correlations.
 ↪  It is
# a wrapper for the tools wtransform and wrecon, and works
 ↪  in two steps:
# 1) wtransform detects probable source pixels within a
 ↪  dataset by repeatedly correlating it with "Mexican Hat"
 ↪  wavelet
# functions with different scale sizes.
# 2) wrecon generates a source list with information from
 ↪  each wavelet scale. For each source, a cell is computed
 ↪  that
# contains the majority of the source flux, and source
 ↪  properties are computed within that cell.
# The input files below are the output files from the
 ↪  fluximage call above
punlearn wavdetect
wavdetect \
  infile=acisf${ref_OBSID}_broad_thresh.img \
  psffile=acisf${ref_OBSID}_broad_thresh.psfmap \
```

```
  expfile=acisf${ref_OBSID}_broad_thresh.expmap \
  scales="1 2 4 6 8 12 16 24 32" \
  outfile=acisf${ref_OBSID}_broad.src \
  scell=acisf${ref_OBSID}_broad.cell \
  imagefile=acisf${ref_OBSID}_broad.recon \
  defnbkg=acisf${ref_OBSID}_broad.nbkg \
  interdir=./ mode=h clob+

cd ${parent_dir}

# Get ObsIDs (Note the extra parentheses, this makes our
 ↪  output a bash array)
OBSIDS=($(awk '/ACIS-I/ {print $1}' chandra_obsids))

# Iterate over the array and operate on the non-reference
 ↪  ObsIDs only
for OBSID in "${OBSIDS[@]}"; do
   if [[ "$OBSID" != "$REF_OBSID" ]]; then
      echo ${OBSID}
      cd ${OBSID}/working/base

      # Create counts image and exposure map in the same
       ↪  way as above. Also generate a psf map using the
       ↪  [psfecf=]
      # parameter. This parameter in fluximage uses the
       ↪  mkpsfmap tool, and the value used should be the
       ↪  same effective
      # energy used when creating the exposure maps
       ↪  (described in the bands parameter)
```

```
fluximage ../../repro/acisf${OBSID}_repr⌋
 ↪  o_evt2.fits acisf${OBSID} bin=1
 ↪  band=broad psfecf=0.9 clobber=yes

# Run wavdetect (again, same as above,
 ↪  with the inputs being the file
 ↪  outputs from fluximage)
punlearn wavdetect
wavdetect \
infile=acisf${OBSID}_broad_thresh.img \
psffile=acisf${OBSID}_broad_thresh.psfma⌋
 ↪  p
 ↪  \
expfile=acisf${OBSID}_broad_thresh.expma⌋
 ↪  p
 ↪  \
scales="1 2 4 6 8 12 16 24 32" \
outfile=acisf${OBSID}_broad.src \
scell=acisf${OBSID}_broad.cell \
imagefile=acisf${OBSID}_broad.recon \
defnbkg=acisf${OBSID}_broad.nbkg \
interdir=./ mode=h clob+

# Generate two source list, running the
 ↪  wcs_match tool as detailed above
wcs_match infile=acisf${OBSID}_broad.src
 ↪  \
```

```
refsrcfile=${parent_dir}/${ref_OBSID}/wo⌋
 ↪  rking/base/acisf${ref_OBSID}_broad.s⌋
 ↪  rc
 ↪  \
outfile=acisf${OBSID}_broad_corrected.fi⌋
 ↪  ts
 ↪  \
wcsfile=acisf${OBSID}_broad_thresh.img \
method=trans

# Use the outfile from wcs_match to
 ↪  update the aspect solution file for
 ↪  each ObsID
wcs_update
 ↪  infile=${parent_dir}/${OBSID}/repro/⌋
 ↪  pcadf${OBSID}_000N001_asol1.fits
 ↪  \
outfile=pcadf${OBSID}_corrected_asol1.fi⌋
 ↪  ts
 ↪  \
transformfile=acisf${OBSID}_broad_correc⌋
 ↪  ted.fits
 ↪  \
wcsfile=acisf${OBSID}_broad_thresh.img

# Copy and rename the reprocessed evt2
 ↪  file to be updated
```

```
                  dmcopy ${parent_dir}/${OBSID}/repro/acisf${OBSID}_repro_evt2.fits acisf${OBSID}_corrected_evt2.fits

                  # Run the wcs_update tool on the copied evt2 file, with the outfile having the same file name
                  pset wcs_update infile=acisf${OBSID}_corrected_evt2.fits
                  pset wcs_update outfile=
                  wcs_update

                  cd ${parent_dir}

    fi

done
```

A.2.5 Deflare

```
#! /bin/bash

# This script removes lightcurves (flares, agn, other sources contaminating our image) to generate a good-time-interval
# (GTI) file that we can use to clean our level=2 event file for each ObsID

# Move into the directory where the data is stored
parent_dir='/home/jhuber/Abell_2219/data'
```

```
cd ${parent_dir}

# Repeat the step from download_data to define that we only want to process data taken with the ACIS-I instrument
# (ignoring ACIS-S) and iterate over this list. See that script for more details.
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

# Define the directory (/deflare) in which we wish to store the data outputs from this script
# We want to explicitly define that we are using the WCS-corrected evt2 file output from the update_wcs script
for OBSID in $A2219_OBSIDS; do
                cd ${OBSID}/working/deflare
          evt2_file=$(ls ../base/acisf${OBSID}_corrected_evt2.fits)

    # Copy our WCS corrected evt2 file, and filter it over a preferred energy range
    dmcopy "${evt2_file}[energy=500:9000]" acisf${OBSID}_evt2_cor_filtered.fits clobber=yes

                # Use the fluximage command to "Create exposure-corrected images and exposure maps for an observation"
                # Based on testing various solutions, it appears that you have to manually specify the asol file
```

```
# being used
fluximage
 ↪  acisf${OBSID}_evt2_cor_filtered.fits
 ↪  binsize=1 bands=broad
 ↪  outroot=acisf${OBSID} \
asolfile=../base/pcadf${OBSID}_corrected_a⌟
 ↪  sol1.fits
 ↪  \
badpixfile=${parent_dir}/${OBSID}/repro/ac⌟
 ↪  isf${OBSID}_repro_bpix1.fits
 ↪  \
maskfile=$(echo ${parent_dir}/${OBSID}/rep⌟
 ↪  ro/acisf${OBSID}_*_msk1.fits)
 ↪  \
psfecf=0.393 clobber=yes

# Use the wavdetect command
wavdetect
 ↪  infile=acisf${OBSID}_broad_thresh.img
 ↪  psffile=acisf${OBSID}_broad_thresh.psf⌟
 ↪  map outfile=. scellfile=.
 ↪  \
imagefile=. defnbkgfile=. regfile=.
 ↪  scales="1.0 2.0 3.0 4.0 8.0 16.0"
 ↪  sigthresh=4e-06 clobber=yes

# Create another copy of the image, using
 ↪  the above generated region file to mask
 ↪  out point sources
```

```
dmcopy "acisf${OBSID}_evt2_cor_filtered.fits[exclude sky=region(acisf${OBSID}_broad_reg.img)]" \
acisf${OBSID}_evt2_cor_filtered_bg.fits clobber=yes

# Get the lightcurve using the dmextract command
dmextract infile="acisf${OBSID}_evt2_cor_filtered_bg.fits[bin time=::10]" \
outfile=acisf${OBSID}_evt2_cor_filtered_bg_lc.fits opt=ltc1 clobber=yes

# The deflare routine is a Python script which allows the user to run lc_sigma_clip() on an input light curve file
# for filtering flares
deflare acisf${OBSID}_evt2_cor_filtered_bg_lc.fits method=sigma \
outfile=acisf${OBSID}_evt2_cor_filtered_bg_lc_clip.gti plot=no save=acisf${OBSID}lcsc_plot

```

```
                # dmcopy on the evt2 file using the gti
                 ↪  output from deflare
                # This output file (per ObsID) will be used
                 ↪  repeatedly throughout the rest of the
                 ↪  reduction and subsequent analysis
                dmcopy "acisf${OBSID}_evt2_cor_filtered.fi⌋
                 ↪  ts[@acisf${OBSID}_evt2_cor_filtered_bg⌋
                 ↪  _lc_clip.gti]"
                 ↪  \
                acisf${OBSID}_evt2_deflare.fits clobber=yes

                cd ${parent_dir}

done

```

A.2.6 ACIS Backgrounds

```
#! /bin/bash

# Create backgrounds for each ObsID to be used during the
 ↪  mosaic process. The blanksky command tailors an
 ↪  unscaled
# background file to be compatible with our level=2 event
 ↪  files through combinations and reprojections from
 ↪  background
# files in CalDB

# Move into the directory where the data is stored
parent_dir='/home/jhuber/Abell_2219/data'
```

```
cd ${parent_dir}

# Repeat the step from download_data to define that we only
 ↪  want to process data taken with the ACIS-I instrument
# (ignoring ACIS-S) and iterate over this list. See that
 ↪  script for more details.
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

for OBSID in $A2219_OBSIDS; do
      echo ${OBSID}
      cd ${OBSID}/working/backgrounds

# We are using the broad_thresh.img file from the
 ↪  lightcurves script to be called in the next step
# The blanksky script automatically locates the correct
 ↪  blanksky background files, and reprojects them to match
 ↪  the
# observation
      pset blanksky
       ↪  evtfile="${parent_dir}/${OBSID}/working/base/aci⌋
       ↪  sf${OBSID}_corrected_evt2.fits[@../deflare/acisf⌋
       ↪  ${OBSID}_evt2_cor_filtered_bg_lc_clip.gti]"
      pset blanksky outfile=acisf${OBSID}_blank.evt
      pset blanksky asolfile=${parent_dir}/${OBSID}/workin⌋
       ↪  g/base/pcadf${OBSID}_corrected_asol1.fits
      pset blanksky tmpdir=./
      blanksky mode=h clobber=yes

            cd ${parent_dir}
```

```

done
```

A.2.7 Mosaic ObsIDs

```
#! /bin/bash

# Use the merge_obs CIAO tool to reproject and combine our ObsIDs to created a merged event file and exposure-corrected
# images for a variety of energy bands. This is our final data piece prior to running the analysis pipeline.

# Move into the directory where the data is stored
parent_dir='/home/jhuber/Abell_2219/data'
cd ${parent_dir}

# Repeat the step from download_data to define that we only want to process data taken with the ACIS-I instrument
# (ignoring ACIS-S) and iterate over this list. See that script for more details.
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

# Run merge_obs:
# Inputs: deflared evt files for each obsid, bkg evt file for each obsid
# ex2 on ahelp page: "5826/repro/acisf05826_repro_evt2.fits,5827/repro/acisf05827_repro_evt2.fits m87"
```

```
# Outputs: data evt file (all obsids), bkg evt file (all obsids)

# Run merge_obs first with bands=[energy], input=acisf${OBSID}_evt2_deflare.fits
merge_obs \
13988/working/deflare/acisf13988_evt2_deflare.fits,\
14355/working/deflare/acisf14355_evt2_deflare.fits,\
14356/working/deflare/acisf14356_evt2_deflare.fits,\
14431/working/deflare/acisf14431_evt2_deflare.fits,\
14451/working/deflare/acisf14451_evt2_deflare.fits,\
20588/working/deflare/acisf20588_evt2_deflare.fits,\
20589/working/deflare/acisf20589_evt2_deflare.fits,\
20785/working/deflare/acisf20785_evt2_deflare.fits,\
20951/working/deflare/acisf20951_evt2_deflare.fits,\
20952/working/deflare/acisf20952_evt2_deflare.fits,\
21966/working/deflare/acisf21966_evt2_deflare.fits,\
21967/working/deflare/acisf21967_evt2_deflare.fits,\
21968/working/deflare/acisf21968_evt2_deflare.fits \
outroot=mosaic/ \
asolfiles=13988/working/base/pcadf13988_corrected_asol1.fits,\
14355/working/base/pcadf14355_corrected_asol1.fits,\
14356/working/base/pcadf14356_corrected_asol1.fits,\
14431/working/base/pcadf14431_corrected_asol1.fits,\
14451/working/base/pcadf14451_corrected_asol1.fits,\
20588/working/base/pcadf20588_corrected_asol1.fits,\
20589/working/base/pcadf20589_corrected_asol1.fits,\
20785/working/base/pcadf20785_corrected_asol1.fits,\
```

```
20951/working/base/pcadf20951_corrected_asol1.fits,\
20952/working/base/pcadf20952_corrected_asol1.fits,\
21966/working/base/pcadf21966_corrected_asol1.fits,\
21967/working/base/pcadf21967_corrected_asol1.fits,\
21968/working/base/pcadf21968_corrected_asol1.fits \
bands=csc,broad,0.6:1.2:0.92 psfecf=0.9 clobber=yes

for OBSID in $A2219_OBSIDS; do
    cd ${parent_dir}/mosaic

    # Create a properly scaled blank sky image with matching WCS, dimensions, and using the same energies as
    # are used in the input image file. The script also creates a background subtracted image.
    pset blanksky_image bkgfile=../${OBSID}/working/backgrounds/acisf${OBSID}_blank.evt
    pset blanksky_image outroot=${OBSID}_blank
    pset blanksky_image imgfile=${OBSID}_broad_thresh.img
    pset blanksky_image tmpdir=./
    blanksky_image mode=h clob+

    cd ${parent_dir}

done

  # Run merge_obs again, this time on the reprojected background evt2 files from make_bkg_event script
  # (/backgrounds/acisf${OBSID}_blank.evt)
```

```
merge_obs \
13988/working/backgrounds/acisf13988_blank.evt,\
14355/working/backgrounds/acisf14355_blank.evt,\
14356/working/backgrounds/acisf14356_blank.evt,\
14431/working/backgrounds/acisf14431_blank.evt,\
14451/working/backgrounds/acisf14451_blank.evt,\
20588/working/backgrounds/acisf20588_blank.evt,\
20589/working/backgrounds/acisf20589_blank.evt,\
20785/working/backgrounds/acisf20785_blank.evt,\
20951/working/backgrounds/acisf20951_blank.evt,\
20952/working/backgrounds/acisf20952_blank.evt,\
21966/working/backgrounds/acisf21966_blank.evt,\
21967/working/backgrounds/acisf21967_blank.evt,\
21968/working/backgrounds/acisf21968_blank.evt \
outroot=mosaic_bg/ bands=broad,0.6:1.2:0.92
```

A.2.8 CCD and Region Files Wrapper

```
#! /bin/bash

# This script identifies the CCDs that were on for each
↪  observation, as well as the primary CCD, through the
# find_acis_ccds script. This information is printed to a
↪  simple text file. Region files are then generated for
↪  each
# CCD, using the ccd_regions script. These regions are then
↪  transformed to WCS coordinates to match later data used
↪  in
# the analysis

```

```
parent_dir='/home/jhuber/Abell_2219/data'
code_dir='/home/jhuber/Abell_2219/code'

cd ${parent_dir}
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)

# Run the find_acis_ccds.sh script
for OBSID in $A2219_OBSIDS; do

     cd ${parent_dir}/${OBSID}/working/ccds
     rm *
     cp "${code_dir}/find_acis_ccds.sh" .
     chmod +x find_acis_ccds.sh
     ./find_acis_ccds.sh ${parent_dir}/${OBSID}/working/de⌋
      ↪  flare/acisf${OBSID}_evt2_deflare.fits

 done < chandra_obsids

cd ${parent_dir}

# Note, region output (e.g. chip_s.reg) is in CIAO
 ↪  coordinates, NOT WCS
for OBSID in $A2219_OBSIDS; do

     cd ${parent_dir}/${OBSID}/working/regions
     rm *
     cp "${code_dir}/ccd_regions.sh" .
     chmod +x ccd_regions.sh
     cp "${code_dir}/change_coords_to_ciao_wcs.sh" .
```

```
    chmod +x change_coords_to_ciao_wcs.sh

    evt2file=${parent_dir}/${OBSID}/working/deflare/acisf⌋
     ↪  ${OBSID}_evt2_deflare.fits
    asolfile=${parent_dir}/${OBSID}/working/base/pcadf${O⌋
     ↪  BSID}_corrected_asol1.fits

#     echo "awk '{print \"./ccd_regions.sh\",
 ↪  \"$evt2file\", \"$asolfile\", \$1}'
 ↪  ../ccds/evt2_clean.fits_ccds_all" | sh
    echo "awk '{print \"./ccd_regions.sh\", \"$evt2file\",
     ↪  \"$asolfile\", \$1}'
     ↪  ../ccds/evt2_clean.fits_ccds_all | sh" | sh

    # Change the coords from CIAO to WCS to match our
     ↪  contbin regions created in the analysis pipeline
    ./change_coords_to_ciao_wcs.sh ${parent_dir}/${OBSID}⌋
     ↪  /working/deflare/acisf${OBSID}_evt2_deflare.fits
     ↪  .

done
```

A.2.9 CCD Identification

```
#!/bin/bash

# Find the ACIS CCDs switched on and the one that is
 ↪  priority, save to text files per ObsID
# ./find_acis_ccds
# remember substr(string,start index,length)
```

```

file=$1

# all
ccd_id_1=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,1,1)}')
ccd_id_2=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,2,1)}')
ccd_id_3=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,3,1)}')
ccd_id_4=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,4,1)}')
ccd_id_5=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,5,1)}')
ccd_id_6=$(dmlist $file opt=header | grep DETNAM | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,6,1)}')

# priority
#ccd_pri=$(dmlist $file opt=header | grep CCD_ID | awk
 ↪  '{print $3}')
#ccd_pri=$(dmlist $file opt=header | grep FEP_CCD | awk
 ↪  '{print $3}' | sed 's/ACIS-//g' | awk 'NR==1{print
 ↪  substr($0,2,1)}')
```

```
ccd_pri=3

echo $ccd_id_1

# if 0 then 3 if no 0 then 7
#if [ $ccd_id_1 -lt 1 ]; then ccd_pri=3
#if [ $ccd_id_1 -gt 1 ]; then ccd_pri=7

echo $ccd_pri

echo "CCDS: " $ccd_id_1 $ccd_id_2 $ccd_id_3 $ccd_id_4
 ↪  $ccd_id_5 $ccd_id_6 " PRIORITY: " $ccd_pri

rm evt2_clean.fits_ccds_all
rm evt2_clean.fits_ccds_pri
echo $ccd_id_1 >> evt2_clean.fits_ccds_all
echo $ccd_id_2 >> evt2_clean.fits_ccds_all
echo $ccd_id_3 >> evt2_clean.fits_ccds_all
echo $ccd_id_4 >> evt2_clean.fits_ccds_all
echo $ccd_id_5 >> evt2_clean.fits_ccds_all
echo $ccd_id_6 >> evt2_clean.fits_ccds_all
echo $ccd_pri >> evt2_clean.fits_ccds_pri
```

A.2.10 CCD Region Files

```
#!/bin/bash

# Create region files for the ccds

# Call as:
```

```
# ./ccd_regions.sh evt2file asolfile ccd_id

evt2file=$1
asolfile=$2
ccdid=$3

roll=$(dmkeypar $evt2file ROLL_NOM echo+)

echo "roll is: $roll"

echo "dmcoords $evt2file asolfile=$asolfile \
      chip_id=$ccdid opt=chip chipx=512.5 chipy=512.5 \
      verbose=1 | grep \"SKY(\" | awk 'BEGIN {printf(\"#
       ↪  Region file format: CIAO version 1.0\nrotbox(\")}
       ↪  \
      {printf(\"%f,%f,1024,1024,%f)\n\", \$2, \$3,
       ↪  360-'$roll')}' > chip_s${ccdid}.reg"  | sh

# For testing/debugging
#echo "dmcoords $evt2file asolfile=$asolfile \
#      chip_id=$ccdid opt=chip chipx=512.5 chipy=512.5 \
#      verbose=1 | grep \"SKY(\" | awk 'BEGIN {printf(\"#
 ↪  Region file format: CIAO version 1.0\nrotbox(\")} \
#      {printf(\"%f,%f,1024,1024,%f)\n\", \$2, \$3,
 ↪  360-'$roll' }' > chip_s${ccdid}.reg"  | sh

```

A.2.11 Regions to WCS

```
#!/bin/bash

# A simple script to change physical coordinates to WCS
 ↪  using DS9

# To run:
# [script] [binmap] [list_of_regions]
# ./change_coords_to_ciao_wcs.sh
 ↪  ../../CONTBIN/SN00_SM00/contbin_binmap.fits /outreg

bin_map=$1
outreg=$2

ds9 ${bin_map} &
sleep 3
#xpaset -p ds9 lower

for region in ${outreg}/*.reg; do
  echo ${region}

    xpaset -p ds9 region load ${region}
    xpaset -p ds9 region format ciao
    xpaset -p ds9 region system wcs
    xpaset -p ds9 region save ${region}
    xpaset -p ds9 region delete all

done

```

```
xpaset -p ds9 exit
```

APPENDIX B

ABELL 2219 ANALYSIS PIPELINE

B.1 Description of Use

See Chapter 3 for an overview of the analysis pipeline and the code contained therein. Appendix B includes all scripts required to analyze a dataset, given that all proper data reduction steps (as detailed in Appendix A) have been ran. A wrapper script (`analysis_init`) is used to automate the analysis pipeline, but each script can be tailored to the desired output for other analysis.

B.2 Source Code

The beginning of each script contains a brief introduction/explanation in plain text of that script and its purpose.

B.2.1 Analysis Pipeline

```
#! /bin/bash

# Wrapper for the data analysis scripts

# Define our directories structure
obsids_dir="/home/jhuber/Abell_2219/data"
code_dir="/home/jhuber/Abell_2219/code"

# Run wavdetect and dmfilth to mask point sources
```

```
./dmfilth_ptsources

# Run contbin tool
./run_contbin

# Make spectra
./make_spec

# Fit spectra
./fit_spec

# Create thermodynamic maps
./turn_spec_to_map.ipynb
```

B.2.2 Detect & Remove Point Sources

```
#! /bin/bash

# Using the CIAO tools wavdetect and dmfilth, we detect point sources in the mosiacked image and then infill them with
# data matching the local background around the point source

# Run script, and then check the result visually to see if the removed point sources make sense
# If there are some that don't (likely centrally near the BCGs) manually remove them from the region file and
# run the script again with new, edited region file to get a final version
```

```
# If multiple energy bands were generated during the mosaic step (e.g. broad, soft, etc.), this process would need to
# be repeated for all of bands

parent_dir="/home/jhuber/Abell_2219/data/mosaic"

# Run wavdetect to locate point sources in our mosaicked broad-band image, generating a region file that is a list of
# those sources
wavdetect infile=${parent_dir}/broad_thresh.img psffile=${parent_dir}/broad_thresh.psfmap outfile=. \
scellfile=. imagefile=. defnbkgfile=. regfile=${parent_dir}/wav_expweighted_mean.reg \
scales="1.0 2.0 3.0 4.0 8.0 16.0" sigthresh=4e-06 clobber=yes

# Create another copy of the image, using the above generated region file to mask out the point sources
dmcopy "acisf${OBSID}_evt2_cor_filtered.fits[exclude sky=region(acisf${OBSID}_broad_reg.img)]" \
acisf${OBSID}_evt2_cor_filtered_bg.fits clobber=yes

# Run dmfilth to remove the sources from the image and infill these sources with data from the local background
```

```
# using annuli (Note: you need to update input/output files
 ↪  for awk and dmfilth and run again as explained below)
awk '{print $0}' ${parent_dir}/wav_expweighted_mean.reg |
 ↪  sed 's/ellipse/annulus/g' > ${parent_dir}/tmp_1

awk -F "," '{print $1","$2","$3","$3*1.5")"}'
 ↪  ${parent_dir}/tmp_1 >
 ↪  ${parent_dir}/bkg_wav_expweighted_mean.reg
rm ${parent_dir}/tmp_1
punlearn dmfilth
# Visually inspect the result to ensure removed regions
 ↪  make sense (i.e. a point source in the BCG or
 ↪  surrounding area)
# Our iteration required removing 3 central regions from
 ↪  above region files
# BE SURE to rerun dmfilth each time you edit region files,
 ↪  edited files have "MOD" added to them
# (e.g wav_expweighted_mean_MOD.reg)
dmfilth ${parent_dir}/broad_flux.img
 ↪  ${parent_dir}/broad_flux_nopt.img DIST \
@${parent_dir}/wav_expweighted_mean.reg
 ↪  @${parent_dir}/bkg_wav_expweighted_mean.reg clobber=yes
```

B.2.3 Contbin

```
#! /bin/bash

# Run contbin software on the mosaic. Contbin is a contour
 ↪  binning and accumulative smoothing software. The most
 ↪  useful
```

```
# outputs will be the the binned image (contbin_out.fits)
 ↪  and the binmap (contbin_binmap.fits). The binmap is a
 ↪  FITS
# image where each pixel is numbered according to which bin
 ↪  it is in (counting from 0, NOT 1). The binmap can be
 ↪  used
# to generate region files (this step is handled in the
 ↪  make_spec script).
# See README documentation on the contbin github repo:
 ↪  https://github.com/jeremysanders/contbin)

parent_dir="/home/jhuber/Abell_2219/data/mosaic"

# Make the contbin directory and make a variety of contbin
 ↪  maps
cd ${parent_dir}
mkdir CONTBIN
cd CONTBIN

# You can use DS9 to verify the counts in the image (if
 ↪  desired), typically looking for 2000, 3000, 5000, and
# 10,000 counts
infile="${parent_dir}/broad_thresh.img" # use be
 ↪  broad_thresh.img from the mosaic
cp $infile .
expmap="${parent_dir}/broad_thresh.expmap" # use the
 ↪  broad_thresh.expmap from the mosaic
cp $expmap .

```

```
### Currently need to initialize the heasoft download each
 ↪  time before running farith
### export HEADAS=/usr/local/heasoft/PLATFORM
### source $HEADAS/headas-init.sh

# Create a mask file, removing point sources from
 ↪  wcs_update wavdetect input/output on our mosaic
# Effectively, we are saying there are 1's everywhere
 ↪  EXCEPT where wavdetect finds AGN
farith $infile 0 tmp.fits MUL
farith tmp.fits 1 allones.fits ADD
rm tmp.fits

# Point sources from wavdetect - use the "MOD" versions as
 ↪  detailed in dmfilth_ptsources
cp ../wav_expweighted_mean_MOD.reg .
# This "MOD2" file is simply a circular region encompassing
 ↪  the merger, primarily to make contbin to run more
 ↪  quickly
# This region was drawn manually as a circular region in
 ↪  DS9 over the binmap and saved as 'MOD2.reg'
cp ../MOD2.reg .
dmcopy "allones.fits[exclude
 ↪  sky=region(wav_expweighted_mean_MOD.reg)][opt full]"
 ↪  mask_tmp.fits clobber=yes
dmcopy "mask_tmp.fits[sky=region(MOD2.reg)][opt full]"
 ↪  mask.fits clobber=yes
mask='mask.fits'

```

```
# backgrounds - save for later once make_mosaic is fixed
#bkimage='B_bgimg1_0.6-7.0.fits.gz.fits'
#cp ../ALL_IMAGE/$bkimage .
#bgexpmap='B_Sbkgd_expmap1_3.75kev.fits.gz.fits'
#cp ../ALL_IMAGE/$bgexpmap .

# Run the contbin tool, creating a new directory with name matching the parameters for that run
mkdir SN30_SM15
cd SN30_SM15
contbin ${parent_dir}/broad_thresh.img --expmap=${parent_dir}/broad_thresh.expmap --sn=30 --mask=../$mask \
--constrainfill --constrainval=1.8 --smoothsn=15

# Other run example with changed parameters, repeat as necessary when trying new parameters
mkdir SN60_SM15
cd SN60_SM15
contbin ${parent_dir}/broad_thresh.img --expmap=${parent_dir}/broad_thresh.expmap --sn=60 --mask=../$mask \
--constrainfill --constrainval=1.8 --smoothsn=15
```

B.2.4 Generate Spectra

```
#! /bin/bash

```

```
# Using the contbin software, generate region files
 ↪  matching each bin from the binmap generated in the
 ↪  previous
# script. Once generated, you need to determine which CCDs
 ↪  each region lies on (as each CCD has a different
 ↪  response
# profile and background data). If a region covers more
 ↪  than one CCD, we split the region and save new regions
# (e.g xaf_1.reg becomes xaf_1_0.reg and xaf_1_3.reg). This
 ↪  step is performed by the region_split script. Next,
 ↪  using
# the extract_spec script, we extract spectra for each
 ↪  region PER ObsID.

parent_dir="/home/jhuber/Abell_2219/data"
code_dir="/home/jhuber/Abell_2219/code/"

# Dirname is name of binning dir so can have same name e.g.
 ↪  SN100_SM30
# NOTE: This currently needs to be changed manually each
 ↪  time a new dataset is ran
dirname=SN30_SM15

# minx and miny are the minimum points (bottom left corner
 ↪  of the image) of the contbin_binmap.fits
minx=1441
miny=1608

### STEP ONE ###
```

```
# Make region files to match the contbin_binmap regions, and convert the regions to WCS coordinates
cd ${parent_dir}

mkdir SPEC
cd SPEC

mkdir ${dirname}
cd ${dirname}

mkdir outreg

# Generate the regions using contbin. Make sure your binning matches the binning in your images (in our case it is 8)
make_region_files --bin=8 --minx=${minx} --miny=${miny} --outdir=outreg \
${parent_dir}/mosaic/CONTBIN/${dirname}/contbin_binmap.fits

# This is a simple script used to transform the binmap regions from physical to WCS coordinates
cd outreg
cp ${code_dir}/change_coords_to_ciao_wcs.sh .
chmod +x change_coords_to_ciao_wcs.sh
./change_coords_to_ciao_wcs.sh ${parent_dir}/mosaic/CONTBIN/${dirname}/contbin_binmap.fits .

### STEP TWO ###
```

```
# Make the individual ObsID directories and extract the
 ↪  spectra
cd ${parent_dir}
A2219_OBSIDS=$(awk '/ACIS-I/ {print $1}' chandra_obsids)
specdir=${parent_dir}/SPEC/${dirname}

for OBSID in $A2219_OBSIDS; do
    cd ${specdir}
    mkdir -p ${OBSID}
    cd ${OBSID}

# Loop over the regions and extract the spectra
    region_path="${specdir}/outreg"
    shopt -s nullglob
    ls "${region_path}"/*.reg

    for region in "${region_path}"/*.reg; do
        # find if the regions need to be split (i.e. if
         ↪  they are covering multiple CCDs)
        evt2file=${parent_dir}/${OBSID}/working/deflare/ac⌟
         ↪  isf${OBSID}_evt2_deflare.fits
        # This is the reprocessed, GTI, deflared evt2 file
         ↪  in /deflare
        bckscaledir=${parent_dir}/${OBSID}/working/backgro⌟
         ↪  unds
        cp ${code_dir}/region_split.sh .
        chmod +x region_split.sh
        ./region_split.sh "${region}" "${evt2file}"
         ↪  "${parent_dir}/${OBSID}/working/ccds"
```

```
        # NOTE: this makes ccd_tmp (used by extract_spec)
         ↪  in the directory where we will do the
         ↪  extraction
        cp ${code_dir}/extract_spec.sh .
        chmod +x extract_spec.sh
        echo ${region} > region_name.tmp
        region_name=$(awk '{split($0,array,"/"); print
         ↪  array[9]}' region_name.tmp)
        echo ${region_name}
        # extract spectra per region, per ObsID
        ./extract_spec.sh ${evt2file} ../outreg/
         ↪  ${region_name} ${bckscaledir} ${OBSID} yes
    done

done
```

B.2.5 Split Regions by CCD

```
#!/bin/bash

# Determine if a region file is covering two or more chips
 ↪  and if so split region file
# Region file should be in WCS units

# run as:
# [script] [region] [evt2_file] [ccd_list]
# ./region_split.sh region.reg evt2file
 ↪  ../$obsid/work_ci45_ca459/ccds/

```

```
# list ccds on which region file is based in the file
 ↪  ccd_tmp
#
# e.g. ccd_tmp is [minccd=ccdid?] [area]
#
# 0
# 1
# 2
# 3
#

parent_dir="/home/jhuber/Abell_2219/data"
code_dir="/home/jhuber/Abell_2219/code/"

inputreg="$1"
evt2="$2"
ccd_dir="$3"

#rm reg.tmp
cp ${code_dir}/get_region_area.sh .
chmod +x get_region_area.sh

# if dmstat [cols ccd_id] min and max are not the same
# Debugging
echo "inputreg: $inputreg"
echo "dmstat string: ${evt2}[sky=region(${inputreg})][cols
 ↪  ccd_id]"

```

```
min=$(dmstat "${evt2}[sky=region(${inputreg})][cols ccd_id]" | grep min | awk '{print $2}')
max=$(dmstat "${evt2}[sky=region(${inputreg})][cols ccd_id]" | grep max | awk '{print $2}')
good=$(dmstat "${evt2}[sky=region(${inputreg})][cols ccd_id]" | grep good | awk '{print $2}')
echo $min $max
if [ "$min" == "$max" ]; then
  rm ccd_tmp
  if [ "$good" != "0" ]; then
    echo "Only one region"
    regions=0
      # put the number of pixels into the file too
      ./get_region_area.sh ${evt2} ${inputreg}
      area=$(awk '{print $0}' area.tmp)
    echo ${min} ${area} > ccd_tmp
  else
    echo "Region not on any chip"
  fi
else
  echo "More than one region"
  rm ccd_tmp
  regions=1
  # if more than one region find which ccds to use
  while read id
    do echo ${id}
      minccd=$(dmstat "${evt2}[ccd_id=${id},sky=region(${inputreg})][cols ccd_id]" | grep min | awk '{print $2}')
```

```
      maxccd=$(dmstat "${evt2}[ccd_id=${id},sky=region($⌋
       ↪  {inputreg})][cols ccd_id]" | grep max | awk
       ↪  '{print $2}')
      goodccd=$(dmstat "${evt2}[ccd_id=${id},sky=region(⌋
       ↪  ${inputreg})][cols ccd_id]" | grep good | awk
       ↪  '{print $2}')

      # create the new region - this line may differ
      chip_region=$(awk '{if (NR==2) print "*"$0}'
       ↪  ${ccd_dir}/../regions/chip_s${id}.reg)
      echo "awk '{print \$0\"${chip_region}\"}'
       ↪  ${inputreg} > reg.tmp " | sh
           ./get_region_area.sh ${evt2} reg.tmp
      area=$(awk '{print $0}' area.tmp)
      if [ "${minccd}" != "${maxccd}" ]; then echo
       ↪  "Problem with chip regions"
      else
        if [ "${goodccd}" != "0" ]; then
          echo "Chip ${id} in region"
          echo ${minccd} ${area} >> ccd_tmp
        else
          echo "Chip ${id} not in region"
        fi
      fi
  done < ${ccd_dir}/evt2_clean.fits_ccds_all
fi


```

B.2.6 Get Region Area

```bash
#!/bin/bash

# Simple script that gets the area of a region for a given region file.

# Run as:
# [script] [evt2] [outfile] [regionfile]
# ./get_region_area.sh ../filter/evt2_clean.fits ../outreg_old/ region_file.reg

evt2_file=$1
region=$2

rm area.tmp

area=$(dmlist "${evt2_file}[sky=region(${region})]" subspace | grep area | awk '{if (NR==1) print $11}')

echo "Region using: " ${region}
echo "Area: " ${area}

echo "${area}" > area.tmp

```

B.2.7 Extract Spectra

```bash
#!/usr/bin/env bash

```

```
# Extract spectra per input region file

# ./extract_spec.sh $evt2file ../outreg/ $region $bckscaledir yes

# NOTES
#
# Will turn a region file of xaf_0.pi into xaf_0_0.pi if on chip 0
# I.e. if on two chips such as 0 and 2 will write
# xaf_0_0.reg xaf_0_2.reg with correct backgrounds for each so they can be fit simultaneously
#

evt2name=$1
reg_dir=$2
reg=$3
bckscaledir=$4
OBSID=$5
clobberval=$6

rm tmp tmp2 tmp3
# do we need to split reg into two extracted spectra?
while read ccd_line # for ccd_line in ccd_temp; do
  do echo ${ccd_line} > tmp
    ccd=$(awk '{print $1}' tmp)
    # specific area
    sp_area=$(awk '{print $2}' tmp)
    # total area
```

```
total_area=$(awk '{SUM+=$2} END {print SUM}' ccd_tmp)
echo "specific area is: ${sp_area} total area is: ${total_area}"
echo ${sp_area} ${total_area} > tmp2
areascal2=$(awk '{printf "%4.3f", $1/$2}' tmp2)
echo "AREASCAL keyword should be: ${areascal2}"

# BACKGROUND STUFF
# Will need bkg_evt_file dmkeypar bkgscal{ccd_id}, store in value (instead of grep?)
bckgrdevt2name=${bckscaledir}/acisf${OBSID}_blank.evt
echo "Background file is: " ${bckgrdevt2name}
dmkeypar ${bckgrdevt2name} BKGSCAL${ccd} echo=yes > bck_tmp
corrfac=$(awk '{printf "%4.8f", 1.0/$1}' bck_tmp)
echo "Correction factor for ccd ${ccd} is: ${corrfac} * ${areascal2}"
echo ${corrfac} ${areascal2} > tmp3
corrfac2=$(awk '{print $1*$2}' tmp3)
echo ${corrfac2}
# extract background
# input fits file and region
# output .pi file
echo ${reg} ${ccd}
echo ${reg/.reg/_${ccd}.bckgd}
bkgrndspec=${reg/.reg/_${ccd}.bckgd}
echo bkgrndspec
punlearn dmextract
# Testing line
```

```
#    dmextract
 ↪  $bckgrdevt2name"[ccd_id=${ccd},sky=region(${reg})][bin
 ↪  pi]" ${bkgrndspec} error=gaussian clobber=$clobberval
    # Actual line
    dmextract
     ↪  $bckgrdevt2name"[sky=region(${reg_dir}${reg})][bin
     ↪  pi]" ${bkgrndspec} error=gaussian
     ↪  clobber=${clobberval}
    # this output is the extracted spectrum of the
     ↪  background
    dmhedit ${bkgrndspec} filelist=none operation=add
     ↪  key=AREASCAL value=${corrfac2}

    # SOURCE STUFF
    # extract source
    # input fits file and region
    # output .pi file and quick wmap
    outspecpi=${reg/.reg/_${ccd}.pi}
    punlearn dmextract
      dmextract "${evt2name}[ccd_id=${ccd},sky=region(${re⌋
       ↪  g_dir}${reg})][bin pi]" ${outspecpi}
       ↪  wmap="[energy=600:7000][bin det=8]"
       ↪  error=gaussian clobber=$clobberval
      dmhedit ${outspecpi} filelist=none operation=add
       ↪  key=AREASCAL value=${areascal2}

    # make source rmf file
    # input CALDB
    # output .rmf file
```

```
# good wmap not usually needed - just use dmextract one
 ↪  (becky note)
# commented parameters are used only if no WMAP is used
outspecrmf=${reg/.reg/_${ccd}.rmf}
punlearn mkacisrmf
  mkacisrmf infile=CALDB outfile=${outspecrmf}
   ↪  energy=0.28:9.0:0.01 channel=1:1024:1
   ↪  chantype=PI wmap="${outspecpi}[WMAP]" gain=CALDB
   ↪  clobber=$clobberval #chipx=1 chipy=1
   ↪  ccd_id=$chip_val

# make source arf file
# input wmap (can use asphist and sky2tdet to get
 ↪  better weight map if needed)
# output .arf file
outspecarf=${reg/.reg/_${ccd}.arf}
outspecweight=${reg/.reg/_${ccd}.weight}
punlearn mkwarf
  mkwarf infile="${outspecpi}[WMAP]"
   ↪  outfile=${outspecarf} egridspec=0.28:9.0:0.01
   ↪  threshold=0 weightfile=${outspecweight}
   ↪  spectrumfile=""  clobber=$clobberval
   ↪  pbkfile="NONE" dafile="NONE"

# group the channels incase have few counts in channels
# input source.pi file
# output grouped .pi file
# this is ftool version can also do with dmgroup
```

```
    # chkey keyword new_value changes the value of the
     ↪  keywords within the PHA file
    # Do not group up the background spectrum, xspec will
     ↪  handle the area rescaling and data grouping using
     ↪  the header information in both spectral files.
     ↪  xspec also automatically bins up the response
     ↪  matrix to have the same number of channels as the
     ↪  spectrum.
    # this has made min group occupation to be 30 counts
    outspecgrp=${reg/.reg/_${ccd}_grp1.pi}
    punlearn grppha
      grppha infile="$outspecpi" outfile="$outspecgrp"
       ↪  chatter=0 comm="group min 1 & chkey BACKFILE
       ↪  $bkgrndspec & chkey RESPFILE $outspecrmf & chkey
       ↪  ANCRFILE $outspecarf & exit" clobber=$clobberval

    # modify header
    # input source.pi files
    # output source.dat files
    punlearn dmhedit
#     dmhedit ${reg/.reg/_${ccd}_grp1.pi} ${reg/.reg/.dat}
    pset dmhedit operation=add filelist="" mode=h
    dmhedit ${reg/.reg/_${ccd}.pi} key=BACKFILE
     ↪  value=${reg/.reg/_${ccd}.bckgd}
    dmhedit ${reg/.reg/_${ccd}.pi} key=RESPFILE
     ↪  value=${reg/.reg/_${ccd}.rmf}
    dmhedit ${reg/.reg/_${ccd}.pi} key=ANCRFILE
     ↪  value=${reg/.reg/_${ccd}.arf}

```

```
    dmhedit ${reg/.reg/_${ccd}_grp1.pi} key=BACKFILE
     ↪  value=${reg/.reg/_${ccd}.bckgd}
    dmhedit ${reg/.reg/_${ccd}_grp1.pi} key=RESPFILE
     ↪  value=${reg/.reg/_${ccd}.rmf}
    dmhedit ${reg/.reg/_${ccd}_grp1.pi} key=ANCRFILE
     ↪  value=${reg/.reg/_${ccd}.arf}

    dmhedit ${reg/.reg/_${ccd}.bckgd} operation=add
     ↪  key=RESPFILE value=${reg/.reg/_${ccd}.rmf}
    dmhedit ${reg/.reg/_${ccd}.bckgd} operation=add
     ↪  key=ANCRFILE value=${reg/.reg/_${ccd}.arf}

done < ccd_tmp
```

B.2.8 XSpec Spectra Fitting

```
#! /bin/bash

# Take the extracted spectra per contbin region and use
 ↪  XSpec to fit a model to each, giving us statistics per
 ↪  region
# (stacked for all ObsIDs) that can then be used to plot
 ↪  thermodynamic maps of the object

parent_dir="/home/jhuber/Abell_2219/data"
code_dir="/home/jhuber/Abell_2219/code/"

# dirname is name of binning dir so can have same name e.g.
 ↪  SN100_SM30
dirname=SN30_SM15
```

```

### fit all obsids together
# Make the region files for the contbin maps
  cd ${parent_dir}
  cd SPEC
  cd $dirname
#  pwd
cp ${code_dir}/fit_contbin_phabsapec.tcl .

# find how many xaf_0_*.reg files exist
no_of_regions=$(ls outreg/xaf_*.reg | wc | awk '{print
 ↪  $1-1}')

# loop through them and list which files to read in
for i in $(eval echo {0..${no_of_regions}});
# can also do 'for i in $(eval echo {0..$range..2}); do
 ↪  echo $i; done' if wish to increment
  do
    # list the data input command - have to go into file to
     ↪  find the right rmf/arf/bkgd
    rm import_for_fit import_data.tcl
    ls */xaf_${i}_*_grp1.pi > import_for_fit
    awk -F "/" '{print "cd", $1"; data 1:"NR, $2"; cd
     ↪  ../"}' import_for_fit > import_data.tcl
    # if data empty then delete import_data
    if [ -s import_data.tcl ]
    then
      cat import_data.tcl
      no_files=$(wc import_data.tcl | awk '{print $1}')
```

```
      echo $no_files
      # now run fit
xspec<<EOF
set no_files [expr $no_files]
@import_data.tcl
@fit_contbin_phabsapec.tcl
EOF
      mv results.txt results_${i}.txt
    else
#      rm  import_data.tcl
      echo "Te 0 0 0" >> results_${i}.txt
      echo "Ab 0 0 0" >> results_${i}.txt
      echo "No 0 0 0" >> results_${i}.txt
      echo "Stats 0 0 0" >> results_${i}.txt
    fi
    rm test.out
  done # loop over region files
```

B.2.9 XSpec Model & Parameters

```
# XSpec initialization file - this sets the model and
 ↪  parameters that XSpec will use to fit the spectra and
 ↪  extract
# statistics


### imputs needed
set z [expr 0.2256]
set n_h [expr 0.0176]

```

```
### general setup
cpd /xs
setplot energy

### specific setup
ignore bad
ignore 1-$no_files: **-0.6
ignore 1-$no_files: 7.0-**
abund angr
cosmo ,,0.7
statistic cstat

model phabs(apec) & /*

newpar 1 $n_h
newpar 2 10.0
newpar 3 0.3
newpar 4 $z
newpar 5 1.0E-4

freeze 1 4 3
thaw 2 5

query yes
show parameters
fit 1000

# fit with MC
chain length 1000
```

```
chain burn 200
chain run test.out

# bin at 5 sig or in sets of 5 bins
setplot rebin 10 10
pl ldata resid
show parameters

#2.706 is 90 percent conf
    tclout dof
    set deg [lindex $xspec_tclout 0]

    tclout stat
    set statparam [lindex $xspec_tclout 0]

    tclout param 2
    set Te [lindex $xspec_tclout 0]
    tclout param 3
    set Ab [lindex $xspec_tclout 0]
    tclout param 5
    set No [lindex $xspec_tclout 0]

    error maximum 1.0 2
    tclout error 2
    set eTm [lindex $xspec_tclout 0]
    set eTp [lindex $xspec_tclout 1]
    error maximum 1.0 3
    tclout error 3
    set eAm [lindex $xspec_tclout 0]
```

```
    set eAp [lindex $xspec_tclout 1]

    error maximum 1.0 5
    tclout error 5
    set eNm [lindex $xspec_tclout 0]
    set eNp [lindex $xspec_tclout 1]

    set resultsfile [open "results.txt" w]
    puts $resultsfile "Te $Te $eTm $eTp"
    puts $resultsfile "Ab $Ab $eAm $eAp"
    puts $resultsfile "No $No $eNm $eNp"
    puts $resultsfile "Stats $statparam $deg $deg"

    close $resultsfile

# /* is called the terminator - When a user inputs a line, the first few characters are checked for the EOF string, /*, which denotes the end of a read.
#
#For example:
#/*  ! This is a generated EOF
# /* ! This is a single argument input string "/*".

#set i [expr $i+1]

#}

#exit
```

B.2.10 Thermodynamic Mapping

```python
#%%
# import stuff
import numpy as np
import matplotlib.pyplot as plt
from astropy.io import fits
import glob
import pandas as pd
import os
from astropy.wcs import WCS
from astropy.nddata.utils import Cutout2D
from astropy import units as u
from astropy.coordinates import SkyCoord
from matplotlib.patches import Circle, Rectangle
#%%

joe = "/home/jhuber/Abell_2219/code"
filename = "/home/jhuber/Abell_2219/data/mosaic/CONTBIN/SN30_SM15/contbin_binmap.fits"
#%%
res_files=glob.glob("/home/jhuber/Abell_2219/data/SPEC/SN30_SM15/results*.txt")
#%%
res_files
#%%
hdul = fits.open(filename)
#%%
data = hdul[0].data
wcs = WCS(hdul[0].header)
```

```
#%%
data
#%%
#plt.imshow(data)
plt.subplot(projection=wcs)
plt.imshow(data, origin='lower')
plt.grid(color='white', ls='solid')
plt.xlabel('Galactic Longitude')
plt.ylabel('Galactic Latitude')
#%%
length_of_files=len(res_files)
print(length_of_files)
#%%
temp_map=np.copy(data)*0.0
abund_map=np.copy(data)*0.0
norm_map=np.copy(data)*0.0
stat_map=np.copy(data)*0.0

# Finding the data
for i in range(0,length_of_files, 1):

  # A results file
  results_file=f'/home/jhuber/Abell_2219/data/SPEC/SN30_SM15/results_{i}.txt'

  # Does the file exist
  if os.path.exists(results_file):
    df = pd.read_csv(results_file, delimiter=' ', header=None, names=['Name', 'best', 'min', 'max'])
```

```

    # Where is the data
    n=np.where(data==i)

    # Get values
    temp=df[df.Name=='Te']
    abund=df[df.Name=='Ab']
    norm=df[df.Name=='No']
    stat=df[df.Name=='Stats']

    best_temp=temp.best.values[0]
    best_abund=abund.best.values[0]
    best_norm=norm.best.values[0]
    best_stat=stat.best.values[0]

    temp_map[n]=best_temp
    abund_map[n]=best_abund
    norm_map[n]=best_norm
    stat_map[n]=best_stat

  else:
    print("File does not exist.")
#%%
#data_new = np.log10(norm_map)
data_new = temp_map

# Center RA/Dec and radius of circle in degreees
circle = (250.0768968, 46.6932881, 0.1738334)

```

```
# cut out a subregion of my image and update the WCS
 ↪  accordingly
center = SkyCoord(circle[0]*u.degree, circle[1]*u.degree)
cut = Cutout2D(data_new, center, 0.17*60*60/2, wcs=wcs)

#make figure and plot image
figure = plt.figure(figsize=(8,8))
ax = figure.add_subplot(1,1,1, projection=cut.wcs)
im = ax.imshow(cut.data,
              cmap=plt.cm.YlOrRd,
              origin='lower',
              interpolation='none',
              vmin=5,
              vmax=40)

#vmin==0.0, vmax==30,

# make the colour bar the right height, 42 is indeed the
 ↪  answer!
cb = plt.colorbar(im,fraction=0.042, pad=0.04)
#cb = pyplot.colorbar(im)
#cb.set_label("Flux Density (Jy/Beam)", size=12)

# add a circle
# circles are in sky coords!
#c = Circle(circle[:2],circle[2],
#           edgecolor='red',
#           transform=ax.get_transform('fk5'),
#           facecolor='none',
```

```
#            lw=3, ls='-.')
#ax.add_patch(c)

# fancy the plot
ax.grid()
ax.set_xlabel("RA J2000", size=14)
ax.set_ylabel("Dec J2000", size=14)
#ax.set_title("Field 8, 185MHz", size=14)
#%%
(0.1738334*u.deg).to(u.arcsec)
#%%
hdr = hdul[0].header
temp_hdu = fits.PrimaryHDU(data=temp_map, header=hdr)
temp_hdul = fits.HDUList([temp_hdu])
temp_hdul.writeto("temp.fits", overwrite=True)
```

REFERENCES

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33, doi: 10.1051/0004-6361/201322068

Astropy Collaboration, Price-Whelan, A. M., Sipőcz, B. M., et al. 2018, AJ, 156, 123, doi: 10.3847/1538-3881/aabc4f

Astropy Collaboration, Price-Whelan, A. M., Lim, P. L., et al. 2022, ApJ, 935, 167, doi: 10.3847/1538-4357/ac7c74

Brunetti, G., Setti, G., Feretti, L., & Giovannini, G. 2001, MNRAS, 320, 365, doi: 10.1046/j.1365-8711.2001.03978.x

Canning, R. E. A., Allen, S. W., Applegate, D. E., et al. 2017, MNRAS, 464, 2896, doi: 10.1093/mnras/stw2384

de Bernardis, P., Ade, P. A. R., Bock, J. J., et al. 2000, Nature, 404, 955, doi: 10.1038/35010035

Efstathiou, G., Kaiser, N., Saunders, W., et al. 1990, MNRAS, 247, 10P

Ensslin, T. A., Biermann, P. L., Klein, U., & Kohle, S. 1998, A&A, 332, 395, doi: 10.48550/arXiv.astro-ph/9712293

Fruscione, A., McDowell, J. C., Allen, G. E., et al. 2006, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 6270, Observatory Operations: Strategies, Processes, and Systems, ed. D. R. Silva & R. E. Doxsey, 62701V, doi: 10.1117/12.671760

Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Nature, 585, 357, doi: 10.1038/s41586-020-2649-2

Kravtsov, A. V., & Borgani, S. 2012, ARA&A, 50, 353, doi: 10.1146/annurev-astro-081811-125502

Markevitch, M., Gonzalez, A. H., David, L., et al. 2002, ApJL, 567, L27, doi: 10.1086/339619

Markevitch, M., & Vikhlinin, A. 2007, PhR, 443, 1, doi: 10.1016/j.physrep.2007.01.001

Markevitch, M., Ponman, T. J., Nulsen, P. E. J., et al. 2000, ApJ, 541, 542, doi: 10.1086/309470

Million, E. T., & Allen, S. W. 2009, MNRAS, 399, 1307, doi: 10.1111/j.1365-2966.2009.15359.x

Owen, F. N., White, R. A., & Burns, J. O. 1992, ApJS, 80, 501, doi: 10.1086/191674

Perlmutter, S., Aldering, G., della Valle, M., et al. 1998, Nature, 391, 51, doi: 10.1038/34124

Petrosian, V. 2001, ApJ, 557, 560, doi: 10.1086/321557

Riess, A. G., Filippenko, A. V., Challis, P., et al. 1998, AJ, 116, 1009, doi: 10.1086/300499

Russell, H. R., McNamara, B. R., Sanders, J. S., et al. 2012, MNRAS, 423, 236, doi: 10.1111/j.1365-2966.2012.20808.x

White, S. D. M., & Rees, M. J. 1978, MNRAS, 183, 341, doi: 10.1093/mnras/183.3.341

VITA

Joe Huber enrolled at the University of Central Missouri in the fall of 2012, before transferring in 2015 to Missouri State University. He then graduated with a Bachelor of Science in Physics, with an emphasis in Astronomy in 2019. He entered the University of Missouri - Kansas City in 2022, Receiving a Master of Science Degree in Physics, with an emphasis in Astronomy, in May 2025.