Astrostatistics Class Project

April 29, 2025

1 Introduction

Within this project we will simulate astronomical images and then will examine the various methodologies that may be used for point source detection and characterization. We additionally will investigate the impact of broken power laws on clustering algorithms as well as the influence that priors have on our MCMC results.

2 General Methodology

2.1 Point Creation

For the initial source creation we simulated 500 points, randomly & uniformly distributed among x,y. I then randomly assigned band 1 flux to each of these points by sampling from a flux distribution

$$\frac{\partial N}{\partial F} = F^{-5/2} \tag{1}$$

and we also decide the color by randomly sampling from a log gaussian distribution with a mean of 2 and standard deviation of 0.5. The band 2 flux is determined via the equation below.

$$f_2 = e^C * f_1 \tag{2}$$

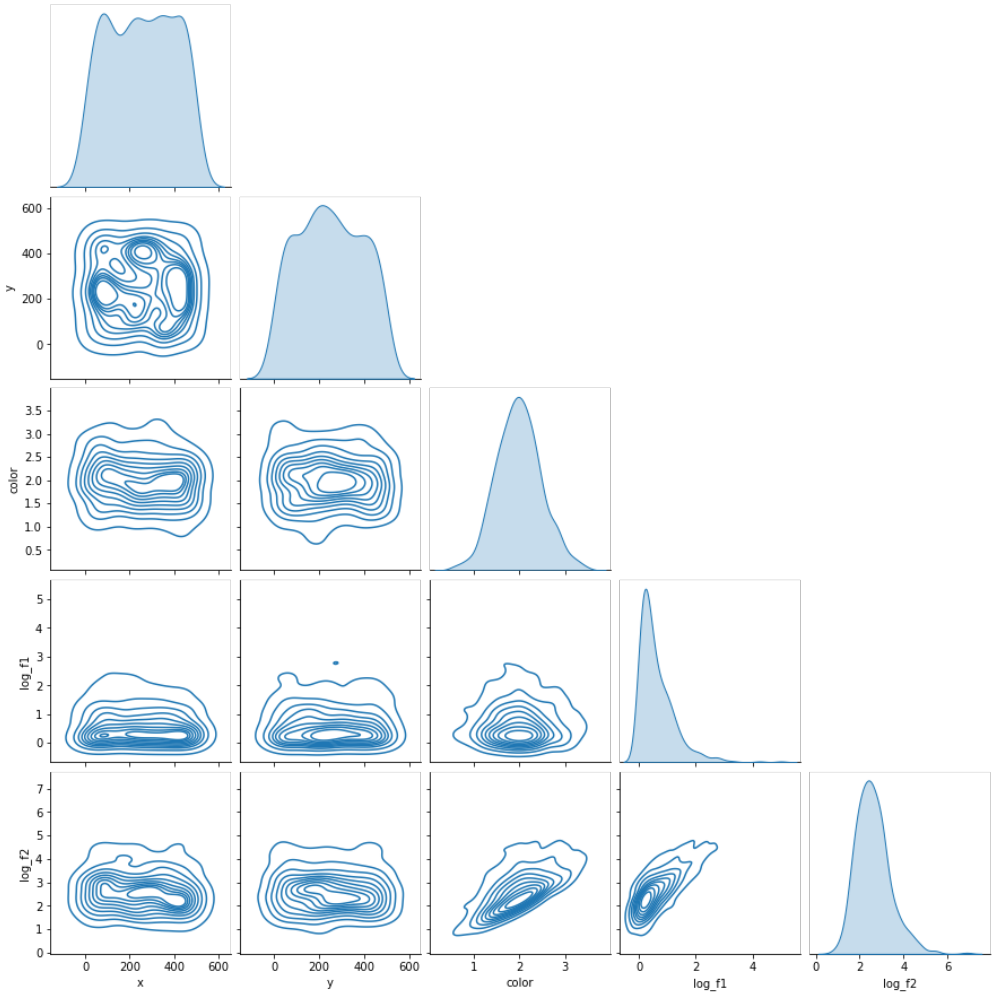


Figure 1: Above is a corner plot displaying the probability distribution of our sources over x, y, band 1 and 2 flux, and color.

We note that only 239 of our sources are considered truly detectable, meaning they have an SNR greater than or equal to 5.

2.2 Simulated Images

With these sources, we can then create simulated images, applying a FWHM of 2.5 pixels.

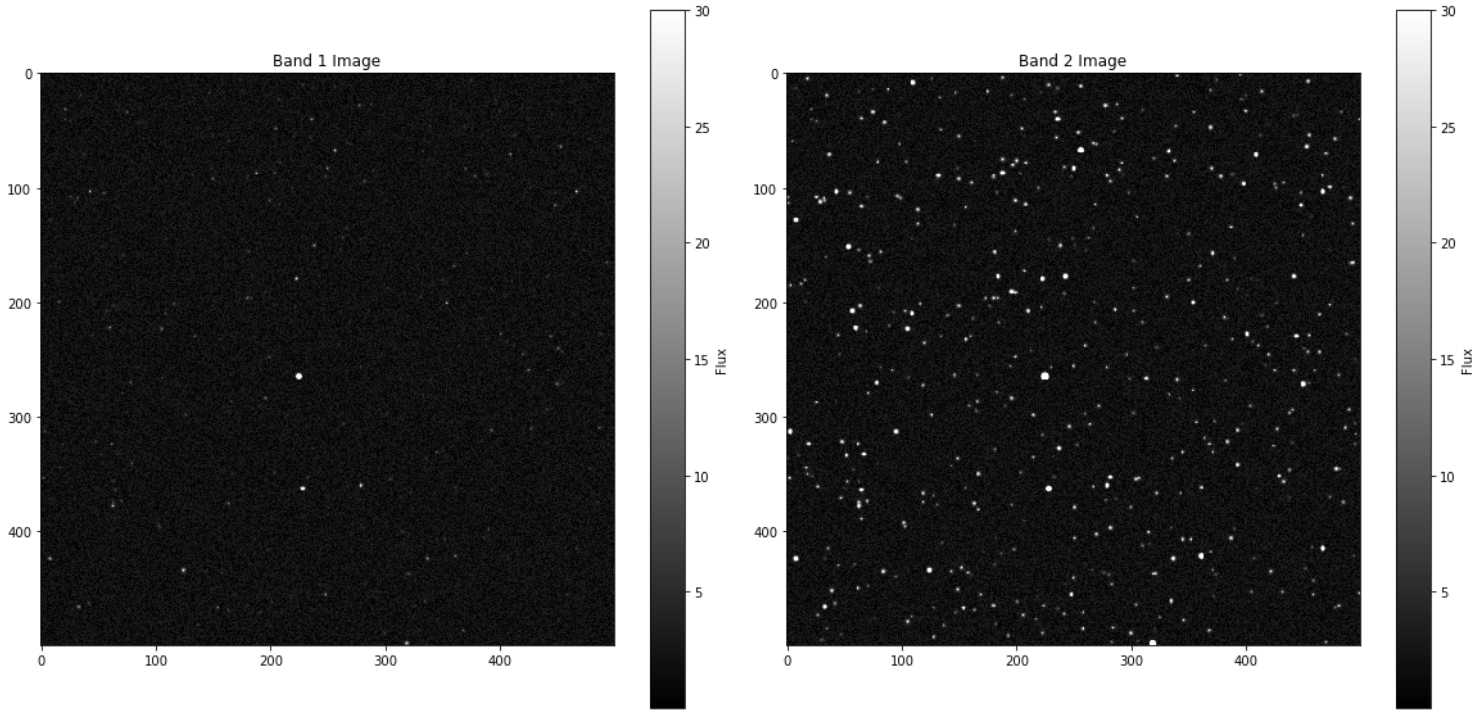


Figure 2: Above are the images resulting from our simulated point sources with an isotropic background and point spread functions set for each point source. On the left is the image in band 1 and on the right is the image in band 2.

3 Candidate Point Source Detection

There are multiple methods we may use to initially identify candidate point sources.

3.1 Outlier Detection

Outlier detection is one such method which relies on viewing sub-images and determining if a pixel is an outlier compared to the general mean. Specifically, we classify an outlier as a pixel with flux greater than 3σ. If there are outliers within 2 pixels of one another we take them to all be part of the same source, whose position we take to be the mean of the outlier pixel positions.

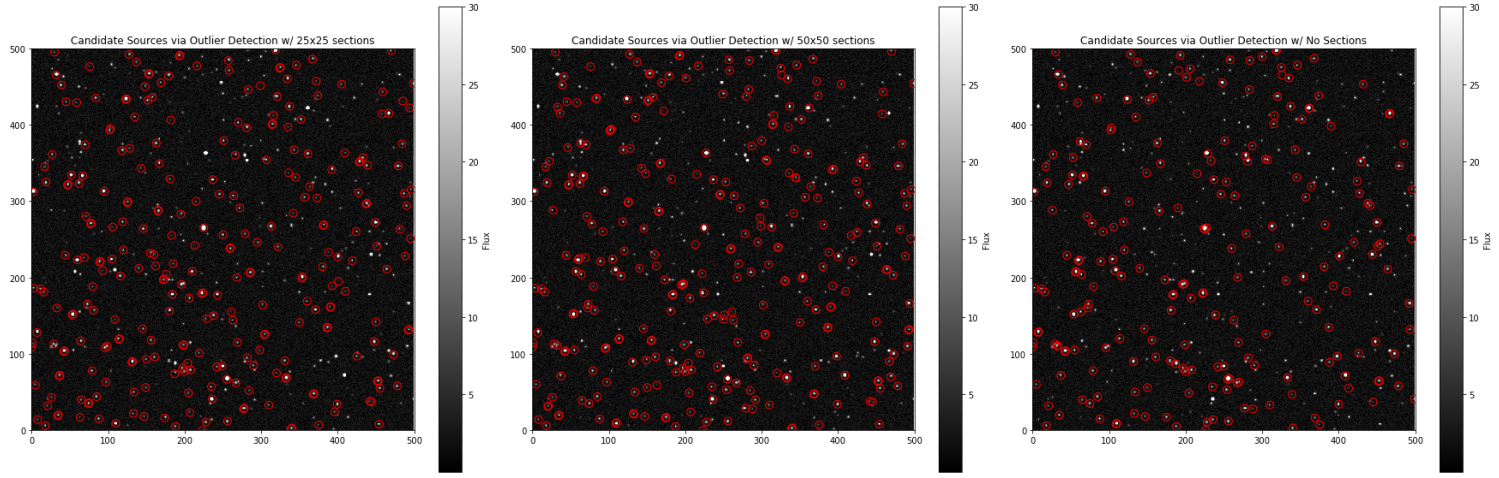


Figure 3: Above are plots showing the results of our outlier detection. Potential sources noted by the outlier detection algorithm are denoted by a red circle and the background is our true band 2 flux image. On the left we see the results when using 25x25 pixel sub-images, in the middle are the results when using 50x50 pixel sub-images, and on the right are the results when using the entire image with no sub-images.

We note from the figure above that the outlier detection method appears slightly sensitive to the sub-image size chosen. In particular, going from the 25x25 sub-images to the 50x50 sub-images and then finally to no sub-images we get 284, 307, and 228 sources respectively. For our purposes going forward we will adopt 50x50 sub-image sizes.

Additionally, the manner in which our candidate detection is applied makes it so that you preferentially detect sources with a higher relative f_1 even if it has a low color and doesn't appear very bright in f_2. This is because we apply outlier detection to both fluxes and then take the list from both bands, however, as is apparent from the image, f_1 is much dimmer and so it doesn't take nearly as much flux in order for a point to become an outlier.

3.2 Template Matching

An additional method of point source detection is to utilize template matching. The theory behind template matching is simple, we start with some "template" or base image that we expect sources to look like. We iterate throughout the image and determine points that fit relatively well with our template. For the purposes of this project, we utilized a variety of templates all based on the same idea, some point source of varying flux with a gaussian point spread function with FWHM of 2.5.

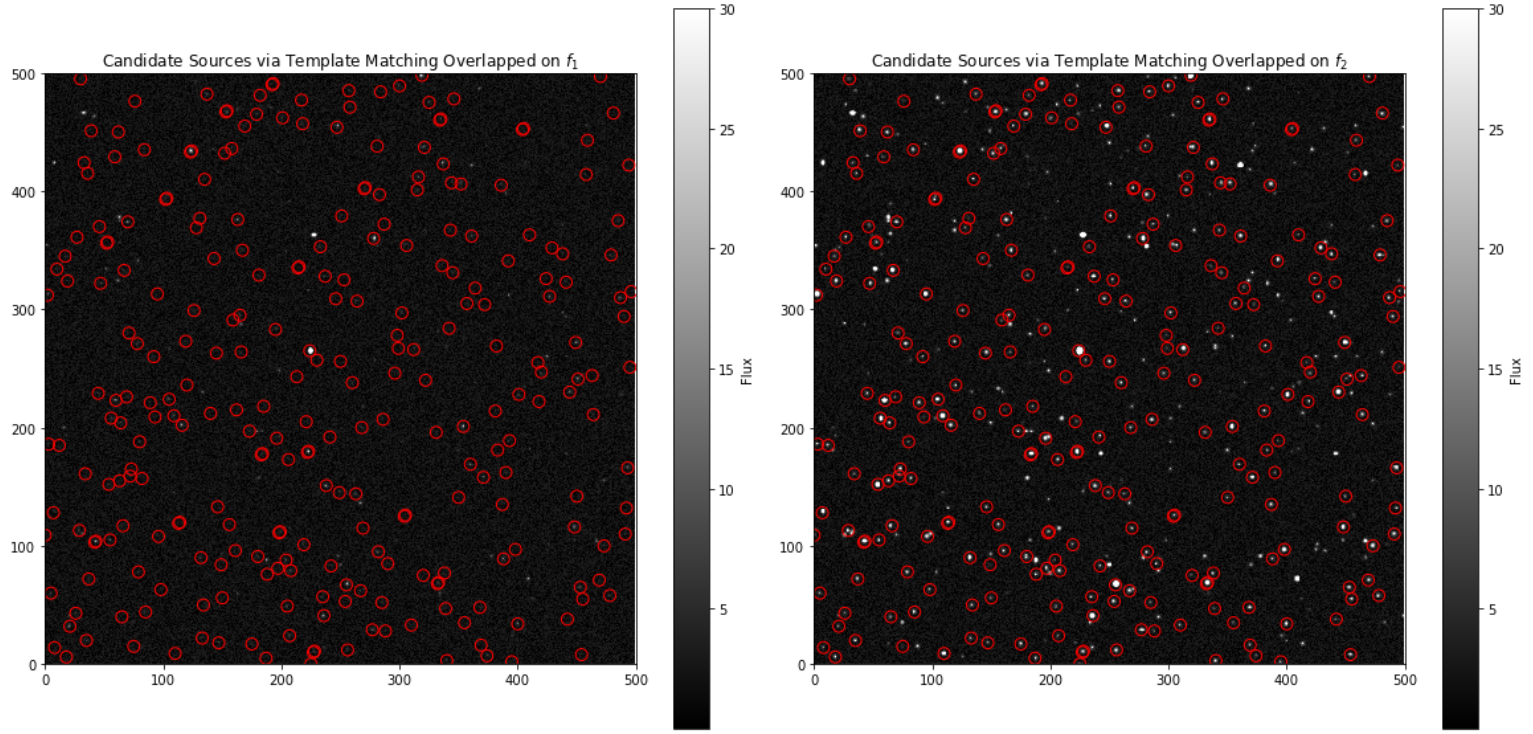


Figure 4: Above are the true band 1 (left) and band 2 (right) images over-plotted with circles centered around the proposed source from template matching. Note that the proposed sources are based on the result from both bands, hence why some of the proposed sources don't make sense when only considering the band 1 image.

Between both bands you get 307 total candidate sources. It's important to note that, when compared to our outlier detection algorithm, a few notably bright sources aren't detected by template matching.

3.3 Precision vs Recall

In order to better compare methods of point source detection we may plot the precision vs recall graphs for each method over a variety of threshold values. In order to avoid potential influences due to our source centering method, which merges any proposed sources within a radius of 2 pixels of one another, we exclude that step when creating these precision recall curves.

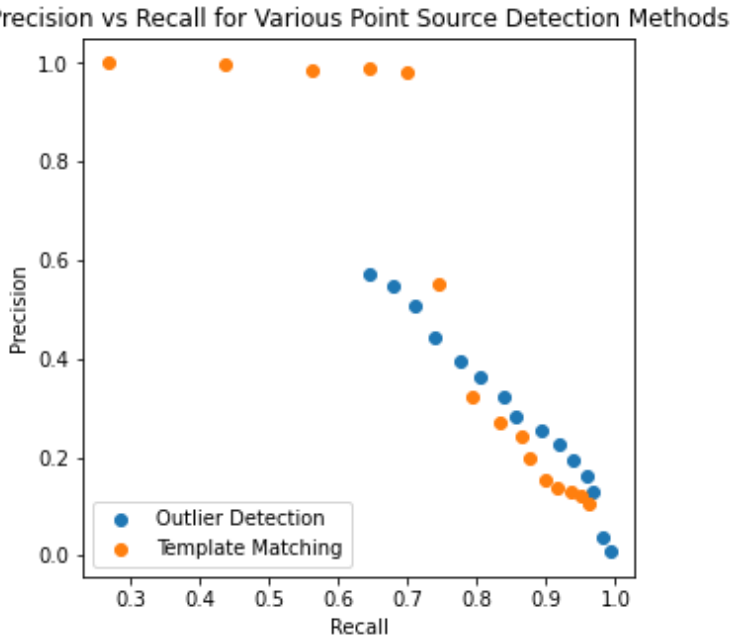


Figure 5: Above are the precision vs recall curves for outlier detection as well as template matching.

Looking at the curves above we note that template matching results in higher maximum precision of nearly 100% meanwhile outlier detection only reaches a maximum of around 60% from what we can tell. This is likely because, without our source centering method, it is likely that a single bright source results in multiple outlier pixels, increasing the number of false positives. We also note that there is a much more dramatic drop off in precision for template matching, meanwhile outlier detection results in a more gradual downward slope.

In general, our recall values may be lowered by the fact that they were taken based on the full list of sources as opposed to simply the list of our "detectable" sources. This means that there are sources begin counted as false negatives that we don't expect to be able to resolve. It is notable though that, despite this, outlier detection manages to reach 100% recall and similarly template matching reaches very close too that 100% recall.

Based on the results of these curves, our inferred source positions will be taken from hereon out using template matching with a image similarity threshold of 0.73 which resulted in a 98% precision and a recall of about 70%.

4 Advanced Source Characterization

Given some set of source positions, we may use more sophisticated methods to characterize the flux and color of a given source. Specifically, we used the Weighted Least Squares (WLS) and Markov Chain Monte Carlo (MCMC) methods, which are described more below.

4.1 Weighted Least Squares

Weighted least squares is a method of determining model parameters in order to reduce residuals using matrix operations. Specifically, it is distinguished from other "least square" functions because WLS considers the "weights" or variance-covariance matrix which allows you properly consider if you expect non-uniform residuals. If you expect different parameters to influence one another as well that information would be basked into the covariance matrix, although this is not the case for any of our parameters as, even though f_1 and f_2 are very heavily tied to one another, the covariance still ends up being 0 because of the independence between color and f_1. This leaves only the diagonal of the variance for each parameter, which for our sources are $\frac{1}{f_1}$ and $\frac{1}{f_2}$.

Our WLS only examined the band 1 and 2 flux, and consequently the color. The x-y positions of the sources were taken from our previous inferred source positions.

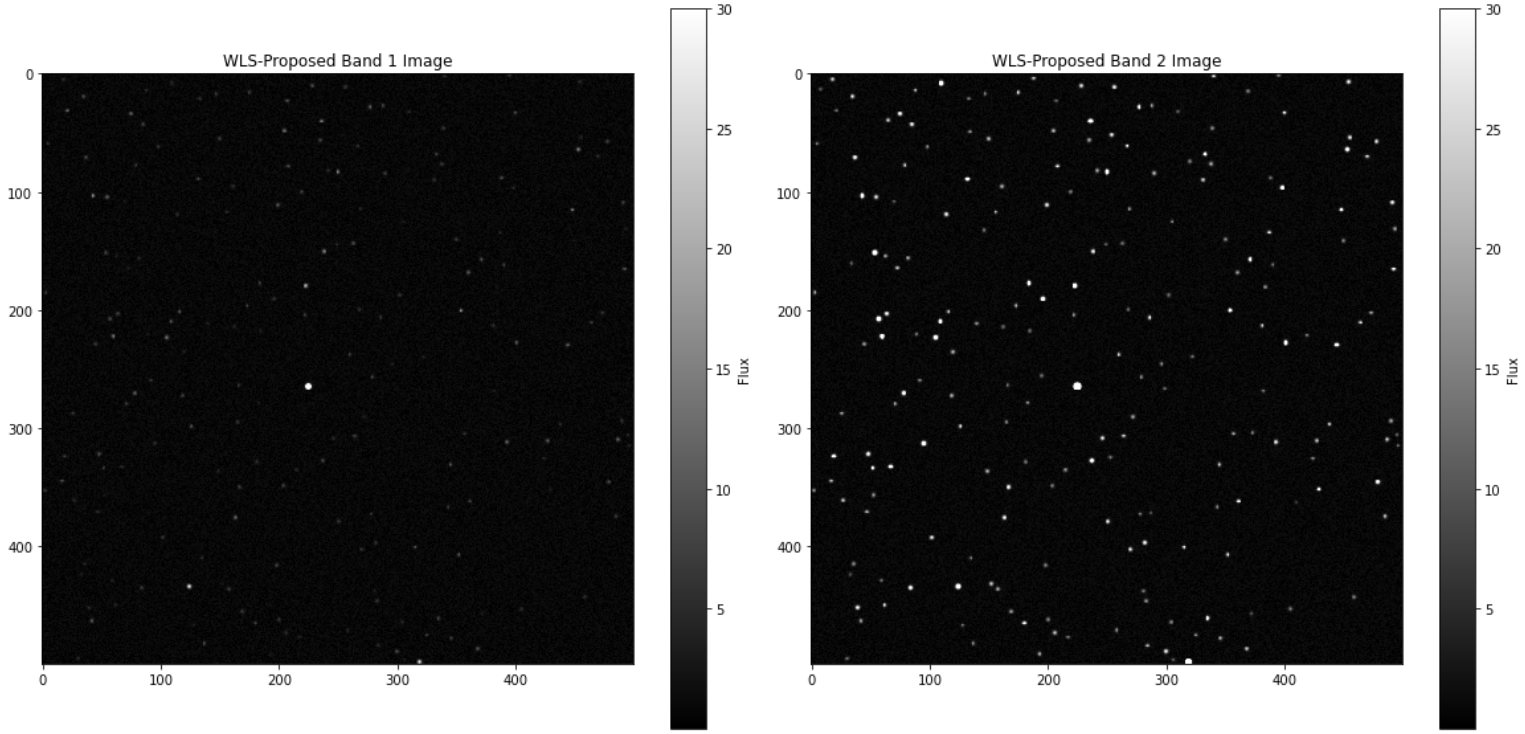


Figure 6: Above are the proposed images using weighted least squares in band 1 and band 2.

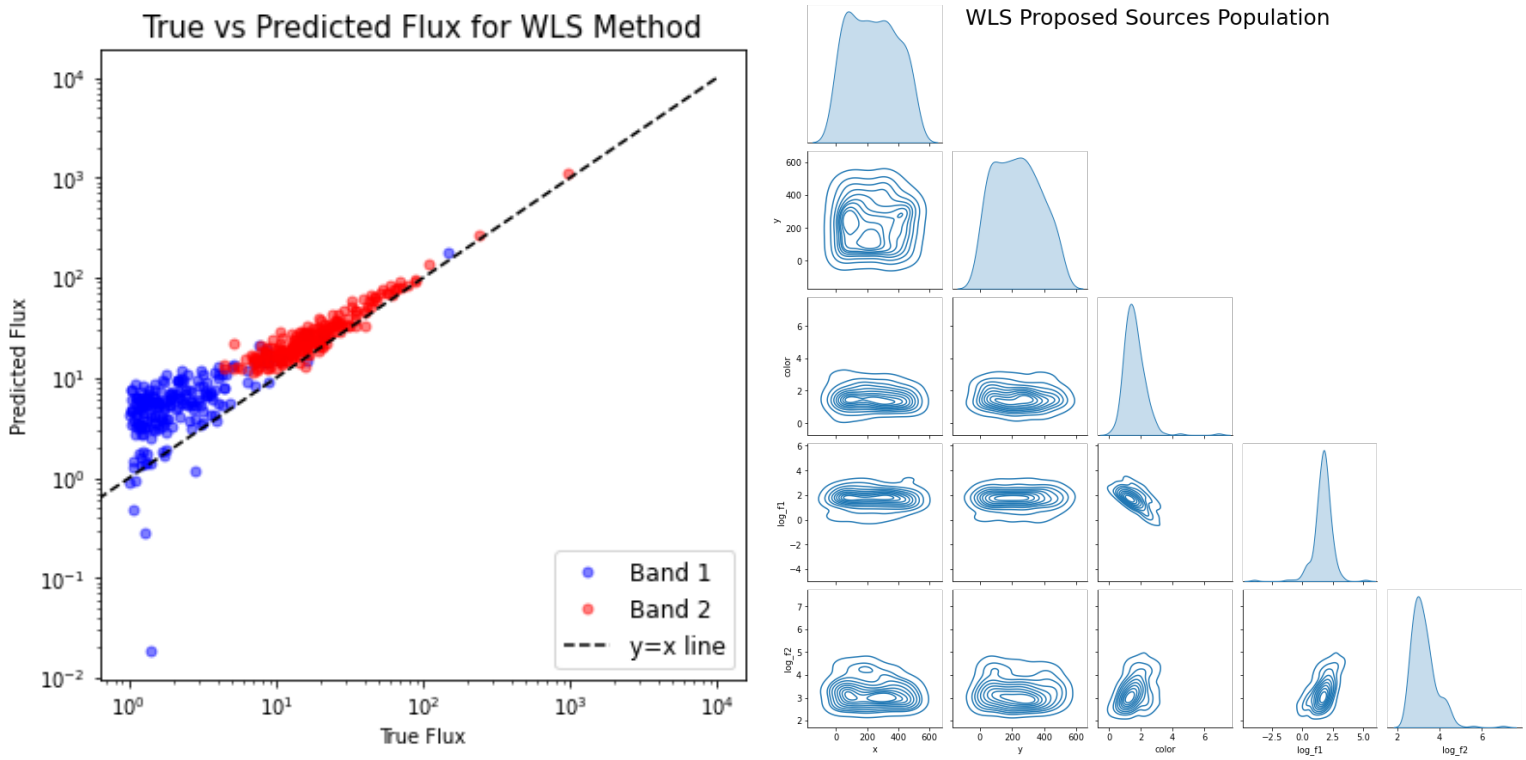


Figure 7: Above on the left is a plot of the true vs predicted flux of sources using the WLS method. Specifically, this is only demonstrating the sources that were properly identified. Incorrect predicted sources or missed sources aren't included. The left however shows the corner plot for the population of sources proposed with the WLS method.

Looking at the true versus predicted flux plot above, it appears that our weighted least squares behaves increasingly well for higher fluxes. Additionally, we see that generally the predicted flux overestimates the true flux, especially for band 1. There are however a few sources where our model greatly underestimated their flux. The average relative error for correctly identified sources is around 1.89 for band 1, and around 0.557 for band 2, supporting the observation that band 1 has a worse fit. This is counterintuitive as, given the variance-covariance matrix, one would expect for there to be a more accurate model for smaller fluxes.

The corner plot further supports this conclusion, with us seeing that for a few sources the $log(f_1)$ dips to be negative, suggesting that there are a few sources with much smaller flux than expected. We also see that the color distribution is shifted to the left towards

smaller color values, although there are a few sources with high color. This makes sense as a lower average color naturally results from having a accurate f_2 and an overestimated f_1.

Overall, the predicted fluxes being consistently higher than the true fluxes suggests that our model doesn't consider the impact of background enough and upon checking this appears correct. Specifically, our average background was predicted to be 0.05 meanwhile in reality the mean background contribution was 2.

4.2 MCMC

Within our MCMC algorithm, due to time constraints, we narrowed our scope to focus on a randomly selected 60x60 pixel square within our image. Below is the sub-image that was selected, although it should be noted that the tick marks don't represent the true position of the square relative to the full image. Within this sub-image as opposed to our full 500 point sources we have only 8 point sources, 7 of which are detectable.

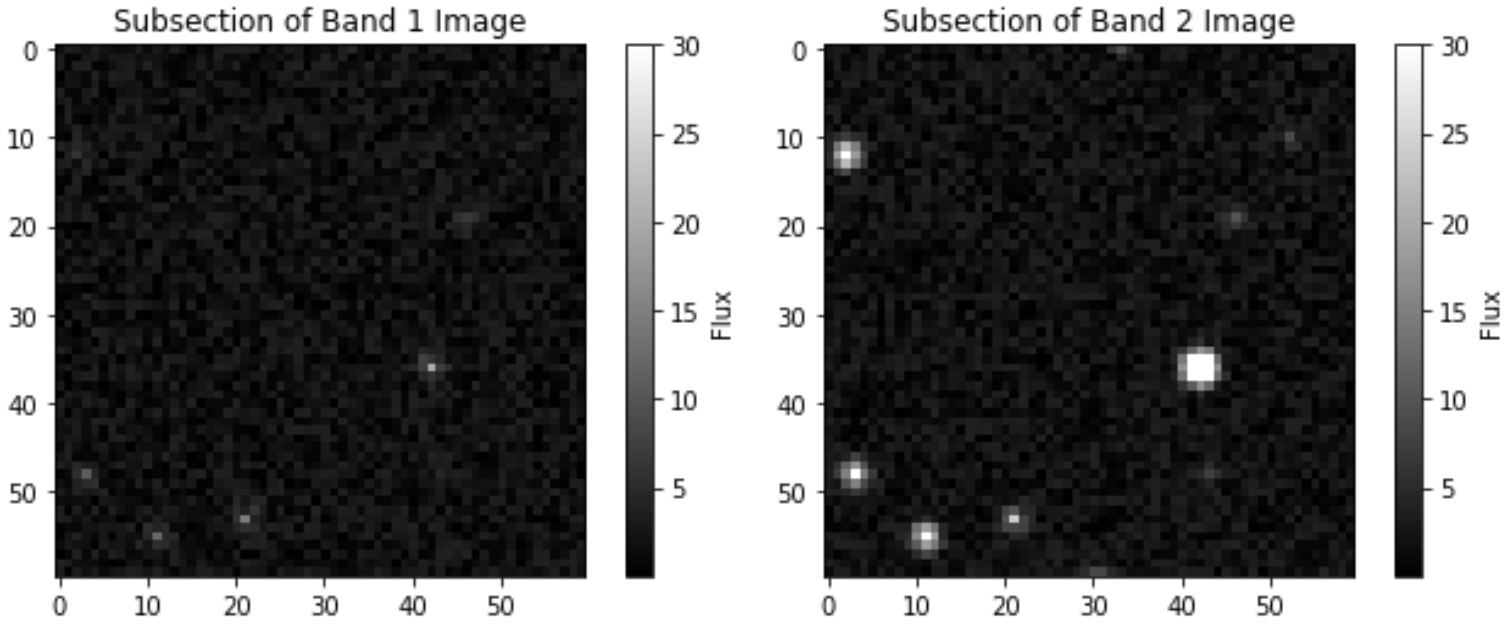


Figure 8: Above are the band 1 and band 2 sub-images that are used for our MCMC tests.

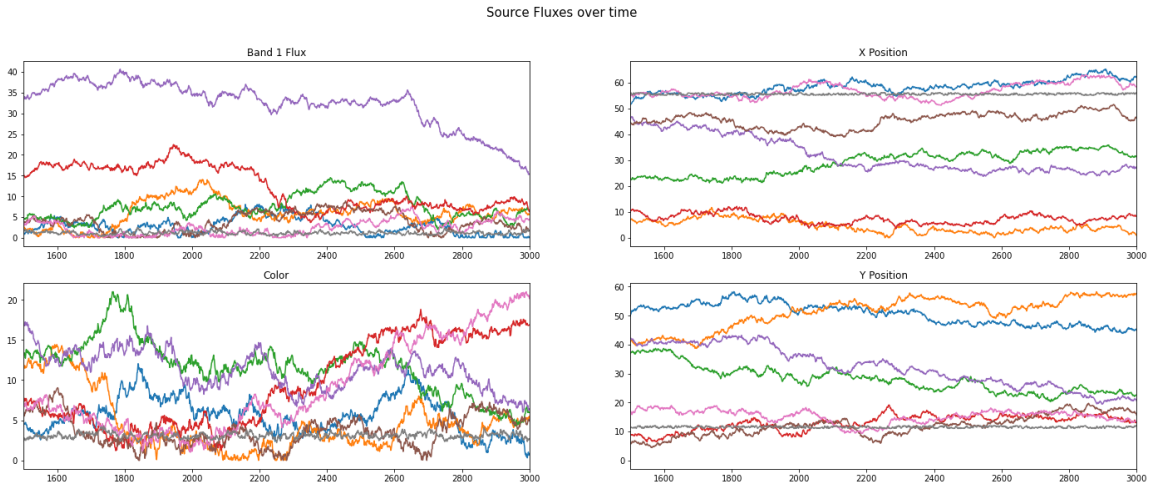
For the MCMC algorithm we went through 30,000 cycles and we save the results from every tenth cycle. From there we also examine the average change in flux and position over time and only choose the data once it appears that the flux and position have stabilized in a process known as burnout. For our particular run this was after 15,000 cycles. Additionally, each step in color or flux was randomly taken with a standard deviation relative to the current value of 0.1.

Something else that is incredibly important to note is that, due to some bugs and coding difficulty, my MCMC algorithm does not predict the background flux.

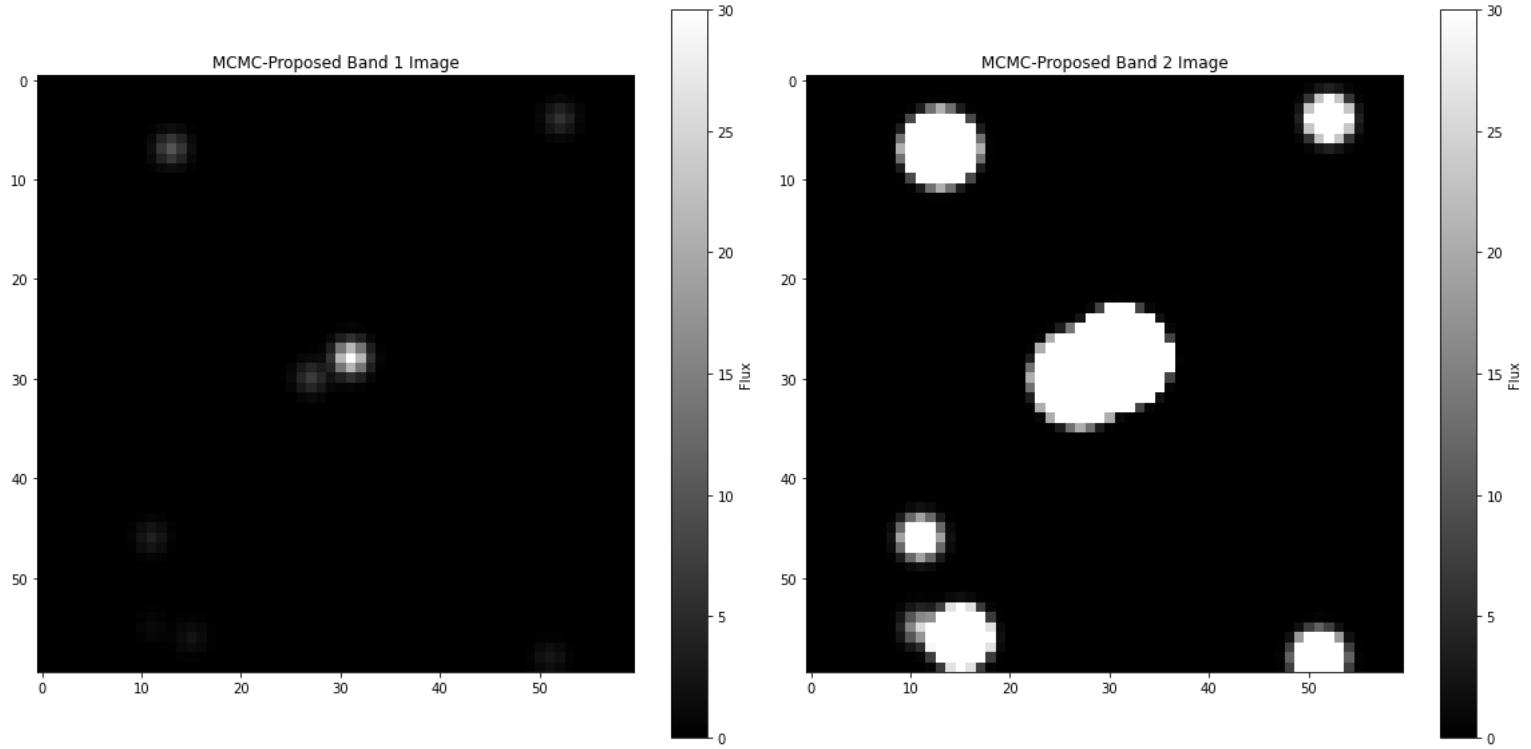
4.2.1 Initial Position Influence

With our MCMC algorithm we may test it using a variety of initial positions. Specifically, we tested how it performed when you gave it the true initial positions of the sources and the inferred positions that we assumed via template matching. We originally intended to also investigate the results when beginning with the true positions biased by ± 1 pixel (which for our case is about 0.4*FWHM), however due the long runtime of our algorithm and additional time constraints, this chain didn't finish and thus won't be included.

True Positions For this section we began by utilizing the true positions of our 7 detectable sources. We then gave an initial flux and color by randomly sampling from the prior distribution. The result was an monte carlo chain that had an average acceptance rate of about 18% and the following history for our four parameters.



Looking at the history as well as the images below we can see that our MCMC algorithm overestimated the color and band 2 flux. The images below were specifically created by taking the median positions and fluxes for each point source, which remains labeled throughout our code.



Looking in the band 2 image one can see that you had the three sources in the bottom left. However, their relative positions and many of the other sources are placed incorrectly. This is surprising as, while the flux didn't remain very stable and therefore can be expected to be a bit incorrect, the positions remained relatively constant.

	x	y	color	f1	log_f1	f2	log_f2	in_subsection	adjusted_x	adjusted_y
19	84	410	1.400503	1.646813	0.498842	6.681516	1.899345	True	48	43
42	46	419	1.312972	1.325778	0.281999	4.928188	1.594971	True	10	52
118	55	413	1.156184	1.740804	0.554347	5.531901	1.710531	True	19	46
134	48	369	3.066131	1.123192	0.116175	24.102257	3.182305	True	12	2
393	72	409	2.831829	5.974624	1.787521	101.428066	4.619350	True	36	42
458	84	370	1.726746	4.103351	1.411804	23.070397	3.138550	True	48	3
480	89	388	1.570163	2.473876	0.905786	11.892988	2.475949	True	53	21
482	91	378	2.106968	3.181281	1.157284	26.160534	3.264252	True	55	11

	x	y	f1	color	f2	state	pointSource
0	58	51	2.744046	5.539758	6.271504e+02	median	0
1	4	52	5.958545	3.422639	2.139313e+02	median	1
2	30	27	6.970583	12.129357	1.328706e+06	median	2
3	7	13	10.005161	8.125468	2.484838e+04	median	3
4	28	31	33.409239	11.467658	3.064115e+06	median	4
5	46	11	3.833103	3.382829	1.123287e+02	median	5
6	56	15	2.318292	6.828970	2.232441e+03	median	6
7	55	11	1.107738	3.064336	2.443395e+01	median	7

Figure 9: Above on the left is the table of the true sources within our sub-image, with the adjusted x and y values indicating the position relative to this smaller sub-image. Meanwhile on the right is the table of sources recovered via our MCMC algorithm.

Looking at the tables above, though, it appears possible that our MCMC algorithm resulted in a few of the sources, namely source 6 and 7, potentially being "merged" or nearly merged.

Inferred Positions With our new sub-images we ran template matching with the same parameters and code as before and detected 5 point sources, shown below.

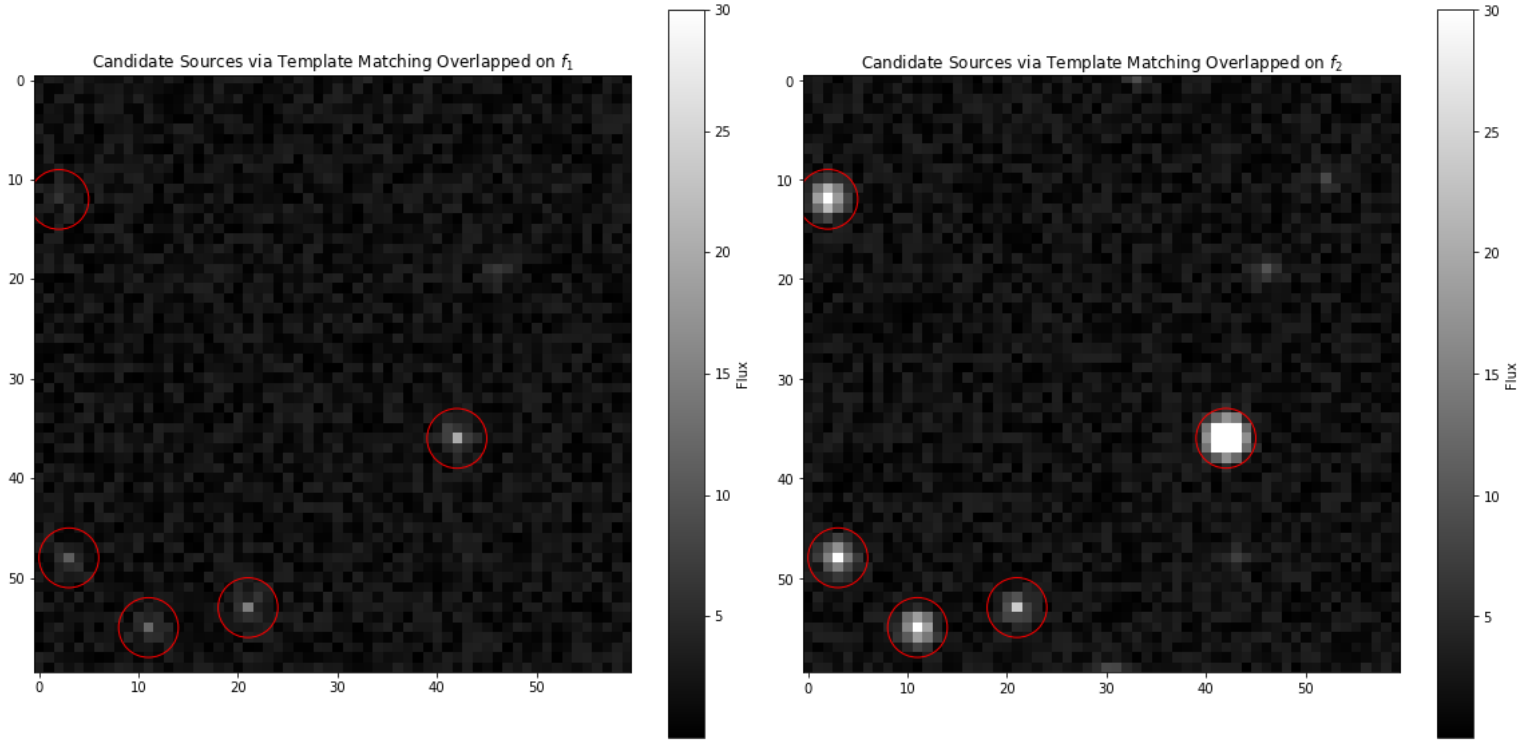
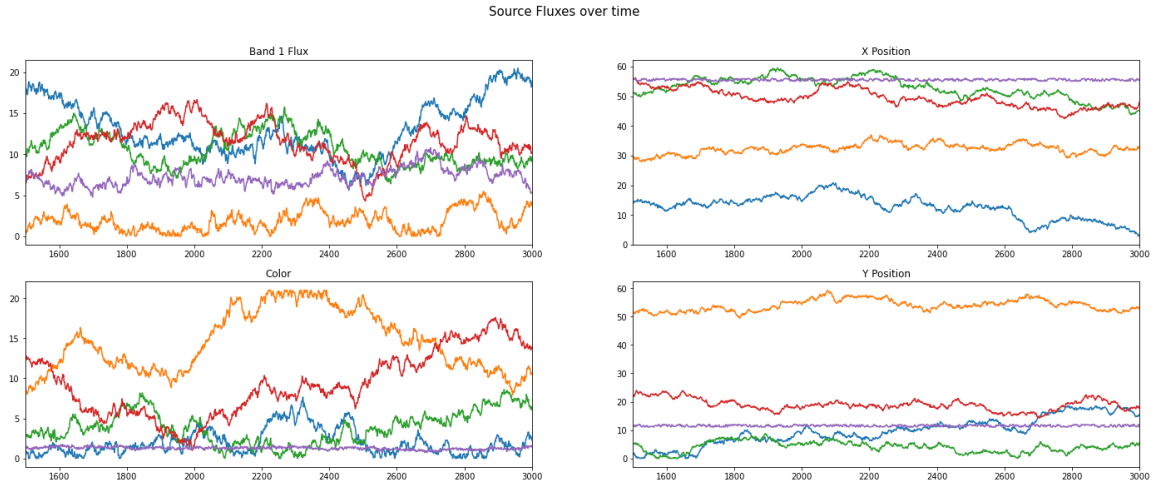


Figure 10: Above are the band 1 (left) and band 2 (right) sub-images over-plotted with circles centered around the proposed source from template matching.

The more limited view allows us to note some ways that our template matching could be improved in the future, namely, our optimized threshold from the full image precision-recall curve appears to be too strict for our sub-image. We notably miss a relatively bright source in the bottom right region as well as two dimmer, but still notable sources in band 2 around the same region. When testing with a smaller threshold of only 0.5, we managed to recover INPUT HERE sources in total with only INPUT HERE of them being false detections. In order to remain consistent and to give our MCMC algorithm a bit of a challenge, we will maintain our original threshold of 0.73.

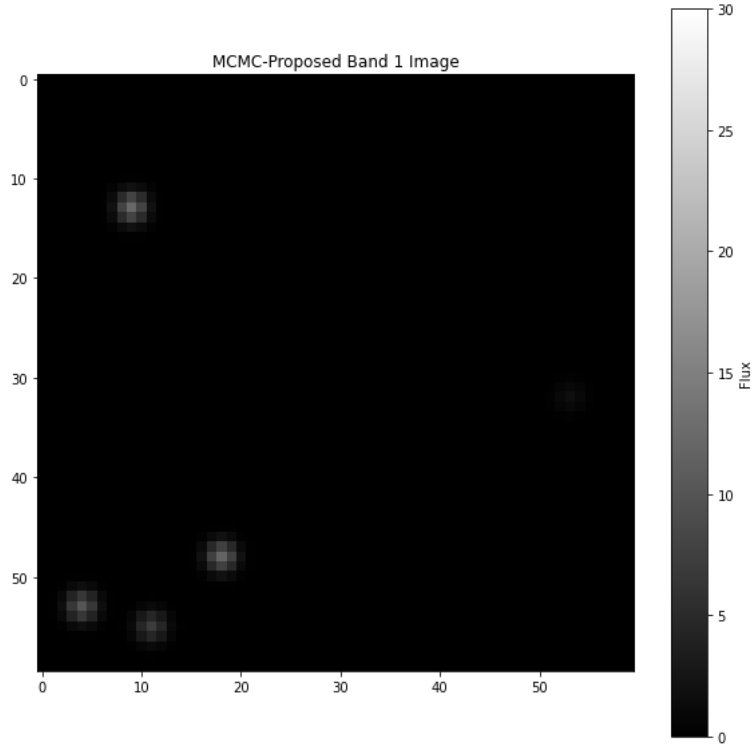
For our initial fluxes within the MCMC we utilized the fluxes as determined by our WLS algorithm.

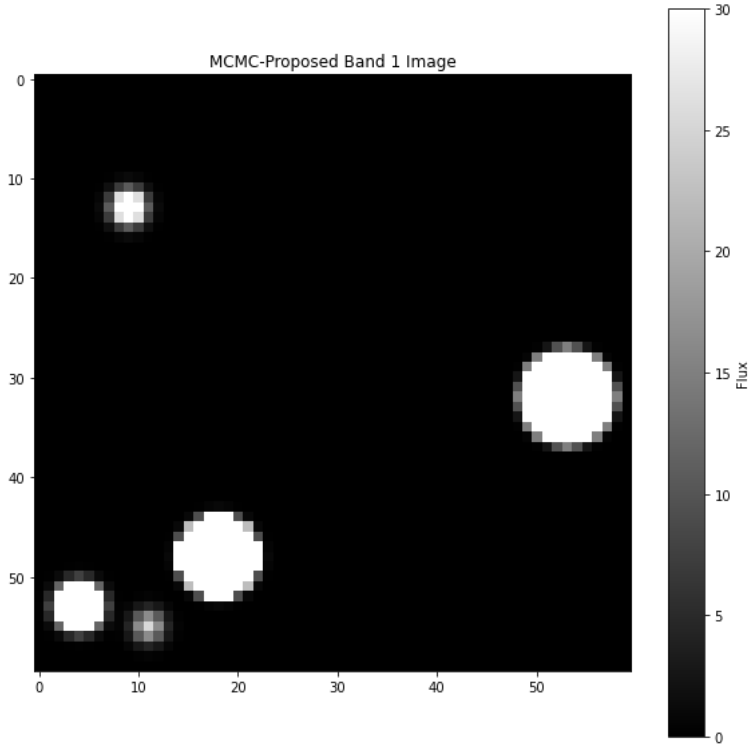
The result of all of this was an monte carlo chain that had an average acceptance rate of about 26% and the following history for our four parameters.



Similar to our MCMC results using the true positions, it is obvious that our algorithm vastly overestimated the band 2 flux. However, it does appear from the image below that we roughly got the relative positions correct, you see the three sources bunched to the bottom left, one in the upper left, and one off to the middle or bottom of the right. That being said, there is definitely room

for improvement, especially within the relative y positions of the three sources in the corner, and with respect to the x position of the two other sources.

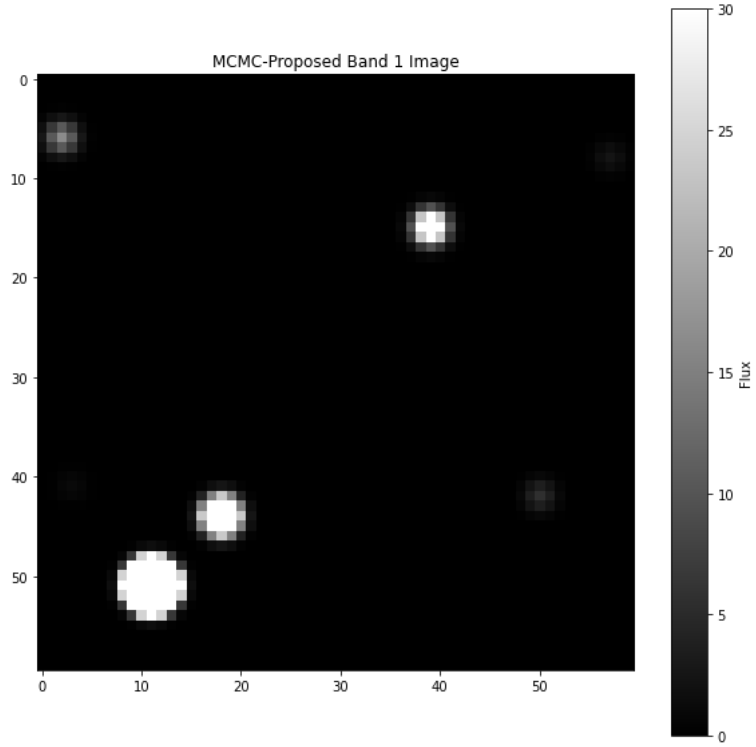


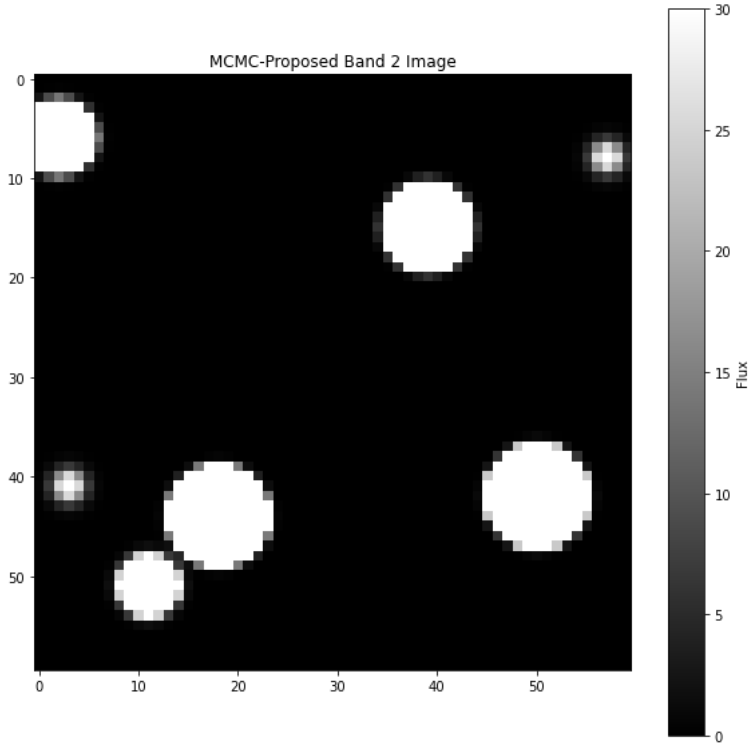


4.2.2 Prior Influence

Next, we sought to test how much our priors influenced the results. Specifically, we tested what would happen if you were to give it a flux prior with $\alpha = 3/2$ as opposed to the correct $5/2$. For this test, to keep everything consistent, we provided it with the true initial positions of sources. Judging from the results above, one would expect that the band 2 flux should sky-rocket even higher than before, and similar the band 1 flux should increase since there are not as tight constraints with regards to the prior. However, with higher flux above what we truly expect to see comes smaller likelihood values, and so that may limit any run-away that occurs.

Looking at our true images below, it appears that the constraints imposed by the likelihood wasn't enough and we did indeed get much brighter sources in band 1.





This demonstrates the importance that having a correct prior may have on your model, especially if your algorithm, like my

MCMC, is less than ideal.

We do note though that, interestingly enough, it does do a slightly better job of recovering the true positions compared to our initial test with the true positions and the true flux prior. This is likely because, with the higher flux, the likelihood made it much more important for the sources to at least be in their proper position.

5 Broken Power Law Source Clustering

To further expand our work we also examined the case where you have a broken power law in flux given by the equation below.

$$\begin{cases} \frac{\partial N}{\partial F} = F^{-7/2} & x < 2 \\ \frac{\partial N}{\partial F} = F^{-5/2} & x \geq 2 \end{cases}$$

This produces the following population distributions.

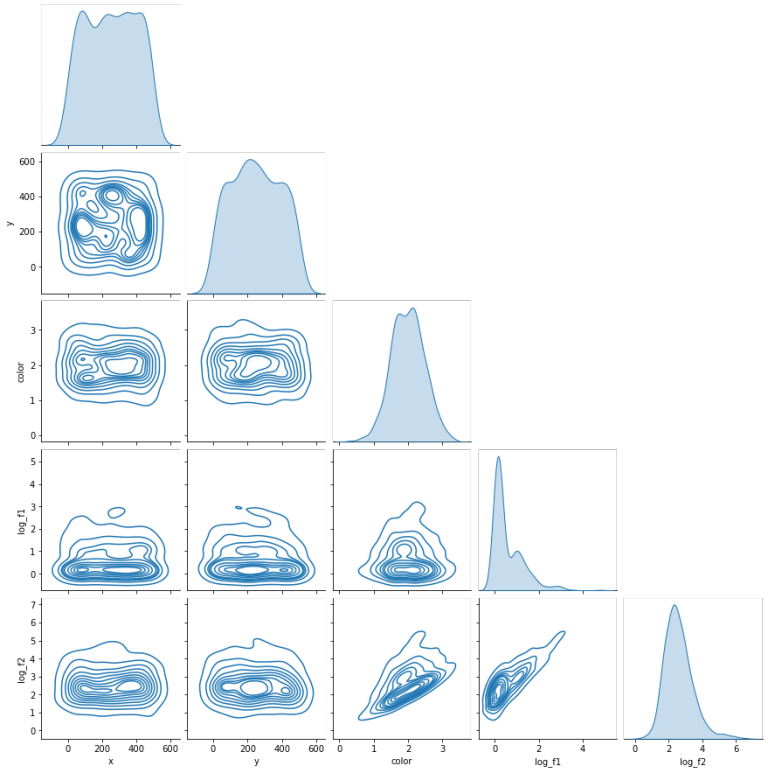


Figure 11: Above is a corner plot of x and y positions, $log(f_1)$, $log(f_2)$, and color for our broken power law population.

Given these sources, we may examine various cluster algorithms to determine if there's any notable clusters that may influence our results.

5.1 KMeans Clustering

KMeans clustering is a classic clustering algorithm commonly used in a variety of applications. It aims to partition a data set into k groups where each cluster is centered around some centroid and data points belong to the cluster with the nearest centroid. The true KMeans clustering problem is NP-hard, meaning there's no polynomial time algorithm that will always give you an optimal solution, however the algorithm that we use from sklearn, known as the Lloyd algorithm, will give you a local optimum relatively quickly.

We used the elbow method in order to determine the optimal number of clusters, which we determined was 3.

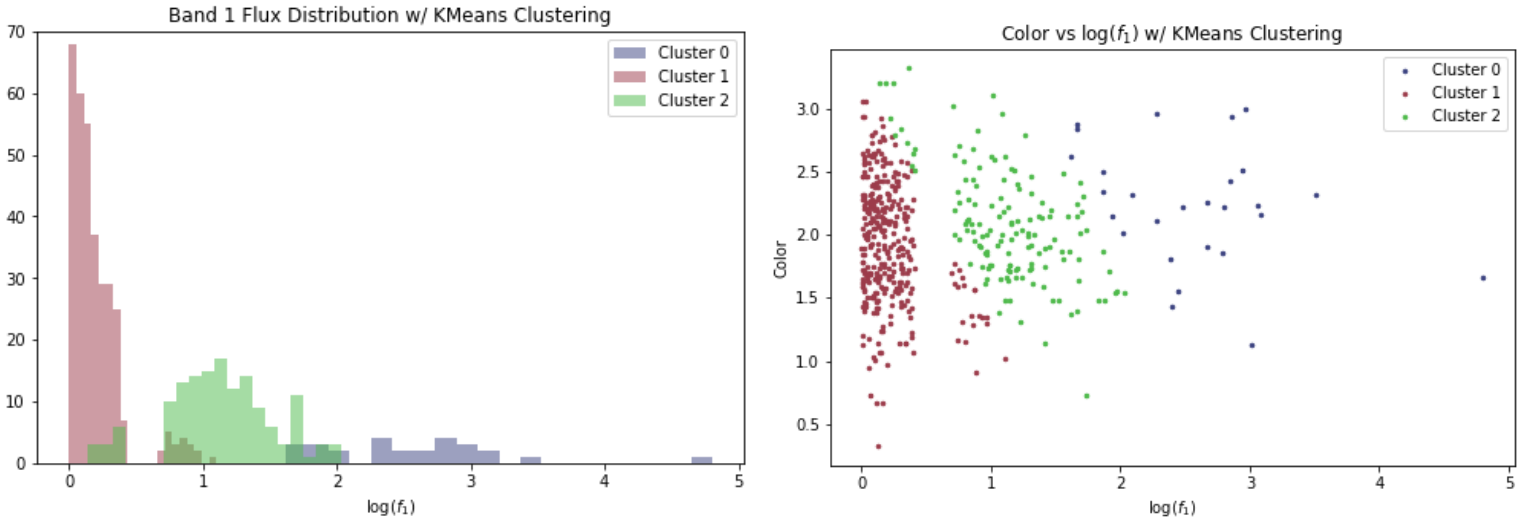


Figure 12: Above are the plots of the band 1 flux distribution (left) as well as the band 1 flux vs color (right) for each cluster.

We see above that, while the clusters aren't perfect, they fit much better with what we expect. There are 330 sources in cluster 1, 143 sources in cluster 2, and 27 sources in cluster 0 and looking at their placements it appears that cluster 1 captures primarily the sources that fall in the first section of the power law where $\alpha = 7/2$ while cluster 2 captures the second section of the power law where $\alpha = 5/2$, leaving cluster 0 to represent all of the outliers with absurdly high flux. The ratio of the number of sources that fall into cluster 1 vs cluster 2 however doesn't fully fit our expectation that about 35.35% of sources to fall in the first part of the power law. This may partially be because of the overlap between the power laws. Specifically, the second part of the power law overlaps for lower flux values, making the cluster proportions make sense.

5.2 Spectral Clustering

Spectral clustering is a clustering algorithm which determines cluster labels using the eigenvalues of the similarity matrix between data points. It is particularly useful when examining non-convex or high-dimensional data. For our data I assumed that we knew there should be two clusters.

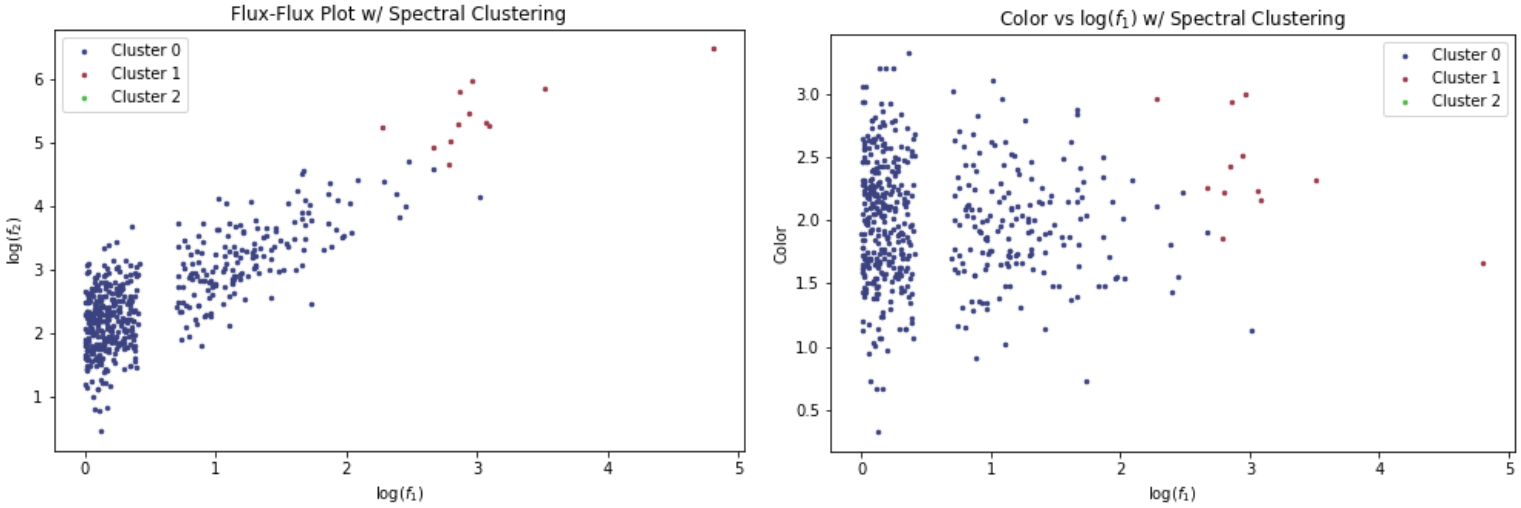


Figure 13: Above are the plots of band 1 vs band 2 flux (left) and band 1 flux vs color (left) for each cluster.

Looking at the plots above it's apparent that spectral clustering didn't perform very well. Specifically we note that there are 488 sources in cluster 0 and only 12 in cluster 1, making it seem like cluster 1 is less of a true cluster but is instead a set of outliers. The fundamental clusters, however, aren't distinguished. Even when we set the number of clusters to be 3 we get similar results, with the third cluster being very small and only capturing a few high flux outliers.

This example demonstrates an interesting fact that in the world of clustering, a more sophisticated algorithm isn't always better. Spectral clustering is typically a much more complex approach to clustering when compared to KMeans, yet due to the nature of our data KMeans still outperforms it.

5.3 Gaussian Mixture Model

Gaussian mixture models are another way that we may cluster, where clusters are assumed to take the form of gaussians within the data space, that may or may not overlap.

For the gaussian mixture model, we choose the ideal number of clusters by using the elbow method with the BIC (say what it is) and AIC (say what it is) metrics. From these values we determined that the ideal number of clusters is 3.

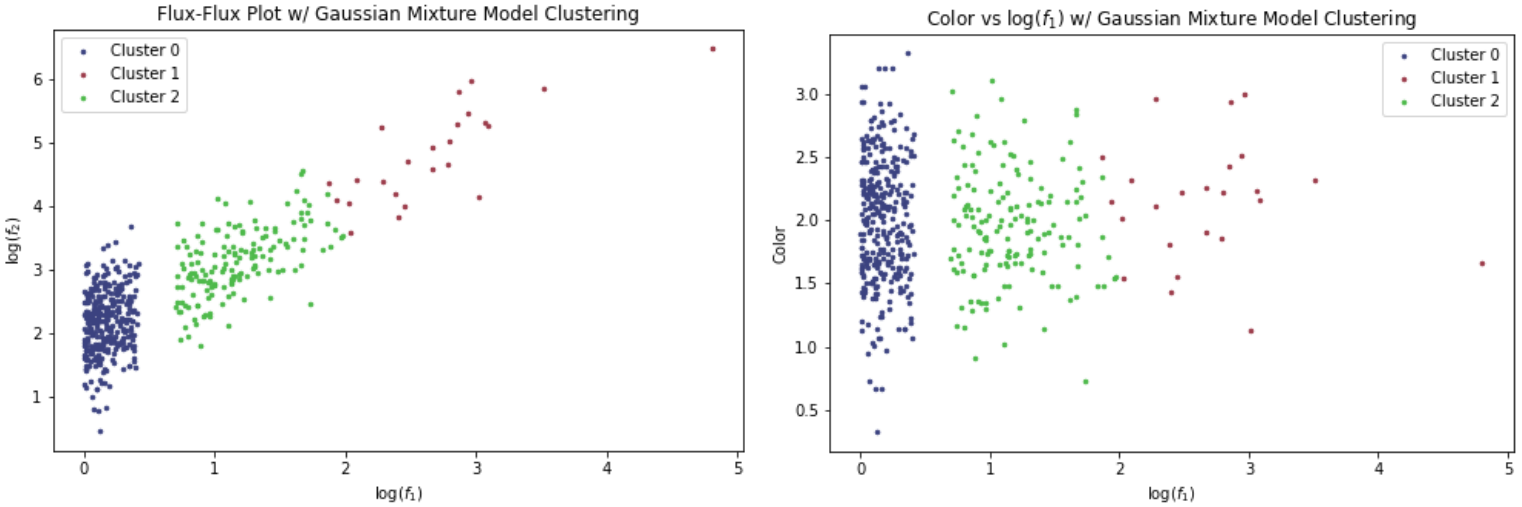


Figure 14: Above are the plots of band 1 vs band 2 flux (left) and band 1 flux vs color (left) for each cluster.

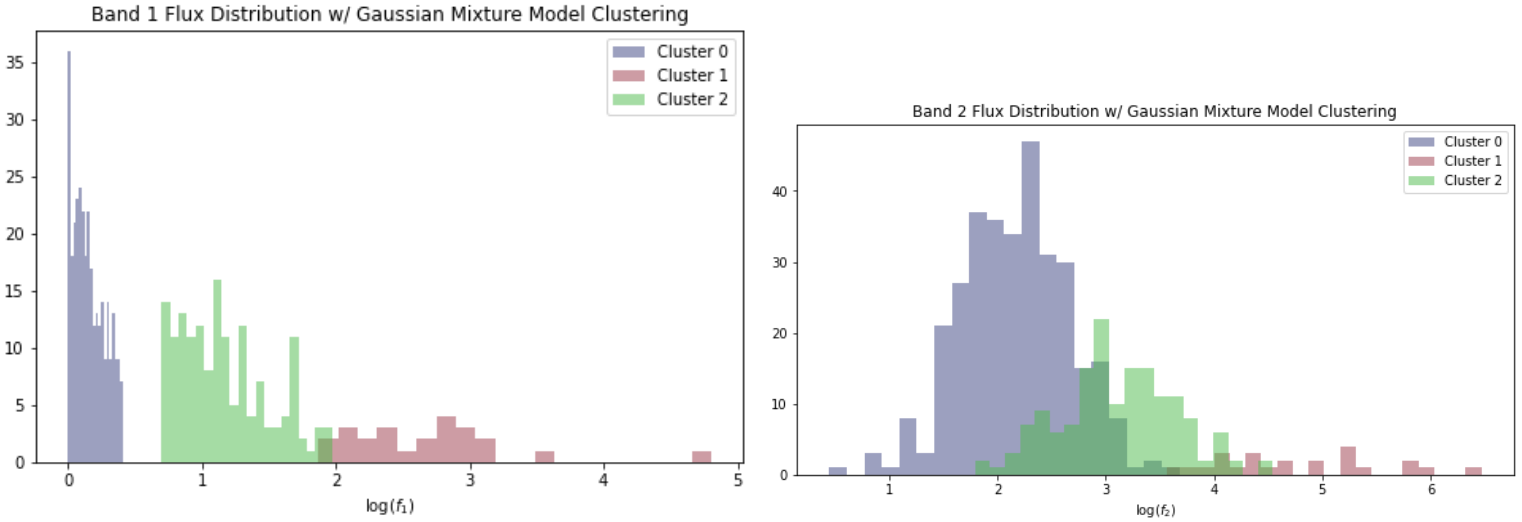


Figure 15: Above are the band 1 (left) and band 2 (right) flux distributions for each cluster.

We see from the figures above that the gaussian mixture model works extremely well and has very similar interpretation of the various clusters compared to the KMeans results. Something that is interesting to note is that each cluster has a gaussian f_2 distribution that overlaps some. This makes sense because while there are the distinct power laws within f_1, the color gaussian is the same for all points and so you do expect some gaussian centered around the mean f_1 value times the exponent of the mean color value for each cluster.