**Angell Observatory CCD Characterization**

**Via Nielson**

**ASTRO 361 Lab 3 Report**

**Abstract**

We conduct an experiment to test the Mean-Variance “Method II” to determine the

electronic gain and detector read noise of the University of Michigan’s Angell Hall Observatory CCD camera by characterizing the data obtained in the form of flat field and bias frames. We used the techniques of “Method II” to look at the root mean square data in the "depth" direction of the cube, which is time. This allowed us to obtain an estimate of the mean and standard deviation for each pixel of each flat field frame, and their respective errors and uncertainties for plot-fitting values. The fit values were further tested for errors using the Bootstrap Method, which was found to be sub-optimal for the tested data. We find that electronic gain is approximately and that the detector read noise is. We also find that the camera saturates at an exposure time of 50 seconds with noticeable non-linearity starting at 30 seconds. We conclude that the gain is the inverse of the mean-variance slope, that the read noise is the square root of the y-intercept of the mean-variance fit, and finally, that the “Method II” techniques are sufficient in estimating these values with minimal error.

**Introduction**

Charge-coupled devices (CCDs) are extremely sensitive and accurate photon detectors, but there are limitations on their performance. The read noise is an important factor that determines the ability of the CCD. One limitation of the camera, studied in this experiment, is the read noise. The read noise is the noise inherent to the detector generated as the charge present in the pixels is transferred to the camera. This noise is generated by the electronics that convert the charge of each CCD pixel into a signal for conversion into an ADU (analog to digital unit) value. Because read noise is independent of signal, it will set a detectable noise floor. The noise can be measured by taking repeated images with the same amount of charge in each pixel. Since the amount of charge in a pixel depends on the amount of light entering that pixel, we take bias frame shots to ease the process because bias frame shots, theoretically, have no light entering.

A bias frame is the response of the detector with no incident light and with an integration time of zero. Because of the finite range of the ADC, there is some non-zero offset in ADU per bias. Knowing this level varies from frame-to-frame due to read noise, for this experiment we took 10 bias frame shots and averaged them together to form a “master bias” with then negligible read noise. The variation in this set of sequential bias frames provides an excellent estimate of the read noise of the CCD camera.

To detect the gain of the camera, (and still the read noise, although perhaps slightly less accurately), we took flatfield images at varying exposure times. Flat field images are the dark-corrected response of a detector to a uniformly illuminated field. We take these to measure the detector variance at multiple light levels (integration times) and plot the relation to ultimately deduce the gain (and read noise) of the detector. This is possible through the resulting analysis of the Mean-Variance test. The accurate detection of the gain and read noise characteristics of the CCD camera is important because it allows us to quantify the capabilities and limitations of the detector.

**Theory**

CCDs (charge-coupled devices) are integrated circuits containing an array of semiconductor-based photodiodes that convert light photons into electrons. In principle, the CCD captures images and then translates them into electronic data. When exposed to light, different sections of the CCD build up charge proportional to the light’s intensity. We can then measure that charge and know precisely how bright that section of the image should be. The CCD uses clever geometry for creating individual “pixels” (picture elements) and to prevent capacitive coupling (that results if pixels are connected to wires). The electronics utilize the “bucket brigade” method, a pivotal innovation for a successful CCD Readout. A CCD is made from a slab of silicon. Engineers have made spots for pixels by creating insulating sections called channel stops (these divide the slab into rows). The surface is coated in a thin layer of insulating silicon-dioxide, and perpendicular to the channel stops, engineers deposit thin strips of metal, typically aluminum. Each pixel then is just one section of this grid. So, when the CCD is exposed to a light source, we then have a specific voltage held by each silicon pixel. The CCD then shifts charges row to row, until they reach the bottom when a readout register transfers the charge. The noise inherent to this readout process is what we define as the readout noise.

To derive our general formula for the readout noise, we first find that:

Our “signal” will be this number of photons multiplied by the quantum efficiency of the detector (incident photon to converted electron ratio) to gather the actual number of processed photons:

From here, we can convert this number of electrons into the number of ADU by dividing by the conversion factor between the number of electrons (e-) recorded by the CCD and the number of digital units ("counts") contained in the CCD image. This is called the camera’s gain.

Moreover, once we have the value, we can derive the photon noise of the system. We know that from Poisson statistics that the noise from the signal itself varies by the square root of the number of realizations. Therefore we can reason that

and finally that

.

This is in units of electrons. To obtain this in units of ADU, we simply divide by the gain:

To find the read noise, you begin by taking 2 (or more) bias images. Subtracting one image from another results in a differential image of the biases. If you take the standard deviation of the differential image on a pixel per pixel basis, this will show you the time-independent source of uncertainty when reading the array; this is the read noise in electrons. We can then multiply by gain to retrieve the read noise in proper units:

To now find how one uses least-squares fitting of a straight line to come up with estimates (and errors) for the read noise and detector gain, we define to be the total noise in terms of recorded electrons. We remember that the different noise sources are independent of each other, so they add in quadrature:

But again, because we know that photon noise obeys the laws of Poissonian statistics, we can rewrite the above equation as:

Knowing how the gain relates units of electrons and counts, we can modify this equation to read as follows:

Which then gives

This is the equation of a line in which is the y-axis, is the x-axis, and the slope is . Solving for the gain directly, we find:

This shows that then to calculate the readout noise, you would find the y-intercept of the mean-variance graph and square-root it. Knowing the gain then allows this to be converted to a readout noise in the standard units of electrons. However, finding the intercept of the line is not the best method, because the readout noise is a relatively small quantity and the exact path where the line passes through the y-axis is subject to much uncertainty when using a least-squares line fit. This is shown quantitatively later in the uncertainty approximations made by the bootstrapping method.

The bootstrapping method is a statistical method that utilizes random sampling with replacement. Bootstrapping the data used in this experiment will pluck only certain data points from the mean-variance plot and will fit a new least-squares line to the chosen data points. This is repeated multiple times to obtain many least-squares lines. This will result in a varying slope and y-intercept to estimate uncertainty, although the values obtained in this lab will be large because the data taken for this experiment only consists of a few points. As will be discussed, the y-intercept will vary by a lot because the bias frame that anchors the y-intercept will not always be chosen by the bootstrapping method. The advantages of this method, however, are proven in its simplicity in deriving error estimates when there is enough data.

**Experiment and Methodology**

**Data Accumulation:**

To collect the data used in this experiment, first the telescope was pointed at the flat field lamp. The flat field lamp was tuned to 50 (even illumination, in theory). The V-band filter was used and the camera chiller was set to -5C. Pixels have different sensitivities to different wavelengths, due to differing photon energies, since they have to excite charge carriers across the band gap. So although the band choice affected the counts per second of the data measured for the flat field frames, it did not affect the bias frames since they had an exposure of zero. The light level was adjusted beforehand so that a thirty second exposure gave about 40,000 counts using the V filter, which is about 2/3 the digital saturation of 65536 ADU. We confirmed that the image was relatively flat for the purposes of this experiment.

Rather than take the frames one-by-one, we took a series of data and customized the filename using the 'Autosave' feature in the 'Camera Control' GUI. First, we took 10 "bias" frames (setting 'Frame Type' to ‘bias’). This set the integration time to 0. We take these initial bias frames to later create a “master bias” which the purpose of has been stated in the “Theory” section. Next, the 'Frame Type' was set to 'FLAT' and then was integrated for 5 seconds. 5 frames were taken in this manner. Next, we collected data for a range of times corresponding to totally dark to nearly saturating the detector. Flat frames were taken for integration times of 5, 10, 20, 30, 40, 50, and 60 seconds. 5 frames were taken for each integration time, with the exception of only 2 frames for 60 seconds as we expected complete camera saturation. For each light level, the mean pixel levels were noted in the lab log sheet. When the frames were finished, the files were transferred to a data folder in the G-Drive.

We collected 5 flat CCD data frames (with the exception of the 60 s frame) at different integration levels to look for variations in photons between the frames. This variation is used to estimate the read noise. The observed noise (variance) will increase as the mean number of counts increases, and we can determine the electronics gain (e-/ADU) and the read-noise (e-) by analyzing this behavior.

**Data Reduction:**

The collected frames were quite large and slow to analyze. Moreover, the "flat field" image is seen to be not entirely flat which can complicate analysis. To combat this, the data to be used was defined by creating a subarray that told the program what part of the chip to read to avoid bad pixels and make the flatfield a true flat fit for analysis. All of the data was loaded into a tuple of arrays. This resulting "list" of arrays was then converted into a 3-dimensional data cube (dimx, dimy, n\_files) for easier analysis. A data cube is optimal for this type of experimentation because it is robust to outliers, such as dead or hot pixels, and averages out noise. The only disadvantage comes in the form of rounding issues that may occur.

**Methodology:**

First, a master bias was created by averaging the ten bias frames together. The noise in this master was theoretically reduced by a factor of The method I chose to analyze the data cube is “Method II”. Method II is to look at the root mean square in the "depth" direction of the cube, which is time. This then gives us an estimate of the mean and standard deviation for each individual pixel. Looping over both the x- and y-dimensions, the master bias was subtracted from the data cube and recorded in a list *datalist*. Within the loop still, for each pixel in the datalist, the standard deviation and the mean were calculated and parsed into their new respective lists. This loop was run for each specified datapath, namely, the fits for the bias frame and flats 5-60, by calling the loop function *parse* and specifying the datapath as a parameter of the loop function. The returned lists were set equal to their new lists, *rms\_list[t]*, *mean\_list[t]*, depending on the integration time. Next, the mean of each flat field and the bias frame were calculated by taking the mean of the corresponding *mean\_list[t]*. The *rms\_list[t]* of each frame was squared to create individual *var\_list[t]*s, and then the variance of each frame was calculated by taking the mean of the corresponding *var\_list[t]*s. Finally, each mean and variance value were added to a list containing all the means, *mean\_t*, and all of the variances, *var\_t*, to be plotted against each other.

Next, two plots were created and the least-squares fit line was determined using NumPy’s *polyfit()* class method. The fit was determined excluding both the flat\_50 and flat\_60 values due to their saturated values to produce the best fit for the unsaturated data. The line fit was produced with a weight equal to that of one over the total error squared to account for the error of each data point. (The error for each data point was calculated by taking the standard deviation of each *var\_list[t]* and dividing it by the square root of the length of the *var\_list[t]*. Each error corresponding to each integration time was then added to a new list called *err\_t,* or the total error, which was the value used to weight the *polyfit()* fit). Both a slope and a y-intercept were extracted from the *polyfit()* method, and was plotted against the mean and variance of each frame.

Finally, the error bars were calculated (again excluding both the flat\_50 and flat\_60 values) with the *errorbar()* method using the list of the mean of each frame (*mean\_t*), the list of the variance of each frame (*var\_t*), and the y-axis error set to the list of the error of each frame (*err\_t*).

**Results:**

Mean-Variance Plot ‘Reduced’ Data

|  |  |  |  |
| --- | --- | --- | --- |
| *Exposure Time* | *Mean* | *Variance* | *Error* |
| 0 | -1.61435E-16 | 442.38424 | 2.08960 |
| 5 | 7148.65108 | 14842.39295 | 104.99945 |
| 10 | 14407.69412 | 29660.01814 | 211.9070 |
| 20 | 28934.00424 | 58968.04461 | 416.29764 |
| 30 | 43320.64042 | 86990.27287 | 618.54091 |
| 40 | 57536.26472 | 112738.21446 | 799.48830 |
| 50 | 64555.76394 | *saturated* | *saturated* |
| 60 | 64555.76394 | *saturated* | *saturated* |

Table 1

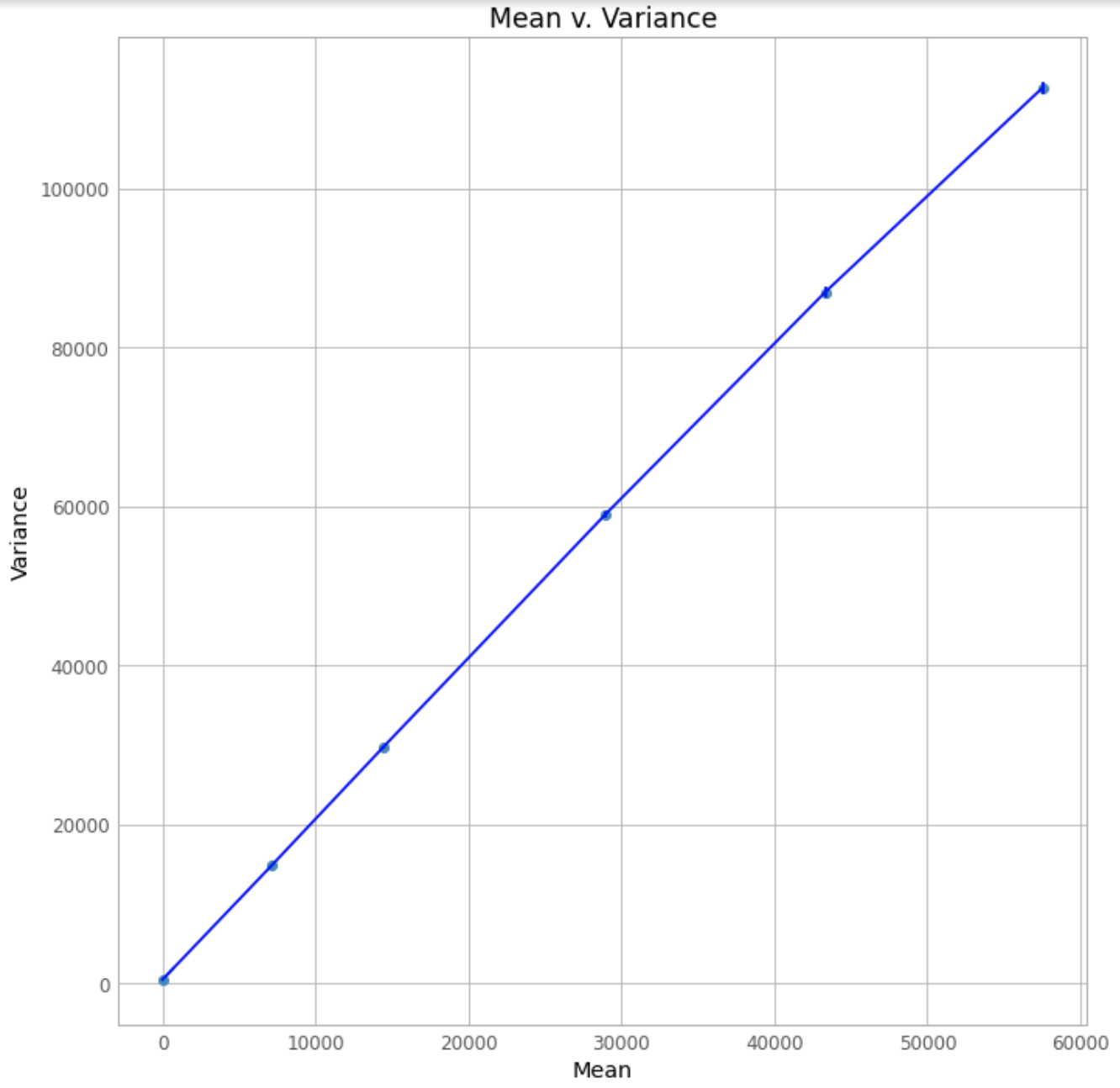
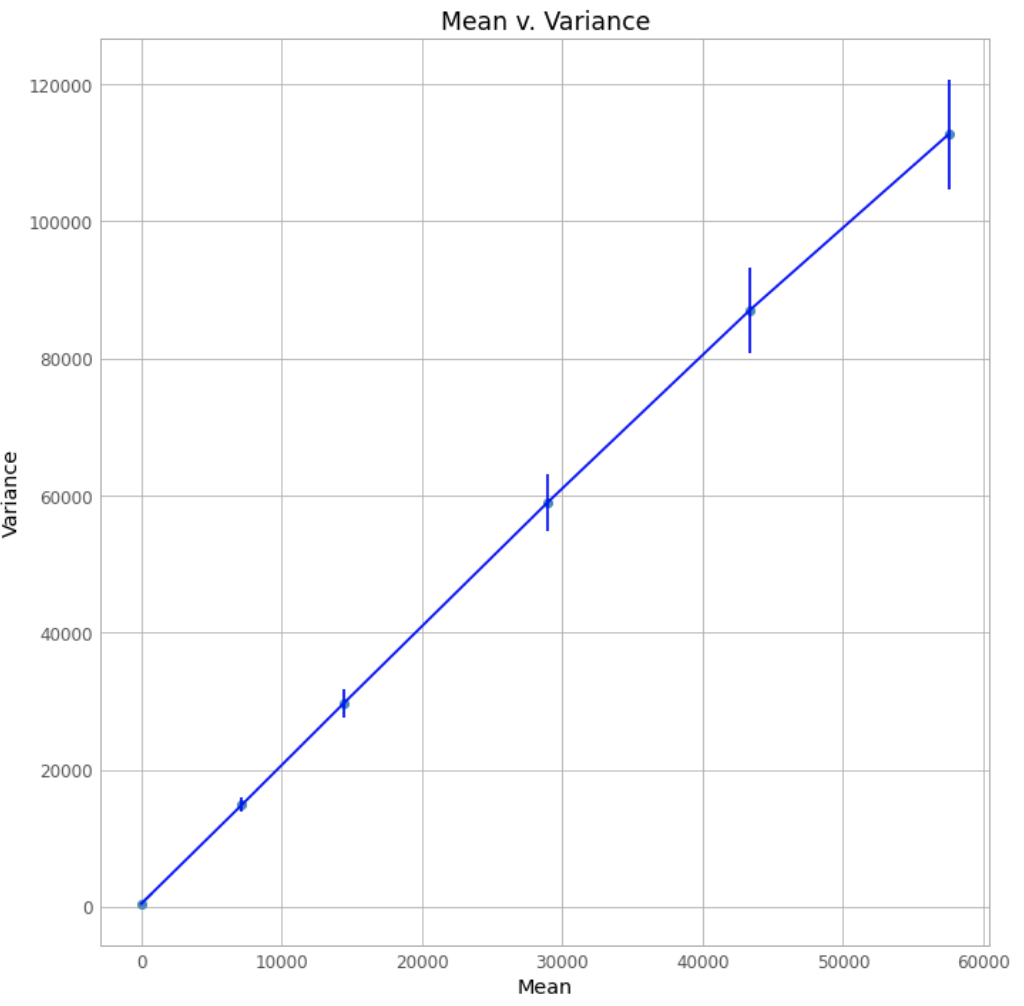


Figure 1 Figure 2

*The mean and variance (ADU) of each frame are plotted against each other in these figures. Error bars grow as the mean does. The first point represents the bias frame. As integration time increases, so do the mean and variance proportionally as seen. Flat\_50 and flat\_60 are excluded so we observe a linear relationship. The only difference between Figure 1 and Figure 2 is that the error bars are multiplied by 10 in Figure 1 so that they can be better viewed.*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

Table 2

These uncertainties were calculated by setting the *polyfit()* ‘cov’ equal to ‘True’. “If given and not False, [‘cov’ returns] not just the estimate [of slope and y-intercept,] but also its covariance matrix. By default, the covariance are scaled by chi2/sqrt(N-dof), i.e., the weights are presumed to be unreliable except in a relative sense and everything is scaled such that the reduced chi2 is unity” [1]. The uncertainty values on the slope and y-intercept fit, however, were also more crudely calculated using the bootstrap method.

A loop was created to resample the data based on the estimates of the mean and sigma. *bootstrap\_x* is a NumPy array that was filled by choosing random indexes from the list of mean ADU values from each flat frame (*mean\_t*). Similarly, *bootstrap\_y* is a NumPy array that was filled by choosing random indexes from the list of variance ADU values from each flat frame (*var\_t*). For each resulting bootstrap sample, a new line was fit to the resampled data to calculate the slope and y-intercept uncertainty. The results are as follows:

|  |  |
| --- | --- |
|  |  |
|  |  |

Table 3

It is understood that the bootstrapping method is suboptimal for this type of data analysis as the number of data points is relatively small. The y-intercept is seen to vary so largely because when the data is re-sampled, the value that anchors the y-intercept (the bias frame mean and variance) is only sometimes selected which produces much variation. Knowing this, the Monte-Carlo method would have produced more accurate results.

**Interpretation:**

Gain:

As determined in the Theory section of this report, the gain is equal to the positive inverse of the slope of the mean-variance plot presented above.

The uncertainty here was calculated by the *polyfit()* method. This number means that the camera digitizes the CCD signal so that each ADU corresponds to ~ photoelectrons.

Read Noise:

The read noise we know is the square root of the y-intercept of the mean-variance plot presented above as derived in the Theory section of this report.

The uncertainty here was again calculated by the *polyfit()* method.

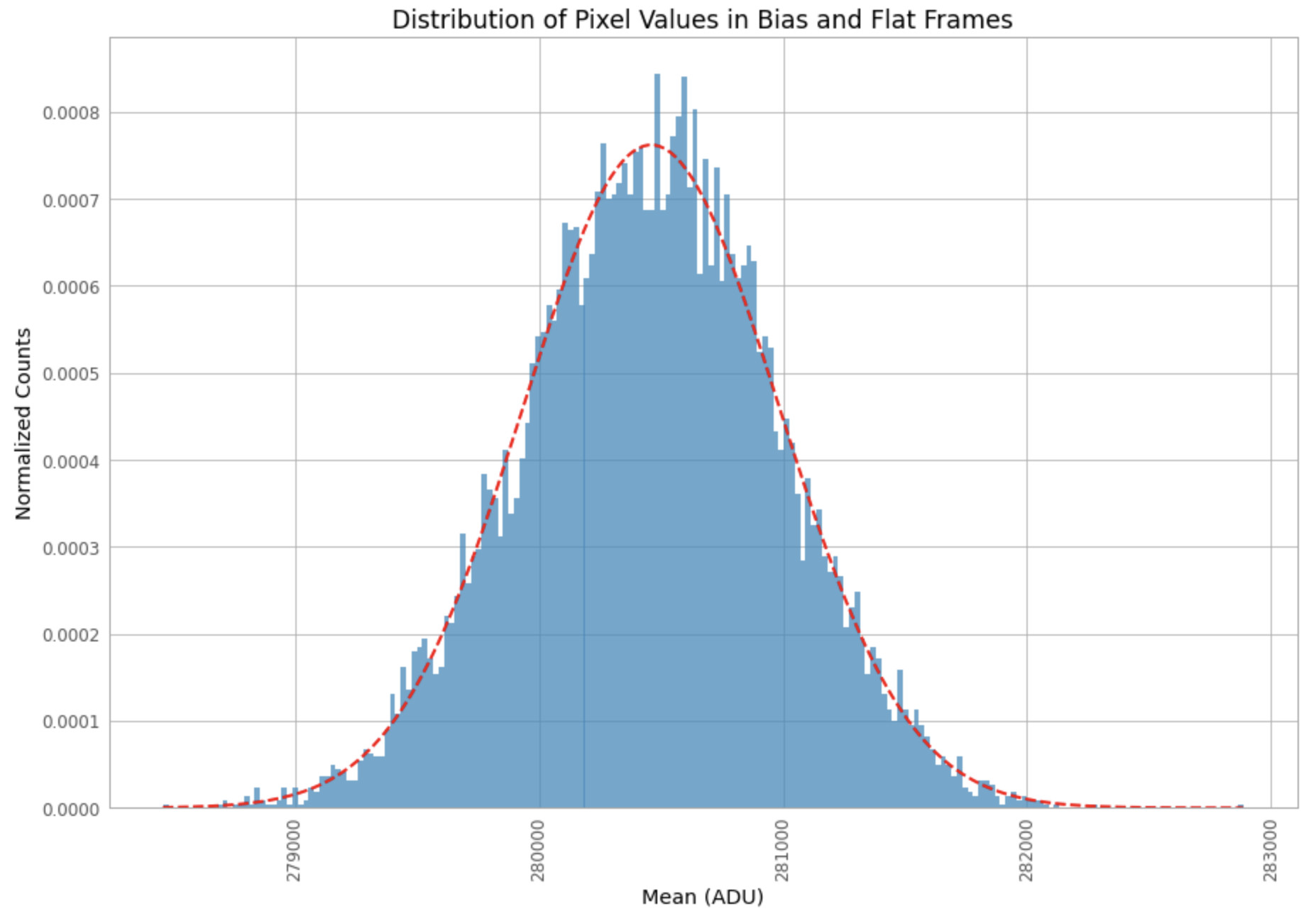


Figure 3

*Above is a histogram plot showing the distribution of pixel values seen in both the bias and flat frames. The dashed red fit is a Gaussian distribution. Both the data and the fit are normalized. We observe a mu value of 280458.78246 ADU and sigma of 523.47688 ADU.*

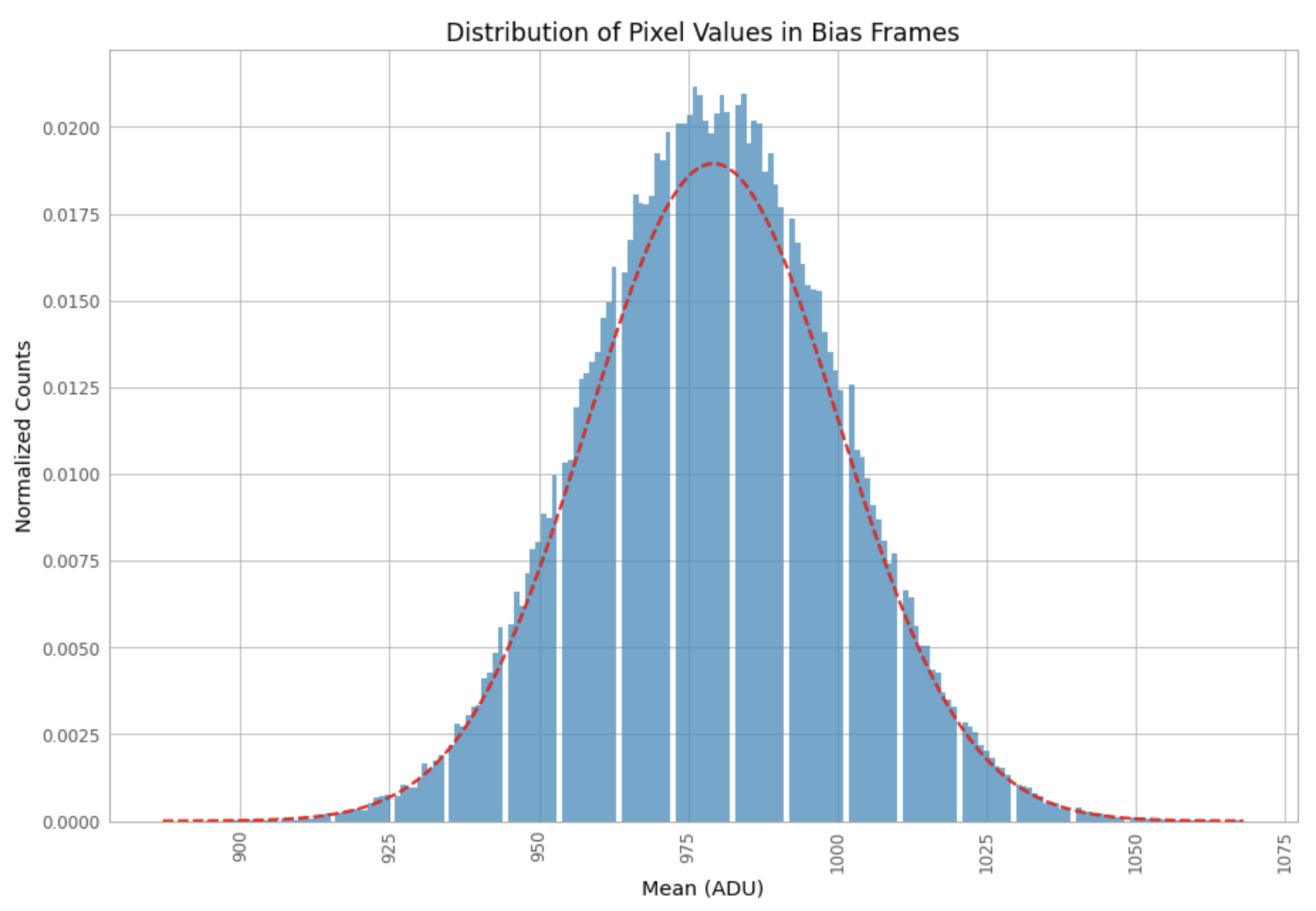


Figure 4

*Above is a histogram plot showing the distribution of pixel values seen in the bias frames only. The dashed red fit is a Gaussian distribution. Both the data and the fit are normalized. The mean in the bias frames seems to be centered around about 975 ADU. We note that this is lower than the value seen for the flat values.*

As observed in *Figure 3* and *Figure 4* above, the Gaussian distribution fits the distribution of pixel values in the frames very well, as expected.

Linearity:

Next, the counts versus integration time was plotted to determine the light level at which the detector will saturate and to quantify the deviation from linearity.

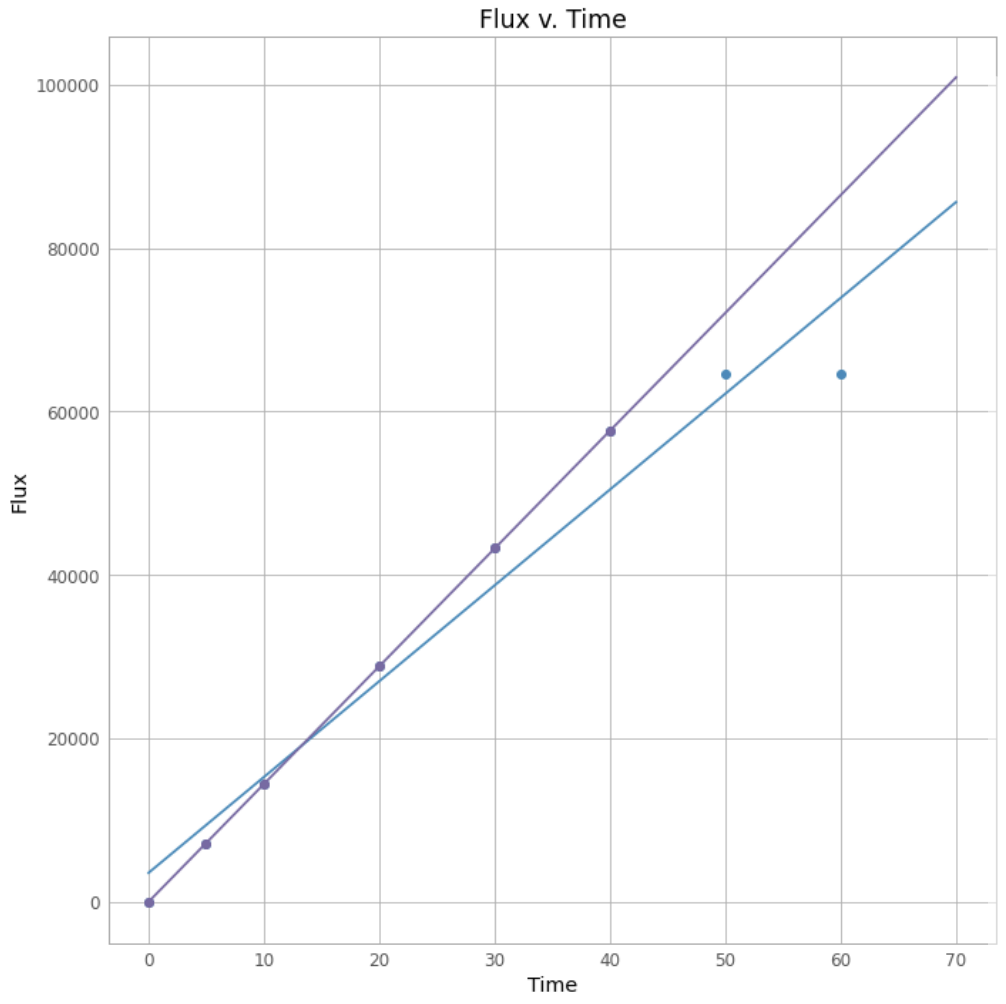


Figure 5

*The mean counts (ADU) of each frame are plotted against the integration time in seconds of the frame in this figure. The purple line represents the fit excluding the saturated values, while the blue line represents the fit including the saturated values.*

The *polyfit()* slope and y-intercept calculations for each fit are shown below:

|  |  |
| --- | --- |
|  |  |
|  |  |

Table 4

As is obvious from a quick visual analysis of the flux versus time graph fits, there is a large detectable deviation from linearity for the flat\_50 and flat\_60 data points. This is because the detector saturation occurs at an integration time around 50 seconds; this happens where the signal that needs to be measured is larger than the dynamic range of the sensor. When this happens, the output of the sensor becomes the limiting value of the sensor range and the graph of flux versus time will asymptote at the saturation point as seen in *Figure 5*.

The non-linearity is quantifiable by calculating the percent difference between saturated and unsaturated slopes:

This large percentage can be interpreted to mean that there is a significant decrease in linearity about the saturation point. Moreover, it is seen in *Figure 6* how there is detectable nonlinearity in the unsaturated data points as well. The points that fit well within the unsaturated data fit from above were further tested to show their own nonlinearity:

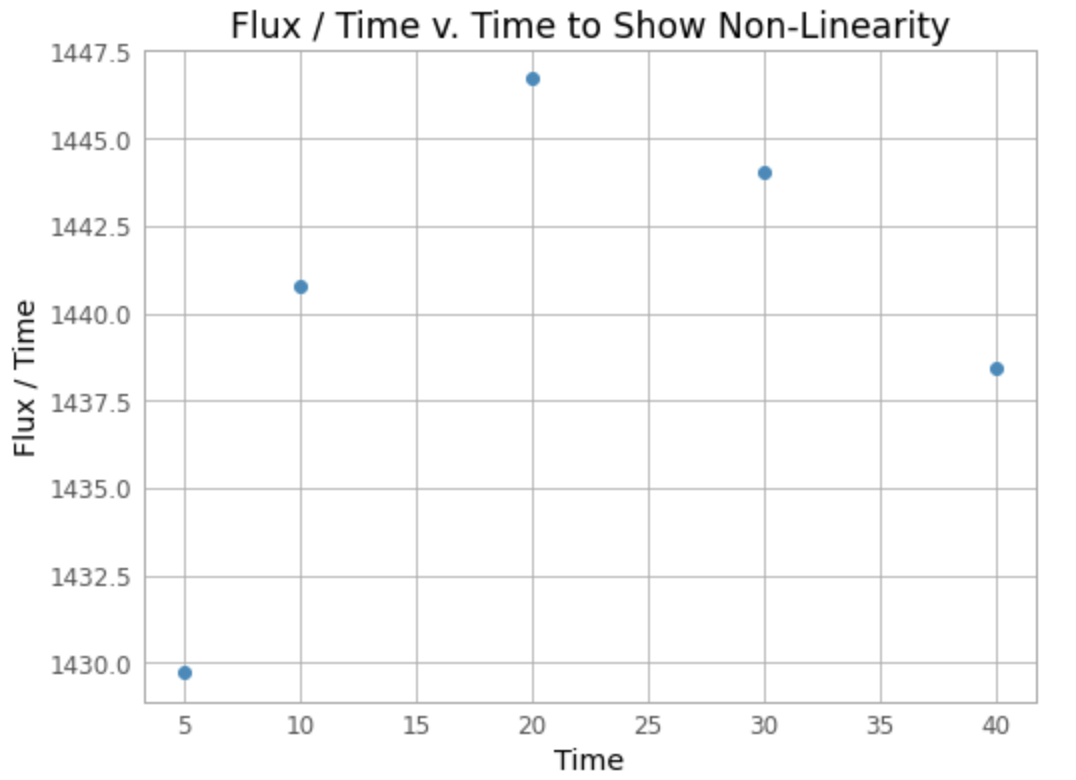


Figure 6

*The flat frames for integration times 5-40 seconds are shown plotted against the amount of flux per time the data reads.*

The flat\_5 through flat\_20 data is seen with a photon flux per time of increasing value, mostly linearly (positively correlated, although concave trajectory). However, the flat\_30 data point and on show a decreasing trend that deviates from the otherwise mostly linear trend when we would expect it to continue to rise positively.

We see that between an integration time of 5 seconds and 10 seconds, there is an increase of flux/time of approximately. However between 10 and 20 seconds, although we would expect twice the increase, only a jump of is seen. Then, between 20 and 30 seconds, a decrease of , and then a more dramatic decrease ofis seen between 30 and 40 seconds. CCDs are not perfectly linear systems. After bias subtraction, the number of ADU counts is not exactly proportional to the number of incident photons as shown in *Figure 6*. The reason for this nonlinearity is that the controller gain is not a constant function of the number of electrons. When the CCD approaches saturation, there is a much larger nonlinearity due to the reduced probability of capturing photons in nearly full pixels [3]. This deviation can also be seen in the *Table 1* Error section as we note that for increased integration time the program reports increasingly large error. This is further quantified in *Table 5* (below) as seen in the negative Data-Model values for flat\_30 and flat\_40, meaning they fall further and further below the theoretical fit.

Chi-Squared Interpretation:

The chi-squared statistic is a single number that tells you how much difference exists between your observed counts and the counts you would expect if there were no relationship at all in the population. A low value for chi-square means there is a high correlation between the two sets of data. Conversely, a high value indicates that there is a poorly fit model or the error bars have been grossly underestimated (or both). We expect, for this experiment, a chi-square value of about 1. This is to say that the model prediction is generally well within the error bars of the data.

First thedata-model data was calculated by subtracting the weighted line model’s theoretical prediction (as described in the Methodology section) from the actual (mean, var) pair of the specified frame. Then the individual chi-squared numbers were calculated:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Bias | flat\_5 | flat\_10 | flat\_20 | flat\_30 | flat\_40 |
| Data-Model | 3.01618E-07 | -11.35505 | 172.36108 | 195.99012 | -784.60257 | -3694.72956 |
| Chi-Squared | 2.08346E-14 | 0.01170 | 0.66159 | 0.22165 | 1.60902 | 21.35704 |

Table 5

These individual values in *Table 5* were taken together to calculate the total chi-squared value:

Then, the reduced chi-squared (or chi-squared per degree of freedom) value was calculated as follows:

where the denominator is the number of data points minus the number of parameters in the model. In this case, the number of data points was 6 and the number of parameters was 2. These methods produced the following:

|  |  |
| --- | --- |
| Chi-Squared | Reduced Chi-Squared |
| 23.86099 | 11.93050 |

Table 6

Despite an expectation of ~1, we receive a reduced chi-squared value of ~11.93050. The reason for this large of a value can be inferred from the nonlinearity that occurs specifically for the flat\_30 and flat\_40 frames; the data begins to fall under the model’s predictions and therefore increases the chi-square value.

**Summary and Conclusion**

In conclusion, we find that the use of the mean-variance Method II is sufficient in estimating the CCD gain and detector read noise with minimal noise. Although the bootstrapping method estimates a large uncertainty, we find that this is because this method of resampling data is suboptimal for this type of experiment due to the small number of data points used for analytics. We conclude through mean-variance analysis that the gain is the positive inverse of the least-squares line fit slope, and that the read noise is the square root of the y-intercept. We find that electronic gain is approximately and that the detector read noise is when multiplied by the gain. The professional MDM4K detector, in comparison, has “... a gain of 2.2-2.4 electrons per ADU and a read noise of 5 electrons (rms)” [2]. Although these four readout amplifiers have about two times a smaller read noise than the Angell Hall Observatory CCD, they have over four times the electronic gain.

We find through visual analysis of the error bars that the frame’s error increases with increasing integration time, as expected. We also can observe that both the histogram plot showing the distribution of pixel values seen in the bias frames only, and the plot showing the distribution of pixel values seen in the bias frames and the flat frames follows a Gaussian distribution curve quite well.

We also find through visual analysis of the flux versus integration time graph that the camera saturates at approximately fifty seconds. We conclude this implies that an integration time of about fifty seconds means that the signal that needs to be measured is larger than the dynamic range of the sensor, and so we observe an asymptotic convergence at this time. Furthermore, there is detectable nonlinearity within the saturated values themselves because the controller gain is not a constant function of the number of electrons. As the integration time approaches 30 and 40 seconds, there is larger and larger nonlinearity due to the reduced probability of capturing photons in nearly full pixels. The chi-squared statistic proves that this nonlinearity affects the validity of the fit, as we receive a value of approximately ~11.93050 as opposed to the preferred value of ~1.

In conclusion, it is important to understand the CCD characteristics because although a CCD can make accurate measurements of the charge accumulated in each CCD pixel, it is not perfect due to a few sources of noise. Understanding the noise and accounting for it experimentally will help with the correct interpretation of data collected using the CCD. We conclude that the Method II procedure is fairly accurate in estimating CCD characteristics by looking at the root mean square data of all the bias and flat frames in the "depth" direction of the cube, which is time, and plotting the mean versus variance accordingly.

Works Cited

[1] “Numpy.polyfit.” *Numpy.polyfit - NumPy v1.19 Manual*, numpy.org/doc/stable/reference/generated/numpy.polyfit.html.

[2] “MDM Blue 4Kx4K CCD.” *OSU Blue 4K CCD Imager*, www.astronomy.ohio-state.edu/MDM/MDM4K/.

[3] Baldry, Ivan Karim. “Time-Series Spectroscopy of Pulsating Stars.” *University of Sydney*, University of Sydney, 1999, p. 21.