DISTRIBUTION OF STARS AND DUST WITHIN NGC 3576 EMISSION NEBULA

By Samwel Amani Njoroge

Abstract

Assessing certain regions of OB stars and dust within the *NGC 3576* Nebula from three separate fit file images of Hydrogen Alpha (H_α), Oxygen III $(O\ III)$ and Sulphur II $(S\ II)$ in relation to the photon intensity within each pixel. Intensity of pixels within the images are split into 100 different levels for assessment of each layer. Gaussian fitting is used to get readings on H_α temperatures. A linear relationship between levels and H_α temperatures is used to extrapolate temperate for dust. Distribution coefficient of total Dust in the image is 0.199. Surface temperatures of OB and M stars found to be $(18300 \pm 400)\ K\ (39\%\ error)$ and $(2720 \pm 60)\ K\ (9.\overline{33}\%\ error)$, respectively.

Temperatures reradiated from Sulphur and Oxygen dust are in the range $T \geq (5400 \pm 200)\ K$ and $T \geq (12500 \pm 500) K$, respectively.

ST. NO: 4060924

12/08/2022
PHY327 Main Project

Table of Contents

Introduction

Nebulae are known as giant clouds of dust and gas in outer space that can span to many light years in relative diameter. They are structures that can originate from the explosion of a dying star and are able to repopulate the universe with younger and hotter stars. In 1834, John Frederick William Herschel discovered a gas cloud, and it was officially named as NGC 3567. (Bot, 2018). It is widely known as the Statue of Liberty Nebula due to its famous resemblance to the statue in New York. (See **Image 1**).

Source: https://astrodonimaging.com/gallery/statue-of-liberty-nebula/

Image 1: Optical image of the Statue of Liberty Nebula (*NGC 3576*)

It is an emission nebula found within the constellation of Carina with a distance of 2.5 kiloparsecs from earth. The emission from this bundle of gas and dust comes from the surplus of Hydrogen ions that give Hydrogen Alpha (H_α) emission. This signifies that there are a multitude of young OB stars that are present in order to produce this substantial number of photons that reach earth.

Star formation is due to the property of mass and the force of gravity that matter exerts on each other. This force is directly proportional to the mass of the object. The fact that hydrogen alpha is already present indicates that certain regions of the gas cloud was heavier than Jeans

Mass meaning the mass was great enough for gravity to overpower any form of pressure and temperature to cause a gravitational collapse. That foretells the total mass of *NGC 3576* exceeding that of one hundred solar masses. It is known that all stars must be undergoing fusion in their core, and so that means the core's temperature must be about 10 - 15 million kelvins. (Newman & Whitlock). However, the research of interest here is the surface temperature of the young OB stars as well as the dust that encompasses them (if possible).

An observatory in Chile referred to as 'ObsTech' managed to capture an image of NGC 3567 and had given public access of the data in fit files. The data from this site was stored within three separate fit files with each fit file containing two-dimensional data. This is because each fit file pertains to data that is already filtered with regards to H_α, O III (2nd ionization of Oxygen) and S II (1st ionization of Sulphur). All these emission spectra are within the Near IR spectrum and are given in greyscale.

The focus of investigation will mostly be on H_α data. H_α is known as an emission line in the Balmer series when an electron emits a photon of energy as it drops form energy level 3 to 2. This photon of energy corresponds to the wavelength of 656.3 nm. This wavelength of energy is allowed to pass through the H_α filter and be captured by the telescope. Data is given as a matrix of photons per pixel in a $(400, 400)$ pixel matrix. (See **Image 2**)

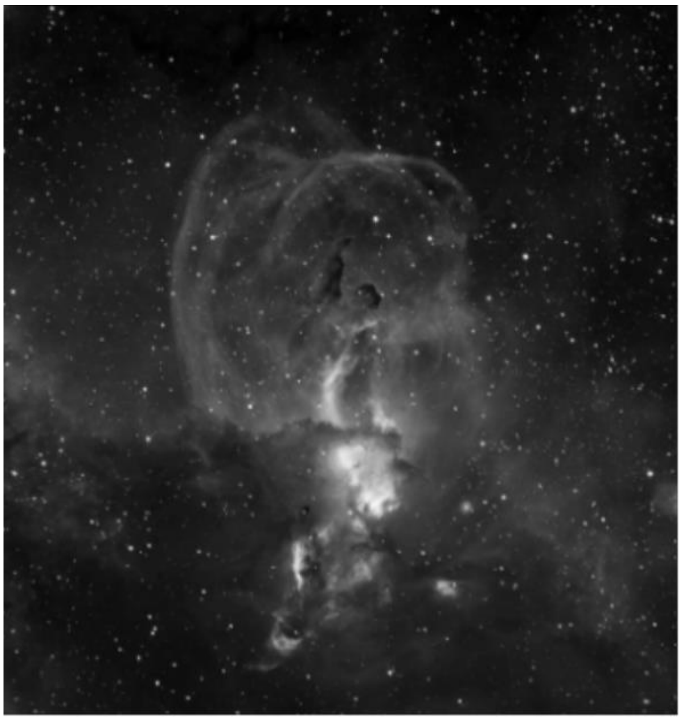


Image 2: Unfiltered $\boldsymbol{H_\alpha}$ pixel matrix data

Data analysis was conducted with written code (python based) and has been publicly posted on my GitHub Account[1]. Pixel intensity can be regarded as the number of photons per pixel

[1] https://github.com/Amani5576/Nebula_Intensity.git

within the matrix. The code is essentially dependent on user input. The user is guided through their input to produce a filtered matrix of their desired pixel intensity (or intensities).

For discussion, the user defined levels of splitting the data will be of 100 levels of intensity with the highest intensity labelled as Level 1. Level 1 refers to the hottest regions of the matrix where stars exist. The lowest regions of intensity (with Level value 100) will give insights of regions that are being covered with dust or lacking in star formation or the inexistence of young stars.

The header of the fit file did not give information about Right Ascension (RA) or Declination (DE) and so referring to a specific location in the matrix (if necessary) will be given in regard to rows and columns. Additionally, there was no flux information given within the fit header information. Notwithstanding, Weins law can be used to find an estimation of Temperature readings.

Weins law states that the spectral radiance of a black body peaks at particular wavelength. (see eq.1 below).

$$T\ (in\ K) = \frac{2\ 897\ 771.955\ nm \cdot K}{\lambda_{peak}\ (in\ nm)} \tag{1}$$

Eq. 1 represents an inverse proportionality between Temperature and Peak wavelength (λ_{peak}) of a given black body curve. This will assist in acquiring temperature readings.

Gaussian Fitting will be used in the rough approximation of the peak wavelength that can be used to find temperature. The gaussian function is a particularly useful statistical function that can be used for approximation. (Weisstein, 2022). The Gaussian equation is represented as eq. 2, below.

$$f(x) = A \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2}$$

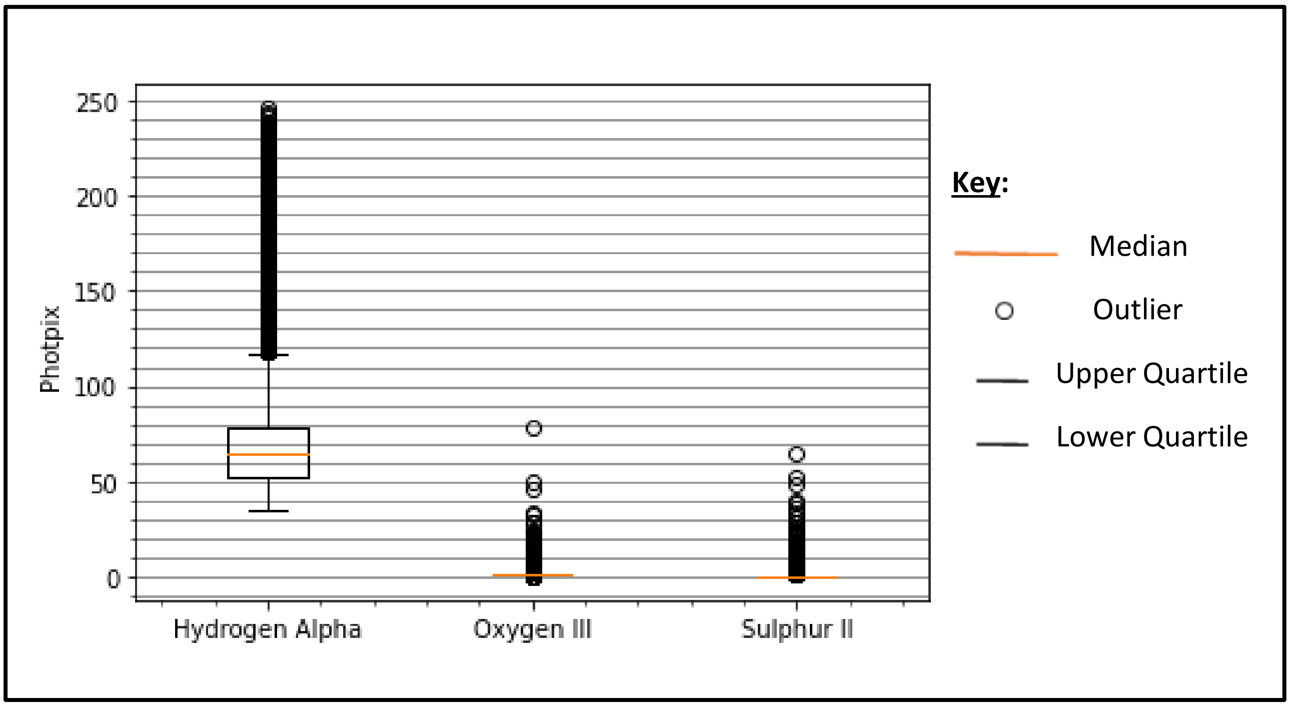
$$\begin{aligned} &\mu = Mean\ of\ the\ Gaussian\ (Centre\ of\ bell\ curve) \\ &\sigma = standard\ deviation \\ &A = Amplitude\ of\ the\ bell\ curve \end{aligned}$$

Throughout the report the term **Photpix** will be used as a proxy for the concept of photons per pixel within the three different data matrices.

Results and Discussions

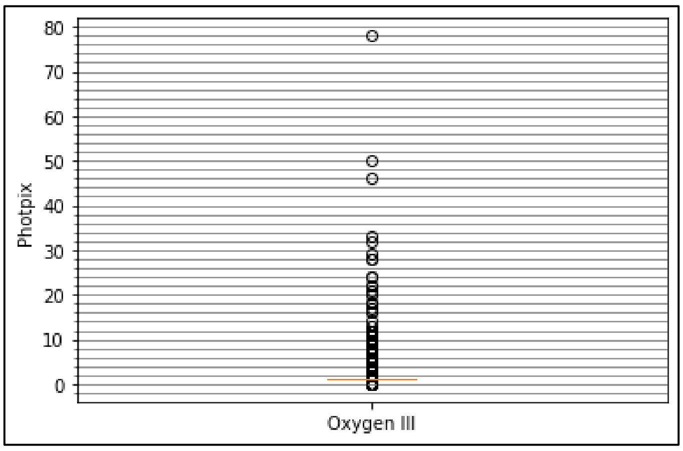
Emission	λ (nm)	Max Photpix	Min Photpix	Photpix Upper quartile	Median Photpix	Photpix Lower quartile	Modal Photpix	Mean Photpix	Standard Deviation (3 d.p.)
H_α	656.3	246	35	78	64	52	49	68.660	23.376
$O\ III$	495.9	78	0	1	1	1	1	1.014	0.402
$S\ II$	671.6	65	0	0	0	0	0	0.216	0.624

Table 1: Statistical Data on Hydrogen Alpha, Oxygen 3, and Sulphur 2 fit files after extraction

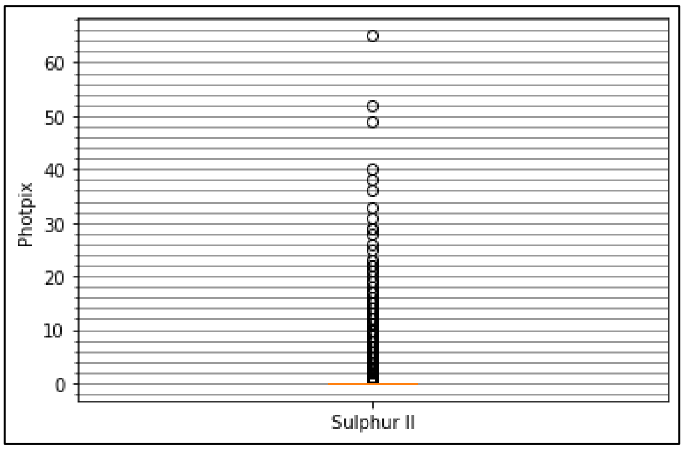


Graph 1: Quartile Data on the Photons per pixel of each image with regards to its filter

Graph 1 is a representation of the data given in **Table 1**. It represents the interquartile range of the three separate fit files in terms of the photpix. It can be seen that H_α image has a huge distribution of different photpix in comparison to the others, with its median photpix of 64. The outliers of H_α are reasonable based on the standard deviation in **Table 1** given as 23.376. The code written to produce this output Graph was considering all information which limits getting further insights on $OIII$ and SII interquartile range data. These two other data sets were drawn separately to see more coherent interquartile range data. (See **Graph 2** and **Graph 3**)



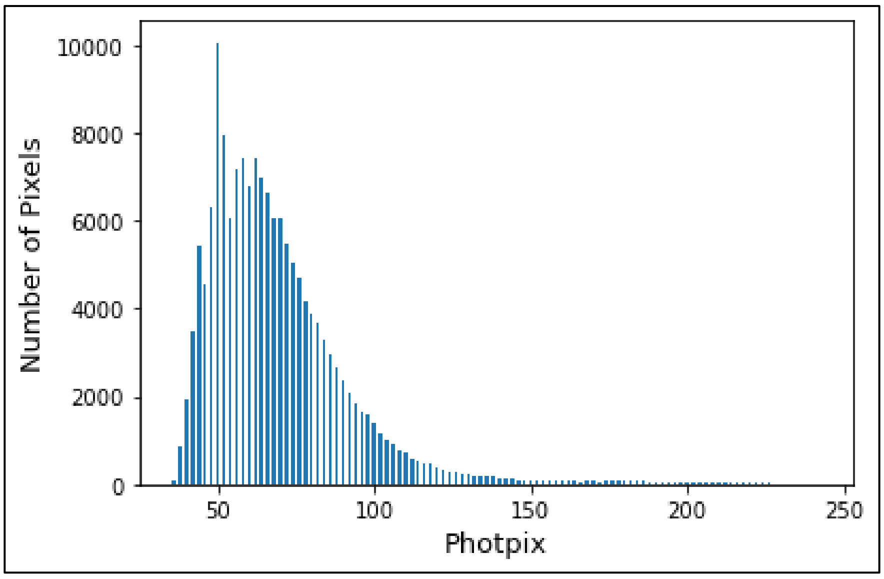
Graph 3: Interquartile Data for $\boldsymbol{O\ III}$



Graph 2: Interquartile Data for $\boldsymbol{S\ II}$

Graph two and **Graph 3** have similar characteristics as per the previous **Graph 1** because even if the interquartile ranges are no longer being obscured by H_α data, they still are on one level. This can be argued by the mean, upper quartile and lower quartile data represented in **Table 1**. For $O\ III$ and $S\ II$, most elements in the matrices is 1 and 0, respectively.

The standard deviation for these two fit files strongly suggests this to be true as the dispersion of the data away from the mean is close to 0. This reveals that not much intensity is coming from heavier elements where there is dust. Another argument would be that there was not enough exposure for more photons of light to be collected by the filtered telescope.



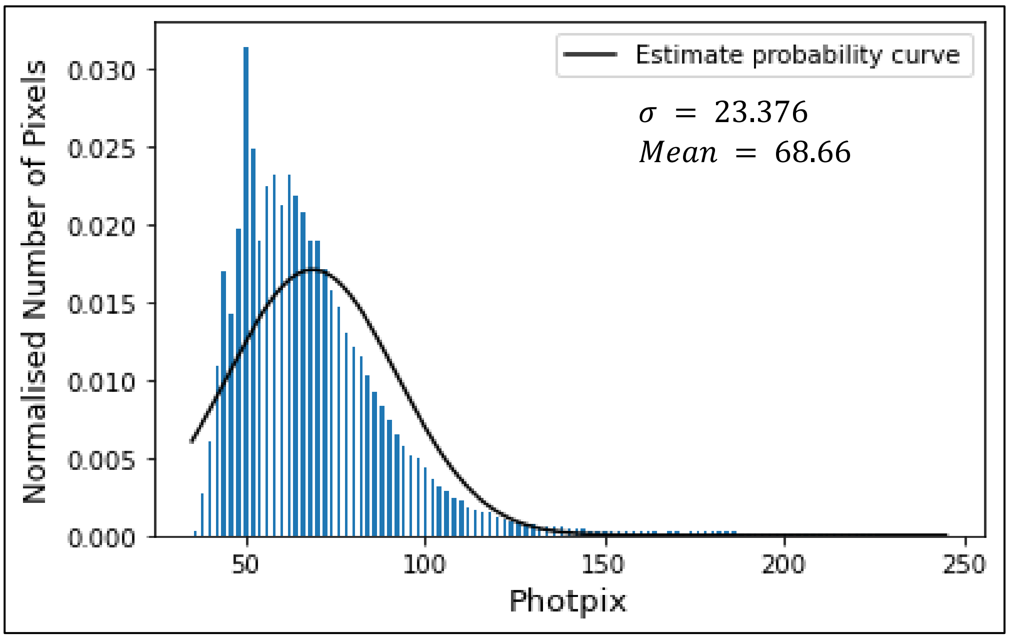
Graph 4: Histogram data of the Photpix of Hydrogen Alpha (Bin Width of 2)

However, there are outliers that are more easily identifiable. **Graph 3** has more outliers than that of **Graph 2**. This shows that there are more regions in the Sulphur matrix with diverse photpix values than that of Oxygen. This can also be proven by the comparison of their standard deviation with Sulphur having a higher deviation by 0.222. It means that *NGC 3576* has more probability of reddening of young OB stars due to Sulphur dust than that of Oxygen dust.

Graph 4 shows the number of pixels that have a specific photpix value. The graph embodies similar characteristics to that of a black body radiation curve. However, there are differences such as a rugged peak, or more importantly the x-axis representing intensity of photons rather than the Temperature. However, since Weins law only looks at the peak wavelength, it is only the peak of this graph that is of importance. “Wein’s… law states that the product of maximum wavelength corresponds to maximum intensity…” (Vedantu, 2022). Note that **Table 1** gives $\lambda_{H_\alpha} = 656.3\ nm$. By using eq. 1:

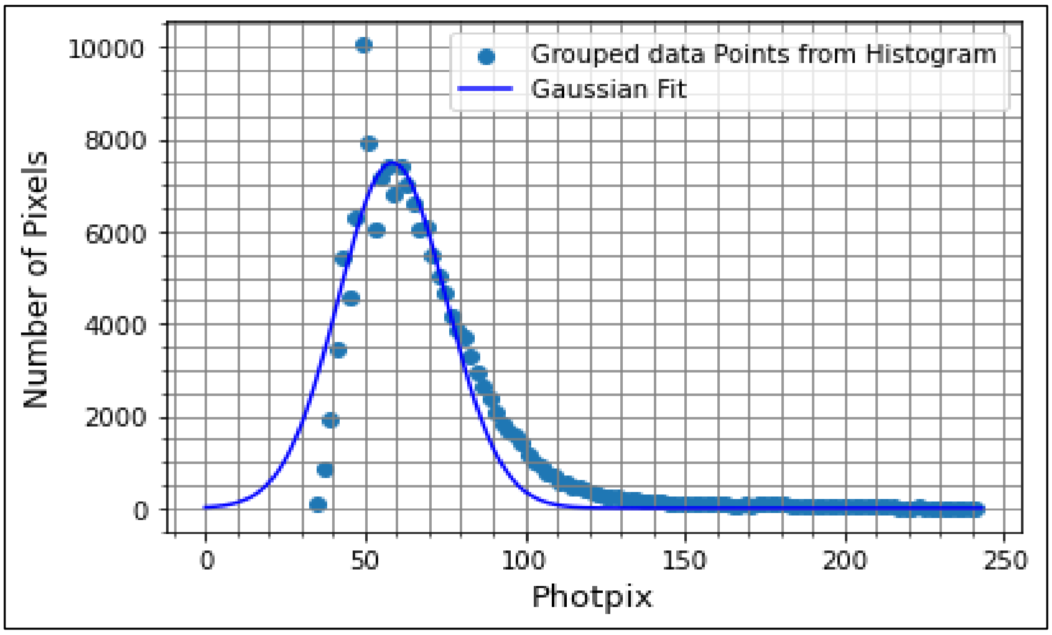
$$T\ (in\ K) = \frac{2\,897\,771.955\ nm \cdot K}{656.3\ (in\ nm)} = \ 4\,415.444\ K\ (3\ d.p)$$

This temperature reading is of the peak wavelength of the graph. However, the peak is not well defined. The first attempt of getting a smooth peak was by fitting a probability density curve (See **Graph 5**).

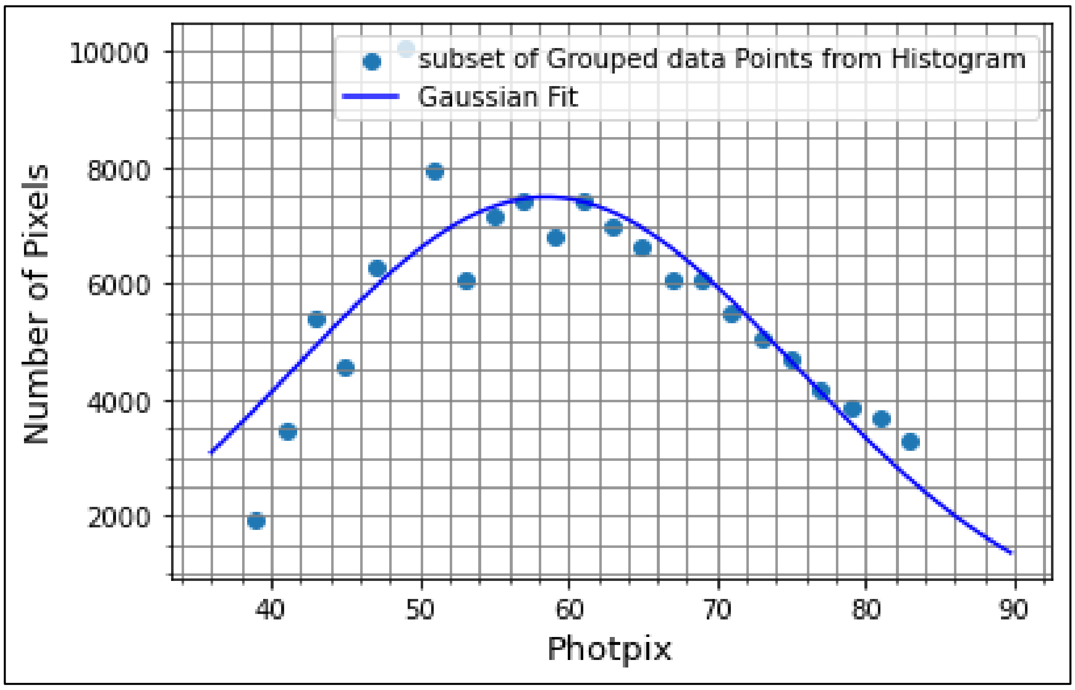


Graph 5: Fitting of Probability Density curve onto the graph

The problem with the fitting is the lack of alignment of the smooth peak (in black) with that of the rough peak (from the histogram). The second approach was by fitting a Gaussian that would produce a better approximation of the true peak. Doing so reaped a good estimate of the peak Photpix. This enables one to find a relation between Photpix values with temperature.



Graph 7: Fitting a Gaussian to the Bar Graph



Graph 7: Subset that was taken for Gaussian Fitting

Graph 6 has data points (in light blue) that depict midpoints of the top of the bars seen in **Graph 5**. The dark blue line is the superimposed Gaussian function. Certain guesses were used in the parameters of μ, σ and A (see Appendix C) for a good estimation to be approximated to the data points. **Graph 7** is a zoomed in version of **Graph 6**. As seen in **Graph 6**, through use of a Gaussian fit, there now lies a smooth peak. The approximated parameter for the mean was found to be:

$$Mean\ Photpix\ (Gaussian) = \ \mu = 59 \pm 1\ (3.d.p)$$

By restating eq.2 the gaussian fit was found to be of the following function:

$$f(x) = (7\ 488.798\ ) \cdot \exp\left(-\frac{(x-59)^2}{2(24)^2}\right)$$

$$f(Photpix) = (7\ 488.798\ ) \cdot \exp\left(-\frac{(Photpix-59)^2}{1\ 152}\right)$$

$$\therefore Number\ of\ Pixels = (7\ 488.798\ ) \cdot \exp\left(-\frac{(Photpix-59)^2}{1\ 152}\right)$$

This function is only true for most data points found in **Graph 7**, however for other data points (as seen in **Graph 6**) there starts to be a deviation whereby the gradient is increasing faster before reaching the peak and, after the peak, it starts to decrease faster when moving towards larger photpix values. Statistically speaking, the uncertainties will also be varying according to this deviation (; more so for larger values).

The relationship between a Photpix and Temperature can be assumed to be linear in order to give other Photpix values their own temperature readings. The uncertainty in the mean Photpix value (from the peak) is accounted for when calculating uncertainty of temperature readings. (See **Table 2** and **Table 3**)

Photpix	*No. of Pixels*	*Temp(K)*	*δ Temp (K)*
36	97	2 720	60
38	879	2 870	60
40	1 942	3 020	60
42	3 478	3 170	60
44	5 415	3 320	70
46	4 558	3 470	70
48	6 297	3 620	70
50	10 050	3 770	80
52	7 938	3 920	80
54	6 044	4 080	80
56	7 160	4 230	90
58	7 424	4 380	90
60	6 784	4 530	90

Table 2: H_α photpix values and their corresponding Temperatures from the ratio acquired by fit. (Lowest Temperatures)

Photpix	*No. of Pixels*	*Temp(K)*	*δ Temp (K)*
218	18	16 500	300
220	18	16 600	300
222	19	16 800	300
224	25	16 900	300
226	17	17 100	300
228	10	17 200	400
230	12	17 400	400
232	4	17 500	400
234	4	17 700	400
236	2	17 800	400
238	2	18 000	400
240	3	18 100	400
242	2	18 300	400

Table 3: $\boldsymbol{H_\alpha}$ photpix values and their corresponding Temperatures from the ratio acquired by fit. (Highest Temperatures)

Table 2 and **Table 3** shows the relative photon intensities for the H_α matrix data as well as the number of pixels that have that average pixel value. There is also the temperatures with their uncertainties from the decisive assumption of linearity between temperature and photon intensity. **Table 2** features the lowest temperature readings while **Table 3** features the highest temperatures.

Table 2 has the low uncertainty values of temperature compared to that of the **Table 3**. This is because of the way in which data deviates away from the gaussian fitting leading to higher uncertainties when reaching higher values. However, when it comes to statistics, lower values reap lower uncertainties. This is because uncertainty from the Peak Photpix was converted to percentage error. This error was applied to lower and higher Photpix values leading to direct proportionality between Temperatures and their uncertainties.

There are no odd values of Photpix values due to using histogram data from **Graph 4**. The average of the bin width was used and rounded off because photons cannot be regarded as a fraction of a packet of energy. It would go against the laws of quantisation of energy.

The same gaussian fitting could not have been done for the other fit files due to the lack of sufficient data distribution to reproduce the depiction of a black body radiation which was confirmed by **Graph 2** and **Graph 3**.

Further tabulated results of Photpix values and their relative temperatures can be seen in Appendix A. All tabulated information of temperature can be seen by the **Chart 1**.

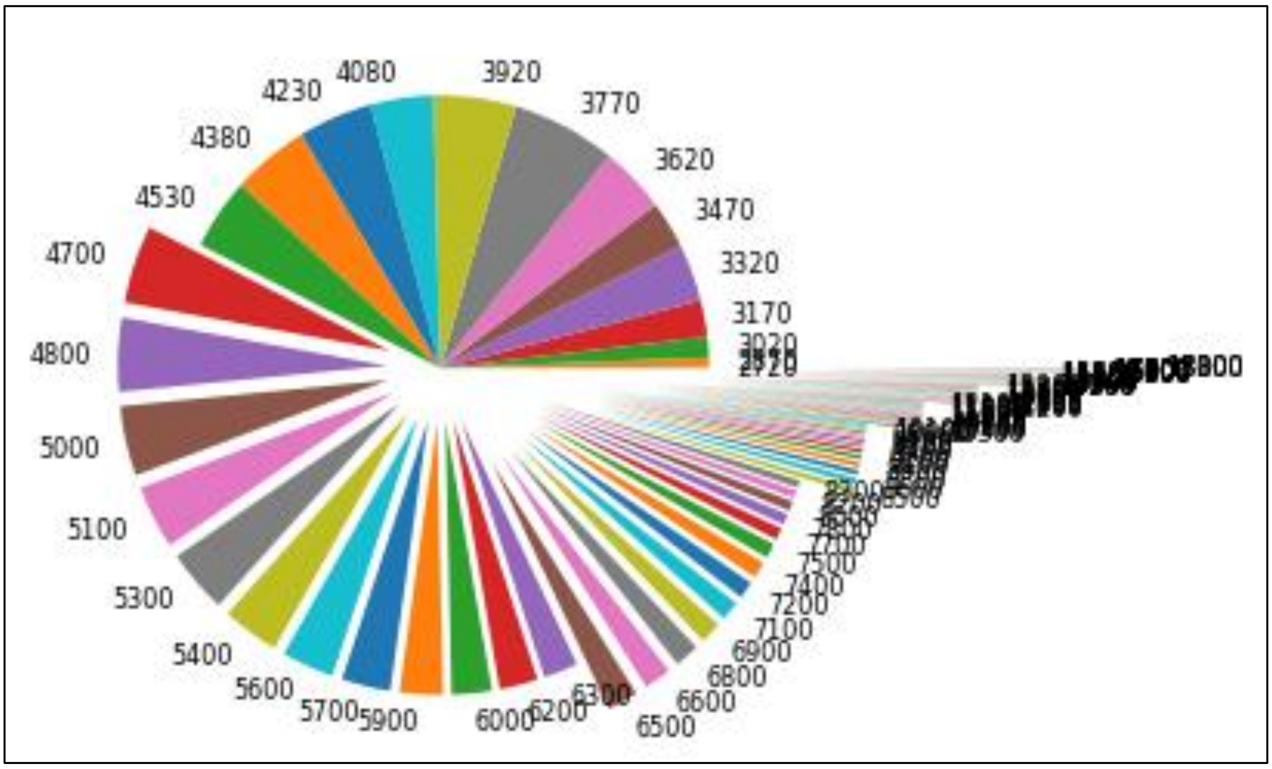


Chart 1: Pie Chart representing all temperature readings (in Kelvin) belonging to a certain number of pixels in $\boldsymbol{H_\alpha}$ fit file

Chart 1 represents the Temperature distribution within *NGC 3576* in regards to $\boldsymbol{H_\alpha}$**.** The bigger the piece the more the number of Pixels are within that range of temperature. Certain pieces seem to detach from the centre of the pie chart. This detachment is proportional to the temeperature value. Higher temperatures float further away from the centre than lower temperatures. This also correlates with the uncertianty of the temperature as previously discussed. Certain labels of temperature cannot be seen in **Chart 1**. Refer to Appendix B for all subsets of pie chart data Grouped into 9 different categories of Temperature.

The largest grey piece has a temperatrue label of $3\ 770\ K$. This measurement is very close to that of the peak Photpix Temperature and can be regarded with very low uncertainty.

Chart 1 feautres a relatively large amount of low temperatures within certain regions of *NGC 3576*. Less than a third of the chart is of higher tmeperatures. This data is of H_α signifying the existence of young OB stars. If this were true then it would mean that most pieces should be detached at a good distance from the centre to feature high temperatures. However, this is not the case. It raises high suscpion that there must be <u>dust</u> that prohibits sufficient readings due to its reddening effect.

To look at the distribution of dust, the splitting of 100 levels was constucted using python (See Appendix F) . Each of these levels resembles the image of H_α with only intensities of that level being displayed. Particular image-levels have been selected to show regions where dust resides in the nebula. (See **Image 3**)

Image 3 features specifically chosen levels to represent the regions of *NGC 3576* that have low and high intenisty of light. The white regions represent the areas that have that specific intensity level whilst the black regions refers to the regions where there is no photon intensity of that level. E.g. The first image-level has only small regions at its centre that have light of intensity level 28. However, the last image-level only has regions at the top that have light of intensity level 98. When moving from left to right (or from up to down) the temperature value of the image increases until the final image belonging to level 98.

This shows how the temperature is distributed around the Nebula from Earths point of view. It is seen that after image-level 79, the white spots within the next image-levels seem more prominent. This is only due to many same-photpix-level pixels that are close enough to each other to produce this effect. The only explanation as to why this occurs is because these images of higher levels have little to no dust that would reduce the photpix values. However, the lower levels have fainter features of white spots which gives a clue as to regions covered by dust (from image-level 79 or 67 to lower image-levels).

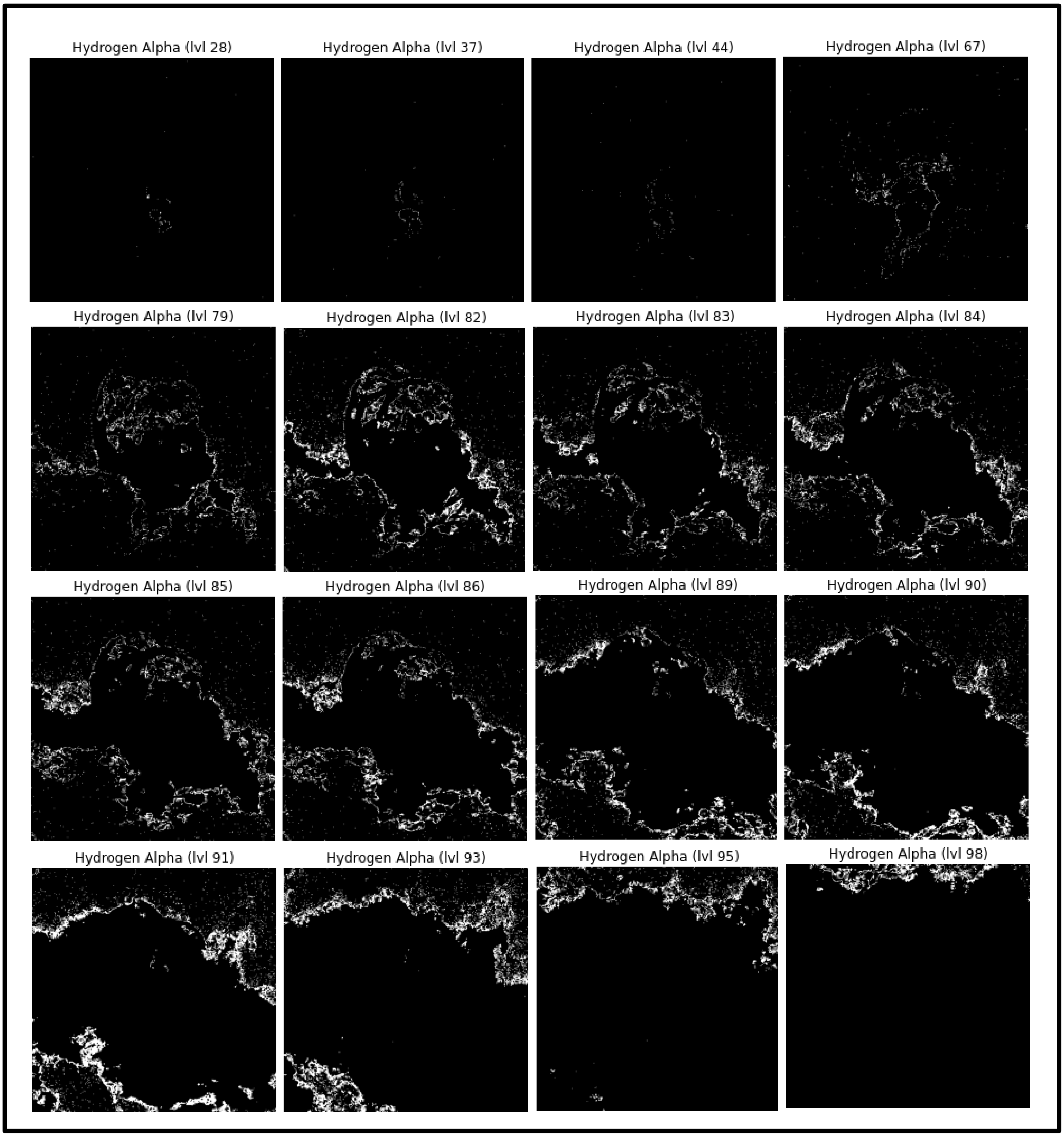


Image 3: Collage of specific image-Levels after splitting $\boldsymbol{H_\alpha}$ data into 100 levels.

To confirm this, $O\ III$ and $S\ II$ fit file were also split into 100 levels. **Table 1** gave the mean photpix of the matrices of $O\ III$ and $S\ II$ fit file as being 1.014 and 0.216, respectively, and both standard deviations were smaller than 1 showing that most of the data had extremely low photpix values. That is the reason as to why the $O\ III$ and $S\ II$ Image-Levels have little to no difference in appearance, except for $O\ III$ level 98 and $S\ II$ level 99. These two image-levels were combined to show the metallicity regions of *NGC 3576*. (see **Figure 1**)

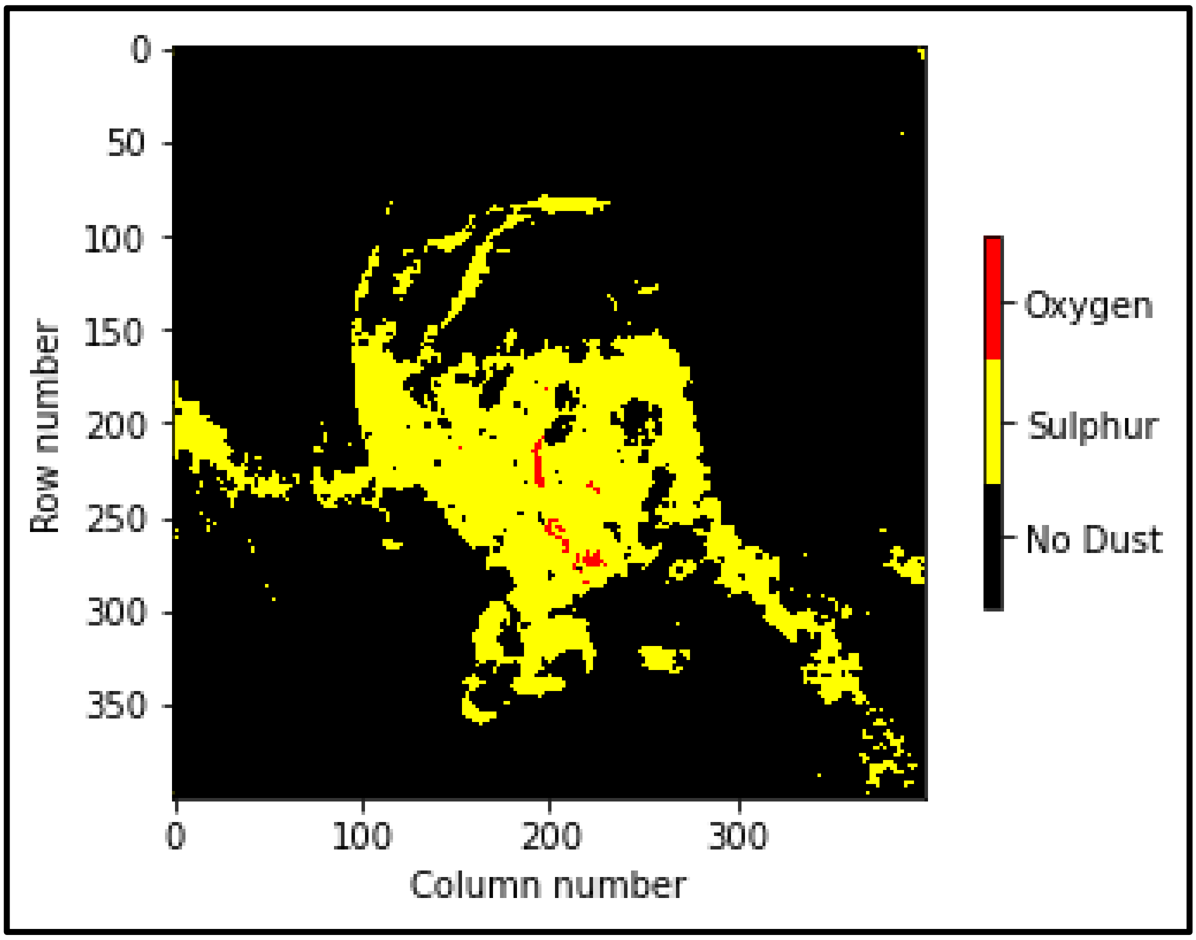


Figure 1: Distribution of dust within *NGC 3576* with regards to $O\ III$ and $S\ II$

Figure 1 shows great abundance of Sulphur dust than Oxygen. However, **Image 2** has H_α image-level of 37 that approximately alligns with Oxygen level 98 in **Figure 1**. Similarly, **Image 2** has a H_α image-level of 82 that approximately aligns with Sulphur level 99 in **Figure 1**. The higher the image-level, the lower the temperatures of the OB stars as seen in the H_α images. This leads to the notion that Oxygen dust resides in the region of higher temperatures than that of Sulphur. From this analogy, an approximation of the temperature of most of the dust can be retrieved.

Figure 2 shows the superimposing of all the specifically correlating level-images discussed in reference to **Image 2**:

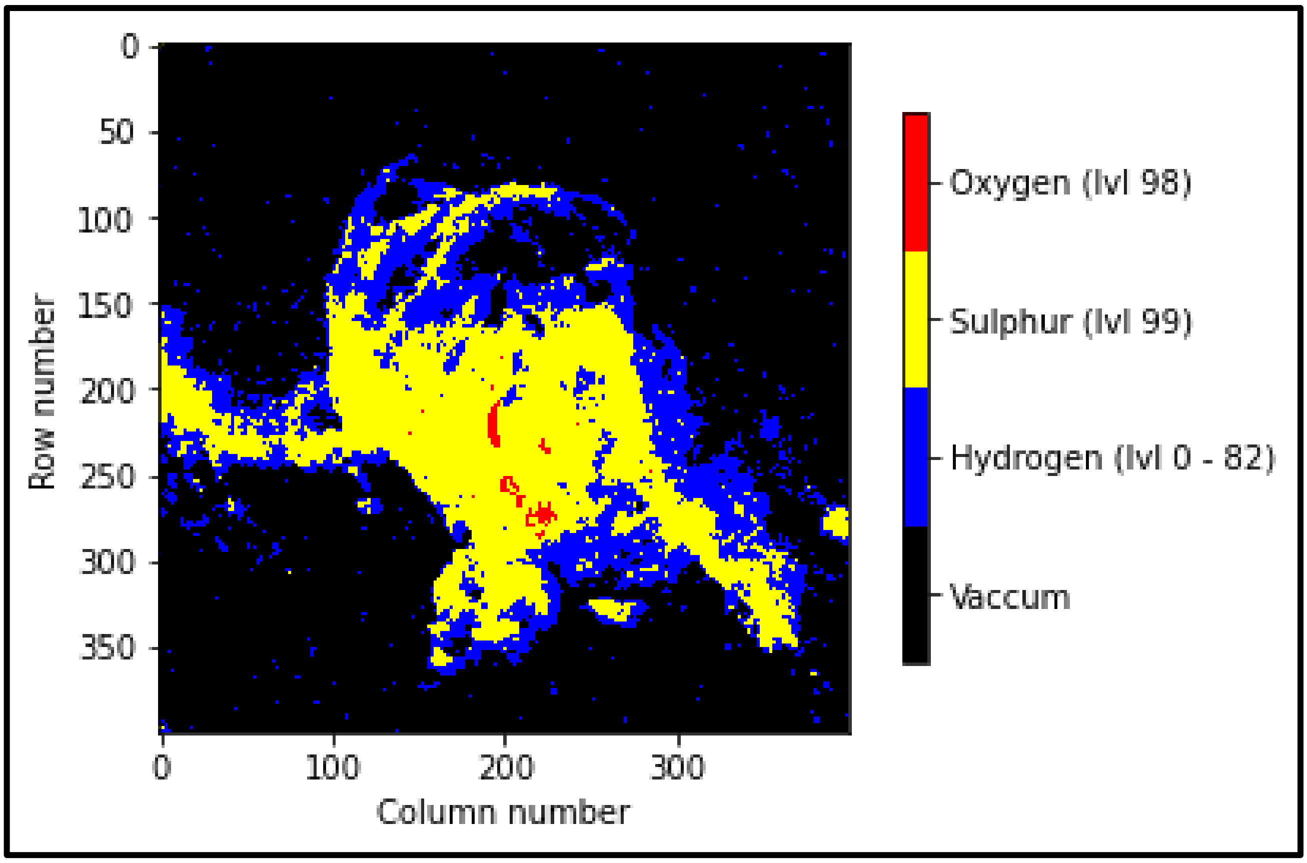


Figure 2: Comparison between Dust and young OB stars within *NGC 3576*

There are a few blue regions that are very far away from the central red region of Oxygen and some are misleading due to the noise coming from black bodies in the background.

Since the data had been split into 100 levels. Level 1 corresponds to the top 1% of temperature readings in H_α (highest values of temperature) and similarly, level 100 corresponds to the bottom 1% (lowest values of temperature). **Table 2** and **Table 3** show $(2\,720 \pm 60)K$ and $(18\,300 \pm 400)K$, for H_α, respectively. These are the minimum and maximum values that can be related to the 100 level-images. With the assumption that there is a constant proportionality (C) that relates temperature of H_α with levels, the following can be conducted:

$$C = \frac{(18\,300 \pm 400)\;K - \;(2\,720 \pm 60)\;K}{level\;1 - level\;100} = \frac{(18\,300 \pm 2.186\%)\;K - \;(2\,720 \pm 2.206\%)\;K}{level\;1 - level\;100}$$

$$= \frac{(18\,300 - 2\,720)\;K(levels)^{-1}}{-99} \pm (2.186\% + 2.206\%)$$

$$= -157.4\;K\;(level)^{-1} \pm (4.392\%)$$

$$= (-157.4 \pm 6.912)\;K\;(level)^{-1}$$

$$\therefore C = (-157\ \pm 7)\ K(level)^{-1}$$

$$T_{H_\alpha}\left(L_{H_\alpha}\right) = \ (-157\ \pm 7)\ L_{H_\alpha} + 18\,300 \qquad (3)$$

Where:

$L_{H_\alpha} = Level\ number\ from\ H_\alpha\ (in\ levels)$

$T_{H_\alpha} = Temperature\ correpsonding\ to\ that\ of\ H_\alpha\ temperatures\ (in\ Kelvin)$

By using eq.3, a good estimate of the temperature of dust could be acquired. As seen in **Figure 2**, dust emitting $S\ II$ radiation covers a wide range whereby the correspoding regions of H_α emissions covers temperatures from level (from 0 to 82). Oxygen can be seen to be covering the level range of H_α from levels 1 to 37.

$$T_{H_\alpha}\left(L_{H_\alpha} = 0\right) = (18\,300 \pm 800)\ K$$

$$T_{H_\alpha}\left(L_{H_\alpha} = 100\right) = (2\,720 \pm 100)\ K$$

Notice how how the use of eq. 3 reaps larger uncertainties. E.g. Temperature at 18300 K has an uncertianty that has increased by 600 K with the method of attaining it from eq. 3. This indicates that by using this linear relationship, there will be higher uncertianties in temeprature for the dust than there is of H_α. This is even regardless as to whether the best estimates of dust are identical to that of H_α.

Note that the percentage error (4.392%) was used instead of the uncertinaty of C (± 7).

$$T_{O\ III_{min}} = \ T_{H_\alpha}(37) = (12\,500 \pm 500)K$$

$$T_{S\ II_{max}} \ = \ T_{O\ III_{max}} \ = \ T_{H_\alpha}(0) = (18\,300 \pm 800)\ K$$

$$T_{S\ II_{min}} \ = \ T_{H_\alpha}(82) = (5\,400 \pm 200)K$$

Hence , the Oxygen and Sulphur dust reside within the following temperature inequalities:

$$(12\,500 \pm 500)K \leq T_{O\ III} \leq \ (18\,300 \pm 800)\ K$$

$$(5\,400 \pm 200)K \leq \ T_{S\ II} \ \leq (18\,300 \pm 800)\ K$$

$$0.432 \leq \frac{T_{S_{II}}}{T_{O_{III}}} \leq 1$$

These calculations proves that the Sulphur in abundance is of lower temperature than that of oxygen. But the ratio $\frac{T_{S_{II}}}{T_{O_{III}}} \leq 1$ signifies that there are regions where both the different dust have maximum temperatures. The region in **Figure 2** that agrees with the absolute equality $\frac{T_{S_{II}}}{T_{O_{III}}} = 1$ is where there is oxygen superimposed onto the Sulphur (row number 190 to 290 and column number 200 to 240). The further away one moves from this region the more the inequality $\frac{T_{S_{II}}}{T_{O_{III}}} < 1$ holds true till the limit at which $\frac{T_{S_{II}}}{T_{O_{III}}} = 0.432$.

It should be noted that the reason as to why the combined image-levels of H_α was capped at the limit of 82 was because its was found that the image matrix of H_α had no element 0. This is also feared in **Table 1** where the minimum Photpix value was of the value 35. Combining all image-levels ould have produced **Figure 2** to have a blue background with little to no balck to feature regions of a vaccum. The purpose was to superimpose certain regions to show the distribution of dust as an explanation to the faintess of the first image-levels in **Image 3**.

"Type O stars have the highest surface temperatures and can be as hot as 30,000 Kelvins. On the other extreme, type M stars have the lowest surface temperatures and can be as cool as 3,000 K" (csun.edu, undated). This differs from the highest acquired temperature even after looking at its upper limit $(18\ 700\ K)$. However, the upper limit of the lowest temperature reading is $2\ 780\ K$ is much closer to the theoretical value of $3\ 000\ K$. Weins law relates higher temperatures to have most intensity in shorter wavelengths. Hence, the higher the temperature the shorter the wavelength. This means that OB stars emit light of much higher energy which are reddened **more** by dust than larger wavelengths from M stars. That is why the upper limit of the highest surface temperature is much further from the theoretical value of $30\ 000\ K$ whilst the upper limit of the lowest surface temperature is not that far off from the temperature of M stars. This can be confirmed by the percentage error below.

$$\%\ error = \left|\frac{Observed - Expected}{Expected}\right|$$

$$\%\ error_{max\ Temp} = \left|\frac{18\ 300 - 30\ 000}{30\ 000}\right| = 0.39 = 39\%\ error$$

$$\%\ error_{min\ Temp} = \left|\frac{2\ 720 - 3000}{3000}\right| = 0.09\overline{3} = 9.\overline{33}\%\ error$$

The area of dust covered (in terms of number of pixels) from **Figure 1** was calculated to look at the ratio of distribution between Oxygen and Sulphur. (See Appendix D). The (400×400) pixel matrix gives the total number of $160\ 000$ pixels in total. The Number of $O\ III$ pixels and the Number of $S\ II$ pixels can be denoted by $N_{(O\ III)}$ and $N_{(S\ II)}$, respectively.

$$\therefore \frac{N_{(O\ III)}}{N_{(S\ II)}} = \frac{486}{31\ 641} = 0.0154\ (3\ s.f.)$$

Note that also the Distribution of total dust can also be acquired by finding the Distribution Coefficient (N_D). (See Appendix E). There is overlapping of Oxygen and Silicon that shouldn't be considered.

$$N_D = \frac{Dust}{Total\ Pixel\ Area} = \frac{31785}{160\ 000} = 0.199\ (3\ s.f.)$$

However, there are many assumptions that have been made in order to acquire temperature readings on the regions of OB stars and dust. Most assumped relationships have been considered to be linear which may not be the case. A better method of acquiring the temperature readings would be to outsource flux information. However, it was very difficult use this methodology due to the lack of research papers open to the public that specifically gives insight on hydrogen alpha emission. There were a few sources that could help salvage the temperature however, the research conducted were not of the same product of study.

For example, the first attempt was to get readings from a publically posted data base with user friendly features that gave *NGC 3576* flux and frequency information (in Jansky). However, the energy of those regions were of gamma radiation which indicates byproducts of massive or rapid activity such as a supernova remnants; neutron stars/pulsars, etc.

The second attempt was to get readings from a publically posted reasearch paper that contained temperature values for *NGC 3576*. Unfotunately, they were for H-113β and H90-α emissions. The α and β represented a change in 1 and 2 optical numbers (δn), respectively. The 90 and the 113 are the original n_1 (before the transition) which is different from H_α emissions; transition $n_1 = 3$ to $n_2 = 2$.

Thus, outsourcing data from either of the examples above would not have been a scientifically huge mistep in when conducing calculations and relations.

Additionally, there were sources of black body radiation (as seen **image 2**) that were in the background acting as noise and increasing error in calculations.

Conclusion

In summary, *NGC 3576* clearly has many young OB stars with highest surface temperatures of $(18300 \pm 400)\ K$ and M stars exhibit lower surface temperatures of up to $(2720 \pm 60)\ K$ with percentage errors 39% and $9.\overline{33}\%$, respectively, due to dust.

A large region of the nebula contains Sulphur dust that reradiates temperatures $T_{S\,II} \geq (5400 \pm 200)\ K$ whilst a small region of the nebula contains Oxygen dust that reradiates temperatures $T_{O\,III} \geq (12500 \pm 500) K$ leading to the ratio $\frac{T_{S_{II}}}{T_{O_{III}}} \leq 1$.

Occupation of the ratio of Oxygen to Sulphur dust within the observed region is of the value 0.0154 . Distribution coefficient of dust (N_D) was found to be 0.199.

An improvement of these readings would be to acquire fit files that have information of flux and frequency data. However, with the given fit files, were sufficient enough to give a good estimation as to the effect of the light and temperature from Earths point of view.

Bibliography

(no date) The hertzsprung - russell diagram. Available at: http://www.csun.edu/~boregan/astrolab/manual/unit080.htm#:~:text=Type%20O%20stars%20have%20the,as%20cool%20as%203%2C000%20K (Accessed: September 30, 2022).

Bot, P. (2018) Category:NGC 3576, Wikimedia Commons. Available at: https://commons.m.wikimedia.org/wiki/Category:NGC_3576 (Accessed: September 1, 2022).

Newman, P. and Whitlock, L.A. (no date) Starchild: Stars, NASA- Stars -Medium Stars. The StarChild Team. Available at: https://starchild.gsfc.nasa.gov/docs/StarChild/universe_level2/stars.html#:~:text=Once%20the%20temperature%20reaches%2015%2C000%2C000,the%20evolution%20of%20a%20star (Accessed: September 20, 2022).

Physical properties (2016) The MOSFIRE Deep Evolution Field Survey. WordPress. Available at: https://mosdef.astro.berkeley.edu/for-the-public/public/physical-properties/#:~:text=There%20are%20two%20lines%20from%20singly%2Dionized%20nitrogen%20(%5BNII,many%20weaker%20lines%20as%20well (Accessed: September 1, 2022).

Physical properties (2016) The MOSFIRE Deep Evolution Field Survey. WordPress. Available at: https://mosdef.astro.berkeley.edu/for-the-public/public/physical-properties/#:~:text=There%20are%20two%20lines%20from%20singly%2Dionized%20nitrogen%20(%5BNII,many%20weaker%20lines%20as%20well (Accessed: September 1, 2022).

Table of Galaxy Emission lines from 700Å to 11,000Å. (no date) Table of UV/optical emission lines observed in Galaxies. Available at: http://astronomy.nmsu.edu/drewski/tableofemissionlines.html (Accessed: September 13, 2022).

Vedantu (2022) Wien's law, VEDANTU. Vedantu. Available at: https://www.vedantu.com/physics/wiens-law (Accessed: September 19, 2022).

Weisstein, E. (2022) Gaussian function, from Wolfram MathWorld. Wolfram Research. Available at: https://mathworld.wolfram.com/GaussianFunction.html (Accessed: September 16, 2022).

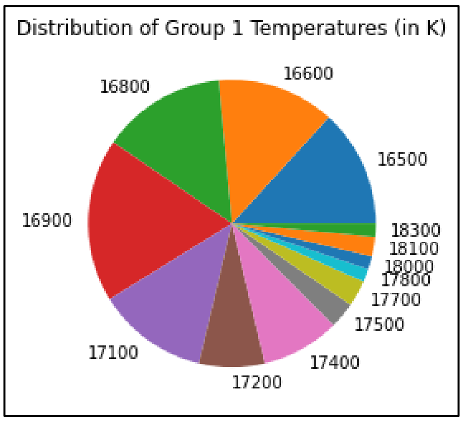
Appendices

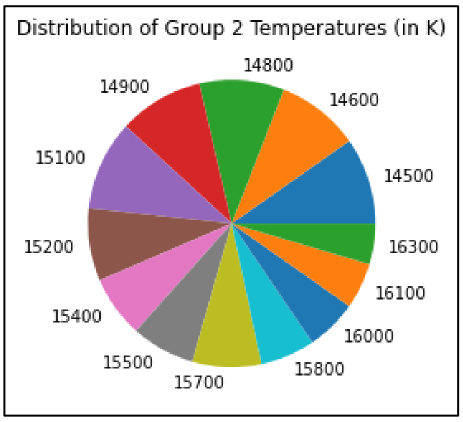
Appendix A

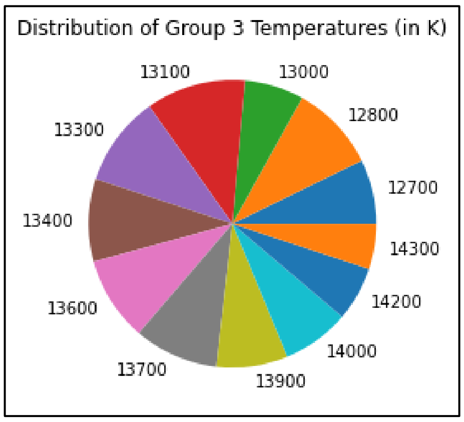
Photpix	No. of Pixels	Temp(K)	δ Temp (K)	Photpix	No. of Pixels	Temp(K)	δ Temp (K)
62	7400	4700	100	140	153	10600	200
64	6988	4800	100	142	132	10700	200
66	6629	5000	100	144	130	10900	200
68	6045	5100	100	146	101	11000	200
70	6072	5300	100	148	104	11200	200
72	5488	5400	100	150	108	11300	200
74	5031	5600	100	152	97	11500	200
76	4696	5700	100	154	93	11600	200
78	4155	5900	100	156	91	11800	200
80	3851	6000	100	158	79	11900	200
82	3689	6200	100	160	84	12100	200
84	3304	6300	100	162	88	12200	300
86	2959	6500	100	164	86	12400	300
88	2648	6600	100	166	50	12500	300
90	2388	6800	100	168	64	12700	300
92	2074	6900	100	170	86	12800	300
94	1846	7100	100	172	59	13000	300
96	1652	7200	100	174	99	13100	300
98	1571	7400	200	176	90	13300	300
100	1396	7500	200	178	82	13400	300
102	1144	7700	200	180	84	13600	300
104	1009	7800	200	182	84	13700	300
106	917	8000	200	184	71	13900	300
108	750	8200	200	186	67	14000	300
110	703	8300	200	188	54	14200	300
112	572	8500	200	190	45	14300	300
114	546	8600	200	192	49	14500	300
116	473	8800	200	194	47	14600	300
118	472	8900	200	196	48	14800	300
120	392	9100	200	198	48	14900	300
122	351	9200	200	200	51	15100	300
124	289	9400	200	202	41	15200	300
126	266	9500	200	204	35	15400	300
128	243	9700	200	206	36	15500	300
130	248	9800	200	208	39	15700	300
132	206	10000	200	210	31	15800	300
134	200	10100	200	212	29	16000	300
136	185	10300	200	214	26	16100	300
138	164	10400	200	216	23	16300	300

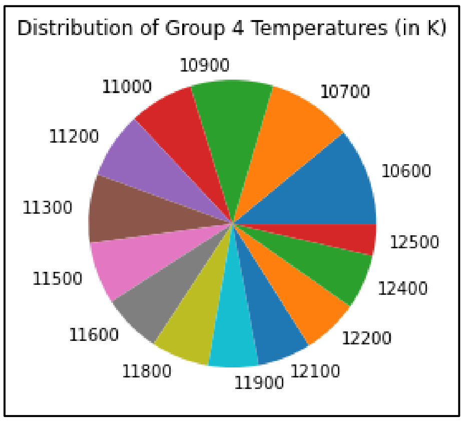
Appendix B

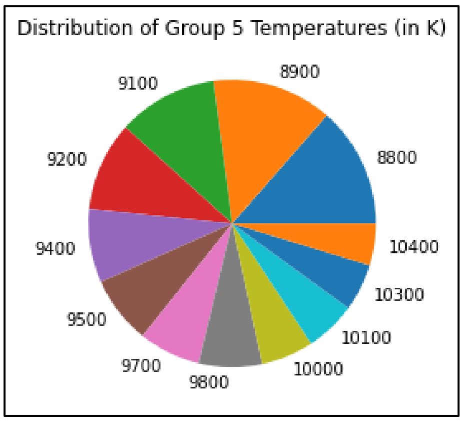
Note: Group 1 is of highest temperatures whilst Group 9 is of lowest temperatures

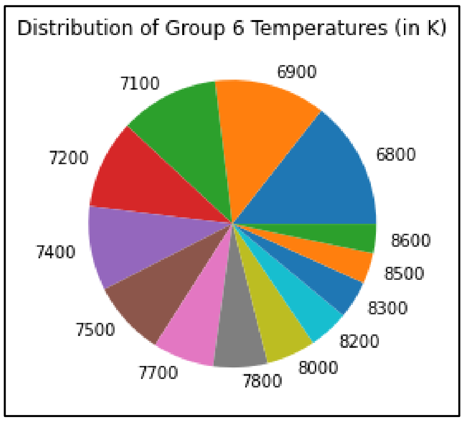


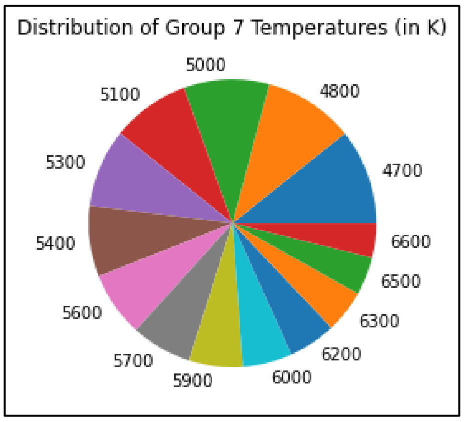


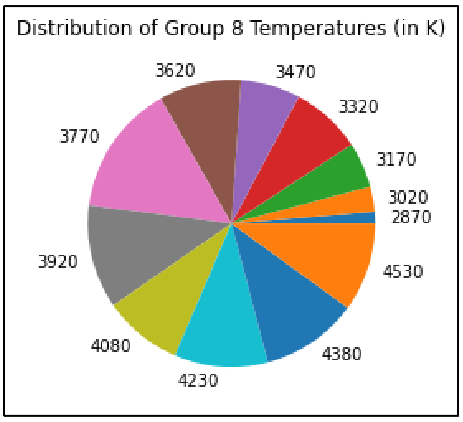


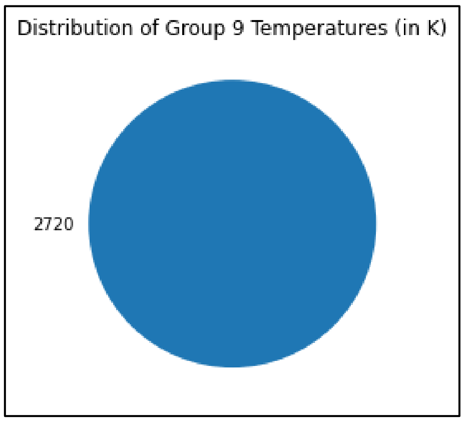








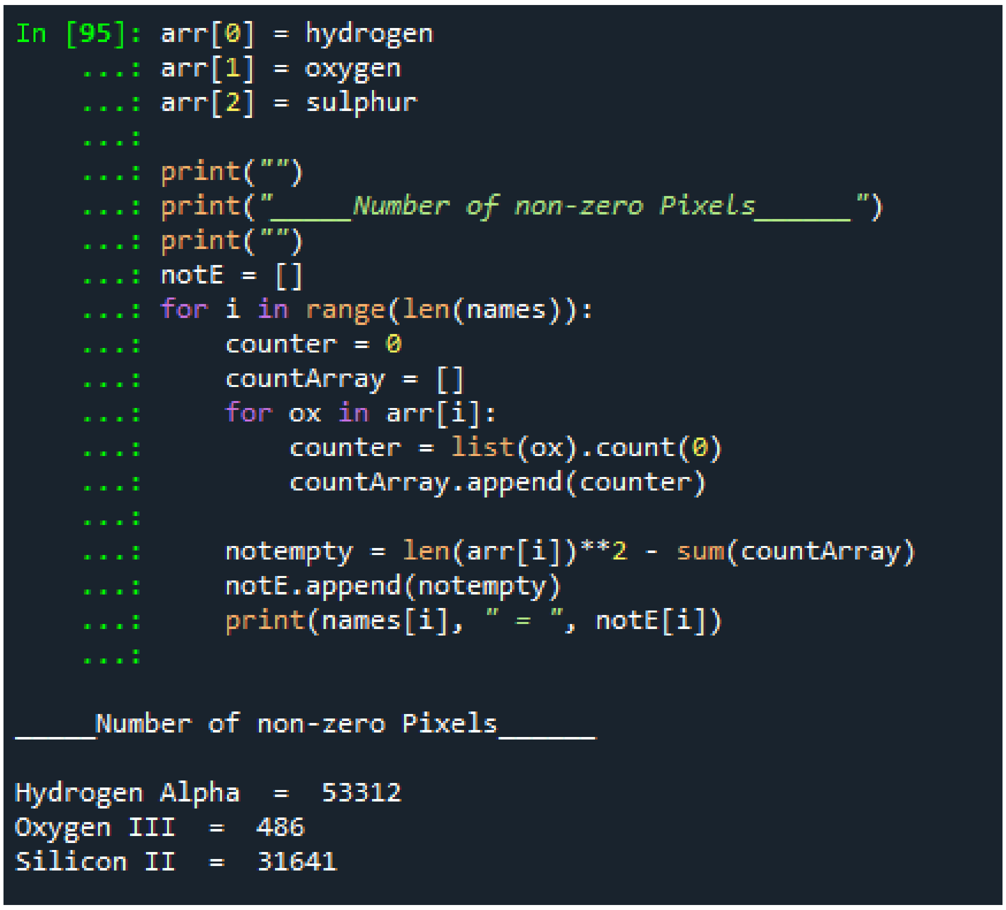




Appendix C

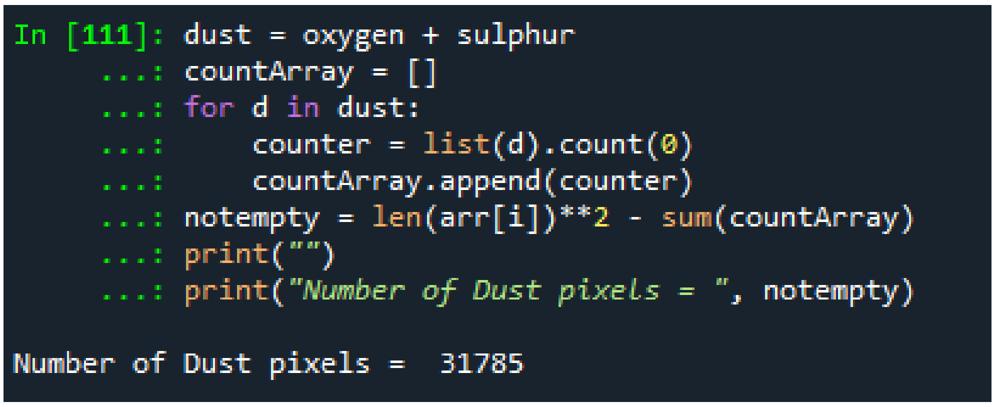
```
#GAUSSIAN_________________________________________________________________________

    #Taking subset of the Data
        yVals , xVals = list(hist1[0]) , list(hist1[1]) #Converting numpy array to list

        x_dats = xVals[xVals.index(39):xVals.index(85)] #(From 42 to 84 )
        y_dats = yVals[xVals.index(39):xVals.index(85)]

        #popt -> Gives optimal parametres that make the gaussian curve best fit the data points
        #pcov -> Covariance matrix giving estimate of errors

        #NOTE: Gaussian Function as a function
            #gauss(x,A,mu,sig)
            #A = Amplitude (Guessing it to be 7500)
            #mu = Centre of the graph (Guessing the value 59)
            #sig = Sigma is about 10 (assumption)

        popt , pcov = curve_fit(gauss, x_dats, y_dats, p0=[7500,60,10])

        A_opt, mu_opt, sig_opt = popt

        # A_opt -> Optimal value for A
        # mu_opt -> Optimal value for mu
        # sig_opt -> Optimal value for sig

        x_fit = np.linspace(0,xVals[-1],1000) # x coords for Gaussian fit
        y_fit = gauss(x_fit, A_opt, mu_opt, sig_opt) # y coords of Gaussian fit

        Unc_mu = np.diag(pcov)[1] #Uncertianty in the peak value of mu
        percUnc_mu = (Unc_mu/mu_opt)*100 #Percetnage uncertanty of Mu


        plt.figure(9)
        plt.minorticks_on()
        plt.grid(visible = True, which="both" , axis="both" , color = "grey")
        plt.xlabel("Photpix", size = 13)
        plt.ylabel("Number of Pixels", size = 13)
        plt.scatter(xVals[:-1],yVals, label = "Grouped data Points from Histogram")
        plt.plot(x_fit,y_fit,color = "blue", label = "Gaussian Fit")
        plt.legend()

        plt.figure(10)
        plt.minorticks_on()
        plt.grid(visible = True, which="both" , axis="both" , color = "grey")
        plt.xlabel("Photpix", size = 13)
        plt.ylabel("Number of Pixels", size = 13)
        plt.scatter(x_dats,y_dats, label = "subset of Grouped data Points from Histogram")
        plt.plot(x_fit[148:370],y_fit[148:370],color = "blue", label = "Gaussian Fit")
        plt.legend()
```

Appendix D



```
In [95]: arr[0] = hydrogen
    ...: arr[1] = oxygen
    ...: arr[2] = sulphur
    ...:
    ...: print("")
    ...: print("_____Number of non-zero Pixels______")
    ...: print("")
    ...: notE = []
    ...: for i in range(len(names)):
    ...:     counter = 0
    ...:     countArray = []
    ...:     for ox in arr[i]:
    ...:         counter = list(ox).count(0)
    ...:         countArray.append(counter)
    ...:
    ...:     notempty = len(arr[i])**2 - sum(countArray)
    ...:     notE.append(notempty)
    ...:     print(names[i], " = ", notE[i])
    ...:

_____Number of non-zero Pixels______

Hydrogen Alpha  =  53312
Oxygen III  =  486
Silicon II  =  31641
```

Appendix E



```
In [111]: dust = oxygen + sulphur
     ...: countArray = []
     ...: for d in dust:
     ...:     counter = list(d).count(0)
     ...:     countArray.append(counter)
     ...: notempty = len(arr[i])**2 - sum(countArray)
     ...: print("")
     ...: print("Number of Dust pixels = ", notempty)

Number of Dust pixels =  31785
```

Appendix F

```python
# Intensity Spectra of Nebulae
# S. Amani Njoroge
# 4060924
# DataAnalysis

from FitsExtraction import HDUs, corrector #importing HDU Data Sets from FIT files as well as copies of the Fit files
import numpy as np #Importing numpy for useful array manipulation.
from statistics import median, multimode, stdev
import sys

"""
Converting PrimaryHDU's into numpy arrays by firstly extracting them from
FitsEctract.py package
"""
arrs = [] #Storing numpy matrix of ImageHDU of HA, OIII and SII respetively.
names = ["Hydrogen Alpha", "Oxygen III", "Sulphur II"]

for x in range(len(HDUs)):
    #HDUs[x][0] means the Primary HDU whilst HDUs[x][1] would have been ImageHDU
    arrs.append(np.array(HDUs[x][1].data))

    #Making sure to close each Fits file after accessing.
    #Must come after converting the data to numpy array first
    #HDUs[x].close()

matrix_title = ["_______HA array/matrix______", "______OIII array/matrix________", "______SII array/matrix_______"]

unfiltMTX = input("""
Show unfiltered matrices of fit file images?

Y/N ?

""")
```

```
if unfiltMTX == "Y":
    for x in range(len(arrs)): #Looping through length of arrs (starting from 0 to length-1)
        print(matrix_title[x]) #Print the title matrix before printing the matrix data
        print("")
        print(arrs[x]) #Print the matrix
        print("")
elif unfiltMTX == "N":
    pass
else:
    corrector()

minToMax_arr = []   #Matrix needed to store 1D data for checking Max and Min
                    #photons in particular array

#Array storing string elements for ouptutting stat titles
stat_titles = ["_____HA Stats________", "_____OIII Stats________", "_____SII Stats________" ]
max_vals = [] #storing Maximum value of HA, OIII and SII into array, RESPECTIVELY
min_vals = [] #storing Minimum value of HA, OIII and SII into array, RESPECTIVELY
median_arr = [] #storing Median value of HA, OIII and SII into array, RESPECTIVELY
modes_arr = [] #storing Modal(s) value of HA, OIII and SII into array, RESPECTIVELY
stDev_arr = [] #storing standard Deviation value of HA, OIII and SII into array, RESPECTIVELY
up_q_arr = []
low_q_arr = []

#Statistical Data
statDats = input("""
Print statistical Data outputs of each fit file?

Y/N ?

""")

for x in range(len(stat_titles)):
    for i in arrs[x]: #Looping over each individual sub-araay list
        for j in i: #Looping through each element in specific sub-array list
            minToMax_arr.append(j) #Adding each element into 1D array
    minToMax_arr.sort() #Sorts the array in ascending order
    maxim, minim = minToMax_arr[-1], minToMax_arr[0]
    up_q, low_q = np.percentile(arrs[x],75), np.percentile(arrs[x],25) #getting upper and lower quartile of each matrix data
    up_q_arr.append(up_q) , low_q_arr.append(low_q)
    med =  median(minToMax_arr)
    median_arr.append(med)
    mode = multimode(minToMax_arr)
    modes_arr.append(mode)
    mean = np.mean(arrs[x])
    std = stdev(minToMax_arr, xbar = mean)
    stDev_arr.append(std)
    max_vals.append(maxim)
    min_vals.append(minim)
    minToMax_arr.clear() #Clearing array of all content for OIII and SII data storing
    if statDats == "Y":
        print(stat_titles[x]) #print the current stat title
        print("")
        print("Maximum photons in a pixel = ", maxim) #Max pixel value
        print("Minimum photons in a pixel = ", minim) #Min Pixel value
        print("Upper quartile = ", up_q) #Printing upper quartile info
        print("Lower Quartile = ", low_q) #Printing lower quartile info
        print("Median = ", med) #Getting median value
        print("Mode(s) = ", mode) #Getting modal value. Using multimode in case of two modes or more
        print("mean= %.3f " % mean)
        print("StDev = ", std) #Standard Deviation
        print("")
    elif statDats == "N":
        pass
    else:
        corrector()

################################################################################
"""
Below is the part where I Look at levels of high to relatively low intensity
The levels of intensity will be rated by the maximum and minimum values that
were previously collected.
```

```

All other lower levels of intensity are counted as negligible if user chooses
a number of levels.

By increasing the scaling factor, the data is categorized in more levels.
By inputting the number of levels (starting from the highest level; Level 1),
the data recorded will be categorized upto that level.
"""
#################################################################################
print("_______________________________________________________________________________________________")

# Limitation based on Colour of an object in digital systems
maxColor = 2**8-1 #0 -> 255

#Creating user-input based Scale-Levels for relative intensity
scF = int(input("""

Please input the number of levels.

Its advisable to choose a high number such as %d or higher.
Highest value to be chosen is %d due to a limitation of
Colour of an object in digital systems.


""" % (int(maxColor/4),maxColor)))

Levels = int(input("""
Levels have been successfully constructed.

Note: Highest intesity level = 1
      Lowest intesity level = %d

Are you sure you want all the data to be processed and inputted till level %d?
If so, then type %d again.

Input the number of level intensities you desire.
(e.g. if you type 6, data will process from level 1 to 6)
Recommended to choose upto the top quarter tier level such as level %d or lower. .
(Warning: can take longer depending on processessing power of your device)

""" % (scF,scF,scF,int(scF/4))))

print("_______________________________________________________________________________________________")
print("")

if scF > maxColor: #If the chosen number of levels are bigger than the Scaling Factor:
    print("""
          You have split the data into %d levels but the limitation (based on 2D pixel data) is %d in length/width.
          """ % (scF, maxColor))

    sys.exit("Please rerun the code an choose wisely.") #exiting the code with a message if the preceeding if statement is met

if Levels > scF: #If the chosen number of levels are bigger than the Scaling Factor:
    print("""You have split the data into %d levels but have chosen your levels of interests to level %d.
          """ % (scF, Levels))

    sys.exit("Please rerun the code and choose wisely.") #exiting the code with a message if the preceeding if statement is met

scales = [] #Array holding constructed scales for HA, OIII and SII

for x in range(len(max_vals)):
    scales.append((max_vals[x] - min_vals[x])/scF)

XYm_gList = [] #List that holds tuples of (x,y,intensity level) for g_arr
XYm_hList = [] #List that holds tuples of (x,y,intensity level) for h_arr
XYm_iList = [] #List that holds tuples of (x,y,intensity level) for i_arr

#array containing all XY_()List
XYm_Lists = [XYm_gList, XYm_hList, XYm_iList]

for q in range(len(XYm_Lists)): #Sorting/Categorising levels.
    i_idxNum = 0 #Index number
```

```python
    for i in arrs[q]: #Looping through array of HA, then OIII then SII
        y = i_idxNum #row number or y coordinate of element
        j_idxNum = 0 #column number
        for j in i: #for every element in the list "i"
            mltp = 0 #creating multiplier variable
            x = j_idxNum #column number or x coordinate of element
            while mltp <= scF and mltp<=Levels+1: #If the multiplier vairable smaller than or equal to the scaling factor
                                                  #Also making sure to dump the rest of the other data inside an extra lower level
                if j >= max_vals[q] - mltp*scales[q]:
                    """Checking condition if data is in specific intensity level
                    in terms of the pixel's photon number. First level is from maximum (inclusive)
                    to lower region of that first level (also inclusive)"""
                    if mltp == Levels+1 or (x,y,mltp-1) in XYm_Lists[q]: #If data lands in extra level or if coordinate already exists
                        break #Get out of while loop rather than recording it
                    else:
                        temp_tup = (x,y,mltp) #Trapping tuple of coords and their corresponding multiplier
                        #tuples t in (<x>,<y>, t) where t = 0 are just the pixel coordinates that have maximum number of photons.
                        XYm_Lists[q].append(temp_tup)
                mltp += 1  #Increasing multiplier to find the next relative intensity
            j_idxNum += 1 #Incrementing the x-cooordinate
        i_idxNum += 1 #Incrementing the y-coordinate
    #Sorting all tuple elements in the List array
    XYm_Lists[q].sort() #Sorting list in ascending order
```