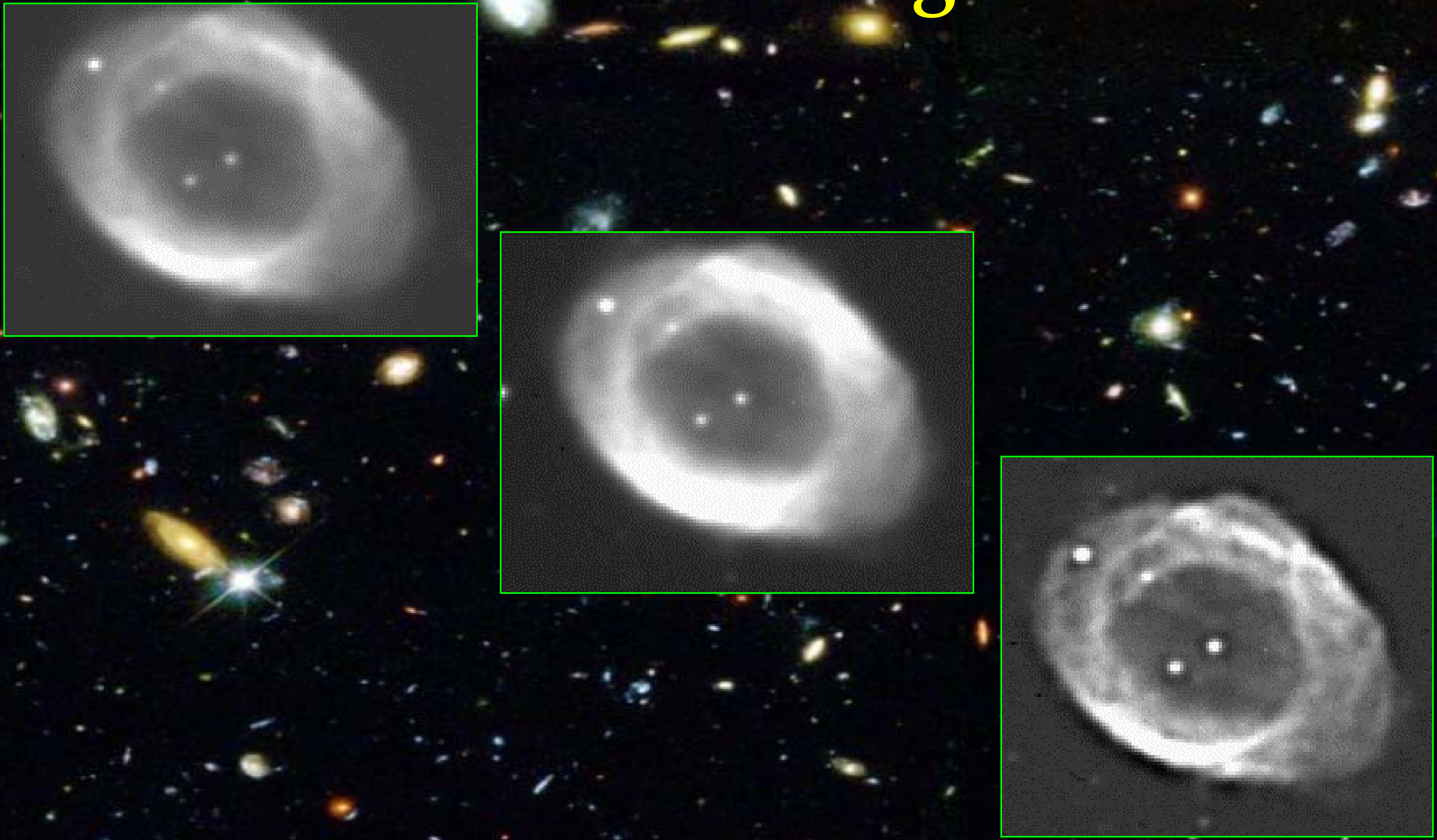
# Introduction to Digital Image Processing



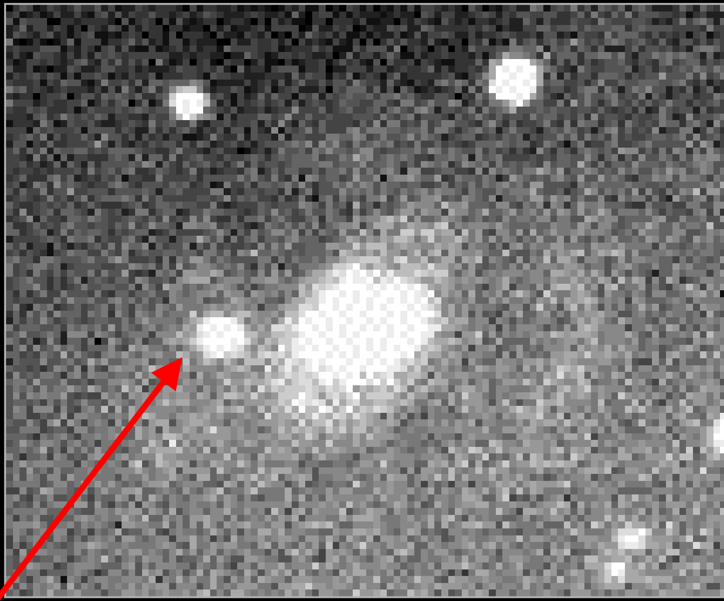
# In this activity you will learn:

1. How a digital image is recorded by your CCD
2. How to remove electronic noise from your raw image
3. Techniques for displaying your image
4. How to design and apply digital filters to emphasize certain features of your image

# 1. What exactly *is* a digital image?

- A digital image is a picture which is divided into a grid of "pixels" (picture elements).
- Each pixel is defined by three numbers, (x,y,z), and displayed on a computer screen.
- The first two numbers give the x and y coordinates of the pixel, and the third gives its intensity, relative to all the other pixels in the image.
- The intensity is a relative measure of the number of photons collected at that photosite on the CCD, relative to all the others, for that exposure.

For example: Here is an image of the quasar Markarian 205, which was taken by Mark Parker with the 14" telescope atop the physics building at the University of California, Santa Barbara, using a thermoelectrically cooled CCD of 576 x 384 pixels.



Markarian 205 is the smaller elliptical object to the left of the galaxy.

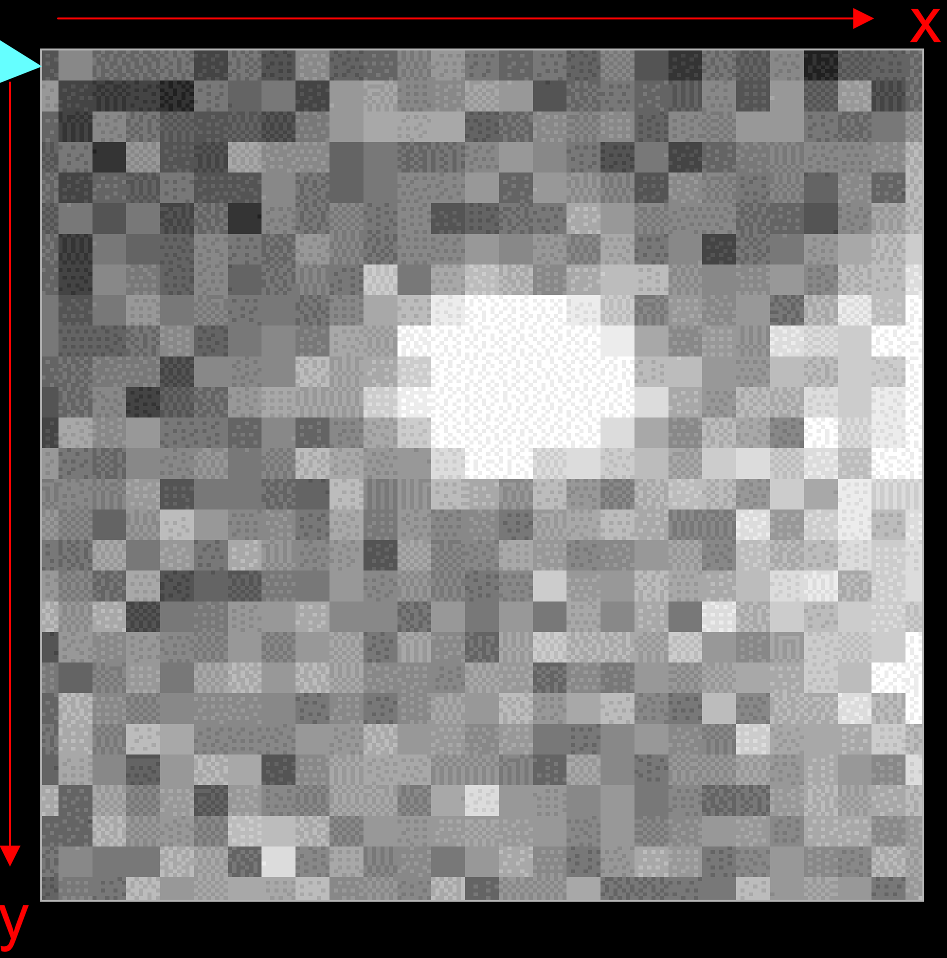
Replace with GTN image!

***UCSB Remote Optical Telescope on the roof of Broida Hall***

Here’s an expanded view of Markarian 205 on the previous slide, showing the individual pixels in the image:

(0,0) are the (x,y) coordinates of the upper left-hand corner.

X-values increase to the right, while y-values increase from top to bottom.



The z coordinates range from 0 to 255, where 0 = blackest of black, and 255 = brightest white, relative to all pixels in the image.

The clarity of a digital image depends on the number of "bits" the computer uses to represent each pixel.

The most common type of representation in popular usage today is the "8-bit image", in which the computer uses 8 bits, or 1 byte, to represent each pixel.

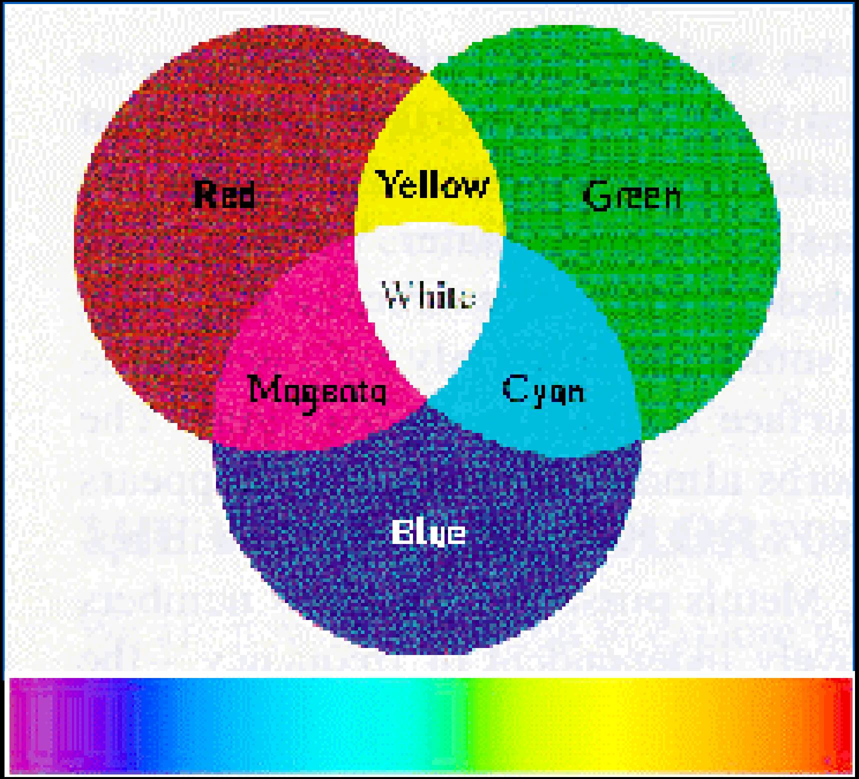
This yields $2^8$ or 256 brightness levels available within a given image.

These brightness levels can be used to create a black-and-white image, with shades of gray between black (0) and white (255).

To create a color image, your software assigns RGB values to each pixel between (0,0,0) and (255,255,255).

In this way it creates "color vectors" that map each pixel value to a color within its pallette.

xRed + yGreen + zBlue = pixel color



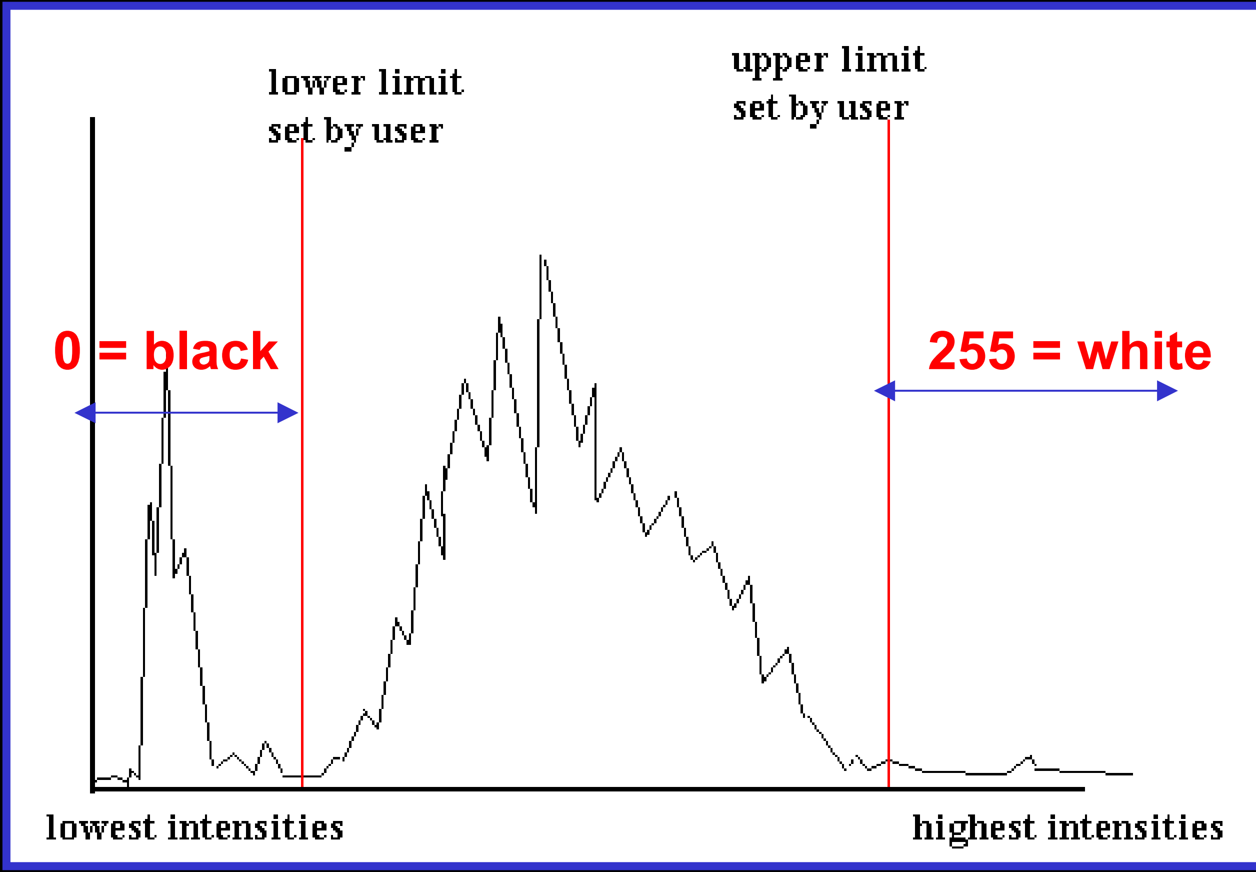
The range of intensity values in an image also depend on the way in which a particular CCD handles its ANALOG TO DIGITAL (A/D) conversion.

12-bit A/D conversion means that each image is capable of $2^{12}$ (4096) intensity values.

If your image processing program only handles $2^8$ (256) brightness levels, these must be divided among the total range of intensity values in a given image.

# This histogram shows the number of pixels in a 12-bit image that have the same intensity, from 0 to 4095.

Suppose you have software that only handles 8-bit information. You assign black and white limits, so that all pixels with values to the left of the lower limit are set to 0, while all those to the right of the upper limit are set to 255. This allows you to look at details within a given intensity range.



So... a digital image is a 2-dimensional array of numbers, where each number represents the amount of light collected at one photosite, relative to all the other photosites on the CCD chip.

It might look something like this:

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 88 | 98 | 107 | 145 | 126 | 67 | 93 | 154 | 223 | 155 | 180 | 232 | 250 | 242 | 207 | 201 |
| 29 | 72 | 159 | 159 | 131 | 76 | 99 | 245 | 211 | 165 | 219 | 222 | 181 | 161 | 144 | 131 |
| 99 | 157 | 138 | 97 | 106 | 55 | 131 | 245 | 202 | 167 | 217 | 173 | 127 | 126 | 136 | 129 |
| 100 | 156 | 110 | 114 | 91 | 70 | 128 | 321 | 296 | 208 | 193 | 191 | 145 | 122 | 135 | 138 |

...only much larger!

***Can you find the brightest pixel in this "image"?***

*... now, what do we DO with it?*

# 2. Correcting the raw image

Every intensity value contains both signal and noise.

Your job is to extract the signal and eliminate the noise!

But first you have to understand what the sources of noise can be...

# *Step 1:*
# *Removing the Dark Current*

*DESIRABLE* photons are those collected by exposure to light from the sky...

...but *UNDESIRABLE* photons come from heating of the chip by the electronics that surround it.

These are the *thermal electrons,* and they create an unwanted signal even when the CCD is covered, with no incident light at all.

Since electrons in motion through a metal or semiconductor create a *current*, these thermally agitated electrons are called the *Dark Current.*

...so, to eliminate thermal electrons, the CCD must be *COOLED* as much as possible.

The cooler one can make one's CCD, the less dark current one will generate.

In fact, the dark current decreases roughly by a factor of 2 for every 7°C drop in temperature of the chip.

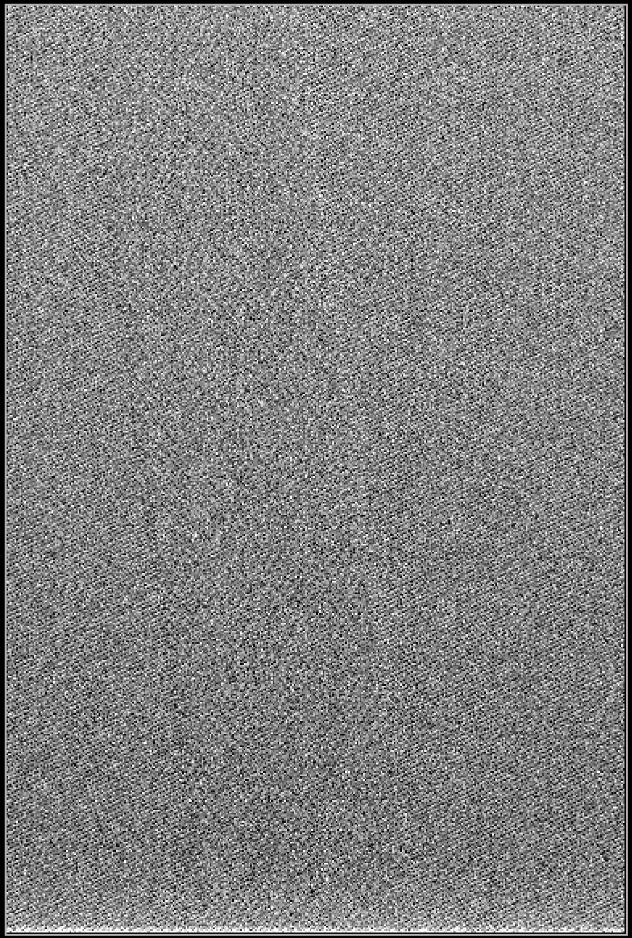
At –100°C the dark current is negligible.

When you process an image correctly, you must account for this dark current, and subtract it out from the image. This is done by taking a "closed shutter" image of a dark background, and then subtracting this dark image from the "raw" image you are observing.

The exposure time of the dark image should match that of the image of the object or starfield you are viewing.

In fact, folks who regularly take CCD images keep files of dark current exposures that match typical exposure times of images they are likely to take, such as: 10, 20, 45, 60, or 300 seconds, which are updated regularly, if not nightly.

Here is a 10-second exposure dark current image taken with the RAAP telescope at UCSB.



Notice the graininess...that is because each photosite is responding to the ambient heat of the circuitry, even though our telescope is reverse-Peltier cooled to -45° F!

Since these electrons are NOT produced by incident photons, but are present as noise in every image, this unwanted information must be subtracted out before you can make measurements such as magnitude determinations.

# *The Bias Correction*

CCD cameras typically add a bias value to each image they record. If you know that the same specific bias value

has been added to each pixel, you can correct for this by subtracting a constant from your sky image.

# *Step 3: Correcting for*
# Pixel-to-pixel Sensitivity variation

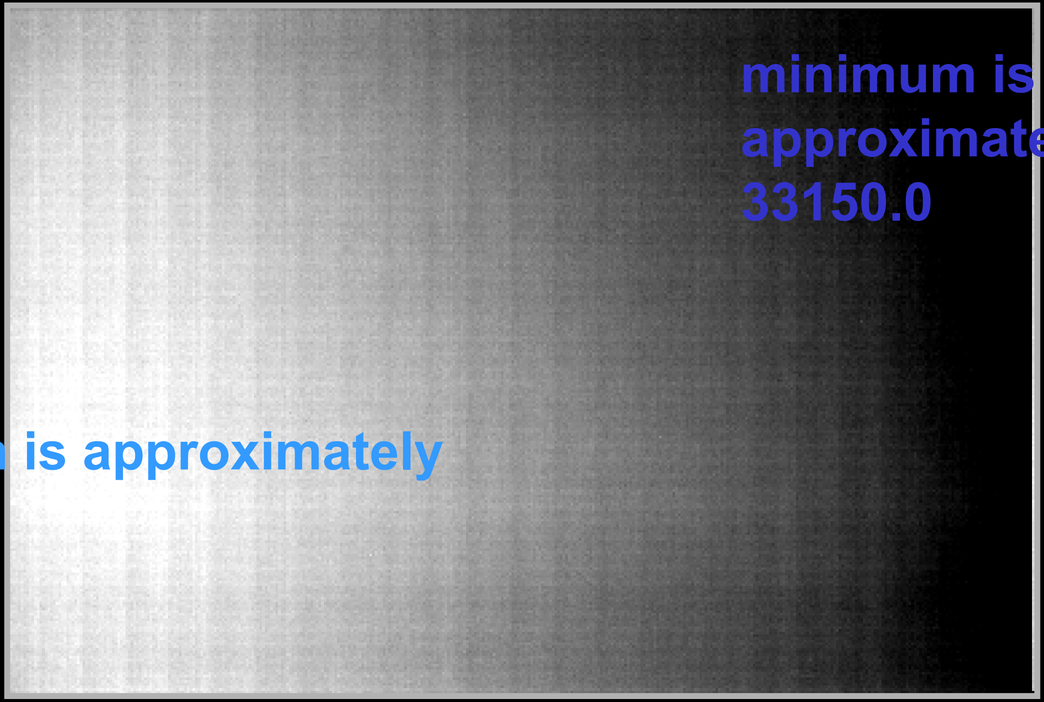
Another source of noise is the inherent variation in the response of each pixel to incident radiation.

Ideally, if your CCD is functioning properly, there should be no variation in pixel value when you measure a uniformly-illuminated background. However, nothing is perfect, and there usually is some slight variation in the sensitivity of each photosite, even if the incident radiation is totally uniform.

This can be accounted for by taking a picture of a uniformly bright field, and dividing the raw image by this "flat" field – a process called *Flat Fielding.*

Here is a 2-second exposure "dome flat" – a flat field image taken inside the dome at Virginia Polytechnic Institute. Although the image was taken of a uniformly bright background,

note the variation in sensitivity across the chip of the pixels!



minimum is approximately 33150.0

maximum is approximately 39814.0

*"Flats" are usually taken each night, sometimes more than once!*

http://www.phys.vt.edu/~jhs/SIP/images/m42/flat.fit

- The length of time to expose the flat image should be enough to saturate the pixels to the 50% or 75% level.

***So how many pictures must you take before I can begin to process my image?***

Well, to put it bluntly, FOUR.

You need 4 images to create a “noiseless” image of the sky:

1) the original;

2) a dark exposure of the same integration time as youoriginal;

3) a flat exposure;

4) and another dark exposure, of the same integration time as your flat exposure!

**FINAL IMAGE =**

**(RAW IMAGE - DARK $_{\text{RAW}}$)**

**(FLAT - DARK $_{\text{FLAT}}$)**

- So...to correct raw images taken with any CCD, you first must :

1) subtract the bias correction from each image;

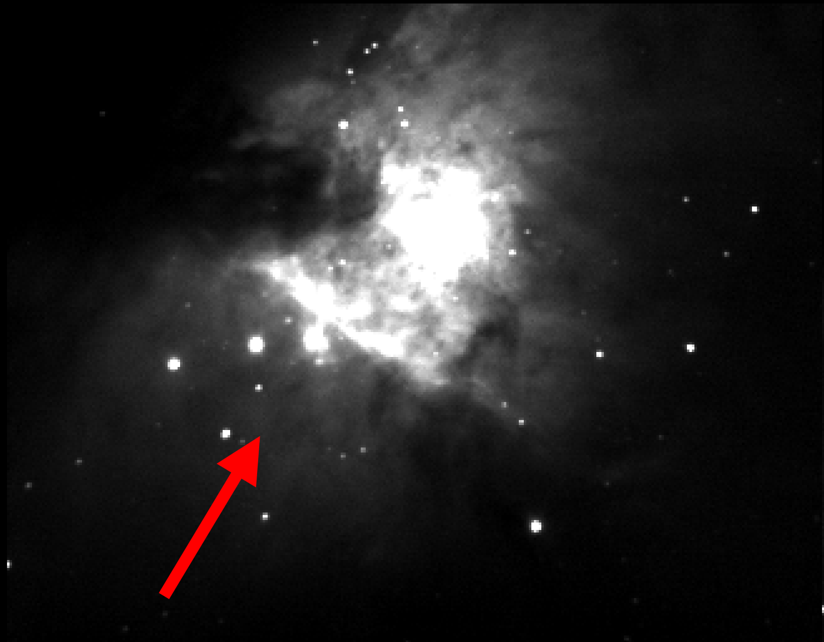
2) subtract the dark current image from the raw image;

3) subtract the dark current image from the flat field image;

4) divide the dark-subtracted raw image by the dark-subtracted flat image.

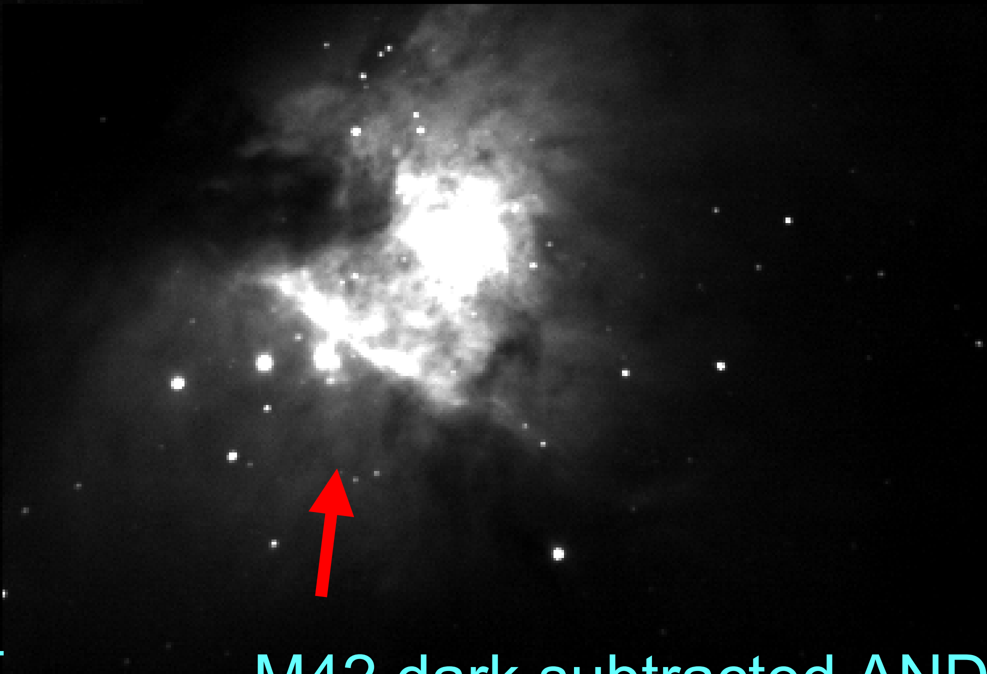
*two examples of M42 are shown on the next slide...*

*Raw - Dark*



M42, raw image, dark current subtracted out

$$\frac{\textit{Raw – Dark}}{\textit{Flat - Dark}}$$

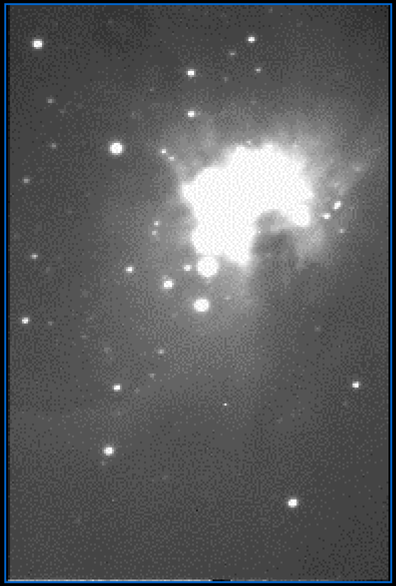


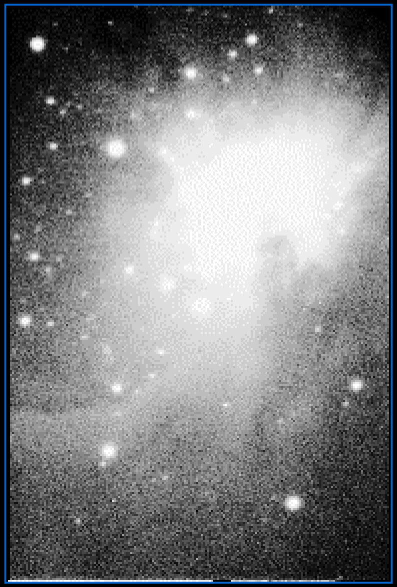
M42 dark subtracted AND flat fielded

***It may not be obvious to the naked eyeball, but the image on the right is the corrected image, ready for whatever further processing you want to do...***

*Which brings us to the next topic:*

# 3. Ways of Displaying your Image to Enhance Certain Features





Once you have dark-subtracted and flat-fielded your image, there are many techniques you can use to enhance your signal, once you have eliminated the noise.

These manipulations fall into two basic categories:

1) CHANGING THE WAY IN WHICH THE INFORMATION IS PLOTTED ON YOUR SCREEN

these methods are basically mapping routines, and include:

- * limiting the visualization thresholds within the histogram
- * plotting the image using different transfer functions
- * histogram equalization

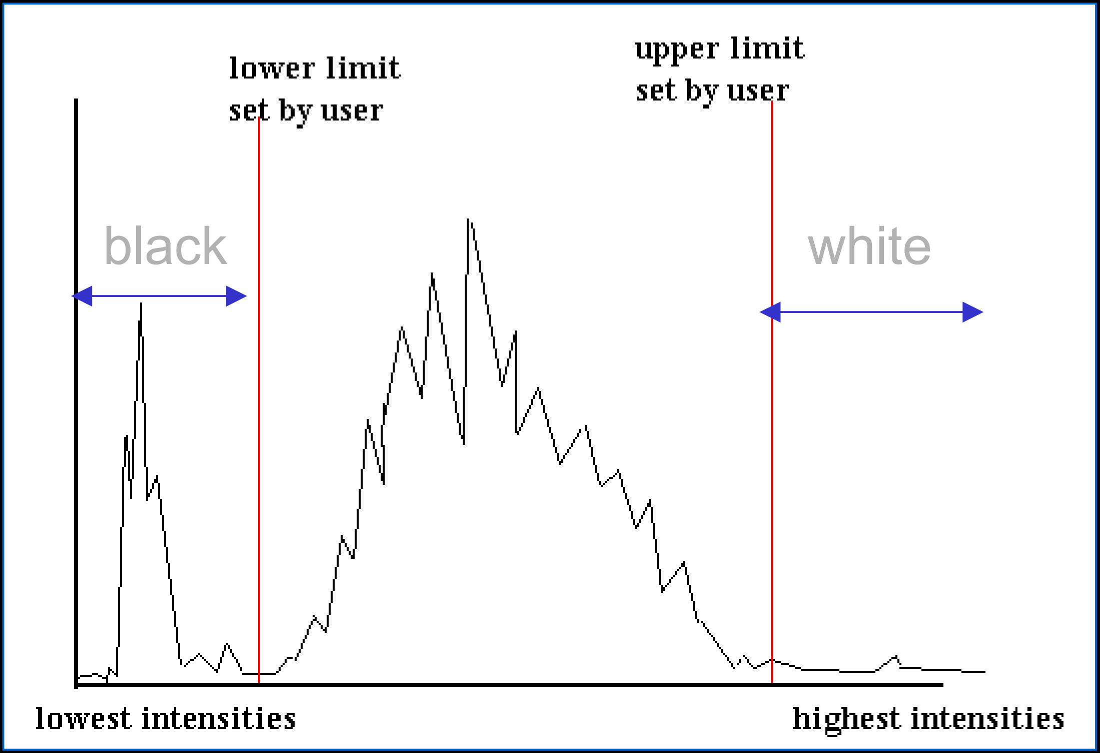
2) MATHEMATICAL METHODS OF MASSAGING YOUR DATA

these methods employ various matrix multiplications, Fourier transforms, and convolutions, and we will address them in the next section.

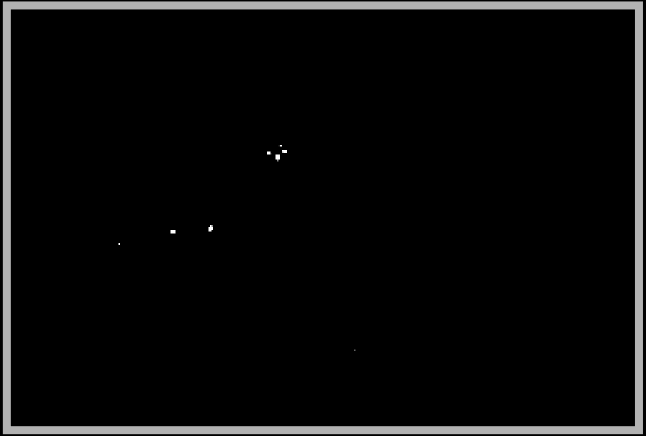
# Limiting the visualization thresholds within the histogram

- We already saw that the histogram function shows you the distribution of brightness values in an image, and the number of pixels with the same brightness value.

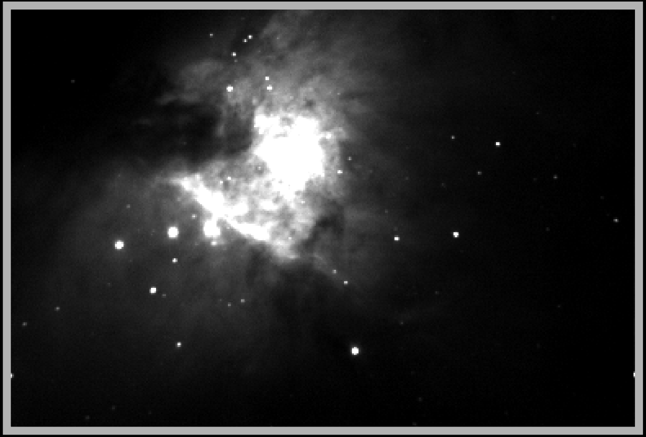
In the histogram shown at right, most of the useful information is contained between the user-defined limits.
The peak of intensities on the lower end could possibly be some faint feature, which could be enhanced in a variety of ways...



- By changing the visualization limits in the histogram, the user can pre-define the black and white levels of the image, thus increasing the level of detail available in the mid-ranges of the intensities in a given image.

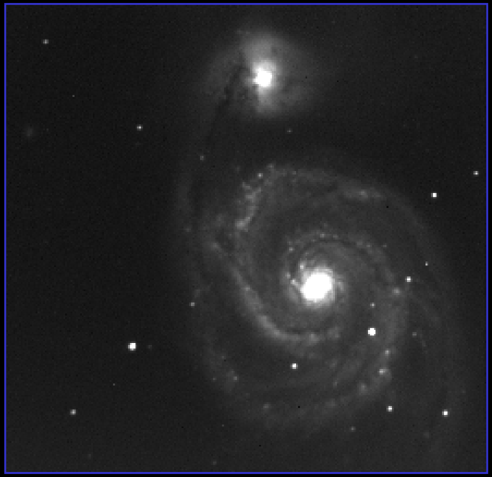


Here is M42 shot with an ST7 camera at Virginia Polytechnic Institute, utilizing the maximum intensity range from 0 to 4095. Only the brightest stars are visible in this rendition, including the hot young stars of the Trapezium.

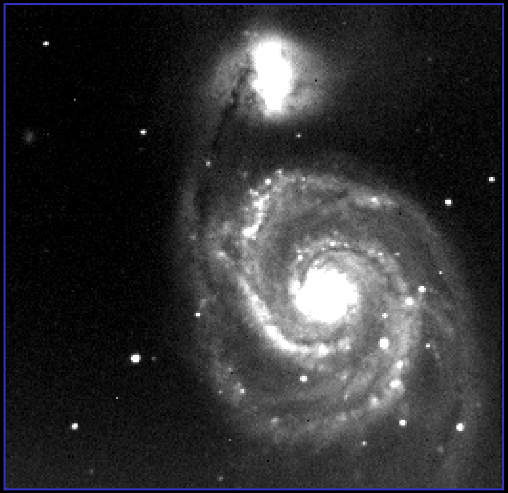


Here is the same image, this time with the histogram delimited so that all values below 162 are set to black, and all values above 1718 are set to white.
Lo, and behold: the nebula is now visible!

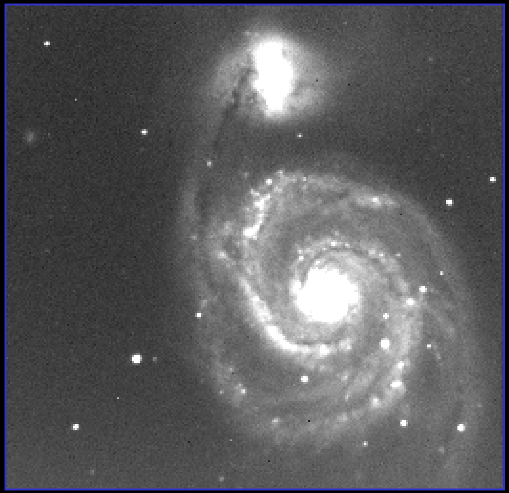
- Here are some examples of histogram limitation is used to examine different features of M51:



pixel range: 0 to 4095
the full range



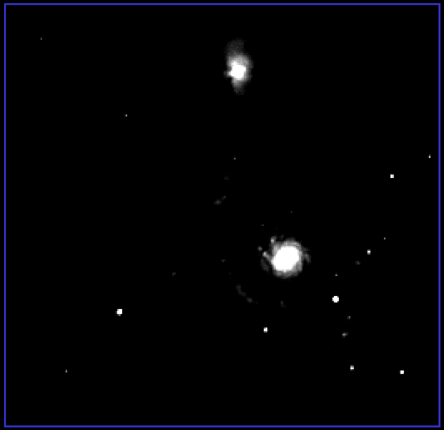
pixel range: 178 to 819
standard deviation



pixel range: 0 to 819
low intensities

pixel range: 819 to 4095
high intensities

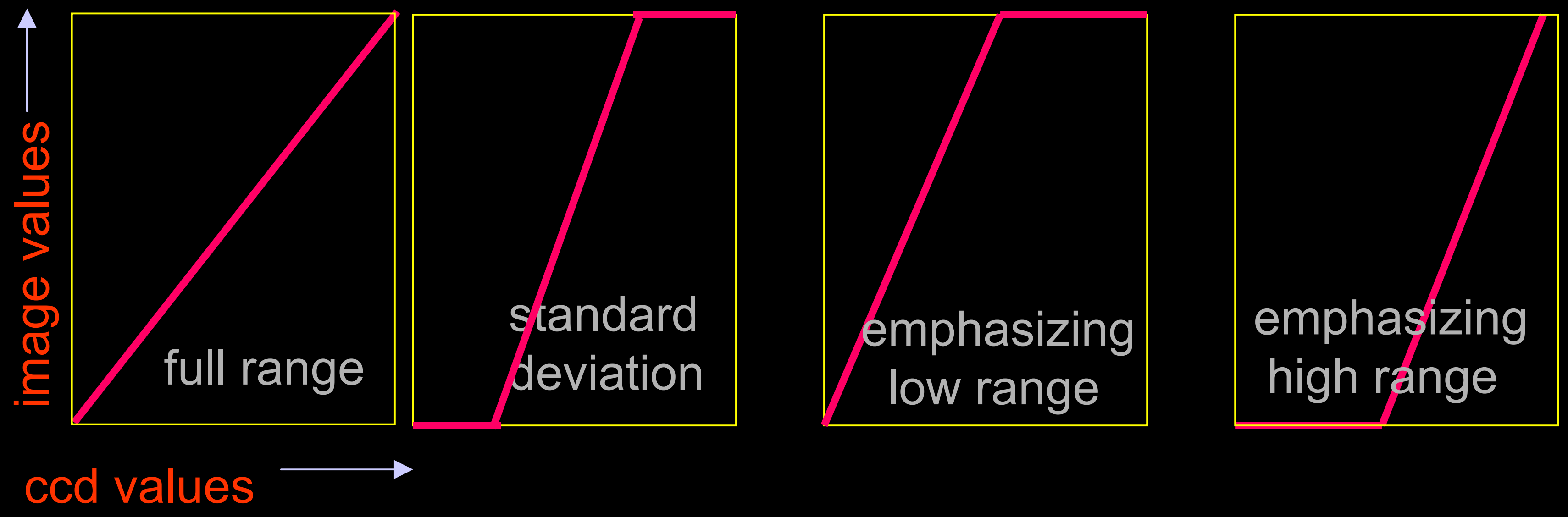
*what happened to the galaxy?*



***Notice how each range allows you to see different features of the data...***

# Plotting the image using different transfer functions

A transfer function maps the pixel values from the CCD to the available brightness values in the imaging software



All the images so far have been plotted using linear transfer functions.

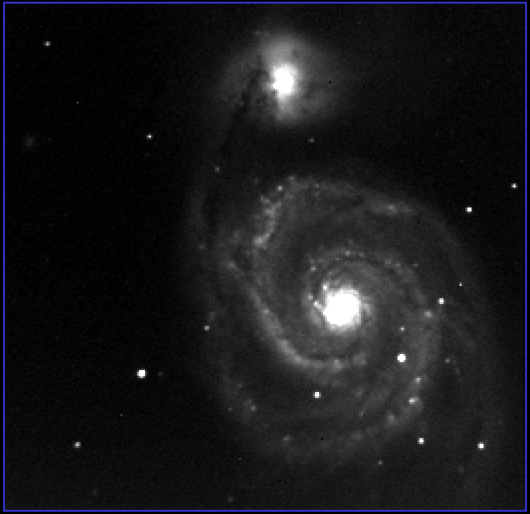
# ...but you can also use non-linear scaling

- Human eyes see a wide range of intensities because our
- vision is *LOGARITHMICALLY* scaled.
- When you plot digital images logarithmically, it allows you to see a broader range of intensities, and can give a
- more “natural” look... as if you could see the object with
- your naked eyes.

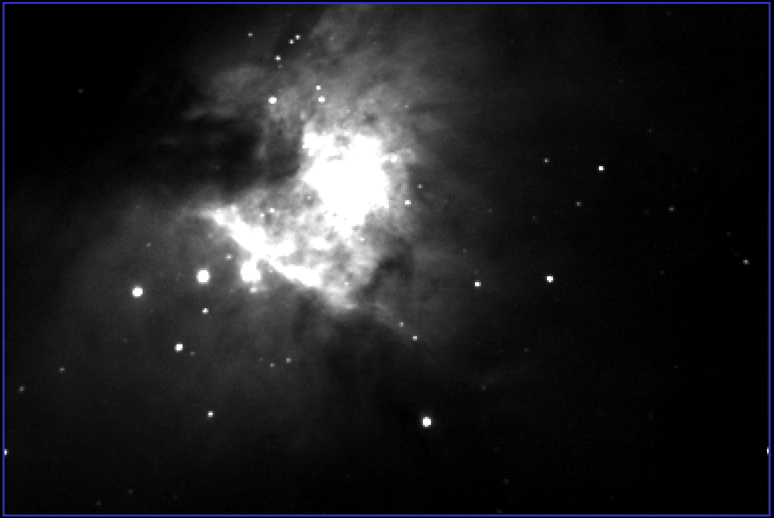
Compare linear, standard deviation plots of M51 and M42 with logarithmic plots on the next slide... →

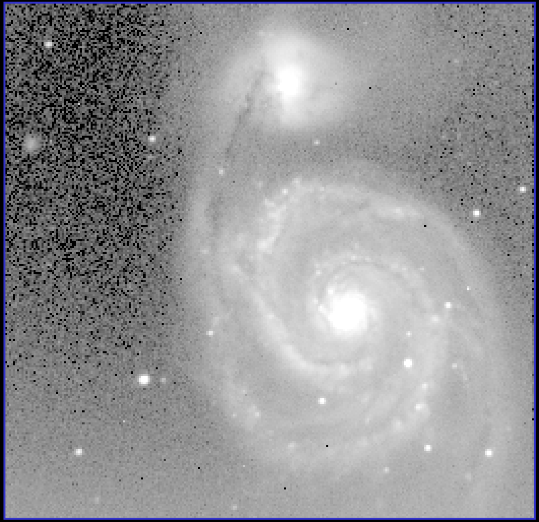
M51

M42

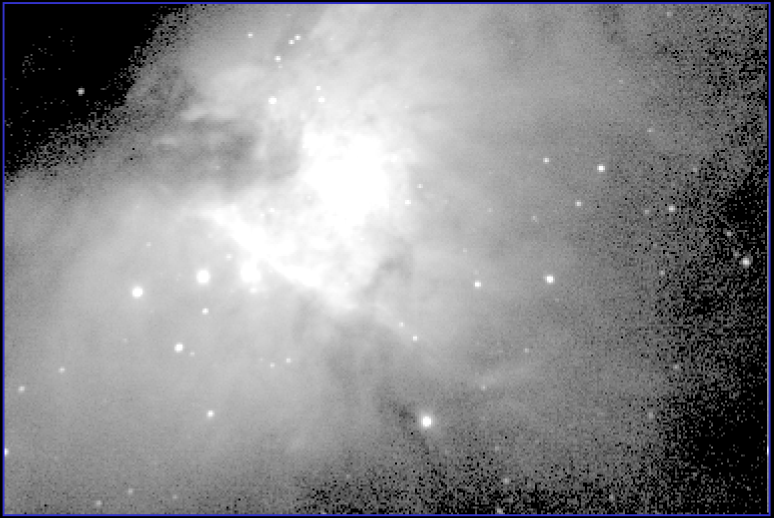


LINEAR

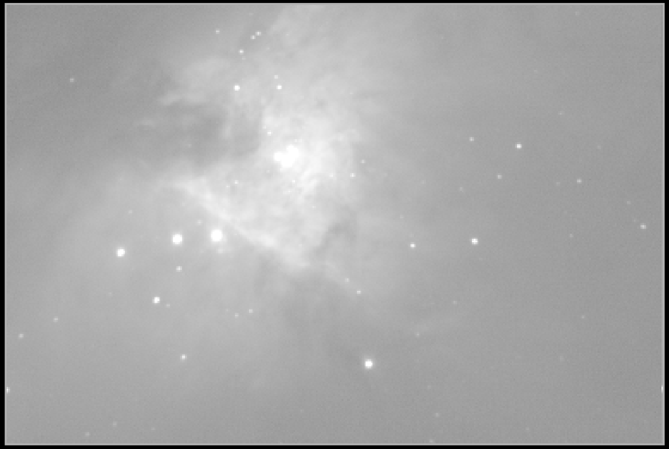




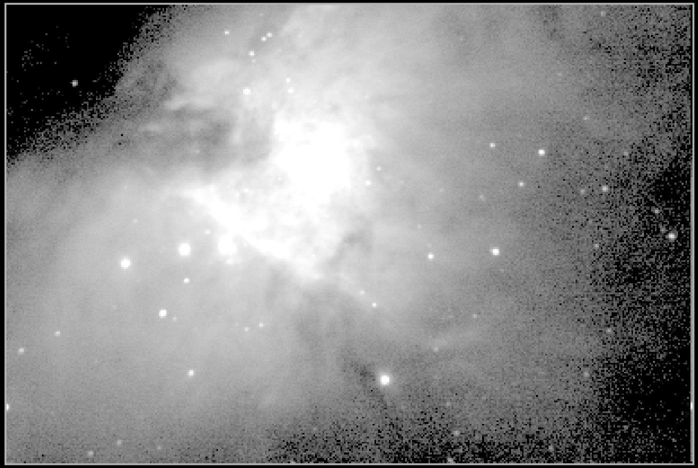
LOGARITHMIC



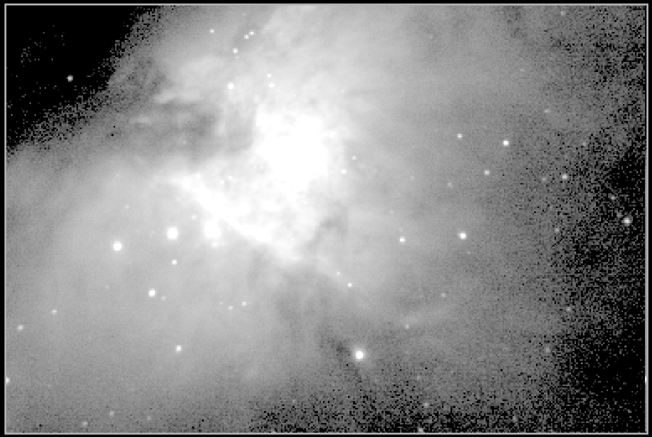
- You can also apply a combination of linear and logarithmic scaling to enhance different features...



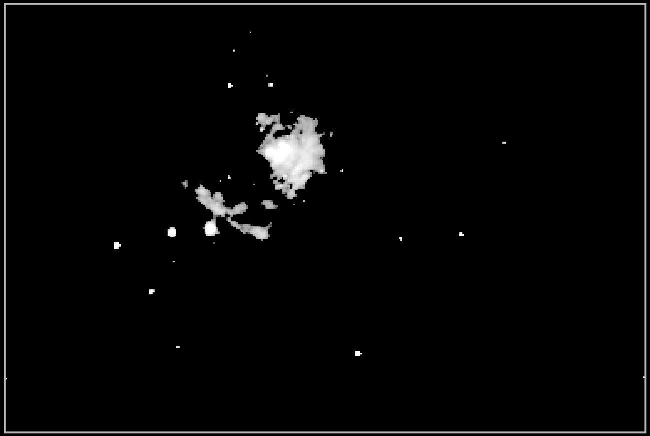
log plot of full range of values



standard deviation applied after log plot

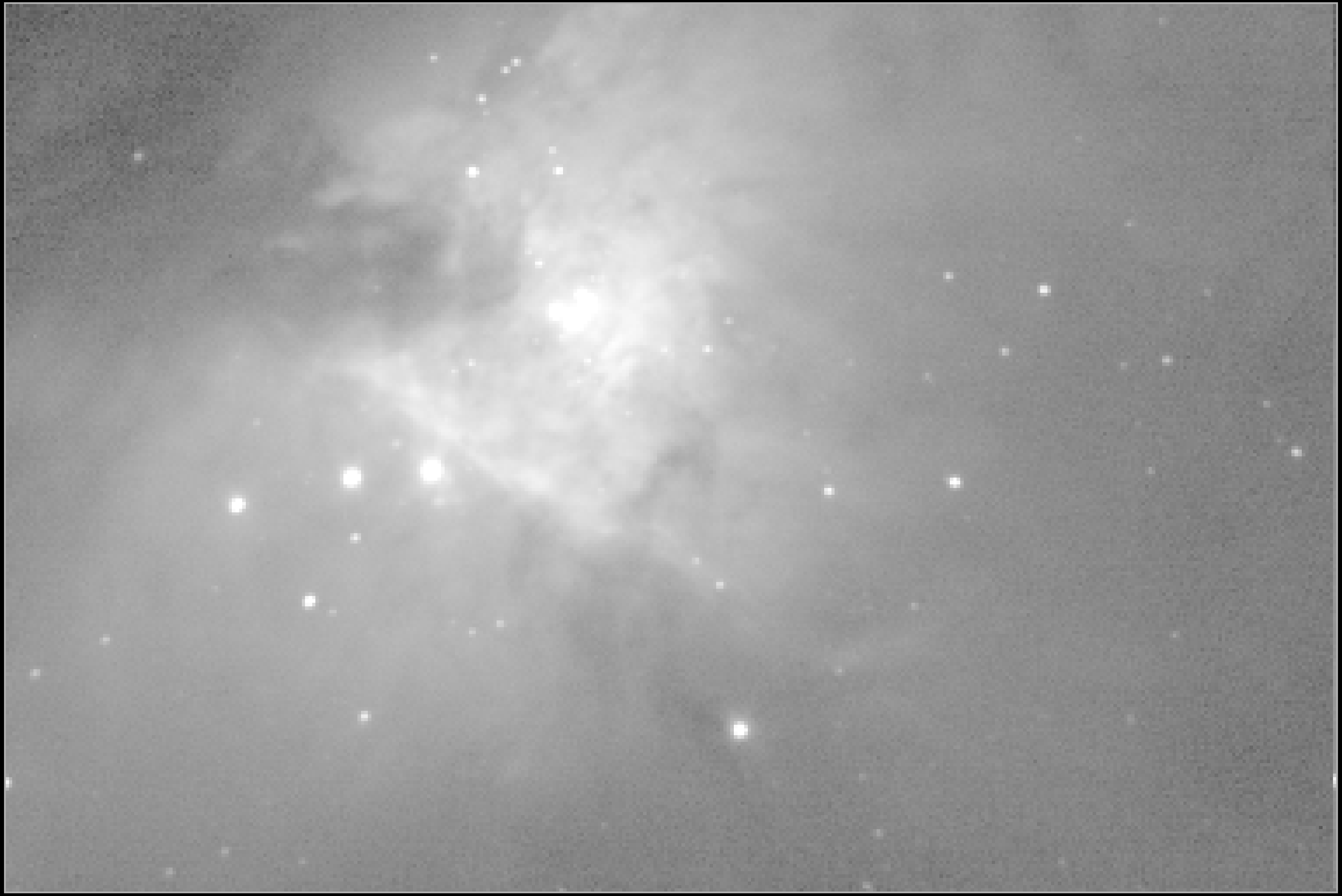


standard deviation applied before log plot



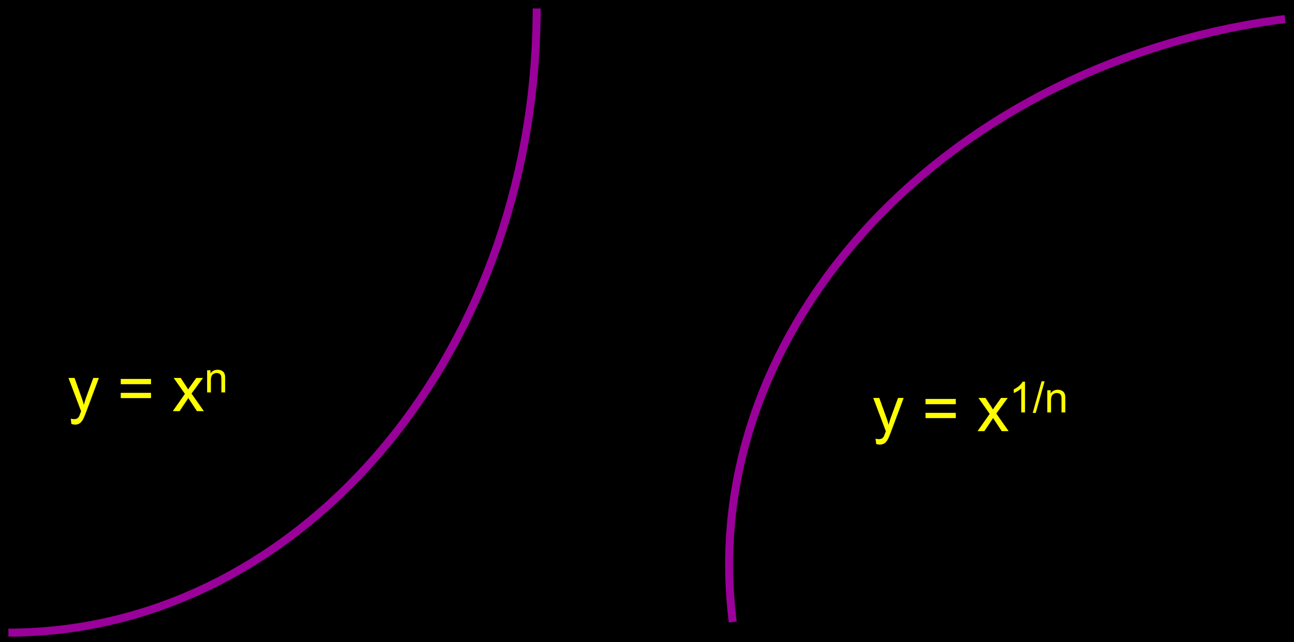
high range only, log plot

- Here is M42, logarithmically plotted with black set to 100 and white set to 4000. Notice the whispy nebulosity that is enhanced in this plot:



# ...or, you could use a Power Law scaling

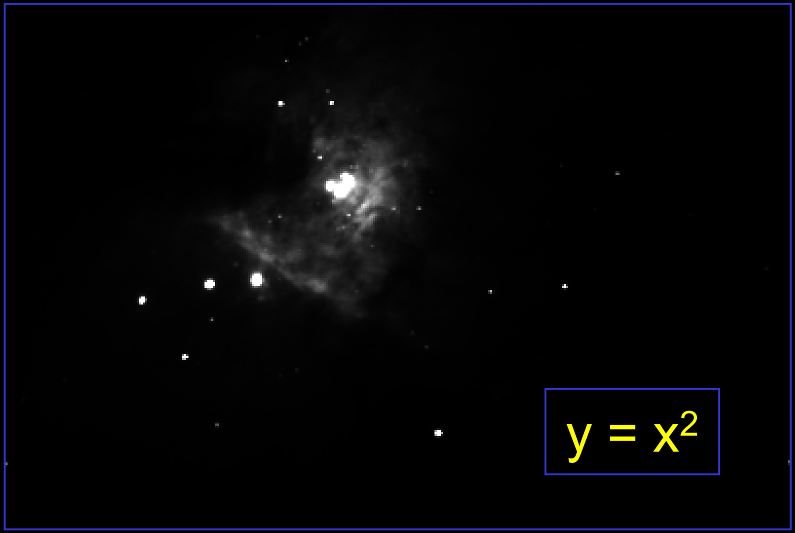
- Fractional powers enhance low intensity features, while
    - powers greater than 1 enhance high intensity features.



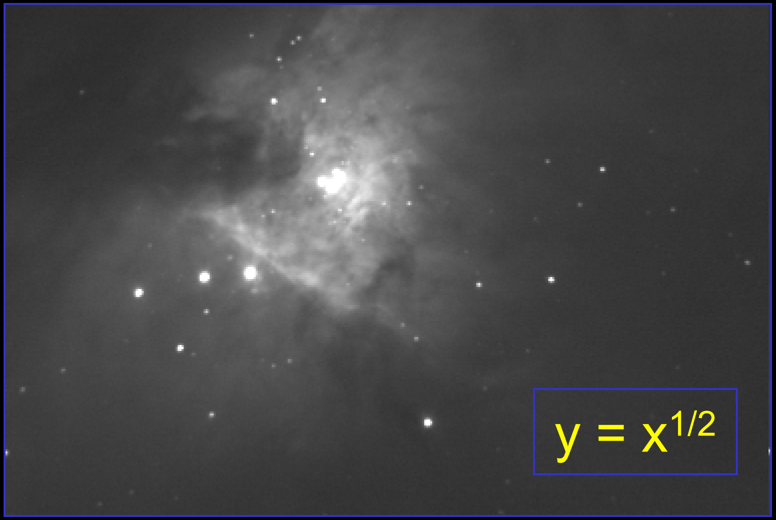
see next slide...

→

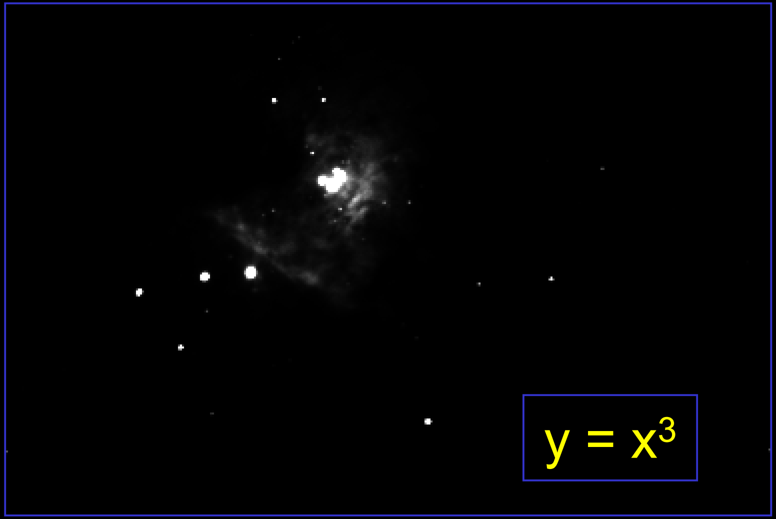
# Examples of power law scaling using M42



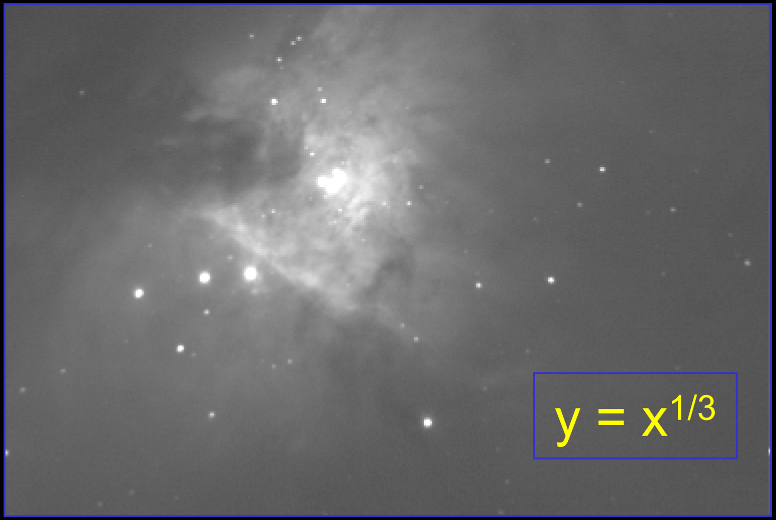
$y = x^2$



$y = x^{1/2}$

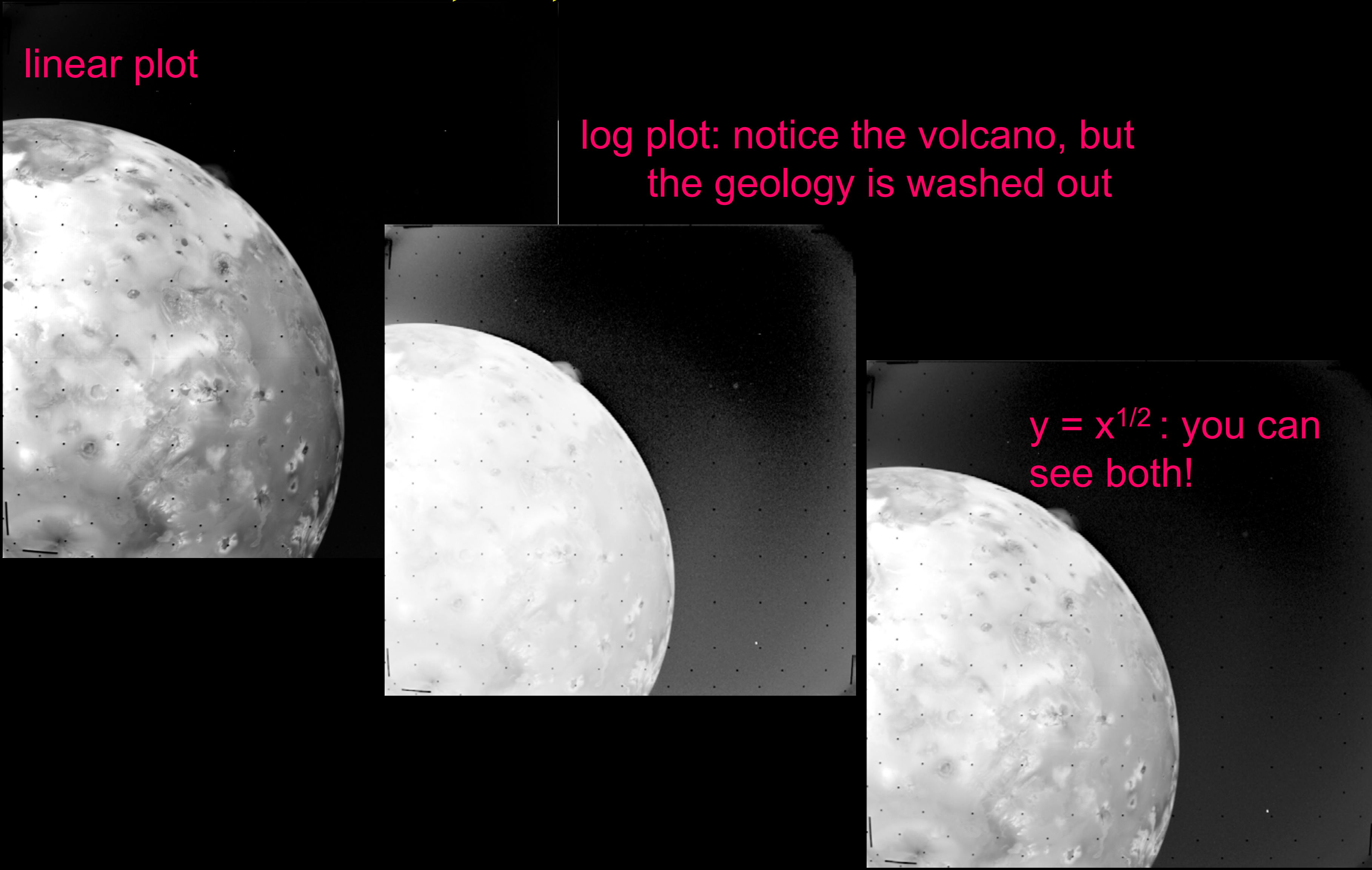


$y = x^3$



$y = x^{1/3}$

# Different transfer functions bring out faint planetary features, too, like this volcano on Io



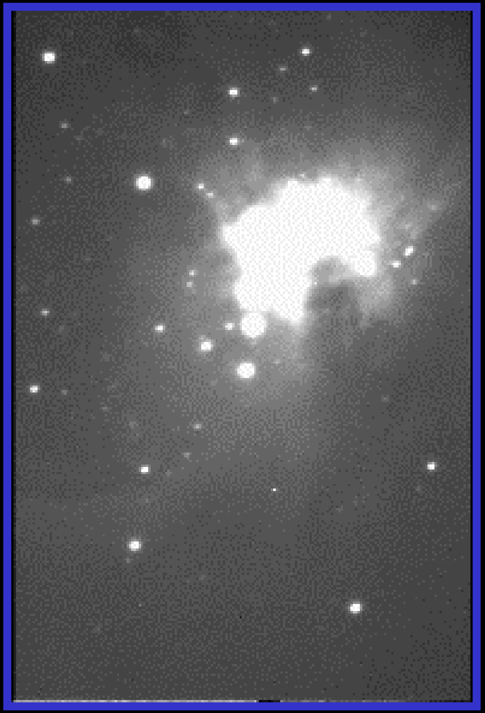
linear plot

log plot: notice the volcano, but the geology is washed out

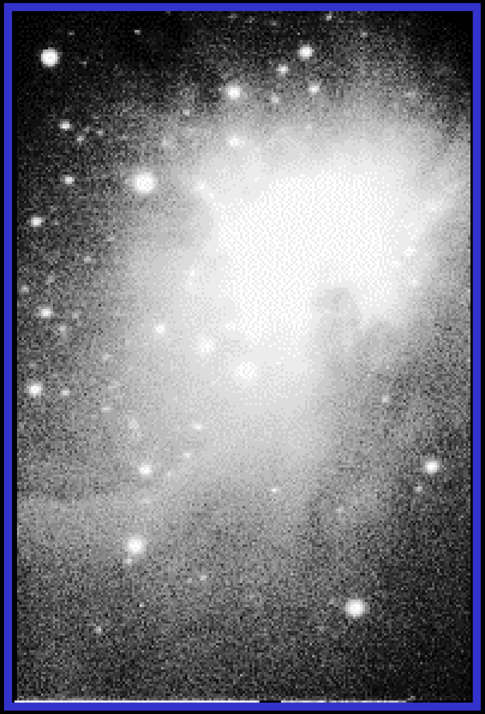
$y = x^{1/2}$ : you can see both!

# HISTOGRAM EQUILIZATION

- ...a means of flattening your histogram by putting equal numbers of pixels in each “bin”, it  serves to enhance mid-range features in an image with a wide range of intensity values:

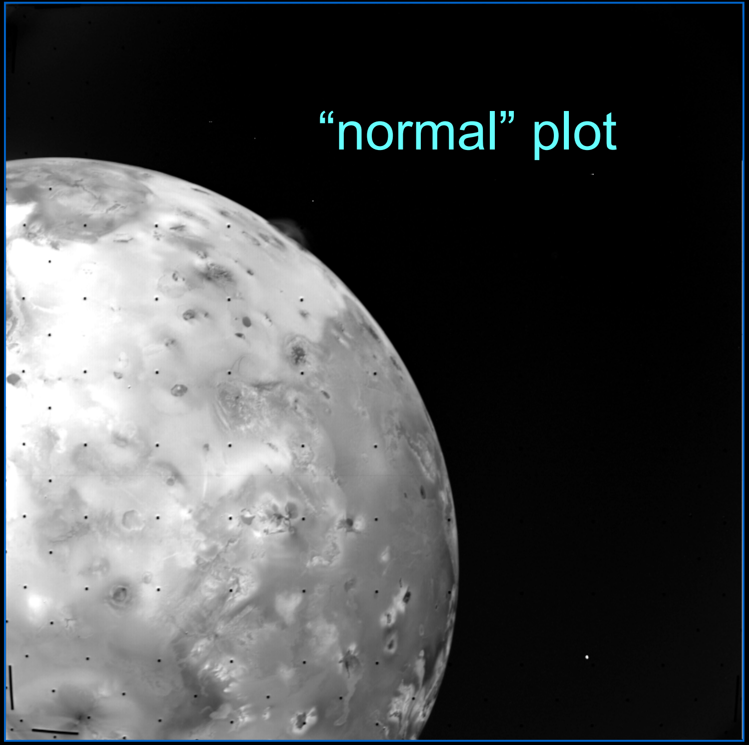


“normal”

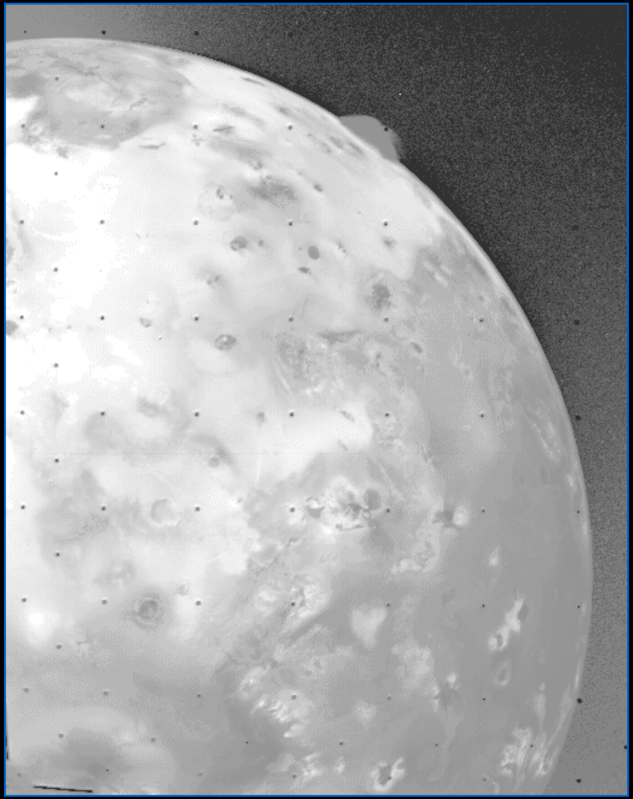


“histogram equalized”

- When you equalize your histogram, you distribute your 4096 intensity values from your CCD equally among the intensity values available in your software. This can be particularly useful for bringing out features that are close to the sky background, which would otherwise be lost.

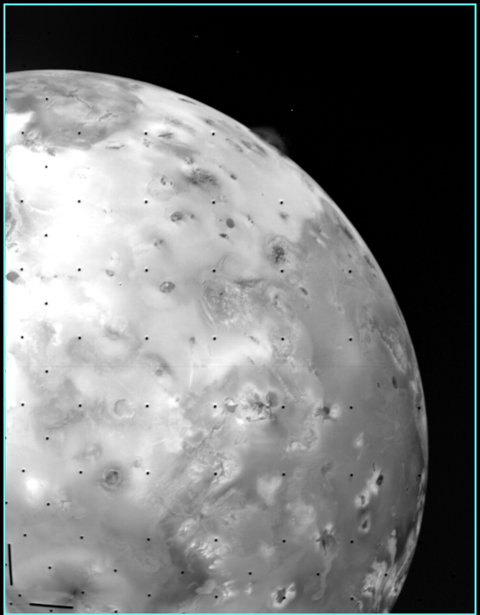


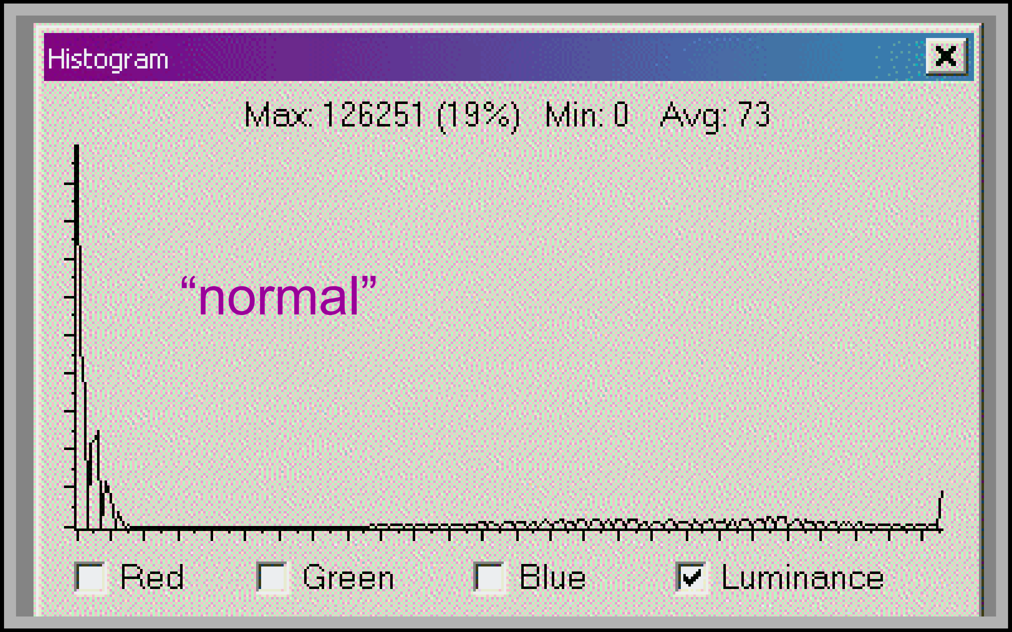
“normal” plot



With histogram equalization the surface features, as well as the volcano, are visible.

The histograms for these images are shown on the next slide.



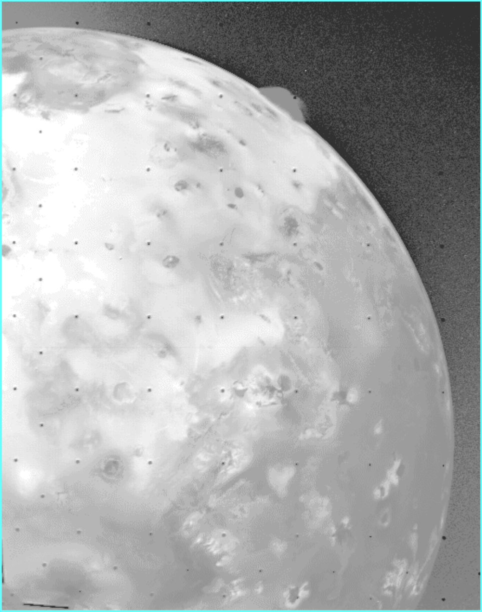


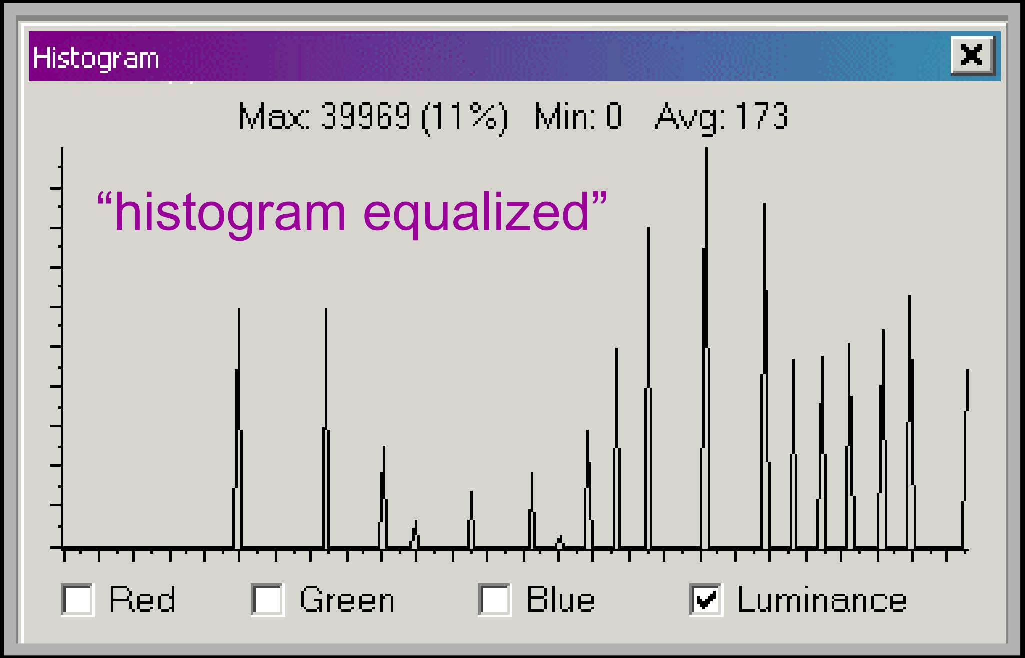
Histogram

Max: 126251 (19%) Min: 0 Avg: 73

“normal”

□ Red □ Green □ Blue ☑ Luminance





Histogram

Max: 39969 (11%) Min: 0 Avg: 173

“histogram equalized”

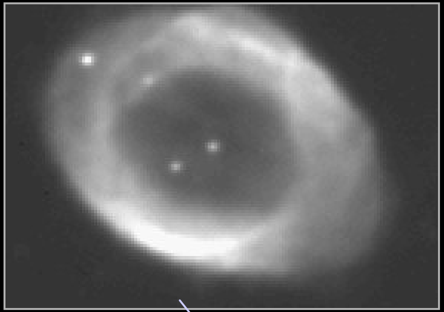
□ Red □ Green □ Blue ☑ Luminance

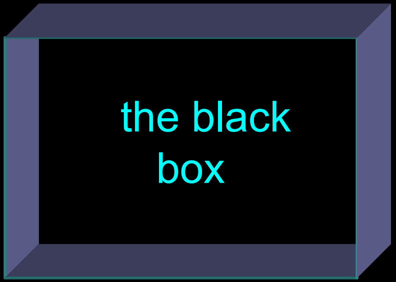
After you have corrected your raw image so that you are confident that what you are seeing really comes from incident photons and not electronic noise of your CCD, you may still have unwanted components in your image.

It is time now to perform mathematical operations on your signal which will enhance certain features, remove unwanted noise, smooth rough edges, or emphasize certain boundaries.

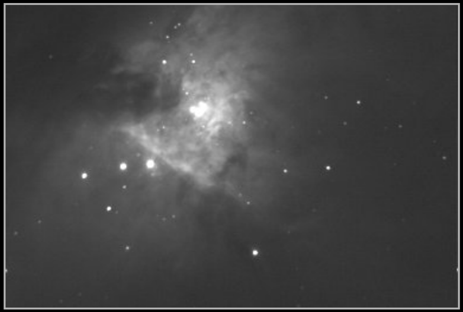
- *…and this brings us to our last topic for this module:*

# Filters, Masks and other Mathematical Manipulations



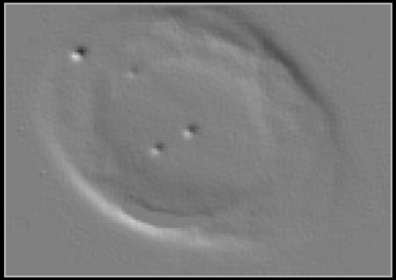


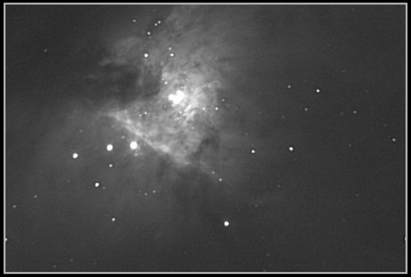
the black box



| | | |
|---|---|---|
| -2 | -1 | 0 |
| -1 | 0 | 1 |
| 0 | 1 | 2 |

| | | |
|---|---|---|
| -1 | -1 | -1 |
| -1 | 9 | -1 |
| -1 | -1 | -1 |





# the rhyme and reason

- Basically, ANY signal contains information of varying frequencies and phases. In digital signal enhancement, we attempt to accentuate the components of that signal which carry the information we want, and reduce to insignificance those components which carry the noise.

- Audio equipment, such as your stereo or CD player, has filters which do this for one-dimensional, time-varying audio signals.

- In digital image analysis we extend these techniques to 2-dimensional signals which are spatially varying.

  - In any case, the basic idea is the same:
    - *get the most out of your data*
    - *for the least amount of hassle!*

# Here’s How it Works:

- You create an “n x n” matrix of numbers, such as 3 x 3 or 5 x 5, and you move this across your image, like a little moving window, starting at the upper left corner (that’s 0,0, recall).
- You “matrix multiply” this with the pixel values in the image directly below it to get a new value for the center pixel.
- You move the window across your image, one pixel at a time, and repeat the operation, until you have changed
- the appearance of the entire image.

*This process is called CONVOLUTION.*

# Here’s one example that is easy to visualize:

Convolution of an image with this matrix will replace the center pixel in the window with the weighted average intensity of all the points located inside the window:

$$\boxed{\begin{array}{ccc}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{array}}$$

$$\begin{array}{ccccc}I_{11} & I_{12} & I_{13} & I_{14} & I_{15} \ldots\\ I_{21} & I_{22} & I_{23} & I_{24} & I_{25} \ldots\\ I_{31} & I_{32} & I_{33} & I_{34} & I_{35} \ldots\\ I_{41} & I_{42} & I_{43} & I_{44} & I_{45} \ldots\\ I_{51} & I_{52} & I_{53} & I_{54} & I_{55} \ldots\end{array}$$

You plop this window down over a 3 x 3 section of your image, do a matrix multiplication on the center pixel, $I_{22}$ in this example, and the new value for $I_{22}$ which is returned is:

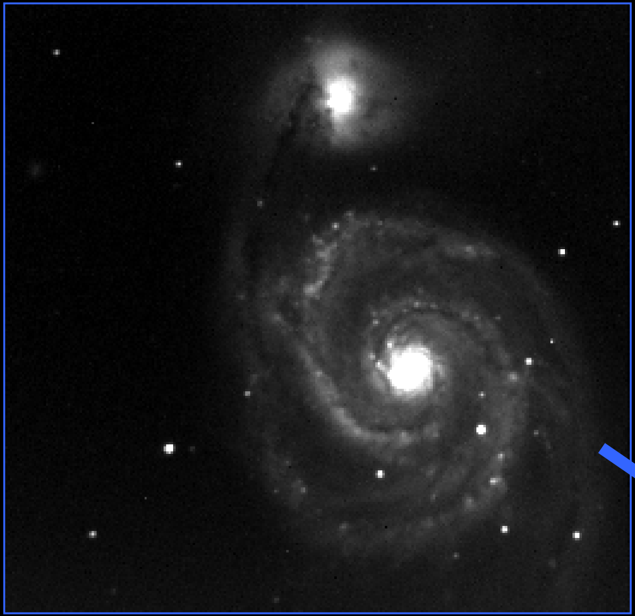
$$\boxed{I_{22}' = \frac{I_{11} + 2\,(I_{12}) + I_{13} + 2(I_{21}) + 4(I_{22}) + 2(I_{23}) + I_{31} + 2(I_{32}) + I_{33}}{16}}$$

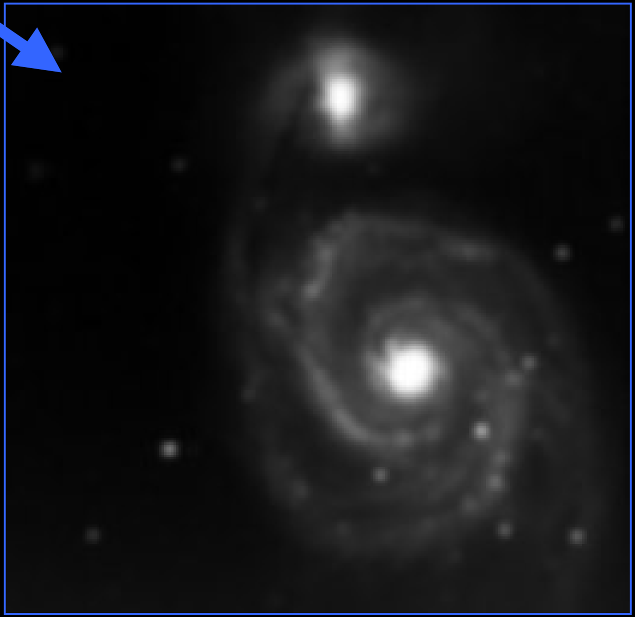
Imagine your image in 3 dimensions, where the intensity is plotted as an elevation. Features with low spatial frequency will appear as hills and valleys, while high frequency objects like stars will appear as sharp spikes. Some features will have steep gradients, while others shallow. Some features may have jagged edges, while others are smooth.

You can design little n x n windows to investigate and enhance these kinds of features of your image.

In the slides that follow, we will show examples of Low Pass, High Pass, Edge Detection, Gradient Detection, Sharpening, Blurring, Bias filtering your image.

# Low-pass filters enhance the larger scale features in your image:





The image acquires a blurry look.

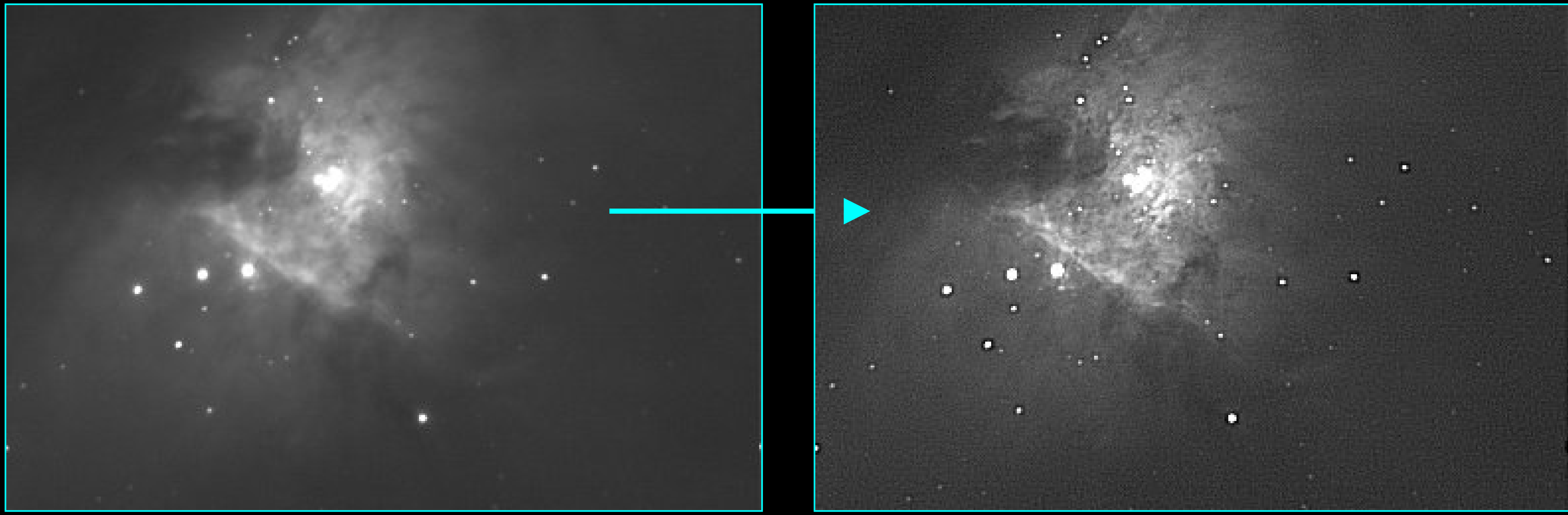
Some examples of low-pass filters:

$$\boxed{\begin{matrix} 0 & 1 & 0 \\ 1 & 16 & 1 \\ 0 & 1 & 0 \end{matrix}}$$

$$\boxed{\begin{matrix} 1 & 1 & 1 \\ 1 & 12 & 1 \\ 1 & 1 & 1 \end{matrix}}$$

$$\boxed{\begin{matrix} 1 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 1 \end{matrix}}$$

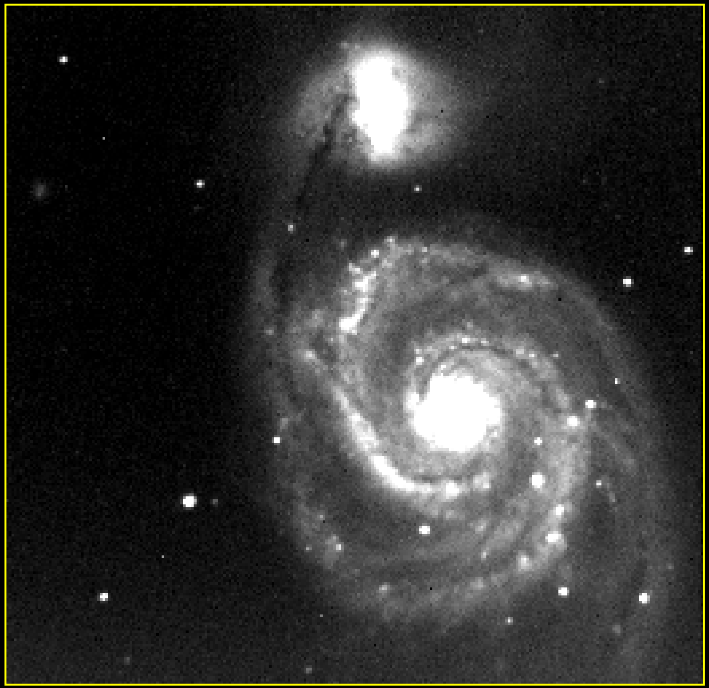
- High pass filters enhance the short period features in your image, giving it a sharper look.



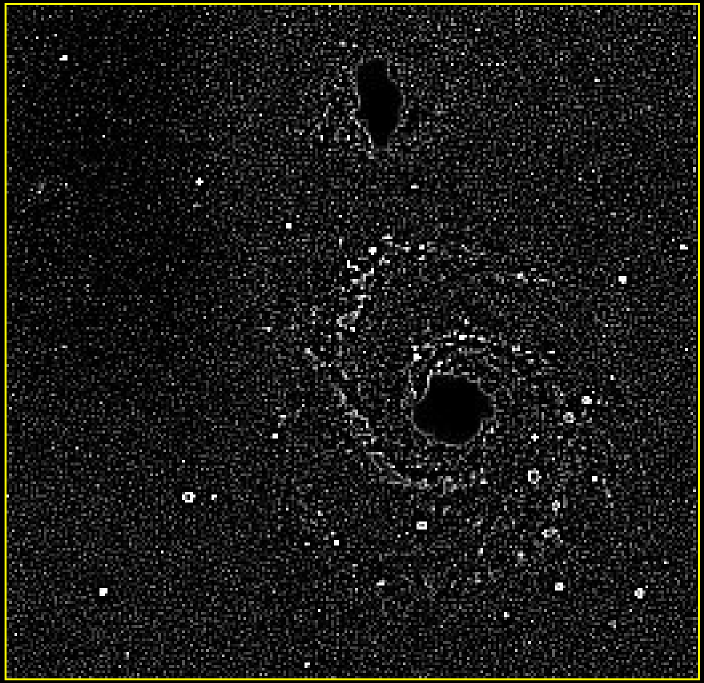
Some examples of high pass filters:

$$\boxed{\begin{array}{ccc} 0 & -1 & 0 \\ -1 & 20 & -1 \\ 0 & -1 & 0 \end{array}} \quad \boxed{\begin{array}{ccc} 0 & -1 & 0 \\ -1 & 10 & -1 \\ 0 & -1 & 0 \end{array}} \quad \boxed{\begin{array}{ccc} -1 & -1 & -1 \\ -1 & 10 & -1 \\ -1 & -1 & -1 \end{array}} \quad \boxed{\begin{array}{ccc} -1 & -1 & -1 \\ -1 & 16 & -1 \\ -1 & -1 & -1 \end{array}}$$

- Edge Detection filters are used to locate the boundaries between regions of different intensities.



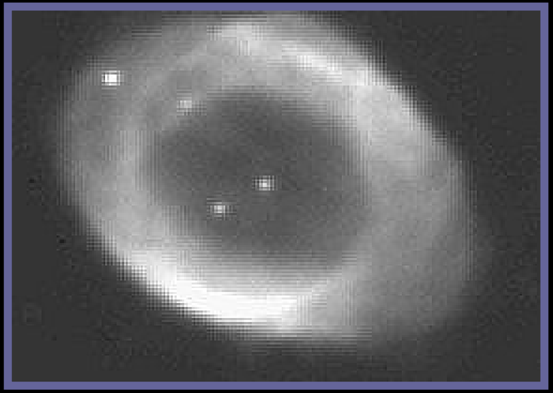
$$\begin{matrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{matrix}$$

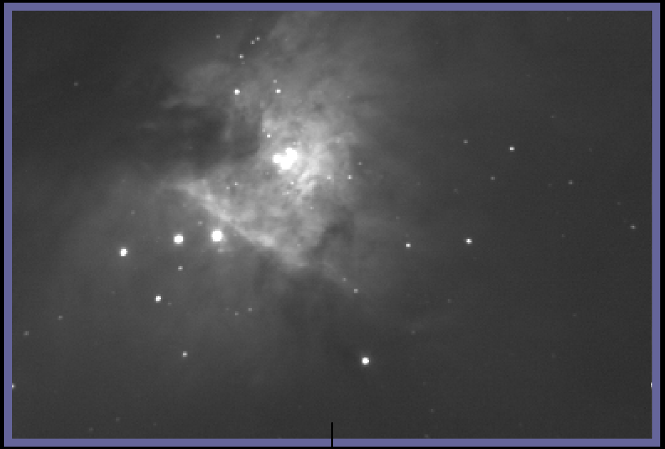


These techniques can be used on other types of images, too!

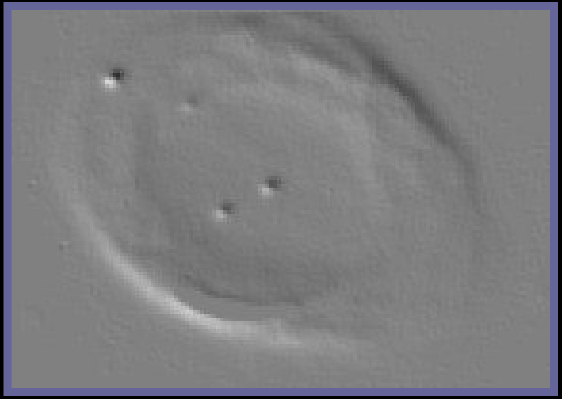
*...such as this one of the author and her dog Tasha with their Bright Blue Ford pickup truck, in Santa Barbara, California*

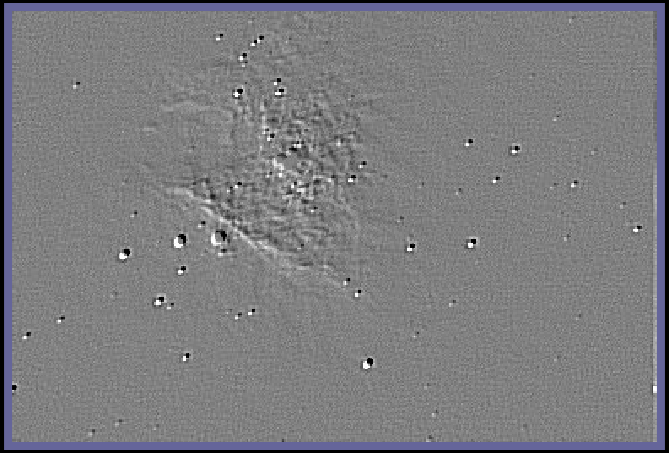
# A “bias filter” makes an image look like a bas relief with shadows. This can be useful for examining certain details.





$$\begin{array}{ccc} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{array}$$





# You can also combine processes – such as low pass,high pass, and image subtraction in a process called UNSHARP MASKING.

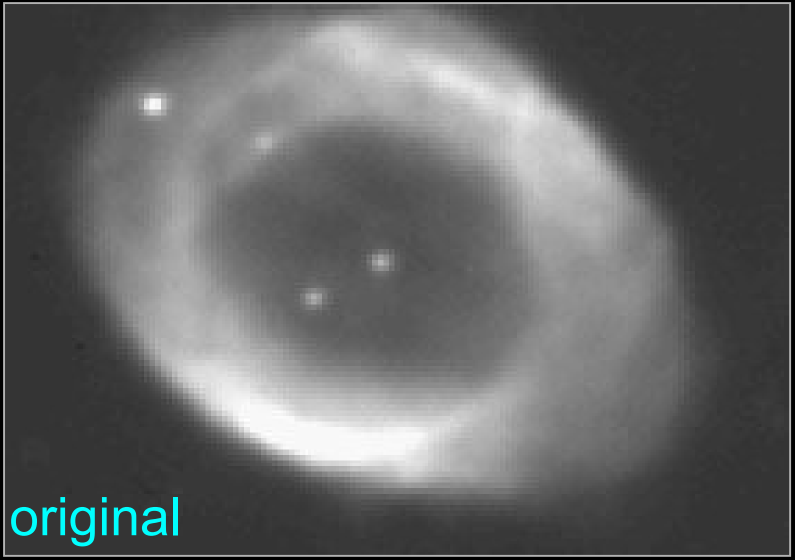
Unsharp masking consists of a 3-step process:

1) Make a copy of the image where each pixel is the average of the group of pixels surrounding it, so that the large features are not disturbed, but the small ones are blurred. (This is the unsharp mask.)

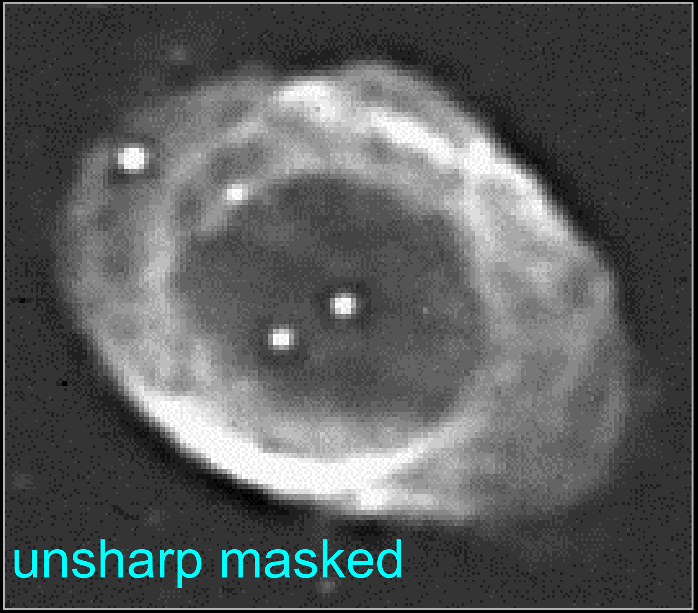
2) The pixel values of the original image are multiplied by a constant ("A"), and then the pixel values of the unsharp mask are subtracted from this one or more times ("B"). In this way, the large features are not changed by much, but the small ones are enhanced.

3) Finally a low-pass filter is applied to the result.

# Here is an example of how unsharp masking can reveal details that were previously hidden:



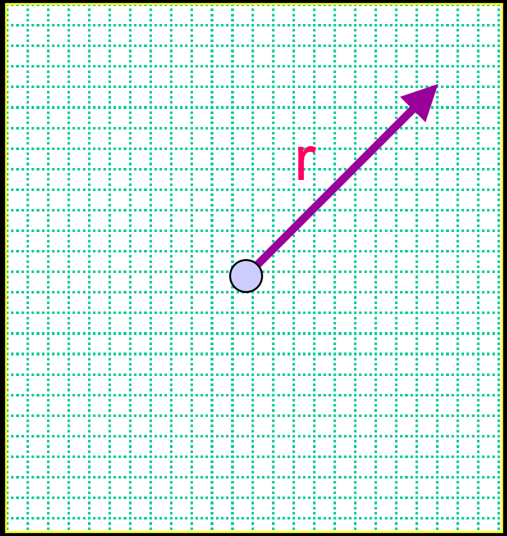
*3 x (original) – (average) then convolved with a 9x9 low-pass filter gives the final product:*



Note the whispy filaments in this planetary nebula that become apparent after processing.

- You can also create your own filter to smooth or otherwise operate on your image.
- Here is a 9 x 9 Gaussian filter that I created to smooth the statistical noise expected in the Cosmic Microwave Background from a “horn” antenna with a 10° beam width back in the “COBE” days.
- (Today’s microwave antennas measure angular sizes of less than 1°.)

$$G_{IJ} = e^{(-1/2(r/\sigma)2}$$



where $r = (x^2 + y^2)^{1/2}$ is the distance from the center pixel to each pixel in the 9x9 grid and $\sigma$ is the beam width, 10°.

Here’s how it looks: →

$G_{IJ} = e^{(-1/2(r/\sigma)2}$
where $r = (x^2 + y^2)^{1/2}$ is the distance from the center pixel to each pixel in the 9x9 grid and $\sigma$ is the beam width of 10°.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 5.657 | 5 | 4.472 | 4.123 | 4 | 4.123 | 4.427 | 5 | 5.657 |
| 5 | 4.423 | 3.606 | 3.162 | 3 | 3.162 | 3.606 | 4.423 | 5 |
| 4.472 | 3.606 | 2.828 | 2.236 | 2 | 2.236 | 2.828 | 3.606 | 4.472 |
| 4.123 | 3.162 | 2.236 | 1.414 | 1 | 1.414 | 2.236 | 3.162 | 4.123 |
| 4 | 3 | 2 | 1 | 1 | 1 | 2 | 3 | 4 |
| 4.123 | 3.162 | 2.236 | 1.414 | 1 | 1.414 | 2.236 | 3.162 | 4.123 |
| 4.472 | 3.606 | 2.828 | 2.236 | 2 | 2.236 | 2.828 | 3.606 | 4.472 |
| 5 | 4.423 | 3.606 | 3.162 | 3 | 3.162 | 3.606 | 4.423 | 5 |
| 5.657 | 5 | 4.472 | 4.123 | 4 | 4.123 | 4.427 | 5 | 5.657 |

The effect of this filter is to smooth out random variations that are expected to be less than the size of the grid, in terms of pixels. *Moral of the story: you can create any filter to suit your needs!* *Just be careful not to introduce noise! But that gets into sampling theory, for which we refer you to a good book on Fourier Analysis.*

# Summary

- In this Activity, we have examined:

1. The basic definition of a digital image

2. Correcting the raw image for instrument noise

3. Highlighting features of the image by manipulating the display function

4. Changing the appearance of the image by digital filtering

# Image Credits

Images for this module were taken from the websites of The Remote Access Astronomy Project at the University of California, Santa Barbara
www.deepspace.ucsb.edu

and Sky Image Process at Virginia Tech
www.phys.vt.edu