

# AIC 2010 CCDStack Workshop

Basic concepts

What's new in CCDStack V2

Slope and Intercept: normalization and color correction

Deconvolution

# CCDStack Basic Concepts

- CCDStack is a sophisticated tool kit, not an in/out black-box
- intuitive user interface
- fast on-demand processing (batch scripting in development)
- data rejection and Missing Values
- data visualization aids:
  - stack paradigm
  - image navigation
  - Multi-speed blink of selected images
  - process This/All/Included
  - view image data with real-time WYSIWYG non-destructive scaling
  - view alignment overlays prior to resampling
  - view rejected pixels prior to stacking
  - view color adjustments prior to data modification
  - Histogram and selected area statistics

CCDStack is an advanced tool kit for astronomical image processing that provides the user with a rich set of procedures and comprehensive and flexible controls. It is not a back-box that automatically processes images. The software is primarily intended for expert level users who want to explore their data and produce optimal results using sophisticated tools. CCDStack is decidedly not an easy “app” for dummies and is geared toward the intellectually curious and adventurous. If you are uninterested in what goes on under the hood and just want something to spit-out pretty pix then you should probably go elsewhere.

A major consideration in the design and implementation of CCDStack has been the user interface. In fact, my original impetus for creating CCDStack was frustration with the clunky and at times infuriating user interfaces of the astro software offerings that were available more than a decade ago. I won't name names. CCDStack was designed specifically to stack astronomical images and the user interface is crafted to naturally assist the user.

To date, CCDStack has been a profoundly real time / on-demand program. This means that the user directs all processes as they occur. However, I am currently working on a macro type Process Manager that will allow users to create and execute scripts and well as save, reuse and edit the scripts. More on that later.

CCDStack's handling of data rejection and Missing Values uniquely distinguishes it from all other image processing software. Although the concept of missing values is implicit in some data cleaning schemes such as “sigma reject”, no other software makes it so explicit and visceral. One of my other hats is programming and analysis of large scale statistical data where missing data is a serious issue, so it was natural for me to incorporate that concept into CCDStack.

Very briefly, “missing values” means that the actual value of a data point is unknown

# CCDStack Basic Concepts

- wealth of processes, options and controls:
  - reads most formats including DSLR
  - image edit and transforms (e.g. crop re-size, rotate, mirror, duplicate)
  - advanced calibration and repair
  - pixel math and file math, including built-in C# compiler with simplified coding
  - kernel filters (Gaussian, mean, median, and mode)
  - deconvolution and unsharp masking
  - gradient handling (flattening)
  - many Data Rejection algorithms for individual image(s) or stack
  - many alignment/registration methods with visual confirmation
  - normalization
  - multiple combine methods (mean, sum, median, minimum, maximum)
  - Mean and Sum combine methods are weighted and impute rejected pixels
  - Color creation, adjustment, and conversions
  - Bayer (OSC, DSLR) processing with many options
  - more and growing...

Of course, a fancy user interface is worthless unless there is something for the user to do! And in that regard, CCDStack provides a wealth of processes and controls over those processes. Here is a partial list...

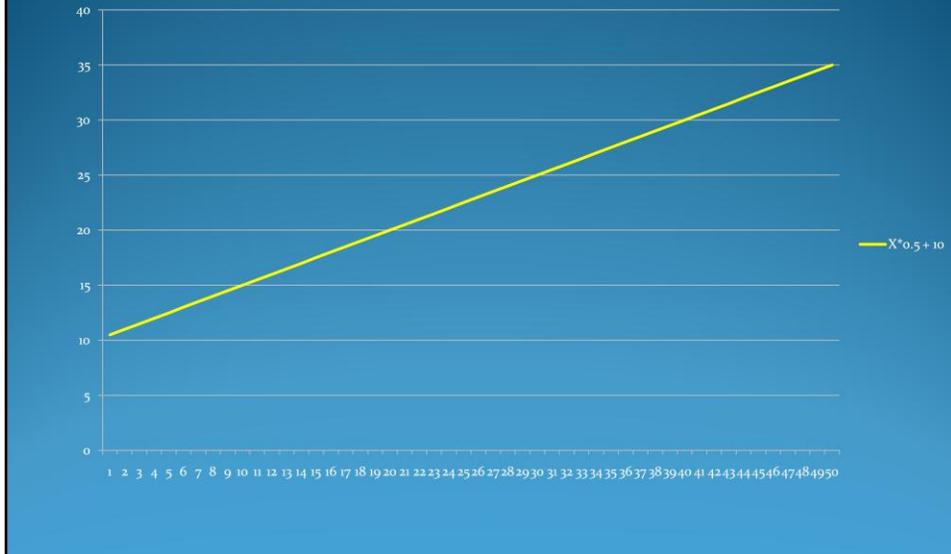
## New in CCDStack v2

- context sensitive Help - press "F1" to view Help for the active form
  - extensive parallel processing to exploit multi-core CPUs
  - deconvolution controls
  - automatic star-matching registration
  - auto-star selection procedure
  - new resample methods
  - repair bad pixel/columns
  - enhanced support for DSLR raw
  - Save and Open Stacks
  - Recent Image and Stack
  - Histogram
- near future:***
- Process Manager – macro/batch/script

Here is a partial list of features that are new in Version 2

- context sensitive Help - press "F1" to view Help for the active form
  - extensive parallel processing to exploit multi-core CPUs
  - deconvolution controls
  - automatic star-matching registration
  - auto-star selection procedure
  - new resample methods
  - repair bad pixel/columns
  - enhanced support for DSLR raw
  - Save and Open Stacks
  - Recent Image and Stack
  - Histogram
- near future:***
- Process Manager – macro/batch/script

# Slope and Intercept



Several processes used to develop astronomical images are mathematically related to the concept of slope and intercept.

Raw data from digital cameras is essentially linear, which means that the data values have a simple relationship to the number of photons collected.

Linear data is mathematically and graphically represented by a straight line that can be described using only two numbers: slope and intercept. Slope is basically the angle of the line and is usually specified as a factor of X. Intercept is the line's Y offset at X = zero. The linear equation is:

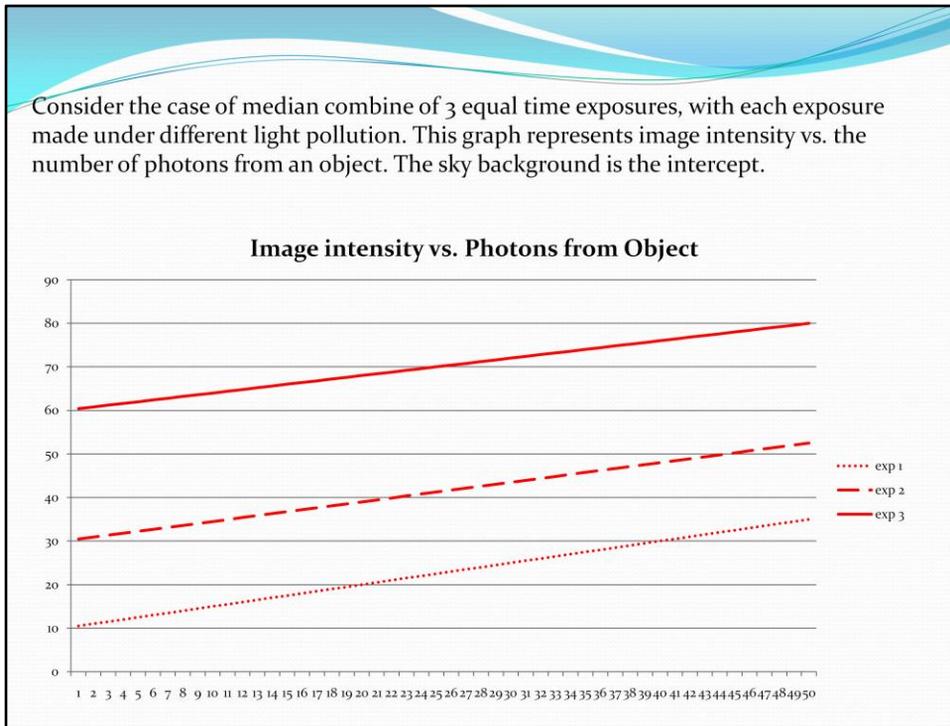
$$y = \text{slope} * x + \text{intercept}$$

## Image Normalization for Data Rejection

Data rejection methods, including median are predicated on the assumption that the various areas in each image have very similar average intensities. If one of more images violates that assumption then data rejection will malfunction and produce a defective final image. For successful data rejection it is necessary to normalize the images using the slope and intercept concept.

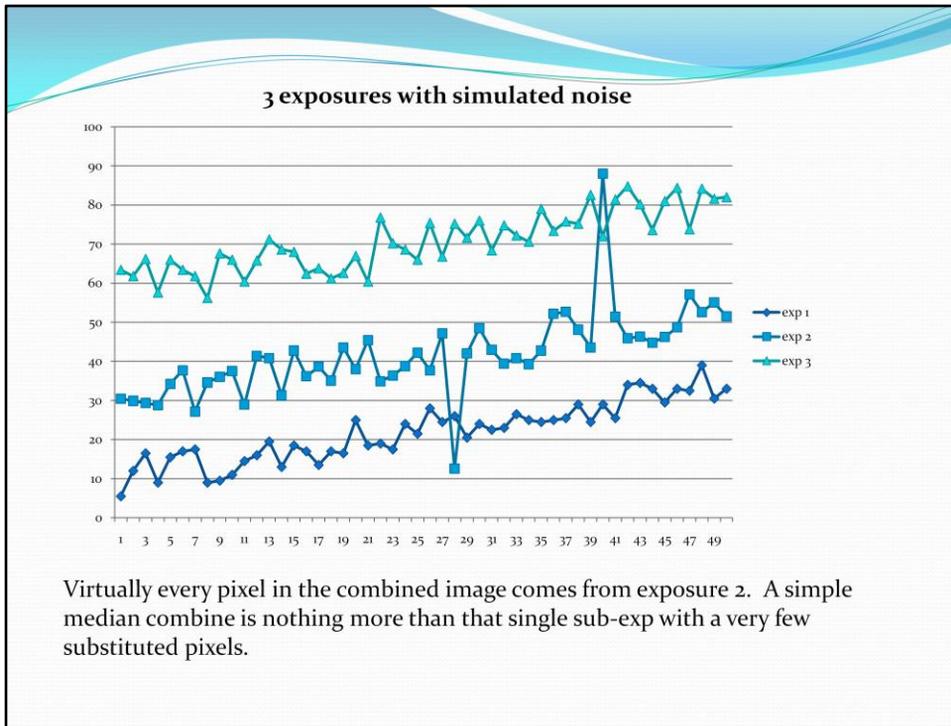
Data rejection methods, including median are predicated on the assumption that the various areas in each image have very similar average intensities. If one of more images violates that assumption then data rejection will malfunction and produce a defective final image. For successful data rejection it is necessary to normalize the images using the slope and intercept concept.

Consider the case of median combine of 3 equal time exposures, with each exposure made under different light pollution. This graph represents image intensity vs. the number of photons from an object. The sky background is the intercept.



For example, consider the case of median combine of 3 equal time exposures, with each exposure made under different moonlight pollution: before moonrise, at moonrise, and with the moon high in the sky. Obviously the sky background in image 1 will be dimmer than image 2 which is itself dimmer than image 3. This graph represents image intensity (electrons) vs. the number of photons from an object (gathered by the scope). At zero object photons, each exposure has a different sky bias; any object photons are added to that bias. Thus the sky background is the intercept.

Of course, the actual intensities of various pixels will not be exactly the values above because of noise, one of the reasons for combining exposures.

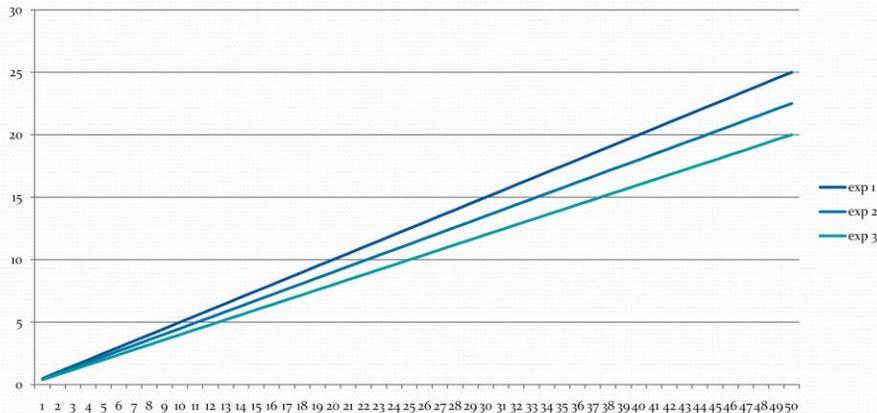


Now it is easy to see what happens when these images are combined via median. Virtually every pixel in the combined image comes from exposure 2 because that's the mid-value for each pixel stack. A very few far outlier pixels in exp 2 will be substituted from exp 1 or exp 3 but it is rare and the substitutions are defective (too low or too high). Thus the median combine is nothing more than a single sub-exp with a very few poorly "fixed" pixels.

# Offset Normalization

Subtract the sky offset from each image. Now each image has intercept = 0.

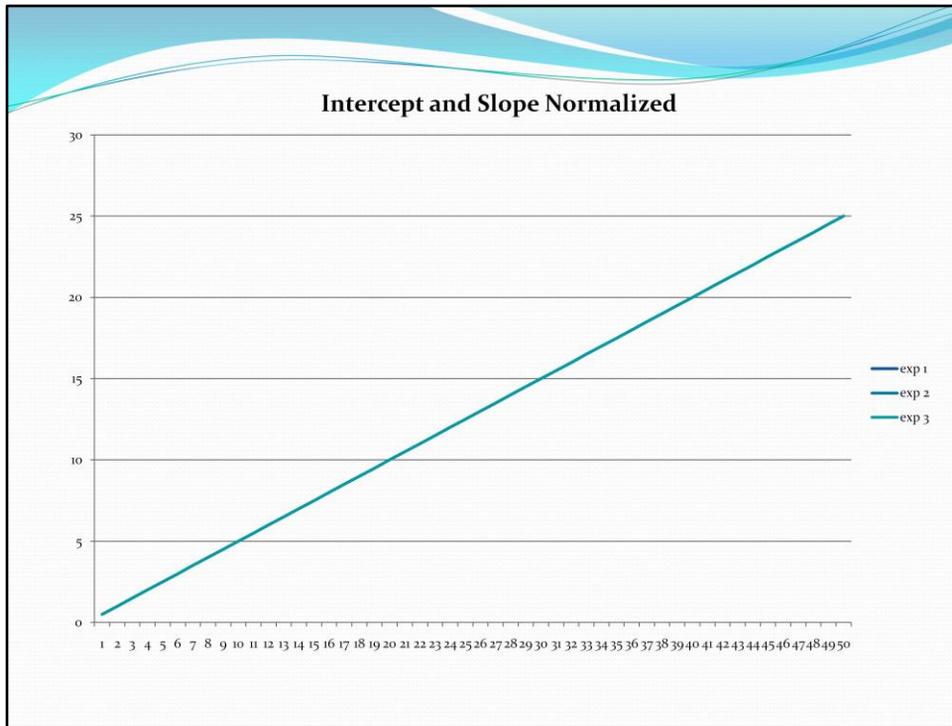
Intercept Normalized



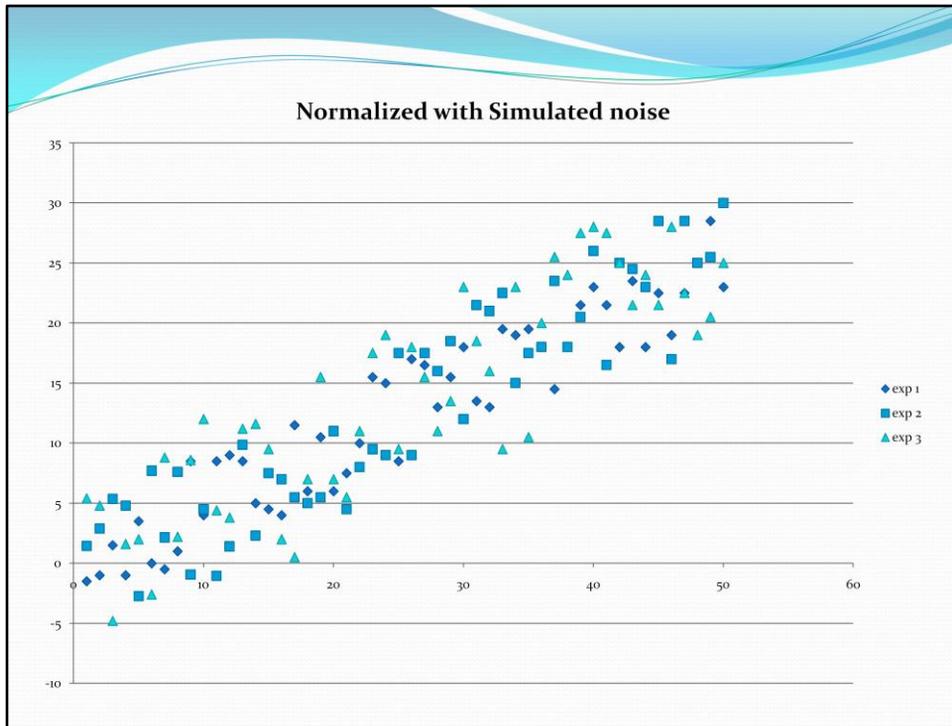
## Offset Normalization

To effectively combine those exposures, it is necessary to equalize the sky background intensities. The correct method is to measure the sky intercept for each exposure to calculate an offset to be subtracted from each image. In the example, the offsets are: exp 1 = 10, exp 2 = 30, exp 3 = 60. This results in a new intercept of zero for each exposure. Here is the graph after subtracting those offsets.

Now the exposures are much closer to each other but they are still not identical. This is because while the moon was rising the target was getting lower in the sky and thus each successive exposure was increasingly dimmed by changing sky transparency. Note that the effect of varying transparency is “slope”, which can be normalized by factoring each exposure:



Here are the theoretical intensities after intercept and slop normalizations.



Now the median will be drawn from the best pixels and each image will contribute substantial data to the result. Note that normalization does not decrease or increase the relative noise of each image. Thus noisier frames will make fewer contributions to the median

(exp 3 in the example is noisier due to the bright moonlight).

# CCDStack Normalization Procedures

## Auto

Performs offset and scalar normalizations based on percentiles  
Option to select a sub-area

## Control

Scalar: adjust slope only (assumes zero intercept)

Offset: adjusts intercept only

Control – Both: adjust slope and intercept for full linear normalization

CCDStack provides several ways to normalize images. It is possible to measure the images and directly calculate offsets and factors that can be applied via “Pixel Math”. Easier methods are implemented in the “Stack”; “Normalize” menu. The top image is the reference image to which all included images will be normalized; thus the top image does not change.

\* Auto: This method calculates percentiles for each image and uses an iterative algorithm to derive slope and intercepts that produce similar percentiles. There is an option to select an area that will be used to calculate the percentiles; this option is useful to avoid distortions from blown-out or other problem areas.

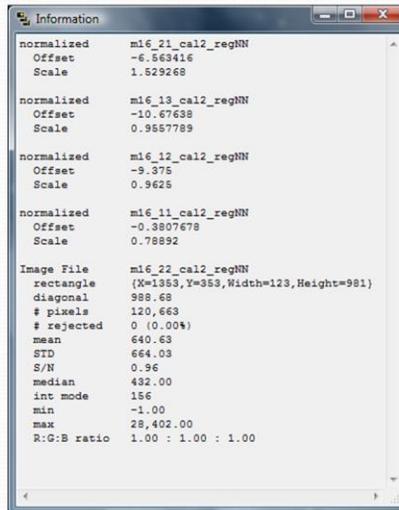
The “Control” methods allow the user to select regions in the image that will be used to directly calculate the normalization.

Control Scalar: This method calculates only factors (slope) to normalize the images. An area is chosen by the user for reference intensity. Factors to equalize that area are calculated and applied to all included images. This is intended for situations where the intercept is naturally zero. For example, a dark subtracted flat has a zero intercept and it is appropriate and accurate to slope-normalize zero-bias flats.

\* Control – Offset: This method calculates only offsets (intercepts) to normalize the images. An area is chosen by the user for reference intensity. Offsets to equalize that area are calculated and applied to all included images. This is intended for situations where the intercept varies and the slope remains constant. However, such situations are rare in the “real world” of astronomy and this option is mostly included for mathematical completeness.

\* Control – Both: This method allows the user to select areas that are used to calculate both the intercept and slope. The user selects a background area that is used to calculate offsets (intercepts) and a bright highlight area that is used to calculate scalar factors (slope); an example would be the non saturated core or inner arms of a galaxy.

# CCDStack Normalization Procedures



```
Information
normalized    m16_21_cal2_regNN
Offset       -6.563416
Scale        1.529268

normalized    m16_13_cal2_regNN
Offset       -10.67638
Scale        0.9557789

normalized    m16_12_cal2_regNN
Offset       -9.375
Scale        0.9625

normalized    m16_11_cal2_regNN
Offset       -0.3807678
Scale        0.78892

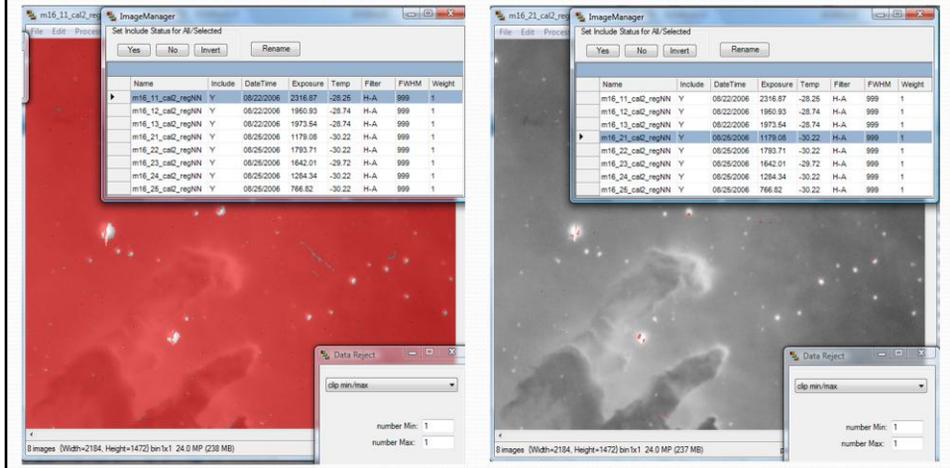
Image File    m16_22_cal2_regNN
rectangle    (X=1353,Y=353,Width=123,Height=981)
diagonal     988.68
# pixels     120,863
# rejected   0 (0.00%)
mean         640.63
STD          664.03
S/N          0.96
median       432.00
int mode     156
min          -1.00
max          28,402.00
R:G:B ratio  1.00 : 1.00 : 1.00
```

The Information window will report the factors and offsets applied to each image. Additionally, the factor will be used to calculate new weights. Statistical weights are used by CCDStack combine methods to optimize the signal-to-noise of the combined image; for example, a frame with exp time only half as long as the other frames would be normalized via a factor near 2x and weighted near 0.5, then when a Mean or Sum image is created that frame would only contribute  $\frac{1}{2}$  as much information as other frames.

**Warning!** All normalization procedures assume that the images are aligned. Do not forget to register the stack before normalization.

# CCDStack Normalization Procedures

## Symptoms of Faulty Normalization



## Symptoms of Faulty Normalization

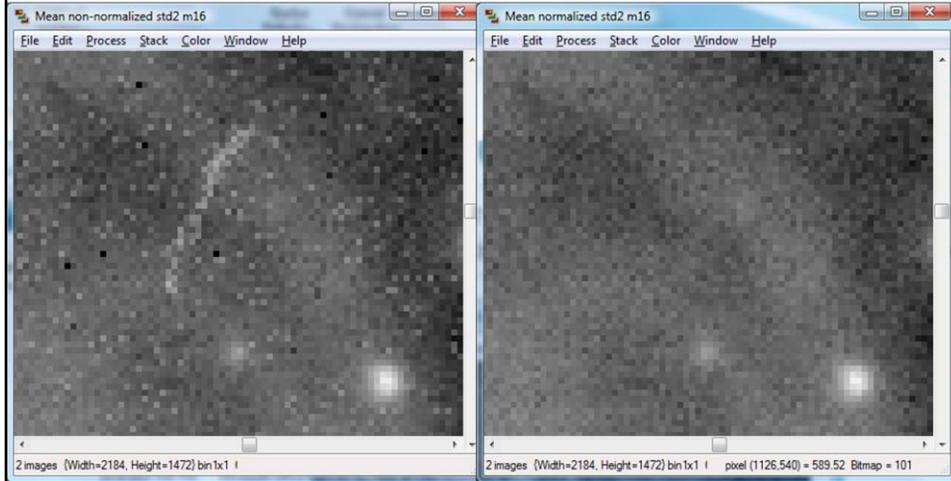
Stack data reject procedures should generally result in rejection percentages and spatial distributions that are similar for all images. For example, if a sigma-reject specification results in 2% rejection in the top image then each of the other images should also have approximately 2% rejection rates. If there are images in the stack that are over or under rejected then it might be due to incorrect normalization.

There are valid reasons that properly normalized images would have dissimilar rejection rates. Valid discrepancies can be caused by significantly different signal-to-noise variations in the stack. Example include: stacks composed of different exposure times, or from different scopes, or made under very different conditions.

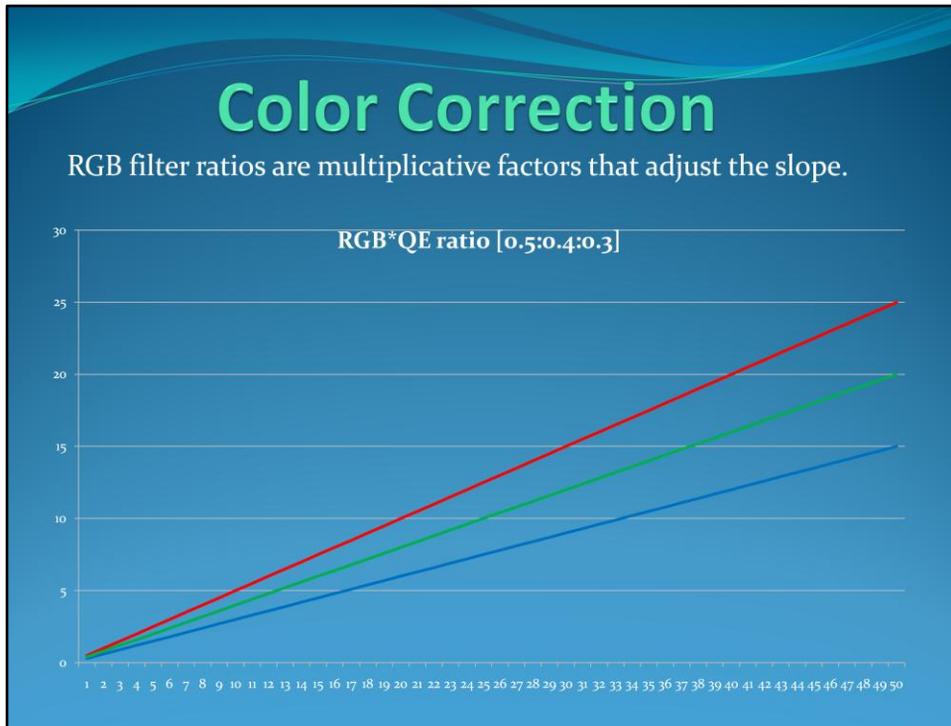
Data reject shows incorrect normalization (not normalized)

# CCDStack Normalization Procedures

Example of final image quality difference due to normalization



Example of final image quality difference due to normalization.  
Close-up of stacks of 8 exposures of varying length; STD data reject factor = 2:



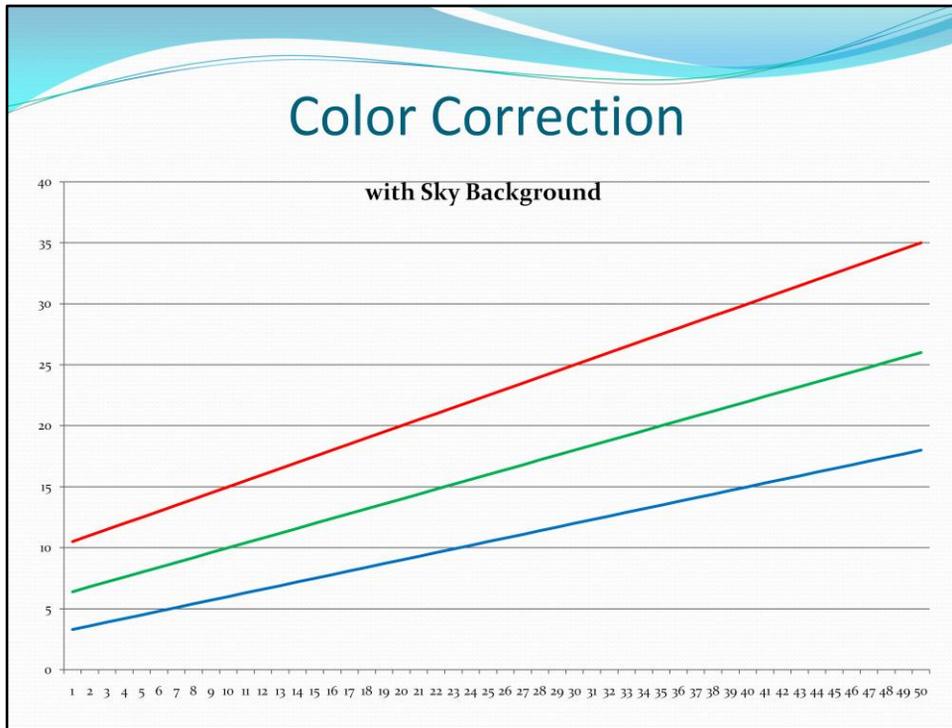
Slope and intercept applies to astronomical color correction.

RGB filter ratios are multiplicative factors that adjust the slope.

The reason that each line has a different slope is due to transmission factor of the filter for its pass-band as well as the quantum efficiency of the CCD in that same pass-band.

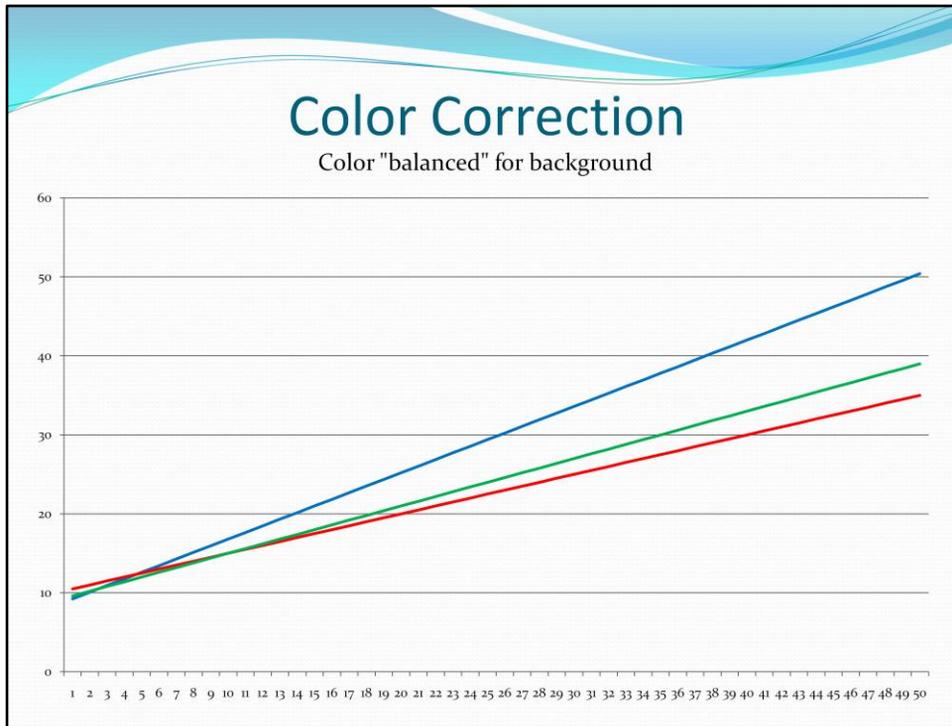
Note the zero convergence at low intensities. That is a zero intercept. Terrestrial imaging has a natural intercept of zero.

But astronomical imaging has an important difference – the sky background... (next slide)



Sky background is the zero-point bias for all astronomical objects.  
It is an intercept (or offset or bias).

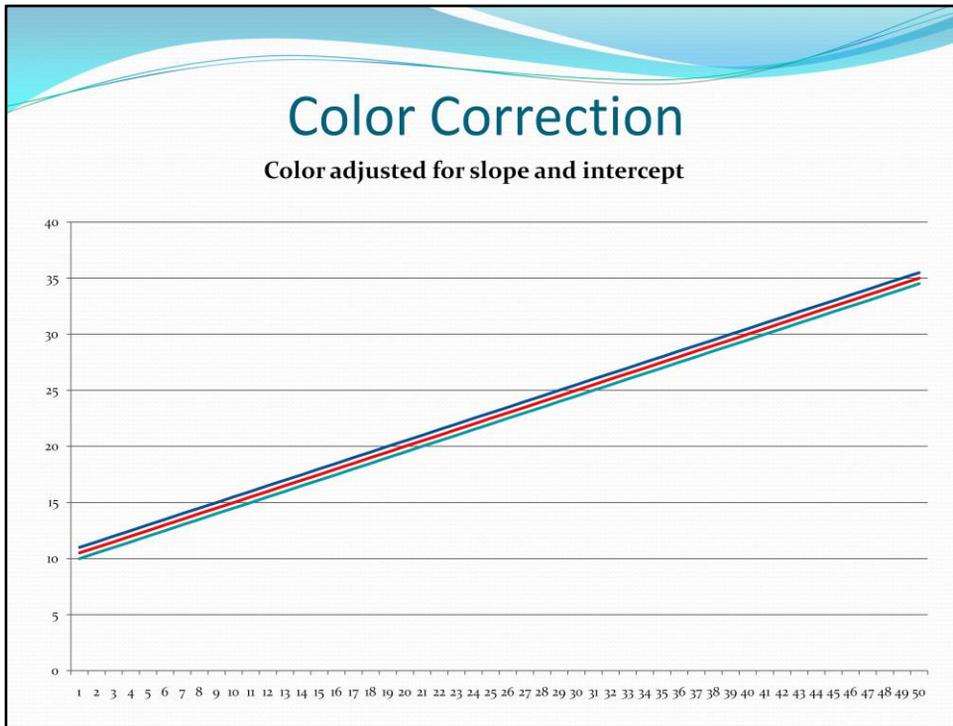
Note that each filter has a different sky background because the sky itself is colored.



A common mistake is attempting to correct the sky background color by adjusting color ratios (slope).

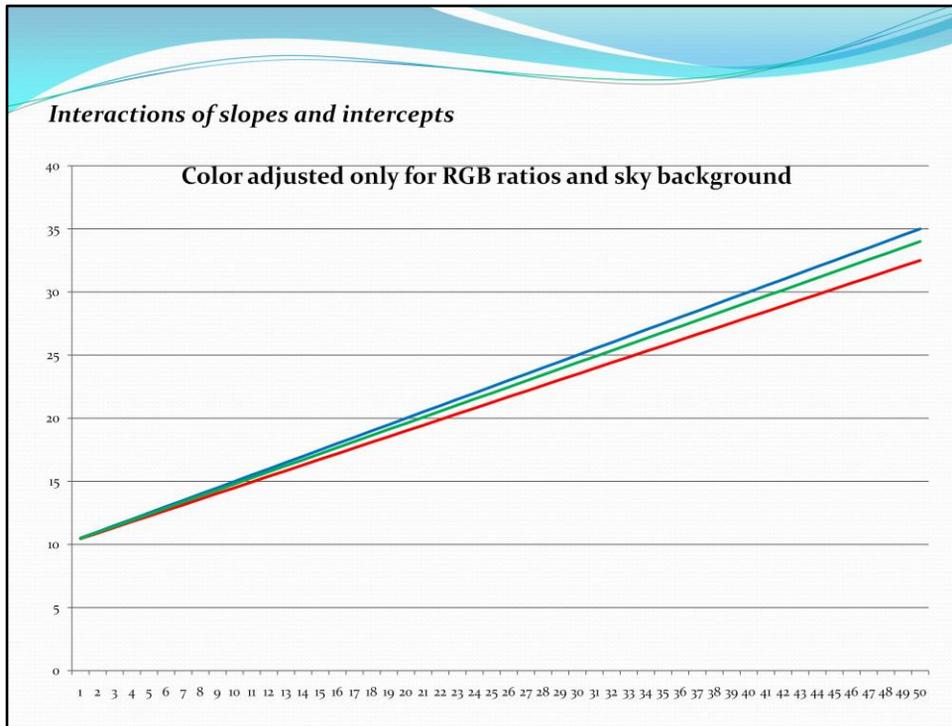
Adjusting ratios to balance the background produces distorted hues of bright objects.

I've often observed someone state that they corrected the color but then ask "why is my galaxy so blue?"



The correct way to handle sky color is to subtract an offset, or in other words, change the intercept.

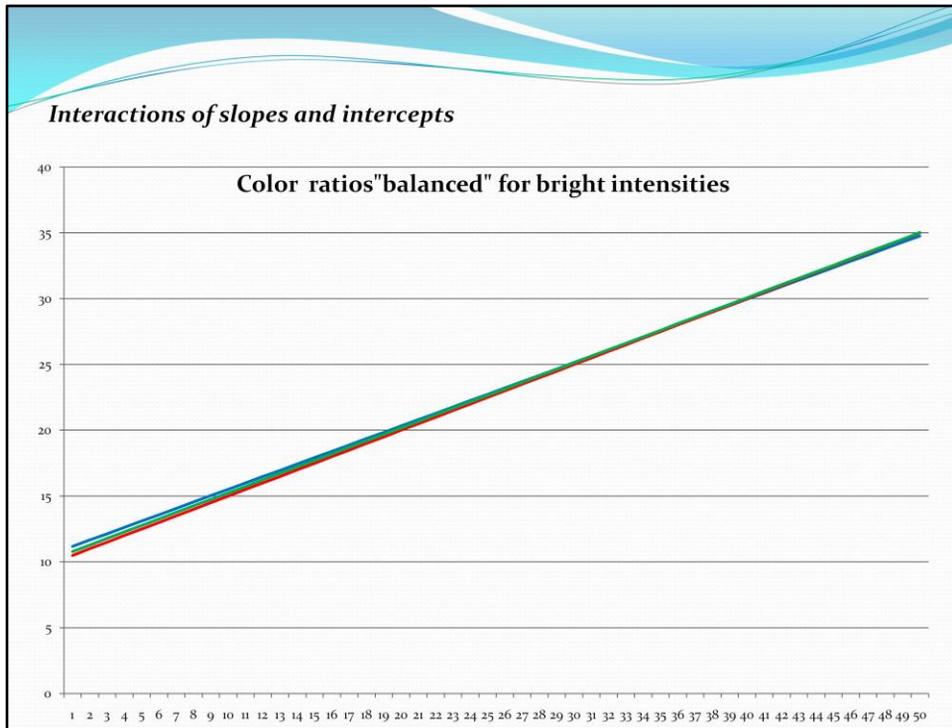
Actually all three lines should overlay but they are separated here just to illustrate that all 3 colors are balanced.



***Interactions of slopes and intercepts***

The additive nature of intercept and the multiplicative nature of slope can combine to produce unexpected or confusing results. This is especially true in color processing. The problem occurs when applying a factor to offset data vs. applying an offset to factored data. The two methods yield different results.

Suppose we have applied standard RGB ratios and correctly removed sky color via offsets. But the color balance is not quite right because the object was imaged thru excessive atmosphere or interstellar dust.



Attempt to correct the imbalance by only adjusting ratios to whiten bright objects

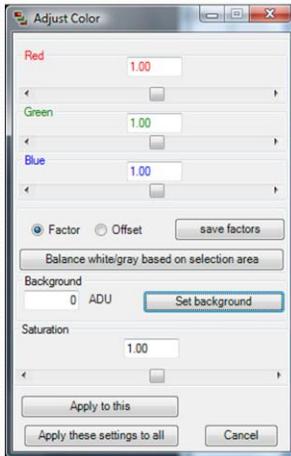
The attempt to balance a bright object inadvertently colorized the background. It may be necessary to iterate this process several times to produce a correct balance across all intensities.

***CCDStack to the Rescue!***

To prevent that situation, CCDStack recalls the background offset (from "set background") and automatically calculates color ratio adjustments using that offset as an intercept term. This method avoids color shifts at different intensities and obviates the need for iterative color balancing.

# CCDStack Color Adjustments

*Integral and explicit slope and intercept system of color adjustments*



The Red, Green, blue sliders and text boxes are contextual depending on whether “Factor” (slope) or “Offset” (intercept) is checked.

Ability to directly enter numbers in the textboxes.

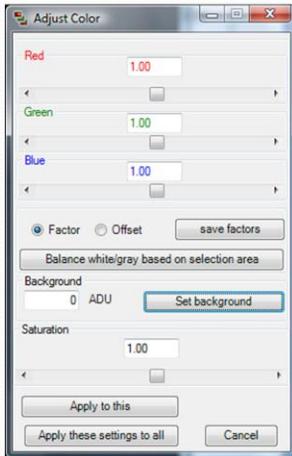
Double-click the blank area to the left or right of a textbox to normalize the ratios so the selected color has ratio = 1.0.

Save factors will remember the factors as RGB ratios for future application.

Set Offsets = 0 (in Offset mode) resets the offsets.

# CCDStack Color Adjustments

## *Automatic white/gray balancing*



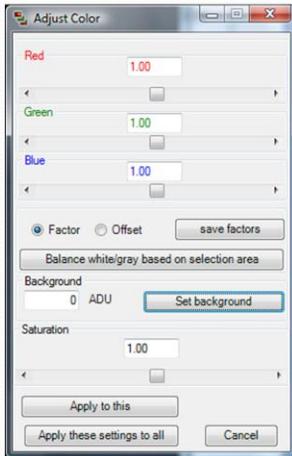
Balance white/gray based on selection area:

in “Factor” mode, this calculates RGB ratios so that the selected area has an overall color balance = white/gray (use on G2v star or spiral galaxy or star cluster);

in “Offset” mode it calculates color offsets so that the selected area has an overall color balance = white/gray (similar to “set background” except that it does not remember the intercept and is not directly applied).

# CCDStack Color Adjustments

## *Set Background*



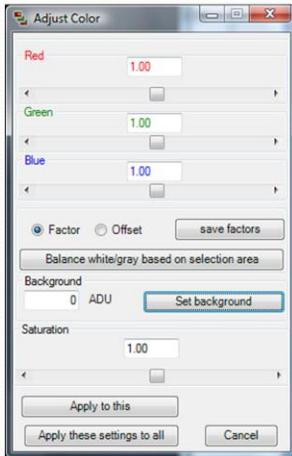
Background is the mean intensity of the selected background from the “Set Background” process.

Set background:

Calculates the mean ADU for each color within the selected area and applies offsets to each color to equalize all intercepts to the luminance value. Additionally, the equalized intercept is remembered to use when subsequently changing color ratios.

Option to de-saturate the background removes color information for pixels with ADU near or below the selected background intensity.

# CCDStack Color Adjustments



Saturation enhances the color differentials. For example, if an object (set of pixels) has red signal that is 20% brighter than green then increasing the saturation by 2 results in a red that is 40% brighter than the green.

Apply buttons change the underlying data.

Prior to "apply" the settings are temporary.

## Flat Field Slope and Intercept

Flat fielding non-zero bias frames results in under or over correction due to the non-zero intercept.

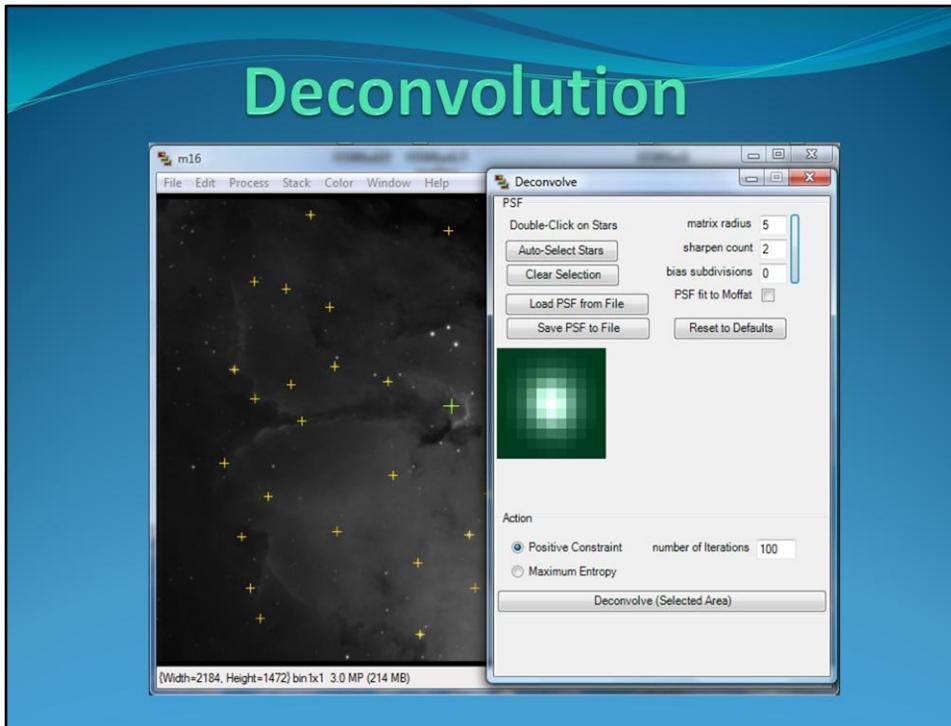
A master flat constructed from incorrectly normalized frames can produce a Slope and intercept dynamic that impairs flat fielding.

Flat fielding non-zero bias frames results in under or over correction due to the non-zero intercept. Correct use of software generally avoids this problem. Both the flat and light image must be dark subtracted or otherwise remove the bias.

Different software handles calibration in different ways. CCDStack uses a zero pedestal while others may use a 100 ADU pedestal to avoid negative numbers. It is inadvisable to mix-and-match software within the calibration regime (e.g. if you make a master flat in brand X then try to calibrate in CCDStack you need to know exactly what brand X did to that master, e.g. is there a pedestal? Did it crop the image, as is common for DSLR).

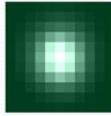
Slope and intercept dynamic that can potentially impair flat fielding – a master flat constructed from incorrectly normalized frames can produce a distorted master. It is advised that master flats be combined via simple mean with no normalization or data rejection. Use CCDStack master flat procedure for automatic normalization and rejection. To make a master flat via normal stacking: remove dark subtract the flats then use “control” scalar normalization before data rejection.

# Deconvolution



CCDStack V2 exposes new deconvolution controls. The core algorithms are similar to CCDStack version 1 but the new controls facilitate more flexible and precise processing by tailoring the Point Spread Function (PSF).

## PSF



The Point Spread Function (PSF) describes how a single point in the object is represented in the image.

Deconvolution uses the PSF to infer the non-convolved object from the convolved image.

The Point Spread Function (PSF) describes how a single point in the object is represented in the image. Stars are single point objects that are convolved (blurred) by optical diffractions and aberrations as well as seeing and tracking/guiding errors.

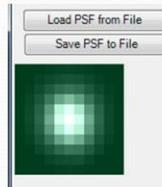
Deconvolution uses the PSF to infer the non-convolved object from the convolved image.

It is important to derive an accurate and appropriate PSF.

## PSF - Star Selection

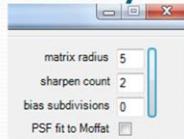
- Select particular star(s) and/or use the Auto-Star Selection procedure
- Double-click to select a star
- Double-click a selected star to de-select
- PSF from multiple selections is the mean; brighter stars have more impact
- avoid non-linear stars (near saturation)
- avoid noisy stars (too dim)
- avoid multiple stars and crowded star fields
- avoid stars within objects (the PSF wings may be biased by object gradients)
- handle optical field aberrations such as coma:
  - select from the center
  - select from all sectors
  - Moffat fit
  - separately deconvolve multiple tiles

## PSF - Save and Reuse



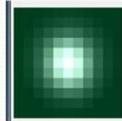
- PSF is saved and loaded as a small FITS
- for nearly constant PSF, an optimal PSF can be saved and reused indefinitely
- saved PSF can be used on a similar-PSF image with no stars or no suitable stars
- saved PSF can be custom modified in any software
- image can be divided into tiles with different PSF for each tile

# Modify PSF

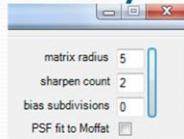


## *matrix radius*

- matrix radius set the size of the pixel matrix that encloses the PSF
- matrix width and height =  $1 + 2 * \text{radius}$ ; e.g. radius = 3 creates 7x7 matrix
- matrix size should be large enough to encompass the desired PSF
- matrix size should not be any larger than necessary
- processing speed depends on matrix size
- suggest matrix radius =  $\text{FWHM} + 1$  to  $\text{FWHM} * 2$
- to trim PSF wings - use smaller matrix and “bias subdivisions”
- matrix is displayed



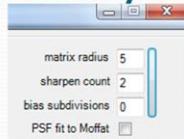
# Modify PSF



## ***bias subdivisions***

- bias subdivisions suppress PSF wings
- inaccurate PSF wings are responsible for “panda eyes”
- bias subdivisions = number of divisions between the raw bias level and the maximum
- values less than the lowest subdivision are set to zero:  
$$\text{Clipped bias} = \text{raw\_bias} + ((\text{max\_value} - \text{bias}) / \text{bias\_subdivisions})$$
- fewer subdivisions result in more aggressive clipping
- more subdivisions result in less aggressive clipping
- clipping is performed prior to Moffat fitting (if any)
- Moffat fit will re-value the clipped wings
- if not Moffat fit, clipped wings = zero and so radius may be unnecessarily large
- set to zero to disable

## Modify PSF



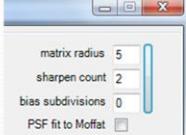
### *sharpen count*

- effect of sharpening is “sample limited” deconvolution
- PSF sharpening is a way to minimize some artifacts
- PSF sharpening does not sharpen the image
- sharpen transfers PSF flux inward via simple flux redistribution
- count = number of PSF sharpen iterations applied
- set to zero to disable (turn off “sample limited” deconvolution)

### *sharpen count*

- effect of sharpening is “sample limited” deconvolution
- PSF sharpening is a way to minimize some artifacts
- PSF sharpening does not sharpen the image
- sharpen transfers PSF flux inward via simple flux redistribution
- count = number of PSF sharpen iterations applied
- set to zero to disable (turn off “sample limited” deconvolution)

## Modify PSF



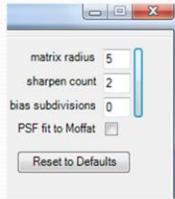
***PSF fit to Moffat***

- fit empirical PSF to a Moffat function
- Moffat functions model star images and are more accurate than Gaussian
- fitted PSF will be symmetrical
- useful for differentially aberrated image
- fitted PSF can improve a problematic PSF
- use empirical PSF (not fitted) to correct for systematic asymmetrical defects such as small tracking errors

### ***PSF fit to Moffat***

- fit empirical PSF to a Moffat function
- Moffat functions model star images and are more accurate than Gaussian
- fitted PSF will be symmetrical
- useful for differentially aberrated image
- fitted PSF can improve a problematic PSF
- use empirical PSF (not fitted) to correct for systematic asymmetrical defects such as small tracking errors

## Modify PSF



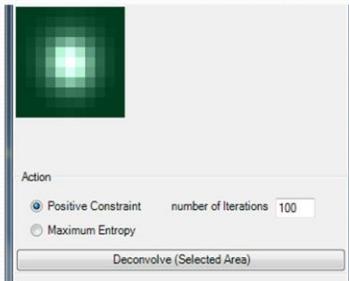
### ***Reset to Defaults***

resets parameters to approximate CCDStack Version 1 deconvolution

### ***Reset to Defaults***

resets parameters to approximate CCDStack Version 1 deconvolution

# Deconvolve



## ***Positive Constraint***

- variant of Lucy-Richardson
- usually produces the least artifacts
- somewhat less sensitive to the settings

## ***Maximum Entropy***

- temperamental and unpredictable
- fine results from some images but
- terribly wrong for other images
- often very sensitive to settings

## ***Number of iterations***

- Positive Constraint usually be between 20-200 iterations
- Maximum Entropy usually be between 50-500 iterations

## ***Maximum ADU (Maximum Entropy only)***

- maximum ADU near the upper limit of interesting features
- features above maximum ADU will blow-out
- too high maximum ADU results in sub-optimal sharpening and takes longer

## ***Positive Constraint***

- variant of Lucy-Richardson
- usually produces the least artifacts
- somewhat less sensitive to the settings

## ***Maximum Entropy***

- temperamental and unpredictable
- fine results from some images but
- terribly wrong for other images
- often very sensitive to settings

## ***Number of iterations***

- Positive Constraint usually be between 20-200 iterations
- Maximum Entropy usually be between 50-500 iterations

## ***Maximum ADU (Maximum Entropy only)***

- maximum ADU near the upper limit of interesting features
- features above maximum ADU will blow-out
- too high maximum ADU results in sub-optimal sharpening and takes longer

# Deconvolve



**Selected Area (option)**

Before pressing the Deconvolve button, drag-out a rectangle to restrict deconvolution to the selected area:

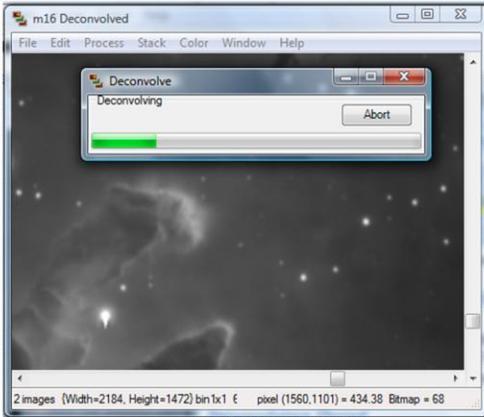
- smaller area runs faster than larger area
- small areas can be used to quickly test parameters
- use selected area to limit the deconvolution to an object within the image
- use selected areas to divide the image into panes for separate deconvolution

## Selected Area (option)

Before pressing the Deconvolve button, drag-out a rectangle to restrict deconvolution to the selected area:

- smaller area runs faster than larger area
- small areas can be used to quickly test parameters
- use selected area to limit the deconvolution to an object within the image
- use selected areas to divide the image into panes for separate deconvolution

# Deconvolution Thread



Abort interrupts the current iteration then performs a normal finish.

Deconvolution runs its own master thread, separate from the main user interface

Deconvolution sub-routines split into parallel threads to exploit multi-core computers

Image is frequently be updated during deconvolution

During deconvolution, stack images can be viewed rescaled, sharpened, magnified, etc.

Blink the original and deconvolving images to observe changes

Double-click star to see the improving DWHM in Image Manager

# The End... (?)

AIC 2010 CCDStack Workshop

Questions?