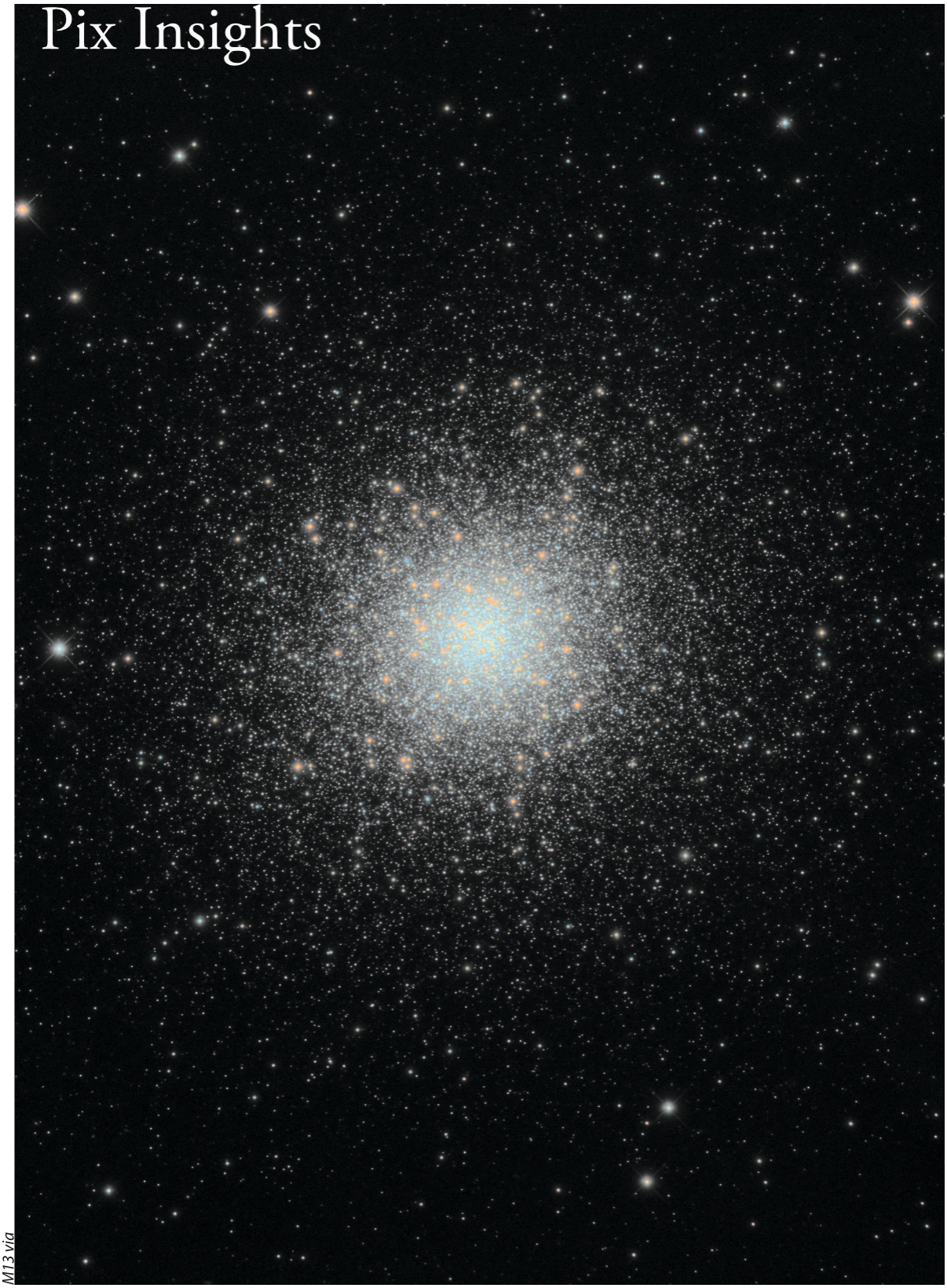
Pix Insights



M13 via

Pre-Processing

This set of tasks is often automated for convenience but a little extra effort often improves image quality.

This chapter and the four that follow are new to edition two. Their inclusion is designed to push the quality envelope of your image processing using PixInsight. These five chapters discuss selected processes in more detail, the first of which looks at the often overlooked and highly automated process of image pre-processing. Pre-processing is a key step that is often taken for granted, on account that powerful tools can automate the process. Pre-processing consists of four main steps:

1 selection
2 calibration
3 registration
4 integration

Careless use of automation may fail to discriminate between good and bad images and choose sub-optimum strategies to minimize image noise, register images and reject bad pixels.

Selection (1)

The saying "you have got to be cruel to be kind" applies to astrophotography. After all the effort of honing your skills, ensuring your equipment is reliable and after patient hours of acquisition and tuning focusing and autoguiding parameters, it is often with some reluctance that we accept that some of our precious subframes are best discarded. This is really tough, especially when the pickings are meager. These images are not necessarily the ones with aircraft or light trails, since these can be removed statistically, but those where the focus and tracking issues spoil star shapes and the image has excessive noise. It is important to select images prior to calibration rather than afterward, since the data from these bad boys influence the statistical analysis of the good frames.

PixInsight has several tools that help identify those images for relegation of which the SubframeSelector and the Blink tool are the most useful. The three main indicators of a poor image are the star size (FWHM), eccentricity and signal to noise ratio. These measures are not unique to PixInsight; CCDInspector and other tools, such as Main Sequence's free FITS Image Grader have simpler versions. In each case the images are loaded into the tool and sorted according to the selected discriminating parameter. As you might expect, PixInsight, which revels in statistical detail, has many adjustable parameters behind the simple default tool settings.

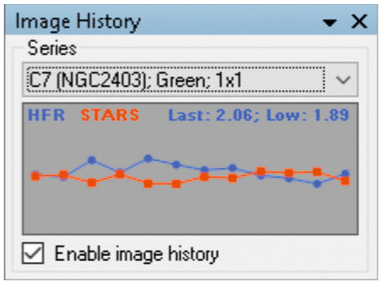


fig.1 Sometimes, a simple monitor during image capture will detect oddballs or worsening trends (e.g. star count and HFR).

Let us initially back up and start at a simpler level, to the business of image acquisition. Problem images, arising from special causes do happen from time to time but more often than not, an indication that something has gone off the boil is apparent during image acquisition. Prevention is always better than cure and it is possible to detect focus drift or tracking issues in real time. This provides an opportunity to pause and fix things, discard obvious duds and add further sub-frames to the sequence to make up for the loss. I use Sequence Generator Pro for image acquisition and one of its panels provides a simple image history graph, with HFR and star-count metrics for each exposure (fig.1). An adverse trend or special issue are easily spotted and uniquely, in SGP, a dud image can be marked as "bad" during a sequence and the event count is automatically compensated. If the HFR suddenly reduces after a temperature-triggered autofocus run, it may be a sign that the temperature trigger point is too large. Not everyone uses SGP and it is useful to know that CCDInspector can be set running and watching your image folder. As the images roll in, it measures FWHM, eccentricity, background and other parameters and can equally plot them on a graph to identify trends or outliers.

Blink Tool

The PixInsight blink tool has a number of uses, including the ability to generate a movie file from separate images. In this instance, it is used to initially visual compare and assess images. In practice, load all the files for a target

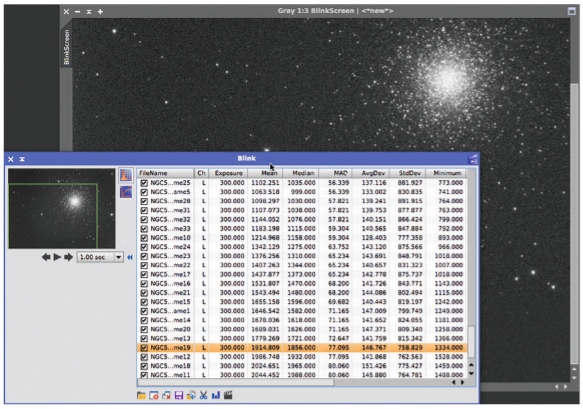


fig.2 The blink tool in operation on M3. The frames are displayed in stretched form onto the main screen and quickly updated from one image to another, making differences obvious.

image and hit the histogram button. This applies an automatic histogram transformation (similar to the screen stretch function) to all images. The subsequent comparison is then much easier since the images from all the filter sets have the same general appearance on screen. (This does not change the image data, only its appearance on screen.) I then set the time-scale to 1 second and "play" the images to the screen. Although simple in concept, the quick overlay of successive images is surprisingly effective at weeding out a few strays. The blink tool also provides a data comparison for each image that has some diagnostic use, for instance, correlating particular imaging conditions with poor performance.

SubframeSelector Tool

This tool is considerably more sophisticated and discriminating. It calculates useful image data to sort and identify rogue images. This is a script that uses several PixInsight tools in a convenient way. After loading the images, open the system parameters and enter the camera gain and angular resolution. The tool analyzes stars and uses further parameters to optimize star-detection and exclusion. The pop-up dialogs show how the various sliders can be used to exclude highly distorted stars, hot pixels, saturated stars and characterize the point spreading function (PSF) for star detection and measurement. The aim is to detect about 200–1,000 stars, avoiding saturated ones and hot pixels. If in doubt, just add a few images and experiment with some settings before loading the entire image folder. Finally, click measure and put the kettle on.

The result is a considerable amount of quantitative data and the next step is to make some sense of it. The SubframeSelector tool produces a table of results with all

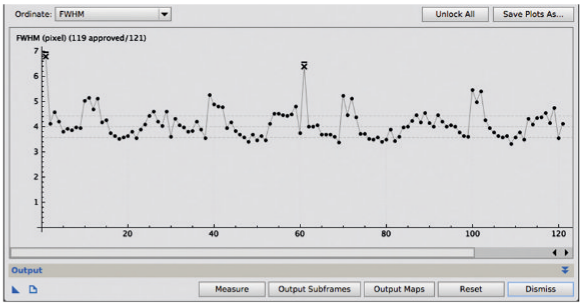


fig.3 The output of the SubframeSelector Tool is a table and a set of useful graphs. Here is the one for FWHM, showing a few outliers.

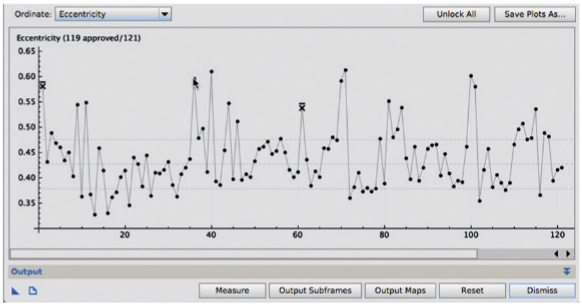


fig.4 As fig.3 but this time measuring star eccentricity or elongation, normally as a result of tracking issues caused by autoguider issues.

kinds of information. This generally consists of data in two forms: its absolute value and its deviation from its mean value. The same data is also presented in graphical form, one for each parameter. Principally, I use star size (FWHM), star shape (eccentricity) and a general noise weighting factor. At a simple level, outlier points in the plots section (fig.3 and fig.4) are clicked on to exclude them (shown with an x). In this case, those with high eccentricity and FWHM. The output section of the tool has the ability to copy or move approved and rejected image files to a folder and add an appropriate suffix to identify its status.

If you prefer a single balanced assessment of a number of criteria, one can enter in an expression, which combines attributes and weighting factors, to provide an overall goodness indicator. In the "goodness" expression below, it is set to a combination of three performance parameters, whose contribution is weighted and scaled within a range:

(25*(1- (FWHM-1)/(5-1)) + 15*(1- (Eccentricity - 0.2) /(0.5 - 0.2)) + 10*(1- (Noise - 25)/(70-25))) + 50

where:

FWHM	*1–5*	*- weighted 50%*
Eccentricity	*0.2–0.5*	*- weighted 30%*
Noise	*25–70*	*- weighted 20%*

In the example, I rejected about 10% of the sub-frames, mostly due to guiding and focus issues, and noted the image with the best parameters as a reference for later on.

Calibration (2)

The science of calibration has its own extensive chapter, explaining how and why we calibrate images and including best practices and techniques. All the image processing tools have semi-automatic methods to do this, removing the boring bits. This includes PixInsight too, in which its standard tools have been incorporated into a script to generate master bias, flat and dark files and calibrate light frames. I often use the BatchPreprocessing script for convenience to generate the master calibration files, calibrate and register subframes, but leave out the final integration. Here, we lift the lid off these processes to see if there is some further quality improvement we can eke out using the individual standard tools.

Manual calibration processing is formed by two main steps: generating master calibration files and applying them to the individual light subframes, using the ImageIntegration and ImageCalibration tools in turn. (Another, the Superbias tool, is a specialized filter that improves the master bias file, especially with small data sets, by extracting column and/or row artefacts in the presence of random noise.) With refractors, and providing one is scrupulous about cleanliness, the dull task of generating master calibration files is an infrequent event. One might not be so lucky with open-design reflectors though, as both mirrors attract dust in addition to the camera filter. In this case master flat files may be required for each imaging session.

Master Bias and Darks

The master bias and dark files are generated using the ImageIntegration tool. This tool has many uses and it has a range of options that selectively apply to the task at hand. In this case it is important to disable some of the features that we are accustomed to using on image subframes. The key parameters are shown in (fig.5).

- Do not normalize the images, either in the ImageIntegration or Pixel Rejection (1) settings.
- Disable the image weighting feature, as we want to average a large number of frames and reject obvious outliers.
- Choose Winsorized Sigma Clipping option to reject outliers, with a 3–4 Sigma clipping point or, if you know your sensor characteristics, try the CCD Noise Model option.

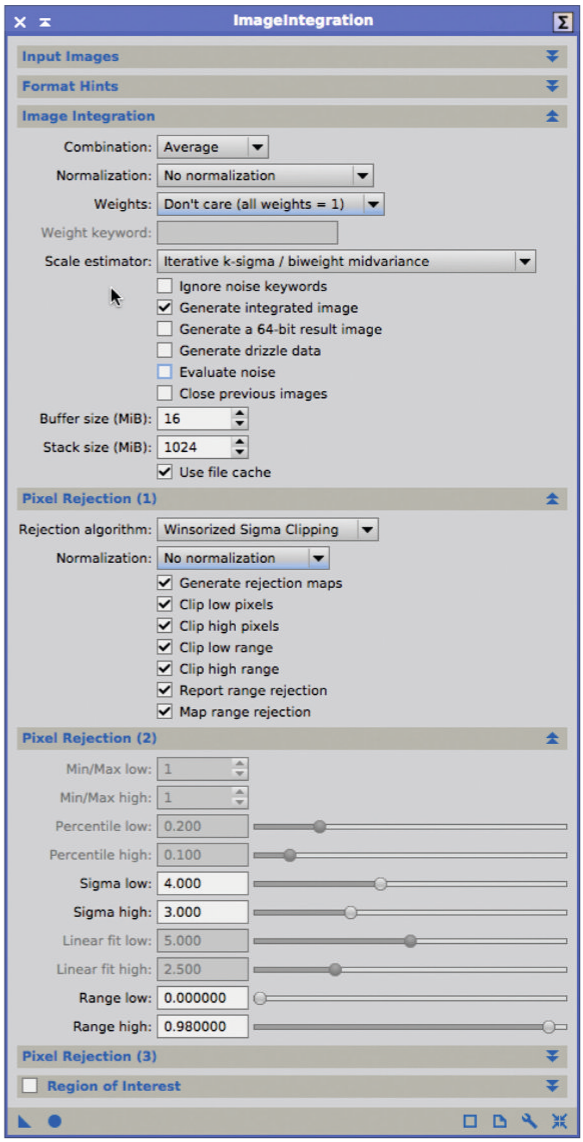


fig.5 The typical settings for creating a master bias or dark file. Note, there is no normalization or weighting.

In each case, select the files according to the filter, binning and exposure setting and generate an overall average master file for each combination. These files should be good for a while, though most CCD cameras develop additional hot pixels over time. It takes three days to acquire all my dark frames, but they generally last a year.

Superbias

This unique PixInsight tool improves image calibration beyond a straight integration of bias frames. A bias frame has very faint pattern noise that is obscured by read noise

and it requires a large number of integrated frames to reveal it. The Superbias tool uses multiscale processing to extract this pattern noise from a master bias file that has been made with a modest number of images. In practice, the default settings work well with 20 bias frames or less (fig.6). Fig.7 and fig.8 show the dramatic reduction in noise in a master bias file. With every doubling of the count, it may be possible to lower the layer count by 1. Noisier CCD sensors and CMOS sensors require more bias frames for the same quality outcome. CMOS sensors present some unique issues; the Superbias tool is optimized for column or row-oriented sensors like a CCD camera and most CMOS cameras have both row and column patterns causing a combination of orthogonal variations. It is still possible to establish a good Superbias for a CMOS sensor, but it requires a few more steps:

- run Superbias on the master bias image using the Column mode
- using PixelMath, subtract the superbias from the master bias and add an offset of 0.1
- run Superbias on this new image using Row mode
- using PixelMath, add the first and second superbias images and subtract the offset of 0.1

The resulting Superbias should have both column and row patterns. The comparison in fig.9 and fig.10 show a master bias from an EOS 60Da and after the application of Superbias. If you look carefully, there are faint horizontal bands in the Superbias image.

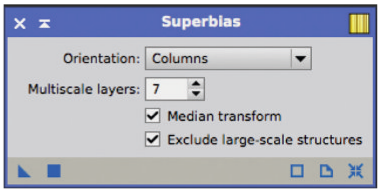


fig.6 The default settings for the Superbias tool, to improve master bias generation.

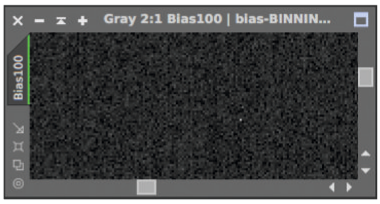


fig.7 A master bias of 100 integrated frames from a KAF8300 CCD sensor.

fig.8 The master bias in fig.7, after application of the the Superbias tool.

Master Flats

The master flat frames are treated differently, to account for the fact that each flat subframe has bias and dark noise in it. This is a two-step process; first, we calibrate each subframe (fig.11) and then we integrate them (fig.12). Open up the ImageCalibration tool and select the master bias and dark frame. The master dark frame will have a different exposure (typically longer) to the flat's and check the optimize option to ensure it is appropriately scaled to optimize the signal to noise ratio. After the tool has done its work on all the frames, we move on to combining them. In this case the images are normalized multiplicatively (especially with sky-flats) and with equal weighting. Pixel rejection has a few alternative approaches, depending on how one took the flat frames. Fig.12 shows a typical setting for a limited number of sky-flats. In this case the recommended rejection algorithm is Percentile Clipping, with very low clipping points (under 0.02). Some experimentation may be required to determine the optimum value to reject outliers. I use an electroluminescent panel and take 10–50 flat subframes for each filter position, allowing me to use Winsorized Sigma Clipping for the rejection algorithm. Pixel rejection benefits from normalized frames. For flat frames choose the Equalize fluxes option for the normalization method.

After applying the tool, the outcome is a series of master flat files on your desktop. As these master files are created outside of the BatchPreprocessing script, give each file a meaningful title and then use the FITSHeader tool to add the filter name and binning level to each master file. This is useful for later on, for instance, the BatchPreProcessing script uses the information in the FITS headers to match up the filter and binning settings with light frames during the image calibration process.

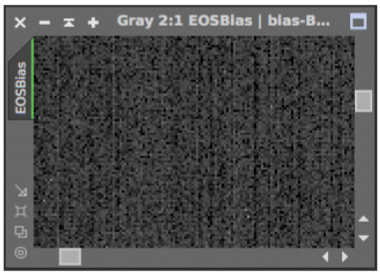


fig.9 The master bias from 100 EOS frames.

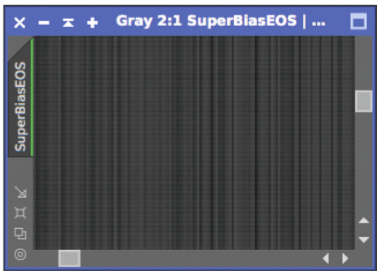


fig.10 Superbias, run in separate column and row modes on the data in fig.9.

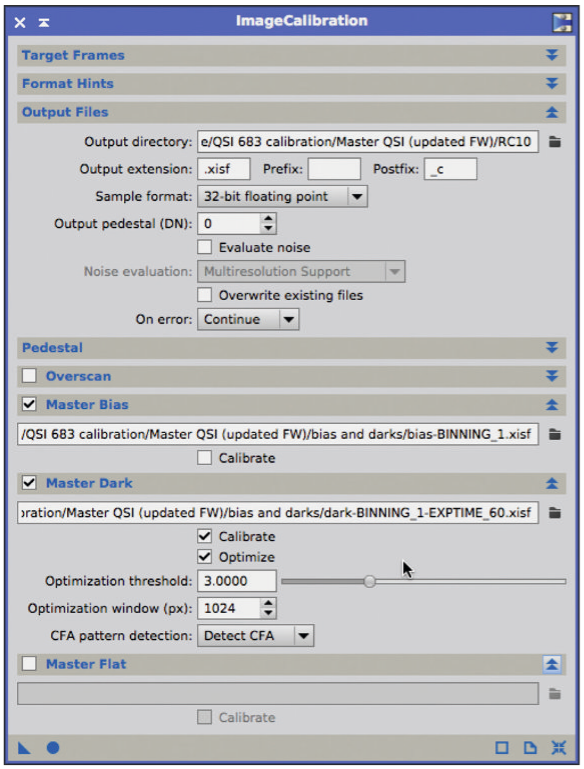


fig.11 The typical settings for creating calibrated flat files, using the previously generated master bias and dark files.

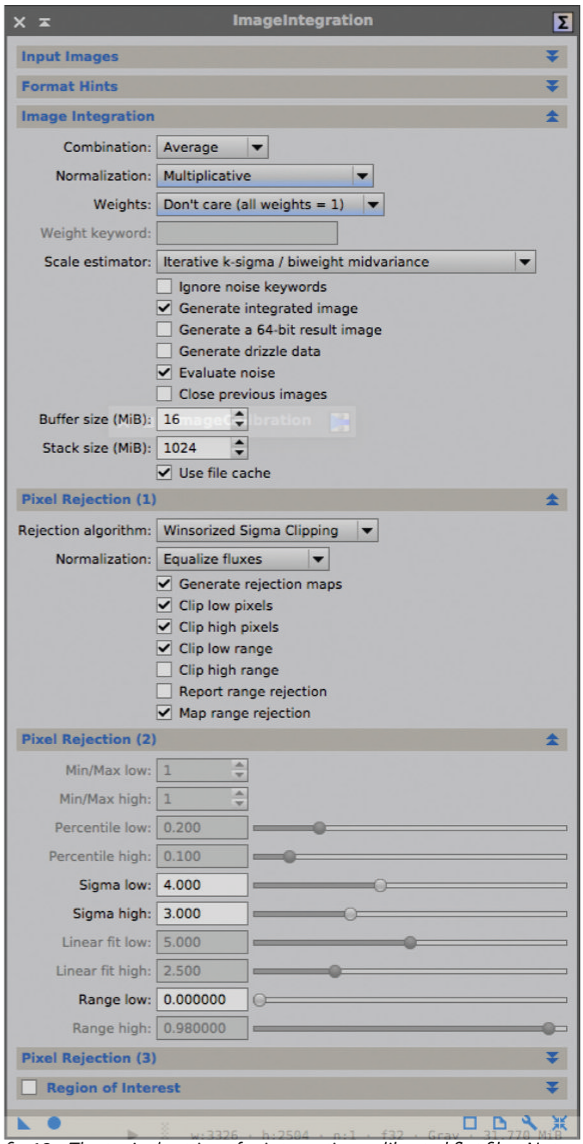


fig.12 The typical settings for integrating calibrated flat files. Note the weighting and normalization settings are different to calibrating lights in both cases.

Calibrating Lights

Lights are our precious image subframes. Before registering these they require individual calibration. The ImageCalibration tool is swung into action once more, but with some additional settings: We replace the individual flat files in the Target Frame box with our light frames (for a particular filter, binning and exposure time) and identify the matching master bias, dark and flat files in their respective choosers. In the Master Flat section, remember to un-check the calibrate box (since we have already calibrated the flat files) but leave the Optimize option enabled. Repeat for all the various combinations of exposure, filter and binning, using the matching dark, bias and flat masters in each case.

Note: In the ImageCalibration tool the Optimize option scales the master dark frame, before subtraction, to maximize the calibrated image's signal to noise ratio (as before during flat file calibration). In some instances, this may not fully eliminate hot pixels, or if new hot pixels have developed since the dark frames were taken, miss them altogether. This is especially noticeable in the normally darker image exposures after stretching (fig.13). There are a number of strategies to remove these; if you dither between exposures, the hot pixels move around the image and with the right pixel rejection settings, the final integration process removes them. At the same time, these settings may be overly aggressive on normal image pixels, reducing the overall image SNR. A better way is to remove them from the calibrated images before registration (this also reduces the possibility of false matches) using DefectMap or CosmeticCorrection.

Fixing Residual Defect Pixels

The principle is to identify and replace defect pixels with an average of the surrounding pixels before registration and

integration. The DefectMap and CosmeticCorrection tools do similar jobs. They are applied to the individual calibrated lights in both cases. To use the simpler DefectMap, load all your calibrated images into an image container. Load the master dark image and open the Binarize tool. Look at the image at full scale and move the threshold slider to find a setting that just picks up the hot pixels (typically 0.01). Close the preview and apply to the master dark image. Now choose Image/Invert from the PI menu to form a white image with black specks and then select this file in the DefectMap tool dialog. Drag the blue triangle from ImageContainer onto the DefectMap bottom bar to apply the correction to each image in the container.

The second tool, CosmeticCorrection is more powerful and is a combination of DefectMap and statistical corrections. Again, it is applied to calibrated images before registration. Its setup parameters include a simple threshold (to identify hot pixels from a master dark file in a similar manner to DefectMap) in addition to statistically comparing neighboring pixels. Most CCDs develop additional hot pixels over time and in practice, the CosmeticCorrection can fix those pixels that escape the calibration process (fig.13). The benefits are two-fold; not only are there fewer outlier pixels but these, in turn, improve the accuracy of the registration algorithm by reducing false matches. The benefits extend to image

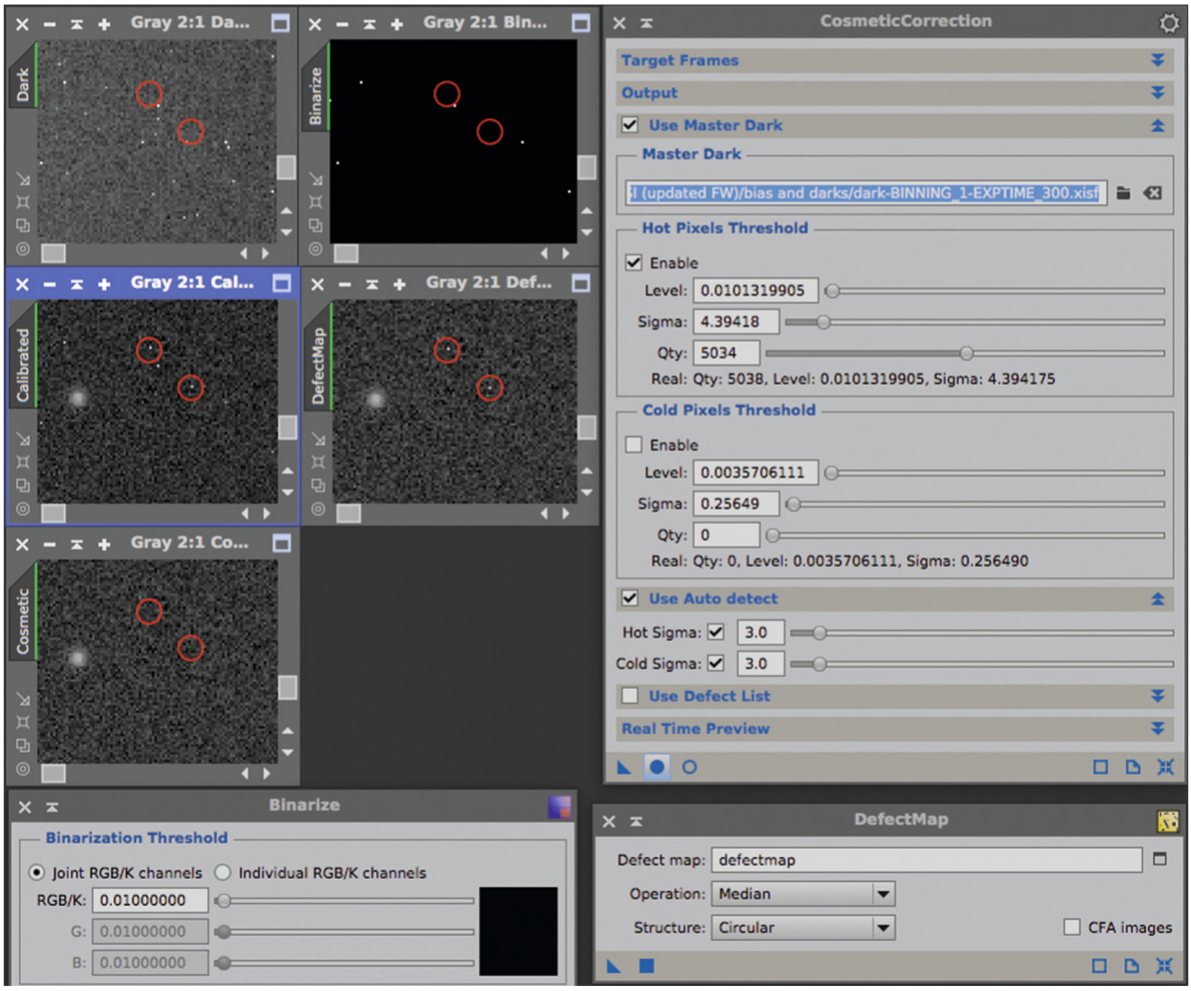


fig.13 This screen shot shows a selection of image pixel correction tools and results using DefectMap and CosmeticCorrection. The array of images are shown at 200%, starting top left with a 2015 master dark and its binarized version alongside. Beneath these are the calibrated light frame (taken in 2016) and after DefectMap has been applied. The bottom image shows the calibrated light frame after CosmeticCorrection has been applied. Note the absence of hot pixels in the old dark file, which are then missed in the DefectMap application. It is the Auto detect method in Cosmetic correction that fixes them.

integration too, since there is less need for extensive pixel rejection and hence the integrated result benefits from a mildly improved signal to noise ratio. In the example in fig.13, the registration errors in the normal integrated image (with hot pixels) are 4x higher than the same files using CosmeticCorrection. The default hot and cold settings of 3 Sigma work well in many cases but as always, a little experimentation on a sample image, using the live preview at different settings, may achieve a better result.

A cosmetic correction option also appears in the BatchPreprocessing script. In practice, use the real-time preview of the CosmeticCorrection tool on an uncalibrated image file and select the matching dark file in the dialog. Zoom in (or use a preview of the image file) to see what hot and cold thresholds are needed to identify the hot and cold pixels. One can additionally use the auto settings too, adjusting the hot and cold sigma sliders, to remove the ones that refuse to disappear. Once you have the right settings, close the preview and drag the blue triangle onto the desktop to produce a new process instance icon and give it a useful name. It is this process icon that is selected in the BatchPreprocessing script to apply CosmeticCorrection to all the light frames.

Registration (3)

Image registration is also integrated into the BatchPreprocessing Script and, using its default parameters is often perfectly adequate for the job. It utilizes the StarAlignment tool with a set of basic settings. As with the other tools, however, one can perform registration independently, using the StarAlignment tool in all its glory. This may be a requirement in tricky situations, for instance when aligning tiled images (with minimal overlap) for a mosaic. It can also be used to align stars to astrometry data, to form a solid reference.

Image registration for deep sky objects uses image features to align subframes; here, we have plenty of them, namely stars. (Image registration for planetary images uses other techniques that match images in the frequency or spatial domains.) Simple star alignment is called a rigid transformation, in so much that it only shifts, rotates and scales an image uniformly to make a match. The more advanced algorithms stretch and distort images to match one another, a handy feature for accurate star-matching towards the edge of an image. The StarAlignment tool has a number of settings and fortunately, the best starting point is its default values. It can automatically register a group of images or a single image. During mosaic panel alignments, it can not only create the separate mosaic components but also merge them and adapt frames to make them seamless. Mosaics pose the biggest challenge and the StarAlignment tool can restrict the star matching to selected areas and use other mathematical techniques (FFT-based intersection estimation) to improve alignments. Just before going into the clever workings behind the scenes, it is important to note that the choice of reference image is critical. If you noted the best image during subframe selection, this is the one that is best suited for use as the reference for the others, typically the one with the smallest FWHM and eccentricity.

Star Detection

The Working mode is set to Register/Match Images by default, used for normal single-pane images. For mosaics, the Register/Union-Mosaic and Register/Union -Separate generate a new combined mosaic image and separate ones on a full canvas. This latter setting allows one to use powerful tools like GradientMergeMosaic that hides the joins between panes. (An example of this is used to combine a 5-tile mosaic of the California Nebula in one of the first light assignment chapters.)

After several dialogs on file input and output, we come to the Star Detection section. Although we can instinctively identify stars in an image, computers need a little help to discriminate between star sizes, hot pixels, nebula, cosmic rays and noise. The default values work well but a few may be worth experimenting with. The Detection scale is normally set to 5; a higher number favors bigger stars and a smaller value will include many smaller stars. If the default value has difficulty finding enough stars, a setting of 4 or 3 may fix the issue. The following parameters refer to noise rejection; the default Noise scale value of 1 removes the first layer (the one with most of the noise) prior to star detection. I have never had the need to change this value and the PixInsight documentation suggests that a value of zero may help with wide-field shots where stars may be one pixel, or larger, in the case of very dim stars. Similarly, the Hot pixel removal setting changes the degree of blurring before structure detection. This uses a median filter, which is particularly effective at removing hot pixels.

Two further settings affect the sensitivity of star detection, Log(sensitivity) and Peak response. A lower Log(sensitivity) setting favors dimmer stars. Again, the default value works well and including more stars with a lower value may be counter-productive. The Peak response parameter is a clever way to avoid using stars with several saturated pixels at their core. This setting is a compromise like the others; too small and it will not detect less pronounced stars, too high and it will be overly sensitive and potentially choose saturated stars.

Star Matching

Now that the star detection settings are confirmed, the next section looks at the matching process parameters. Here, there are some freaky terms that refer to RANSAC or RANdom SAmple Consensus. The tolerance term sets the number of pixels between allegedly matched stars. A larger value will be more tolerant of distortion but too high and one may get some false matches. The Overlapping, Regularity and RMS error parameters should be left at their default values in all but the trickiest mosaics. The Descriptor type changes the number of stars in the matching process. The default Pentagons settings works well on standard images, but on mirrored sets of images, change it to Triangles similarity. Lastly, the Use scale differences and its tolerance parameter constrict the allowable scale range between images, often used to prevent false matches on mosaics.

Interpolation

The last section worth discussing is Interpolation. This is where things become interesting. Why do we need interpolation? We know that autoguider algorithms can detect the centroid of a star to a $1/10^{th}$ pixel. The alignment algorithms are no different in that they can register an image with sub-pixel accuracy, added to which, there may also be an overall scaling factor required during the registration process. If the mode is set to auto, Lanczos-4 is employed for stacks of images of the same scale, and for down-scaling, Mitchell-Netravali and Cubic B-spline. The default value of 0.3 for the clamping threshold is normally sufficient to stop dark pixels appearing around high-contrast objects. This works similarly to the deringing controls in deconvolution and sharpening tools. Lowering the value softens the edges further. Since these dark pixels are random, one can also remove isolated dark pixels using statistics during image integration.

Integration (4)

The last of the four processing steps is the one in which user-input plays the biggest part in the outcome and where the default settings may be some way off optimum. Integration is about combining your calibrated and registered images in a way that keeps good data and rejects the poor. Keeping and rejecting are relative terms: It can physically mean that certain pixels from a subframe are totally ignored and at the same time that the remaining pixels contribute to the final image in proportion to the subframe quality. This selection and weighting process assumes that the computer can compare subframes with some certainty. Subjectively, our brains can determine a hot pixel in an image as it sees things in context to its surroundings, but consider a program trying to compare images of different brightness.

The ImageIntegration tool (fig.12) takes things in its stride and the statistical methods that underpin its workings are not for the faint-hearted. We can conceptualize what the tool needs to do and the settings it requires to make the best judgments. At its heart are two processes; statistical combination (in its simplest form, averaging) and pixel rejection. The tool identifies which pixels to reject in each image and statistically combines the remainder. Pixel rejection is very important and is done by comparing a pixel value in a particular subframe to the corresponding pixels in the others and a fixed reference. To do this reliably it needs to make an adjustment to each subframe (normalization) so that it can directly compare images statistically to identify reject pixels. Normalization is also required prior to the image combination process, to adjust the images so that they are similar. To explain this last point, which is not necessarily obvious, if we consider the case of a set of near-identical images, a simple statistical average may suffice, but in the case where the background illumination, exposure or camera gain changes between sub-frames, the brightest images will dominate the overall average and these may be brighter due to mist and reflected light pollution! Remember, scaling an image does not change its signal to noise ratio.

Normalization takes several forms and the precise setting is optimized for its purpose; rejection or combination. Given that we have two sets of normalized images, the user now has to choose the statistical method to correspondingly identify reject and combine pixels. Finally, during the combination stage, one has the choice to favor (weight) some subframes more than others based on particular criteria, for example, subframe signal to noise ratio (SNR).

With a broad understanding of what is going on under the hood, it is easier to review the principal options in each of the tool sections:

Input Images

This section looks familiar with many other tools but there is a gotcha that requires a little explanation. In addition to the standard image add, clear and selection buttons, there are a few concerning drizzle that we can disregard for the moment and another, named Set Reference. The referenced file is a registered subframe that is used as the template for registering and matching the other images to and from which the quality and image weighting is judged. By default, it is set to the first file in the list but for best results, choose it carefully.

fig.14 These two images show the effect of image integration on an image stack. (Both images have had an automatic screen stretch.) On the left is a single uncalibrated sub-frame and on the right, the final integrated stack of the best of 30 images. The background noise has diminished slightly but more importantly, the signal level had been boosted enormously.

This image should ideally have the best SNR of the group and at the same time have the most even illumination and the least defects, such as plane trails etc. I often start imaging low in the east and track across the meridian until I run out of night, horizon or weather. As a result, my first image always has the worst sky gradient and poorest seeing and is a long way from being the optimum reference image. To select the best reference, identify the top ten images using the report from the SubframeSelector tool (using SNR as a guideline) and examine them for gradients and defects. Then choose the best of these as the reference file.

Image Integration

In the main Image integration section, we have the combination, image normalization, weighting and scale options. For image combination, the Average (mean) or Median options are preferred. Of the two, Average has better noise performance. In one of the practical chapters in edition 1, I used median combination to eliminate an aircraft trail. I have since realized that I can do that with average combination, which achieves a better overall signal to noise ratio and tune the rejection settings to remove the pesky pixels. The image weighting is normally left at the default setting of Noise evaluation, which provides automatic image weighting based on image data. (The Average signal strength option uses the image data too. It may not provide such a good result though if there are illumination variations, for instance due to a changing sky gradient.)

As mentioned earlier, the ImageIntegration tool has two distinct settings for normalization: one for image integration and another for pixel rejection, as the needs of each process are slightly different. Many statistical tools work well with symmetrical distributions, like pure Gaussian noise. In practice, however, other alternatives often work better with real data. In the case of image integration, since we are trying to match the main histogram peaks and dispersion of each image, choose the Additive with scaling option for normalizing the light frames.

There are a number of methods to calculate the normalization parameters. These are selected in the Scale estimator option box. Several of these statistical methods analyze an image using its median pixel value to locate the hump of its histogram (to identify the additive bit) and then work out the data distribution (to calculate the scale bit). The exception is the Iterative K-sigma / biweight mid variance scheme, or IKSS for short. This default value is the preferred safe choice since it accepts real data with skewed distributions and is less sensitive to random pixel values (that have not been rejected).

Lastly, before leaving the Image integration section, check the Evaluate noise and Generate integrated image options (that create your image stack). All these myriad settings are wonderful but at first, the selections are guided by science and faith. Help is at hand. The evaluate noise option is useful since it generates a report with numerical values for the image noise level (and improvement) of the final image and hence, is an ideal way to directly compare the performance of the various integration options. In the final report out (shown in the process console window after the ImageIntegration tool has completed) the goal is to maximize the median noise reduction figure. Every situation is unique and it is likely that each will benefit

from a unique deviation from the default settings to maximize its SNR.

Pixel Rejection –1

Pixel rejection removes individual pixels from an image that arise from special causes (cosmic ray hits, satellites, meteors, airplanes) and common causes (tracking issues, focusing issues, excessive noise). After normalization, the pixels are statistically identified and removed. If the rejection settings are too aggressive, there will be no spurious pixels but the signal to noise ratio will suffer as a result of less combined data. The opposite is true and obviously, the aim is to find a setting that just removes the unwanted pixels. The rejection algorithm itself is an interesting dilemma: No one algorithm works in all cases and hitting the tool's reset button changes it to No rejection, encouraging experimentation. Although trying out different techniques is illuminating, broadly speaking, the selection principally depends upon the number of sub-frames and to a lesser extent the image conditions. The following are useful starting points:

3–6 images	Percentile Clipping
8–10 images	Averaged Sigma Clipping
> 10 images	Sigma Clipping
> 15 images	Winsorized Sigma Clipping
> 15 images	Linear Fit Clipping (see text)

The Linear Fit Clipping algorithm is subtly different to the others: Rather than use a static mid-point about which to set rejection limits, it can adapt to a changing set of pixel values over time, for instance, a changing sky gradient. Although overall mean background values are normalized in the rejection process, subframes at low altitude will have a more extreme gradient to those at high altitude. Linear fit clipping works best with a large number of images.

We have already discussed that normalization occurs before the algorithms are set to work and in this case there are two main methods: Scale + zero offset and Equalize fluxes. The first is used for calibrated subframes and the second is the better choice for flat frames, uncalibrated subframes or images with severe changes in illumination across the frame (for example, sky gradients before and after a meridian flip).

The Generate rejection maps option instructs the tool to produce two images that indicate the positions of all the rejected (high and low pixels) from all the frames. These are used to check the rejection settings. In practice, apply a screen stretch to them and compare these to those images with known issues (for instance satellite trails). Beneath these options are the ones for clipping: The Clip low and high pixels options enable the statistical rejection of dark and bright pixels identified by the chosen algorithm. The Clip low and high range options exclude pixels outside an absolute value range (independent of the algorithm). The Clip low range option can be quite useful: On one occasion I had to rotate my camera after a meridian flip to find a suitable guide star. The image frame overlap was poor and I used this option to reject all the empty border space in the registered files. Without that, I created a patchwork quilt!

Pixel Rejection –2

There are quite a few alternative rejection algorithms. Once chosen, the irrelevant slider settings are greyed-out. Each algorithm has a low and high setting permitting asymmetrical clipping. This is particularly useful in astrophotography since most special causes only add electrons to a CCD well and in practice "high" clipping values are typically more aggressive than their "low" cousins. For those algorithms that use standard deviation (sigma) settings, the default values of 4 and 2 will almost certainly need some modification to find a value that is just sufficient to remove the unwanted pixels. In both cases, a higher value excludes fewer pixels. For the low setting, I find the point where I start to have very dark

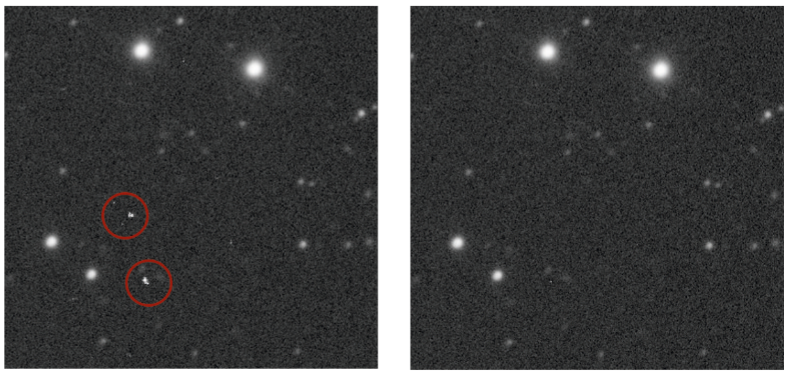


fig.15 The two image crops above show the difference between the default rejection settings and optimized settings (using a linear fit algorithm with tuned clipping parameters). The left image, using default settings, shows the integration of 30 images and has some spurious pixels from cosmic ray hits. The image on the right has been optimized and is a much cleaner result, but with very slightly more noise as a result of more rejected pixels (although this may not be visible in print).

pixels and on the high setting, since I invariably have a plane trail in one of my subframes, I gradually decrease the Sigma high value until it disappears. The other thing to note is that optimum settings may well be different for luminance, color and narrowband filters. The Clip low and high range settings are here too and populated by the default values 0 and 0.98 respectively.

Region of Interest

Experimentation is the key here but it is time-consuming. In common with many other PixInsight tools, to speed up the evaluation, it helps to try different settings on a portion of the image, defined in the Region of Interest section. Conveniently, one can open a registered frame and choose an image preview, that ideally covers a representative range of background and objects. PixInsight usefully has an image buffer and different rejection settings are quickly re-evaluated from the calculations in memory. The golden rule to PixInsight is to experiment and find the right compromise between improvement and destruction. As the figures show, integration always improves the depth of an image and it is principally the rejection criteria that strike a balance between random noise increase and the eradication of spurious pixels.

LRGB Luminance Enhancement

There is a further consideration in the image integration process. We know that high-quality imaging processes the luminance and color information follow two distinct workflows, optimized for enhancing detail and color information respectively. For instance, if one takes images with a DSLR, the DeBayered color image contains both color and luminance information and although some image processing tools can restrict application to the luminance information (for instance some noise reduction ones) an easier method is to extract the luminance information from the color image and process through a separate workflow. The "aha" moment is to realize an RGB image has both color and luminance information and when one considers image capture through separate filters, the LRGB processing workflow discards this luminance information with the LRGBCombination tool. This is a lost opportunity; the luminance information from those exposures taken through colored filters can be combined with the dedicated luminance channel to produce a result with improved SNR (the impact is dependent upon the quality of the color data). It also improves potential color separation too, since the luminance information from the color images is tuned to those wavelengths, as in the case of enhancing red separation by combining it with Hα data and assuming that the RGB images are taken with the same binning level as the luminance too, to preserve the luminance resolution.

There are many potential ways of combining the data and one has to sit back and consider the integration process and likely outcomes. The data from each channel is quite different; each has a different bandwidth and object color as well as potentially having a different exposure duration and quantity. Each noise level, signal level and distribution will be distinct from the other channels. A simple integration of all image subframes would have difficulty rejecting pixels reliably (the rejection criteria assume a single signal statistical distribution and potentially, in this case, there would be multiple distributions).

A more reliable method is to integrate the discrete luminance and color information as normal, optimizing each in their own right and then combine these stacks. A simple average with scaling, using the noise level as a scaling parameter and with no pixel rejection, produces a substantial noise reduction. Note: the ImageIntegration tool can only work on 3 or more images. This is not an issue when working with LRGB or LRGBHα information but in the case of combining the luminance information from a color image with separate luminance data, you will need to first extract the separate RGB channels from the color image using the ChannelExtraction tool.

When it comes to LRGBCombination, there is a further trick to help with the predictability and quality of the outcome. When the luminance of the RGB image and Luminance image are very different, LRGBCombination struggles. I found this useful trick from the PixInsight forum to make the process more robust:

1 Apply the ChannelExtraction tool to the RGB image (set to CIE L*a*b* mode).
2 Use LinearFit to match the L* channel to the Luminance image.
3 Reassemble the L*a*b* channels using the ChannelCombination tool.
4 Apply LRGBCombination

The outcome of introducing these extra steps makes the image much easier to tune with the LRGBCombination tool's brightness and saturation sliders.