In Part 1, we covered the basic notion of SNR and in Part 2, we covered SNR in a single pixel. If you've not read those bits yet, head back and give them a look. If you have, you've had your nose in the books for a bit now and it's time for a break. In this installment, we're going to take on the practical aspect of testing your camera and figuring out just what kinds of camera noise you're up against. Warning - this is a long one. You may find it helpful to grab the PDF of the article that I've put up (along with others in the series) on my personal website.

Believe it or not, you can get very accurate measurements on your camera with only a minimum of hardware, skills, and time. Here, you will set out to measure:

- System gain (number of electrons per ADU)
- Read noise (in ADU and electrons)
- Dark current (in ADU and electrons)
- Dark current stability

You can also go on to probe some of the inner workings of your camera and look for it's "fingerprint" as it work by some detailed analyses of the read noise. Here, we'll look at:

- Histogram of the read-noise (just how Gaussian is it?)
- Amount of fixed-frequency / variable-location noise (the worst kind!)

Believe it or not, you only need a few tools to do all of this and all of this can be done with the camera sitting on the desk next to you (no need for the telescope). All we need is the ability to take clean dark frames and to take reasonable flat frames. So, here's a parts-list:

- (optional) An SLR camera lens you can attach to your camera. If you've got this, you can take better flats and control the amount of light hitting your chip. If not, you'll live.
- A metal lenscap for the camera or SLR lens. If you don't happen to have this, a piece of tin foil and a rubber band will do.
- About 15 sheets of white paper roughly 4x4 inches or so apiece.
- ImageJ. Freeware image processing and analysis software. Much can be done in other programs, but ImageJ does give you nice FFTs
- (optional) A spreadsheet program to graph your results and do things like fit a linear regression line. Excel, of course can do this, but even though I own Excel, I end up using a Mac version of OpenOffice called NeoOffice. If you've not got Excel, OpenOffice is free and available for most any platform. Yes, you can do what you need to old-school with graph paper, but... c'mon. You can also use Google Docs, but you'll need to do one thing by hand rather than right on your plot.
Getting the Data

We're going to collect a bunch of bias frames, several dark frames, and some flat frames. Get your camera setup, but don't get it turned on and going yet. Keep it at ambient temperature. I do all of this on a desk without a telescope attached, as there is no need at this point for any kind of lens.

First up are the biases and darks. Here, we need to make sure that no light whatsoever is getting to the sensor. Believe it or not, black plastic lenscaps are often pretty transparent to IR light. This is why I said you need a metal lenscap. A perfectly good solution is to use your camera's 1.25" or 2" nosepiece and to wrap a piece of tin foil over the nosepiece. Hold it all in place with a rubber band. Voila! Perfect dark frames.

Most cameras these days are very light-tight, but some still aren't. If you're worried that yours isn't (or you know it isn't -- look at a dark frame and see if there is one side that's brighter than another), you'll need to shade the camera body from any ambient light. One way to do this is to work in the dark (just don't aim your computer screen at the camera). Another is to put a box over the camera. If you go that route, make sure there's enough ventilation still to keep the camera from getting abnormally hot. Your goal here is typically to keep the camera shaded and not in direct light. If it can't deal with a small amount of reflected light, you've got bigger fish to fry.

Now, fire up the camera and connect in your capture software. Right away, fire off:

1. A set of 1-minute dark frames (at least 30). These are used for your dark stability measurement. Since the camera is at ambient, we get to see how its dark current changes as you get going. If you've got cooling, it'll start to drop as you head towards your set-point or the max-cool level. If not, your camera will start to warm up here. Once done, your camera should be at some kind of thermal equilibrium. Some cameras do need more than 30 minutes to hit this, though.

2. A large stack of bias frames. These will be used in a number of measures. These days, I grab 250 of them to be safe. 50 would probably do just fine, but unless there's a compelling reason not to, grab at least 100. The exposure duration here is typically set to 1 ms, so it's not like this should take a long time. Note, even if your camera isn't thermally equilibrated yet, you can still get these. At 1 ms, there's no dark current to speak of.

3. A set of dark frames of varying length. These will be used in calculating the dark current. I typically grab 1 m, 2 m, 5 m, and 10 m frames. Note, if you think your camera may have thermal stability issues (i.e., it's uncooled), you may want to space these out. So, wait 15 min or so after the bias frames and then grab the 1 m frame. Wait 15 min again and get the 2 m. The goal here is to let the camera's temperature stabilize.

After this, you'll want to setup for flats. This will enable you to calculate the system gain of your camera. Here, you need to have the ability to take pairs of flat frames of varying brightnesses, ideally without changing the exposure duration. The goal here is to take pairs of flats, perfectly matched in intensity, for a range of intensities. There are two methods that I've used:

- An EL-panel with variable brightness that sits on the front of an SLR lens (camera and lens aimed up). This is the uber-cool way to grab the flats as you can dial in any brightness desired. Set the exposure duration to something like 0.1 s and the panel to
something dim and grab a test shot. Adjust the exposure duration and/or f-ratio on the lens to be above the level of a bias frame by just a hair. You can now adjust the brightness of the panel to increase the brightness of the flat.

- A stack of white office paper acting as a diffuser. Start with ~4 sheets on the nose of the camera or on the front of your lens and aim the rig at something like a white ceiling. You won't need to be perfectly flat and this will get you quite close. Setup for a short duration (e.g. 0.1 s or so) and use your capture program's histogram to see how bright the image really is. The goal here is to be near but not entirely at the top of the histogram. To adjust the brightness of the flat, you'll simply add another piece of paper onto the stack.

Now, take pairs of flats at various brightness levels. Make sure that the overall brightness level covers a good range of the intensity scale. The figure here shows the histograms (plotted from Nebulosity) for ten different intensity levels I used in testing the Atik 314L+ here. You don't want to bottom out and be looking like a bias frame, but you don't want to saturate the CCD either. Err on the side of being in the lower-half here as above this, your sensor may be non-linear. I'll typically use at least five brightness levels. You can do this with just one, but you'll be less prone to error with more. Make sure you name them with a convention that will make sense later. For example, you might have Flat1_001.fit, Flat1_002.fit, Flat2_001.fit, Flat2_002.fit, Flat3_001.fit, Flat3_002.fit, etc.

While here, I like to confirm where saturation is on the camera. Take the paper off if you like and dial in a much longer exposure. Mouse-around the image and see if you can read off values of 65535 (the maximum possible in a 16-bit camera). If not, increase the exposure to something like 10 s. If you still can't get to 65535, note the approximate maximum you can get to. This will be useful for estimating the approximate full-well (technically the maximum number of electrons you can record).

A few things of notes on the flats. First, you don't have to worry about dust motes too much. We can work around them by either ignoring them or by cropping around them. Second, some sensors may behave oddly with no lens attached or with a very low f-ratio lens attached. Most don't, but some do want a reasonable light cone. Here, using an SLR lens or your telescope will be needed. If your flats look reasonable, don't worry about this though as most sensors are fine. Third, make sure you're capturing your data in a raw format. If you've got a color camera here, we don't want it to be a de-Bayered color image here.

**Analyzing your Images: Basic Specifications**

Now comes the fun part - seeing just how your camera's behaving. We'll cover a range of measurements, starting with the one that's most annoying. We do this not only to get it out of the way, but also because it's what gives us the ability to convert from simple intensity units (ADU) into actual electrons. I'll be using test data collected for a review of the Atik 314L+ I'm working on right now as an example.
System Gain

The system gain of your camera is the conversion rate between the raw numbers you get out of the camera (ADU or Analog Digital Units) and actual electrons. Knowing it helps you interpret the other measures as you get to express things like read noise in real units (e-) rather than in arbitrary units (ADU). It also gives you an assessment of just how many electrons you can record (which is an estimate of the full-well capacity, or at least places a lower-bound on the full-well capacity of the sensor). There are two ways to calculate the system gain: a quick and dirty one and a more involved one. I favor the more involved one described by Tim Abbot as it's more tolerant of errors (a very similar one can be found on the Apogee CCD University page).

If you decide you want to do the quick and dirty one, you only need a pair of flats and your master bias. The formula you need to compute is:

$$\text{gain} = \frac{\text{mean}(\text{Flat1} + \text{Flat2})}{\text{var}(\text{Flat1} - \text{Flat2})}$$

where \text{var} is the variance, here of the difference image between your two flats, and \text{mean} is the mean of the image (here of the sum of the two flats). Since the average signal in Flat1 is really the average signal in Flat2, you can simplify this into:

$$\text{gain} = \frac{2 \times \text{mean}(\text{Flat1})}{\text{var}(\text{Flat1} - \text{Flat2})}$$

You can compute this with ImageJ, but we're going to take the longer route here. We're going to do this because any issue you may have with either of your flats will drastically throw off your estimate of the system gain without giving you any way of knowing there was an issue.

The longer route is really just an extension of this shorter route. The shorter one is using two points to estimate a line and the longer one is using several (based on the number of pairs of flats you took). It's really not so bad to do the longer route:

First, start a spreadsheet with two columns. Label them \text{v} and \text{m} for \text{variance} and \text{mean}. For each pair of flats, you'll calculate a value for \text{v} and \text{m}. We'll do \text{m} first.

Second, for each flat pair, calculate the mean intensity level (or median intensity level) across the whole image for one of the flats and multiply this by 2. This is your \text{m}. Your image capture / processing software may give you this. If it doesn't, it's trivial to calculate in ImageJ. Pull down \text{Analyze, Measure} and a dialog will appear that includes the mean signal level in your image.

$$\text{m} = 2 \times \text{mean}(\text{Flat1})$$

So, in my first pair of images, looking at Flat1, I have a mean of 6244. In the first entry in my \text{m} column, I'd then enter 12488.

Next, for each flat pair, make a difference image. Start off by load both images in ImageJ. Before we actually subtract one image from another, we will add a constant value into one of the images. This is so that we can cleanly subtract \text{Flat2} from \text{Flat1} without "clipping" the
data. If a given pixel in Flat2 is 100 and in Flat1 is 110, life is good and we have a difference of 10. If the pixel in Flat1 is 90, however, we have -10 for the difference. These images don't allow negative numbers, though, so it will get clipped to 0. This will throw off our estimate of v.

The solution to this is simple. Select Flat1 (which may be actually called Flat1_001.fit or something) and pull down Image, Math, Add and type in a number like 5000. (The actual value here won't matter. It needs to be big enough to cover the maximum difference between the images, though). Next, we'll subtract Flat2 from this new Flat1. Pull down Process, Image Calculator... In the dialog that pops up, have one flat be Image1 and the other flat be Image2. Select Subtract in the Operation section.

As before, we now want to measure this resulting image. So, pull down Analyze, Measure and that dialog will again pop up. Here, we're interested in the standard deviation measure. (If, for some reason, you don't see a standard deviation value, pull down Analyze, Set Measurements and check Standard Deviation). The standard deviation is just the square root of the variance (i.e., the variance is the standard deviation squared). So, we can calculate v as just:

\[ v = \text{var}(\text{Flat1} - \text{Flat2}) \]
\[ = \text{stdev}(\text{Flat1} - \text{Flat2})^2 \]

When I ran this on my first pair of flats, I see that the mean of this difference image (Result of Flat1_001) is 5000.43 with a min of 3948 and a max of 6052. This is good as it shows that my difference image doesn't have any zeros in in (min > 0) and it isn't clipped on the top end either. The StdDev column shows 212.495 here, so for the v column in my first pair of images, I'd enter 45154.

Repeat this process for each of your pairs of flats. You should end up with a row of numbers for each pair of flats with each row having a pair of numbers. If you like, you can, of course, have your spreadsheet do a bit of the math for you by calculating m and v from the means and standard deviations given in ImageJ. As you do this, keep an eye on the Min and Max values reported when you run Measure to make sure that you're not hitting 0 or 65535 and clipping your data.

In the end, you should have something that looks a bit like this. Here, I've entered values from four of the flat pairs from this Atik 314L+. Next, we need to perform a linear regression analysis. All this means, is that we need to fit a line to the four points we've just created. Select your data and tell your spreadsheet program to insert a chart. When asked what kind of chart to make, tell it to make an "XY Scatter". With luck, your points will all line up nicely.
with each other. If visually, things look like a line, proceed to the next step. If, you've got most that form a nice line but a few that are way out of line, simply delete those points from your data. Outliers typically come about from errors in your processing or image capture process or from clipping the data (e.g., hitting the saturation point of the CCD).

Next, it's time to fit that regression line. If you select your data series in the chart by clicking on one of the points in it, you'll typically have the option to add a "trend line". Different programs let you get to this in different ways, but most spreadsheets will let you do this. What you want to do is to fit a "Linear" regression and to "Show the equation" in the chart.

The equation will have two parts. In the example here, it says that the regression is equal to "0.27x + 292.09". That bit before the "x" is the slope of the line (you may recall the formula for a line is $y=mx+b$ - this is the $m$). That slope is your system gain. It is the number of electrons per ADU. Note, typical values for this will be between 0.2 and 1.5. If you've got a number a lot higher than this, you may have flipped your $m$ and $v$. If so, your y-axis will have smaller numbers than your x-axis and your system gain is $1/YourValue$.

From this slope, you can estimate the full-well capacity (or the maximum number of electrons that can be recorded before the ADC saturates, whichever is less). Multiply your slope by the maximum intensity you can get out of your image (probably 65535, but on some cameras it'll be a bit less). Here, I get about 17,700 e-.

**Special Note 1:** This works very well if your flats are fairly flat. If they're not and if you're vignetting a lot or if you've got a whole dust-bunny warren in the image, you may want to crop a section of the image out of each flat. If you do, make sure you are cropping the exact same portion of the image out of each flat. You can either do this by carefully watching the cursor position as you crop each image or by using a cropping tool that lets you specify where to place the crop. *ImageJ's Adjust Canvas Size will let you do this.*

**Special Note 2:** In addition, if you've got a one-shot-color camera, the "Bayer Matrix" or "Color Filter Array" on your camera may cause issues. The problem is that each color channel can have a decidedly different mean in your flats. For these cameras, I use a tool to extract one of the color channels from the raw, Bayer-encoded image. A number of programs will let you do this (e.g., Iris, Nebulosity, Maxim DL, etc.)

**Special Note 3:** If your spreadsheet does not have the ability to give you the equation for the line on the plot there, fear not. You can use the **LINEST** function, passing in the $v$ values for the x-data and the $m$ values for the y-data. The slope parameter returned is the number you're looking for.
Read Noise

The system gain was by far the worst one to do, but we've gotten it out of the way and it will now let us have the other measurements be in real numbers. Next up is the camera's read noise. Recall that every time you read an image, you have some noise. This is why even with no light hitting the sensor and no dark current (bias frames), images look different.

You can typically get a good estimate of the read noise by just taking the standard deviation of a single bias frame. So, if you open up a bias frame in ImageJ and with a bias frame pull down Analyze, Measure you'll end up pretty close to the real value. But, if you want to do it right, you need to do a few extra steps.

First, if you've not made a "master bias" from all those bias frames, make one now. Use your image processing software to stack all of your bias frames (no alignment, of course) and average them all together.

Next, load up that master bias image and three or four individual bias images in ImageJ. As in the system gain measurement, add something like 5000 to your master-bias image. Then, subtract an individual bias image from the master bias image using the Image Calculator. Do a Measure on this and look at the standard deviation. This is one estimate of your read noise in ADU.

Repeat this for each of the individual bias images. It's a good idea to either keep these images open or to save them as you'll need these (and the master bias image) later on. On the Atik 314L+ here, an individual bias frame had a standard deviation of 13.93. The standard deviation of this difference image is 13.8. As you can see, we're pretty close with the two methods. The next two bias frames I tested, when subtracted from that master bias, read 13.8 as well. So, I know this is a nice, reliable measure. Average your numbers and this is your read noise in ADU. Multiply that number by your system gain (0.27 here) and you have your read noise in e-/ADU. Here, the Atik turns in an exceptional 3.7 e- of read noise.

Dark Current

On many cameras, dark current can be measured very easily. If you've got a cooled camera, all that is needed is to measure the mean of a bias frame and subtract this from the mean of a long dark frame. In the Atik 314L+ I have on the bench here, the mean of a bias frame is 232.5 and the mean of a 10-minute dark frame is 234.2. That means that in a 10 minutes of exposure, my average intensity went up by 1.7 ADU or 0.46 electrons. Typically, this is specified as electrons per second, so we divide this by the number of seconds in this interval (600 seconds) and get 0.00076 e-/second. This is a very low number (and is why I've often said that regulated cooling and the use of dark frames is really unnecessary on these Sony sensors - a cooled dark frame is almost exactly the same as a bias frame).

If your camera isn't cooled or if you think there might be something odd going on (or if you just want a bit cleaner estimate of the dark current), you can do the same thing you did in coming up with the system gain. In a spreadsheet, make one column for the exposure time and another column for the mean value of the dark frame at that time. Plot time on the x-axis and the dark current value on the y-axis and again do a linear fit. The data should fall on a line. If they don't something is odd as doubling the exposure duration should double the number of photons from dark current being recorded. Note, when done this way, the Atik turns in an even lower dark current of 0.0005 e-/second. The current is so low, it's really tough to estimate!
Dark Stability

When you collected your images, I had you collect at least 30 1-minute dark frames. This was so that you could evaluate how much the dark current changes over time. Load up each image in ImageJ and calculate the mean (average) signal, again with the Analyze, Measure tool. In your spreadsheet program, make one column (time) and enter the numbers 1-30 in there (or whatever numbers correspond to the number of darks you took here) and enter in the mean signal for the corresponding dark frame.

Again, do an X-Y plot of these data (if you like, you can select just the mean dark value and do a simple line or column plot as the x-axis is evenly spaced). You’ll probably find that the camera’s dark current changes a bit early on. For cooled cameras, you’ll see it drop down to the set-point or to the deepest cooling point it can muster and stay relatively stable. How long does it take to get there? This will let you know how long you should let the camera stabilize before imaging. For uncooled cameras, does it reach a relatively stable point and rise no more after some amount of use? Again, this will tell you how long you should run the camera before you expect the dark current to be repeatable.

Analyzing Bias Frames and Read Noise

At this point, you’ve gone through and come up with some key benchmarks on your camera. You know its system gain, its read noise, its average dark current, and how stable the dark current is. Hopefully, you’ve also learned some tools and are now a bit more comfortable analyzing the performance of your camera. We’re now going to look a bit deeper into the camera’s performance by investigating the bias frames and the character of the read noise.

Before turning to your camera, it’s probably worth seeing how an ideal camera would behave, as much of what we’ll be looking at here isn’t as clear as a simple number. In ImageJ, we can create an ideal bias frame from a camera with a clean sensor and nothing but pure, Gaussian noise. Pull down File, New and enter an image size of 256x256 with a background set to black. Next, add an offset to this by entering Process, Math, Add and entering a value of 100. You should now have a small gray image. If you were to run the Measure tool on this, you’d end up with a Min, Max, and Mean of 100.

Next, add some random, Gaussian noise to the image by pulling down Process, Noise, Add Specified Noise, and give it a standard deviation there of 10. Running the Measure tool now should give you a Mean of about 100 still, but the Min and Max will now be different - perhaps about 50 and 150 respectively. The standard deviation should be about 10 (since we made an image with a mean of 100 and added noise with a standard deviation of 10...). It’s probably worth saving the simulated image at this point.

Histogram of Simulated Bias

Pull down Analyze, Histogram at this point and you should see a nice, smooth histogram of your image. Again, it will show you the mean, standard deviation, minimum, and maximum. Hit the button marked Log to look at a logarithmic-based histogram. All this is doing is making the y-axis (height) of the histogram use a
logarithmic rather than a linear scale. In log scales, the y-axis is distorted. For example, the
distance between values of 1 and 10 would be the same as the distance between 10 and 100
or 100 and 1000 (this would be a log_{10} scale).

The figure here shows what you should see. Keep this figure on hand as it shows what a
clean image really looks like. Deviations from this are not desired. We want something
symmetric and that roughly resembles the nose cone of a rocket. Of course, it can have
various widths, but it should have this basic shape.

**FFT of Simulated Bias**

A bright bloke named Fourier came up with the idea that any signal - be it an sound, an
image, a 3D shape, etc. - can be broken down into a series of sine waves. If you were to take
sine waves of all the possible frequencies and combine them, adding varying amounts of
each frequency, you could build up anything. If you've ever looked at the dancing lights of a
spectrum analyzer on a stereo system's graphic equalizer, what you're looking at is the
amount of energy in each of several audio frequency bands. This information is being derived
by a Fast Fourier Transform or FFT. What we're about to do here is to analyze not the audio
frequencies in sound, but the spatial frequencies in an image. If, as you move across an
image you slowly ramp from dark to bright to dark, there is some energy at a low frequency.
If, as you move across, you go very rapidly from dark to bright to dark again in only a few
pixels, there is energy at a high spatial frequency. Our goal here is to determine how much
energy there is at all possible frequencies in the image. (Note, we never have all frequencies
in an image as there is a limit on the highest possible frequency that can be in an image. The
Nyquist Theorem tells you how high a frequency can be encoded in an image. Spatially, this
is two-pixels wide.)

If you've still got your image before you added the noise around (re-make it if you don't), pull
down **Process, FFT, FFT**. You'll see a black square with one bright pixel in the middle. The
middle of the FFT refers to 0 Hz, or "DC", or the constant offset in the image. What this is telling
us is that we could recreate your frame here by adding only a single constant to the image. It's
right, as the image at this point is a perfectly even gray.

Now, run the FFT on that bias image you faked. You may need to zoom in, but what you should
see should look roughly like this. A bright dot in the middle with some random noise around this.
What this is saying is that you can re-create your image by adding in a constant offset (the bright
dot in the middle), and a number of entirely random values of random frequencies. No
frequency (other than 0 Hz) is over-represented in the image.

This is really as good an image as we can ever hope for. There will always be noise in our
images, but what we hope is that the noise is entirely random. Random noise will go away
when stacking frames. Noise that isn't random will not go away and will build up. Remember,
that's exactly what we're trying to do with our signal. Our signal consists of spatial frequencies that we want in our image. Stacking lets these remain while the noise goes away.

**For Fun:** If you want to get a better handle on FFTs, try doing this. First, open an image of a normal daytime shot. It may be useful to rescale it down to something a bit smaller than full-size. Here, I've taken a shot of one of my sons, Miles, at the beach. In **Process, FFT, FFT Options**, turn on **Complex Fourier Transform**. Now, do an FFT of the image. Two FFT windows will appear, looking something like the next two images here. This is the full FFT of the image. Select one of these and pull down **Process, FFT, Inverse FFT**. You'll now end up with an exact replica of the original image (top row, right). You took your image, converted it into the Fourier domain (into a frequency and phase pair of images) and then took that Fourier representation and converted it back to an image. Pretty slick eh?

In the next row here, I blanked out portions of the frequency image. By doing so, I'm cutting out a range of frequencies in the image. Pixels closer to the middle of the frequency image are lower and those closer to the edges are higher. So, here, I've cut out the higher frequency components. In one, I cut out a lot more than the other. The inverse FFTs of these restricted-frequency images now look a bit softer don't they? That's the loss of the high frequency detail. See how much you can remove before the image starts to degrade. Think this could be a good way of compressing, smoothing, or sharpening your images?
Analyzing your Cameras Read Noise Frame

If you've not looked at FFTs before, I don't expect this quick introduction will have you feeling like you've mastered the ideas. Hopefully, at this point you have some ideas what to look for. There are a number of good descriptions of this on the web with the one at QSI being a particularly good example for us. A perfect FFT will show a bright dot in the center and simple noise elsewhere. If there are bright dots or lines elsewhere in the image, it means there are spatial frequencies in the image. That is, there is structured noise. Our goal here is to examine this noise and to determine just how repeatable the noise is. If it's repeatable, it's removable with things like bias frames and/or dark frames.

Go back to (or open up) the master bias frame you saved before and one of the images you made by subtracting an individual bias from that master bias back when we were measuring the read noise. If you run an FFT on the master bias frame, you certainly may see something that doesn't look ideal. Here, for example, is the master bias frame from the Atik and its FFT.

You can see in the average bias that there is an odd banding on the left side of the image. The bias stack here is stretched incredibly as the total swing in the image from the dark bands to the light is about 4 ADU (on a full 16-bit scale). Likewise, the histogram shows that it's not the perfect shape (the fact that the histogram is made up of just a few spikes, though, shows that the variance in this bias frame is extremely small). Nevertheless, something is here. Something happens during the readout of the sensor to cause this slight variation in the intensity level. Since we're seeing this in a stack of 200 bias frames, odds are this is something that exists in the same place in each bias frame (or we'd never have seen it build up). If it is there in every frame, it'll come out of our light frames by subtraction. If not, or if
there is anything else that is in the bias that varies from frame to frame, we'll see this in our read noise frame.

The image you calculated before - this master bias minus a single bias - is a read noise frame. What is left over in this subtraction is what the camera is doing differently each time it reads the image. What it’s doing the same each time got subtracted away. This is what it does differently each time and what will show us the “fingerprint” of the camera as it were.

So, instead of running on the master bias frame, have a look at the histogram and FFT of this difference image - your read noise image. On the Atik 314L+ here, visually, the read noise image looks very clean. Those bands have disappeared and we’re left with something that looks like pure noise.

Looking at the read noise histogram, we see excellent performance. There are no clear "shoulders" to the histogram and overall it has a good shape. It’s not perfect, as if you squint there is a hint of a "tail" on the right, but this is excellent. We can see just how good it is by again creating a blank image, adding an offset, and adding Gaussian noise to match the
values in the camera's image. I've included one here as a sample (note, if you do this, make sure the range from Min-Max is about the same in your simulation as it is in your camera's image or the histogram will differ considerably in width). Having tested a lot of cameras, I can say without reservation that this one is very, very good and there is nothing to complain about here.

Next, we can turn to the FFT of the read noise frame. Here, on the left, we see the read noise frame itself and on the right we have its FFT. As noted, the read noise frame looks very nice and smooth and it's clear from the FFT that there is nothing periodic about the noise. There are no bright lines, extra dots, etc. in the FFT. If one zooms in on it, the central dot is clearly visible (as it must be), but there is little else in the image.

Thus, we can conclude that this camera's read noise performance is excellent. The histogram is excellent and we'd be reaching here to find anything wrong with the camera. The FFTs show that the read noise is nicely random and there are no large patterns that will easily detract from the image.

**Conclusions**

This certainly was a long entry here and I hope that at least after several attempts, you've made it here to the end. While long, we covered a lot of ground. We covered how to get critical basic performance specifications on your camera that you might have thought were well beyond your reach, yet only required very simple tools and math. We also covered how to go deep into the analysis of your camera's electronics to see what might lie deep in the noise, but that might build up to hurt your final image.

We're not quite done with SNR here yet. We still have topics to cover like how what we know about SNR now should influence things like how we choose an image scale and what implications this has for the infamous f-ratio "myth". At this point, what I'd like to do though, is to hear from you, the reader. What parts of this haven't made sense? What questions do you have on this? I'm sure you've got questions, so drop me a line either here on the forums or by direct e-mail. I'll try my best to answer them and to shed some light on things in an upcoming entry.

Until next time, clear skies!

Craig