Astro 121, Spring 2014
Week 5 (February 20)

Topic: Characterization of CCD data
Break: Sara

Topics: Our main concern next week is how one actually measures the CCD characteristics we have been discussing (and in general just getting comfortable with taking and analyzing CCD data). Once you get through this week's work, you'll be in good shape for working with some real astronomical CCD data next week for a photometry project.

Note that we are skipping over a few chapters of Chromey and going in a slightly different order. Chromey discusses optics in Chapter 5 and telescopes in Chapter 6. This is a logical ordering (the light has to go through the telescope before it reaches the detector), but I want to get you started working with data, so we're skipping ahead to start to talk about CCD data. We'll return to this material in a few weeks. Chromey also discusses the physics of how photons interact with matter in Chapter 7. This covers the fundamental principles used in detectors, but in the interest of time, we won't cover it in this course so we can spend more time than Chromey does in discussing observations outside the visible spectrum.

Reading:

- Chromey, Chapter 8, pp. 235–260. Chromey goes into more detail on the different types of CCDs than we will discuss, so pp. 254–260 are less crucial than the first parts of the chapter.

- *Handbook of CCD Astronomy*, Howell. Chapters 3, Sec. 3.4 to end of chapter 3; and through Sec. 4.3 of Chapter 4. Note that the sentence about read noise in the third paragraph on p. 46 is incorrect. Read noise is an *uncertainty* in the number of electrons detected, and thus it can increase or decrease the readout value compared to the true value; it is not a number simply *added* to each pixel.

- Optional: *The Handbook of Astronomical Image Processing*, Berry and Burnell. Chapter 1, pp. 1–17; Chapter 4. In Chapter 1, you can skim pp. 1–8; we'll come back to some of those concepts in a few weeks. I'll put a copy of this book in the astro lab.

Problems

1. Derive the four equations on pp. 72–73 (pp. 53–54 in the first edition) of Howell. Watch out for Howell's inconsistent notation, though. For the first equation, F represents the average counts (in ADU) in the flat field after the bias level has been subtracted, but for the next two equations, F_1 and F_2 represent the average counts (in ADU) in the flat field images *including the bias level*. (In practice this doesn't matter much since the counts in a flat are likely to be much higher than those in a bias.)

2. The detector for the Pan-STARRS survey (http://pan-starrs.ifa.hawaii.edu/public/home.html) is a 64x64 array of CCDs, each of which is 600x600 pixels, all on one silicon chip. If this whole thing were one massive conventional CCD, how long would it take to read this array through a single amplifier at a pixel frequency of 1 MHz? This is why many large, modern CCDs have multiple readout amplifiers (for example, one at each corner of a chip) or (as in the Pan-STARRS case) employ a different kind of readout altogether that can address individual pixels directly.

Data analysis: Now that you've been exposed to the basic ideas of CCD calibration and of signal-to-noise measurements, the best way to get really comfortable with them is to deal with some real data. The project outlined below will help you do this. I haven't laid it out in a step-by-step cookbook fashion. Instead, it is more like a real scientific problem: you have a set of questions you want to answer, and you have some resources that help you figure out how to get started finding the answers. (Admittedly there is no new science in determining these quantities; but a good understanding of your detector is often the first step toward using it to *do* some real science.)

So, your assignment for next week is to assess the vital statistics of one of our CCDs: this year we'll use the imaging CCD we have for the telescope, an Apogee U16M. The analysis includes measuring the following quantities:

1. Read noise, in electrons/pixel.

2. Gain (electrons/ADU)

3. Dark current (expressed in e^-/minute/pixel at operating temperature).

4. Uniformity of flat field (i.e., are there any notable features observable in the flat?)

5. Signal at which the CCD saturates.

6. If it's possible to find the information, how any or all of the above compare to the manufacturer's specifications for that chip. (To be clear, in the above I am asking you to make your *own* determination of those quantities for *our* particular CCD, and here I'm asking you to look up some representative values for that kind of CCD. There definitely can be sample variation due to the manufacturing process for the silicon in the CCD, so each Apogee U16M may not exactly match the manufacturer's specs. Typically when you buy an astronomical-grade CCD, you receive a test sheet from the manufacturer showing the properties of your particular copy of the CCD, tested before shipping.)

Numbers 3, 4, and 5 are fairly straightforward and can be determined from examination of just one or a few images each. The others will require somewhat more detailed analysis.

In order to take and process the data necessary to do this, you need to be able to do a few major things, listed below. Hopefully, after this week's experience, you are comfortable tackling these, but feel free to ask for any help you need.

- How to operate a CCD camera to take data. The main control software we use in the observatory is MaximDL.

- How to log in to our cluster of Linux machines and start up programs. A number of you have done this already, but if you haven't, let me know and I'll show you the basics you need to know to get around.

- How to use IRAF or AstroImageJ to analyze CCD data. Even though AstroImageJ is a little more user-friendly for processing data, you may find IRAF a little easier to use for these problems, which involve some image arithmetic and statistics.

Once you can do these things, not only will you be able to determine the quantities mentioned above, but you will also be well on your way to having the tools necessary to do real observing. (The elements still missing will be some experience with locating celestial objects with a telescope, and doing real photometry of objects in your data. But it's helpful not to have to learn *too* many new things at one time!)

All of the above can be determined by using bias, dark, and flat-field frames. (A flat is just a high-signal image of a uniformly-illuminated source.) Thus, you can do all of this with observations made during the day. The bias and dark don't require the camera shutter to be open, and the flat doesn't require a dark sky. Be sure the camera is cooled down before taking data. For the flats, you will need to observe around dusk – the daytime sky saturates the CCD too quickly. Through trial and error, I have found that the sky brightness is about right to get good flats at around *civil twilight*, defined as the time when the Sun has set and is 6 degrees below the horizon. There are also other kinds of twilight (nautical and astronomical), defined at http://aa.usno.navy.mil/faq/docs/RST_defs.php. Also linked from that USNO page is one where you can look up the twilight times for our location in order to plan your observations.

Resources:

- IRAF documentation: IRAF has built-in help files for each task (just type "help *taskname*" from within IRAF; if you don't know the task name, see the next point). Also, there is more detailed documentation available. A number of IRAF documents are printed out and in a binder on the shelf in the astronomy computer lab (SC 124). Also, IRAF help is available on-line; see below. Finally, you should feel free to ask me questions. The point of this is to learn by doing, not to beat your head against a wall trying to get IRAF to do what you want.

- There is an introductory IRAF exercise that I've posted on the course web page. If you haven't used IRAF before, you should sit down and go through this to get a feel for using IRAF. It should take an hour or so, but you'll be in much better shape to start working productively with your data.

- If you're trying to find a particular IRAF task to do what you want, you can search all of the IRAF task descriptions by going to the IRAF web page (http://iraf.noao.edu/) and clicking on the "Documentation: IRAF help" link on the left; there's a search form at the bottom of the page. You can also use the command "reference" within IRAF, e.g. "reference gaussian" will list all of the tasks the mentions Gaussians in their task descriptions.

- DS9 documentation. DS9 is an image display tool for use with IRAF. Docs are available on the web at http://hea-www.harvard.edu/saord/ds9/ref/index.html .

Suggestions:

- The procedure for measuring read noise and gain depends on the assumption that the standard deviation within the image is due only to Poisson noise and read noise. But sometimes other noise sources come in, such as cosmic rays, which cause very high flux levels in single pixels on the detector. These can skew your standard deviation and lead to incorrect results for gain and/or read noise. (You might not think a single pixel out of a million would affect the standard deviation, but a cosmic ray hit could be 10,000 e^- or more above the mean, and the standard deviation is based on the *squared* deviation from the mean.) IRAF has a routine to detect and remove cosmic rays (it's called *cosmicray*), but it doesn't work as well on bias or flat images as on actual star fields. One way to roughly get rid of these bad pixels is to use the task *imreplace*, which allows you to specify a set value to replace pixels with certain values. Look at a histogram of the difference images (see next point). You should see a roughly Gaussian shape (which should be centered at zero), with very long tails out to high and low values. If necessary (and it may not be) you can trim these tails by removing the highest or lowest pixels. For example

  ```
  imreplace flat_diff.fits  0  upper=-2000
  ```

  will look at the image flat_diff.fits and find all pixels with values *less* than –2000 (the "upper" parameters specifies the upper bound of the range *to be replaced*), replacing them all with zero. You can cut off the high tail with "lower=2000". Be careful with this; if you cut off high or low values that are due to real Poisson or read-noise variations, you're biasing your answer. I'd recommend calculating gain and read noise without clipping the images like this unless there are

obvious problems. (If you do need to do this, work with a copy of the original image; you can simply make copies of the images you're working on and clip one but not the other).

- Plotting histograms of pixel values can be a very useful way to examine your data, especially for biases, darks, and flats, where all of the pixel values ought to be fairly similar. IRAF will let you plot histograms in a few different ways. One straightforward way is to use the "imexam" command. Display the image first (using *display*), then run *imexam*. Point the cursor to any point in the displayed image, and type "h"; you will then get a histogram plot of the pixel values. The size of the area of the image sampled for the histogram is set by parameters in the *himexam* set (i.e. do "epar himexam" to set this before running imexam). While *imexam* is fine for simple histograms, you can get a bit more control with the *imhistogram* task. If you want to print the histogram (rather than plot it on screen), add "device=stdplot" on the command line after the task name. Using these histograms is important because you are doing these statistics under the assumption that the distribution of counts follows a particular distribution (Gaussian or Poisson), and you should check that that assumption is justified.

- If you have a plot (such as a histogram) displayed in IRAF that you want a hardcopy of, you can usually get one by pressing the "=" key for any task that makes plots interactively. This applies to tasks like *imexam* and *implot* that are still active after making a plot, not to things like *surface* or *imhistogram* that exit after making the plot; for those, set device=stdplot (the printer) rather than the default device=stdgraph (the screen). If you make a plot and then quit the task, but wish you had printed a copy of the plot still on the screen, typing "=gcur" from the IRAF prompt will temporarily put your cursor back in the graphics window, from which you can hit "=" to print.

- You can use the *imstat* task to get the basic statistics of an image (such as mean, min, max, standard deviation, etc.) You can also get stats for only a subsection of an image by specifying the image name as image.fits[xmin:xmax,ymin:ymax]. For example, " imstat bias.fits[100:1024,100:1024]" will ignore the first 99 rows and columns when calculating the statistics. This syntax works with any task, not just *imstat*; for example, you could use it to copy only a part of an image into a new, smaller image using *imcopy* and specifying the input image name as above. Or you could use it to get *imhistogram* to plot the histogram of only part of an image. This can be very handy if the edges of one of your images seem different from the middle, and you want to check out one part or the other.

To turn in for this part:

In addition to the usual written solutions for the conventional problems listed at the start of this assignment, for the analysis part I'd like a well-organized, written description of what you find and how you find it, including (where relevant) images or graphs of the data. (In most cases, it's fairly straightforward to get IRAF to print a hardcopy of any plot you make – see comments above.) Your report doesn't have to follow any set format. If you just take *careful* notes while you're working with the data and doing the calculations, including a photocopy of part of your notes as part of your report is OK, but you should still write up something to give it some overall structure. You will also want to include some revealing hardcopy plots from your investigations of the data, and some explanation of how you're calculating or determining the relevant quantities.