UCF Physics: AST 5765/4762: (Advanced) Astronomical Data Analysis

Fall 2019 Homework 8

Due Tuesday 22 October 2019

Work:
Become sufficiently familiar with image corrections to:

1. know what calibration data to acquire given the observing objectives and the sky conditions,
2. create calibration frames appropriate to a given dataset,
3. remove dark current, bias, background, and flat-field variations from an object frame.

Resources:

1. Chapter 6 of Howell (**DUE** before class Tuesday, 22 October 2019)
2. Appendix C of Howell (**DUE** before class Thursday, 24 October 2019)

Hand in:

This assignment continues the correction of the data in `Files/data/hw7`. Be careful to follow the instructions above the questions in HW7.

1. (10 points) Copy the routines and homework files from the previous homework (including any inclusions of prior assignments) into the directory for this assignment. Correct any errors you may have had, making a comment that says "`# FIXED: `" and giving the date. You may refer to the posted solution, but if you do so you must state what you used from it. The only non-comment for this problem should be the prior homework file run as a batch job (i.e., `from hw7_sol import *`). This will run all the homework files as batch files, each calling the prior one as its first thing, back to HW7. Of course, make sure they all still run without errors.

2. (10 points) Write a function called `normmedcomb` that implements the normalized median combination method. It should accept a 3D array of dark-subtracted data containing sky flux (i.e., your dithered object dataset as processed in HW7). It should take a tuple containing the corner coordinates of the normalization region (`((y1, x1), (y2, x2))`) as an optional keyword (the default region should be the whole array). The normalization region is the region from which to calculate the normalization factor in each frame. The routine should return a tuple containing:

   (a) the normalized, median-combined, 2D array,

   (b) a 1D array containing the normalization factors.

   Be sure not to modify the 3D input array in the caller!

3. (10 points) Run the new routine on your dark-subtracted object data from HW7. Use this normalization region: `((225, 225), (-225, -225)) # ((y1, x1), (y2, x2))` This large normalization region avoids obviously-bad areas of the array (and if everyone uses the same one, grading is easier!). Assign this to a variable and use it **throughout the rest of the homeworks**. Make sure there are no stars left in the resulting normalized sky frame. Write the results to `sky_13s_mednorm.fits`. Write as **32-bit** floating-point data. Print the normalization factors and include them in a comment or dummy string in your main homework file.

4. (10 points) Write a function called `skycormednorm` that implements sky subtraction using your normalized sky frame (i.e., denormalize and apply it). The function should accept two 2D data arrays, the object frame and the normalized sky frame (both already dark-subtracted). You might have created the sky frame from only a subset of the frames, so rather than passing the normalization factor (which you might not have), calculate it again using the same normalization region tuple as before (pass it into the function). Return the resulting sky-subtracted frame. Be careful not to modify any data in the caller.

5. (10 points) Apply `skycormednorm` in a loop to all the object data in place (i.e., modify the 3D object array). Write the last resulting frame (32-bit floating point) to a file named `stars_13s_9_nosky.fits`. Print the value of pixel `[217, 184]` in this frame.

6. Consider a mid-infrared observation of a star. You have an object-and-sky image pair in which the sky background is 10^5 counts/s/px and the star's total signal is 10^6 counts/s. Under the point-spread function, 30% of the star's signal falls in the peak pixel. The integration time is 0.1 seconds. The read noise is 10^2 electrons/px. The gain is 10 electrons per ADU. You consider summing sets of 10, 100, 1000, 10,000, 100,000, and 1,000,000 images.

   (a) (4 points) Write formulae (variables, no numbers) for the electrons per pixel from all sources in each image. Calculate it for the peak pixel in both images.

   (b) (4 points) Write formulae for the noise per pixel and from all sources in each image. Calculate it for the peak pixel in both images.

   (c) (4 points) Write the (two-term) conceptual image background-subtraction formula. Then, substitute the signal-per-pixel formulae from the prior steps, without simplifying, to make obvious what is involved. Then, simplify. Calculate it for the peak pixel.

   (d) (4 points) Looking at the middle equation you wrote in the last item, write a noise-per-pixel formula for the final image. Calculate it for the peak pixel.

   (e) (4 points) Write the per-pixel signal-to-noise formula. Calculate it for the peak pixel.

   (f) (5 points) Write the per-pixel signal-to-noise formula for multiple images. Calculate it for the peak pixel for each contemplated set of images. Plot it *vs.* number of images in the set on a log-log plot. Include horizontal lines at S/N = 3 and 10. How many frames are required to reach each of these limits? You may read this from the plot. Print these values.

7. (10 points) Include a copy of your class log file in your handin. Print the Git log for your main homework file.