Astro465 - Optical Part

September 17, 2024

Contents

1 CCD Characterization

1.1 Introduction

In this lab experiment we will develop a basic reduction procedure to apply standard calibrations to a set of CCD observations, and then use this data to derive some of the detector characteristics, including gain and readnoise levels.

As discussed in the lecture overview over CCD (Charge Coupled Device) detectors, observational data needs to undergo some pre-processing to make it suitable for scientific analysis. At the very least, this includes removal of the bias signature, and correcting for pixel sensitivity variations using a flatfield exposure.

1.2 Preparation

Read Chapter 3 and Sections 4.1-4.5 in Howell's "Handbook of CCD Astronomy" to familiarize yourself with the basic functioning of CCD detectors and the way the different detector systematics can be addressed with the various calibration products such as bias, dark, and flat-field exposures.

All data processing will be handled in python, with FITS images being the data format of choice for the optical astronomer. Read the documentation for the astropy.io.fits[1] package that provides all functionality to read, manipulate, and write image data.

1.2.1 Optical specifications

Aperture size 16 inches, 0.4m

Focal ratio f/10

Focal length $\sim$ 4m

Camera Apogee Alta F8300 (see below)

Filter wheel QHY, 7 positions, 36mm diameter

Available bandpass filters Baader Planetarium Sloan u/g/r/i/z/y, 6.5nm O[III], 6.5nm Hα (u not currently installed)

1.2.2 Technical Specs of our CCD

Array Size (pixels) 3326 x 2504 (8.3 Megapixel)

Pixel Size 5.4 x 5.4 μm

Sensor Size 18 x 13.5 mm (243 mm^2), 22.5 mm diagonal

[1] https://docs.astropy.org/en/stable/io/fits/

Pixel Well Depth (typical) 40,000 e^-

Dark Current 0.016 e^-/pixel/sec

Linearity Better than 99%

1.3 Data acquisition

For this exercise we will need two sets of images:

- Several (10-25) bias frames, all using a constant CCD temperature.
- A sequence of flat-field pairs with varying exposure times, all using the same bandpass filter in the beam. Experience shows that g, r, and i filters are good choices, allowing to accumulate sufficient signal without requiring too much integration times.
- Individual flat-field images in the other broad-band filters to distinguish between detector and filter artifacts. Aim for a middle intensity range (10-40K counts).

For the flatfield images choose a range of exposure times to cover the full dynamic range, from very few counts above bias level, to a peak intensity level below saturation level (aim for 40-50K counts in the longest integrations). Also note that due to the workings of the diaphragm shutter, the minimum integration time should be at least 0.1 seconds to allow the shutter to fully open and close.

1.4 Analysis

1. Based on the technical specs of the telescope and CCD, derive the plate-scale (in arcseconds per pixel) and field-of-view.
   Hint: What is the angle of a triangle, with opposite the size of a pixel, and an adjacent the length of the focal length of the telescope? Watch your units (and make sure to convert radians to/from degrees where needed).
2. Open each image in your jupyter notebook. The package `astropy.io.fits` has the appropriate `open(filename)` function (see the help-pages on the astropy website for more pointers on how to get to the actual data part). To avoid issues with the raw data, convert all INTEGER values to "regular" numbers using the `...data.astype(float)` conversion.
3. From your set of bias images, derive the readnoise of the detector in counts/ADUs, $\sigma_{readnoise,ADU}$. If you have a single frame, you can derive the readnoise as the standard deviation across all pixels. If you have multiple images, you can derive the readnoise for each pixel as the standard deviation across all measurements of this pixel.
4. For each pair of flat-fields (i.e. two flat-field images with identical filter and exposure time), calculate the average intensity level, as well as the noise of the difference image (i.e. standard deviation of flat1-flat2).
   Note: explain why using a difference image here rather than the image data directly makes sense.
5. Compute the gain of your detector (for each pair of flat-fields with your different exposure times) using this equation:

$$g = \frac{(\overline{F_1 + F_2}) - (\overline{B_1 + B_2})}{(\sigma^2_{\Delta_F} - \sigma^2_{\Delta_B})} \quad [e^-/ADU]$$

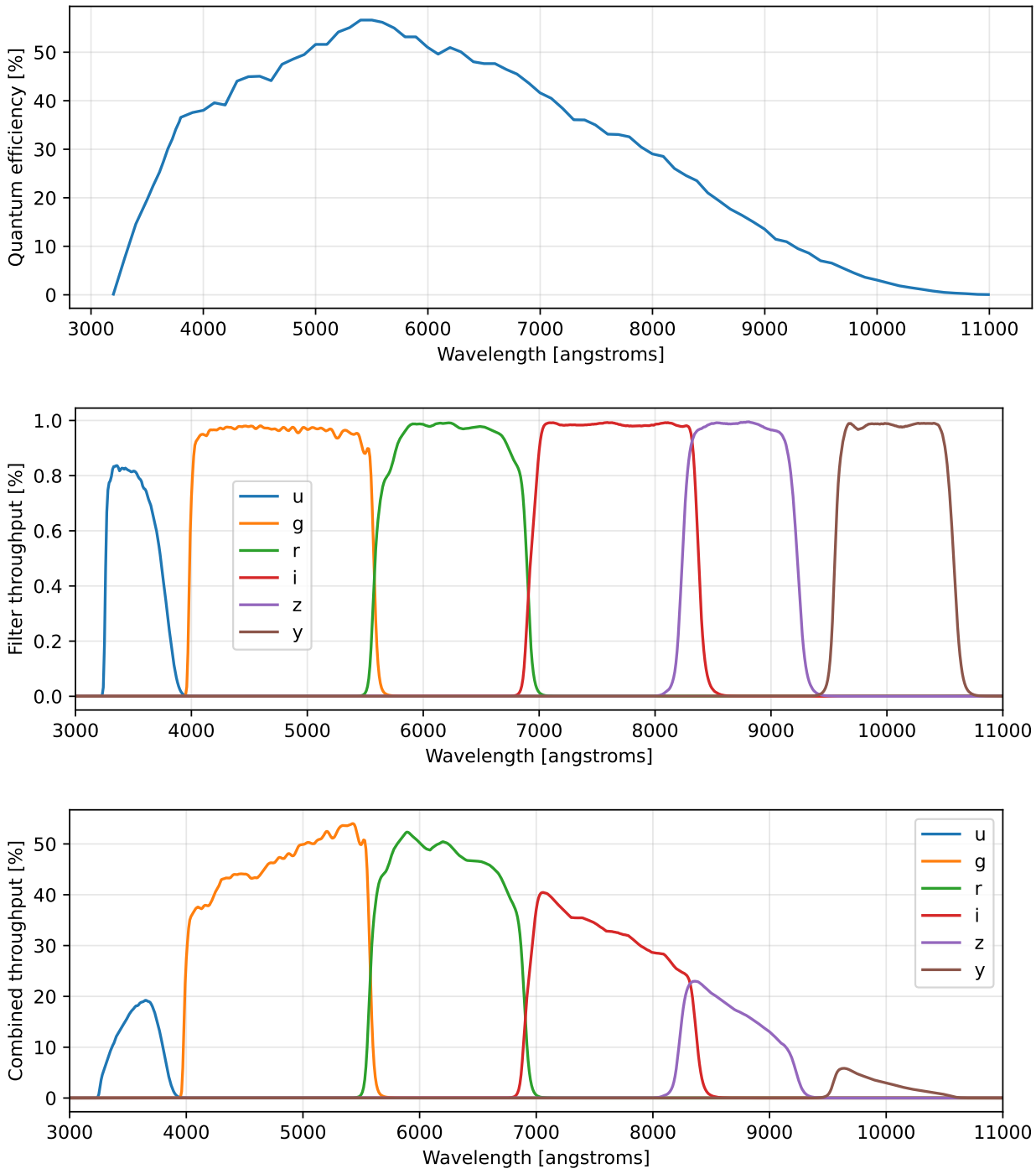

Figure 1.1: Quantum efficiency (top panel) and throughput curves for available broad-band filters (middle panel), and combined filter and quantum efficiency curves (bottom panel). Not included are wavelength-dependent mirror reflectivity for primary and secondary mirror, or the transparency curve of the primary field corrector lens.

where F_1 and F_2 are your two flat-fields, $\overline{X}$ is the average intensity of image X, and σ_X is the standard deviation measured across all pixels in image X.

Are the gain values you get consistent across all exposure times and/or illumination levels?

6. plot the observed standard deviation as a function of the mean image intensity, in a log-log plot. The theoretical curve for the data follows the shape:

$$\sigma(I[e^-]) = \sqrt{I + \sigma^2_{readnoise[e^-]}}$$

Since CCDs do not directly return the number of detected electrons, but rather the number of ADUs, where $1ADU = N_{e^-}/gain$, this equation can be rewritten as

$$\sigma(I[ADU]) = gain^{-1} \times \sqrt{I_{ADU} \times gain + \sigma^2_{readnoise,ADU}}$$

Using this equation and your earlier measurements for the readnoise, derive the gain of your CCD detector.

7. From your flatfield data derive the number of hot and cold pixels (defined as deviating from the overall sensitivity by more than 20%).

There is a quick and easy way of counting pixels using numpy and masks.
Example: `numpy.sum(image > 20)` counts all occurences in the image (internally handled as a 2-d array of numbers) with values >20.

8. Describe the overall flat-field cosmetics. Do you notice any peculiar features, or regions of the detector with depressed sensitivity (you may want to avoid those when taking actual on-sky data)?

9. Compare your flat-fields from the different bandpasses. What is the main source of contamination (bandpass filters, or detector)?

10. In preparation for the upcoming observing labs, develop a routine that combines a number of biases and flat-fields, generates a normalized flat-field, and applies it to an arbitrary image frame.

2 Photometry of star clusters

2.1 Introduction

Star clusters are important objects in astrophysics as they provide some insight into the structure and formation of our Milky Way, as well as into the lifes of stars. Star clusters include a range of objects, including massive, centrally concentrated globular clusters; intermediate mass open star cluster, and low mass, generally loosely concentrated star associations. Important catalogs of bright star clusters include the Messier catalog (including sources such as M13 and M15), the new General Catalog (NGC; e.g. NGC 891), and the extended Index Catalog (IC).

Stellar associations are loose conglomerates of stars, with concentrations often just slightly above their surroundings. Due to their low star numbers and thus low total stellar masses, the gravitational forces binding these stars together are weak, limiting the typical lifetimes. For this reason, associations are often made of predominantly young, short-lived stars such as O- and B-type stars.

Open star clusters include 10s to 100s of stars, with a total stellar mass of the order of 10^2 to 10^3 solar masses. Our Milky Way contains roughly 1000 open clusters, but dust effects close to the Milky Way limits their observability, and the true number is likely much greater. Typical sizes of open star clusters range from 1-10 pc. Combined with their larger masses star clusters are typically a few hundred Myrs old, with notable exceptions at much younger or older ages. Prominent examples are the pleiades and hyades in the constellation Taurus, or the double star cluster h and χ Persei.

A typical globular star clusters (GC) contains several 10^5 stars, with total stellar masses of 10^3-10^6 solar masses. It also contains a bright (often unresolved, at least from the ground) nucleus of 0.3-10 pc, with stellar densities rapidly falling as the distance from the core increases. Typical diameters of GCs range from 20-150 pc, with absolute luminosities of -2 to -10, making it possible to study star clusters out to large distances. Our Milky Way contains 150-200 GCs, most of which are very old and metal poor, making them ideal laboratories to study the early formation of our Milky Way.

2.1.1 Hertzsprung-Russell and Color-Magnitude diagrams

The Hertzsprung-Russell diagram plot the absolute magnitude of stars as a function of their spectral type, or effective temperature. It is named after Ejnar Hertzsprung and Henry Norris Russell that around 1910 studied the connection between spectral type and luminosity. Since stars come with a range of radii, luminosities, and surface temperatures, one might expect that stars are more or less randomly scattered throughout the HRD. In reality, most stars arrange themselves in a number of well confined areas. The **main sequence** runs roughly diagonal from luminous white-blue stars down to faint, red stars. Stars above the main sequence are named "giants", as they show higher luminosities for a given temperature, whereas stars on the main sequence are referred to as dwarfs.

However, deriving absolute luminosities and surface temperatures are somewhat cumbersome (absolute luminosity, for example, requires knowledge of a distance), so in practice we can replace these physical quantities with observables in the shape of apparent (or absolute, if distance is known) magnitudes and color indices. Color indices in particular have the benefit of being photometric pa-

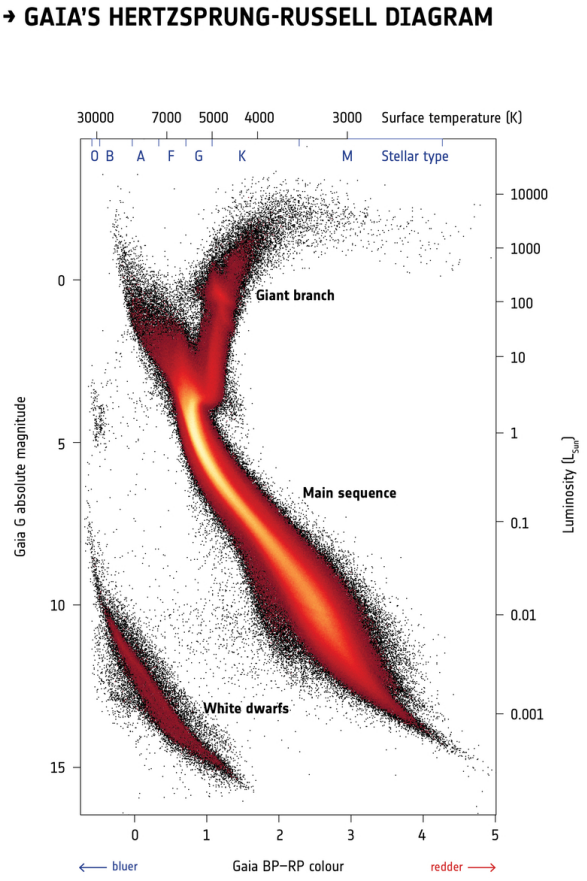

Figure 2.1: An observational Hertzsprung–Russell diagram obtained by the GAIA mission. Source: https://sci.esa.int/web/gaia/-/60198-gaia-hertzsprung-russell-diagram

rameters that do not require stellar spectroscopy, and thus can be obtained with modest observing time even for faint stars.

2.2 Relevant terminology

2.2.1 Instrumental, apparent, and absolute magnitudes, distance modulus

Instrumental magnitude m_{inst} is a measure for the brightness of a star, measured as it has been observed with the detector before any other calibrations are applied. It can be obtained by

$$m_{inst} = -2.5 \times \log_{10}(I_\star)$$

With intensities typically measured in counts or electrons, instrumental magnitudes are therefore typically negative, ranging from -5 to -30 mag.

Apparent magnitudes m_{app} are a measure for the observed brightness of a star. It can be derived from the observed flux as

$$m_{app} = -2.5 \times \log_{10}(f_\star) + const$$

Comparing instrumental and apparent magnitude we can see they are nearly identical, with just an additive offset (or a multiplicative scale factor when considering fluxes rather than magnitudes). This offset is typically referred to as photometric zeropoint or photometric calibration constant. While obtaining a absolutely calibrated detector would be ideal as it would directly yield apparent magnitudes, in practice we can compare the observed instrumental magnitudes with cataloged apparent magnitudes for a number of photometric calibration stars to compute the desired photometric zeropoint.

As the observed flux $f(r)$ of an object decreases as $1/r^2$, the apparent magnitude of an object also depends on its distance from us:

$$f(r) = F_0/r^2$$
$$m = -2.5\log_{10}(F_{0,\star}) - 5\log_{10}(r) + const$$

Absolute magnitudes M are defined as the apparent magnitude of a source at a standard distance of 10 pc.

Distance Modulus is the difference between apparent and absolute magnitude of an object, so $(M_{app} - M$. So if we know the absolute magnitude of an object (e.g. from its location in the HRD), and its observed apparent magnitude, we can calculate the distance to this source:

$$(m - M) = 5\log_{10}\left(\frac{r}{10pc}\right)$$
$$r = 10pc \times 10^{0.2(m-M)} = 10^{0.2(m-M+5)}pc$$

2.2.2 Preparation

- What spectral classes are there and how do they correlate with the observed colors of stars?
- What is the interpretation if points in the HRD are more densely populated with sources than other areas.
- Locate our sun in the HRD.
- What are supergiants and white dwarfs and where would they be located in the HRD?

2.2.3 Target selection

From the catalog of Kharchenko (2013) select one or two clusters for observations. Suitable targets should be suitably located in the sky for observing during the semester, have a large enough number of stars to allow for analysis, and not be too extended so we can fit a good fraction of cluster members into the field-of-view.

Best way to access the catalog is via the CDS Vizier service: https://vizier.cds.unistra.fr/viz-bin/VizieR. In the "Find catalog" field, search by either author and year (Kharchenko 2013) or alternatively (and if known), catalog identifier (J/A+A/558/A53/catalog). Alternatively you can search the catalog of Milky Way Globular Clusters by Harris (1997; catalog ID VII/202). The latter

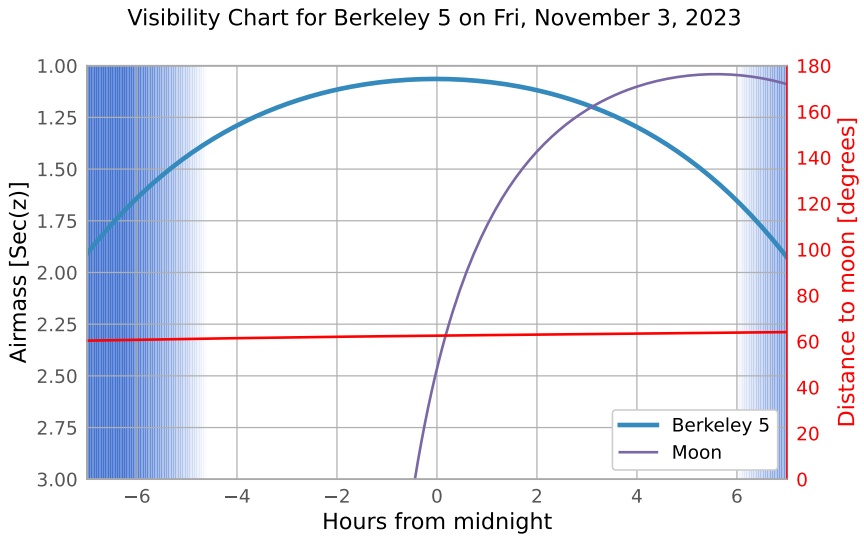

Figure 2.2: Example airmass plot for star cluster "Berkeley 5" for the night of November 3rd, 2023.

contains targets with significantly higher star densities, which is helpful for fitting CMDs but can make data analysis somewhat more challenging.

For each target generate a airmass plot, showing the observability of each target during the observing night, and its distance from the moon (see Fig. 2.2 for an example)[1].

Also prepare a finder chart for your target, especially if the concentration of member stars is low and/or the target is located close to the galactic plane where contamination from background star makes identifying the target less obvious.

2.2.4 Observations

Required observations are

1. At least one exposure of the target in two different bandpasses. Considerations for the filter choice are the wavelength baseline covered, as well as system throughput in each band.
2. All required calibration products necessary to reduce and calibrate the science frames. This includes both bias and flat-fields.
3. Associated calibration products for photometric calibration. For most targets there likely will exist calibrated reference data (e.g. from PanSTARRS), allowing to perform a photometric calibration without having to observe separate standard star fields.

2.3 Data analysis – Lab 2a

1. Optional, but highly recommended: Organize your files. One common approach is to sort each data type into its own sub-directory, sorted by filters. An example might look like this:
   - biases
     - bias_1.fits

[1] https://github.com/rkotulla/astro465/tree/main/visibility_tool

  - bias_2.fits
- flats_g
  - flat_g_1.fits
  - flat_g_2.fits
- flats_i
  - flat_i_1.fits
  - flat_i_2.fits
- science
  - ngc1234_g.fits
  - ngc1234_i.fits

2. Reduce your science images using your calibration products

   a) You can open each FITS file using the `astropy.io.fits.open` function. This will give you a HDUList item (don't worry about the details). To get to the image data, using the `hdulist[0].data` property. Also, to avoid trouble down the line, convert the `.data` from the native 16-bit integer (only positive whole numbers) to a "normal" number using the `.astype(float)` addition. In a single line, you could get the image data via `astropy.io.fits.open("your_filename.fits)[0].data.astype(float)`.

   b) Combine all bias frames into a master bias (both median or mean should work)

   Note: Based on page 78 in the "CCD Handbook", an average of 10 or more bias images is needed to sample well CCD's CD variations. This is why we need to calculate a mean image of all our bias frames. This, the master bias image, will be calculated for each pixel.

   c) Subtract your master bias from each individual flat-field. Normalize each bias-subtracted flatfield to have an average intensity of 1. Then combine all flat-fields taken in the same filter into a master flatfield (you need to have one master flat for each filter you are using; do not mix & match filters!).

   Note: Based on pages 79–80 in the "CCD Handbook", an average of 5 or more flat field images is needed to sample pixel-to-pixel variations in the CCD response as well as any nonuniform illumination of the detector itself. One way of doing this is to average all flat field images first, subtract the master bias image, and then normalize the resultant image so it has an average intensity of 1. Another way is to subtract your master bias image from each flat field image, normalize each bias-subtracted flat field image, and then average all normalized (bias-subtracted) flats. Both methods should give you the same result. This will be your master flat field image, calculate for each filter.

   d) To keep things simple, please use a single science image, subtract your master bias, then divide by your normalized master flat *for the correct filter*.

   Note: This is shown as an equation on page 82 of the "CCD Handbook".

3. For your single science image, extract photometry for all sources of interest in your images. For the time being ignore sources that are blended or too contaminated by nearly sources to yield accurate and reliable photometry. To do this we will use a python package named `photutils` that comes with all functionality we need. Especially take a look at the sections "background

estimation" (needed to estimate photometry uncertanties), "Source Detection" and "Aperture Photometry".

a) As the package for detecting sources requires a background-subtracted image, start by estimating the sky background using astropy.stats package with sigma-clipping. We need to know the sky background so we can select sources that have fluxes significantly higher than the sky background.
An example is shown here: https://photutils.readthedocs.io/en/stable/background.html .

Note: Sky background in the I-band is higher than in other bands.

b) Source detection: follow examples from https://photutils.readthedocs.io/en/stable/detection.html . The input is a background-subtracted image. We want to detect sources that are 5x or 10x above the image noise. The package will provide a list of sources with source positions and fluxes given in the number of sigmas. Make a scatter plot of source fluxes.

c) Run aperture photometry to estimate source fluxes: examples are at
https://photutils.readthedocs.io/en/stable/aperture.html

First provide source positions in the format that the aperture tool requires. Then define an aperture circular with radius equal 4 pixels. Then use the aperture-photometry tool. This will provide source fluxes in counts but over the entire aperture. Then calculate flux uncertainties.

Source flux in magnitudes is equal $-2.5log_{10}(aperture\_sum)$.

d) Derive uncertainties for source fluxes estimated using the photometry package. This is done by applying the CCD equation. In your report, discuss the largest contributions to the total uncertainty, and describe potential ways to decrease uncertainties and improve your data quality.

Note: To estimate uncertainties you can use this equation:

$$N = \sqrt{S_* + n_{pix}(S_{sky} + t \times d + R^2)} \tag{2.1}$$

Here we can ignore the dark current, read-noise (R) is the value from the Lab 4 (i.e. 7 counts = 7 e-). As 1 ADU = gain × N[e-], with gain = 1, counts and electrons are the same, and we can plug in counts into the equation directly. The rest of the parameters are delivered by the Python photometry routine for source detection: S_* is flux for the source, S_{sky} is the ky background, n_{pix} is the size of the aperture expressed in number of pixels (provided by the source catalog).

As a quick check, plot plot S/N for all detected sources, this is flux over aperture/N. You should get a range from get a few to few ×1000.

e) Optional: Based on the position of sources in your image, perform an astrometric calibration of your data. You may find the `astrometry` python package useful in this endeavor.

2.3.1 Some python snippets to get you started

Packages to consider importing

```
import astropy.io.fits as pyfits # open FITS files
import numpy                     # do calculations on image data
```

```
import glob                    # get filenames in a directory
```

Process a bunch of files in a directory all at once

```
# get all FITS files in the biases directory
bias_filelist = glob.glob("biases/*.fits")

# initialize a variable to hold each of the bias images
biases = []

# repeat for each bias image
for fn in bias_filelist:
    # open image
    hdu = pyfits.open(fn)
    # read image, and convert to floating point numbers
    data = hdu[0].data.astype(float)
    # add to the list of biases for processing
    biases.append(data)

# convert list of images to a image stack for processing in numpy
biases = numpy.array(biases)

# apply median combination to each pixel
masterbias = numpy.median(biases, axis=0)
```

Estimate the sky level in an image

```
# estimate background
from astropy.stats import sigma_clipped_stats
mean, median, std = sigma_clipped_stats(image_data, sigma=3.0)
```

Find sources in an image

```
# from https://photutils.readthedocs.io/en/stable/detection.html
from photutils.detection import DAOStarFinder
daofind = DAOStarFinder(fwhm=3.0, threshold=10.*std)
sources = daofind(skysubtracted_image)
for col in sources.colnames:
    if col not in ('id', 'npix'):
        sources[col].info.format = '%.2f' # for consistent table output
sources.pprint(max_width=76)
```

Derive aperture photometry for a bunch of sources

```
from photutils.aperture import CircularAperture, aperture_photometry

# define where our apertures are
positions = [ (s['xcentroid'], s['ycentroid']) for s in sources ]
# this converts the array of source positions and properties to a list that
```

```
# contains only the source positions in the format we need for the aperture photometry

# define all source apertures, using an fixed aperture radius of 4.0 pixels
apertures = CircularAperture(positions, r=4.0)

# now extract the actual photometry
photometry_table = aperture_photometry(skysubtracted_image, apertures)
photometry_table['aperture_sum'].info.format = '%.8g' # for consistent table output
print(photometry_table)

# also add the new data back into the original catalog
sources['aperture_sum'] = photometry_table['aperture_sum']
sources['mag'] = -2.5*numpy.log10(sources['aperture_sum'])
```

Compute uncertainties for a number of sources

```
# calculate uncertainty for each source

R = 8.                  # define a readnoise
gain = 1.0              # define a gain
npix = numpy.pi * 4**2  # aperture size in pixels
S_source = sources['aperture_sum']
noise_b = numpy.sqrt( gain*S_source + npix*(gain*skylevel + R**2) ) / gain
```

2.4 Data analysis – Lab 2b

1. Continue your analysis with the reduced data from Lab 2a. We need reduced data for two bands. Our goal for this part of the Lab is to generate a color-magnitude diagram: for example, plot the color (B-V) on the x-axis, and the V-band intensity (in magnitudes) on the y-axis. We will then overplot the isochrone lines to estimate the cluster age and metallicity. You can do this by skipping the optional part of absolute photometric calibration.

2. **OPTIONAL - Absolute photometric calibration.** Compare photometry for some brighter sources in your data with published photometry and derive a photometric zeropoint. One common approach to this is to compare observed magnitudes of several stars in your images with known photometry from a well-calibrated survey such as SDSS, LegacySurvey, or PanSTARRS. All surveys come with convenient web interfaces[2][3] that allow to identify stars and obtain their calibrated magnitudes, thus allowing to compute a reliable photometric zeropoint.

   Do you notice any significant trends in your zeropoints with either magnitude and/or color index of the catalogued sources?
   If you images do not have a astrometric calibration, the best way of doing this is to write your source catalog as a VOTable (use the astropy table `.write` function, specifying the format as votable), then overlay this catalog over your image in ds9. That makes it easier to compare your instrumental magnitudes and the calibrated photometry for the same stars in a separate spreadsheet.

[2] https://www.legacysurvey.org/viewer
[3] https://outerspace.stsci.edu/display/PANSTARRS/PS1+Image+Cutout+Service

3. We have two source catalogs obtained for two bands, we now need to match sources by comparing coordinates.

   Here the `astropy.coordinates` has all features you need – Read the section titled "Matching Catalogs". To yield better results, you can take out any potential offsets between the star positions in each frame first by adding/subtracting the offsets from the respective coordinates. This will allow you to select a smaller matching radius and thus a cleaner combined sample, with the added benefit of providing more matches.

4. Generate a color-magnitude diagram from your data. If you have doen the absolute photometric calibration, you can apply the zeropoints you derived to your instrumental magnitudes to get the calibrated apparent magnitudes you need for your plot.

5. Overplot a theoretical isochrone, adjusting age, metallicity, and offsets in both color and apparent magnitude. Determine the best-fit and extreme values for each of the parameters. This will provide you with an age, metallicity, distance, and foreground reddening value for your cluster.
   In your report describe how you tuned your parameters for the best fit: Did you consider all stars equal, or attribute more importance to some stars over others? Did you run into any unforeseen issues or challenges in determining your best solution?

6. Only after you have completed the previous step, compare your results with published values. Discuss any discrepancies.

2.4.1 Some more python snippets for your journey

Combine two catalogs

```
import astropy.coordinates as astcoo
import astropy.units as u

# To match catalogs we can exploit some astropy functionality
# If you have frames with valid WCS (World coordinate solution), convert x/y to
# RA,DEC -- this is the proper way of doing
# if you have NO wcs and so have to work with pixel coordinates we have to
# "cheat" a bit to make this work; to do so, we convert source positions
# in x/y to sky positions by assuming an origin position of 0,0, and a pixelscale
pixelscale = 0.27 * u.arcsec
src_b = astcoo.SkyCoord(sources['xcentroid']*pixelscale, sources['ycentroid']*pixelscale)
src_i = astcoo.SkyCoord(sources_i['xcentroid']*pixelscale,
    sources_i['ycentroid']*pixelscale)

# match both catalogs
idx_i, d2d, d3d = src_b.match_to_catalog_sky(src_i)
# idx_i: position in the src_i catalog for the closest
#       match to a source in the src_b-catalog
# d2d:  projected distance between sources in b and i
# d3d:  3-d distance (ignore this, since we do not have distances)

# re-organize the i-band catalog, so the N-th source in the new catalog
# is the source closest to the N-th source in the original b-band catalog
matched_catalog_i = sources_i[idx_i]

# Now add data from the matched (i-band) catalog into our catalog
# that already holds data for the b-band
```

```
sources['mag_b'] = sources['mag']
sources['mag_i'] = matched_catalog_i['mag']
```

Loading and learning about isochrones

You will have to download the isochrones.zip file from Canvas to make these steps work

```
# import the Isochrone class from the downloaded isochrone.py file
from isochrones import Isochrone

# load the isochrone file -- there are two options
# isochrones_ubvrijhk.dat is you are using archival data from WIYN, in the B/V/R/I filters
# isochrones.dat if you are using roof-top data in u/g/r/i/z filters
iso = Isochrone("isochrones_ubvrijhk.dat")

# you can query the class to what ages and metallicities are available:
# for log(ages)
iso.get_list_of_ages()

# for different magnitude bandpasses
iso.get_list_of_magnitudes()

# and for metalliticities
iso.get_list_of_metallicities()

# get data for a single isochrone with a given log(age) and metallicity
iso1 = iso.select_isochrone(9.3, -1.0)

# get a list of available column names
iso1.columns
```

Plotting isochrones

```
# for a very simple plot of a color-magnitude diagram
fig,ax = plt.subplots()
ax.scatter(iso1['Bmag']-iso1['Imag'], iso1['Imag'])
```

Alternatively, plot the isochrone and your data in the same plot for the visual fitting:

```
fig,ax = plt.subplots()

# plot CMD for our data, using B-I vs I
ax.scatter(final_catalog['mag_b']-final_catalog['mag_i'], final_catalog['mag_i'], s=0.5)

# select a isochrone for a chosen log(age) and metallicity
iso1 = iso.select_isochrone(9.85, -0.0)
# some data points are not useful, so we can omit that
bad_iso = iso1['Imag'] > 30
iso1 = iso1[~bad_iso]

# get B-I and I magnitudes from this isochrone to tidy up the plot command
# the numpy.array(XXX) call is to work around some python/pandas complaints
```

```
iso_bi = numpy.array(iso1['Bmag']-iso1['Imag'])
iso_i = numpy.array(iso1['Imag'])
# note: unless you've done your photometric calibration you'll likely need to tweak
# the offsets in X (i.e. B-I) and y (i.e. I mag) to match the isochrone to your data
ax.plot(iso_bi-0.02, iso_i-16.05, color='red')

# zoom into the relevant area for a nicer looking plot
ax.set_ylim((-9,-17))
ax.set_xlim((0.5,3))
```

2.5 Useful pointers to get started

Much of the functionality needed for this lab were derived as part of the CCD lab (basic data reduction), or are included in the PHOTUTILS[4] python package. The latter in particular includes all required functionality to detect sources in astronomical images, as well as derive aperture photometry. Matching the catalogs from the two bandpasses can be accomplished in a number of different ways, depending on the quality of image alignment between both frames, but a good, reliable start can be found as part of the ASTROPY.COORDINATES package[5].

While absolute astrometric calibration would be great and can be accomplished e.g. via astrometry.net (also available as a python package), it can be a challenge at first and thus is less critical for this experiment (although it certainly makes some of the other steps easier). One important calibration step, however, is the absolute photometric calibration. One common approach to this is to compare observed magnitudes of several stars in your images with known photometry from a well-calibrated survey such as SDSS, LegacySurvey, or PanSTARRS. All surveys come with convenient web interfaces[6][7] that allow to identify stars and obtain their calibrated magnitudes, thus allowing to compute a reliable photometric zeropoint with limited effort.

2.6 What to include in your lab report:

Please include the following sections:

Observations Please include and explain the observational setup (what different frames have been observed?

Data Analysis Please include details images for each analysis step, show the final calibrate images for two different bands, and include your Python code. Provide a plot of all detected sources in two bands (RA, Dec . Provide a plot of the color-magnitude diagram with isochrones. Provide an estimate of the cluster age and metallicity and compare your results with published ones.

Conclusions Discuss any differences and sources of uncertainties.

[4] https://photutils.readthedocs.io/en/stable/index.html
[5] https://docs.astropy.org/en/stable/coordinates/matchsep.html#matching-catalogs
[6] https://www.legacysurvey.org/viewer
[7] https://outerspace.stsci.edu/display/PANSTARRS/PS1+Image+Cutout+Service