Draft version December 15, 2022
Typeset using LaTeX **twocolumn** style in AASTeX631

Lab Report n°1: 47 Tucanae Globular Cluster Analysis

Berretti Michele and Fontana Aleksandar

Abstract

We present a new study of the 47 Tucanae Globular Cluster imaged during Hubble Space Telescope (HST) Cycle 17 for 121 orbits using the Wide Field Channel of the Advanced Camera for Surveys. During this analysis we tested different approaches in order to realise a complete catalog of the stars in 47 Tucanae Globular Cluster and the superimposed Small Magellanic Cloud. The main goal of the lab experience was to try different techniques for photometry and implement novel software to a well-established field such as astrometry.

Keywords: 47 Tucanae Globular Cluster — Small Magellanic Cloud — Aperture Photometry — Point-Spread Function Photometry — TrackPy — Hubble Space Telescope

1. INTRODUCTION

The globular clusters spread throughout the Milky-Way have proven to be irreplaceable objects for astronomists over the past century. Indeed, they provide essential informations on cosmology and galaxy formation, the evolution of low-mass stars, stellar dynamics and much more. Therefore the improvement in photometric studies with the advent of highly sensitive and photometrically accurate two-dimensionals detectors, such as Charge-Coupled Devices (CCDs), are a dramatic improvement in the scope and quality of such data Vandenberg & Bell (1985). Indeed, the introduction of CCDs in the late 1970s provided high quantum efficiency and low read-noise, but on the other hand they introduced new constraints such as that the dynamic range is not infinite Heasley (1999).

During the past decades many catalogs were produced in order to keep lists of up-to-date and readily available parameters of globular clusters by many authors (e.g. Shapley (1930), Sawyer Hogg (1959), Kukarkin & Kireeva (1974), Harris & Racine (1979), Vandenberg & Bell (1985), Harris (1996), Kalirai & Richer (2010), Kalirai et al. (2012)) and thanks to the transition to a digital world is now easier than ever to work on rapidly updated catalogs and ensure to a wide community a set of ready-made datasets.

Building on the succes of Kalirai et al. (2012), we will focus on the 47 Tucanae Globular Cluster, one of the most luminous and best studied clusters in the sky.

In Kalirai & Richer (2010) the author observerd 47 Tuc for 121 orbits using the Wide Field Channel of the Advanced Camera for Surveys of the Hubble Space Telescope (HST, Jakobsen & Laurance (1989)) and collected extremely deep images of the globular cluster in two filters: F606W (broad V-band) and F814W (I-band). These two filter were chosen as they provide an accurate characterization of the faintest white dwarves present in 47 Tuc Kalirai & Richer (2010). Basing off white dwarves cooling models for low masses and previous experiences in deep studies (Richer et al. (1995), Hansen et al. (2007)) the luminosity function of such stars is expected to span over 5 magnitudes in the Color-Magnitude Diagram (CMD) and be most efficiently

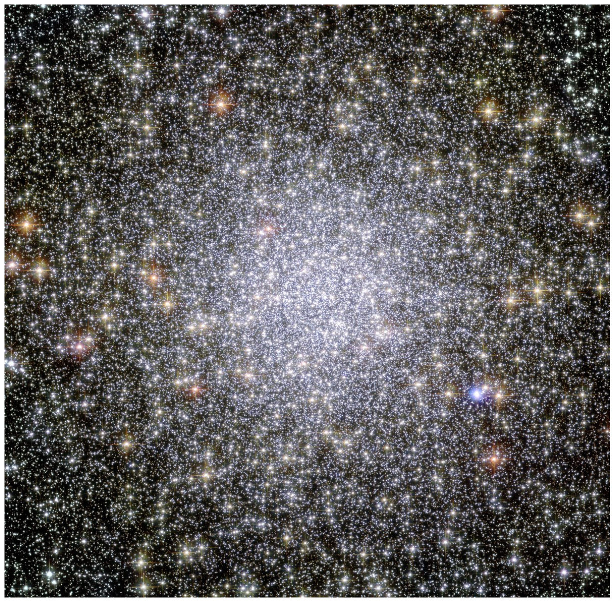
Figure 1. 47 Tucanae Globular Cluster Image. After Omega Centauri, 47 Tucanae is the brightest globular cluster in the night sky.

measured using observations at visible wavelength (Kalirai et al. (2012)).

Using these observations, Kalirai & Richer (2010) were able to construct one of the deepest and most complete CMD of stellar population. As an added bonus, members of the Small Magellanic Cloud (SMC), represent background sources.

We will use the stacked ACS mosaics (FITS Files) released by the team of Kalirai et al. (2012) to experiment with various feature detection algorithms to exploit the didactical value of such dataset and explore the different techniques currently used in the field such as Aperture and PSF photometry (Da Costa (1992), Mighell (1999)) and a novel software suite, TrackPy (Allan et al. (2021)).

2. DATASET

The FITS mosaic in both bands were processed and stacked using Multidrizzle (Brammer et al. (2003)), the mosaics are supersampled at 0.03 arcseconds per pixel and there is a small offset in position between the F606W and F814W stacked mosaics that will require to readjust the positions of the stars once they have been detected to create a 1-to-1 correlation between the catalogs for the two bands. This fix has been necessary to avoid creating a labelling scheme for the stars independent of the positions in order to be able to then compare the same star in the two catalogues despite the shifted position.

3. METHODS

The different methods used to study the FITS files are:

1. Aperture Photometry
2. Point Spread Function (PSF) Photometry with Astropy
3. TrackPy Feature Detection Software Suite

3.1. *Aperture Photometry*

The process of determining the brightness of a star can be surprinsingly complex even with techniques as straightforward as aperture photometry. Indeed, the concept of aperture photometry is quite simple, all one needs to do is sum the brightness of the pixels near the center of a star, subtract the contribution of the nearby sky and then convert the remaining photon number to a stellar magnitude.

In this lab experience we developed a rudimental and very simple python program for aperture photometry that despite its simplicity returns acceptable (but far from good) results.

Here we illustrate the various parts of our code.

```
1      (...)
2      f606 = r"
       hlsp_deep47tuc_hst_acs_47tuc_f606w_v1_drz.
       fits"
3      (...)
4      f606_raw_image = fits.open(f606)
5      f606data = np.array(f606_raw_image[0].data
       [2000:9000,2000:9000])
6      f = find_peaks(f606data, 50, box_size=8,
       border_width=10)
7      catalog606 = f.to_pandas()
8
9        x_peak  y_peak  peak_value
10   0     5681    10      347.172607
11   1      322    11    17860.667969
12   2     1717    11      546.045471
13   3     6345    11      458.307037
14   4     1024    12      198.728195
15   ...    ...     ...     ...
```

This first part of the program (above) finds the stars in the given FITS file. The **find_peaks** function finds local peaks in an image that are above a specified threshold value. Thus, peaks are defined as maxima above the threshold within a local region. The local region size is defined by the box_size parameter and we considered it to be a square box. Obviously, the approximation of the star PSF (Point Spread Function) to a box is wrong and the better way to give the function the shape of the local region would have been using the centroid_func parameter and using a 2D-Gaussian or 2D-Moffat function. If multiple pixels within a local regions have identical intensities, then the coordinates of all such pixels are returned. Otherwise, there will be only only one pixel per local region. Thus, the local region size actually imposes a minimum separation between peaks.

As we can see in line 5 of the above code, we only considered a cropped version of the image. This is in order to remove the borders of the mosaic and not having to worry about handling the borders' pixels. In addition to that we also used the border_width parameter in the find_peaks function to smooth the borders of the cropped image and further reduces boundary effects.

In order to decide the box size we accounted for the FWHM of the PSF of HST, the thickness of the outer shell defining the sky's region and the thickness of the middle shell separating the star's region from the sky's region. The thicknesses of both shells have been chosen equal to the FWHM. This is in order to achieve an excellent practical compromise between concerns about systematic centering errors and diminishing signal-to-noise ratios typically obtained with larger apertures Mighell (1999).

Once we found the peaks we created a catalogue containing all of the stars that have been found by the **find_peaks** function. A sample of it is reported in the above code.

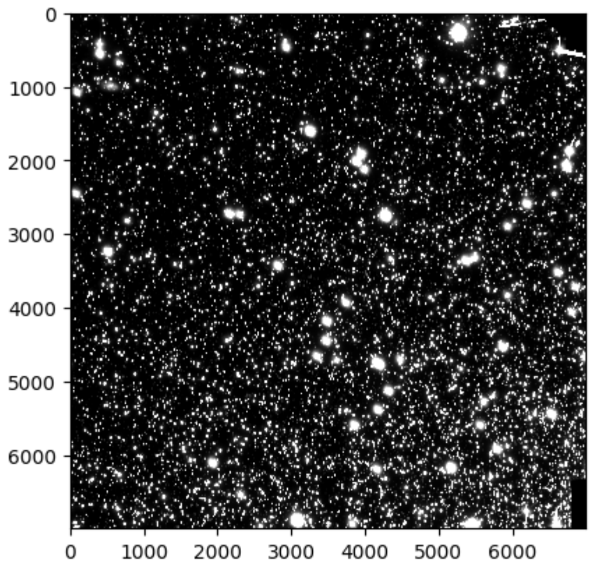

Figure 2. Cropped image considered for aperture photometry. As we can see in the right corners some of the contours of the mosaic are still present. They will be excluded thanks to the border_width parameter in the find_peaks function.

We can now find a rudimental value for the magnitude of the stars found in the image using only the brightness of the centroid and therefore without using any aperture photometry technique.

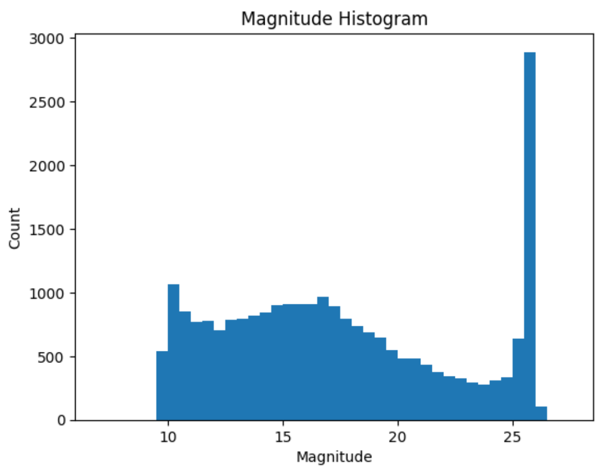

Figure 3. Magnitude Histogram. Magnitude computed using only the brightness value of the stars' centroid.

In order to apply aperture photometry we need to define both the contours of the blobs of pixels representing the image of each star and the contours of the corresponding nearby sky. The below code is a simple function to define such contours.

```python
def neighbors(a, radius, row_number, column_number):
    return [[a[i][j] if  i >= 0 and i < len(a) and j >= 0 and j < len(a[0]) else 0
                for j in range(column_number-1-radius, column_number+radius)]
                    for i in range(row_number-1-radius, row_number+radius)]
```

We can now proceed and define the stars' contours centered on the centroid coordinates and their nearby sky region with the associated brightness values.

```python
1  tib = [] #Totally Integrated Brightness
2  sky = []
3
4  for i in range(catalog606.shape[0]):
5      x0 = catalog606["x_peak"][i]
6      y0 = catalog606["y_peak"][i]
7
8      in_box = neighbors(f606data, 2, y0,x0)
9      mid_box = neighbors(f606data,3,y0,x0)
10     out_box = neighbors(f606data, 4,y0,x0)
11
12     tib.append(np.sum(in_box))
13     sky.append(0.45*np.sum(out_box)-0.65*np.sum(mid_box))
```

We can see in line 13 that we multiplied the sums of the outer and mid shell brightness with 0.45 and 0.65 to emulate what a Gaussian profile would do without having to implement a Gaussian or Moffat instead of a box.

We then compile a catalogue with the newly computed brightness, magnitudes and sky values with the following code:

```python
tib = np.array(tib)
sky = np.array(sky)

mag = 2.5*np.log(tib-sky)


f606cat = pd.DataFrame()

f606cat["x_centroid"] = np.array(catalog606["x_peak"])

f606cat["y_centroid"] = np.array(catalog606["y_peak"])

f606cat["TBI"] = tib

f606cat["mag"] = mag

f606cat["sky"] = sky
```

Finally, we can plot the magnitudes histogram:

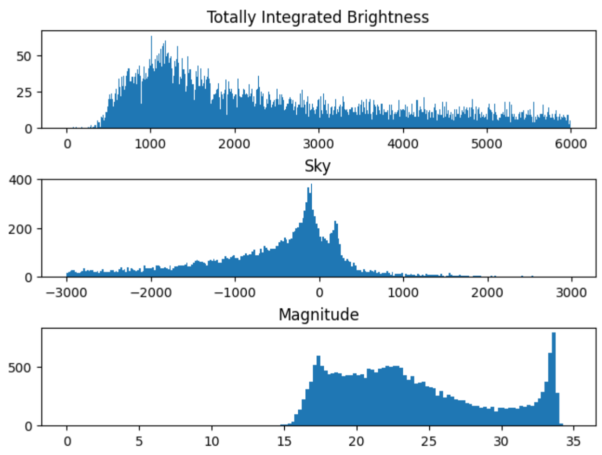

Figure 4. Totally Integrated Brightness, Sky and Magnitude histograms of the 47 Tucanae Globular Cluster obtained with a barebone version of aperture photometry.

3.2. *PSF Photometry with Astropy*

PSF Photometry is based on two main assumptions:

1. All the point-like sources pointed by the telescope can be represented by a Point Spread Function. This function is either assumed to be space invariant or it is knowable and can be modeled.

2. Response of the imaging system is linear to incoming radiation (CCDs have an upper limit on their linearity Heasley (1999)).

Given that the shape of the star on the focal plane is known, its amplitude will scale linearly with the brightness of the star forming the image (Heasley (1999)). Therefore, for any stellar image we must fit the PSF template on the observed image after taking care of the background sky level on top of which the star ”sits”. Having this concepts in mind, the process for PSF photometry is pretty straightforward: we must identify the stars, remove the background noise due to the sky and use the PSF template to fit the stars and compute the brightness of each one of them.

The **photutils** library previously used for aperture photometry provides a modular set of tools to perform PSF photometry for different science cases. We used the **DAOStarFinder** function that implents the **DAOFIND** algorithm introduced in Stetson (1987).

DAOFIND (Stetson (1987)) searches images for local density maxima that have a peak amplitude greater than the threshold value and have a size and shape similar to the defined 2D Gaussian Kernel. The gaussian kernel is defined by the ”FWHM” and ”Ratio” parameters. The threshold value is the absolute image value above which to select sources, in our case we opted for 5σ to be sure to avoid false positive due to pixels of saturated stars (the one belonging to the Small Magellanic Cloud) polluting the background of nearby 47 Tuc’s stars.

The FWHM is the Full Width Medium Heigth of the PSF in units of pixels, we used the one given by Kalirai et al. (2012) but this could have been easily obtained knowing the parameters of the Hubble Space Telescope (Jakobsen & Laurance (1989)) optical system.

The ”Ratio” parameter is the ratio of the minor to major axis standard deviations of the Gaussian 2D kernel. A better kernel for stars detection would be a Moffat 2D kernel, which essentially is an elongated version of a Gaussian kernel, we therefore decided to use a ratio of 0.7 to emulate the slight asimmetry of the Moffat kernel.

Rather than cropping a-priori the image like in the previous section, this time we preferred to use the exclude$_{b}ordersparameterofthe$**DAOStarFinder**$functiontoignore$

Here we provide the code used to create the catalogs for each band with **DAOStarFinder**:

```
print("Locating stars on F606W...")
mean, median, std = sigma_clipped_stats(
    f606data, sigma=3.0)
daofind = DAOStarFinder(fwhm=1.7,ratio=.69,
    threshold=5.*std, exclude_border=True)
sources = daofind(f606data-median)


catalog606 = sources.to_pandas()
catalog606.to_csv("catalog606.csv")



print("Locating stars on F814W...")
mean, median, std = sigma_clipped_stats(
    f814data, sigma=3.0)
daofind = DAOStarFinder(fwhm=1.7,ratio=0.69,
    threshold=5.*std, exclude_border=True)
sources = daofind(f814data-median)


catalog814 = sources.to_pandas()
catalog814.to_csv("catalog814.csv")
```

To realise the Color Magnitude Diagram, we need to link each star in the F606W catalog to the same star in the F814W catalog. In order to do that we first need to fix the shift between the two images we have previously noted. The fix is simple and straightforward, we simply shift back the shifted image so that the two match. The following code shows both the fix for the shift and how the 1-to-1 correlation among stars in the two catalogs is carried out.

```
print("Applying fix to F814W...")
fix_x_814 = np.array(catalog814["xcentroid"])
    +440
fix_y_814 = np.array(catalog814["ycentroid"])
    +13

```

```
print("Preparing for merge...")
catalog814["xcentroid"] = fix_x_814
catalog814["ycentroid"] = fix_y_814

rep=5
tolerance = 5
print(f"Tolerance set to {tolerance}.")

magbg = []
magsm = []
ybg = []
ysm = []
xbg = []
xsm = []

who = 0

total_width = 0

if catalog814.shape[0]>catalog606.shape[0]:
    total_width=catalog606.shape[0]
    bigger = catalog814
    smaller = catalog606
    who = 814
else:
    total_width = catalog814.shape[0]
    bigger = catalog606
    smaller = catalog814
    who = 606
print(f"The bigger array is {who}.")

for i in progressbar(range(smaller.shape[0]), "
    Creating a 1-to-1 match between the two
    catalogs: ", 40):
    #print(f'Sto alla stella {i+1} di {n}', end
    ='\r')
    p_ini = 0
    p_fin = bigger.shape[0]-1
    x_target = smaller["xcentroid"][i]
    y_target = smaller["ycentroid"][i]
    for k in range (rep):
        p_ini, p_fin = range_limiter(bigger,
    p_ini, p_fin, x_target)
    for j in range(p_ini, p_fin):
        if np.abs(bigger["xcentroid"][j] -
    x_target)<tolerance:
            if np.abs(bigger["ycentroid"][j] -
    y_target)<tolerance:
                ybg.append(bigger["ycentroid"][
    j])
                xbg.append(bigger["xcentroid"][
    j])
                magbg.append(bigger["mag"][j])
                xsm.append(smaller["xcentroid
    "][i])
                ysm.append(smaller["ycentroid
    "][i])
                magsm.append(smaller["mag"][i])
```

We can now verify that the star in the two catalogs have been correctly linked to each other checking the correlation plots and finally we realise the Color Magnitude Diagram.

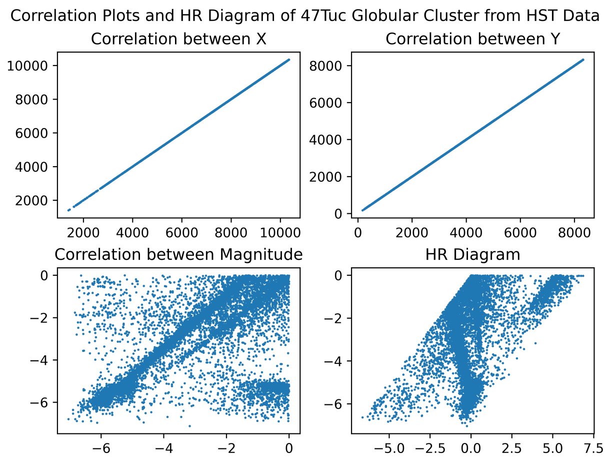

Figure 5. Correlation and CMD, before removing outliers.

To reduce noise we can remove outliers in the plots a-posteriori applying the Isolation Forest Machine Learning Algorithm.

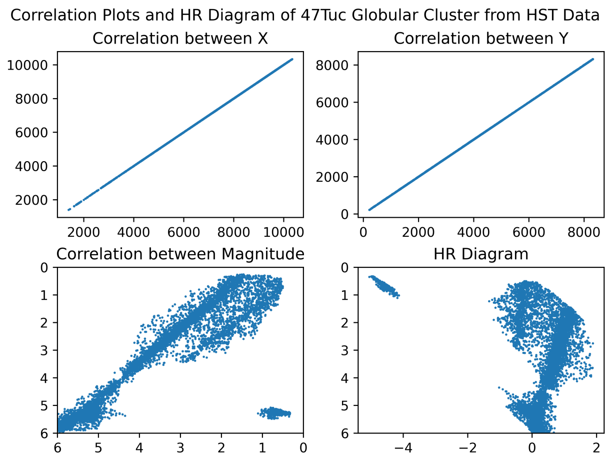

Figure 6. Correlation plots and Color Magnitud Diagram after outliers removal.

3.3. *TrackPy Photometry*

Trackpy is a Python library for tracking blob-like features in video images, following them through time, and analyzing their trajectories. It started from a Python implementation of the widely-used Crocker–Grier algorithm and is currently in transition towards a general-purpose Python tracking library. Here we make an original use of this library by only exploiting its feature detection capabilities with the **locate** function. This function locates gaussian-like blobs of a given size, in our case the size was set to be twice the FWHM of the PSF, characterizes the neighborhoods of the peaks and take only those with given totally integrated brightness. Finally, the position of the peaks is refined. We disabled any kind of image pre-processing made by the trackpy library and decided to detect stars with a minimum separation of 2 pixel, equal to a rounded value of the FWHM. Once the stars were detected, the catalog making procedure was exactly the same we have already illustrated in the previous paragraph.

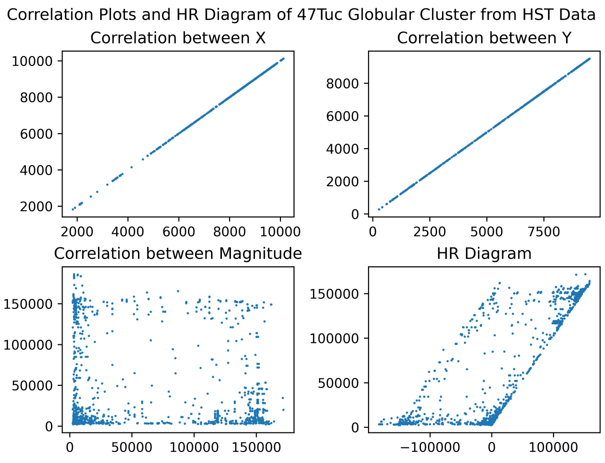

Figure 7. TrackPy Correlation plots and CMD.

4. RESULTS & DISCUSSIONS

4.1. *Aperture Photometry*

As we can see in Fig. 3, the resulting histogram of the magnitudes, computed by simply considering the centroid brightness, is very noisy but we can still detect the double peak profile that highlights the stars of 47 Tuc on the fainter peak and the Small Magellanic Cloud on the other. But once we implement a very barebones version of aperture photometry, the magnitudes histogram is surprisingly decent given the amount of approximations carried out. Indeed, as we can see in Fig. 4 we can clearly detect the double peak profile.

4.2. *PSF Photometry*

As we can see the CMD and the Magnitude Correlation in Fig. 5 is quite noisy compared to the spatial correlation between stars. This is probably due to the fact that the saturation of the stars belonging to the Small Magellanic Cloud is different in the two bands and so the trails of the saturated stars are getting in the way of the star detection algorithm and their brightness is drastically different in the two bands. A quick and easy way to solve this problem without having to deal with a long and time consuming optimization of the detection parameters would be to use a machine learning outliers removal algorithm.

Indeed, using the **Isolation Forest** algorithm we succesfully removed the outliers and "fake" stars from the catalog, resulting in a much less noisy correlation plot and CMD as we can see in Fig. 6.

Using the catalog we built with **DAOStarFinder** we can now check the magnitudes histrogram for the magnitudes in the two different bands:

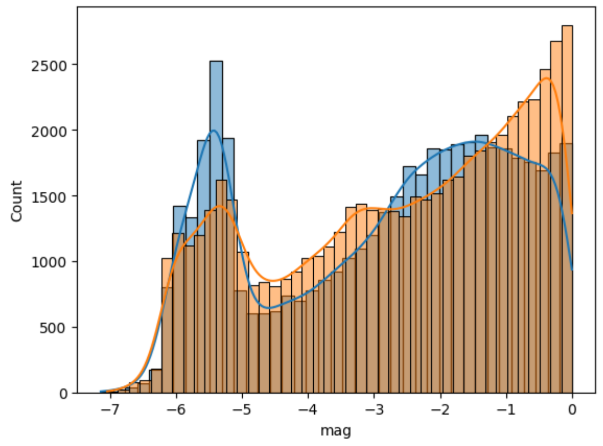

Figure 8. Magnitude Histrogram. Blue for F606W and orange for F814W. Curves are the Kernel Density Estimation (KDE) curves.

We can clearly see the double peak profile with a first peak representing the brighter Small Magellanic Clouds and a second peak given by the stars of the 47 Tucanae Globular Cluster.

We can now differentiate the KDE curves based on the sharpness of the star in Fig. 9. Interstingly enough, we can see that for sharper objects the double peak profile is much more obvious. A third peak seems to appear in the F814W band for the sharpest objects.

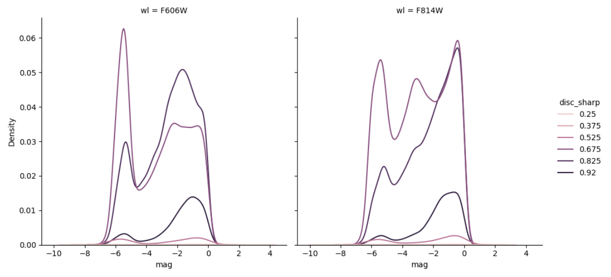

Figure 9. KDE Curves for different sharpenesses.

4.3. *TrackPy Photometry*

Whilst TrackPy was quicker than **DAOStarFinder** at detecting stars, the end results are quite disappointing. The brightness of the detected stars, as we can see in Fig. 7, is much less reliable. This is probably due to the fact that TrackPy wasn't created with star detection and brightness estimation in mind and so it struggles to compute the actual values of brightness even though it correctly detects the majority of the stars. We can exclude that the poor performances of TrackPy are due to a bad link between stars in the two catalogs thanks to the position correlation plots.

5. CONCLUSION

We have presented a detailed analysis of a complex and rich HST data set from cycle 17. We described the observational design of the data used in this lab experience and analysed the stacked mosaics using both conventional and novel techniques with very different results. Through our experimentation, we found that the element that most affects the quality of the detection and therefore analysis of the globular cluster is the kernel used for the fit of the stars in PSF Photometry and for the shape of the boxes in Aperture Photometry.

Regarding strictly the different techniques used for the making of the catalogues we noted that whilst TrackPy was the fasted among the three in detecting stars, the brightness estimation was performed poorly by the program and therefore useless. This comes as no surprise as algorithms such as **DAOFIND**, implemented in the **DAOStarFinder** function we used for PSF Photometry with Astropy, are specifically built with star detection and magnitude estimation in mind. In future we believe that TrackPy could be used for the detection of the stars and then use the location of those to perform aperture photometry, essentially replacing **photutils**' **find_peaks** function in the aperture photometry section.

Software: Astropy Astropy Collaboration et al. (2013), TrackPy Allan et al. (2021)

REFERENCES

Allan, D. B., Caswell, T., Keim, N. C., van der Wel, C. M., & Verweij, R. W. 2021, soft-matter/trackpy: Trackpy v0.5.0, v0.5.0, Zenodo, doi: 10.5281/zenodo.4682814

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33, doi: 10.1051/0004-6361/201322068

Brammer, G., Koekemoer, A. M., & Kiziltan, B. 2003, in HST Calibration Workshop : Hubble after the Installation of the ACS and the NICMOS Cooling System, 325

Da Costa, G. S. 1992, in Astronomical Society of the Pacific Conference Series, Vol. 23, Astronomical CCD Observing and Reduction Techniques, ed. S. B. Howell, 90

Hansen, B. M. S., Anderson, J., Brewer, J., et al. 2007, ApJ, 671, 380, doi: 10.1086/522567

Harris, W. E. 1996, AJ, 112, 1487, doi: 10.1086/118116

Harris, W. E., & Racine, R. 1979, ARA&A, 17, 241, doi: 10.1146/annurev.aa.17.090179.001325

Heasley, J. N. 1999, in Astronomical Society of the Pacific Conference Series, Vol. 189, Precision CCD Photometry, ed. E. R. Craine, D. L. Crawford, & R. A. Tucker, 56

Jakobsen, R., & Laurance, R. J. 1989, ESA Bulletin, 58, 91

Kalirai, J. S., & Richer, H. B. 2010, Philosophical Transactions of the Royal Society of London Series A, 368, 755, doi: 10.1098/rsta.2009.0257

Kalirai, J. S., Richer, H. B., Anderson, J., et al. 2012, AJ, 143, 11, doi: 10.1088/0004-6256/143/1/11

Kukarkin, B. V., & Kireeva, N. N. 1974, Soviet Ast., 18, 346

Mighell, K. J. 1999, in Astronomical Society of the Pacific Conference Series, Vol. 189, Precision CCD Photometry, ed. E. R. Craine, D. L. Crawford, & R. A. Tucker, 50

Richer, H. B., Fahlman, G. G., Ibata, R. A., et al. 1995, ApJL, 451, L17, doi: 10.1086/309674

Sawyer Hogg, H. 1959, Handbuch der Physik, 53, 129, doi: 10.1007/978-3-642-45932-0_4

Shapley, H. 1930, Star Clusters, Vol. 2

Stetson, P. B. 1987, PASP, 99, 191, doi: 10.1086/131977

Vandenberg, D. A., & Bell, R. A. 1985, ApJS, 58, 561, doi: 10.1086/191052