Field restoration

II assignment of *Astrofisica Osservativa*

Bernardo Vettori

April 11, 2025

Abstract

This exercise aims to simulate the reconstruction of a sample of stars from an acquired image. The generating of the artificial acquisition starts from a star sample obtained by drawing values from an Initial Mass Function. Each star is randomly located in an empty frame to which a background normal distributed noise is added. This image is then convolved with a Gaussian kernel to simulate the atmospheric seeing effect. Finally, an additive Gaussian noise is involved. After the initialization process, we discuss the procedure applied to restore the initial brightness distribution. A Bayesian-based iterative method is used to deconvolve the field. The results show evidence of the Malmquist bias, which is induced by setting a lower threshold as an acceptance condition to detect an object. In the end, a field with Poisson fluctuations in pixel counts is reported and we explain why the deconvolution algorithm fails in this case.

1 Introduction

The second assignment of *Astrofisica Osservativa* requires the implementation of a script to recover the brightness distribution of stars from an artificial image of the sky.

A python package called `skysimulation` collects all the python modules to generate the field, to detect the stars, and to restore their brightness and it is discussed in Section 2. Section 3 reports the steps to produce the image, particularly the star sample, the kernel, and the noise. In Section 4 in addition to the definition of a stellar object we explain how restoring the initial brightness distribution through a Bayesian-based iterative method and discuss the results in Section 5. Finally, Section 6 presents the analysis of the recovery algorithm in the case of Poisson fluctuations.

2 The python package

To accomplish the tasks of the assignment we wrote a Python package named `skysimulation`[1], which is organized into four modules:

- `stuff.py`
  This module collects many useful methods such as the ones to load and save data, the function for the convolutions, and the classes for the used distributions.
- `display.py`
  This module contains the function to display the generated matrices as images.
- `field.py`
  This module implements all the methods to initialize the star sample and generate the frame.
- `reconstruction.py`
  This module provides the methods required to restore the initial brightness distribution.

The package documentation contains a detailed description of each function and module. In this report, only the main steps of the implementation are discussed.

[1] See https://github.com/00-berni/proj_2

3 Field generating

The first part of the assignment is generating the artificial photometric acquisition. The building blocks of the procedure involve: the mass sample, the background, the seeing effect, and the detector noise.

3.1 Mass and brightness sample

The method `generate_mass_array()` provides the mass sample through a Monte Carlo method, drawing values from a power law taken as the Initial Mass Function $\Phi(M)$ (IMF) of the stars:

$$\Phi(M) = \left(M\,[M_\odot]\right)^{-\alpha} \tag{1}$$

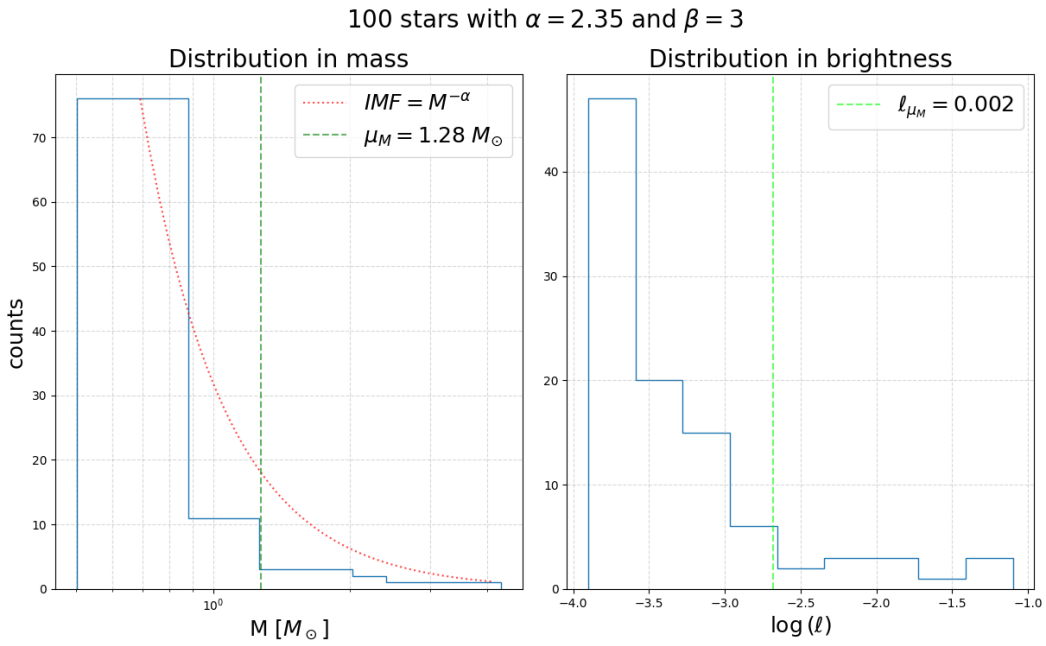

Figure 1: Sample distribution in mass (left) and in brightness (right). μ_M refers to the mean mass of the sample (see in the following sections) and ℓ_{μ_M} to its brightness.

where M is the mass of a star in the mass range $[0.5\,M_\odot, 10\,M_\odot]$. The mass-luminosity relation is a power law, too:

$$\ell(M) = K \cdot M^\beta \tag{2}$$

where ℓ is a dimensionless effective luminosity and K is a normalization constant[2]. The generated sample is shown in Figure 1 for the parameters[3] $\alpha = 2.35$ and $\beta = 3$.

Then the method `star_location()` locates the stars in an empty $N \times N$ matrix ($N = 100$). The procedure is designed to assign a position in the frame to a star from a uniform distribution. The image[4] $\mathcal{S}$ thus obtained (named *Source Image*) is characterized by the fact that the brightness of any star of the sample is all gathered in one single pixel, as shown in Figure 2 (left).

3.2 Disturbance effects

After the initialization process, the next step is to compute the effects of background, atmospheric seeing, and detector noise. The background $\mathcal{B}$ and detector $\mathcal{N}$ noises are generated from two different normal distributions. The seeing effect of the atmosphere is gathered by the convolution with a Gaussian kernel Π, instead. Before the convolution, the field matrix is located in a bigger frame filled with background noise to avoid artifacts at the edge of the image. The generated image $\mathcal{I}$ (named *Light Frame*) is then [1, 3, 5]:

$$\mathcal{I} = (\mathcal{S} + \mathcal{B}) * \Pi + \mathcal{N} \tag{3}$$

where $*$ means the convolution.

The parameters used for the implementation presented in this report are listed in Table 1.

i	μ_i	σ_i/μ_i
$\mathcal{B}$	$3.2 \cdot 10^{-4}$	20%
$\mathcal{S}$	$5 \cdot 10^{-5}$	50%
Π	$\sigma_\Pi = 3$ px	

Table 1: Parameters for background and detector noises. The last row is the STD of the Gaussian kernel.

3.3 Master Light

To simulate the acquisition process the method `field_builder()` generates a set number of Light Frames (6 by default) with different realizations of the background and detector noise (Figure 15) and then averages them pixel to pixel (Figure 2 right). The uncertainty in the values of the mean-field (named *Master Light*) is the STD pixel to pixel. The method also simulates a certain number of Dark acquisitions, which are frames filled with detector noise only [1, 3, 5], and computes the Master Dark by averaging them (Figure 16).

[2] $K \equiv M_{max}^{-\beta}$

[3] The Salpeter's law and an approximation of the mass-luminosity relation for main sequence stars [5], respectively.

[4] The term *image* refers to the graphic representation of a *matrix* $k \times t$, but in this report, these two words are used synonymously.

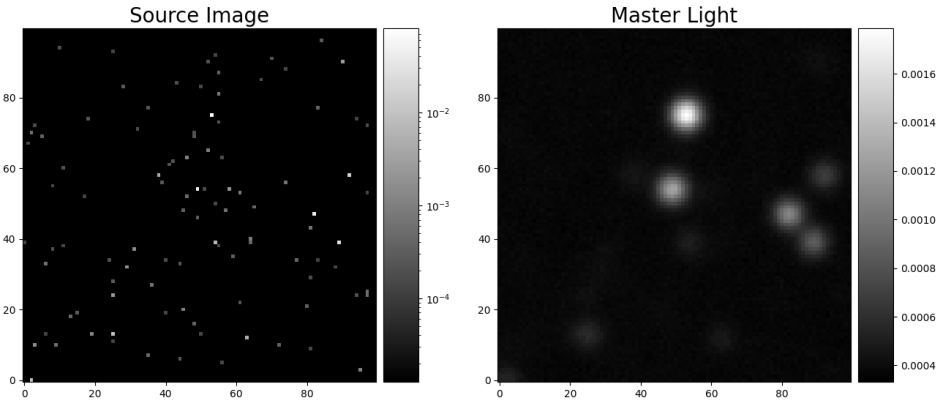

Figure 2: Comparison between the Source Image (left) and the Mater Flat (right).

4 Restoration process

At the end of the initialization process, the Source Image is lost. After subtracting Master Dark from Master Light to remove the effect of detector noise and obtain the called *Science Frame* (Figure 3) the restoration process starts. It consists of four main steps: (a) the evaluation of the background, (b) the estimation of the seeing kernel, (c) the deconvolution of the field, and (d) the recovery of the initial distribution in brightness.

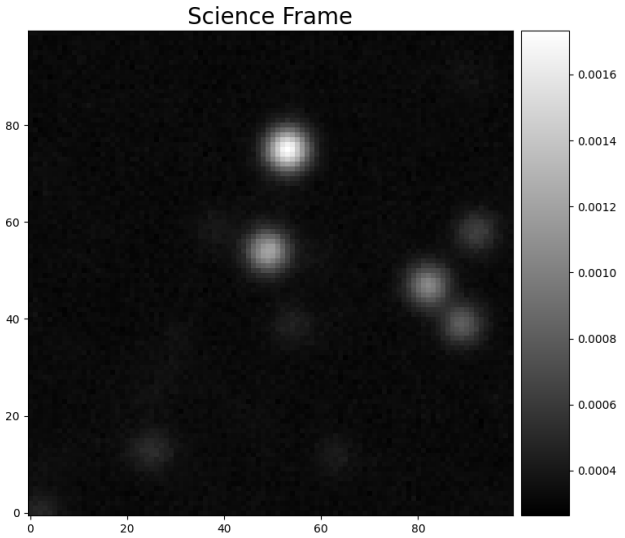

Figure 3: The computed Science Frame.

4.1 Background mean

The first step of the restoration process is to estimate the mean of the background noise. This value is significant for the routines in the following because it sets the lower limit for an object to be detected. The method `bkg_est()` approximates $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ with the median

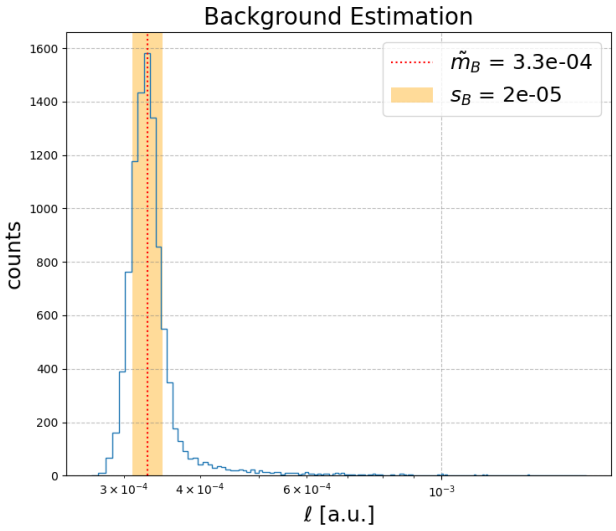

Figure 4: Distribution of the Science Frame brightness. The red dotted line is the median of the data and the orange area is the region at one STD from the value.

$\tilde{m}_{\mathcal{B}}$ and the STD $s_{\mathcal{B}}$ of the data, respectively[5]. For the latter, the method selects only values around $\tilde{m}_{\mathcal{B}}$ to avoid the effect of the brightest stars. Figure 4 reports the result of this procedure. The mean is well estimated by the median, instead $s_{\mathcal{B}}$ seems to approximate $\sigma_{\mathcal{N}}$ better than $\sigma_{\mathcal{B}}$.

To avoid handling negative values during the deconvolution process the estimated value of the background is not subtracted immediately from the frame.

4.2 How to detect stars

Recovering the kernel used for atmospheric seeing requires fitting a Gaussian profile to that of a star. In the case of the Source Image, the meaning of "a star" is clear, that is, a not-empty pixel, but after the convolu-

[5] We adopt the Greek letters for the parameters of the original distribution and the Latin letters for their estimations.

tion stars become extended objects. Therefore, defining what we mean with a *stellar object* is essential before the implementation of any detecting procedure.

4.2.1 A stellar object

In this implementation a *stellar object* is a section of the field (a matrix $L \times J$) around a bright pixel, called the *center*[6], where one can identify (approximately) a decreasing trend in brightness as progressing from the center toward any direction. We divide stellar objects into two categories according to the half value of their central pixel: *bright* whether it is higher than $\tilde{m}_{\mathcal{B}}$ and *faint* otherwise. Depending on the type of stellar object, the acceptance conditions change. By default, an acceptable *bright* object is an object with either a decreasing mean trend in brightness as progressing to the edge and a signal-to-noise ratio (SNR) higher than a chosen threshold; for *faint* object only the condition on SNR is used, instead. The mean trend in brightness is computed by the method `average_trend()`, which averages the brightness of pixels at the same distance from the center of the object inside a frame wide as the FWHM. This is the reason for the classification criterion: by definition, assuming $\tilde{m}_{\mathcal{B}}$ as the lowest brightness value, FWHM can not be estimated for *faint* objects. In this case, the frame size is the mean FWHM of the previously detected *bright* objects if any, or otherwise a preset value.

The identification and classification of a stellar object is provided by the method `object_check()`.

4.2.2 The detecting algorithm

A procedure to detect and store stellar objects is one of the main implementation tasks. This is provided by the implemented iterative algorithm named `searching()`. Each cycle of the routine can be summarized in a few key points:

1. find the position of the brightest pixel
2. compute the size of the object by studying the trend in brightness along all directions
3. check whether it is acceptable as a stellar object (according to the previous definition) and identify the type (*faint* or *bright*)
4. subtract $\tilde{m}_{\mathcal{B}}$ to the object and fit a 2D-Gaussian to its profile.
5. store the object thus estimated and subtract it from a copy of the field (to prevent loss of data)

If step 3 fails, the object is completely removed from the copied image, that is those pixels are set to 0. The process stops when the value of the brightest pixel is lower than $\tilde{m}_{\mathcal{B}} + s_{\mathcal{B}}$.

[6]Regardless of whether L and J are odd or not.

The method returns two lists, one of the estimated objects and the other of the corresponding uncertainties, which are set as the STD of the fit residuals. The coordinates of each object are returned as well.

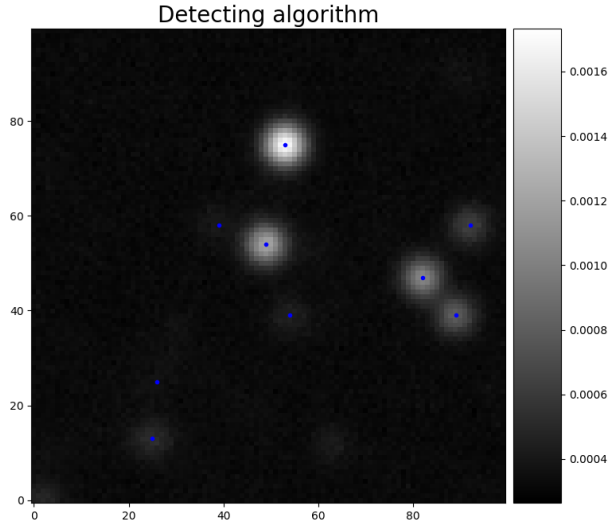

Figure 5: The selected bright objects to find the kernel.

4.3 Estimation of seeing effect

Estimating the kernel used to simulate the seeing requires the analysis of the brightest objects in the field. In this implementation, the number of selected targets is limited to just 10 at the most with a strict condition on the SNR (Figure 5). The method `kernel_estimation()` estimates σ^2_{Π} through a harmonic mean of the variances of the objects weighted on their central brightness. Each variance $s^2_{\Pi,k}$ is obtained by the 2D-Gaussian fit from the previous routine. Figure 6 shows the kernel thus computed.

For this implementation, we get $s_{\Pi} = (3.0 \pm 0.1)$ px in accordance with the expected value.

4.4 Deconvolution of the field

At this point, the kernel used for the convolution is known so that we can deconvolve the field. The chosen algorithm is the Bayesian-based iterative method explained in [2, 4] and is provided by the method `LR_deconvolution()`. According to this procedure at the rth iteration we get:

$$\mathcal{I}^{(r)} = \left(\mathcal{S}^{(r)} * \Pi_{est}\right)$$
$$S^{(r+1)} = S^{(r)} \cdot \left(\frac{\mathcal{I}}{\mathcal{I}^{(r)}} * \Pi_{est}\right)$$

where Π_{est} is the estimated kernel. We take the same precautions adopted for the convolution in the field-generating process.

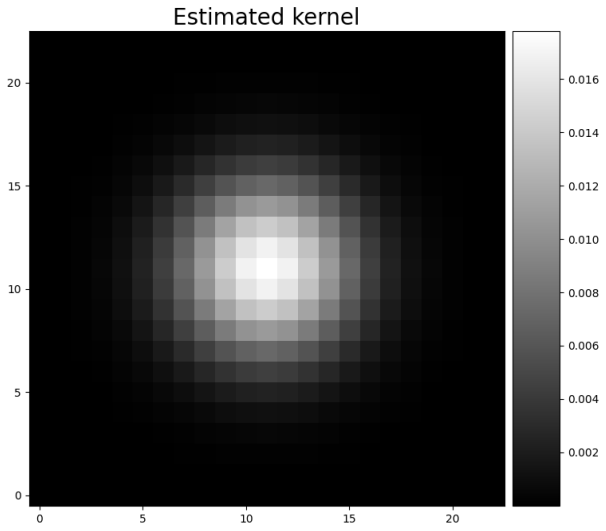

Figure 6: The estimated kernel.

By default, the algorithm stops when the difference between two subsequent steps integrated over the entire image reaches a certain threshold. This difference $\Delta\mathcal{I}_r^{r+1}$ is defined as:

$$\Delta\mathcal{I}_{(r)}^{(r+1)} \equiv \sum_{i,j} \frac{\left|\mathcal{I}_{i,j}^{(r+1)} - \mathcal{I}_{i,j}^{(r)}\right|}{\sum_{k,t}\mathcal{I}_{k,t}}$$

where the subscripts refer to the pixel in the position (i,j) or (k,t). The convergence of the method and the field obtained from the deconvolution are reported in Figures 7 and 8, respectively.

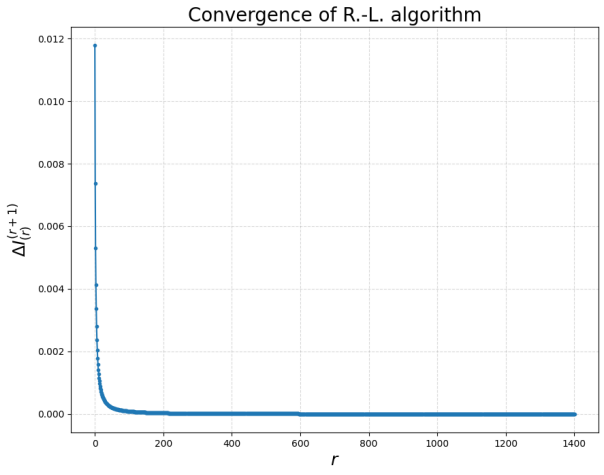

Figure 7: The trend of the difference of two subsequent images as the iteration step grows.

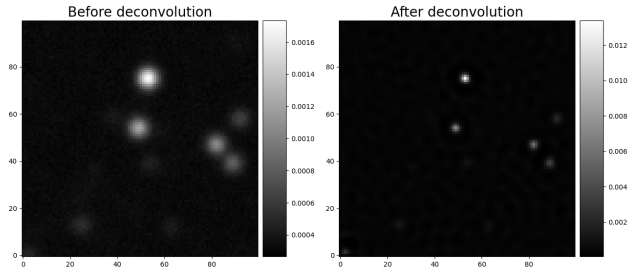

Figure 8: Comparison between the image before (left) and after (right) deconvolution.

4.5 Light restoration

Finally to restore the distribution in brightness the method `light_recover()` calls the method `searching()` and collects the value of the object's cumulative brightness, that is the sum over the entire object.

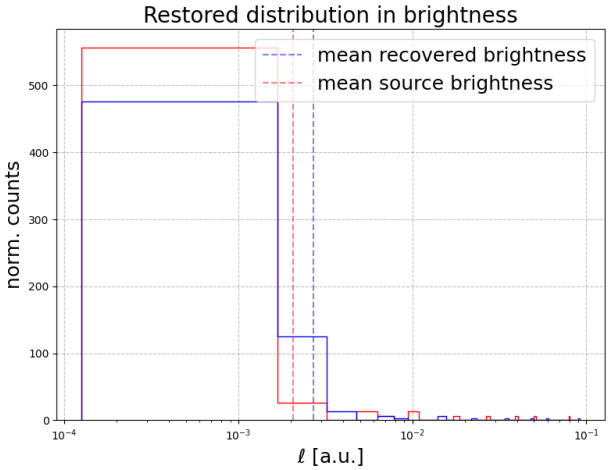

Figure 9: Brightness distribution of the initial sample (red) and the restored one (blue). The counts are normalized so that the histogram area is unitary. As explained in the Section, a sample brighter than the original on average is evidence of the Malmquist bias.

5 Comments

From the comparison between the original and recovered brightness distributions, shown in Figure 9, two main features can be noticed: the number of the faintest stars is underestimated and the average brightness of the recovered objects is greater than the brightness of the mean mass[7] of the generated sample. These two re-

[7] The mean mass is defined as:

$$\mu_M \equiv \frac{\int M\,\Phi(M)\,dM}{\int \Phi(M)\,dM} = \frac{\alpha-1}{\alpha-2}\frac{M_{min}^{-(\alpha-2)} - M_{max}^{-(\alpha-2)}}{M_{min}^{-(\alpha-1)} - M_{max}^{-(\alpha-1)}}$$

Hence, its brightness is $\ell_{\mu_M} = K\,\mu_M^{\beta}$.

sults are easily explained by the fact that we set a lower threshold for the brightness of a star. For values equal to or lower than the background estimated mean it is impossible to distinguish between stars and noise fluctuations. Hence, the procedure introduces a cutoff in the brightness distribution shifting the sample average to higher values. This kind of bias is called the Malmquist bias [3, 5].

In general, there may be the possibility that the restored sample is affected by other biases that are induced by the detecting algorithm and reproduce the same results as the Malmquist one. In this case, we would expect that the detected objects are not distributed uniformly in space. To check this effect we compare the distribution of the relative distance between recovered objects with that between uniformly distributed objects. The latter was obtained by generating a set of 2000 Source Images with the same mass sample. The accordance between the two distributions, shown in Figure 10, is quite good, proving that the algorithm applies no preferential conditions to a target other than the ones on the brightness.

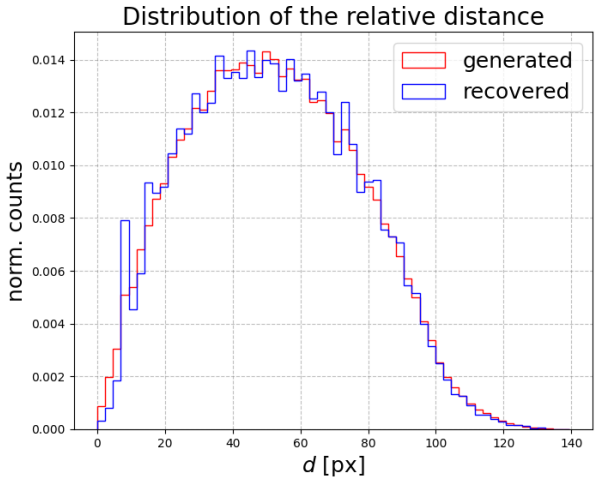

Figure 10: Comparison of the distribution of the relative distance between objects for a uniformly distributed sample and the recovered one. The counts are normalized so that the histogram area is unitary. The overestimates, particularly at small distances, are probably due to the appearance of artifacts after the deconvolution procedure. The discussed algorithm tends to increase the brightness of a bright pixel by the light of its surroundings, hence a noise fluctuation could become a fake star at the end. However, an analysis of the artifacts is out of the scope of this assignment.

To show more clearly the evidence of the Malmquist bias, keeping fixed the Source Image we performed 20 synthetic acquisitions for each of 6 different values of $\mu_{\mathcal{B}}$. We collected the mean brightness $\bar{\ell}_i(\mu_{\mathcal{B}})$ $(i = 0, \dots, 20)$ for each restored sample and averaged the estimated 20 values to get the mean brightness $\bar{\ell}(\mu_{\mathcal{B}})$ per background mean $\mu_{\mathcal{B}}$. The results are shown in Figure 11.

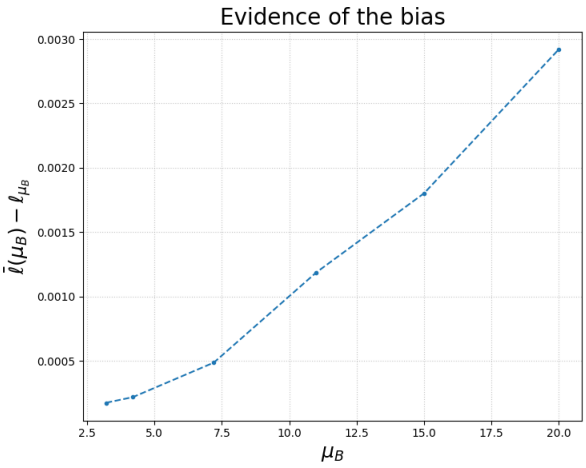

Figure 11: The growth of the discrepancy between the average brightness of the sample and the original respect to the mean of the background noise.

6 A peculiar case

In this section, we investigate a particular case in which the chosen deconvolution approach fails.

Field generating The image is generated in the same way as before but without noise, that is the source convolved with the kernel only. At this point, we suppose that the counts in each pixel of the detector image are Poisson-distributed. So assuming $\ell_{i,j}$ the brightness of the pixel (i, j) and $n_{i,j}$ the corresponding counts detected in the same position we get $\mu_{n_{i,j}} = \ell_{i,j}$ and $\sigma_{n_{i,j}} = \sqrt{\ell_{i,j}}$ for each pixel. Figure 12 shows the two images. The Master Light is obtained by averaging 6 different acquisitions.

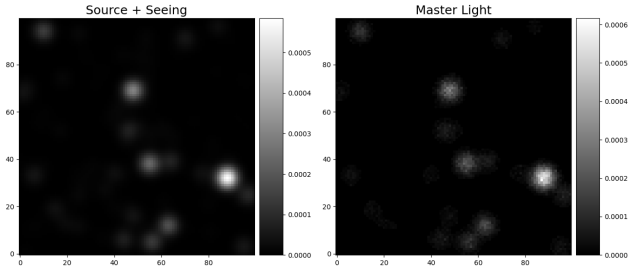

Figure 12: The effect of the Poisson fluctuations in the counts at the detector.

Restoration process We follow the same steps as in the previous case. We estimate the kernel by fitting a Gaussian to the brightest objects (Figure 13). Despite the Poisson fluctuations, which induce a higher

uncertainty in central pixel localization, we obtain a less precise but acceptable value for the STD $s_\Pi = (2.7 \pm 0.8)$ px. However, the deconvolution process is strongly affected by the Poisson noise. Since the fluctuations are on average proportional to the square root of the pixel brightness, faint objects can easily be fainter or vanish, while a pixel in the halo of a bright star could be brighter and become a fake object after the deconvolution. The recovered brightness distribution indeed matches in no way the original, underestimating the low brightness values.

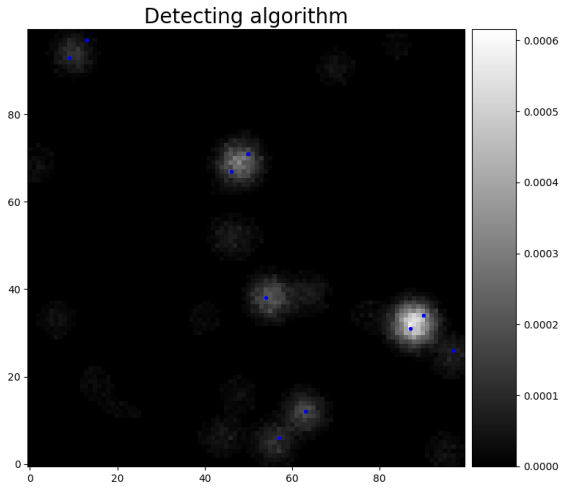

Figure 13: Selected objects for the estimation of the kernel.

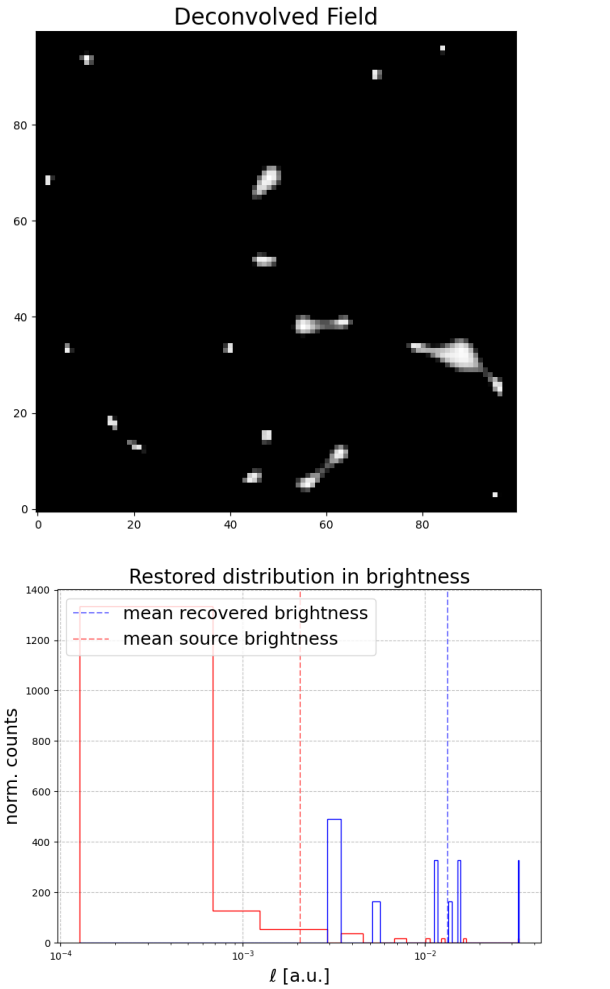

Figure 14: Top: the image after the deconvolution in logarithmic scale. Bottom: the brightness distributions of the restored sample (blue) and the original one (red). The counts are normalized so that the histogram area is unitary.

References

[1] M. Gallaway. *An Introduction to Observational Astrophysics*. Undergraduate Lecture Notes in Physics. Springer Cham, 1st edition, 2016. doi: 10.1007/978-3-319-23377-2.

[2] L. B. Lucy. An iterative technique for the rectification of observed distributions. *ApJ*, 79:745, June 1974. doi: 10.1086/111605. URL https://ui.adsabs.harvard.edu/abs/1974AJ.....79..745L.

[3] Notes of *Astrofisica Osservativa* course, 2021-2022.

[4] W. H. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America (1917-1983)*, 62(1):55, January 1972. URL https://ui.adsabs.harvard.edu/abs/1972JOSA...62...55R.

[5] S. N. Shore. *Tapestry of Modern Astrophysics*. John Wiley & Sons, Inc., 2003. ISBN 0-471-16816-5.

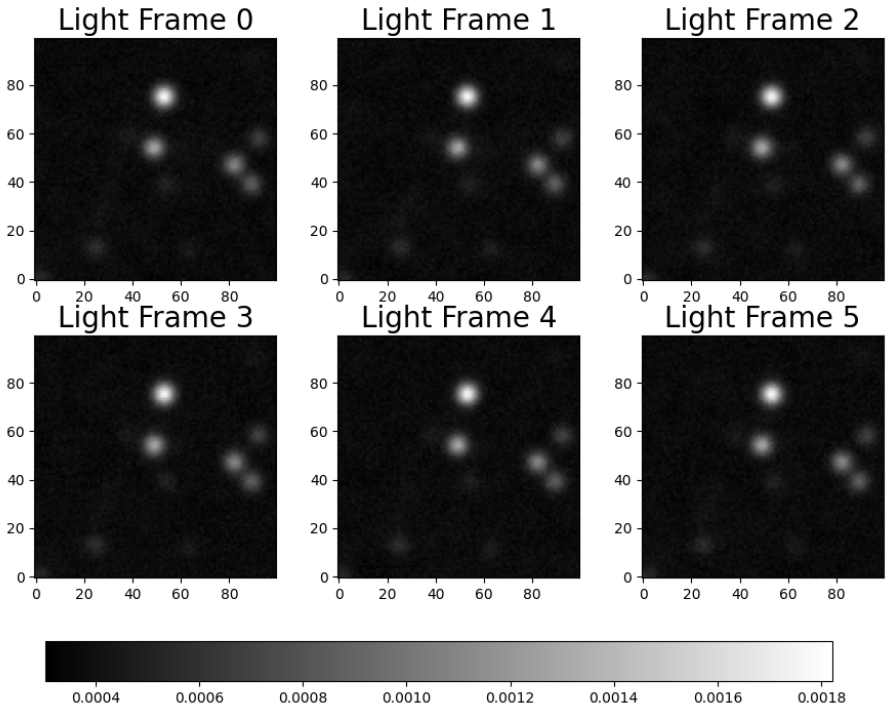

Figure 15: The Master Light is computed by averaging 6 different acquisitions, reported here.

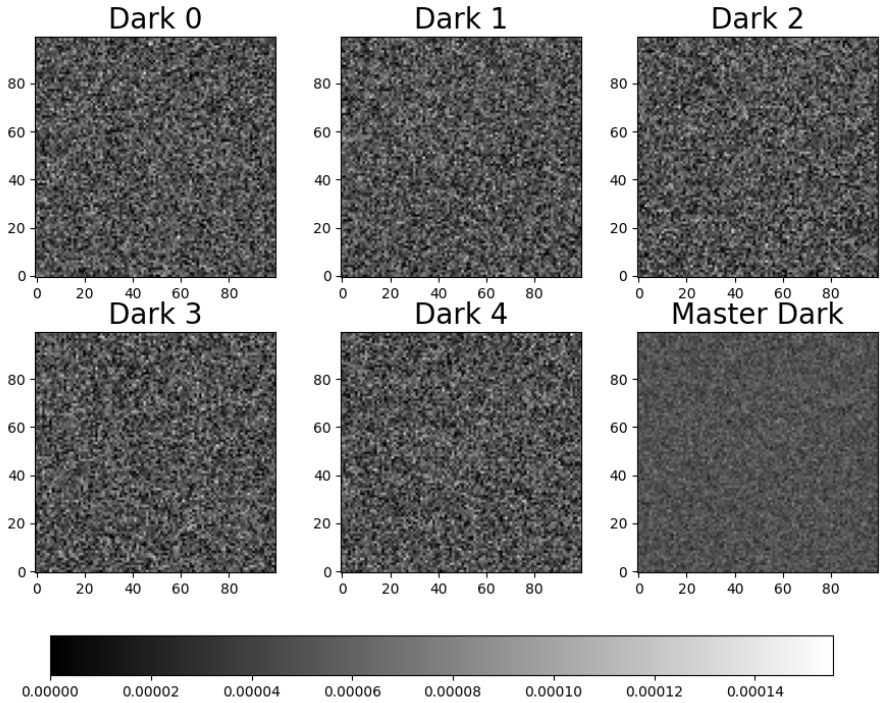

Figure 16: The Master Dark is computed by averaging 5 different Darks.