PATCH Tutorial

Andrea La Camera, Laura Schreiber

November 2023

1 Introduction

Software `Patch` is an IDL software developed for the deconvolution of mono-pupil images with space-variant point-spread functions (PSFs) [1, 2]. The input image is decomposed in partially overlapping sub-domains, the size of which will depend on the properties of the PSFs.

The reconstruction is obtained by applying a deconvolution method with boundary effects compensation to each sub-domain. The final object is formed by a mosaic of the partial reconstructed images, after the removal of the common parts of adjacent sub-domains.

The aim of this tutorial is to help the users to deconvolve a mono-pupil image with space-variant PSFs by means of the software `Patch` . In this example we are using a popular data set: the simulation of a star cluster for the Hubble Space Telescope's Wide Field Camera.

From the repository Tutorial[1] of the software `Patch` , you can download the archive called "`hst_cluster.zip`". The zip file contains the input image, the PSFs and a README file describing all the necessary parameters for the deconvolution. You can extract the archive within the "`patch_data`" sub-folder.

The tutorial is split in three parts: the first one shows how to load the inputs (image and PSFs) and how to set up the sub-domain parameters (described in Sect. 2, the second one describes the deconvolution process and the necessary parameters to be set (Sect. 3), and the last part concerns the visualization and the necessary steps for saving the results (Sect. 4).

Before starting, be sure that the software is correctly installed, i.e. you are able open the Graphical User Interface (GUI), shown in Fig. 1. Follow

[1] `https://www.ict.inaf.it/gitlab/laura.schreiber/patch/tutorial`

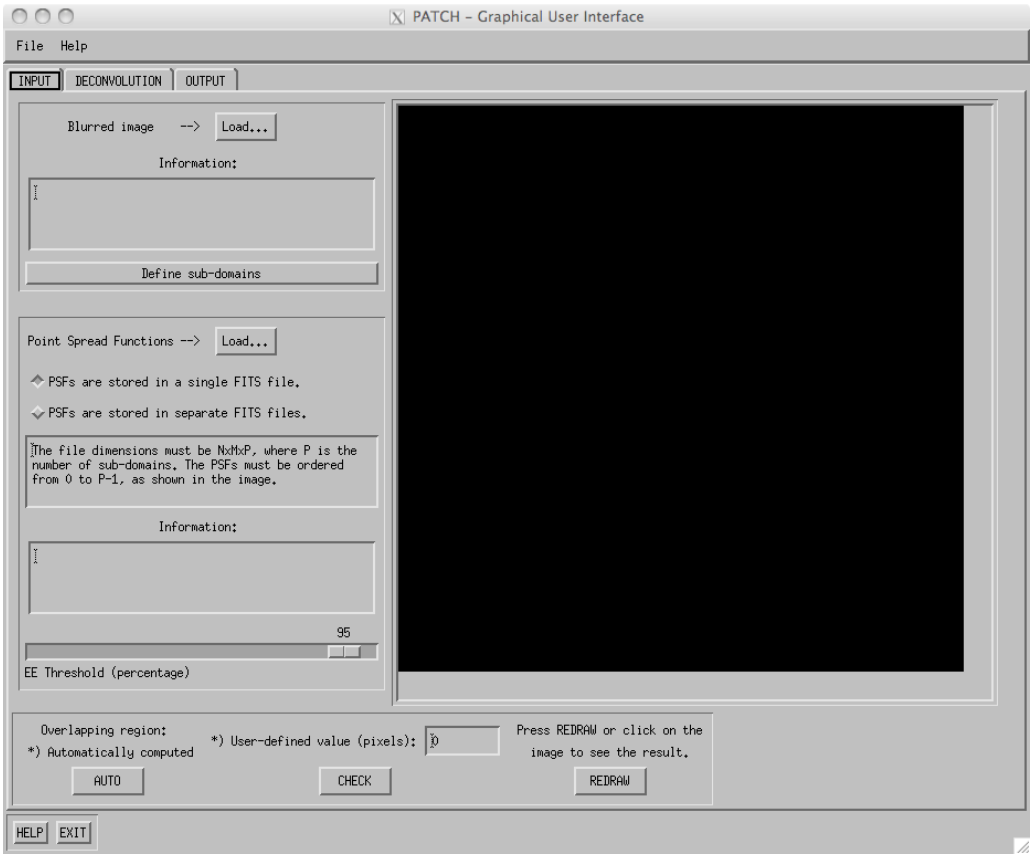

Figure 1: The main GUI of `Patch` . Note that there are three "panels", or "tabs", one for the inputs, the second one for the deconvolution and the last one for showing and saving the results.

the installation instructions contained within the "`README.txt`" file included in the archive. Then start IDL and type "`patch`" at the prompt: the GUI of the program appears. Note that there are three "panels", or "tabs", one for dealing with the inputs, the second one for the deconvolution and the last one for showing and saving the results.

2 First Part - Load the Input Data

2.1 Blurred Image

First of all, you should load the input image to be deconvolved. Click on the "`Load...`" button and select the "`sim5.fits`" file. The image is read and displayed within the graphical area, on the right part of the GUI (Fig. 2).

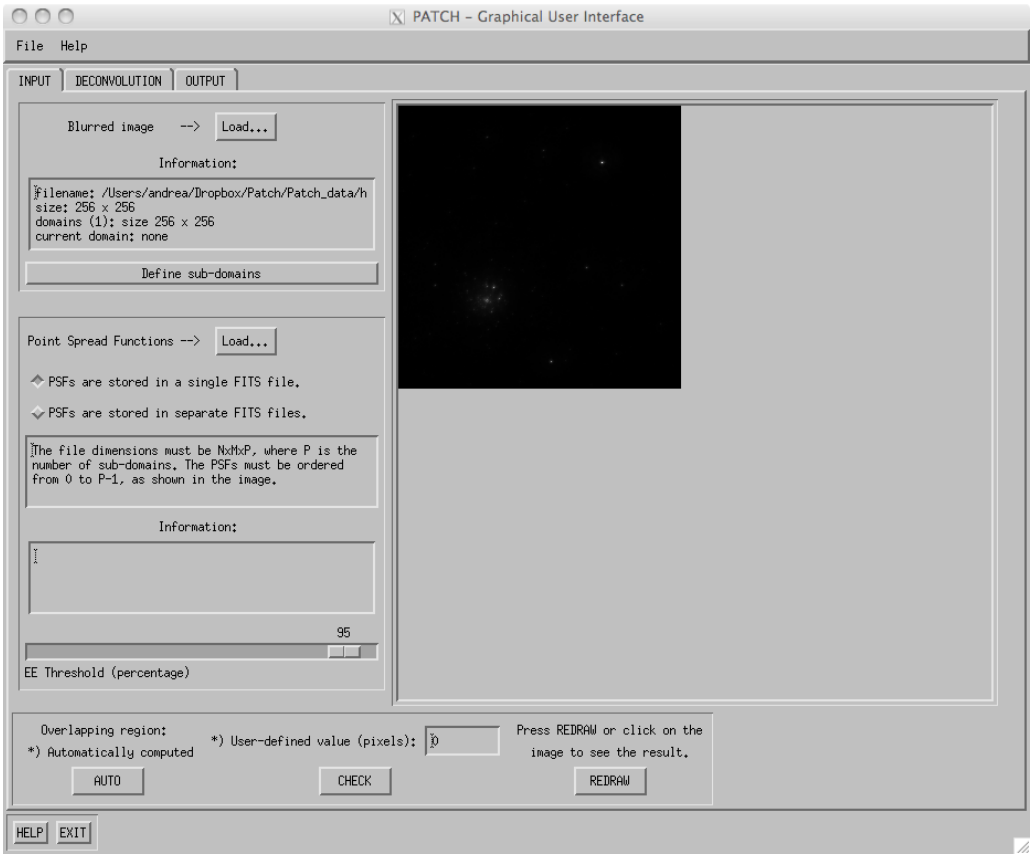

Figure 2: The input image is now displayed on the GUI of `Patch` .

Remark. Only `.FITS`*,* `.FIT`*, or* `.FTS` *file formats are supported.*

2.2 Visualization Options

Right click on the image to open a pop-up window. Within the window you can change the zoom, the scale and the color map of the image. Press "`Apply and close`" to apply your settings. See Fig. 3.

2.3 Sub-domains

At this point you should define the sub-domains that decompose the input image. Since in this example 25 PSFs are provided, you should define a 5×5 grid. Click on the "`Define sub-domains`" button to open a pop-up window. Click on the first numeric field and write "5" (the number of rows you want to divide the image) and press "`enter`".

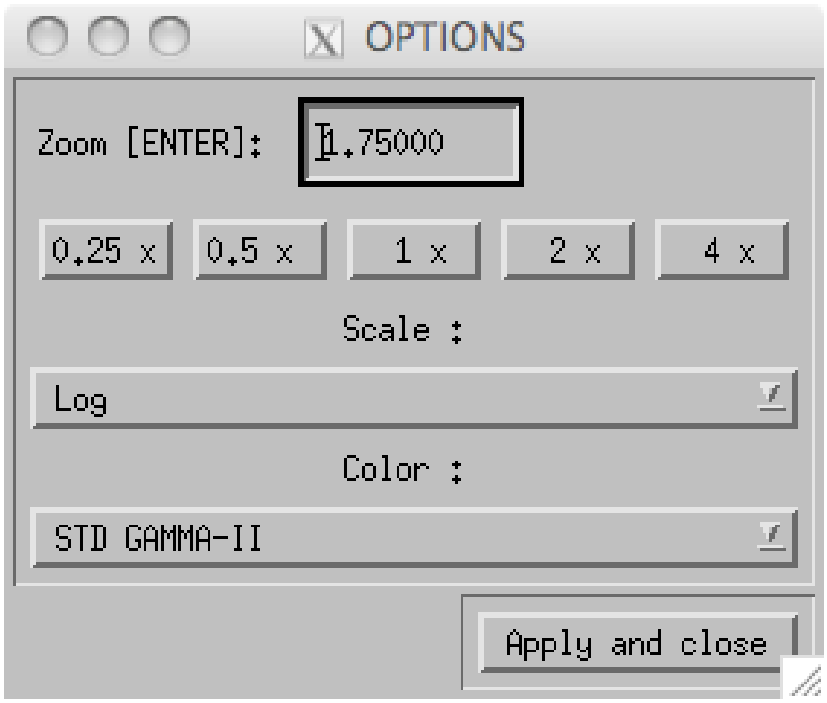

Figure 3: The visualization options available on `Patch` .

In general, if the size of the image is divisible by your input, the resulting size of each sub-domain is updated (second row of the window). Otherwise (as in this case) a pop-up message will inform you that a small zero-padding is necessary in order to enlarge the size of the input image. Answer "`Yes`" to the question and the size is now changed from 256 to 260 pixels.

Repeat now the same operations with the number of columns (5 again) in the second numeric field. Each sub-domain is now defined as an array of 52x52 pixels.

Finally press the "`Save and close`" button. See Fig. 4.

2.4 Point-Spread Functions

The PSFs can be stored (and therefore loaded) in a unique FITS file (cube of PSFs) or in separate files, one PSF for each sub-domain. In both cases, the order is **very important**: the first PSF of the cube (or the first file) must concern the first sub-domain, similar for the second one and so on.

If P is the number of sub-domains, the PSF must be ordered from 0 to $P-1$, as shown in Fig. 5. In order to help the users to remember the order of the sub-domains, a grid is drawn over the image and a progressive number

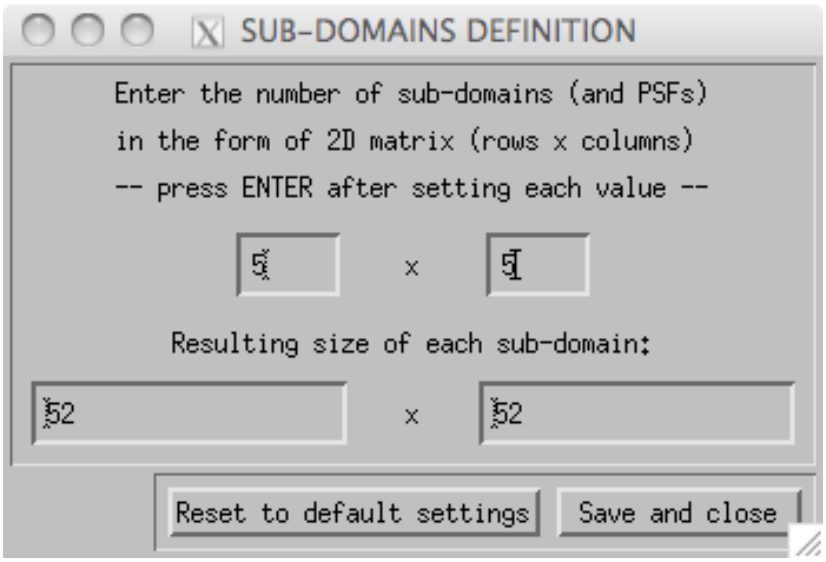

Figure 4: The defined 5×5 sub-domains.

counts the sub-domains.

2.5 Load PSFs...

Choose the "`separate files`" case and then click the "`Load...`" button to load the PSFs. Within the loading window, select all the 25 PSFs (from "`mypsf00.fits`" to "`mypsf24.fits`") at once and press "`OK`".

The files are then read and the dimensions of the PSFs are compared with the ones of the generic sub-domain. Indeed, in order to correctly apply the deconvolution methods with boundary effects compensation (see Sect. 3), a larger PSF is necessary. For this reason, a message informing you about a zero-padding will appear during the loading process. The dimensions of the enlarged PSF depend on the dimensions of each sub-domain and on the width of the PSF itself.

In order to find out this value, given a distance r from the centre of the PSF, we define the *Enclosed Energy* (EE) as the sum of the values (counts) of the PSF within the square box of length $= 2r$ and centered on the centre of the PSF.

The $EE(r)$ is a crescent function and, since the PSF is normalized to a unitary volume, it is defined between 0 and 1. Therefore $EE(r)$ crosses a certain threshold (called *EE Threshold*) in $r = \Delta$. The EE Threshold parameter can be set by moving a slider from left to right. For this example, set to 97% the threshold percentage.

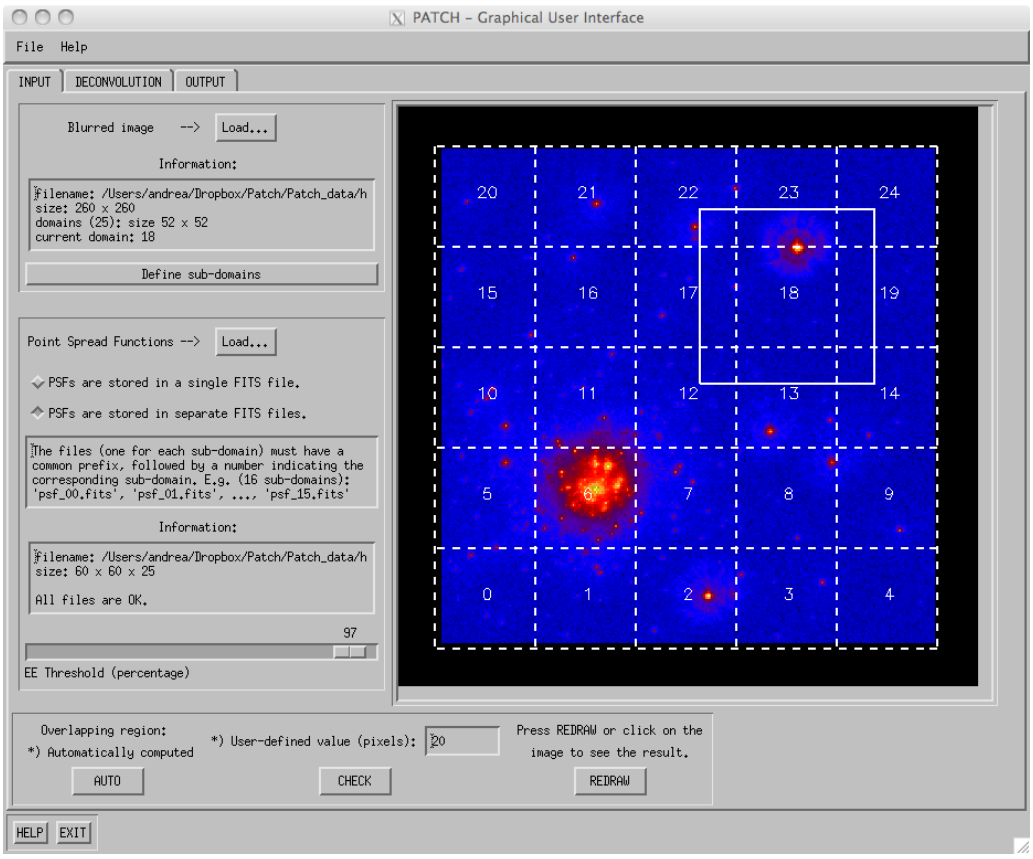

Figure 5: The 5×5 grid drawn over the image helps the users to remember the order of the sub-domains and, therefore, the order of the input PSFs.

2.6 Overlapping Region

Since the deconvolution of disjoint domains can introduce *artifacts* in the common boundaries, it is necessary to extend the sections to partially overlapping regions either by choosing an automatically-computed value or by entering a user-defined number.

In this last case, the number must be checked (by pressing the corresponding button) because it must be greater than the auto value (for more details see the following math part).

Of course, you can choose a larger number by entering the value on the text field. By pressing the "`redraw`" button the image will show the new settings. In our example, the auto value is 20 pixels and, by looking at the image, we accept this value.

2.7 Extra 1 - Equations

The overlapping region is automatically determined as follows:

- given the dimensions of the PSFs ($N_P \times M_P$) and of the sub-domains, we call $O_A = max[(N_P - dom_x)/2, (M_P - dom_y)/2, 0])$;
- given Δ, we call $O_B = \Delta$;
- the overlap is the largest value between O_A and O_B.

Of course, a larger user-defined number can be used (for example, by looking at the input image, you may want to adopt a value in order to include a particular feature within a specific sub-domain).

2.8 Extra 2 - More Equations

As discussed above, depending on the size of the input image ($N \times M$) and on the number of sub-domains ($n_{col} \times n_{row}$), the dimensions of each sub-domain ($dom_x \times dom_y$) are computed by following this approach:

- From $N \times M$ to $N' \times M'$ (first zero-padding) such as N' is multiple of n_{col} and M' of n_{row}. This step is skipped if the divisibility rule is satisfied: in such a case $N' = N$ and $M' = M$. The resulting dimensions of each sub-domain are: $dom_x = N'/n_{col}$ and $dom_y = M'/n_{row}$.
- From $N' \times M'$ to $N'' \times M''$ (second zero-padding) such as $N'' = N' + 2 * overlap$ and $M'' = M' + 2 * overlap$. The resulting dimensions of each sub-domain are: $dom'_x = dom_x + 2 * overlap$ and $dom'_y = dom_y + 2 * overlap$.

The resulting dimension of the sub-domains is used as "working dimension" (eventually enlarged to the next power of 2).

Example - Our input image is 256×256 and we want to decompose the image in 5×5 sub-domains with $overlap = 20$ pixels. Therefore:

- The first zero-padding is necessary since we have to enlarge the size of the image to 260×260 (divisible by 5). Thus, $dom_x = dom_y = 52$ pixels.
- Then we apply the second zero-padding to the image and we have $N'' = M'' = 300$ pixels and each sub-domain is $dom'_x = dom'_y = 92$ pixels.

3 Second Part - Deconvolve the Blurred Image

In the second part of the software, you can set the necessary parameters for running the deconvolution process. Click on "`Deconvolution`" panel and continue with the following steps (Fig. 6).

3.1 Sky Background

Set the sky background ($b = 10.0$) in the upper-left numeric field of the panel. In this example, the background is considered to be a constant in the whole image and must not have been subtracted from the input data (this is mandatory for our algorithms). Alternatively, you can load a FITS file in which the background is stored.

3.2 Read-Out Noise and Gain

Set the Read-Out Noise value ($RON = 13$ electrons) in the numeric field under the sky background and set the GAIN value to 7.5 electrons/ADU. According to Snyder et al.[3], the variance of the Read-Out noise is added to the input image and to the background in order to approximate the Gaussian noise with a suitable Poisson noise.

3.3 Restoration Method

Now you can choose the deconvolution algorithm between **Richardson-Lucy (RL)** and **Scaled Gradient Projection (SGP)**. Both algorithms

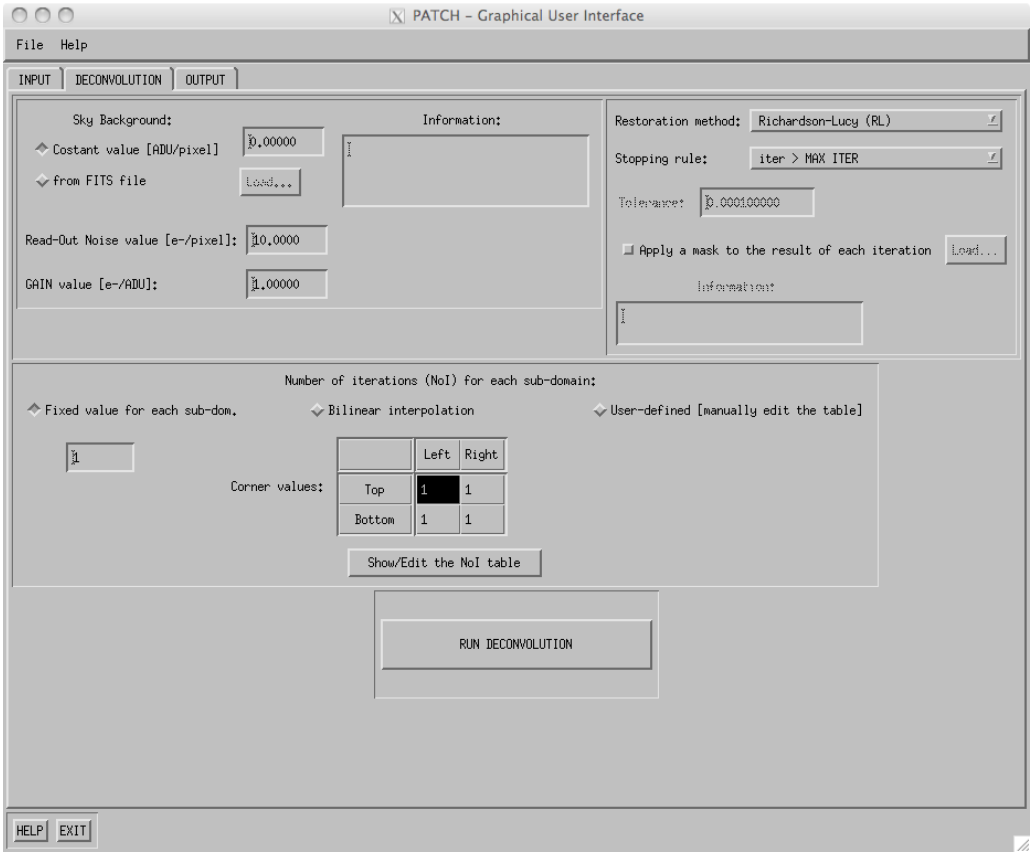

Figure 6: The deconvolution panel of the software `Patch` .

Name	Stopping rule
#1: iter > MAX ITER	if (k > MAX_ITER) then stop the iterations
#2: distance of successive iterations	$(\|\|f^{(k)} - f^{(k-1)}\|\| < tol * \|\|f(k)\|\|$ OR #1
#3: data-fidelity function	$(\|J(f^{(k)}) - J(f^{(k-1)})\| < tol * \|J(f^{(k)})\|$ OR #1
#4: discrepancy principle	$(2/N * J(f^{(k)}) < tol$ OR #1

Table 1: Stopping rules available for the SGP algorithm. Stopping rule #1 is also available for RL.

take into account the boundary effects correction, the first is well-known in astronomy, while the latter is a first-order method developed by S. Bonettini et al. ([4]).

The two algorithms are iterative and, except in the case of point-wise objects, early stopping of the iterations is required for obtaining sensible solutions of the restoration problem. Four stopping rules are available for the SGP algorithm, as shown in the Tab. 1. The stopping rule #1 is also available for the RL method.

In the table, $f^{(k)}$ is the reconstructed object at k-th iteration, $J(f^{(k)}$ is the *data-fidelity function* (or Kullback-Leibler divergence), N is the total number of pixels of f, and *tol* is a user-defined tolerance of the stopping rules #2 – #4. Choose #3 in order to push the algorithm to convergence. In the case of #4, called *Discrepancy principle*, *tol* should be a given number close to 1. You can find more information on [5].

Indeed, since this example concerns the deconvolution of a star cluster, we want to push the algorithm to convergence. In the case of RL, this means to set a high number of total iterations (10^4 for example) and to wait for their end, while in the case of SGP you can choose stopping rule #3 with a small tolerance. Assuming you want to test the SGP algorithm, select it and then set the "data fidelity function" rule and write "1e-7" (i.e. 10^{-7}) within the tolerance numeric field.

3.4 Max. Number of Iterations

Regardless of the chosen stopping rule, the algorithm will stop the iterations when they reach the number set in this field. For this reason, you must set the maximum number of iterations of the algorithm in the middle part of this panel of the GUI (see Fig. 6).

You can choose a fixed number of iterations for all sub-domains (left), a bi-linear interpolation (middle) or insert user-defined values (right).

Remark. The four corners refer to the definition of sub-domains as de-

scribed in the first part of this tutorial.

For this example (and for the SGP choice of the previous step) set 1000 iterations in the numeric field "`Fixed value for each sub-dom.`".

3.5 Run Deconvolutio Button

By pressing the button at the bottom of this panel, the algorithm is launched and the program waits for its end. A message printed in the IDL prompt informs you when a sub-domain is done and the following begins the deconvolution (the elapsed time is also shown).

Depending on the working dimensions, on the number of sub-domains, and, above all, on your computer, it might take from a few seconds to several minutes to obtain the final result.

A mosaic of the reconstructed sub-domains forms the output image.

4 Third Part - Results

In the last panel (shown in Fig. 7) you can display and save the results of the deconvolution process.

4.1 Reconstructed Object

You can save the reconstructed object as a FITS file. The header of the file (that you can also read by clicking "`Display FITS header`") contains all the parameters of the deconvolution, such as number of iterations for each sub-domain, executing time, algorithm and stopping rule used, etc.

4.2 Visualization Options

Right click on the image to open a pop-up window. Within the window you can change the zoom, the scale and the color map of the image. Press "Apply and close" to apply your settings.

4.3 Residual Image

You can display the so-called **residual image** in the same graphical area of the result. During the deconvolution the residual image is computed as follows:

$$R_i^{(k)} = \frac{g_i - (K_i * f_i^{(k)} + b_i)}{\sqrt{K_i * f_i^{(k)} + b_i}}$$

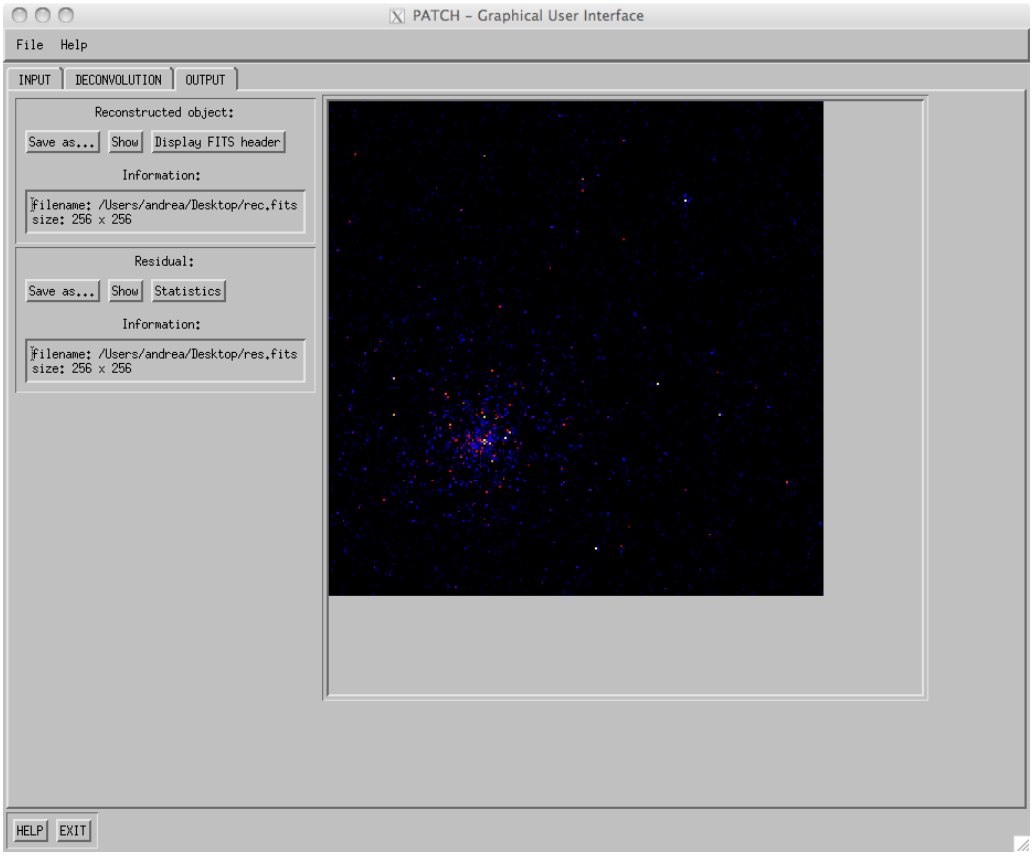

Figure 7: The last panel of the software `Patch` . Here you can visualize and save the results of the deconvolution.

where g_i is the part of the input image corresponding to the i-th sub-domain, similarly K_i is the PSF, b_i the background, $f_i^{(k)}$ the reconstructed object at the k-th iteration and $*$ is the convolution operator. All the other operations are intended pixel-by-pixel.

As for the reconstructed object, at the end of the iterations, a mosaic of the $R_i^{(k)}$ images forms the final residual image R.

4.4 Visualization Options (Residual)

Again, right click on the image to open a pop-up window and choose the visualization options. However, when you close this window, by default the reconstructed object is shown in the graphical area; therefore if you want to display again the residual image, you must click another time on the "`show`" button.

4.5 Statistics

Click on this button to show a separate window in which, through the Astrolib routine `HISTOGAUSS`, the histogram of the distribution of the residual image is shown. In addition, mean value and standard deviation are also shown. Note that the "perfect" reconstruction has a mean value equal to zero and variance equal to 1.

5 Final Remarks

All the parameters set in the GUI can be saved in a `.sav` file and can be used later. Click on the "`Save parameters...`" item in the "`File`" menu, choose a filename and press "`OK`".

To load a set of parameters that you previously saved, click on "`Load parameters...`" item in the "`File`" menu and select your file. Then all parameters are set in the GUI (both the panels) and you can see the input image already loaded within the graphical area.

If you want to deconvolve an image and to obtain the output starting from an already saved set of parameters (for example stored in the "`mytest.sav`" file) you can run the program without load the GUI simply by entering the following command at the IDL prompt:

```
PATCH > patch, 'mytest.sav'
```

6 Contacts

Comments, corrections and suggestions about the content of this tutorial are always welcome. For any question please contact:

- Andrea La Camera (DIBRIS, now with Teiga srls) [andrea AT teiga.it]
- Laura Schreiber (INAF) [laura.schreiber AT inaf.it]

Document created: 2014/10/20 - Andrea La Camera.
Last Update: 2023/11/02 - Andrea La Camera

References

[1] Andrea La Camera et al. "A method for space-variant deblurring with application to adaptive optics imaging in astronomy". In: *Astronomy and Astrophysics* 579 (2015), A1. DOI: 10.1051/0004-6361/201525610. URL: http://www.aanda.org/10.1051/0004-6361/201525610.

[2] Andrea La Camera and Laura Schreiber. "Deblurring of post-adaptive optics images". In: *SPIE Newsroom* (Sept. 2015). ISSN: 18182259. DOI: 10.1117/2.1201508.006107. URL: http://www.spie.org/x115428.xml.

[3] D. L. Snyder et al. "Compensation for readout noise in CCD images". In: *J. Opt. Soc. Am.* A-12 (1995), pp. 272–283. DOI: 10.1364/JOSAA.12.000272.

[4] S. Bonettini, R. Zanella, and L. Zanni. "A scaled gradient projection method for constrained image deblurring". In: *Inverse Problems* 25.1 (2009), p. 015002.

[5] M. Bertero et al. "A discrepancy principle for Poisson data". In: *Inverse Problems* 26 (2010), p. 10500.