

User documentation v1.0

of

SPICE Imaging Tool (SPICE-IT)

Mission: Solar Orbiter

Instrument: SPICE (spectrometer)

Project: Software development

Purpose: Quick view of SPICE data

Release: 2025

Created by: IAS team (Orsay, France)

Developer: David Picard

Supervisors: Frédéric Auchère, Éric Buchlin, Stéphane Caminade, Susanna Parenti, Alfred Voyeux

Last update: 2025-02-14 by David Picard

Contents

- I. Introduction3
 - Software description.....3
 - Installation3
 - Initial configuration.....3
- II. Usage & Features.....4
 - Opening SPICE data.....4
 - Global display.....4
 - General4
 - Window area.....5
 - Left menu9
 - Menu bar 10
 - Footer..... 10
 - Summary on dimension browsing (x, y, λ, t) 10
 - Throw x 11
 - Throw y 11
 - Throw λ 12
 - Throw t 12
 - Fit maps..... 12
- III. Contact & support 13
 - Main IAS contacts 13

I. Introduction

➤ Software description

Thanks to this software, you will be able to open SPICE data files (FITS level 2) and have a quick view of the content in it. Files can be opened one at a time. The content of the file will be shown, typically there are several windows (areas on the SPICE detectors i.e. over the wavelength dimension, related to known chemical elements, e.g. O III, C II, ...). For each window you will be able to navigate through the dimensions (x, y, λ, t).

Users aimed are anybody who wants to open data file from the SPICE instrument and look for interesting SPICE images, typically (x, y) , (λ, y) images or $I(\lambda)$ profiles.

The software is developed in Python and use libraries whose PyQtGraph, it is very easy to install but the user will have to install these requirements first. Everything needed is explained below.

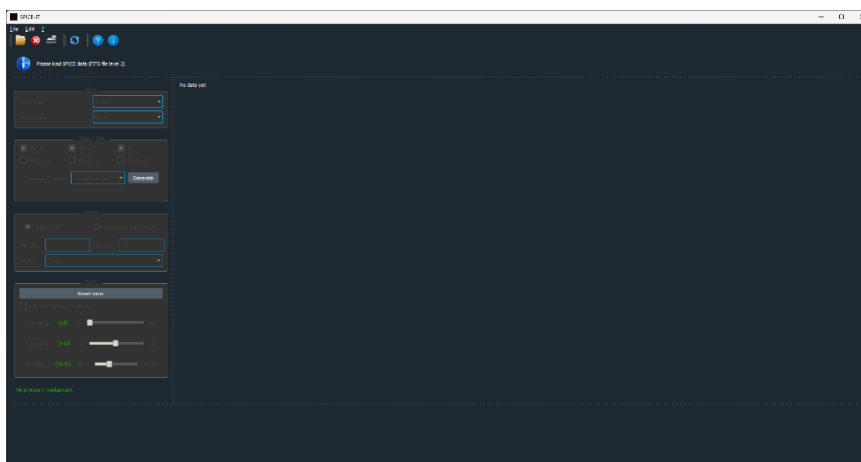
➤ Installation

- 1) Go the IAS Gitlab > SPICE projects, an access will be required (see S. Caminade contact in the [Contact & Support](#) part at the end of this document)
 - 2) Download it as a ZIP file then extract the project or use the Git environment, typically:
`git clone https://git.ias.u-psud.fr/spice/data_quicklook.git`
 - 3) Ensure Python is installed on your computer
 - 4) Install the requirements specified in requirements.txt
 - 5) Run python run.py
- N.B.: Steps 3, 4, 5 can be done easily thanks to the Pycharm software ([free version downloadable](#))

➤ Initial configuration

The software uses a configuration by default that is adapted for most of computer displays. However, advanced users can change some of the parameters in the *settings/const.py* file. E.g. according to your screen, you can adjust the display by updating *window_width* and *window_height* variables. If display is not satisfactory after editing the file, the user should switch the const.py file back on.

If the software opens with a large empty area with a disabled menu on the left, then you can go to the [Usage & Features](#) part.



II. Usage & Features

➤ Opening SPICE data

The first step is to open a compatible SPICE data file, you need to ensure that:

- It is data from SPICE instrument
- It has ".fits" extension
- It belongs to level 2 data files

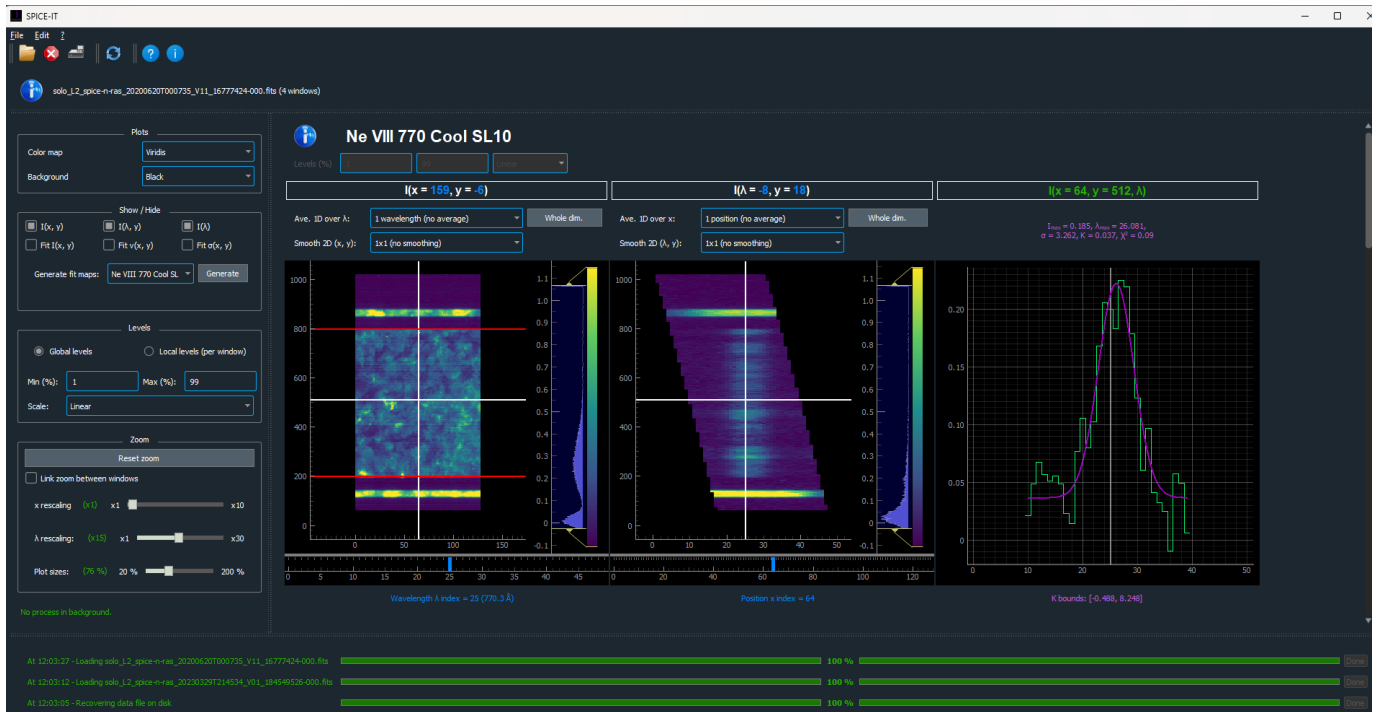
The file must contain usual headers in SPICE data (e.g. NAXIS, STUDYTYP, CDELT1, ...) and windows with the related data cubes. The software can handle several study types: raster, single exposure (full spectrum) and time series (sit and stare). Typically, name of data files can be:

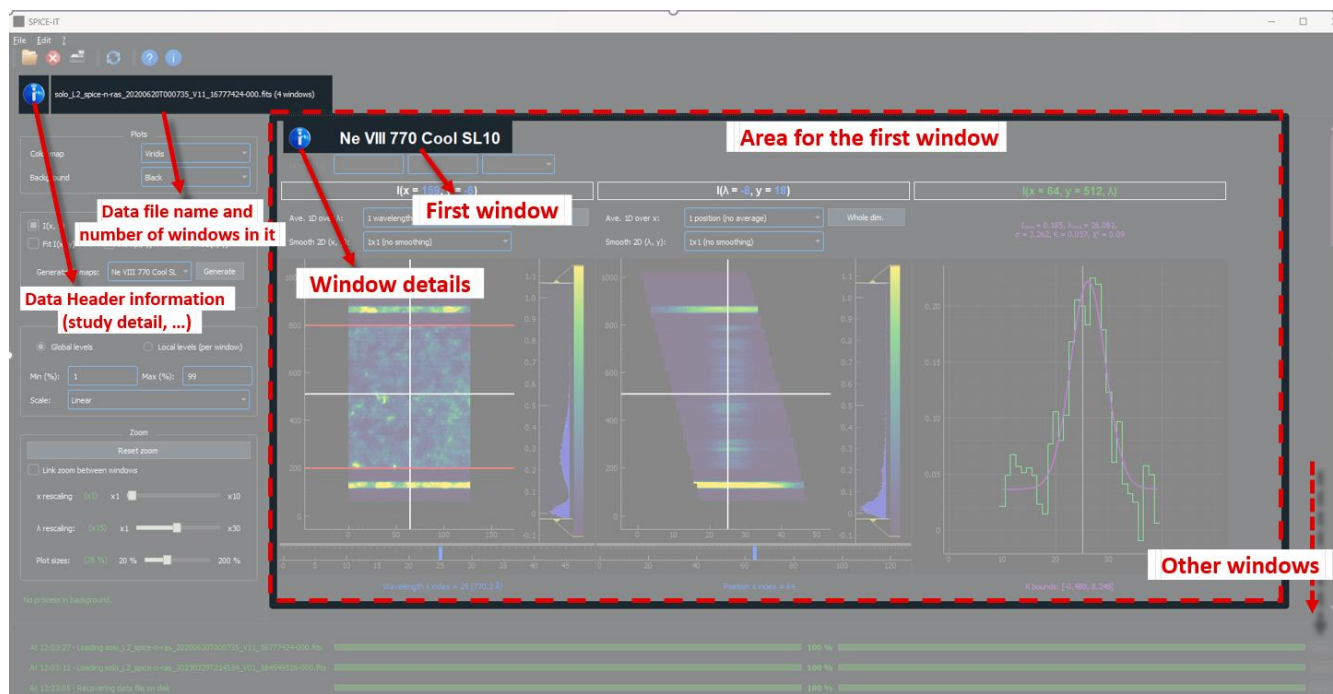
solo_L2_spice-n-ras_20230329T214534_V01_184549526-000.fits (raster)
solo_L2_spice-n-sit_20210420T214022_V05_50331906-000.fits (sit-and-stare)
solo_L2_spice-w-exp_20210912T031019_V09_67109110-000.fits (single exposure)

➤ Global display

General

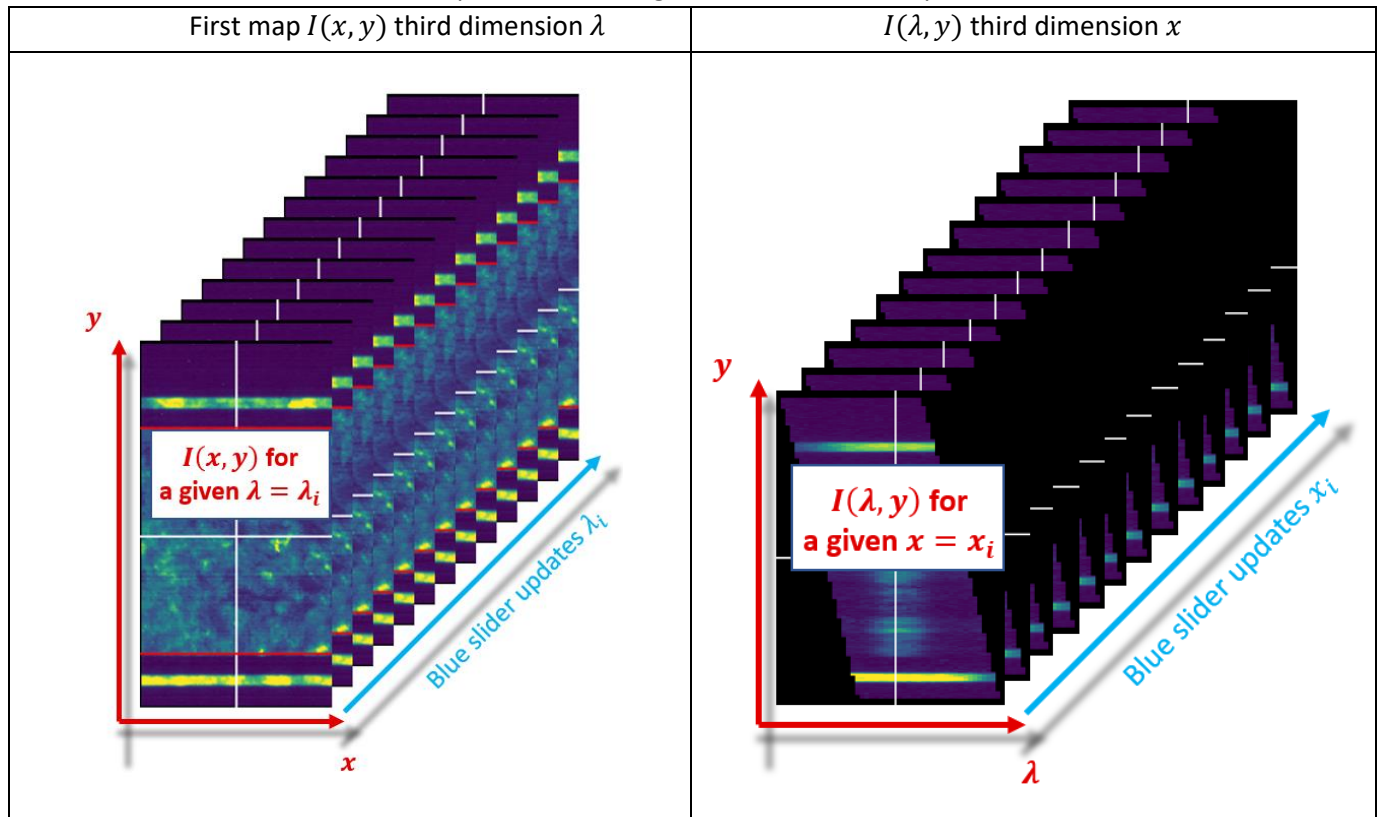
Once loaded, you are supposed to have a screen like the one below. The next screenshots will explain every part of the display.



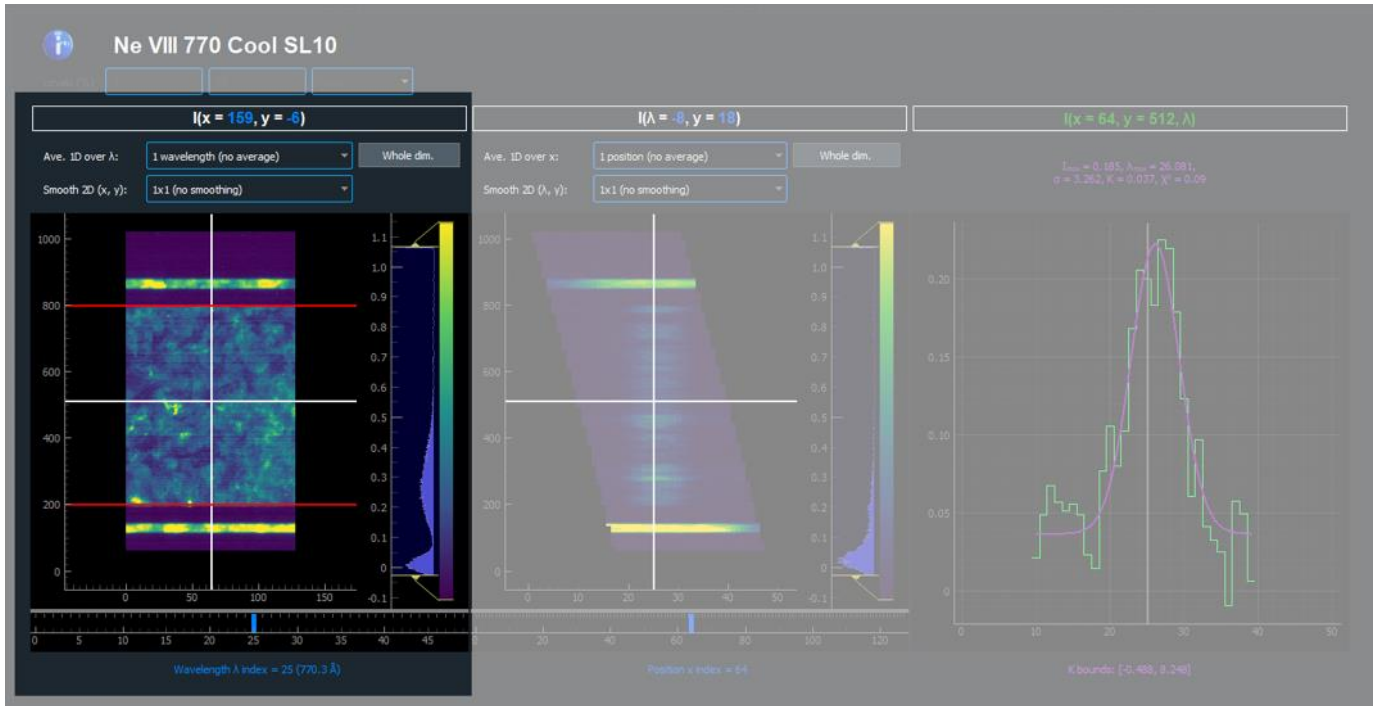


Window area

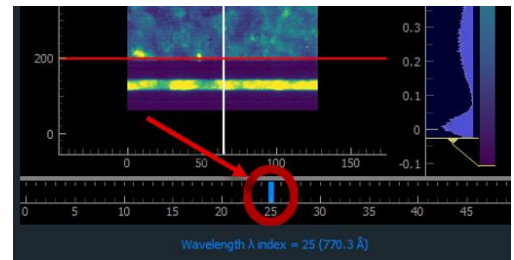
Now let's see the window area, this block (dashed block above) is duplicated for each existing window found in the loaded data. By using the scroll on the very right side, you will be able to navigate through the other windows. Let's see the basics, there are two maps and one histogram. The two first maps can be seen as data cubes:



$I(x, y)$ map



On this first map, the intensity is displayed in the (x, y) plane for a given λ . The blue slider below allows to update the λ index, the map will update dynamically according to the selected λ .



On top (in the title $I(x, y)$), the blue coordinates update according to the mouse position in the map. If a pixel exists, intensity is also displayed.

$$\text{e.g. } I(x = 10, y = 12) = 0.5 \text{ W/m}^2/\text{sr/nm}$$

Average 1D over λ

Option "Ave. 1D over λ " will do an average over the λ dimension. It means an average between several maps (related to the other λ indexes) will make new maps (x, y) which are averaged maps. The selectable number in the list allow to specify how many maps must be averaged together, as a result:

- 1 wavelength: no average, original data cube is used
- 2 wavelengths: average between first and second map gives an averaged map, third and 4th ones give another one, and so on
- N wavelengths: the number of maps in result is equal to (initial λ size) / N
- A shortcut "Whole dim" allow to select the greatest N available i.e. $N = \text{initial } \lambda \text{ size}$ which leads to one unique averaged map of all the λ_i "slices"

Once this option is selected, the whole data cube is affected, it means averages are performed on all the maps " M_i ": $I(x, y, \lambda_i)$

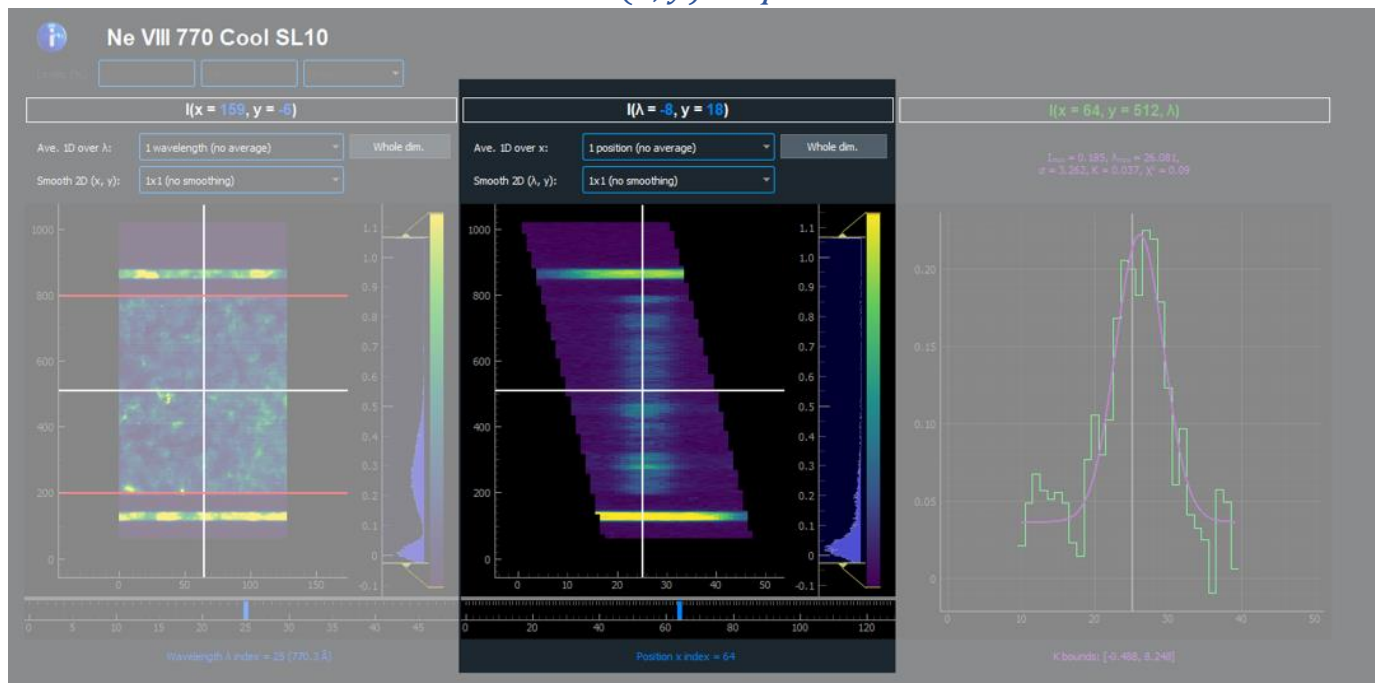
Smooth 2D (x, y)

Option "Smooth 2D over (x, y)" will do a smooth on the map itself i.e. on (x, y), each pixel is an average of its pixel neighbors in 2 dimensions. The list allows to define how many neighbors we need to consider, that's why you can select a kernel $N \times N$ that means a squared kernel where the centered position will be the average of the other neighbors in the kernel. Again, once this option is selected, the whole data cube is affected, it means smooths are performed on all the maps " M_i ": $I(x, y, \lambda_i)$

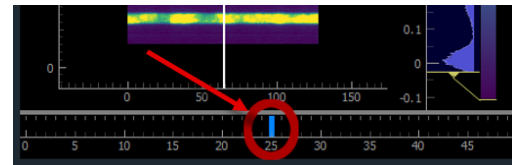
N.B.: Both average 1D and smooth 2D can't be performed at the same time. Using one of them will disable the other one.

You will find a tilted histogram that represents the intensity distribution on the right side of the maps, you can change the thresholds (thanks to the yellow bar). This is the level adjustment by "intensity values".

$I(\lambda, y)$ map



On this first map, the intensity is displayed in the (x, y) plane for a given λ . given λ . The blue slider below allows to update the λ index, the map will update dynamically according to the selected λ .



On top (in the title $I(x, y)$), the blue coordinates update according to the mouse position in the map. If a pixel exists, intensity is also displayed.

$$\text{e.g. } I(x = 10, y = 12) = 0.5 \text{ W/m}^2/\text{sr/nm}$$

Average 1D over λ

Option “Ave. 1D over λ ” will do an average over the λ dimension. It means an average between several maps (related to the other λ indexes) will make new maps (x, y) which are averaged maps. The selectable number in the list allow to specify how many maps must be averaged together, as a result:

- 1 wavelength: no average, original data cube is used
- 2 wavelengths: average between first and second map gives an averaged map, third and 4th ones give another one, and so on
- N wavelengths: the number of maps in result is equal to (initial λ size) / N
- A shortcut “Whole dim” allow to select the greatest N available i.e. $N = \text{initial } \lambda \text{ size}$ which leads to one unique averaged map of all the λ_i “slices”

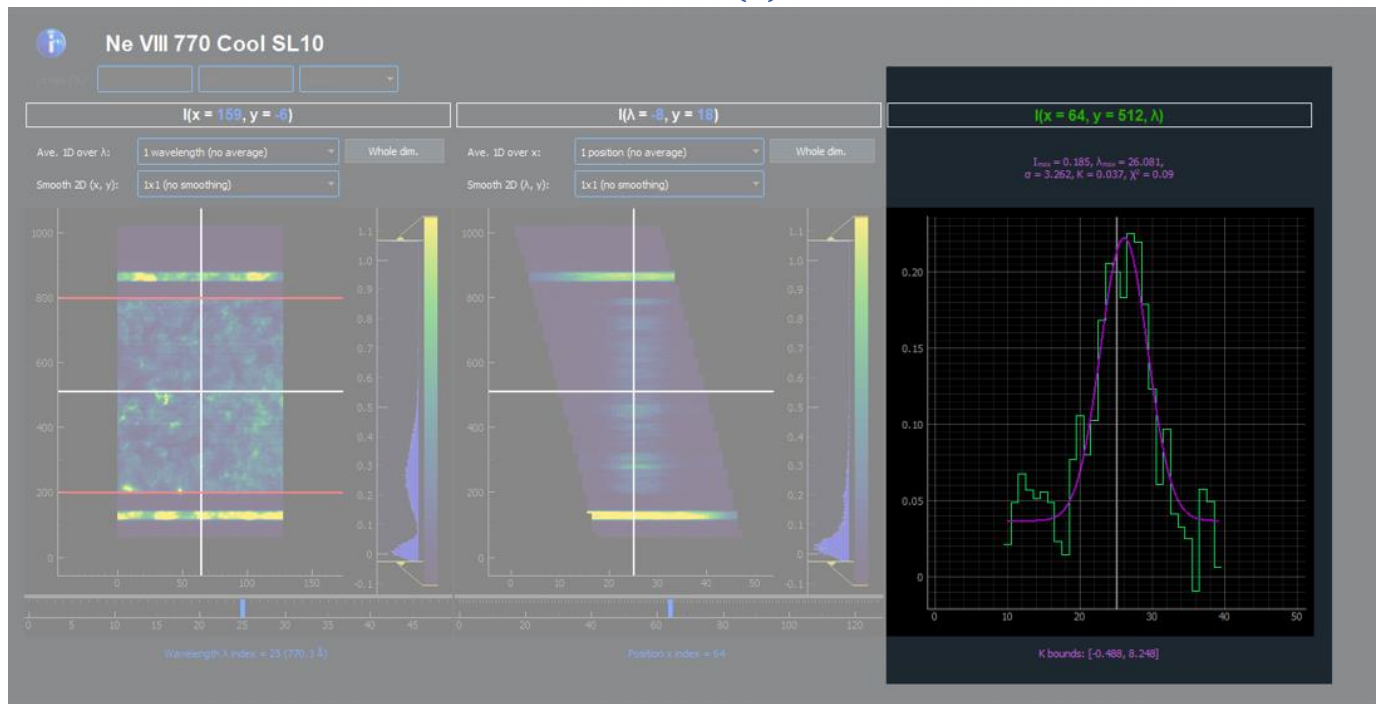
Once this option is selected, the whole data cube is affected, it means averages are performed on all the maps " M_i ": $I(x, y, \lambda_i)$

Smooth 2D (x, y)

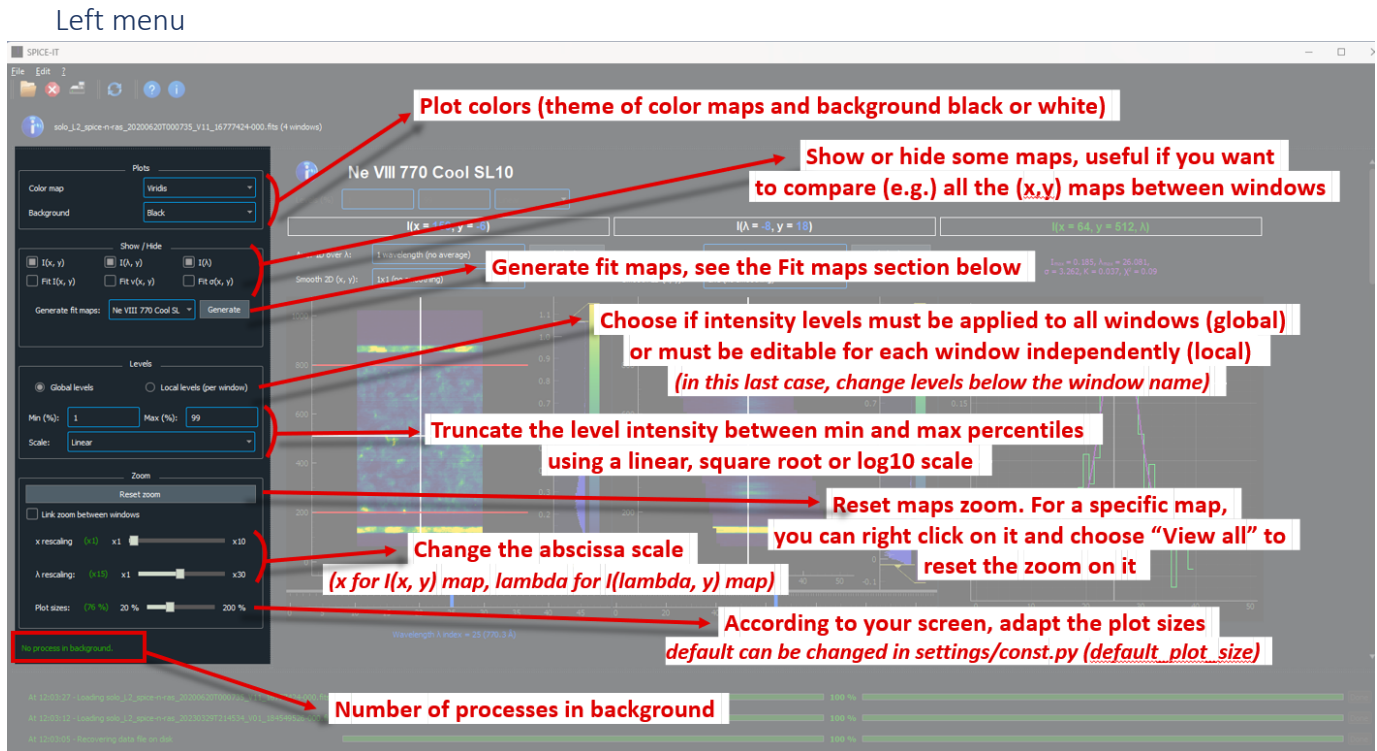
Option “Smooth 2D over (x, y) ” will do a smooth on the map itself i.e. on (x, y) , each pixel is an average of its pixel neighbors in 2 dimensions. The list allows to define how many neighbors we need to consider, that’s why you can select a kernel $N \times N$ that means a squared kernel where the centered position will be the average of the other neighbors in the kernel. Again, once this option is selected, the whole data cube is affected, it means smooths are performed on all the maps " M_i ": $I(x, y, \lambda_i)$

N.B.: Both average 1D and smooth 2D can’t be performed at the same time. Using one of them will disable the other one.

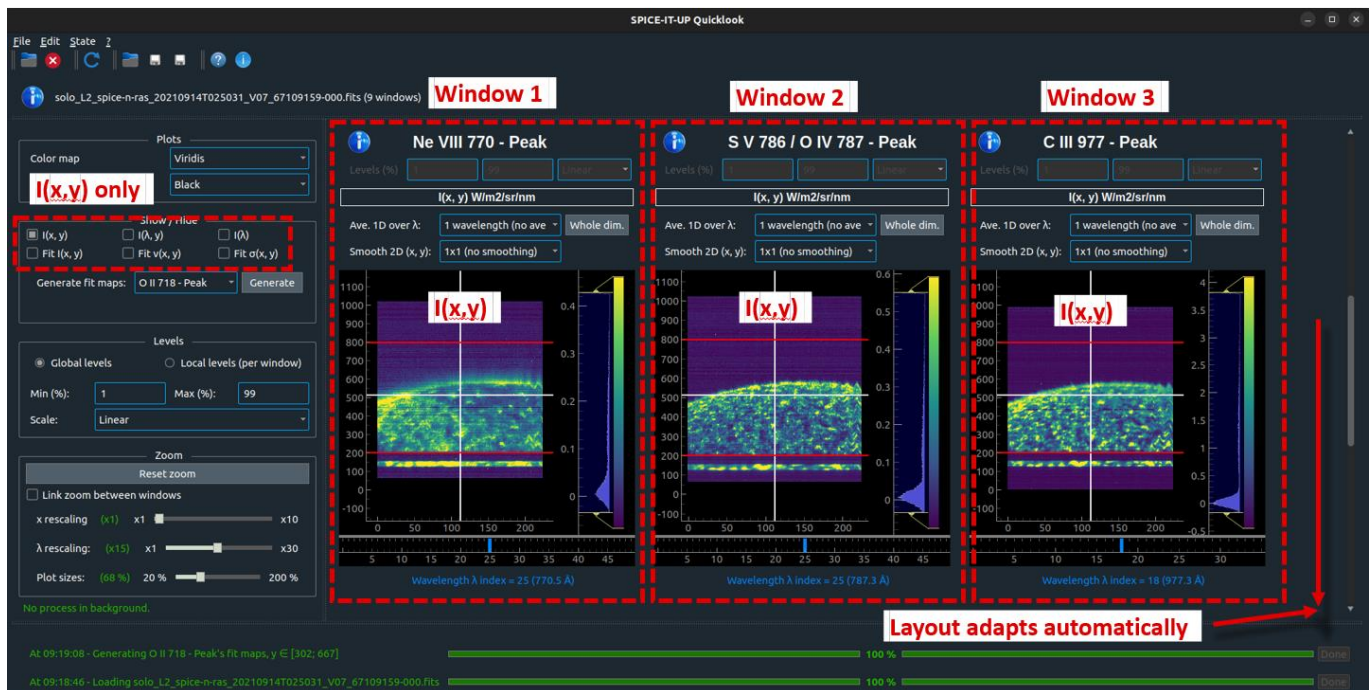
Profile $I(\lambda)$



This plot shows the intensity vs the wavelength i.e. $I(\lambda)$ for a given x and y . The data is represented as a histogram (in green), a classic Gaussian fit (in purple) is applied to it. You can retrieve the Gaussian parameter values above the plot.



Let's see an example where only $I(x,y)$ maps are checked in order to show them side by side on window areas:



Menu bar

Data file

We saw *File > Open data file* to open data files. There is a “recent files” feature, each time you load a file, it is stored in the recent files. Then, you can load a previous file automatically (be sure the file path still exists i.e. file didn’t move and the destination disk is mounted) by checking the list *File > Recent files*. Files are kept even after closing the app. Tip: use *Ctrl+L* to load the last one.

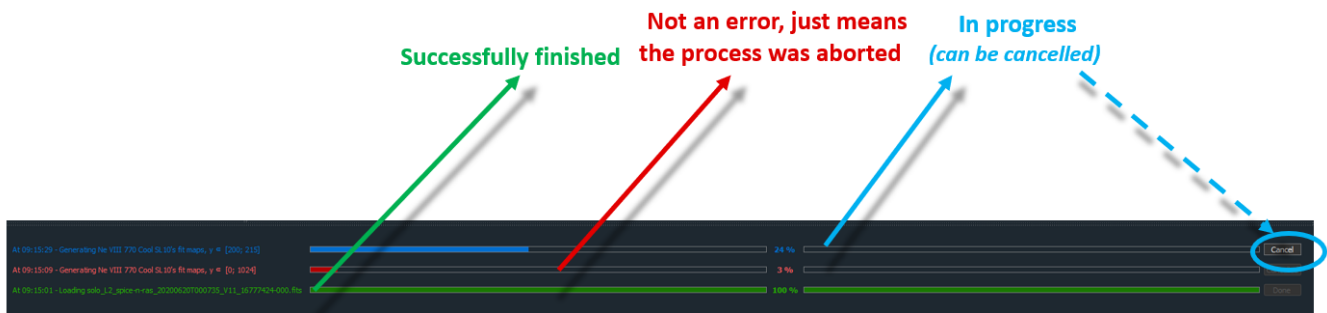
JSON state

Even better, you can store the whole app state. It will store the data file used, the parameters in the left menu, the intensity levels, averages, smooths and other modified options.

- ➔ Use *State > Save state* (*Ctrl+Shift+S*) to export the state of the app in a JSON file
- ➔ Use *State > Load state* (*Ctrl+Shift+L*) to restore the saved state

Footer

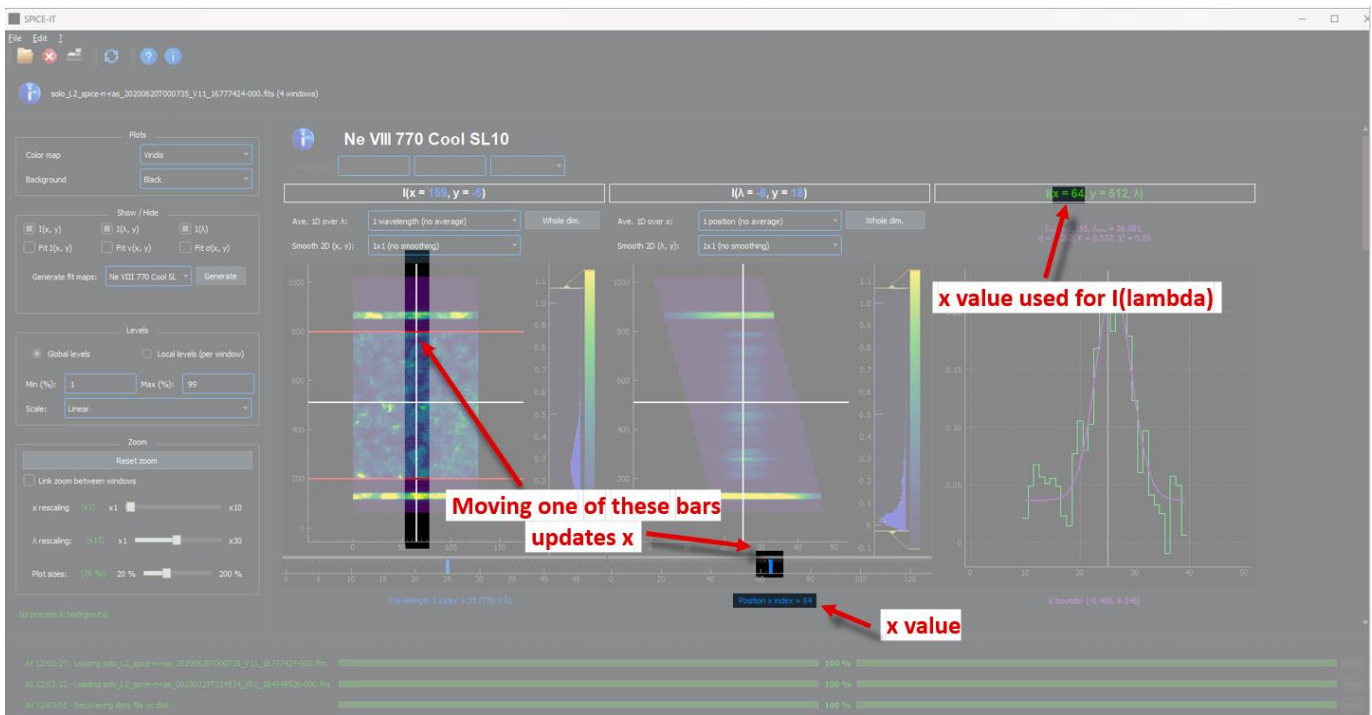
You will retrieve the processes running in background (only some of them, the main ones). Processes can be in 3 states, see below:



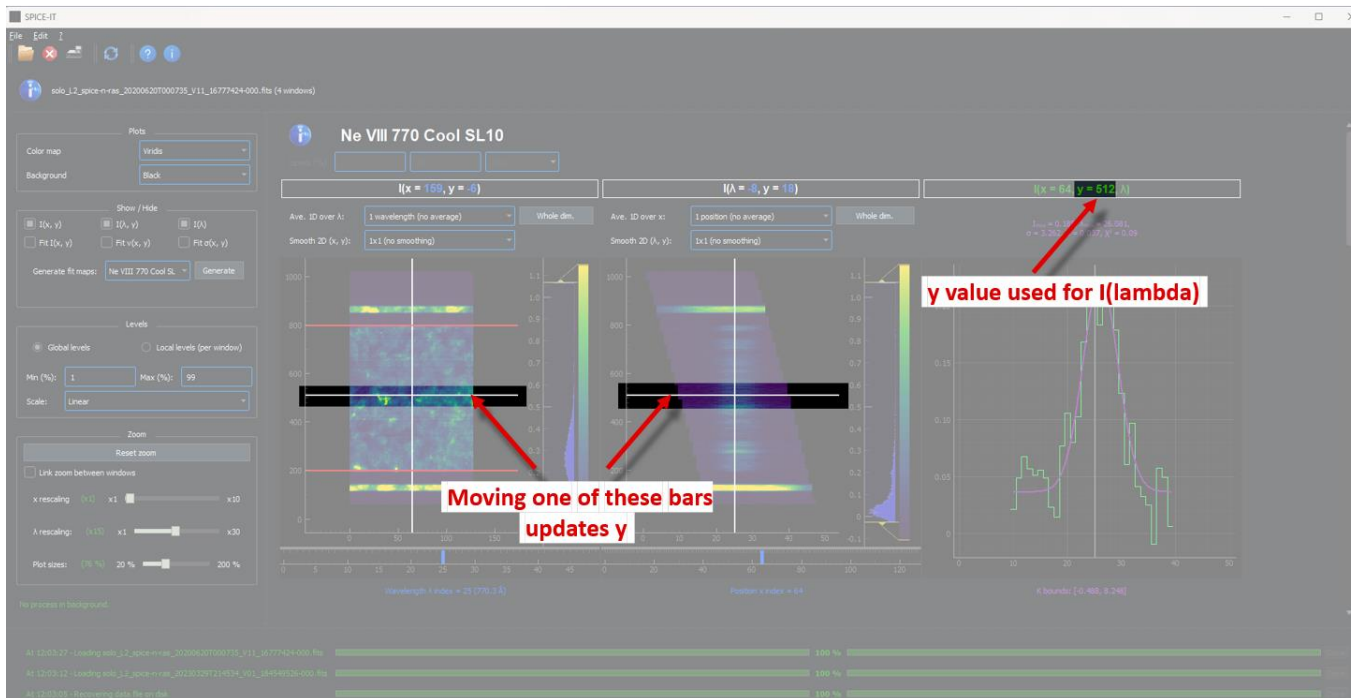
➤ Summary on dimension browsing (x, y, λ, t)

This is how you can easily navigate through the dimensions i.e. get images and plots for specific x, y, λ, t values.

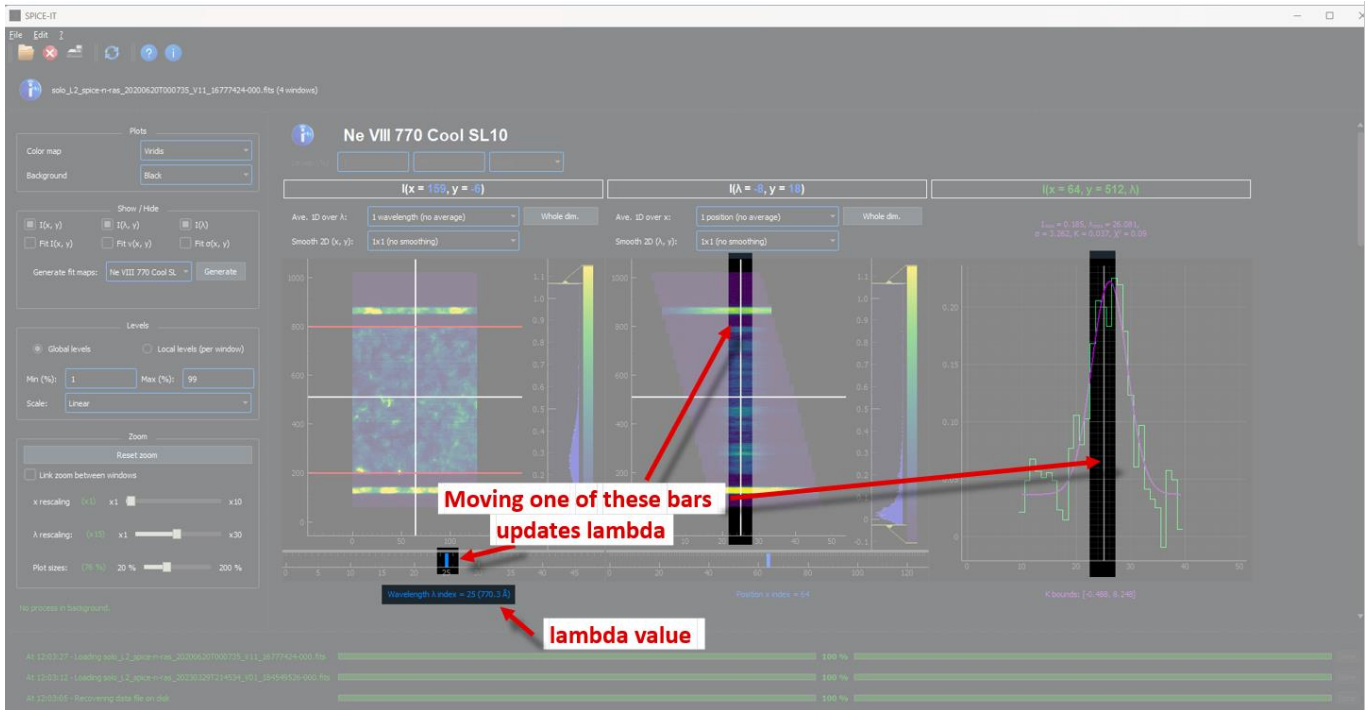
Throw x



Throw y



Throw λ



Throw t

For a raster, the time dimension of a data file is equal to 1 and you can open FITS files one at a time, this means there is no time to navigate throw. However, in a sit-and-stare file, the x dimension is 1 and there are several t indexes. In this case the x on the display is replaced by the t , this means you can't navigate throw the x dimension but you can throw the t one. As a result you never have 4 real dimensions but only 3 $\rightarrow x, y, \lambda$ or y, λ, t .

Fit maps

So far, for each window, we had 3 plots: $I(x, y)$, $I(\lambda, y)$ and $I(\lambda)$. Three other maps can be displayed for each window, these are the "Fit maps" results. For each pixel in the (x, y) map, a Gaussian fit is computed (one of them is displayed for a given x and y , this is the $I(\lambda)$ profile). Generate the fit maps in the left menu (for a specific window or for all windows) will do the Gaussian fit for every pixel in the (x, y) map. The best Gaussian fit is found each time, as a result we have Gaussian parameters for every pixel in the (x, y) map.

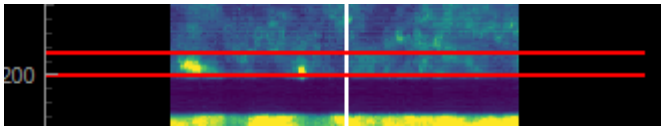
$$I(x, y) = I_{max} \exp \left[-\frac{(\lambda - \lambda_{mean})^2}{2\sigma^2} \right] + K$$

The 3 maps in result are:

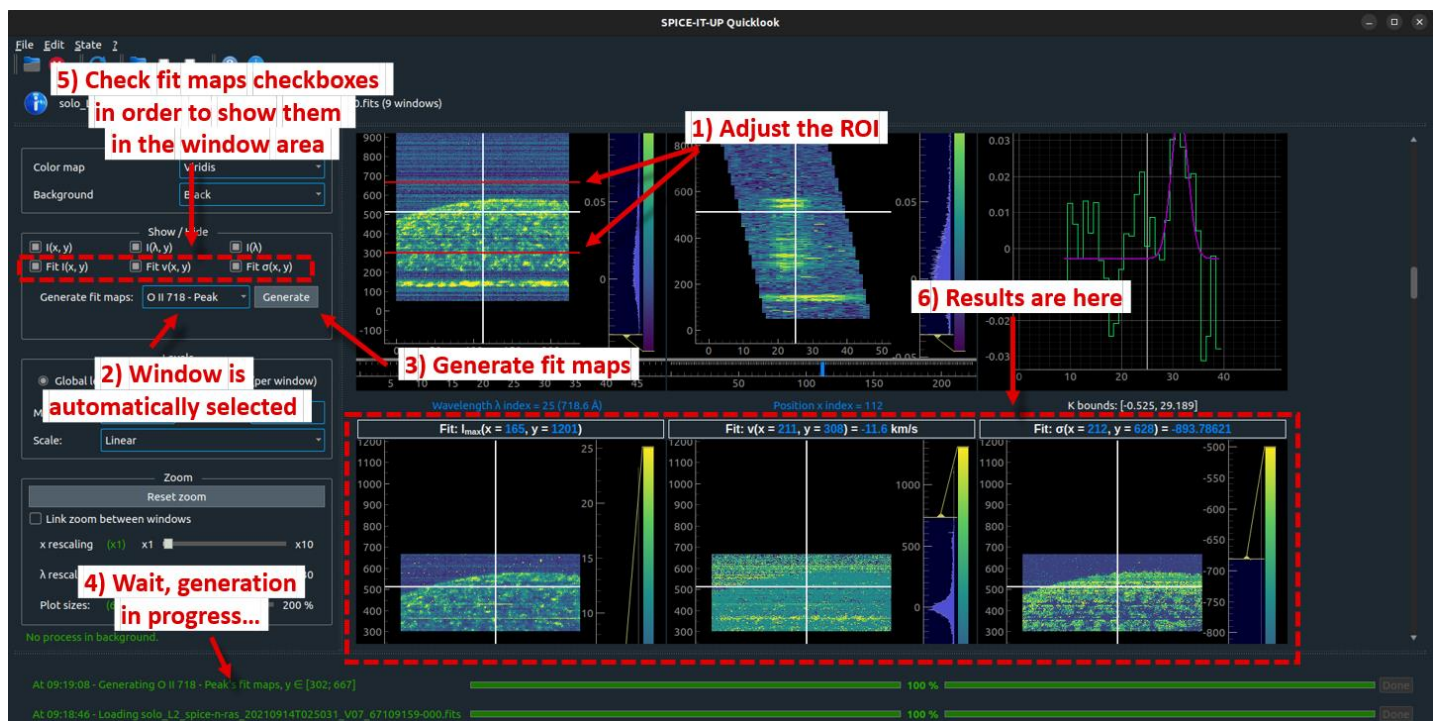
$$I_{max}(x, y) \quad ; \quad \lambda_{mean}(x, y) \text{ that is converted to velocity (Doppler formula) } v(x, y) \quad ; \quad \sigma(x, y)$$

In order to save time, you can specify a region of interest (ROI) where the fit maps will be computed. Use the red bars in the $I(x, y)$ before clicking on the "Generate fit maps" button. When bars moved, the related window is automatically selected in the left menu. This is how this looks like:

First, define the ROI:



Then, in the left menu click on “Generate fit maps”. If it takes time and if you want to generate it for a smaller region, you are free to cancel the process in the footer, then change the red bars position and generate them again. Once the process in the footer is done, make sure the Fit maps checkboxes are checked (See in the left menu, in the Show/Hide block, checkboxes “Fit [...]”). Let’s summarize in a real case:



III. Contact & support

➤ Main IAS contacts

- david.picard@universite-paris-saclay.fr (David Picard)
- stephane.caminade@universite-paris-saclay.fr (Stéphane Caminade)
- frederic.auchere@universite-paris-sud.fr (Frédéric Auchère)
- eric.buchlin@universite-paris-saclay.fr (Éric Buchlin)