



Fast turn-around reduction of NIRSpec MOS data in Datalabs

Giovanna Giardino

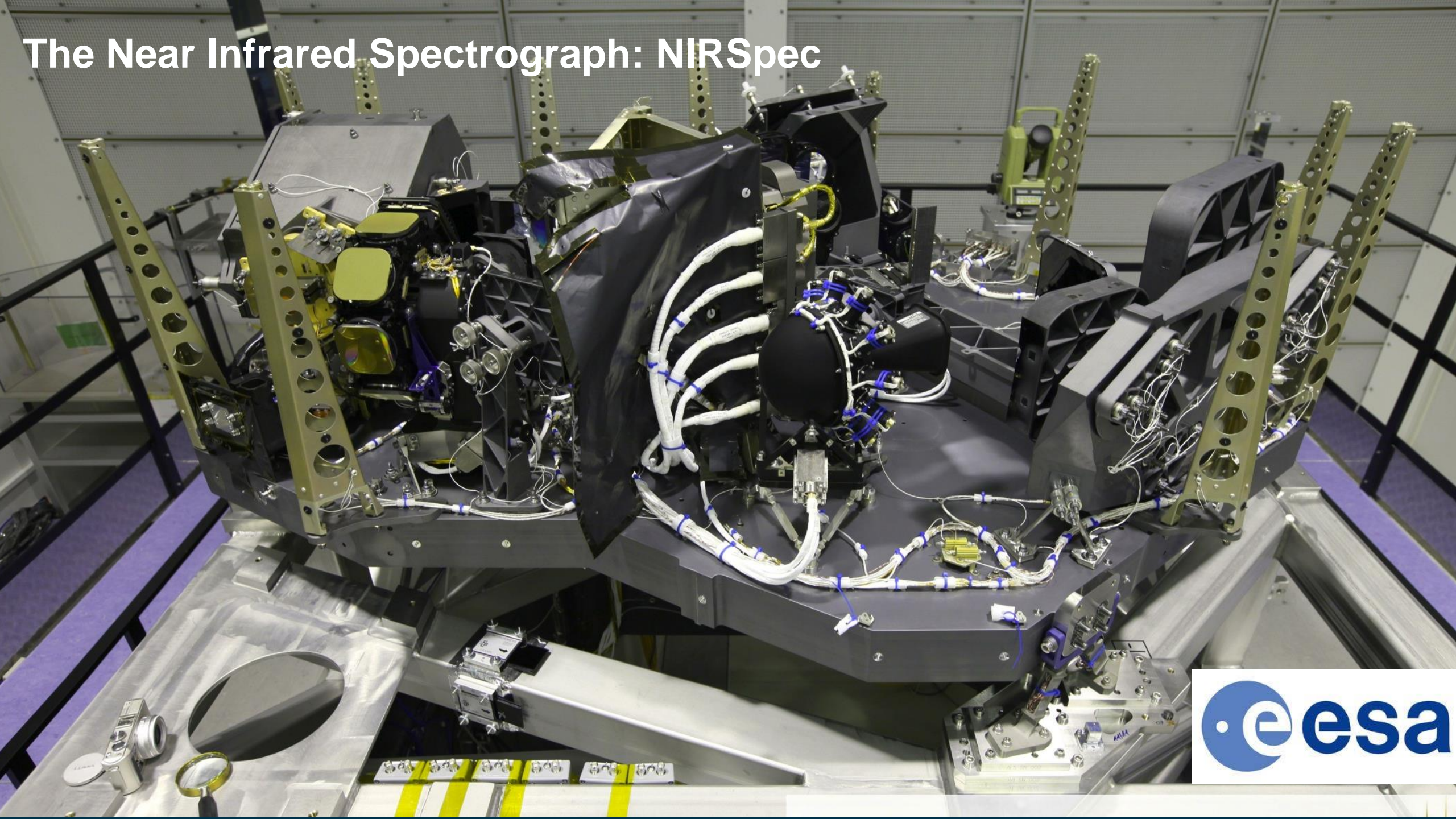
ESA UNCLASSIFIED – For ESA Official Use Only



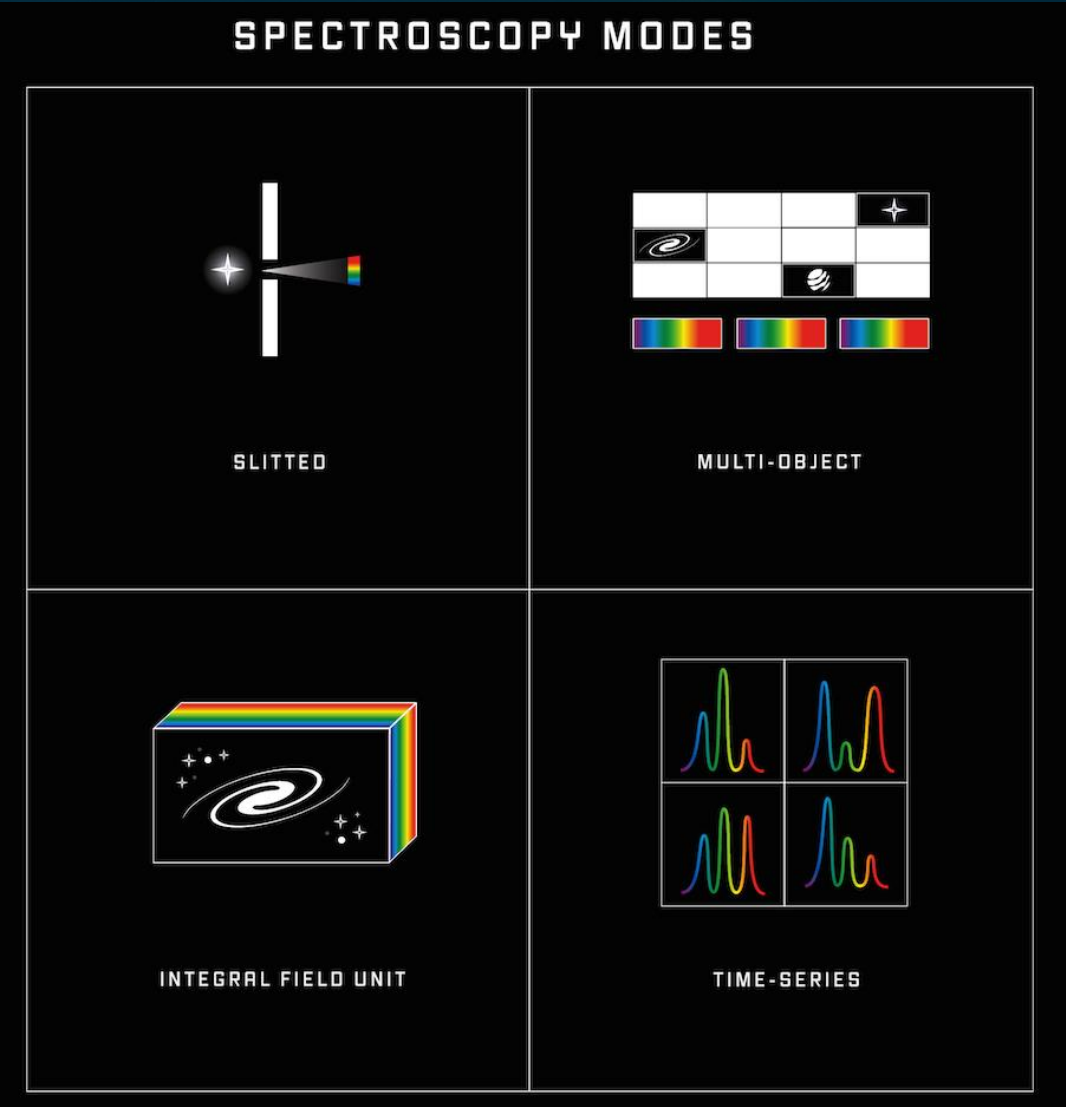
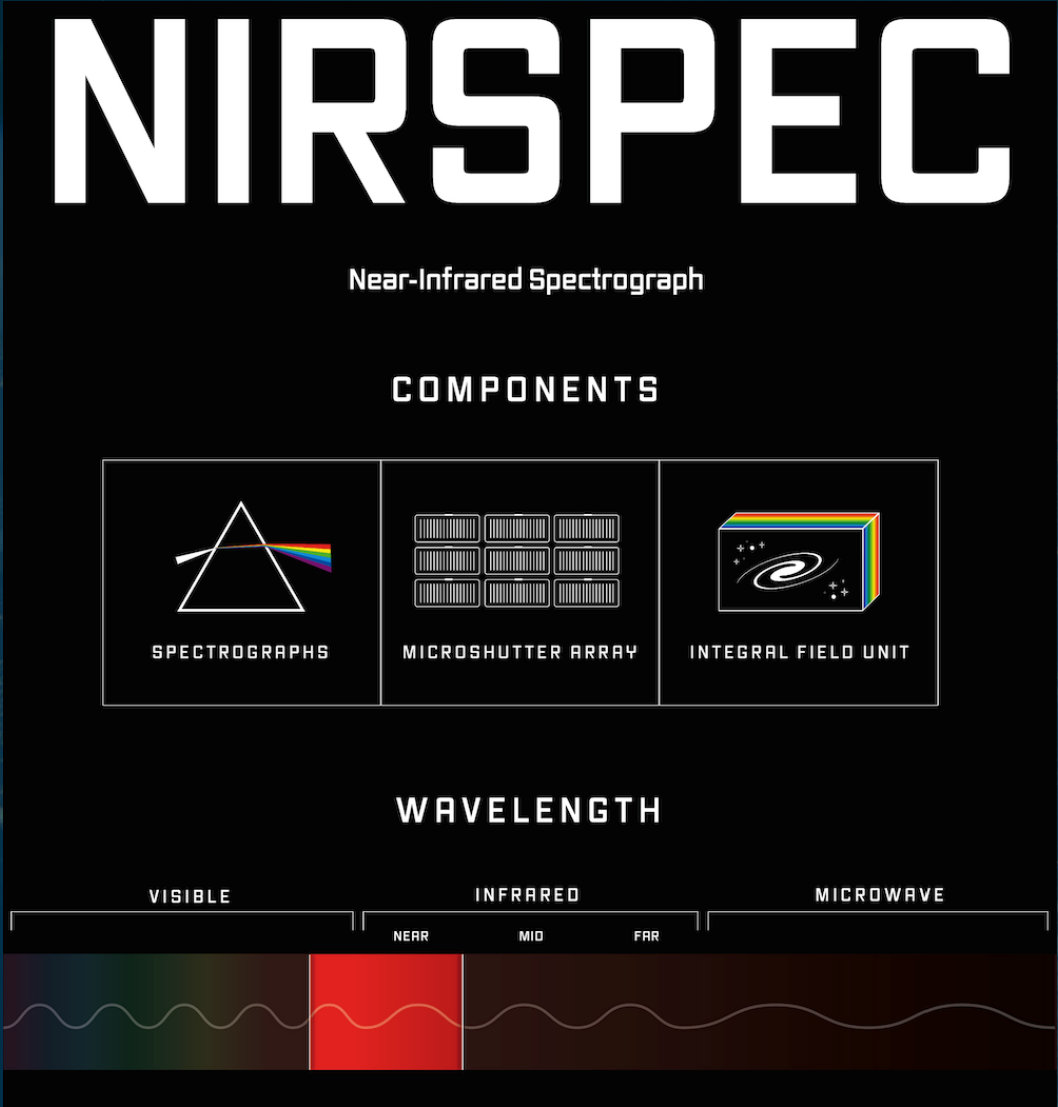
THE EUROPEAN SPACE AGENCY

Image Credit: NASA, ESA, CSA, and STScI

The Near Infrared Spectrograph: NIRSpec



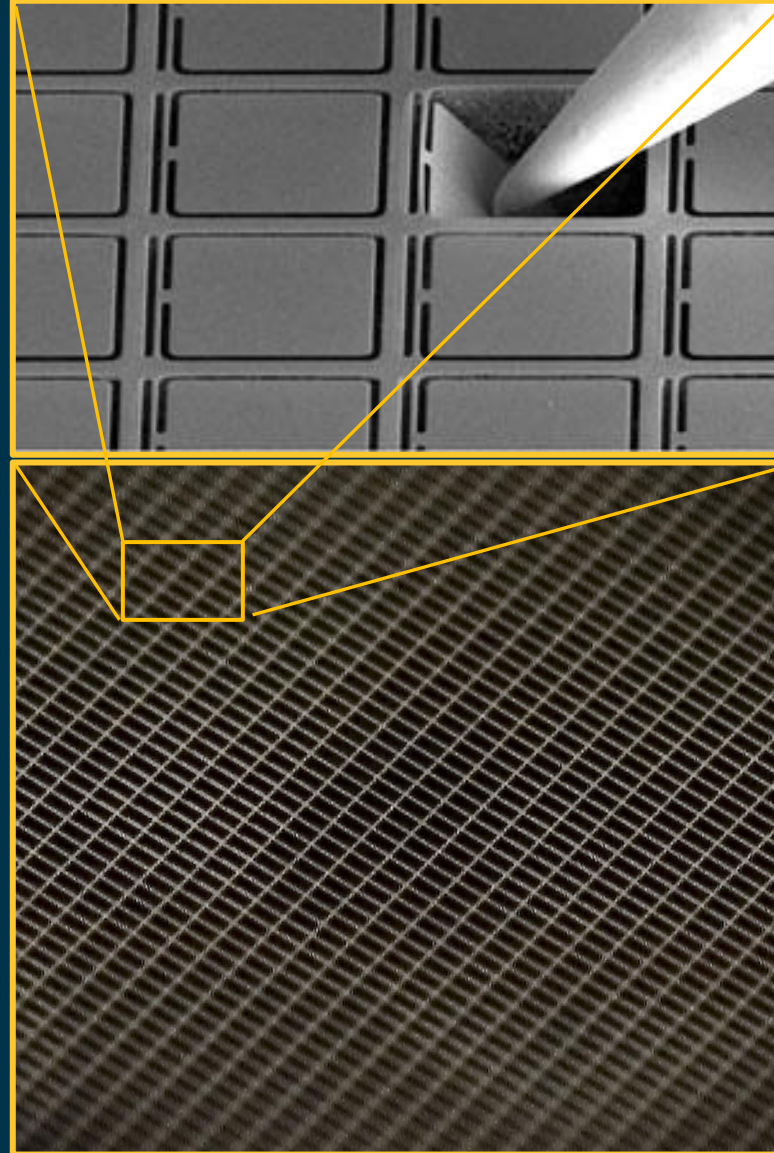
NIRSpec supports 4 different observing modes



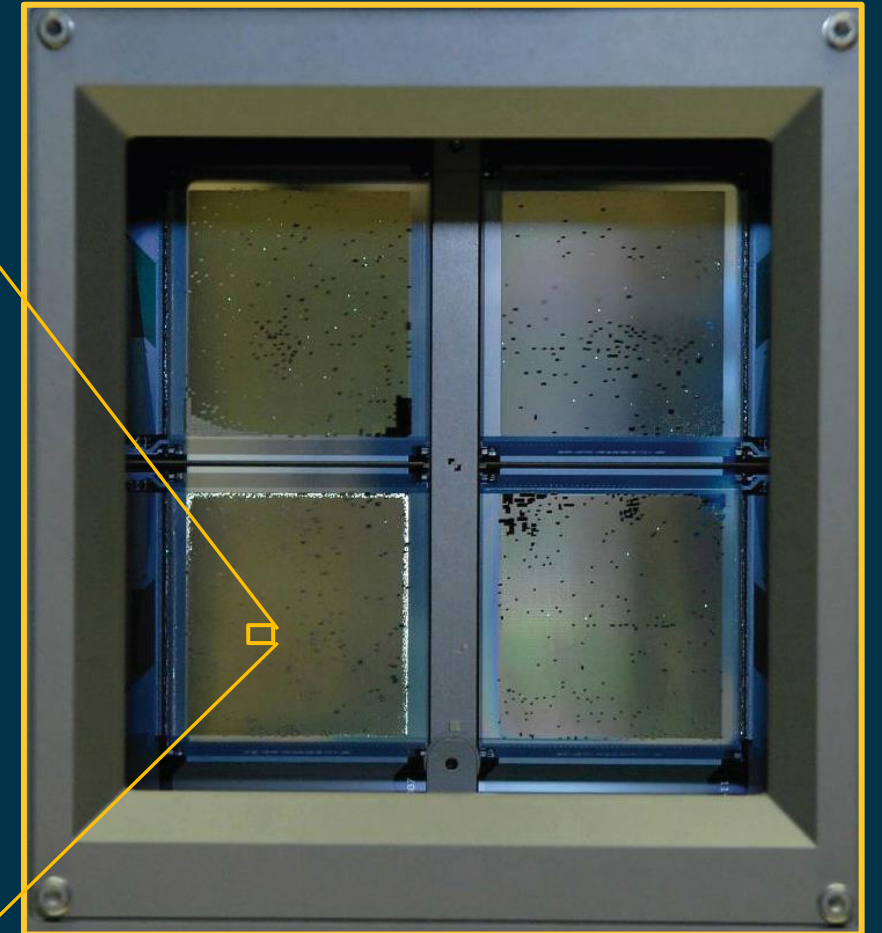
Micro Shutters Assembly of NIRSpec

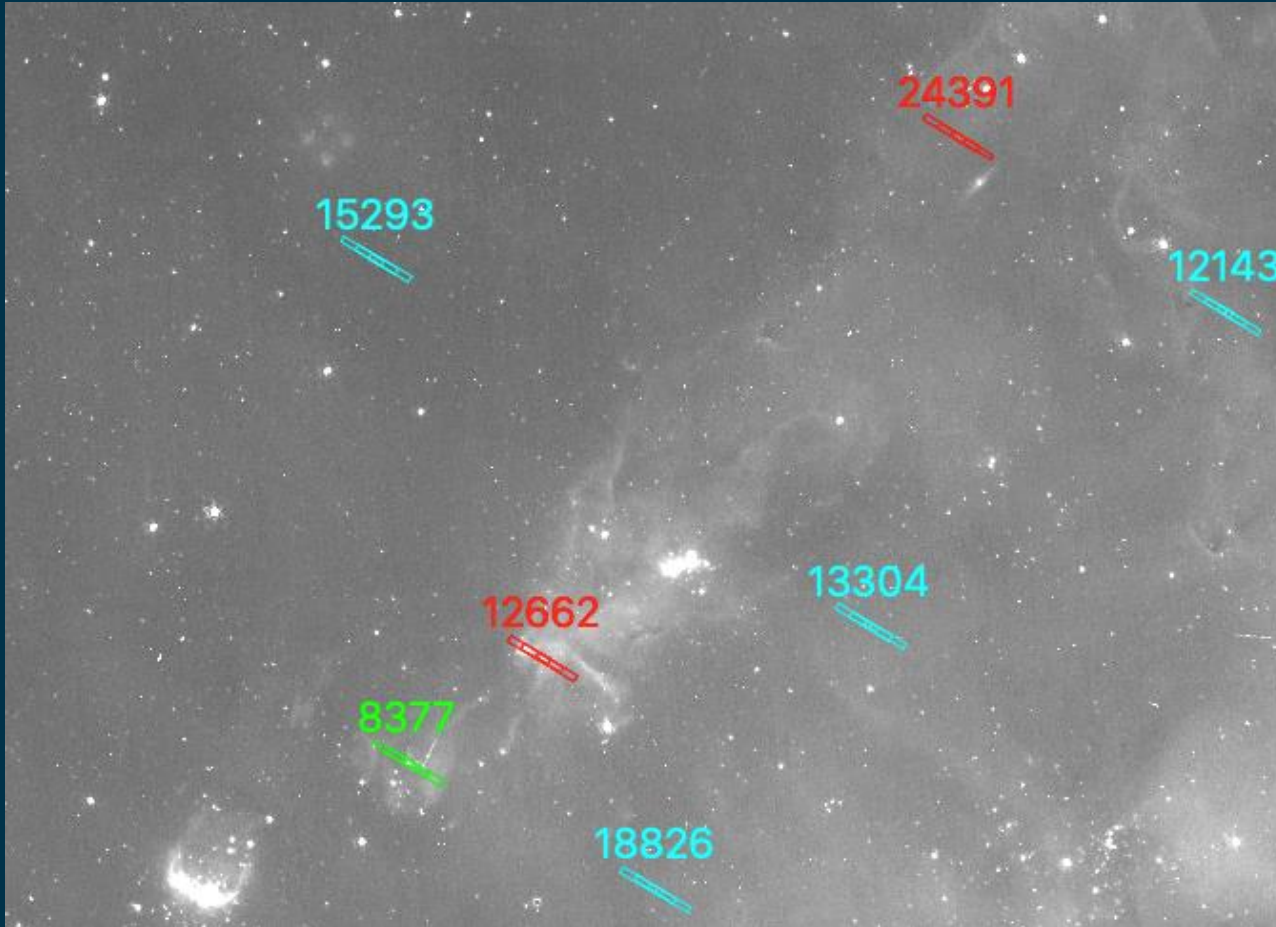
Tiny windows with shutters: 100x200 μm

- Works at cryogenic temperature (40K)
- Each shutter opens individually and wide



365x171x4 micro-shutters
 $\sim 250,000$





NIRCam image F187N

GTO program

led by Guido De Marchi

MOS Observations of two star-formation regions:

- NGC364 in SMC
- NGC3603 in our galaxy

Excuted in July/August

Observation strategy:

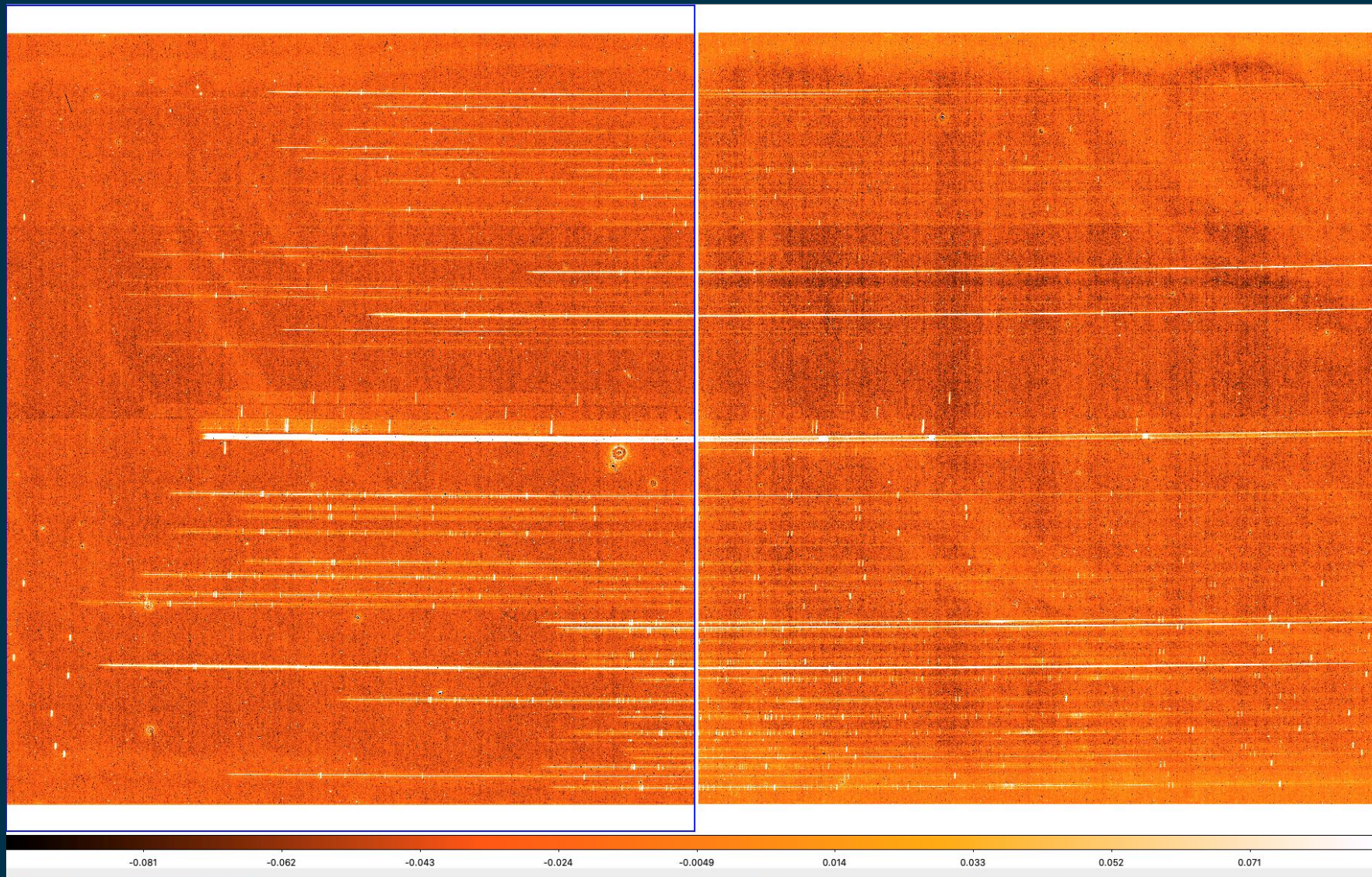
3-shutters per targets

3-point nod

5-sky regions sampled over three exposures

→ Standard approach for direct background subtraction

The data: exposures



- MOS observations
- 40 stars
 - Medium resolution spectroscopy
F170LP-G235M
 $\Delta\lambda = 1.7 - 3.2 \mu\text{m}$

Tasks:


- Run ESA team pre-processing pipeline (implemented in C): generates count-rate images from raw exposures (up-the-ramp integrations)
- Run ESA pipeline NIPS (python 3): generate flux- and wavelength calibrated spectra from detector traces
- Run python scripts to plot and analyze individual spectra

What else did we need?

What did we need?

- Access to the archived data in NIPS format
- A “private” place to work collaboratively on these non-public datasets, trying to avoid as much as possible to spread them all over the place.

➔ Remotely accessible platform

- ❖ Small team of 4 people running the data reductions based in Italy and the Netherlands
- ❖ The raw data from our observations become available in July and August, when we were on holiday and traveling to different places
- ❖  We needed a quick solution... a solution NOW...

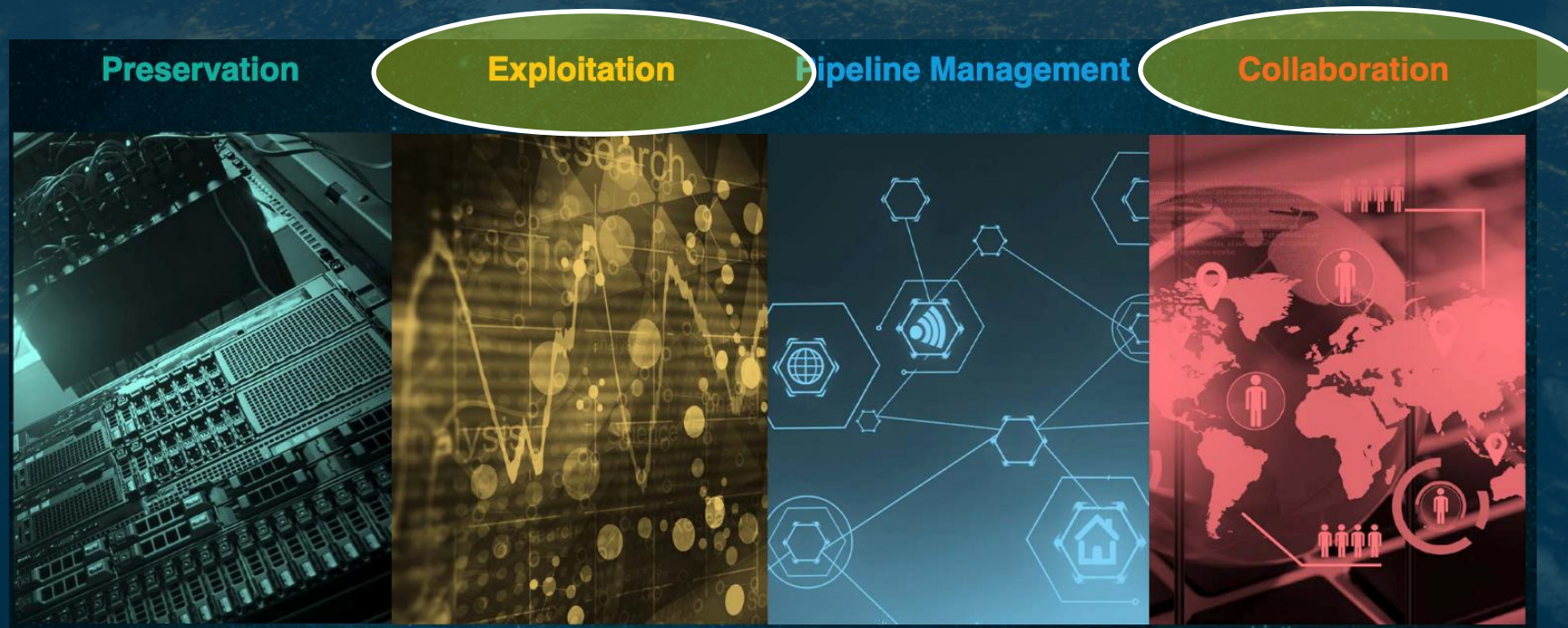
ESA Datalabs entered the scene...

I had some experience of working with the Datalabs from the analysis of some parts of commissioning data that were conducted with external support from the GTO team -

Tech talk by P. Ferruit

This was the right tool for our project!! 😊 😊

Within a few days all of us could run the data reduction

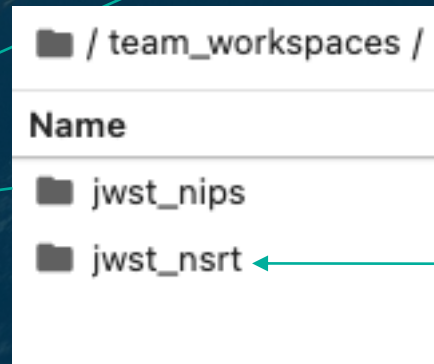
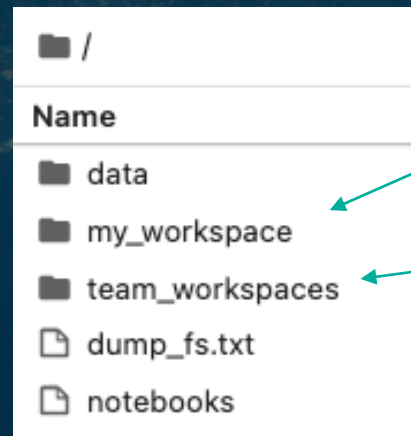
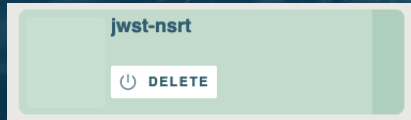


Note that we were not talking of “big data” in our case.

Accessing “private” work area in Datalabs

A shared NSRT volume was created only visible by the Datalabs users registered GTO users.

- Manual process during this development phase, expecting it to be “standard” in the released version.
- “Persistent” area, will survive the deletion of a jwst-nsrt Datalabs instance.
- Work performed by Datalabs team in collaboration with P. Ferruit
- Extensive support by Marcos



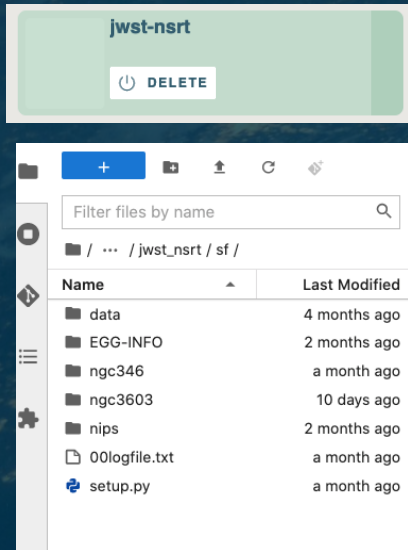
Standard folder in Datalabs. What users put here is only visible by themselves and is “persistent”. Not a shared space.

What users put here is only visible by NSRT users and is “persistent”. This is the shared space we requested.

(users belonging to multiple teams will see more folders)

Setting up our “private” Python environments

This was done by creating “local” environments stored in the shared volume. Their maintenance, configuration-control was our business.



Specific call to the conda commands used to create (once!) and activate these environments.

```
55 # creating the trunk environment
56 conda create -p /media/nsrt/envs/processing python=3.9
57 conda activate /media/nsrt/envs/processing
```

With every new terminal sessions:

```
(base) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/home$ conda activate /media/nsrt/envs/processing
(processing) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/home$
(processing) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/home$ cd /media/nsrt/sf/ngc346/
```


Typical session: running python script in terminal



The screenshot displays the ESA Datalabs interface, version 0.3.0/BETA. The interface is divided into several sections:

- File Browser:** Located on the left, it shows a list of files and folders. A green circle highlights the 'Terminal' icon in the bottom toolbar.
- Notebook:** The central area shows a notebook titled 'team_workspaces/jst_nrst/sf/ngc346'. It contains a 'Python 3 (ipykernel)' kernel and a 'processing' kernel. A green arrow points from the 'Terminal' icon to the 'Terminal' window.
- Terminal:** The bottom right section shows a terminal window with the following commands and output:

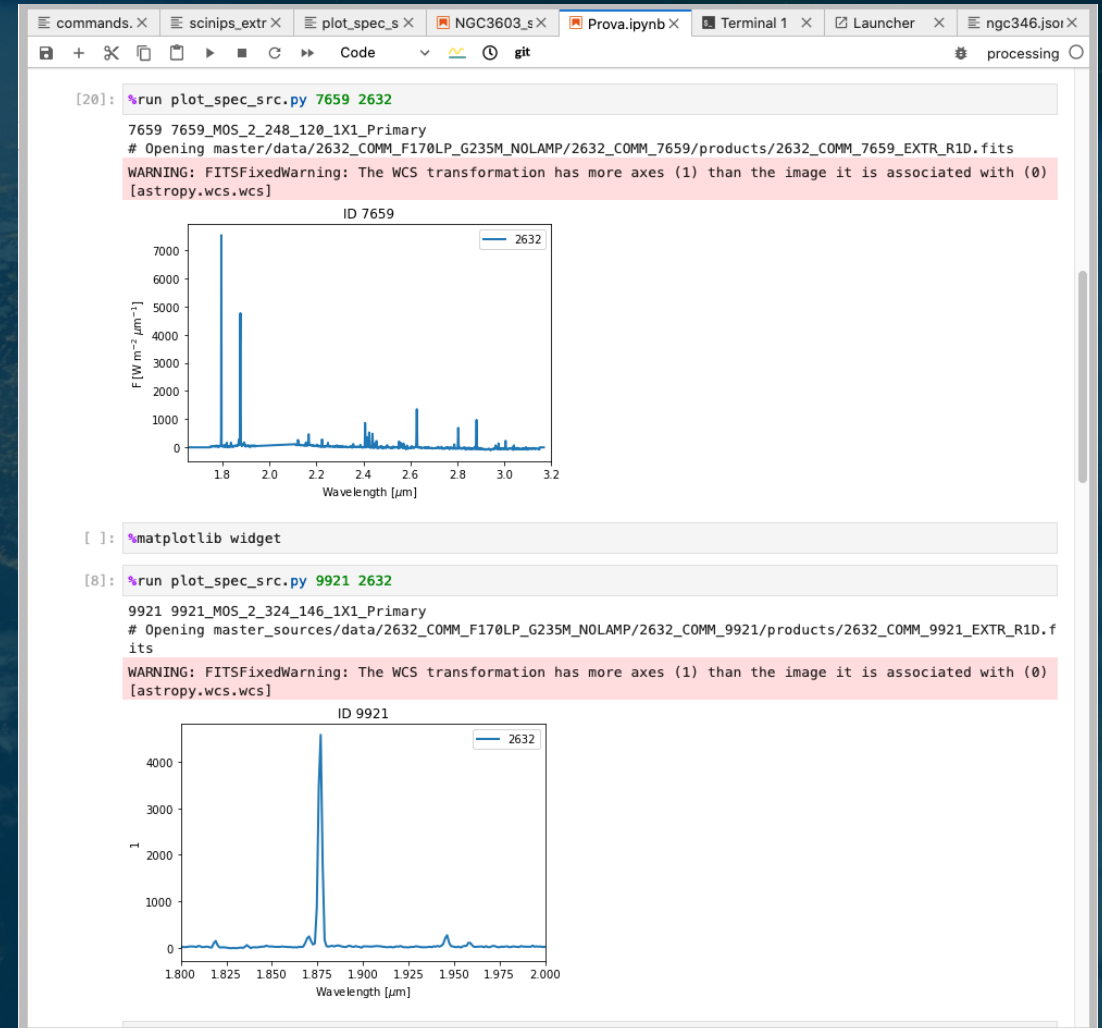
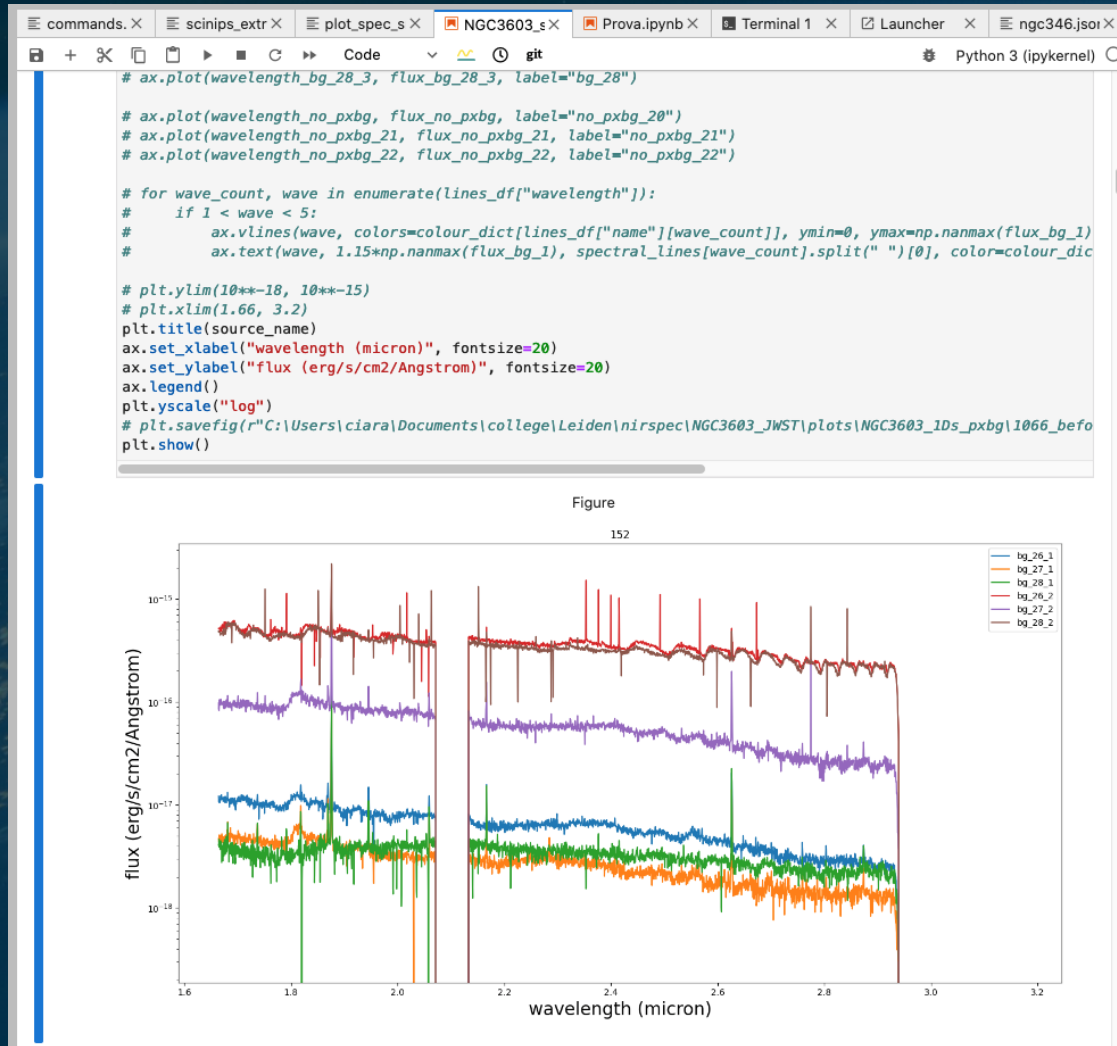
```
(base) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/home$ conda activate /media/nsrt/envs/processing
(processing) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/home$ cd /media/nsrt/sf/ngc346/
(processing) ggiardin@datalab-81c9cf7dc99114-6577557ff9-q8zc6:/media/nsrt/sf/ngc346$ python set_up_targets.py --conf ngc346.json

['set_up_targets.py', '--conf', 'ngc346.json']
Version: 1.0.1
2022-11-22T09:30:37.053419

# Input configuration file: ngc346.json
# Extraction slitlet: 1X1
# Loading the configuration file.
# ./
# Main loop on the exposures.
# Processing exposure: 2632_COMM
# * Associated background exposures: ['2633_COMM', '2634_COMM']
# Loading the MPT file ./csv_1227/1227-obs14-cl1n1-G235M-F170LP.csv
# 201_MOS_3_063_149_1X1_Primary (14.835007,-72.166427)
/media/nsrt/sf/ngc346/set_up_targets.py:160: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use
`float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Depreciated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
target.m_set_aperture('MOS', np.array([dx, dy], dtype=np.float),
# 3159_MOS_4_192_151_1X1_Primary (14.751402,-72.192740)
# 5102_MOS_4_324_094_1X1_Primary (14.791109,-72.196906)
# 7659_MOS_2_248_120_1X1_Primary (14.715198,-72.165266)
# 8377_MOS_2_331_070_1X1_Primary (14.745653,-72.166891)
# 9685_MOS_4_262_003_1X1_Primary (14.820643,-72.186239)
# 9884_MOS_1_354_126_1X1_Primary (14.824727,-72.154485)
```



We also used Jupiter notebooks



What worked well:

- We were able to set up our work environment in Python without any problem. We were able to manage it by ourselves.
- Datalabs is very intuitive environment – in particular thanks to the “Terminal” application that allows one to work as one is used, on a standard computer running a Unix-like OS
- Speed!! NIPS pipeline is parallel software so having access to many cores meant we could re-run processing significantly faster than on our laptops: fast-turn around!
- Many re-processing due to criticalities of background subtraction...

What did not work so well:

- Network... not a Datalabs-specific issue
- Visualisation

Adapting our way of working to Datalabs

Datalabs is a notebook-oriented environment if one wants to display things, when using python, notebook is the way to go





















Generally, one cannot use applications with GUIs in Datalabs unless they are available in the Datalabs catalog.

ESA Datalabs [0.3.0/BETA]

Create Datalab

Find a datalab in ESA datalabs catalog

Filter results

 aladin Aladin is an interactive sky atlas allowing the user to visualize digitized astronomical images or full surveys, superimpose entries from astronomical catalogues or databases, and interactively access related data and information from the <i>Simbad</i> database, the <i>VizieR</i> service and other archives for all known astronomical objects in the field.	 filezilla FileZilla	 fv FV - An image display and visualization tool for astronomical data
 jl-esdc Jupyterlab ESDC	 jl-euclid-dps Euclid DPS JupyterLab	 jl-herschel Herschel JupyterLab
 jl-juice JupyterLab with JUICE moon coverage tool (0.8.0).	 jl-pangaia PanGaia JupyterLab	 jupyterlab Plain JupyterLab for demonstration of basic functionality.
 jwst Jupyterlab JWST	 jwst-miricle Jupyterlab JWST Miricle	 jwst-nips Jupyterlab JWST NIPS
 jwst-nsrt Jupyterlab JWST NSRT	 qfitsview QfitsView - An image display and visualization tool for astronomical data	 theia-python Theia Python Editor
 x-cosmographia Cosmographia	 x-ds9 SAOImageDS9 - An image display and visualization tool for astronomical data	 x-glab GNSS-Lab Tool (gLAB)
 x-octave Scientific Programming Language.	 x-topcat Tool for Operations on Catalogues And Tables	

→ This point is actively worked on by the datalabs team.

- We needed a platform to work collaboratively on reducing and analyzing NIRSpec data using specific software: ESA team data reduction software
- We found Datalabs a stable and intuitive environment for us to efficiently process the data and share the high level products
- Scientific analysis of these products is on going (lines, their intensities, ratios, etc.)... soon to be revealed!
-