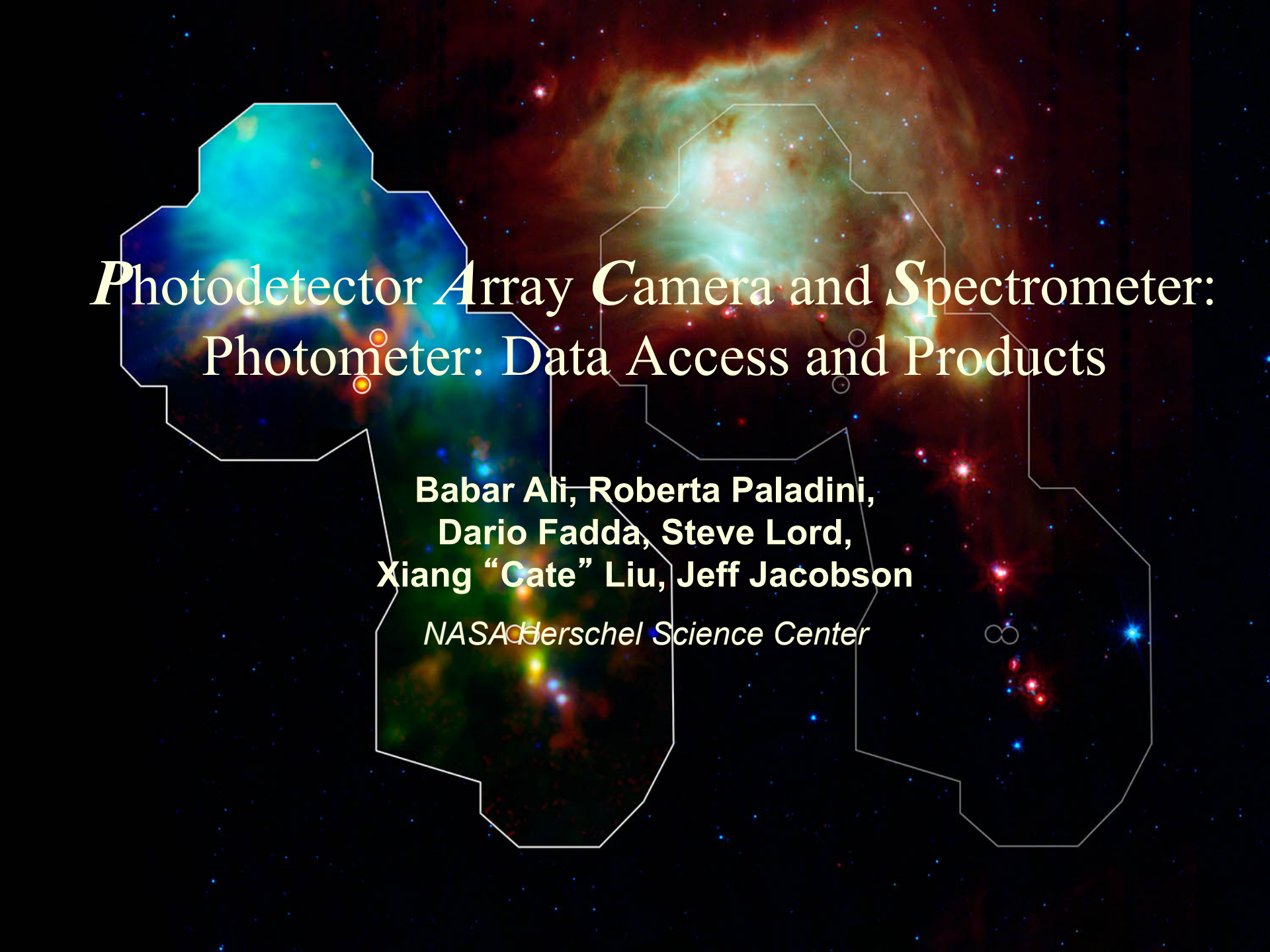


*P*hotodetector *A*rray *C*amera and *S*pectrometer: Photometer: Data Access and Products

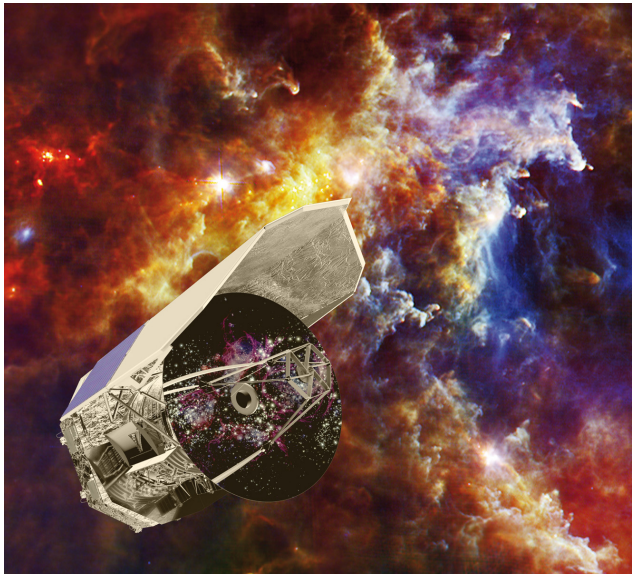
Babar Ali, Roberta Paladini, Dario Fadda, Steve Lord, Xiang "Cate" Liu, Jeff Jacobson

NASA Herschel Science Center

Two Topics

I. Accessing PACS data

II. Photometer Products



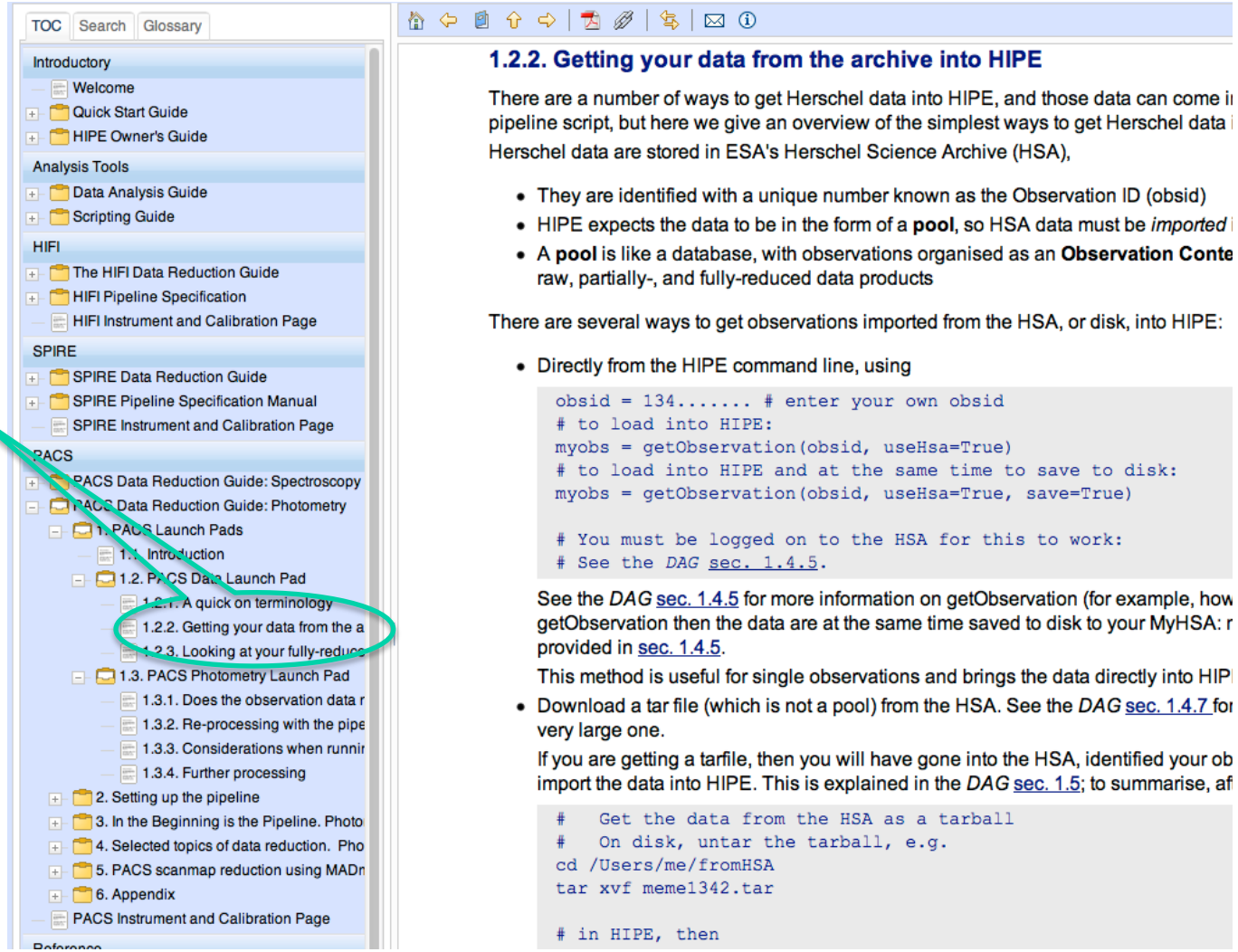
Accessing PACS photometer data

Introductory Notes

1) PACS data may be accessed and viewed via many different and independent methods.
2) For data processing and analysis within HIPE, one convenient method is '**getObservation**', which uses the HIPE command line interface.
3) Access to data is provided via the unique observation identifier (**OBSID**).

Relevant Section in HIPE 11 documentation

1.2.2



get data via 'getObservation'

Define your observation

The command in its most basic form.

```
1
2   # Define OBSID
3
4   obsid = 1342185553
5
6   # Access the observations from Herschel Science Archive
7   #
8   # obsid = Your observation identifier
9   # useHsa = True/False means obtain from local disk or the Science Archive
10  # instrument = 'PACS'  is only needed for parallel mode observations.
11
12  obs = getObservation(obsid,useHsa=True,instrument='PACS')
13
14
```

Script name: "*Example-accessSingleOBSID.py*"

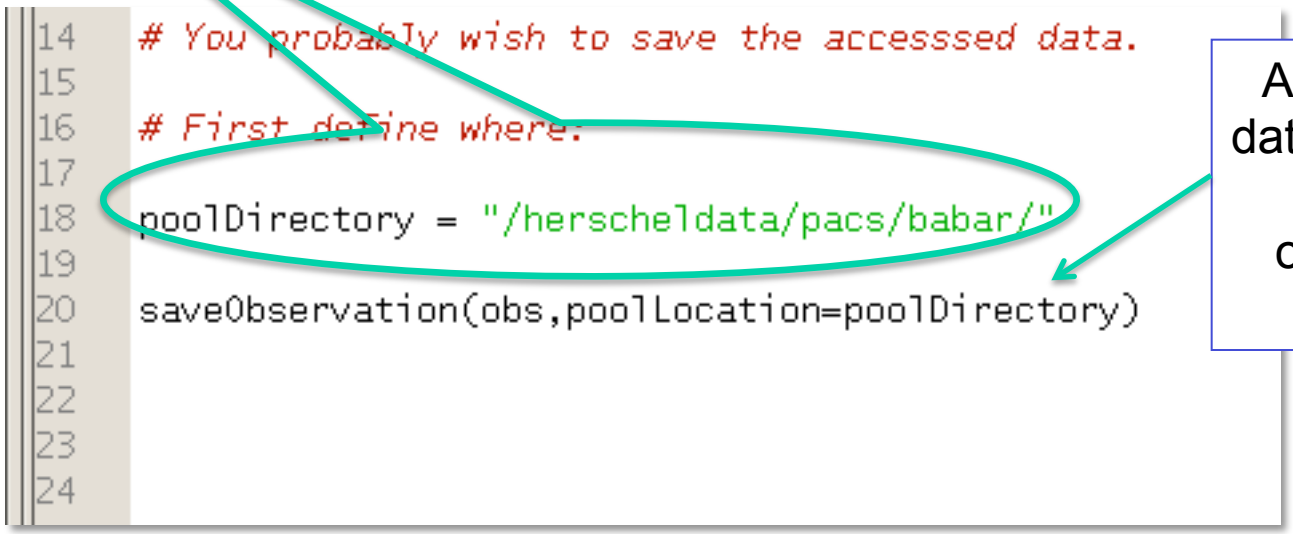
What just happened?

- In fact, you have only established a link to the data.
- The variable ‘obs’ is called the ‘**observation context**’ of your observation.
- Access to actual data is proceeds via further manipulation of the observation context.
- We will discuss ‘obs’ in later slides.

Saving your data.

The command to use is '**saveObservation**'

Define location



And, save all data associated with the observation context.

Script name: "*Example-access-and-save-SingleOBSID.py*"

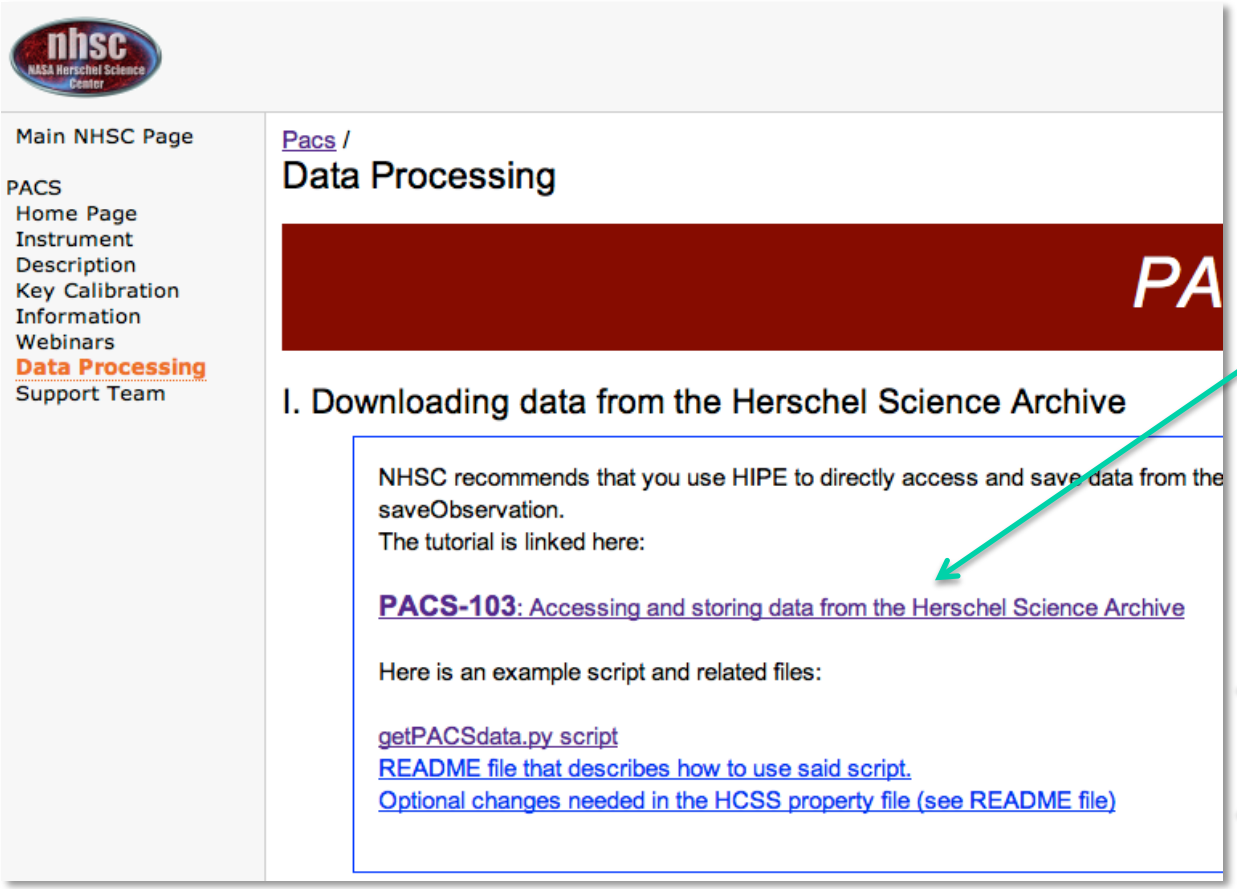
Access & Saving Multiple OBSIDs

```
# Define OBSIDs in a python array

obsids = [1342185553,1342185554]

# EDIT and Define where you wish to save the data.

poolDirectory = "/herscheldata/pacs/babar/"

# Loop through each obsid to save the data.

for obsid in obsids:
   print "Starting OBSID=%i ..." % (obsid)
   obs = getObservation(obsid,useHsa=True,instrument='PACS')
   saveObservation(obs,poolLocation=poolDirectory)
   print "... finished"
```

Script name: "*Example-access-and-save-MultipleOBSIDs.py*"

NHSC/PACS team has prepared a dedicated tutorial on how to access and download PACS data in HIPE:

https://nhscsci.ipac.caltech.edu/sc/index.php/Pacs/DataProcessing



Click on this link to download or view the tutorial

Scripts and additional info on configuring HIPE

PACS photometer products

First there is observation context

- The observation context is a pointer to all PACS data associated with a particular observation.
- Individual observations are identified via their identifiers (OBSIDs).
- Interactive processing starts by loading the observation context first.
- Data are pulled into HIPE for processing via the observation context.
- In HIPE observation context appears as a variable.
- All PACS processed data products are linked via the observation context.

What are product levels?

- PACS pipelines save data at a few natural stopping points.
- The “level” in product level refers to the amount of processing applied to the raw signal.
- Higher levels imply more processing.
- The products at all levels are accessed in HIPE via the observation context link.

PACS product levels

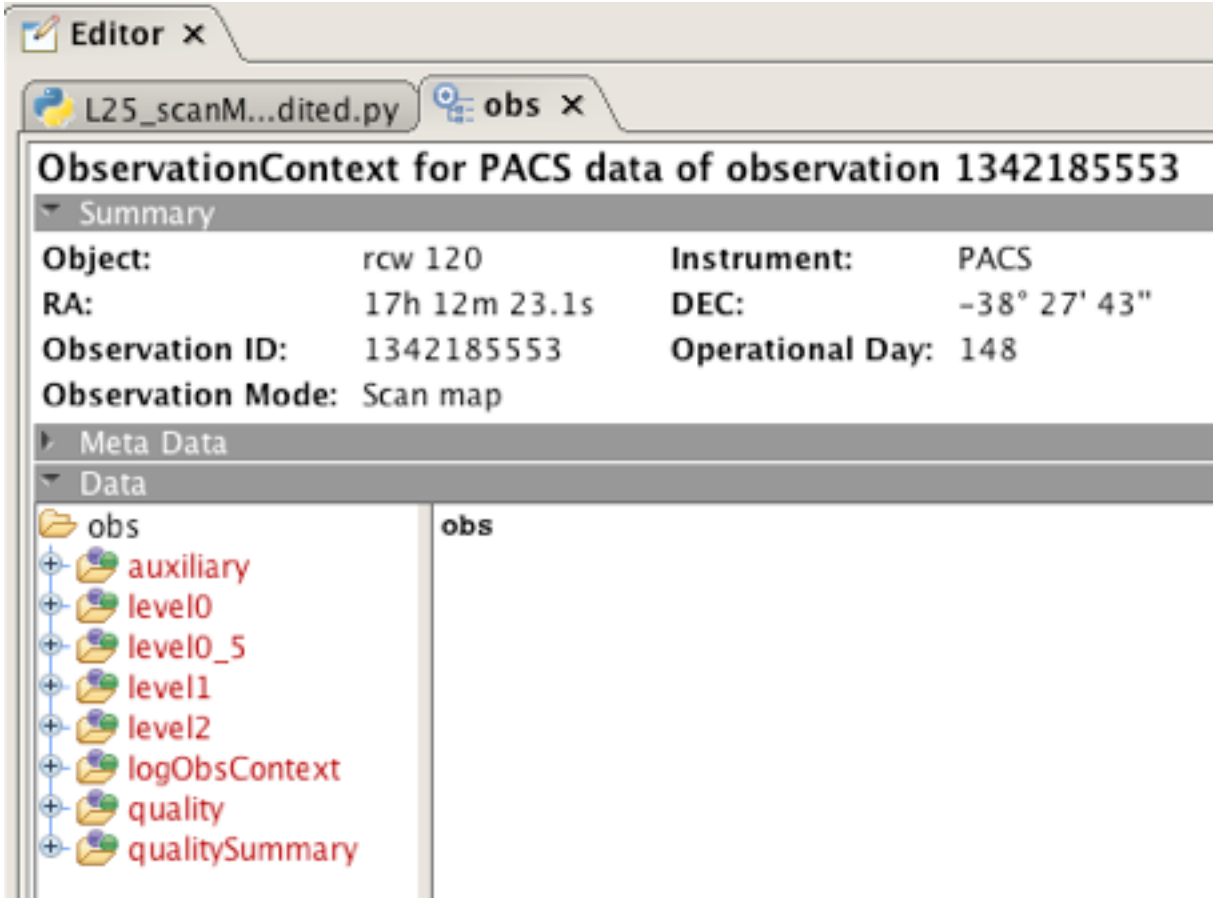
- **Level 0**: Raw signal values. Astrometry and housekeeping information is not merged.
- ~~**Level 0.5**: Basic reorganization of data and associating housekeeping and astrometry with the signal.~~
- **Level 1**: Calibrated cube of PACS bolometer readouts.
- **Level 2**: Projected maps.
- **Level 2.5**: Projected maps using two observations (scan and cross-scan).
- **Level 3.0**: Projected maps using all observations.

In addition, there are ..

- Auxilliary products, Housekeeping products and calibration products

PACS data in HIPE

- You can use the GUI aspects of HIPE to navigate though PACS data products via the observation context.



HIPE 11 documentation

Section 6.2.1