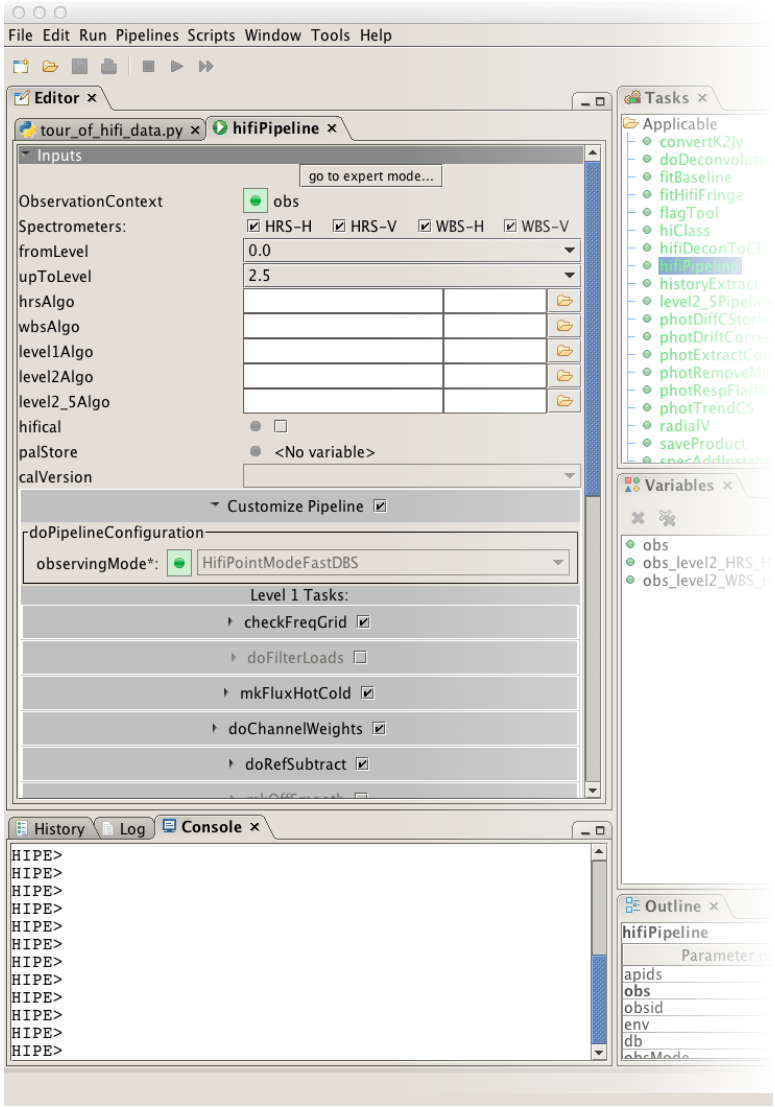


Reprocessing HIFI Data

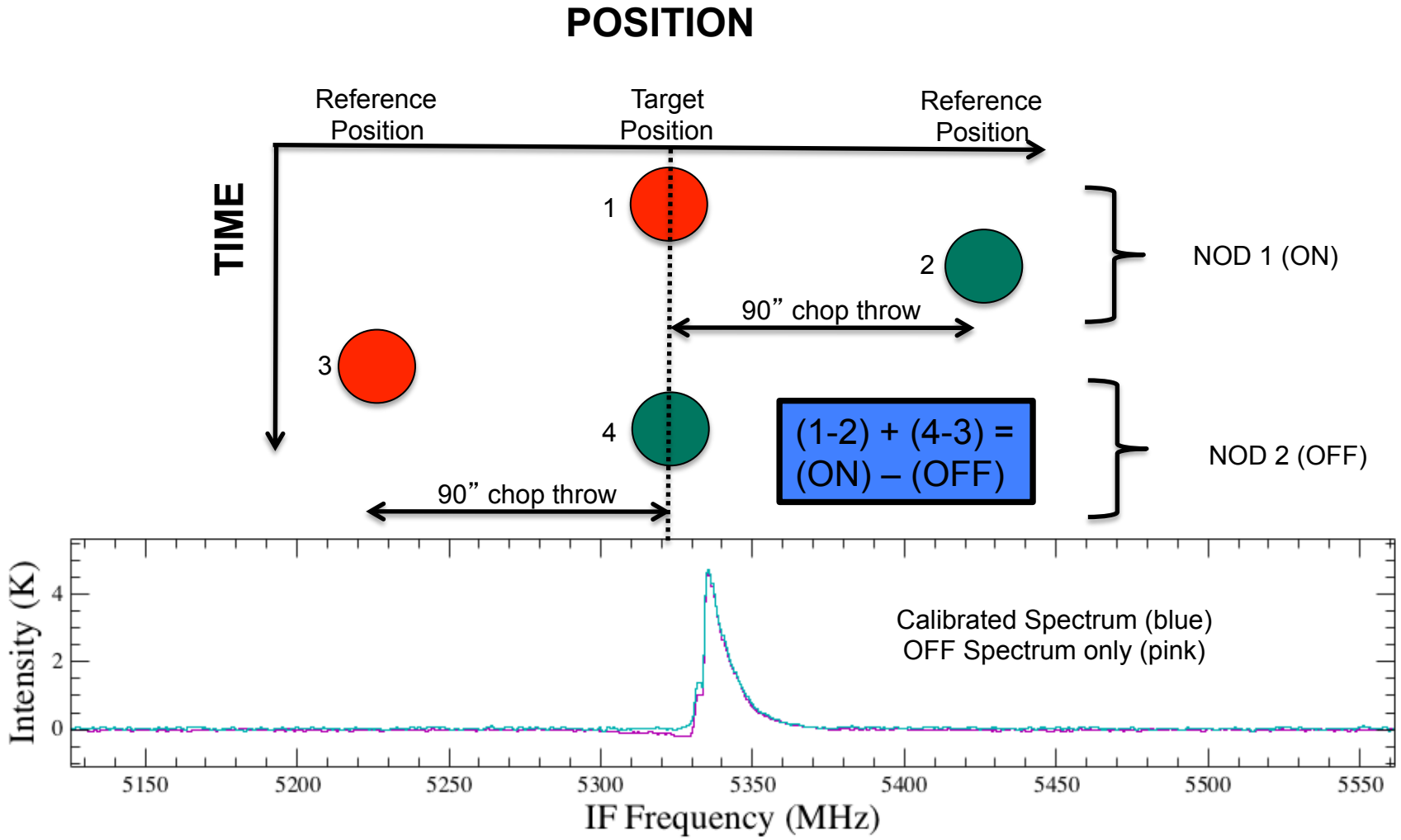
Outline

- **Why** would it be necessary to reprocess data?
  - Take advantage of new calibration updates (pointing updates require additional finesse)
  - Take advantage of improved pipelines
  - Adjust parameters in existing pipeline
  - Include new or optional steps in the pipeline
  - The pipeline is a useful aid for gauging the quality of the data

- **How** to reprocess HIFI data
  - HSC on-demand re-processing
  - Command line based
  - GUI based re-pipelining (configurable)
  - Script based re-pipelining

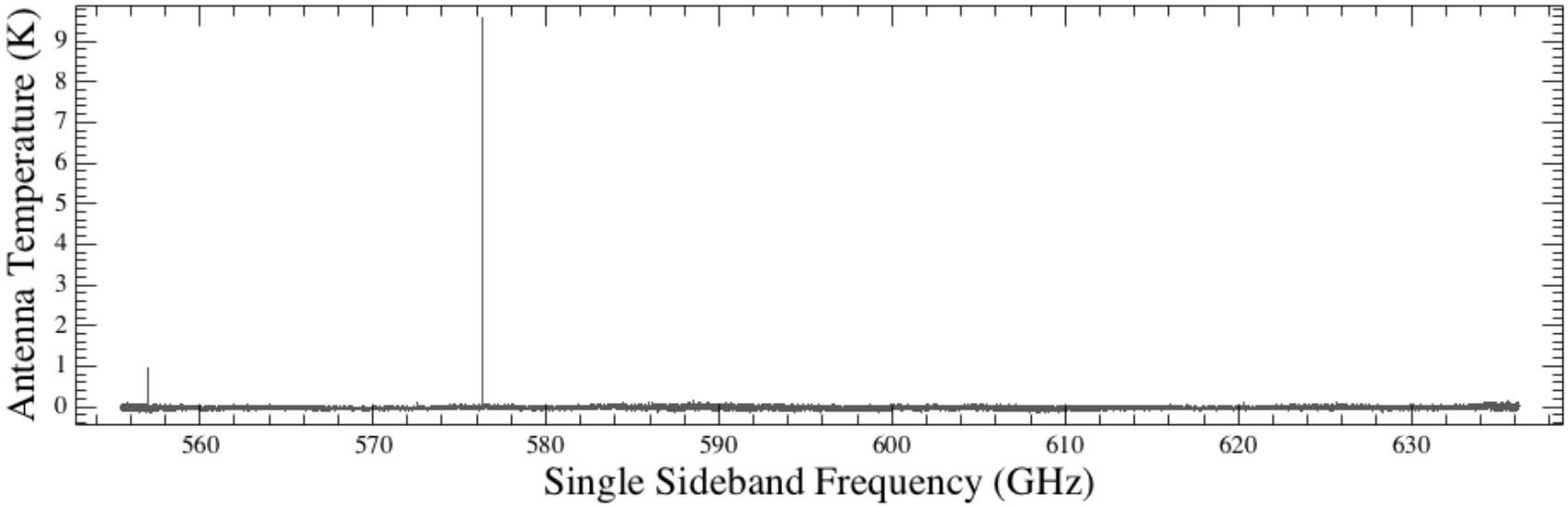
Data used in this demo

- 1342190183: Point Mode DBS with emission in a reference position



Data used in this demo

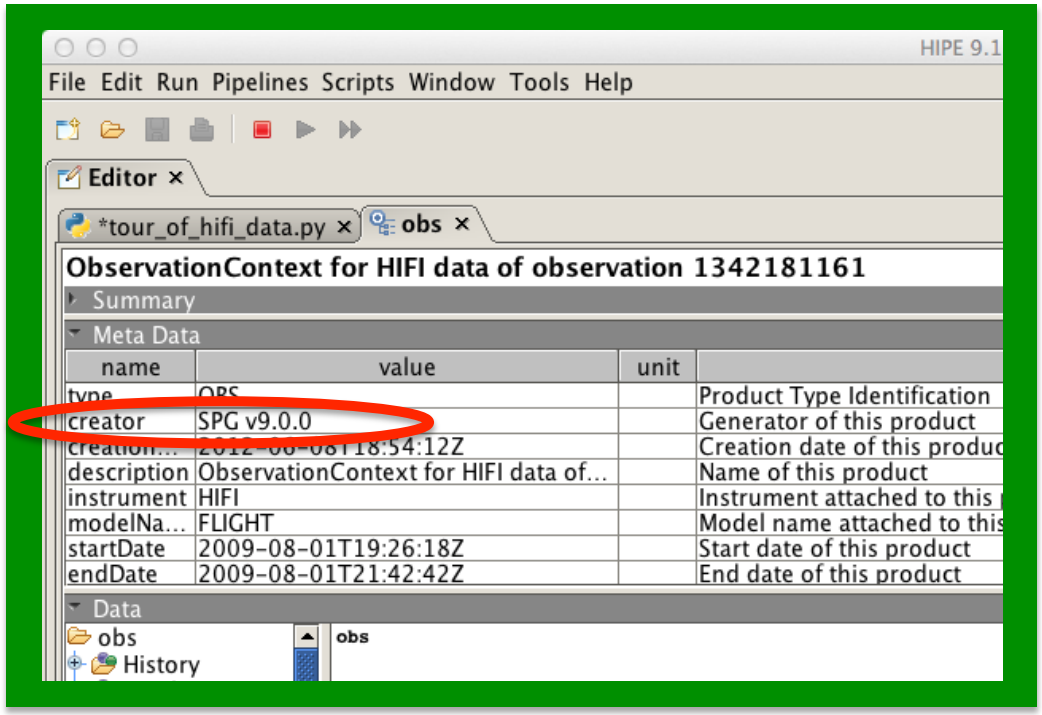
- 1342181161: DBS Spectral Scan
  - No emission in the reference beams
  - Pipeline performs an extra step compared to point mode
  - Deconvolution (discussed tomorrow) combines the spectra at different frequencies into a single spectra.

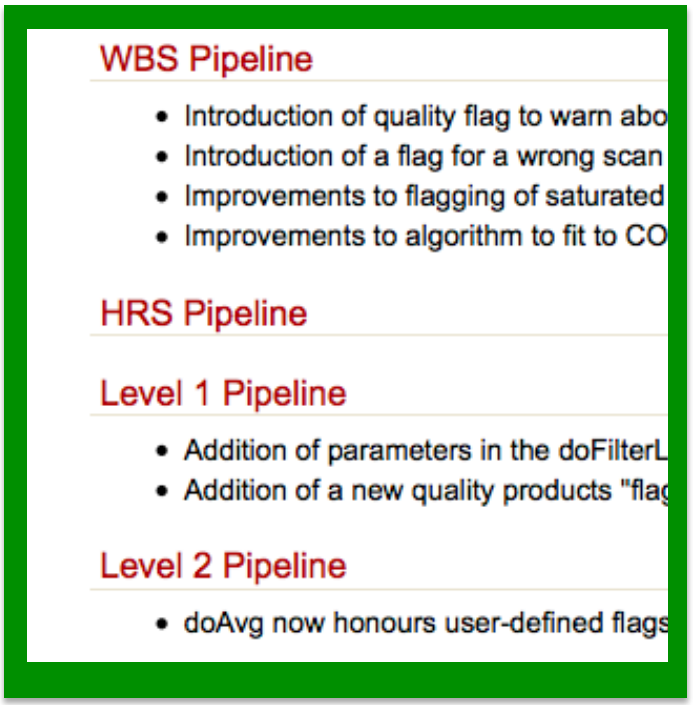


Changes in algorithms

- Compare the SPG meta data against the `what's new' web

http://herschel.esac.esa.int/twiki/bin/view/Public/HipeWhatsNew

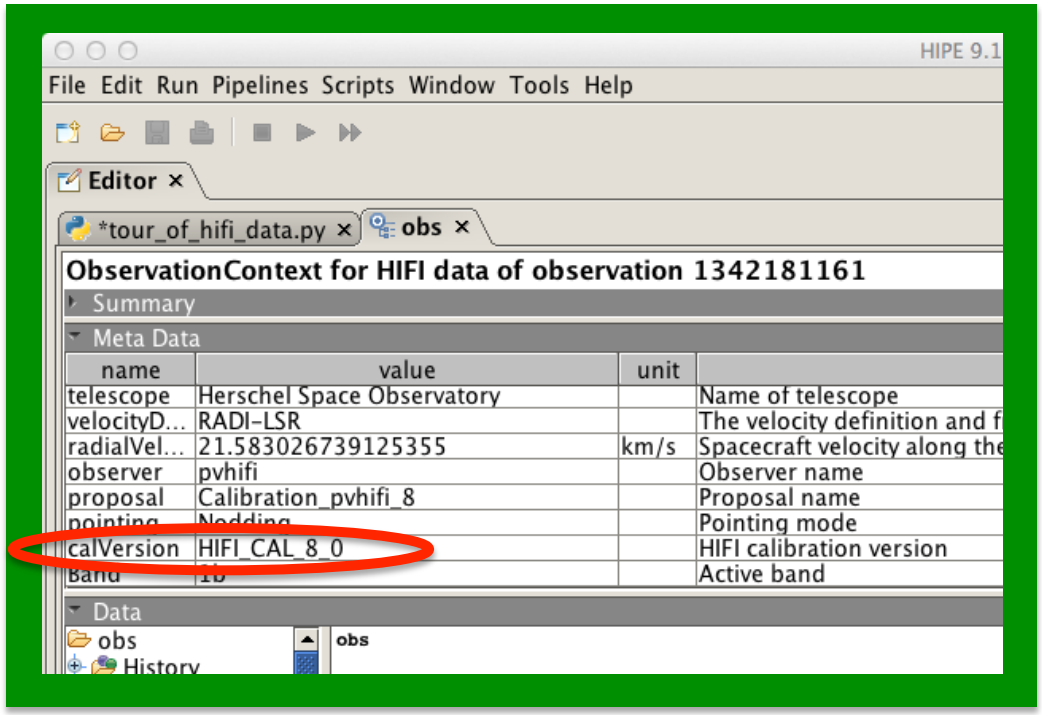


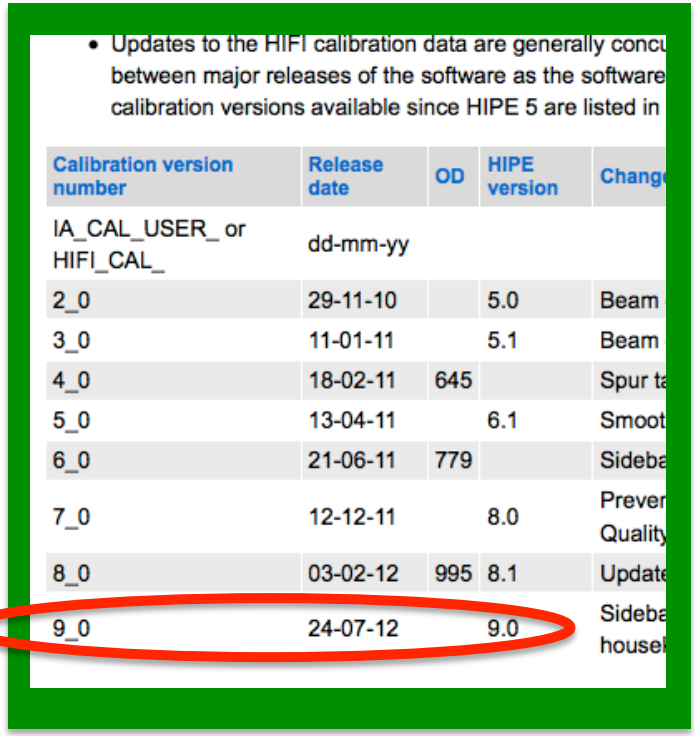


Changes in calibration

- Compare **`calVersion`** meta data in the observation context against the list on the HIFI calibration web page.
- Cal updates usually tied to major releases but not always.

http://herschel.esac.esa.int/twiki/bin/view/Public/HifiCalibrationWeb





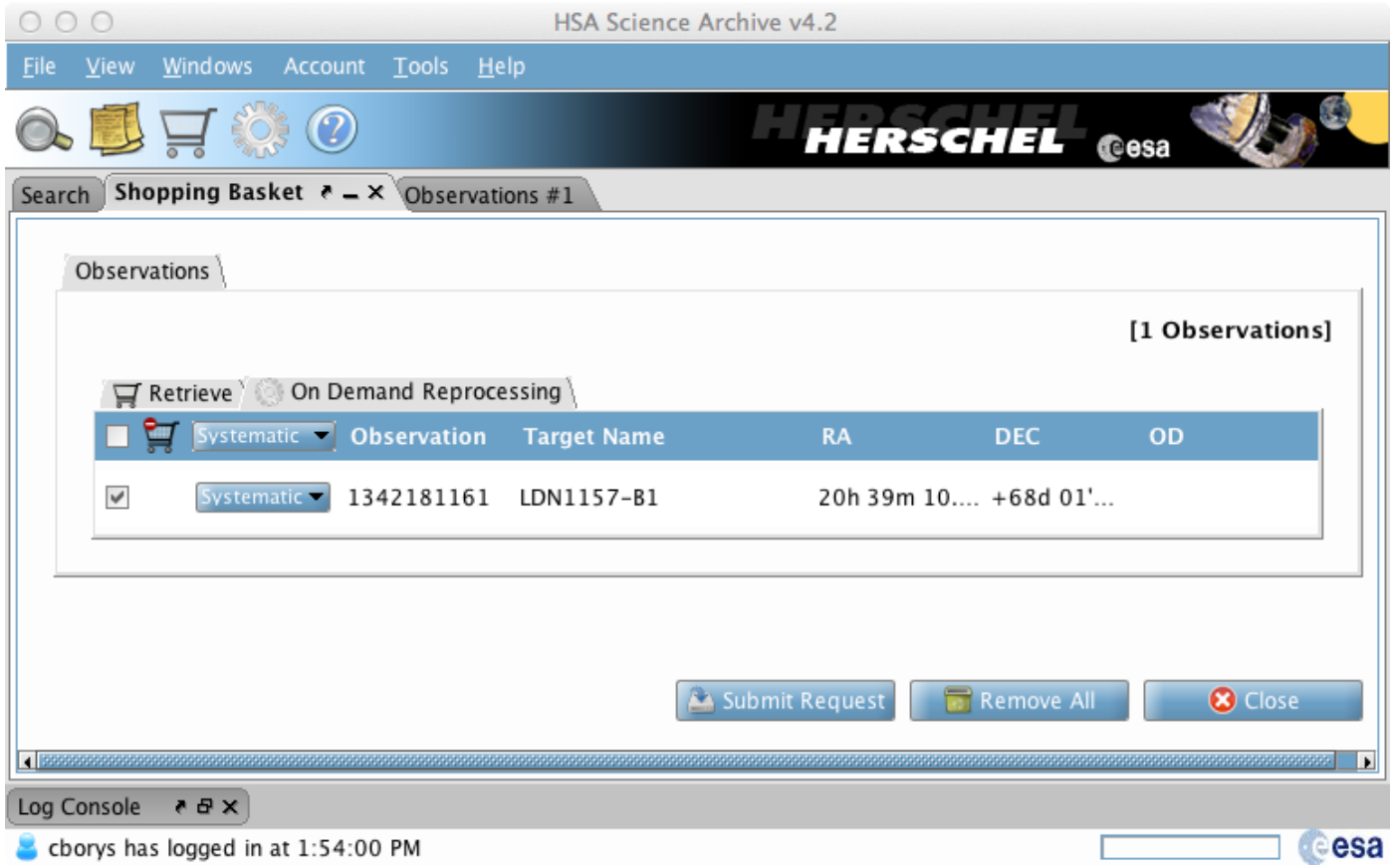
- Updates to the HIFI calibration data are generally concu
between major releases of the software as the software
calibration versions available since HIPE 5 are listed in

Calibration version number	Release date	OD	HIPE version	Change
IA_CAL_USER_ or HIFI_CAL_	dd-mm-yy			
2_0	29-11-10		5.0	Beam
3_0	11-01-11		5.1	Beam
4_0	18-02-11	645		Spur ta
5_0	13-04-11		6.1	Smoot
6_0	21-06-11	779		Sideba
7_0	12-12-11		8.0	Prever Quality
8_0	03-02-12	995	8.1	Update
9_0	24-07-12		9.0	Sideba housel

HSC on-demand reprocessing

- Use HSA web page or via HIPE:

  `Window->Show View->Data Access-> HSA`



Other reasons to re-pipeline

- Two other major reasons for re-pipelining are to tweak the parameters of the existing pipeline, or to include optional pipeline steps.
- The On-demand reprocessing cannot accommodate these, and thus one must use HIPE.
- The configurable pipeline is particularly useful for getting access to all aspects of the pipeline

HIFI pipeline command line use

- Two other major reasons for re-pipelining are to tweak the parameters of the existing pipeline, or to include optional pipeline steps.
- The On-demand reprocessing cannot accommodate these, and thus one must use HIPE.

- The follwing demo includes a script called '**`hifi_pipeline_demo.py`**' which contains more advanced examples, but in its simplest form, the pipeline is run via:

```
obs_new = hifiPipeline(obs=obs)
```

- The pipeline OVERWRITES the observation context, which is passed via reference. Thus **`obs`** is identical in content to **`obs_new`** after running the pipeline. For this reason, I recommend simply:

```
hifiPipeline(obs=obs)
```

HIFI GUI use

- The HIFI pipeline GUI is a great way to 'learn' about the pipeline. It generates a command line version of what the GUI executes, so you can easily cut and paste it into a script.

- Due to how TASKS work in HIPE however, the GUI always returns a new variable. i.e obs1, obs2 ,etc. Remember that all of these variables have the same contents because the pipeline overwrites data.

- Main advantage of the GUI is to tweak parameters of existing tasks, or to include (or exclude) other tasks.

HIFI Configurable Pipeline

- The user has the option to add additional steps to the pipeline post level 2.
- Compatible tasks include **`FitHifiFringe`**, **`FitBaseline`**, and **`doDeconvolution`**.
- You can also write custom scripts that the pipeline can use (more advanced topic)