Doc 4.13, ver. 1 | October, 2016

ALMA Science Pipeline User's Guide for CASA 4.7.0

Interferometric and Single-Dish Data

www.almascience.org

User Support:

For further information or to comment on this document, please contact your regional Helpdesk through the ALMA User Portal at **www.almascience.org**. Helpdesk tickets will be directed to the appropriate ALMA Regional Center at ESO, NAOJ or NRAO.

Revision History:

Version	Date	Editors
3.13v1.0 CASA 4.5.1	January 2016	Pipeline Team
4.13v1.0 CASA 4.7.0	October 2016	Pipeline Team

In publications, please refer to this document as:

ALMA Pipeline Team, 2016, ALMA Science Pipeline User's Guide, ALMA Doc 4.13v1.0

Table of contents

1 The ALMA Science Pipeline

1.1 Purpose of this document

The ALMA Science Pipeline is being developed for the automated processing of ALMA interferometry and single-dish data, and for the combination of data taken using multiple ALMA arrays. The Cycle 4 Pipeline is commissioned for the calibration of “standard mode” ALMA interferometry datasets (see Sec. 5.2 of the Cycle 4 Proposer’s Guide), for the provision of diagnostic calibrator images, and for preliminary science target imaging (using a simple clean mask and shallow clean, potentially for only a subset of science targets). It is also commissioned for the end-to-end calibration and imaging of ALMA Single-Dish data acquired from Cycle 3 onwards. Combined science target imaging for interferometric data taken from multiple arrays is not yet commissioned, nor is pipeline processing of “non-standard mode” observations.

This document describes how to obtain the ALMA Pipeline, how to use it to calibrate ALMA interferometric (IF) and single-dish (SD) data, and a description of the Pipeline WebLog (collection of web pages with diagnostic information describing the pipeline run). Since interferometric and single-dish data are calibrated and imaged using different procedures and diagnostics, their recalibration procedures and WebLogs are described separately.

1.2 Pipeline Overview

The Pipeline calibrates and (optionally, in the case of interferometric data) images ALMA data automatically using Pipeline tasks. ALMA datasets are comprised of all of the individual executions that result from completing a Scheduling Block (SB) (provided they pass QA0). An individual execution is called an ASDM (for ALMA Science Data Model), and the collection of ASDMs from a single SB are collected into a data structure called a Member Observing Unit Sets (MOUS). See Chapter 10 of the ALMA Technical Handbook for details.

Pipeline tasks use CASA tasks wherever possible to perform the data reduction, and Pipeline tasks can be viewed and executed within CASA in exactly the same way as CASA tasks. For example, in a CASA version also containing the Pipeline, view the possible inputs for the task `hifa_importdata` by typing `inp  hifa_importdata`. To see all the tasks available in CASA, type `tasklist`.

The Pipeline is data-driven: i.e. the characteristics of each dataset drive the calibration and imaging strategy (the **Pipeline Heuristics**). During the Pipeline run, critical information (for example, which calibration tables are used) are stored in the pipeline **Context**. Both the **Heuristics** and the **Context** are implemented as python classes.

In order to determine if the Pipeline was used in the processing of an ALMA dataset, please consult the README file in the data delivery package. Some projects may contain a mix of both manually and Pipeline-calibrated data.

2 What's New in Cycle 4

New features of the Cycle 4 pipeline include:

- The interferometric calibration pipeline has a low signal-to-noise heuristic that will calculate the temporal phase variations by combining spectral windows (stage `hifa_spwphaseup`).
- The Interferometry Pipeline now includes science target imaging, as well as including "check sources" in the calibrator imaging stage and improved calibrator quality assurance scores.
- The Single-dish Pipeline has been refactored to use Measurement Set rather than scan table format.
- There are improved defaults for the `hif_gainflag` task.
- Files previously exported as .tar.gz are now exported as .tgz.

Known limitations of the Cycle 4 pipeline include:

- No flux equalization between the different executions of multi-epoch observations.
- No automated science target flagging (although a flag template file is available for pipeline to apply manually identified flags).
- The frequency ranges for interferometric continuum identification and subtraction are done in an automated manner that works well over a very broad range of observing modes and source properties. In some cases (e.g. hot core line emission, noisy broadband continuum), it is expected that better results can be obtained by more careful examination of individual sources and/or spectral windows.
- The frequency ranges for single dish line identification and spectral baseline subtraction are done in an automated manner that has been optimized to detect moderate channel width (wider than 100 channels) emission lines at the center of a spectral window. It is expected that better results can be obtained by more careful examination of individual sources and/or spectral windows. The following cases are most strongly affected:
  - Narrow emission lines (less than 100 channels wide), especially in TDM mode.
  - Emission at the edge of spectral window.
  - Cubes with a "forest" of emission lines.
- Science target deconvolution is done with a generic mask and shallow dynamic-range limited clean thresholds, meaning that images with moderate to strong emission will benefit from more carefully defined masks and deeper cleaning thresholds.
- The clean may terminate early if a (conservative) divergence criteria is met (warning is given in WebLog).
- The pipeline does not include science target self-calibration. Therefore, the pipeline imaging products of bright sources may be dynamic range limited.

- The interferometric imaging pipeline commands should work with measurement sets calibrated outside the pipeline, but this has not been tested extensively and may have as-yet undetermined failure modes.

A list of pipeline “known issues” is available on the ALMA Science Portal: http://almascience.org/documents-and-tools/pipeline-documentation-archive. This will be updated as issues are discovered during the cycle.

3 Cycle 4 Pipeline Version

3.1 Pipeline & CASA Versions

The pipeline tasks have a specific version number, and are bundled with a specific version of CASA. These versions are reported in the README file that is archived with the pipeline data products, and are also reported on the Home page of the WebLog for each pipeline-processed dataset (see Figure 5 for an example). For ALMA Cycle 4, the pipeline is released with CASA 4.7.0.

In general, the version of Pipeline+CASA used in ALMA Operations to calibrated and image the archived data will be the same as the publicly posted pipeline version available from the “Obtaining CASA” page, but they can be slightly different (e.g. for bugs that have an operational workaround but which are fixed in the posted public version). There is a “**Pipeline Version Tracker**” available from the **ALMA Science Portal** at https://almascience.nrao.edu/documents-and-tools/pipeline-documentation-archive#version, which lists the versions of CASA+Pipeline used in ALMA Operations as well as the versions which should be used for any restoring or reprocessing of the data from the same cycle (as described elsewhere in this document).

3.2 Obtaining the Pipeline

A download of CASA 4.7.0 that includes the ALMA pipeline is available, along with installation instructions, from the Pipeline section of the **Documents & Tools** page of the **ALMA Science Portal**: (http://www.almascience.org/). If any issues are encountered with CASA 4.7.0 installation, please contact the ALMA Helpdesk via the link on the ALMA Science Portal.

4 Pipeline-related Documentation

The User documentation currently relating to Pipeline is available from the ALMA Science Portal at http://almascience.org/documents-and-tools/pipeline-documentation-archive. This includes:

- **ALMA Science Pipeline User’s Guide**: This document.
- **ALMA Science Pipeline Reference Manual**: Description of individual Pipeline tasks.

In addition, Chapter 13 of the **ALMA Technical Handbook** describes ALMA pipeline processing, and the **ALMA QA2 Data Products** docment for Cycle 4 provides a description of ALMA data deliveries, including pipeline products. These are also available from the ALMA Science Portal “Documents and Tools” page at https://almascience.nrao.edu/documents-and-tools.

5 ALMA Interferometric Data

ALMA Interferometric data refers to observations obtained with either the 12-m Array or 7-m Array. This section describes how to reconstitute calibrated interferometric data and how to re-run interferometric pipeline calibration and imaging commands.

5.1 Restoring IF Pipeline calibration using scriptForPI.py

If the Pipeline was used to calibrate the data, you will find several special files in the "/script" directory of your delivery package, e.g., script/**casa_piperestorescript.py**. In order to convert ALMA raw data (ASDMs) into calibrated Measurement Sets, for example before re-imaging the data, the script **scriptForPI.py** should be used. **scriptForPI.py** directly applies calibration and flagging tables determined during the ALMA Quality Assurance process to raw data to create calibrated measurement sets (MS). The instructions for using this script will be in the **README** file of the ALMA delivery. **Using scriptForPI.py is the recommended and fastest method of obtaining calibrated ALMA data from the delivery.** For deliveries that used the ALMA pipeline, **scriptForPI.py** must be run from within a version of CASA that includes the pipeline tasks, preferably the “User Pipeline” version for the appropriate operations version as listed in the Pipeline Version Tracker (see Sec. 3). In general, this will be very close to the version listed on the Home Page of the pipeline WebLog (Sec. 7).

The **scriptForPI.py** calls the script **casa_piperestorescript.py**. At the present time, the **casa_piperestorescript.py** script should not be executed directly. When data calibrated by the pipeline are restored using **scriptForPI.py**, additional directories such as calibrated/working and calibrated/rawdata are created.

5.2 Re-running Pipeline calibration tasks using casa_pipescript.py

Under some circumstances, it is desirable to re-run all of the pipeline tasks from scratch. For instance, if CASA or pipeline version incompatibilities prevent running the restore as described in Section 5.1, then re-running the pipeline tasks will also recreate the calibrated MS (and/or images). This also allows you to change the calibration and flagging from what was performed during the ALMA Quality Assurance Process.

The complete set of pipeline commands are given in the script **casa_pipescript.py**. This is a python script that includes all tasks and parameter values, in the correct sequence, that were used for the pipeline run. For data that were both calibrated and imaged in the pipeline, the **casa_pipescript.py** file will include both the calibration and imaging pipeline commands (a typical script shown in Figure 1). In that case, a dummy **scriptForImaging.py** file may be added which only contains the note that no manual imaging was carried out. For data that were calibrated in the pipeline but imaged outside of the pipeline, the **casa_pipescript.py** file will only include the calibration pipeline commands; the manual (CASA) imaging commands will be included in a separate **scriptForImaging.py** file.

The tasks names, order, and parameter values in the **casa_pipescript.py** script reflect the processing recipe used for each individual delivery. Additionally, the `pipelinemode` parameter is set to “automatic” for each task. In this mode, the task takes the default settings for each tasks and only a limited number of parameters are exposed for editing by a user. Setting the pipeline mode to “interactive” will usually enable the values of a larger number of parameters to be changed. See the **ALMA Science Pipeline Reference Manual** for more details, and Sec. 5.4 below for some examples.

```
__rethrow_casa_exceptions = True
h_init()
try:
    hifa_importdata(dbservice=False,
vis=['uid___A002_X877e41_X452'], session=['session_1'])
    hifa_flagdata(pipelinemode="automatic")
    hifa_fluxcalflag(pipelinemode="automatic")
    hif_rawflagchans(pipelinemode="automatic")
    hif_refant(pipelinemode="automatic")
    hifa_tsyscal(pipelinemode="automatic")
    hifa_tsysflag(pipelinemode="automatic")
    hifa_antpos(pipelinemode="automatic")
    hifa_wvrgcalflag(pipelinemode="automatic")
    hif_lowgainflag(pipelinemode="automatic")
    hif_gainflag(pipelinemode="automatic")
    hif_setjy(pipelinemode="automatic")
    hifa_bandpass(pipelinemode="automatic")
    hifa_spwphaseup(pipelinemode="automatic")
    hifa_gfluxscale(pipelinemode="automatic")
    hifa_timegaincal(pipelinemode="automatic")
    hif_applycal(pipelinemode="automatic")
    hif_makeimlist(intent='PHASE,BANDPASS,CHECK')
    hif_makeimages(pipelinemode="automatic")
    hif_exportdata(pipelinemode="automatic")
# Start of pipeline imaging commands
    hif_mstransform(pipelinemode="automatic")
    hifa_flagtargets(pipelinemode="automatic")
    hif_makeimlist(specmode='mfs')
    hif_findcont(pipelinemode="automatic")
    hif_uvcontfit(pipelinemode="automatic")
    hif_uvcontsub(pipelinemode="automatic")
    hif_makeimages(pipelinemode="automatic")
    hif_makeimlist(specmode='cont')
    hif_makeimages(pipelinemode="automatic")
    hif_makeimlist(width='')
    hif_makeimages(pipelinemode="automatic")
    hif_exportdata(pipelinemode="automatic")
finally:
    h_save()
```

Figure 1: Example of a Pipeline casa_pipescript.py script for a dataset that was run through the Pipeline for both calibration and imaging

There are two ways to run re-run the pipeline from scratch, described in the following two sub-sections. One involves calling **casa_pipescript.py** directly, and one invokes it indirectly using **scriptForPI.py**.

5.2.1 Rerun Pipeline manually using casa_pipescript.py directly

Execute the following steps to modify and re-run the Pipeline calibration:

- Copy **casa_pipescript.py** from the **script** directory to the **raw** directory.
- Copy **flux.csv** and ***flagtemplate.txt** and **antennapos.csv** (if present) from the **calibration** directory to the **raw** directory.

In the **raw** directory:

- Make sure the naming of the raw ALMA data is consistent with those provided in the script (e.g. if the data ends in **.asdm.sdm** then move to names which do not have this suffix).
- Modify the appropriate ***flagtemplate.txt** file to add any additional flags, and edit **casa_pipescript.py** to comment out the pipeline steps you do not wish to repeat (e.g. imaging, which is very computationally expensive).
- Start the version of CASA containing Pipeline using `casapy --pipeline`, then type `execfile('casa_pipescript.py').`
- Alternatively, you can sequentially execute individual commands from **casa_pipescript.py**, stopping at any point to run other CASA commands (plotms, etc).

Running the script through the first `hif_makeimages` command will create:

- A calibrated MS for each ASDM in the same directory.
- Images of the bandpass, phase, and (if present) check source calibrators (1 per spectral window, in *.image format. To view a *.image file e.g. use casaviewer image_file_name).
- A pipeline*/html directory containing:
  - o the Pipeline WebLog - the html pages of plots from the calibration process. Access using e.g. firefox index.html.
  - o **casa_commands.log** –a list of the CASA tasks used by Pipeline tasks during calibration.

Note that to re-run the Pipeline multiple times, it is recommended to start each time from a clean directory containing only the raw data, flux.csv, antennapos.csv file (if present) and *flagtemplate.txt files and the casa_pipescript.py script.

If the **flux.csv** and ***flagtemplate.txt** files are not present in the directory, Pipeline will create new default versions of these files, which will not contain any edits made to them by ALMA staff during the Quality Assurance Process. Also note that it is advisable to have 8-10 GB RAM and 50-75 GB disk space per ALMA raw data file (ASDM) available to perform the Pipeline calibration (note that the requirements for running the pipeline imaging commands are much more demanding). Please contact ALMA via the Helpdesk if assistance is needed with data reprocessing.

5.2.2 Reproduce Pipeline run using scriptForPI.py

If the **casa_piperestorescript.py** is not found in the 'scripts' directory, then executing **scriptForPI.py** will fall-back to running the pipeline commands by executing **casa_pipescript.py,** instead of simply applying the calibration tables as described in Sec. 5.1. A user can force this behavior by moving or renaming **casa_piperestorescript.py** and executing **scriptForPI.py** from within the 'scripts' directory. A /calibrated directory will be created by **scriptForPI.py,** and should not exist at the time the the script is initially run. The working area for this run is 'calibrated/working/'. As described above, the hif* command arguments in the **casa_pipescript.py** file may be edited before using this method. Fine-grained control over (e.g.) continuum subtraction and flagging will be more difficult.

5.3 Running Imaging tasks

5.3.1 For manually imaged deliveries

For manually imaged datasets, the imaging commands will be included in a separate **scriptForImaging.py** file, containing all the CASA commands used to create the delivered products. In order to use this imaging script after using **casa_pipescript.py**, the science spectral windows must first be "split" out from the calibrated measurement sets and the measurement sets output with a `.split.cal` suffix. To perform the split, in CASA e.g.:

```
split('uid__A002_X89252c_X852.ms',
 outputvis='uid__A002_X89252c_X852.ms.split.cal',spw='17,19,21,23')
```

The science spectral windows are specified in the Pipeline WebLog (Home > Observation Summary > Measurement Set Name > Spectral Setup, in the ID column) or can be determined using the CASA task listobs e.g. `listobs('uid___A002_X89252c_X852.ms')` and results will be in the CASA logger.

If a script named **scriptForFluxCalibration.py** is present in the script directory, this must also be executed prior to running the imaging script.

5.3.2 For pipeline imaged deliveries

To re-run imaging using the pipeline imaging tasks, you simply run the appropriate `hif_makeimlist` and `hif_makeimages` commands from the **casa_pipescript.py** file. However, be aware that the pipeline imaging tasks are very computationally expensive and the default outputs (all sources, all spectral windows, all channels at full spectral resolution) result in very large files. For running the imaging pipeline, it is advisable to have a system with >64 GB RAM, and the available disk space needs to be 10 – 100 times the expected size of the final imaging products.

In practice, it is unlikely that the imaging pipeline commands would need to be rerun in their entirety. It would be much quicker and demand much less computing resources to only image the sources and or spectral windows (spw) or channels of interest, at an appropriate spectral resolution. This can be done by finding the corresponding `tclean()` command in the provided **casa_commands.log** file, modifying it as desired, and running it in CASA. These commands work on the measurement set created by the pipeline `hif_mstransform()` command, so that part of the imaging script would need to be run first.

5.4 Modifying the Pipeline Run

As a rule, it does not make sense to rerun the **casa_pipescript.py** exactly as delivered, since this will merely reproduce the calibrated measurement set (which is much more easily generated using **scriptForPI.py** as described in Sec. 5.1 above) and/or already-delivered products. Instead, it is likely that the user may want to redo the calibration after some modifications or produced modified imaging products. This section describes a few of the more common calibration and imaging changes. See the **ALMA Science Pipeline Reference Manual** for more complete details on the pipeline tasks and their inputs.

5.4.1 Removing ASDMs from the processing

Problematic datasets (ASDMs) can be removed from the processing by editing the `vis=` and `session=` lists in `hifa_importdata` in **casa_pipescript.py**.

5.4.2 Introducing additional flagging during calibration

Additional manual flagging can be introduced to any Pipeline reduction by editing the ***flagtemplate.txt** files. Examples of the syntax to use in editing these files are given at the top of the files. The flags are applied when `hifa_flagdata` command is run (right after `hifa_importdata`).

5.4.3 Editing calibrator fluxes used by the Pipeline

Pipeline currently reads quasar calibrator fluxes from the **flux.csv** file. If a regularly-monitored quasar has been used as the flux calibrator, the flux of this provided from the ALMA Quality Assurance process can be over-ridden by editing it in this file. Only values for the flux calibrator can be over-ridden, since values for the bandpass and phase calibrators are derived during data calibration.

5.4.4 Manipulating the Pipeline Context

It is recommended to always run the Pipeline using python scripts. New Pipeline runs/scripts need to be initialised using `h_init` in order to create an empty pipeline **Context**.

If it is wanted to halt a Pipeline run before the end of processing, the **Context** should be saved at that point using `h_save`. In order to resume the run, use `h_resume` to load the saved **Context** before resuming the run. See the **ALMA Science Pipeline Reference Manual** for more information.

If it is wanted to mainly use the Pipeline to calibrate a dataset but to e.g. insert a different bandpass table into the processing, the following procedure should be followed:

- Run the pipeline until the end of the bandpass table creation task `hifa_bandpass`.
- View the calibration tables that Pipeline will use with `hif_show_calstate`.
- Export the calibration tables Pipeline uses to a file on disk using `hif_export_calstate`.

- Edit the file to replace the name of the Pipeline-created bandpass table with the one it is wanted to use instead.
- Import the file back to the **Context** using `hif_import_calstate` and resume the processing.

5.4.5 Modifying the Pipeline Imaging Commands

The pipeline imaging commands can be modified to produce different products. Examples are given in a “CASA Guide” at https://casaguides.nrao.edu/index.php/ALMA_Imaging_Pipeline_Reprocessing. There you will find examples of the following:

- Making aggregate continuum image with all channels of all spectral windows.
- Redoing continuum subtractions with user-derived continuum ranges.
- Making a cube of subset of sources, spectral windows, with a different robust weight and channel binning factor.

6 ALMA Single Dish (Total Power) Data

6.1 Re-running Pipeline calibration tasks using scriptForPI.py

For ALMA single dish (also known as Total Power) data, there is currently no script to restore raw data to calibrated measurement sets by directly applying the Observatory-determined calibration tables. Instead, for ALMA SD data that have been calibrated and imaged by the Single Dish Pipeline, when **scriptForPI.py** is used, it will call **casa_pipescript.py** to perform a full re-calibration of the data. Please note that it may take a significant amount of processing time to re-calibrate the SD data. To know approximately how long it will take, see the “Execution Duration” which is shown on the top-page of the WebLog.

The **scriptForPI.py** calls **casa_pipescript.py**. A typical script shown in Figure 2.

```
__rethrow_casa_exceptions=True
h_init()
hsd_importdata(vis = ['uid___A002_X877e41_X452'])
hsd_flagdata(pipelinemode='automatic')
hifa_tsyscal(pipelinemode='automatic')
hifa_tsysflag(fnm_byfield=True)
hsd_skycal(pipelinemode='automatic')
hsd_k2jycal(pipelinemode='automatic')
hsd_applycal(pipelinemode='automatic')
hsd_baseline(pipelinemode='automatic')
hsd_blflag(pipelinemode='automatic')
hsd_baseline(pipelinemode='automatic')
hsd_blflag(pipelinemode='automatic')
hsd_imaging(pipelinemode='automatic')
hsd_exportdata(pipelinemode='automatic')
h_save()
```

Figure 2: Example of the Single Dish Pipeline calibration script casa_pipescript.py.

6.2 Re-running Pipeline calibration tasks using casa_pipescript.py

For SD data, the easiest way to reproduce the Single Dish Pipeline calibration performed as part of the ALMA Quality Assurance Process is to run **scriptForPI.py which can be found** in the ALMA delivery package. Alternatively, the following steps can be performed manually to reproduce the calibration, but it is essentially equivalent:

- Copy **casa_pipescript.py** from the **script** directory to the **raw** directory.
- Copy **jyperk.csv** and ***flagtemplate.txt** from calibration directory to the **raw** directory.

In the raw directory:

- Make sure the naming of the raw ALMA data is consistent with those provided in the script.
- Start the version of CASA containing Pipeline using **casapy --pipeline**, then type **execfile('casa_pipescript.py').**

Running the script will create:

- A calibrated MS for each ASDM in the same directory.
- A **pipeline-*/html** directory containing
    - The Pipeline WebLog.
    - The **casa_commands.log** file

6.3 Manual Imaging after running casa_pipescript.py

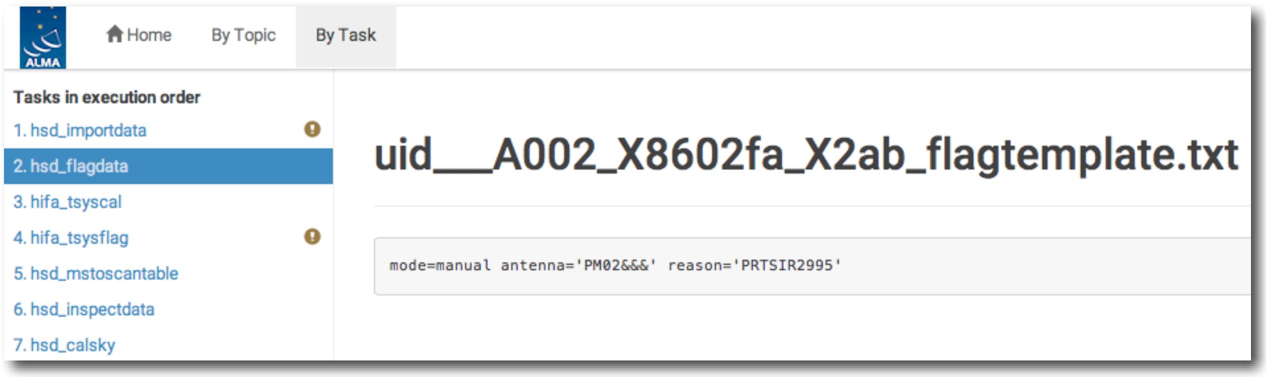
After calibration with the script casa_pipescript.py, it is possible to re-image using the CASA Single Dish task, `sdimaging`, with user-defined parameters. Single Dish Pipeline creates a calibrated MS with a suffix (filename extension) of "***.ms_bl**" for each ASDM that is specified in the **infiles** keyword parameter in the `sdimaging` command. For other parameters in `sdimaging`, refer to the **script/*casa_commands.log file** under the delivered package.

Note that the images included in the delivery package have native frequency resolution and a cell size of one-ninth of the beam size, as recommended in the SD "CASA Guide" (https://casaguides.nrao.edu/index.php/M100_Band3_SingleDish_4.3). If you want to change them to your preferable frequency resolution and cell size, we recommend that you import the delivered FITS data cubes to CASA and regrid it using the CASA task `imregrid`.

It is also possible to revise the baseline subtraction using your prefered mask range instead of the pipeline-defined range. We recommend doing this on the images using the CASA tasks `imcontsub` or `tsdbaseline` during your own manual calibration (refer to the CASA Guides).

6.4 Modifying the Pipeline Run

It is possible to insert manually (in a text file that should be called ***flagtemplate.txt**) additional flagging not identified by pipeline if necessary. These can be any valid CASA `flagdata` command. In single dish data you will be flagging autocorrelation, so be sure to add "&&*" (which means cross- and auto-correlation) or "&&&" (which means auto-correlation) in the **antenna** keyword and with the name of the antenna to flag. You can find an example in Figure 3.



uid___A002_X8602fa_X2ab_flagtemplate.txt

```
mode=manual antenna='PM02&&&' reason='PRTSIR2995'
```

Figure 3: WebLog view of hsd_flagdata showing an example of flagtemplate.txt.

7 The Pipeline WebLog

7.1 Overview

The WebLog is a set of html pages that give a summary of how the calibration of ALMA data proceeded, of the imaging products, and provides diagnostic plots and Quality Assurance (QA) scores. The WebLog will be in the **qa** directory of an ALMA delivery. To view the WebLog, untar and unzip the file using e.g. **tar zxvf *WebLog.tar.gz .** This will provide a **pipeline*/html** directory containing the WebLog, which can be viewed using a web browser e.g. **firefox index.html**.

The WebLog provides both a quick overview of datasets and also gives methods for exploring each pipeline stage in detail. Therefore most calibration pages of the WebLog will first give a single "representative" view, with further links to a more detailed view of all the plots associated with that calibration step. Some of these will have a "Plot command" link that provides the CASA command to reproduce the plot (see Figure 4). For some stages, the detailed plots can be filtered by a combination of outlier, antenna and spectral window criteria. Where histograms are displayed, in modern web browsers it is possible to draw boxes on multiple histograms to select the plots associated with those data points. All pipeline stages are assigned QA score to give an "at a glance" indication of any trouble points.

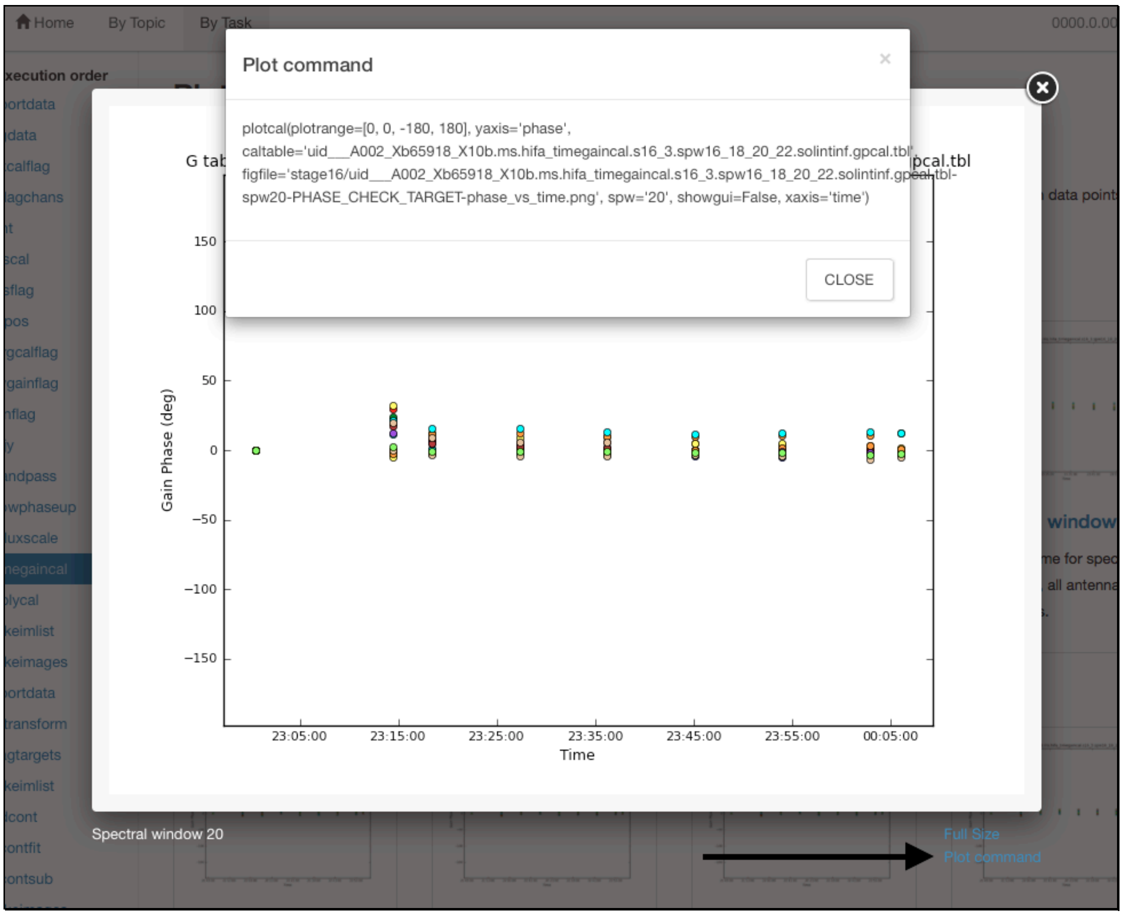
WebLog Quick Tips

Any text written in blue, including headings, is a link to further information

To go straight to viewing calibrated science target plots, go to **By Task > hif_applycal** and scroll down to the bottom

Histograms can have selector boxes drawn on them using the mouse

Commands for re-creating some WebLog plots in CASA are now provided



```
plotcal(plotrange=[0, 0, -180, 180], yaxis='phase',
caltable='uid___A002_Xb65918_X10b.ms.hifa_timegaincal.s16_3.spw16_18_20_22.solintinf.gpcal.tbl',
figfile='stage16/uid___A002_Xb65918_X10b.ms.hifa_timegaincal.s16_3.spw16_18_20_22.solintinf.gpcal.tbl-
spw20-PHASE_CHECK_TARGET-phase_vs_time.png', spw='20', showgui=False, xaxis='time')
```

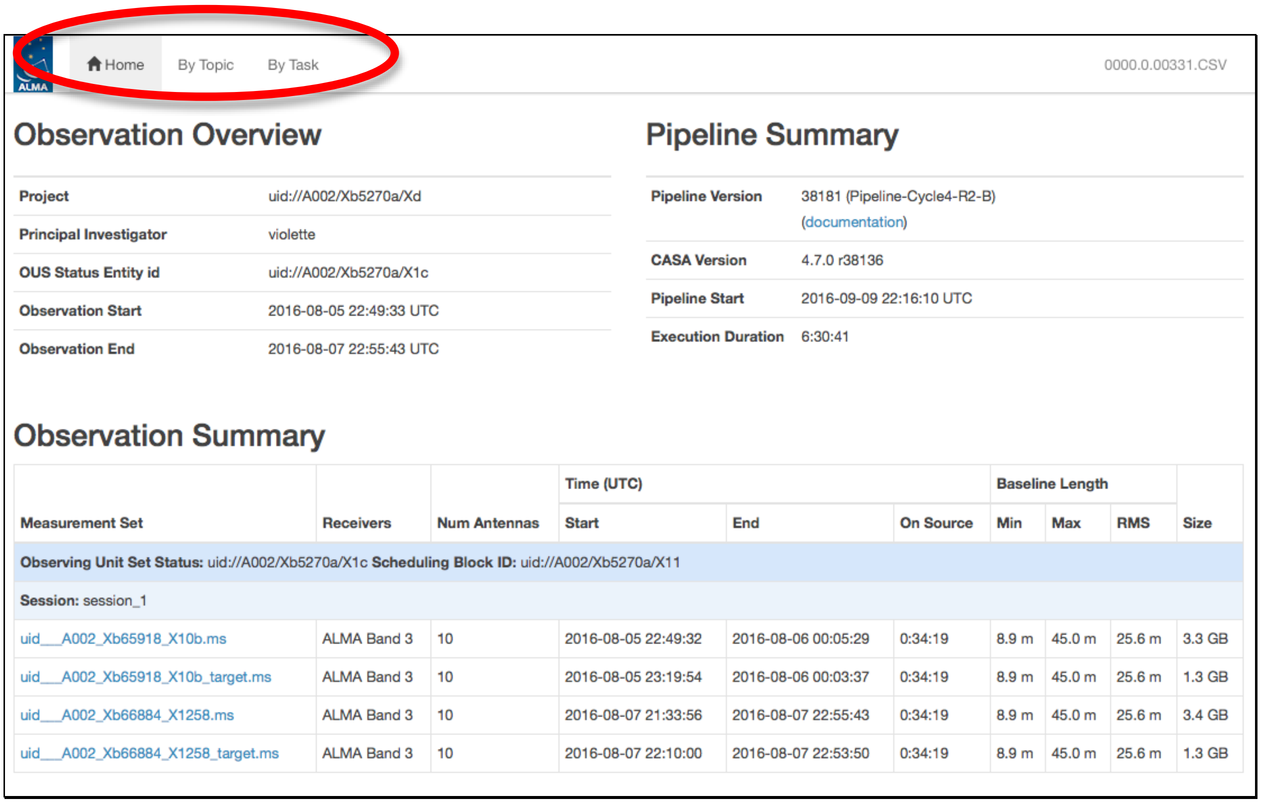
Figure 4: Example of WebLog plot with a "Plot command" link (arrow) that provides the CASA command for reproducing the plot.

7.2 Navigation

To navigate the main pages of the WebLog, click on items given in the bar at the top of the WebLog home page. Also use the **Back** button provided at the upper right on some of the WebLog sub-pages. Avoid using “back/previous page” on your web browser (although this can work on modern browsers). Throughout the WebLog, links are denoted by text written in blue and it is usually possible to click on thumbnail plots to enlarge them.

7.3 Home Page

The first page in the WebLog gives an overview of the observations (proposal code, data codes, PI, observation start and end time), a pipeline execution summary (pipeline & CASA versions, link to the current pipeline documentation, pipeline run date and duration), and an **Observation Summary** table. Clicking on the bar at the top of the home page (see Figure 5) enables navigation to **By Topic** or **By Task**.



Observation Overview

Project	uid://A002/Xb5270a/Xd
Principal Investigator	violette
OUS Status Entity id	uid://A002/Xb5270a/X1c
Observation Start	2016-08-05 22:49:33 UTC
Observation End	2016-08-07 22:55:43 UTC

Pipeline Summary

Pipeline Version	38181 (Pipeline-Cycle4-R2-B) (documentation)
CASA Version	4.7.0 r38136
Pipeline Start	2016-09-09 22:16:10 UTC
Execution Duration	6:30:41

Observation Summary

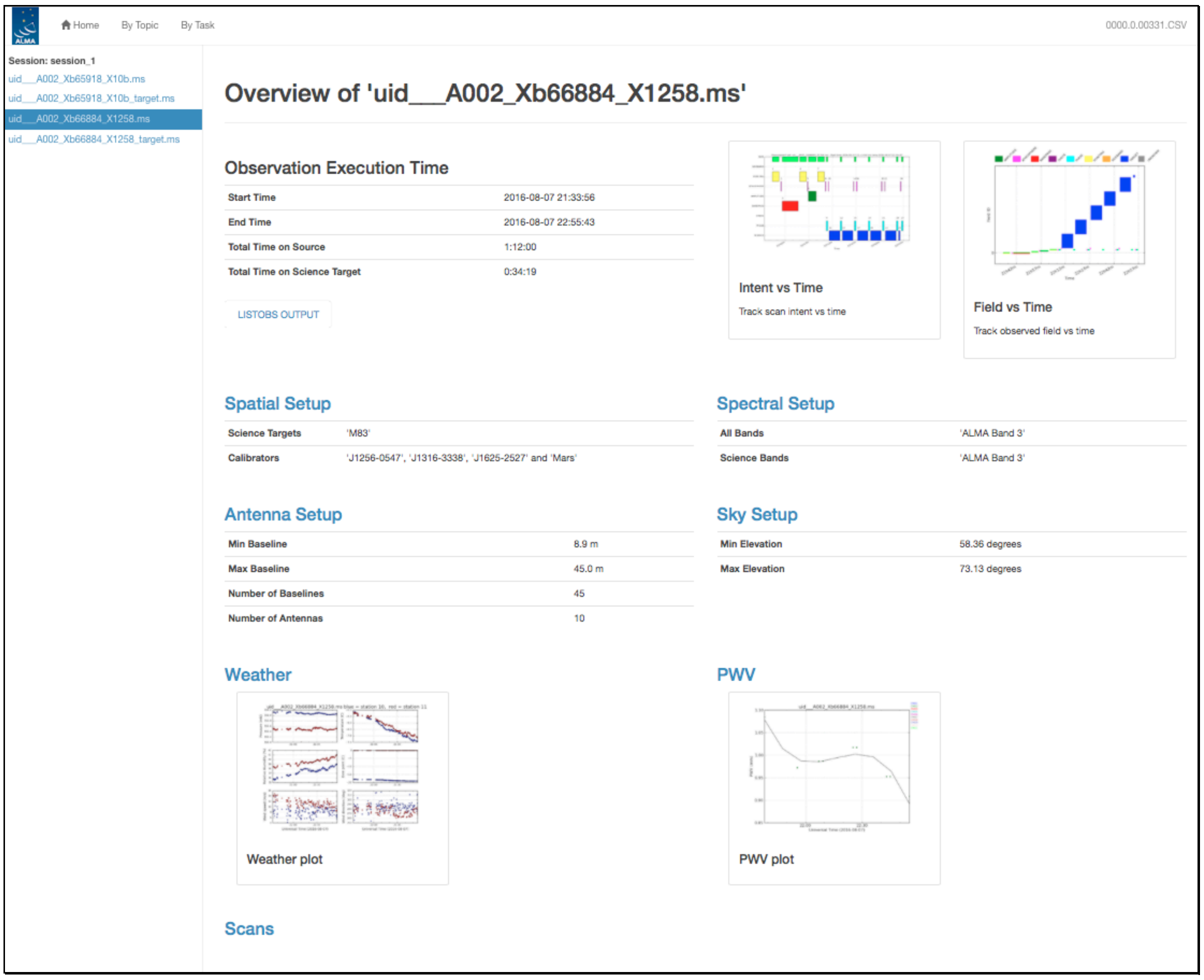
			Time (UTC)			Baseline Length			
Measurement Set	**Receivers**	**Num Antennas**	**Start**	**End**	**On Source**	**Min**	**Max**	**RMS**	**Size**
Observing Unit Set Status: uid://A002/Xb5270a/X1c **Scheduling Block ID:** uid://A002/Xb5270a/X11									
Session: session_1									
uid___A002_Xb65918_X10b.ms	ALMA Band 3	10	2016-08-05 22:49:32	2016-08-06 00:05:29	0:34:19	8.9 m	45.0 m	25.6 m	3.3 GB
uid___A002_Xb65918_X10b_target.ms	ALMA Band 3	10	2016-08-05 23:19:54	2016-08-06 00:03:37	0:34:19	8.9 m	45.0 m	25.6 m	1.3 GB
uid___A002_Xb66884_X1258.ms	ALMA Band 3	10	2016-08-07 21:33:56	2016-08-07 22:55:43	0:34:19	8.9 m	45.0 m	25.6 m	3.4 GB
uid___A002_Xb66884_X1258_target.ms	ALMA Band 3	10	2016-08-07 22:10:00	2016-08-07 22:53:50	0:34:19	8.9 m	45.0 m	25.6 m	1.3 GB

Figure 5: WebLog Home Page. The Navigation Bar is circled in red.

The **Observation Summary** table lists all the measurement sets included in the pipeline processing, grouped by observing “sessions”. Each measurement set is calibrated independently by the pipeline. For data that have been run through the imaging stages of the pipeline, two MS will be listed – the original one including all data and spectral windows, and a target.ms containing only science target data. The table provides a quick overview of the ALMA receiver band used, the number of antennas, the start/end date and time, the time spent on source, the array minimum and maximum baseline length, the rms baseline length and the size of that measurement set. To view the observational setup of each measurement set in more detail, click on the name of it to go to its overview page.

7.3.1 Measurement set Overview pages

Clicking on the measurement set name in the **Observational Summary** table brings up the **Measurement set Overview page** (Figure 6). Each measurement set **Overview** page has a number of tables: **Observation Execution Time**, **Spatial Setup (includes mosaic pointings)**, **Antenna Setup**, **Spectral Setup** and **Sky Setup (includes elevation vs. time plot)**. For more information on the tables titled in blue text, click on these links. There are additionally links to **Weather, PWV, Scans** and **Telescope Pointings** (in the case of Single Dish observations) information. Two thumbnail plots, which can be enlarged by clicking on them, show the observation structure either as **Field Source Intent vs Time** or **Field Source ID vs Time**. To view the CASA listobs output from the observation, click on **Listobs Output**.



Session: session_1

uid___A002_Xb65918_X10b.ms

uid___A002_Xb65918_X10b_target.ms

uid___A002_Xb66884_X1258.ms

uid___A002_Xb66884_X1258_target.ms

Overview of 'uid___A002_Xb66884_X1258.ms'

Observation Execution Time

Start Time	2016-08-07 21:33:56
End Time	2016-08-07 22:55:43
Total Time on Source	1:12:00
Total Time on Science Target	0:34:19

LISTOBS OUTPUT

Intent vs Time

Track scan intent vs time

Field vs Time

Track observed field vs time

Spatial Setup

Science Targets	'M83'
Calibrators	'J1256-0547', 'J1316-3338', 'J1625-2527' and 'Mars'

Spectral Setup

All Bands	'ALMA Band 3'
Science Bands	'ALMA Band 3'

Antenna Setup

Min Baseline	8.9 m
Max Baseline	45.0 m
Number of Baselines	45
Number of Antennas	10

Sky Setup

Min Elevation	58.36 degrees
Max Elevation	73.13 degrees

Weather

Weather plot

PWV

PWV plot

Scans

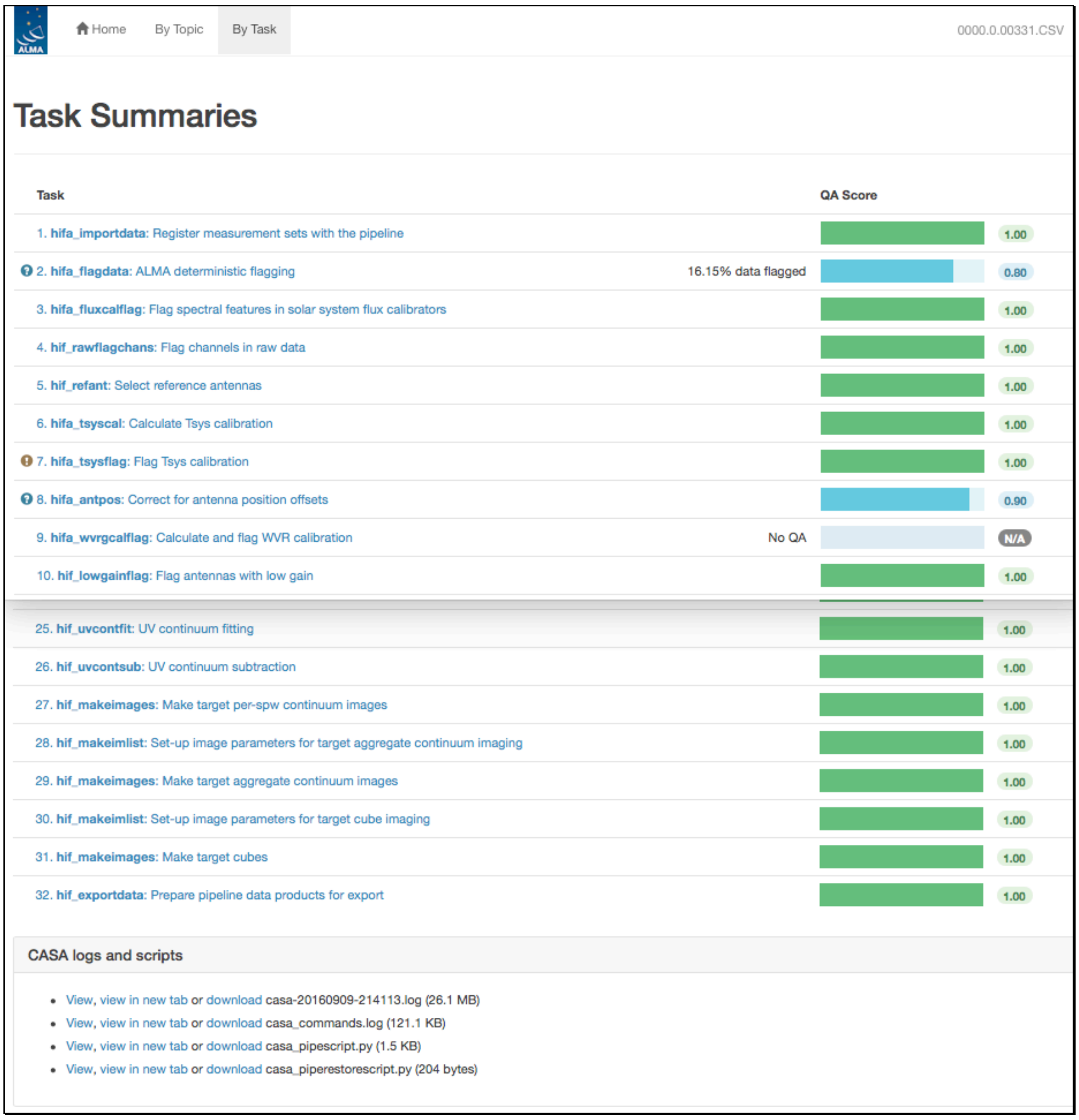
Figure 6: Measurement Set Overview Page. Click on the table headings in blue for more information about each.

7.4 By Topic Summary Page

The **By Topic** summary page provides an overview of all **Warnings and Errors** triggered, a Quality Assessment overview in **Tasks by Topic** and **Flagging Summaries** for the processing.

7.5 By Task Summary Page

The **By Task** summary page (Figure 7) gives a list of all the pipeline stages performed on the dataset. It is not displayed per measurement set as the Pipeline performs each step on every measurement set sequentially before proceeding to the next step; e.g. it will import and register all measurement sets with the Pipeline before proceeding to perform the ALMA deterministic flagging step on each measurement set. The name of each step on the By Task page is a link to more information. On the right hand side of the page are colored bars and scores that indicate how well the Pipeline processing of that stage went. Green bars should indicate a fairly problem-free dataset, while blue or red bars indicate less than perfect QA scores. Encircled symbols to the left of each task name (“?”, “!” or “x”), indicate that there are informative QA messages on the subtask pages.



Home By Topic By Task 0000.0.00331.CSV

Task Summaries

Task		QA Score
1. **hifa_importdata**: Register measurement sets with the pipeline		1.00
2. **hifa_flagdata**: ALMA deterministic flagging	16.15% data flagged	0.80
3. **hifa_fluxcalflag**: Flag spectral features in solar system flux calibrators		1.00
4. **hif_rawflagchans**: Flag channels in raw data		1.00
5. **hif_refant**: Select reference antennas		1.00
6. **hifa_tsyscal**: Calculate Tsys calibration		1.00
7. **hifa_tsysflag**: Flag Tsys calibration		1.00
8. **hifa_antpos**: Correct for antenna position offsets		0.90
9. **hifa_wvrgcalflag**: Calculate and flag WVR calibration	No QA	N/A
10. **hif_lowgainflag**: Flag antennas with low gain		1.00
25. **hif_uvcontfit**: UV continuum fitting		1.00
26. **hif_uvcontsub**: UV continuum subtraction		1.00
27. **hif_makeimages**: Make target per-spw continuum images		1.00
28. **hif_makeimlist**: Set-up image parameters for target aggregate continuum imaging		1.00
29. **hif_makeimages**: Make target aggregate continuum images		1.00
30. **hif_makeimlist**: Set-up image parameters for target cube imaging		1.00
31. **hif_makeimages**: Make target cubes		1.00
32. **hif_exportdata**: Prepare pipeline data products for export		1.00

CASA logs and scripts

- View, view in new tab or download casa-20160909-214113.log (26.1 MB)
- View, view in new tab or download casa_commands.log (121.1 KB)
- View, view in new tab or download casa_pipescript.py (1.5 KB)
- View, view in new tab or download casa_piperestorescript.py (204 bytes)

Figure 7: By Task summary view. The page has been truncated so both the top and bottom can be seen. Each pipeline stage is listed, along with its QA score (colored bars to the right), and links to the CASA logs and scripts.

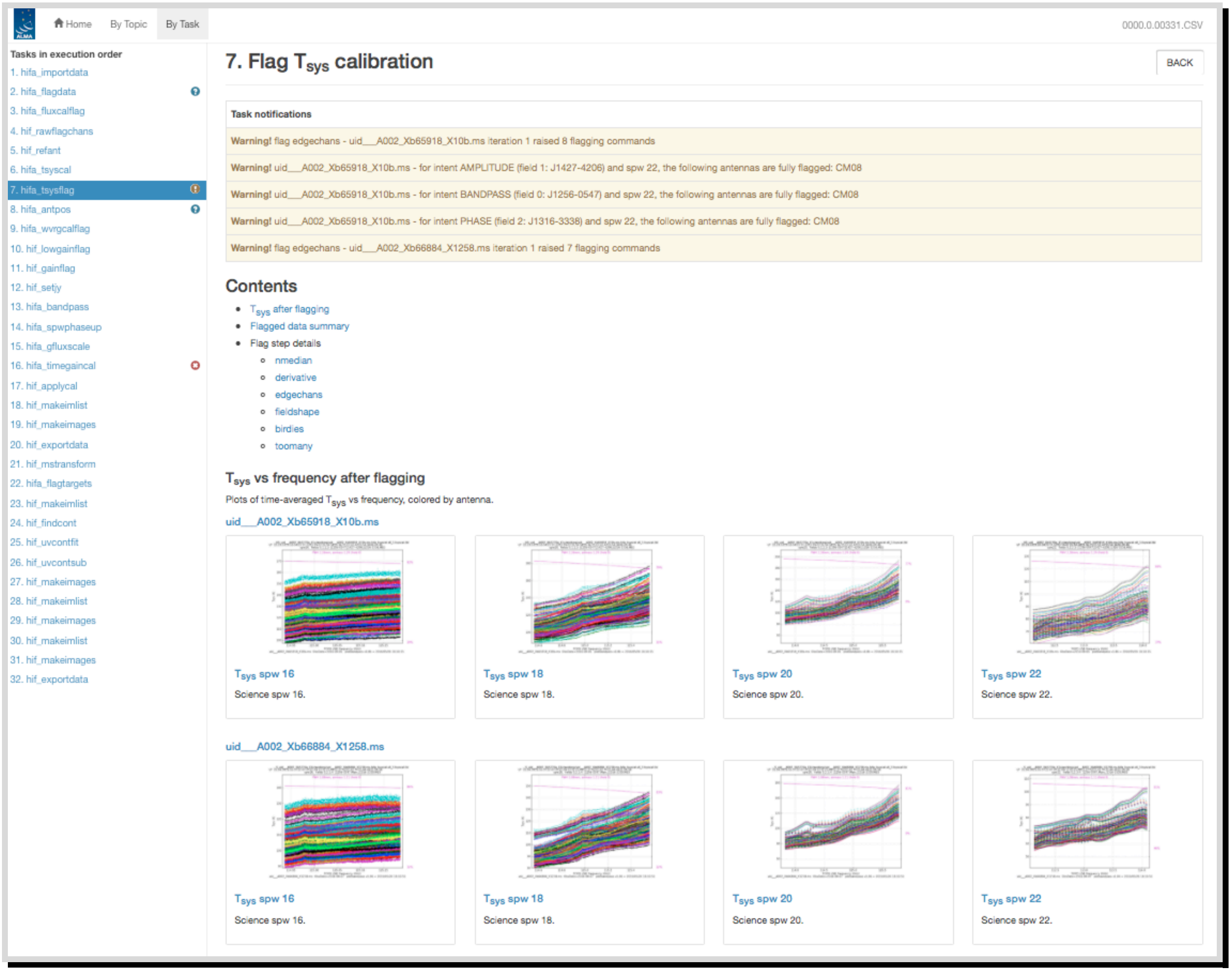
7.5.1 CASA logs and scripts

At the bottom of the **By Task** summary page are links to the CASA logs and supporting files and scripts. These include the complete CASA log file produced during the pipeline run, the **casa_commands.log** file, and the pipeline restoration scripts described in Sec. 5: **casa_pipescript.py** and **casa_piperestorescript.py**.

The **casa_commands.log** file is written by the pipeline to provide a list of the equivalent CASA task commands (as opposed to Pipeline tasks) used by the Pipeline to process a dataset. While this log cannot be used to create a CASA reduction script that is identical to the Pipeline processing, it provides executable CASA commands with the parameter settings used by the pipeline. The log is commented to indicate which Pipeline stage the tasks were called from and why. The imaging commands given in this file can be easily modified to produce new imaging products with more finely tuned inputs (e.g. interactive masks and deeper cleaning thresholds).

7.6 Task Pages

Each task has its own summary page that is accessed by clicking on the task name on the **By Task** summary page or in the left navigation menu from other pages. The task pages provide the outcome, or the representative outcome, of each Pipeline task executed. **For a fast assessment of the calibration results, go straight to the applycal page.** At the top of the page will be any Task Notification (see Figure 8). These provide informative messages or warnings generated from the QA scoring and should be reviewed carefully.


Home By Topic By Task

0000.0.00331.CSV

Tasks in execution order
1. hifa_importdata
2. hifa_flagdata
3. hifa_fluxcalflag
4. hif_rawflagchans
5. hif_refant
6. hifa_tsyscal
7. hifa_tsysflag
8. hifa_antpos
9. hifa_wvrgcalflag
10. hif_lowgainflag
11. hif_gainflag
12. hif_setjy
13. hifa_bandpass
14. hifa_spwphaseup
15. hifa_gfluxscale
16. hifa_timegaincal
17. hif_applycal
18. hif_makeimlist
19. hif_makeimages
20. hif_exportdata
21. hif_mstransform
22. hifa_flagtargets
23. hif_makeimlist
24. hif_findcont
25. hif_uvcontfit
26. hif_uvcontsub
27. hif_makeimages
28. hif_makeimlist
29. hif_makeimages
30. hif_makeimlist
31. hif_makeimages
32. hif_exportdata

7. Flag T_{sys} calibration

BACK

Task notifications

Warning! flag edgechans - uid___A002_Xb65918_X10b.ms iteration 1 raised 8 flagging commands

Warning! uid___A002_Xb65918_X10b.ms - for intent AMPLITUDE (field 1: J1427-4206) and spw 22, the following antennas are fully flagged: CM08

Warning! uid___A002_Xb65918_X10b.ms - for intent BANDPASS (field 0: J1256-0547) and spw 22, the following antennas are fully flagged: CM08

Warning! uid___A002_Xb65918_X10b.ms - for intent PHASE (field 2: J1316-3338) and spw 22, the following antennas are fully flagged: CM08

Warning! flag edgechans - uid___A002_Xb66884_X1258.ms iteration 1 raised 7 flagging commands

Contents

- T_{sys} after flagging
- Flagged data summary
- Flag step details
  - nmedian
  - derivative
  - edgechans
  - fieldshape
  - birdies
  - toomany

T_{sys} vs frequency after flagging

Plots of time-averaged T_{sys} vs frequency, colored by antenna.

uid___A002_Xb65918_X10b.ms

T_{sys} spw 16
Science spw 16.

T_{sys} spw 18
Science spw 18.

T_{sys} spw 20
Science spw 20.

T_{sys} spw 22
Science spw 22.

uid___A002_Xb66884_X1258.ms

T_{sys} spw 16
Science spw 16.

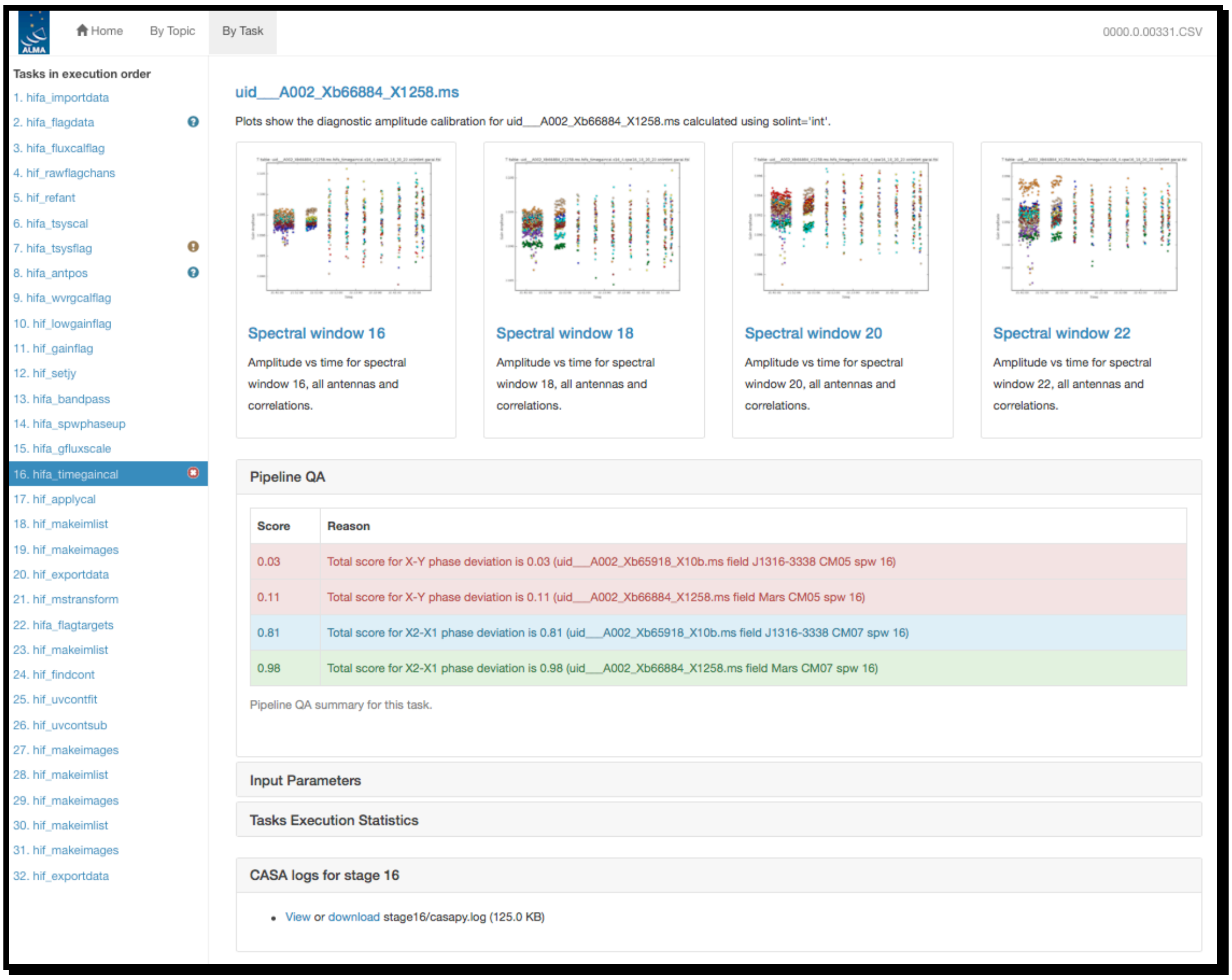
T_{sys} spw 18
Science spw 18.

T_{sys} spw 20
Science spw 20.

T_{sys} spw 22
Science spw 22.

Figure 8: hifa_tsysflag task page, showing the task notifications at the top, and diagnostic plots (Tsys for each spw grouped by MS). Further down on the page are flagging summary tables. To see the sub-page for this task, click on the measurement set name in blue above each set of plots. This will take you to a page of detailed plots for individual MS/antenna/spectral windows (see Figure 10 for an example).

At the bottom of each task page are expandable sections for **Pipeline QA**, **Input Parameters** and **Task Execution Statistics**, and links to the CASA log commands for the specific task. An example is given in Figure 9.



Home By Topic By Task 0000.0.00331.CSV

Tasks in execution order

1. hifa_importdata
2. hifa_flagdata
3. hifa_fluxcalflag
4. hif_rawflagchans
5. hif_refant
6. hifa_tsyscal
7. hifa_tsysflag
8. hifa_antpos
9. hifa_wvrgcalflag
10. hif_lowgainflag
11. hif_gainflag
12. hif_setjy
13. hifa_bandpass
14. hifa_spwphaseup
15. hifa_gfluxscale
16. hifa_timegaincal
17. hif_applycal
18. hif_makeimlist
19. hif_makeimages
20. hif_exportdata
21. hif_mstransform
22. hifa_flagtargets
23. hif_makeimlist
24. hif_findcont
25. hif_uvcontfit
26. hif_uvcontsub
27. hif_makeimages
28. hif_makeimlist
29. hif_makeimages
30. hif_makeimlist
31. hif_makeimages
32. hif_exportdata

uid___A002_Xb66884_X1258.ms

Plots show the diagnostic amplitude calibration for uid___A002_Xb66884_X1258.ms calculated using solint='int'.

Spectral window 16

Amplitude vs time for spectral window 16, all antennas and correlations.

Spectral window 18

Amplitude vs time for spectral window 18, all antennas and correlations.

Spectral window 20

Amplitude vs time for spectral window 20, all antennas and correlations.

Spectral window 22

Amplitude vs time for spectral window 22, all antennas and correlations.

Pipeline QA

Score	Reason
0.03	Total score for X-Y phase deviation is 0.03 (uid___A002_Xb65918_X10b.ms field J1316-3338 CM05 spw 16)
0.11	Total score for X-Y phase deviation is 0.11 (uid___A002_Xb66884_X1258.ms field Mars CM05 spw 16)
0.81	Total score for X2-X1 phase deviation is 0.81 (uid___A002_Xb65918_X10b.ms field J1316-3338 CM07 spw 16)
0.98	Total score for X2-X1 phase deviation is 0.98 (uid___A002_Xb66884_X1258.ms field Mars CM07 spw 16)

Pipeline QA summary for this task.

Input Parameters

Tasks Execution Statistics

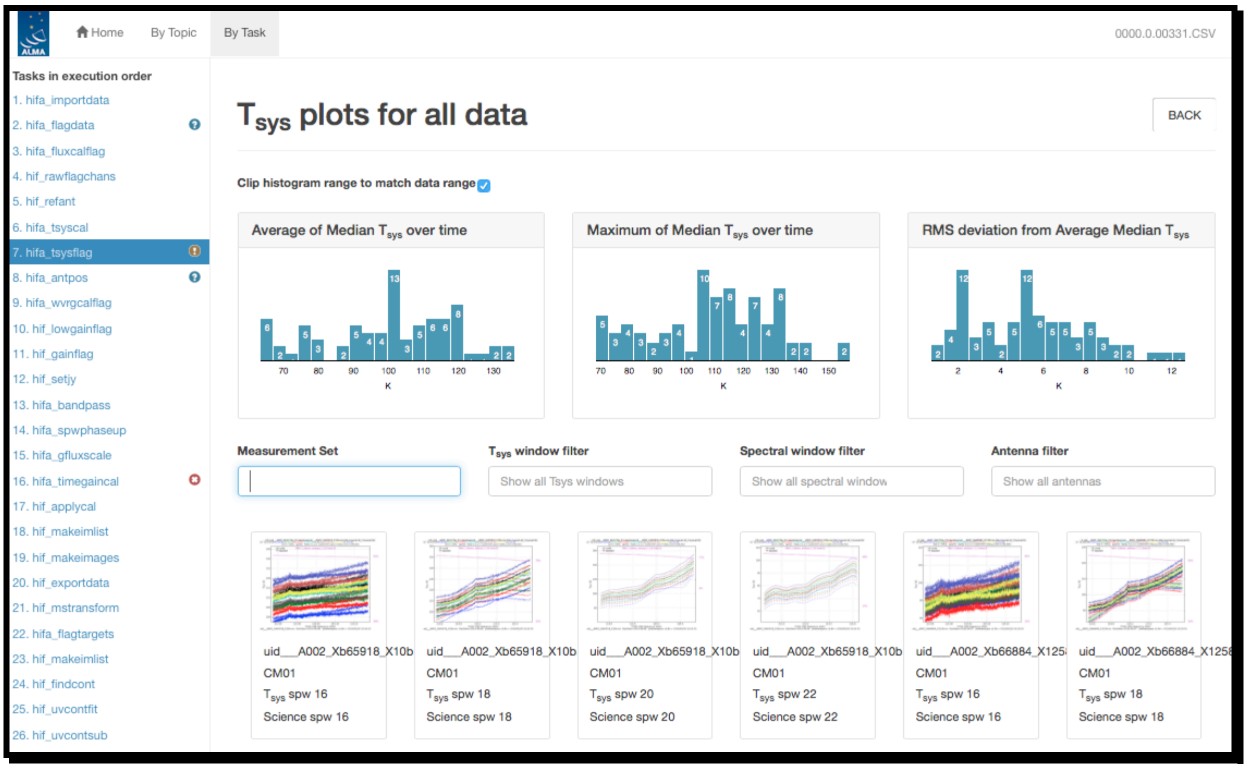
CASA logs for stage 16

- View or download stage16/casapy.log (125.0 KB)

Figure 9: Bottom of the hifa_timegaincal page, showing the expanded Pipeline QA section, as well as the expandable sections for Input Parameters, Task Execution Statistics and link to the CASA logs for this stage.

7.6.1 Task sub-pages and plot filtering

Most sub-pages have further links in order to access a more detailed view of the outcome of each task. These links are often labelled by the measurement set name. Some of these plots can be filtered by entering one or more MS, antenna, or spectral window in the appropriate box. Still others have histograms of various metrics than can be selected using the cursor in a drop-and-drag sense to outline a range of histogram values and displays the plots for the MS/antenna/spw combinations that are responsible for those histogram values. An example of these subpages and plot filtering is given in Figure 10 – Figure 12 below, using the **By Task > hifa_tsysflag: Flag Tsys calibration** pages.


Home By Topic By Task 0000.0.00331.CSV

Tasks in execution order
1. hifa_importdata
2. hifa_flagdata
3. hifa_fluxcalflag
4. hif_rawflagchans
5. hif_refant
6. hifa_tsyscal
7. hifa_tsysflag
8. hifa_antpos
9. hifa_wvrgcalflag
10. hif_lowgainflag
11. hif_gainflag
12. hif_setjy
13. hifa_bandpass
14. hifa_spwphaseup
15. hifa_gfluxscale
16. hifa_timegaincal
17. hif_applycal
18. hif_makeimlist
19. hif_makeimages
20. hif_exportdata
21. hif_mstransform
22. hifa_flagtargets
23. hif_makeimlist
24. hif_findcont
25. hif_uvcontfit
26. hif_uvcontsub

T_{sys} plots for all data BACK

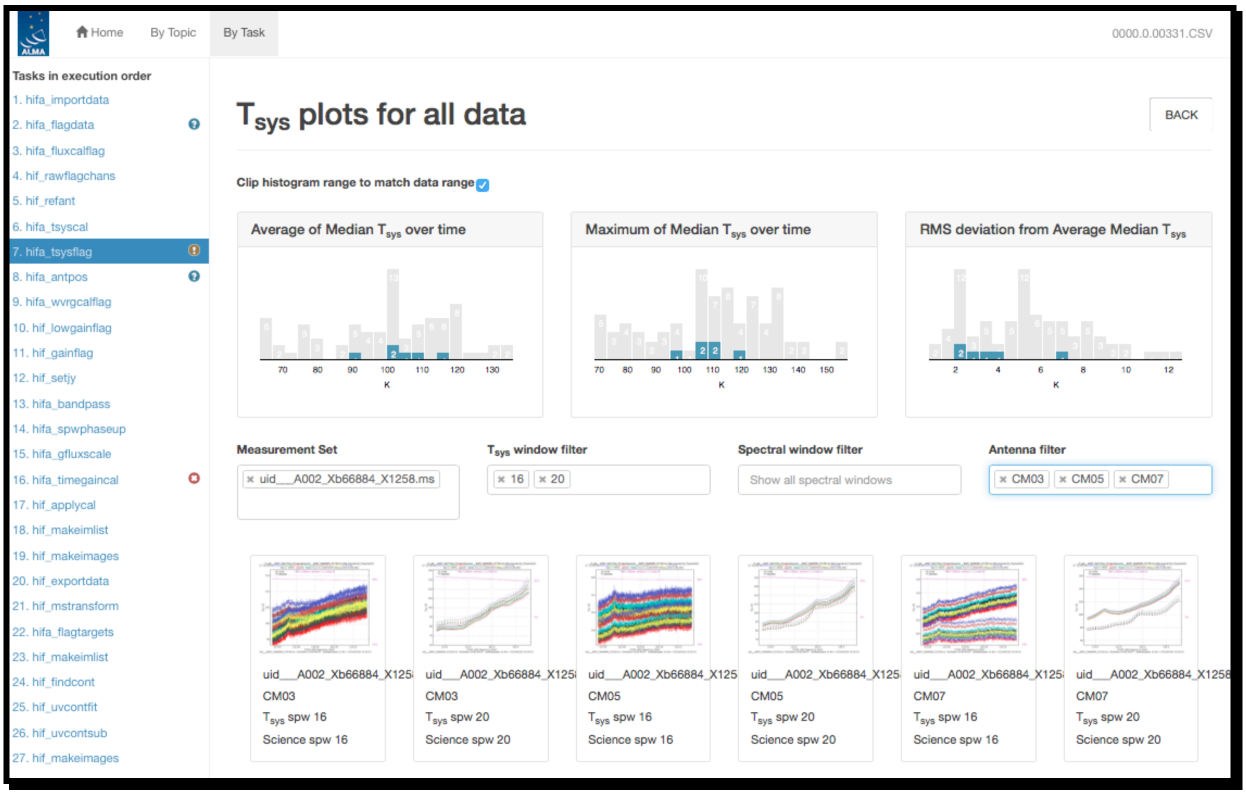
Clip histogram range to match data range ✓

Average of Median T_{sys} over time | Maximum of Median T_{sys} over time | RMS deviation from Average Median T_{sys}

Measurement Set | T_{sys} window filter: Show all Tsys windows | Spectral window filter: Show all spectral window | Antenna filter: Show all antennas

uid___A002_Xb65918_X10b CM01 T_{sys} spw 16 Science spw 16
uid___A002_Xb65918_X10b CM01 T_{sys} spw 18 Science spw 18
uid___A002_Xb65918_X10b CM01 T_{sys} spw 20 Science spw 20
uid___A002_Xb65918_X10b CM01 T_{sys} spw 22 Science spw 22
uid___A002_Xb66884_X125 CM01 T_{sys} spw 16 Science spw 16
uid___A002_Xb66884_X1258 CM01 T_{sys} spw 18 Science spw 18

Figure 10: Unfiltered view of the hifa_tsysflag sub-page. The page is arrived at by clicking on the measurement set link from the hifa_tsysflag task page (Figure 8). Only the first row of plots are shown; many more appear below (one for each MS, antenna, spw combination). This page has histograms of three metric scores based on the median Tsys that can also be used to filter the plots that are displayed.


Home By Topic By Task 0000.0.00331.CSV

Tasks in execution order
1. hifa_importdata
2. hifa_flagdata
3. hifa_fluxcalflag
4. hif_rawflagchans
5. hif_refant
6. hifa_tsyscal
7. hifa_tsysflag
8. hifa_antpos
9. hifa_wvrgcalflag
10. hif_lowgainflag
11. hif_gainflag
12. hif_setjy
13. hifa_bandpass
14. hifa_spwphaseup
15. hifa_gfluxscale
16. hifa_timegaincal
17. hif_applycal
18. hif_makeimlist
19. hif_makeimages
20. hif_exportdata
21. hif_mstransform
22. hifa_flagtargets
23. hif_makeimlist
24. hif_findcont
25. hif_uvcontfit
26. hif_uvcontsub
27. hif_makeimages

T_{sys} plots for all data BACK

Clip histogram range to match data range ✓

Average of Median T_{sys} over time | Maximum of Median T_{sys} over time | RMS deviation from Average Median T_{sys}

Measurement Set: × uid___A002_Xb66884_X1258.ms | T_{sys} window filter: × 16 × 20 | Spectral window filter: Show all spectral windows | Antenna filter: × CM03 × CM05 × CM07

uid___A002_Xb66884_X125 CM03 T_{sys} spw 16 Science spw 16
uid___A002_Xb66884_X125 CM03 T_{sys} spw 20 Science spw 20
uid___A002_Xb66884_X125 CM05 T_{sys} spw 16 Science spw 16
uid___A002_Xb66884_X125 CM05 T_{sys} spw 20 Science spw 20
uid___A002_Xb66884_X125 CM07 T_{sys} spw 16 Science spw 16
uid___A002_Xb66884_X1258 CM07 T_{sys} spw 20 Science spw 20

Figure 11: Same as Figure 10, but with a specific MS, Tsys window, and antenna filter set. The corresponding plots are displayed below, and their metric scores are shown by blue shading in the histogram plots.

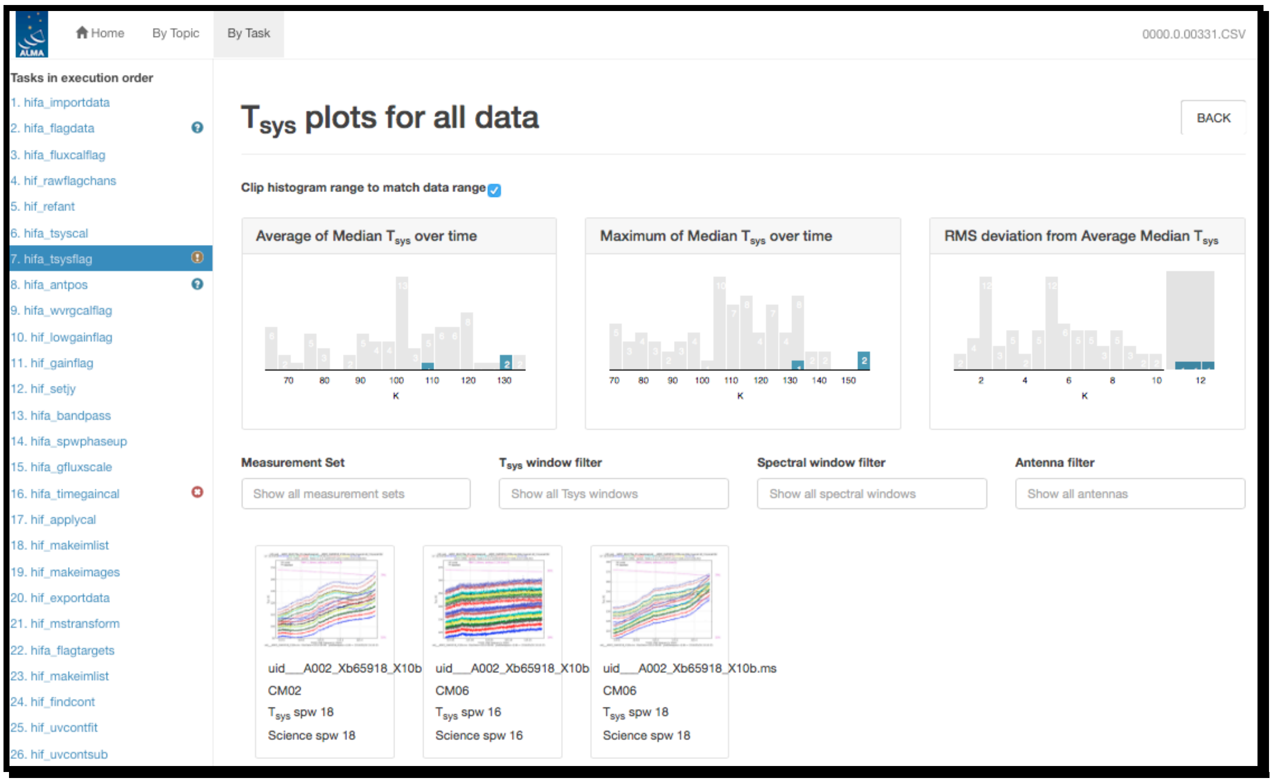


Figure 12: Same as Figure 10, but filtering to the plot of interest by using the mouse to draw a grey box on the highest histogram values in the RMS deviation from Average Median Tsys histogram plot (upper right). To clear the grey box filters on the histograms, click on any white space in the histograms.

7.7 WebLog Quality Assessment (QA) Scoring

Pipeline tasks have scores associated with them in order to quantify the quality of the dataset and the calibration. The scores are between 0.0 and 1.0 and are colourized according to the following table:

Score	Colour	Comment
0.90-1.00	Green	Standard/Good
0.66-0.90	Blue	Below standard
0.33-0.66	Yellow	Warning
0.00-0.33	Red	Error

7.7.1 Interferometric Pipeline QA Scores

Pipeline Task	Pipeline QA Scoring Metric	Score
hifa_importdata	Checking that the required calibrators are present	1.0 all present
		0.1 subtracted for missing bandpass or flux calibrator
		1.0 subtracted for missing phase calibrator or Tsys calibration
		0.5 subtracted for existing processing history

hifa_flagdata	Determining percentage of incremental flagging	0 < score < 1 === 60% < fraction flagged < 5%" (for 'online', 'shadow', 'qa0', 'before' and 'applycal') where "0 < score < 1 === HIGH% < fraction flagged < LOW%" means • Score is 0 if flag fraction is >= HIGH% • Score is 1 if flag fraction is <= LOW% Score is linearly interpolated between 0 and 1 for fractions between HIGH% and LOW%
hifa_fluxcalflag	Determining percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0
hif_rawflagchans	Determining percentage of data flagged due to deviant channels in rawdata	0% flagged: score=1.0; 100% flagged: score=0.0
hif_refant	Determining if a reference antenna centrally located and not flagged a lot	Sum of two scores. Score1: 1- [(distance from array center) / (distance of furthest antenna from array center)] Score2: 1- [(#good visibilities)/max(# good visibilities)]
hifa_tsysflag	Determining percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0
hifa_antpos	Determining if antenna positional corrections were applied	1.0 if no corrections needed; 0.9 if one or more antennas were corrected.
hifa_wvrgcalflag	Checking phase RMS improvement	0.0 if RMS(before)/RMS(after) < 1, 0.5 ... 1.0 for ratios between 1 and 2, and 1.0 for ratios > 2
hif_lowgainflag	Determining percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0
hif_gainflag	Determining percentage of incremental flagging	Score 1: Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0 Score 2: if a flagging view could be created then 1.0, otherwise 0.0
hifa_bandpass	Judging phase and amplitude solution flatness per antenna, spectral window and polarization (interim measure, future scoring will be based on data with solutions applied)	two algorithms: Wiener entropy and derivative deviation, and signal-to-noise ratio (scores: Wiener entropy: error function with 1-sigma deviation of 0.001 from 1.0; derivative deviation: error function with 1-sigma deviation of 0.03 for the outlier fraction; signal-to-noise ratio: error function with 1-sigma deviation of 1.0 for the signal-to-noise ratio)
hifa_spwphaseup	Determining fraction of spectral windows without phase solutions transferred from other windows	Score is the fraction of spectral windows for which phase solutions are unmapped to expected number of spectral windows
hifa_gfluxscale	Determining SNR of fitted flux values	Fitted flux values with SNR < 5.0 are assigned a score of 0.0, SNR > 20.0 a score of 1.0, and

		a linearly scaled value in between
hifa_timegaincal	Determining X-Y / X2-X1 phase solution deviations	Standard deviation of X-Y phase difference converted to path length: 1.0 if lower than 4.25e-6 m, 0.0 if higher than 7.955e-2 m, with an exponential decrease in between. Standard deviation of X2-X1 phase differences of subsequent integrations converted to path length: 1.0 if lower than 3.08-e-5 m, 0.0 if higher than 2.24e-2 m, with an exponential decrease in between. NB: The high limits are currently dummies. Determining their realistic values is still under development.
hifa_applycal	Determining percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5, >50%: score=0.0
hif_makeimlist	Determine if expected targets/spw will be imaged	1.0 when all objects with desired intent appear in list for all science SPW

hif_makeimages (non-checksource calibrators & science targets)	Determine if noise is close to theoretical	Ratio of sensitivity measured in non-pbcor image in a 0.3 – 0.2 PB annulus compared to 0.25 times clean threshold;
		Score=1 when ratio is 1 or lower
		Score=0 when ratio is 5 or higher

hif_makeimages (Checksources)	Determine if phase transfer worked for checksource, by checking for decorrelation and positional shift	Geometric mean of following two scores:
		Score1=1.0 – abs[(catalog position – fitted position)/beam size]
		Score2=1.0-abs[gfluxscale flux – fitted image flux)/gfluxscale flux]

hif_exportdata	Determining Pipeline products have been exported	1.0 when files successfully exported
hif_mstransform	Determine if proper files were created	1.0 when target.ms files successfully created; otherwise 0.0
hifa_flagtargets	Determine if any target flags were applied	1.0 when no flagging commands applied
hif_findcont	Determine if continuum could be identified for all spw	1.0 if continuum frequency ranges found for all spw
hif_uvcontfit	Determine if continuum could be fit	1.0 if continuum fit table created
hif_uvcontsub	Determine if continuum could be subtracted	Always set = 1.0

7.7.2 Single-Dish Pipeline QA scores

Pipeline Task	Pipeline QA Scoring Metric	Score
hsd_importdata	Checking that the required calibrators are present	1.0 ATMOSPHERE intents are present
		0.5 subtracted for existing processing history
		0.5 subtracted for existing model data
		1.0 one continuous observing session
		1.0 all source coordinate are present
hsd_flagdata	Determining percentage of incremental flagging	0 < score < 1 === 60% < fraction flagged < 5%" (for 'online', 'shadow', 'qa0', 'before' and 'applycal') where "0 < score < 1 === HIGH% < fraction flagged < LOW%" means • Score is 0 if flag fraction >= HIGH% • Score is 1 if flag fraction <= LOW% Score is linearly interpolated between 0 and 1 for fractions between HIGH% and LOW%
hsd_k2jycal	Checking that all Kelvin-to-Jy conversion factors are provided	0.0 Missing Kelvin-to-Jy conversion factor for some data
hsd_applycal	Determining percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5, >50%: score=0.0
hsd_baseline	Checking that one or more than one emission line is detected by line-finder.	1.0 there is more than one emission line detected in at least one spw.
		0.0 No line is detected in all spw.
hsd_blflag	Determining percentage of incremental flagging per source per spw.	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5, >50%: score=0.0
hsd_exportdata	Checking that the required files are exported	1.0 pipeline processing request file is exported
		1.0 pipeline WebLog file is exported
		1.0 pipeline script file is exported
		1.0 pipeline restore script file is exported
		1.0 pipeline commands log file is exported
		1.0 No missing final flag version files
		1.0 No missing final apply commands files
		1.0 No missing caltables files

8 The “By task” WebLog for Interferometric Data

This section describes navigation of the Task sub-pages for each Interferometric Pipeline task starting from the “By Task” tab. For a fuller description of each task, refer to **ALMA Science Pipeline Reference Manual**.

8.1 hifa_importdata

In this task, ASDMs are imported into measurement sets, Binary Data Flags are applied, and some properties of those MSs are calculated. The WebLog page shows a summary of imported MSs, and flux densities of calibrators. Flux densities are read from the Source table of the ASDM, which is recorded by the online system at the time of observation by interpolating in frequency the recent measurements in the calibrator catalog (see Appendix C of the ALMA C4 Technical Handbook). The flux densities for each calibrator in each science spw in each MS are written to the file flux.csv in the calibration/ subdirectory of a data delivery package. The values in this file can be edited before continuing with the pipeline execution if you first use the importonly option of eppr.executeppr.

8.2 hifa_flagdata

In this task, the online (XML format) flags, which includes the QA0 flags for antenna pointing calibration failures, are applied along with the rest of the deterministic flagging reasons (unwanted intents, autocorrelations, shadowed antennas, and TDM edge channels). The WebLog page shows the percentage of flagged data per MS. The “Before Task” column contains only the effect of the Binary Data Flags (BDF) applied during `hifa_importdata`. The additional flags are applied in the order of columns shown in the table. The percentage in each column reflects the additional amount of data flagged when applying this flag reason. The QA score for this stage is based on BDF+QA0+online+template+shadow flagging.

8.3 hifa_fluxcalflag

The WebLog shows any flagging or spwmap that was required. If the flux calibrator is a solar system object, known lines in the object (e.g. CO in Titan’s atmosphere) are flagged by this task. If >75% of a given spw is flagged on the flux calibrator for this reason, then a spwmap is calculated to transfer the flux scale from another spw. The WebLog shows if any flagging or spwmap was required.

8.4 hif_rawflagchans

This task was designed to detect severe baseline-based anomalies prior to performing antenna-based calibration. These bad data are often due to hardware problems during the observation. Outlier channels and outlier baselines are detected in the uncalibrated visibilities of the bandpass calibrator.

The WebLog page links to the images of the values used for flagging. Any flagged data are shown on the plots along with a summary of all flagging performed in this task. The following two rules are used to evaluate the need for flagging:

1) "bad quadrant" matrix flagging rule:

This starts with the "baseline" vs. "channel" flagging view. In this view, some data points may already be flagged, e.g. due to an earlier pipeline stage.

First, outliers are identified as those data points in the flagging view whose value deviates from the median value of all non-flagged data points by a threshold factor times the median absolute deviation (MAD) of the values of all non-flagged data points, where the threshold is 'fbq_hilo_limit' (default: 8.0).

In formula: *flagging mask = (data - median(all non-flagged data)) > (MAD(all non-flagged data) * fbq_hilo_limit)*

Next, the flagging view is considered as split up in 4 quadrants of channels (since some problems manifest in only one or more quadrants), and each antenna is evaluated separately as follows:

a) Select baselines belonging to antenna and select channels belonging to quadrant.
b) Determine number of newly found outlier datapoints within selection.
c) Determine number of originally unflagged datapoints within selection.
d) Determine fraction of "number of newly found outliers" over "number of originally unflagged datapoints".
e) If the latter fraction exceeds the fraction threshold 'fbq_antenna_frac_limit' (default: 0.2), then a flagging command is generated that will flag all channels within the evaluated quadrant for the evaluated antenna.
f) Otherwise, no action is taken (i.e. the newly found outlier datapoints are not individually flagged by this rule),

Next, the flagging view is still considered as split up in 4 quadrants of channels, and each baseline is evaluated separately, as follows:

a) Select baseline and select channels belonging to quadrant.
b) Determine number of newly found outlier datapoints within selection.
c) Determine number of originally unflagged datapoints within selection.
d) Determine fraction of "number of newly found outliers" over "number of originally unflagged datapoints".
e) If the latter fraction exceeds the fraction threshold 'fbq_baseline_frac_limit' (default: 1.0), then a flagging command is generated that will flag all channels within the evaluated quadrant for the evaluated baseline.
f) Otherwise, no action is taken (i.e. the newly found outlier datapoints are not individually flagged by this rule).

2) **"outlier" matrix flagging rule:**

Data points in the flagging view are identified as outliers if their value deviates from the median value of all non-flagged data points by a threshold factor times the median absolute deviation of the values of all non-flagged data points, where the threshold is 'fhl_limit' (default: 20.0).

In formula: *flagging mask = (data - median(all non-flagged data)) > (MAD(all non-flagged data) * fhl_limit)*

Flagging commands are generated for each of the identified outlier data points.

If the number of data points in the flagging view are smaller than the minimum sample 'fhl_minsample' (default: 5), then no flagging is attempted.

8.5 hif_refant

An ordered list of preferred reference antennas is calculated, with preference given to central array location and low flagging fraction. The WebLog page shows that list, and the score for each antenna can be found in the casa log for this stage.

8.6 hifa_tsyscal

System temperature (Tsys) as a function of frequency is calculated from the atmospheric calibration scan data by the online system at the time of observation. These spectra are imported to a table of the MS during `hifa_importdata`. In `hifa_tsyscal`, these spectra are copied into a CASA calibration table by the gencal task, which flags channels with zero or negative Tsys. The WebLog shows the mapping of Tsys spectral windows to science spectral windows, and plots Tsys before flagging.

8.7 hifa_tsysflag

This task flags the Tsys cal table created by the `hifa_tsyscal` pipeline task. Erroneous Tsys measurements of several different kinds are detected, including anomalously high Tsys over an entire spectral window, spikes or “birdies” in Tsys, and discrepant “shape” or Tsys as a function of frequency. Details are provided in the WebLog for each kind of flagging performed, and all of the Tsys spectra are plotted again. In these plots, all of the anomalies should be gone.

Tsysflag provides six separate flagging metrics, where each metric creates its own flagging view and has its own corresponding flagging rule(s). In the current standard pipeline, all six metrics are active, and evaluated in the order set by the parameter "metric_order" (default: 'nmedian, derivative, edgechans, fieldshape, birdies, toomany').

1) **Metric 1: "nmedian"**

   A separate view is generated for each polarisation and each spw. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is the median value of the Tsys spectrum for that antenna/time.

   The views are evaluated against the "nmedian" matrix flagging rule, where data points are identified as outliers if their value is larger than a threshold-factor * median of all non-flagged data points, where the threshold is 'fnm_limit' (default: 2.0).

   Flagging commands are generated for each of the identified outlier data points.

2) **Metric 2: "derivative"**

   A separate view is generated for each polarisation and each spw. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is calculated as follows:

   - calculate "valid_data" as the channel-to-channel difference in Tsys for that antenna/timestamp (for unflagged channels)
   - calculate *median(abs(valid_data - median(valid_data))) * 100.0*

The views are evaluated against the "max abs" matrix flagging rule, where data points are identified as outliers if their absolute value exceeds the threshold "fd_max_limit" (default: 5).

Flagging commands are generated for each of the identified outlier data points.

3) **Metric 3: "edgechans"**

A separate view is generated for each spw and each of these intents: ATMOSPHERE, BANDPASS, and AMPLITUDE. Each view contains a "median" Tsys spectrum where for each channel the value is calculated as the median value of all selected (spw,intent) Tsys spectra in that channel (this combines data from all antennas together).

The views are evaluated against the "edges" vector flagging rule, which flags all channels from the outmost edges (first and last channel) until the first channel for which the channel-to-channel difference first falls below a threshold times the median channel-to-channel difference, where the threshold is "fe_edge_limit" (default: 3.0).

A single flagging command is generated for all channels newly identified as "edge channels".

4) **Metric 4: "fieldshape"**

A separate view is generated for each spw and each polarization. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is a measure of the difference of the Tsys spectrum for that time/antenna from the median of all Tsys spectra for that antenna/spw in the "reference" fields that belong to the reference intent specified by "ff_refintent" (default: "BANDPASS").

The exact fieldshape value is calculated as: 100 * mean(abs(normalized tsys - reference normalized tsys)), where a 'normalized' array is defined as: "array / median(array)"

The views are evaluated against the "max abs" matrix flagging rule, where data points are identified as outliers if their absolute value exceeds the threshold "ff_max_limit" (default: 5).

5) **Metric 5: "birdies"**

A separate view is generated for each spw and each antenna. Each view contains a "difference" Tsys spectrum calculated as:

"channel-by-channel median of Tsys spectra for antenna within spw" - "channel-by-channel median of Tsys spectra for all antennas within spw".

The views are evaluated against the "sharps" vector flagging rule, which flags each view in two passes:

a. flag all channels whose absolute difference in value to the following channel exceeds a threshold "fb_sharps_limit" (default: 0.05).

b. around each newly flagged channel, flag neighboring channels until their channel-to-channel difference falls below 2 times the median channel-to-channel difference (this is intended to flag the wings of sharp features).

A single flagging command is generated for all channels newly identified as "birdies".

6) **Metric 6: "toomany"**

A separate view is generated for each polarisation and each spw. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is the median value of the Tsys spectrum for that antenna/time. (This is the same as for "nmedian" metric).

The views are evaluated against two separate flagging rules:

a. "tmf" (too many flags): This evaluates each timestamp one-by-one, flagging an entire timestamp when the fraction of flagged antennas within this timestamp exceeds the threshold "tmf1_limit" (default: 0.666). Flagging commands are generated per timestamp.

b. "tmef" (too many entirely flagged): This evaluates all timestamps at once, flagging all antennas for all timestamps within current view (spw, pol) when the fraction of antennas that are entirely flagged in all timestamps exceeds the threshold "tmef1_limit" (default: 0.666). Flagging commands are generated for each data point in the view that is newly flagged.

8.8 hifa_antpos

Sometimes the antenna positions were refined after the science data were recorded. If such refinements have been located, they are applied in this task. The corrections are listed in the WebLog, and the uvw values for the visibility data are recalculated.

8.9 hifa_wvrgcalflag

Water Vapor Radiometer (WVR) power measurements are converted into a phase correction table that can be applied to the science data. The phase rms during observation of the bandpass calibrator, with and without the WVR correction, is used 1) to detect poorly performing WVR units on individual antennas, and 2) to determine of the WVR correction helps overall.

The WebLog shows the effects of the phase correction in several ways, if any antennas' WVR data are flagged (the required phase correction is then interpolated from nearby antennas), and also prints a warning of the correction is deemed not helpful enough to apply at all.

8.10 hif_lowgainflag

Antennas with persistently low amplitude gains are detected and flagged. The WebLog links to grayscale images of the relative gain of each antenna calculated using the observation of the bandpass calibrator, and shows if any antennas are flagged.

This task first creates a bandpass caltable, then a gain phase caltable, and finally a gain amplitude caltable. This final gain amplitude caltable is used to identify antennas with outlier gains, for each spw. Flagging commands for outlier antennas (per spw) are applied to the entire MS.

A separate view is created for spw. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is the absolute gain amplitude for that antenna/timestamp.

The views are evaluated against the "nmedian" matrix flagging rule, where data points are identified as outliers if:

a. Their value is smaller than a threshold-factor * median of all non-flagged data points, where the threshold is 'fnm_lo_limit' (default: 0.7), or

b. Their value is larger than a threshold-factor * median of all non-flagged data points, where the threshold is 'fnm_hi_limit' (default: 1.3).

Flagging commands are generated for each of the identified outlier data points.

8.11 hif_gainflag

Antennas whose gain as a function of time shows anomalously high scatter are detected and flagged. The WebLog links to grayscale images of the gain rms per antenna, showing any that are flagged.

This task first creates a phased-up bandpass caltable, then a gain phase caltable, and finally a gain amplitude caltable. This final gain amplitude caltable is used to identify antennas with outlier gains, for each spw. Flagging commands for outlier antennas (per spw) are applied to the entire MS.

Gainflag offers two separate flagging metrics, where each metric creates its own flagging view and has its own corresponding flagging rule(s). In the Cycle 4 pipeline release, only the "rmsdeviant" metric is active. This works as follows:

a. A separate view is created for each spw. Each view is a matrix with axes "time" vs. "antenna". Each point in the matrix is the "standard deviation of the gain amplitudes for that antenna and all timestamps, divided by the median absolute deviation of the gain amplitude for all antennas and all timestamps".

b. The views are evaluated against the "max abs" matrix flagging rule, where data points are identified as outliers if their absolute value exceeds the threshold "frmsdev_limit" (default: 3.5).

Flagging commands are generated for each of the identified outlier data points

8.12 hif_setjy

The model flux density of the amplitude calibrator is set, either from an internal CASA model (solar system objects), or the results of observatory calibrator monitoring (quasars) which ultimately appear in the file flux.csv (see `hifa_importdata`). These flux densities are listed on the WebLog page, along with plots of the amplitude calibrator as a function of uv distance (which is useful to assess resolved solar system objects).

8.13 hifa_bandpass

In this task, the bandpass calibrator is self-calibrated (phase only is first calibrated on as short a time interval as allowed by signal-to-noise, listed on the WebLog page). The antenna-based bandpass phase and amplitude solution is then calculated using a S/N-dependent frequency interval, also listed on the WebLog page. Finally, the WebLog page links to plots of all of the bandpass solutions, with the atmospheric transmission curve overlaid.

8.14 hifa_spwphaseup

The relative phase offsets between spectral windows are determined for each antenna using the observation of the bandpass calibrator. (The offset is assumed to be constant in time during each execution.) If narrow spectral windows are present, a mapping is determined so that the calculated phase calibration as a function of time can be subsequently transferred (during subsequent gaincal and applycal tasks) from wider, higher S/N spectral windows to the narrow ones. If any such reference spwmaps are required, then they are listed on the WebLog page.

8.15 hifa_gfluxscale

In this task, the flux densities of the phase and bandpass calibrators are calculated and listed on the WebLog page. In this way, the absolute flux scale is transferred from the amplitude calibrator to the other calibrators and ultimately to the science target (via the phase calibrator). If the absolute flux calibrator is resolved (solar system object), plots of its amplitude as a function of uv distance are shown, and only data on short baselines are used to calculate the flux densities of the secondary calibrators. Note that phase-only self-calibration is performed on all calibrators prior to this flux calculation.

8.16 hifa_timegaincal

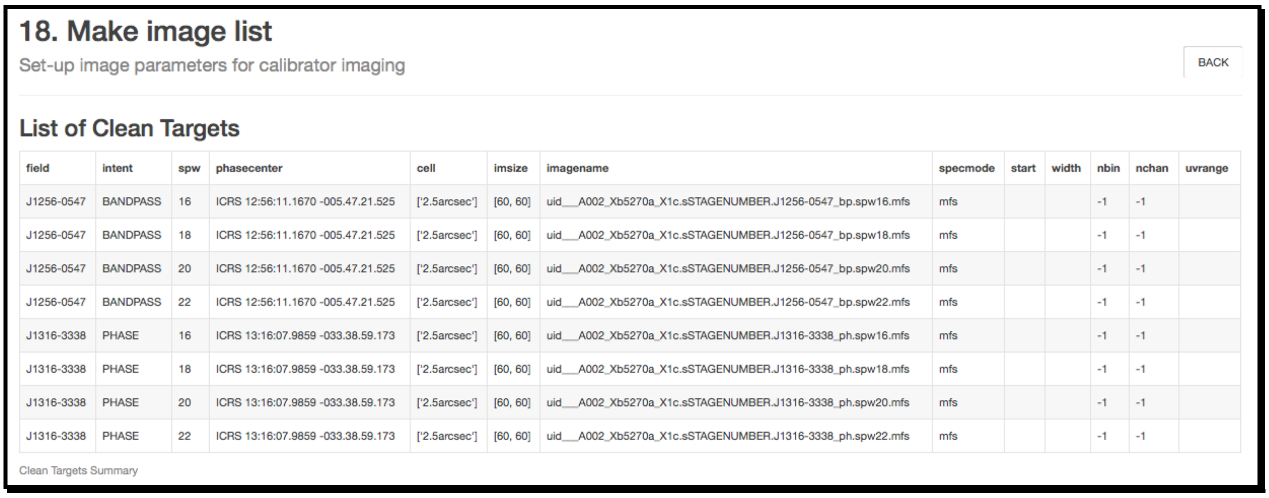
In this task, gain as a function of time is calculated from observations of the phase calibrator. The WebLog page shows plots of this gain, both on a scan timescale (as will be interpolated to the science target), and on an integration timescale (useful for assessing weather and calibration quality).

8.17 hif_applycal

In this task, all previously calculated calibration tables are applied to the science data. Any failed calibration solutions, and flagged Tsys scans, will result in flagging of actual science data in this stage, so the WebLog shows a summary of that additional flagging, and high flagging will result in a low QA score. The WebLog page also includes many useful plots of the calibrated data as a function of time and frequency. Outliers in these plots can indicate any remaining bad data.

8.18 hif_makeimlist: Set-up parameters for calibrator images

This stage determines image parameters (image size, cell size, etc) to be used in the subsequent `hif_makeimages` stage, and reports them on the WebLog page (See Figure 13). The "specmode" can be mfs for per-spw continuum multi-frequency synthesis images, "cont" for mfs continuum images of several spectral windows, or "cube" for spectral cubes. The first time the task is run is in preparation for making per-spw mfs images of the calibrators.


18. Make image list

Set-up image parameters for calibrator imaging

BACK

List of Clean Targets

field	intent	spw	phasecenter	cell	imsize	imagename	specmode	start	width	nbin	nchan	uvrange
J1256-0547	BANDPASS	16	ICRS 12:56:11.1670 -005.47.21.525	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1256-0547_bp.spw16.mfs	mfs			-1	-1	
J1256-0547	BANDPASS	18	ICRS 12:56:11.1670 -005.47.21.525	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1256-0547_bp.spw18.mfs	mfs			-1	-1	
J1256-0547	BANDPASS	20	ICRS 12:56:11.1670 -005.47.21.525	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1256-0547_bp.spw20.mfs	mfs			-1	-1	
J1256-0547	BANDPASS	22	ICRS 12:56:11.1670 -005.47.21.525	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1256-0547_bp.spw22.mfs	mfs			-1	-1	
J1316-3338	PHASE	16	ICRS 13:16:07.9859 -033.38.59.173	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1316-3338_ph.spw16.mfs	mfs			-1	-1	
J1316-3338	PHASE	18	ICRS 13:16:07.9859 -033.38.59.173	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1316-3338_ph.spw18.mfs	mfs			-1	-1	
J1316-3338	PHASE	20	ICRS 13:16:07.9859 -033.38.59.173	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1316-3338_ph.spw20.mfs	mfs			-1	-1	
J1316-3338	PHASE	22	ICRS 13:16:07.9859 -033.38.59.173	['2.5arcsec']	[60, 60]	uid___A002_Xb5270a_X1c.sSTAGENUMBER.J1316-3338_ph.spw22.mfs	mfs			-1	-1	

Clean Targets Summary

Figure 13: Example of the WebLog for the `hif_makeimlist` stage. This example is for setting up the parameters for calibrator per-spw mult-frequency synthesis (mfs) continuum images.

8.19 hif_makeimages: Make calibrator images

This stage actually creates the images requested by the most recent `hif_makeimlist`. The first time it is run is to create per-spw mfs continuum images of the calibrators. See the Stage 27 description for more information and examples of the `hif_makeimages` stage. Low QA scores for non-Check source calibrators may indicate the need for additional flagging.

8.20 hif_exportdata

Calibration tables, calibrator images (exported in fits format), and other products are moved from the pipeline /working to the /products directory.

(Subsequent stages only present if the imaging pipeline was run)

8.21 hif_mstransform

For each execution, calibrated visibilities for the science target(s) are split to a new MS with "target.ms" in the name, as listed on the front WebLog page.

8.22 hifa_flagtargets

Flagging of the science target data, if determined to be necessary by an observatory scientist, is performed as listed in the ***flagtargetstemplate.txt** files linked to the WebLog page. The WebLog also shows a summary table of any flagging performed.

8.23 hif_makeimlist: Set-up parameters for target per-spw continuum imaging

Imaging parameters are determined and listed for creation of per-spw mfs continuum images of each science target. This run of `hif_makeimlist` also controls the parameters used to create the dirty cubes used by the `hif_findcont` stage, including any channel binning (listed in the "nbins" column of the `hif_makeimlist` table).

8.24 hif_findcont

In this task, dirty image cubes are created for each spectral window of each science target. The cubes are made at the native channel resolution unless the nbins parameter was used in the preceding `hif_makeimlist` stage. The signal as a function of frequency is determined and plotted on the WebLog page as a spectrum (see examples in Figure 14). This is not simply a mean spectrum of the target, but rather a quantity that is more sensitive to spectral features. In most cases, it is the mean spectrum above a S/N level that depends on the number of channels. If no features are found, the spectrum is recomputed over the central area of the cube in search of faint compact emission. An alternative algorithm (the per-channel peak / MAD) is invoked in certain situations, including when significant atmospheric lines are present. In either case, frequency ranges are calculated that are the least likely to contain any line emission or absorption, and these are listed in the LSRK frame on the WebLog page, as well as being indicated by the cyan colored horizontal line(s) on the spectra.

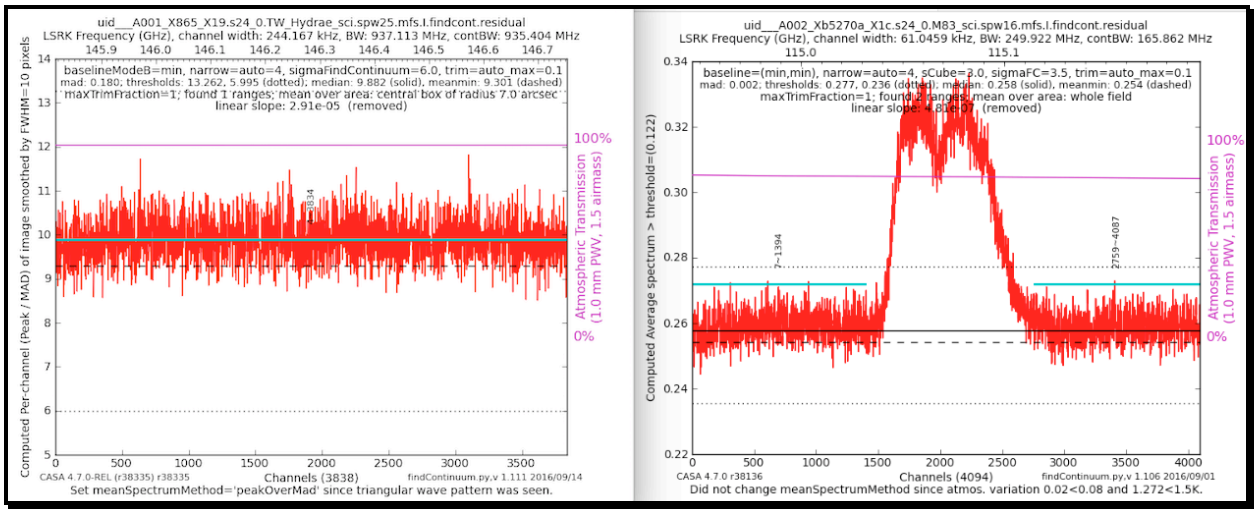


Figure 14: Two examples of `hif_findcont` plots, one with the entire window identified as continuum (left), and another with two identified continuum regions (right; identified continuum indicated by cyan lines).

The continuum frequency ranges are also printed to a file called "cont.dat". If this file already exists before `hif_findcont` is executed, then it will first examine the contents. For any spw that already has frequency ranges defined in this file, it will not perform the analysis described above in favor of the a priori ranges. For spws not listed in a pre-existing file, it will analyze them as normal and update the file. In either case, the file cont.dat is used by the subsequent `hif_uvcontfit` and `hif_makeimages` stages.

8.25 hif_uvcontfit

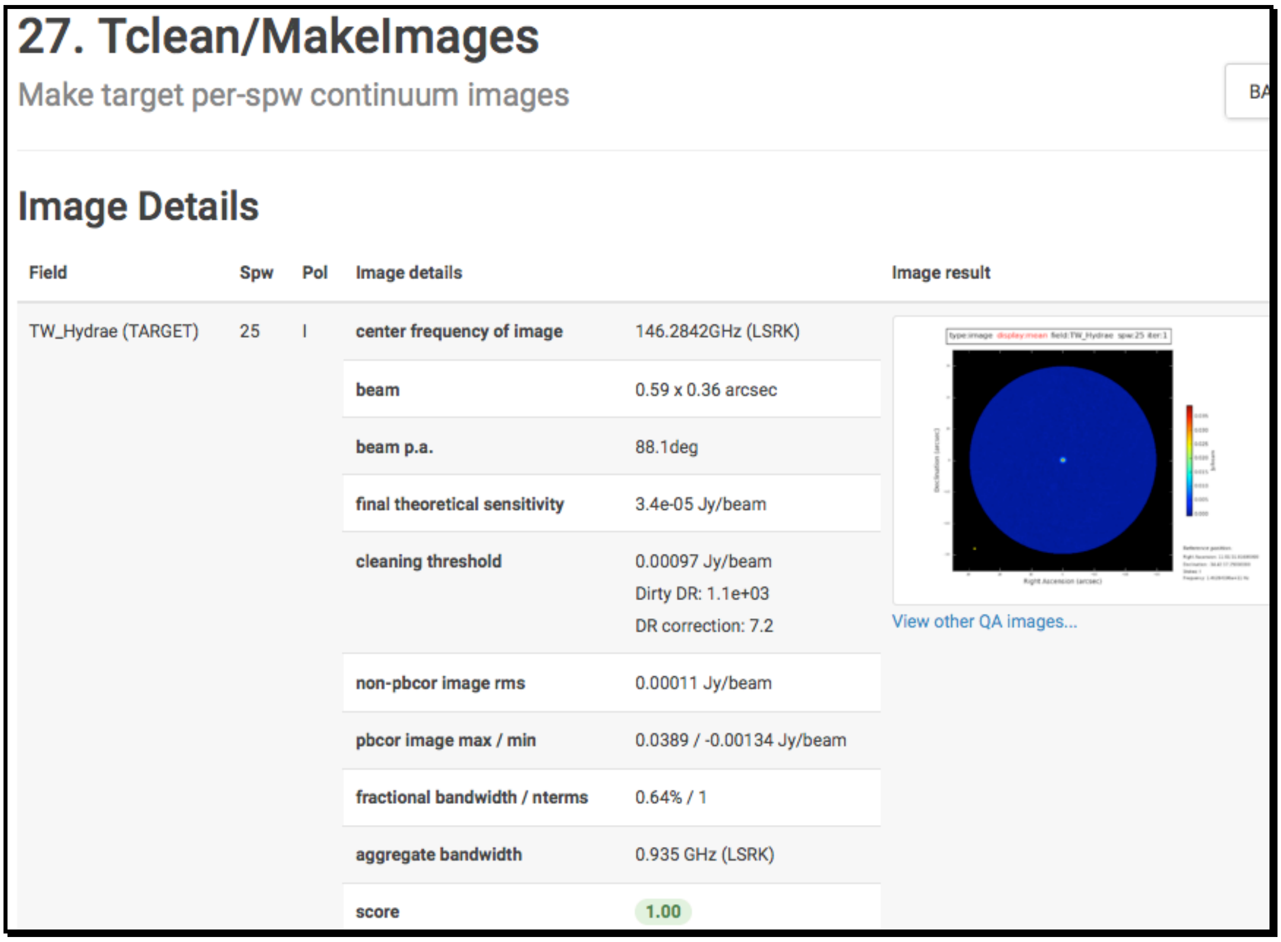
The previously determined continuum frequency ranges as shown in the cont.dat file are used to fit the continuum of each visibility. The fit is performed for each spw independently using a fitorder=1, and a calibration table is used to store the resulting fits called "uvcont.tbl". The WebLog for this stage reports the continuum ranges from `hif_findcont` in LSRK (cont.dat) but translated to the topocentric (TOPO) frame for each MS.

8.26 hif_uvsub

The `hif_uvcontfit` calibration table is applied to the data. After this step, the original continuum + line emission is contained in the DATA column of the MS, while the continuum subtracted data are written to the CORRECTED column.

8.27 hif_makeimages: Make target per-spw continuum images

Cleaned continuum images are created for each spectral window, each science target, using the continuum frequency ranges determined from `hif_findcont` (as written in the cont.dat file) from the DATA column. The resulting non-primary beam corrected images are displayed on the WebLog page. For each image, the properties are shown next to the associated image png (see Figure 15). In particular, the center frequency, beam, theoretical sensitivity, cleaning threshold (4 x theoretical sensitivity x DR correction), dynamic range of dirty image, observed rms noise measured in the non-primary beam corrected image in an annulus between the 0.3 to 0.2 response point of the primary beam, image max /min of the primary beam corrected image, fractional bandwidth, aggregate bandwidth, and image QA score are shown.



27. Tclean/MakeImages

Make target per-spw continuum images

Image Details

Field	Spw	Pol	Image details		Image result
TW_Hydrae (TARGET)	25	I	center frequency of image	146.2842GHz (LSRK)	View other QA images...
			beam	0.59 x 0.36 arcsec	
			beam p.a.	88.1deg	
			final theoretical sensitivity	3.4e-05 Jy/beam	
			cleaning threshold	0.00097 Jy/beam Dirty DR: 1.1e+03 DR correction: 7.2	
			non-pbcor image rms	0.00011 Jy/beam	
			pbcor image max / min	0.0389 / -0.00134 Jy/beam	
			fractional bandwidth / nterms	0.64% / 1	
			aggregate bandwidth	0.935 GHz (LSRK)	
			score	1.00	

Figure 15: Example of `hif_makimages` WebLog page for per-spw images. Clicking on the thumbnail will enlarge the image. Clicking on the "View other QA images" link will bring up the detailed image page (Figure 16).

The “View Other QA Images’ links for each image show the primary beam corrected image, residual, clean mask (red area), dirty image, primary beam, psf, and clean model (Figure 16).

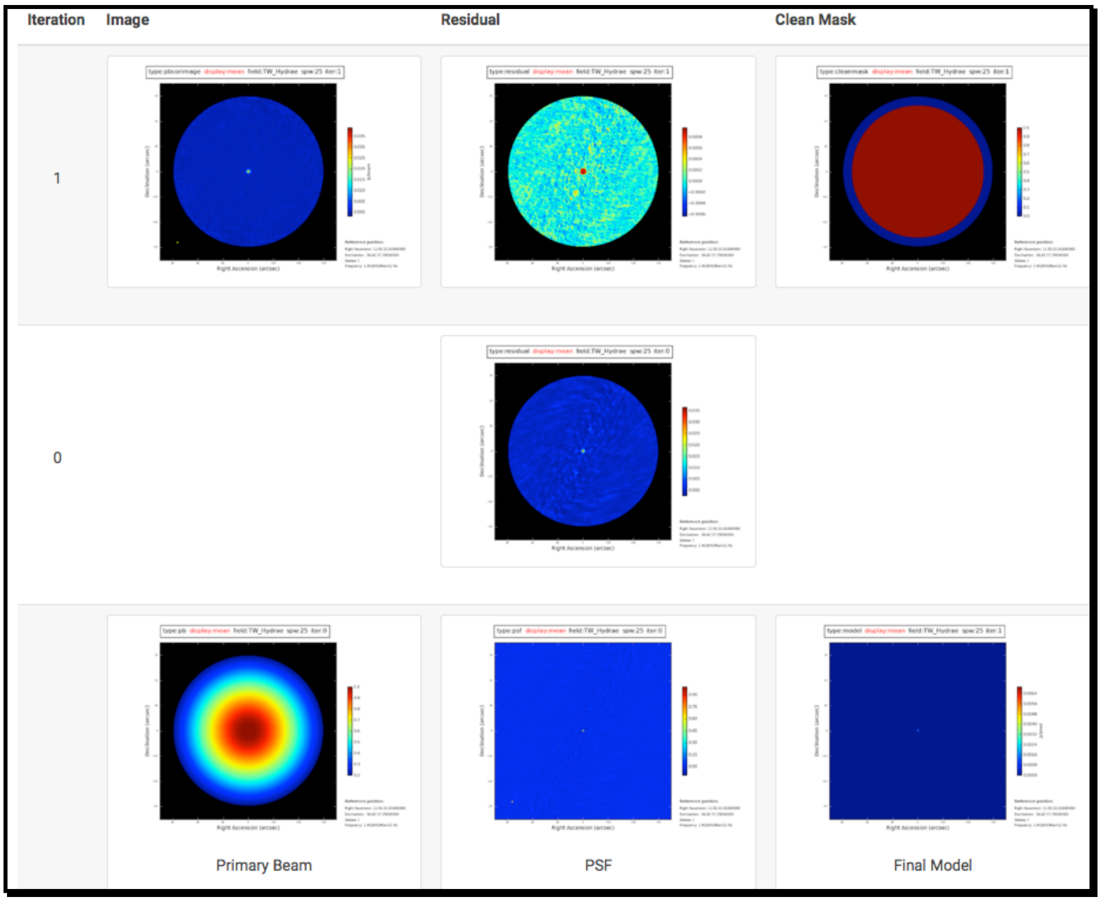


Figure 16: Details page that is displayed after clicking on the “View other QA images” link on the `hif_makimages` WebLog page.

8.28 hif_makeimlist: Set-up parameters for target aggregate continuum images

Imaging parameters are calculated and listed for creation of an aggregate (all spectral windows combined) continuum image (specmode=’cont’) of each science target.

8.29 hif_makeimages: Make target aggregate continuum images

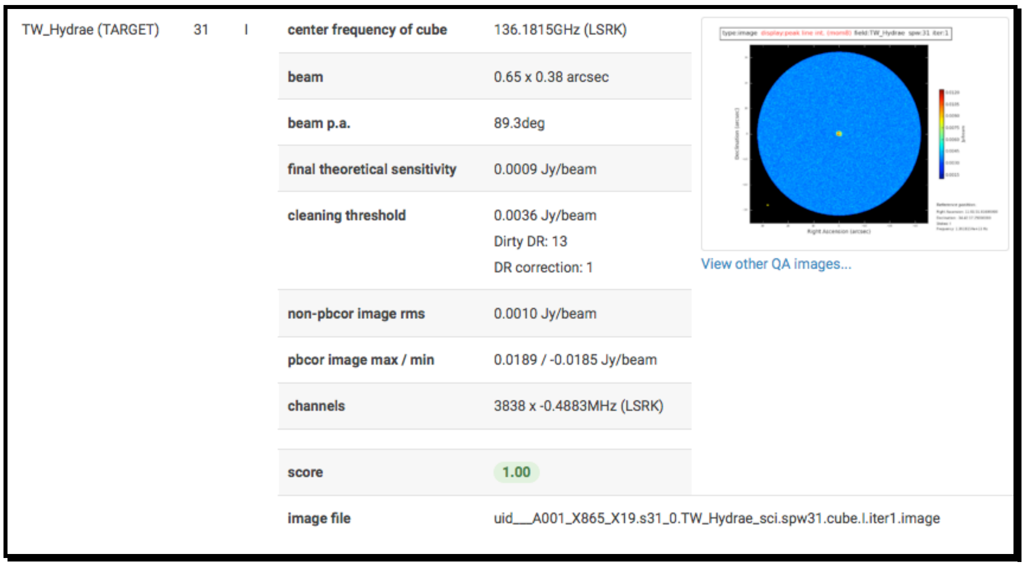
A cleaned aggregate continuum image of each science target is formed from the `hif_findcont` channels (as listed in the cont.dat file) is created. The aggregate continuum image(s) are made with nterms=2 if the fractional bandwidth is ≥ 10% (only currently possible for ALMA Bands 3 and 4 data). The resulting non-primary beam corrected images are displayed on the WebLog page. The “View Other QA Images” links show the primary beam corrected image, psf, clean model, dirty image, and residual image (Figure 16).

8.30 hif_makeimlist: Set-up image parameters for target cube imaging

Parameters are calculated and listed for creation of spectral cube images of each continuum-subtracted spectral window of each science target.

8.31 hif_makeimages: Make target cubes

Cleaned continuum-subtracted cubes are created for each science target and spectral window at the native channel resolution (unless channel binning has been selected using nbins in the preceding `hif_makeimlist`) from the CORRECTED column. Cubes are made in the radio LSRK frequency frame. Only channels that have not been designated as continuum channels are cleaned. The WebLog page displays non-primary beam corrected peak intensity images for each cube (“moment 8”) along with properties of the cubes (see Figure 17). The information is similar to that shown in Stage 27 for continuum images, except that the noise is the median rms over all channels (still measured in a 0.3 – 0.2 PB annulus), and instead of fractional and aggregate bandwidth the “channel” information is given as the number of channels imaged times the channel width. Recall that if no online nor nbins (pipeline option) channel averaging is done, the velocity resolution will be twice the channel width.



TW_Hydrae (TARGET)	31	I	**center frequency of cube**	136.1815GHz (LSRK)
			beam	0.65 x 0.38 arcsec
			beam p.a.	89.3deg
			final theoretical sensitivity	0.0009 Jy/beam
			cleaning threshold	0.0036 Jy/beam Dirty DR: 13 DR correction: 1
			non-pbcor image rms	0.0010 Jy/beam
			pbcor image max / min	0.0189 / -0.0185 Jy/beam
			channels	3838 x -0.4883MHz (LSRK)
			score	1.00
			image file	uid___A001_X865_X19.s31_0.TW_Hydrae_sci.spw31.cube.I.iter1.image

View other QA images...

Figure 17: Example of `hif_makimages` WebLog page for image cubes.

In addition to the “View other QA images” for continuum images demonstrated for Stage 27, an additional plot is included for continuum subtracted cubes: an integrated intensity (“moment 0”) image using the `hif_findcont` continuum frequency ranges (labeled “Line-free Moment 0”; see Figure 18), which should be noise-like if the continuum subtraction worked well.

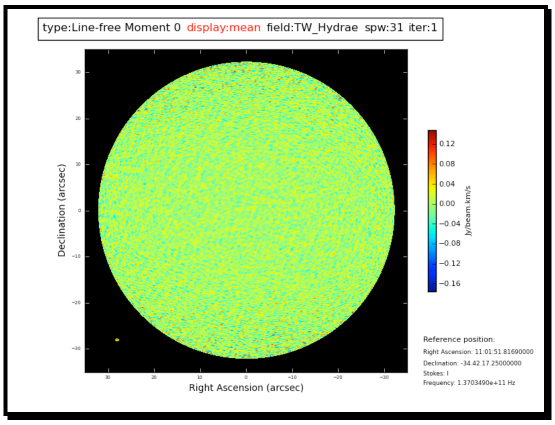


Figure 18: Example of a line-free moment 0 map shown on the details page of image cubes.

8.32 hif_exportdata

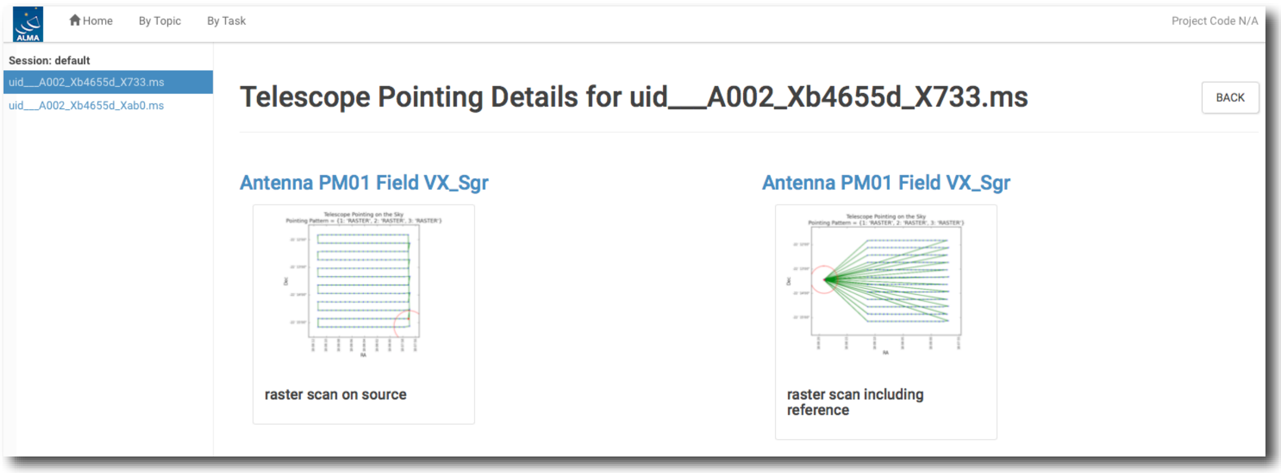
Science target images are converted to fits format and copied to the /products subdirectory as well as the cont.dat file from the `hif_findcont` stage.

9 The "By task" WebLog for Single-Dish Data

This section describes navigation of the Task sub-pages for each Single Dish Pipeline task starting from the "By Task" tab. For a fuller description of each task, refer to the **ALMA Science Pipeline Reference Manual**.

9.1 hsd_importdata

The WebLog for `hsd_importdata` task shows the summary of imported MSs, grouping of spws to be reduced as a group[1], and spw matching between Tsys and science spws. This task also generates figures of Telescope Pointings, which are available in the MS Summary page (i.e. from the Home page, click the MS name, and then click on "Telescope Pointing"). There are two types of plots that can be found containing full information on all pointings and just on-source pointings (Figure 19). In these plots, the red circle indicates the beam size of the antennas and its location is the starting position of the raster scan. The Red (small) dot indicates the last position of the raster. The green line represents the antenna slewing motion, and in the right panel of Figure 19 the green line going to/from the red dot indicates that the antenna goes to the last scan and returns to the OFF position.



Home
By Topic
By Task
Project Code N/A
Session: default
uid___A002_Xb4655d_X733.ms
uid___A002_Xb4655d_Xab0.ms
Telescope Pointing Details for uid___A002_Xb4655d_X733.ms
BACK
Antenna PM01 Field VX_Sgr
raster scan on source
Antenna PM01 Field VX_Sgr
raster scan including reference

Figure 19: The detailed page of Telescope Pointing in the MS summary page.

9.2 hsd_flagdata

The WebLog for the `hsd_flagdata` task shows the summary of flagged data percentage per MS due to binary data and online flagging, manually inserted file (***flagtemplate.txt**), shadowing, unwanted intents, and edge channels. Note that the value in the "Before Task" column corresponds to the percentage of flagged data by binary data flagging.

[1] When several EBs are processed at once, the Pipeline needs to group their respective spws, based on spw Name.

9.3 hifa_tsyscal

It shows the associations of Tsys and science spectral windows to be used for Tsys (amplitude-scale) calibration, and also shows the original Tsys spectra per spectral window.

9.4 hifa_tsysflag

It shows the flagged Tsys spectra per spectral window after heuristic flagging is applied.

9.5 hsd_skycal

The WebLog shows the integrated OFF spectra per spw and per source. The y-axis is the direct output from the correlator, which means the values are dominated by signals from both the atmosphere and receivers (Figure 20). The different colors indicate different scans (times).

The time-averaged plots of the OFF spectra are also shown in this page for the purpose of assessing the time variability of the spectra. The different colors here indicate different spws. Note that the OFF spectrum is not averaged over the spectral windows yet, but it will be in the future.

The coordinates of the OFF position can be confirmed in the Reference Coordinates table.

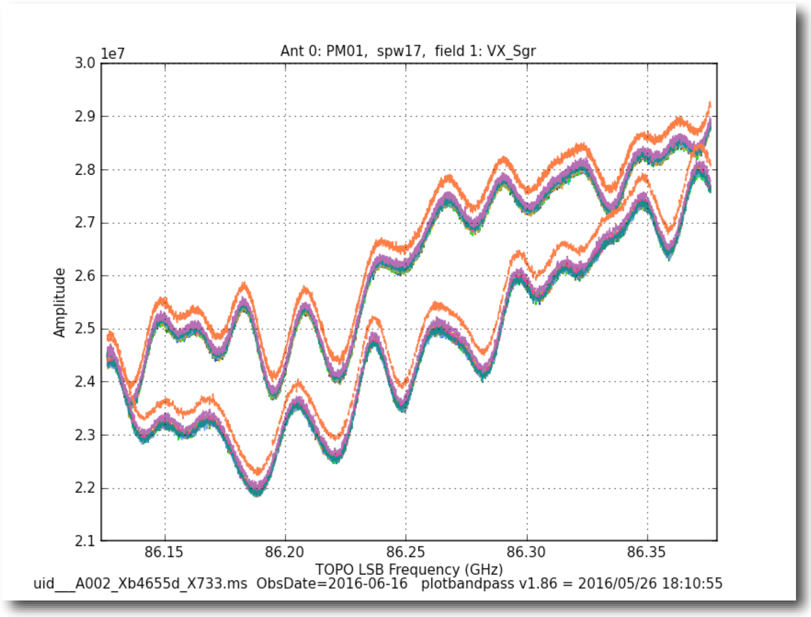


Figure 20: An example of OFF spectrum

9.6 hsd_k2jycal

This page shows the list of Kelvin to Jy conversion factors that Pipeline has read from a file “jyperk.csv”, which shall contain the factors per spw, per antenna, and per polarization.

9.7 hsd_applycal

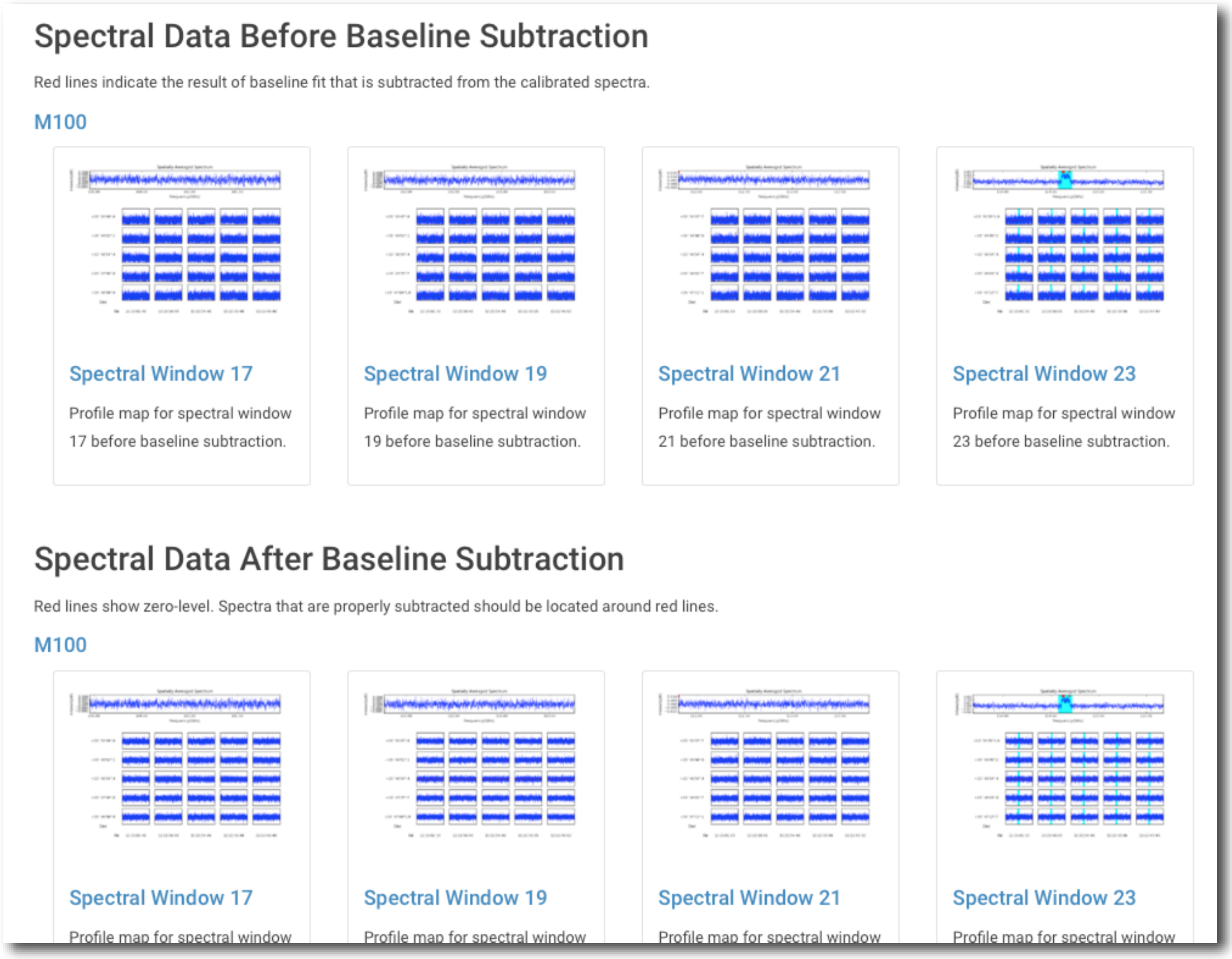
It shows a list of the calibrated MSs with the name of the applied Tsys, Sky and amplitude calibration (Kelvin to Jy conversion) tables, and also shows the integrated spectra after calibration.

9.8 hsd_baseline

Spectral data before/after baseline subtraction

The `hsd_baseline` page of the WebLog show the grid of spectra before (top) and after (bottom) baseline subtraction (Figure 21). The plots on the `hsd_baseline` summary page (just after clicking the `hsd_baseline` link of the WebLog) show a representative spectral map for each spw out of all maps in the detail pages (when you click the "Spectral Window" link below the grid of spectra in the summary page). The detail page has similar plots but this time for each EB, antenna, and polarization.

On the top panel of each grid of spectra, a spatially integrated spectrum per EB, antenna, spw and polarization is shown. The cyan filled regions indicate the mask channels containing emission line that are identified in the entire map. Below the top panel, there is a grid of spectra aligned along R.A./Decl. coordinates. Each small panel shows *one* representative spectrum per grid cell (which sometimes we call "sparse profile map"). The red (horizontal) line over-plotted on the spectrum indicates the fitted function to be used for baseline subtraction for spectral data before baseline subtraction, while the zero-level for spectral data after baseline subtraction.



Spectral Data Before Baseline Subtraction

Red lines indicate the result of baseline fit that is subtracted from the calibrated spectra.

M100

Spectral Window 17	Spectral Window 19	Spectral Window 21	Spectral Window 23
Profile map for spectral window 17 before baseline subtraction.	Profile map for spectral window 19 before baseline subtraction.	Profile map for spectral window 21 before baseline subtraction.	Profile map for spectral window 23 before baseline subtraction.

Spectral Data After Baseline Subtraction

Red lines show zero-level. Spectra that are properly subtracted should be located around red lines.

M100

Spectral Window 17	Spectral Window 19	Spectral Window 21	Spectral Window 23
Profile map for spectral window	Profile map for spectral window	Profile map for spectral window	Profile map for spectral window

Figure 21 An example of the summary page of hsd_baseline.

R.A. vs Dec. plots

There are four different plots per spw, i.e. “clustering_detection”, “clustering_validation”, “clustering_smoothing”, and “clustering_final”. The number of plots in each figure is the same as that of the candidate line components. The “cluster_detection” plot (Figure 22a) shows the grid cells having emission line exceeding the threshold. In the plot, yellow grid cells show a region where there is a single time-domain group with detected emission lines. Cyan squares indicate grid cells where there are more than one time-domain groups with detected emission lines.

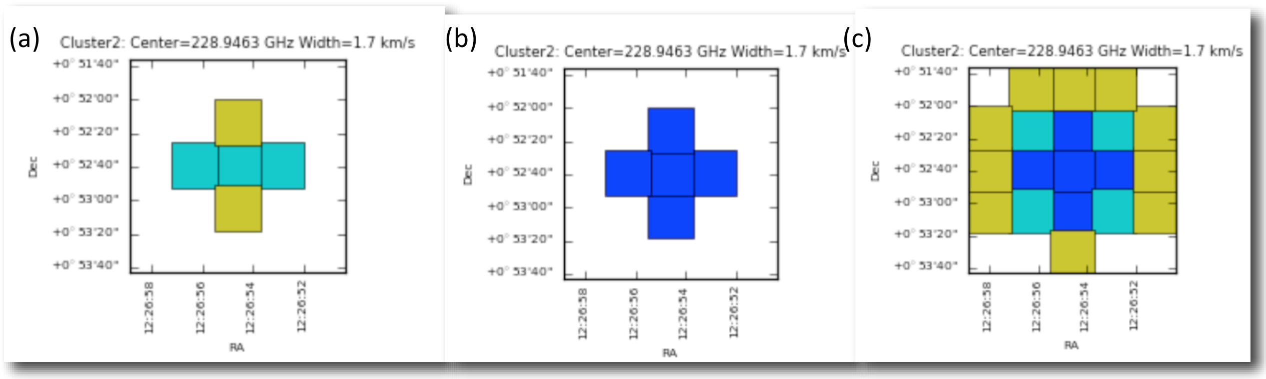


Figure 22 Examples of (a) clustering_detection, (b) clustering_validation, and (c)clustering_smoothing.

After line detection, the algorithm calculates how many spectra containing emission lines are included in the grid cell in order to judge whether the grid cell possibly contains true emission lines. At this line detection validation step, the ratio of the number of spectra having detected emission lines (defined as “Nmember”) per grid cell and the number of total spectra belonging to the grid cell (“Nspectra”) is calculated. The “clustering_validation” plot (Figure 22b) shows this ratio for each grid cell, i.e., the grid cell is marked as:

- “Validated” if Nmember/Nspectra > 0.5 (Blue squares in Figure 22b)
- “Marginally validated” if Nmember/Nspectra > 0.3 (Cyan squares)
- “Questionable” if Nmember/Nspectra > 0.2 (Yellow squares)

After the validation step, the grid containing the Nmember/Nspectra rate per grid cell is smoothed by a Gaussian-like grid function. This is to eliminate the isolated grid cells having a single emission line candidate while enhancing the grid cells with detected emission line in neighboring grid cells.

Figure 22c shows an example of “clustering_smoothing”. Blue squares represent the grid cells with points exceeding the defined threshold, i.e., the grid cells having promising detections of emission lines that are also found in the neighboring grid cells. Cyan and yellow squares are the grid cells with points slightly below the threshold (Border), or lower than the threshold (Questionable).

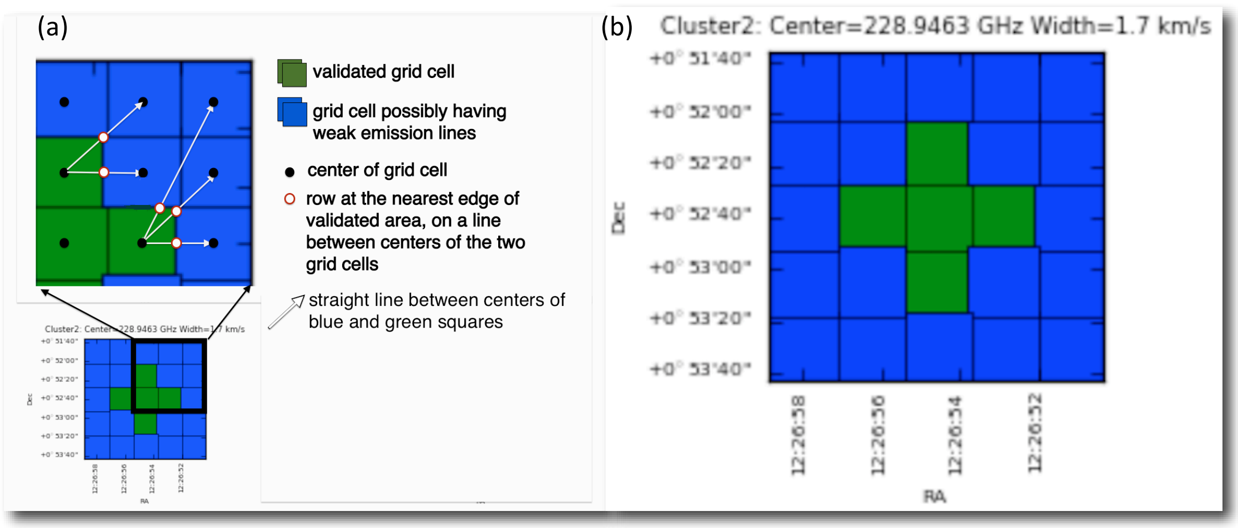

(a)
validated grid cell
grid cell possibly having weak emission lines
center of grid cell
row at the nearest edge of validated area, on a line between centers of the two grid cells
straight line between centers of blue and green squares
Cluster2: Center=228.9463 GHz Width=1.7 km/s
(b)
Cluster2: Center=228.9463 GHz Width=1.7 km/s
+0° 51'40"
+0° 52'00"
+0° 52'20"
+0° 52'40"
+0° 53'00"
+0° 53'20"
+0° 53'40"
Dec
12:26:58
12:26:56
12:26:54
12:26:52
RA

Figure 23 (a) An example of how the mask range is calculated. In the blue squares, the mask channel range is the range obtained at the nearest edge of any validated area by interpolation mask channel ranges in the validate grid cells (white-filled red circle). (b) An example of clustering_final.

As a final step, the mask region for each grid cell is determined. In the validated area after the validation and the smoothing steps (blue squares in Figure 22c or green squares in Figure 23), mask channel ranges are calculated over the spatial domain by inter/extrapolating the mask ranges of the integrated spectra in the validated cells, and over each single non-integrated spectrum. The mask channel range is determined and used in baseline subtraction in the green and blue squares of Figure 23a. An example of “clustering_final” is shown in Figure 23b.

Line Center vs. Line Width plot

This plot shows the extent of each identified emission line candidate on the parameter space of the line width versus the line center. The small dots indicate spectra containing identified emission line. The red ovals show each clustering region with a size of the cluster radius.

Number of Clusters vs. Score plot

This plot shows the number of clusters and corresponding scores based on the cluster size determined from the “line width” v.s. “line center” plot using clustering analysis (K-means algorithm). The scoring is empirically defined so that the score gets better (smaller) when the cluster size is smaller, the number of clusters is smaller, and the number of outliers is fewer than those of other clusters. The users will know which number of clusters is more plausible by searching for the number of clusters with a lower score. This plot is basically for developers.

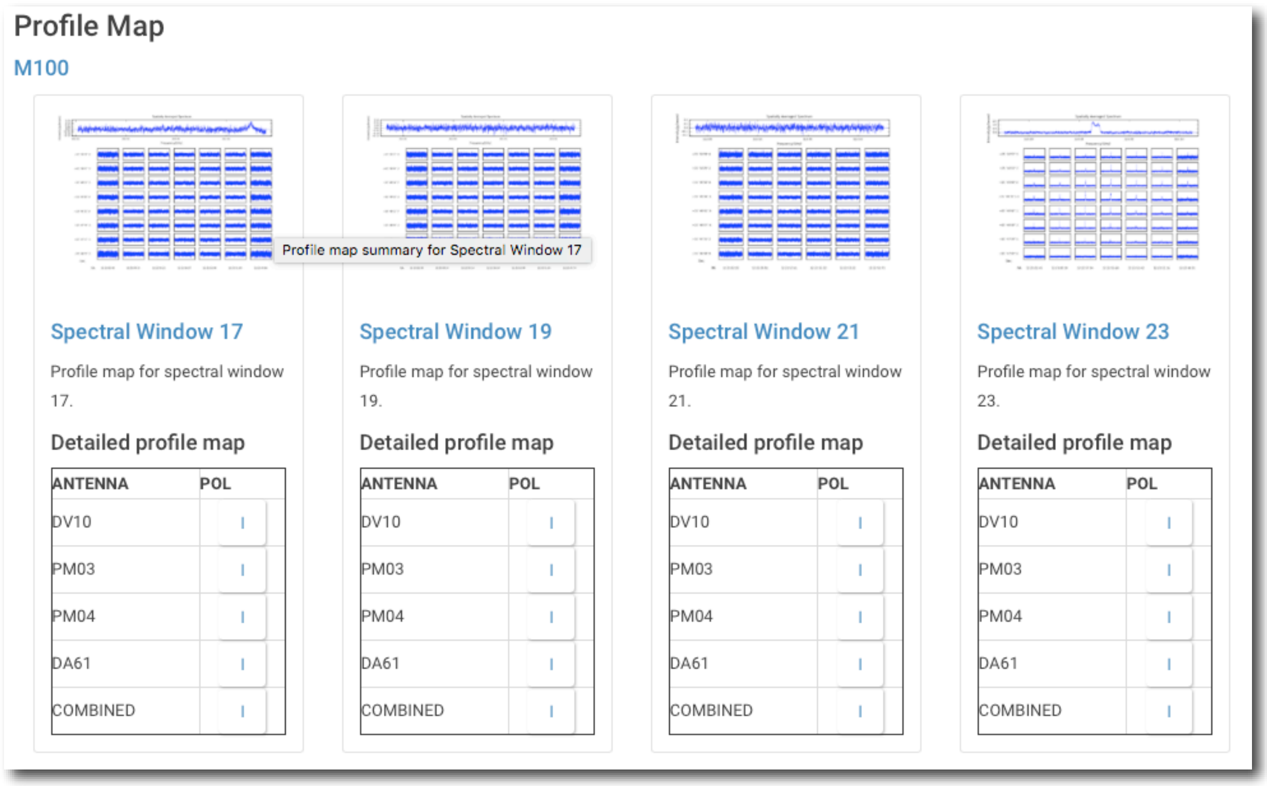
9.9 hsd_blflag

The WebLog shows the list of flagged data percentage using five criteria which are explained in the ALMA Pipeline Reference Manual. When you click on “details”, you will get the detailed figures to evaluate these criteria as a function of rows (one row corresponds to a spectrum for one integration). The flagged and unflagged data are shown in red and blue, respectively.

9.10 hsd_imaging

Profile Map

Figure 24 shows the top of the summary page. Three types of profile maps are available in the WebLog: 1) The simplified profile map of the combined image per spw at the top, 2) a simplified profile map per antenna, and 3) a detailed profile map. To access the simplified profile map per antenna, click the corresponding "Spectral Window". Each spectrum of the simplified profile maps (either 1. or 2.) corresponds to an averaged spectrum in an area of ⅛ of the image size (imsize), so that the total number of spectra in the profile map is 8 times 8. If the number of pixels (along x- or y-axis) is less than eight, it shows all spectrum per pixel. To see the detailed profile maps, click the icon with a symbol of polarization in the polarization column (see Figure 24). Each bin of the profile map is equivalent to a pixel, but with an interval of three cells. Due to the limitation of the allowed number of plots per page (max 5 x 5 plots per page), the rest of the plots are displayed in other pages.


Profile Map
M100
Profile map summary for Spectral Window 17
Spectral Window 17
Profile map for spectral window 17.
Detailed profile map
Spectral Window 19
Profile map for spectral window 19.
Detailed profile map
Spectral Window 21
Profile map for spectral window 21.
Detailed profile map
Spectral Window 23
Profile map for spectral window 23.
Detailed profile map

ANTENNA	POL
DV10	I
PM03	I
PM04	I
DA61	I
COMBINED	I

Figure 24 An example of the profile map.

Channel Map

The number of channel maps per spw corresponds to the number of emission lines that have been identified by the clustering analysis. In each channel map (see Figure 25), the top-middle plot shows the identified emission line and the determined line width (bracketed by two red vertical lines), overplotted on the averaged flux spectrum (in Jy) as a function of frequency (in GHz).

The top-left plot shows the zoom-up view of the identified emission line, but with velocity axis. The vertical axis is the averaged flux in Jy and the horizontal axis is in units of km/s. The (center) velocity of 0 km/s corresponds to the central frequency of the emission line, while the velocity range is equivalent to the masked region where the emission line was identified. The line velocity width is gridded into 15 bins, which are shown as red vertical lines. The top-right plot shows the total integrated intensity map (in Jy/beam km/s) over the all channels in the spw. Finally the channel maps within the velocity range of the identified emission line are shown in the panel at the bottom. Each channel plot corresponds to a bin in the top-left plot.

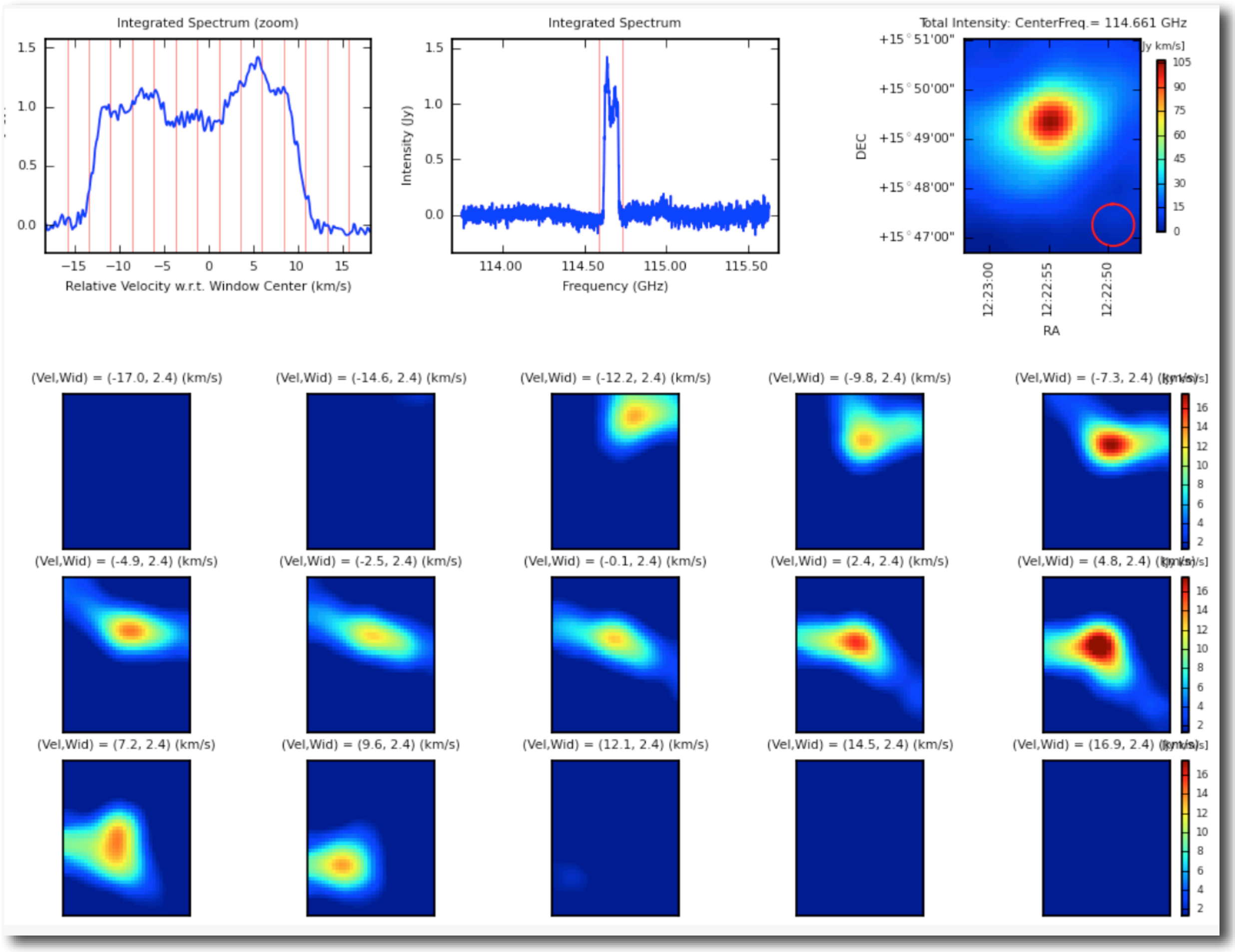


Figure 25 An example of channel map.

The **Baseline RMS Map** is created using the baseline RMS stored in the baseline tables. The baseline RMS is calculated by `hsd_baseline` using emission free channels.

The **Integrated Intensity Map** for each spw is generated using immoments task with all the available channel range.