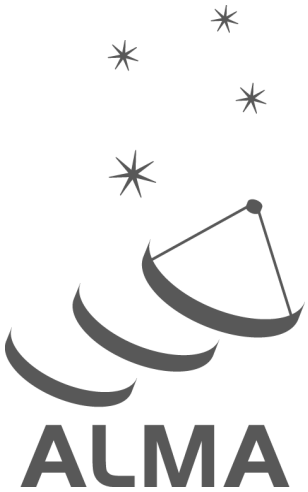
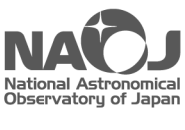
Doc 2023, ver. 1.0 | Oct 2023

User Support:

ALMA Science Pipeline User's Guide for Release 2023.1.0.124, CASA 6.5.4-9, python3.8

Interferometric and Single-Dish Processing





www.almascience.org

For further information or to comment on this document, please contact your regional Helpdesk through the ALMA User Portal at http://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
4.13v2.0 CASA 4.7.2	July 2017	Pipeline Team
5.13v1.0 CASA 5.1.1	November 2017	Pipeline Team
6.13v1.0 CASA 5.4.0	October 2018	Pipeline Team
7.13v1.0 CASA 5.6.1	October 2019	Pipeline Team
7.13v2.0 CASA 6.1.1	October 2020	Pipeline Team
2021.2v1.0 CASA 6.2.1	October 2021	Pipeline Team
2022.2v1.0 CASA 6.4.1	October 2022	Pipeline Team
2022.2v1.1 CASA 6.4.1	April 2023	Pipeline Team
2023.1v1.0 CASA 6.5.4	October 2023	Pipeline Team

In publications, please refer to this document as:
ALMA Pipeline Team, 2023, ALMA Science Pipeline User's Guide, ALMA Doc 2023v1.0

Contents

1 The ALMA Science Pipeline

1.1 Purpose of this document

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

This document is applicable for the Cycle 10 version (2023.1.0) of the ALMA Pipeline that is packaged with CASA 6.5.4. Version "casa-6.5.4-9-pipeline-2023.1.0.124" was deployed for use in ALMA Operations in October 2023. This version is labeled in the WebLog as Pipeline Version 2023.1.0.124 with CASA Version 6.5.4-9. The versions of the major included packages are: `Astropy`: 5.2.1, `NumPy`: 1.23.5, `Matplotlib`: 3.3.3, `SciPy`: 1.10.0 (see Table 1 in § 4.4 for a more complete list of version changes in third party packages between PL2022 and PL2023).

1.2 Pipeline Overview and Nomenclature

The ALMA Science Pipeline is used for the automated calibration and imaging of ALMA interferometric and single-dish data. ALMA Interferometric data refers to observations obtained with either the ALMA 12-m Array or 7-m Array, while single-dish data refers to observations obtained with the 12-m dishes of the ALMA Total Power Array.

The Pipeline consists of modular calibration and imaging tasks within the Common Astronomy Software Applications (CASA) data reduction package that are selected and put together in a specific order based on standard prescriptions or **recipes**. The ALMA pipeline recipes cover the processing requirements of what were formerly known as "standard" interferometric and single-dish observing modes. Datasets resulting from other observing modes are, as a rule, processed outside the pipeline, using manually modified CASA scripts (this typically amounts to less than a few percent of all ALMA data). In previous Cycles, the standard and non-standard observing modes were defined in the Proposer's Guide. For this Cycle, see section §3.3.2 of this document. The science pipeline is not yet commissioned for the combination of datasets obtained from different array components (i.e., separate IF array MOUS observations, or IF plus SD combinations).

The pipeline operates on a completed dataset that is comprised of all of the quality assured individual executions that result from completing a Scheduling Block (**SB**). An individual SB execution results in a dataset referred to as an **ASDM** (for ALMA Science Data Model), or **EB** (Execution block) and the collection of ASDMs (EBs) from a single SB are collected into a data structure called a Member Observing Unit Set (**MOUS**), which is the data unit that the pipeline operates on. The pipeline produces the following: calibration products for each ASDM (including calibration and flagging files and tables); imaging products (FITS images) made from all ASDMs (although not necessarily for all science targets – see §9.31); informative logs and scripts; and a **WebLog** consisting of a collection of webpages with diagnostic messages, tables, figures, and "Quality Assurance" (**QA**) scores. These products are reviewed as part of the ALMA Quality Assurance process, and, if satisfactory, are stored into the ALMA Science Archive. Calibrated MeasurementSets (**MS**), one for each EB, are also produced by the act of running the pipeline, but they are not stored in the archive. See the ALMA Technical Handbook for details on the ALMA data structures, quality assurance criteria, and archiving system.

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 look at the WebLog or consult the README file in the data delivery package. Some projects may contain a mix of both manually and Pipeline-calibrated data.

2 Quick Start

- If you want to understand what data is in a downloaded package, and the steps and quality of how it was processed see, see the **Pipeline WebLog** (§8).
- If you want to restore the calibrated MS, run **scriptForPI.py** (§5.1) or **casa_piperestorescript.py** (§5.3) to restore calibrated MSs.
- If you want to see, edit, or rerun the pipeline task commands that were run, you want **casa_pipescript.py** (§5.4).
- If you want to see the CASA task calls that were used, either look at the casa log linked at the bottom of each pipeline processing stage of the "By Task" section of the WebLog, or see the full **casa_commands.log** file (§5.5).
- If you want to cite the ALMA pipeline in a publication, we encourage you to cite the refereed journal article published in July 2023: PASP, volume 135, id 074501. This document contains additional details on the heuristics that are not included in this user's guide.

3 What's New in 2023.1

3.1 Major new capabilities

- For full-polarization interferometric data, the hifa_polcal stage now solves for the instrumental polarization and provides plots and QA in the WebLog, as described in §9.22.
- For single-field, non-ephemeris interferometric data, the hif_selfcal stage now attempts to perform self-calibration for each source, using the aggregate continuum image, as described in §9.47. If any source is sufficiently bright that self-calibration gains improve the signal-to-noise without changing the beam by more than 5%, then the gains are applied to continuum and uv-continuum-subtracted visibilities, which can be imaged by requesting datatype "SELFCAL_CONTLINE" or "SELFCAL_LINE" in hif_makeimlist (selfcal). Only sources with successful selfcal will be imaged with that option.
- To reduce instances of divergence in cleaning, hif_makeimages (cube) now detects spectral windows with high dirty dynamic range and sparse uv coverage and modifies the cyclefactor and cleaning threshold.
- In the continuum-finding task (hif_findcont), the application of a boxcar pre-smoothing of the mean spectrum of the joint mask is now based on the user-specified expected linewidth of the representative target rather than the bandwidth for sensitivity. This improvement will eliminate the cases of excessive smoothing when the bandwidth for sensitivity was based on continuum detections rather than spectral line detections. Also, in TDM spectra, spectral smoothing is disallowed if "cont" appears in the Transition name string associated with the spw by the user (and no "ID=x" with x>0), which increases the continuum bandwidth in some cases, often leading to "AllCont". Finally, moment 0 and 8 images are now generated in the same call to the CASA task immoments, which saves some processing time but results in a different file-naming convention compared to prior releases.
- The low-level function correctedampflag, which is used by hifa_bandpassflag, hifa_gfluxscaleflag, hifa_polcalflag and hifa_targetflag has been modified to improve the detection of general decorrelation by requiring it to be seen on a significant number of antennas. This avoids desensitizing the heurisitic to bad data that involve some (or all) baselines to only a few antennas and more than 10% of the timestamps. This will lead to flagging of additional bad data that was missed in prior versions.
- The DB (database) access in hsd_k2jycal stage is re-enabled with the default value of "dbservice = True". The relevant csv file (jyperk_query.csv) is now provided. As of CASA 6.4.3 we replace Jy/K DB access in hsd_k2jycal stage by using "gencal" task ("caltype=jyperk"). Corresponding pipeline codes that provide access to DB are not provided in the current release.
- To estimate the baseline flatness as much as possible, new step-wise logics are implemented to hsd_baseline stage. Namely, (i) Try computing binned spectrum with default `nbin` of 10 or 20. (ii) If it failed, re-compute binned spectrum with 2× previous `nbin`. (iii) If flatness was successfully obtained, create plots with binned spectrum. Relevant warnings will appear. (iv) If flatness was failed to compute, create plots without binned spectrum. Relevant warnings will appear. (v) If input spectrum is invalid, neither binned spectrum is computed nor any plot is created. Relevant warnings will appear.
- Flagging threshold was increased to avoid target mis-flagging in hsd_blflag stage. All the thresholds related to the hsd_blflag functions are doubled. This has certain merits. For example, when some line channels are unidentified, a higher threshold could prevent the emission in those channels being flagged out.
- There were roughly 0.15% of the FITS files in 2021.1 projects which have their frequency order *reversed* compared to all the other files, i.e., CDELT3 < 0. Single dish pipeline is now improved to re-invert such data, i.e., now producing all cubes to have CDELT3 > 0.

3.2 New features and improvements by WebLog section and processing stage.

3.2.1 General features

- **Main overview (per-EB) pages:**

  - The total number of EBs is shown in the Observation Overview heading.
  - The Correlator name has been added below the Science Bands in the Spectral Setup heading.
  - A colorized plot of spw ID vs. frequency has been added.
- **Spectral setup (per-EB) pages:**
  - Low frequency Band 3 SQLD spws are no longer labeled as Band 2.
  - Band 1 spws above 45 GHz are now labeled Band 1 instead of Unknown.
  - New columns added for the number of correlation bits and the receiver feed angles

3.2.2 New features and fixes specific to the interferometric pipeline

- **hifa_importdata:** (§ 9.1) The spectralDynamicRangeBandwidth, which is 1/3 of the expected line width specified by the user, has been added to the Representative Target information table.
- **hifa_flagdata:** (§ 9.2) The "Low Transmission" heuristic will no longer crash on Cycle 0-2 data due to the use of the J2000 coordinate frame. The new Tsys subscan intent #TEST that was introduced in Cycle 10 will be flagged like the rest of the Tsys subscans.
- **hifa_bandpass:** (§ 9.13) The fillgaps parameter is now exposed and settable in the PPR.
- **hif_lowgainflag:** (§ 9.10) The threshold for the `too_many_entirely_flagged` condition has been reduced to 0.666.
- **hifa_spwphaseup:** (§ 9.14) Flagging is now accounted for in the SNR estimates, which avoids selecting a heavily flagged spw for spw-mapping.
- **hifa_wvrgcalflag:** (§ 9.9) The QA score and notification will be blue instead of yellow if the improvement is < 1.0 but phase stability is good (generally meaning that the "phase correction with/without WVR" plot for the bandpass scan is between +/-50deg, showing stable phase even without WVR applied).
- **hifa_gfluxscale:** (§ 9.19) This task now prevents rare cases of two solutions for the same timestamp. The values in the flagging summary table are now correct for TARGET intent. The WebLog plot of spw flux density vs. frequency is now more color-blind friendly.
- **hifa_timegaincal:** (§ 9.20)
  - The phase offsets QA now works correctly for single-polarization data and no longer triggers red if an spw has all antennas flagged.
  - The duplicate set of offset points and erroneous yellow QA score have been eliminated when the phase calibrator is also the bandpass or flux calibrator.
- **hif_applycal** (§ 9.23): The plotms command shown in the WebLog for corrected/model amplitude now has the correct syntax.
- **hif_makeimlist (checksource)**: (§ 9.29) Multi-EB MOUS with a check source no longer crash.
- **hifa_imageprecheck** (§ 9.30): No longer crashes on a mosaic with an integer source name.
- **hifa_renorm:** (§ 9.32)
  - The parameter `bwthreshspw` is now exposed and settable in the PPR.
  - Segment sizes dynamically change to automatically avoid science lines on the segment boundaries (a manually input `bwthreshspw` value will override it).
  - If segment sizes change differently from scan-to-scan or field-to-field (common for mosaics), the summary plot will show a warning to be aware that it happened.
- **AQUA report:**
  - The image name is included in the Sensitivity entries.
  - The specifier "Regcal" or "selfcal" is now in filenames, and more importantly, datatype used for each image is in the manifest.xml entries.

  - The spw field for the hifa_timegaincal phaseoffset QA is now listed correctly.
  - The version of the report after imaging is now linked in the WebLog (it used to be the cal-only version).
- **CASA log messages:** The order of copytree commands in **casa_commands.log** will now be consistent across all ARCs.

3.2.3 New features specific to the single-dish pipeline

- **hsd_importdata** (§ 10.1): Now the pipeline properly handles Execution Blocks (EBs) that do not contain representative source.
- **hsd_importdata** (§ 10.1): Tied to the re-enabled DB access, the current pipeline does not automatically create `flux.csv`. Any information about `flux.csv` is now omitted from weblog.
- **hsd_atmcor** (§ 10.8): Inappropriate messages in the "Applied corrections" table, "Default" is now corrected to "Fallback".
- **hsd_baseline** (§ 10.9): The treatment of the flagged data, when one spectrum in a raster row is completed flagged, is improved. This is relevant to the quality of the cubic spline fitting to the data in the hsd_baseline stage, and the resultant baseline fit quality has improved dramatically.
- **hsd_blflag** (§ 10.10): Changed flagging threshold of running mean.
- **hsd_imaging** (§ 10.11): In the previous pipelines, sensitivities were calculated in the native (effective) channel resolution first, and then scaled to estimate the sensitivity in a representative channel width specified by PIs for the images of representative SPW regardless of the source, in hsd_imaging stage of a weblog. However, this process was changed, and the sensitivity in the native (effective) channel resolution is now displayed in the hsd_imaging stage of a weblog.
- **hsd_imaging** (§ 10.11): The calculation procedure of the weight for spectra that are completely flagged in only one polarization is now improved, which reduces the artifact spotty images generated in the hsd_imaging stage.
- **hsd_imaging** (§ 10.11): In addition to moment-0 maps shown in the channel map plots, now moment-8 (maximum value of the spectrum) maps are also shown in weblog.

3.3 Current Known Limitations of the 2023 Pipeline

3.3.1 General limitations

- All raw data (ASDMs) run through the pipeline must have complete and properly formatted binary and metadata. This is not always the case for ASDMs from earlier ALMA cycles. In particular:
  - The SD pipeline can only be run on data from Cycle 3 or later.
  - The IF pipeline will not work with ALMA Cycle 0 data, nor with some Cycle 1 – 2 data (those that do not have complete and accurate calibration intent labelling).

  Manually calibrated data from Cycles 1 – 3 are likely to have problems if run through the pipeline.
- The representative source name must be observed at least once with TARGET intent in the representative spectral window name. A corollary statement is that the representative spectral window cannot be used to observe only calibration intents.
- The raw data (ASDMs) run through the pipeline should have a "quality assurance level 0" (QA0) assessment of "QA0 Pass". Running the pipeline on non-quality assured data ("QA0 SemiPass" or "QA0 Fail") is not expected to give sound results and may fail.
- In general, CASA assumes that it has access to all of the available RAM on the node where it is run. If other processes use significant amounts of this RAM, the pipeline may fail. For example, if running with a resource allocator such as Torque, part of the CASA tclean task (major cycle gridding) will respect CGROUP memory limits, but other parts of CASA will not. Please contact the pipeline working group (PLWG) via the ALMA Helpdesk for advice on pipeline usage in complex computing environments.

IF Observing modes supported by ALMA 2023 Pipeline

*Single field and/or pointed mosaic with dual-polarization and:	1 SpectralSpec for Science/GainCalibration (implies a single receiver band)	>1 SpectralSpec for Science/GainCalibration (includes any number of receiver bands)
1 Phase calibrator 1 Bandpass calibrator 1 Flux calibrator 1 (optional) Check source (BP/Flux can be the same)	**A.** Single source in continuum and lines **B.** Multiple sources in continuum and/or lines that are within a few degrees, and with similar LSR velocity (e.g. within a single Milky Way GMC)	**A.** Spectral scan of a single source **B.** Line observations of >1 source, all within a few degrees with diverse LSR velocities requiring different tunings **C.** Spectral scans of >1 source, all within a few degrees
>1 Phase calibrator	Continuum and/or lines in multiple widely-separated sources (> a few degrees) with similar LSR velocity	**Not supported:** Line observations (or spectral scans) of widely-separated sources (> a few degrees).
>1 Flux calibrator, or >1 Bandpass calibrator	Not supported (nor is it needed by current execution block lengths, < 90 minutes)	Not supported (nor is it needed by current execution block lengths, < 90 minutes)
No Bandpass calibrator, or No Flux calibrator, or No Science target	Not supported (missing calibrators would require "sessions" and/or an extensive database of antenna gains vs. frequency and time of day)	Not supported (missing calibrators would require "sessions" and/or an extensive database of antenna gains vs. frequency and time of day)
Full polarization (i.e., including XY and YX correlator products)	**Supported** (but full polarization imaging of the science target must be done manually) Single polarization (XX or YY) is supported, but the flux scale will be affected if flux calibrator is polarized.	Unknown: no test data provided
Different SpectralSpec used for Gain Calibration (not supported)	Not supported A. Bandwidth switching (low aggregate science BW) B. Band-2-band transfer (no nearby strong phase calibrator)	Not supported A. BW-switching on multiple objects of diverse velocity B. B2B transfer on multiple objects of diverse velocity

***Notes:** Ephemeris objects are fully supported (cube imaging must be done in serial because of still-unfixed CASA bugs); VLBI, Solar are **not** supported

4

Figure 1: Summary of supported observing modes in PL2023.

3.3.2 Additional limitations of the Interferometric Pipeline

The IF pipeline is commissioned only for the observing modes shown in Figure 1, subject to these additional restrictions:

- While the IF pipeline calibration and flagging tasks do include low signal-to-noise heuristics, they will produce poor results if the calibrators are too weak. If the sources are so weak that solutions cannot be found, including on check sources, then the pipeline will crash first in hifa_gfluxscale.
- In order to increase delivery rates of data to PIs, the archived imaging products may be binned in frequency, limited in the imaged field of view, and/or restricted to a subset of sources (see §9.31). Users can make the missing products by making small modifications to the scripts that are archived with the data.
- 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. If the data are heavily binned in frequency before this task is run, the results may be compromised. The user can edit the file **cont.dat** (Sec.7.6) and rerun sections of the imaging pipeline to obtain their own continuum subtracted visibilities and new line images.
- The IF PL imaging steps use the "effective channel bandwidth" from the raw data file to calculate the theoretical image sensitivity and hence clean thresholds. This information is not correctly stored in raw ALMA data from Cycles 2 and earlier; as a result, the clean thresholds will be higher than intended when such data is run through the imaging pipeline.
- The standard recipes used in Operations in 2023 include self-calibration for single-field non-ephemeris targets, but there are still cases for which pipeline imaging products of bright sources can be dynamic range limited: bright mosaics and ephemeris sources, bright sources for which the self-calibration doesn't converge and thus is not applied, or if the selfcal stage is removed from the recipe by a data reducer.

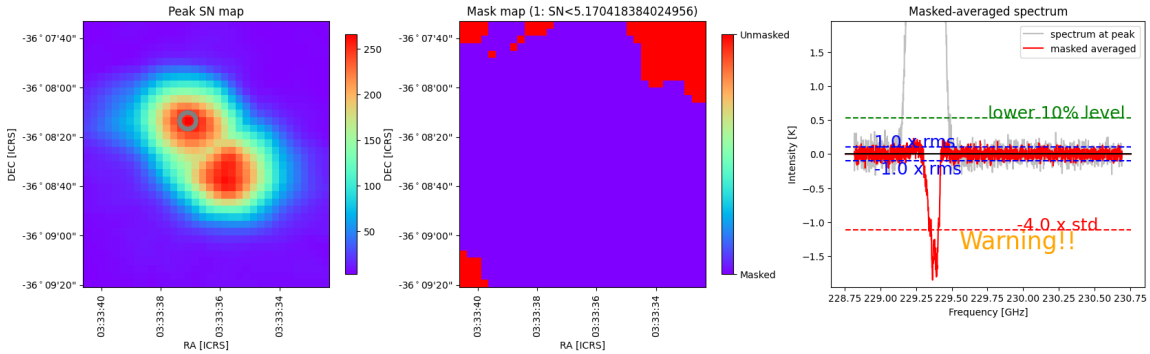


Figure 2: An example of the contamination plot for the case the OFF position is contaminated by the astronomical signal. The spectrum with the negative peak indicates that the OFF position is contaminated by the astronomical signal.

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

3.3.3 Additional limitations of the Single-dish Pipeline

- 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), especially in TDM mode.
  - Cubes with a "forest" of emission lines.
- The SD pipeline imaging results may be unusable if there is emission in the "off" position and/or if the atmospheric line features still remain in the calibrated data. However, as described below diagnostic plots aide in clearly identifying this situation. Figure 2 shows an example of the contamination plot for the case the OFF position is contaminated by the astronomical signal.
- If hsd_baseline is run manually to subtract the baseline in an individual spw, hsd_blflag and hsd_imaging should be run before proceeding to subtract the baseline from the next spw. Otherwise, hsd_baseline will overwrite the baseline solutions for the previous spws.
- Strong emission is flagged in hsd_blflag. In some cases, emission components at some channels (i.e. wing component) are not identified as the line. This increases the RMS at corresponding channels. Essentially, improving the line identification algorithm is required to fix this issue. In operation, this issue can be avoided by changing the threshold of blflag manually.

A list of pipeline "known issues" that arise after the publication date of this document is maintained on the ALMA Science Portal at http://almascience.org/processing/. This list will be updated as issues are discovered during the cycle.

4 Pipeline Versions & Documentation

4.1 Obtaining the Pipeline

A link to the CASA+pipeline package is available, along with installation instructions and supporting documentation, from the **Overview and Pipeline** section of the **ALMA Science Portal** at `http://www.almascience.org` (under the "Processing" tab, or directly at `http://almascience.org/processing/`). If any issues are encountered with CASA installation, please contact the ALMA Helpdesk via the link on the ALMA Science Portal.

The pipeline tasks become available by starting up CASA using the command:

% `casa --pipeline`

Or to run CASA with pipeline tasks using MPI (multi-core parallelization):

% `mpicasa -n 8 casa --pipeline`

Note that you may need to provide the full path to `casa` in the `mpicasa` command line to initiate the desired version if you have multiple versions installed.

4.2 Pipeline-related Documentation

The User documentation currently relating to the Pipeline is also available from the Overview and Pipeline section of the Science Portal referenced above. This includes the **ALMA Science Pipeline User's Guide** (this document), and the ALMA Pipeline Reference Manual (a detailed description of individual Pipeline tasks parameters).

Examples of common re-imaging modifications to the IF pipeline script are given at: `https://casaguides.nrao.edu/index.php/ALMA_Imaging_Pipeline_Reprocessing`

In addition, Chapters 10, 11, and 13 of the ALMA Technical Handbook provide more information on calibration in general, how Quality Assurance is performed, and how data is archived.

4.3 Pipeline and CASA Versions

The pipeline heuristic 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 12 for an example).

CASA produces numerous releases, but only the versions listed in the table on the Science Portal `http://almascience.org/processing/` have been scientifically validated, are accepted for operations, and are supported by ALMA. That table additionally lists the versions which can be used to restore previously pipeline-calibrated archival visibility data.

Beware that each CASA release often includes one or more updates the third party python modules that are packaged with it. A summary of the changes in the CASA versions used between PL2022 and PL2023 are shown in Table 1.

4.4 Pipeline and CASA tasks

The pipeline heuristics are written as special CASA tasks, appearing with a `hif_` or `hifa_` (for interferometric) or `hsd_` (for single-dish) prefix. They can be viewed and executed within CASA just as any python function (if one has launched CASA with "--pipeline"). For example, one can view the possible inputs for the task hifa_importdata by typing `?hifa_importdata`.

The pipeline heuristics use CASA tasks wherever possible to perform the data reduction or imaging. E.g. the pipeline bandpass calibration & flagging task hifa_bandpassflag calls the CASA bandpass task, and the

module	6.4.1py3.6 (PL2022)	6.5.4py3.8 (PL2023)
almatasks[a]	1.3.1	1.6.1
astropy	4.1	5.2.1
attrs	19.3.0	22.2.0
backcall	0.1.0	0.2.0
bdsf	1.10.1	1.10.2
certifi	2020.12.05	2022.12.07
cycler	0.10.0	0.11.0
decorator	4.4.2	5.1.1
grpcio	1.26.0	1.29.0
ipython	7.15.0	7.15.0 (no change)
jedi	0.17.0	0.18.2
kiwisolver	1.3.1	1.4.4
matplotlib	3.3.3	3.3.3 (no change)
mpi4py	3.0.3	3.1.3
numpy	1.18.4	1.23.5
packaging	20.4	23.0
parso	0.7.0	0.8.3
pip	20.1.1	23.0.1
pluggy	0.13.1	1.0.0
prompt_toolkit	3.0.5	3.20.1
ptyprocess	0.7.0	0.8.0
Pygments	2.6.1	2.14.0
pyparsing	2.4.7	3.0.9
pytest	5.4.2	7.2.1
pytz	2020.1	2022.7.1
scipy	1.4.1	1.10.0
setuptools	39.0.1	56.0.0
traitlets	4.3.3	5.8.1
wcwidth	0.2.2	0.2.6
wheel	0.37.1	0.41.2

[a]contains wvrgcal.py, but there were no changes to it

Table 1: Changes to third party python modules in CASA. The ones likely to be of most interest to users are highlighted in bold.

interferometric imaging task hif_makeimages calls the CASA imaging task tclean.

The standard pipeline processing recipes are deterministic and should always give the same result for the same data. However, the CASA pipeline tasks are designed to be highly flexible, so that they can have the default inputs over-ridden with user-specified values, or be added, subtracted, or rearranged to produce alternative processing recipes. This enables a manual "mix and match" mode for data reduction and imaging that combines standard CASA pipeline tasks with other CASA commands or python code to produce scripts that are better tuned to the idiosyncrasies of a specific dataset. The exact pipeline commands that will reproduce the standard recipe are delivered with each dataset, in a script called **member.<mous_uid>.<recipe>.casa_pipescript.py** (see §5.4 below). One can edit and add to that script to implement "mixed mode" processing.

Some common "manual mode" modifications are presented in §6 below. A complete list of the variables for each pipeline task is given in the ALMA Pipeline Reference Manual.

This document, along with the ALMA Pipeline Reference Manual, describe key aspects of the pipeline tasks. Important changes to other CASA tasks are documented in the Release Notes for the corresponding CASA release, available from the CASA page `https://casa.nrao.edu`.

5 Data Processing Files

5.1 Archived scripts

There are several scripts that are archived with ALMA data deliveries. These are described in the document **ALMA QA2 Data Products** (sometimes cycle-specific) available from ALMA Science Portal under the "Processing" tab. The particular scripts for a specific dataset should also be described in the QA2 report archived with the data products. This report will vary based on how the data were processed (pipeline calibrated & imaged; pipeline calibrated & manually imaged; manually calibrated & pipeline imaged, manually calibrated & manually imaged).

The scripts produced by the pipeline are archived with the data and have file names like:
member.<mous_uid>.<recipe>.casa_pipescript.py and
member.<mous_uid>.<recipe>.casa_piperestorescript.py.

The former includes all pipeline processing commands that were run on the data, and is more fully described below. The latter "restores" the data, which means that rather than re-running the pipeline calibration commands, it uses previously derived calibration and flagging tables and applies them directly to the raw data, producing a calibrated MeasurementSet. This is much quicker and requires less computing resources than re-running the pipeline calibration commands. However, expert users should be aware that if the latter, faster method is used, then the state of the MeasurementSets are not exactly the same as in a complete run (e.g. the model of the calibrators will not be set).

Every delivery package also includes a master script with a file name like **member.<mous_uid>.scriptForPI.py**, that will reproduce the calibrated data regardless of how it was processed (manual or pipeline). This script is not created by the pipeline, but instead by the data packaging software so that it is produced for both pipeline and manually reduced data. For pipeline calibrated data, it will simply invoke the pipeline-produced **casa_piperestorescript.py** or **casa_pipescript.py** scripts mentioned above.

Using scriptForPI.py is the recommended and easiest method of obtaining calibrated ALMA data from the delivery. However, one can also run the pipeline **casa_piperestorescript.py** using the steps in §5.3. To *change* the calibration results, one would re-run the commands in **casa_pipescript.py** after making modifications, as described in §6.

5.2 Pipeline "Helper" text files

Both the IF and SD pipeline use a number of text files that, if present, will affect the pipeline results (e.g. by applying manually identified flags or by updating calibrator fluxes or antenna positions before calculating the calibration tables). These files are particularly useful for users to over-ride the default pipeline behavior when re-running the pipeline at home, as more fully described in §6 below. They include the following:

- **flux.csv**: This file is used by the IF pipeline to update the flux of calibrators. The flux of the calibrator with the "AMPLITUDE" intent will affect the overall flux scale of the data. If this file is not present where the pipeline is run and hifa_importdata parameter `dbservice` is True, the pipeline will attempt to contact the ALMA source catalog (at the URL specified by the environment variable `FLUX_SERVICE_URL`) for previously recorded flux densities, and if that doesn't succeed, the fluxes in the ASDM(s) will be used, representing the best flux estimate at the time the SB was executed. If no flux value appears in either the flux.csv file or the ASDM, a flux of 1.0 Jy is adopted.
- **jyperk.csv** and **jyperk_query.csv**: These files are used by the SD pipeline to update the "Kelvin to Jansky" calibration factors which set the overall flux scale of the data. The SD pipeline will use a file specified by hsd_k2jycal parameter `reffile` (**jyperk.csv** as default). If they are not present where the pipeline is run and the pipeline database query parameter is True, conversion factors are obtained via the database. If they are not present and the pipeline database query parameter is False, conversion factors of unity are assumed.
- **antennapos.csv**: This file is used by the IF pipeline to update the positions of the antenna elements. If

it is not present where the pipeline is run, the positions in the ASDM(s) will be used.

- **uid*flagtemplate.txt**: This file is used to add additional CASA flagging commands that will be applied to the data before the calibration tables are calculated.
- **uid*flagtsystemplate.txt**: This file is used to add additional CASA flagging commands that will be applied to the tsys spws before the calibration tables are calculated.
- **uid*flagtargetstemplate.txt**: This file is used to add additional CASA flagging commands that will be applied to the data after the calibration tables are calculated, but before science target imaging is performed.
- **cont.dat**: This file is used to specify the continuum frequency ranges used for constructing the continuum images and creating the continuum-subtracted cubes. This particular file is described in more detail below (§7.6) and in the reimaging casaguide https://casaguides.nrao.edu/index.php/ALMA_Imaging_Pipeline_Reprocessing.

The format of each of these files is given in §7.

5.3 The Pipeline script to restore calibrated MSs: casa_piperestorescript.py

To restore data calibrated by the pipeline, one can either run **scriptForPI.py** as described in **ALMA QA2 Data Products** document available from ALMA Science Portal under the "Processing" tab (or directly at https://almascience.org/processing), or one can run the pipeline-provided **casa_piperestorescript.py** script:

- Create **rawdata**/, **working**/, and **products**/ subdirectories.
- Download the raw ASDMs from the archive and put them in **rawdata**/. 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 rename to not have this suffix).
- Copy or move ***manifest.xml, *caltables.tgz, *flagversions.tgz,** and ***calapply.txt** to **products**/.
- Copy **uid*casa_piperetorescript.py** to **working**/**casa_piperestorescript.py**.
- In **working**/, start `casa -pipeline`, and `execfile(''casa_piperestorescript.py'')`.

5.3.1 Results from running the SD casa_piperestorescript.py

- A calibrated MS for each ASDM with a name like **uid___A002_Xe50c9e_X1297.ms**.

Running the script through hsd_atmcor command will additionally create:

- A calibrated, atmospheric-corrected MS for each ASDM with a name like

  uid___A002_Xe50c9e_X1297.ms.atmcor.atmtype1.

  The pipeline "automatic" mode reproduces correction for atmospheric effects.

Note that the baseline subtraction is not done for the restored calibrated MS. Running the script through hsd_atmcor and hsd_baseline commands will additionally create:

- A calibrated, atmospheric-corrected, baseline-subtracted MS for each ASDM with a name like

  uid___A002_Xe50c9e_X1297.ms.atmcor.atmtype1_bl.

  The pipeline "automatic" mode reproduces the baseline subtraction. If instead the user may want to set the mask ranges to be used for baseline subtraction, CASA task sdbaseline is recommended. In this case, please be aware that a WebLog is not generated for CASA tasks. If the baseline subtraction is done with the CASA task sdbaseline, any further Pipeline tasks cannot be used.

Running the script additionally through hsd_blflag command will result in:

- flagging based on the baseline rms for each ASDM. The hsd_blflag command has to be run after hsd_baseline at least once. In the standard recipe, hsd_baseline and hsd_blflag are repeated twice to improve the quality of baseline detection.

Running the script through the hsd_imaging command will additionally create:

- native resolution images per spectral window, antenna, and source.

5.3.2 Results from running the IF casapiperestorescript.py

- A calibrated MS for each ASDM with a name like **uid___A002_Xe50c9e_X1297.ms**, containing all sources including calibrators, with calibrated data in the CORRECTED column.

It is often desirable to subsequently run the first few steps of the imaging pipeline, to recover uv-subtracted target visibilities. Detailed instructions are found here `https://casaguides.nrao.edu/index.php/ALMA_Imaging_Pipeline_Reprocessing`. In brief:

1. navigate to the **calibrated/working** directory
2. copy **cont.dat** (§7.6) into that directory - it is likely to be found inside **calibration/*auxproducts.tgz**
3. if you still have casa running from having just run **scriptForPI.py** or **casa_piperestorescript.py**, then you have an active Pipeline session, and new pipeline task calls will use the active **Context** – for example, the MSs are already known in that **Context**.
4. if not, you will have to start `casa -pipeline`, and run `h_init` and then hifa_importdata with the list of recently-restored, calibrated MSs, to start a new pipeline session.
5. run `hifa_mstransform()` to create ***_targets.ms**, with calibrated continuum+line target data in the DATA column.
6. next, run `hif_makeimlist(specmode="mfs"); hif_findcont()`. It should use your existing **cont.dat** and not have to recalculate anything.
7. finally, run `hifa_uvcontsub()`. Now your ***.targets_line.ms** will have continuum-subtracted line visibilities in the DATA column.

5.4 The Pipeline processing script: casa_pipescript.py

5.4.1 Format of casa_pipescript.py

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. A typical **casa_pipescript.py** script for a SD Pipeline run (including both calibration + imaging steps) is shown in Figure 3, while a typical IF pipeline script including both pipeline calibration and imaging steps is shown in Figure 4.

For data that were both calibrated and imaged in the pipeline (including all SD data run through the pipeline), the **casa_pipescript.py** file will include both the calibration and imaging pipeline commands. For IF data that were calibrated in the pipeline but imaged outside of the pipeline, the **casa_pipescript.py** file will only include the IF calibration pipeline commands (up to the line "# Start of pipeline imaging commands"), and the archived data will include a separate **scriptForImaging.py** script containing the manual (CASA) imaging commands. If instead the IF data were manually calibrated and pipeline imaged, there will be a separate **scriptForCalibration.py** script (one for each EB) in the archived data, containing the manual (CASA) calibration commands, and the IF pipeline imaging commands (those following the line "# Start of pipeline imaging commands" in Figure 4) would be included in a separate **scriptForImaging.py** script.

The tasks names, order, and parameter values in the **casa_pipescript.py** script reflect the processing recipe used for each individual delivery. Most tasks are run with default parameters; to see what task parameters are

```
__rethrow_casa_exceptions=True
h_init()
hsd_importdata(vis = ['uid___A002_X877e41_X452'])
hsd_flagdata() ## Uses *flagtemplate.txt
h_tsyscal()
hsd_tsysflag()
hsd_skycal()
hsd_k2jycal() ## Uses jyperk.csv
hsd_applycal()
hsd_atmcor()
hsd_baseline()
hsd_blflag()
hsd_baseline()
hsd_blflag()
hsd_imaging()
h_save()
```

Figure 3: Example of the Single Dish Pipeline calibration + imaging script **casa_pipescript.py**. The "##" comment line identifies the pipeline command that uses one of the pipeline "helper" text files described in §7.

available, type `?<task_name>` at the CASA command line or consult the ALMA Pipeline Reference Manual for more details, and §6 below for examples of modified pipeline re-runs.

```
__rethrow_casa_exceptions = True
context=h_init()
try:
   hifa_importdata(dbservice=False, vis=['uid___A002_X877e41_X452'], session=['session_1']) ## Uses flux.csv
   hifa_flagdata() ## Uses *flagtemplate.txt
   hifa_fluxcalflag()
   hif_rawflagchans()
   hif_refant()
   h_tsyscal()
   hifa_tsysflag()
   hifa_antpos() ## Uses antennapos.csv
   hifa_wvrgcalflag()
   hif_lowgainflag()
   hif_setmodels()
   hifa_bandpassflag()
   hifa_bandpass()
   hifa_spwphaseup()
   hifa_gfluxscaleflag()
   hifa_gfluxscale()
   hifa_timegaincal()
   hifa_targetflag()
   hif_applycal()
   hif_makeimlist(intent='PHASE,BANDPASS,AMPLITUDE')
   hif_makeimages()
   hif_makeimlist(per_eb=True, intent='CHECK')
   hif_makeimages()
   hifa_imageprecheck()
   hif_checkproductsize(maxproductsize=350.0, maxcubesize=40.0, maxcubelimit=60.0)
   hifa_renorm()
   hifa_exportdata()
# Start of pipeline imaging commands
   hif_mstransform()
   hifa_flagtargets() ## Uses *flagtargetstemplate.txt
   hif_makeimlist(specmode='mfs')      ## Uses cont.dat
   hif_findcont() ## Modifies cont.dat
   hif_uvcontsub()
   hif_makeimages() ## Uses cont.dat
   hif_makeimlist(specmode='cont') ## Uses cont.dat
   hif_makeimages() ## Uses cont.dat
   hif_makeimlist(specmode='cube') ## Uses cont.dat
   hif_makeimages() ## Uses cont.dat
   hif_makeimlist(specmode='refBW') ## Uses cont.dat
   hif_makeimages() ## Uses cont.dat
finally:
   h_save()
```

Figure 4: Example of an non-polarization IF Pipeline **casa_pipescript.py** script.

5.4.2 Results from running the single dish casa_pipescript.py

Running the script will create:

- A calibrated, atomospheric-corrected MS for each ASDM with a name like
  uid___A00X_XXXX_XXX.ms.atmcor.atmtype1
- A calibrated, atomospheric-corrected, baseline-subtracted MS for each ASDM with a name like
  uid___A00X_XXXX_XXX.ms.atmcor.atmtype1_bl
- Baseline subtracted image cubes of the the science targets in ***.image** format (1 per spectral window, all antennas combined, at the native correlator frequency spacing).
- A **pipeline-*/html** directory containing
  - The Pipeline WebLog (see §8).
  - The **casa_commands.log** file (see §5.5)

5.4.3 Results from running the interferometric casa_pipescript.py

Running the script through the first hif_makeimages command (calibrator imaging) will create:

- A calibrated MS for each ASDM with a name like **uid___A00X_XXXX_XXX.ms**. This ms includes both calibrator and science data and all spectral windows, with the raw data in the DATA column, and the calibrated continuum + line data in the CORRECTED column.
- Continuum images of the bandpass, phase, and (if present) check source calibrators (1 per spectral window for the bandpass and phase and 1 per spectral window per EB for the check source, in ***.image** format). To view a ***.image** file e.g. use casaviewer `image_file_name`.
- A **pipeline-*/html** directory containing:
  - The Pipeline WebLog (see §8).
  - The **casa_commands.log** file (see §5.5)

Deprecation Warning: CASA support for the standalone viewer is not expected to continue indefinitely, and users are encouraged to switch to the CARTA viewer `http://cartavis.org` for CASA images.

Running the script through the hif_mstransform command will additionally create:

- A calibrated MS for each ASDM containing only science target data (only science targets and spectral windows), with a name like **uid___A00X_XXXX_XXX_targets.ms**. This ms will have the calibrated continuum + line data in the DATA column.

Running the script through hif_uvcontsub command will result in:

- The science-target only MS **uid___A00X_XXXX_XXX_targets_line.ms**, with the calibrated continuum-subtracted line data in the DATA column.

Running the script through the final hif_makeimages command (science target spectral line imaging) will additionally create:

- Per-spw continuum images, aggregate continuum images, and continuum subtracted image cubes of at least some science targets (the number of targets may be reduced automatically – see §9.31).

5.5 CASA equivalent commands file: casa_commands.log

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 does provide the 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).

6 Modifying a Pipeline Run using casa_pipescript.py

6.1 Pipeline re-processing considerations

As a rule, it does not make sense to rerun the **casa_pipescript.py** exactly as delivered, since this will merely reproduce the calibrated MeasurementSet (which for IF Pipeline calibrated data is much more easily generated using **scriptForPI.py** or **casa_piperestorescript.py** to "restore" the calibration, as described in §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 produce modified imaging products. This section describes a few of the more common calibration and imaging changes for both the IF and SD Pipeline tasks. See the ALMA Pipeline Reference Manual for more complete details on the pipeline tasks and their inputs.

Re-running the pipeline can be very resource-intensive, both from a compute-time and disk-space perspective. For the compute time, an idea of how long the pipeline took when can be inferred from the WebLog (using the **Execution Duration** shown on the top of the "Home" page of the WebLog – see Figure 12, or the **Task Execution Statistics** that are listed for each task in the "By Task" part of the WebLog – see e.g. Figure 15. Those times, however, reflect the run times using the ALMA Operations processing clusters, which have $\gtrsim$ 256 GB RAM, and likely use parallel processing (multi-core) for imaging. Concerning disk space, to re-run SD or IF pipeline calibration, it is advisable to have a system with at least 8 GB RAM, and 50 – 75 GB free disk space per ASDM. To re-run the IF imaging pipeline, it is advisable to have a system with $\geq$64 GB RAM, and the available disk space needs to be 10 – 100 times the expected size of the final imaging products.

The above resource requirements for the IF imaging pipeline are rather daunting. However, 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, and often a reduced spectral range. 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 MeasurementSet created by the pipeline hif_mstransform command, so that part of the imaging script would need to be run first.

Please contact ALMA via the ALMA Helpdesk if assistance is needed with data reprocessing.

6.2 Preparing to run casa_pipescript.py

The following steps describe how to modify and re-run the Pipeline, starting from the archived products and directory structure created after downloading the data:

- Create **rawdata**/, **working**/, and **products**/ subdirectories
- Copy **uid*casa_pipescript.py** to **working**/**casa_pipescript.py** .

To re-run IF calibration:

- copy **flux.csv**, **antennapos.csv** (if present), and **uid*flagtemplate.txt** to the **working**/ directory (there will be one flagtemplate.py file per EB). Depending on the delivery method, **flux.csv** and **antennapos.csv** are likely to be found in **uid*auxproducts.tgz** which will need to be unzipped.

To re-run IF imaging also:

- Copy **uid*flagtargetstemplate.txt** to the **working**/ directory (note there is one per ASDM).
- Copy **cont.dat** (there will only be one per MOUS) to the **working**/ directory.

To re-run SD calibration & imaging:

- copy **jyperk.csv** and **uid*flagtemplate.txt** to the **working**/ directory (there will be one file per ASDM).

In the **rawdata**/ 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).

In the **working/** directory:

- Modify the pipeline "helper" files as desired (e.g. editing the ***flagtemplate.txt** file to add any additional flags – see §7 for other options).
- Edit **casa_pipescript.py** to only include the pipeline steps you wish to repeat (e.g. commenting out the hif_findcont or imaging steps, which are very computationally expensive).
- Start the version of CASA containing Pipeline using `casa -pipeline`
- You are now ready to run the script by typing `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).

Note that to re-run the Pipeline multiple times, it is recommended to start each time from a clean working directory containing only CASA "helper" text files and the casa_pipescript.py script.

6.3 Modifying Calibration Commands

The pipeline calibration commands can be modified to produce different results.

For instance, problematic datasets (ASDMs) can be excluded from the processing by editing the `vis=` and `session=` lists in hifa_importdata or hsd_importdata tasks in the **casa_pipescript.py** script.

As a second example, a user-specified prioritized reference antenna list can be specified via the `refant` parameter in calibration tasks, over-riding the pipeline reference antenna heuristics, by passing the desired refant list. E.g. hifa_bandpass(refant='DV06,DV07')

See the ALMA Pipeline Reference Manual for more options.

Another use case is to keep the default pipeline commands, but to change the values in the Pipeline "helper" text files to e.g. change the flux scaling, or update antenna positions (see §7 for details). The new values will be used when the relevant `hif_` commands are run.

6.4 Modifying IF Pipeline Imaging Commands

The pipeline imaging commands can be modified to produce different products. Typical reasons for re-imaging include:

- Imaging improvements to be gained from interactively editing an emission specific clean mask and cleaning more deeply. The pipeline generates a clean mask automatically (see §9.39 for specifics). Cases with moderate to strong emission (or absorption) can benefit from deeper clean with additional interactive clean masking, with the most affected property being the integrated flux density.
- Non-optimal continuum ranges. The pipeline uses heuristics that attempt to identify continuum channels over a very broad range of science target line properties. Particularly for strong line forests (hot-cores) and occasionally for TDM continuum projects the pipeline ranges can be non-optimal – too much in the first case and too little in the second.

Other science goal driven reprocessing needs may include:

- Desire to use wide image channels in imaging to increase the S/N of cubes.
- Desire to use a different Briggs `robust` image weighting than the default value of `robust`=0.5 (smaller value = smaller beam, poorer S/N; larger value = larger beam, better S/N).
- Desire to uv-taper images to increase the S/N for extended emission.
- Desire to use different continuum frequency ranges than determined by the pipeline, by modifying the **cont.dat** file (§7.6).

Some re-imaging 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.

6.5 Manual imaging after running casa_pipescript.py

6.5.1 SD Data

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. As mentioned earlier, the Single Dish Pipeline creates a calibrated MS with a filename extension of ***.ms.atmcor.atmtype1_bl** for each ASDM. The sdimaging command will make images of all MS that are specified in the `infiles` parameter. For other parameters in sdimaging, refer to the ***casa_commands.log** file.

Note that the images included in the delivery package have the 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`. If you want to change the frequency resolution and cell size, we recommend that you import the delivered FITS data cubes into CASA and regrid them using the CASA task imregrid.

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

6.5.2 IF Data

For IF data that are pipeline calibrated but *manually* imaged, the imaging commands will be included in a separate **scriptForImaging.py** script, containing all the CASA commands used to create the delivered products. In order to use this imaging script, after using **casa_pipescript.py** to recalibrate, the science spectral windows must first be "split" out from the calibrated MeasurementSets and the MeasurementSets output with a ***.split.cal** suffix. Perform the split in CASA with a command like this: `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 > MeasurementSet Name > Spectral Setup, in the ID column) or can be determined using the CASA task listobs, e.g. `listobs('uid___A002_X8925`
where the results will be reported in the CASA logger.

If the pipeline-calibrated data is restored using **scriptForPI.py**, that script can, with the appropriate parameters set, perform the split command for the user.

If a script named **scriptForFluxCalibration.py** is present in the **script/** directory, this must also be executed prior to running **scriptForImaging.py**. **scriptForPI.py** will run this script if it is present.

6.6 Manipulating the Pipeline Context

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

If the script is modified to only run a subset of the pipeline tasks, the **Context** should be saved after the last task by using `h_save`. To resume the run, use `h_resume` to load the saved **Context** before executing any pipeline tasks. See the ALMA Pipeline Reference Manual for more information.

7 Description of Pipeline "Helper" Text Files

As mentioned in §5.2, both the IF and SD pipeline use a number of text files that are read by various pipeline tasks (as indicated by comments ## in Figure 3 and 4), and which affect the pipeline results (e.g. by applying manually identified flags or by updating calibrator fluxes or antenna positions before calculating the calibration tables). These files are particularly useful for users to over-ride the default pipeline behavior when re-running the pipeline at home, as described in the following section. Below we describe all of the currently available control files, identifying whether they are used by the IF pipeline, SD pipeline, or both in the subsection heading.

7.1 flux.csv (IF Pipeline)

From Cycle 4 onward, the fluxes of standard ALMA quasar calibrators at the observed frequencies for each spw are written into the ASDM, using extrapolated values calculated from entries in the ALMA Source Catalog available at the time of observation. These fluxes are sometimes updated subsequently (thereby bracketing the observation in time), allowing for more accurate interpolated fluxes to be used for the absolute flux calibration.

Since the pipeline is usually run days to weeks after an observation is completed, better flux densities are often available at that time, so the pipeline hifa_importdata task does the following:

1. If `dbservice=True`, an online observatory database service is queried and the best flux densities for the time and frequency of the observation are interpolated, overriding values in the ASDM.
2. If the **flux.csv** text file exists in the working directory, any values therein, for example as retrieved by `analysisUtils::getALMAFluxCsv()`, will in turn override results from the online database.
3. If no flux density is available in ASDM, dbservice, or **flux.csv**, a flux of 1Jy will be assumed.
4. After evaluating this sequence of preferred sources, the flux densities for each source and spw are written into the **flux.csv** text file. Note that even if all values are taken from **flux.csv**, they will be written back to **flux.csv**, so the file's modification date will be updated.
5. The new flux value of the flux calibrator (the source with intent=AMPLITUDE) is then used in the subsequent hif_setmodels task. Values for the other calibrator intents (BANDPASS, PHASE, CHECK) are also updated, but these values are only shown for comparison against the values derived from the pipeline calibration calibration (both are shown in a table in the hifa_gfluxscale stage of the WebLog – see §9.19.

The format of the **flux.csv** file is shown in Figure 5 below. It contains one row for every spw of every calibrator (intents of AMPLITUDE, BANDPASS, PHASE or CHECK) in every ASDM in the MOUS. This file can be edited by users and the pipeline re-run in order to scale the fluxes of each ASDM to a different value for the AMPLITUDE calibrator. Changing the values of other calibrators will not have an effect on the calibration.

The original **flux.csv** file written by the pipeline upon the initial run of the hifa_importdata task, starts out with "origin=Source.xml" as part of the comment on all lines. Lines updated with the online database will have "origin=DB".

7.2 jyperk.csv (SD pipeline)

ALMA single-dish observations do not include observations of absolute amplitude calibrators. Instead, the observatory conducts regular observations of standard single-dish calibrators and stores them in an observatory database. In the hsd_k2jycal stage, the CASA task gencal retrieves the best value of these "Kelvin to Jansky" calibration factors, based on the observing date, frequency, Tsys, and source elevation from the observatory database. The appropriate values are written into the **jyperk_query.csv** text file that is read and applied.

The format of the **jyperk_query.csv** file is shown in Figure 6 below. It contains one row for every spw in every ASDM in the MOUS. This file can be edited by users and the pipeline re-run in order to scale the fluxes of each ASDM to a different value.

ms,field,spw,I,Q,U,V,spix,comment

uid___A002_Xd0adbe_Xd5a.ms,0,25,0.8818,0.0,0.0,0.0,-0.750167691515,"# field=J1550+0527 intents=AMPLITUDE,ATMOSPHERE,BANDPASS,POINTING,WVR origin=DB age=4 queried_at=2018-09-14 20:47:41 UTC # +-0.0849Jy, freq=109.517GHz, spec_index=-0.750+-0.168, Band3/7_separation=0 days, spixAge=-14 days, Band3age=5 days, setjy parameters for field 0 (J1550+0527): spix=-0.7502, reffreq='109.5167GHz', fluxdensity=[0.881754,0,0,0], au.getALMAFluxcsv v1.4207 executed on 2018-09-14 21:27:01 UT"

uid___A002_Xd0adbe_Xd5a.ms,0,27,0.8926,0.0,0.0,0.0,-0.750167691515,"# field=J1550+0527 intents=AMPLITUDE,ATMOSPHERE,BANDPASS,POINTING,WVR origin=DB age=4 queried_at=2018-09-14 20:47:42 UTC # +-0.0868Jy, freq=107.746GHz"

uid___A002_Xd0adbe_Xd5a.ms,0,29,0.9630,0.0,0.0,0.0,-0.750167691515,"# field=J1550+0527 intents=AMPLITUDE,ATMOSPHERE,BANDPASS,POINTING,WVR origin=DB age=4 queried_at=2018-09-14 20:47:42 UTC # +-0.1032Jy, freq=97.383GHz"

uid___A002_Xd0adbe_Xd5a.ms,0,31,0.9767,0.0,0.0,0.0,-0.750167691515,"# field=J1550+0527 intents=AMPLITUDE,ATMOSPHERE,BANDPASS,POINTING,WVR origin=DB age=4 queried_at=2018-09-14 20:47:43 UTC # +-0.1061Jy, freq=95.563GHz"

uid___A002_Xd0adbe_Xd5a.ms,1,25,0.1435,0.0,0.0,0.0,-0.577110147696,"# field=J1458+0416 intents=ATMOSPHERE,PHASE,WVR origin=Source.xml age=N/A queried_at=N/A # +-0.0076Jy, freq=109.517GHz, spec_index=-0.577+-0.121, Band3/7_separation=0 days, spixAge=+4 days, Band3age=4 days, setjy parameters for field 1 (J1458+0416): spix=-0.5771, reffreq='109.5167GHz', fluxdensity=[0.143542,0,0,0]"

uid___A002_Xd0adbe_Xd5a.ms,1,27,0.1449,0.0,0.0,0.0,-0.577110147696,"# field=J1458+0416 intents=ATMOSPHERE,PHASE,WVR origin=Source.xml age=N/A queried_at=N/A # +-0.0077Jy, freq=107.746GHz"

Figure 5: Example of a **flux.csv** file used by the interferometric pipeline (one per MOUS) – blank lines added here for readability.

```
MS,Antenna,Spwid,Polarization,Factor

uid___A002_Xb1d975_Xf65.ms,PM02,17,I,43.785
uid___A002_Xb1d975_Xf65.ms,PM02,19,I,43.782
uid___A002_Xb1d975_Xf65.ms,PM02,21,I,43.664
uid___A002_Xb1d975_Xf65.ms,PM02,23,I,43.63
uid___A002_Xb1d975_Xf65.ms,PM02,25,I,43.638
uid___A002_Xb1d975_Xf65.ms,PM02,27,I,43.64
uid___A002_Xb1d975_Xf65.ms,PM02,29,I,43.641
uid___A002_Xb1d975_Xf65.ms,PM04,17,I,43.785
uid___A002_Xb1d975_Xf65.ms,PM04,19,I,43.782
uid___A002_Xb1d975_Xf65.ms,PM04,21,I,43.664
uid___A002_Xb1d975_Xf65.ms,PM04,23,I,43.63
uid___A002_Xb1d975_Xf65.ms,PM04,25,I,43.638
uid___A002_Xb1d975_Xf65.ms,PM04,27,I,43.64
uid___A002_Xb1d975_Xf65.ms,PM04,29,I,43.641
uid___A002_Xb1cc39_X1e46.ms,PM02,17,I,43.782
uid___A002_Xb1cc39_X1e46.ms,PM02,19,I,43.778
uid___A002_Xb1cc39_X1e46.ms,PM02,21,I,43.661
uid___A002_Xb1cc39_X1e46.ms,PM02,23,I,43.627
uid___A002_Xb1cc39_X1e46.ms,PM02,25,I,43.635
uid___A002_Xb1cc39_X1e46.ms,PM02,27,I,43.636
uid___A002_Xb1cc39_X1e46.ms,PM02,29,I,43.638
uid___A002_Xb1cc39_X1e46.ms,PM04,17,I,43.782
uid___A002_Xb1cc39_X1e46.ms,PM04,19,I,43.778
uid___A002_Xb1cc39_X1e46.ms,PM04,21,I,43.661
uid___A002_Xb1cc39_X1e46.ms,PM04,23,I,43.627
uid___A002_Xb1cc39_X1e46.ms,PM04,25,I,43.635
uid___A002_Xb1cc39_X1e46.ms,PM04,27,I,43.636
uid___A002_Xb1cc39_X1e46.ms,PM04,29,I,43.638
```

Figure 6: Example of a **jyperk_query.csv** file used by the single-dish pipeline (one per MOUS)

```
name,antenna,xoff,yoff,zoff,comment

uid___A002_Xbb63ba_X18b0.ms,DA42,1.27536e-04,-3.54105e-04,-2.38014e-04,
uid___A002_Xbb63ba_X18b0.ms,DA46,1.98098e-04,-5.34528e-04,-3.65393e-04,
uid___A002_Xbb63ba_X18b0.ms,DA49,1.69321e-04,-2.81896e-04,-1.76309e-04,
uid___A002_Xbb63ba_X18b0.ms,DA52,-4.06882e-05,3.45109e-04,3.15047e-04,
uid___A002_Xbb63ba_X18b0.ms,DA62,-1.79249e-04,2.50696e-04,7.12701e-05,
uid___A002_Xbb63ba_X18b0.ms,DV03,-3.92453e-04,2.85912e-04,3.14499e-04,
uid___A002_Xbb63ba_X18b0.ms,DV08,-2.76083e-04,7.41071e-04,1.87197e-04,
uid___A002_Xbb63ba_X18b0.ms,DV14,-5.41156e-05,2.61746e-04,3.44329e-04,
uid___A002_Xbb63ba_X18b0.ms,DV15,-1.21313e-04,3.67910e-04,1.49062e-04,
uid___A002_Xbb63ba_X18b0.ms,DV23,1.73257e-04,1.36402e-04,-1.23099e-04,
uid___A002_Xbb63ba_X18b0.ms,DV25,3.12879e-03,-5.08802e-03,-2.87630e-03,
uid___A002_Xbb63ba_X18b0.ms,PM03,1.78948e-04,-5.00918e-04,-2.14580e-04,
uid___A002_Xbb63ba_X1626.ms,DA42,1.27536e-04,-3.54105e-04,-2.38014e-04,
uid___A002_Xbb63ba_X1626.ms,DA46,1.98098e-04,-5.34528e-04,-3.65393e-04,
uid___A002_Xbb63ba_X1626.ms,DA49,1.69321e-04,-2.81896e-04,-1.76309e-04,
uid___A002_Xbb63ba_X1626.ms,DA52,-4.06882e-05,3.45109e-04,3.15047e-04,
...
```

Figure 7: Example of a **antennapos.csv** file used by the interferometric pipeline (one per MOUS); the offset units are in meters. Corrections that are comparable, or larger than the observing wavelength are consequential.

7.3 antennapos.csv (IF pipeline)

The position of every antenna in an interferometric observation must be known in order to properly transfer the calibration from the phase calibrator to the science targets. If these positions have errors, it will lead to phase errors in the imaging of the science target (increasing with telescope position error and separation between the phase calibrator and science target).

The antenna positions are calculated by special observatory observations taken outside of PI science observing, and the positions stored in an observatory database. This database is queried at the time of an SB execution, and the appropriate antenna positions are written into the ASDM. These positions are sometimes updated subsequently, especially if the observation happened shortly after an array reconfiguration or if an array element was recently moved.

Since the pipeline is usually run days to weeks after an observation, ALMA staff run commands outside of the pipeline to get the best-available antenna positions at the time the pipeline is run. These are written into the **antennapos.csv** text file, which is then read in by the pipeline hifa_antpos task (if it exists in the directory where the pipeline is run) and used to over-ride the values in the ASDM.

The format of the **antennapos.csv** file is shown in Figure 7 below. It contains one row for every antenna in every ASDM in the MOUS. This file can be edited by users and the pipeline re-run in order to correct antenna position errors.

7.4 uid*flagtemplate.txt & uid*flagtsystemplate.txt (both pipelines)

The extensive pipeline flagging heuristics may sometimes prove inadequate, and users may wish to add additional flagging commands to exclude these data from the calibration. These manually-identified flags can be introduced to any Pipeline reduction by editing the **uid*flagtemplate.txt** files that are provided with the archived pipeline products and rerunning the pipeline calibration steps. There should be one file for every MS that needs additional flagging, with a name matching the MS uid. The flag commands can be any valid CASA `flagdata` command. For interferometric data, use the <AntID> syntax to flag only cross-correlation data for <AntID>, while for single dish data use the "<AntID>&&*" syntax to flag both cross- and auto-correlation data for <AntID>, and

```
# User flagging commands file for the calibration pipeline
# Examples
# Note: Do not put spaces inside the reason string !
#
# mode='manual' antenna='DV02;DV03&DA51' spw='22,24:150~175' reason='QA2:applycal_amplitude_frequency'
#
# mode='manual' spw='22' timerange='2018/02/10/00:01:01.0~00:02:01.0' reason='QA2:timegaincal_phase_time'
#
# TP flagging: The 'other' option is intended for bad TP pointing
# mode='manual' antenna='PM01&&PM01' reason='QA2:other_bad_pointing'
#
# Tsys flagging:
# mode='manual' antenna='DV02;DV03&DA51' spw='22,24' reason='QA2:tsysflag_tsys_frequency'
#
mode='manual' timerange='2016/12/05/03:55:30.1440' reason='QA2:applycal_outlier_amp'
mode='manual' antenna='PM02&&&' reason='PRTSIR2995'
```

Figure 8: Example of a **uid*flagtemplate.txt** file used by both the interferometric and single-dish pipeline (one per ASDM)

the "<AntID>&&&" syntax to flag auto-correlation data for <AntID>. Examples of the syntax to use in editing these files are given at the top of the files **uid*flagtemplate.txt** (see Figure 8).

These flag files will be picked up by the hifa_flagdata/hsd_flagdata tasks which are run before the calibration tasks, therefore excluding the manually identified data from being used to generate the calibration tables.

Since the tsys spectra are calculated from a different ASDM subtable, any commands that the user desires to flag the tsys spectral windows have to be applied differently by the pipeline, and so have to be put into the separate ***.flagtsystemplate.txt** file. The flagging syntax is the same, only that those commands should refer to tsys spectral windows in particular.

7.5 uid*flagtargetstemplate.txt (IF imaging pipeline)

Users should examine the science data (e.g. using the CASA task plotms, or examining the MS using the CASA viewer). If bad data are found, flagging commands can be added to the **uid*flagtargetstemplate.txt** files that are provided with the archived pipeline products to exclude these data from subsequent imaging. There should be one file for every MS that needs additional flagging, with a name matching the MS uid. As for the **{uid*flagtemplate.txt}** files, the flag commands can be any valid CASA `flagdata` command. If these files are found in the directory where the pipeline is run, they will be picked up by the hifa_flagtargets task and applied to the data before science target imaging.

Deprecation Warning: hifa_flagtargets is rarely used in operations and may be removed from the standard recipe, although it is expected to remain a supported pipeline task.

7.6 cont.dat (IF imaging pipeline)

The pipeline-identified continuum frequency ranges, in LSRK units, for each spectral window of each source are entered into a file called **cont.dat** that is delivered with the pipeline products. This file lists the LSRK frequency ranges that were used to make the per-spw and aggregate continuum images, and for fitting and subtracting the continuum for the image cubes. When this file is in the directory where the pipeline is (re)run, the pipeline will use these entries directly instead of using its own heuristics (via the hif_findcont task) to determine them. Therefore, a user can edit this file (or create their own) in order to use a different continuum range. Alternatively, a user-defined file name can be passed as an argument to the hif_makeimlist task. An example **cont.dat** file is shown in Figure 9.

```
Field: G09_0850-0019
SpectralWindow: 17
NONE

SpectralWindow: 19
337.659971874~339.253995016GHz LSRK

SpectralWindow: 21

SpectralWindow: 25
349.755169752~351.067897111GHz LSRK
351.271057297~351.380451244GHz LSRK
```

Figure 9: Example of a **cont.dat** file used by the interferometric pipeline (one per MOUS). This example is for an MOUS that has 5 spectral windows; the entry for spw 21 is empty and spw 23 is omitted, which will result in the hif_findcont task determining the frequency ranges for these spectral windows.

The behavior of hif_findcont and the subsequent continuum subtraction and continuum and line imaging commands is as follows:

1. If the SpectralWindow line in **cont.dat** is followed by one or more frequency ranges, hif_findcont will not run its heuristics on the spw. The task hif_uvcontsub will use these frequency ranges to fit and subtract the continuum from this spw. Subsequent continuum images will include only these frequency ranges for this spw, and the spw line cubes will be made from the continuum subtracted data.
2. If the SpectralWindow line is followed by a line containing "NONE", hif_findcont will not run its heuristics on the spw (if the delivered **cont.dat** file contains spw entries with "NONE", this indicates that the hif_findcont task failed to find any continuum frequency ranges). The task hif_uvcontsub will skip fitting this spw. Subsequent continuum images will include the full frequency range for this spw (logging a message in the WebLog), and the spw line cubes will have had no continuum subtraction performed.
3. If a spw is not followed by a frequency range or is missing from **cont.dat** when hif_findcont is run, then it will try to find the frequency ranges, and these will be used to make subsequent continuum images, and for continuum subtraction.

8 The Pipeline WebLog

This section gives an overview of the Pipeline WebLog, which is a collection of webpages with diagnostic messages, tables, figures, and "Quality Assurance" (**QA**) scores. It is reviewed, along with the pipeline calibration and imaging products, as part of the ALMA Quality Assurance process, but also provides important information to investigators on how the pipeline calibration and imaging steps went.

The section describes common elements to the single dish and interferometric Pipeline WebLogs. Subsequent sections present descriptions of the SD- or IF- specific "By Task" part of the WebLog.

8.1 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.tgz`. This will provide a **pipeline***/**html** directory containing the WebLog, which can be viewed using a web browser e.g. `firefox index.html`.

Note about browser security: Most modern browsers now prevent javascript when using a file:// URL (e.g. viewing a WebLog on a local directory). The page in Figure 10 should appear, which describes the mitigation options. One can either use a localhost web server that is now delivered with CASA+Pipeline, or one can run a local http server with a command like "python3 -m http.server 8080 –bind 127.0.0.1", or one can adjust the security settings in the browser by going to about:config and setting either ***privacy.file_unique_origin*** *or* ***security.fileuri.strict_origin_policy*** *to False, whichever is available.*

The WebLog provides both an overview of datasets and details of the pipeline processing. Therefore many 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 (those produced by the CASA tasks plotms and plotbandpass) will have a "Plot command" link that provides the CASA command to reproduce the plot (see Figure 11). When viewing image products, a similar link will provide the tclean command that produced the image. 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 a QA score to give an "at a glance" indication of any trouble points.

8.2 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.

8.3 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 "environment" link next to the CASA version will open a popup detailing hardware and software used, and number of cores if MPI; see Figure 13). Clicking on the bar at the top of the home page (see Figure 12) enables navigation to **By Topic** or **By Task**. CASA relies on earth Geodetic information to determine the geometry of the array - this information is stored in the IERS Earth Orientation Parameters (eop2000, measured values), and IERS Predicted Earth orientation. Since it takes time to analyze measurements and update those in CASA, usually data processed within a month or two of the observation date uses the Predict table.

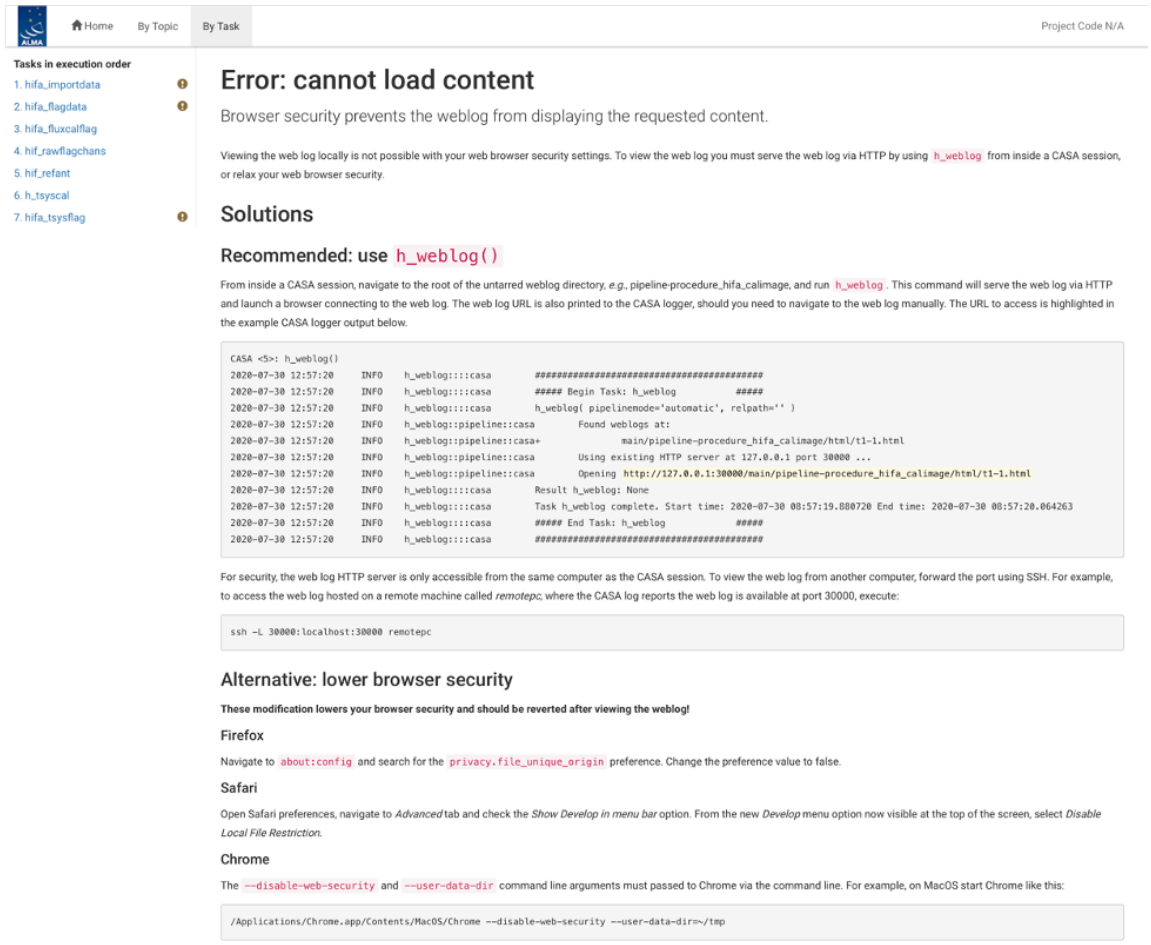


Figure 10: This page will usually appear if your browser is blocking javascript in viewing a local WebLog (a `file://` URL). It is recommended to run a local html server and viewing the local file with `http://` instead.

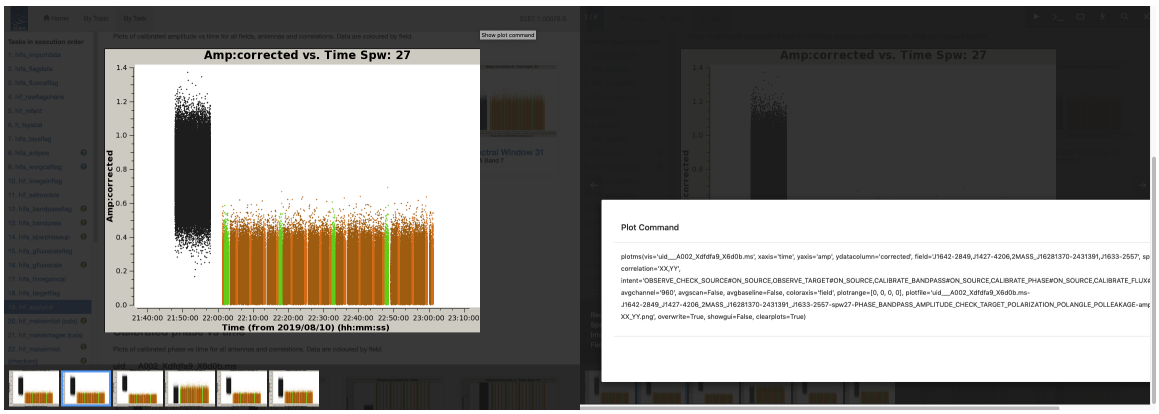
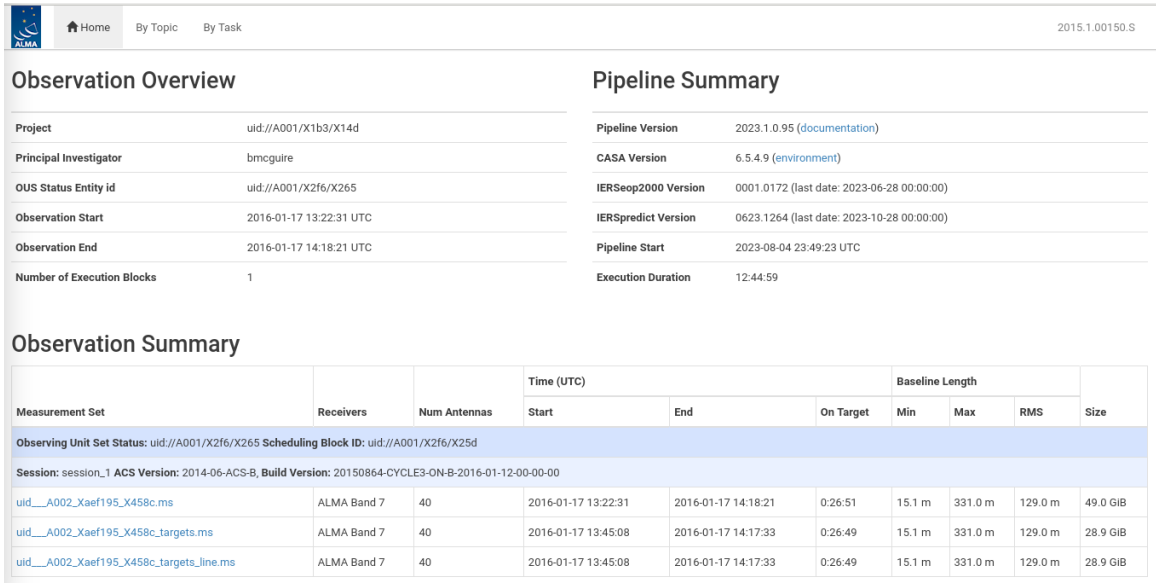


Figure 11: Example of WebLog plot with a "Plot command" link (>_) that, when clicked, opens a popup window containing the CASA command for reproducing the plot that can be copied and pasted elsewhere.



Home By Topic By Task 2015.1.00150.S

Observation Overview

Project	uid://A001/X1b3/X14d
Principal Investigator	bmcguire
OUS Status Entity id	uid://A001/X2f6/X265
Observation Start	2016-01-17 13:22:31 UTC
Observation End	2016-01-17 14:18:21 UTC
Number of Execution Blocks	1

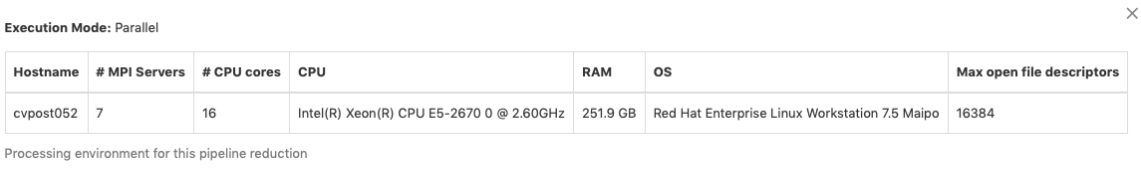
Pipeline Summary

Pipeline Version	2023.1.0.95 (documentation)
CASA Version	6.5.4.9 (environment)
IERSeop2000 Version	0001.0172 (last date: 2023-06-28 00:00:00)
IERSpredict Version	0623.1264 (last date: 2023-10-28 00:00:00)
Pipeline Start	2023-08-04 23:49:23 UTC
Execution Duration	12:44:59

Observation Summary

			Time (UTC)			Baseline Length			
Measurement Set	**Receivers**	**Num Antennas**	**Start**	**End**	**On Target**	**Min**	**Max**	**RMS**	**Size**
Observing Unit Set Status: uid://A001/X2f6/X265 **Scheduling Block ID:** uid://A001/X2f6/X25d									
Session: session_1 **ACS Version:** 2014-06-ACS-B, **Build Version:** 20150864-CYCLE3-ON-B-2016-01-12-00-00-00									
uid___A002_Xaef195_X458c.ms	ALMA Band 7	40	2016-01-17 13:22:31	2016-01-17 14:18:21	0:26:51	15.1 m	331.0 m	129.0 m	49.0 GiB
uid___A002_Xaef195_X458c_targets.ms	ALMA Band 7	40	2016-01-17 13:45:08	2016-01-17 14:17:33	0:26:49	15.1 m	331.0 m	129.0 m	28.9 GiB
uid___A002_Xaef195_X458c_targets_line.ms	ALMA Band 7	40	2016-01-17 13:45:08	2016-01-17 14:17:33	0:26:49	15.1 m	331.0 m	129.0 m	28.9 GiB

Figure 12: WebLog Home Page



Execution Mode: Parallel

Hostname	# MPI Servers	# CPU cores	CPU	RAM	OS	Max open file descriptors
cvpost052	7	16	Intel(R) Xeon(R) CPU E5-2670 0 @ 2.60GHz	251.9 GB	Red Hat Enterprise Linux Workstation 7.5 Maipo	16384

Processing environment for this pipeline reduction

Figure 13: Processing Environment popup window

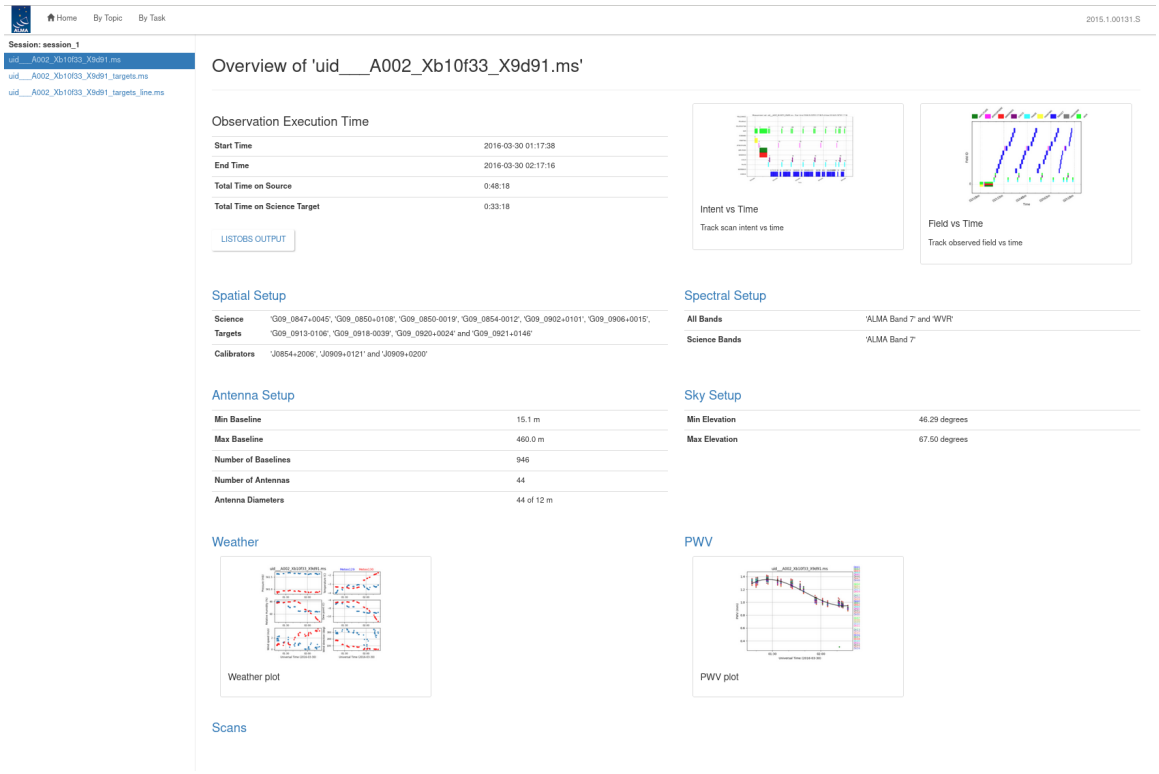


Figure 14: MeasurementSet Overview Page. Click on the table headings in blue for more information about each category.

The **Observation Summary** table lists all the MeasurementSets included in the pipeline processing, grouped by observing "sessions". Each MeasurementSet is calibrated independently by the pipeline. For data that have been run through the imaging stages of the pipeline, three MS will be listed – the original one including all data and spectral windows, a targets.ms containing only calibrated continuum+line science target data, and a targets_line.ms containing only calibrated continuum-subtracted science target line 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 MeasurementSet. To view the observational setup of each MeasurementSet in more detail, click on the name of it to go to its overview page.

8.3.1 MeasurementSet Overview pages

Clicking on the MeasurementSet name in the **Observational Summary** table brings up the **MeasurementSet Overview page** (Figure 14). Each MeasurementSet **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**.

8.4 By Topic Summary Page

The **By Topic** summary page provides an overview of the lowest QA scores for tasks grouped by processing topic, followed by a listing of all "Error!" or "Warning!" level task notifications, and then a **Flagging Summaries** section which presents a graphic depicting the fraction of data flagged by antenna & spw for every calibrator and science target.

8.5 By Task Summary Page

The **By Task** summary page (Figure 15) gives a list of all the pipeline stages performed on the dataset. It is not displayed per MeasurementSet as the Pipeline performs each step on every MeasurementSet sequentially before proceeding to the next step; e.g. it will import and register all MeasurementSets with the Pipeline before proceeding to perform the ALMA deterministic flagging step on each MeasurementSet. The name of each step on the By Task page is a link to the corresponding Task Page which provides detailed information for the task as described below (Sec. 8.6).

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 other colors indicate less than perfect QA scores following the assessment described in Sec. 8.7. Encircled symbols to the left of each task name (⑦, ⓘ, ⓧ), indicate that there are informative QA messages or important notifications on the task pages. Stages with a ⓘ symbol next to them can indicate either poor QA scores (QA Score progress bar on the Task Summaries page will be yellow with a short descriptive text) or "Warning" notifications, which are important messages about the stage execution but which are not thought to indicate a quality issue with the data (i.e. the QA Score progress bar on the Task Summaries page may still be green, e.g. the hifa_tsysflag and hif_lowgainflag stages in Figure 15).

8.5.1 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 pipeline restoration scripts described in §5.1: **casa_pipescript.py** and **casa_piperestorescript.py**, and the **casa_commands.log** file described in §5.5.

8.6 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 hif_applycal page.** At the top of the page will be the **Pipeline QA** scores and associated messages and any **Task Notification** (see Figure 16). If there are more than one QA messages, the message corresponding to the lowest score will be displayed with a link to all QA scores and messages. Clicking this link will expand the QA score table to show all entries. Similarly, if there are more than one notifications, the most severe will be displayed along with a link to all notifications. These messages are color-coded by severity: green means the quality check were fine, blue are informative messages that should not impact the quality of the processing, and yellow and red indicate important notifications or that a QA heuristic was triggered (see §8).

At the bottom of each task page are expandable sections for **Input Parameters** and **Task Execution Statistics**, and links to the CASA log commands for the specific task. An example is given in Figure 17.

8.6.1 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 MeasurementSet name. Some of these plots can be filtered by entering one or more

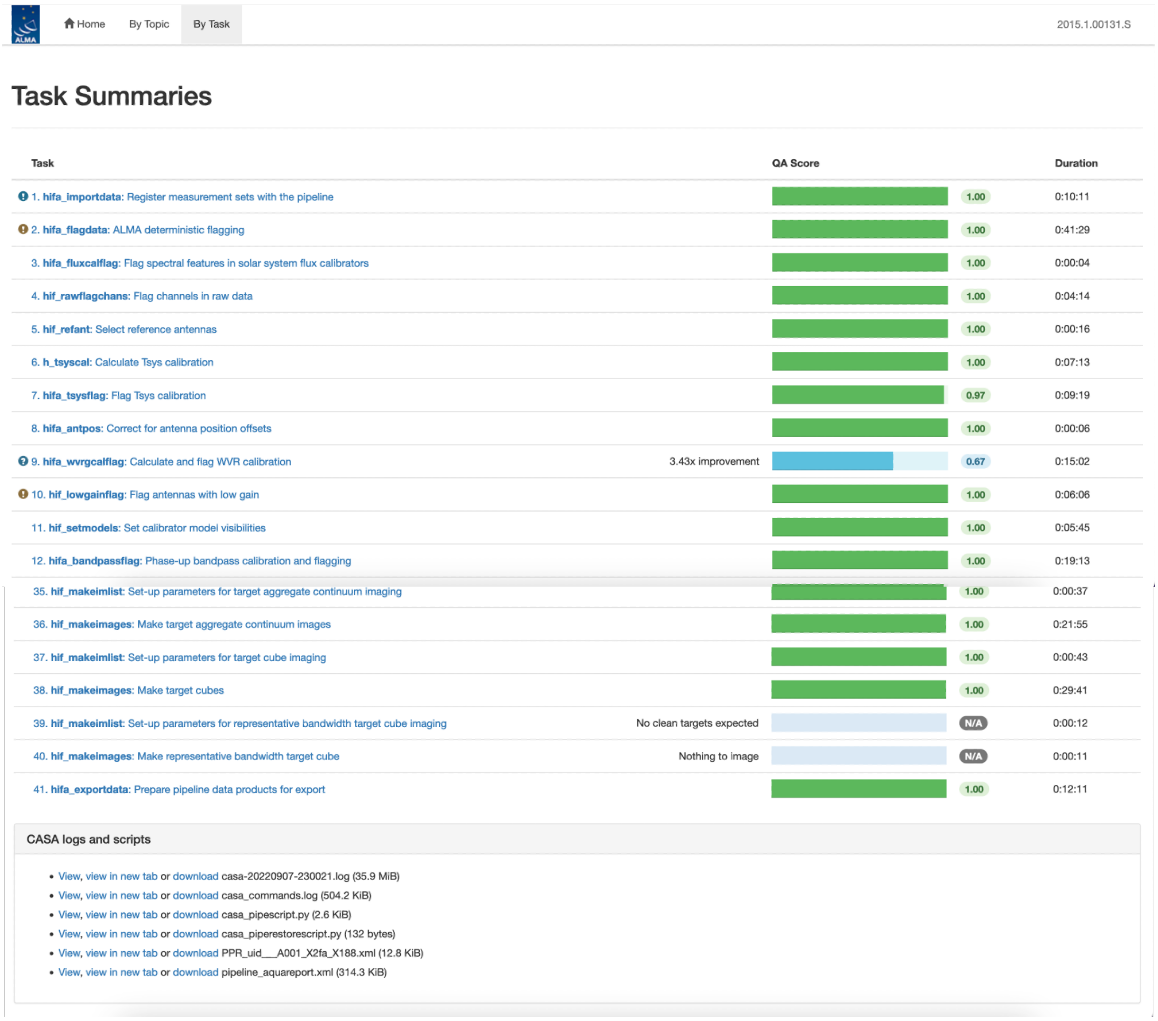


Figure 15: The By Task summary view. The figure has been truncated so both the top and bottom can be seen. Each pipeline stage is listed, along with its QA score (colored bars to the right), computing run-time for each stage, and links to the CASA logs and scripts.

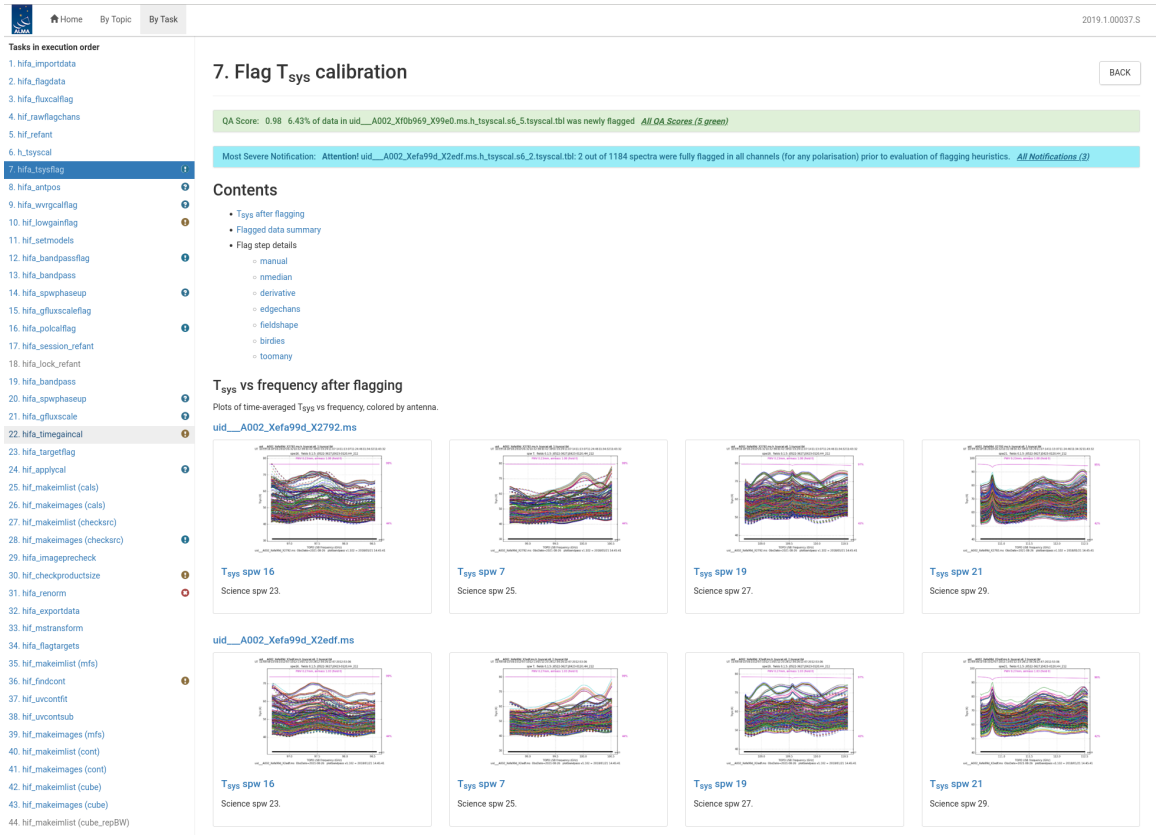


Figure 16: The hifa_tsysflag task page, showing the task notifications and QA score at the top, and diagnostic plots (Tsys for each spw grouped by MS). Further down on the page are flagging summary tables. To see the sub-page for this task, click on the MeasurementSet name in blue above each set of plots. This will take you to a page of detailed plots for individual MS/antenna/spectral windows (see Figure 18 for an example).

Figure 17: Bottom of the hifa_timegaincal page, showing the expandable sections for Input Parameters, Task Execution Statistics and link to the CASA logs for this stage.

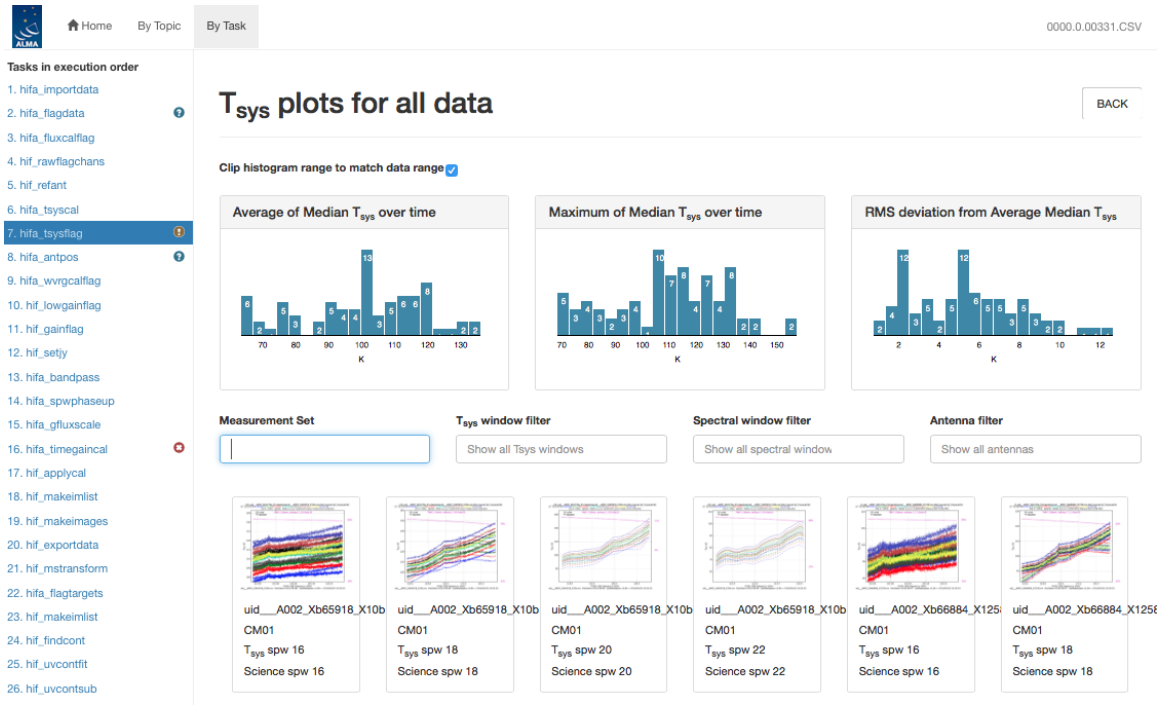


Figure 18: Unfiltered view of the hifa_tsysflag sub-page. The page is arrived at by clicking on the MeasurementSet link from the hifa_tsysflag task page (Figure 16). Only the first row of plots are shown; many more appear below (one for each MS, antenna, spw combination). This page has histograms of three metric scores based on the median Tsys that can also be used to filter the plots that are displayed.

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 18 – Figure 20, using the **By Task** > **hifa_tsysflag: Flag Tsys calibration** pages.

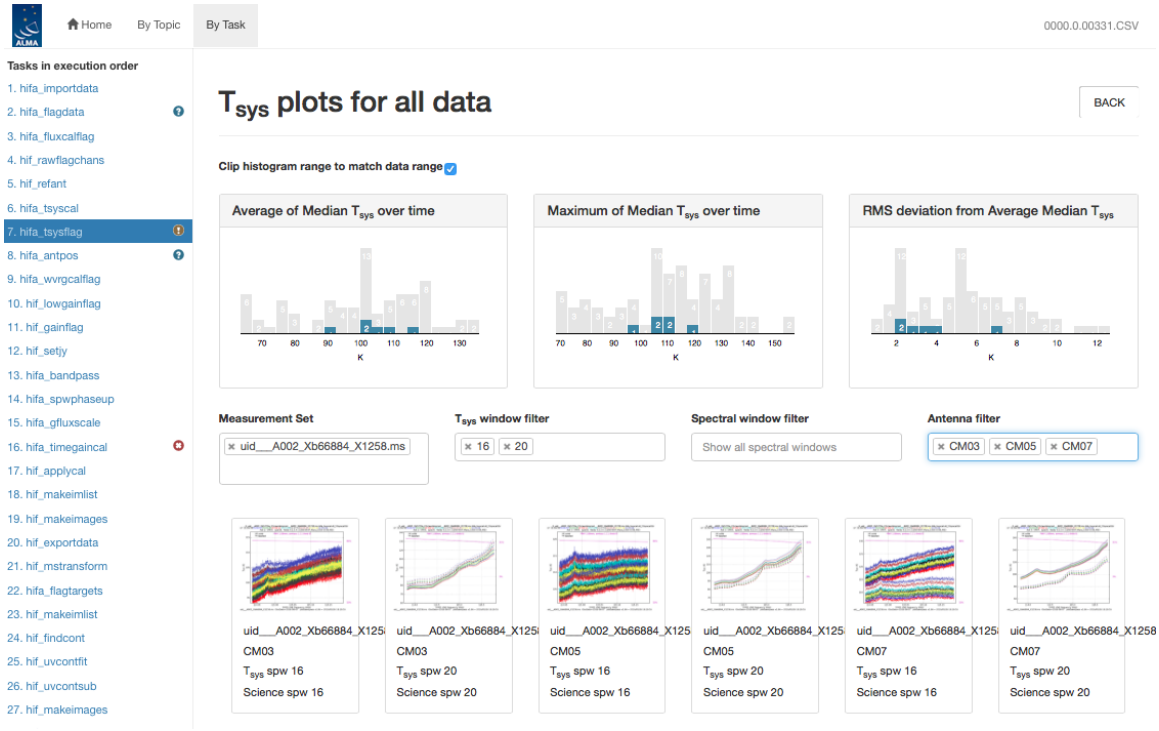


Figure 19: Same as Figure 18, but with a specific MS, Tsys window, and antenna filter set. The corresponding plots are displayed below, and their metric scores are shown by blue shading in the histogram plots.

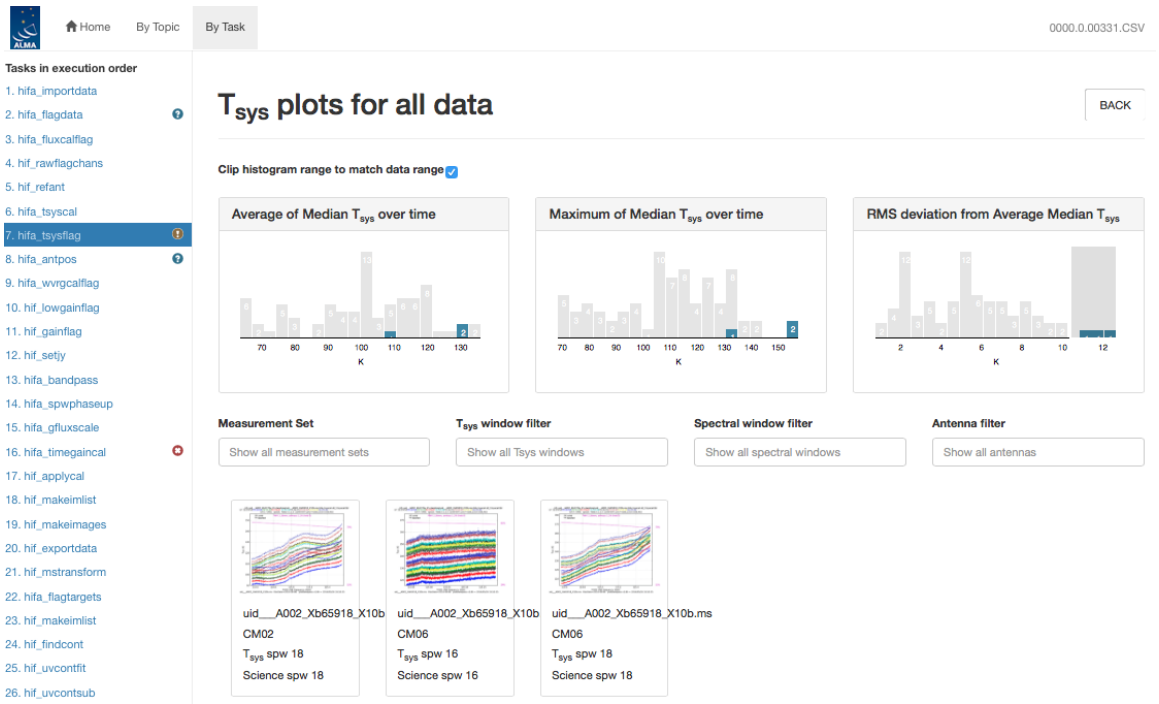


Figure 20: Same as Figure 18, but filtering to the plot of interest by using the mouse to draw a grey box on the highest histogram values in the RMS deviation from Average Median Tsys histogram plot (upper right). To clear the grey box filters on the histograms, click on any white space in the histograms.

8.7 WebLog Quality Assessment (QA) Scoring

Pipeline tasks have scores associated with them in order to quantify the quality of the dataset and the calibration. These scores are designed to inform data inspection as part of the ALMA quality assurance or "QA2" process. When scores are calculated per ASDM, the overall task QA score is taken as the lowest of the per-ASDM scores. The scores are between 0.0 and 1.0 and are colorized according to the following table:

Score	Color	Meaning
>0.90-1.00	Green	No issues identified
>0.66-0.90	Blue	No serious issues identified, but a note has been added
>0.33-0.66	Yellow	QA warning triggered; carefully inspect the results for this stage
0.00-0.33	Red	Serious issue; may not meet quality standards

The individual QA scores and associated messages appear at the top of each task WebLog page. If there is more than one QA score and message, this section is expandable by clicking on the "All QA Scores" link (Figure 16).

8.7.1 Interferometric Pipeline QA Scores

Pipeline Task	QA Scoring Metric	Score
hifa_importdata	Required calibrators are present	0.0 if required calibrators are not present; 0.5 if polarization calibrator present but pipeline was not run via the polarization recipe; else 1.0.
	Appropriate calibrator flux densities if `dbservice=True`	0.3 if the calibrator database cannot be reached, or it returns a failure code; else 1.0. 0.5 if the database returns a warning, or the nearest measurement is more than 14 days old; else 1.0.
	Appropriate calibrator flux densities if `dbservice=False`	0.3 if `Origin=Source.xml` in **flux.csv**; else 1.0.
	The input data are "clean"	0.0 if there is an existing processing history or missing RA/Dec for calibrators; 0.9 if high frequency data; else 1.0.
	Parallactic angle coverage (polarization recipes only)	0.66 if the parallactic angle coverage of the polarization calibrator over all ms in a session is less than 60°, else 1.0.
hifa_flagdata	Percentage of incremental flagging	Considering the sum of the following flag types: 'Before Task', 'QA0', 'Online Flags', 'Flagging Template', and 'Shadowed Antennas': Score is 0.0 if flag fraction is ≥50%, 1.0 if flag fraction is ≤5%, and linearly interpolated between 0 and 1 for fractions between 60% and 5%.
	SPWs with low mean transmission	0.33 if the representative spw has a mean transmission <5%; 0.90 if any non-representative spw has a mean transmission <10%; else 1.0.
hifa_fluxcalflag	Percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0.
	spws mapped as a result of flagging	0.66 if there are any mapped spws, else 1.0.

table continued on next page

table continued from previous page

hif_rawflagchans	Percentage of data flagged due to deviant channels in rawdata	A score equal to (1-percentage of data newly flagged). e.g., score = 1.0 if 0% flagged, 0.75 if 25% flagged, etc..
hif_refant	Ability to choose a refant	1.0 if suitable reference antenna is found.
h_tsyscal	Ability to calculate tsys calibration tables	1.0 if all SPWs mapped to Tsys window.
hifa_tsysflag	Percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0.
hifa_antpos	If antenna positional corrections were applied	1.0 if no corrections needed; 0.9 if one or more antennas were corrected.
hifa_wvrgcalflag	Evaluate whether WVR corrections improve calibrator phases (12m Array only)	Score based on the WVR improvement ratio = RMS(without WVR)/RMS(with WVR), the number of antennas with WVR corrections flagged, the number of antennas with elevated (> 0.5 mm) path length RMS or channel-to-channel "discrepancy", and the phase-RMS, according to the scoring workflow given in Figure 21.
hif_lowgainflag	Flagging view created and Percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5; >50%: score=0.0
hif_setmodels	Model flux for Amplitude calibrator successfully set	1.0 if the flux density of the Amplitude calibrator is successfully set for all spw, and if the spectral index of the bandpass calibrator is set.
hifa_bandpassflag	Percentage of incremental flagging	For BANDPASS intents, a score equal to (1-percentage of data newly flagged).
hifa_bandpass	Signal-to-noise of amplitude bandpass solution	An error function with 1-sigma deviation of 1.0 for the amplitude signal-to-noise ratio.
	phase vs. frequency derivative deviation (DD)	An error function with 1-sigma deviation of 0.03 for the "outlier fraction" (fraction of channels with a phase change between channels that is greater than 5 MAD), mapped to a linear score of 0.34 – 0.66 for DD < 0.2 (more than 12% outliers), to 0.67 – 0.9 for DD between 0.2 – 0.3, and 0.91 – 1.0 for DD >= 0.3 (less than 8% outliers).
hifa_spwphaseup	Median achieved SNR	Phase calibrators: 1.0 if SNR > 24; 0.9 if 16 < SNR ≤24; 0.66 if 9.6 < SNR ≤ 16; else 0.33. Check sources: 1.0 if SNR > 24; 0.9 if 16 < SNR ≤ 24, 0.8 if 9.6 < SNR ≤ 16; else 0.7.
	Median phase RMS over the phase calibrator cycle time on the outermost baselines	1.0 for RMS< 30°, 0.9 for RMS between 30° – 50°, 0.66 – 0.34 for RMS between 50° – 67°, 0.33 – 0.0 for RMS > 67°. 0.9 if unable to assess.

table continued on next page

table continued from previous page

hifa_gfluxscaleflag	Percentage of incremental flagging	Each intent gets a score equal to (1-percentage of data newly flagged), and the final score is the multiplication of the per-intent scores; i.e., if AMPLITUDE has 10% data newly flagged, and PHASE has 40% data newly flagged, then the total score will be (1-0.1)*(1-0.4)=0.54
hifa_polcalflag (polarization recipes only)	Percentage of incremental flagging	For POL* intents, a score equal to (1-percentage of data newly flagged).
hifa_session_refant (polarization recipes only)	Ability to determine a refant	1.0 if suitable reference antenna is found
hifa_gfluxscale	SNR of fitted flux values	1.0 if SNR >20, 0.0 if SNR < 5, and a linearly scaled value in between. If there are missing derived fluxes, then the ratio of the number of derived values to the number of spws is computed, and the lessor of this and the SNR score is used.
	consistency of the derived flux densities across spw compared to catalog fluxes	Defining Rspw = (derived flux for an SPW) / (catalog flux for that SPW), and Kspw = Rspw / (Rspw of the spw with the highest signal-to-noise), then the QA score is based on the the spw with the largest deviation of Kspw from 1.0. The QA score is 1.0 for Max(| 1 - Kspw|) < 0.1; 0.75 for Max(| 1 - Kspw|) between 0.1 – 0.2, and 0.5 for higher values.
hifa_timegaincal	Outliers in phase offset plots	There are three tests comparing each antennas per-ms/spw/polarization/scan phase solution against the phase solution for that antenna calculated by combining all spws: (1) the maximum offset for any scan; (2) the mean offset over all scans; and (3) the standard deviation of the offsets over all scans. If the standard deviation of all offsets over all antennas (σ_{off}) for a spw is > 15°, then that spw is considered too noisy and given a QA score of 0.82 and no other tests are performed. Otherwise if one of the following tests is met, the score is set to 0.5: the maximum offset is > 30° or 6×σ_{off}; the mean offset is > 30° or 6×$\sigma_{off}/\sqrt{N_{ant}}$; or if the number of scan solutions is >3 and the standard deviation of the offsets is > 30° or 4×σ_{off}. Otherwise if any of the above tests are met using limits of (15°, 5°, 15°) for the max, mean & standard deviation tests, the score is set to 0.8. If a spw has a mix of antennas with scores of 0.5 and 0.8, then a score of 0.5 is adopted and the antennas with that score indicated in the QA message in bold text. If any antenna has <4 solutions, there is an additional score of 0.85 and a message reporting this. The stage score is the minimum of all these scores.

table continued on next page

table continued from previous page

hifa_targetflag	Percentage of incremental flagging	For TARGET intents, a score equal to (1-percentage of data newly flagged).
hifa_polcal (polarization recipes only)	Gain Ratios	1.0 if between 0.90 – 1.10 inclusive, else 0.65.
	Gain Ratio RMS	1.0 if the value after polarization calibration for all scans is less than or equal to 0.02, else 0.60.
	D-term solutions	1.0 if less than or equal to 0.10, 0.75 if between 0.10 – 0.15, else 0.55.
	Residual polarization	1.0 if less than or equal to 0.001, else 0.50.
hif_applycal	Percentage of incremental flagging	Additional 0%-5% flagging: score=1.0; flagging 5%-50% => 1.0...0.5, >50%: score=0.0. This QA score calculation is restricted to scans matching the 'TARGET' intent if present. If no 'TARGET' intent data is present, a warning is raised, and the QA score calculation reverts to using scans for any available intent.
	Determine if corrected Amp/Phase vs. Frequency of any antenna differs from the mean for all antennas	0.9 if linear fit for any antenna, calculated on a per-intent, per-polarization basis, differs significantly from the mean fit for all antennas (for amplitude) or from zero (for phase). Otherwise 1.0. The fits are calculated on a per-scan basis as well as averaging over all scans.
hif_makeimlist	Determine if expected targets/spw will be imaged	The number of clean targets divided by the number expected.
hif_makeimages (all sources)	`tclean` divergence	0.0 if the `tclean` algorithm diverges, else 1.0.
	Expected image not made	0.34 if an expected image is not created, else 1.0.
	Determine if noise is close to theoretical	Ratio of sensitivity measured in non-pbcor image in a 0.3 – 0.2 annulus of the primary beam (PB) compared to the "dynamic range correction factor" times the theoretical noise (see §9.25 and §9.39.3 for the DR correction factors). Score=1.0 when this ratio is 1 or lower; Score=0.0 when the ratio is 5 or higher, and linear in between.
hif_makeimages (additional QA for polarization calibrators)	Check that the polarization fraction and angle can be calculated	For each session, the score is 1.0 if a Gaussian fit to the I, Q, U images of the polarization calibrator succeeded, else 0.34. The stage score is the minimum score over all sessions.
hif_makeimages (additional QA for check sources)	Determine if phase transfer worked for the check source(s)	For each check source image, the overall score is the Geometric mean of following three scores: Score1=1.0 – abs[(catalog position – fitted position)/beam size] Score2= max[1, (peak flux) / (fitted total flux)] Score3= max[1, (fitted total flux) / (gfluxscale flux)] The stage score is the minimum score over all check source images.

table continued on next page

table continued from previous page

hifa_imageprecheck	For the representative target & spw check that a `robust` parameter between 0 – 2 can meet the PI-requested angular resolution (AR)	0.5 if no representative target / frequency is found. If the PI's desired AR is found in the ASDM, a QA score is assigned as follows: 1.0: PI's AR achieved in both axes with `robust`=0.5. 0.85: PI's AR achieved in both axes with a different `robust`. 0.5: At least one axis is out of range, but the beam area is still within the PI's range. 0.25: not even the beam area can be brought within the PI's desired range by changing `robust`.
hif_checkproductsize	Check whether target cubes need to be made with non-default parameters or fewer targets imaged	0.85 if the products had to be imaged with non-default parameters (larger cells, wider channels, smaller FOV, fewer targets or spw); 0.25 if the images cannot be made without exceeding the set limits for image or product size; else 1.0.
hifa_renorm	Check renormalization scaling spectrum for values above 1.02 (2%)	1.0 if scaling spectrum are all below 1.02 so are not applied. 0.9 if scaling spectrum have values above 1.02 but are not applied and are likely caused by an atmospheric feature, or if scaling spectrum have values above 1.02 and are applied. 0.85 if a user-excluded region of the scaling spectrum has been applied. 0.66 if the scaling spectrum are applied and an atmospheric feature is detected above 1.02. N/A if there are no FDM spectral windows in the dataset (no score is calculated).
hif_mstransform	Check that proper files were created	1.0 when target.ms files successfully created; else 0.0
hif_findcont	Determine if continuum could be identified for all spw	1.0 if continuum frequency ranges found for all spw. If not, the score is the number of spws with ranges divided by the number of expected spws.
hif_uvcontsub	Determine if continuum could be fit and subtracted	1.0 if a continuum fit table was created for all science spws, else 0.0.
hif_makeimages (additional QA for target cubes)	Check for emission in "line-free moment 8" image ("mom8fc", see §9.44.1)	Score based on three measures: PeakSNR=the peak above the median in the mom8fc image compared to the median absolute deviation (MAD) in the cube; HistAsym = the difference between an intensity histogram constructed from the mom8fc with that constructed from a "line-free mom10" image (=minimum value at each pixel in the cube over the findcont channels); and MaxSeg=the largest "segment" of contiguous pixels with values above a threshold in the mom8fc image. If PeakSNR>5 and HistAsym>0.2, or if PeakSNR>3.5 and HistAsym>0.05 and MaxSeg> 1 beam area, then the score is the minimum of 0.65 and the value of an error function (erf) between 0.33 and 1.0 based on the fraction of the image represented by the largest segment (smaller values for larger segments). If that condition is not met, then the score is given by the value of the error function (higher values for a smaller segment).

table continued on next page

table continued from previous page		
hif_selfcal	No QA defined	Always N/A
hifa_exportdata	Check that Pipeline products have been exported	1.0 when all standard pipeline products were successfully copied to the `/products` directory, else 0.0.

8.7.2 Single-Dish Pipeline QA scores

Pipeline Task	QA Scoring Metric	Score
hsd_importdata	Check that the required calibrators are present	1.0 ATMOSPHERE intents are present.
		1.0 one continuous observing session.
		1.0 all source coordinate are present.
		0.5 subtracted for existing processing history.
		0.5 subtracted for existing model data.
hsd_flagdata	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_skycal	Check elevation difference between ON and OFF	1.0 if elevation difference between ON and OFF is <=3°. 0.8 if elevation difference between ON and OFF is >3°.
hsd_k2jycal	Check that all Kelvin-to-Jy conversion factors are provided	1.0 if Kelvin-to-Jy conversion factor present. 0.0 if Kelvin-to-Jy conversion factor is missed for some data.
hsd_applycal	Percentage of incremental flagging	1.0 if additional flagging is 0%-5%. 1.0...0.5 if additional flagging is 5%-50%. 0.0 if additional flagging is >50%. QA score calculation is restricted to scans matching the 'TARGET' intent if present. If no 'TARGET' intent data is present, a warning is raised, and the QA score calculation reverts to using scans for any available intent.
hsd_atmcor	Check that ATM correction is applied	1.0 if ATM correction is applied. N/A if ATM correction is not applied. 0 if there is an error in the application of the correction.
hsd_baseline	Check that one or more than one emission line is detected by line-finder	1.0 if there is more than one emission line detected in at least one spw. 0.0 if no line is detected in all spw.
	Evaluate the baseline flatness	QA score is calculated by the following criterion. • 0.33 if MAX(mean)-MIN(mean)>3.6σ • 0.33–1.0 if 1.8σ <MAX(mean)-MIN(mean)≤3.6σ • 1.0 if MAX(mean)-MIN(mean)<1.8σ Emission-free channels are divided into 10 or 20 bins. "mean" is the mean in each bin. MAX(mean) and MIN(mean) are the maximum and minimum of "mean" among bins.

table continued on next page

table continued from previous page		
hsd_blflag	Percentage of incremental flagging per source per spw	1.0 if additional flagging is 0%-5%. 1.0...0.5 if additional flagging is 5%-50%. 0.0 if additional flagging is >50%.
hsd_imaging	Determine if observed area is not masked	1.0 if no pixel masked. 0.5 if any of the pixels in the pointing area masked. 0.0 if 10% of the pixels in the pointing area masked. Score is linearly interpolated between 0 and 0.5 for fractions between HIGH% and LOW%
hsd_exportdata	Check that Pipeline products have been exported	1.0 when files successfully exported.

9 Interferometric pipeline tasks and "By Task" WebLog pages

This section describes each Interferometric Pipeline task and its associated task WebLog page. For a detailed description of task inputs and parameters, refer to the ALMA Pipeline Reference Manual.

Note that stages that don't perform any actions are in grey font in the "By Task" WebLog navigation sidebar, such as hifa_wvrgcalflag for 7m antenna data which have no Water Vapor Radiometers (WVRs), or hif_makeimages (rep BW cube) when no representative/user bandwidth cube is required.

9.1 hifa_importdata

In this task, ASDMs are imported into MeasurementSets, 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. If the flux intent is not shown in this table, then most likely the flux calibrator was a solar system object, which can be seen on the hif_setmodels page. Flux densities are first 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 Technical Handbook). The online ALMA calibrator flux service is queried, and if that succeeds in returning flux density values, the values read from the Source table are replaced with the query values. This allows observatory measurements performed after the science observation to be used to obtain a better estimate of the calibrator flux density. 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`.

If a POLARIZATION intent is present in the dataset, the parallactic angle coverage of each polarization session is shown graphically, and reported quantitatively. If the polarization recipe (`hifa_polcal`) is not specified in the context in the casa_pipescript.py file, the QA score will be set to 0.5 with a message about unexpected polarization calibrations in the MS file, which can be safely ignored if this was intended. If the recipe is used without specifying that explicitly in the context, the polarization calibrator is calibrated and imaged in hifa_makeimages, but not exported by hifa_exportdata.

9.2 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, partial polarizations, and TDM edge channels). The first table on the WebLog page shows whether any data in these categories were flagged (a check mark in the first table means yes, an X means no). The Flagged data summary table shows the percentage of flagged data per MS. The "Before Task" column contains only the effect of the Binary Data Flags (BDF) from the correlator 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 partial polarization flagging agent identifies all visibilities where 1..N-1 polarization products were flagged by the BDF flags, where N=number of polarization products in the science spectral windows. Because these visibilities are assessed prior to applying the other flagging agents, the percentage can appear as 0.000% even when there is a check mark in the first table. Note that the ALMA BDF and online flags do not attempt to perform any channel-based flagging. The QA score for this stage is based on BDF+QA0+online+template+shadow flagging.

New in PL2022 is the "Low Transmission" flagging agent which will flag spws whose transmission across 60% of their bandwidth is less than 10% on non-representative spws and less than 5% on the representative spw. This heuristic can be disabled with the boolean value `lowtrans` in the PPR. Alternatively, the 10% threshold can be adjusted via the parameter `mintransnonrepspws` and the 5% threshold can be adjusted by the parameter `mintransrepspw`. The QA subscore will be reduced to 0.9 if a non-representative spw is flagged and 0.33 if the representative spw is flagged. This feature uses the following fixed meteorological values to avoid being susceptible to faulty weather station values: pressure=563 mb, altitude=5059 m, temperature=273 K, maxAltitude=48 km,

humidity=20%, (water vapor scale height) h0=1.0 km, (initial pressure step) dP=5.0 mb, and (multiplicative factor of successive pressure steps) dPm=1.1. The PWV is taken from the actual median across the EB, while the airmass is the mean of the first and final integration of the scan in question. **Note:** Because the CASA task plotbandpass uses h0=2 km, which is arguably less accurate on most days at the ALMA site than the smaller value employed by the pipeline, there may be edge cases where the transmission curve may look sufficiently lower than the threshold to trigger flagging in the plots vs. frequency in hifa_tsysflag and hifa_bandpass, while the flagging is (correctly) not performed. For example, near the 325 GHz water line at PWV=0.62 mm and airmass=1.3, the model difference in h0 amounts to a net difference of 4%, meaning that h0=1 km will predict 16% transmission while h0=2 km will predict 12% transmission.

9.3 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 more than 75% of a given spw is flagged on the flux calibrator for this reason, then a reference spwmap (`refspwmap`) is calculated and stored in the pipeline context so that later in hifa_gfluxscale, that parameter of the fluxscale task will be set in order to transfer the flux scale from the nearest other spw. The WebLog shows if any flagging or spwmap was required. In Mars, Venus, Titan, and Neptune, ^{12}CO is flagged in all ALMA bands. In Mars, Venus, and Titan, ^{13}CO is also flagged. In Titan, HCN, H^{13}CN, and HC^{15}N are also flagged, as is HCN v_2=1. Finally, in Titan, CH_3CN is flagged up through Band 8. The frequency width that is flagged is based on published spectra of 1 or 2 transitions of the species. For other transitions, this frequency width is scaled to maintain consistent velocity widths. Also, because the flags are applied in topocentric frame, the final width that is flagged is further broadened by the maximum relative velocity between the object and the geocenter (computed over a decade). A detailed list of topocentric frequency ranges flagged is given in the following table:

Object	Species:	Topocentric frequency ranges flagged (GHz)
Mars	CO:	[115.204,115.338], [230.404,230.672], [345.595,345.997], [460.773,461.309], [691.071,691.875], [806.184,807.120], [921.265,922.335]
	^{13}CO:	[110.190,110.212], [220.377,220.421], [330.555,330.621], [440.721,440.809], [661.001,661.133], [771.108,771.260], [881.196,881.370]
Venus	CO:	[115.206,115.337], [230.407,230.669], [345.600,345.992], [460.779,461.303], [691.081,691.865], [806.194,807.110], [921.277,922.323]
	^{13}CO:	[110.192,110.210], [220.380,220.418], [330.560,330.616], [440.727,440.803], [661.011,661.123], [771.118,771.250], [881.208,881.358]
Titan	CO:	[114.92,115.67], [229.49,231.74], [343.82,347.62], [458.29,463.80], [687.75,694.66], [803.46,809.85]
	^{13}CO:	[110.18,110.22], [220.28,220.52], [330.36,330.82], [440.42,441.15], [660.60,661.53], [770.65,771.72], [880.73,881.82]
	HCN:	[88.45,88.81], [176.73,177.80], [264.96,266.81], [353.29,355.72], [441.74,444.52], [618.45,622.16], [707.01,710.74], [795.56,799.31], [883.82,887.86]
	HC^{15}N:	[86.04,86.07], [172.05,172.16], [258.04,258.27], [430.02,430.45], [601.95,602.60], [773.88,774.64], [859.84,860.62]
	H^{13}CN:	[86.33,86.35], [172.63,172.73], [258.91,259.11], [431.44,431.88], [603.98,604.56], [776.48,777.16], [862.72,863.42]
	HCN v2-1:	[177.192,177.286], [178.088,178.184], [265.782,265.924], [267.128,267.270], [354.365,354.555], [356.161,356.351], [442.942,443.178], [445.184,445.422], [620.058,620.390], [623.197,623.529], [708.596,708.974], [712.182,712.562], [797.117,797.543], [801.149,801.577], [885.619,886.091], [890.096,890.572]
	CH_3CN:	[91.938,92.008], [110.304,110.409], [128.659,128.809], [147.039,147.209], [165.415,165.608], [183.792,184.006], [202.168,202.403], [220.543,220.798], [238.916,239.194], [257.289,257.587], [275.660,275.980], [294.030,294.371],

table continued on next page

table continued from previous page		
		[312.399,312.761], [330.766,331.149], [349.205,349.534], [367.572,367.920], [385.938,386.303], [404.301,404.683], [422.735,423.062], [441.098,441.440], [459.459,459.814], [477.881,478.186], [496.239,496.557]
Neptune	CO:	[113.99,116.51], [226.98,234.52], [339.97,351.54], [454.95,467.55], [685.93,696.57], [802.92,810.58]

9.4 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).

   *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).

   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. As of PL2023, if channels to be flagged coincide with strong ozone lines, then these channels are removed from the list to be flagged.

9.5 hif_refant

An ordered list of preferred reference antennas is calculated, with preference given to antennas closest to the center of the array and those with a low flagging fraction through the following metric (M):

$M = n_{ant}\big([1 - (\texttt{normalized\_distance\_from\_center})] + \texttt{normalized\_fraction\_of\_unflagged\_data}\big)$

The center of the array is defined by the median values of the lists of antenna latitudes and longitudes. The WebLog page shows that ordered list of antennas, and the metric for each antenna can be found in the casa log for this stage. A single refant can be selected manually in the PPR (but it will be applied to all EBs of the MOUS).

9.6 h_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 h_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. Mapping is often necessary because so far, Tsys can only be measured in TDM windows on the 64-station baseline correlator.

9.7 hifa_tsysflag

This task flags the Tsys cal table created by the h_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" in 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. If there is a Tsys flag template file included during the pipeline run these 'manually' added flags will also be applied.

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 "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).

   Individual sources are evaluated separately with the default setting of `fnm_byfield=True`; this is to prevent elevation differences between targets from causing unnecessary flags (mostly affects high frequencies).

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

2. **Metric "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 "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 outermost 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 "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 a percentage 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: 13).

5. **Metric "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: 0.15).

   (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 "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.

9.8 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 data are corrected via

Initial Assessment	Secondary Scoring	Metric Score	Colour	Warning	Analyst Action
Phase RMS improvement ratio **> 1**	No other issues	Fix to 1.0	**GREEN**	None	None
Phase RMS improvement ratio **> 1**	**Issues** - Flagged antenna, poor 'disc' or 'rms' in wvrgcal, low SNR phase cal - **set score 0.9, and deduct**: flagged antenna: -0.1 poor 'rms' wvrgcal: -0.1 poor 'disc' wvrgcal: -0.1	Linear Fit score between 0.67 - 0.9	**BLUE**	About Issues	None*
Phase RMS improvement ratio **> 1**	**Issues** - Flagged antenna, poor 'disc' or 'rms' in wvrgcal, low SNR phase cal and Rudimentary assessment shows Bandpass phase RMS >85deg **- set score 0.66 and deduct:** flagged antenna: -0.1 poor 'rms' wvrgcal: -0.1 poor 'disc' wvrgcal: -0.1	Linear Fit score between 0.34 - 0.66	**YELLOW**	**Possible high phase RMS** data and issues	These data could be in unsuitable conditions - phase RMS decoherence plots will look noisy - see spwphaseup stage to examine
Phase RMS improvement ratio **< 1**	**Issues** - Flagged antenna **- score will be reduced by:** flagged antennas: -0.1 **Check the Phase RMS** on the Bandpass scan and Phase Calibrator scans are <1 radian	Linear Fit score between 0.67 - 0.9	**BLUE**	State that Phase RMS is good, even though no improvement, and about any issues	Confirm the plot "Phase correction with/without WVR" shows the deviation from the median phase is generally within +/-50 deg
Phase RMS improvement ratio **< 1**	**Issues** - Flagged antenna, poor 'disc' or 'rms' in wvrgcal - **set score 0.66, and deduct**: flagged antenna: -0.1 poor 'rms' wvrgcal: -0.1 poor 'disc' wvrgcal: -0.1	Linear Fit score between 0.34 - 0.66	**YELLOW**	No WVR improvement and issues	Visual inspection to see why there was no improvement: • issues with sporadically poor 'disc' and 'rms' possibly point to high PWV conditions and poor solutions due to clouds - possible remcloud pre-cursor (see below) -plots will show large phase spread
Phase RMS improvement ratio **< 1**	**Issues** - median of all disc' or 'rms' values in wvrgcal are poor - **set score 0.33, and deduct**: flagged antenna: -0.1 poor 'rms' wvrgcal: -0.1 poor 'disc' wvrgcal: -0.1	Linear Fit score between 0.0 - 0.33	**RED**	No WVR improvement, possibly Remcloud needed, phase RMs to be checked, plus issues	Need visual inspect to see why there was no improvement. As median of all 'rms' and 'disc' returned by wvrgcal are poor, this is the clear precursor for clouds - i.e. remcloud needed trigger. Phases could also be poor - also see spwphaseup stage

Figure 21: QA scoring workflow for the hifa_wvrgcalflag task indicating the initial and secondary scoring criteria, the resulting QA metric score range and corresponding color according to the table at the begining of §8.7. The "Warning" column summarizes the meaning of the score, and the final column describes the instructions given to the QA analysts reviewing the WebLog.

a calibration table.

9.9 hifa_wvrgcalflag

Water Vapor Radiometer (WVR) sky brightness temperature measurements in four subbands surrounding the 183 GHz water line are converted by the CASA task wvrgcal 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 if 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 by `wvrgcal`, as long as there are at least `minnumant` (default=2) within `maxdistm` (default=500 m) of the WVR-flagged antenna), and also prints a warning if the correction is deemed not helpful enough to apply at all. Note that for datasets with heterogeneous antenna diameters, there must be at least `ants_with_wvr_nr_thresh` (default=3) antennas with WVRs (i.e. 12 m antennas) **and** at least `ants_with_wvr_thresh` (default=0.2) fraction of antennas with WVRs (e.g. 3 out 15, such as 3 PM with 12 CM), otherwise this stage will not attempt any correction.

The per-antenna values of path length rms ("RMS") and channel-to-channel discrepancy ("Disc") are provided in the WebLog table. Further information on the meaning of these columns can be found in the documentation of wvrgcal at:
`https://casadocs.readthedocs.io/en/v6.5.3/api/tt/almatasks.wvrgcal.html`

In hifa_wvrgcalflag, a QA score is produced for each MeasurementSet of an OUS. The QA score is produced after a two stage metric process. First, the RMS improvement ratio is assessed for the "BANDPASS" and "PHASE"

calibrator sources. The CASA log attached to the WebLog lists the ratios of with-wvr phase RMS / wihout-wvr phase RMS (i.e. 1/improvement ratio). In the second stage any issues with the data - including flagged antennas in the solutions, values of the "RMS" and "Disc" exceeding 0.5 mm, or whether the "BANDPASS" or "PHASE" calibrator data have low SNR or point to high atmospheric phase variation are used to cap the QA metric score before further reducing the score for each issue found. If a low SNR issue was found for the "PHASE" calibrator data, when the phase RMS is >90 deg on the longer baselines, it is excluded from the scoring assessment and only the improvement ratio for the "BANDPASS" intent is used. Depending on the scoring tree as shown in Figure 21 the final QA score is set by fixed range linear fits. Corresponding notifications and warnings are also shown at the top of the hifa_wvrgcalflag page. The final stage score is the lowest score from all MeasurementSets.

9.10 hif_lowgainflag

Antennas with persistently discrepant 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 each 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:

1. Their value is smaller than a threshold-factor * median of all non-flagged data points, where the threshold is `fnm_lo_limit` (default: 0.5), or
2. Their value is larger than a threshold-factor * median of all non-flagged data points, where the threshold is `fnm_hi_limit` (default: 1.5).

Flagging commands are generated for each of the identified outlier data points. If any antennas have a significant fraction of data flagged, the reference antenna ranked list will be reordered, moving the flagged antennas to the end and the new order will be displayed. A "Warning!" notification will appear at the top of the page listing which antennas were moved to the end of the list.

9.11 hif_setmodels

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 §9.1). 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). If the bandpass calibrator is distinct from the amplitude calibrator and is a frequently monitored quasar, its model is also set at this stage.

9.12 hifa_bandpassflag

After calculating an initial phaseup solution and bandpass solution and applying it, flagging is performed on outlier visibilities by comparing the scalar difference between the amplitudes of the calibrated visibilities and the model for the bandpass calibrator (Figure 22). If flags are found, a second iteration is performed. At the end, the flagging state at the beginning of the task is restored, and all of the flags found by this stage are applied.

Note about flagging summary table: the “before” flagging fraction in hifa_bandpassflag may differ from the “after” flagging fraction in hifa_flagdata, because hifa_bandpassflag’s “before” summary is done on an MS that has temporarily already had some caltables applied (and thus some flagging already propagated). This is done

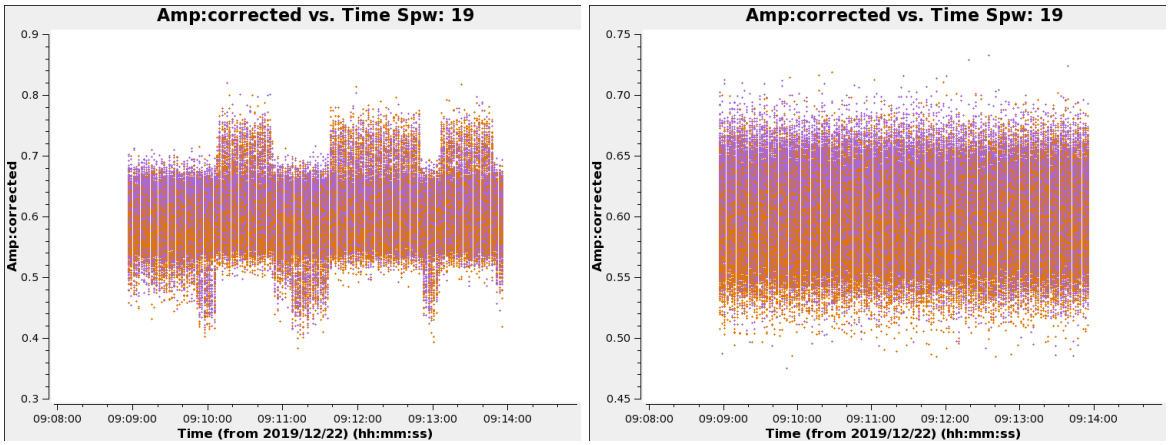


Figure 22: Example of hifa_bandpassflag removing visibilities that show outlier amplitudes – before flagging (left panel) and after flagging (right panel).

because the before/after summary in hifa_bandpassflag is intended to clearly show how much new flagging is done by hifa_bandpassflag.

9.13 hifa_bandpass

In this task, the bandpass calibrator is self-calibrated (phase only solutions are first obtained on as short a time interval as allowed by signal-to-noise, listed on the WebLog page). The antenna-based bandpass phase and amplitude solutions are then calculated using a S/N-dependent frequency interval, also listed on the WebLog page.

The top-level WebLog page for this stage includes plots of the amplitude vs. frequency and phase vs. frequency bandpass tables for all spws for both the reference antenna and for a typical antenna. The atmospheric transmission curve is overlaid in these plots. There are also links to a sub-page that shows the corresponding plots of the bandpass solutions on a per-ms/spw/antenna basis, also with the atmospheric transmission curve overlaid. There are fields to filter these plots by ms/spw/antenna, and at the top is a histogram of the QA values for each table based on the metrics described in the §8.7.1 entry for this task: the signal-to-noise metric for the amplitude vs. frequency plots, and the phase derivative deviation metric for the phase vs. frequency plots.

As of PL2023, the `fillgaps` parameter of the bandpass task is exposed for use in a PPR, allowing the user to interpolate across a spectral feature in the bandpass calibrator spectrum.

9.14 hifa_spwphaseup

In this stage, there are three major activities. First, 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, but is tested later in hifa_timegaincal as described below.) Second, the temporal gain calibration strategy is decided for each phase calibrator and check source independently. The best performance will be obtained when each spw has sufficient SNR to calibrate itself, even if the spw offsets appear to be stable in time. Only when one or more spws has insufficient SNR does the pipeline invoke the more complicated "low-SNR" heuristics, as described below. Third, an assessment is made of the phase RMS as a function of projected baseline length using the high-SNR observation of the bandpass calibrator. Such an assessment is useful as a proxy of atmospheric phase stability conditions and is used to make an assessment of possible phase decoherence that may exist in the target data after calibration (note that the difference in elevation between the bandpass calibrator and target are not taken into account, i.e. typically lower elevation sources will have a more unstable, worse, phase RMS).

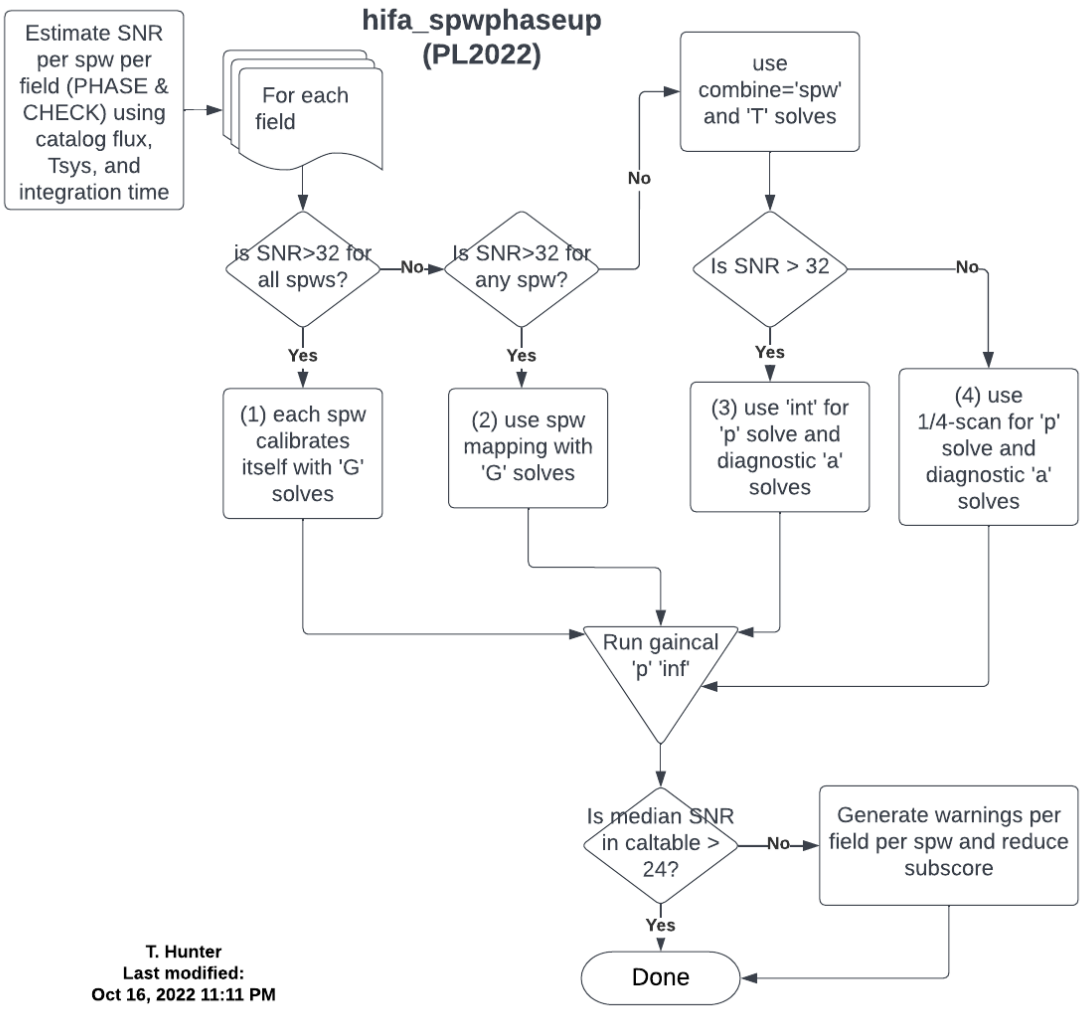


Figure 23: Logic flowchart of hifa_spwphaseup for determining the temporal gain strategy.

9.14.1 Determination of expected SNR

To achieve these goals, we use the flux density estimates from the ALMA calibrator database query (performed earlier in the pipeline and loaded to the pipeline context). The expected SNR is computed for each spectral window and each phase calibrator and check source independently (earlier pipeline versions used only the first phase calibrator). For each field, the SNR is computed for a length of time matching the scan length of the scan closest to the first Tsys scan used for that field. As of PL2023, the integration time is adjusted for the percentage of flagging (per spw). The bandwidth used for the spw accounts for the maximum effective bandwidth of 1875 MHz for TDM spws. A pre-computed list of per-Band sensitivities for a nominal Tsys value with 16 $\times$ 12 m antennas, 8 GHz bandwidth, and 1 minute integration with dual polarization is consulted and scaled for the actual median Tsys, actual total collecting area of antennas, actual bandwidth, and single polarization. Using these values, the logic in Figure 23 is followed.

9.14.2 Choosing the spw mapping strategy

If each spw of the field in question has sufficient SNR to calibrate itself, then that simple direct calibration path is chosen for that field. The SNR must be >32 on a per-scan basis, thereby guaranteeing SNR>5 on a per-integration basis considering that there can be up to 30 integrations per scan ($30 \times 2.016 = 60.48$ s) and allowing for a 20% over-estimate in the catalog flux density of the calibrator. If any spw's SNR is below 32, then if the spw with the highest S/N has sufficient S/N to calibrate itself, then all spws that have insufficient SNR are mapped to this spw so that the calculated phase calibration as a function of time can be subsequently transferred (during subsequent gaincal and applycal tasks) from this higher SNR spw to the those that need it. If any such reference spwmaps are required, then they are listed in the table on the WebLog page. If the S/N is sufficiently poor on all spws, then all spws are combined for the subsequent calculation of time-varying gains, and the estimated combined SNR for that field is shown in the table on the WebLog.

9.14.3 Gain calibration

Next, for each field, a gain calibration is performed using its chosen strategy. If the SNR is less than 75% of the SNR threshold of 32, then a warning is generated and the QA subscore is reduced. For check sources, the QA score is 0.9 for 16<SNR≤24, 0.8 for 9.6<SNR≤16, or 0.7 for SNR<9.6. For phase calibrators, the score is 0.9 for 16<SNR≤24, 0.66 for 9.6<SNR≤16, and 0.33 for SNR<9.6. **Note that these warnings can occur even if spw combination has not been deemed necessary by the initial estimate. This can happen when the actual flux density of the field is significantly below the ALMA catalog estimate.** In this scenario, the pipeline can be re-run by setting the `phasesnr` parameter sufficiently higher than 32 in order to trigger combination for the field in question.

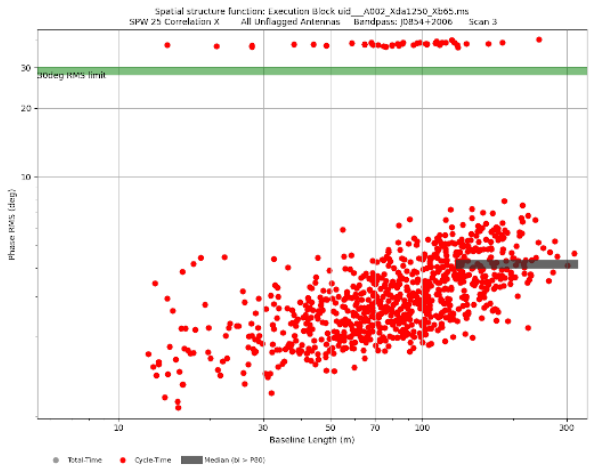
Regardless of the usage of spw mapping or combine, the assumption of constant phase offsets vs. time is tested downstream in hifa_timegaincal by solving for new time-based phase solutions per spw with the spwphaseup table (and associated mapping) applied.

9.14.4 Assessment of phase decoherence

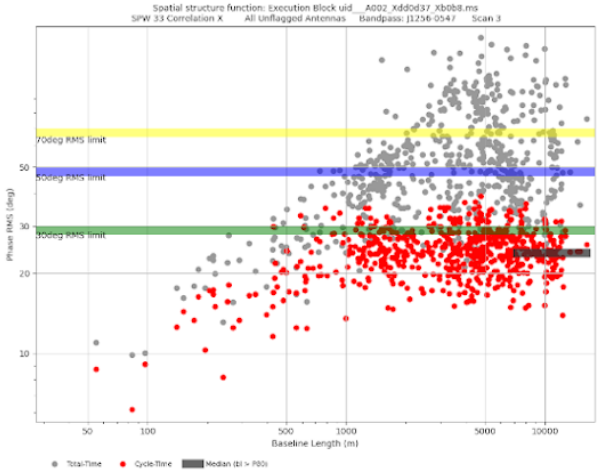
Finally the phase decoherence assessment is made. Using the high SNR sort solint phase(time) solutions (self-calibration from the hifa_bandpass stage) of the bandpass calibrator from the widest spw, which are per antenna with respect to the reference antenna, the baseline based phases can be reconstructed. For each baseline the phase RMS is calculated (data set to zero mean) using the entire phase(time) stream of the bandpass observation (total-time). The plots in Figure 24 show the phase RMS as a function of baseline length for the total-time as the gray points. These are the longest timescale phase RMS assessment possible within these data and are closely related to so-called Spatial Structure Function (SSF) plots that provide a means to fundamentally investigate the variability of the turbulent atmosphere and typically indicate that longer baselines (regardless of array configuration) will have higher phase RMS over >5-10 minutes. (see ALMA Memo 592 and references therein for more detail about the atmospheric structure above ALMA).

More importantly, for interferometric observations the technique of phase referencing is used for calibration with a repeat interval known as the phase referencing cycle time. The action of phase calibration (see hifa_timegaincal), when solutions are interpolated to a target (see hif_applycal), acts only to correct timescales longer than the phase referencing cycle time, but cannot act to calibrate any short term phase variations which are not sampled (only self-calibration can provide such a correction). (see ALMA Memo 582 and references therein for more detail about phase referencing and WVR correction). The assessment of the phase RMS as a function of baseline length for a time interval equal to the phase referencing cycle time provides a proxy of the phase variably that can remain un-corrected in the target source visibilities. In Figure 24 the red points indicate the phase RMS as a function of baseline length when calculated over a time period equal to the phase referencing cycle time.

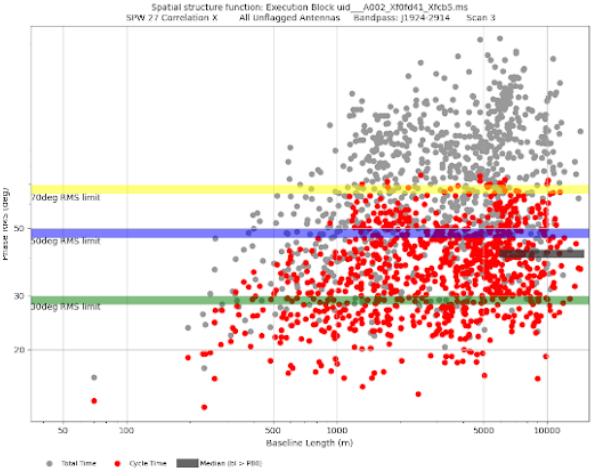
Values of the phase RMS are then extracted from all baselines longer than the 80^{th} percentile and the median phase RMS value is reported in the table in the WebLog, and as indicated by the semi-transparent black bar in Figure 24. Note, in some cases for short baseline observations the phase referencing cycle time can be 5 minutes or longer, typically longer than the bandpass calibrator scan used for the total-time, and therefore the cycle-



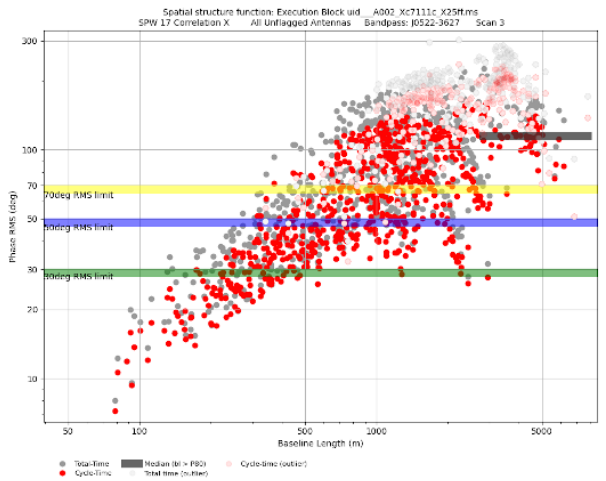
A) Excellent stability - One outlier antenna - SCORE = 0.9



B) Long Baseline data - Short cycle-time. Excellent stability - SCORE = 1.0



C) Long Baseline data - Short cycle-time. Good Stability - SCORE = 0.9



D) Mid Baseline data - Cycle-time < total-time - Poor Stability - SCORE = 0.0

Figure 24: Example phase decoherence plots, phase RMS vs. Baseline length. Example A shows a short baseline dataset where the cycle-time values are set to those of the total-time and the phase RMS is indicating excellent stability. The score would be 0.9 for these data as there is an outlier antenna where all baselines to that antenna show phase RMS values constant and at ∼40°. Example B and C are both long baseline data where the cycle-time are of the order 80 seconds such that the phase RMS is notably lower than that measured over the total-time of the bandpass scan (5 minutes in these cases). Example B has excellent stability (median ∼25°) where baselines after ∼2 km will all have a phase RMS ∼30° over the cycle time, while example C is a case of good stability with a median phase RMS of ∼45°. Example D is a mid-baseline length (max baseline ∼5 km) and has poor phase stability (∼80°). High phase RMS outlier antennas were also detected and excluded (semi-transparent points) in a re-assessment loop, however the phase RMS is still poor and the data will suffer from high decoherenece.

time phase RMS value will be set equal to the total-time phase RMS value. From a phase-metrics standpoint short baselines are insensitive to large atmospheric turbulent cells that usually cause very large, long-timescale phase changes over >5-10 minutes, and only see the smaller turbulent cells with shorter timescale <1-2 minute variations and hence for timescales longer than a few minutes the phase RMS will remain unchanged. For long baselines configurations (maximal baselines >2-3 km) the phase referencing cycle times are shorter than a typical bandpass calibrator scan and thus the phase RMS assessed over the shorter timescale will be reduced compared to the total-time value.

Outlier antennas are also identified if the phase RMS exceeds 180°, pointing to pure phase noise; if the median phase RMS is >50° a limit of 4×MAD + median phase RMS is used; while if the median phase RMS is <50° the maximum value of either 6×MAD + median phase RMS or 2× median phase RMS is used. In the case of data with phase RMS >50° a re-assessment of the phase RMS is made excluding the outliers to examine if they were skewing the median phase RMS value. The outliers values are plotted as semi-transparent symbols (Figure 24 Example D). Any outlier antennas are identified in the WebLog table and also listed in the log file and could point to problematic antennas to be investigated in hifa_timegaincal.

9.14.5 QA scoring

A QA score assessment is made based on the median phase RMS value. Phase RMS <30° is indicative of excellent phase stability and has a QA score of 1.0, but can reduce to 0.9 if there are outlier antennas. Phase RMS between 30-50° is indicative of good stability and has a score of 0.9, between 50-70° the phase RMS is notably elevated and the score is set relatively between 0.5 to 0.3, while for very poor phase stability the phase RMS is >70° and the score is set relative from 0.3 down to 0.0. Excessive phase RMS can cause signal decoherence and even after phase calibration the target source can have a reduced flux caused by the phase RMS that acts to scatter flux around an image rather than to focus it on the correct location of the source. Phase RMS levels of 30, 50, 70° can lead to decoherence at the levels of ∼13, 32 and 53 %, these limits are indicated by semi-transparent green, blue and yellow horizontal lines in the plots as shown in Figure 24. If possible phase self-calibration of the target source is advisable to help improve the short term phase calibration, especially for data with higher phase RMS.

Overall, the lowest score is presented for the stage as a whole, whereas individual notifications or warnings are issued for the action of spw mapping, combination or poor phase RMS.

9.15 hifa_gfluxscaleflag

As with hifa_bandpassflag, a temporary calibration is performed and applied, then outliers in the the calibrated visibilities of the flux, phase, and check source calibrators are flagged. The WebLog shows the before and after flagging plots as a function of time and of uv distance. The heuristics for multi-scan calibrators (typically the phase calibrator and sometimes the check source) differ slightly from single-scan calibrators.

9.16 hifa_polcalflag (polarization recipes only)

As with hifa_bandpassflag and hifa_gfluxscaleflag, a temporary calibration is performed and applied, then outliers in the calibrated visibilities of the polarization calibrators are flagged. The WebLog shows the before and after flagging plots as a function of time and of uv distance. The heuristics for multi-scan calibrators (of which the polarization calibrator always is) differ slightly from single-scan calibrators.

9.17 hifa_session_refant (polarization recipes only)

A single reference antenna is selected for each entire session (for ALMA polarization observations, this is often 2-3 EBs). To choose the best antenna, the antennas are first ranked by the product of their per-EB ranking (which is based on flagging fraction and central location in the array, as in hif_refant). The task next performs a gaincal 'int' 'p' 'G' (`minsnr`=3) on all PHASE intent scans for each EB starting with the highest ranking antenna as the sole refant, and checks the resulting caltable to see if the refant ever changes. It chooses the first antenna

that does not result in any change of refant. If none of the top 3 antennas qualify (which should be rare), then the antenna with the most solutions as refant is chosen, and a message is displayed with the number of phase solution outliers, where the word "outlier" is where the refant phase was non-zero, which indicates that another refant was chosen for some integrations. The total number of possible solutions is: $N_{EBs} * N_{spws} * N_{integrations} * N_{pol}$. If a single refant was requested the hif_refant stage in the PPR, then a warning is generated: "Measurement sets for session "session_1" have only one reference antennas in common (DV13), which will be set as the final best reference antenna."

9.18 hifa_lock_refant (polarization recipes only)

Sets the refant to a single antenna and refantmode="fixed" for all subsequent calibration tasks. This can be "unlocked" with the `hifa_unlock_refant` task, but that is not needed in the standard polcal and polcalimage recipes. Note that in the polcal and polcalimage recipes, hifa_bandpass and hifa_spwphaseup are called a second time following the locking of the reference antenna, to ensure that the bandpass and spw-offsets cal tables are using that fixed reference antenna.

9.19 hifa_gfluxscale

In this task, the absolute flux scale is transferred from the amplitude calibrator to the other calibrators and ultimately to the science target (via the phase calibrator). A phase-only self-calibration is performed on all calibrators prior to this flux calculation. The phase solution uses gaintype='G' (per-polarization solutions) but the subsequent amplitude solve (in which the phase solution is pre-applied) uses gaintype='T' (polarizations combined). This method effectively calibrates to the Stokes I flux densities measured by the ALMA calsurvey team, and allows polarized calibrators to have differing flux densities in their calibrated XX and YY visibilities, and thereby avoids introducing any false polarization into the science targets.

This task will fail to find flux densities if the same field shares calibration intent between these groups: (BANDPASS, FLUX, PHASE, POLARIZATION, CHECK_SOURCE), with the exception of BANDPASS==FLUX.

In the high SNR regime, the flow chart of gaincal commands is shown in Figure 25.

If low SNR heuristics were activated in hifa_spwphaseup, then they are used in this task. For example, for phase calibrators and check sources, if spw mapping was triggered for a field due to low SNR in hifa_spwphaseup, then that map will be applied in this task, as shown in Figure 26.

In general, the solutions are performed on the integration timescale, computing the phase solution for that field in this stage. However, if a field triggered spw combination in hifa_spwphaseup, then in addition to using spw combination, the solution interval for the phase solution will be increased to 1/4 of the scan length for that field and 'T' solutions (i.e. polarization-averaged) will be created instead of the default of 'G' (per-polarization), as shown in Figure 27.

The phase solutions for the checksources are shown in hifa_timegaincal in the plot of Diagnostic Phase vs. time.

The WebLog for this stage lists the derived flux scale factors, which is what gets written to the model column via setjy. It also lists the calibrated flux densities (measured by the vector averaged calibrated visibility amplitude) of the non-amplitude calibrators (usually phase and bandpass calibrators), along with the flux values extracted from the ALMA Source Catalog. Plots of amplitude as a function of uv distance are shown, and if the absolute flux calibrator is resolved (i.e., it has decreasing flux with increasing uv distance, which is usually only the case for solar system objects), only data from antennas with sufficiently short baselines are used to calculate the flux densities of the secondary calibrators. The exact steps followed are:

1. Estimate the solar system object calibrator size.
2. Determine the longest observing wavelength across the science spws.
3. Determine the shortest unprojected baseline to the reference antenna.
4. Estimate the peak intensity of the visibility pattern at that baseline.
5. Find the baseline length the where transform of a uniform disk drops to 20% of that peak.

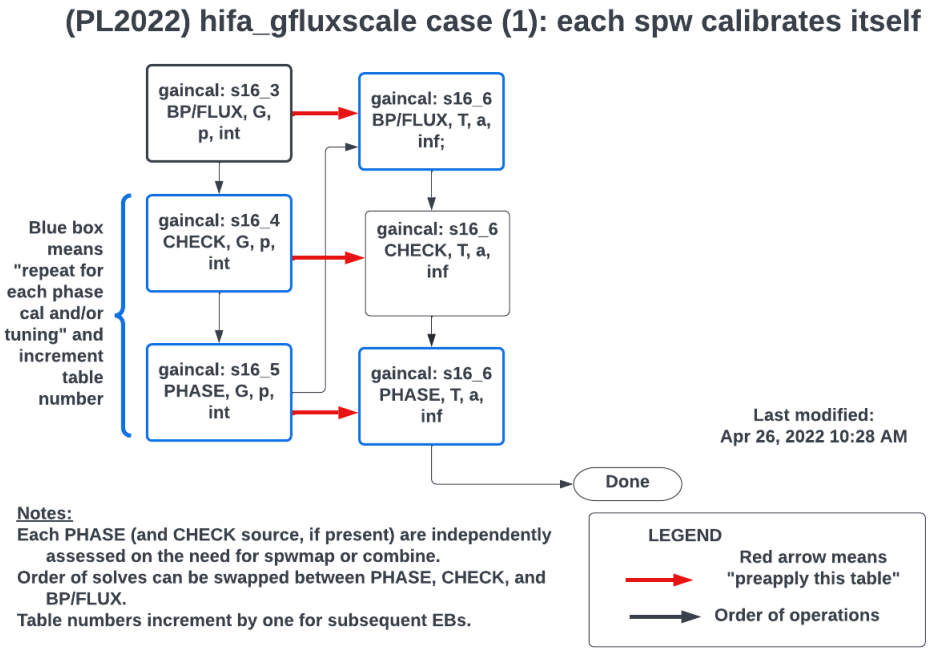


Figure 25: Logical flow in hifa_gfluxscale for the case of high SNR in all spws.

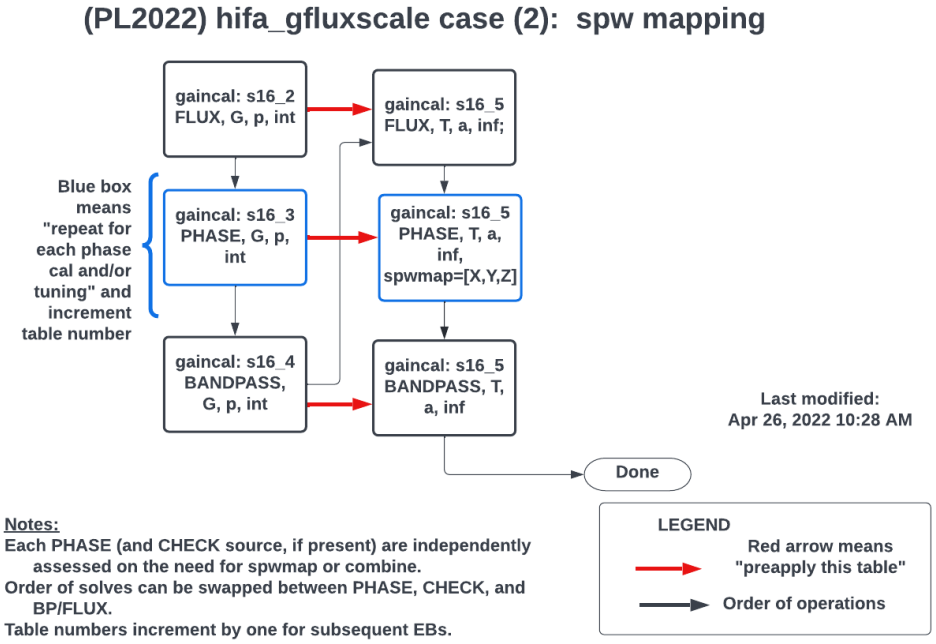


Figure 26: Logical flow in hifa_gfluxscale for the case of high SNR in at least one spw and low SNR in at least one spw.

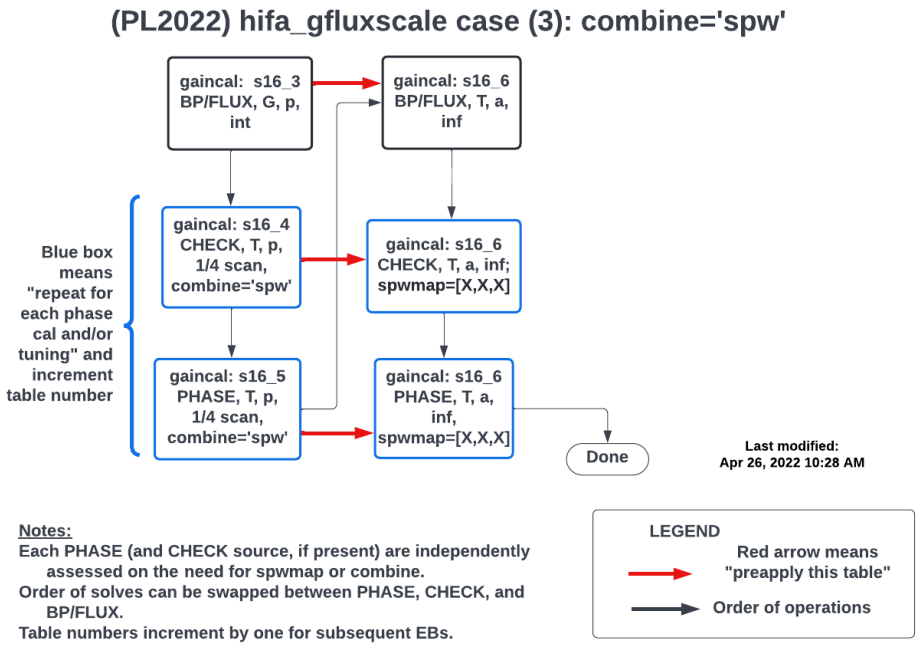


Figure 27: Logical flow in hifa_gfluxscale for the case of low SNR in all spws.

6. Select antennas whose separation from the refant are all within that length.
7. If the number of qualifying antennas id < 3, then default back to selecting all antennas.

The antennas used are listed in the table at the top of the WebLog page (Figure 28). Blank entries mean that all antennas were used, which will be the case for quasars.

Derived flux density vs catalog flux density plots are shown for non-amplitude calibrators (see Figure 29). Faint sources can have systematically elevated hifa_gfluxscale derived flux densities in low SNR spws. The pipeline also calculates a QA metric to detect such spw-spw discrepancies (see QA Table in §8.7.1), and gives an informational statement (blue QA) if the spw-spw variation is between 10–20%, and a warning (yellow QA) for larger variations.

Antennas Used for Flux Scaling

The following antennas were used for flux scaling, entries for unresolved flux calibrators are blank

Measurement Set		UV Range	Antennas
uid___A002_Xa25bbf_X6823.ms	Titan	<319.210m	
uid___A002_Xa216e2_X1e58.ms	Titan	<318.242m	
uid___A002_X9ec9e7_X1804.ms	Titan	<322.242m	
uid___A002_X88063e_X1f1.ms	QSO		
uid___A002_X87f18c_Xd0.ms	Mars	<32.306m	CM01, CM02, CM10, CM05, CM12, CM09, CM11, CM03

Figure 28: An example of limited uv ranges used for deriving the fluxscale table.

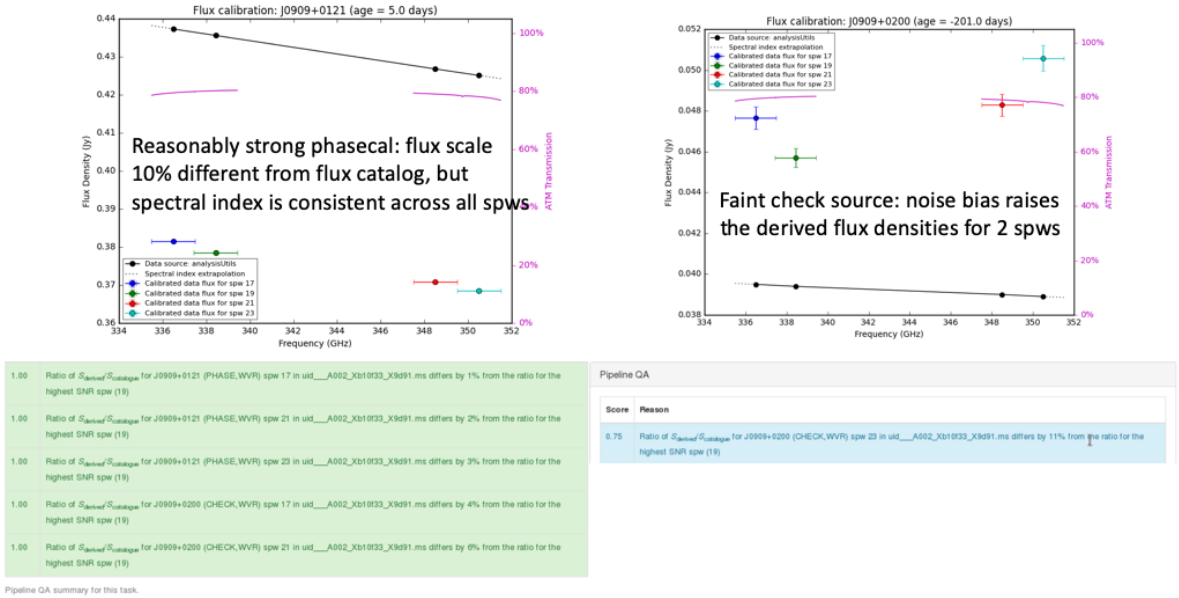


Figure 29: Examples of the plot comparing derived flux densities to source catalog flux densities, and associated QA messages at the top of the hifa_gfluxscale web page.

9.20 hifa_timegaincal

In this task, gain as a function of time is calculated from observations of the bandpass, flux, polarization, and phase calibrator(s). The flowcharts in Figures 30, 31, and 32 show the gaincal calls for the case of high-SNR in all spws, high-SNR in at least one spw, and low-SNR. In all cases, solutions are computed on a scan basis (`solint` = 'inf') is produced. The phase is also solved on an 'int' timescale and pre-applied for the amplitude scan-based solve. These scan-based solutions from the phase calibrator(s) will be subsequently applied to the phase calibrator(s), check source(s), and science target(s) with interpolation by time in hif_applycal. The WebLog page shows plots of this gain vs. time at the top of the page. As of PL2022, the `fldmap` is set to the appropriate phase calibrator in the cal library entry, which prevents the polarization calibrator's solutions from being used in the interpolation onto the check source(s) and science target(s) in the occasional case that it was observed adjacent in time to these objects.

Phase solutions and amplitude solutions are also performed on an integration time interval. The results of these rapid solutions for the bandpass, flux, and polarization calibrators are shown on the Diagnostic plots of Phase vs. time. Those plots also contain the short time interval solutions for the phase calibrators along with the solutions for the checksources previously derived in hifa_gfluxscale. As explained in section 9.19, these solution intervals can be 1/4 scan length (rather than 'int') and 'T' solutions (rather than 'G') if the per-spw SNR is low in all spws. The 'a' solves are always 'T' solutions, in order to allow polarized calibrators to differ in their XX and YY intensities, and still obtain the correct Stokes I flux density.

For phase calibrators and check sources, a final 'int' solve is executed in hifa_timegaincal that combines all spws, regardless of the status of the low SNR test in hifa_spwphaseup in order to be able to search for time dependence in the spw-to-spw phase offsets. By pre-applying this table and re-solving for the residual phases on a per-spw, per-scan, per-polarization basis, the task produces an additional plot of these residual phase offsets which will show any time variations. Although the bandpass, flux, and polarization calibrators are included on this plot, they will always appear at zero phase because spw combination is not used in their solution. In optimal cases, the scatter about zero for the phase calibrator(s) should be small with no detectable drift. If any antenna exhibits

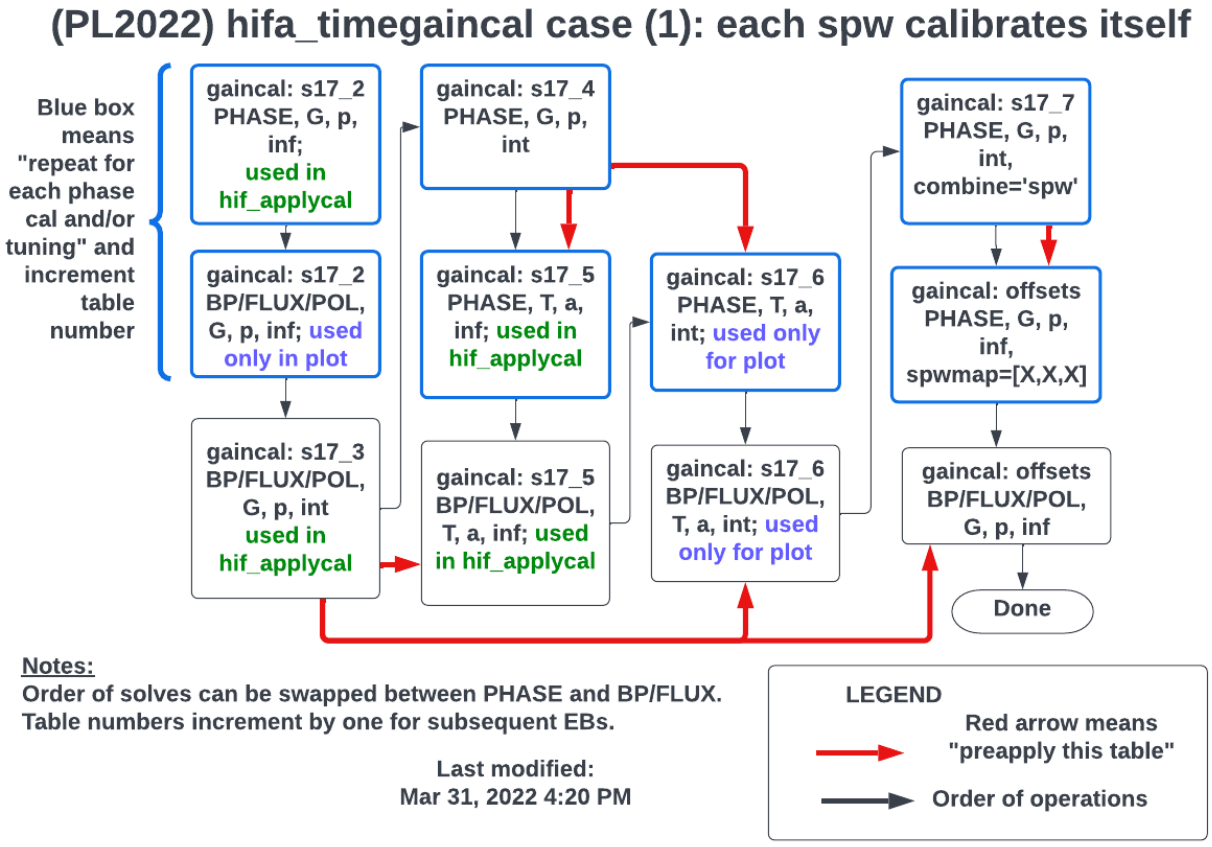


Figure 30: Logical flow in hifa_timegaincal for the case of high SNR in all spws.

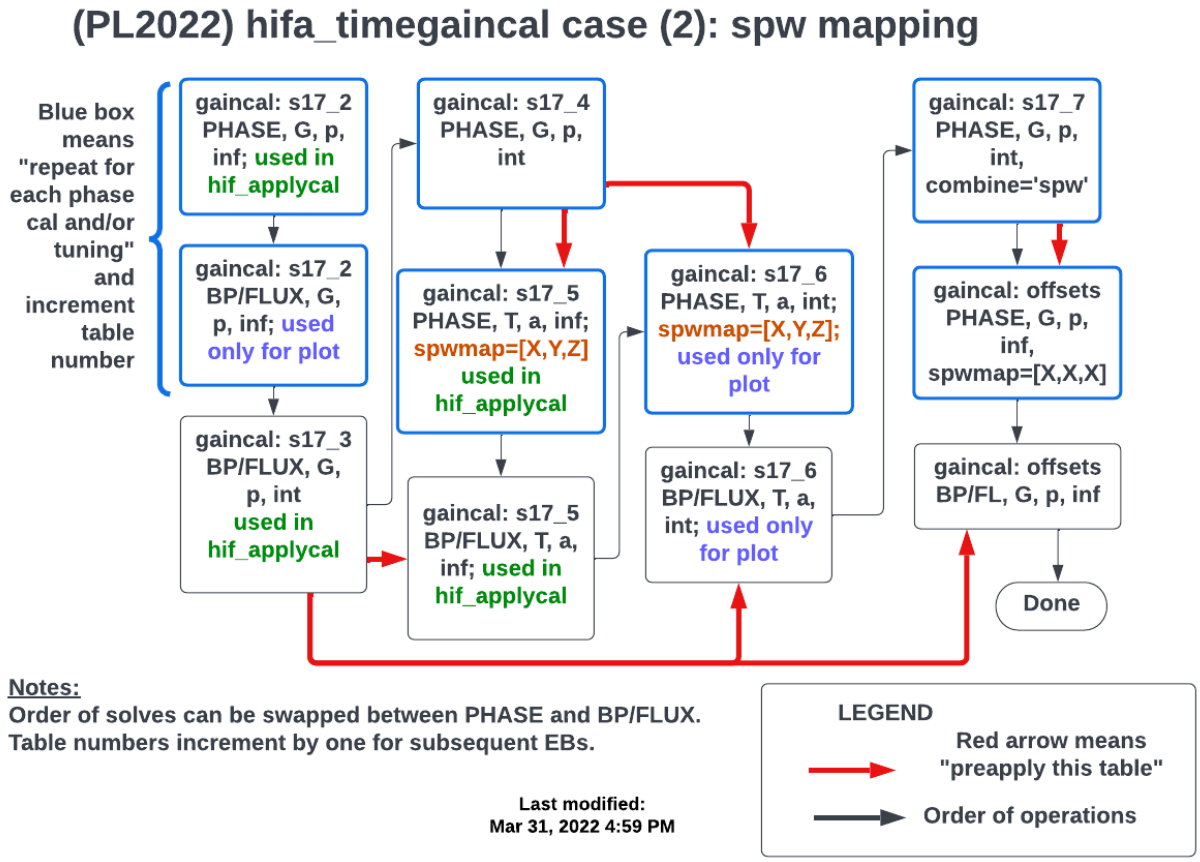


Figure 31: Logical flow in hifa_timegaincal for the case of at least one spw with high SNR but some with low SNR.

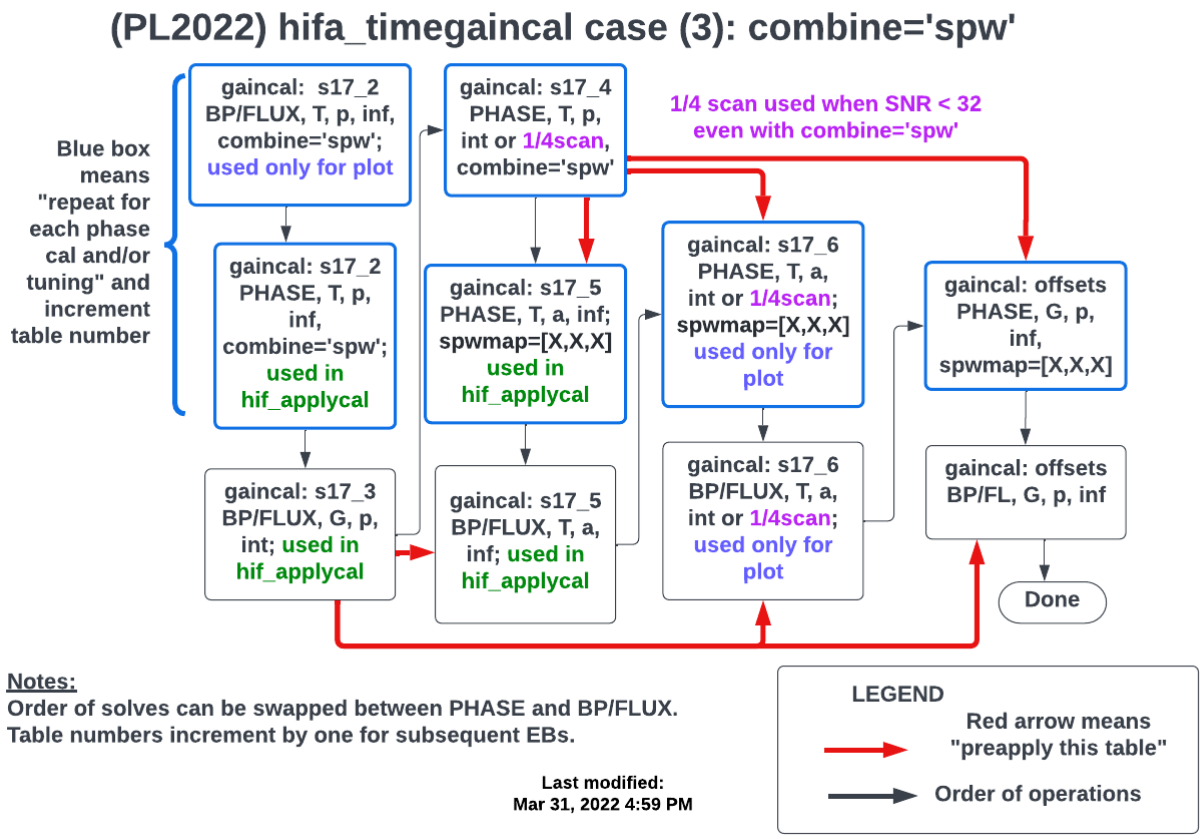


Figure 32: Logical flow in hifa_timegaincal for the case of low SNR in all spws.

significant jumps away from zero, then a warning message and poor score will be generated. The offending antenna(s) should be manually flagged or manually excluded from science target imaging commands in order to obtain the best imaging performance. Jumps on only one or a few antennas can significantly raise the image rms if a relatively bright source is present.

9.21 hifa_targetflag

Because science targets are generally not point sources, the flagging algorithm needs to be more clever than hifa_bandpassflag, hifa_gfluxscaleflag, and hifa_polcalflag. The algorithm identifies outliers by examining statistics within successive overlapping radial uv bins, allowing it to adapt to arbitrary uv structure. Outliers must appear to be a potential outlier in two bins to be declared an outlier. This stage does add processing time, particularly in making the plots. So to save time, we only make the amp vs. time plots if flags are generated, and the amp vs. uv distance plots are made for only those spws that generated flags. Also, to avoid confusion in mosaics or single field surveys, these uv distance plots only show field IDs with new flags (Figure 33).

9.22 hifa_polcal (polarization recipes only)

This task performs polarization calibration per session. More information on ALMA's observing strategy and calibration for polarization observations can be found in Section 8.7 of the Technical Handbook (https://almascience.nrao.edu/documents-and-tools/cycle10/alma-technical-handbook). This stage performs

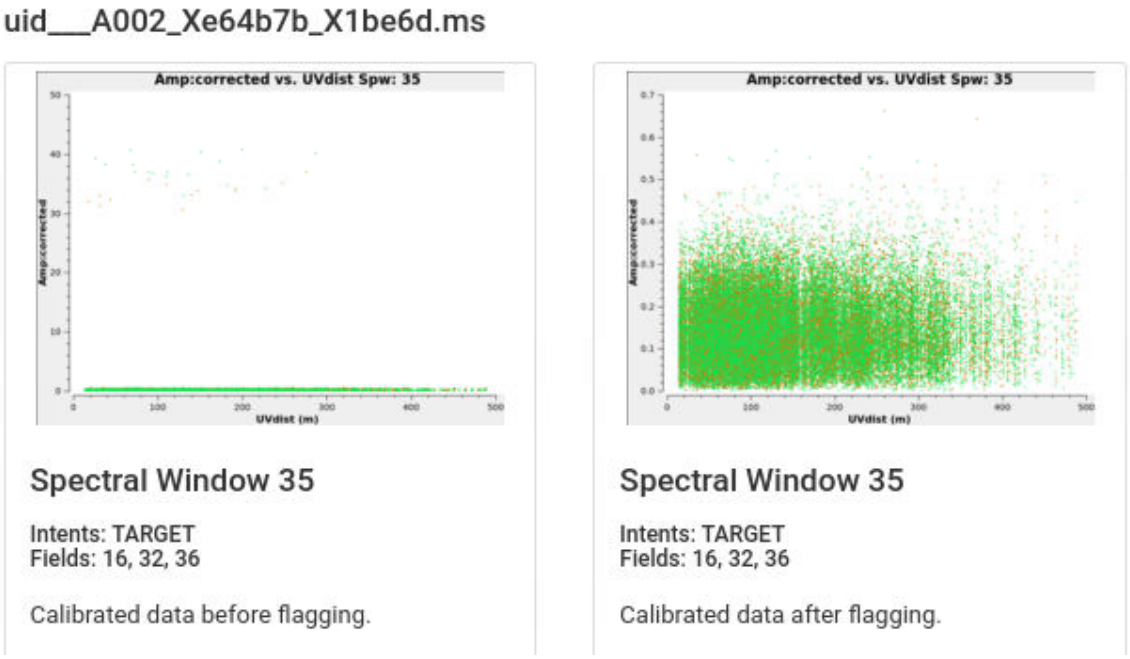


Figure 33: Example flagging of high outliers in science target visibilities in hifa_targetflag.

the calibration and displays the following information in tables:

- Session information including MeasurementSets, polarization calibrator and reference antenna used per session
- Residual polarization of the polarization calibrator after calibration (per spw and an average over all spws), per session
- Derived polarization of the polarization calibrator (per spw and an average over all spws), per session

and the following plots for the polarization calibrator (per session):

- Amplitude vs. parallactic angle plots per spw
- Gain Amplitude Polarization Ratio vs. Scan
- Cross-hand Phase vs. Channel
- D-terms Solutions Gain vs. Channel per spw for all antennas
- Gain Ratio RMS vs. Scan
- X,Y amplitude vs. antenna per spw
- X/Y amplitude gain ratio vs. antenna per spw, and
- Real vs. Imaginary component of the polarization calibrator after calibration (XX, YY, XY and YX)

The following QA scores and messages are displayed (Figure 34):

- Yellow QA score of 0.5 if the residual polarization is $> 0.1\%$, with a message identifying the spws and sessions, green QA score of 1 otherwise
- Yellow QA score of 0.6 if the Gain Ratio RMS after calibration is $> 2\%$, with a message identifying the scans, green QA score of 1 otherwise
- Blue QA score of 0.75 if the D-terms solutions are between 0.10 – 0.15, or a yellow QA score of 0.55 if they are greater than 0.15, with a message identifying the spws and antennas, green QA score of 1 otherwise
- Yellow QA score of 0.65 if the gain ratio is outside the range 0.9 – 1.10, with a message identifying the spws and antennas, green QA score of 1 otherwise

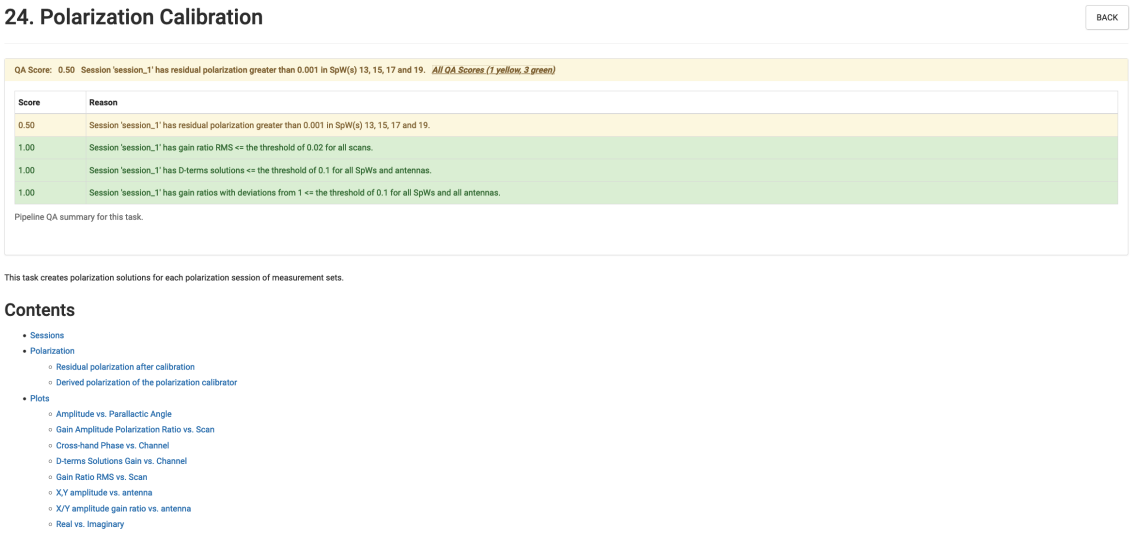


Figure 34: Example of the WebLog QA for the hifa_polcal stage.

The Stokes I, Q, U, V images and polarization intensity and angle of the polarization calibrator are imaged and computed in §9.27.

9.23 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.

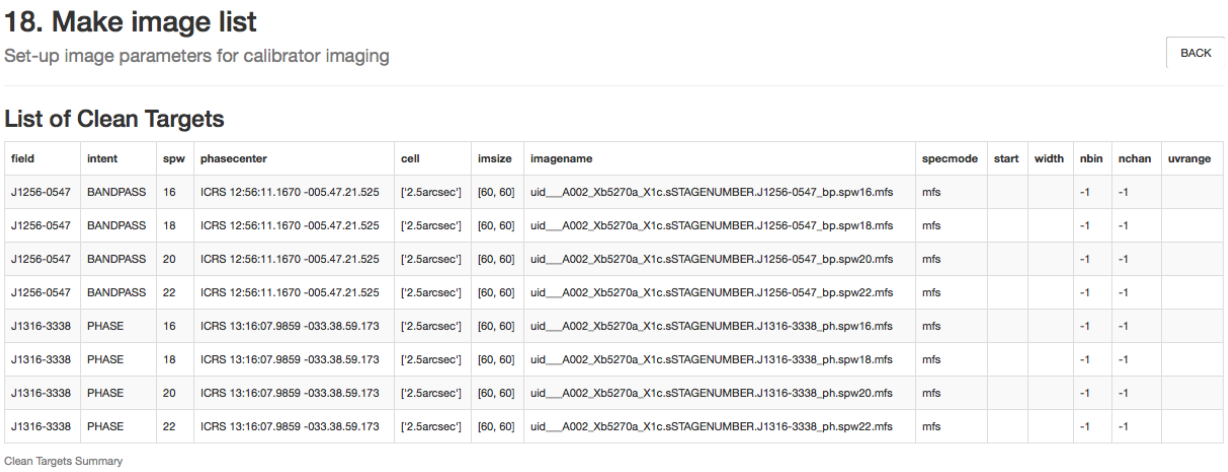
The WebLog page also includes many useful plots of the calibrated data as a function of time and frequency. To reduce the number of plots and processing time to create them, plots of targets only include the representative target, and for mosaics, only the brightest field.

Outliers in these plots can indicate remaining bad data. To help identify these, a QA score is calculate based on the corrected Amplitude-vs-Frequency and Phase-vs-Frequency plots for each calibrator (which are produced for each MeasurementSet and spw). This score is based on fitting a linear function to the corrected data for each antenna, and seeing if the slope or offset of that fit differs significantly from a similar fit to the calibrated data of all antennas. These fits are done on a per-scan, per-polarization basis. If any antenna is found to have a significant difference, the QA score for this stage is set to 0.9 (blue color), and details of the deviant antennas are reported in the expandable QA messages at the top of the page, as well as in an "applycalQA_outliers.txt" file linked to at the bottom of the page. It is important to note that not all reported outliers are (1) visible in the corresponding plots in hif_applycal (which are averaged over all scans and polarizations), or (2) consequential to the final products (since the mean calibration solutions are still robust and adequate to calibrate the final data). However, if problems with the calibration are subsequently found, these messages provide clues on where to look for problems. For data that are delivered as QA2 Pass, one can assume that the ALMA data reviewers have checked these messages and concluded that the overall calibration is not significantly compromised.

Finally, a plot of the uv coverage (original and after all calibration flags are applied) is provided for the representative Source and spw.

9.24 hif_makeimlist: Set-up parameters for calibrator images

Non-default parameters: `intent=PHASE,BANDPASS,AMPLITUDE, POLARIZATION`



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 35: Example of the WebLog for the hif_makeimlist stage. This example is for setting up the parameters for calibrator per-spw multi-frequency synthesis (mfs) continuum 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 35). The `specmode` can be `mfs` for per-spw continuum multi-frequency synthesis images, `cont` for aggregate mfs continuum images of several spectral windows, or `cube` for spectral cubes. The first time the task is run in standard recipes is in preparation for making per-spw mfs images of the calibrators.

9.25 hif_makeimages: Make calibrator images

This stage actually creates the images requested by the most recent hif_makeimlist. The first time it is run in standard recipes is to create per-spw mfs continuum images of the calibrators, using briggs weighting with `robust=0.5`. Calibrator images are cleaned to a threshold of 2 x (predicted rms noise) x (dynamic range correction factor). The dynamic range (DR) correction factor accounts for the fact that targets with a high dynamic range will have larger imaging artifacts, which should not be cleaned. The artifacts are worse for poorer UV coverage, so different dynamic range corrections factors are adopted for 12-m Array and 7-m Array observations according to the following tables:

Calibrator Dynamic Range	12-m array dynamic range correction factor
≤ 1000	1
1000 – 3000	DR/1000
≥ 3000	DR/3000

Calibrator Dynamic Range	7-m array dynamic range correction factor – 1EB
≤ 200	1
≥ 200	DR/200

See the description in §9.39 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 and/or significant decoherence.

9.26 hif_makeimlist: Set-up parameters for polarization calibrator imaging (polarization recipes only)

Non-default parameters: `intent=POLARIZATION, specmode=cont, per_session=True`

Prepare to create aggregate continuum Stokes I, Q, U and V images of the polarization calibrators per session. The default image size is set to 256 in this stage.

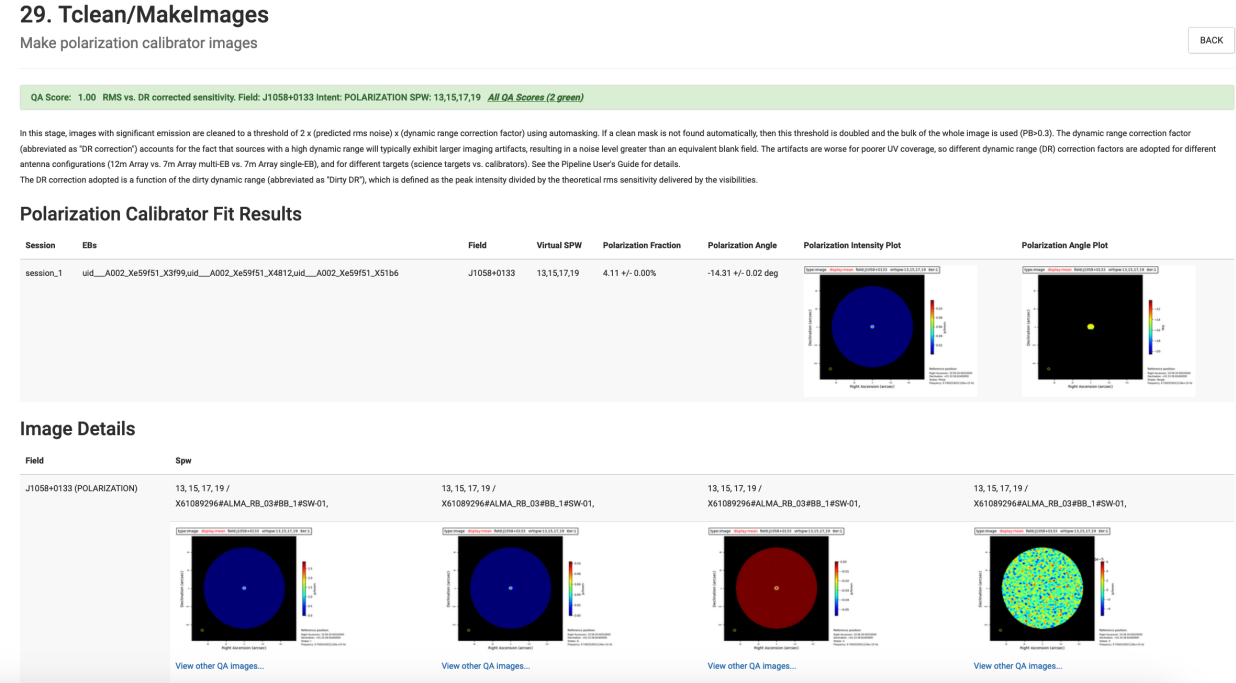


Figure 36: Example of the hif_makeimages stage for the Polarization calibrator (§9.27). First it displays the polarization fit results for each session along with the polarization intensity and angle plots, followed by Stokes I, Q, U, and V images. Clicking on the thumbnails will enlarge the image. Clicking the View other QA images link will bring up the detailed image page.

9.27 hif_makeimages: Make polarization calibrator images (polarization recipes only)

This stage creates and displays per-session aggregate continuum Stokes I, Q, U and V images of all polarization calibrators, performs Gaussian fits to each Stokes I, Q, and U image, and uses the fit results to calculate the polarization fraction and angle. The WebLog displays this information along with the polarization intensity and angle plots and I, Q, U and V images (see Figure 36). A yellow QA score and warning message is displayed if the Stokes I, Q, or U fits fail for a calibrator.

9.28 hif_makeimlist: Set-up parameters for check source images

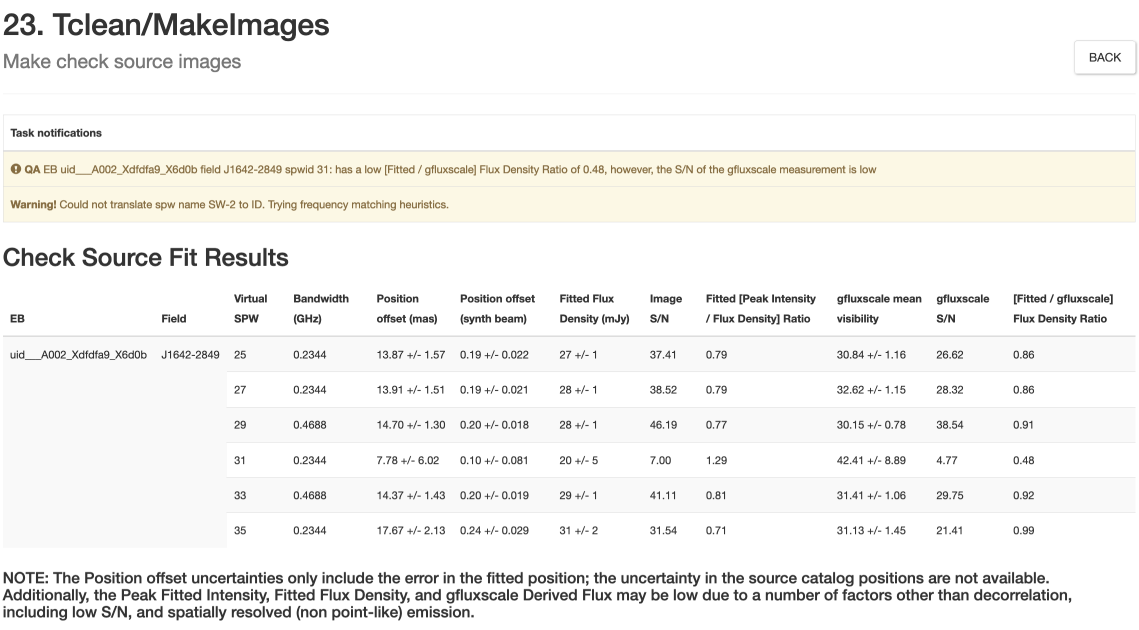
Non-default parameters: `intent=CHECK, per_eb=True`

Prepare to create check source images, one per EB per spw.

9.29 hif_makeimages: Make check source images (if check source present)

After creating per-EB, per-SPW images of all check sources, the pipeline performs a Gaussian fit to each image and shows a table of the fit results (Figure 37). The table displays:

- Check source position offset from catalogue position in mas and synthesized beams, warning level $>$ 0.30*synthesizedBeam (was 0.15 prior to PL2020)
- Fitted [Peak Intensity / Flux Density] Ratio can help to assess decorrelation AND presence of resolved emission, warning level $<$ 0.8



23. Tclean/MakeImages

Make check source images

BACK

Task notifications

QA EB uid___A002_Xdfdfa9_X6d0b field J1642-2849 spwid 31: has a low [Fitted / gfluxscale] Flux Density Ratio of 0.48, however, the S/N of the gfluxscale measurement is low

Warning! Could not translate spw name SW-2 to ID. Trying frequency matching heuristics.

Check Source Fit Results

EB	Field	Virtual SPW	Bandwidth (GHz)	Position offset (mas)	Position offset (synth beam)	Fitted Flux Density (mJy)	Image S/N	Fitted [Peak Intensity / Flux Density] Ratio	gfluxscale mean visibility	gfluxscale S/N	[Fitted / gfluxscale] Flux Density Ratio
uid___A002_Xdfdfa9_X6d0b	J1642-2849	25	0.2344	13.87 +/- 1.57	0.19 +/- 0.022	27 +/- 1	37.41	0.79	30.84 +/- 1.16	26.62	0.86
		27	0.2344	13.91 +/- 1.51	0.19 +/- 0.021	28 +/- 1	38.52	0.79	32.62 +/- 1.15	28.32	0.86
		29	0.4688	14.70 +/- 1.30	0.20 +/- 0.018	28 +/- 1	46.19	0.77	30.15 +/- 0.78	38.54	0.91
		31	0.2344	7.78 +/- 6.02	0.10 +/- 0.081	20 +/- 5	7.00	1.29	42.41 +/- 8.89	4.77	0.48
		33	0.4688	14.37 +/- 1.43	0.20 +/- 0.019	29 +/- 1	41.11	0.81	31.41 +/- 1.06	29.75	0.92
		35	0.2344	17.67 +/- 2.13	0.24 +/- 0.029	31 +/- 2	31.54	0.71	31.13 +/- 1.45	21.41	0.99

NOTE: The Position offset uncertainties only include the error in the fitted position; the uncertainty in the source catalog positions are not available. Additionally, the Peak Fitted Intensity, Fitted Flux Density, and gfluxscale Derived Flux may be low due to a number of factors other than decorrelation, including low S/N, and spatially resolved (non point-like) emission.

Image Details

Figure 37: Check source imaging diagnostic table.

- [Fitted / gfluxscale] Flux Density Ratio, warning level < 0.8
- Warning is also given if S/N of fitted or `hifa_gfluxscale`-derived flux densities are low < 20
- The images themselves which are per EB / spw are located below the table, and except for being per EB are unchanged from the standard hif_makeimages layout.
- Check source imaging uses the dynamic range modifiers for science targets §9.39.3

The QA score per EB/spw is computed as an aggregate of 3 subscores as follows:

- Score1 = 1 - min(1, frac_beam_offset)
- Score2 = max[1, (Fitted Peak Intensity) / (Fitted Flux Density)]
- Score3 = max[1, (Fitted Flux Density) / (gfluxscale Flux Density)]

The FinalScore is the geometric mean of the three scores, i.e. $\sqrt{Score1 * Score2 * Score3}$. The aggregate score for the whole stage is the minimum of the individual EB/spw aggregate scores. Note that Score2 is an indicator of decorrelation, although the score may be low for other reasons, including a weak calibrator or low atmospheric transmission in an spw.

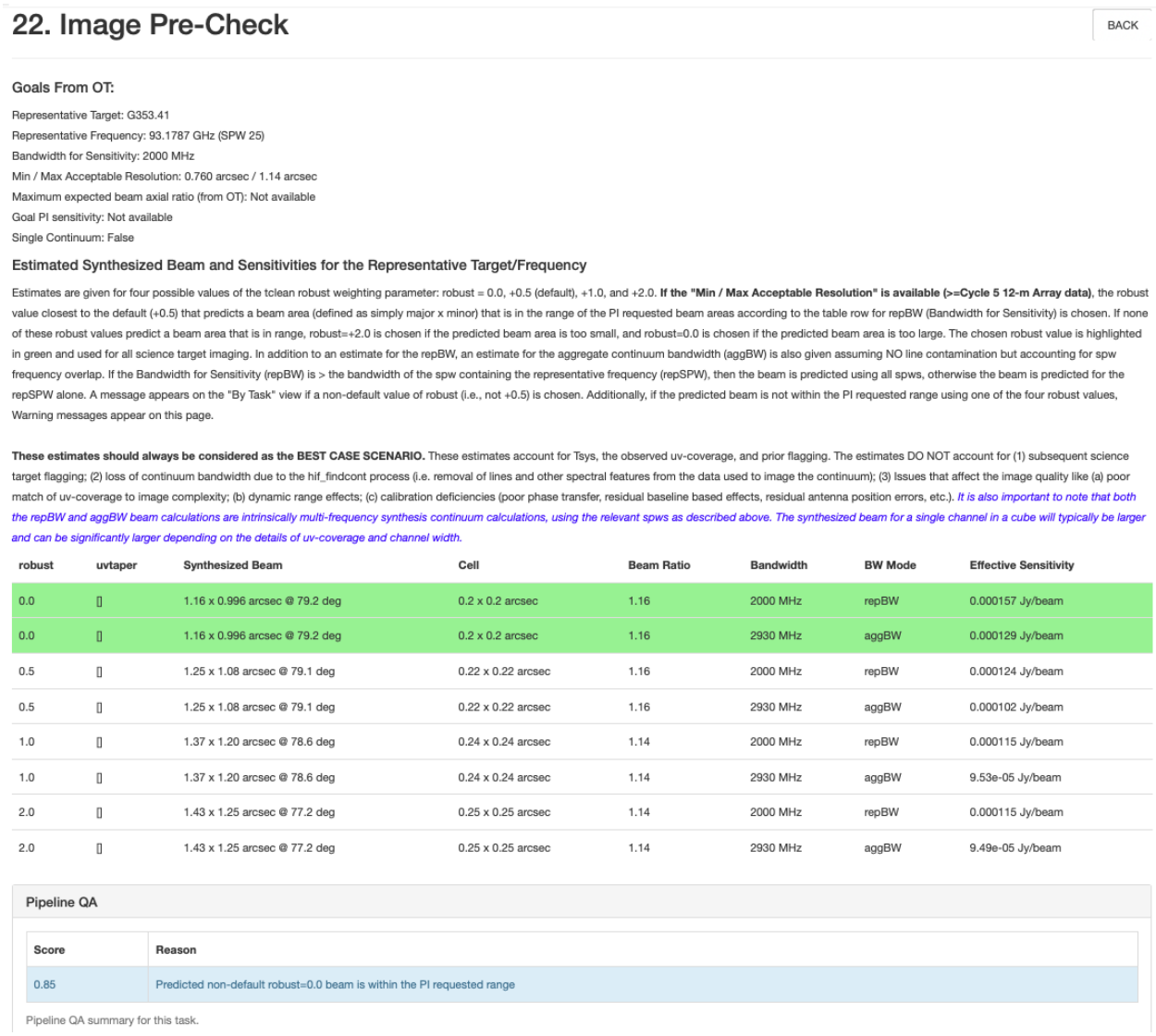
9.30 hifa_imageprecheck

The representative source and spw containing the representative frequency selected by the PI in the OT are used to calculate the synthesized beam and to make sensitivity estimates for the aggregate bandwidth and representative bandwidth for several values of the `robust` parameter. This information is reported in the hifa_imageprecheck WebLog page, including a table like the example shown in Figure 38. If no representative target/frequency information is available, it defaults to the first target and center of first spw in the data (i.e. pre-Cycle 5 data does not have this information available). The best Briggs weighting `robust` parameter in the range [1] of 0.0 – 2.0 that best matches the PI's requested angular resolution is chosen automatically:

[1] Smaller values of `robust` are not considered, since they result in images with poorer imaging characteristics (higher noise and a compromised ability to recover extended emission), especially for mosaics or when the uv coverage is sparse

- The values of `robust` are considered in order: +0.5, +1.0, 0.0, +2.0.
- If one value has a predicted beam with both axes within the PI desired range, that value is used. If that value is not the default 0.5, a warning is printed.
- If no value of `robust` produces a beam with both axes within range, the value that produces a predicted beam area closest to the mean of the PI's range is chosen.
- If no value of `robust` can produce a beam area within the PI's range, the value that produces a predicted beam area closest to the mean of the PI's range is used, a red QA score is assigned, and an error message is printed at the top of the webpage.

The chosen value is used for all subsequent target images (except for hif_findcont, which uses `robust=1`). For the ACA only `robust`=0.5 is considered. Note: the `cell` and `imsize` chosen in this and the following stage is stored in the pipeline context so that all product images (mfs, cubes) have the same `cell` and `imsize`. If these stages are run with non-default intent/source selections, slightly different `cell` and `imsize` may naturally result.



22. Image Pre-Check

BACK

Goals From OT:

Representative Target: G353.41
Representative Frequency: 93.1787 GHz (SPW 25)
Bandwidth for Sensitivity: 2000 MHz
Min / Max Acceptable Resolution: 0.760 arcsec / 1.14 arcsec
Maximum expected beam axial ratio (from OT): Not available
Goal PI sensitivity: Not available
Single Continuum: False

Estimated Synthesized Beam and Sensitivities for the Representative Target/Frequency

Estimates are given for four possible values of the tclean robust weighting parameter: robust = 0.0, +0.5 (default), +1.0, and +2.0. **If the "Min / Max Acceptable Resolution" is available (>=Cycle 5 12-m Array data)**, the robust value closest to the default (+0.5) that predicts a beam area (defined as simply major x minor) that is in the range of the PI requested beam areas according to the table row for repBW (Bandwidth for Sensitivity) is chosen. If none of these robust values predict a beam area that is in range, robust=+2.0 is chosen if the predicted beam area is too small, and robust=0.0 is chosen if the predicted beam area is too large. The chosen robust value is highlighted in green and used for all science target imaging. In addition to an estimate for the repBW, an estimate for the aggregate continuum bandwidth (aggBW) is also given assuming NO line contamination but accounting for spw frequency overlap. If the Bandwidth for Sensitivity (repBW) is > the bandwidth of the spw containing the representative frequency (repSPW), then the beam is predicted using all spws, otherwise the beam is predicted for the repSPW alone. A message appears on the "By Task" view if a non-default value of robust (i.e., not +0.5) is chosen. Additionally, if the predicted beam is not within the PI requested range using one of the four robust values, Warning messages appear on this page.

These estimates should always be considered as the BEST CASE SCENARIO. These estimates account for Tsys, the observed uv-coverage, and prior flagging. The estimates DO NOT account for (1) subsequent science target flagging; (2) loss of continuum bandwidth due to the hif_findcont process (i.e. removal of lines and other spectral features from the data used to image the continuum); (3) Issues that affect the image quality like (a) poor match of uv-coverage to image complexity; (b) dynamic range effects; (c) calibration deficiencies (poor phase transfer, residual baseline based effects, residual antenna position errors, etc.). *It is also important to note that both the repBW and aggBW beam calculations are intrinsically multi-frequency synthesis continuum calculations, using the relevant spws as described above. The synthesized beam for a single channel in a cube will typically be larger and can be significantly larger depending on the details of uv-coverage and channel width.*

robust	uvtaper	Synthesized Beam	Cell	Beam Ratio	Bandwidth	BW Mode	Effective Sensitivity
0.0	[]	1.16 x 0.996 arcsec @ 79.2 deg	0.2 x 0.2 arcsec	1.16	2000 MHz	repBW	0.000157 Jy/beam
0.0	[]	1.16 x 0.996 arcsec @ 79.2 deg	0.2 x 0.2 arcsec	1.16	2930 MHz	aggBW	0.000129 Jy/beam
0.5	[]	1.25 x 1.08 arcsec @ 79.1 deg	0.22 x 0.22 arcsec	1.16	2000 MHz	repBW	0.000124 Jy/beam
0.5	[]	1.25 x 1.08 arcsec @ 79.1 deg	0.22 x 0.22 arcsec	1.16	2930 MHz	aggBW	0.000102 Jy/beam
1.0	[]	1.37 x 1.20 arcsec @ 78.6 deg	0.24 x 0.24 arcsec	1.14	2000 MHz	repBW	0.000115 Jy/beam
1.0	[]	1.37 x 1.20 arcsec @ 78.6 deg	0.24 x 0.24 arcsec	1.14	2930 MHz	aggBW	9.53e-05 Jy/beam
2.0	[]	1.43 x 1.25 arcsec @ 77.2 deg	0.25 x 0.25 arcsec	1.14	2000 MHz	repBW	0.000115 Jy/beam
2.0	[]	1.43 x 1.25 arcsec @ 77.2 deg	0.25 x 0.25 arcsec	1.14	2930 MHz	aggBW	9.49e-05 Jy/beam

Pipeline QA

Score	Reason
0.85	Predicted non-default robust=0.0 beam is within the PI requested range

Pipeline QA summary for this task.

Figure 38: Example hifa_imageprecheck WebLog page, showing the "Goals from the OT" including the PI desired sensitivity. The table shows the sensitivity and predicted beam for a range of `robust` values. The pipeline then chooses the best value (see text).

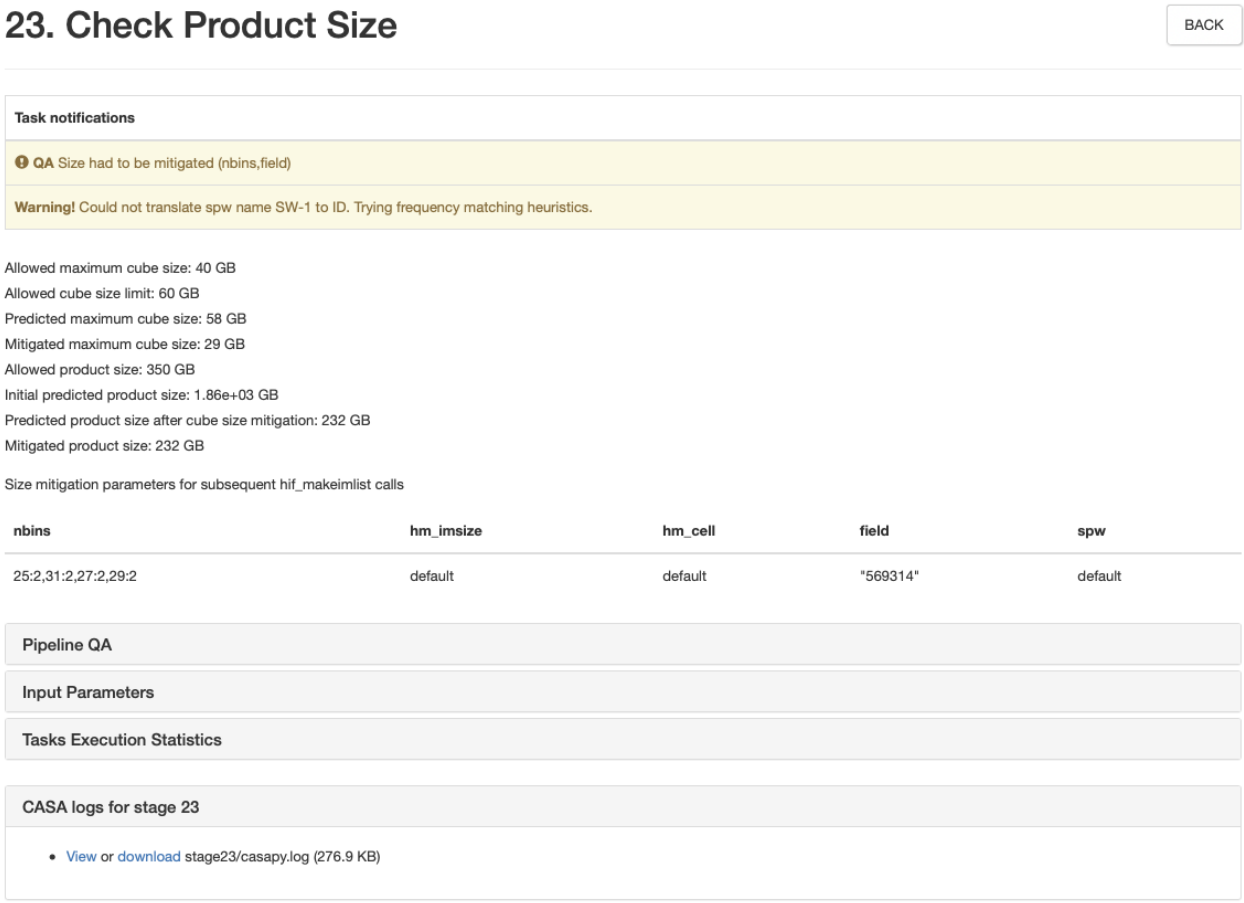


Figure 39: Screenshot of the hif_checkproductsize stage of IF Pipeline. In this example, the spws had to be binned by a factor of 2 and a single field selected in order to get the products below the default thresholds. Before the mitigation the maximum cube size would have been 58 GB and the product size would have been 1.86TB; after the mitigation, the maximum cube size is predicted to be 29 GB and the product size 232GB.

9.31 hif_checkproductsize: Mitigation to avoid overly long runs

This task will modify the characteristics of the imaging products in order to decrease their size, thereby decreasing the time needed to make them so that data can be delivered to PIs more expediently. Figure 39 shows an example WebLog page for a mitigated dataset.

Datasets that have been mitigated will have imaging products with different characteristics than those that have not been mitigated. Full imaging products can be recreated by users, by modifying the tclean commands that are in the **casa_commands.log** file, or by calling the appropriate hif_makeimlist, hif_makeimages pair with the defaults (which will make full imaging products without mitigations – be aware that this could take many days to weeks to complete). The mitigations are done in a priority order, with the mitigation halted once the predicted sizes fall below the thresholds. The parameters that control the mitigation and their default limits are:

- maxcubesize (40 GB): cube size at which to start triggering mitigation
- maxcubelimit (60 GB): maximum cube size that will be produced. Also controls the number of large cubes.
- maxproductsize (500 GB): maximum size of products that will be produced

The pipeline recipe explicitly encodes these values so it can be easily changed universally for all pipeline runs. The **casa_pipescript.py** also encodes these values explicitly, so they can be easily changed on a per-MOUS basis, and the pipeline re-run using the modified file.

The size calculations (in GB) are based on the following:

- mfssize = 4. * nx * ny / 1e9
- cubesize = 4. * nx * ny * nchan / nbin / 1e9
- productsize = 2.0 * (mfssize + cubesize)

where the 4 is the number of bytes per pixel and the 2.0 is to account for the intensity image and the primary beam image, which are both delivered to the user.

The mitigation cascade is as follows:

Step 1: If cubesize > maxcubesize, for each spw that exceeds maxcubesize:

a. If (nchan == 3840) or (nchan in (1920, 960, 480) AND online channel averaging was NOT already performed), then set nbin=2.

b. If still too large, then calculate the Gaussian primary beam (PB) response level at which the largest cube size of all targets is equal to the maximum allowed cube size. The cube sizes are initially calculated at primary beam power level PB=0.2. For an image of width d, the response level at the edge will be PB=exp(-d^2*ln(2)/$FWHM^2$), the image size $d^2 \propto$ -ln(PB), and the required power level to create an image of size = maxcubesize is:

$$PB\_mitigation = \exp(\ln(0.2) * maxcubesize / current\_cubesize)$$

   i. Then account for imsize padding: PB_mitigation = 1.02 * PB_mitigation

   ii. Then limit the size reduction to PB=0.7: PB_mitigation = min(PB_mitigation, 0.7)

   iii. Then round to 2 significant digits: PB_mitigation = round(PB_mitigation, 2)

   NOTE: this mitigation cannot be applied to mosaics, only single fields, and the same mitigated FoV is used for all science target image products.

c. If still too large, change the pixels per beam (cell size) from 5 to 3.25 (if `robust`=+2) or 3.0 otherwise.

d. If still too large, *stop with error*, the largest size cube(s) cannot be mitigated.

Step 2: If productsize > maxproductsize

a. If the number of science targets (single fields or mosaics) is greater than 1, reduce the number of targets to be imaged until productsize < maxproductsize. The representative target is always retained.

b. If productsize still too large, repeat steps 1a, 1b, and 1c, recalculating productsize each time.

c. If productsize is still large, *stop with error*, the productsize cannot be mitigated.

Step 3: For projects with large cubes that can be mitigated, restrict the number of large cubes that will be cleaned:

a. If there are cubes with sizes greater than 0.5 * maxcubelimit, limit the number of large cubes to be cleaned to 1. The spw encompassing the representative frequency shall always be among the cubes retained.

Step 4: For projects that have many science targets, limit the number to be imaged to 30, the representative target is always retained in the list.

The QA scoring for this stage is: 1.0=no mitigation, 0.5=mitigation was performed, or 0.0=error encountered.

When the cube or product size cannot be mitigated, the following warning will appear at the top of the hif_checkproductsize stage:

"QA Maximum cube size cannot be mitigated"

and then the pipeline will stop in the first hif_makeimlist that creates cubes with the message: “Error! Size mitigation had failed. Will not create any clean targets.”

In the example shown in Figure 39, the initial data products were estimated to include a cube that would be 58 GB. This triggered two mitigations: spectral windows were binned by a factor of 2, and the field was restricted to one field. This was sufficient to get the cube size down to 29 GB, so the mitigation cascade stopped. The total product size after the cube mitigation is 232 GB.

9.32 hifa_renorm

This task makes an assessment, and optionally applies a correction, to data suffering from incorrect amplitude normalization caused by bright astronomical lines detected in the autocorrelations of some target sources. A more complete description of ALMA's amplitude normalization and the effects of bright emission lines can be found here: `https://help.almascience.org/kb/articles/623`. In brief, the effect occurs when there is bright enough line emission to be strong in a single-dish spectrum of the source.

The main component of the code is to extract the autocorrelations that were already used for normalizing the data at the correlator level and create renormalization spectra (or scaling spectra). When applied, the scaling will compensate for the previously under-scaled amplitudes by rescaling the channels affected at the spectral window, scan, field, antenna, and correlation level. The scaling spectra are applied to the data when a scaling of >1.02 (>2%) is found.

The main premise of the renormalization correction is to utilize a "clean" autocorrelation spectrum and compare this with a suspected contaminated autocorrelation spectrum from the science target. Simply, the Bandpass source autocorrelations are assumed as line-free and are divided from the science target autocorrelations leaving a unit-less scaling per channel. If the Bandpass and target autocorrelations were identical, except for an astronomical line detection on the target, this division would already create the scaling spectrum. However, due to differences in observation time and airmass, the respective autocorrelations differ slightly and must be baseline fitted in order to establish whether there is any contaminant astronomical line and by how much each contaminated channel should be rescaled. This is done through iteratively fitting the baseline with increasing polynomial orders up to a fifth order polynomial. Due to the differences in airmass between the Bandpass source and the target, the scaling spectra of some observations may exhibit slight residual scaling features related to an inability to properly fit residual atmospheric profiles. The hifa_renorm stage will, by default, have `atm_auto_exclude=True` and therefore act to exclude incorrect scaling that can be caused by strong atmospheric features.

Throughout the renormalization assessment a number of heuristics are run and act to repair any incorrect values in the scaling spectra due to, e.g., birdies in the autocorrelation spectra, divergent baseline fitting near edges of segments, and poor or flagged antennas. Deviations from the median spectrum of more than 0.25% are flagged and replaced by the median spectrum of the matching correlation.

After the scaling spectra are produced, the maximum renormalization value that was found (on a per target, per spw, per scan, per field level) is reported in the main table in the stage summary (see Figure 40). Each row will have a link to a PDF that contains all the scaling spectra for that Execution Block, target, and spectral window. A screenshot of a PDF is shown in Figure 41. Scaling spectra are produced at the per spectral window, per scan, per field, per antenna, per correlation level, and are shown in diagnostic plots made within the stage in the PDF. The summary spectra plots showing the cumulative averages for each antenna to provide a representative overview of what scaling corrections are required are indicated on the main WebLog page. If the peak scaling value is found to be over 1.01 (1%) then it is explicitly labeled in this plot per correlation (X in red, Y in blue). Diagnostic spectra plots are the actual scaling corrections that will propagate to the rescaling application if needed (the plots shown in Figure 41). In these plots, the green and black dotted lines show the median spectra for the X and Y correlations (respectively). In all plots, thin vertical lines show where the spectrum was broken up during the baseline fitting procedure and the atmospheric profile is shown as the magenta line. While `atm_auto_exclude=True` any atmospheric features identified and excluded from the scaling spectra will be covered by a semi-opaque grey region as to indicate the exclusion range. The scaling spectra that will be excluded will still be visible through the region as to allow a visual check. Regions of the spectra that have a frequency (channel) range excluded either by manual inclusions to the `excludechan` parameter or by the automatic atmospheric feature exclusion assessment will be shown by a yellow bar to the bottom of each plot

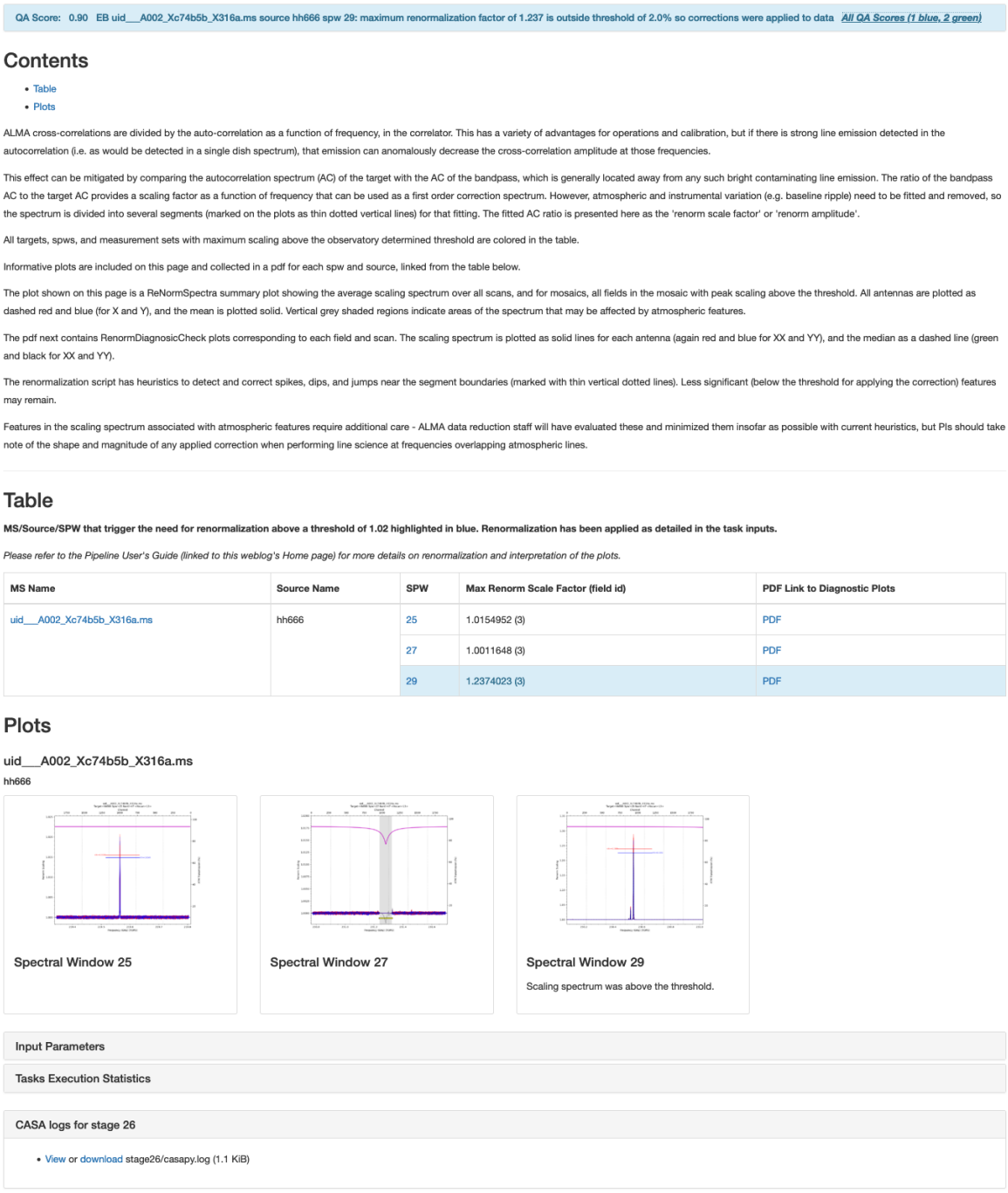


Figure 40: WebLog for the hifa_renorm stage of the IF Pipeline. Results of the renormalization assessment are shown in the form of a table and plots on the main page. In the table the rows are highlighted in blue when they have values above the threshold (default 1.02 or 2%) and will therefore have the renormalization applied when the parameter `apply=True` (spectral window 29 in this example). When hifa_renorm is run in an assessment only sense, with `apply=False`, the rows that are above the 1.02 (2%) threshold will be highlighted in red and the associated stage notifications and warnings will change. The summary plots on the main WebLog page show the cumulative average renormalization spectra for each field and spectral window. Renormalization is performed and reported for every Execution Block, Target field and FDM spectral window. Renormalization is not necessary for TDM spectral windows.

(Figure 42).

Note that Bands 9 and 10 may require special treatment in this stage as the receiver systems are double-sideband and as such there can be complex interactions from the atmospheric features propagating from both the signal and image sidebands. Band 9 and 10 observations have rarely been seen to require renormalization in testing.

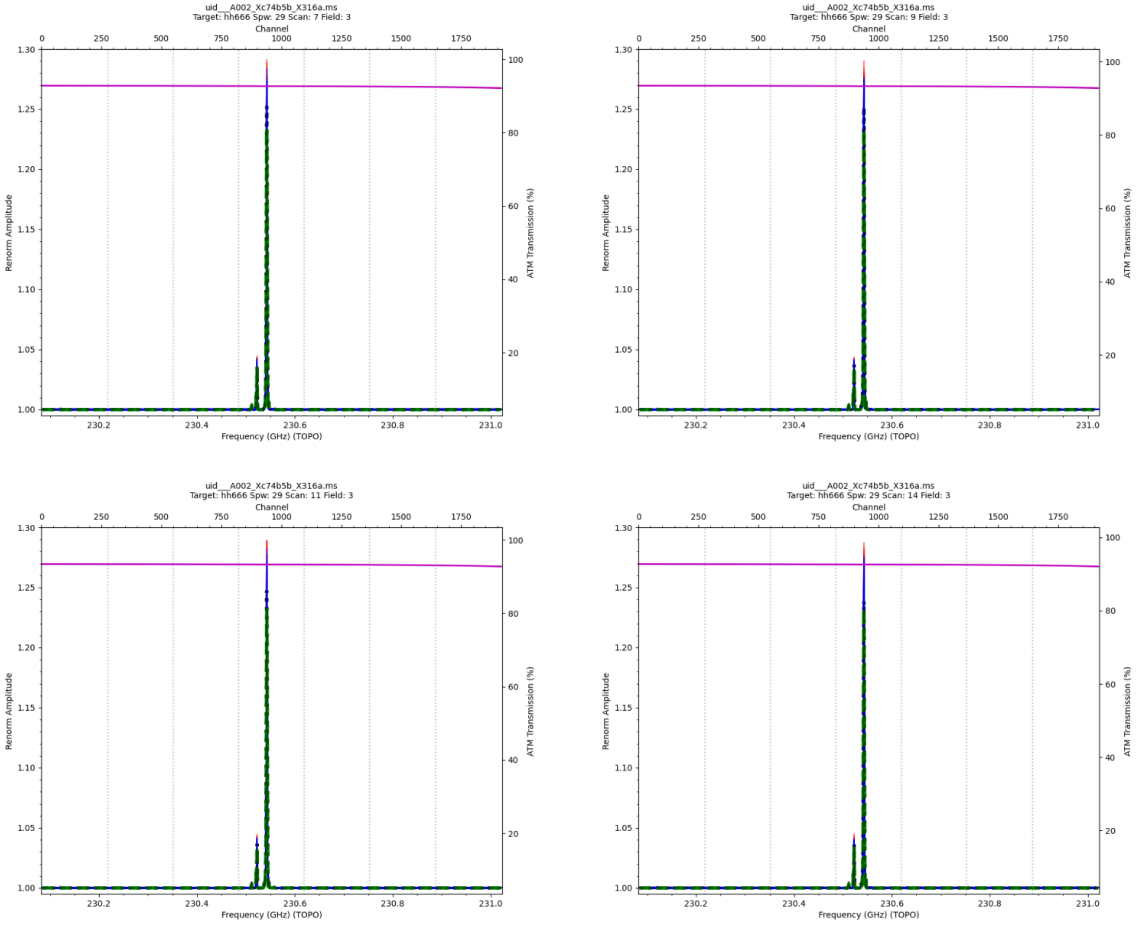


Figure 41: Screenshot of a PDF file generated for an Execution Block, science target, and FDM spectral window during the updated hifa_renorm stage of the IF Pipeline for which the renormalization will be applied. In this example the Carbon Monoxide astronomical line is the reason for the renormalization and is causing an atypically high scaling value exceeding 1.25 (25%).

9.33 hifa_exportdata

Calibration tables, calibrator images (exported in fits format), and other products are moved from the pipeline **working**/ to the **products**/ directory. If the polarization recipe `hifa_polcal` is not specified in the context in the **casa_pipescript.py** file, the polarization calibrator image fits files are not exported in this step. For combined calibration and imaging pipeline runs an intermediate calibration only WebLog tar file will also be created.

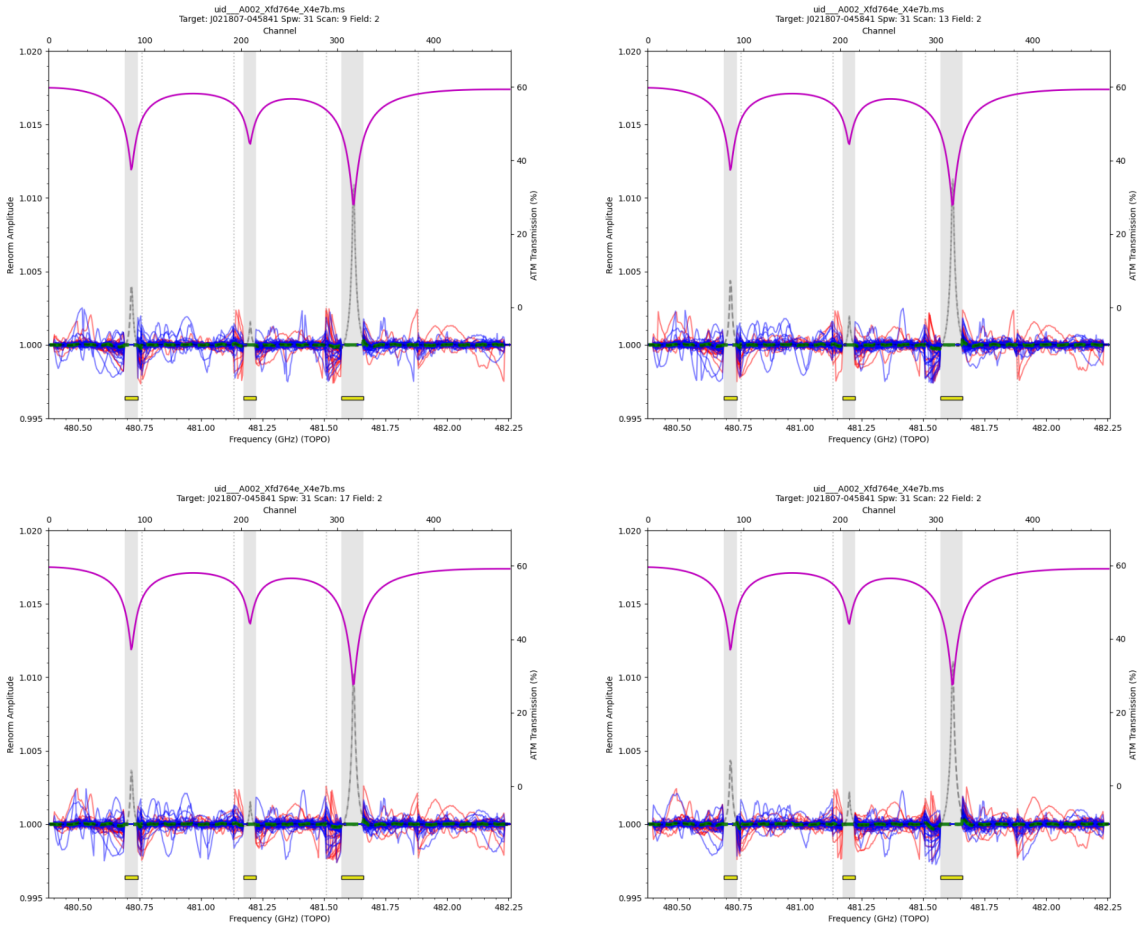


Figure 42: Screenshot of a PDF file as Figure 41 but where atmospheric regions have been automatically excluded and are covered by semi-opaque grey regions. It is clear to see that the excluded scaling spectra (visible by the grey dashed lines) are coincident with the three deep atmospheric lines shown by the magenta line. The section of frequencies (channels) that will be excluded are also illustrated by the yellow boxes towards the bottom of each plot. In this example renormalization will not be applied as there are no real astronomical lines to scale.

NOTE: The subsequent stages are only present if the imaging pipeline was run.

9.34 hif_mstransform

For each execution, calibrated visibilities for the science target(s) are split using the mstransform task to a new MS with ***targets.ms** in the name, as listed on the front WebLog page. The targets MS at this point contain only the calibrated continuum and line emission data.

9.35 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.

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

Non-default parameters: `specmode='mfs'`

In general, this task determines imaging parameters for subsequent hif_makeimages commands. The `specmode` can be `mfs` for per-spw continuum multi-frequency synthesis images, `cont` for aggregate mfs continuum images of several spectral windows, or `cube` for spectral cubes.

In this task, imaging parameters are determined and listed for creation of per-spw mfs continuum images of each science target. This task 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).

9.37 hif_findcont

9.37.1 Overview

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 `nbin` parameter was used in the preceding hif_makeimlist stage, in which case a coarser cube will be made. Each cube is built with `robust`=1 Briggs weighting for optimal line sensitivity, even if a different value had been chosen in hifa_imageprecheck to match the PI-requested angular resolution. The pipeline then runs the function **findContinuum** (available in the source code file **findContinuum.py**), which performs the rest of the work.

9.37.2 Mean spectrum from joint mask

First, it generates the mean spectrum of a masked region of the dirty line+cont image constructed using SNR-based thresholds on the moment0 (integrated) and moment8 (peak) images. The noise measurement of the mask images is based on imstat robust statistics (Chauvenet MAD). This 2D mask image is then "pruned" to remove mask islands with fewer than X pixels where:

$$X = \max([4,\text{int}(\text{beamAreaInPixels}*\text{minbeamfrac})]). \quad (1)$$

Initially, `minbeamfrac`=0.3, but if all regions are pruned and it is ACA 7m data, then another attempt is made with `minbeamfrac`=0.5. At this point, if fewer than 4 pixels of contiguous emission are found after pruning, the whole field at the >30% primary beam level is used as the mask region. In either case, the mean spectrum constructed, analyzed, and displayed really is a spectrum of the source (created by the `ia` tool). See examples in Figure 43. The mask image can be viewed if necessary in the working directory (***.joint.mask2** if present, otherwise ***.joint.mask**, except if Amend Mask was run, see below).

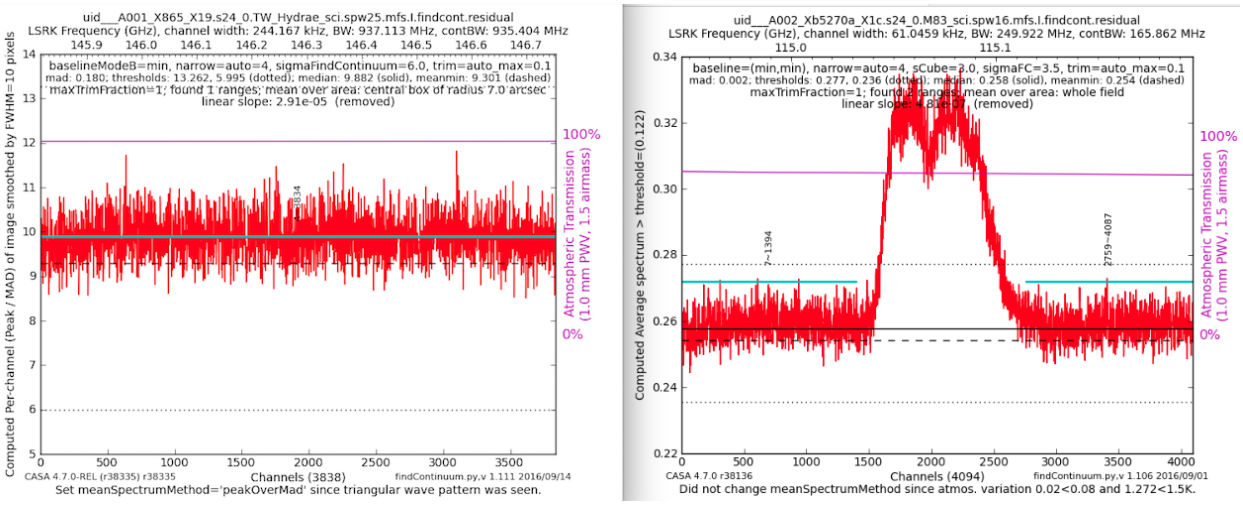


Figure 43: Two examples of hif_findcont plots, one with the entire window identified as continuum (left panel), and another with two identified continuum regions (right panel). The identified continuum range(s) is indicated by the horizontal cyan lines.

9.37.3 Improvement in pre-smoothing of mean spectrum

In order to promote the detection of faint lines, prior to analysis of the mean spectrum of the joint mask hif_findcont will apply a boxcar pre-smoothing of this spectrum, but limited to be $<$ `nchan`/13 in order to preserve some large-scale structure in the spectrum if the user has set $\nu_{\rm sens}$ to be a large fraction of the spw bandwidth. For Cycle 10 data, for which the spectralDynamicRangeBandwidth (equal to 1/3 of the user-specified expected linewidth of the representative target) is newly available in the ASDM, the width of the smoothing kernel is set by the ratio of this quantity to the channel width, rounded to the nearest integer.

For older data, the heuristic introduced in PL2022 will be used instead: if the bandwidth for sensitivity ($\nu_{\rm sens}$) set by the user is sufficiently wider than the channel width of the representative spw as to invoke the creation of a representative bandwidth cube. The smoothing kernel is nominally set to the `nbin` factor shown in the table on the hif_makeimlist (cube_repBW) stage. Additional heuristics are then run to determine if any of the non-representative spws should *not* be similarly smoothed, as the user may have set $\nu_{\rm sens}$ based on continuum rms rather than line rms. First, if strong line emission (peak SNR $>$ 10) is present in the raw mean spectrum, then nbin will be further limited to 2 channels on the 12m array and 3 channels on the 7m array. Second, if the spw bandwidth is less than $\nu_{\rm sens}$, then nbin smoothing is disabled (meaning that no smoothing is done) on that spw. Third, because smoothing broad windows that lack line emission can lead to reduced continuum bandwidth selections (due to ripple structure in the spectral baseline), nbin smoothing is disabled if the spw being processed is wide (>650 MHz, i.e. 937.5 or 1875 MHz) **and** narrow spws (<215 MHz, i.e. 58 or 117 MHz) are also present in the MOUS. The latter combination also strongly suggests that the user is expecting to detect narrow lines as well as continuum, so invoking any smoothing in the wide windows could completely dilute any potential line emission in those windows.

In all cases, if the transition name string associateed with an spw contain the word "cont" and "id=0" (case insensitive), then smoothing is disallowed for that spw.

After the possible pre-smoothing step, the mean spectrum is assessed to find frequency ranges that are the least likely to contain any line emission or absorption. These ranges are listed (in the LSRK frame) on the WebLog page, as well as being indicated by the cyan colored horizontal line(s) and corresponding channel range labels on the spectra.

9.37.4 Moment difference image assessment of line contamination

The initial set of channels found by the previous heuristics are used to construct mom8fc and mom0fc images, then the mom0fc is scaled and subtracted from the mom8fc to remove continuum sources (to zeroth order), creating a "momDiff" image. The presence of significant residual emission in the momDiff image, defined as the peak SNR of the momDiff image > 8 (> 11.5 in spws where the atmospheric variation is deemed large), then indicates line contamination. In order to eliminate the excess (and thereby reduce the peak momDiffSNR), two possible paths can be followed:

- "Amend Mask" Path (multi-letter logic code starts with A)
- "Only Extra Mask" Path (multi-letter logic code starts with E)

Two paths are needed, because growing the size of the mask (amendMask) can be beneficial in some cases, while harmful in others (due to dilution effects). OnlyExtraMask works best in the latter case. Note: The mask shown in the hif_findcont WebLog will be the amended mask (if created), otherwise it will be the original mask. After each stage, a new momDiffSNR is computed to determine if additional work is needed. If amendMask or onlyExtraMask are insufficient, then the channel ranges found by these steps will be intersected with the original ranges. If the momDiffSNR is still significant and we had run amendMask, then extraMask will be tried. If momDiffSNR is still significant (or we had run onlyExtraMask), then the autoLower stage will be run in which the sigmaFindContinuum threshold is further reduced to try to eliminate the apparent remaining line contamination.

The logic path followed is indicated by the logic path letter code in the upper legend of the plot (e.g., ADIEX, where AD=amendMask, I=intersectChannels, E=ExtraMask, X=autoLower). The "D" of "AD" (or possibly "ED") indicates that it was the momDiff image statistics that led to the increase in the mask, while "AC" or "EC" would indicate that it was the mom8fc image statistics (having at least 1 pixel above 8.5 sigma, 9 pixels above 7.5 sigma, and less than twice above 7.25 sigma as above 7.5 sigma).

In addition, a 4-letter improvement code (L=Lower, S=Same, H=Higher) on stats of the final momDiff image (Peak, PeakOutsideMask, Sum, ScaledMADOutsideMask). If either of the first two codes are H(igher), then the channel selection is "reverted" to its original state when the function exists. If the mask is amended and the subsequent new channel range is not reverted, then the relevant mask image in the working directory is called ***.amendedJointMask.original**.

If the logic code is "S, SSSS", then it means the spectrum was assessed but no changes were deemed necessary by the new heuristics. If the logic code is blank, "PR" or "P#", then it was excluded from consideration, including mom8fc images with a large fraction of negative pixels and strong emission pixels (P#), high dynamic range continuum use cases (PR), and long baseline (>400m) TDM cases (blank), all of which are dominated by false positives that result in unnecessary processing, and a generally undesirable decrease in continuum bandwidth. The final value of momDiffSNR is also displayed in the top right legend. Values below 10 usually indicate no (or very little) contamination, and values below 8 are almost always good. But there is a scenario (in some strong hot cores) where the momDiffSNR is not a foolproof indicator of line contamination. Hopefully, a further improvement in this area will be made for the Cycle 10 release.

9.37.5 Results and output

In addition to display on the plot and the WebLog table, the selected 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_uvcontsub and hif_makeimages stages.

If no lines ranges are found, then the corresponding continuum-subtracted cube will not be subsequently cleaned (in order to save processing time). Similarly, if only a single range is found that occupies $>=$ 92.5% of the channels ($>$ 91% if the spw has fewer than 75 channels, i.e. 64-channel full-polarization TDM spws), then the

spw is declared as AllContinuum and no cleaning will be attempted in the subsequent hif_makeimages cube stage.

9.38 hif_uvcontsub

In this task, the previously determined continuum frequency ranges as shown in the **cont.dat** file are used to fit the continuum of each visibility and subtract it, creating a new copy of the measurement set in the process. The fit is performed for each spw independently using a fitorder=1. The WebLog for this stage reports the continuum ranges from hif_findcont in LSRK but translated to the topocentric (TOPO) frame for each MS. After this stage, the original continuum + line emission is contained in the DATA column of the input MS called *_targets.ms, while the continuum subtracted data are written to the DATA column of the new *_targets_line.ms.

9.39 hif_makeimages: common task functionality

9.39.1 Image coordinates

In all hif_makeimages stages, the image will be centered on the ICRS equinox 2000 position requested in the Observing Tool, or in the case of ephemeris objects, the ICRS ephemeris direction evaluated at the time of first integration. For objects with non-zero proper motion rates and/or parallax values entered in the Observing Tool, the image coordinates will be ICRS equinox 2000 but for the epoch of observation (i.e., not 2000.0). In this case, the difference between the Source direction and the Field direction shown in the WebLog is the parallax term, typically only a fraction of an arcsecond, and only when specified for the target in the Observing Tool.

9.39.2 Automatic clean boxes

The pipeline uses the tclean auto-masking method "auto-multithresh" (Kepley et al. 2020) in all hif_makeimages stages where cleaning is performed. This algorithm is intended to mimic what an experienced user would do when manually masking images while interactively cleaning. The parameters `sidelobethreshold` and `noisethreshold` control the masking of the image. The sidelobethreshold indicates the minimum sidelobe level that should be masked, while the noisethreshold indicates the minimum signal-to-noise value that should be masked. The threshold used for masking is the greater of the two values calculated for each minor cycle based on the rms noise and sidelobe levels in the current residual image. Due to a feature that "prunes" small ($<$ `minbeamfrac`) noise-like automask regions, real emission can have all mask regions "pruned", resulting in no clean mask for very compact, typically high S/N emission or absorption. For continuum imaging stages, tclean is run again but falling back to a clean mask that simply corresponds to the area above a fraction of the primary beam response (0.3, if no mitigation of the field of view has occurred). To save time, this step is not done for cube imaging stages.

The pipeline tclean automask parameters vary as a function of imaging type, and the 75th percentile baseline length, b_{75}. These differences are needed because, for example, the smaller 12m-array configurations tend to have better uv-coverage and psfs than more extended configurations. The parameter `fastnoise=True` calculates the noise via a simple median absolute deviation, which is fast, but may overestimate the noise in cases where the field is filled with emission. `fastnoise=False` uses the Chauvenet method to estimate the noise, which may be more accurate in this case, but it is slower.

table continued from previous page

Automask parameter	**7m-array**	**12m-array $b_{75} < 300$m**	**12m-array $b_{75} = 300$m to 400m**	**12m-array $b_{75} > 400$m**
noisethreshold	5.0	4.25	5.0	5.0
sidelobethreshold	1.25	2.0	2.0	2.5
lownoisethreshold	2.0	1.5	1.5	1.5
minbeamfrac	0.1	0.3	0.3	0.3
negativethreshold	0.0	0.0 (continuum) 15.0 (line)	0.0 (continuum) 7.0 (line)	0.0 (continuum) 7.0 (line)
fastnoise	False	False	False	True

table continued on next page

9.39.3 Cleaning Threshold

Images are cleaned to a threshold of 2 × (predicted rms noise) × (dynamic range correction factor). The dynamic range correction factor accounts for the fact that sources with a high dynamic range will have larger imaging artifacts, which may not be able to be cleaned. The artifacts are worse for poorer UV coverage, so different dynamic range (DR) corrections factors are adopted for 12m Array and 7m Array observations, for science targets according to the following tables:

Source Dynamic Range	**12m Array DR correction factor**
≤ 20	1
20 – 50	1.5
50 – 100	2
100 – 150	2.5
≥ 150	max (2.5, DR/150)

Source Dynamic Range	**Dynamic range correction factor**	
	7m Array – 1EB	**7m Array – 2 or more EBs**
≤ 4	1	1
4 – 10	1.5	1.5
10 – 20	2	2
20 – 30	2.5	2.5
30 – 55	max (2.5, DR/30)	2.5
55 – 75	max (2.5, DR/30)	3.0
≥ 75	max (2.5, DR/30)	max(3.5, DR/55)

In addition to this, to prevent divergence in cases of high dynamic range along with poor UV coverage, when the on-source integration time is less than 60s and the dirty dynamic range of a spectral window cube image is greater than 30, the clean threshold will instead be set as 5 × (predicted rms noise) × (dynamic range correction factor).

9.39.4 WebLog

The resulting non-primary beam corrected images are displayed on the respective hif_makeimages stage WebLog page. For each image, the properties are shown next to the associated image png (see Figure 44). In particular, the following are reported: the center frequency, beam parameters (major and minor FWHM resolution & position angle), theoretical sensitivity, cleaning threshold, the "dirty dynamic range" (dirty DR) of the dirty image (image peak to theoretical noise) and corresponding DR correction factor, the non-pbcor image rms (the noise measured in the non-primary beam corrected image over an annulus between the 0.3 to 0.2 response point of the primary beam, or a smaller analogous annulus if field of view mitigation has occurred), image max /min of the primary beam corrected image, fractional bandwidth, aggregate bandwidth, and the image QA score (meant to indicate how close the measured noise is to the theoretical noise, considering also the DR correction factor – see below and §8.7.1).

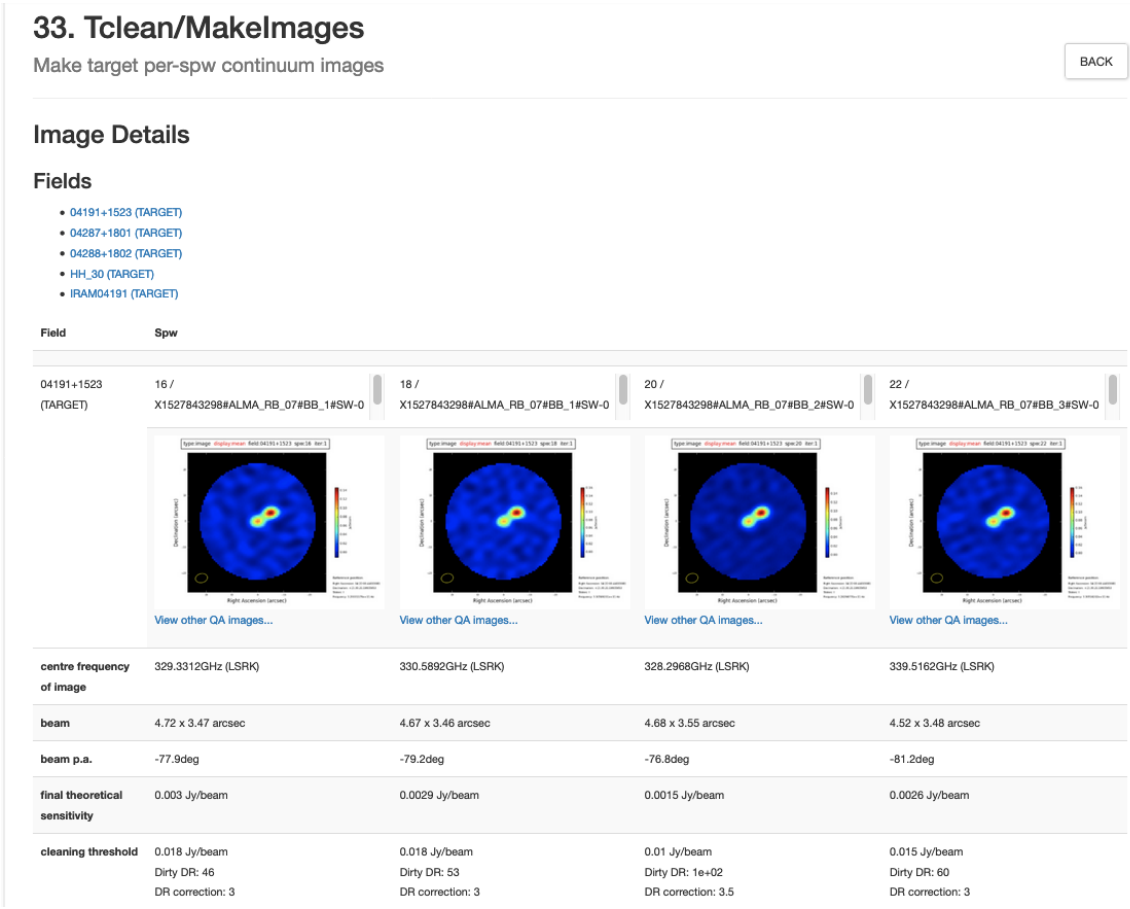


Figure 44: Example of hif_makeimages stage for per-spw continuum images. Clicking on the thumbnail will enlarge the image. Clicking the View other QA images link will bring up the detailed image page (Figure 45).

A clickable link (denoted by the symbol: >_) is available from the image display page to the tclean command used to create that image, similar to the `plotms` commands shown in Fig. 11.

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 45). For cube imaging calls, additional diagnostic images are shown.

9.39.5 Data types and imaging recipe

As described in 9.47, the pipeline now includes additional data types beyond the traditional “DATA" and “CORRECTED" columns, which the different imaging stages will refer to depending on the imaging recipe. These include distinctions between REGCAL and SELFCAL and CONTLINE (non-continuum-subtracted) and LINE (continuum subtracted). The default imaging recipe has the following stages which reference the REGCAL_CONTLINE_SCIENCE datatype (mfs / aggregate continuum images) and REGCAL_LINE_SCIENCE datatype (cube and representative bandwidth cube images):

- hif_makeimages(mfs): Target per-spw continuum images
- hif_makeimlist(cont): Set-up parameters for target aggregate continuum images
- hif_makeimages(cont): Make target aggregate continuum images
- hif_makeimlist(cube): Set-up parameters for target cube images

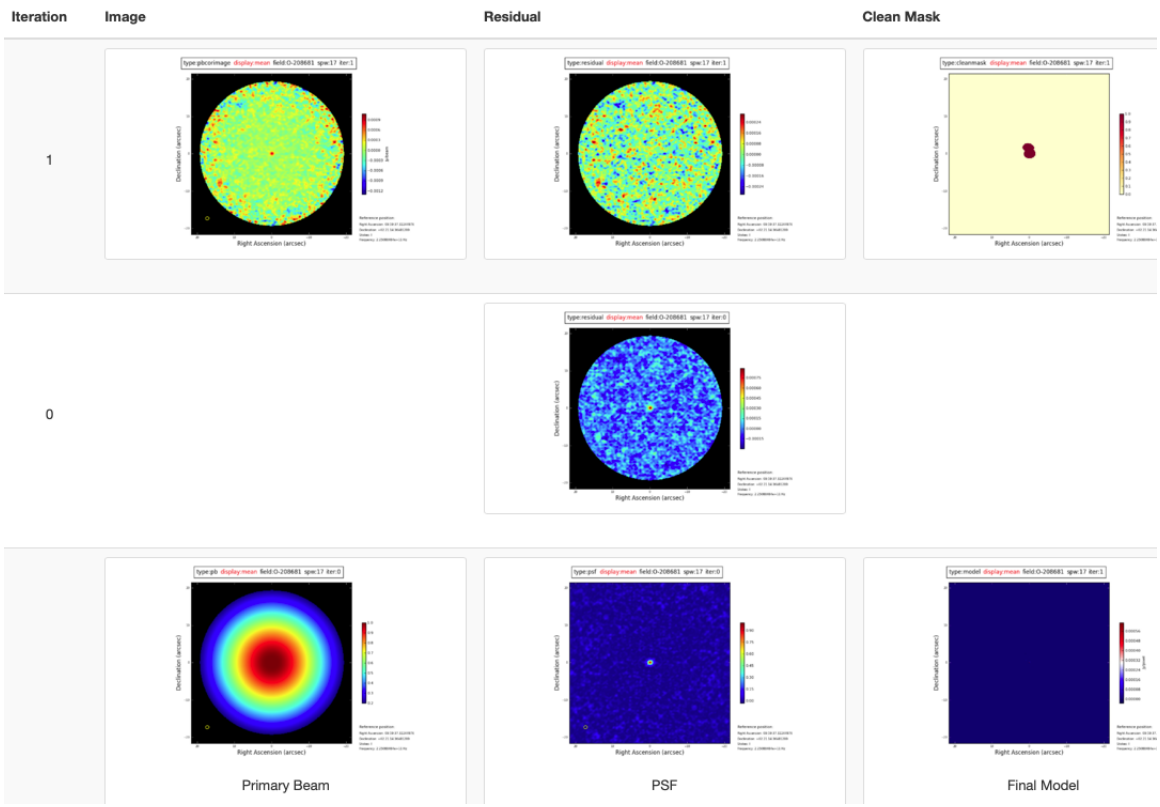


Figure 45: Details page that is displayed after clicking on the View other QA images link on the hif_makeimages WebLog page.

- hif_makeimages(cube): Make target cube images
- hif_makeimlist(repBW): Set-up parameters for target representative bandwidth cubes
- hif_makeimages(repBW): Make target representative bandwidth cubes

Each of these stages are described in more detail below. In recipes where selfcal is performed (see 9.47), the same imaging stages may then be repeated with the same heuristics, but with the datatypes of SELFCAL_CONTLINE_SCIENCE and SELFCAL_LINE_SCIENCE, respectively.

9.40 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), the `robust` selected from the hifa_imageprecheck stage, and any size mitigation triggered by the hif_checkproductsize stage.

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

Non-default parameters: `specmode='cont'`

Imaging parameters are calculated and listed for creation of an aggregate (all spectral windows combined) continuum image of each science target and the `robust` selected from the hifa_imageprecheck stage and any mitigation triggered by the hif_checkproductsize stage.

9.42 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 $\geq 10\%$ (only currently possible for ALMA Bands 3 and 4 data).

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

Non-default parameters: `specmode='cube'`

Parameters are calculated and listed for creation of spectral cube images of each continuum-subtracted spectral window of each science target. The cube parameters use the `robust` selected from the hifa_imageprecheck stage and any mitigation triggered by the hif_checkproductsize stage.

9.44 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). 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 46). The information is similar to that described in the continuum imaging WebLog pages, 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 or nbins (pipeline option) channel averaging is done, the velocity resolution will be twice the channel width.

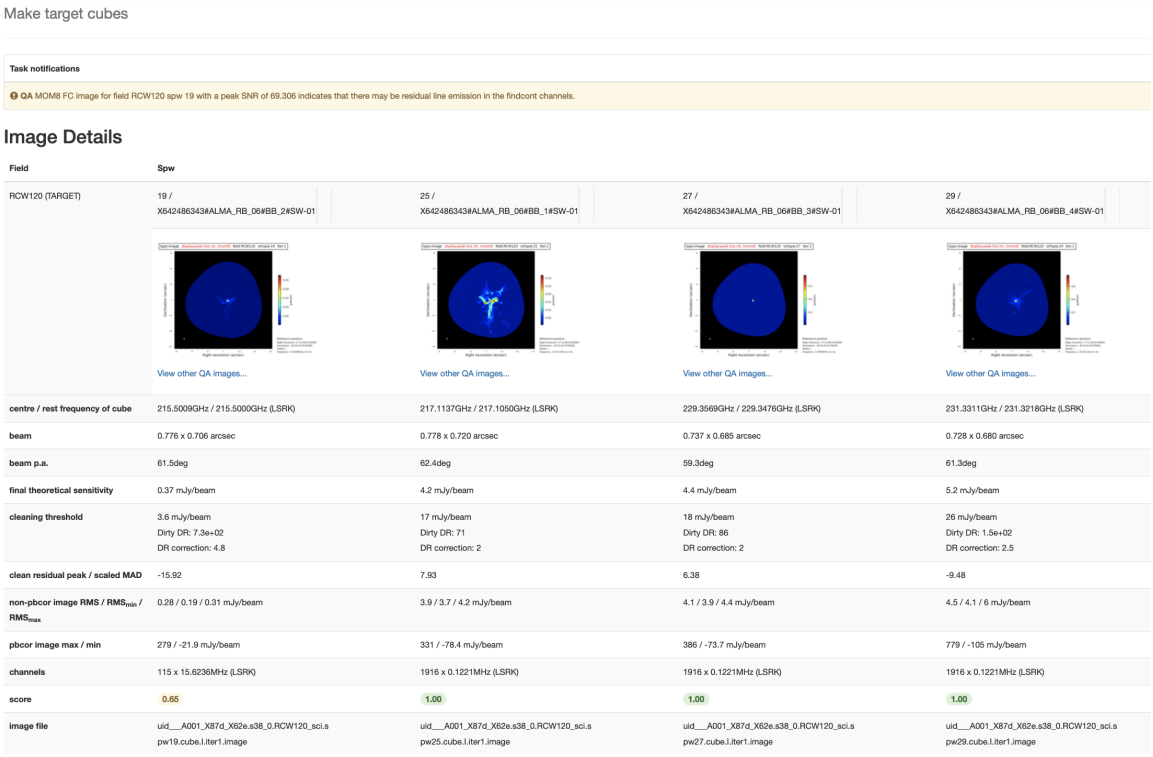


Figure 46: Example of hif_makeimages WebLog page for image cubes. The first spw has a reduced QA score based on the **mom8_fc** image as described in §9.44.1.

9.44.1 hif_makeimages: specmode=cube additional QA

The accuracy of the line-free spectral ranges determined by hif_findcont is assessed by creating a moment 8 (peak intensity) image of those line-free ranges of the cube, called the **mom8_fc** image. In the ideal case, that image will only contain noise. There is a QA metric for this stage to help identify if that image has evidence of extended emission (see corresponding entry in Sec. 8.7.1).

Sometimes there is no simple way to denote continuum (e.g. a line forest), and other times the signal-to-noise is too low to determine whether the hif_findcont selection is adequate, but the QA metric identifies cases that merit manual examination (by looking at spectra through the cube at locations of peaks in the **mom8_fc** image). In most cases, the cubes do not need to be re-generated, because a small change in the continuum ranges will not have a significant effect on continuum fitting and subtraction. But in some cases one may want to create a new aggregate continuum image outside of the pipeline to avoid potential line contamination (the data reducer will do so if they deem it appropriate).

9.45 hif_makeimlist: Set-up image parameters for representative bandwidth target cube

If the PI-requested spectral resolution (bandwidth for sensitivity) is at least 4x larger than the correlator channel width, then in addition to cubes created at that correlator width, the representative source and spw are imaged at the PI's requested resolution, in this and the next stage.

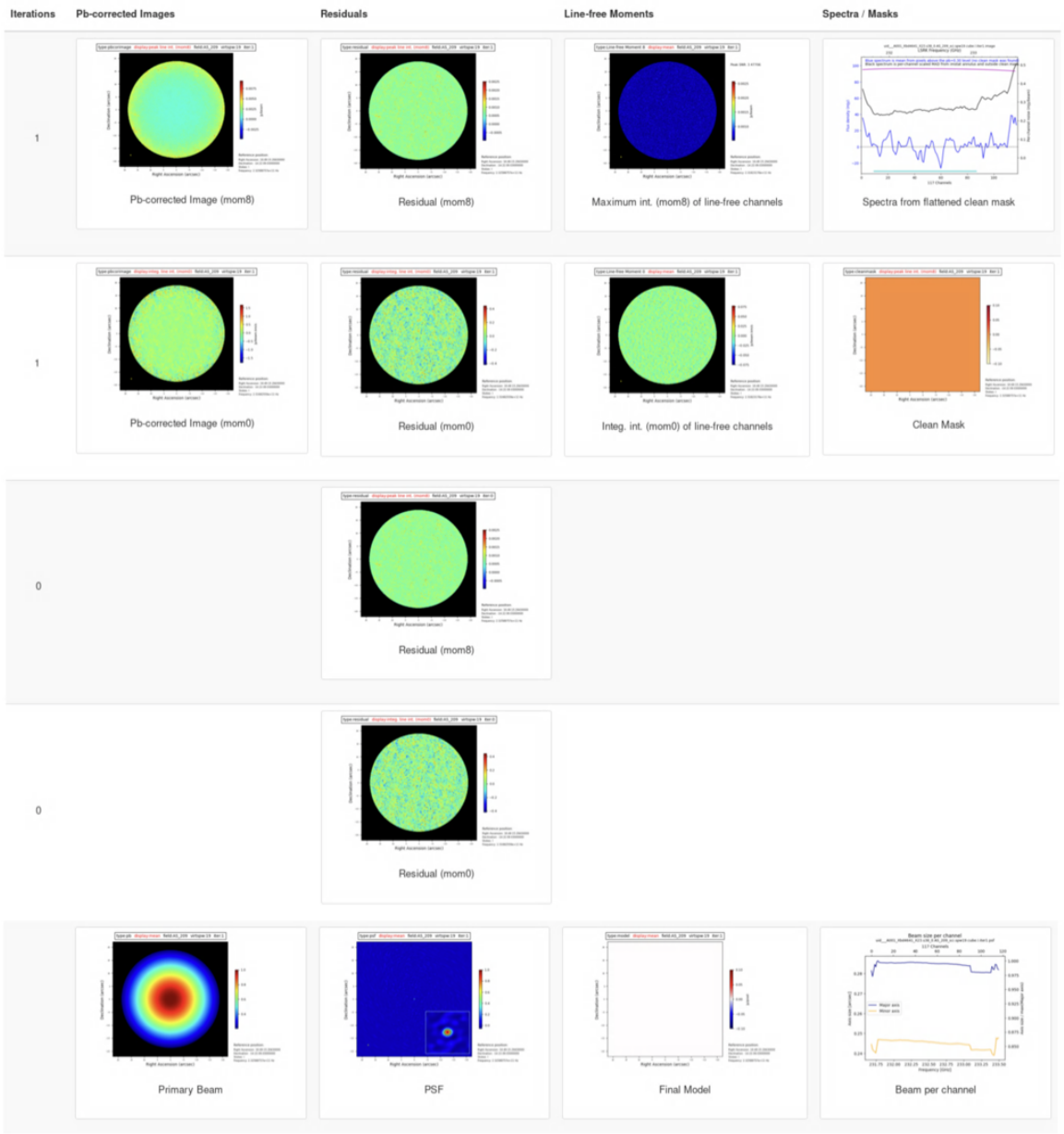


Figure 47: Example of an image cube details page including the line-free moment 0 and moment 8 images.

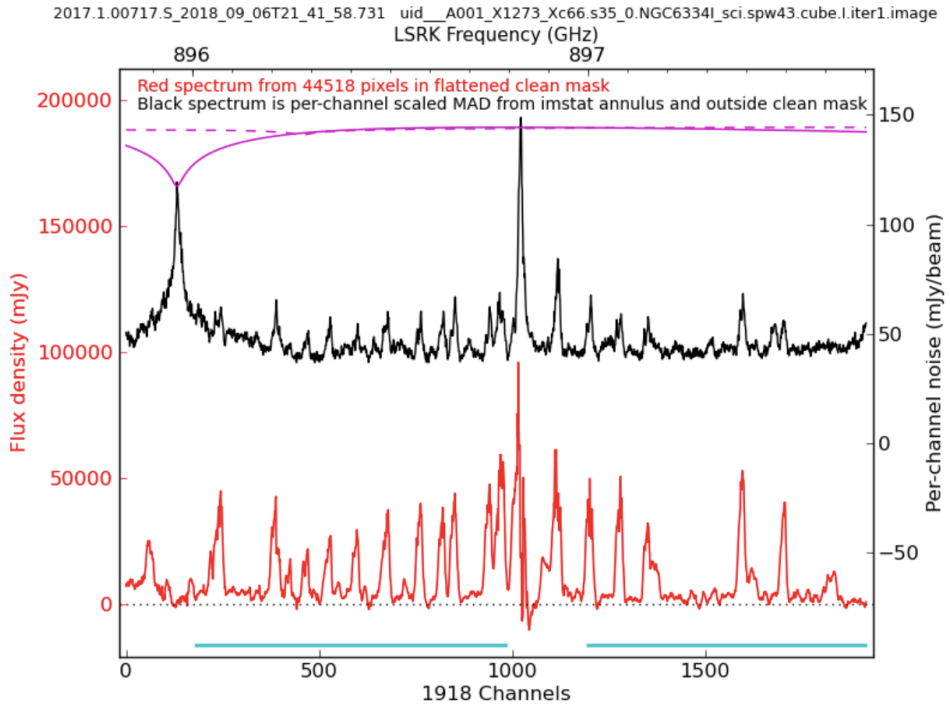


Figure 48: Example of a cube spectrum (red) constructed from pixels inside the clean mask, overlaid with a noise spectrum (black) constructed from the (portion of the) noise annulus outside the clean mask.

9.46 hif_makeimages: Make representative bandwidth target cube

If the PI-requested bandwidth for sensitivity (representative bandwidth) is significantly coarser ($>$ 4x) than the native correlator channel width, an additional cube is created at the PI-requested bandwidth (note: this stage is always created even if it is not populated).

9.47 hif_selfcal

9.47.1 Overview

This task attempts to perform self-calibration on all science targets for which the estimated SNR per-EB per-antenna is $>$ 3. In the event that self-calibration succeeds for a given target, the successful self-calibration solutions are applied to that target. For sources where self-calibration does not succeed, or for which self-calibration is not attempted due to the estimated SNR per-EB per-antenna being too low, no solutions are applied.

9.47.2 Handling of Data and Imaging During hif_selfcal

In order to accommodate the new types of data that will be provided for sources that had successful self-calibration, the pipeline now includes additional data types beyond the traditional “DATA" and “CORRECTED" columns. These new data types are:

- RAW: The data for all targets, calibrator and science, prior to the application of any calibrations, typically residing in the DATA column of the original *.ms files for the dataset.
- REGCAL_CONTLINE_ALL: The data for all targets, calibrator and science, with regular calibrations applied, typically residing in the CORRECTED column of the original *.ms files for the dataset.

- REGCAL_CONTLINE_SCIENCE: The data for science targets only, with regular calibrations applied, typically residing in the DATA column of the *_targets.ms files created by hif_mstransform.
- SELFCAL_CONTLINE_SCIENCE: The data for science targets only, with self-calibration solutions applied, typically residing in the CORRECTED column of the *_targets.ms files created by hif_mstransform.
- REGCAL_LINE_SCIENCE: The line data for science targets only, with regular calibrations applied, typically residing in the DATA column of the *_targets_line.ms files created by hif_uvcontsub.
- SELFCAL_LINE_SCIENCE: The line data for science targets only, with self-calibration solutions applied, typically residing in the CORRECTED column of the *_targets_line.ms files created by hif_uvcontsub.

These datatypes will be referred to in the following discussion rather than the DATA/CORRECTED columns and their corresponding MS files.

At the beginning of this task, lines found by hif_findcont (i.e., the complement of the channel ranges found for continuum) are flagged, the data for each science target are split from the REGCAL_CONTLINE_SCIENCE column into temporary individual MS files with channels averaged to 15.625 MHz, and the original data is reverted to its pre-line flagging state. The task then uses these averaged, per-source temporary MS files for all imaging and calibration during the self-calibration process. If self-calibration is successful for a given science target, the solutions are applied to both the REGCAL_CONTLINE_SCIENCE and REGCAL_LINE_SCIENCE columns and stored in the SELFCAL_CONTLINE_SCIENCE and SELFCAL_LINE_SCIENCE columns.

All images generated as a part of hif_selfcal are made from all SPWs for a given source and use **robust=0.5** along with auto-masking to define the mask during the cleaning process. The thresholds cleaned to, however, vary over the course of the self-calibration process and are described further below.

9.47.3 Self-calibration Workflow

hif_selfcal starts by making a dirty image of each source to be self-calibrated and follows with a cleaned initial image with a threshold determined by in the same way as is done for hif_makeimages, using the predicted rms noise and a dynamic range correction factor. This initial image is used to assess whether there is sufficient SNR per-EB per-antenna to attempt self-calibration and also to set the thresholds for each successive iteration of self-calibration (described further in the next section).

The task will then attempt self-calibration of all science targets that were deemed to have sufficient signal (SNR per-EB per-antenna > 3) to attempt the process. For each such target, the task will loop through the list of solution intervals to attempt and perform the following operations:

1. Generate a “pre" image of the data, with all calibrations, including previous self-calibration solution intervals when this is not the first iteration of self-calibration, applied. The clean threshold used is described below. The model generated by this imaging process is saved in the MODEL column.
2. Use the model generated in the previous step, placed in the MODEL column, along with the **gaincal** task to calculate gain solutions over the solution interval specified for this iteration. These solutions are applied to the science target.
3. Generate a “post" image of the data, with the gain solutions from this interval of self-calibration applied. The clean threshold used is *exactly* the same as the threshold used for the “pre" image of the same solution interval.
4. Evaluate the success of the gain solutions from this solution interval at improving the calibration of the data, discussed further in a subsequent section.
5. If the gain solutions are deemed to have improved the calibration for this science target, move on to the next solution interval and repeat from step 1, or end self-calibration for this target if there are no further solution intervals to attempt. If the gain solutions, however, do not improve the calibration of the target then the calibration is reverted to its state prior to this iteration.

Once these iterations have been completed for each science target for which self-calibration was to be attempted, a final image is made with a clean threshold set by $3\times$ the rms determined from the final successful self-calibration iteration for that source.

9.47.4 Solution Intervals and Thresholds

The self-calibration solution intervals to attempt and the thresholds to clean to for each of those solution intervals are calculated in advance of the self-calibration process. hif_selfcal will start by cleaning relatively shallowly and with long solution intervals for gain calibration, but will decrease both the solution intervals and clean depths with each successive iteration.

The first solution interval, dubbed "inf_EB", is always set to use *combine="scan"*, *solint="inf"*, and *gaintype="G"* i.e. it is calculated over an entire EB, with separate solutions for each EB if multiple EBs are present. This first solution interval initially attempts to calculate solutions per-polarization and per-spw, but will also test whether using either a spwmap to map some spws to others with better solutions or *combine="scan,spw"* produces sufficiently less flagging. If successful, the gain table from this solution interval is pre-applied before calculating the gains for subsequent solution intervals.

Subsequent solution intervals all use *combine="spw"* and *gaintype="T"*. The second solution interval is typically *solint="inf"*, though in the event that a science target has only a single scan this will be will skipped as it would be almost equivalent to the inf_EB solution interval, and finish with *solint="int"*. Between the "inf" solint, or inf_EB if inf is not included, and the "int" solution intervals, hif_selfcal will attempt solution intervals that split the median scan time approximately evenly into multiple solution intervals. The task will target a total of 5 iterations of self-calibration including the inf and int solution intervals, and amount that the current solution interval is divided by to reach the next solution interval is set to reach this maximum number of 5 iterations, with some adjustments to avoid single integrations from being left out.

The solution intervals are, with the exception of inf_EB, not cumulative. That is to say that if the solution intervals to attempt are [inf_EB, inf, int], the inf_EB solutions will be pre-applied when calculating gains for inf and int, but inf will not be pre-applied when calculating int. At the end of self-calibration, only the gain table from the final successful solution interval, along with the gain table from inf_EB if it is not the final successful interval, will be applied. Any successful intermediate steps will be discarded.

Clean thresholds are set initially as multiples of the rms of the image. The first (inf_EB) solution interval is set by the SNR of the initial image divided by a factor of 15, the final, int, solution interval is set to clean to a threshold of 5× the rms of the image, and the thresholds in-between are scaled logarithmically between these two values. While the planned thresholds for each solution interval are set as multiples of the rms and remain unchanged, the rms of the images is measured from the "post" image of each solution interval, stored as the current rms, and multiplied by the planned multiple of the rms for the next solution interval.

9.47.5 Solution Acceptance/Rejection

Two criteria are considered when determining whether a given self-calibration solution has successfully improved the calibration of a science target: First, the beam area in the image made after applying the solutions for the solution interval must not increase by more than 5% compared with the beam size of the initial image, pre-self-calibration. Additionally, the SNR of the "post" image for the current solution interval (step 3 above) must increase compared with the SNR of the "pre" image of the current solution interval (step 1 above). The rms of both "pre" and "post" images are measured outside of the clean mask from the "post" image so that they are calculated from the same area of the image. For the inf_EB solution interval, a slight ($< 2\%$) reduction in the SNR is allowed to enable the routine to go on to additional, shorter solution intervals. If either criterion is not met, then the solution interval is deemed unsuccessful.

9.47.6 Weblog

The hif_selfcal stage Weblog page shows a summary table describing the targets considered, the solution intervals to be attempted, and whether self-calibration was successful for that target. For each target, there is then a table showing the initial and final images, listing statistics for each such as the SNR and rms, along with a brief description of whether self-calibration was successful, what the final solution interval was, and why self-calibration was stopped. Then an additional table is provided providing statistics such as the SNR, improvement of the SNR,

List of Self-cal Targets

Field	Band	spw	phasecenter	cell	imsize	Solints to Attempt	Success	Contline applied	Line applied
IRS48 (rep.source)	Band 7	25,27,29,31	ICRS 16:27:37.1797 -024.30.35.480	['0.052arcsec']	[540, 540]	inf_EB, inf, 151.20s, 48.38s, 12.10s, int	✔	✔	✔

Self-calibration Target(s) Summary

Self-cal Target Details

IRS48 Band 7 back to top SUMMARY PER-SOLINT DETAILS

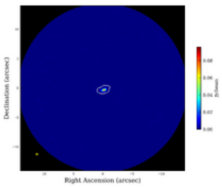
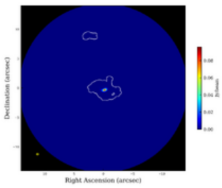
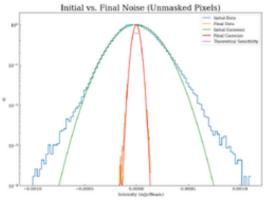
Data Type	Initial	Final	Brightness Dist. / Ratio
Image			
Integrated Flux	193.219 ± 2.278 mJy	200.061 ± 2.276 mJy	1.035
SNR	482.931	3050.983	6.318
SNR (N.F.)	484.433	3053.878	6.304
RMS	0.172 mJy/bm	0.031 mJy/bm	0.182
RMS (N.F.)			0.183
Beam	0.330"x0.258" -84.362 deg	0.331"x0.258" -84.180 deg	1.002
Success / Final Solint	Yes / int		
Stop Reason	None		

Figure 49: Example WebLog for the hif_selfcal stage. The List of Self-cal Targets table shows the list of science targets considered for self-calibration, the imaging parameters used, the solution intervals to be attempted, and whether or not self-calibration was successful and applied. Then the Self-cal Target Details shows further details of the before and after self-calibration status of the data, including SNR, rms, beam size, along with before and after images and details about whether self-calibration was successful or not, the final successful solution interval, and why self-calibration stopped for this source.

RMS, improvement of the RMS, integrated flux within the clean mask, beam size before and after that solution interval, and whether that solution interval succeeded or failed (and why it failed) for each solution interval attempted for that source. Clicking on the "QA Plots" link for each solution interval leads to a separate page showing the before and after images as well as plots of the gain solutions for each EB and antenna combination.

Statistics dubbed with the moniker "near-field" (or "NF") use the rms measured in a mask that extends from slightly beyond the clean mask out to a distance several times the largest angular scale beyond that rather than the rms calculated outside of the clean mask out to the extent of the image. These statistics are tracked by hif_selfcal at this time and are provided on an informational basis, but are not used in any decision-making at this time.

An example Weblog for the hif_selfcal stage is shown in Figure 49.

9.48 hifa_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. This stage is run in operations, but is not included in the **casa_pipescript.py** script.

9.49 hifa_restoredata

The hifa_restoredata task restores calibrated MeasurementSets from archived ASDMs and pipeline flagging and calibration data products. It contains many of the same parameters as hifa_importdata is called at the beginning of the imaging recipe, but is not called in the cal+imaging recipe. When importing the ASDM and converting it to a MeasurementSet (MS), if the output MS already exists in the output directory, then the importasdm conversion step is skipped, and instead the existing MS will be imported.

10 Single Dish pipeline tasks and WebLog pages

This section describes each Single Dish Pipeline task and its associated task WebLog page. For a detailed description of parameters for each task, refer to the ALMA Pipeline Reference Manual.

10.1 hsd_importdata

The WebLog for hsd_importdata task shows the summary of imported MSs, grouping of spws to be reduced as a group, 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 including reference sources and just on-source pointings (Figure 50). 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 50 the green line going to/from the red dot indicates that the antenna goes to the last scan and returns to the OFF position. The grey dots indicate flagged data. hsd_importdata generates pointing pattern plots with ephemeris correction in addition to the plots without correction if target is moving source.

10.2 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, autocorrelation, edge channels, and low transimission. Note that the value in the "Before Task" column corresponds to the percentage of flagged data by binary data flagging (BDF).

10.3 h_tsyscal

This page 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.

10.4 hsd_tsysflag

This page shows the flagged Tsys spectra per spectral window after heuristic flaggings are applied.

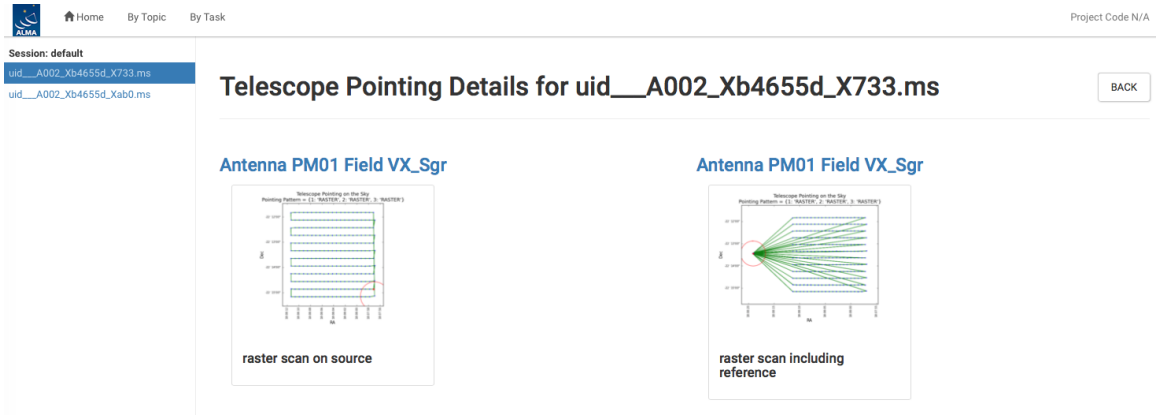


Figure 50: The detailed page of Telescope Pointing on the MS summary page.

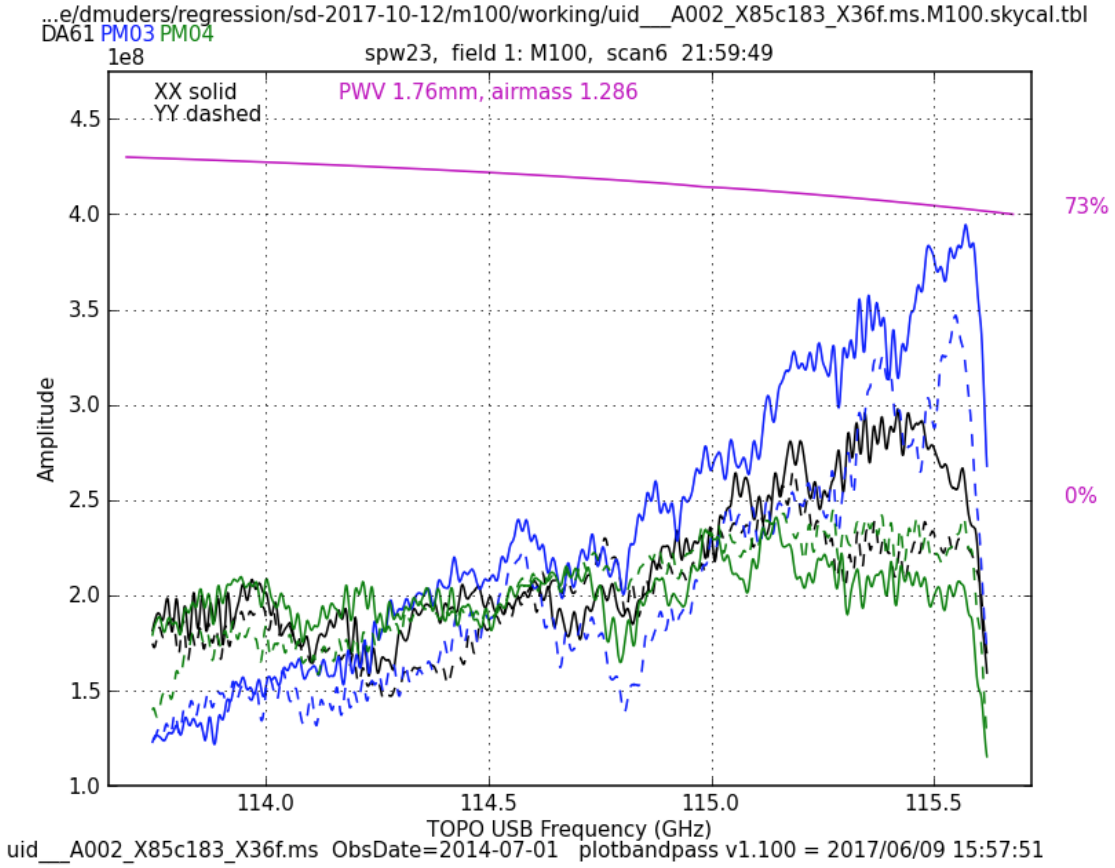


Figure 51: An example of an OFF spectrum. Different antennas are shown in different colors. The atmospheric transmission curve is shown in magenta.

10.5 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 51). The different colors indicate different scans (times). The magenta lines indicate the atmospheric transmission at each frequency.

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.

In addition, amplitude versus time plots for the OFF_SOURCE data and elevation difference between ON_SOURCE and OFF_SOURCE data plots are shown in this page.

10.6 hsd_k2jycal

This page shows the list of Kelvin-to-Jansky conversion factors that Pipeline has read from a file "jyperk.csv", which shall contain the factors per MS, per spw, per antenna, and per polarization. With a parameter of dbservice=True (default), Kelvin-to-Jansky conversion factors are obtained via the data base (DB) and a file "jyperk_query.csv." is produced. This csv file is read in the pipeline.

10.7 hsd_applycal

This page shows a list of the calibrated MSs with the name of the applied Tsys, Sky and amplitude (Kelvin-to-Jansky conversion) calibration tables, and also shows the integrated spectra (amplitude vs frequency) after calibration.

10.8 hsd_atmcor

This page shows the list of the calibrated MSs with model parameters of atmType, h0, and dTem_dh. atmType, h0, and dTem_dh, are the atmospheric type, scale height for water, and derivative of temperature with respect to height. The automatic procedure, as default, evaluates and selects the best model to be applied to correct for atmospheric effects from four different models: atmType=1 (tropical), 2 (mid-latitude summer), 3 (mid-latitude winter), and 4 (subarctic summer), with fixed temperature gradient (dTem_dh) of -5.6 K km^{-1} and fixed scale height for water (h0) of 2 km. In case user-defined parameters are provided, the heuristics will be turned off. More explanations can be found on this page. This page also shows the integrated spectra (amplitude vs frequency) after the atmosphere correction. The integrated spectra before the correction can be see in the page of hsd_applycal. The details of the atmosphere model are described in Sawada et al. 2021, PASP, 133c4504S.

10.9 hsd_baseline

Spectral data before/after baseline subtraction

The first three rows of the hsd_baseline page of the WebLog shows the five grids of spectra per source: the ones on the top and in the third rows correspond to the spectra before and after the baseline subtraction, respectively. The one in the second row is obtained by averaging all the spectra associated with each grid before baseline subtraction (see Figure 52). Averaging the data improves the S/N ratio making the spectral line features more prominent. These plots, which appear just after clicking the hsd_baseline link of the WebLog, show a representative spectral grid of each spw. Normally they correspond to the spectral grid of a certain antenna. The spectral grids are shown in R.A./Decl. coordinates. Each small panel shows **integrated** spectrum per grid cell over the time. The red line (basically toward the horizontal direction) over-plotted on the spectrum indicates the fitted function to be used for baseline subtraction for spectral data before baseline subtraction, while it indicates the zero-level for spectral data after baseline subtraction.

On the top panel of each spectral grid map, a spatially integrated spectrum per ASDM, antenna, spw and polarization is shown. The magenta lines indicate the atmospheric transmission at each frequency. The cyan filled regions indicate the mask channels containing emission lines that are identified in the entire map, and red thick bars indicate the channels masked by a "deviation mask" algorithm, designed to exclude atmospheric lines and lines at the band edge from the baseline fit.

Detailed plots of the spectra can be seen in the detail pages, which can be opened by clicking the Spectral Window link of each summary page. In the detail pages the spectral maps of all the antennas are shown. In the upper part of the detail pages there are boxes that can be used to set filters to plot spectral maps by antenna, field, spectral window and polarization.

Fitting order determination

Pipeline performs baseline fitting using a cubic spline that connects an empirically defined number of segments of each spectrum.

In a first step, the spectra are grouped in space and time domains. Pipeline groups the spectra that were observed close in time and position with respect to each other. Subsequently, Pipeline analyzes each (emission-masked) spectrum through Fast Fourier Transform (FFT) to obtain the power spectra. Note that input for the discrete Fourier Transform is (spectrum-average)*flag, where "flag" is set to zero for the emission-masked channels, while set to 1 for other channels. The power spectra of all integrations in a group are summed together and divided by its average value (averaging over frequency channels). Based on the peak value of the normalized Fourier spectra (P_FFT), the number of segments for cubic spline fitting ($N\_segment$) is defined empirically:

Figure 52: An example of the summary page of hsd_baseline. First three rows are shown.

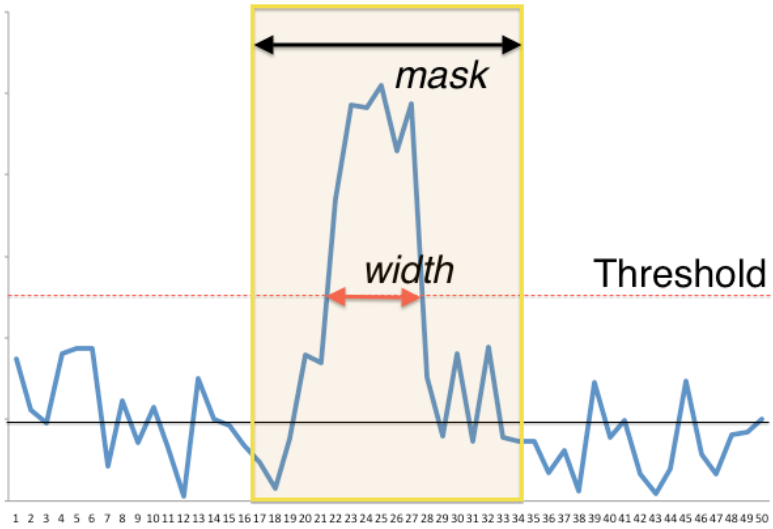


Figure 53: An example spectrum to explain Pipeline-defined mask range. "Width" corresponds to the number of channels where the emission is above the threshold.

1. If $1<P\_FFT<3$, then $N\_segment=3$,
2. If $3\leqq P\_FFT<5$, then $N\_segment=4$,
3. If $5\leqq P\_FFT<10$, then $N\_segment=5$,
4. If $P\_FFT\geqq 10$, then $N\_segment=F\_FFT\times 2+1$, where F_FFT is the frequency corresponding to the peak P_FFT.

In a second step, in order to take into account the proportion of masked channels, the obtained $N\_segment$ is newly defined as $N\_segment\times(Nch\text{-}N(mask))/Nch$, where Nch is the number of spectral channel and $N(mask)$ is the number of channels masked. Note that unmasked channels are equally divided into segments, i.e., number of unmasked channels is same for all segments. Finally, the Pipeline performs the baseline fitting and baseline subtraction using cubic splines, which are third order polynomial that meet the boundary condition at the joint between the segments.

Mask range determination

When baseline fitting is performed, the emission channel range is masked out. The mask range is determined by following equation:

```
mask = (maxW-width*(2*minW+10)+(width-minW)*maxW) / (maxW-minW)
```

where `minW` is minimum channel width, `maxW` is maximum channel width, *width* is channel width where the emission is above a threshold with at least five adjacent channels (see Figure 53). The values of `minW` and `maxW` are empirically determined and set to 2 and 500, respectively. Finally, the range of central channelmask divided by 2 will be masked out. In this definition, the relatively narrow channel width will have slightly larger mask range. For example, if `width`=5 channels, the mask range will be 8 channels at the central channel. For `width`=500 channels, the mask range will be 250 channels at the central channel.

Evaluation of baseline flatness

In the last fourth row of this page, pipeline evaluates the flatness of the baselines per field, per MS, per spw, per antenna, and per polarization. Emission-free channels are divided into 10 or 20 bins. QA score is calculated by the following criterion related to RMS, MAX(mean) and MIN(mean), where RMS, "mean", MAX(mean), and MIN(mean) are the rms estimated from emission-free channels, the mean in each bin and the maximum and minimum of "mean" among bins. Figure 54 shows an example of the baseline flatness. QA score for THE criterion is below.

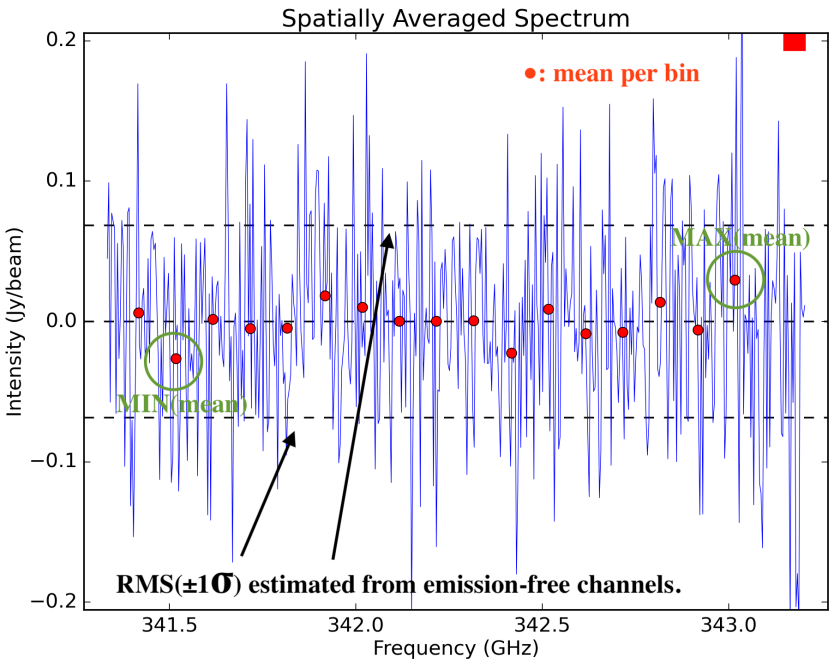


Figure 54: An example of the baseline flatness evaluation.

- 0.33 if MAX(mean) - MIN(mean) $> 3.6\sigma$
- 0.33–1.0 if $1.8\sigma \leq$ MAX(mean) - MIN(mean) $< 3.6\sigma$
- 1.0 if MAX(mean) - MIN(mean) $\leq 1.8\sigma$

Line identification

The plots related to line identification, which are basically useful for developers, are not displayed in the single dish pipeline as default. The users are able to display these plots by setting `plotlevel`='all' in `h_init`. The information summarized below describes the plots disabled in the single dish pipeline, i.e., plots for the line identification heuristics. However, it is useful to understand how line identification happens in the pipeline.

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 55a) 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.

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 55b) shows this ratio for each grid cell, i.e., the grid cell is marked as:

- "Validated" if Nmember/Nspectra > 0.5 (Blue squares in Fig. 55b)
- "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

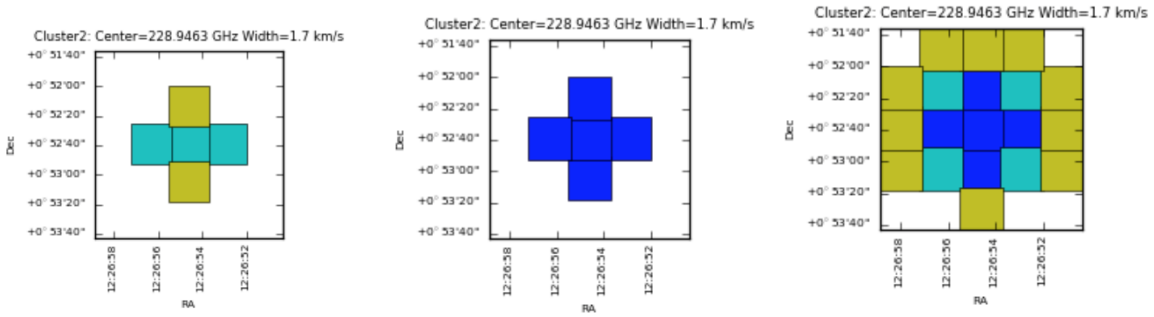


Figure 55: Examples of (a) clustering_detection, (b) clustering_validation, and (c)clustering_smoothing

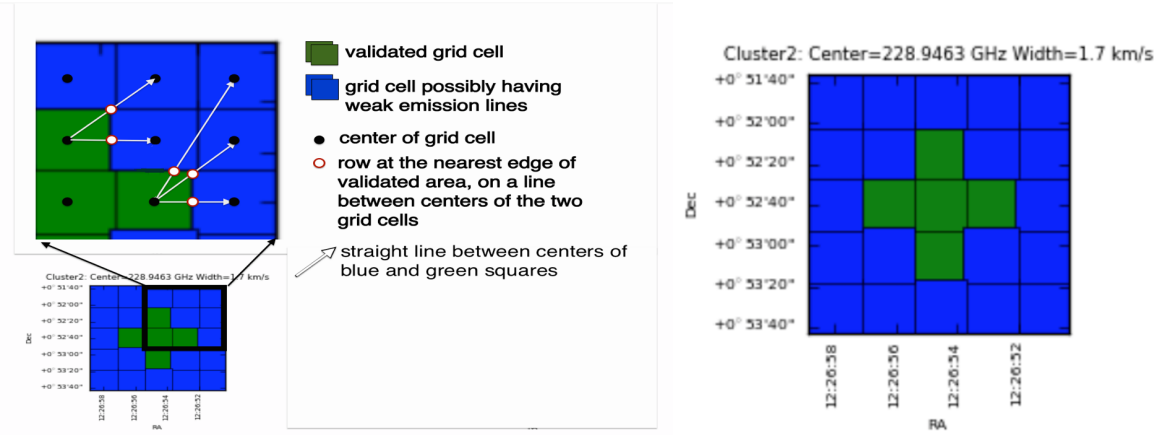


Figure 56: (a) An example of how the mask range is calculated. In the blue squares, the mask channel range is the range obtained at the nearest edge of any validated area by interpolating mask channel ranges in the valid grid cells (white-filled red circle). (b) An example of clustering_final.

enhancing the grid cells with detected emission line in neighboring grid cells.

Figure 55c 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).

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 55c or green squares in Figure 56), 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 56a. An example of "clustering_final" is shown in Figure 56b.

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[2]. The small dots indicate spectra containing identified emission line. The red ovals show each clustering region with a size of the cluster radius.

[2]Starting with the Cycle 7 pipeline, the hierarchy algorithm is adopted instead of kmean. The hierarchy algorithm tends to identify larger number of clusters with smaller radii compared with kmean. Please keep in mind this property when Cycle 7 results are compared with prior Cycles

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).[3] 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.

10.10 hsd_blflag

The WebLog shows the list of flagged data percentages using five criteria that are explained in the ALMA Pipeline Reference Manual. When you click on "Plots", you will get the detailed figures to evaluate these criteria. The flagged and unflagged data are shown in red and blue, respectively.

10.11 hsd_imaging

10.11.1 Image Sensitivity Table

The achieved sensitivity for the final cubes per Spw and Source as well as the theoretical RMS taken into account the flagging fraction are shown in the table. Note that all sensitivities are calculated with the native bandwidth from Cycle 10.

10.11.2 Profile Map

Figure 57 shows the top of the summary page. Three types of profile maps are available in the WebLog: 1) a 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. In the simplified profile map, the magenta lines indicate the atmospheric transmission at each frequency. One transmission profile is plotted for each ASDM processed. 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 $\frac{1}{8}$ 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. 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.

10.11.3 Channel Map

The number of channel maps per spw corresponds to the number of emission lines that have been identified by the clustering analysis of the pipeline. In each channel map (see Figure 58), 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 spectral range where line emission was detected, 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.

[3] Starting with the Cycle 7 pipeline, hierarchy algorithm is adopted instead of kmean so the current plots are the dummy.

Figure 57: An example of the profile map.

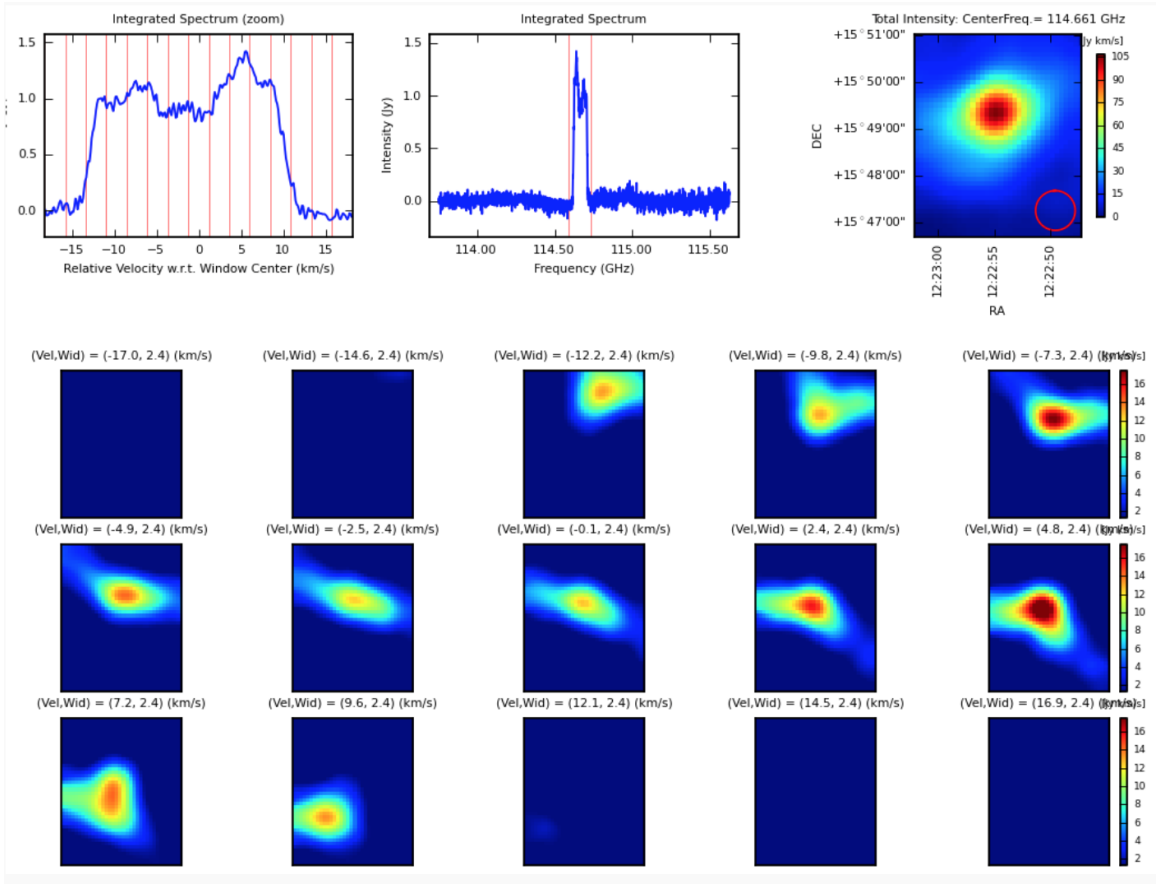


Figure 58: An example of a channel map.

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 **Max Intensity Map** (moment-8) for each spw is generated using immoments task with all the available channel range.

The **contamination plot** contains three plots of Peak SN map, mask map, and masked-averaged spectrum. The open circle in Peak SN map indicate the peak SN position. Mask map indicate the pixels that have a lower 10% SN ratio values. In masked-averaged spectrum, the spectrum in red indicates the spectrum averaged over the masked pixels, while the spectrum in grey indicates the spectrum at the peak SN position. If the negative peak of the masked-averaged spectrum is less than -4 * standard deviation, the pipeline gives a warning.

10.12 hsd_exportdata

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

11 Imaging weights in cubes

Since CASA 5.6, the calculation of imaging weights for cubes can be performed either per-channel or for all channels, according to the tclean parameter `perchanweightdensity`. This can have significant effects on the image produced. Users should be aware of these effects when creating new images, either using pipeline tasks or with tclean. For a detailed description, see https://casadocs.readthedocs.io/en/stable/notebooks/synthesis_imaging.html.

11.1 History of weighting parameter choices

- The tclean `perchanweightdensity` parameter was effectively False in CASA 5.4, the Cycle 6 pipeline, and all prior versions of CASA and pipeline (the parameter did not exist prior to CASA 5.5.0).
- As of CASA 5.6.0 (i.e. the version used for ALMA Cycle 7 data reduction), `perchanweightdensity=True` is the default in tclean.
- ALMA decided that the Cy7 and 2020.1 Imaging Pipelines would create cubes with `perchanweightdensity=False` (consistent with all previous versions of the imaging PL).
- For PL2021, PLWG developed the new `briggsbwtaper` weighting to be used with `perchanweightdensity=True`
- `briggsbwtaper` is only applicable to cube imaging, and `briggs` remains the default weighting scheme for mfs imaging.

	`weighting` default	`perchanweightdensity` default
CASA<5.6	natural	effectively False
C6 pipeline	briggs	effectively False
CASA≥5.6	natural	True
C7 Pipeline PL2020	briggs	False
≥PL2021	briggsbwtaper	True

11.2 Summary of the effects of weighting scheme choices

Different channels can span multiple cells in uv-space because of the frequency difference. This is the basis of multi-frequency-synthesis (mfs) continuum imaging, which takes advantage of this property to increase the *effective* uv-coverage. In CASA 5.5 onward, the `perchanweightdensity` parameter determines whether the imaging weights are calculated using only the (u,v) points for each channel of interest (`perchanweightdensity=True`), or using the points corresponding to all channels in the spw (`perchanweightdensity=False`) similar to an mfs continuum image.

- `perchanweightdensity=False` results in a systematic variation of the beam size across a spectral window, generally larger in the center, smaller on the edges.
- In general, cubes produced with `perchanweightdensity=False` will have higher noise on the edges than center of the spectral window, even after all channels are convolved to the same beam (as is standard for the Pipeline)
- In general, `perchanweightdensity=True` with `briggs` weighting results in a smaller dynamic range in uv density, and thus changing `briggs robust` will have less effect - one will find that all beams are larger for a given `robust` value, and the endpoint of Uniform weighting has changed to be larger than with `perchanweightdensity=False`.
- For a given `robust` value, `briggsbwtaper` weighting will recover with `perchanweightdensity=True` a similar beam to that achieved with `briggs` weighting and `perchanweightdensity=False` - i.e. the beam and noise are relatively constant across a spectral window, but reducing `robust` allows a significant reduction of the beam size.

- In addition, `briggsbwtaper` with `perchanweightdensity=True` results in a cube beam size very similar to the mfs beam size for the same `robust` value (so Pipeline continuum and line images will have similar beams).