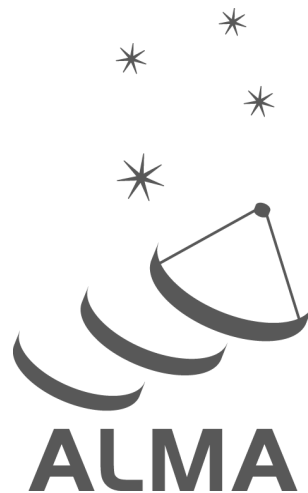


User Support:

ALMA Science Pipeline User's Guide for Release 2025.1.0.35, CASA 6.6.6-17, python3.10 Interferometric and Single-Dish Processing



www.almascience.org

ALMA, an international astronomy facility, is a partnership of ESO (representing its member states), NSF (USA) and NINS (Japan), together with NRC (Canada), NSC and ASIAA (Taiwan), and KASI (Republic of Korea), in cooperation with the Republic of Chile. The Joint ALMA Observatory is operated by ESO, AUI/NRAO and NAOJ.

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
2024.1v1.0 CASA 6.6.1	October 2024	Pipeline Team
2025.1v1.0 CASA 6.6.6	October 2025	Pipeline Team

In publications, please refer to this document as:
ALMA Pipeline Team, 2025, ALMA Science Pipeline User’s Guide, ALMA Doc 2025v1.0



Contents

1	The ALMA Science Pipeline	6
1.1	Purpose of this document	6
1.2	Pipeline Overview and Nomenclature	6
2	Quick Start	7
3	What's New in 2025.1	8
3.1	Major new capabilities	8
3.2	New features and improvements by WebLog section and processing stage.	8
3.3	Current Known Limitations of the 2025 Pipeline	11
4	Pipeline Versions & Documentation	14
4.1	Obtaining the Pipeline	14
4.2	Pipeline-related Documentation	14
4.3	Pipeline and CASA Versions	14
4.4	Pipeline and CASA tasks	14
5	Data Processing Files	17
5.1	Archived scripts	17
5.2	Pipeline “Helper” text files	17
5.3	The Pipeline script to restore calibrated MSs: <code>casa_piperestorescript.py</code>	18
5.4	The Pipeline processing script: <code>casa_pipescript.py</code>	19
5.5	CASA equivalent commands file: <code>casa_commands.log</code>	22
6	Modifying a Pipeline Run using <code>casa_pipescript.py</code>	24
6.1	Pipeline re-processing considerations	24
6.2	Preparing to run <code>casa_pipescript.py</code>	24
6.3	Modifying Calibration Commands	25
6.4	Modifying IF Pipeline Imaging Commands	25
6.5	Manual imaging after running <code>casa_pipescript.py</code>	26
6.6	The Pipeline Context	26
7	Description of Pipeline “Helper” Text Files	28
7.1	<code>flux.csv</code> (IF Pipeline)	28
7.2	<code>jyperk.csv</code> or <code>jyperk_query.csv</code> (SD pipeline)	28
7.3	<code>uid*antennapos.json</code> or <code>antennapos.csv</code> (IF pipeline)	30
7.4	<code>uid*flagtemplate.txt</code> & <code>uid*flagssystemtemplate.txt</code> (both pipelines)	31
7.5	<code>uid*flagtargetstemplate.txt</code> (IF imaging pipeline)	31
7.6	<code>cont.dat</code> (IF imaging pipeline)	32
8	The Pipeline WebLog	34
8.1	Overview	34
8.2	Navigation	34
8.3	Home Page	34
8.4	By Topic Summary Page	36
8.5	By Task Summary Page	39
8.6	Task Pages	39
8.7	WebLog Quality Assessment (QA) Scoring	44

9	Interferometric pipeline tasks and “By Task” WebLog pages	45
9.1	hifa_importdata	45
9.2	hifa_flagdata	45
9.3	hifa_fluxcalflag	46
9.4	hif_rawflagchans	47
9.5	hif_refant	48
9.6	h_tsyscal	48
9.7	hifa_tsysflag	49
9.8	hifa_tsysflagcontamination (standard IF recipes only, not diffgain or full-pol)	50
9.9	hifa_antpos	52
9.10	hifa_wvrgcalflag	52
9.11	hif_lowgainflag	52
9.12	hif_setmodels	54
9.13	hifa_bandpassflag	54
9.14	hifa_bandpass	54
9.15	hifa_spwphaseup	56
9.16	hifa_gfluxscaleflag	62
9.17	hifa_polcalflag (polarization recipes only)	62
9.18	hifa_session_refant (polarization recipes only)	62
9.19	hifa_lock_refant (polarization recipes only)	63
9.20	hifa_gfluxscale	63
9.21	hifa_diffgaincal (diffgain recipes only)	67
9.22	hifa_timegaincal	68
9.23	hifa_renorm	72
9.24	hifa_targetflag	76
9.25	hifa_polcal (polarization recipes only)	76
9.26	hif_applycal	77
9.27	hif_makeimlist: Set-up parameters for calibrator images	78
9.28	hif_makeimages: Make calibrator images	78
9.29	hif_makeimlist: Set-up parameters for polarization calibrator imaging (polarization recipes only)	79
9.30	hif_makeimages: Make polarization calibrator images (polarization recipes only)	79
9.31	hif_makeimlist: Set-up parameters for check source images	79
9.32	hif_makeimages: Make check source images (if check source present)	81
9.33	hifa_imageprecheck	81
9.34	hif_checkproductsizes: Mitigation to avoid overly long runs and/or tclean failures	82
9.35	hifa_exportdata	86
9.36	hif_mstransform	87
9.37	hifa_flagtargets	87
9.38	hif_makeimlist: Set-up parameters for target per-spw continuum imaging	87
9.39	hif_findcont	87
9.40	hif_uvcontsub	90
9.41	hif_makeimages: common task functionality	90
9.42	hif_makeimages: Make target per-spw continuum images	95
9.43	hif_makeimlist: Set-up parameters for target aggregate continuum images	96
9.44	hif_makeimages: Make target aggregate continuum images	96
9.45	hif_makeimlist: Set-up image parameters for target cube imaging	96
9.46	hif_makeimages: Make target cubes	96
9.47	hif_makeimlist: Set-up image parameters for representative bandwidth target cube	97
9.48	hif_makeimages: Make representative bandwidth target cube	99
9.49	hif_selfcal	100
9.50	hifa_exportdata	104
9.51	hifa_restoreddata	104

10 Single Dish pipeline tasks and WebLog pages	105
10.1 hsd_importdata	105
10.2 hsd_flagdata	105
10.3 h_tsyscal	106
10.4 hsd_tsysflag	106
10.5 hsd_skycal	106
10.6 hsd_k2jycal	107
10.7 hsd_applycal	107
10.8 hsd_atmcor	107
10.9 hsd_baseline	108
10.10 hsd_blflag	112
10.11 hsd_imaging	112
10.12 hsd_exportdata	116
10.13 Single-Dish Pipeline QA scores	116
11 Imaging weights in cubes	119
11.1 History of weighting parameter choices	119
11.2 Summary of the effects of weighting scheme choices	119

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 12 version (2025.1.0) of the ALMA Pipeline that is packaged with a CASA 6.6.6 Python 3.10 monolithic release. Version “casa-6.6.6-17-pipeline-2025.1.0.35” was deployed for use in ALMA Operations in October 2025. This version is labeled in the WebLog as Pipeline Version 2025.1.0.35 with CASA Version 6.6.6-17. The versions of the major included packages are: **Astropy**: 6.1.7, **NumPy**: 2.0.1, **Matplotlib**: 3.9.2, **SciPy**: 1.14.1 (see Table 1 in § 4.4 for a more complete list of version changes in third party packages between PL2024 and PL2025).

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 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 (e.g. VLBI and Solar) are, as a rule, processed outside the pipeline, using manually modified Common Astronomy Software Applications ([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 officially 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 §9.34); 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 open access refereed journal article “*The ALMA Interferometric Pipeline Heuristics*”, Hunter et al. 2023, [PASP, volume 135, id 074501](#). That paper contains additional details on the IF pipeline heuristics that are not included in this user’s guide.

3 What’s New in 2025.1

3.1 Major new capabilities

- The calibration of data with faint (low SNR) calibrator sources, which often includes high frequency, long baseline, and Band-to-band Interferometry cases, has been extended to handle more severe cases that previously failed to calibrate.
- Self-calibration has been extended to cover mosaic targets, and its heuristics for long baseline data have been improved.
- IQUV science target imaging capabilities have been introduced and added to the polarization recipes.
- The locations on the sky of atmospheric calibration scans are now correctly displayed on Spatial Setup page of the weblog.
- QA scores have been improved for [hifa_bandpass](#) and the Amp/Ph-Freq component of [hif_applycal](#).
- New QA have been developed in SDPL for calibrated amplitude vs time, and for residual emission in line-free channels of images.
- New plots of Jy/K as a function of time, and amplitude vs time have been added to SDPL.

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

3.2.1 General features

- **Main overview (per-EB) pages:**
 - ObservingMode is displayed per session (Fig. 13)
 - On the Spatial Setup Details page for mosaics, the Mosaic Pointings plot shows the location of the target Tsys “OFF” scans with a red dotted circle labeled with the scan number.
 - On the Spectral Setup Details page, the spw name column has been added to the “All windows” tab.
- **CASA log messages:**
 - In [hifa_importdata](#), default values have been defined for `FLUX_SERVICE_URL` and `FLUX_SERVICE_BACKUP`, which will be used when the corresponding environment variables are undefined: “Switching to default <https://almascience.org/sc/flux>” and “Switching to default <https://asa.alma.cl.sc/flux>”.
 - In [hifa_importdata](#), a log message is written if the MOUS “contains EBs with mixed spw naming conventions. Spw names may not exactly match across EBs.”
 - In [hifa_importdata](#), INFO messages have been added for “Calculating offset(s) for intent CALIBRATE_ATMOSPHERE#OFF_SOURCE” with subsequent lines indicating the “Median offset” in azimuth and elevation and the subsequent “Tsys Field radec” direction.

3.2.2 New features and fixes specific to the interferometric pipeline

- [hifa_importdata](#): (§ 9.1)
 - A poor QA score results if the flux of any non-SSO amplitude calibrator (for any spw) is from the ASDM Source.xml file rather than a more recent Source Catalog query.
- [hifa_antpos](#): (§ 9.9)
 - Antenna positions are now queried from an online database rather than by supplying a CSV file with corrections to apply. The functionality to use a provided CSV file remains.
- [hifa_bandpass](#): (§ 9.14)

- The low SNR heuristics now allow the ‘phase=up’ pre-bandpass solution to combine spectral windows before increasing the solution interval in the process of improving solution SNR.
- For FDM spectral windows and the baseline correlator (BLC), a new QA score evaluates whether the bandpass tables have evidence of sub-band issues.
- [hifa_spwphaseup](#): (§ 9.15)
 - Modifications were made to use combine=‘spw’ for all INTENTS (except the Polarization calibrator) if required: In some cases, the BANDPASS, AMPLITUDE and/or DIFFGAIN (for Band-to-Band (B2B) data) may not be strong enough for a per-spw ‘int’ timescale phase-up. If this is the case, spectral window combination can be triggered before also combining polarizations and extending the time solution interval.
- [hifa_diffgaincal](#): (§ 9.21)
 - Modifications were made as part of the low SNR heuristics to allow spectral window combination for all parts of the process, ‘reference’, ‘B2B-offset’, and ‘residual’ solve processes if the sources are deemed too weak.
- [hif_applycal](#): (§ 9.26)
 - The QA score that identifies amplitude-frequency or phase-frequency outliers no longer results in a fixed QA score of 0.9, but instead a score that gets lower based on how significant the outlier is, except for: amplitude-frequency offsets that are anomalously large but symmetric, or phase-frequency offsets for CHECK sources.
- [hif_checkproductsizes](#) (§ 9.34):
 - The stage was adapted to account for additional product size in IQUV full polarization imaging recipes.
- [hif_findcont](#) (§ 9.39):
 - The moment difference SNR value is now displayed in the weblog table. Values less than about 10 generally mean minimal line contamination in the selected channel ranges.
 - The "AllCont" label has been added to the spectral plot.
 - When a cube of all zeros is encountered, it will proceed to the next cube.
 - Improvements to findContinuum.py:
 - * Prevent erroneous AllCont status.
 - * Prevent undefined variable error for jointMaskTmp.
 - * Keep up with syntax changes in NumPy.
 - * Fix deprecation warnings.
- [hif_makeimlist](#) (§ 9.45):
 - Full polarization IQUV imaging has been added as a new capability. New stage variants are mfs_fullpol, cont_fullpol, cube_fullpol and cube_repBW_fullpol
- [hif_makeimages](#) (§ 9.46):
 - Spurious yellow warning messages in checksource imaging were fixed. (checksource [hif_makeimages](#))
 - Full polarization IQUV imaging has been added as a new capability (mfs, cont, cube, and repBW [hif_makeimages](#) stages). New stage variants are mfs_fullpol, cont_fullpol, cube_fullpol and cube_repBW_fullpol
 - Overlapping sky plot colorbar and labels were fixed (all [hif_makeimages](#) stages)
 - Backend math was changed to calculate polarization intensity using $\sqrt{Q^2 + U^2}$, with no V contribution (polarization calibrator [hif_makeimages](#) stage)
 - Plot rendering issues were fixed for DSB receiver cases (cube [hif_makeimages](#))
- [hif_selfcal](#) (§ 9.49):
 - New heuristics have been added to handle the self-calibration of ALMA mosaic data.

- Solints that fail due to the beam-size growing by more than 5% will invoke a new fallback mode that attempts to pass through longer-baselines with flagged calibration solutions uncalibrated and unflagged to preserve the benefits of good solutions on shorter baselines
- [hifa_exportdata](#) (§ 9.35):
 - The stage was adapted to account for additional products generated in IQUV full polarization imaging recipes.
- **AQUA report:**
 - Single-dish pipeline reports observed and theoretical rms as `ObservedSensitivityJyPerBeam` and `TheoreticalSensitivityJyPerBeam` in AQUA report.
 - *Deprecation Warning: Both SD and IF now report ObservedSensitivityJyPerBeam and TheoreticalSensitivityJyPerBeam. In PL2026, the ambiguous and now redundant "SensitivityJyPerBeam" will be removed.*
 - Sensitivity entries for IF tasks such as [hif_makeimages](#) and [hifa_imageprecheck](#) now use the correct XML tag.

3.2.3 New features specific to the single-dish pipeline

- [hsd_flagdata](#) (§ 10.2):
 - A new function to flag outlier pointing data has been implemented. The correct map size is set, which prevents the pipeline from crashing. New QA scoring has been implemented accordingly.
- [hsd_skycal](#) (§ 10.5):
 - The plot of ‘elevation difference vs time’ excludes flagged data, which caused an undesired scale of the display.
 - The plot of ‘time vs interval’ has been removed.
- [hsd_k2jycal](#) (§ 10.6):
 - The plot of ‘summary of Jy/K conversion factor’ has been improved. The new plot shows frequency (SPW ID) vs Jy/K conversion factor. A scatter plot appears in MSs with less than 5 EBs, and a box plot is created for MSs with 5 EBs or more.
- [hsd_applycal](#) (§ 10.7):
 - New QA scoring has been implemented to assess an amplitude difference between the two polarizations. A new diagnostic plot of ‘amplitude difference (XX-YY) vs frequency’ has been implemented accordingly.
 - A new plot of ‘calibrated amplitude vs time’ has been implemented to check the time variance of amplitude caused by unavoidable reasons (*e.g.*, weather conditions, instrumental problems)
- [hsd_atmcor](#) (§ 10.8):
 - The latest atmospheric model has been applied.
- [hsd_baseline](#) (§ 10.9):
 - A new capability to set [fitfunc](#) and [fitorder](#) for each spectral window has been implemented.
 - New QA scoring has been implemented to remove unnecessary warnings. The QA messages have been updated accordingly.
 - Frequency reference frame (TOPO) has been added in the spectral figures.
- [hsd_imaging](#) (§ 10.11):
 - New QA scoring has been implemented to check for potential lines that have not been detected at the [hsd_baseline](#) stage. If potential lines are detected, a new diagnostic plot appears on this page.
 - QA scoring for a significantly small percentage of mask pixels has been improved. The warning messages have been updated accordingly.
 - Frequency reference frame (LSRK) has been added in the spectral figures.
 - Bugs in the calculation of Theoretical RMS have been fixed.

Interferometric Observing modes supported by ALMA 2025 Pipeline

*Single field and/or pointed mosaic with dual-polarization correlation products 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 (BP) calibrator 1 Flux calibrator (can be BP) 1 (optional) Check source	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, < 2 hr)
Atmospheric calibration of each frequency range	Required on: (BP/Flux calibrator) and (Phase calibrator and/or one Science target). Spw may differ but must be same source ID .	Same as 1 SpectralSpec
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., XY and YX correlation products)	Supported, including imaging. Single pol. (XX or YY) is also supported, but flux scale will be compromised if flux cal. is polarized.	Unknown: no test data provided
Different SpectralSpec used for Gain Calibrator vs. Science target	Supported for Band-to-Band transfer in dual-polarization (needed when there is no nearby strong phase calibrator). Not Supported for Bandwidth switching	Not supported A. BW-switching on multiple objects of diverse velocity B. B2B transfer on multiple objects of diverse velocity

*Notes: (1) Ephemeris objects are fully supported; VLBI and Solar are **not** supported

(2) Phase calibrator can be same object as Bandpass or Flux calibrator, but only if Tsys is recorded on at least 1 science target ⁴

Figure 1: Summary of supported observing modes in PL2025. Changes are indicated by the two red circles.

3.3 Current Known Limitations of the 2025 Pipeline

(none of these are new for 2025)

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 or slurm, 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.

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 ratio (SNR) heuristics, which have been further improved in PL2025, it remains the case that the calibration recipe 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 (for single field cases), and/or restricted to a subset of sources (see §9.34). Users can generate 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 determined 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 sometimes be obtained by more careful examination of individual sources and/or spectral windows. If the data are heavily binned in frequency either by the online system, or 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 (the SpectralWindow.xml file of the ASDM) to calculate the theoretical image sensitivity and hence the cleaning thresholds. This information was not correctly stored in raw ALMA data in Cycles 2 and earlier; as a result, the cleaning thresholds will be higher than intended when such data is run through the imaging pipeline (results will still be correct, merely not as deeply cleaned).
- The standard recipes used in operations in 2025 include self-calibration for single-field and mosaicked 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 [hif_selfcal](#) stage is removed from the recipe by a data reducer.
- 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 most optimized to detect moderate channel width emission lines at the center of a spectral window. It is, however, expected that better results can be obtained by more careful examination of individual sources and/or spectral windows. The following cases are most notably affected:
 - Very broad emission lines that potentially occupies a full spw, both in FDM and TDM modes.
 - 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.
- When strong emission is flagged in [hsd_bflag](#) due to unidentified line channels, this can increase the RMS at the corresponding spws. In operation, this issue can be resolved by manually redefining the line ranges in the corresponding spws and reprocess the data. However, in few extreme cases (e.g. Solar system objects observation), altering line ranges is not enough to resolve the issue. This issue can be circumvented by turning off the corresponding [hsd_bflag](#) flagging heuristics. If a user faces at situations shown above, we recommend to contact the [ALMA Helpdesk](#) for detailed advice.

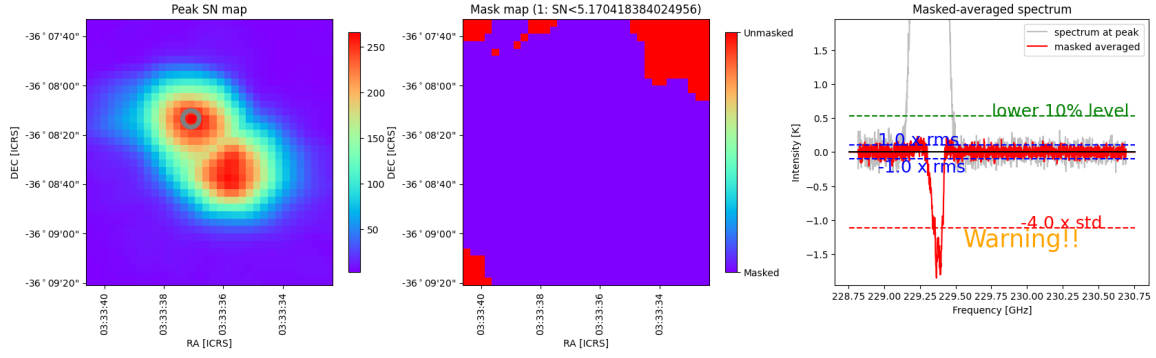


Figure 2: An example of the contamination plot for the case the OFF position is contaminated by the astronomical signal. The spectrum with the negative peak indicates that astronomical signal has been detected at the OFF position.

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, including back dating the scope of new issues to prior cycles.

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

An example of the pipeline processing for the total power data is available at: [https://casaguides.nrao.edu/index.php/Single-Dish_\(Total_Power\)_Data_Processing_with_Pipeline](https://casaguides.nrao.edu/index.php/Single-Dish_(Total_Power)_Data_Processing_with_Pipeline)

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 13 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.

Note that each CASA release often includes one or more updates to the third-party Python modules that are packaged with it. A summary of the changes in the CASA versions used between PL2023, PL2024, and PL2025 are shown in Table 1.

4.4 Pipeline and CASA tasks

The pipeline heuristics are written as python functions appearing with a `hif_` or `hifa_` (for interferometric) or `hisd_` (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 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, if run on the same hardware allocation (number of cores, memory). 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.<mouse_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>.

module	6.5.4py3.8 (PL2023)	6.6.1py3.8 (PL2024)	6.6.6py3.10 (PL2025)
almatasks ^a	1.6.1	1.7.1	–
Astropy	5.2.1	5.2.2	6.1.7
attrs	22.2.0	–	–
backcall	0.2.0	0.2.0	–
bdsf	1.10.2	1.10.3	1.13.0.post2
cachetools	5.2.0	5.3.1	5.5.2
casacconfig	–	–	1.1.1
certifi	2022.12.07	2023.07.22	2024.7.4
csscompressor	0.9.5	0.9.5	0.9.5
cycler	0.11.0	0.11.0	0.12.1
decorator	5.1.1	5.1.1	5.1.1
grpcio	1.29.0	1.26.0	1.66.0
intervaltree	3.1.0	3.1.0	3.1.0
ipython	7.15.0	7.34.0	8.26.0
jedi	0.18.2	0.19.0	0.19.1
kiwisolver	1.4.4	1.4.5	1.4.5
logutils	0.3.5	0.3.5	0.3.5
mako	1.2.4	1.2.4	1.3.10
Matplotlib	3.3.3	3.5.0	3.9.2
mpi4py	3.1.3	3.1.5	3.1.5
NumPy	1.23.5	1.24.4	2.0.1
Open MPI	1.10.4	5.0.1	5.0.1
packaging	23.0	23.1	24.1
parso	0.8.3	0.8.3	0.8.4
pip	23.0.1	22.3.1	22.3.1
pluggy	1.0.0	1.3.0	1.5.0
prompt_toolkit	3.0.36	3.0.39	3.0.47
ps_mem	3.14	3.14	3.14
ptyprocess	0.8.0	0.7.0	0.7.0
Pygments	2.14.0	2.26.1	2.28.0
pyparsing	3.0.9	3.1.1	3.1.2
pypubsub	4.0.3	4.0.3	4.0.3
pytest	7.2.1	7.4.2	8.3.2
pytz	2022.7.1	2023.3.1	2024.1
SciPy	1.10.0	1.10.1	1.14.1
setuptools	56.0.0	70.1.1	65.5.0
traitlets	5.8.1	5.10.0	5.14.3
wcwidth	0.2.6	0.2.6	0.2.13
wheel	0.41.2	–	–

^acontained the CASA task [wvrgcal](#), which has been migrated to [casatasks.wvrgcal](#) in PL2025.

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

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** or **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**. If they are not present where the pipeline is run and the pipeline database query parameter (**dbservice**) is True, conversion factors are obtained via the database. If they are not present and **dbservice** is False, conversion factors of unity are assumed.
- **uid*antennapos.json** or **antennapos.csv**: **uid*antennapos.json** files are used by the IF pipeline to

update the positions of the antenna elements. The pipeline will, by default, query an online database to retrieve these files, but if already present in the working directory then the existing files will be used instead (e.g. if one were to want to manually control the antenna position corrections). Alternatively, an `antennapos.csv` file can be supplied in conjunction with `hm_antpos='file'` to provide antenna position corrections.

- **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*flagssystemtemplate.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`, `*auxproducts.tgz` and `*calapply.txt` to `products/`.
- Copy `uid*casa_piperestorescript.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 `hds_bflag` command will result in:

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

Running the script through the `hds_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 `hif_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 `hif_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

```

context = h_init()
context.set_state('ProjectStructure', 'recipe_name', 'hsd_calimage')
try:
    hsd_importdata(vis=['uid___A002_X85c183_X36f'], session=['default'])
    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()
    hsd_exportdata()
finally:
    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.

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 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.

```

context = h_init()
context.set_state('ProjectStructure', 'recipe_name', 'hifa_calimage')
try:
    hifa_importdata(vis=['uid__A002_Xc46ab2_X15ae'], session=['session_1'], dbservice=True) # use flux.csv
    hifa_flagdata() # uses *flagtemplate.txt
    hifa_fluxcalflag()
    hif_rawflagchans()
    hif_refant()
    h_tsyscal()
    hifa_tsysflag()
    hifa_tsysflagcontamination()
    hifa_antpos()
    hifa_wvrgcalflag()
    hif_lowgainflag()
    hif_setmodels()
    hifa_bandpassflag()
    hifa_bandpass()
    hifa_spwphaseup()
    hifa_gfluxscaleflag()
    hifa_gfluxscale()
    hifa_timegaincal()
    hifa_renorm(createcaltable=True, atm_auto_exclude=True)
    hifa_targetflag()
    hif_applycal()
    hif_makeimlist(intent='PHASE,BANDPASS,AMPLITUDE')
    hif_makeimages()
    hif_makeimlist(intent='CHECK', per_eb=True)
    hif_makeimages()
    hifa_imageprecheck()
    hif_checkproductsizes(maxcubesize=40.0, maxcubelimit=60.0, maxproductsizes=500.0)
    hifa_exportdata()
    hif_mstransform()
    hifa_flagtargets() # uses *flagtargettemplate.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='repBW')
    hif_makeimages()
    hif_selfcal() # uses cont.dat
    hif_makeimlist(specmode='mfs', datatype='selfcal') # uses cont.dat
    hif_makeimages() # uses cont.dat
    hif_makeimlist(specmode='cont', datatype='selfcal') # uses cont.dat
    hif_makeimages() # uses cont.dat
    hif_makeimlist(specmode='cube', datatype='best')
    hif_makeimages()
    hif_makeimlist(specmode='repBW', datatype='selfcal')
    hif_makeimages()
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, atmospheric-corrected MS for each ASDM with a name like `uid___A00X_XXXX_XXX.ms.atmcor.atmtype1`
- A calibrated, atmospheric-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 (it is already gone for MacOS), 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 §9.34).

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 13, or the **Task Execution Statistics** that are listed for each task in the “By Task” part of the WebLog – see e.g. Figure 16. 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 ≥ 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` or `antennapos.json` (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*` are likely to be found in `uid*auxproducts.tgz` which will need to be unzipped. If using the older `antennapos.csv`, the `hifa_antpos()` call should set `hm_antpos='file'` and `antposfile='antennapos.csv'`. If json antenna positions are present, `hifa_antpos()` needs no input parameters set.

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` or `jyperk_query.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 [hisd_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 §9.41 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 signal-to-noise ratio (SNR) of cubes.
- Desire to use a different Briggs **robust** image weighting than the default value of **robust=0.5** (smaller value = smaller beam, poorer SNR; larger value = larger beam, better SNR).
- Desire to uv-taper images to increase the SNR 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, `tsdimaging`, 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 `tsdimaging` command will make images of all MS that are specified in the `infile`s parameter. For other parameters in `tsdimaging`, refer to the `*casa_commands.log` file.

Note that the images included in the delivery package have the native frequency resolution, and the 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_X89252c_X852.ms'),
```

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 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` (the default for production pipeline runs), 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 §9.20).

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` or `jyperk_query.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,uvmin,uvmax,comment
uid__A002_Xca8fbf_X5733.ms,0,23,1.7632620185868495,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,25,1.7450149079637087,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,27,1.7448343973502576,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,29,1.7482019091989398,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,31,1.7474672794635093,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,33,1.766271050283792,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=AM...
uid__A002_Xca8fbf_X5733.ms,0,35,1.7656450126668404,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,0,37,1.7648329718913716,0.0,0.0,0.0,-0.285,0.0,0.0,"# field=J1517-2422 intents=A...
uid__A002_Xca8fbf_X5733.ms,1,25,0.16623488895935037,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,27,0.1662065981594399,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=A...
uid__A002_Xca8fbf_X5733.ms,1,29,0.16673468749683268,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,31,0.16661942767873175,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,33,0.16957947525989395,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,35,0.16948059703943796,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,37,0.16935237463260322,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...
uid__A002_Xca8fbf_X5733.ms,1,23,0.16910442689668115,0.0,0.0,0.0,-0.468,0.0,0.0,"# field=J1532-1319 intents=...

```

Figure 5: Example of a **flux.csv** file used by the interferometric pipeline (one per MOUS)

```

MS,Antenna,Spwid,Polarization,Factor
uid__A002_X85c183_X36f.ms,DA61,17,I,51.890035198
uid__A002_X85c183_X36f.ms,PM03,17,I,51.890035198
uid__A002_X85c183_X36f.ms,PM04,17,I,51.890035198
uid__A002_X85c183_X60b.ms,DA61,17,I,51.890035198
uid__A002_X85c183_X60b.ms,PM03,17,I,51.890035198
uid__A002_X85c183_X60b.ms,PM04,17,I,51.890035198
uid__A002_X8602fa_X2ab.ms,PM02,17,I,51.4634859397
uid__A002_X8602fa_X2ab.ms,PM03,17,I,51.4634859397
uid__A002_X8602fa_X2ab.ms,PM04,17,I,51.4634859397
uid__A002_X8602fa_X577.ms,PM02,17,I,51.4634859397
uid__A002_X8602fa_X577.ms,PM03,17,I,51.4634859397
uid__A002_X8602fa_X577.ms,PM04,17,I,51.4634859397
uid__A002_X864236_X2d4.ms,PM03,17,I,49.437094842
uid__A002_X864236_X2d4.ms,PM04,17,I,49.437094842
uid__A002_X864236_X693.ms,PM03,17,I,49.437094842
uid__A002_X864236_X693.ms,PM04,17,I,49.437094842
uid__A002_X864236_X693.ms,DV10,17,I,49.437094842
uid__A002_X86fcfa_X664.ms,PM03,17,I,54.5385894402
uid__A002_X86fcfa_X664.ms,PM04,17,I,54.5385894402
uid__A002_X86fcfa_X664.ms,DV10,17,I,54.5385894402
uid__A002_X86fcfa_X96c.ms,PM03,17,I,54.5385894402
uid__A002_X86fcfa_X96c.ms,PM04,17,I,54.5385894402
...

```

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

```

{
  "data": {
    "CM05": [2225063.5461666198, -5440128.205155838, -2481550.079322571],
    "CM04": [2225074.067549907, -5440115.249648992, -2481568.944019133],
    "CM03": [2225076.734462043, -5440122.932430074, -2481549.8243493875],
    "CM02": [2225070.957887418, -5440127.671403772, -2481544.6552856914],
    "CM01": [2225080.354745021, -5440132.956785359, -2481524.789613484],
    "CM12": [2225089.438089277, -5440119.773602419, -2481545.39917787],
    "CM11": [2225065.4337890167, -5440120.433856066, -2481565.3136376073],
    "CM10": [2225078.9290363584, -5440126.085189326, -2481541.0211708946],
    "CM08": [2225063.5325220255, -5440134.134238726, -2481537.201932615],
    "CM07": [2225090.9997122493, -5440126.601164151, -2481529.1336664273],
    "CM06": [2225084.2403706782, -5440114.998424933, -2481560.411721414]
  },
  "metadata": {
    "caltype": "ALMA antenna positions",
    "description": "ALMA ITRF antenna positions in meters",
    "product_code": "antposalma",
    "outfile": "uid__A002_Xe1f219_X9dbf.antennapos.json",
    "hosts": [
      "https://asa.alma.cl/uncertainties-service/uncertainties/versions/last/measurements/casa/"
    ],
    "asdm": "uid://A002/Xe1f219/X9dbf",
    "search": "auto",
    "successful_url": "https://asa.alma.cl/uncertainties-service/uncertainties"
      "/versions/last/measurements/casa//?asdm=uid%3A%2F%2FA002%2FXe1f219%2FX9dbf&search=auto",
    "timestamp": "2025-08-25 18:22:13.785714"
  }
}

```

Figure 7: Example of a **uid*antennapos.json** file used by the interferometric pipeline (one per EB). File lists the total positions for each antenna, which are converted into corrections to be applied to the values written into the MS by the **gencal** task.

7.3 uid*antennapos.json or 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, the **hifa_antpos** task will query the online database to get the best-available antenna positions at the time the pipeline is run. These are written into **uid*antennapos.json** text files, which is then read in by the **gencal** task and used to correct the values in the ASDM.

The format of the **uid*antennapos.json** file is shown in Figure 7. It contains a dictionary entry for every antenna in an EB with the total position of that antenna. Though a user need not supply such a file, as **hifa_antpos** will automatically query the service for it, if these files are present in the directory where the pipeline is run a priori, the existing files will be used in lieu of querying the online database.

Alternatively, for datasets originally processed through the 2024 pipeline or earlier, a **antennapos.csv** file may be present in the ***auxproducts.tgz** from the original run. If this file exists in the directory where the pipeline is run, and the **hifa_antpos** task iparameters are set to **hm_antpos='file'** and **antposfile='antennapos.csv'**, the

```

name,antenna,xoff,yoff,zoff,comment
uid___A002_Xca8fbf_X5733.ms,DA41,-5.29597e-06,-1.16080e-05,-1.60051e-04,
uid___A002_Xca8fbf_X5733.ms,DA42,-8.69576e-06,-2.61175e-04,-8.79318e-05,
uid___A002_Xca8fbf_X5733.ms,DA43,-9.54466e-06,-1.70737e-04,-1.47686e-04,
uid___A002_Xca8fbf_X5733.ms,DA44,1.88754e-04,-2.52803e-04,-7.59289e-05,
uid___A002_Xca8fbf_X5733.ms,DA46,-2.86531e-04,-4.77569e-04,-4.69737e-04,
uid___A002_Xca8fbf_X5733.ms,DA47,-2.63745e-05,-2.20798e-04,-3.75155e-04,
uid___A002_Xca8fbf_X5733.ms,DA49,-3.67966e-05,-3.40138e-05,-1.81810e-04,
uid___A002_Xca8fbf_X5733.ms,DA50,4.44967e-05,-6.21555e-05,-3.29943e-04,
uid___A002_Xca8fbf_X5733.ms,DA51,-7.00033e-05,-1.77003e-04,-2.09113e-04,
uid___A002_Xca8fbf_X5733.ms,DA52,-7.74823e-05,-1.45007e-05,-1.21235e-04,
uid___A002_Xca8fbf_X5733.ms,DA53,1.95067e-04,-4.52641e-04,-5.08577e-05,
uid___A002_Xca8fbf_X5733.ms,DA54,-1.47680e-04,2.42910e-04,1.14313e-04,
uid___A002_Xca8fbf_X5733.ms,DA55,-1.62264e-04,2.22735e-05,-3.31404e-04,
uid___A002_Xca8fbf_X5733.ms,DA58,2.10611e-04,-3.27433e-04,-7.80367e-05,
uid___A002_Xca8fbf_X5733.ms,DA59,4.70094e-05,-5.74067e-05,-1.44642e-04,
uid___A002_Xca8fbf_X5733.ms,DA60,1.71910e-04,4.06828e-04,-4.65860e-04,
uid___A002_Xca8fbf_X5733.ms,DA62,-8.71466e-05,7.18571e-05,-1.87562e-04,
uid___A002_Xca8fbf_X5733.ms,DA64,-6.97952e-05,-9.69870e-05,-1.40777e-04,
uid___A002_Xca8fbf_X5733.ms,DA65,-1.25389e-05,-7.79228e-05,-1.55111e-04,
uid___A002_Xca8fbf_X5733.ms,DV01,4.04226e-04,-6.00860e-04,-2.94583e-04,
uid___A002_Xca8fbf_X5733.ms,DV02,3.10277e-04,-2.40413e-04,-3.90951e-04,
...

```

Figure 8: 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.

corrections it contains will be used in-lieu of the online database.

The format of the **antennapos.csv** file is shown in Figure 8. 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*flagstemplate.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 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 9).

These flag files will be picked up by the [hifa_flagdata](#)/[hds_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 ***.flagstemplate.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

```

#
# 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' field='1' timerange='2018/02/10/00:01:01.0959~2018/02/10/00:01:01.0961' reason='QA2:t...
#
# 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' spw='25:230.513~230.525GHz,33:220.380~220.386GHz' field='J0542-0913' reason='QA2:applycal_amp...'

```

Figure 9: Example of a `uid*flagtemplate.txt` file used by both the interferometric and single-dish pipeline (one per ASDM)

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. *The format of `cont.dat` changed between PL2023 and PL2024. PL2024 and later can read the earlier format, but not vice versa.* An example `cont.dat` file is shown in Figure 10.

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 exists, but is not followed by any ranges, `hif_findcont` will not run its heuristics on the spw (if the delivered `cont.dat` file contains spw entries without ranges, this indicates that the `hif_findcont` task failed to find any continuum frequency ranges). The task `hif_uvcontsub` will currently assume this is an all continuum case and fit using all channels. This is supposed to be changed in future pipeline versions to skip fitting and writing the spw to the line cube. 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 a continuum subtraction performed using all channels which may offset any line emission.

```

Field: hh666

SpectralWindow: 19 X1913589666#ALMA_RB_06#BB_2#SW-01#FULL_RES

SpectralWindow: 25 X1913589666#ALMA_RB_06#BB_1#SW-01#FULL_RES
Flags: ALLCONT
219.3452433728~219.8069245935GHz LSRK

SpectralWindow: 29 X1913589666#ALMA_RB_06#BB_4#SW-01#FULL_RES
230.0866113521~230.0958888260GHz LSRK
230.1017482832~230.1100491809GHz LSRK
230.2062419366~230.4933553394GHz LSRK
230.5939426880~231.0177675888GHz LSRK

```

Figure 10: Example of a **cont.dat** file used by the interferometric pipeline (one per MOUS). This example is for an MOUS that has 4 spectral windows; the entry for spw 19 is empty and spw 27 is omitted, which will result in the [hif_findcont](#) task determining the frequency ranges for these spectral windows.

Cube imaging "FC" moment map computation will fail due to the missing continuum ranges.

3. If a spw 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 through.

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 11 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 12). 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; see Figure 14). Clicking on the bar at the top of the home page (see Figure 13) 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.

[Home](#)
[By Topic](#)
[By Task](#)

Project Code N/A

Tasks in execution order

- hifa_importdata
- hifa_flagdata
- hifa_fluxcalflag
- hif_rawflagchans
- hif_refant
- h_tsyscal
- hifa_tsysflag

Error: cannot load content

Browser security prevents the weblog from displaying the requested content.

Viewing the web log locally is not possible with your web browser security settings. To view the web log you must serve the web log via HTTP by using `h_weblog` from inside a CASA session, or relax your web browser security.

Solutions

Recommended: use `h_weblog()`

From inside a CASA session, navigate to the root of the untarred weblog directory, e.g., `pipeline-procedure_hifa_calimage`, and run `h_weblog`. This command will serve the web log via HTTP and launch a browser connecting to the web log. The web log URL is also printed to the CASA logger, should you need to navigate to the web log manually. The URL to access is highlighted in the example CASA logger output below.

```

CASA <S>: h_weblog()
2020-07-30 12:57:20 INFO h_weblog:::casa #####
2020-07-30 12:57:20 INFO h_weblog:::casa #### Begin Task: h_weblog ####
2020-07-30 12:57:20 INFO h_weblog:::casa h_weblog( pipelinemod="automatic", relpath="" )
2020-07-30 12:57:20 INFO h_weblog:::pipeline::casa Found weblogs at:
2020-07-30 12:57:20 INFO h_weblog:::pipeline::casa+ main/pipeline-procedure_hifa_calimage/html/t1-1.html
2020-07-30 12:57:20 INFO h_weblog:::pipeline::casa Using existing HTTP server at 127.0.0.1 port 30000 ...
2020-07-30 12:57:20 INFO h_weblog:::pipeline::casa Opening http://127.0.0.1:30000/main/pipeline-procedure_hifa_calimage/html/t1-1.html
2020-07-30 12:57:20 INFO h_weblog:::casa Result h_weblog: None
2020-07-30 12:57:20 INFO h_weblog:::casa Task h_weblog complete. Start time: 2020-07-30 08:57:19.888720 End time: 2020-07-30 08:57:20.064263
2020-07-30 12:57:20 INFO h_weblog:::casa #### End Task: h_weblog ####
2020-07-30 12:57:20 INFO h_weblog:::casa #####

```

For security, the web log HTTP server is only accessible from the same computer as the CASA session. To view the web log from another computer, forward the port using SSH. For example, to access the web log hosted on a remote machine called `remotepc`, where the CASA log reports the web log is available at port 30000, execute:

```
ssh -L 30000:localhost:30000 remotepc
```

Alternative: lower browser security

These modification lowers your browser security and should be reverted after viewing the weblog!

Firefox
Navigate to `about:config` and search for the `privacy.file_unique_origin` preference. Change the preference value to false.

Safari
Open Safari preferences, navigate to *Advanced* tab and check the *Show Develop in menu bar* option. From the new *Develop* menu option now visible at the top of the screen, select *Disable Local File Restriction*.

Chrome
The `--disable-web-security` and `--user-data-dir` command line arguments must be passed to Chrome via the command line. For example, on MacOS start Chrome like this:

```
/Applications/Chrome.app/Contents/MacOS/Chrome --disable-web-security --user-data-dir=/tmp
```

Figure 11: 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.

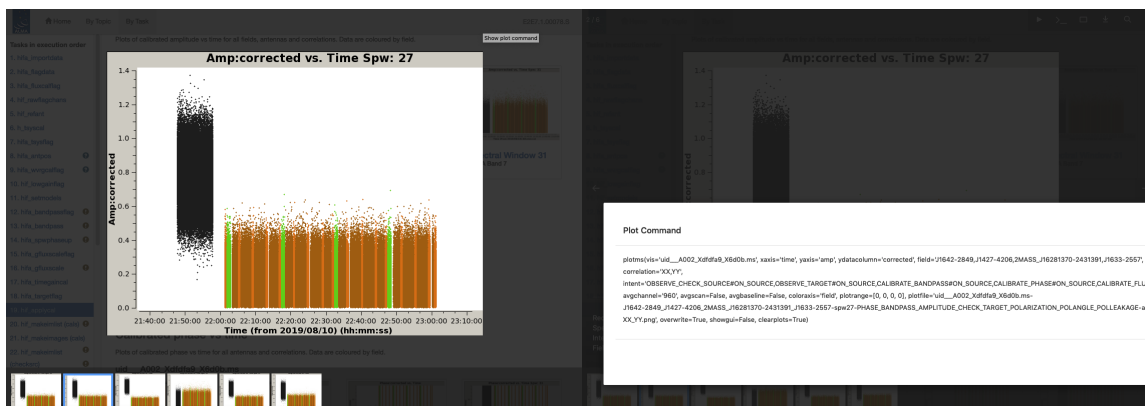


Figure 12: Example of WebLog plot with a “Plot command” link (`>_`) that, when clicked, opens a popup window containing the CASA command for reproducing the plot that can be copied and pasted elsewhere.

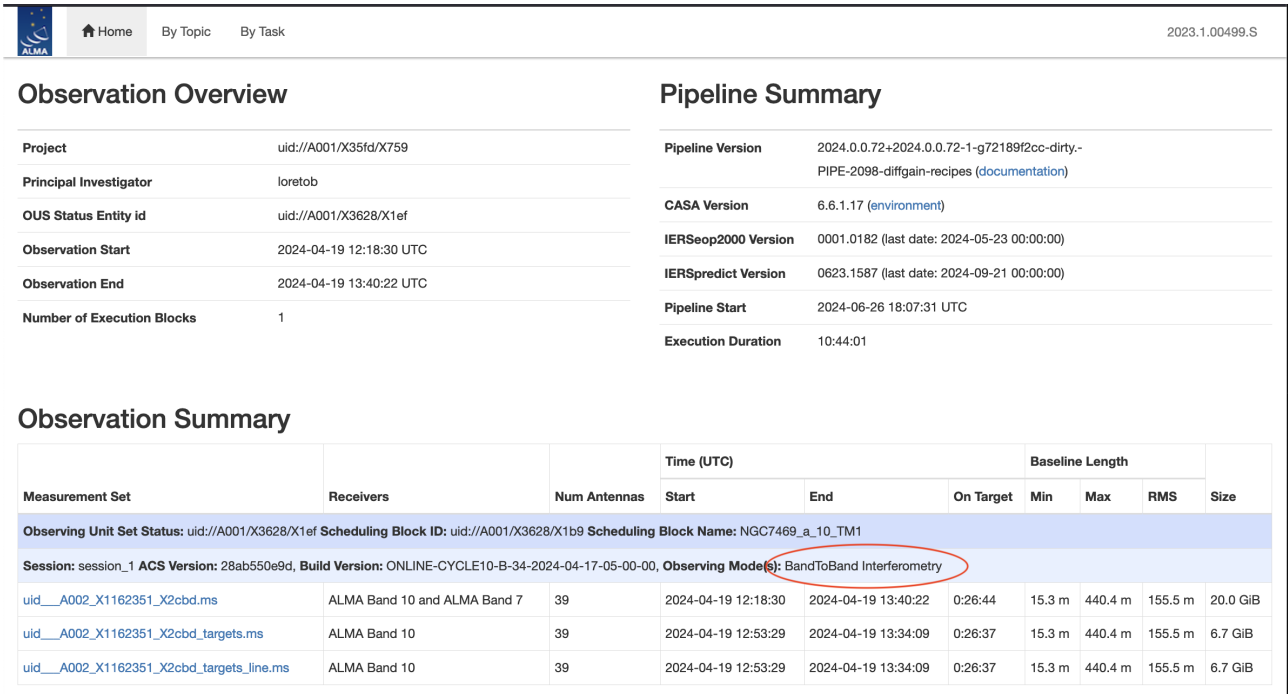


Figure 13: WebLog Home Page: new in PL2024 is the ObservingMode for each session.

The **Observation Summary** table lists all the MeasurementSets included in the pipeline processing, grouped by observing “sessions”. The Observing Mode (StandardInterferometry, StandardSingleDish, BandToBandInterferometry, etc.) is noted for the session. 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 15). 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**, **SpwID vs. frequency**, and **Telescope Pointings** (for 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

Execution Mode: Parallel

Host information	
Hostname	cvpost126
OS	Red Hat Enterprise Linux 8.9 (Ootpa)
Number of MPI servers	7
Max open file descriptors	131072

CPU resources and limits	
Hostname	cvpost126
CPU	Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz
Physical CPU cores	12
Logical CPU cores	24
Cgroup CPU allocation	8
Cgroup CPU bandwidth	100%
CPU time ulimit in seconds	N/A
Max OpenMP threads per CASA instance	1

Available memory and limits	
Hostname	cvpost126
RAM	503.1 GiB
Swap	894.3 GiB
Cgroup memory limit	248.0 GiB
Memory usage ulimit	266287972352
Memory available to tclean	248.0 GiB

Figure 14: Processing Environment popup window



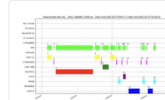
Session: session_1

[uid__A002_Xa8df68_X268f.ms](#)[uid__A002_Xa8df68_X268f.ms](#)[uid__A002_Xa8df68_X268f.ms](#)

Overview of 'uid__A002_Xa8df68_X268f.ms'

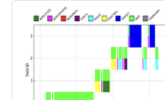
Observation Execution Time

Start Time	2015-08-27 09:24:17
End Time	2015-08-27 10:10:40
Total Time on Source	0:37:11
Total Time on Science Target	0:06:51

[LISTOBS OUTPUT](#)

Intent vs Time

Track scan intent vs time



Field vs Time

Track observed field vs time

Spatial Setup

Science Targets	'Orion_Source_I'
Calibrators	'J0423-013', 'J0522-3627' and 'J0607-0834'

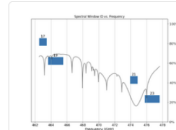
Spectral Setup

All Bands	'ALMA Band 8' and 'WVR'
Science Bands	'ALMA Band 8'
Correlator Name	ALMA_BASELINE

Antenna Setup

Min Baseline	15.1 m
Max Baseline	1.6 km
Number of Baselines	780
Number of Antennas	40
Antenna Diameters	40 of 12 m

SpW ID vs Frequency

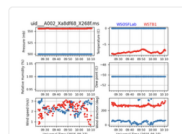


SpW ID vs Frequency plot

Sky Setup

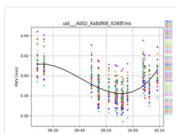
Min Elevation	52.63 degrees
Max Elevation	65.88 degrees

Weather



Weather plot

PWV



PWV plot

Scans

Figure 15: MeasurementSet Overview Page. Click on the table headings in blue for more information about each category.

and science target.

8.5 By Task Summary Page

The **By Task** summary page (Figure 16) 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 ($\textcircled{?}$, $\textcircled{!}$, $\textcircled{\times}$), indicate that there are informative QA messages or important notifications on the task pages. Stages with a $\textcircled{!}$ 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 16).

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_piperscript.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 (or [hsd_applycal](#) page for the case of single dish observations).** At the top of the page will be the **Pipeline QA** scores and associated messages and any **Task Notification** (see Figure 17). 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 18.

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 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

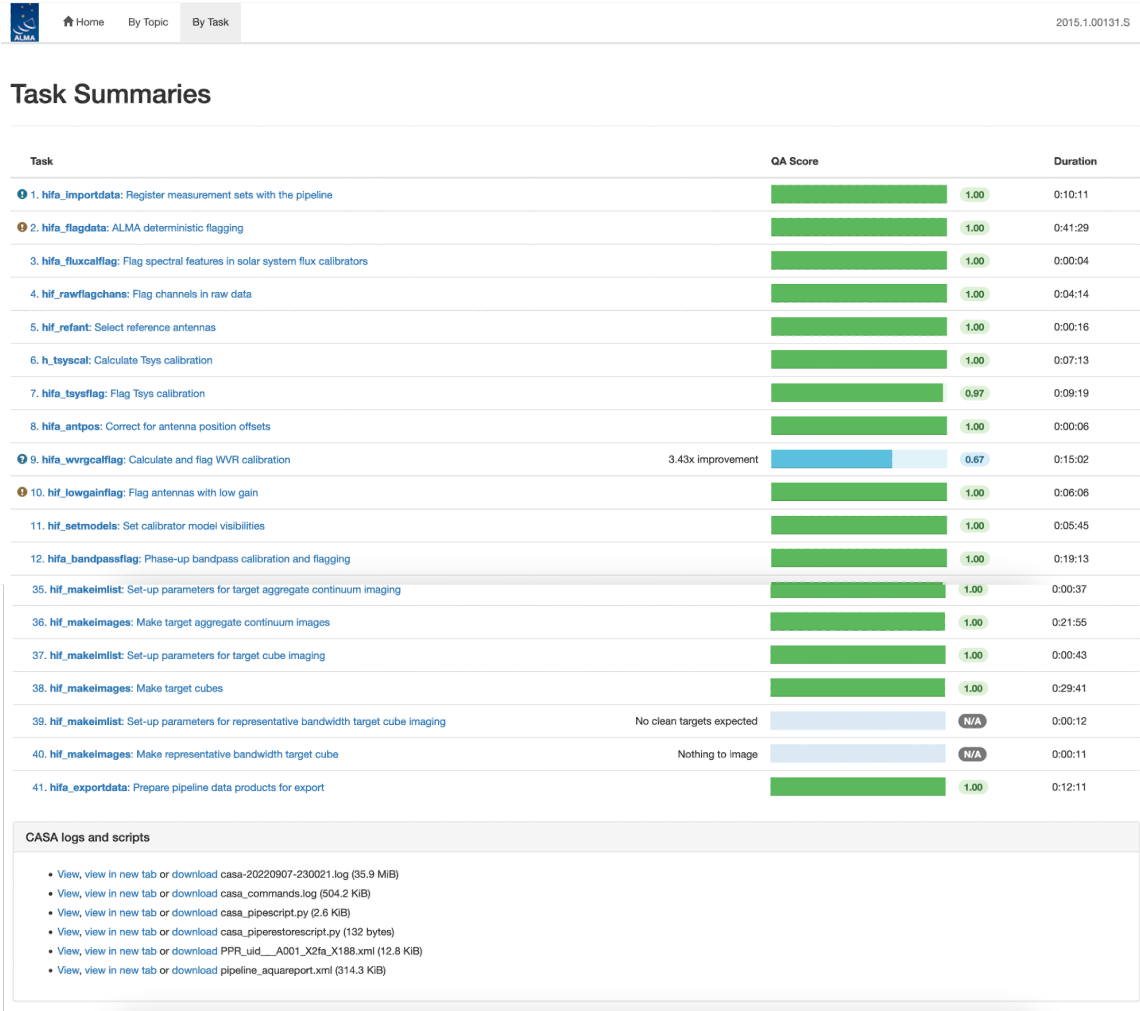


Figure 16: The By Task summary view. The figure has been truncated so both the top and bottom can be seen. Each pipeline stage is listed, along with its QA score (colored bars to the right), computing run-time for each stage, and links to the CASA logs and scripts.

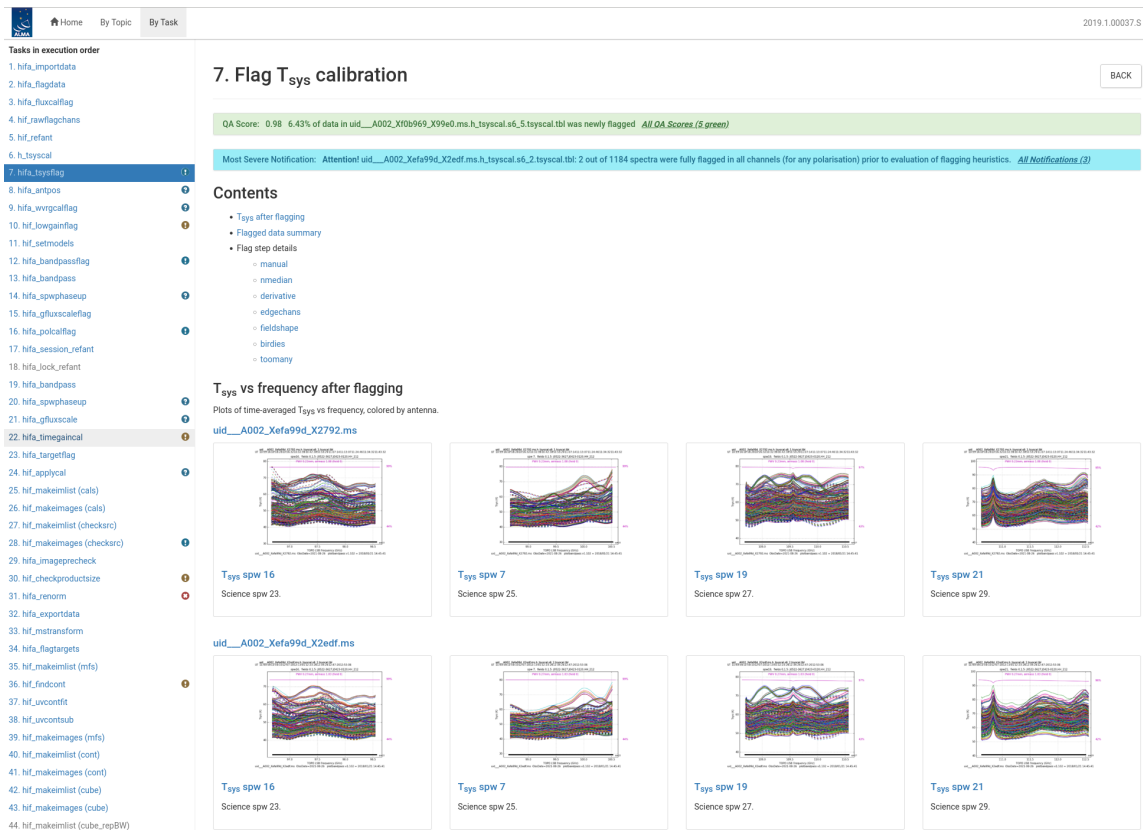


Figure 17: The `hifa_tsysflag` task page, showing the task notifications and QA score at the top, and diagnostic plots (Tsys for each spw grouped by MS). Further down on the page are flagging summary tables. To see the sub-page for this task, click on the MeasurementSet name in blue above each set of plots. This will take you to a page of detailed plots for individual MS/antenna/spectral windows (see Figure 19 for an example).



Figure 18: 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.

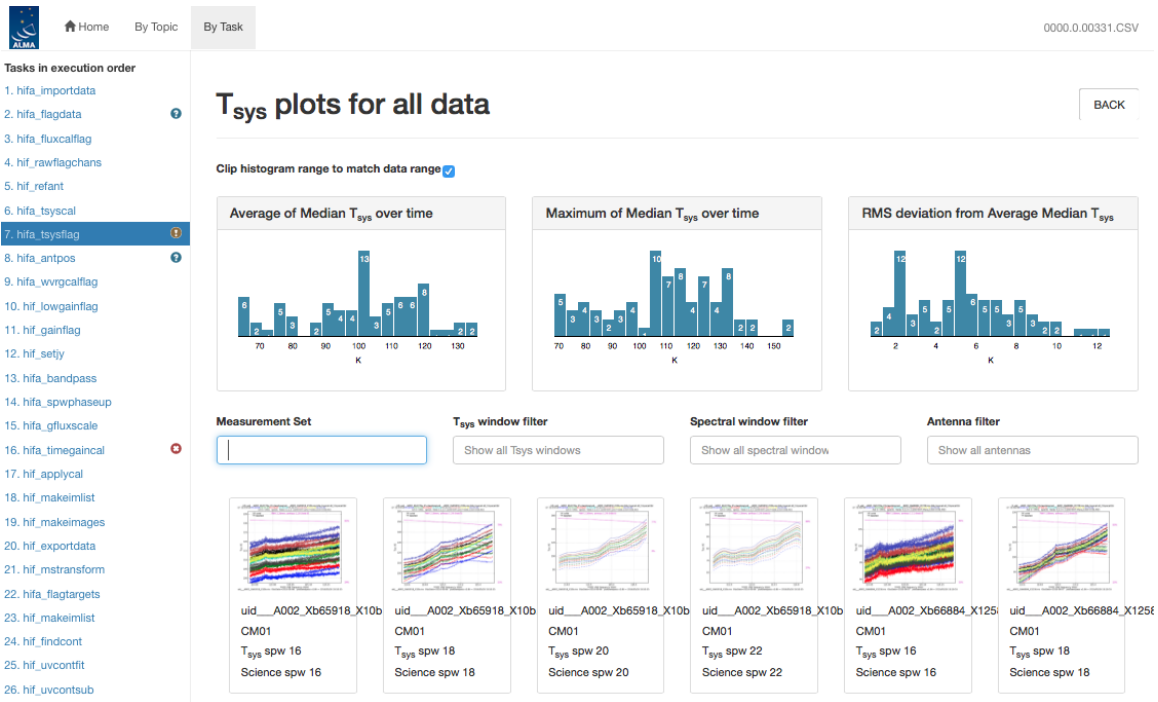


Figure 19: Unfiltered view of the [hifa_tsysflag](#) sub-page. The page is arrived at by clicking on the MeasurementSet link from the [hifa_tsysflag](#) task page (Figure 17). Only the first row of plots are shown; many more appear below (one for each MS, antenna, spw combination). This page has histograms of three metric scores based on the median Tsys that can also be used to filter the plots that are displayed.

these subpages and plot filtering is given in Figure 19 – Figure 21, using the **By Task > hifa_tsysflag: Flag Tsys calibration** pages.

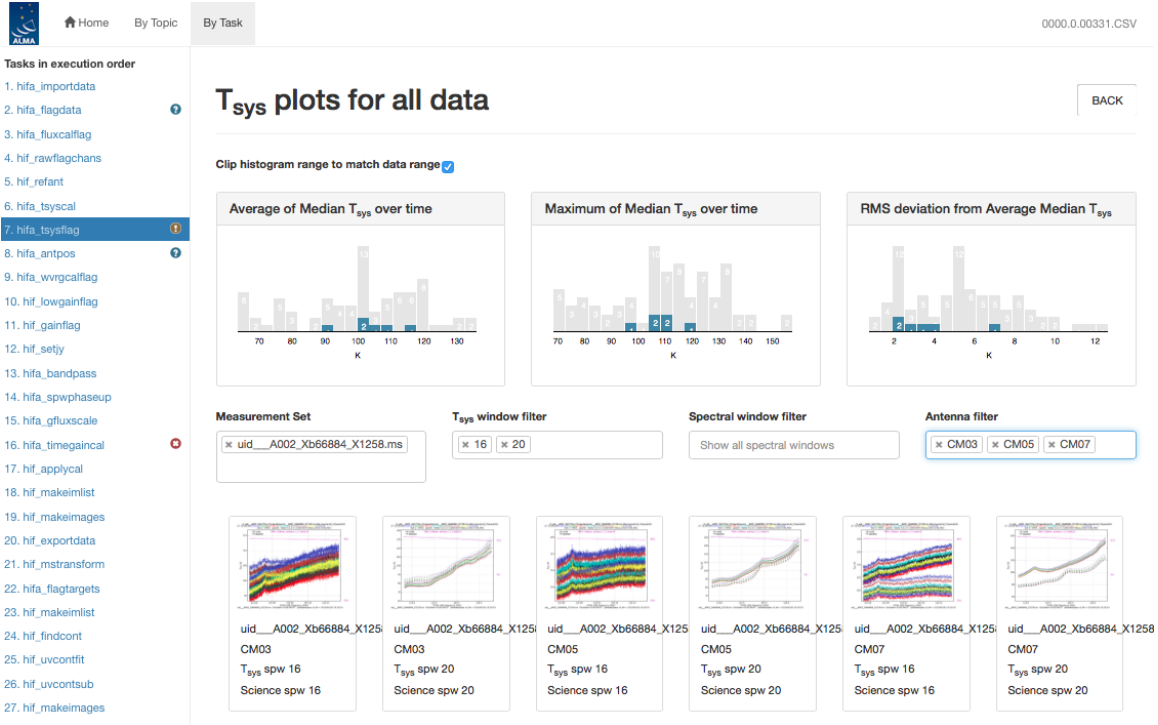


Figure 20: Same as Figure 19, but with a specific MS, Tsys window, and antenna filter set. The corresponding plots are displayed below, and their metric scores are shown by blue shading in the histogram plots.

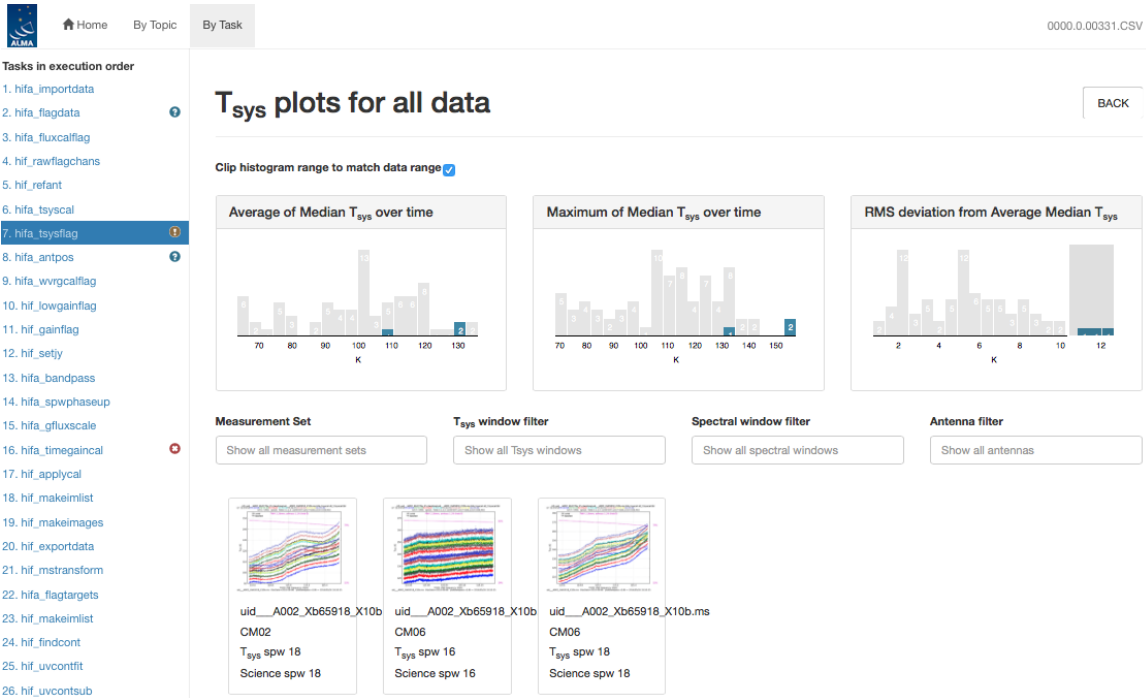


Figure 21: Same as Figure 19, but filtering to the plot of interest by using the mouse to draw a grey box on the highest histogram values in the RMS deviation from Average Median Tsys histogram plot (upper right). To clear the grey box filters on the histograms, click on any white space in the histograms.

8.7 WebLog Quality Assessment (QA) Scoring

Pipeline tasks have QA 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 there are multiple QA heuristics for a stage (each with its own QA score), or scores are calculated separately for each ASDM in an observation, the overall task QA score is taken as the lowest of all computed scores. Valid QA scores have values 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 failure to calculate a QA score results in a red score of -0.1. 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 17).

A detailed description of the IF Pipeline QA scores and their motivation is given in Sec. 7 of [Hunter et al. 2023](#). Any changes to the scores since the date of that paper will be included in §9 below.

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. Poor QA scores are given if the imported data lack key information (missing scan intents or calibrator positions, out of date IERS table), or were modified by previous processing. A QA score of 0.9 is given for high frequency data (Bands 9 or 10), since the calibration of such data deserves more scrutiny.

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.xml 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 section 10.2.6 of the [ALMA Technical Handbook](#) for a description of this catalog). Under normal operations (controlled by setting the parameter `dbservice=True`), the online ALMA calibrator flux service (see section 13.3 of the [ALMA Technical Handbook](#)) 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. A QA score is set based on the result of each catalog query, with score of 0.3 if the database value could not be used and the flux for any spw is from the ASDM, 0.5 if the query returned a warning or had a value that was older than 14 days, otherwise 1.0.

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 (see §7.1) before continuing with the pipeline execution if you first use the `importonly` option of `eppr.executeppr`. This mode uses the non-default value of `dbservice=False` (meaning the intent is to use the values from **flux.csv** and not query the calibrator flux service). A QA value of 0.3 results if `Origin=Source.xml` in **flux.csv**, otherwise 1.0.

If a POLARIZATION intent is present in the dataset, the parallactic angle coverage of each polarization session is shown graphically, and reported quantitatively. A QA score of 0.60 results if this angle is less than 60°. 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 the flag fraction considering the BDF+QA0+online+template+shadow flags: the QA score is 0.0 if the flag fraction is $\geq 60\%$, 1.0 if the flag fraction is $\leq 5\%$, and linearly interpolated between 0 and 1 for fractions between 60% and 5%. There is an additional QA score of 0.8 if the baseband frequency range could not be calculated for a spw.

There is an additional “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 `mintransnonrepsps` and the 5% threshold can be adjusted by the parameter `mintransrepsps`. The QA score will be 1.0 no spw is flagged, 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

If the flux calibrator is a solar system object, known lines in the object are flagged by this task. 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]
table continued on next page	

table continued from previous page		
	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 ₃ CN:	[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], [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]

The WebLog reports if any flagging was required, and if this resulted in the need to use a spw-map. If more than 75% of a given spw is flagged on the flux calibrator for the lines listed above, then a reference spw-map (`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. A QA score of 0.66 is set if a spw-map was required, otherwise 1.0. There is an additional QA score based on the incremental amount of flagging (0.0 if the flag fraction is $\geq 50\%$, 1.0 if the flag fraction is $\leq 5\%$, and linearly interpolated in-between).

Because not all SSOs with atmospheres have been surveyed for spectral lines in all ALMA bands, additional spectral ranges to be flagged can be manually specified to augment the built-in list by specifying the name of a text file in the `linesfile` parameter. The simple format for each line of this file is four values (space-delimited): field name, line name (user-defined), start frequency (GHz), and stop frequency (GHz). Finally, the built-in list of lines can be ignored in favor of the user-specified ranges by setting `appendlines=False`.

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.

The QA score for this stage is equal to (1-percentage of data newly 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):

$$M = n_{ant}([1 - (\text{normalized_distance_from_center})] + \text{normalized_fraction_of_unflagged_data})$$

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).

To avoid picking a reference antenna that is fully flagged on any particular calibrator intent (for example due to shadowing on a low-elevation calibrator, which is unavoidable on a highly positive declination science target),

1. The per-antenna flagging subscore is calculated for each calibrator intent independently.
2. Intent-based flagging subscores are calculated by taking the minimum value across intents to establish the antenna flagging subscore.
3. Antennas with a zero flagging subscore are removed entirely from the refant list.

The QA score is 1.0 if a suitable reference antenna is found, otherwise 0.0.

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.

The QA score is 1.0 if all science spws could be mapped to a Tsys spw, otherwise 0.0.

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 normalized by the median of Tsys in frequency, 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: $100 * \text{mean}(\text{abs}(\text{normalized Tsys} - \text{reference normalized tsys}))$, 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.

The QA score for this stage is based on the total amount of incremental flagging that was done in this stage, and equal to 0.0 if the additional flag fraction is $\geq 50\%$, 1.0 if it is $\leq 5\%$, and linearly interpolated between 0 and 1 for fractions between 50% and 5%, and an additional score of 0.8 results if any spw has an antenna fully flagged.

9.8 hifa_tsysflagcontamination (standard IF recipes only, not diffgain or full-pol)

This task identifies and flags astronomical line emission detected in the system temperature (Tsys) spectrum. As explained in the knowledge-base article <https://help.almascience.org/kb/articles/what-are-the-amplitude-calibration-issues-caused-by-alm-a-s-normalization-strategy>, differences in line emission profiles due to spatial position variations and spectral resolution between the Tsys spectrum and the autocorrelation spectrum can introduce mis-calibration in the affected channels. Thus, astronomical line contamination in Tsys and in the autocorrelations ought to be corrected separately. This task deals with the former issue ([hifa_renorm](#) deals with the latter).

The heuristics work by comparing the Tsys spectra taken during the CALIBRATE_ATMOSPHERE scans toward the science source (and also sometimes toward the phase calibrator) with that Tsys spectrum taken toward the bandpass calibrator. The working hypothesis is that these should be similar save atmospheric lines profiles depending on airmass differences, and Tsys astronomical line contamination. Based on the difference between the source and bandpass Tsys profiles, the task identifies and flags the channel ranges where astronomical line emission is detected.

The task has a only a few configurable parameters: a minimum line-to-continuum ratio (`relative_detection_factor`, default 0.5%) below which the possible astronomical line features are ignored; and a parameter which allows to ignore deviant Tsys profiles (`remove_n_extreme`, default 2). The rest of the currently defined parameters control the input, output, and logging of the task.

The weblog report of the successfully executed task displays three parts: (i) plots similar to those of [hifa_tsysflag](#) but with the detected ranges affected by astronomical lines already flagged, (ii) a set of diagnostic plots, and (iii) a table with a flag summary and the specific flag command template text. Figure 22 shows an example of a diagnostic plot. The diagnostic plot consist of a two-panel graph per source with CALIBRATE_ATMOSPHERE intent, per spectral window (spw), per execution block. Channel ranges which are identified as contaminated and flagged by the task are shown in red. Other ranges with other colors are mainly useful for internal pipeline diagnostics.

The QA score for this stage are based on the result of line contamination heuristic, as follows:

- QA=1.0 if the task ran correctly and no Tsys contamination detected.
- QA=0.9 if the task ran correctly and Tsys contamination ranges detected and flagged.
- QA=0.6 and a message which says “Large difference between the bandpass telluric line and. . .” means that there is an uncorrected telluric line residual which could also appear in the autocorrelations and may need to be manually flagged in the [hifa_renorm](#) stage.
- QA=0.6 and a message which says “Astronomical contamination covering a wide frequency range. . .” means the range identified is too wide for it to be reliable, and no flagging was done.
- QA=0.6 and a message which says “Large residuals. . .” means that the hypothesis that Tsys profiles between the target and the bandpass should be similar is not fulfilled and results from the task could be unreliable. This warning could be triggered by very broad line contamination as well.
- QA=0.6 and a message which says “Heuristic not applied: . . .” means the input data are not supported by the heuristic, so the flags are not applied. This applies to multi-source multi-tuning, double sideband (Bands 9 and 10), and full-polarization data.
- QA=-0.1 if the task did not run correctly.

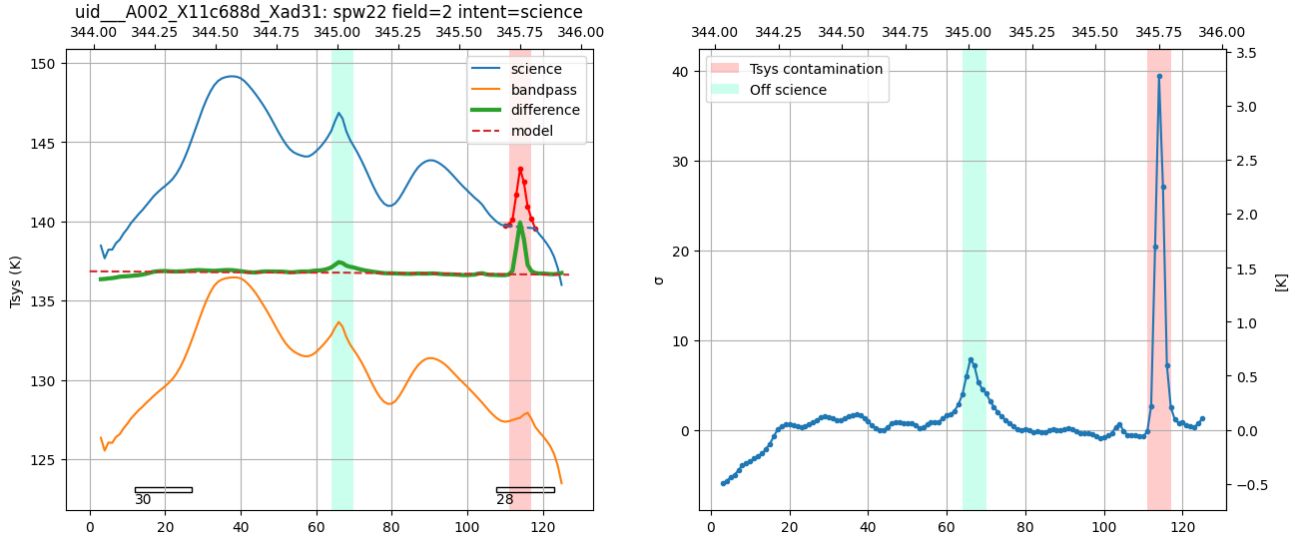


Figure 22: Example diagnostic plot for [hifa_tsysflagcontamination](#) task. The left panel shows the (averaged) Tsys profile taken toward the science fields with CALIBRATE_ATMOSPHERE intent (blue curve), and toward the bandpass (orange curve). A green thick curve shows the difference between these two Tsys profiles (plus a constant added for display convenience). The dashed red curve shows an atmospheric opacity model. A red range indicates detected contamination to be flagged. The right panel shows the difference curve but corrected by the atmospheric model. There are two scales in the y-axis: one in K and the other in approximate (since it varies by channel) signal-to-noise ratio.

9.9 hifa_antpos

Sometimes the antenna positions were refined after the science data were recorded. This task will query an online database for the best available antenna positions and calculate any refinements to the values stored in the MS. Any such refinements are applied in this task. The corrections are shown in two tables in the WebLog, one sorted by antenna name and one sorted by the vector total correction. Values larger than a threshold (default=1.0 wavelength) are highlighted in bold in these tables, and the data are corrected via a calibration table. The QA score is set to 1.0 if no corrections needed, or 0.9 if one or more antenna positions were corrected.

9.10 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 of 15, such as 3 PM with 12 CM), otherwise this stage will not attempt any correction. Antennas with "flagged" WVRs are removed from the refant list, except for heterogeneous arrays in which the CM antennas are never removed the refant list.

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/almataasks.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 / without-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 23](#) 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.

9.11 hif_lowgainflag

Antennas with persistently discrepant amplitude gains are detected and flagged. The WebLog links to greyscale 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 performs an initial phase-up for the BANDPASS intent, then 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.

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 23: 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 beginning 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 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. An "Attention" notification will appear at the top of the page listing which antennas were moved to the end of the list.

The QA score for this stage is based on the total amount of incremental flagging that was done in this stage, and equal to 0.0 if the additional flag fraction is $\geq 50\%$, 1.0 if it is $\leq 5\%$, and linearly interpolated between 0 and 1 for fractions between 50% and 5%, and an additional score of 0.8 results if any spw has an antenna fully flagged.

9.12 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.

The QA score is set to 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, otherwise it is set to 0.0.

9.13 hifa_bandpassflag

After calculating an initial phaseup solution and bandpass solution (see Section 9.14) 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 24). 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 because the before/after summary in [hifa_bandpassflag](#) is intended to clearly show how much new flagging is done by [hifa_bandpassflag](#).

The QA score for this stage is equal to (1-percentage of data newly flagged), and an additional score of 0.8 results if any spw has an antenna fully flagged.

9.14 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) using a point source model. As of PL2025, both the `solint` and `combine` options of the `bandpass` task are computed based on achieving the `phaseupsnr` SNR (default =20) for all spws. If the [hifa_bandpass](#) parameter `solint` is not ‘int’, which is the default in the calibration recipes, the `combine` parameter of `bandpass` is changed from the default value of ‘scan’ to ‘scan,spw’

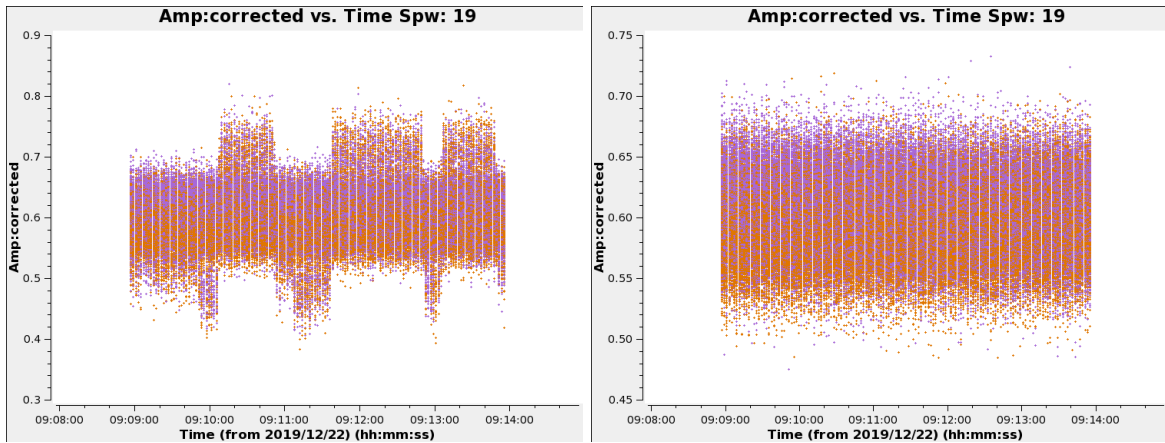


Figure 24: Example of `hifa_bandpassflag` removing visibilities that show outlier amplitudes – before flagging (left panel) and after flagging (right panel).

so as to combine all spws (improving the SNR), and thereafter the `solint` is calculated on the basis of the SNR for the aggregate bandwidth of all spws. The value of `solint` is increased incrementally in units of the integration time up to a maximum value of `phaseupmaxsolint` (default = 60 s¹). Figure 25 indicates the logical workflow. In cases where `combine='scan,spw'` is used, a temporary `gaincal` solution is made to obtain per-spw phase-offsets to align the phases of all spws (similar to the permanent solution made in `hifa_spwphaseup`, see Section 9.15).

The antenna-based bandpass phase and amplitude solutions are then calculated using a SNR-dependent frequency interval, also listed on the WebLog page. Since PL2024, the frequency interval is rounded to counteract the `floor()` function in CASA, so that the interval ends up with the desired number of channels. If the frequency interval needed to reach the required SNR would result in fewer than 8 channels in the bandpass table, the frequency interval is set to one-eighth of the bandwidth, and the result is assigned a QA subscore of 0.70.

Since PL2024, the `hm_auto_fillgaps` parameter, if set to True (which is the default in the calibration recipes), causes the `fillgaps` parameter of the underlying `bandpass` task to be set to 1/4 of each spw width, allowing the user to interpolate across a celestial spectral feature in the bandpass calibrator spectrum. It also avoids gaps due to strong atmospheric lines where the SNR of the solution fails to meet the threshold.

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 following metrics: the SNR metric for the amplitude vs. frequency plots, and the phase derivative deviation metric for the phase vs. frequency plots. These QA metrics are calculated as follows: The amplitude SNR metric is an error function with 1-sigma deviation of 1.0 for the amplitude signal-to-noise ratio. The phase derivative deviation (DD) metric is 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). From PL2025 the combination of spws in the phaseup will present a QA sub-score of 0.9.

Since PL2024, the `auto_fillgaps` parameter, if set to True, causes the `fillgaps` parameter of the underlying `bandpass` task to be set to 1/4 of each spw width, allowing the user to interpolate across a spectral feature in the bandpass calibrator spectrum.

¹In the limiting case, `solint` can very slightly exceed 60 s as an integer of the integration time must be used, e.g. if the integration time is 6.048 s, then ten integration times is the best match to the limiting case, `solint`=60.48 s.

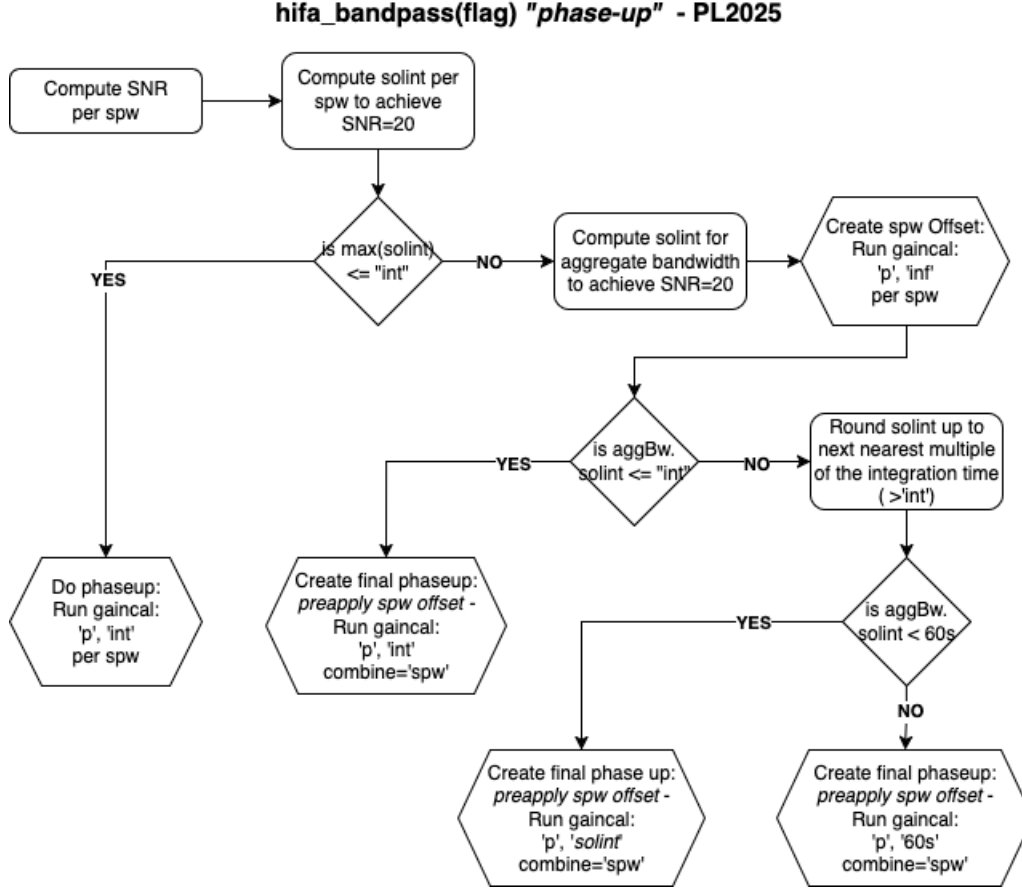


Figure 25: The logical workflow for the temporal phase-up process used in the `hifa_bandpassflag` and `hifa_bandpass` tasks that compute the `gaincal solint` and `combine` parameters.

From PL2025 onward, there is a new bandpass QA for bandpass solutions calculated for FDM spectral windows that use the Baseline Correlator (BLC). In FDM mode, the correlator has subbands with a width of 62.5 MHz which is reduced by a factor of 15/16 (yielding an effective width of 58.59375 MHz) in order to have an overlap between edge channels of two subbands and thereby avoid gaps in sensitivity when combining multiple adjacent subbands (2, 4, 8, 16, or 32) into spws. The subbands can sometimes exhibit anomalous features at subband edges (e.g. offsets and jumps in amplitude and/or phase) if something went wrong with the online process of correlator calibration during the observations. The new QA score is calculated based on five heuristics (two for phase and three for amplitude) evaluated per spw, antenna, and polarization. Two of the heuristics (separately for amplitude and phase) identify subbands with significantly higher noise than other subbands. Two more (separately for amplitude and phase) identify significant offsets between subbands. A fifth identifies abnormally large amplitude spikes. If any heuristic is triggered, the ms is given a QA score between 0.65 and 0.35 based on the fraction of the spw that is affected. The evaluation is skipped and a QA score of 0.70 is assigned if an spw is narrow ($< 2 \times 62.5$ MHz) or if the low SNR heuristics mentioned above result in a frequency interval equal to or larger than the subband width. If no heuristic is triggered, or the data are not from BLC FDM spectral windows, the subband QA score is 1.0.

9.15 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 this assumption is tested later in [hifa_timegaincal](#) as described below). Each Spectral Spec observed is solved separately to generate a separate gaintable. Second, the temporal gain calibration strategy is decided for each of the independent bandpass, amplitude, diffgain, and phase calibrators and the check source. 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(er)-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 science target data after calibration (note that the difference in elevation between the bandpass calibrator and science target(s) are not taken into account, i.e. typically lower elevation sources will have a more unstable, worse, phase RMS. As a result, if the BANDPASS is lower elevation than the science TARGET, the QA score from this assessment may be pessimistic, and vice-versa).

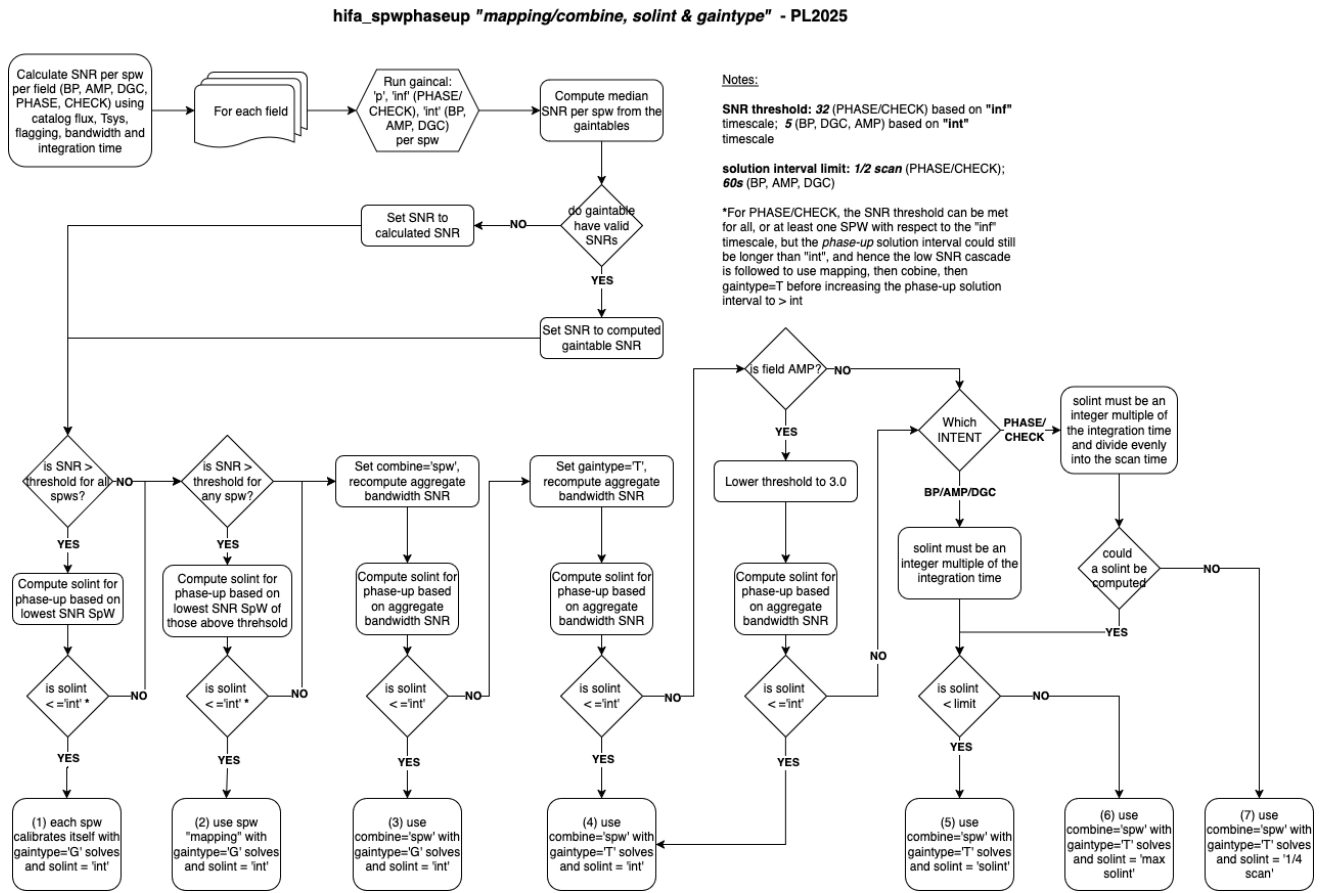


Figure 26: Logic flowchart of [hifa_spwphaseup](#) for determining the temporal gain strategy using the operational pipeline. This logic was implemented in PL2025 and extends the “phase-up” mapping/combine, `solint` and `gaintype` calculation to all intents, except the Polatiration calibrator.

9.15.1 Determination of expected SNR

As of PL2025 we use two approaches to achieve these goals. First, 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 of the bandpass, amplitude, diffgain and phase calibrators, and check source independently (the earliest pipeline versions assessed only the first phase calibrator,

and between PL2022 and PL2025 assessed both the phase calibrator and check source). For each of the phase and check fields, 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, and otherwise the integration time is used for the bandpass, amplitude and diffgain calibrators. Since 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 16×12 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. These SNR values are down weighted to 0.75 of the calculated value as to account for possible source catalog flux variations. The SNRs are shown in the Weblog, column “SNR calculated from source catalog”, and are now only used as a backup in the case the `gaincal` based SNR values are not established.

Second, for each of the bandpass, amplitude, `diffgain`² and phase calibrators, and the check source, a `gaincal` self-calibration with a point source model is made for each spw, using `solint='inf'` for the phase calibrator and check source, and `solint='int'` for the other fields. The median SNR per spw is then computed (also displayed in the Weblog) and used within the logic to select spw mapping or combination, and as the basis for calculating the `solint` and `gaintype` for subsequent phase-up solves for those fields following the logic shown in Figure 26.

9.15.2 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 generally chosen for that field. For the phase and check fields, the SNR must be >32 on a per-scan basis, in most situations guaranteeing $\text{SNR} > 5$ on a per-integration basis considering that there can be up to 30 integrations per typical 60 s scan ($30 \times 2.016 = 60.48$ s). Note that the `solint` for phase-up will still be computed (Section 9.15.3, Figure 26) and could then change the mapping strategy. For the bandpass, amplitude and diffgain calibrators, the SNR must be >5 on the integration time basis directly. If the SNR does not meet the threshold of `phasesnr` (default = 32) for phase calibrator and check source, or `intphasesnr` (default = 5) for the other fields, for a given spw, then that spw can be “mapped” to the spw³ with the highest SNR, meaning that the subsequent phase solutions for the highest SNR spw will also be used to calibrate the less sensitive spw(s). This only occurs if the highest SNR spw is above the respective thresholds. If any such “mapped” spw-maps are required, then they are listed in the table on the WebLog page. Again, the `solint` is always computed and could still further alter the calibration strategy.

If no spw (as part of a Spectral Spec) meets the SNR threshold, then the data from all spws are “combined” (vector averaged) in all subsequent calculation of time-varying phase solves, along with a combined SNR being calculated for later use in computing the `solint`. The estimated combined SNR for that field is shown in the table on the WebLog as is the indication that combine was triggered. Based on the decision in this stage, the workflow of the subsequent calibration stages: `hifa_gfluxscale` and `hifa_timegaincal` are altered with respect to the default high SNR case.

9.15.3 Computing the solint and gaintype phase-up parameters

The ideal phase-up `solint` for calibrators is one integration, typically ranging from 2–6 seconds on the BLC and up to 10 seconds on the ACA Correlator (ACAC), which was retired prior to Cycle 11. Prior to PL2025, other than the phase calibrator and check source, it was always assumed that the calibrators are sufficiently bright point sources as to allow integration based phase-up - this is generally the case for ALMA observations. For the phase calibrator and check source fields, the phase-up `solint` could be increased to a maximum of one quarter of the scan time. Since PL2025, to account for “low SNR” observations, mainly pertaining to the highest frequencies (band 8, 9 and 10) specifically with the ACA 7m array and/or using narrow (<512 MHz) spws, the `solint` and `gaintype` are now computed for each of the bandpass, amplitude, diffgain and phase calibrators,

²The actual intents are DIFFGAINSRC and DIFFGAINREF for the respective high and low frequency scans for the diffgain field in band-to-band observing modes.

³Mapping only occurs within the same Spectral Spec for which the spws simultaneously observed are part of a single correlator spectral setup.

and the check source. These parameters are stored for subsequent phase solves in [gaincal](#), related only to doing phase-up for those fields.

The parameters of `solint` and `gaintype` depend on the mapping or combination process (Section 9.15.2). The `solint` is computed in all cases based upon the difference of the required SNR threshold and the SNR of the data itself. Without mapping or spw combination, all spws will have an SNR over the respective thresholds and typically the computed parameters will be `solint='int'` and `gaintype = 'G'`. For some cases (usually historic data), this might not hold for the phase calibrator or check source if the scan time is very long (i.e. > the modern maximum SSR selection of ~70s on the BLC and ~120s on the ACAC) such that the effective SNR threshold for an 'int' timescale would be higher than the data achieve, and hence the calculated `solint>'int'` and the low SNR cascade would trigger (Figure 26). When any spw are “mapped”, at least one spw has an SNR above the threshold, in which case the `solint='int'` and `gaintype = 'G'` are also generally used based on calculating `solint` from the lowest SNR spw of those above the threshold. The same scenario as detailed above holds for phase calibrators or check sources with long scan times, and if the `solint>'int'` the low SNR cascade will now trigger the spw combination.

When spws are “combined”, the aggregate bandwidth SNR must be above the threshold in order for `solint='int'` and `gaintype = 'G'`, otherwise `gaintype = 'T'` will be triggered, providing a $\sqrt{2}$ improvement in SNR (by combining polarizations in dual or full polarization data), which then must be above the threshold for `solint = 'int'`. If the SNR with `gaintype='T'` is still not above the respective thresholds the solution interval `solint` can be increased as to ensure the data SNR is above the required thresholds. For the phase calibrator and check source, the `solint` is restricted to a whole number of the integration time while also being an integer divisible into the scan time, up to the limit of one half of the scan length. If an optimal solution cannot be found, a default of one quarter of the scan length is set. For the other calibrators, bandpass, amplitude and diffgain, the `solint` can be any multiple of the integration time up to the limit of `phaseupmaxsolint` (default = 60s). For the amplitude calibrator, additionally the SNR threshold can be reduced to `intphasesnrmin` (default=3.0), after setting `gaintype` to 'T' but before increasing `solint`⁴.

9.15.4 QA for Gain calibration

Next, for each field, a diagnostic gain calibration is performed using its chosen strategy, i.e. the `solint` and `gaintype` along with the spw-map. For the phase calibrator and check source the `solint='inf'` is used exclusively. For all assessed fields excluding the check source, if the SNR achieved in the solutions is less than 75% of the SNR threshold, then a warning is generated and the QA subscore is reduced. For check sources, the QA score is 0.9 for $\text{SNR} \leq 0.75 * \text{threshold}$, 0.8 for $\text{SNR} \leq 0.5 * \text{threshold}$, or 0.7 for $\text{SNR} \leq 0.3 * \text{threshold}$, given that it is not a critical calibrator. For the other calibrators, the score is 0.9 for $\text{SNR} \leq 0.75 * \text{threshold}$, 0.66 for $\text{SNR} \leq 0.5 * \text{threshold}$, or 0.33 for $\text{SNR} \leq 0.3 * \text{threshold}$. As of PL2025, in contrast to previous PL versions, low scores (and low SNR warnings) should only occur for very low SNR fields where the employed logic cascade has used spw combination, `gaintype='T'`, and reached the maximum `solint` value. Because [gaincal](#) is used to establish the initial ‘observed’ SNRs feeding into the logic workflow, as opposed to the ‘calculated’ SNR used in previous PL versions, there is no longer any tie with the ALMA source catalog flux density that could have potentially led to an overestimate in the SNR governing the logic cascade⁵. **Note that if these warnings do occur, then the quality of the calibration needs to be questioned. This depends also on the phase stability (see Section 9.15.5) as the use of a sufficiently long solint to deal with low SNR can act to ‘bake-in’ decoherence. Insufficient phase-up correction can significantly lower the visibility amplitudes and subsequently lead to the amplitude calibration overestimating the gains, ultimately resulting in an incorrect flux scale being transferred to the science target(s).**

⁴This staged approach is in an effort to provide phase-up solutions that more optimally correct for temporal atmospheric variations, rather than achieving higher SNR solutions but potentially smoothing over phase fluctuations that will cause decoherence and could ultimately lower the visibility amplitudes.

⁵Prior to PL2025, if the flux density in the source catalog was overestimated (due to a sparse temporal measurement often of phase calibrators and/or for higher frequencies) then the calculated SNR would appear to be higher than the data itself provided and no mapping or combination heuristics would trigger - but the diagnostic post-assessment [gaincal](#) would show the solution SNR to be much lower, likely insufficient for a good calibration, and trigger a warning. This would then require a user to raise the `phasesnr` parameter of [hifa_spwphaseup](#) sufficiently 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 (for the phase calibrator where solutions are transferred to the target(s)).

9.15.5 Assessment of phase decoherence

Finally the phase decoherence assessment is made. Using the higher SNR phase-up solutions of the bandpass calibrator established in the [hifa_bandpass](#) stage, from the widest spw (or combined spw) solutions, 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). Examples of this plot in Figure 27 show the phase RMS as a function of baseline length for the total-time as the gray points. These points represent the longest timescale phase RMS assessment possible within these data and are closely related to the 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).

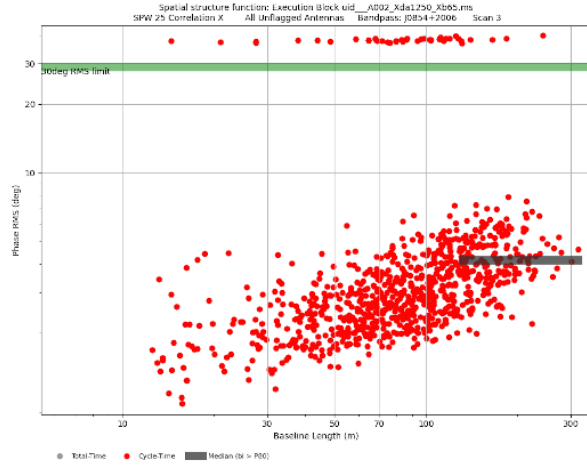
More importantly, for interferometric observations the technique of phase referencing is used for calibration with a repeat interval known as the phase referencing cycle time. The action of phase calibration (see [hifa_timegaincal](#)), when solutions are interpolated to a target (see [hif_applycal](#)), 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 variability that can remain uncorrected in the target source visibilities post application of the phase referencing solutions. In Figure 27, 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 80th 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 27. Note, in some cases for short baseline observations, the phase referencing cycle time can be 5 minutes or longer, which is sometimes longer than the bandpass calibrator scan used for the total-time (which can be 5, 10, 15, 20 or 30 minutes in ALMA observations), and therefore the cycle-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 the shorter baseline arrays, over timescales longer than a few minutes, the phase RMS will remain largely 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.

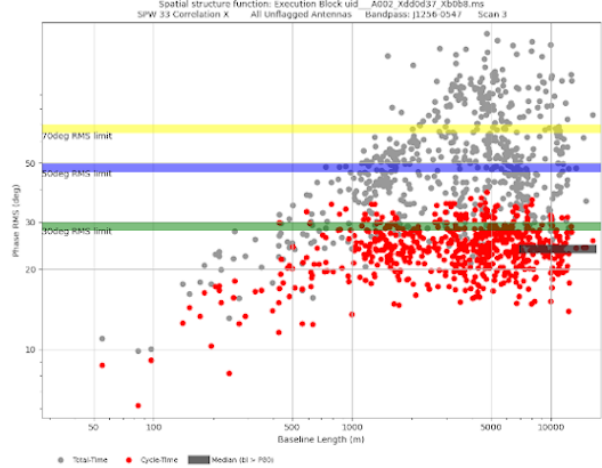
Outlier antennas are also identified if the phase RMS exceeds 180°, pointing to pure phase noise; if the median phase RMS is >50° a limit of 4×MAD + median phase RMS is used; while if the median phase RMS is <50° the maximum value of either 6×MAD + median phase RMS or 2× median phase RMS is used. In the case of data with phase RMS >50°, a reassessment of the phase RMS is made excluding any outliers to examine if they were skewing the median phase RMS value. The outliers values are plotted as semi-transparent symbols (Figure 27 - 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](#).

9.15.6 QA for Phase RMS

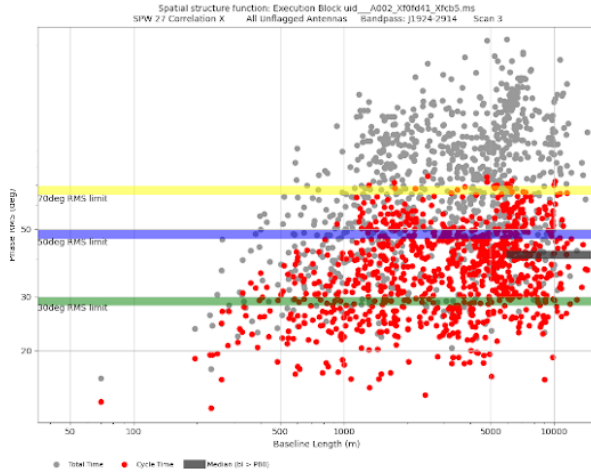
A QA score assessment is made based on the median phase RMS value. Phase RMS <30° is indicative of excellent phase stability and has a QA score of 1.0, but can reduce to 0.9 if there are outlier antennas (see section 9.15.5). Phase RMS between 30-50° is indicative of good stability and has a score of 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



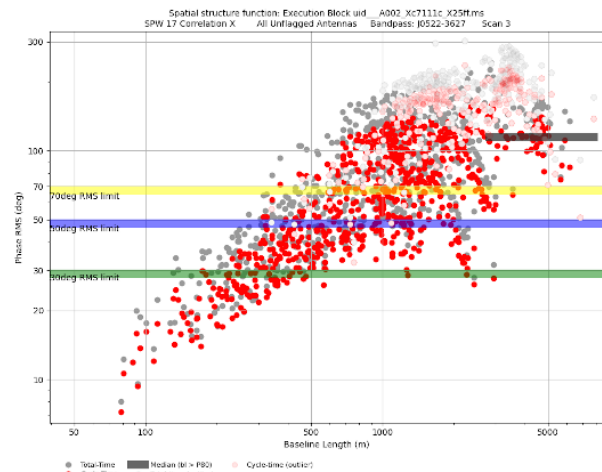
A) Excellent stability - One outlier antenna - SCORE = 0.9



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 27: 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 $\sim 40^\circ$. 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 $\sim 25^\circ$) where baselines after ~ 2 km will all have a phase RMS $\sim 30^\circ$ over the cycle time, while example C is a case of good stability with a median phase RMS of $\sim 45^\circ$. Example D is a mid-baseline length (max baseline ~ 5 km) and has poor phase stability ($\sim 80^\circ$). 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 decoherence.

stability the phase RMS is $>70^\circ$ 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 27. If possible, phase self-calibration (see Section 9.49) 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, low SNR, or poor phase RMS.

9.16 hifa_gfluxscaleflag

As with [hifa_bandpassflag](#), a temporary calibration is performed and applied. The phase-up uses the parameters for spw mapping or combine established in [hifa_spwphaseup](#), but `solint` is always ‘int’ and `gaintype` is always ‘G’, while the amplitude solutions are made using `solint`=‘inf’ and `gaintype`=‘T’. Subsequently, outliers in the calibrated visibility amplitude relative to the model of the flux, diffgain, and phase calibrators, and the check source are flagged. The WebLog shows the “before flagging” plots and (if flags were generated) the “after flagging” plots as a function of both 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 (sometimes the check source).

For the QA score, 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 $(1-0.1)*(1-0.4)=0.54$.

9.17 hifa_polcalflag (polarization recipes only)

As with [hifa_bandpassflag](#) and [hifa_gfluxscaleflag](#), a temporary calibration is performed and applied, then outliers in the calibrated visibility amplitude relative to the model for the polarization calibrator are flagged. The WebLog shows the “before flagging” plots and (if flags were generated) the “after flagging” plots as a function of both time and of uv distance. The heuristics for multi-scan calibrators (of which the polarization calibrator always is) differ slightly from single-scan calibrators.

The QA score for this stage is equal to (1-percentage of data newly flagged), and an additional score of 0.8 results if any spw has an antenna fully flagged.

9.18 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.”

The QA score is 1.0 if a suitable reference antenna is found, otherwise 0.0.

9.19 hifa_lock_refant (polarization recipes only)

This stage sets the refant to a single antenna and refantmode=“fixed” for all subsequent calibration tasks. This setting 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 each called a second time following the locking of the reference antenna, to ensure that the bandpass and spw-offsets calibration tables are using that fixed reference antenna.

9.20 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 calibrator⁶).

⁶Post [hifa_timegaincal](#), during [hif_applycal](#) the amplitude and phase gains from the phase calibrator are transferred to the science target(s). For band-to-band observations, the amplitude gains are rather transferred from the bandpass calibrator (see Section [9.22](#)).

hifa_gfluxscale - PL2025

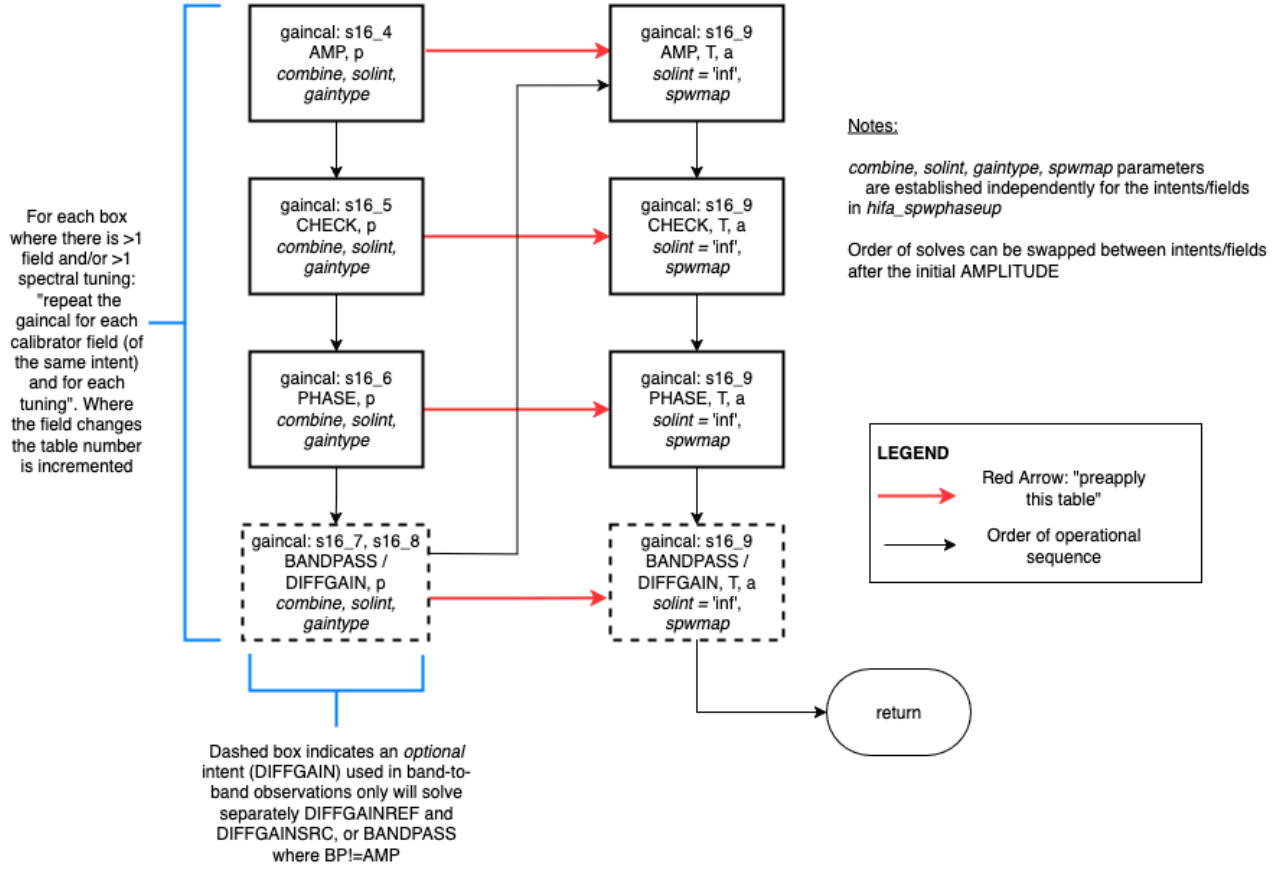


Figure 28: Logical flow in `hifa_gfluxscale`. Each box represents one call to the `gaincal` task, for each intent/field, for each Spectral Spec for one EB, each of which produces a caltable. The content of the boxes indicate the key parameters of the task, including the table number embedded in the resulting `caltable` name, the calibrator intent, the `gaintype`, the `calmode`, the `solint`, and (where appropriate) the values of `spwmap` and `combine`. Italicized parameters are those established in `hifa_spwphaseup`. This logic was updated as part of low SNR work in PL2025, where intents other than PHASE and CHECK can have spw mapping/combine and `solint` or `gaintype` other than 'int' and 'G' respectively. If they are different fields from the AMPLITUDE, the BANDPASS and DIFFGAIN are also solved (dashed box). The DIFFGAIN represents the DIFFGAINREF and DIFFGAINSRC intents that are the low and high frequency Spectral Specs used in band-to-band observations and will have separate caltables.

The workflow is shown in Figure 28. As of PL2025, all parameters for the phase-up previously established by the low SNR heuristics cascade as part of `hifa_spwphaseup`⁷ are used for a phase-only point-source self-calibration on all calibrators (phase-up). If there are other intents, e.g. BANDPASS, DIFFGAIN, that share the same field as the AMPLITUDE and are observed in different scans, then the phase-up is explicitly only performed on the intent AMPLITUDE and effectively only that observing scan. Generally, ALMA uses the same field for BANDPASS and AMPLITUDE and it is usually observed in the same scan. During the `gaincal` processes for

⁷In unique cases, for band-to-band observations, the DIFFGAIN and BANDPASS can be the same field and as such the phase-up call made here is a single caltable produced for the DIFFGAIN/BANDPASS 'shared' field and will use the parameters for the BANDPASS from `hifa_spwphaseup`.

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 29: An example of limited uv ranges used for deriving the fluxscale table on solar system objects when they have been used as flux calibrators.

each field, if there are multiple fields per INTENT and/or more than one Spectral Spec (tuning setup) then a new caltable is created each time. Note that the phase solutions for the check source(s) are saved and are shown in [hifa_timegaincal](#) in the plot of Diagnostic Phase vs. time.

Subsequently, amplitude solves are made (in which the phase solution is pre-applied) using `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. Again, if there are other intents that are the same field as the AMPLITUDE, only the AMPLITUDE intent is explicitly solved. For those intents that are the same field as the AMPLITUDE, a `setjy` call is made to correctly set the flux scale.

High values in the Flagged data summary table on the WebLog page are indicative of low SNR achieved on the corresponding object, even after attempting the best calibration possible according to the [hifa_spwphaseup](#) low SNR cascade. In all cases, the percentages in both the before and after columns will naturally be higher than the corresponding values seen on the later [hif_applycal](#) stage because in that stage the phase solutions are scan-based, thus have higher SNR. It is worth cautioning that in cases of very low SNR data, where the entire heuristics cascade was employed in [hifa_spwphaseup](#), such that the AMPLITUDE calibrator has a phase-up `solint > 'int'` and where the phase variability is unstable over that solutions interval, then decoherence will not be corrected by this phase solve and this will cause subsequent amplitude solves to artificially positively bias amplitudes upwards.

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 assumed to be possible only for solar system objects), only data from antennas with sufficiently short baselines are used in the various [gaincal](#) solves as 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 where transform of a uniform disk drops to 20% of that peak.
6. Select antennas whose separation from the refant are all within that length.
7. If the number of qualifying antennas is < 3, then default back to selecting all antennas.

The antennas used are listed in the table at the top of the WebLog page (Figure 29). Blank entries mean that all antennas were used, which will be the case for quasars.

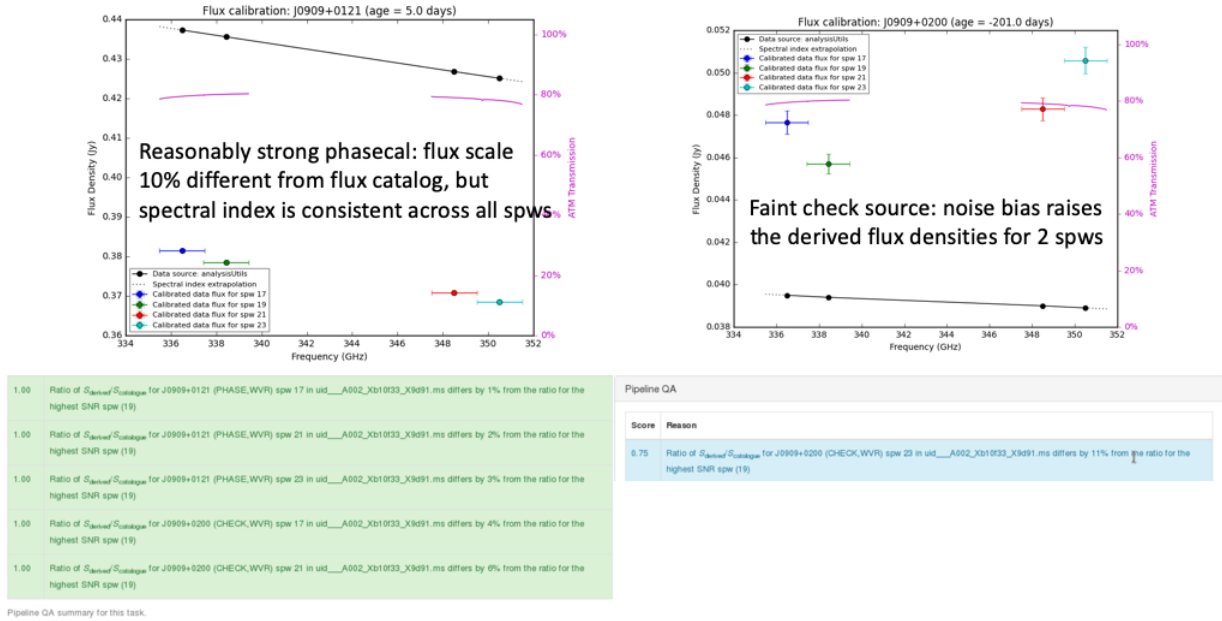


Figure 30: Examples of the plot comparing derived flux densities to source catalog flux densities, and the associated QA messages shown at the top of the [hifa_gfluxscale](#) web page.

Derived flux density vs. catalog flux density plots are shown for non-amplitude calibrators (see examples in Figure 30).

There are three QA scores in this stage to identify potential issues with the flux scale or with calibration transfer by evaluating the derived fluxes for all calibrators except those with the FLUX intent (Hunter et al. 2023). The first evaluates the completeness of the flux determination for each non-FLUX calibrator, and is simply the fraction of spws with derived fluxes (so 1.0 if all spws have a flux determination, 0.50 if only half of them do, and so on). The second evaluates the SNR of the flux determination for each non-FLUX calibrator, to give a blue score if the SNR falls below 20, and a yellow QA score if it falls below a SNR of 5, and a linear mapping in-between.

The final QA score evaluates the consistency of the derived flux densities for each non-FLUX calibrator across all spws by comparing them to the expectation for the same calibrator extracted from the Source Catalog. The comparison is made after globally scaling the catalog values so they match the measured calibrator value in the highest SNR spw (since these sources often vary in time). So this is effectively a comparison of the inferred spectral index of the calibrator compared to its value in the Source Catalog. To calculate this QA score, we define $R_{\text{spw}} = (\text{derived flux for an spw}) / (\text{catalog flux for that spw})$, and $K_{\text{spw}} = R_{\text{spw}} / (R_{\text{spw}} \text{ of the spw with the highest SNR})$. Then the QA score is based on the the spw with the largest deviation of K_{spw} from 1.0 as follows:

- QA=1.0 for $\text{Max}(|1 - K_{\text{spw}}|) < 0.1$
- QA=0.75 for $\text{Max}(|1 - K_{\text{spw}}|)$ between 0.1 – 0.2
- QA=0.5 for $\text{Max}(|1 - K_{\text{spw}}|) > 0.2$

It is worth noting that this QA score can be low for reasons other than the spw fluxscale being incorrect, for instance if the spw has a low SNR (since excessive phase noise will lead to amplitudes biased low), or if there is an atmospheric absorption line in that spw. One should look at other pipeline stages (particularly [hif_applycal](#)) to decide if the low QA actually represents a problem for the science target.

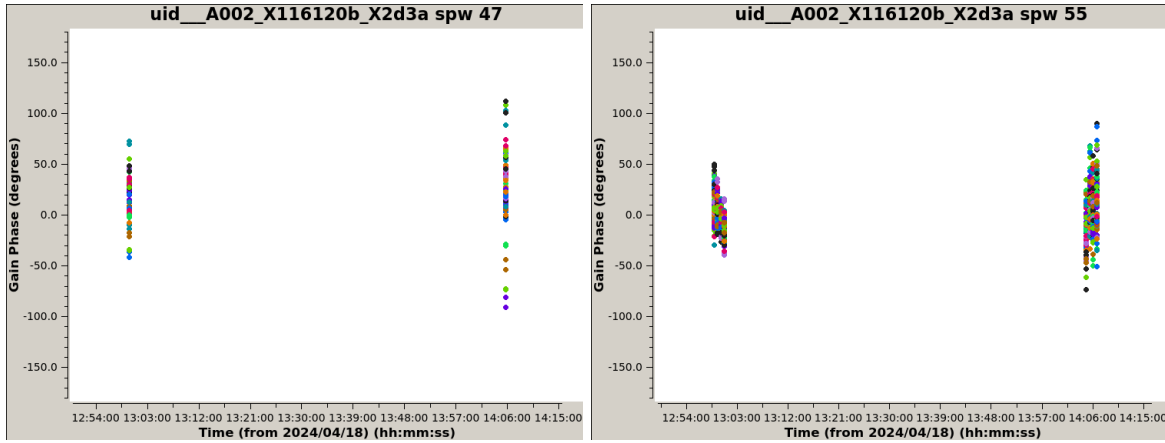


Figure 31: Plots in the [hifa_diffgaincal](#) weblog. The first set show the diffgain phase correction to be applied to the target source (the band-to-band offset). A plot is shown for each spectral window, with phase correction data points plotted per antenna and correlation as a function of time. The phase offsets are computed by pre-applying the LF scan based phase-only solution to the diffgain calibrator data and computing new phase solutions for each spw while combining scan groups at the start and end of the EB. The second set show the diffgain phase residuals as a function of time. The residual phase solutions should scatter about zero with no drift and are calculated by pre-applying the LF to HF phase solutions and the band-to-band offset. The new solutions are not applied to the target.

9.21 hifa_diffgaincal (diffgain recipes only)

Most tasks in the pipeline are now "diffgain-aware" and can run on either band-to-band data with a DIFFGAIN intent or not. In addition,

- [hifa_importdata](#) retrieves the Observing Mode from the ASDM, displays it on the MS Summary page (Figure 13)
- [hifa_spwphaseup](#) calculates and registers the spw-map required so that time-varying gains calculated in [hifa_timegaincal](#) using the low frequency reference spws in the PHASE intent are applied to the high frequency science spws on the TARGET intent.
- as of PL2025, [hifa_spwphaseup](#) also assesses the SNR parameters for the DIFFGAIN which can steer the choice to combine spectral windows in this stage.

The [hifa_diffgaincal](#) stage itself calculates the phase offsets between the LF (low frequency band, e.g. Band 4) and HF (high frequency band, e.g. Band 9) spws, the so-called band-to-band offset. First, scan-length gains are solved on the low frequency (reference) spws, this uses the intent DIFFGAINREF, as implemented since PL2024. Then the 'band-to-band offset' is solved while pre-applying those LF (reference) scan solutions (using linearPD interpolation to correct the frequency scaled temporal phase changes), to determine two temporal solutions (combining groups of DIFFGAINSRC HF scans at the start and end of the EB, respectively) per antenna per polarization (effectively the instrumental offset between bands). If the observation is long, approaching two hours, it is possible that there is an additional DIFFGAIN observation also at the mid-point of the observations, thus making 3 scan groups, and 3 solutions per spw per polarization. Residual solutions, shown also in the Weblog plots, first pre-apply all corrections then solve for the scan based DIFFGAINSRC phases. These solutions are designed to show DIFFGAINSRC spread (about zero degrees), as shown in Figure 31 (right-panel). Residuals that are not signal-to-noise limited should have an ideal (perfect) spread below ± 30 deg, while data acquired in good conditions have a spread within ± 50 deg. There should be no significant offset or outlier solutions from the base zero degree phase.

As of PL2025, updated low-SNR heuristics are employed as to allow `combine='spw'` to be used in any three of the solution steps. The assessment is as follows:

1. If `combine='spw'` was triggered as required for the phase-up (in [hifa_spwphaseup](#) on the DIFFGAIN - using the relevant REF or the SRC) the previously computed SNR and `solint` are used to calculate whether spw combination is required for the scan based phase solves, or in the case of the band-to-band offset - the group of scans, as to meet an SNR of 5⁸. If this SNR threshold is not met, spw combination is triggered.
2. A temporary [gaincal](#) is made to perform the reference, band-to-band offset and residual calibration steps as described above, on a per-spw basis. Then if (i) not all spw solutions are made; (ii) the fraction of flagged solutions exceeds 0.7 of the total solutions, and (iii) the fraction of missing scans exceeds 0.7 of the expected scans, the combine would also be triggered.

This new (PL2025) low SNR implementation accounts for cases where despite selecting a ‘strong’ calibrator⁹, high frequency observations usually made with the ACA and/or narrow spw bandwidths <512 MHz would have a low SNR regardless, resulting in sub-optimal or failed phase solutions without spw combination, and thus poorly calibrated or entirely flagged science data. Note that if `combine='spw'` was triggered for the ‘reference’ solve (LF data), often due to an entirely flagged/bad spw, then because this is a LF to HF transfer solution as will also occur later for the phase transfer from the phase calibrator to the science target(s), then the combine and spw mapping is updated for the phase calibrator (modifying that established in [hifa_spwphaseup](#)) to ensure optimal LF to HF phase transfer to the science target(s). The [gaincal](#) workflow and low SNR logical flow are shown in Figure 32.

The final QA score is the lowest of all sub scores. If the various caltables are successfully calculated the sub-score is 1, and 0 if not, as any missing table effectively invalidates the calibration process. In addition, sub-scores are provided for each of the reference, band-to-band offset and residual solves of 1.0 if `combine='spw'` was not used, and 0.9 if the combination was used. This sub-score is informative.

9.22 hifa_timegaincal

In this task, gains as a function of time are calculated from observations of the bandpass, flux, polarization, phase calibrator(s), and diffgain (reference and source respectively if it is a B2B observation). The flowcharts in Figure 33 show the [gaincal](#) calls for the entire process, with options listed dependent on whether the data have high or low SNR. First, phase solutions of various types are made, followed by amplitude solves, some of which are used and propagate to [hif_applycal](#) while others are only informative or used in diagnostic-based plots for the WebLog, while finally a phase offset assessment is made for the PHASE calibrator. The process follows:

1. Perform the scan-based, `solint='inf'`, phase calibration on the PHASE source and thereafter other calibrators, BANDPASS, AMPLITUDE, POLARIZATION, (DIFFGAINREF if B2B was used¹⁰). Parameters for `combine` and `gaintype` are governed by those established in [hifa_spwphaseup](#) for the PHASE calibrator, irrespective of which calibrator is being solved (`minblperant=4`, `minsnr=3` are also used). This scan-based phase solution is registered only to apply the calibration from the PHASE calibrator to apply to itself, PHASE, and the target(s) and CHECK source. Only for band-to-band cases is the (`interp='linearPD'`) registered to apply temporal phase variations with the correct frequency scaling. Standard data use `interp='linear'`.
2. Perform a ‘phase-up’ on all calibrators. For PL2025, all parameters as established in [hifa_spwphaseup](#) are used for `solint`, `combine`, `gaintype`¹¹. Ideally the `solint='int'`. All solutions will be later used in pre-applies, and all but the PHASE calibrator solution will be used in [hif_applycal](#). The plots from the gaintables are also shown as phase diagnostic plots in the WebLog.

⁸The SNR threshold used is matched with that previously set in [hifa_spwphaseup](#), and is by default 5.0 for the DIFFGAIN intent.

⁹PL2024 assumed the DIFFGAIN calibrator, along with BANDPASS and AMPLITUDE calibrators, were strong enough for a per spw solution in all phase solves.

¹⁰For band-to-band observations only the Spectral Spec of the DIFFGAINREF (the low frequency band the same as the PHASE calibrator) is used in these solves as it is only the PHASE calibrator solution that is used and applied later in [hif_applycal](#).

¹¹If calibrators have a low SNR and in conjunction the atmospheric phase variations are unstable, decoherence will not be well corrected when `solint>'int'`, the visibility amplitudes will be lower, and the gains will therefore be incorrectly scaled in any subsequent amplitude solves. This is most problematic if the AMPLITUDE calibrator is low SNR and stability is marginal to poor

hifa_diffgaincal - PL2025

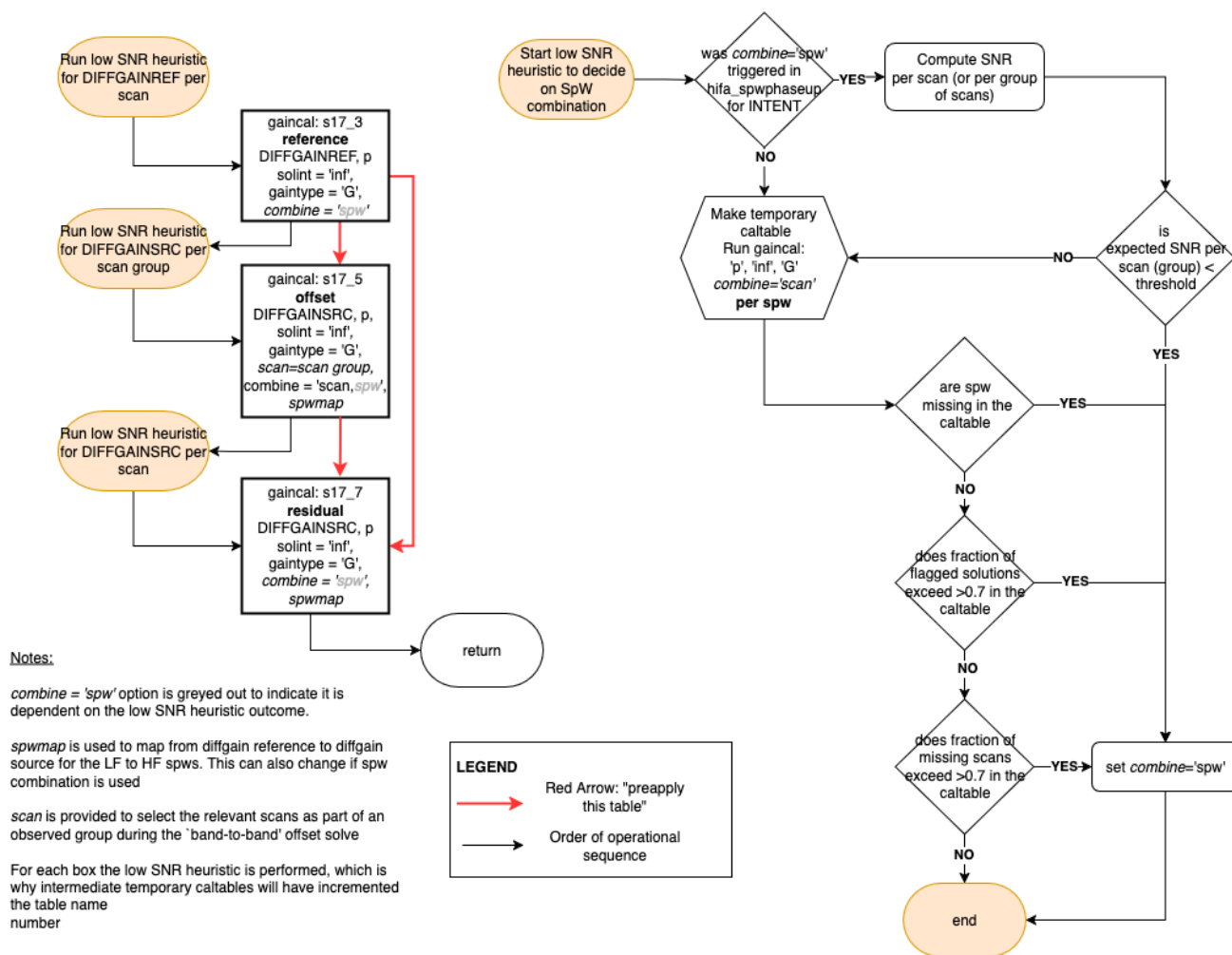


Figure 32: **Left:** Gain solution workflow for the `hifa_diffgaincal` stage. Before each `gaincal` task, the low SNR heuristic is run. The caltable are used to make plots shown in the WebLog for the 'offset' and 'residual' caltables only. Stage numbers and task parameters are also noted. **Right:** Workflow for the low SNR heuristic logic that is used to establish whether spw combination is required for any of the reference, offset and residual `gaincal` solves performed.

3. Pre-apply the ‘phase-up’ solves from the above step and solve for the amplitude gains using `solint=‘inf’` to make scan-based solutions. The parameter `gaintype=‘T’` is always used in amplitude solutions so as not to impart polarization into a target. For all calibrators, these solutions are registered to be self applied given `setjy` has already been run in `hifa_gfluxscale` to set the correct models. In the case of the PHASE calibrator, the amplitude gain solutions are transferred to the target(s) and CHECK source. In the case of B2B observations, the phase calibrator is not at the correct frequency, and hence the CalLibrary is updated to replace the amplitude gains reference field with the BANDPASS calibrator for the target(s) and check source.
4. Solve for the amplitude gains as above, but as a diagnostic only, using `solint` extracted from the `hifa_spwphaseup` for the PHASE calibrator (used as a single common ‘shortest’ `solint` for all diagnostic amplitude solves) and with `minsnr=2`. These solutions are used for diagnostic plots only. Typically `solint=‘int’` for most observations with sufficient SNR.
5. Produce diagnostic phase solutions for the phase offsets. If not already using `combine=‘spw’` for the PHASE calibrator the phase-up gaintable (that used the best `solint` from `hifa_spwphaseup`) is unregistered from the pre-apply for the PHASE calibrator so as not to correct the temporal phases on a per spw basis. Subsequently, `gaincal` is run to make a temporary caltable forcing a solution using `combine=‘spw’` and `solint` from `hifa_spwphaseup` on the PHASE calibrator. These phase solutions combining spws are then pre-applied and the PHASE calibrator is solved again using `solint=‘inf’` and per spw. Solutions are also made for the BANDPASS¹² calibrator (and the POLARIZATION if present, and DIFFGAINREF, DIFFGAINSRC for B2B¹³).

The WebLog page shows the types of plots of all gains vs. time, at the top of the page. The appearance of the plots are shown in a different order to the processing flow, first the longer term timescale phase plots, then the amplitudes plots. Thereafter the results of these rapid solutions for the bandpass, flux, and polarization (and diffgain) calibrators are shown as 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 check sources previously derived in `hifa_gfluxscale`.

Next shown are the phase offset plots used primarily to check the PHASE calibrator and spw to spw comparability. One plot is shown for each spw, with phase offset plotted per antenna and polarization as a function of time in the WebLog. If the instrument is stable, the new phase solutions should scatter about zero with no drift. The difference between these solutions are plotted near the bottom of the WebLog page and should be zero (within the noise) if the variation in the phase solutions is due only to residual non-dispersive atmospheric path variations (Hunter et al. 2023). Any systematic differences from zero indicate instrumental instabilities. These spw-to-spw offsets will correctly calibrate out unless spw mapping or combination is triggered due to the selection of a weak calibrator (§9.15.2). Even with a strong calibrator, these instabilities can limit the SNR of observations with very high dynamic range. In any case, it is always worth identifying these offsets since they generally indicate hardware problems. To this end, there are QA scores based on examining the mean offset (on a per-antenna/polarization/spw basis) for all scans, the maximum deviation for any given scan, and the scatter in the offsets for an antenna compared to the average scatter for all antennas. There is also a check for the scan-to-scan variation in the offsets considering all solutions, which indicates whether the solutions are too noisy to reliably calculate any score.

The QA scores are calculated as follows: If the standard deviation of all offsets over all antennas (σ_{off}) for a spw is $> 15^\circ$, 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 $> 30^\circ$ or $6 \times \sigma_{off}$; the mean offset is $> 30^\circ$ or $6 \times \sigma_{off} / \sqrt{N_{ant}}$; or if the number of scan solutions is > 3 and the standard deviation of the offsets is $> 30^\circ$ or $4 \times \sigma_{off}$. 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. Finally, the above scores are decreased by 0.1 if the spw was mapped, with a QA message reporting that there are possible phase offsets for the listed antennas/spw that could affect calibration. If the spw is not mapped, then the QA message will say that the spw is not mapped so

¹²As the `solint=‘inf’` the bandpass will necessarily be a single solution per spw at exactly zero degrees phase and can be used as guide for the eye in WebLog plots.

¹³For most B2B data taken in Band 9 or 10 these solutions will be largely noise dominated especially if `hifa_spwphaseup` required spw combination.

hifa_timegaincal - PL2025

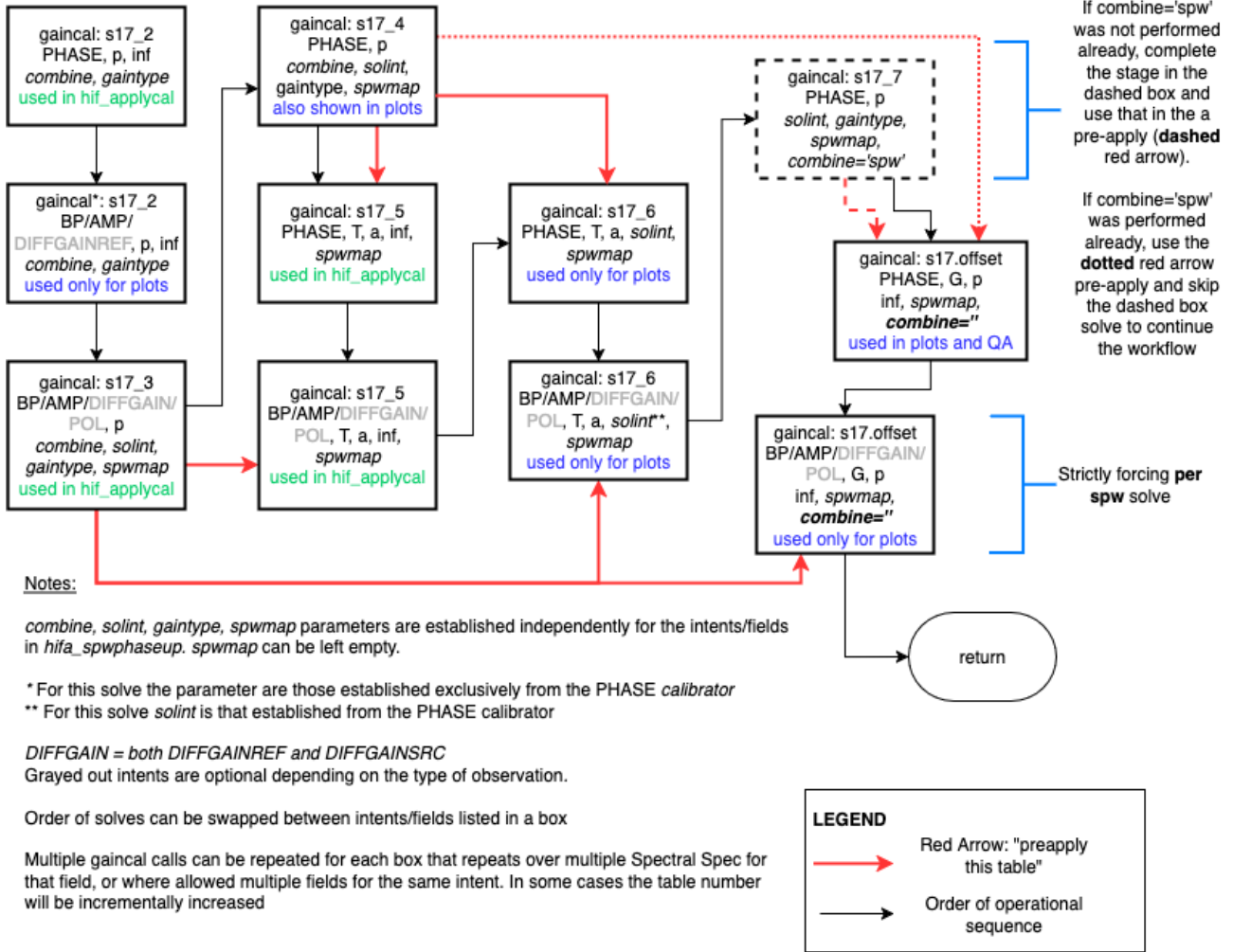


Figure 33: Gain solution workflow for the [hifa_timegaincal](#) stage. This workflow describes all methods where *combine='spw'* may also have been set in [hifa_spwphaseup](#). An additional piece of information in each box is the subsequent usage of the resulting caltable, either for application to the data in [hif_applycal](#) or merely for diagnostic plotting ([plotms](#)). The 'offset' solves are used both for diagnostic plotting and also QA assessment.

that any offsets should calibrate out. Note that for sources with adequate continuum SNR, any reported phase offsets may be corrected by subsequent self-calibration in [hif_selfcal](#).

Finally, the diagnostic short-solint amplitude vs time plots for bandpass, flux, and polarization (and diffgain) calibrators are shown. Note in B2B observations the plots will show those intents where spws exist for those calibrators, for example only the PHASE calibrator is observed at the low frequency setup, while the AMPLITUDE calibrator is observed in both the low and high frequency setups.

9.23 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. As of PL2024, the renormalization spectra are stored as a Tsys-like calibration table, and applied along with all the other cal tables during [hif_applycal](#). Previous versions of the [hifa_renorm](#) task edited the MS directly.

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. Spectral windows with large numbers of channels are split into several “segments” that are fitted independently – the code checks *a posteriori* whether any features are found on segment boundaries and adjusts the segments to avoid that. 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 34). 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 35. 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. Note that the segment checking heuristic is performed on a per target, per spw, per field basis so it is possible that some fields will require different numbers of segments than others. This can lead to what appear to be discontinuities in the summary spectra plots which average these fields together. Therefore, if the number of segments changes across fields that will be averaged together, a warning message is displayed on the summary plot that reads “Number of Segments

Contents

- [Table](#)
- [Plots](#)

ALMA cross-correlations are divided by the auto-correlation as a function of frequency, in the correlator. This has a variety of advantages for operations and calibration, but if there is strong line emission detected in the autocorrelation (i.e. as would be detected in a single dish spectrum), that emission can anomalously decrease the cross-correlation amplitude at those frequencies.

This effect can be mitigated by comparing the autocorrelation spectrum (AC) of the target with the AC of the bandpass, which is generally located away from any such bright contaminating line emission. The ratio of the bandpass AC to the target AC provides a scaling factor as a function of frequency that can be used as a first order correction spectrum. However, atmospheric and instrumental variation (e.g. baseline ripple) need to be fitted and removed, so the spectrum is divided into several segments (marked on the plots as thin dotted vertical lines) for that fitting. The fitted AC ratio is presented here as the 'renorm scale factor' or 'renorm amplitude'.

All targets, spws, and measurement sets with maximum scaling above the observatory determined threshold are colored in the table.

Informative plots are included on this page and collected in a pdf for each spw and source, linked from the table below.

The plot shown on this page is a ReNormSpectra summary plot showing the average scaling spectrum over all scans, and for mosaics, all fields in the mosaic with peak scaling above the threshold. All antennas are plotted as dashed red and blue (for X and Y), and the mean is plotted solid. Vertical grey shaded regions indicate areas of the spectrum that may be affected by atmospheric features.

The pdf next contains RenormDiagnosticCheck plots corresponding to each field and scan. The scaling spectrum is plotted as solid lines for each antenna (again red and blue for XX and YY), and the median as a dashed line (green and black for XX and YY).

The renormalization script has heuristics to detect and correct spikes, dips, and jumps near the segment boundaries (marked with thin vertical dotted lines). Less significant (below the threshold for applying the correction) features may remain.

Features in the scaling spectrum associated with atmospheric features require additional care - ALMA data reduction staff will have evaluated these and minimized them insofar as possible with current heuristics, but PIs should take note of the shape and magnitude of any applied correction when performing line science at frequencies overlapping atmospheric lines.

Table

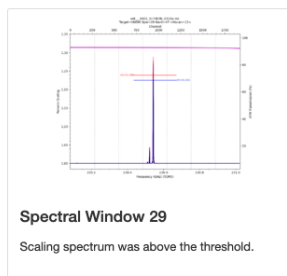
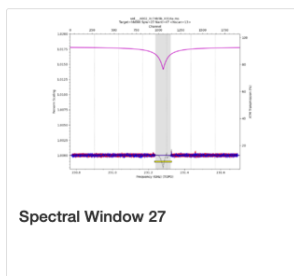
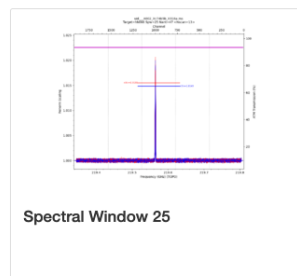
MS/Source/SPW that trigger the need for renormalization above a threshold of 1.02 highlighted in blue. Renormalization has been applied as detailed in the task inputs.

Please refer to the Pipeline User's Guide (linked to this weblog's Home page) for more details on renormalization and interpretation of the plots.

MS Name	Source Name	SPW	Max Renorm Scale Factor (field id)	PDF Link to Diagnostic Plots
uid__A002_Xc74b5b_X316a.ms	hh666	25	1.0154952 (3)	PDF
		27	1.0011648 (3)	PDF
		29	1.2374023 (3)	PDF

Plots

uid__A002_Xc74b5b_X316a.ms
hh666



Input Parameters

Tasks Execution Statistics

CASA logs for stage 26

- [View](#) or [download](#) stage26/casapy.log (1.1 KiB)

Figure 34: 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 spectra saved to a calibration table when the parameter `createcaltable=True` (spectral window 29 in this example). When [hifa_renorm](#) is run in an assessment only sense, with `createcaltable=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.

Changed!!!!” and a blue QA message will appear explaining this. 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 35). 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 36).

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. For this reason, such observations are given a blue QA score of 0.9.

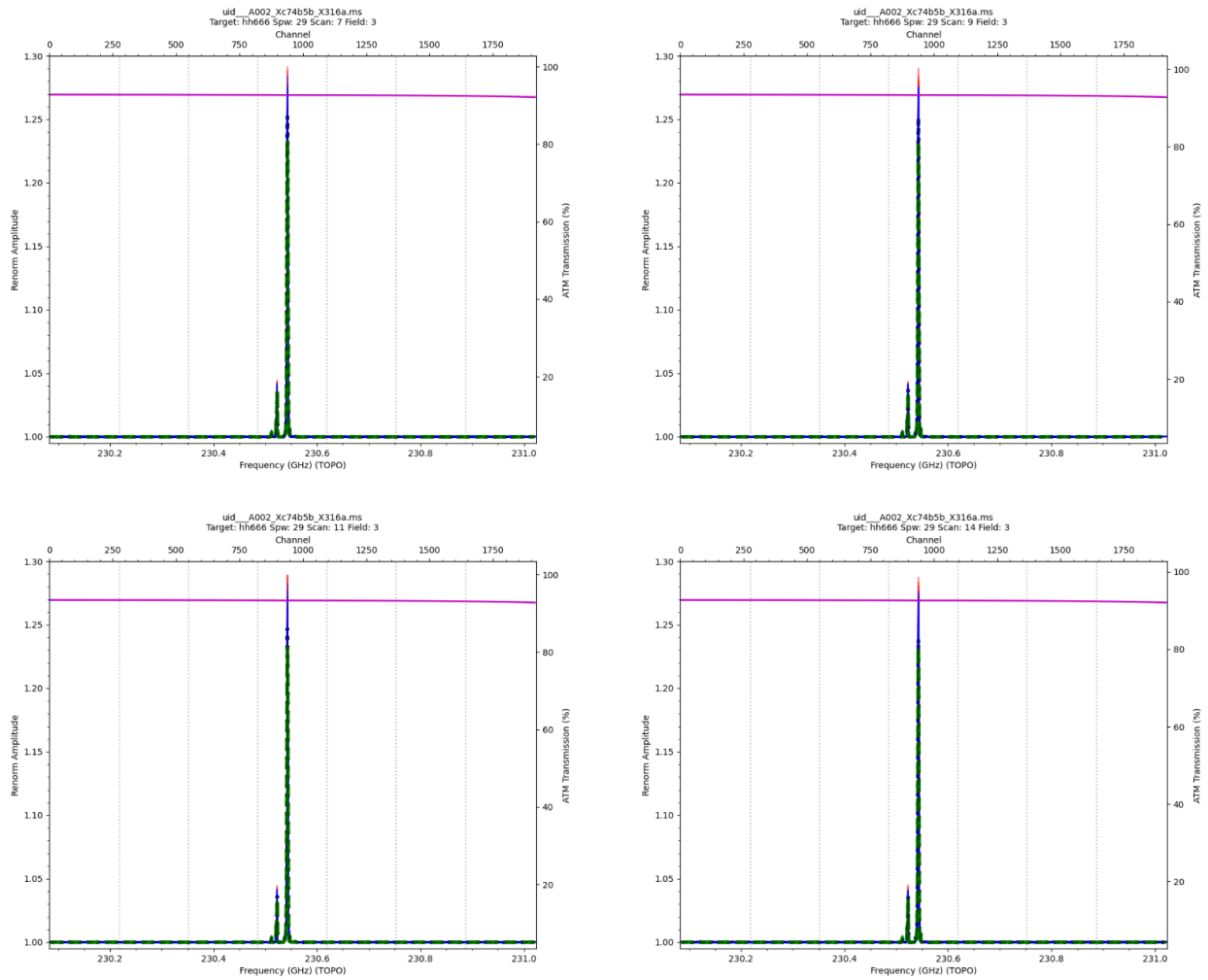


Figure 35: Screenshot of a PDF file generated for an Execution Block, science target, and FDM spectral window during the updated `hifa_renorm` stage of the IF Pipeline for which the renormalization will be applied. In this example the Carbon Monoxide astronomical line is the reason for the renormalization and is causing an atypically high scaling value exceeding 1.25 (25%).

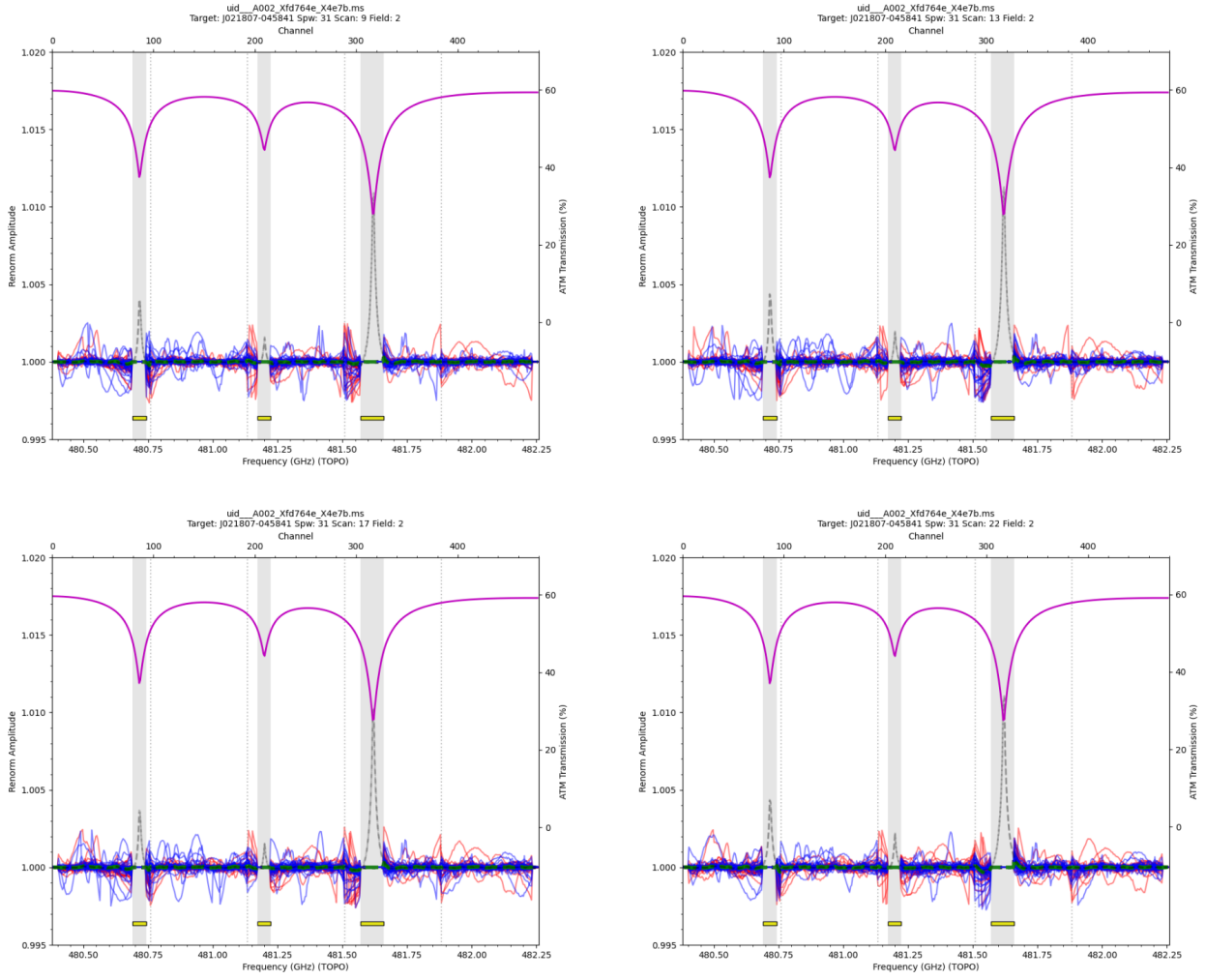


Figure 36: Screenshot of a PDF file as Figure 35 but where atmospheric regions have been automatically excluded and are covered by semi-opaque grey regions. It is clear to see that the excluded scaling spectra (visible by the grey dashed lines) are coincident with the three deep atmospheric lines shown 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.

An overall QA score is calculated for `hifa_renorm` that reflects the magnitude of the correction: If R_{th} is the threshold factor (default=1.02), the QA score is given as follows:

- $QA=1.0 - (1 - 0.90) * (R_f - 1.0) / (R_{th} - 1.0)$ for $R_f < R_{th}$ (i.e. green QA with values between 0.9 — 1.0)
- $QA=0.9 - (0.9 - 0.66)*(R_f - R_{th})/(2.5 - R_{th})$ for $R_{th} < R_f < 2.5$ (i.e. blue QA with values 0.67 — 0.9)
- $QA=0.66$ for $R_f > 2.5$ (i.e. yellow QA)

If the code is unable to calculate a max scaling factor R_f , or that factor is <1 or >2.5 , an additional QA score is created with the message "[ASDM UID]: Erroneous or unrealistic scaling values were found. Error calculating corrections."

uid__A002_Xe64b7b_X1be6d.ms

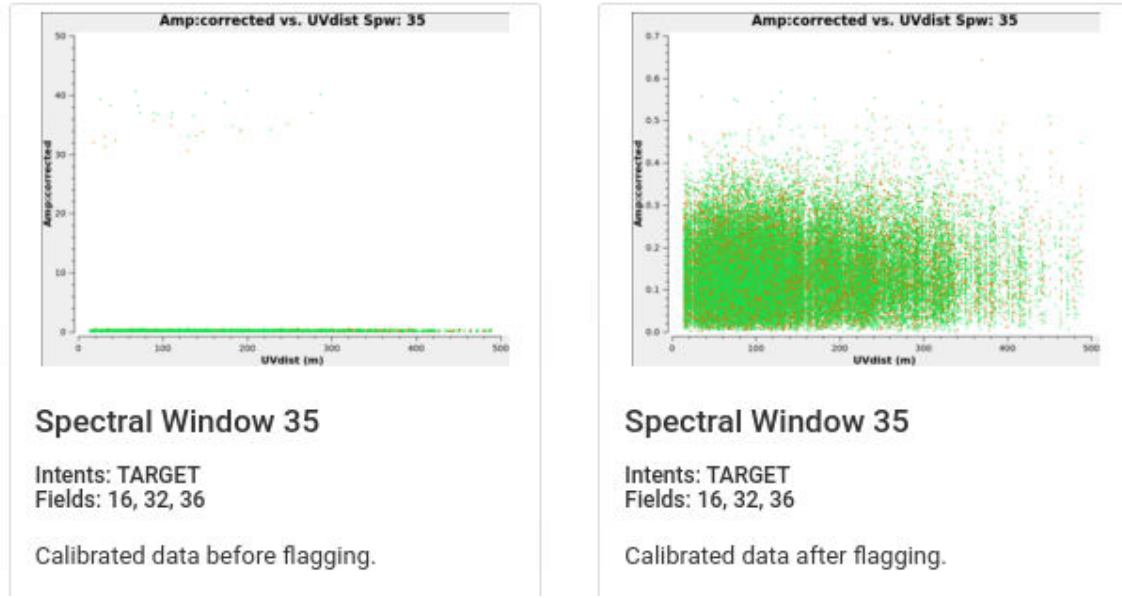


Figure 37: Example of flagging of high outliers in the calibrated science target visibility amplitudes in [hifa_targetflag](#).

9.24 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 in order to be declared an outlier. Because it must evaluate data on a per-field basis, this stage does add processing time, particularly in making the plots. So to save time, we only make the amplitude vs. time plots if flags are generated, and the amplitude 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 37). The QA score for this stage is equal to (1-percentage of data newly flagged).

9.25 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 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

24. Polarization Calibration

BACK

QA Score: 0.50 Session 'session_1' has residual polarization greater than 0.001 in SpW(s) 13, 15, 17 and 19. All QA Scores (1 yellow, 3 green)

Score	Reason
0.50	Session 'session_1' has residual polarization greater than 0.001 in SpW(s) 13, 15, 17 and 19.
1.00	Session 'session_1' has gain ratio RMS <= the threshold of 0.02 for all scans.
1.00	Session 'session_1' has D-terms solutions <= the threshold of 0.1 for all SpWs and antennas.
1.00	Session 'session_1' has gain ratios with deviations from 1 <= the threshold of 0.1 for all SpWs and all antennas.

Pipeline QA summary for this task.

This task creates polarization solutions for each polarization session of measurement sets.

Contents

- Sessions
- Polarization
 - Residual polarization after calibration
 - Derived polarization of the polarization calibrator
- Plots
 - Amplitude vs. Parallax Angle
 - Gain Amplitude Polarization Ratio vs. Scan
 - Cross-hand Phase vs. Channel
 - D-terms Solutions Gain vs. Channel
 - Gain Ratio RMS vs. Scan
 - X,Y amplitude vs. antenna
 - X/Y amplitude gain ratio vs. antenna
 - Real vs. Imaginary

Figure 38: Example of the WebLog QA for the [hifa_polcal](#) stage.

- 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 38):

- 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

The Stokes I, Q, U, V images and polarization intensity and angle of the polarization calibrator are imaged and computed in §9.30.

9.26 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 at this stage, so the WebLog shows a summary of that additional flagging, and a high flagging fraction will result in a low QA score (0.0 if the flag fraction on the science target is $\geq 50\%$, 1.0 if the flag fraction is $\leq 5\%$, and linearly interpolated between 0 and 1 for fractions between 50% and 5%).

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 include only the representative

target, and for mosaics, only the brightest field.

Outliers in these plots can indicate remaining bad data, particularly for sources that were self-calibrated (i.e. all calibrators except for check sources). To help identify these, a QA score is calculated 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 (see §7.3.8 of [Hunter et al. 2023](#) for specifics). These fits are done on a per-scan, per-polarization basis, and (new in PL2025) must be above set thresholds (10% or 10% per 2 GHz for amplitude offset/slope, or 6° or 6° per 2 GHz for phase offset/slope). If outliers are identified, a QA score is calculated with a value that decreases with the significance of the largest outlier, except for: amplitude-frequency offsets that are symmetric in XX YY (since these do not affect imaging), or phase-frequency offsets for CHECK sources (since these are diagnostic and do not affect source calibration). Details of any 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), 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.

9.27 hif_makeimlist: Set-up parameters for calibrator images

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

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 39). 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. The cell and imsize is chosen separately per band, so that more appropriate choices are made for multi-band data including BandToBandInterferometry.

The QA score is set equal to the fraction of the images in the imaging list compared to the total number expected.

9.28 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	Calibrator Dynamic Range	7-m array dynamic range correction factor – 1EB
≤ 1000	1	≤ 200	1
1000 – 3000	DR/1000	≥ 200	DR/200
≥ 3000	DR/3000		

See the description in §9.41 for more information and examples of the `hif_makeimages` stage.

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 39: 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.

There are three QA scores for this stage. A score of 0 occurs if the clean algorithm diverges, meaning that the image may not be properly deconvolved, and a score of 0.34 occurs if an expected image fails to get created. The third score is based on the ratio of the noise measured in the noise annulus (the region of the image with a primary beam response between 0.3—0.2 excluding any regions included in the clean mask - see [Hunter et al. 2023](#) for details) to the product of the theoretical noise and the dynamic range correction factor in the above table. If that value is 1 or lower, the QA score is 1.0, and if it is 5 or higher the QA score is 0, with a linear scaling in-between. Low QA scores for non-Check source calibrators may indicate the need for additional flagging and/or significant decoherence.

9.29 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. The QA score is set equal to the fraction of the images in the imaging list compared to the total number expected.

9.30 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 40](#)). In addition to the three QA scores for other [hif_makeimages](#) stages (§9.28), there is an additional yellow QA score and warning message if the Stokes I, Q, or U fits fail for a calibrator.

9.31 hif_makeimlist: Set-up parameters for check source images

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

29. Tclean/MakeImages

Make polarization calibrator images

BACK

QA Score: 1.00 RMS vs. DR corrected sensitivity. Field: J1058+0133 Intent: POLARIZATION SPW: 13,15,17,19 All QA Scores (2 green)

In this stage, images with significant emission are cleaned to a threshold of $2 \times$ (predicted rms noise) \times (dynamic range correction factor) using automasking. If a clean mask is not found automatically, then this threshold is doubled and the bulk of the whole image is used (PB=0.3). The dynamic range correction factor (abbreviated as "DR correction") accounts for the fact that sources with a high dynamic range will typically exhibit larger imaging artifacts, resulting in a noise level greater than an equivalent blank field. The artifacts are worse for poorer UV coverage, so different dynamic range (DR) correction factors are adopted for different antenna configurations (12m Array vs. 7m Array multi-EB, and for different targets (science targets vs. calibrators). See the Pipeline User's Guide for details. The DR correction adopted is a function of the dirty dynamic range (abbreviated as "Dirty DR"), which is defined as the peak intensity divided by the theoretical rms sensitivity delivered by the visibilities.

Polarization Calibrator Fit Results



Image Details

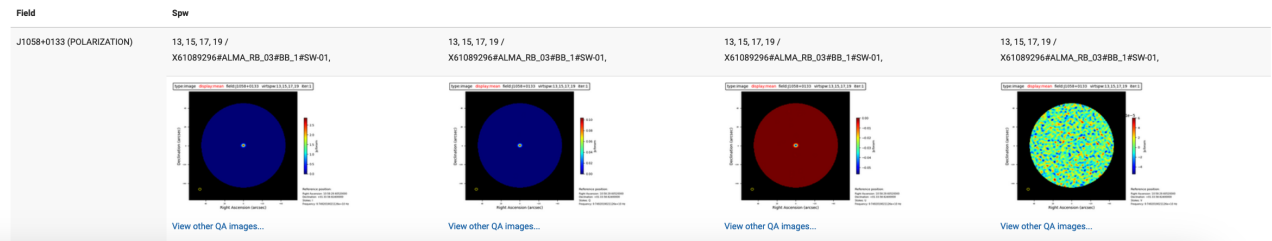


Figure 40: Example of the hif_makeimages stage for the Polarization calibrator (§9.30). First it displays the polarization fit results for each session along with the polarization intensity and angle plots, followed by Stokes I, Q, U, and V images. Clicking on the thumbnails will enlarge the image. Clicking the [View other QA images](#) link will bring up the detailed image page.

Prepare to create check source images, one per EB per spw. The QA score is set equal to the fraction of the images in the imaging list compared to the total number expected.

9.32 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 source and shows a table of the fit results and derived quantities (Figure 41). The table displays the following diagnostic values:

- Check source absolute position offset from the catalogue position, in mas and synthesized beams – this can help evaluate positional shifts due to imperfect phase transfer. A warning notification appears if the offset is $> 0.30 \times \text{synthesized beam}$ (was 0.15 prior to PL2020).
- The ratio of the fitted peak intensity to the fitted total flux density – this can help to assess phase decorrelation, but is also sensitive to low SNR or the presence of resolved emission. A warning notification appears if this value is < 0.7 .
- The ratio of the fitted total flux density to the `hifa_gfluxscale`-derived flux density – this can help evaluate flux scale transfer, but can be affected by low SNR. A warning notification appears if this value is < 0.8 .

The per EB / spw images 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 (§9.41.4).

In addition to the three QA scores for other `hif_makeimages` stages (§9.28), there is an additional QA score per EB/spw computed as an aggregate of 3 subscores as follows:

- $\text{Score1} = 1 - \min[1, \text{positional offset}/\text{beam size}]$
- $\text{Score2} = \max[1, \text{Fitted (Peak Intensity / Flux Density)}]$
- $\text{Score3} = \min[2 - (\text{Fitted} / \text{gfluxscale}) \text{ Flux Density}, (\text{Fitted} / \text{gfluxscale}) \text{ Flux Density}]$

The final value for this QA score is the geometric mean of the above three scores, i.e. $\sqrt{\text{Score1} * \text{Score2} * \text{Score3}}$. 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.

9.33 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 42. 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 ¹⁴ of 0.0 – 2.0 that best matches the PI's requested angular resolution is chosen automatically:

- 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.

¹⁴Smaller 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.

23. Tclean/MakelImages

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 41: Check source imaging diagnostic table.

- 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.

There is a QA score based on the identification of the representative target & frequency (1.0 if identified, otherwise 0.5) and the result of comparing the predicted beamsize to the PI desired range of angular resolutions (AR), with the following values:

- QA=1.0 (green) if both the major and minor axis with `robust=0.5` are within the AR range.
- QA=0.85 (blue) if both the major and minor axis are within the AR range with a `robust` value other than 0.5 but between 0 – 2.
- QA=0.50 (yellow) if at least one axis is outside the AR range, but the beam area is within the range of areas corresponding to the AR range.
- QA=0.25 (red) if the beam area falls outside the areas corresponding to the AR range for any value of `robust` between 0 – 2, since this means that the imaging products are unlikely to meet the PI goals.

9.34 hif_checkproductsizes: Mitigation to avoid overly long runs and/or tclean failures

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, and to prevent `tclean`

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 `clean` robust weighting parameter: `robust = 0.0, +0.5 (default), +1.0, and +2.0`. If the **"Min / Max Acceptable Resolution"** is available (\geq 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	<input checked="" type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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	<input type="checkbox"/>	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 42: 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).

failures on excessively large cubes. Figure 43 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:

- `maxcubysize` (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:

- $\text{mfssize} = 4. * \text{nx} * \text{ny} / 1\text{e}9$
- $\text{cubysize} = 4. * \text{nx} * \text{ny} * \text{nchan} / \text{nbin} / 1\text{e}9$
- $\text{productsize} = 2.0 * (\text{mfssize} + \text{cubysize})$

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. When a full polarization IQUV imaging recipe is used, this is inflated by a factor of 5, to account for the 4 axes in the IQUV image, in addition to the initial Stokes I image.

The mitigation cascade is as follows:

Step 1: If $\text{cubysize} > \text{maxcubysize}$, for each spw that exceeds `maxcubysize`:

- If $(\text{nchan} == 3840)$ or $(\text{nchan} \text{ in } (1920, 960, 480) \text{ AND online channel averaging was NOT already performed})$, then set `nbin=2`.
- 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 $\text{PB}=0.2$. For an image of width d , the response level at the edge will be $\text{PB}=\exp(-d^2*\ln(2)/\text{FWHM}^2)$, the image size $d^2 \propto -\ln(\text{PB})$, and the required power level to create an image of size = `maxcubysize` is:

$$\text{PB_mitigation} = \exp(\ln(0.2) * \text{maxcubysize} / \text{current_cubysize})$$

- Then account for `imsize` padding: $\text{PB_mitigation} = 1.02 * \text{PB_mitigation}$
- Then limit the size reduction to $\text{PB}=0.7$: $\text{PB_mitigation} = \min(\text{PB_mitigation}, 0.7)$
- Then round to 2 significant digits: $\text{PB_mitigation} = \text{round}(\text{PB_mitigation}, 2)$

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

- If still too large, change the pixels per beam (cell size) from 5 to 3.25 (if `robust=+2`) or 3.0 otherwise.
- If still too large, *stop with error*, the largest size cube(s) cannot be mitigated.

Step 2: If $\text{productsize} > \text{maxproductsize}$

- If the number of science targets (single fields or mosaics) is greater than 1, reduce the number of targets to be imaged until $\text{productsize} < \text{maxproductsize}$. The representative target is always retained.
- If `productsize` still too large, repeat steps 1a, 1b, and 1c, recalculating `productsize` each time.
- If `productsize` is still large, *stop with error*, the `productsize` cannot be mitigated.

23. Check Product Size

[BACK](#)

Task notifications

QA Size had to be mitigated (nbins,field)

Warning! Could not translate spw name SW-1 to ID. Trying frequency matching heuristics.

Allowed maximum cube size: 40 GB
Allowed cube size limit: 60 GB
Predicted maximum cube size: 58 GB
Mitigated maximum cube size: 29 GB
Allowed product size: 350 GB
Initial predicted product size: 1.86e+03 GB
Predicted product size after cube size mitigation: 232 GB
Mitigated product size: 232 GB

Size mitigation parameters for subsequent hif_makeimlist calls

nbins	hm_imsz	hm_cell	field	spw
25:2,31:2,27:2,29:2	default	default	"569314"	default

Pipeline QA

Input Parameters

Tasks Execution Statistics

CASA logs for stage 23

- [View](#) or [download](#) stage23/casapy.log (276.9 KB)

Figure 43: Screenshot of the [hif_checkproductsz](#) 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 1.86 TB; after the mitigation, the maximum cube size is predicted to be 29 GB and the product size 232 GB.

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 $0.5 * \text{maxcubelimit}$, 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 if there was no mitigation, 0.85 if mitigation was performed, or 0.0 if an error was encountered. When the cube or product size cannot be mitigated, the following warning will appear at the top of the `hif_checkproducts` 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 43, 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.

9.35 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. The QA score is 1.0 if all standard products were successfully copied to the **products/** directory, otherwise it is set to 0.

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

9.36 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. The QA score is 1.0 if the new MS is created, otherwise 0.

9.37 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.

9.38 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). The QA score is set equal to the fraction of the images in the imaging list compared to the total number expected.

9.39 hif_findcont

9.39.1 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** in the pipeline/extern directory), which performs the rest of the work.

9.39.2 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. In the PL2024 release, the threshold on the moment0 image was raised from 5 to 8 for sources with strong emission in the moment0 image (peak/MAD>90). 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 44. 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). As of PL2024, the primary beam sensitivity mask is applied to the mask image in all cases, so that the thumbnail image shown on the weblog will be black where the sensitivity is below the standard level (20%).

9.39.3 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 $< \text{nchan}/13$ in order to preserve some large-scale frequency structure in the spectrum if the user has set ν_{sens} to be a large fraction of the spw bandwidth. For Cycle 10 data, for which the `spectralDynamicRangeBandwidth` 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. The `spectralDynamicRangeBandwidth` is typically equal to 1/3 of the user-specified expected linewidth of the representative target; however, if the User requested a "User"-defined bandwidth for sensitivity, then the OT copies that bandwidth value into `spectralDynamicRangeBandwidth`.

For older data, the heuristic introduced in PL2022 will be used instead: if the bandwidth for sensitivity (ν_{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 ν_{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 ν_{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 associated with an spw contain the word "cont" (case insensitive) and only one occurrence of "id=X" (case insensitive) where X is a non-negative integer, then smoothing is disallowed for that spw. Prior to PL2024, X had to be zero to prohibit smoothing.

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.

9.39.4 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.

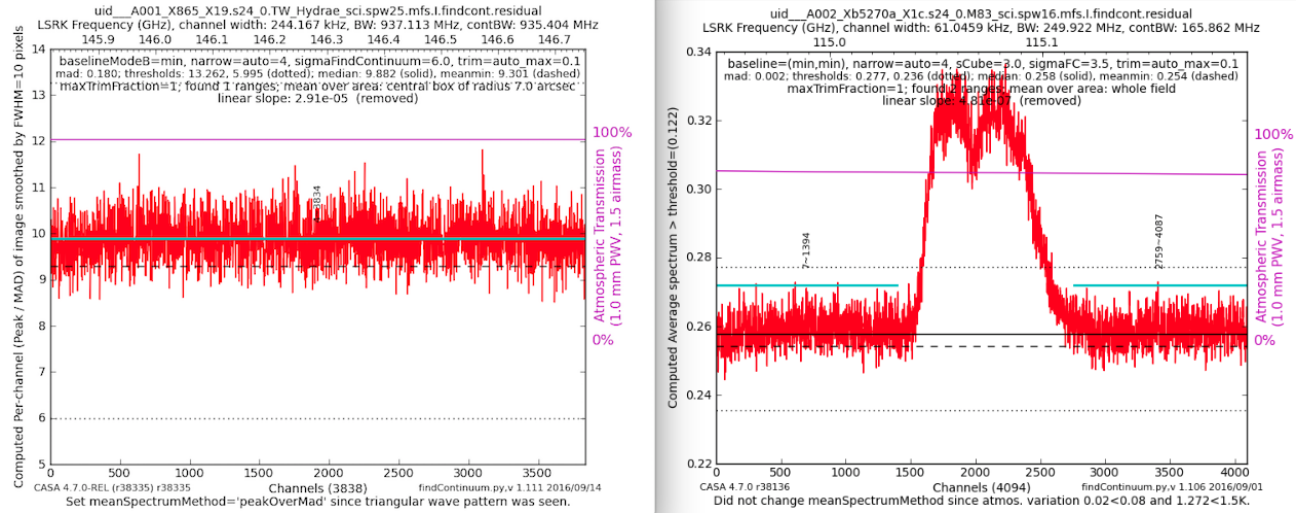


Figure 44: Two examples of `hif_findcont` plots, one with the entire window identified as continuum (left panel), and another with two identified continuum regions (right panel). The identified continuum range(s) are indicated by the horizontal cyan lines. Starting in PL2025, the “AllCont” label will be included on the plot in the left panel.

After each stage, a new `momDiffSNR` is computed to determine if additional work is needed. If `amendMask` or only `ExtraMask` 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 only `ExtraMask`), then the `autoLower` stage will be run in which the `sigmaFindContinuum` threshold is further reduced to try to eliminate the apparent remaining line contamination. A second iteration of `autoLower` is attempted if the first iteration did not reach the threshold.

The logic path followed is indicated by the logic path letter code in the upper legend of the plot (e.g., ADIEXY, where AD=`amendMask`, I=`intersectChannels`, E=`ExtraMask`, X=`autoLower`, Y=`autoLower2`). 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. Also, as of PL2025, the `momDiffSNR` is propagated to the table on the `hif_findcont` WebLog page. 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.

9.39.5 Results and output

The QA for this stage is 1.0 if continuum frequency ranges found for all spw, otherwise it is set equal to the fraction of spws for which a continuum range was identified, unless size mitigation fails, in which case the score is set to 0.0.

In addition to being displayed on the plots and in the WebLog table, the selected continuum frequency ranges are also printed to a file called **cont.dat** that is linked to at the bottom of the page. If this file already exists before [hif_findcont](#) is executed, then it will first examine the contents (see §7.6). 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 the [hif_findcont](#) task is not present in the PPR, then [hif_uvcontsub](#) will use all channels of each spw to perform the continuum subtraction.

If no line ranges are found by the findContinuum process, 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 $\geq 92.5\%$ 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. A final stage of analysis checks for large groups of channels whose intensities are all above the baseline (defined by the median of the blue points in the spectral figure0 and removes those channels from the continuum range as there might be weak (typically broad line) emission present. In this scenario, the prior AllContinuum label will be retained even though the final aggregate bandwidth will now be less than the original threshold.

9.40 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, usually using `fitorder=1`, however beginning in PL2024, `fitorder=0` is used if the LowBW or LowSpread condition was found for the spw in [hif_findcont](#). The WebLog for this stage reports the continuum ranges from [hif_findcont](#) in LSRK but translated into 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`. For any spw that was listed in **cont.dat** but had no ranges specified, then in PL2024 it will be assumed to be "all continuum", whereas in previous releases it was simply skipped and not written to the line MS.

The QA score is 1.0 if a continuum fit table was created, otherwise 0.

9.41 hif_makeimages: common task functionality

9.41.1 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.

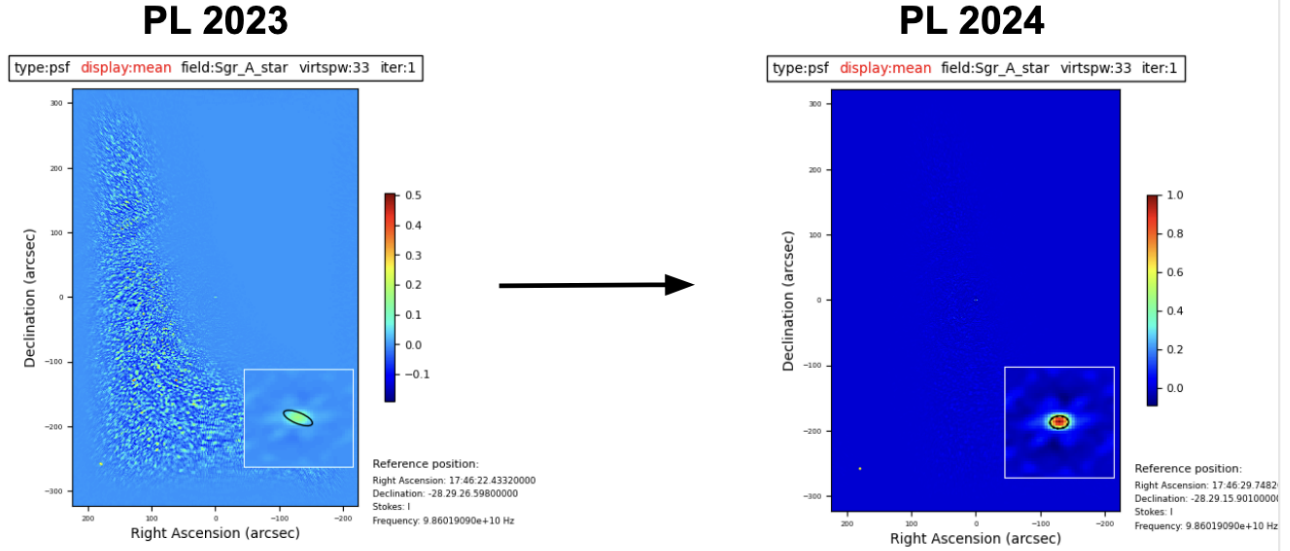


Figure 45: Irregularly shaped mosaics can result in poor psf (as shown) or even tclean crashing due to lack of data at the geometric center. This is automatically handled as of PL2024, by setting the `tclean psfphasecenter` parameter to an explicit mosaic pointing if an irregular mosaic is detected.

9.41.2 Full polarization IQUV imaging

In the `hif_makeimages` stage in the polarization recipes, full polarization Stokes IQUV images are made. The mask from the corresponding Stokes I imaging stage is used with a fallback of no mask (and no cleaning; only iter0) if the Stokes I mask is not found. So for example, `hif_makeimages` cube stage uses the mask from the `hif_makeimages` Stokes I output images (on a per spw basis). This mask is not displayed in the fullpol imaging stages, and is only shown in the corresponding Stokes I imaging stage. Similarly, the spectra from the masked region is not shown in the fullpol imaging stage.

Similarly, the `tclean` threshold is carried forward from the Stokes I imaging stage, and the threshold is recomputed as a fallback if the Stokes I threshold is not found. *Known Issue:* The dynamic range (DR) correction factor shown in the WebLog can be different between I and IQUV imaging in PL2025, but this distinction is immaterial since the threshold is not recomputed using the new DR.

9.41.3 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 ($< \text{minbeamfrac}$) noise-like automask regions, real emission can have all mask regions “pruned”, resulting in no clean mask for very compact, typically high SNR 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.

Automask parameter	7m-array	12m-array $b_{75} < 300\text{m}$	12m-array $b_{75} = 300\text{m to } 400\text{m}$	12m-array $b_{75} > 400\text{m}$
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

9.41.4 Cleaning Threshold

Images are cleaned to a threshold of $2 \times (\text{predicted rms noise}) \times (\text{dynamic range correction factor})$. 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 Dynamic range correction factor
≤ 20	1
20 – 50	1.5
50 – 100	2
100 – 150	2.5
≥ 150	$\max(2.5, \text{DR}/150)$

Source Dynamic Range	7m Array Dynamic range correction factor	
	1EB	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	$\max(2.5, \text{DR}/30)$	2.5
55 – 75	$\max(2.5, \text{DR}/30)$	3.0
≥ 75	$\max(2.5, \text{DR}/30)$	$\max(3.5, \text{DR}/55)$

In addition to this correction factor, 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 $5 \times (\text{predicted rms noise}) \times (\text{dynamic range correction factor})$.

9.41.5 WebLog and QA

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 46). 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

33. Tclean/MakeImages

Make target per-spw continuum images

BACK

Image Details

Fields

- 04191+1523 (TARGET)
- 04287+1801 (TARGET)
- 04288+1802 (TARGET)
- HH_30 (TARGET)
- IRAM04191 (TARGET)

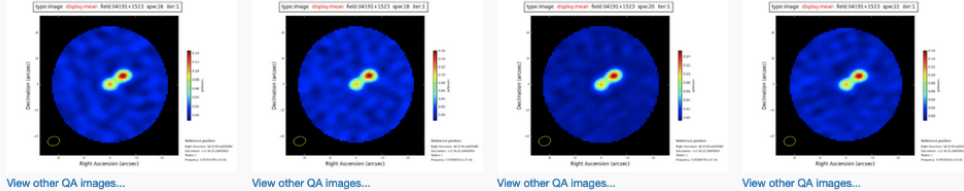
Field	Spw				
04191+1523 (TARGET)	16 / X1527843298#ALMA_RB_07#BB_1#SW-0	18 / X1527843298#ALMA_RB_07#BB_1#SW-0	20 / X1527843298#ALMA_RB_07#BB_2#SW-0	22 / X1527843298#ALMA_RB_07#BB_3#SW-0	
					
centre frequency of image	329.3312GHz (LSRK)	330.5892GHz (LSRK)	328.2968GHz (LSRK)	339.5162GHz (LSRK)	
beam	4.72 x 3.47 arcsec	4.67 x 3.46 arcsec	4.68 x 3.55 arcsec	4.52 x 3.48 arcsec	
beam p.a.	-77.9deg	-79.2deg	-76.8deg	-81.2deg	
final theoretical sensitivity	0.003 Jy/beam	0.0029 Jy/beam	0.0015 Jy/beam	0.0026 Jy/beam	
cleaning threshold	0.018 Jy/beam Dirty DR: 46 DR correction: 3	0.018 Jy/beam Dirty DR: 53 DR correction: 3	0.01 Jy/beam Dirty DR: 1e+02 DR correction: 3.5	0.015 Jy/beam Dirty DR: 60 DR correction: 3	

Figure 46: 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 47).

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 excluding any parts of the clean mask that extend into this region, 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 (see below).

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. 12.

There are three QA scores for all [hif_makeimages](#) stages (including calibrator imaging, so already mentioned in §9.28): A score of 0 occurs if the clean algorithm diverges, meaning that the image may not be properly deconvolved; a score of 0.34 occurs if an expected image fails to get created. The third score is based on the ratio of non-pbcor image rms mentioned above to the product of the theoretical noise and the appropriate “Dynamic range correction factor” from the above tables. If that value is 1 or lower, the QA score is 1.0, and if it is 5 or higher the QA score is 0, with a linear scaling in-between. Some images stages have additional QA scores, which are detailed in the appropriate subsections below.

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

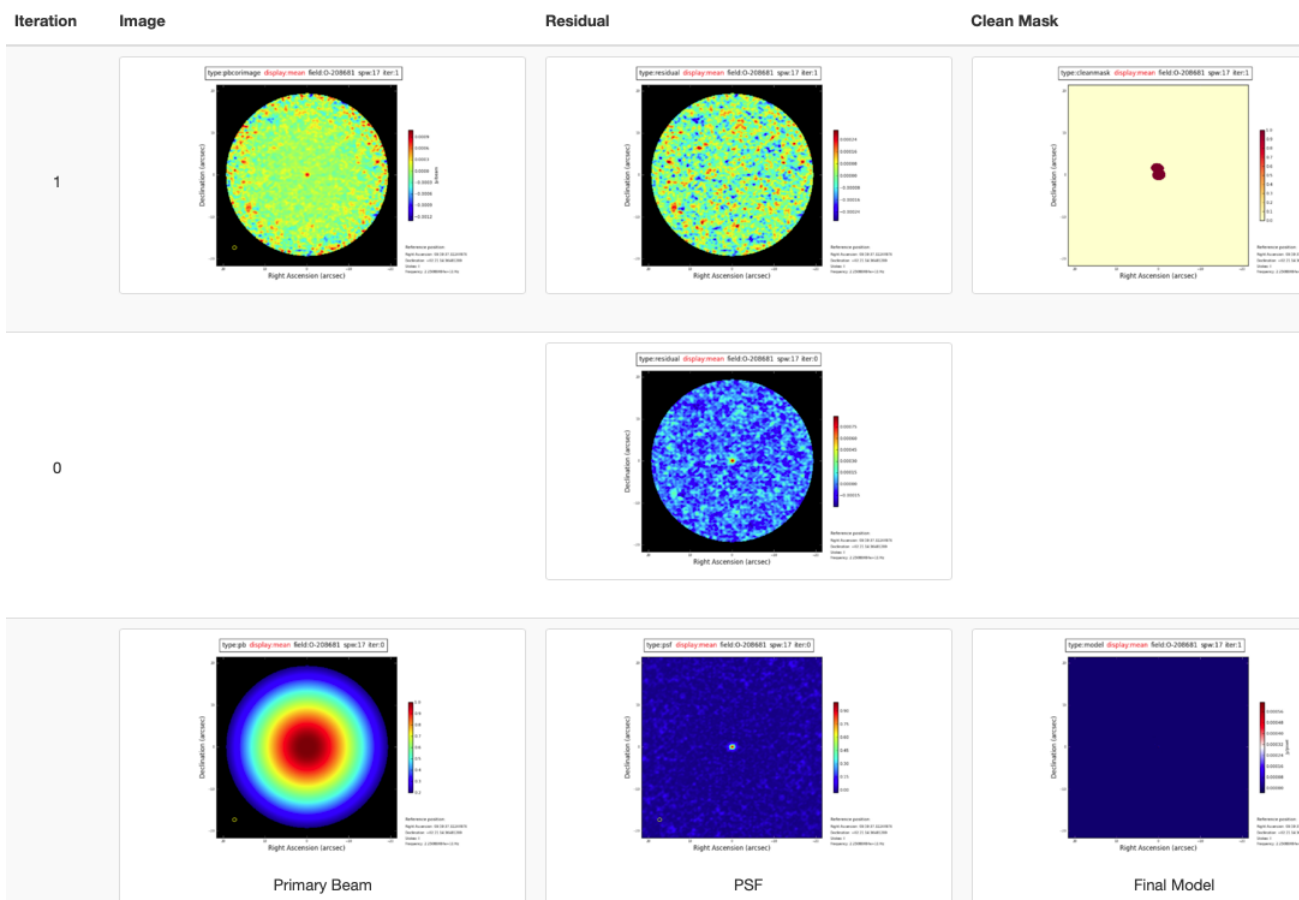


Figure 47: Details page that is displayed after clicking on the [View other QA images](#) link on the [hif_makeimages](#) WebLog page.

9.41.6 Data types and imaging recipe

As described in §9.49, 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 CONTLIN (non-continuum-subtracted) and LINE (continuum subtracted). The default imaging recipe has the following stages which reference the REGCAL_-CONTLIN_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
- [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 9.49), the same imaging stages may then be repeated with the same heuristics, but with the datatypes of SELFCAL_-CONTLIN_SCIENCE and SELFCAL_LINE_SCIENCE, respectively.

In the case of the full polarization imaging recipe, the following stages are use:

- [hif_makeimages](#)(mfs): Target per-spw continuum images, Stokes I only
- [hif_makeimlist](#)(mfs_fullpol): Set-up parameters for target per-spw continuum images, full pol IQUV
- [hif_makeimages](#)(mfs_fullpol): Target per-spw continuum images, full pol IQUV
- [hif_makeimlist](#)(cont): Set-up parameters for target aggregate continuum images, Stokes I only
- [hif_makeimages](#)(cont): Make target aggregate continuum images, Stokes I only
- [hif_makeimlist](#)(cont_fullpol): Set-up parameters for target aggregate continuum images, full pol IQUV
- [hif_makeimages](#)(cont_fullpol): Make target aggregate continuum images, full pol IQUV
- [hif_makeimlist](#)(cube): Set-up parameters for target cube images, Stokes I only
- [hif_makeimages](#)(cube): Make target cube images, Stokes I only
- [hif_makeimlist](#)(cube_fullpol): Set-up parameters for target cube images, full pol IQUV
- [hif_makeimages](#)(cube_fullpol): Make target cube images, full pol IQUV
- [hif_makeimlist](#)(repBW): Set-up parameters for target representative bandwidth cubes, Stokes I only
- [hif_makeimages](#)(repBW): Make target representative bandwidth cubes, Stokes I only
- [hif_makeimlist](#)(repBW_fullpol): Set-up parameters for target representative bandwidth cubes, full pol IQUV
- [hif_makeimages](#)(repBW_fullpol): Make target representative bandwidth cubes, full pol IQUV

9.42 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_checkproductsizes](#) stage.

9.43 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_checkproductsizes` stage.

9.44 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).

9.45 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_checkproductsizes` stage.

9.46 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 48). 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.

9.46.1 hif_makeimages: specmode=cube additional QA

The efficacy of the line-free spectral ranges determined by `hif_findcont` is assessed by creating moment 8 (maximum value along the frequency axis) and moment 10 (minimum value along the frequency axis) images of the line-free ranges of the cube, referred to as the `mom8_fc` and `mom10_fc` images. In an ideal case, these images will only contain noise and their statistics should be similar. To test this, three metrics are calculated (see §7.4.3 of [Hunter et al. 2023](#) for specifics): `PeakSNR` = the peak above the median in the `mom8_fc` image compared to the median absolute deviation (MAD) measured in the line-free channels of the cube; `HistAsym` = the difference between the absolute value of the intensity histogram constructed from the `mom8_fc` image with that constructed from the `mom10_fc` image; and `MaxSeg` = the largest “segment” of contiguous pixels with values above a threshold in the `mom8_fc` image. If `PeakSNR` > 5 and `HistAsym` > 0.2, or if `PeakSNR` > 3.5 and `HistAsym` > 0.05 and `MaxSeg` > 1 beam area, then the QA 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 (resulting in lower QA scores for larger segments). If that condition is not met, then the score is given by the value of the error function (higher QA scores for a smaller segment).

The practical result of this scoring is that if the condition is met (there is brighter emission and noticeable histogram asymmetry or weaker asymmetry and weaker but more extended emission), it will result in a QA

Make target cubes

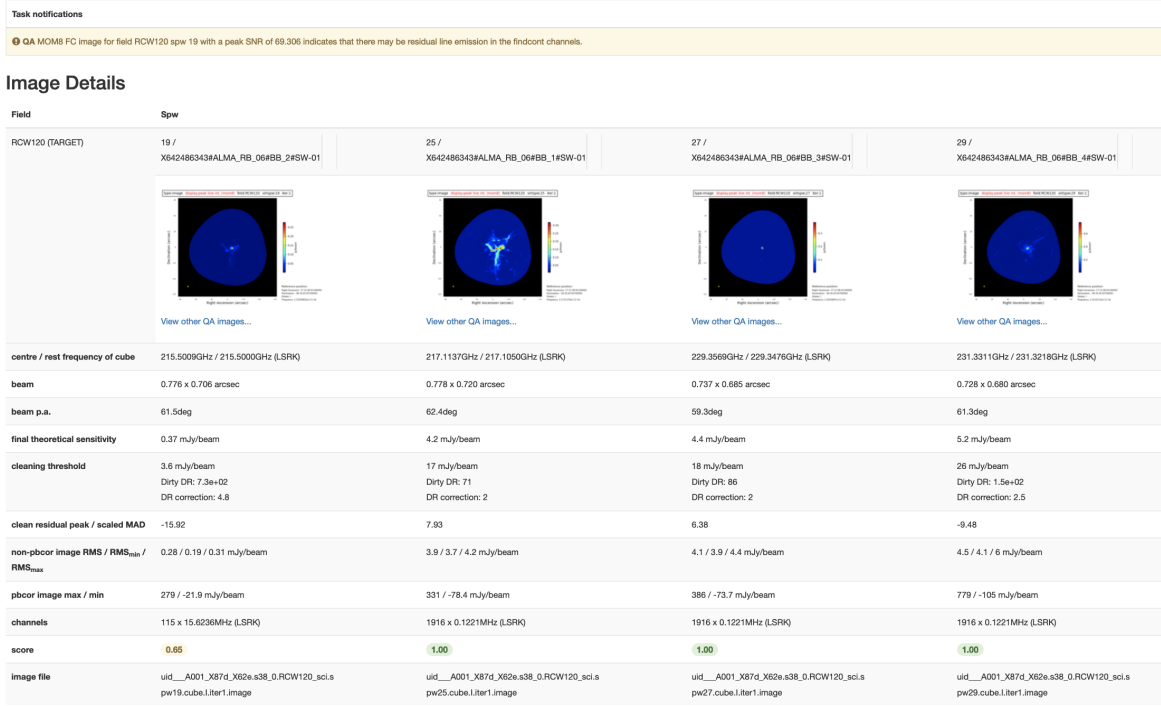


Figure 48: Example of [hif_makeimages](#) WebLog page for image cubes. The first spw has a reduced QA score based on the **mom8_fc** image as described in §9.46.1.

score between 0.33–0.65 (yellow), with poorer scores for more extended emission. Otherwise, the QA score will be between 0.67–1.0 (blue to green) with better scores for less extended emission.

These scores are defined to identify if there is noticeable emission in the **mom8_fc** image, but that does not necessarily mean that something has gone wrong with continuum subtraction or cube imaging. It could be that there is spectrally unresolved line emission, or a line forest, or that there are low SNR features that are missed by the [hif_findcont](#) algorithm. Yellow QA scores identify cases that merit manual examination (by looking at spectra through the cube at locations of peaks in the **mom8_fc** image). **In the majority of cases, the cubes with poor values of this QA score do not need to be regenerated**, because a small change in the continuum ranges will have little to no effect on uv-based continuum 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), or use different channel ranges when making moment images.

There is a final possible modification to the QA scores for cubes, based on whether there are significant deviations in the synthesized beam shape across the cube (which is displayed on the cube details page - see the lower right plot of Figure 49). If the major axis differs by more than a factor of two from the median of all channels, the QA score is reduced by 0.11. If more than 10 channels are deviant, the QA score is reduced by 0.34.

9.47 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.

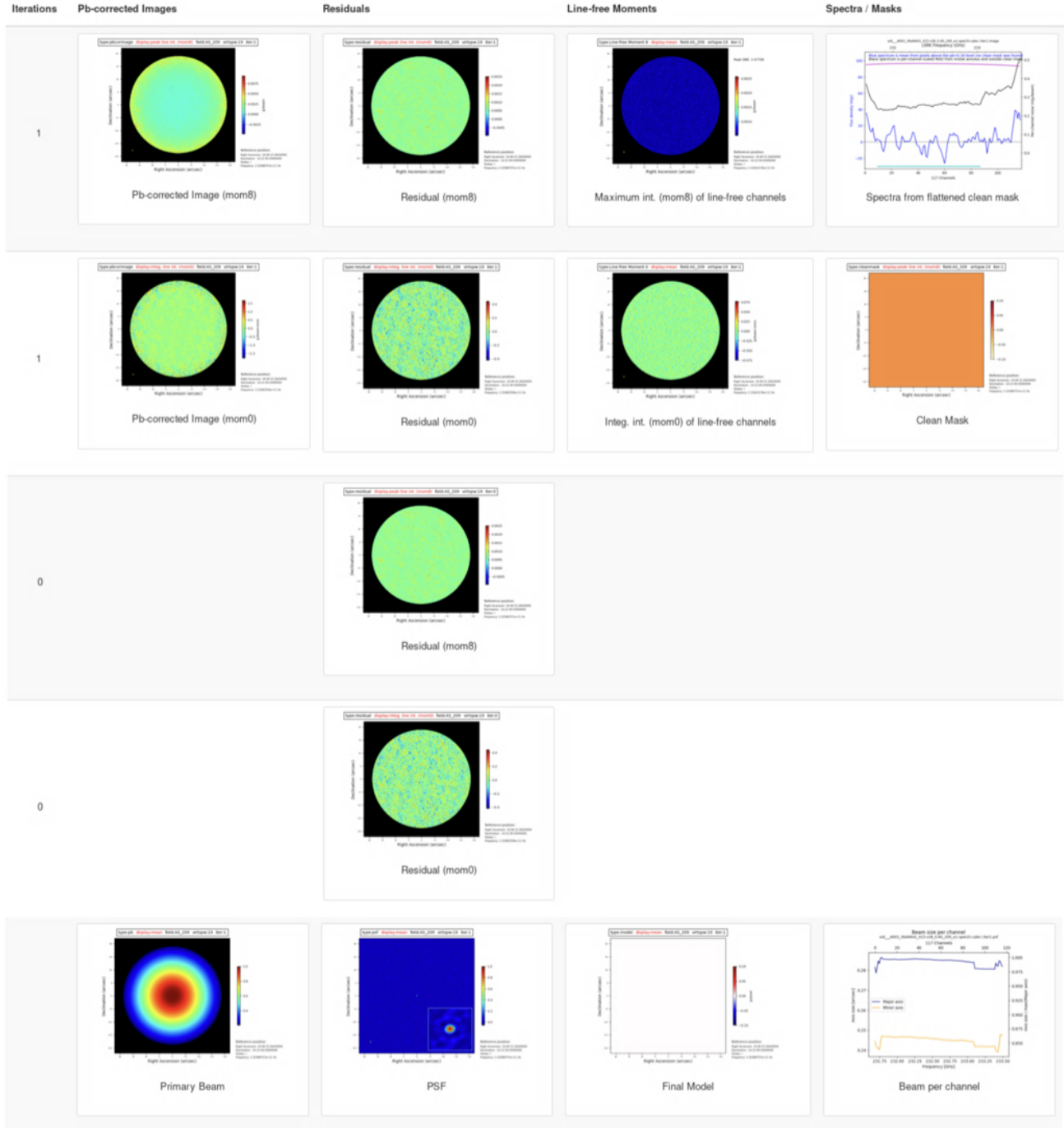


Figure 49: Example of an image cube details page including diagnostic images specific to cubes, such as the line-free moment 0 and moment 8 images (top two rows, third column from the left), the per-channel noise and spectra (upper right), and the synthesized beam major and minor axis across the channels (lower right).

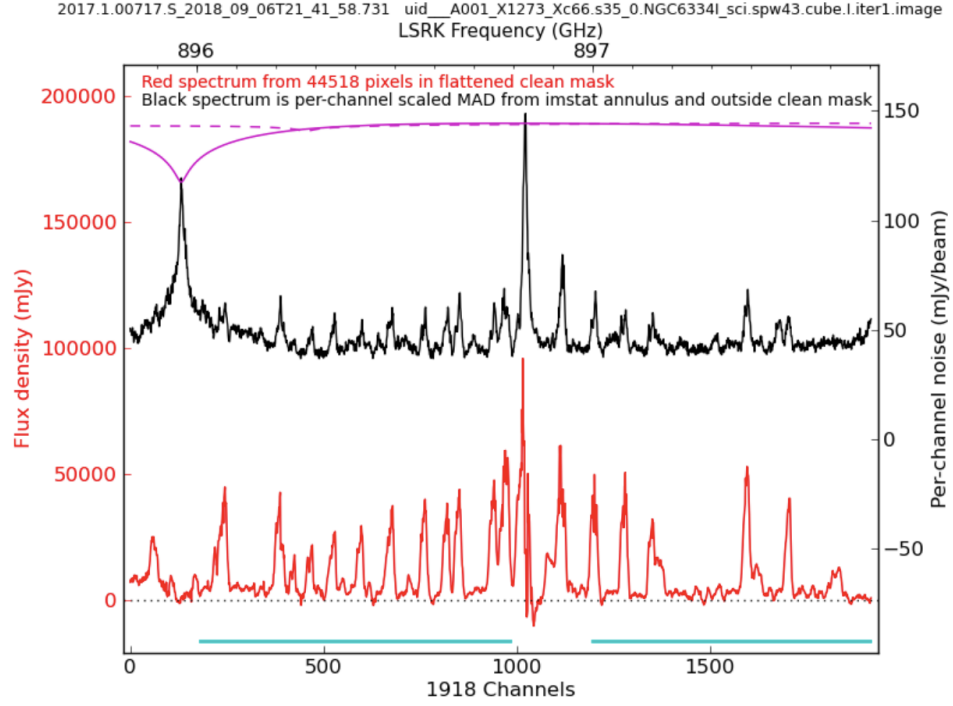


Figure 50: Example of a cube spectrum (red) constructed from pixels inside the clean mask, overlaid with a noise spectrum (black) constructed from the (portion of the) noise annulus outside the clean mask.

9.48 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).

9.49 hif_selfcal

9.49.1 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.

9.49.2 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 includes additional data types tracking beyond the traditional on-disk MS "DATA" and "CORRECTED" columns. These 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](#).
- **SELFAL_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](#).
- **SELFAL_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 calibrated visibility data labeled as REGCAL_CONTLINE_SCIENCE and REGCAL_LINE_SCIENCE and stored in the CORRECTED columns of *_targets.ms and *_targets_line.ms, respectively.

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.

9.49.3 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 (and per-sub-field in the case of mosaics) 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 the minimum of $3\times$ the rms determined from the final successful self-calibration iteration for that source and the clean threshold used to make the initial image prior to self-calibration. The latter is included to ensure that the final, self-calibrated image is always cleaned at least as deep as the initial image.

9.49.4 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 *gain*type=“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 spw-map 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 *gain*type=“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 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 the 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 $5\times$ 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.

9.49.5 Solution Acceptance/Rejection

Several 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.

[hif_selfcal](#) further requires that the “near-field” SNR must also increase in the “post” image as compared with the “pre” image. Statistics dubbed with the moniker “near-field” (or “NF”) that use an 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. The intention of this near-field mask is to measure the rms in regions close to the source, where artifacts are expected to be particularly impactful so that the rms isn’t washed out by large empty regions of the image.

For the inf_EB solution interval, a slight ($< 2\%$) reduction in the SNR and/or the NF SNR is allowed to enable the routine to go on to additional, shorter solution intervals, though if this is allowed and then there are no further successful solution intervals, the inf_EB solution is later failed.

Finally, though some amount of flagging due to selfcal can lead to small increases in the rms, large increases $> 5\%$ in the rms, even with an accompanying increase in the SNR and NF SNR, are not allowed. If any of these criteria are not met, then the solution interval is deemed unsuccessful.

For mosaics, these statistics are computed and evaluated on the mosaic as a whole, and also for each individual component field of the mosaic. To avoid the cost of imaging a large number of sub-fields individually, [hif_selfcal](#) instead corrects the full-mosaic image for the primary beam, makes a sub-image of each field, and then applies the single field primary beam in order to approximate the image that would have been made from an individual imaging run. In the event that an individual sub-field fails a given solint, the solutions for the most recent successful solint for that sub-field are re-applied (if available), and the image of the mosaic as a whole is remade prior to evaluating the success of the mosaic as a whole. For a selfcal solution interval to be considered successful for a mosaic, the mosaic as a whole must succeed, and at least one sub-field must also be improved.

9.49.6 Weblog and QA

The [hif_selfcal](#) stage Weblog page shows a summary table describing the targets considered, the solution intervals to be considered, whether those solution intervals were actually attempted and/or applied, 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, 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. For mosaics, there is an additional per-field sub page that repeats all of the same information, but on a field-by-field basis. An example Weblog for the [hif_selfcal](#) stage is shown in Figure 51.

39. Self-Calibration

Self-calibration using the science target visibilities

QA Score: 0.98 Self-calibrations applied for IRS48 (Band 7).

List of Self-cal Targets

Field	Band	SpW	Phasecenter	Cell	Imsize	Solints to attempt	Success	Cont. 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 Targets Summary: All attempted solution intervals (solints) are shown in **bold**. If a solint is highlighted in **blue**, it represents a final applicable solint.

Self-cal Target Details

IRS48 (Band 7) [back to top](#)

[SUMMARY](#)

[PER-SOLINT DETAILS](#)

Data Type	Initial	Final	Brightness Dist. / Ratio
Image			
Integrated Flux	191.718 ± 2.111 mJy	199.595 ± 2.109 mJy	1.041
SNR	484.096	3019.555	6.238
SNR (N.F.)	485.179	2987.056	6.157
RMS	0.172 mJy/bm	0.032 mJy/bm	0.185
RMS (N.F.)	0.171 mJy/bm		0.187
Beam	0.330"x0.258" -84.328 deg	0.331"x0.258" -84.242 deg	1.001
Success / Final Solint	Yes / int		
Stop Reason	None		

Figure 51: 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.

The QA scores for this stage are assigned as follows:

- QA=1.0 if self-calibration is not attempted because the estimated SNR is too low.
- QA=0.99 if self-calibration is attempted, but does not succeed and is therefore not applied.
- QA=0.98 if self-calibration is attempted, succeeds and is therefore applied.
- QA=0.85 if self-calibration is attempted, succeeds and is therefore applied, but the RMS gets worse for at least one source.
- QA=0.9 if a new mode (at the moment mosaic self-calibration) is run.
- QA=N/A if self-calibration is not attempted due to an unsupported mode (e.g. ephemeris).

9.50 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.

9.51 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 stage. For a detailed description of parameters for each task, please refer to the [ALMA Pipeline Reference Manual](#). Each stage has its own set of QA scores and criteria to trigger its different values. Details of the QA scoring can be found in § 10.13.

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”). Two types of plots can be found; one shows only the on-source positions and the other indicates all positions including OFF positions (Figure 52). In these plots, the red circle indicates the beam size of the antennas and its location corresponds to the starting position of the raster scan. The red (small) dot indicates the last position of the raster scan. The green line represents the antenna slewing motion. In the right panel of Figure 52, 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 gray dots indicate flagged data. [hsd_importdata](#) generates pointing pattern plots with ephemeris correction in addition to the plots without correction if the target is a moving object (*e.g.*, solar system bodies).

10.2 hsd_flagdata

The WebLog for the [hsd_flagdata](#) task shows the summary of flagged data percentage per MS due to various reasons, including what happened in quality assurances, online flagging, manually inserted files (***flagtemplate.txt**), shadowing, unwanted intents, autocorrelation (always disabled), edge channels, pointing issue, and low transmission. Note that the value in the “Before Task” column of the summary table corresponds to the percentage of flagged data by binary data flagging (BDF). The reasons for the flagging are also displayed visually as a function of time.

A new additional flag task for pointing outliers has been implemented in PL2025. The online flags generally remove OFF positions from the final map. However, online flags do not completely remove OFF positions in some cases. In that case, the pipeline attempts to create large maps including OFF positions and finally crashes. To prevent such a case, this new pointing outlier flag works as a safety net. The pipeline calculates the threshold of the map size, and data points outside it are flagged.

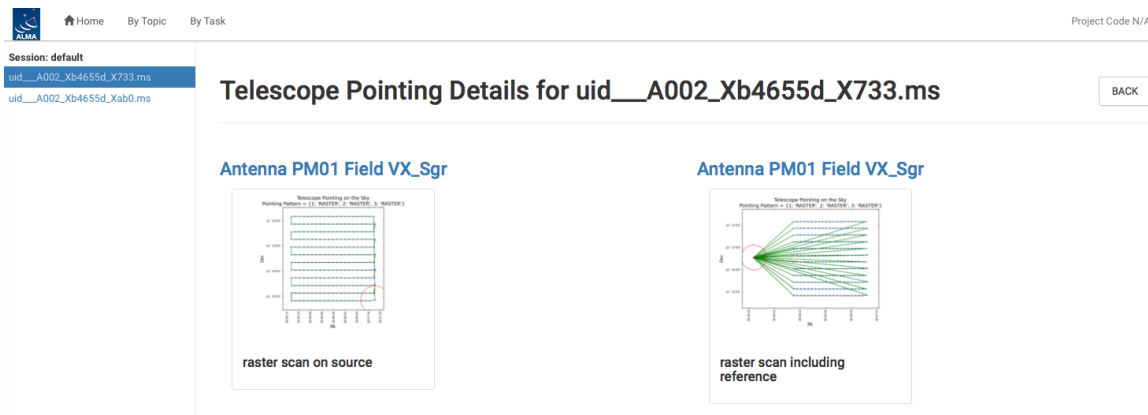


Figure 52: The detailed page of Telescope Pointing on the MS summary page.

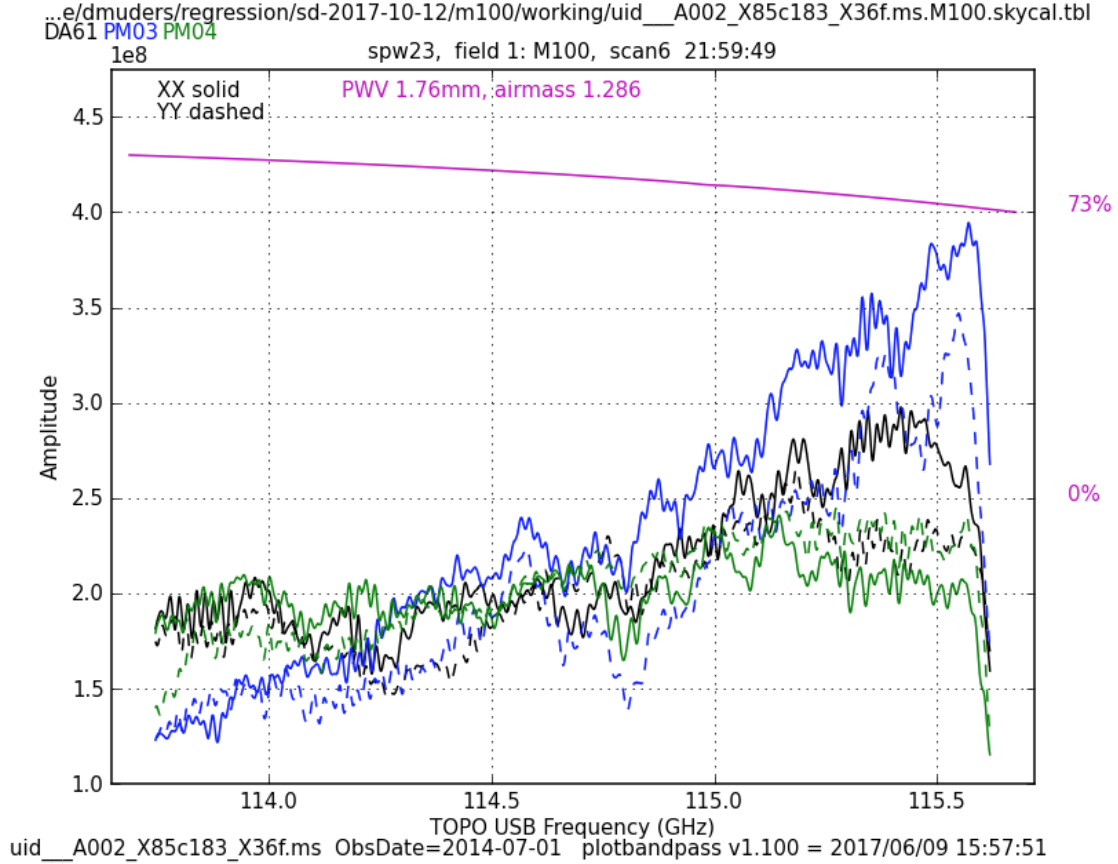


Figure 53: An example of an OFF spectrum. Different antennas are shown in different colors. The atmospheric transmission curve is shown in magenta.

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 and per antenna. The atmospheric transmission curve is shown as a reference in the spectra.

10.4 hsd_tsysflag

This page shows the Tsys spectra per spectral window per antenna after heuristic flagging has been applied.

10.5 hsd_skycal

The WebLog shows the integrated OFF spectra per spw and per source, for each MS. 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 53). The different colors indicate different antennas. The magenta lines indicate the atmospheric transmission curves. Time-averaged (over a certain duration) plots of the OFF spectra are also available on this page for the purpose of assessing the time variability of the spectra.

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

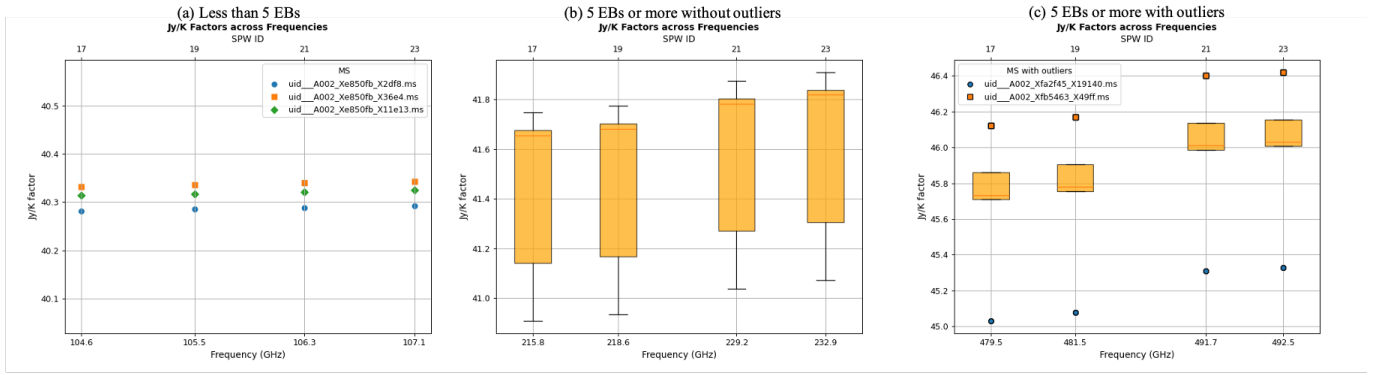


Figure 54: Plots of Jy/K conversion factors. Panel (a) indicates the case of a MS with less than 5 EBs. The scatter plot is applied. Panels (b) and (c) show examples for MSs with 5 EBs or more without and with outliers, respectively. The box plot is applied to both cases. Outliers are plotted as points, and the corresponding EBs are indicated.

10.6 hsd_k2jycal

This page shows the list of Kelvin-to-Jansky conversion factors that Pipeline has read from a file “jyperk_query.csv” or “jyperk.csv”, which contains 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 by the pipeline.

In addition to the list of values, plots showing the applied Kelvin-to-Jansky conversion factors are available at the top of this page (Figure 54). If the MS contains less than 5 EBs, a scatter plot is created (Figure 54a). For MOUSs that contain 5 EBs or more, a box plot is created as shown in Figures 54b and 54c. The definition of outlier follows matplotlib https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.boxplot.html. If an outlier is detected, the corresponding EB is plotted as a point and the EB’s name is indicated in the plot.

10.7 hsd_apprcal

This page summarizes a list of the calibrated MSs with the name of the applied Tsys, Sky, and amplitude (Kelvin-to-Jansky conversion) calibration tables. It shows frequency-averaged-amplitude vs. time plots, and time-averaged-amplitude vs. frequency plots obtained after calibration. It also displays the heuristic plots of a QA score which evaluates the amplitude difference between the XX and YY polarizations (Figure 55). These plots are useful to detect instrumental problems such as receiver instabilities or leakages because these problems are typically dependent on the polarization component.

10.8 hsd_atmcor

This stage evaluates and applies the best atmospheric model to correct residual atmospheric line effects in the science target caused by elevation differences between the ON_SOURCE and OFF_SOURCE positions. There is a list of the calibrated MSs with model parameters of atmType, h0, and dTem_dh. Here, atmType, h0, and dTem_dh, are the atmospheric type, scale height for water vapor, 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. This page also shows the integrated spectra (amplitude vs frequency) after the atmosphere correction. The magenta curves plotted above the spectra in each panel show the atmospheric transmission. The integrated spectra before the correction can be seen in the page of [hsd_apprcal](#). The details of the atmospheric models are described in [Sawada et al. 2021, PASP, 133c4504S](#).

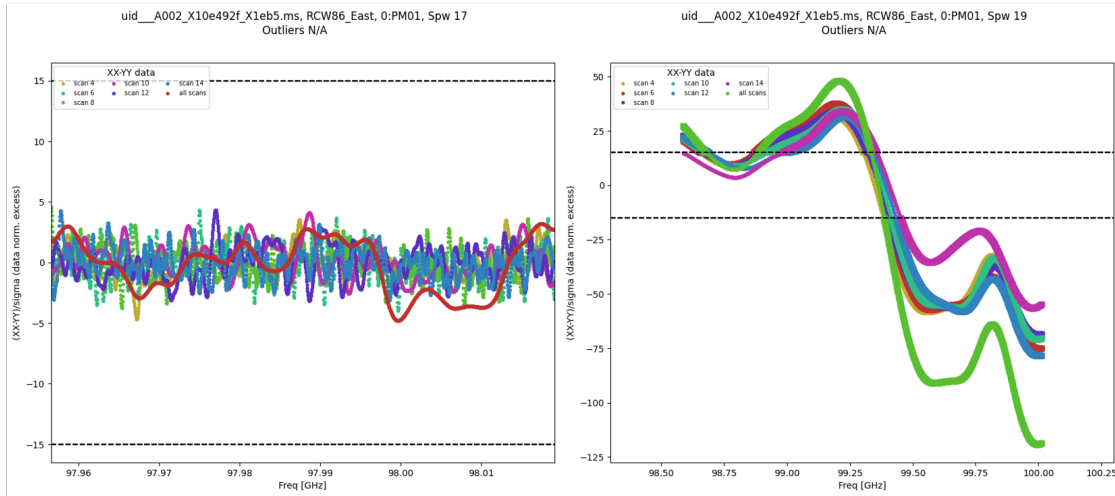


Figure 55: Heuristic plots for the amplitude difference between the two polarizations. The left and right panels are examples of a good case and a case that needs attention, respectively.

10.9 hsd_baseline

This task generates baseline fitting tables and subtracts the baseline from the spectra. Spectral lines are detected by clustering analysis for each spectral window, and line-free channels are used for baseline fitting.

Spectral data before/after baseline subtraction

The first three sections of the [hsd_baseline](#) front page of the WebLog show grids of spectra per source. The panel on top of each grid shows the spatially averaged spectra. The grid plots in the first and third sections correspond to the spectra before and after the baseline subtraction, respectively. Only one representative spectrum is shown for each grid cell. The representative positions are selected as the nearest valid (not fully flagged) points to the average coordinates of all points associated with a specific grid. Examples of the determination of the representative positions are indicated in Figure 56. The grid plot in the second section is obtained by averaging all the spectra associated with each grid before baseline subtraction (see Figure 57). Averaging the data improves the S/N ratio, making the spectral line features more prominent.

These plots, which appear after clicking the [hsd_baseline](#) page of the WebLog, show a representative case of gridding by using a certain antenna: the grids are made in R.A./Decl. coordinates. The red line (basically toward the horizontal direction) overplotted on the spectrum of each grid indicates the fitted function to be used for baseline subtraction for spectral data before baseline subtraction, and 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 displayed. The magenta lines indicate the atmospheric transmission at each frequency. The cyan shaded regions indicate the channels containing emission lines that are identified in the entire map, which are excluded from the baseline fitting, and red thick bars above the spectrum 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 checked 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, some boxes can be used to set filters to plot spectral maps by antenna, field, spectral window, and polarization.

Fitting order determination

The default fitting function for the baseline is a cubic spline function. Fitting order is automatically determined by default. It can be disabled by specifying `fitorder` as a non-negative value. In this case, the value specified by `fitorder` will be used.

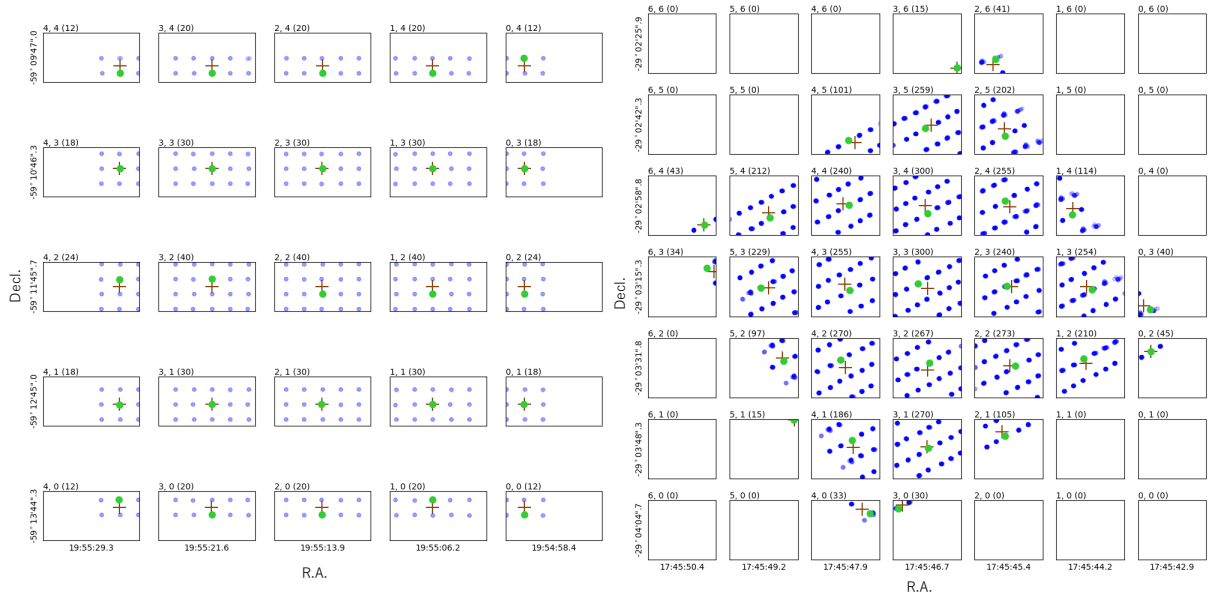
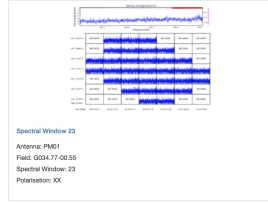
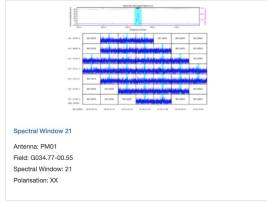
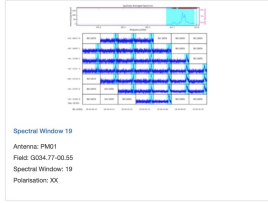
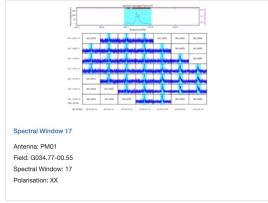


Figure 56: Examples showing how to define representative positions. Blue points are all pointings for each panel and the averaged coordinates are shown in brown crosses. The representative positions indicated in green circles are the representative positions of each grid, and these points are the nearest pointings from the averaged coordinates (brown points).

Spectral Data Before Baseline Subtraction

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

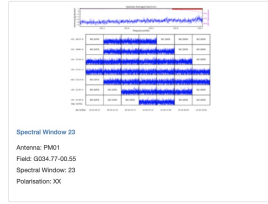
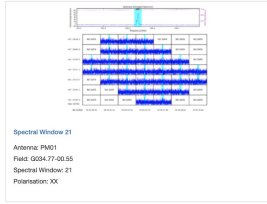
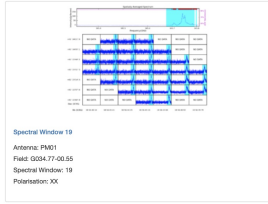
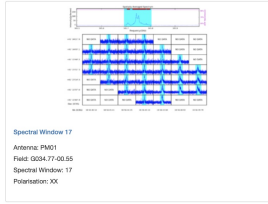
G034.77-00.55



Averaged Spectral Data Before Baseline Subtraction

Pointed data are obtained by averaging all the spectral associated with each grid. Averaging the data improves S/N ratio so that spectral line feature becomes more prominent and it can be easily compared with the line mask for baseline subtraction.

G034.77-00.55



Spectral Data After Baseline Subtraction

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

G034.77-00.55

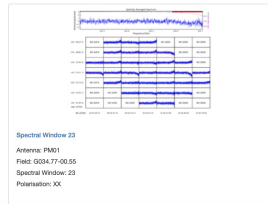
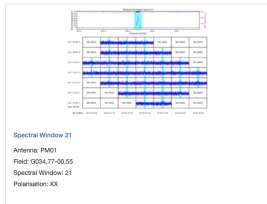
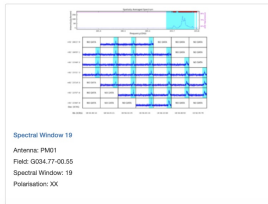
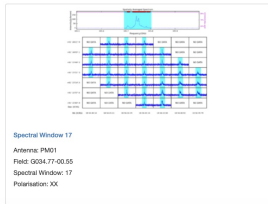


Figure 57: An example of the `hsd_baseline`. First three different rows are shown, but for one spw for display.

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 their 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:

1. If $1 < P_FFT < 3$, then $N_segment = 4$,
2. If $3 \leq P_FFT < 5$, then $N_segment = 5$,
3. If $5 \leq P_FFT < 10$, then $N_segment = 6$,
4. If $P_FFT \geq 10$, then $N_segment = F_FFT \times 2 + 2$, 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 $N_segment \times (Nch - N(mask)) / Nch$, where Nch is the number of spectral channels and $N(mask)$ is the number of channels masked. Note that unmasked channels are equally divided into segments, i.e., the number of unmasked channels is the same for all segments. Finally, the Pipeline performs the baseline fitting and baseline subtraction using cubic splines, which are a third-order polynomial that meet the boundary condition at the joint between the segments.

Mask range determination

When baseline fitting is performed, the emission/absorption channel ranges are masked out. The mask range is determined by comparing the deviation of spectral data from its median value with the threshold evaluated based on the median absolute deviation (MAD) of the data itself. Channels with absolute intensity above the threshold are detected as lines. This will detect both emission and absorption features. This detection process is repeated up to 10 times by excluding detected ranges from the process to detect weak lines. Each detected range is extended as long as the intensity is monotonically decreasing (or increasing in the case of absorption) to include the overall line feature (see Figure 60). To detect wide features, the same detection process is applied to the binned spectra. Binning widths depend on the number of channels. For 4096 channels, they are [1 (no binning), 4, 16, 64]. Threshold value for line detection also depends on binning width, **threshold** = $3.5 + \sqrt{\text{binning_width}} \times MAD$.

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 58 shows an example of the baseline flatness. QA score for the criterion is below.

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

Line validation

The plots related to line validation, which are basically useful for developers, are not displayed in the single-dish pipeline by default. The users can 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 59a) 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.

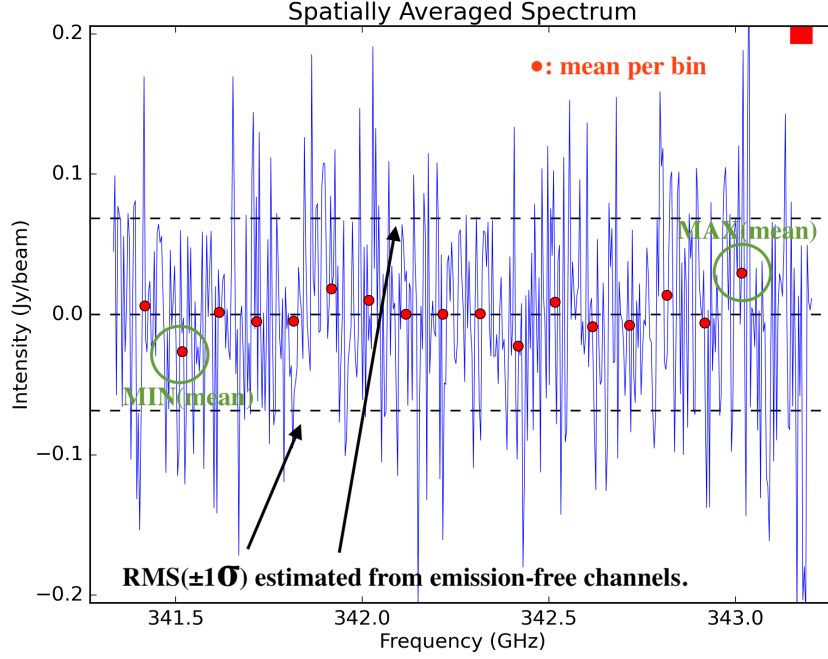


Figure 58: An example of the baseline flatness evaluation.

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 59b) shows this ratio for each grid cell, i.e., the grid cell is marked as:

- “Validated” if $N_{\text{member}}/N_{\text{spectra}} > 0.5$ (Blue squares in Figure 59b)
- “Marginally validated” if $N_{\text{member}}/N_{\text{spectra}} > 0.3$ (Cyan squares)
- “Questionable” if $N_{\text{member}}/N_{\text{spectra}} > 0.2$ (Yellow squares)

After the validation step, the grid containing the $N_{\text{member}}/N_{\text{spectra}}$ rate per grid cell is smoothed by a Gaussian-

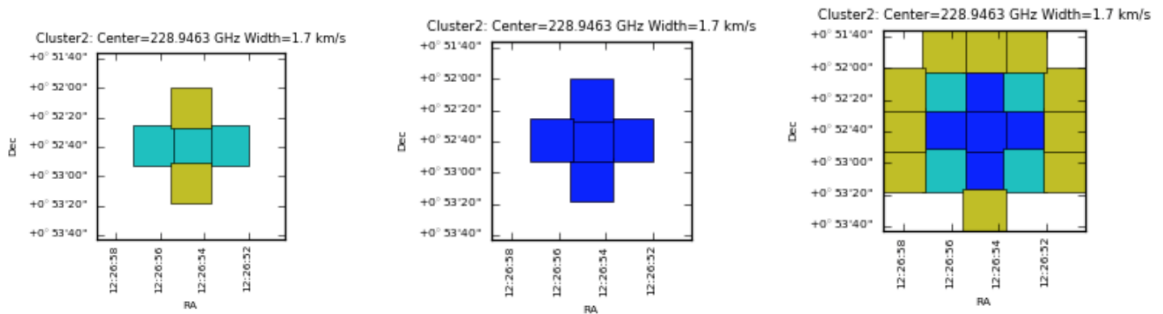


Figure 59: Examples of (a) clustering_detection, (b) clustering_validation, and (c) clustering_smoothing

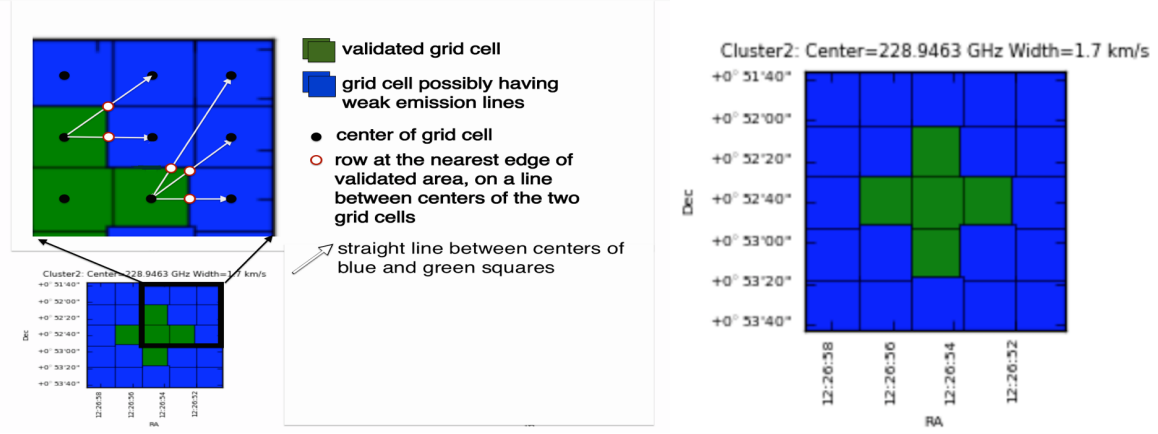


Figure 60: (a) An example of how the mask range is calculated. In the blue squares, the mask channel range is the range obtained at the nearest edge of any validated area by interpolating mask channel ranges in the valid grid cells (white-filled red circle). (b) An example of clustering_final.

like grid function. This is to eliminate the isolated grid cells having a single emission line candidate while enhancing the grid cells with detected emission line in neighboring grid cells.

Figure 59c 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 59c or green squares in Figure 60), 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 60a. An example of “clustering_final” is shown in Figure 60b.

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 taking into account the flagging fraction, are listed in the table.

10.11.2 Profile Map

Figure 61 shows the visible appearance in the summary page. Three types of profile maps are available in the WebLog: 1) a simplified profile map of the combined image per spw in the front page, 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

Profile Map

M100

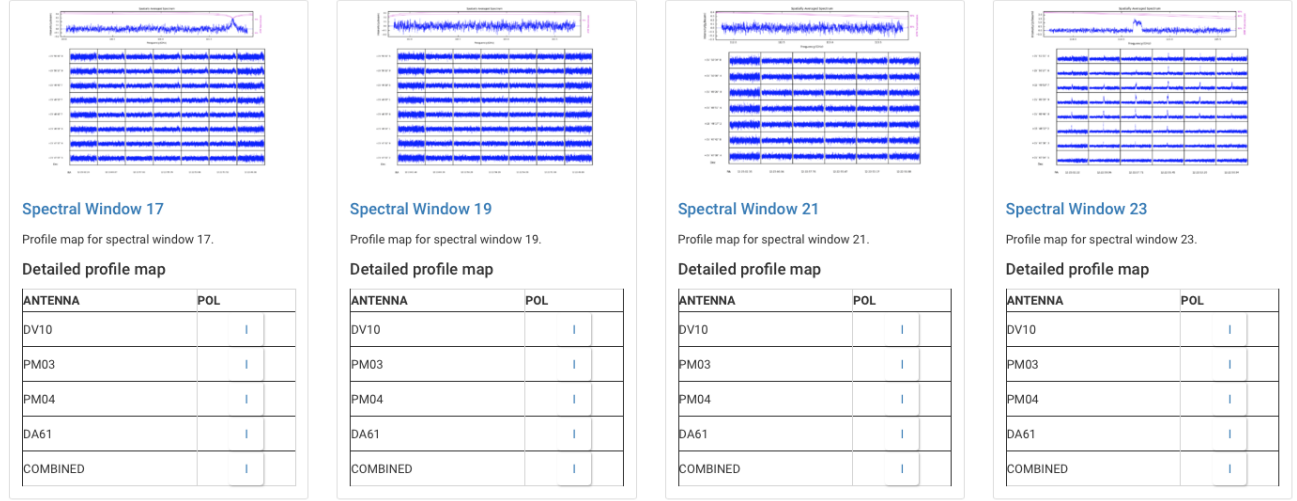


Figure 61: An example of the profile map.

access the simplified profile maps per antenna, click the “Spectral Window”. Note that each spectrum of the simplified profile maps (either combined and per antenna) corresponds to an averaged spectrum in an area of $1/8$ of the image size (imsize), so that the total number of spectra in the profile map is 8×8 as default. If the number of pixels (along the x- or y-axis) is less than eight, it shows all spectrum per pixel. To see the detailed profile maps, go to the “Detailed profile map” table, and 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 5×5 plots per page), the rest of the plots are displayed on other pages.

10.11.3 Channel Map

The number of channel maps per spw (you can see the maps by clicking “Spectral Window n”) corresponds to the number of emission lines that have been identified by the clustering analysis of the pipeline. In each channel map (see Figure 62), the top-middle plot shows the identified emission line in the channel map and the determined line width (bracketed by two red vertical lines; for the case of narrow line, it is difficult to distinguish these two), overplotted on the averaged flux spectrum (in Jy/beam) as a function of frequency (in GHz).

The top-left plot shows the zoom-up view of the identified emission line, but with the velocity axis. The vertical axis is the averaged flux in Jy/beam, 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 by default, which are shown as red vertical lines.

The top-right plot shows the total integrated intensity map (in Jy/beam km/s) over 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.

10.11.4 Baseline RMS Map

This map is created using the baseline RMS stored in the baseline tables. The baseline RMS is calculated by [hsd_baseline](#) using emission-free channels. This map is available only for the combined data and not for per

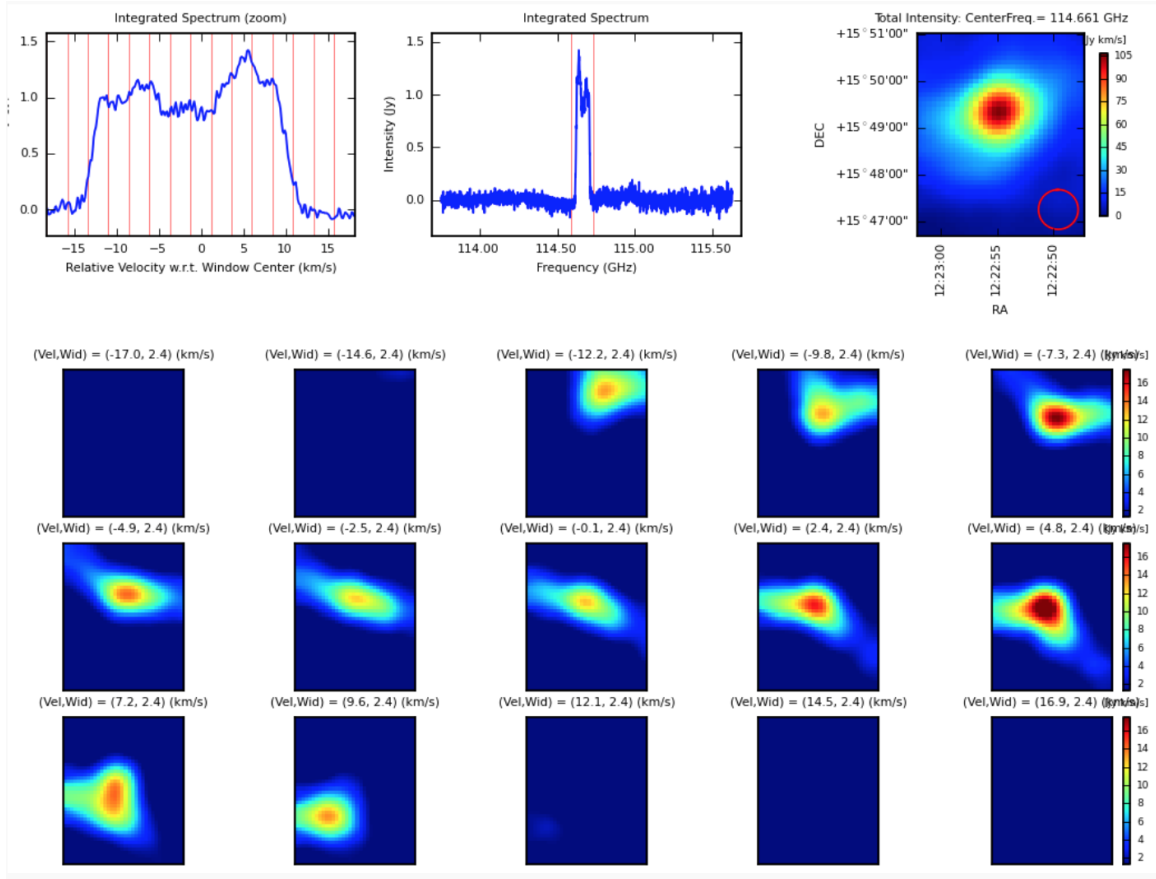


Figure 62: An example of a channel map.

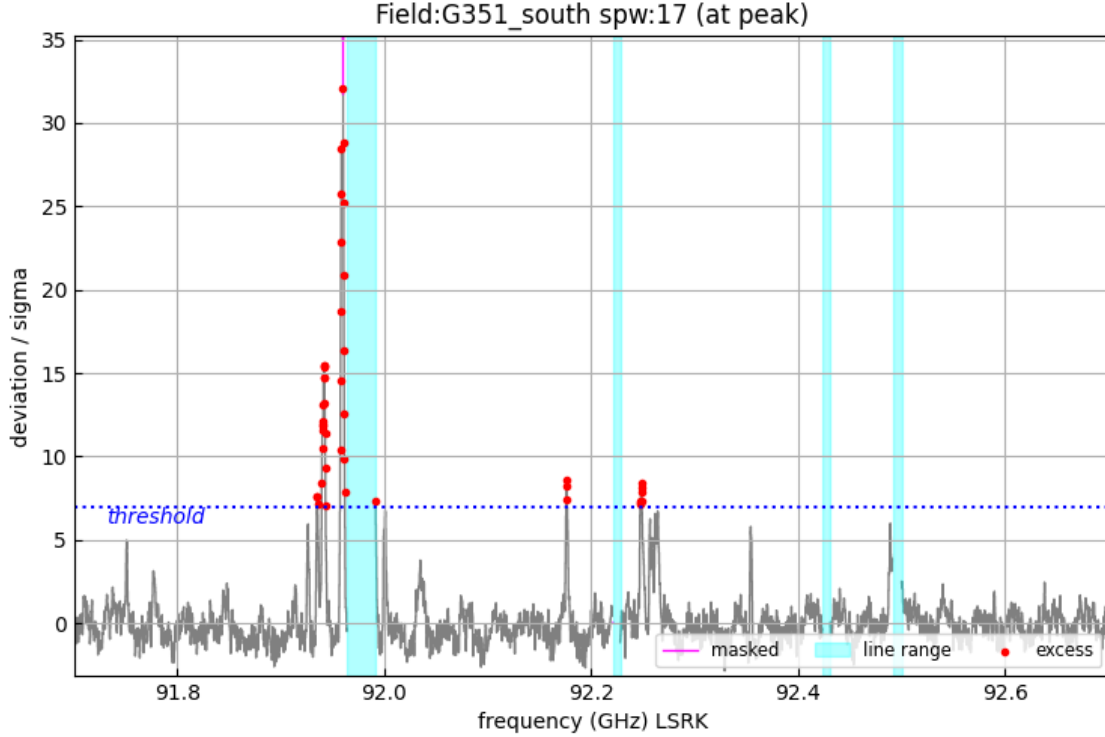


Figure 63: An example of the diagnostic plot for possible missed line channels.

antenna data.

10.11.5 Moment Map

For each spw, three moment maps are generated using `immoments` task: (1) maximum intensity map (moment-8) by using all the available channels, (2) total intensity map (moment-0) by using the line-free channels, and (3) maximum intensity map by using the line-free channels. These maps can be used to diagnose the image quality as well as severe contamination of noise.

10.11.6 Diagnostic plots for possible missed line channels

These plots are generated when line emission is detected outside the line ranges determined at the `hsd_baseline` stage. The default signal-to-noise ratio threshold for the detection of these lines is 7. Figure 63 shows an example of the diagnostic plot for possible missed line emission. This heuristic works per spw and per source at the imaging level, that is, on combined-MS data.

10.11.7 Contamination Plot

This plot contains three plots of Peak S/N map, mask map, and masked-averaged spectrum. The open circle in the Peak S/N map indicates the peak S/N position. Mask map indicates the pixels that have S/N ratio lower than 10%. In the 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 S/N position. If the negative peak of the masked-averaged spectrum is less than $-4 \times$ 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.

10.13 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	$0 < \text{score} < 1 \implies 60\% > \text{fraction flagged} > 5\%$ (for 'online', 'shadow', 'qa0', 'qa2', 'before', and 'template flagging agent') where $0 < \text{score} < 1 \implies \text{HIGH\%} > \text{fraction flagged} > \text{LOW\%}$ means <ul style="list-style-type: none"> Score is 0 if flag fraction $\geq \text{HIGH\%}$ Score is 1 if flag fraction $\leq \text{LOW\%}$ Score is linearly interpolated between 0 and 1 for fractions between HIGH% and LOW%
	Pointing outliers	If no pointing outliers are detected, the QA score is 1.0. If pointing outliers are detected, the QA score is set at 0.83.
hsd_skycal	Check elevation difference between ON and OFF	1.0 if elevation difference between ON and OFF is $\leq 3^\circ$. 0.8 if the elevation difference between ON and OFF is $> 3^\circ$.
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.
<i>table continued on next page</i>		

table continued from previous page

	XX-YY large deviation outlier	<p>QA score is calculated by the following criterion with corresponding warning messages.</p> <ul style="list-style-type: none"> • 1.0 if no significant XX-YY polarization difference is detected • 0.95–0.65 if XX-YY deviation is detected • below 0.65 if XX-YY large deviation outlier is detected
hsd_atmcor	Check that ATM correction is applied	<p>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.</p>
hsd_baseline	Spectral line detection	<p>QA score is calculated by the following criteria with corresponding warning messages. Note that 1/3 of the spw is our criteria to call whether the line is wide or narrow. Edge-line is the one detected at the very edge part of spw. In the following, “main line” does not necessarily be the edge-line (in many cases it will be detected around the center of spw).</p> <ul style="list-style-type: none"> • 1.0 if edge-line is not detected and the main line is narrow • 0.55 if edge-line is detected and the main line is narrow • 0.60 if edge-line is not detected and the main line is wide • 0.55 if edge-line is detected and the main line is wide • 0.80 if no spectral lines are detected • 0.88 if deviation mask is overlapped with spectral lines or atmospheric lines
	Evaluate the baseline flatness	<p>QA score is calculated by the following criterion with corresponding warning messages.</p> <ul style="list-style-type: none"> • 0.33 if $\text{MAX}(\text{mean}) - \text{MIN}(\text{mean}) > 3.6\sigma$ • 0.33–1.0 if $1.8\sigma < \text{MAX}(\text{mean}) - \text{MIN}(\text{mean}) \leq 3.6\sigma$ • 1.0 if $\text{MAX}(\text{mean}) - \text{MIN}(\text{mean}) < 1.8\sigma$ <p>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.</p>
	Deviation from the zero-baseline	<p>If the pipeline detects any significant deviation from the zero-baseline at frequency ranges outside of the candidate lines, it triggers “deviation masks” and sets the QA score as 0.65.</p>
hsd_bflag	Percentage of incremental flagging per source per spw	<p>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%.</p>
hsd_imaging	Determine if observed area (pixel) is not masked	<p>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%</p>

table continued on next page

table continued from previous page

	Astronomical contamination	If any possible astronomical line contamination is detected, QA score is set to 0.65. Users are encouraged to check the contamination plots.
	Missed line channels	If significant off-line-range emission is detected, the QA score is 0.60. If not, the QA score is 1.0.
hsd_exportdata	Check that Pipeline products have been exported	1.0 when files successfully exported otherwise it is 0.0.

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 Cycle 7 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.

Pipeline version	weighting default	perchanweightdensity default
CASA < 5.6	natural	effectively False
C6 pipeline	briggs	effectively False
CASA ≥ 5.6	natural	True
C7 Pipeline	briggs	False
PL2020		
≥ 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).