CASA News
Issue 11 • 27 January 2022

CASA 6.4

The Common Astronomy Software Applications (CASA) Team is pleased to announce that CASA 6.4 has been released. CASA 6.4 is available either as downloadable tar-file, or through pip-wheel installation, which gives users the flexibility to integrate CASA into their own Python (v.3.6 or v.3.8) environment. CASA 6.4 is expected to work on a variety of Operating Systems, as explained elsewhere in this newsletter (/enews/casa_011/index.shtml#os_support) .

The past year included various highlights in terms of CASA development from CASA 6.2 – 6.4. A new task, **phaseshift (/enews/casa_011/index.shtml#phaseshift) ,** was introduced, which replaces the fixvis task for shifting the phase center of visibility data. And a new task, **sdatmcor (/enews/casa_011/index.shtml#sdatmcor)** , allows for atmospheric corrections of single dish data. In addition, a significant refactor of tclean (https://casadocs.readthedocs.io/en/stable/notebooks/memo-series.html#Cube-Refactor) was completed to increase the reliability, flexibility, and performance of imaging data cubes.

Other CASA development highlights include:

- **tclean**: new option *'briggsbwtaper'* and improved *'briggs'* weighting; improved algorithm for fitting the PSF (CASA 6.2); improvements *'savemodel'* step (CASA 6.3+)
- **sdbaseline**: new parameters *'updateweight'* and *'sigmavalue'* (CASA 6.2+)
- **accor**: new parameter support, *'corrdepflags'* (CASA 6.2+)
- **tsdimaging:** *'timerange'* parameter addition (CASA 6.3+)
- **plotcal/plotms**: plotcal deprecation, functionality migrated to plotms (CASA 6.4)
- **plotms**: improvements on averaging, channel selection, and Mueller/Jones tables (CASA 6.2); support of averaging calibration tables with channel selection (CASA 6.4).
- **smoothcal**: smoothing of calibration tables produced by task accor (CASA 6.3+)
- **fringefit**: larger dataset processing due to memory usage reduction (CASA 6.4)
- Consistency in error handling among tasks (CASA 6.2+).

For more details on these and other new features, see CASA Docs (https://casadocs.readthedocs.io/) .

CASA 6.4 can be downloaded here (https://casa.nrao.edu/casa_obtaining.shtml) .

We welcome feedback from users (mailto:casa-feedback@nrao.edu) on this new version of CASA.

CASA Next Generation Infrastructure

Ryan Raba and Jan-Willem Steeb

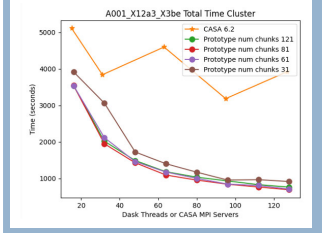


Fig. 1: CASA Next Generation Infrastructure benchmark, comparing the runtime of multi-node parallel dirty image creation using a mosaic gridder in both the current CASA 6 and the CNGI infrastructure. The CNGI performance improvement over current CASA is substantial. CASA 6.2 performance stagnates after a single node because the CASACORE table system cannot write the output images to disk in parallel.

The CASA Team has begun exploring options for a new generation of software to meet the growing demands of current and future radio telescopes, including the next-generation Very Large Array (ngVLA). This CASA Next Generation Infrastructure (CNGI) prototype package is a demonstration of the current state of our research efforts. Its primary purpose is to showcase new data structures (https://cngi-prototype.readthedocs.io/en/stable/data_structures.html) for MeasurementSet and Image contents built entirely in Python atop the popular technology stack of numpy, dask, and xarray.

The CNGI prototype documentation contains Visibility (https://cngi-prototype.readthedocs.io/en/stable/visibilities.html) and Image (https://cngi-prototype.readthedocs.io/en/stable/images.html) overview sections that describe a selection of core mathematics, manipulation, middleware, and analysis functions sections to demonstrate the simplicity and scalability of the technology choices. Notional examples of Calibration (https://cngi-prototype.readthedocs.io/en/stable//calibration.html) , Flagging (https://cngi-prototype.readthedocs.io/en/stable/flagging.html) , and Imaging (https://cngi-prototype.readthedocs.io/en/stable/imaging.html) are provided to illustrate future design and implementation direction. A detailed explanation of technology choices, including the xarray and dask frameworks, the zarr storage format, and the functional design architecture can be found in the Development (https://cngi-prototype.readthedocs.io/en/stable/development.html) section.

Finally, the most computationally intensive areas of CASA imaging are implemented and benchmarked (https://cngi-prototype.readthedocs.io/en/stable/benchmarking.html) to demonstrate the parallel scalability and raw performance now possible from a pure-Python software stack. The figure depicts an example of the creation of a dirty image made through parallel processing on a multi-node computing system, using a mosaic gridder in both the current CASA 6 and the CNGI infrastructure.

Going forward, we will begin to introduce some elements of this technology stack to CASA 6 as opportunities arise in current and future development work. Meanwhile, ngVLA software requirements for development and architecture are underway, with this prototype CNGI demonstration package providing valuable data points to that effort.

We invite anyone interested to explore the new cngi_prototype package. We welcome your feedback (mailto:casa-feedback@nrao.edu) .

New Task Phaseshift

David Mehringer

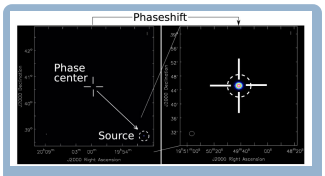


Fig. 1: Task phaseshift changes the phase center of a MeasurementSet

A new task named **phaseshift** was released in CASA 6.3. This task changes the phase center of an MS by modifying the *UVW* coordinates and the specified data column(s). In most cases, it should be used instead of the older **fixvis** task for this purpose. The new task leaves the input MS unaltered and writes a new MS with the updated columns. As with most other MS oriented tasks, **phaseshift** supports many MS selection parameters; the MS it writes will contain only the subset of

specified data. The standard syntax for specifying astronomical world direction coordinates is supported (e.g., 'J2000 19h45m20.56 -50d30m45.7' or 'J2000 19:45:20.56 -50.30.45.7').

There is a Jupyter notebook in Google Colab entitled *Numerical Accuracy of Task Phaseshift* (https://casadocs.readthedocs.io/en/stable/examples/community/phaseshift.html) which details the numerical characterization of **phaseshift**. This notebook provides a complete script and detailed results regarding the correctness of the results produced by **phaseshift**. In summary, in a 1.0 GHz VLA simulation in which the phase center and a source were initially separated by 2.7 degrees, using **phaseshift** to shift the phase center to be coincident with the source resulted in the source being located less than 30 milli-arcsec (0.003 pixels) from the simulated position. In a 150 GHz ALMA simulation, in which the phase center and a source were initially separated by 1.2 arcmin, using **phaseshift** to shift the phase center to be coincident with the source resulted in the source being located less than 90 μarcsec (0.001 pixels) from the simulated position. The algorithms used by **phaseshift** and **tclean**–by specifying the *phasecenter* parameter–are similar, and so are the results, although **phaseshift** seems to have slightly, but statistically significantly, more accuracy. The algorithms used by these tasks are significantly different from the one used by **plotms**, and therefore **plotms** results are likely to differ substantially, the larger differences being associated with larger shifts.

Notable functionality that is currently not supported includes:

1. Time-dependent coordinate frames and ephemeris objects (support planned for ngCASA only and not CASA 6)
2. Specifying the new phase center in terms of an offset from the original phase center will be implemented, if explicitly requested
3. There is currently no support for the possible use case of updating only the *UVW* values (e.g., based on antenna position updates), but not the associated data values. The deprecated task **fixvis** supports this functionality, so it may be used for this purpose. If such support will be needed after **fixvis** is removed, it can be added to **phaseshift** if requested.

New Single-dish Task *sdatmcor*

Takeshi Nakazato

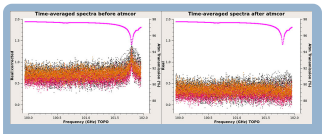


Fig. 1: Illustration of the atmospheric correction by *sdatmcor* task. Left: Calibrated single-dish spectra before the correction. Right: Same spectra after the correction. Magenta line in each panel shows atmospheric transmission curve. There is a strong absorption feature at around 101.8GHz and it causes atmospheric residual feature in the calibrated spectra (left panel). Such feature disappears after the correction (right panel).

Single-dish (auto-correlation) spectrum suffers from strong emission and/or absorption by Earth's atmosphere. The traditional way to extract and calibrate astronomical emission from the detected signal is to observe a "void" or "off-source" region of the sky, free from any astronomical emission, and use it as a reference signal that represents atmospheric emission when target object is observed. However, since "on-source" and "off-source" data are taken in slightly different time and direction, observed reference data deviates from the ideal. Due to this deviation, calibrated spectrum sometimes contains a residual atmospheric feature. Residual becomes prominent when the deviation is large. In particular, when an absorption feature in an atmospheric transmission curve exists inside the spectral window, such feature manifests as a strong and broad absorption or emission in the calibrated spectrum, preventing astronomers from precise analysis of astronomical emission.

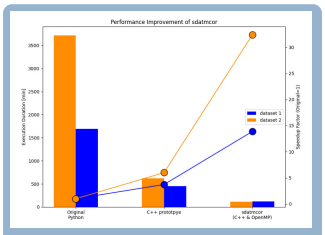


Fig. 2: Performance improvement of C++ re-implementation and OpenMP parallelization. Bar plots show wall clock time to complete the correction (left axis) while circles are the speed increase with respect to the original Python prototype (right axis). Results for two datasets are shown with blue and orange. Speed up factor of C++ re-implementation with optimization of atmosphere tool is ~ 5 while the final speed up factor after OpenMP parallelization is 15-30. In terms of wall-clock time, processing time reduced from 15-30 minutes to a few minutes.

Recently, Sawada et al. (2021) proposed a way to compensate atmospheric residual as part of offline data processing. The correction is based on the atmosphere model called Atmospheric Transmission at Microwaves (ATM) implemented as atmosphere (at) tool in CASA. We have implemented a new task *sdatmcor* for this atmospheric offline correction.

Figure 1 shows the calibrated spectra before and after *sdatmcor*. In the spectra before *sdatmcor* (left panel), there is an emission feature exactly at the frequency where the absorption feature exists (magenta line). This is an artifact created by the atmosphere due to the imperfect calibration. On the other hand, such feature disappears after the correction by *sdatmcor* (right panel) so that resulting spectra become flat.

Although *sdatmcor* yielded great improvement of the data quality, there was a serious performance issue in the original prototype Python script. That is because the correction requires intensive calculation of atmospheric opacity. Therefore, we re-implemented it in C++ with parallel processing using OpenMP. Figure 2 illustrates the performance improvement of the *sdatmcor* task compared with its prototype Python script. After re-implementation in C++ with performance optimization of atmosphere tool, the task became about 5 times faster than the original script. Further improvement with OpenMP parallelization resulted in the final speed up factor of 15~30. By this improvement, *sdatmcor* now provides its capability with practical processing time. Any CASA users can use this functionality. Please see task documentation (https://casadocs.readthedocs.io/en/stable/api/tt/casatasks.single.sdatmcor.html) on how to use *sdatmcor*. Also, ALMA Single-Dish Pipeline incorporated new processing step based on *sdatmcor*, called *hsd_atmcor*, into its starndard processing recipe since ALMA Cycle 8.

Reference

- Documentation of sdatmcor (casadocs) (https://casadocs.readthedocs.io/en/stable/api/tt/casatasks.single.sdatmcor.html)
- Sawada, T. et al., 2021, PASP, 133, 034504 (https://doi.org/10.1088/1538-3873/abe0ab)

CASA Docs

Bjorn Emonts and Ryan Raba

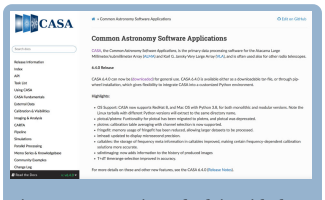


Fig. 1: CASA Docs' new look in Github.

CASA Docs (https://casadocs.readthedocs.io/) is the official CASA user documentation and interface specification. As of CASA 6.2, the CASA Docs content has migrated to Github and is now hosted by Readthedocs (Fig. 1). This new system provides a modern look, faster page loads, and a more natural layout for users familiar with software documentation compared to the old Plone-based system, which was hosted internally at NRAO. It also allows the CASA team to more easily write and release documentation versions. A new version of CASA Docs is being released with every major and minor CASA release.

CASA Docs 6.4

Apart from the wealth of content that was already available in CASA Docs, the latest version of CASA Docs 6.4

introduced updated content on several important aspects.

- Updated API (https://casadocs.readthedocs.io/en/stable/api.html) (Application Programming Interface) content. The API section in CASA Docs includes detailed documentation on CASA tools and tasks, as well as information on configuration and the 'casadata' repository. This is the **official interface specification** of CASA defining all functions intended for public use.
- Version-specific installation instructions (https://casadocs.readthedocs.io/en/stable/notebooks/introduction.html#id1) , as well as compatibility with Operating Systems (/enews/11.11/index.shtml#os_support) .
- Community Examples (https://casadocs.readthedocs.io/en/stable/examples/index.html) which showcase tutorials on using modular CASA, e.g., on how to run advanced simulations. This growing number of Community Examples complements the existing CASA Memo Series and Knowledgebase in CASA Docs.
- Change Log (https://casadocs.readthedocs.io/en/stable/changelog.html) information, which gives a complete overview of all pull requests and API changes. In addition to the standard Release Notes, this detailed Change Log is useful for users who want to compare small changes in the code from version to version, for example for updating scripts or pipelines.

Reliability

Another major change in terms of CASA documentation has been less visible to users. To improve the reliability of the CASA code and documentation, over the past few years the CASA team has adopted a strategy of strictly developing and testing code against the CASA Docs documentation. This included the assembly of a dedicated test-group within the CASA team, led by Sandra Castro (ESO), who have been writing a large suite of new verification tests based on CASA Docs. This is an ongoing effort with the long-term goal of making CASA, and CASA documentation, as reliable as possible for users.

CARTA v2/v3 update

Juergen Ott, John Hibbard, & Ryan Raba

The visualization tool CARTA (https://cartavis.org) (Cube Analysis and Rendering Tool for Astronomy) is still undergoing rapid development. A few months ago, CARTA v2 was released. Its new capabilities include multi-profile plots for spectra, enhanced colorbars, subimage saving options, and Gaussian profile fitting, to mention only some highlights. Catalogs can be overlaid, and catalog entries can be selected through a histogram or other, customizable graphs to be highlighted in the image. An example is in the figure.

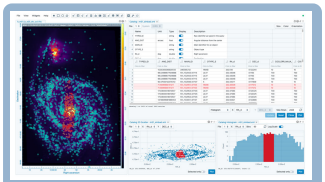


Fig. 1: The CARTA catalog widget. Catalogs can be overlaid on images, and objects are highlighted based on selections in the table, image, histogram, or other customizable plots.

Right now, v3 development is well underway. This version will include substantial new features such as multi-panel display, scripting, and position-velocity generation. A beta version has now been released and is available on the CARTA homepage (https://cartavis.org/) .

CARTA is based on a backend server that is controlled by a frontend that runs in a browser window. This design makes it very efficient to also render images on remote systems. Please see the CASA Docs for updated instructions (https://casadocs.readthedocs.io/en/stable/notebooks/carta.html) on how to run CARTA on the NRAO cluster from a remote computer.

We remind CASA users that the *CASA Viewer* is no longer supported. No more maintenance or improvement to the code will be done, and all effort will be spent on improving CARTA.

For any questions on CARTA, please contact the CARTA team through support@carta.freshdesk.com.

OS Support

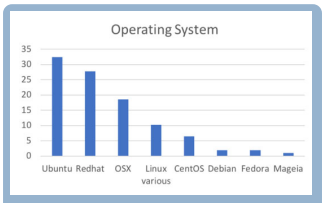
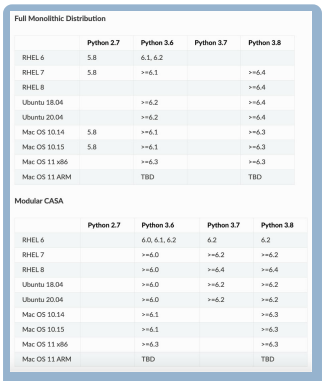


Fig. 1: CASA User Survey, showing the percentage of CASA users plotted against the primary operating system that they use for running CASA. See CASA Memo #6 for details.



Full Monolithic Distribution

	Python 2.7	Python 3.6	Python 3.7	Python 3.8
RHEL 6	5.8	6.1, 6.2		
RHEL 7	5.8	>=6.1		>=6.4
RHEL 8				>=6.4
Ubuntu 18.04		>=6.2		>=6.4
Ubuntu 20.04		>=6.2		>=6.4
Mac OS 10.14	5.8	>=6.1		>=6.3
Mac OS 10.15	5.8	>=6.1		>=6.3
Mac OS 11 x86		>=6.3		>=6.3
Mac OS 11 ARM		TBD		TBD

Modular CASA

	Python 2.7	Python 3.6	Python 3.7	Python 3.8
RHEL 6		6.0, 6.1, 6.2	6.2	6.2
RHEL 7		>=6.0	>=6.2	>=6.2
RHEL 8		>=6.0	>=6.4	>=6.4
Ubuntu 18.04		>=6.0	>=6.2	>=6.2
Ubuntu 20.04		>=6.0	>=6.2	>=6.2
Mac OS 10.14		>=6.1		>=6.3
Mac OS 10.15		>=6.1		>=6.3
Mac OS 11 x86		>=6.3		>=6.3
Mac OS 11 ARM		TBD		TBD

Fig. 2: Compatibility matrix showing the Operating Systems that are expected to work with various versions of CASA and Python. An up-to-date version of this matrix is being maintained on CASA Docs.

CASA support of operating systems has traditionally been limited to the latest two versions of RedHat Linux and Mac OS. This aligns the development and testing of CASA code with the data-processing environments in ALMA and the VLA operations along with a large percentage of the external user community.

Mac OS

The frequency of Mac releases sometimes causes unintentional delays in support of the latest version of Mac OS. The CASA team tries to keep those delays to a minimum. For Mac OS 11 Big Sur, CASA is currently limiting its official Mac support to machines with Intel x86 chips. Users with ARM-based M1 processors may use the x86 Mac OS 11 distribution of CASA through the Apple Rosetta virtualization software (https://support.apple.com/en-us/HT211861) . Initial testing suggests similar performance and output of CASA in both cases. The CASA team is still looking into the possibility for officially supporting Mac M1 machines through native ARM builds, and we welcome feedback from users regarding ARM compatibility at casa-feedback@nrao.edu

Ubuntu

A CASA User Survey (https://drive.google.com/file/d/18OHYERd11S2zZ8Hkx1h6j8e8iqsStkJs/view) in 2018 revealed that the most popular operating system among general users is Ubuntu (Fig. 1). Even though Ubuntu has not been officially supported, user experience is that CASA has in general been working well with Ubuntu. For CASA 6, the CASA team has moved into a direction where CASA versions become gradually more platform agnostic.

While all CASA code testing continues to be done on RedHat Linux, we now expect that the latest CASA releases work as intended on certain versions of Ubuntu. A matrix of all combinations of CASA, OS, and Python versions that are expected to work is shown in Fig. 2. This matrix is kept up to date on CASA Docs (https://casadocs.readthedocs.io/en/stable/notebooks/introduction.html#Compatibility) .

The CASA team now accepts bug reports for compatibility issues with certain versions of Ubuntu and other OSs, as listed in Fig. 2. Please send those bug reports directly to the CASA team at casa-feedback@nrao.edu (mailto:casa-feedback@nrao.edu) .