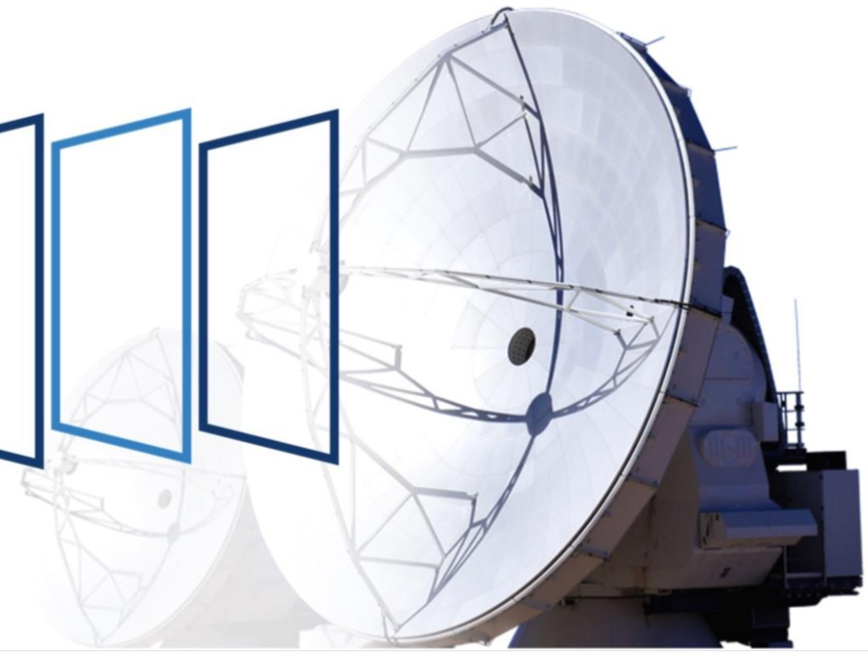
CASA

Common Astronomy Software Applications



The CASA Software for Radio Astronomy

Overview of Framework, Algorithms, and New VLBI Capabilities

Bjorn Emonts

NRAO (Charlottesville, VA)
CASA User Community Liaison

Ilse van Bemmel

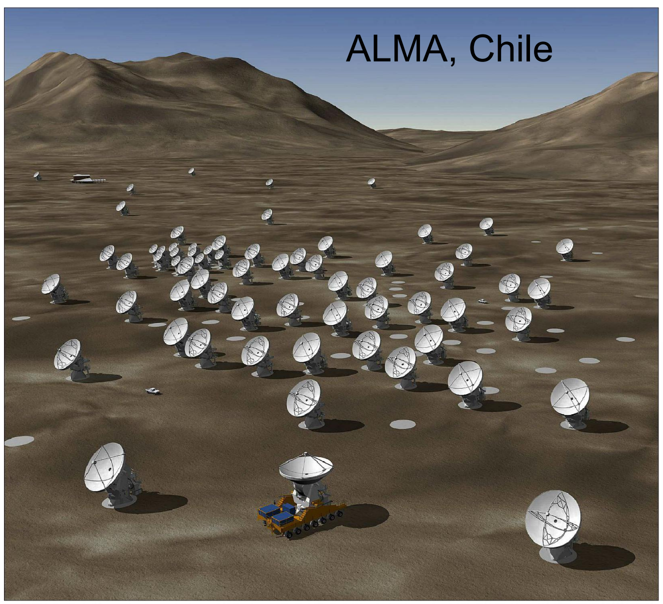
JIVE (Dwingeloo, NL)
CASA-VLBI Project Scientist

Contact: casa-feedback@nrao.edu

CASA

Common Astronomy Software Applications for Radio Astronomy

Primary data processing software *Karl G. Jansky Very Large Array (VLA)* and *Atacama Large Millimeter/submillimeter Telescope (ALMA),* but frequently used also for other radio telescopes.



NRAO ESO NAOJ ASIAA ALMA CSIRO ASTRON JIVE Joint Institute for VLBI ERIC

CASA Team

Urvashi Rao Venkata (NRAO-SO)	*CASA Lead, Lead scientific development*
Sandra Castro (ESO)	*Lead verification testing*
Darrell Schiebel (NRAO-CV)	*Lead visualization, Infrastructure development*
Takeshi Nakazato (NAOJ)	*Lead Single Dish, Scientific development*
Josh Marvil (NRAO-SO)	*Lead scientific validation*
Bjorn Emonts (NRAO-CV)	*User Community Liaison*
Enrique Garcia (ESO)	*Infrastructure development*
Bob Garwood (NRAO-CV)	*Infrastructure, Verification testing*
Kumar Golap (NRAO-SO)	*Scientific development*
Justo Gonzalez Villalba (ESO)	*Scientific development*
Pam Harris (NRAO-SO)	*Data visualization*
Yohei Hayashi (NAOJ)	*Scientific development, Single Dish*
Josh Hoskins (NRAO-CV)	*Scientific development, Infrastructure*
Wataru Kawasaki (NAOJ)	*Scientific development, Single Dish*
Jorge Lopez (NRAO-CV)	*Infrastructure, Scientific development*
Andrew McNichols (NRAO-CV)	*Infrastructure, Scientific development*
Dave Mehringer (NRAO-CV)	*Scientific development, Verification testing*
Renaud Miel (NAOJ)	*Scientific development, Single Dish*
George Moellenbrock (NRAO-SO)	*Scientific development*
Federico Montesino (ESO)	*Infrastructure, Scientific development*
Dirk Petry (ESO)	*Scientific development*
Neal Schweighart (NRAO-CV)	*Scientific development, Verification testing*
Kazuhiko Shimada (NAOJ)	*Scientific development, Single Dish*
Jan-Willem Steeb (NRAO-CV)	*Scientific, Infrastructure development*
Takeshi Shakunaga (NAOJ)	*Scientific development, Single Dish*
Ville Suoranta (NRAO-CV)	*Infrastructure development*
Tak Tsutsumi (NRAO-SO)	*Scientific development, Verification testing*
Akeem Wells (NRAO-CV)	*Verification testing*
Wei Xiong (NRAO-ALBQ)	*Infrastructure, Scientific development*

JIVE Joint Institute for VLBI ERIC

CASA-VLBI

Ilse van Bemmel (JIVE)	*VLBI, Project Scientist*
Mark Kettenis (JIVE)	*VLBI, development*
Des Small (JIVE)	*VLBI, development*
Arpad Szomoru (JIVE)	*VLBI, management*
Marjolein Verkouter (JIVE)	*VLBI, management*
Aard Keipema (JIVE)	*VLBI, Jupyter kernel*

ARDG (Algorithm Research & Development Group)

Sanjay Bhatnagar (NRAO) - ARDG Lead
Mingyu (Genie) Hsieh (NRAO)
Martin Pokorny (NRAO)
Preshanth Jagannathan (NRAO)
Srikrishna Sekhar (NRAO, IDIA)

CASA Stakeholders

- ALMA Users
- ALMA Pipeline
- VLA Users
- SRDP (Science Ready Data Products, NRAO)
- VLBA/VLBI
- ARDG (Algorithm R&D Group, NRAO
- General Users / CASA Users Committee

CASA Development process

- Prioritization (stakeholders, internal) → ~6 months
- Development

  - Definition requirement / specification —— *Documentation*
  - Development
  - Internal verification: testing → functional, unit, stakeholder
  - External validation (pipeline testing)
  - Documentation (benchmarking) ←

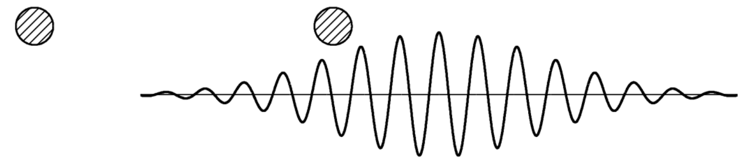
- Release: **General CASA releases → ~2 months**

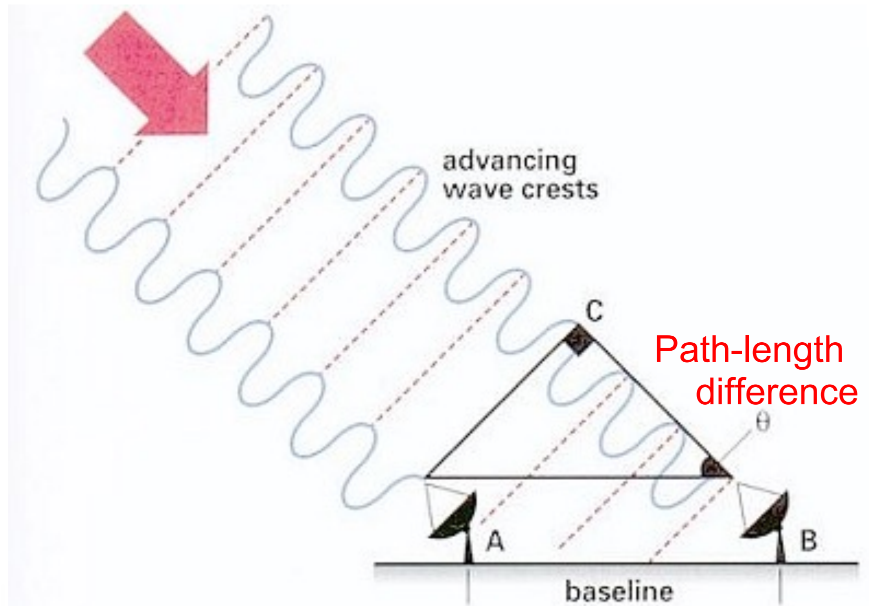
  (CASA+pipeline releases → ~yearly)

CASA Basics

Single Dish & **Interferometry**

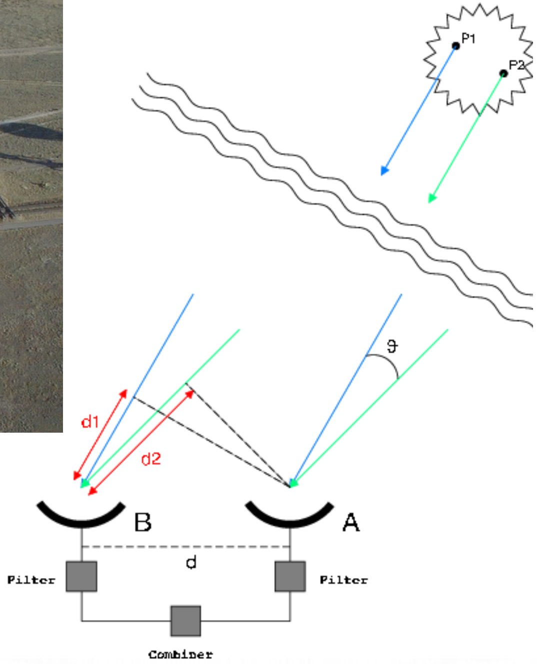
Interferometry

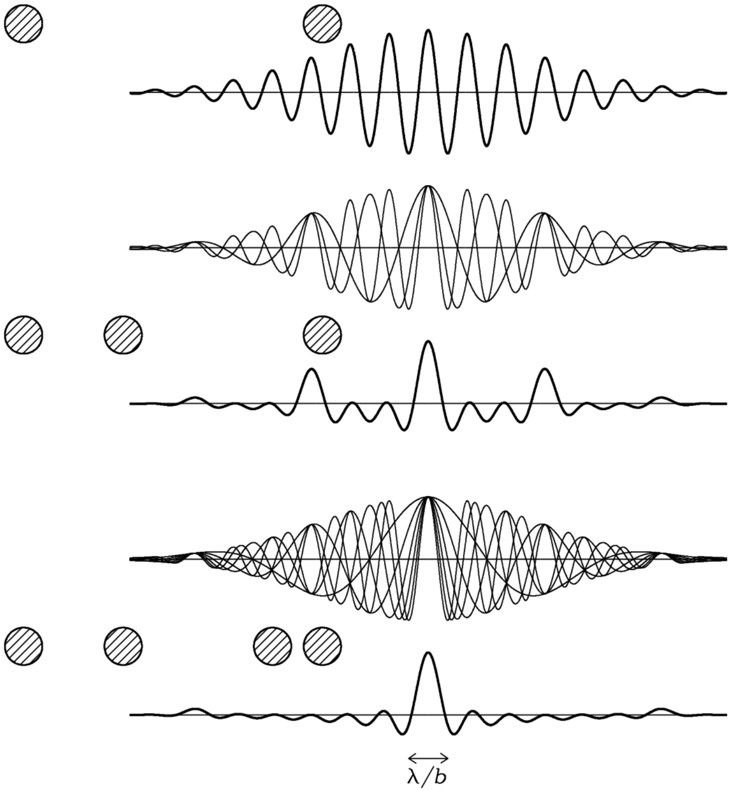




CASA Basics

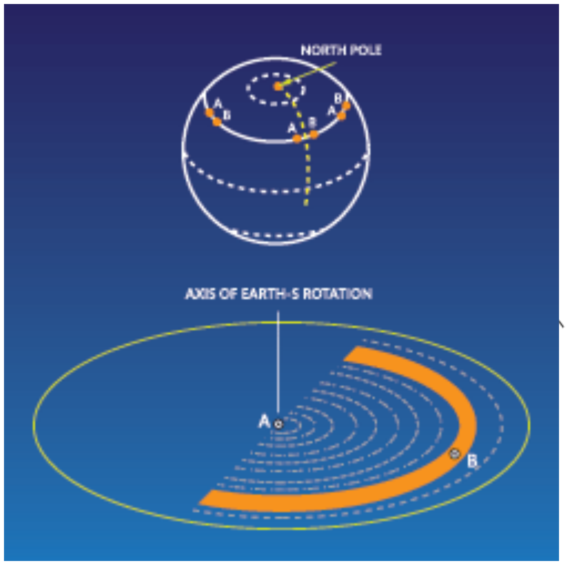
Interferometry

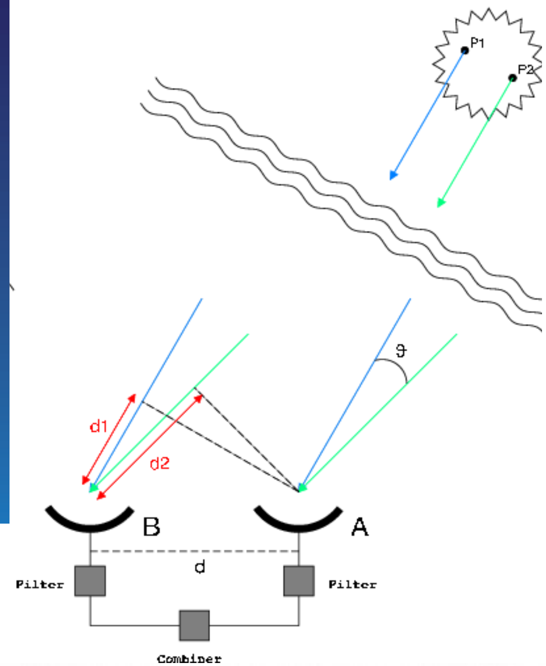


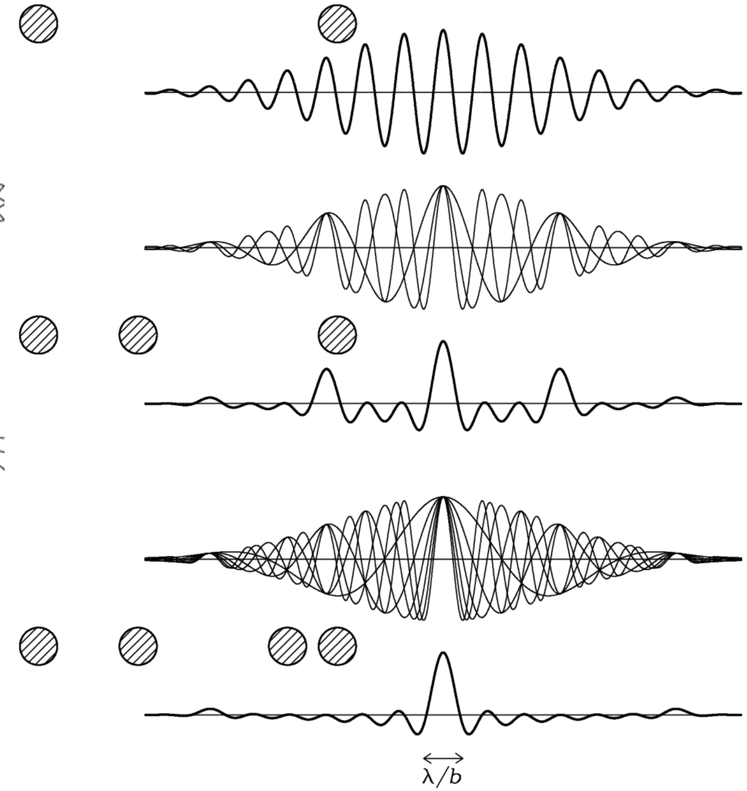


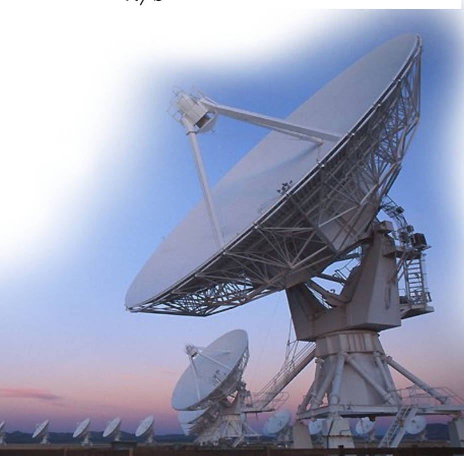
CASA Basics

Interferometry



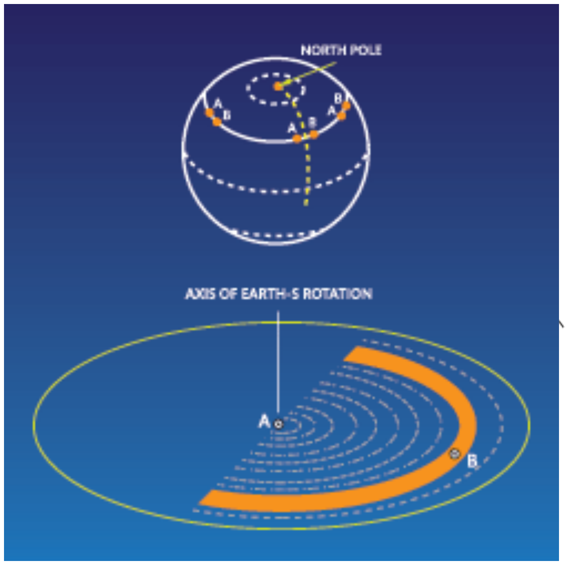


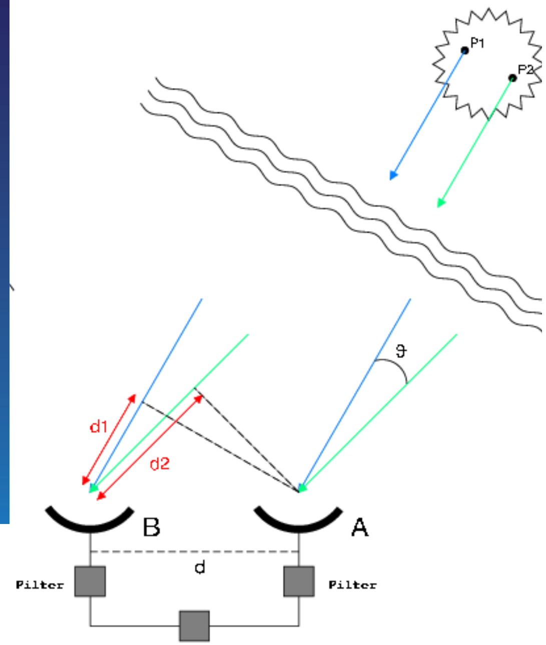


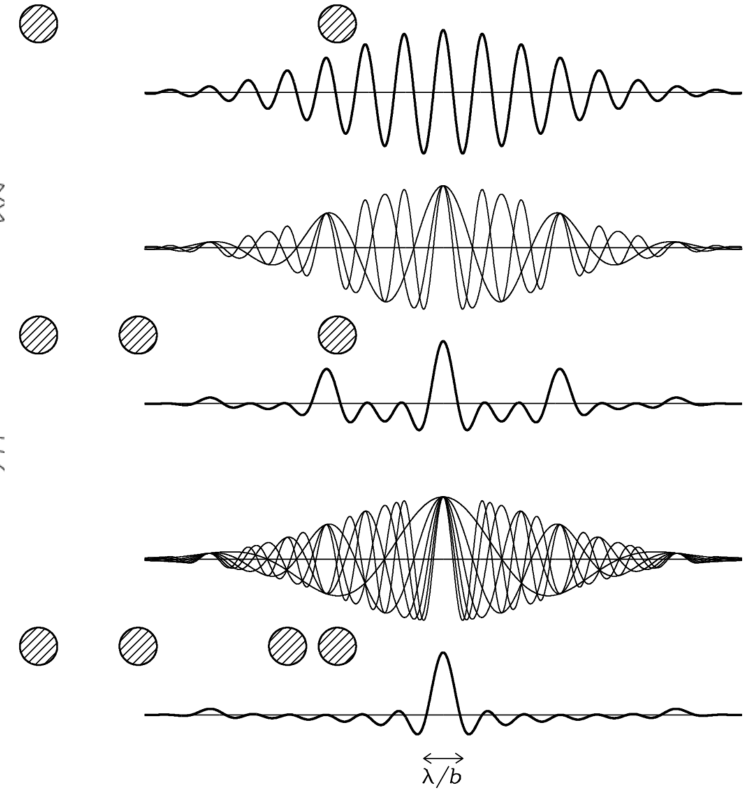


CASA Basics

Interferometry







“Visibility”: interferometer response per antenna pair (*i, j*), Δt, Δf, polarization

→ Fourier Transform of the sky brightness distribution.

→ “complex”, with *amplitude*, *phase* information: $V(u,v) = a_r e^{i\varphi_r}$

→ Source *brightness*, *structure*

CASA Basics

CASA software: raw "visibility" data → science products

- Large data sets!
  Example: N(N-1) baselines, Δt ~ 3s over 12h, 2000 Δf channels, 4 polarizations, ….

- Discrete sampling ('broken mirror') → missing information!
  Iteratively reconstruct model of sky

→ Computationally expensive

→ Complex, advanced algorithms

"Visibility": interferometer response per antenna pair (*i, j*), Δt, Δf, polarization

→ Fourier Transform of the sky brightness distribution.
→ "complex", with *amplitude*, *phase* information: $V(u,v) = a_r e^{i\varphi r}$

→ Source *brightness*, *structure*

The CASA Software

- **Tools**: basic C++ functions linked to Python interface → ***basic operations***
- **Tasks**: bundle tools + Python functionality → ***specific data reduction step***
  → *user friendly, parameter input*
- **GUIs**: Graphical User Interfaces to visualize and examine data/images

Manual, scripting & pipelines *(ALMA calibration & imaging, VLA calibration, VLA Sky Survey)*

The CASA Software

Import/export

Information

Manipulation

Calibration

Imaging

Analysis (Visualization)

Simulations

Calibration in CASA

Import/export

Information

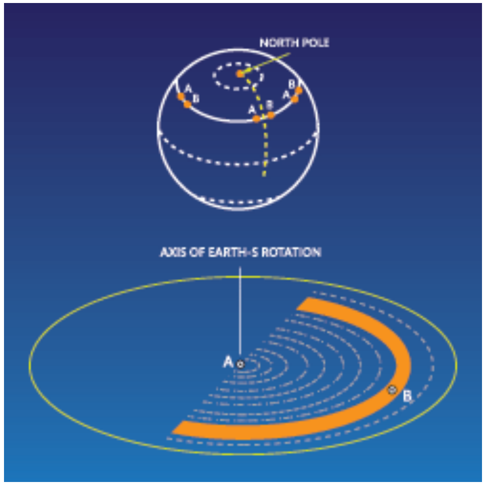
Manipulation

Calibration

Imaging

Analysis

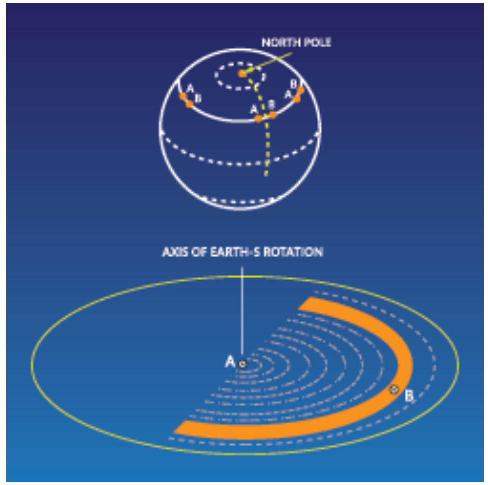
Simulations



$$\vec{V}_{ij} = J_{ij}\,\vec{V}_{ij}^{IDEAL}$$

Calibration in CASA

- Import/export
- Information
- Manipulation
- **Calibration**
- Imaging
- Analysis
- Simulations



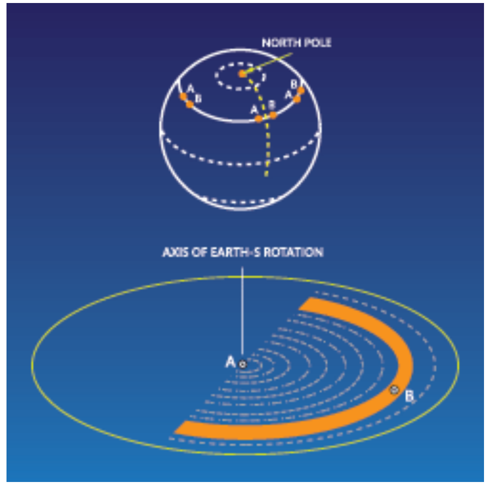
$$\vec{V}_{ij} = J_{ij}\,\vec{V}_{ij}^{IDEAL}$$

$$\vec{V}_{ij} = M_{ij}\,B_{ij}\,G_{ij}\,D_{ij}\,E_{ij}\,P_{ij}\,T_{ij}\,\vec{V}_{ij}^{IDEAL}$$

- T_{ij}: Troposphere
- P_{ij}: Parallactic rotation
- E_{ij}: Optical components tel.
- D_{ij}: Leakage
- G_{ij}: Time dependent gains
- B_{ij}: Frequency dep. gains (bandpass)
- M_{ij}: Baseline-based correlator (non-closing) errors

Calibration in CASA

- Import/export
- Information
- Manipulation
- **Calibration**
- Imaging
- Analysis
- Simulations



$$\vec{V}_{ij} = J_{ij}\,\vec{V}_{ij}^{\,IDEAL}$$

$$\vec{V}_{ij} = M_{ij}\,B_{ij}\,G_{ij}\,D_{ij}\,E_{ij}\,P_{ij}\,T_{ij}\,\vec{V}_{ij}^{\,IDEAL}$$

- T_{ij}: Troposphere
- P_{ij}: Parallactic rotation
- E_{ij}: Optical components tel.
- D_{ij}: Leakage
- G_{ij}: Time dependent gains

```
CASA <31>: gaincal(vis='lowres_data1.ms', spw='1:10~110', refant'4', calmode='p')
```

- ***CASA task*** (gaincal)
- *Input Data* (vis='lowres_data1.ms')
- *Spectral Window selection* (spw='1:10~110')
- *Reference antenna* (refant'4')
- *Phase-only Calibration mode* (calmode='p')

Imaging in CASA

Import/export

Information

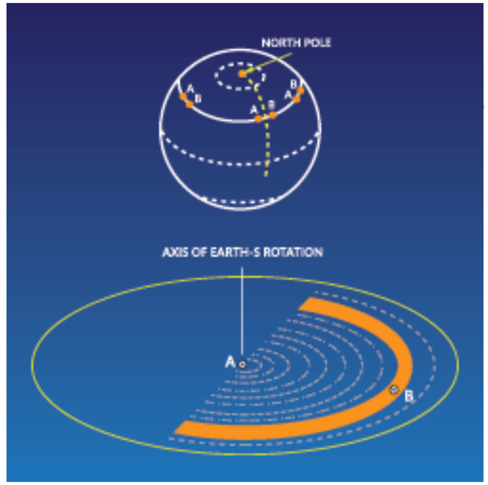
Manipulation

Calibration

Imaging

Analysis

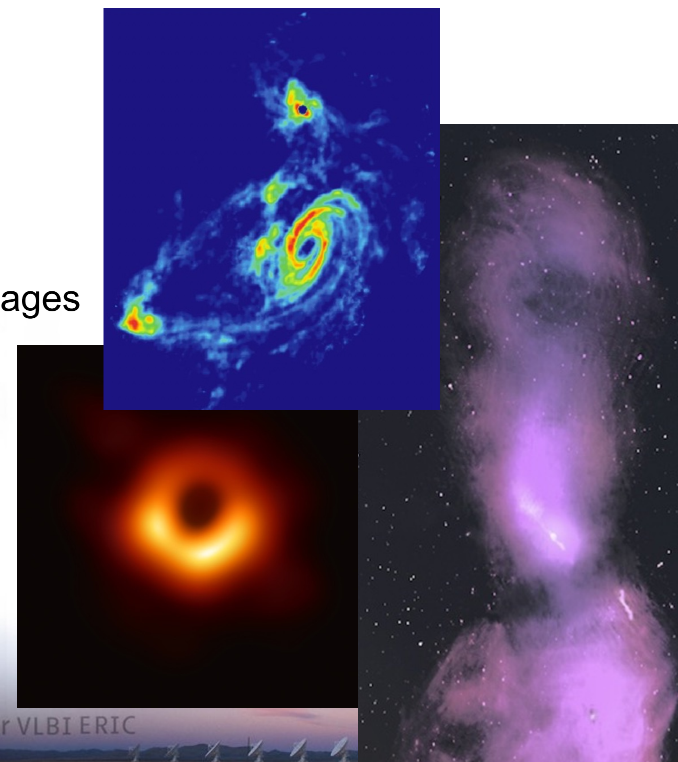
Simulations



$\vec{V}_{ij}^{IDEAL}$

→ Gridding data
→ Weighting data
→ Fourier transform
→ Deconvolution
→ Restoration

2D continuum images
3D image cubes



NRAO ESO 国立天文台 NAOJ ASIAA ALMA CSIRO ASTRON JIVE Joint Institute for VLBI ERIC

Imaging in CASA

Tclean: powerful imaging task

- Fast Fourier Transform (FFT)
- Gridding
- Deconvolution & Restoration
- Primary beam correction
- Data weighting

```
CASA <40>: inp
# tclean -- Radio Interferometric Image Reconstruction
vis                 = ''                # Name of input visibility file(s)
selectdata          = True              # Enable data selection parameters
   field            = ''                # field(s) to select
   spw              = ''                # spw(s)/channels to select
   timerange        = ''                # Range of time to select from dat
   uvrange          = ''                # Select data within uvrange
   antenna          = ''                # Select data based on antenna/bas
   scan             = ''                # Scan number range
   observation      = ''                # Observation ID range
   intent           = ''                # Scan Intent(s)
datacolumn          = 'corrected'       # Data column to image(data,correc
imagename           = ''                # Pre-name of output images
imsize              = [100]             # Number of pixels
cell                = []                # Cell size
phasecenter         = ''                # Phase center of the image
stokes              = 'I'               # Stokes Planes to make
projection          = 'SIN'             # Coordinate projection
startmodel          = ''                # Name of starting model image
specmode            = 'mfs'             # Spectral definition mode (mfs,cu
   reffreq          = ''                # Reference frequency
gridder             = 'standard'        # Gridding options (standard, wpro
   vptable          = ''                # Name of Voltage Pattern table
   pblimit          = 0.2               # PB gain level at which to cut of
deconvolver         = 'hogbom'          # Minor cycle algorithm (hogbom,cl
restoration         = True              # Do restoration steps (or not)
   restoringbeam    = []                # Restoring beam shape to use. Def
   pbcor            = False             # Apply PB correction on the outpu
outlierfile         = ''                # Name of outlier-field image defi
weighting           = 'natural'         # Weighting scheme (natural,unifor
                                        # briggsbwtaper[experimental])
   uvtaper          = []                # uv-taper on outer baselines in u
niter               = 0                 # Maximum number of iterations
usemask             = 'user'            # Type of mask(s) for deconvolutio
   mask             = ''                # Mask (a list of image name(s) or
   pbmask           = 0.0               # primary beam mask
fastnoise           = True              # True: use the faster (old) noise
                                        # calculations
restart             = True              # True : Re-use existing images. F
savemodel           = 'none'            # Options to save model visibiliti
calcres             = True              # Calculate initial residual image
calcpsf             = True              # Calculate PSF
   psfcutoff        = 0.35              # All pixels in the main lobe of t
                                        # beam (the Clean beam).
parallel            = False             # Run major cycles in parallel

CASA <41>:
```

Imaging in CASA: recent development

Tclean: powerful imaging task

- Refactor cube imaging (reliability, flexibility, performance)

- Fast Fourier Transform (FFT)
- Gridding
  - AW-projection
- Deconvolution & Restoration
  - *Adaptive Scale Pixel* deconvolution algorithm
  - *deconvolve* (image plane deconvolution)
  - *sdintimaging* (joint deconvolution SD + Int)
- Primary beam correction
- Data weighting
  - *briggs* (robust) weighting improved
  - *briggsbwtaper* added

CASA download & installation

Website (casa.nrao.edu)

Compatibility Operating Systems

Monolithic (all-inclusive 'plug-and-play')

Pip-wheel (Pythonic, Jupyter Notebooks, Google Colab)

New release every ~2 months!

Latest version: CASA 6.5

The Release Notes and Known Issues of the 6.5 release are a...

CASA 6.5 is based on Python 3, and available either as a downloadable tar-file distribution with Python environment included, or as a modular version that can be installed with pip-wheels.

Manual processing can be done with any CASA version, but ALMA and VLA pipelines may differ and are not always included, so download the correct CASA version for pipeline use.

	Linux (RedHat 6, 7, 8)	**Mac** (OS 11, OSX 10.15)
General Use (Notes)	CASA 6.5.2 (RH7/8 - Py 3.8) CASA 6.5.2 (RH7 - Py 3.6)	CASA 6.5.2 (OS11 - Py 3.8) CASA 6.5.2 (OS11 - Py 3.6)
ALMA Pipeline (Notes)	CASA 6.4.1 (RH7/8)	CASA 6.4.1 (OS11) CASA 6.4.1 (10.15)
VLA Pipeline (Notes)	CASA 6.4.1 (RH7/8)	CASA 6.4.1 (OS11) CASA 6.4.1 (10.15)

The above CASA versions can also be downloaded from our NAOJ CASA mirror site and NAOJ CASA-pipeline mirror site, or via Google Drive.

CASA 6: pip-wheel installation

CASA 6 can optionally be installed through modular pip-wheels, with the flexibility to build CASA tools and tasks into a customized Python environment. Instructions on how to install the pip-wheel version of CASA 6 can be found in CASA Docs: CASA 6 Installation and Usage

The modular pip-wheel version is not yet used in production by ALMA and VLA, and does not include any pipelines.

CASA download & installation

Website (casa.nrao.edu)

Compatibility Operating Systems

Monolithic (all-inclusive 'plug-and-play')

Pip-wheel (Pythonic, Jupyter Notebooks, Google Colab)

Full Monolithic Distribution

	Python 2.7	Python 3.6	Python 3.7	Python 3.8
RHEL 6	5.8	<=6.3		
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 12 ARM*				>=6.4

Modular CASA

	Python 2.7	Python 3.6	Python 3.7	Python 3.8
RHEL 6		<=6.3	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 12 ARM				>=6.4

CASA download & installation

Website (casa.nrao.edu)

Compatibility Operating Systems

Monolithic (all-inclusive 'plug-and-play')

Pip-wheel (Pythonic, Jupyter Notebooks, Google Colab)

New release every ~2 months!

Latest version: CASA 6.5

The Release Notes and Known Issues of the 6.5 release are a...

CASA 6.5 is based on Python 3, and available either as a downloadable tar-file distribution with Python environment included, or as a modular version that can be installed with pip-wheels.

Manual processing can be done with any CASA version, but ALMA and VLA pipelines may differ and are not always included, so download the correct CASA version for pipeline use.

	Linux (RedHat 6, 7, 8)	Mac (OS 11, OSX 10.15)
General Use (Notes)	CASA 6.5.2 (RH7/8 - Py 3.8) CASA 6.5.2 (RH7 - Py 3.6)	CASA 6.5.2 (OS11 - Py 3.8) CASA 6.5.2 (OS11 - Py 3.6)
ALMA Pipeline (Notes)	CASA 6.4.1 (RH7/8)	CASA 6.4.1 (OS11) CASA 6.4.1 (10.15)
VLA Pipeline (Notes)	CASA 6.4.1 (RH7/8)	CASA 6.4.1 (OS11) CASA 6.4.1 (10.15)

The above CASA versions can also be downloaded from our NAOJ CASA mirror site and NAOJ CASA-pipeline mirror site, or via Google Drive.

CASA 6: pip-wheel installation

CASA 6 can optionally be installed through modular pip-wheels, with the flexibility to build CASA tools and tasks into a customized Python environment. Instructions on how to install the pip-wheel version of CASA 6 can be found in CASA Docs: CASA 6 Installation and Usage

The modular pip-wheel version is not yet used in production by ALMA and VLA, and does not include any pipelines.

CASA download & installation

Website (ca

Compatibil

Monolithic

Pip-wheel

We execute tasks just like normal Python functions. Many times they will write information to the log or a specified output file, which we then must display.

```
[ ]: from casatasks import listobs

     rc = listobs(vis='sis14_twhya_calibrated_flagged.ms', listfile='obslist.txt', verbose=False, overwrite=True)
     !cat obslist.txt
```

```
================================================================================
           MeasurementSet Name:  /content/sis14_twhya_calibrated_flagged.ms      MS Version 2
================================================================================
   Observer: cqi     Project: uid://A002/X327408/X6f
Observation: ALMA(26 antennas)
Data records: 80563       Total elapsed time = 5647.68 seconds
   Observed from   19-Nov-2012/07:36:57.0   to   19-Nov-2012/09:11:04.7 (UTC)

Fields: 5
  ID   Code Name                RA               Decl           Epoch   SrcId      nRows
  0    none J0522-364           05:22:57.984648 -36.27.30.85128 J2000   0           4200
  2    none Ceres               06:10:15.950590 +23.22.06.90668 J2000   2           3800
  3    none J1037-295           10:37:16.079736 -29.34.02.81316 J2000   3          16000
  5    none TW Hya              11:01:51.796000 -34.42.17.36600 J2000   4          53161
  6    none 3c279               12:56:11.166576 -05.47.21.52464 J2000   5           3402
Spectral Windows:  (1 unique spectral windows and 1 unique polarization setups)
  SpwID  Name                           #Chans   Frame   Ch0(MHz)  ChanWid(kHz)  TotBW(kHz) CtrFreq(MHz) BBC Num  Corrs
  0      ALMA_RB_07#BB_2#SW-01#FULL_RES    384   TOPO  372533.086       610.352    234375.0 372649.9688        2  XX  YY
Antennas: 21 'name'='station'
   ID=   1-4: 'DA42'='A050', 'DA44'='A068', 'DA45'='A070', 'DA46'='A067',
   ID=   5-9: 'DA48'='A046', 'DA49'='A029', 'DA50'='A045', 'DV02'='A077',
   ID= 10-15: 'DV05'='A082', 'DV06'='A037', 'DV08'='A021', 'DV10'='A071',
   ID= 16-19: 'DV13'='A072', 'DV15'='A074', 'DV16'='A069', 'DV17'='A138',
   ID= 20-24: 'DV18'='A053', 'DV19'='A008', 'DV20'='A020', 'DV22'='A011',
   ID= 25-25: 'DV23'='A007'
```

Another example, lets do channel averaging with MSTransform. Here we need to make sure we've deleted the previous output file if/when running multiple times. Since this task doesn't return anything, we can look at the end of the log file to see what happened.

```
[ ]: from casatasks import mstransform

     os.system("rm -fr chanavg.ms")
     mstransform(vis='sis14_twhya_calibrated_flagged.ms', outputvis='chanavg.ms',
                 datacolumn='DATA', chanaverage=True, chanbin=3)
     !tail casa-202*.log
```

```
2021-10-14 17:43:24     INFO    MSTransformManager::parseMsSpecParams   Tile shape is [0]
2021-10-14 17:43:24     INFO    MSTransformManager::parseChanAvgParams  Channel average is activated
2021-10-14 17:43:24     INFO    MSTransformManager::parseChanAvgParams  Channel bin is [3]
2021-10-14 17:43:24     INFO    MSTransformManager::colCheckInfo        Adding DATA column to output MS from input DATA column
2021-10-14 17:43:24     INFO    MSTransformManager::open        Select data
2021-10-14 17:43:24     INFO    MSTransformManager::createOutputMSStructure     Create output MS structure
2021-10-14 17:43:24     INFO    ParallelDataHelper::::casa      Apply the transformations
2021-10-14 17:43:29     INFO    mstransform::::casa     Task mstransform complete. Start time: 2021-10-14 17:43:23.610120 End time: 2021-10-14 17:43:29.323998
2021-10-14 17:43:29     INFO    mstransform::::casa     ##### End Task: mstransform          #####
2021-10-14 17:43:29     INFO    mstransform::::casa     ##########################################
```

The modular pip-wheel version is not yet used in production by ALMA and VLA, and does not include any pipelines.

CASA Docs (casadocs.readthedocs.io)

CASA

Search docs

Common Astronomy Software Applications

CASA, the *Common Astronomy Software Applications*, is the primary data processing software for the Atacama Large Millimeter/submillimeter Array (ALMA) and Karl G. Jansky Very Large Array (VLA), and is often used also for other radio telescopes.

6.5.2 Release

CASA 6.5.2 can now be downloaded for general use. CASA 6.5.2 is available either as a downloadable tar-file, or through pip-wheel installation, which gives flexibility to integrate CASA into a customized Python environment.

Highlights:

- deconvolve: new task for image-domain deconvolution.
- uvcontsub: new implementation, old uvcontsub task deprecated.
- fringefit: support added for 'uvrange' parameter.
- tclean: new iteration control parameter 'nmajor'.
- sdimaging: new parameter 'enablecache' for improved performance.
- mstransform: parameter 'douvcontsub' deprecated.
- flagdata: mode='shadow' now uses the uvw values from the UVW column.
- tclean/tsdimaging: improved runtime performance of ephemeris imaging.
- simulator tool: new parameter 'simint' in sm.settrop() to control time granularity, down to 0.1s.
- ImageAnalysis tool: new string 'mbret' parameter added to 'image.restoringbeam()'.
- casalog tool: new method 'getOrigin()' implemented to retrieve origin of messages.

For more details on these and other new features, see the CASA 6.5.2 Release Notes.

CASA is developed by an international consortium of scientists based at the National Radio Astronomical Observatory (NRAO), the European Southern Observatory (ESO), the National Astronomical Observatory of Japan (NAOJ), the Academia Sinica Institute of Astronomy and

CASA Docs (casadocs.readthedocs.io)

CASA

Search docs

Release Information
Index
API
Task List
Using CASA

Read the Docs v: stable

Versions
latest stable v6.5.2 v6.5.1 v6.5.0 v6.4.4 v6.4.3 v6.4.1 v6.4.0 v6.3.0 v6.2.1 v6.2.0
Downloads
HTML
On Read the Docs
Project Home Builds Downloads
On GitHub
View Edit
Search

Search docs

Hosted by Read the Docs · Privacy Policy

» Common Astronomy Software Applications

Edit on GitHub

Common Astronomy Software Applications

CASA, the *Common Astronomy Software Applications*, is the primary data processing software for the Atacama Large Millimeter/submillimeter Array (ALMA) and Karl G. Jansky Very Large Array (VLA), and is often used also for other radio telescopes.

6.5.2 Release

CASA 6.5.2 can now be downloaded for general use. CASA 6.5.2 is available either as a downloadable tar-file, or through pip-wheel installation, which gives flexibility to integrate CASA into a customized Python environment.

Highlights:

- deconvolve: new task for image-domain deconvolution.
- uvcontsub: new implementation, old uvcontsub task deprecated.
- fringefit: support added for 'uvrange' parameter.
- tclean: new iteration control parameter 'nmajor'.
- sdimaging: new parameter 'enablecache' for improved performance.
- mstransform: parameter 'douvcontsub' deprecated.
- flagdata: mode='shadow' now uses the uvw values from the UVW column.
- tclean/tsdimaging: improved runtime performance of ephemeris imaging.
- simulator tool: new parameter 'simint' in sm.settrop() to control time granularity, down to 0.1s.
- ImageAnalysis tool: new string 'mbret' parameter added to 'image.restoringbeam()'.
- casalog tool: new method 'getOrigin()' implemented to retrieve origin of messages.

For more details on these and other new features, see the CASA 6.5.2 Release Notes.

CASA is developed by an international consortium of scientists based at the National Radio Astronomical Observatory (NRAO), the European Southern Observatory (ESO), the National Astronomical Observatory of Japan (NAOJ), the Academia Sinica Institute of Astronomy and

CASA Reference Papers

PASP in press.
arXiv: 2210.02276
arXiv: 2210.02275

CASA, the Common Astronomy Software Applications for Radio Astronomy

The CASA Team

Ben Bean,[1] Sanjay Bhatnagar,[2] Sandra Castro,[3] Jennifer Donovan Meyer,[4] Bjorn Emonts,[4] Enrique Garcia,[3] Robert Garwood,[4] Kumar Golap,[2] Justo Gonzalez Villalba,[3] Pamela Harris,[2] Yohei Hayashi,[5] Josh Hoskins,[4] Mingyu Hsieh,[2] Preshanth Jagannathan,[2] Wataru Kawasaki,[5] Aard Keimpema,[6] Mark Kettenis,[6] Jorge Lopez,[4] Joshua Marvil,[2] Joseph Masters,[4] Andrew McNichols,[4] David Mehringer,[4] Renaud Miel,[5] George Moellenbrock,[2] Federico Montesino,[3] Takeshi Nakazato,[5] Juergen Ott,[2] Dirk Petry,[3] Martin Pokorny,[2] Ryan Raba,[4] Urvashi Rau,[2] Darrell Schiebel,[4] Neal Schweighart,[4] Srikrishna Sekhar,[7,2] Kazuhiko Shimada,[5] Des Small,[6] Jan-Willem Steeb,[4] Kanako Sugimoto,[5] Ville Suoranta,[4] Takahiro Tsutsumi,[2] Ilse M. van Bemmel,[6] Marjolein Verkouter,[6] Akeem Wells,[4] Wei Xiong,[1] Arpad Szomoru,[6] Morgan Griffith,[4] Brian Glendenning[2] and Jeff Kern[4]

[1] *National Radio Astronomy Observatory, 800 Bradbury Dr., SE Ste 235, Albuquerque, NM 87106, USA*
[2] *National*
[3] *European South*
[4] *National Rad*
[5] *National Astro*
[6] *Joint Institute f*
[7] *Inter-University Institute for Da*

CASA, the *Common* ...
the Atacama Large Milli...
(VLA), and is frequentl...
from single-dish, apertu...
of its core functionalities...
Survey (VLASS), and t...
basic structure of the C...
radio data in CASA. CA...
engineers based at the ...
Observatory (ESO), the National Astronomical Observatory of Japan (NAOJ), and the Joint Institute for VLBI European Research Infrastructure Consortium (JIV-ERIC), under the guidance of NRAO.

CASA on the fringe – Development of VLBI processing capabilities for CASA

Ilse M. van Bemmel,[1] Mark Kettenis,[1] Des Small,[1] Michael Janssen,[2] George A. Moellenbrock,[3] Dirk Petry,[4] Ciriaco Goddi,[5,6] Justin D. Linford,[3] Kazi L. J. Rygl,[7] Elisabetta Liuzzo,[7] Benito Marcote,[1] Olga S. Bayandina,[1,8] Neal Schweighart,[3] Marjolein Verkouter,[1] Aard Keimpema,[1] Arpad Szomoru,[1] and Huib Jan van Langevelde[1,9,10]

[1] *Joint Institute for VLBI ERIC (JIVE), Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, The Netherlands*
[2] *Max-Planck-Institut für Radioastronomie, Auf dem Hügel 69, D-53121 Bonn, Germany*
[3] *National Radio Astronomy Observatory, P.O. Box O, Socorro, NM 87801, USA*
[4] *European Southern Observatory, Karl-Schwarzschild-Strasse 2, 85748 Garching, Germany*
[5] *Dipartimento di Fisica, Universitá degli Studi di Cagliari, SP Monserrato-Sestu km 0.7, I-09042 Monserrato, Italy*
[6] *INAF - Osservatorio Astronomico di Cagliari, via della Scienza 5, I-09047 Selargius (CA), Italy*
[7] *INAF-Istituto di Radioastronomia & Italian ALMA Regional Centre, Via P. Gobetti 101, I-40129 Bologna, Italy*
[8] *INAF – Osservatorio Astrofisico di Arcetri, Largo E. Fermi 5, 50125 Firenze, Italy*
[9] *Leiden Observatory, Leiden University, Postbus 2300, 9513 RA Leiden, The Netherlands*
[10] *University of New Mexico, Department of Physics and Astronomy, Albuquerque, NM 87131, USA*

ABSTRACT

New functionality to process Very Long Baseline Interferometry (VLBI) data has been implemented in the CASA package. This includes two new tasks to handle fringe fitting and VLBI-specific amplitude calibration

JIVE

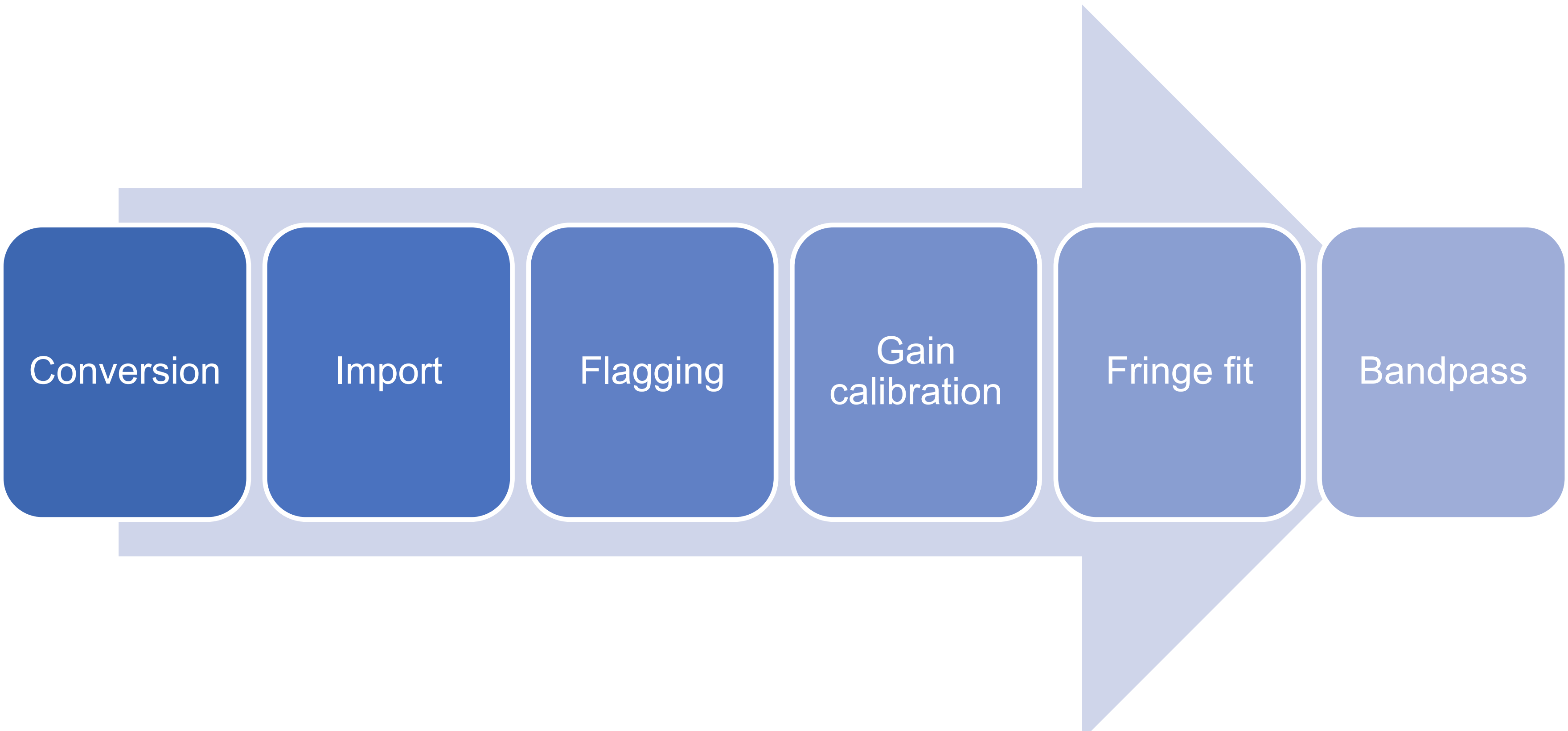
Very Long Baseline Interferometry in the CASA package

Dr. Ilse van Bemmel (JIVE)

Making VLBI accessible

- FAIR data policy
- Develop CASA functionality for VLBI
- Jupyter notebooks with CASA kernel
- CASA-based pipeline development:
  - rPicard (Janssen+ 2019) used for EHT
  - EVN, GMRT, e-MERLIN, others

Calibration of VLBI observations



New CASA tasks

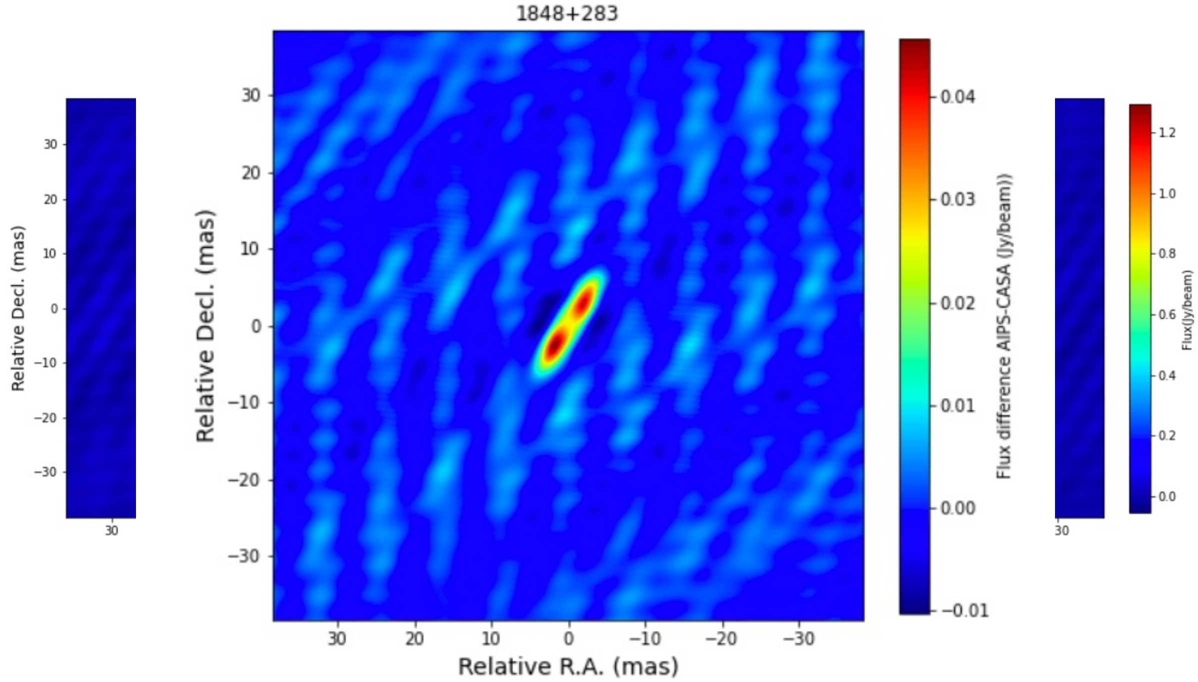
- **`importfitsidi`**
  convert FITS-IDI into MeasurementSet
  includes meta-data tables
  digital corrections for DiFX correlated data

- **`fringefit`**
  correct for residual delay and rate errors
  uses the Schwab-Cotton algorithm (AIPS FRING)
  handles dispersive delay

- **`accor`**
  normalize the visibility amplitudes using auto-correlations
  (AIPS ACCOR & ACSCL)

- Support module to convert system temperature, gain curve and flags

Comparison with AIPS



Future plans

- Pulse cal tone handling
- Fringe fit improvements: baseline stacking, polarization stacking
- Polarization calibration with resolved calibrators
- Ionospheric corrections
- EOP corrections