Simulating ALMA data

CASA

Common Astronomy Software Applications

Bjorn Emonts

NRAO

CASA User Liaison

Credits:

Andrew McNichols (NRAO - CASA)

Remy Indebetouw (NRAO - Pipeline)

CASA Team

NRAO ESO NAOJ ASIAA ALMA CSIRO ASTRON

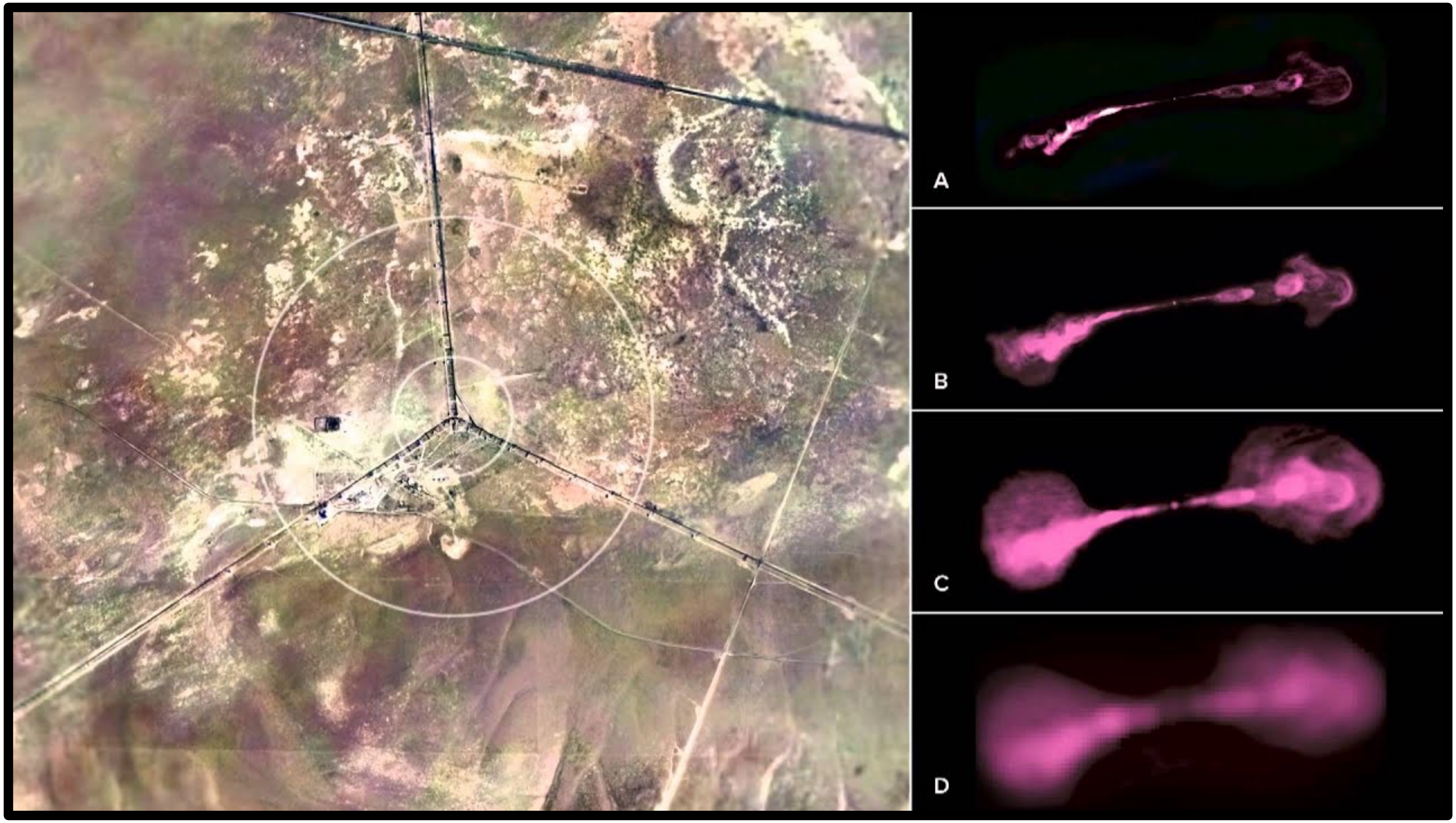
Why simulate radio interferometry observations?

- *Convince TAC that proposed observations are feasible*

- *Convince* ***<u>yourself</u>*** *that proposed observations are feasible!*

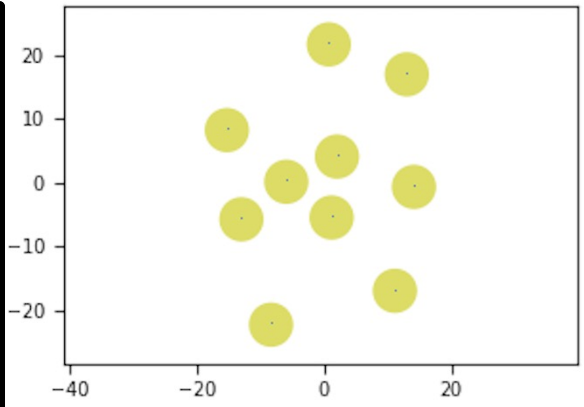
Why simulate radio interferometry observations?

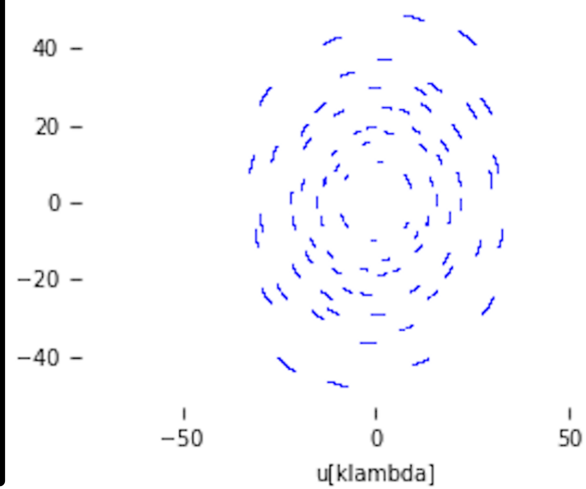
Proposed resolution / array configuration



Why simulate radio interferometry observations?

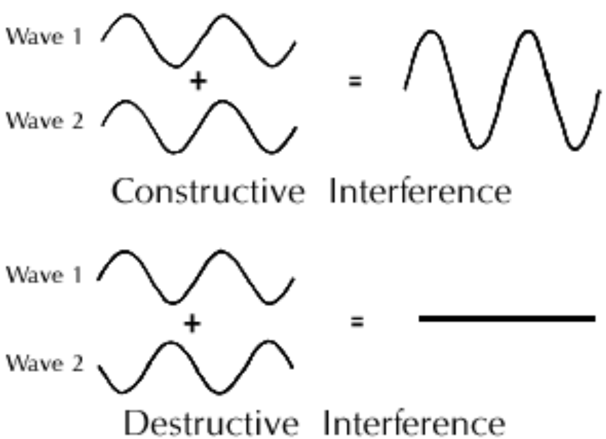
(u,v)-coverage and image quality



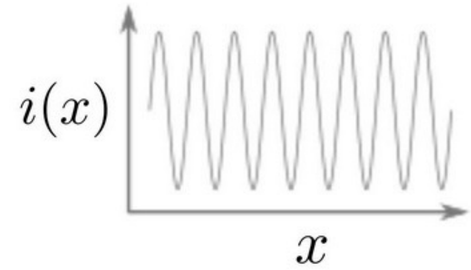


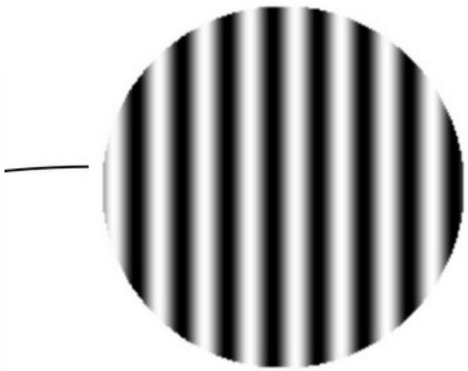
Why simulate radio interferometry observations?

Proposed resolution / array configuration

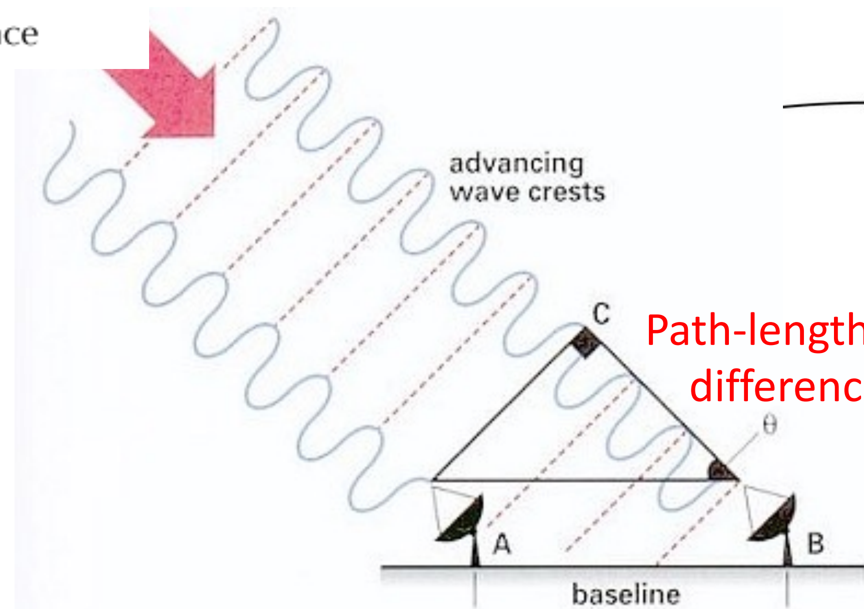


Track point-source on the sky:





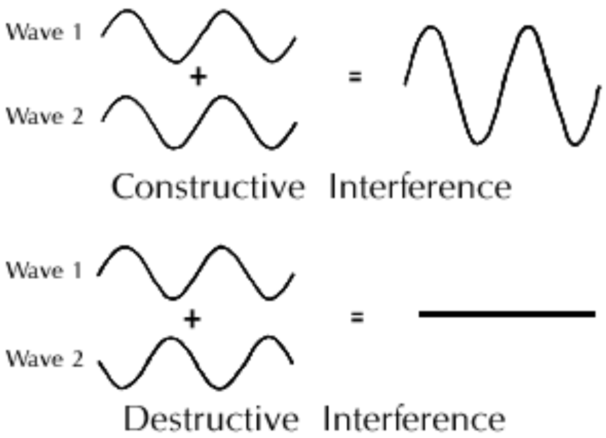
Resolution: **R ~ λ / B**



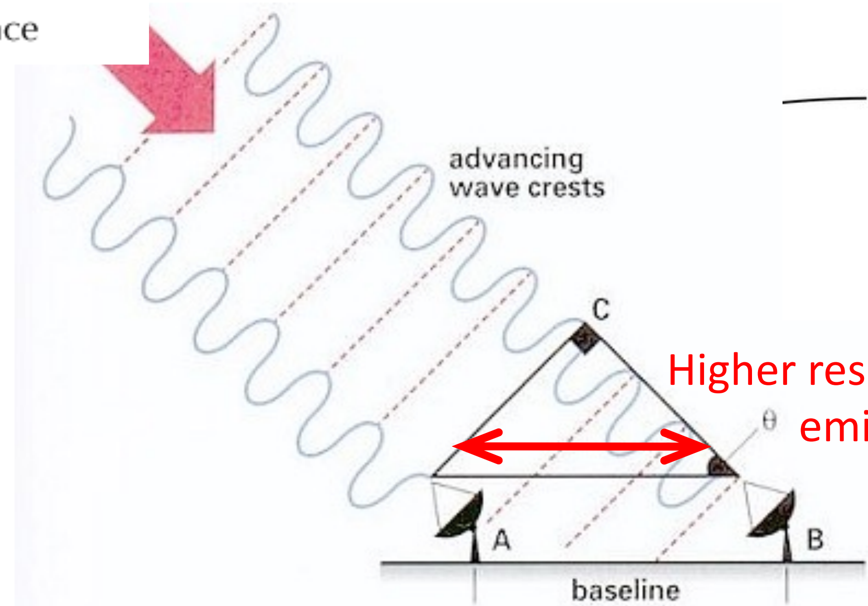
Path-length difference

Why simulate radio interferometry observations?

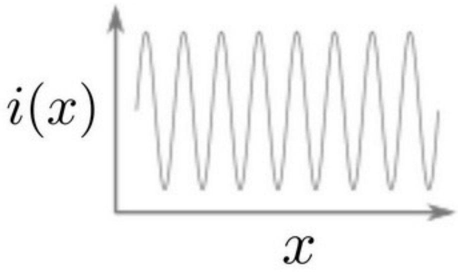
Proposed resolution / array configuration

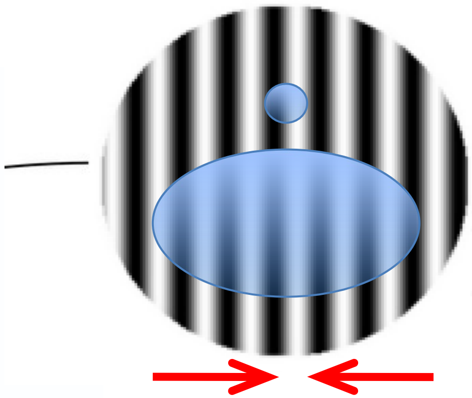


Resolution: **R ~ λ / B**



Track point-source on the sky:





Higher resolution, but resolve out emission on large scales

Why simulate radio interferometry observations?

Proposed resolution / array configuration

- High resolution → imaging small structures

- Surface-brightness sensitivity → imaging extended structures

At mm wavelengths, signal very easily resolved out

Example: ALMA Band 4 (150 GHz):

1km baseline → < 0.5 arcsec

Largest Angular Scale → set by shortest baseline

→ not always useful, need <u>sensitivity</u>

→ For brightness sensitivity, <u>many</u> short baselines

→ Maybe even need for Single Dish *(e.g., Plunkett et al. 2023)*

Why simulate radio interferometry observations?

Proposed resolution / array configuration

- High resolution → imaging small structures
- Surface-brightness sensitivity → imaging extended structures

At mm wavelengths, signal very easily resolved out

Example: ALMA Band 4 (150 GHz):

1km baseline → < 0.5 arcsec

Largest Angular Scale → set by shortest baseline

→ not always useful, need <u>sensitivity</u>

→ For brightness sensitivity, <u>many</u> short baselines

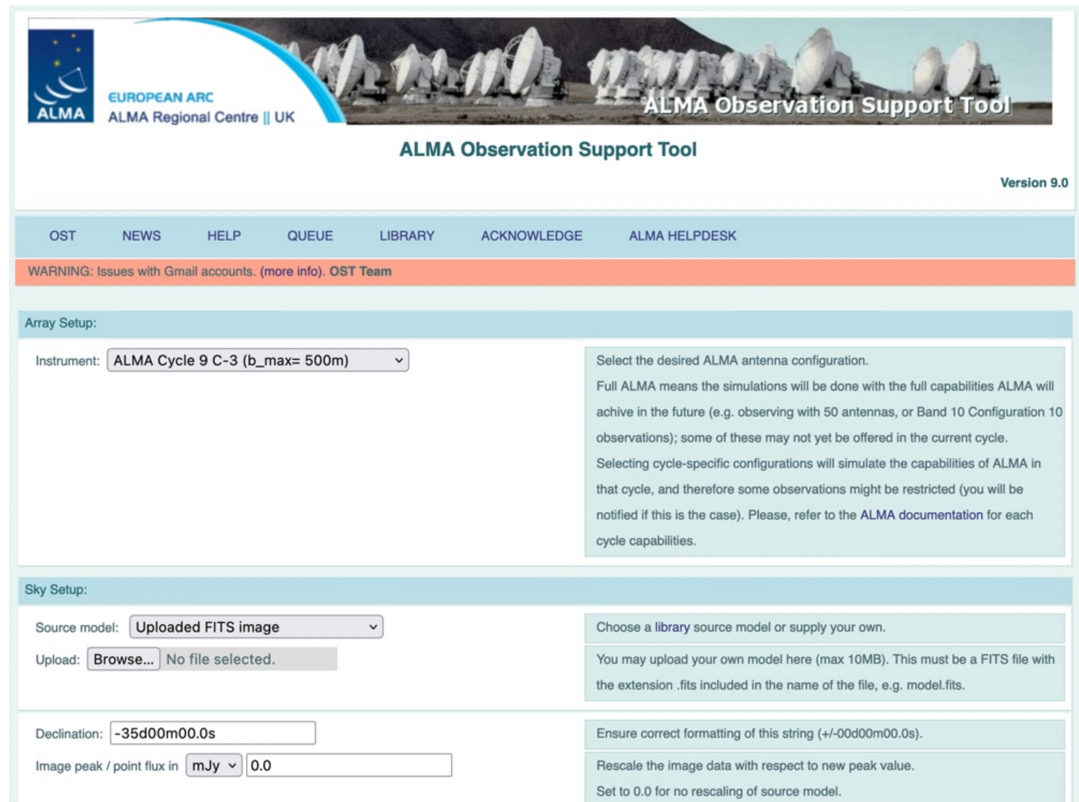
→ Maybe even need for Single Dish *(e.g., Plunkett et al. 2023)*

How to simulate ALMA observations?

1. CASA software
(NRAO, ESO, NAOJ, JIVE)

https://casa.nrao.edu

2. ALMA Observations Support Tool
(online – U.K. ARC, Manchester)



https://almaost.jb.man.ac.uk/

CASA Team

Urvashi Rau (NRAO-SO)	*CASA Lead, Lead scientific development*
Sandra Castro (ESO)	*Lead verification testing*
Josh Marvil (NRAO-SO)	*Lead scientific validation*
George Moellenbrock (NRAO-SO)	*Lead Calibration and VLBI*
Takeshi Nakazato (NAOJ)	*Lead Single Dish, Scientific development*
Darrell Schiebel (NRAO-CV)	*Lead visualization, Infrastructure development*
Jan-Willem Steeb (NRAO-CV)	*Lead infrastructure development*
Ville Suoranta (NRAO-CV)	*Lead Release Engineering*

Victor de Souza Magalhaes (NRAO-ALBQ)	*Scientific development*
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, Infrastructure*
Yohei Hayashi (NAOJ)	*Scientific development, Single Dish*
Josh Hoskins (NRAO-CV)	*Scientific development, Infrastructure*
Andrew McNichols (NRAO-CV)	*Infrastructure, Scientific development*
Dave Mehringer (NRAO-CV)	*Scientific development, Verification testing*
Renaud Miel (NAOJ)	*Scientific development, Single Dish*
Federico Montesino (ESO)	*Infrastructure, Scientific development*
Inna Muzychenko (NAOJ)	*Scientific development, Single Dish*
Dirk Petry (ESO)	*Scientific development*
Neal Schweighart (NRAO-CV)	*Scientific development, Verification testing*
Srikrishna Sekhar (NRAO-SO)	*Scientific development*
Kazuhiko Shimada (NAOJ)	*Scientific development, Single Dish*
Takeshi Shakunaga (NAOJ)	*Scientific development, Single Dish*
Tak Tsutsumi (NRAO-SO)	*Scientific development, Verification testing*
Akeem Wells (NRAO-CV)	*Verification testing*

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 download & installation

New release every 2-3 months!

Website (casa.nrao.edu)

Pipelines (ALMA, VLA)

Monolithic (all-inclusive ‘plug-and-play’)

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

Compatibility Operating Systems

Latest version: CASA 6.6

The Release Notes and Known Issues of the 6.6 release are available in CASA Docs

CASA 6.6 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 7, 8)	Mac (OS 11, 12)
General Use (Notes)	CASA 6.6.3 (RH7) CASA 6.6.3 (RH8)	CASA 6.6.3 (OS12)
ALMA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*
VLA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*

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)

Pipelines (ALMA, VLA)

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

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

Compatibility Operating Systems

Latest version: CASA 6.6

The Release Notes and Known Issues of the 6.6 release are available in CASA Docs

CASA 6.6 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.

ALMA Pipeline release
1x per year

		Mac
General Use (Notes)	CASA 6.6.3 (RH8)	
ALMA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*
VLA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*

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)

Pipelines (ALMA, VLA)

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

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

Compatibility Operating Systems

Latest version: CASA 6.6

The Release Notes and Known Issues of the 6.6 release are available in CASA Docs

CASA 6.6 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.

ALMA Pipeline release 1x per year

		Mac
General Use (Notes)	CASA 6.6.3 (RH8)	
ALMA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*
VLA Pipeline (Notes)	CASA 6.5.4 (RH7/8)	CASA 6.5.4 (OS11)*

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 (cas

Pipelines (A

Monolithic (a

Pip-wheel (F
C

Compatibilit

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     ##########################################
```

include any pipelines.

CASA download & installation

Website (casa.nrao.edu)

Pipelines (ALMA, VLA)

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

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

Compatibility Operating Systems

Full Monolithic Distribution

	Python 2.7	Python 3.6	Python 3.7	Python 3.8
RHEL 6		<=6.3	6.2	6.2
RHEL 7		>=6.0, <6.5.6	>=6.2,<6.5.6	>=6.2
RHEL 8		>=6.0	>=6.4,<6.5.6	>=6.4
Ubuntu 18.04		>=6.0, <6.5.6	>=6.2,<6.5.6	>=6.2
Ubuntu 20.04		>=6.0, <6.5.6	>=6.2,<6.5.6	
Mac OS 10.14		>=6.1, <6.2		<=6.3
Mac OS 10.15		>=6.1, <6.5.2		>=6.3, <6.5.2
Mac OS 11 x86		>=6.3, <6.5.6		>=6.3, <6.5.6
Mac OS 12 ARM*				>=6.4
Mac OS 13 ARM*				>=6.5.5

Modular CASA

	Python 2.7	Python 3.6	Python 3.7	Python 3.8	Python 3.10
RHEL 6		<=6.3	6.2	6.2	
RHEL 7		>=6.0, <6.5.6	>=6.2,<6.5.6	>=6.2	>=6.6
RHEL 8		>=6.0	>=6.4,<6.5.6	>=6.4	>=6.6
Ubuntu 18.04		>=6.0, <6.5.6	>=6.2,<6.5.6	>=6.2	>=6.6
Ubuntu 20.04		>=6.0, <6.5.6	>=6.2,<6.5.6		>=6.6
Mac OS 10.14		>=6.1, <6.2		<=6.3	
Mac OS 10.15		>=6.1, <6.5.2		>=6.3, <6.5.2	
Mac OS 11 x86		>=6.3, <6.5.6		>=6.3, <6.5.6	
Mac OS 12 ARM*				>=6.4	>=6.6
Mac OS 13 ARM*				>=6.5.5	>=6.6

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
- **Data Repository**: Earth Orientation Parameters, reference frames, ephemeris data, beam models, *simulator config files*, etc

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

The CASA Software

Experienced users

- **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
- **Data Repository**: Earth Orientation Parameters, reference frames, ephemeris data, beam models, *simulator config files*, etc

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

CASA Documentation: CASA Docs on github

CASA

Search docs

- Release Information
- Index
- API
- ⊞ Task List
- Using CASA
- CASA Fundamentals
- External Data
- Calibration & Visibilities
- Imaging & Analysis
- CARTA
- Pipeline
- Simulations
- Parallel Processing
- Memo Series & Knowledgebase
- Community Examples
- Citing CASA
- Change Log

Read the Docs v: stable

Versions

latest **stable** v6.6.0 v6.5.6 v6.5.5
v6.5.4 v6.5.3 v6.5.2 v6.5.1 v6.5.0
v6.4.4 v6.4.3 v6.4.1 v6.4.0 v6.3.0

» API » casatasks   Edit on GitHub

casatasks

Tasks in CASA are python interfaces to the more basic toolkit. Tasks are executed to perform a single job, such as loading, plotting, flagging, calibrating, and imaging the data.

The parameters used and their defaults can be obtained by typing `help(<taskname>)` at the Python prompt, where `<taskname>` is the name of a given task. This command lists all parameters, a brief description of the parameter, the parameter default, and any options if there are limited allowed values for the parameter.

Experimental tasks and algorithms

Some tasks and algorithms in CASA are labelled as **Experimental** or **Unverified**. These tasks have not been fully commissioned and/or verified. Such tasks are provided to enhance user capabilities, or because they are required for specific pipeline use.

The label *Experimental* or *Unverified* means that the task/algorithm falls under the following disclaimers:

- Only a subset of modes have been incorporated into CASA unit/regression tests. These are documented in CASA Docs. Other options/modes may be run, and might work just fine, but they are not part of what has been tested carefully.
- Some parameters have been tested for specific use cases (as part of the algorithm development, publication, and CASA test programs), but we have not yet established best practices for all different situations. This information will build over time and will be incorporated into our documentation as appropriate.
- Experimental tasks and algorithms may have Known Issues, representing CASA's current understanding of the state of the code. These Known Issues are clearly defined as part of CASA Docs.
- Parameter names and task structure can change, based on feedback and improved understanding of usability.

It is expected that ALMA and VLA pipelines will begin using experimental tasks only after they have stabilized for stand-alone use.

The complete listing of tasks available in CASA is as follows:

Input / Output

`exportasdm`	Convert a CASA visibility file (MS) into an ALMA or EVLA Science Data Model
`exportfits`	Convert a CASA image to a FITS file
`exportuvfits`	Convert a CASA visibility data set to a UVFITS file:

How to simulate ALMA observations?

Common Astronomy Software Applications

https://casa.nrao.edu

1. CASA simulation tasks:

- simobserve: model → simulated MS
- simanalyze: simulated MS → image (tclean)

simalma

2. Simulator tools:

sm tool / simutil

Use CASA 6 (or at least 5.7+)

(CASA 5.7: simulator upgraded clean → tclean)

How to simulate ALMA observations?

1. CASA simulation tasks:

- simobserve: model → simulated MS
- simanalyze: simulated MS → image (tclean)

simalma

2. Simulator tools:

sm tool / simutil

3. Configuration files:

ALMA Cycle 0 – 11 + ACA

VLA, ngVLA, ATCA, PdbI, WSRT, CARMA, MeerKAT, SMA, VLBA

Note: ALMA Cycle 6-11 config files unchanged

CASA
Common Astronomy Software Applications

https://casa.nrao.edu

How to simulate ALMA observations?

CASA
Common Astronomy Software Applications

https://casa.nrao.edu

1. CASA simulation tasks:

- simobserve: model → simulated MS
- simanalyze: simulated MS → image (tclean)

simalma

2. Simulator tools:

sm tool / simutil

3. Configuration files:

ALMA Cycle 0 – 11 + ACA

VLA, ngVLA, ATCA, PdbI, WSRT, CARMA, MeerKAT, SMA, VLBA

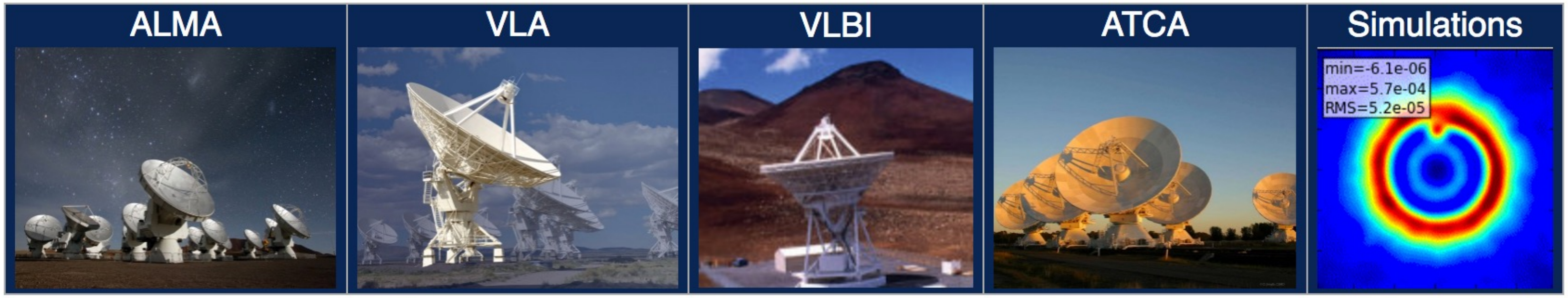
4. Complex (non-ALMA) simulations:

Notebook examples CASA Docs

How to simulate ALMA observations?

CASA Guides

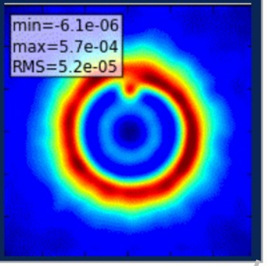
Telescope-specific tutorials
→ data processing strategies
https://casaguides.nrao.edu/



How to simulate ALMA observations?

CASA Guides

ALMA | VLA | VLBI | ATCA | Simulations

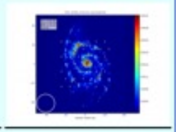


Simulating ngVLA Data (CASA 5.4)

This tutorial shows how to create simulated data for the next generation Very Large Array (ngVLA) either by using simobserve or the sm toolkit. Additionally, it shows how to estimate the scaling parameter for adding thermal noise using the sm.setnoise function and the simplenoise parameter.

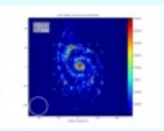
Simalma (CASA 6.4.1)

This tutorial demonstrates how to use **simalma**, a task that simplifies simulations that include the main 12-m array plus the ACA. Like the previous guide, this one is of particular interest to those wishing to explore multi-component ALMA observations.



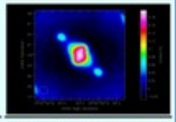
ACA Simulation (CASA 5.4)

A tutorial for simulating ALMA observations that use multiple configurations or use the 12-meter array in combination with the ALMA Compact Array. This tutorial demonstrates combining data from each ALMA component "by hand". This guide is of particular interest to those wishing to explore using the 12-m array in combination with the ACA, and those interested in combining data from multiple 12-m array configurations.



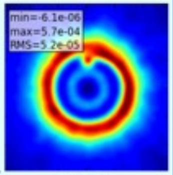
Simulation Guide Component Lists (CASA 6.5.3)

Tutorial for simulating data based on multiple sources (using both a FITS image and a component list). If you are interested in simulating from a list of simple sources (point, Gaussian, disk), rather than or in addition to a sky model image, then read the considerations here.



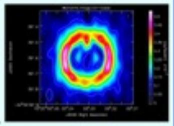
Protoplanetary Disk Simulation (CASA 5.4)

A sky model with a lightly annotated script that simulates a protoplanetary disk. Uses a theoretical model of dust continuum from Sebastian Wolff, scaled to the distance of a nearby star. This is another fairly generic simulation - if you're short on time, you probably don't need to go through this one and the New Users guide, but it can be useful to go through multiple examples.



Protoplanetary Disk Simulation - VLA (CASA 5.5)

This tutorial explains the steps for simulating VLA observations using the same protoplanetary disk sky model that was used for the analogous ALMA tutorial. Observational and analysis parameters are changed step by step and the results are compared to the VLA exposure calculator.



Advanced: Corrupting Simulated Data (Simulator Tool)

simobserve calls methods in the **simulator** tool. For advanced CASA users, the 'simulator' tool has methods that can add to simulated data: phase delay variations, gain fluctuations and drift, cross-polarization, and bandpass and pointing errors. 'simulator' also has more flexibility than simobserve in adding thermal noise. The tutorial linked from this page describes the simulation of data using the task interface only. To learn more about the 'simulator' tool, see the CASA Toolkit Reference Manual. An examples of advanced techniques for corrupting a simulated MeasurementSet can be found in this CASA Guide on Corrupting Simulated Data (Simulator Tool)

SIMALMA

CASA Guides:
https://casaguides.nrao.edu/

```
# Model sky = Halpha image of M51
os.system('curl https://casaguides.nrao.edu/images/3/3f/M51ha.fits.txt -f -o M51ha.fits')
skymodel         =  "M51ha.fits"
```

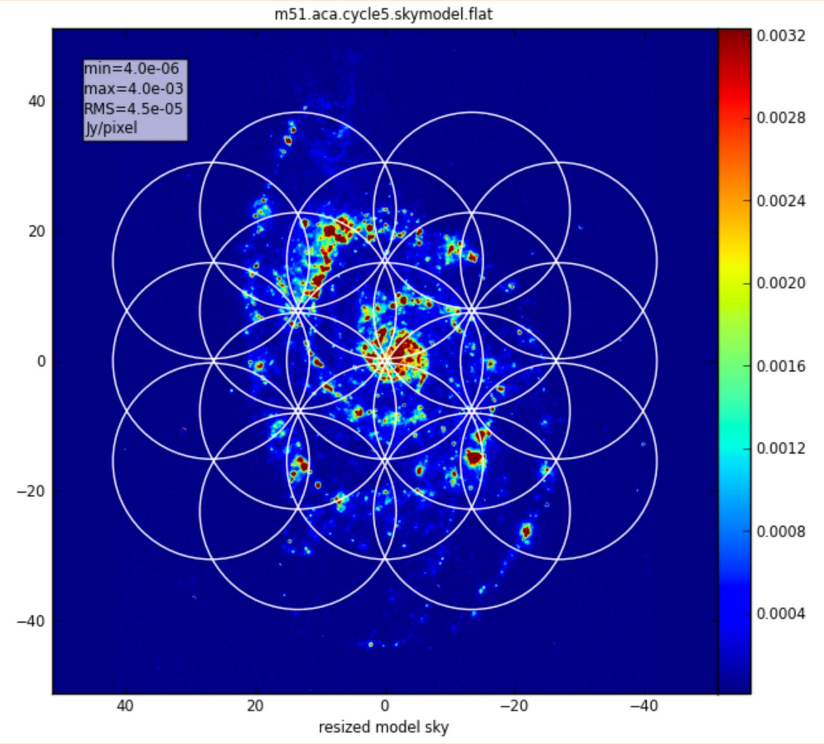
```
# Set model image parameters:
indirection="J2000 23h59m59.96s -34d59m59.50s"
incell="0.1arcsec"
inbright="0.004"
incenter="330.076GHz"
inwidth="50MHz"
```

```
antennalist=["alma.cycle8.3.cfg","aca.cycle8.cfg"]
```

```
totaltime="1800s"
tpnant = 2
tptime="7200s"
pwv=0.6
mapsize="1arcmin"
```

```
inp
```

```
go
```



SIMALMA

```
# Model sky = Halpha image of M51
os.system('curl https://casaguides.nrao.edu/image
skymodel              =  "M51ha.fits"
```

```
# Set model image parameters:
indirection="J2000 23h59m59.96s -34d59m59.50s"
incell="0.1arcsec"
inbright="0.004"
incenter="330.076GHz"
inwidth="50MHz"
```

```
antennalist=["alma.cycle8.3.cfg","aca.cycle8.cfg"]
```

```
totaltime="1800s"
tpnant = 2
tptime="7200s"
pwv=0.6
mapsize="1arcmin"
```

```
inp
```

```
go
```

IPython: CASA_testing/Simulations

File Edit View Search Terminal Help

```
---------> inp()
#  simalma :: Simulation task for ALMA
project             =      'm51'        #  root prefix for output file names
dryrun              =      False        #  dryrun=True will only produce the
                                        #   informative report, not run
                                        #   simobserve/analyze
skymodel            = 'M51ha.fits'      #  model image to observe
     inbright       =    '0.004'        #  scale surface brightness of brighte
                                        #   pixel e.g. "1.2Jy/pixel"
     indirection    = 'J2000 23h59m59.96s -34d59m59.50s' #  set new direction
                                        #   e.g. "J2000 19h00m00 -40d00m00"
     incell         = '0.1arcsec'       #  set new cell/pixel size e.g.
                                        #   "0.1arcsec"
     incenter       = '330.076GHz'      #  set new frequency of center channel
                                        #   e.g. "89GHz" (required even for 2D
                                        #   model)
     inwidth        =     '50MHz'       #  set new channel width e.g. "10MHz"
                                        #   (required even for 2D model)

complist            =         ''        #  componentlist to observe
setpointings        =       True
     integration    =      '10s'        #  integration (sampling) time
     direction      =         ''        #  "J2000 19h00m00 -40d00m00" or "" to
                                        #   center on model
     mapsize        =   '1arcmin'       #  angular size of map or "" to cover
                                        #   model

antennalist         = ['alma.cycle8.3.cfg', 'aca.cycle8.cfg'] #  antenna
                                        #   position files of ALMA 12m and 7m
                                        #   arrays
hourangle           =  'transit'        #  hour angle of observation center e.
                                        #   -3:00:00, or "transit"
totaltime           =    '1800s'        #  total time of observation; vector
                                        #   corresponding to antennalist
tpnant              =          2        #  Number of total power antennas to u
                                        #   (0-4)
     tptime         =    '7200s'        #  total observation time for total
                                        #   power

pwv                 =        0.6        #  Precipitable Water Vapor in mm. 0 f
                                        #   noise-free simulation
image               =       True        #  image simulated data
     imsize         =          0        #  output image size in pixels (x,y) o
                                        #   0 to match model
     imdirection    =         ''        #  set output image direction,
                                        #   (otherwise center on the model)
     cell           =         ''        #  cell size with units or "" to equal
                                        #   model
     niter          =          0        #  maximum number of iterations (0 for
                                        #   dirty image)
     threshold      =   '0.1mJy'        #  flux level (+units) to stop cleanin

graphics            =     'both'        #  display graphics at each stage to
                                        #   [screen|file|both|none]
verbose             =      False
overwrite           =       True        #  overwrite files starting with
                                        #   $project

CASA <67>: go
```

SIMALMA

```
# Model sky = Halpha image of M51
os.system('curl https://casaguides.nrao.edu/image
skymodel            =  "M51ha.fits"
```

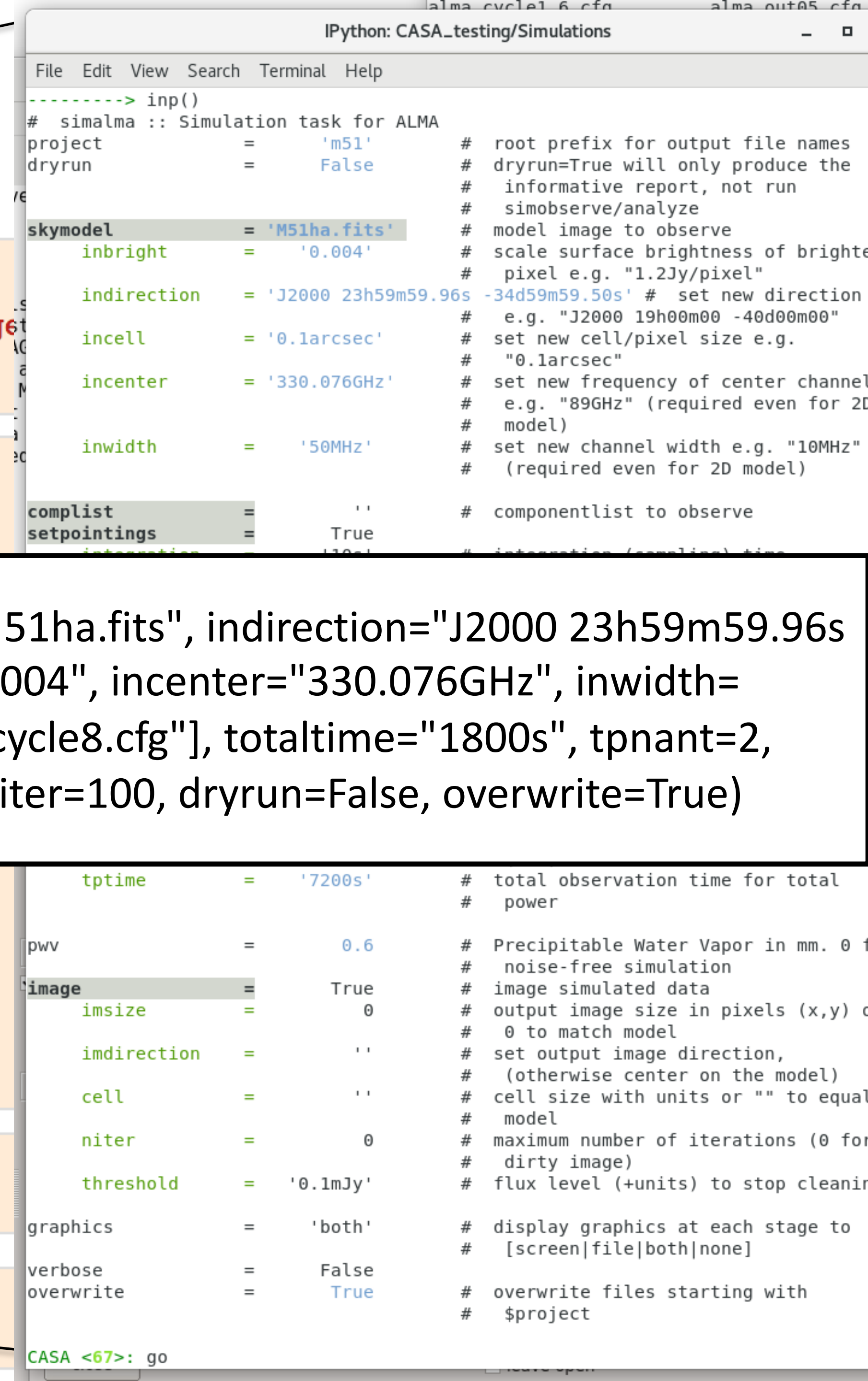
```
# Set model image parameters:
indirection="J2000 23h59m59.96s -34d59m59.50s"
```

simalma(project="M51_compact", skymodel="M51ha.fits", indirection="J2000 23h59m59.96s -34d59m59.50s", incell="0.1arcsec", inbright="0.004", incenter="330.076GHz", inwidth= "50MHz", antennalist=["alma.cycle8.3.cfg","aca.cycle8.cfg"], totaltime="1800s", tpnant=2, tptime="7200s", pwv=0.6, mapsize="1arcmin", niter=100, dryrun=False, overwrite=True)

```
totaltime="1800s"
tpnant = 2
tptime="7200s"
pwv=0.6
mapsize="1arcmin"
```

```
inp
```

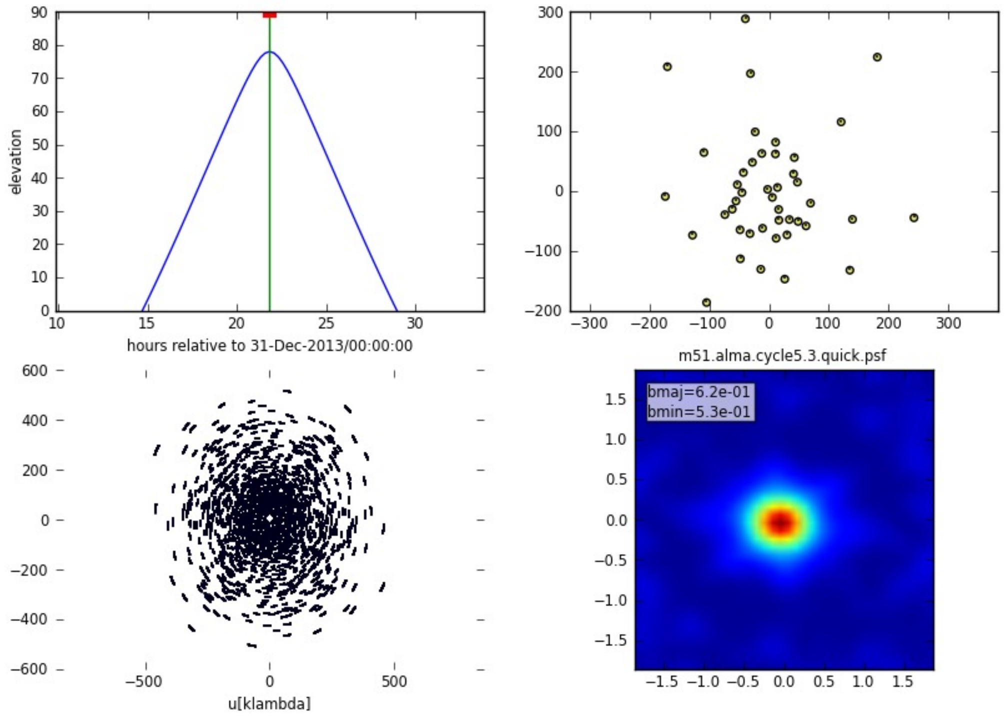
```
go
```



SIMALMA

1. Simobserve

Simulate visibilities (MS) for each configuration



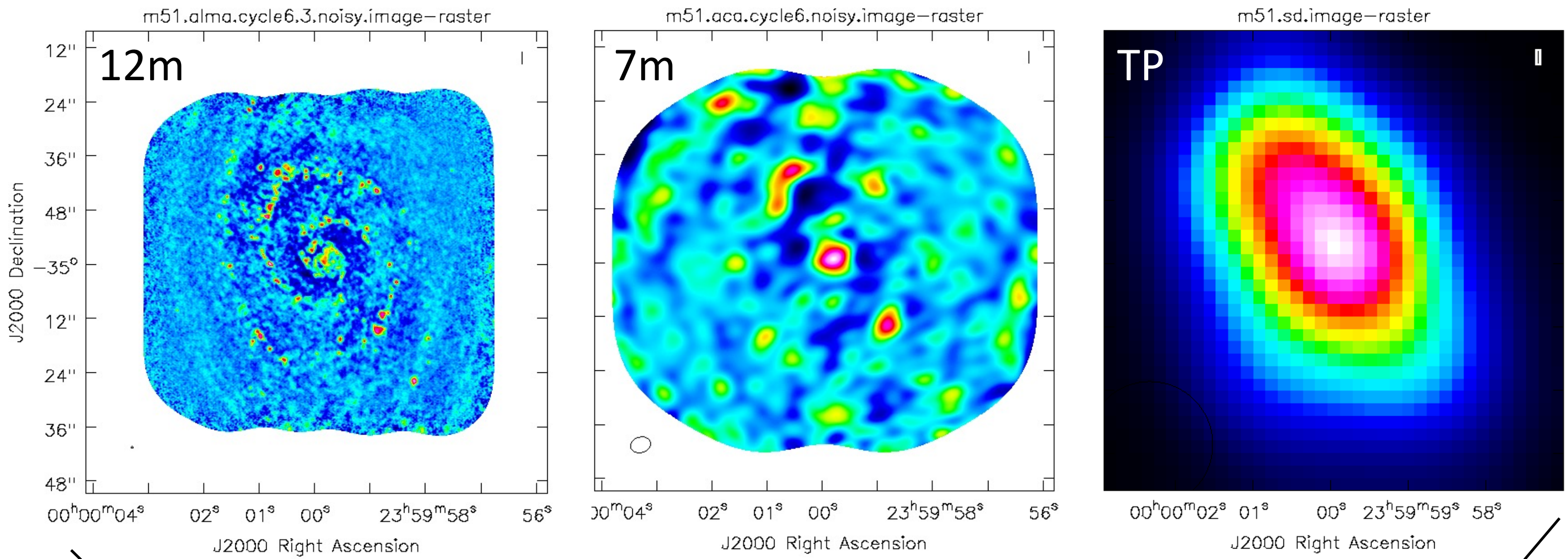
2. Simanalyze

Imaging using simulated MSs

SIMALMA

1. Simc

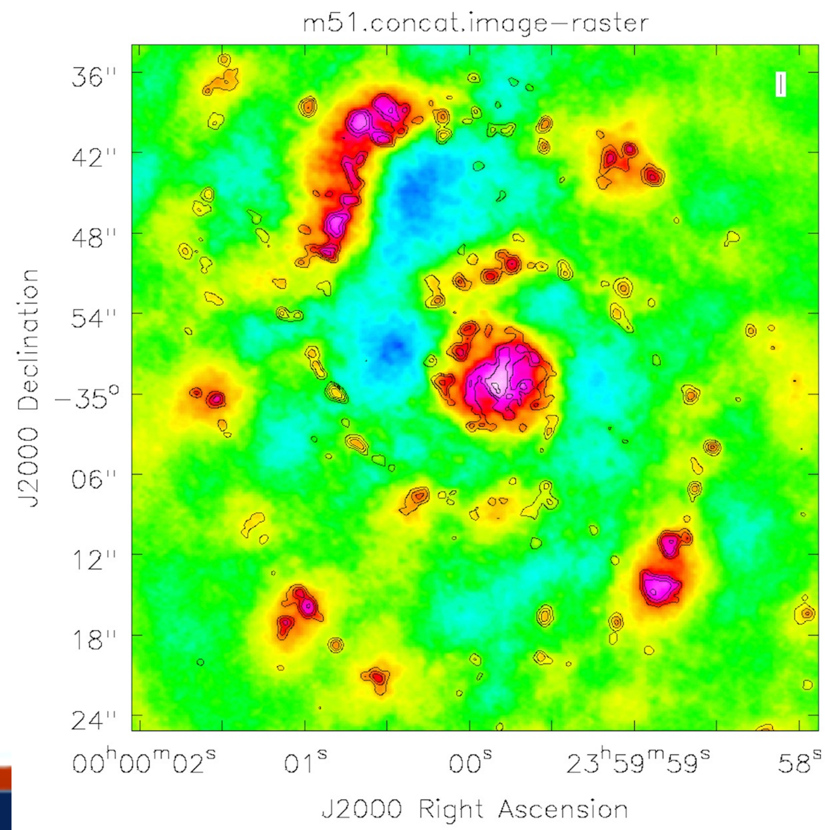
Simula
each c



2. Simanalyze

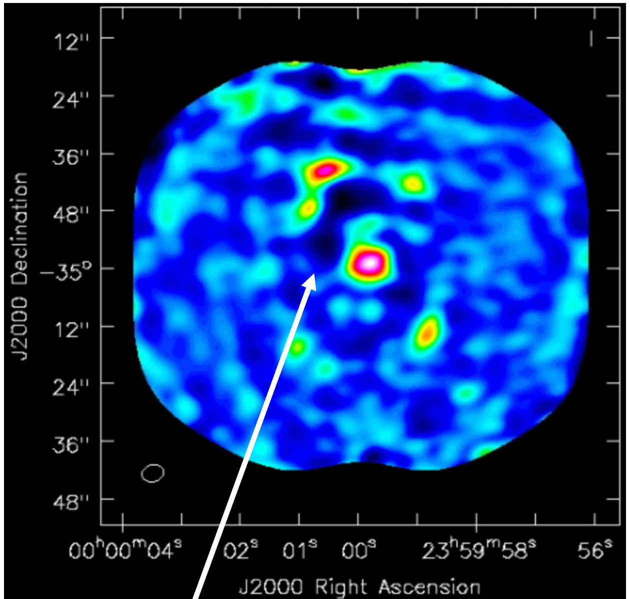
Imaging using
Simulated MSs

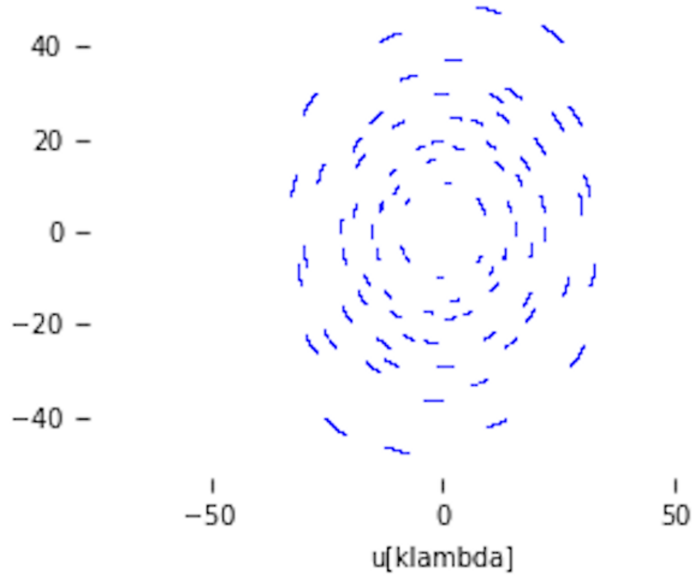
Note: can also use tclean + feather), or sdintimaging



Simulating combinations ALMA / ACA / TP

ACA (7m) → spare (u,v)-coverage

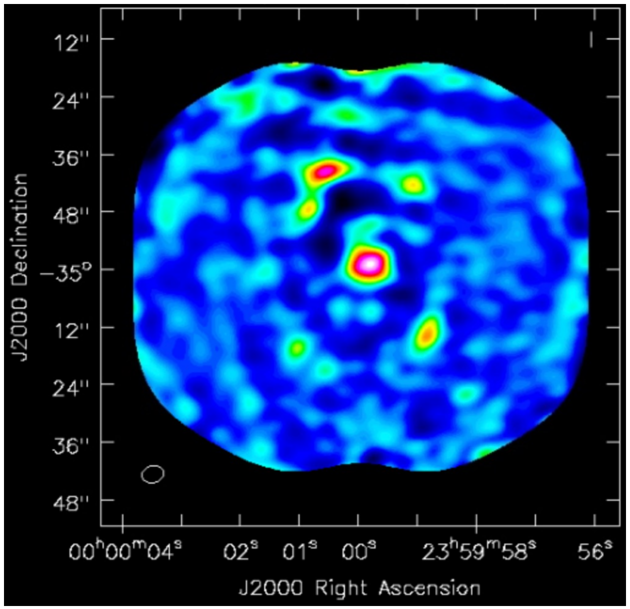




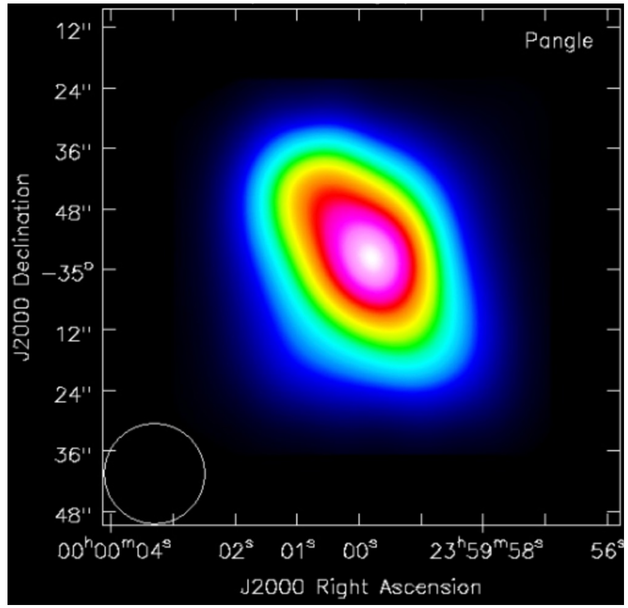
Artifacts large-scale emission

Combining ALMA / ACA / TP

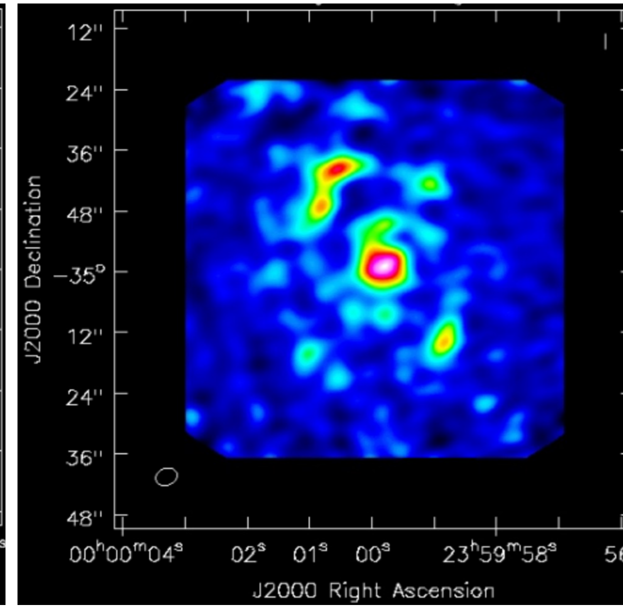
ACA (7m) + single dish (Total Power)



ACA (7m)



TP (single dish)

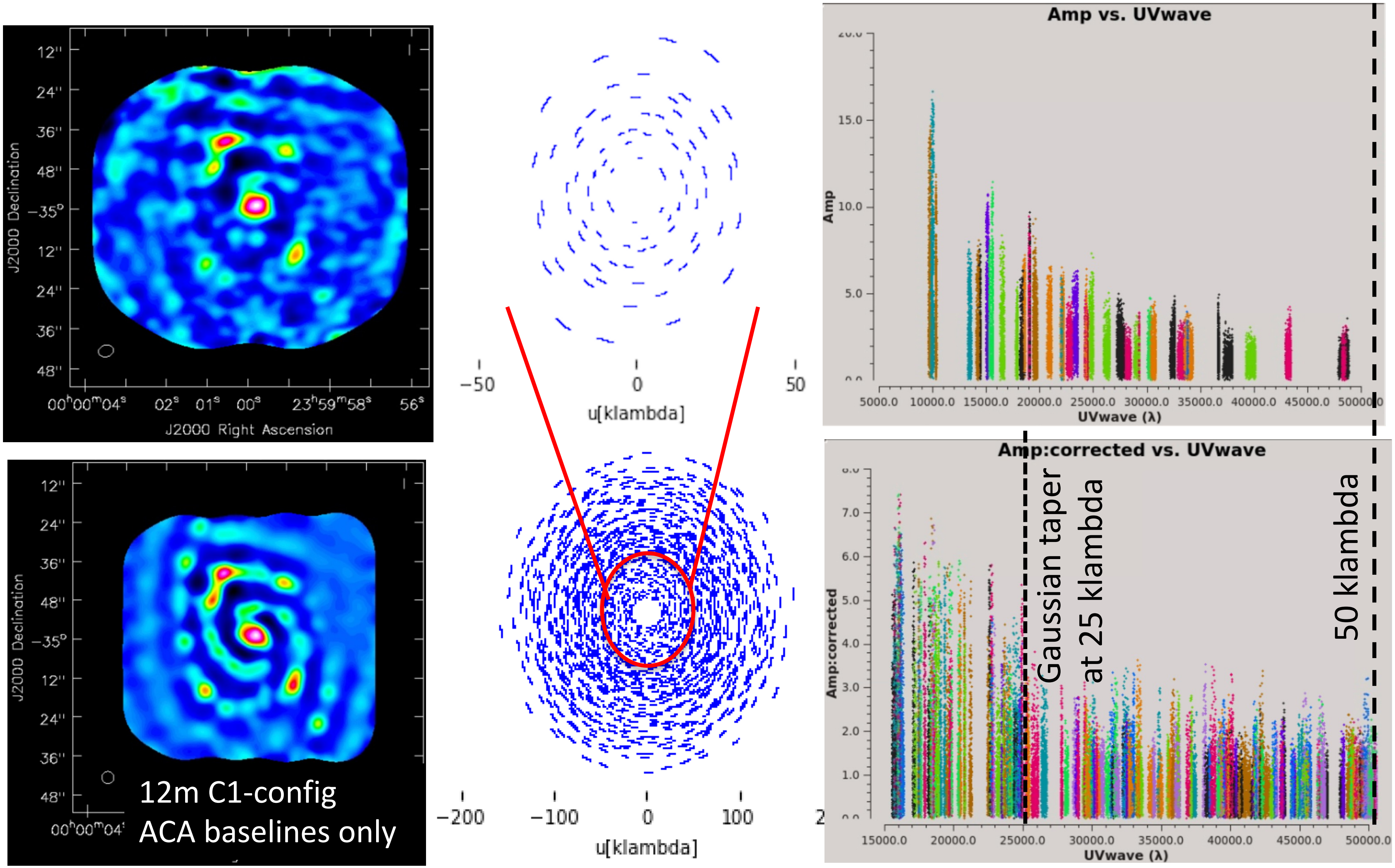


ACA + TP

Simalma → Adding TP reduces artifact of large-scale emission

Simulating combinations ALMA / ACA / TP

ACA (7m) + compact baselines 12m C-1

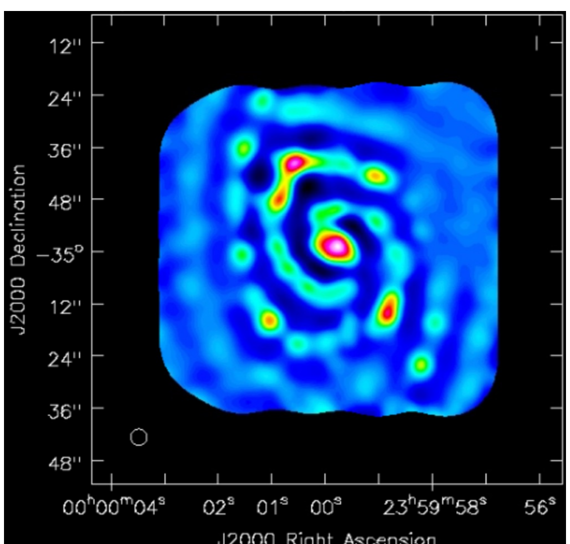


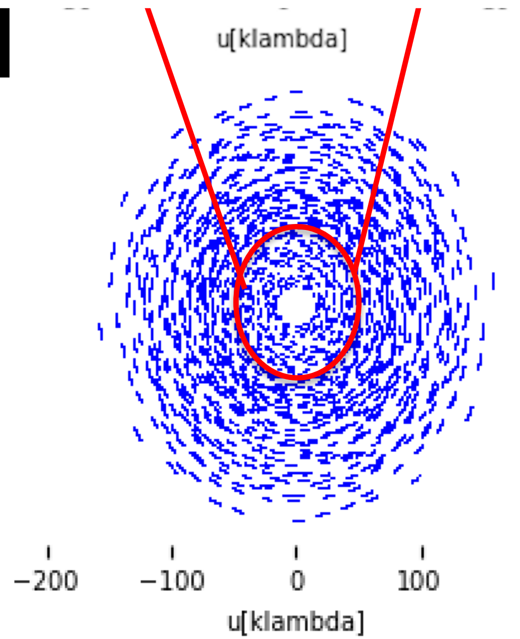
Simulating combinations ALMA / ACA / TP

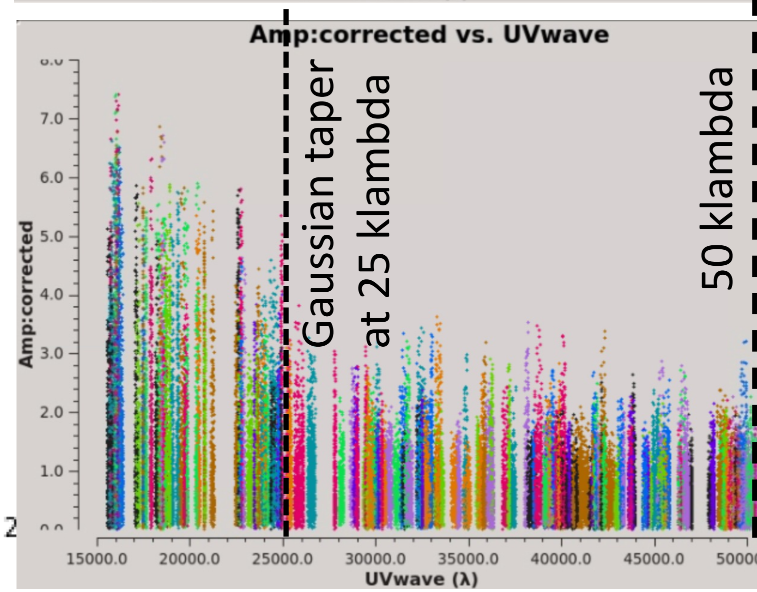
ACA (7m) + compact baselines 12m C-1

Flexibility CASA simulations: output MS simobserve → input tclean

tclean(vis='M51_compact.alma.cycle8.1.noisy.ms', imagename='output_image', datacolumn='corrected', uvrange='0~50klambda', phasecenter='J2000 23h59m59.96s -34d59m59.50s', imsize=[1080], cell=['0.1arcsec'], specmode='mfs', gridder='mosaic', niter=100, pbcor=True, weighting='natural', uvtaper='25klambda')

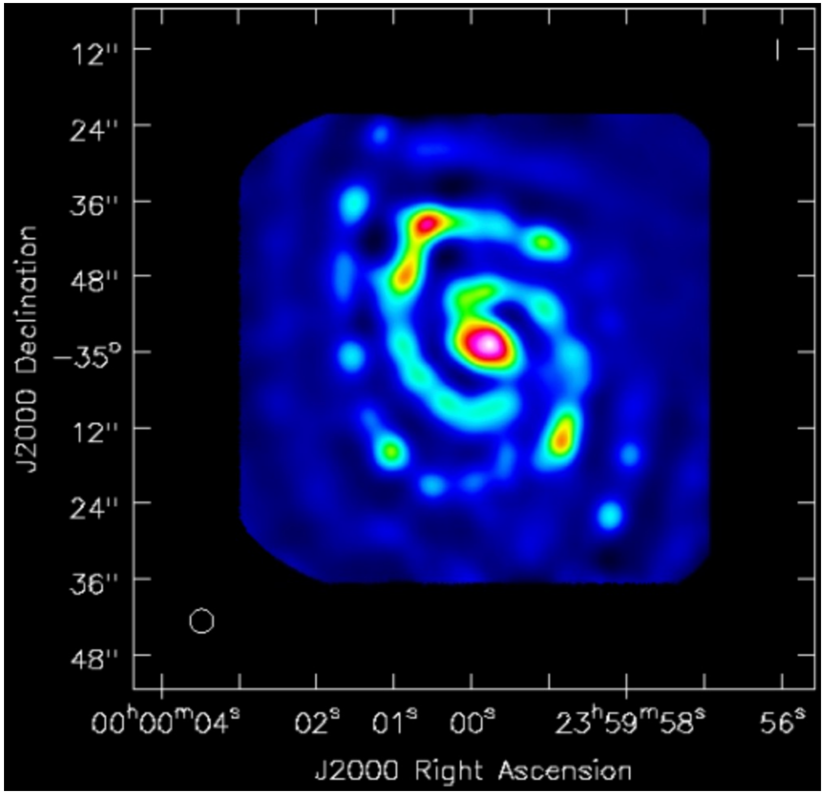




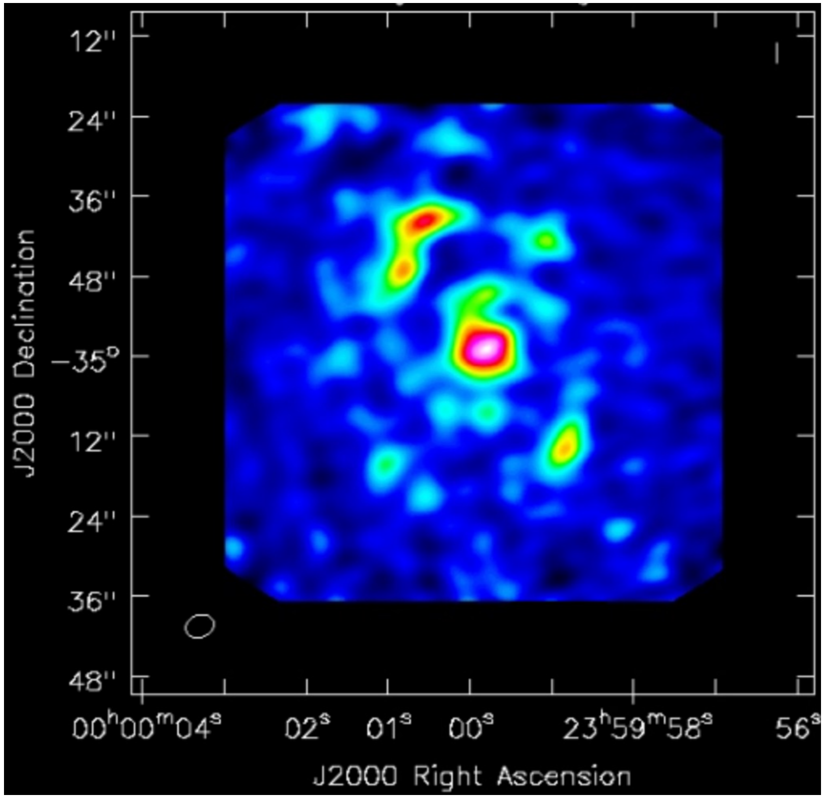


Combining ALMA / ACA / TP

ACA (7m) + compact baselines 12m C-1



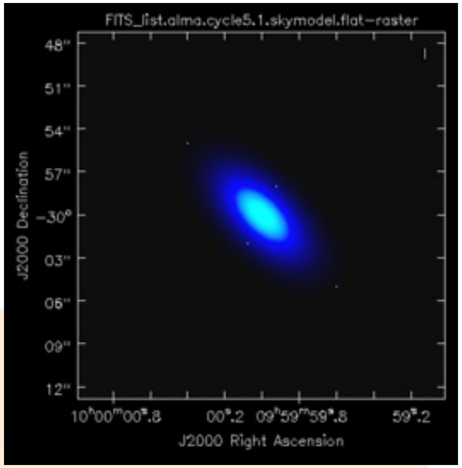
ACA + TP + 12m C1 (ACA baselines)



ACA + TP

Simulating w. Component List

CASA Guides:
https://casaguides.nrao.edu/




```
# In CASA
direction = "J2000 10h00m00.0s -30d00m00.0s"
cl.done()
cl.addcomponent(dir=direction, flux=1.0, fluxunit='Jy', freq='230.0GHz', shape="Gaussian",
                majoraxis="0.1arcmin", minoraxis='0.05arcmin', positionangle='45.0deg')
#
ia.fromshape("Gaussian.im",[256,256,1,1],overwrite=True)
cs=ia.coordsys()
cs.setunits(['rad','rad','','Hz'])
cell_rad=qa.convert(qa.quantity("0.1arcsec"),"rad")['value']
cs.setincrement([-cell_rad,cell_rad],'direction')
cs.setreferencevalue([qa.convert("10h",'rad')['value'],qa.convert("-30deg",'rad')['value']],type="direction")
cs.setreferencevalue("230GHz",'spectral')
cs.setincrement('1GHz','spectral')
ia.setcoordsys(cs.torecord())
ia.setbrightnessunit("Jy/pixel")
ia.modify(cl.torecord(),subtract=False)
exportfits(imagename='Gaussian.im',fitsimage='Gaussian.fits',overwrite=True)
```

```
# In CASA
os.system('rm -rf point.cl')
cl.done()
cl.addcomponent(dir="J2000 10h00m00.08s -30d00m02.0s", flux=0.1, fluxunit='Jy', freq='230.0GHz', shape="point")
cl.addcomponent(dir="J2000 09h59m59.92s -29d59m58.0s", flux=0.1, fluxunit='Jy', freq='230.0GHz', shape="point")
cl.addcomponent(dir="J2000 10h00m00.40s -29d59m55.0s", flux=0.1, fluxunit='Jy', freq='230.0GHz', shape="point")
cl.addcomponent(dir="J2000 09h59m59.60s -30d00m05.0s", flux=0.1, fluxunit='Jy', freq='230.0GHz', shape="point")
cl.rename('point.cl')
cl.done()
```

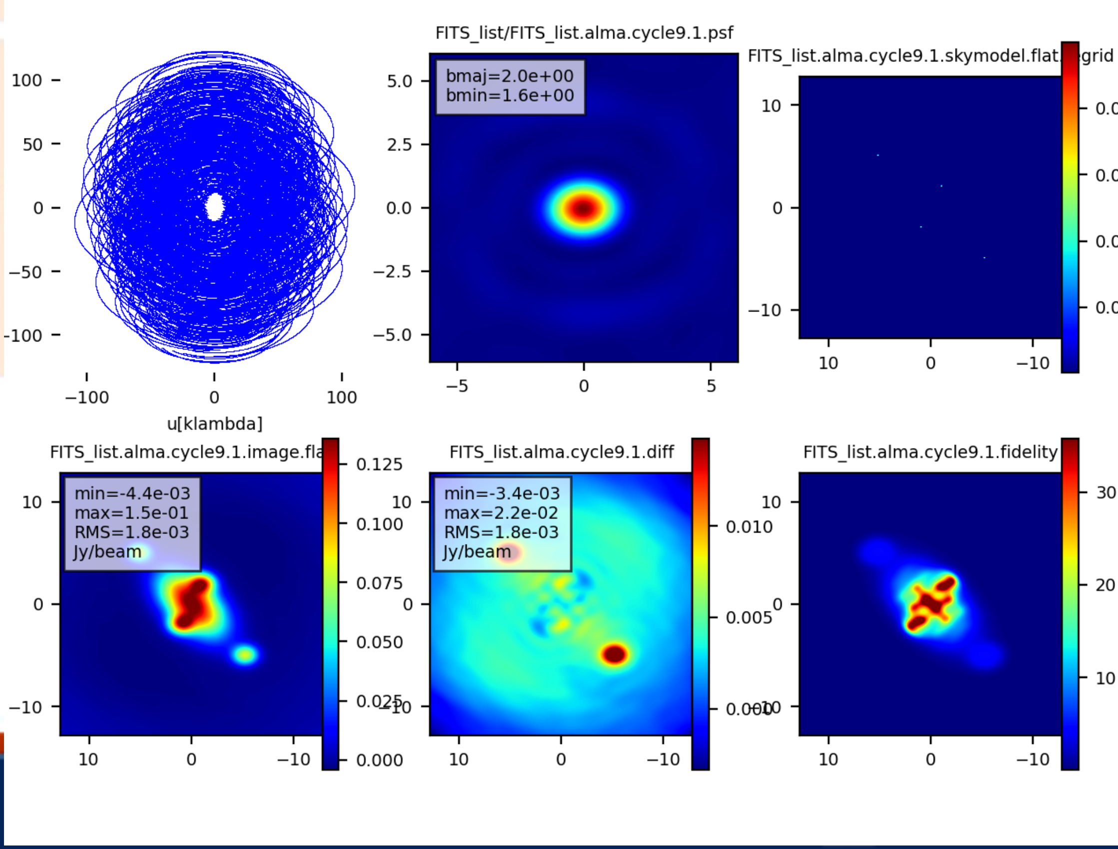
Simulating w. Component List

CASA Guides:
https://casaguides.nrao.edu/

```
# In CASA
default("simobserve")
project = "FITS_list"
skymodel = "Gaussian.fits"
inwidth = "1GHz"
complist = 'point.cl'
compwidth = '1GHz'
direction = "J2000 10h00m00.0s -30d00m00.0s"
obsmode = "int"
antennalist = 'alma.cycle9.1.cfg'
totaltime = "28800s"
mapsize = "10arcsec"
thermalnoise = ''
simobserve()
```

```
default("simanalyze")
project = "FITS_list"
vis="FITS_list.alma.cycle9.1.ms"
imsize = [256,256]
imdirection = "J2000 10h00m00.0s -30d00m00.0s"
cell = '0.1arcsec'
niter = 5000
threshold = '10.0mJy/beam'
analyze = True
simanalyze()
```

Files also saved to disk →



Advanced simulation (Notebook w. modular CASA)

CASA Docs:
https://casadocs.readthedocs.io/

CASA

Search docs

Simulation in CASA

Original Author: rurvashi@aoc.nrao.edu

Description

Get creative with data sets to be used for test scripts and characterization of numerical features/changes. This notebook goes beneath the simobserve task and illustrates simple ways in which developers and test writers can make full use of the flexibility offered by our tools and the imager framework. It also exercises some usage modes that our users regularly encounter and exposes some quirks of our scripting interface(s). Rudimentary image and data display routines are included below.

Topics Covered below

```
- Install CASA 6 and Import required libraries

- Make an empty MS with the desired sub-structure
- Make a true sky model
- Predict visibilities onto the DATA column of the MS
- Add noise and other errors

- A few example use cases
    - Image one channel
    - Cube imaging with a spectral line
    - Continuum wideband imaging with model subtraction
    - Self-calibration and imaging

 - Ideas for CASA developers and test writers to do beyond these examples.
```

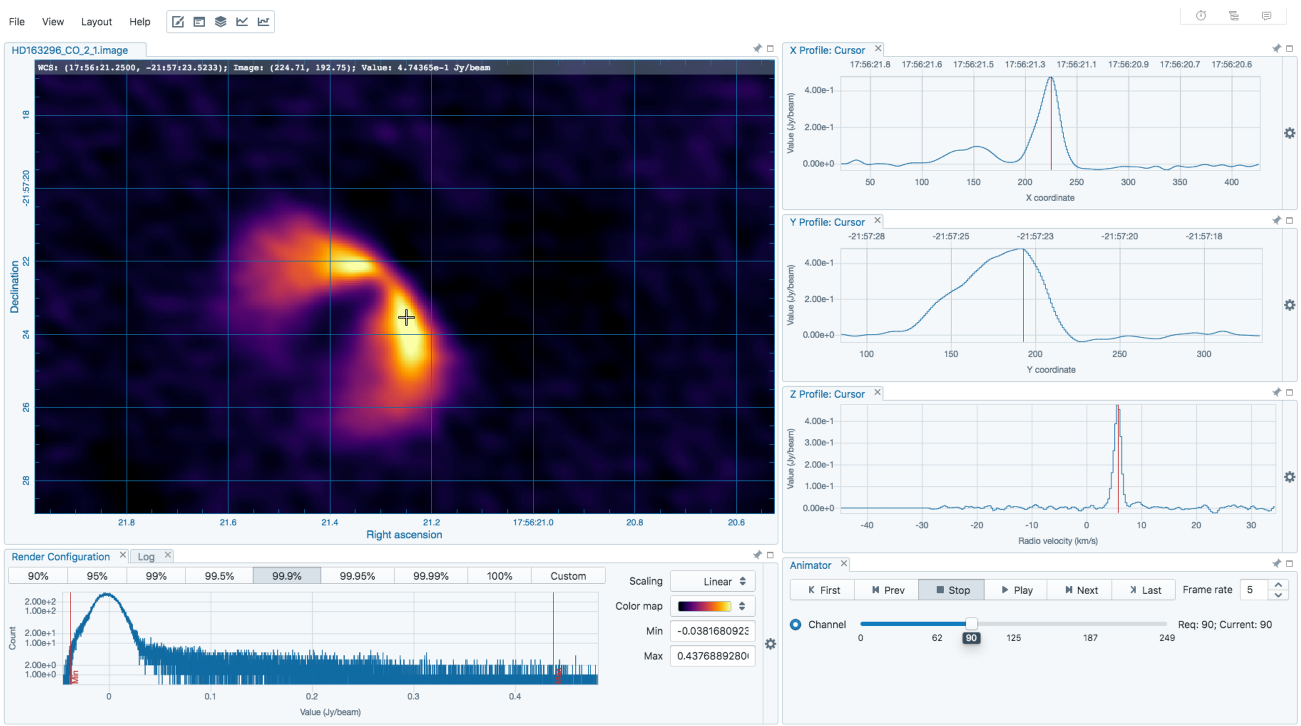
Installation

Option 1 : Install local python3

```
export PPY=`which python3`
virtualenv -p $PPY --setuptools ./local_python3
./local_python3/bin/pip install --upgrade pip
./local_python3/bin/pip install --upgrade numpy matplotlib ipython astropy
./local_python3/bin/pip install --extra-index-url https://casa-pip.nrao.edu/repository/pypi-group/simple casatools
./local_python3/bin/pip install --extra-index-url https://casa-pip.nrao.edu/repository/pypi-group/simple casatasks
./local_python3/bin/pip3 install jupyter
```

Visualizing ALMA simulations (and data)

Cube Analysis and Rendering Tool for Astronomy



https://cartavis.github.io/

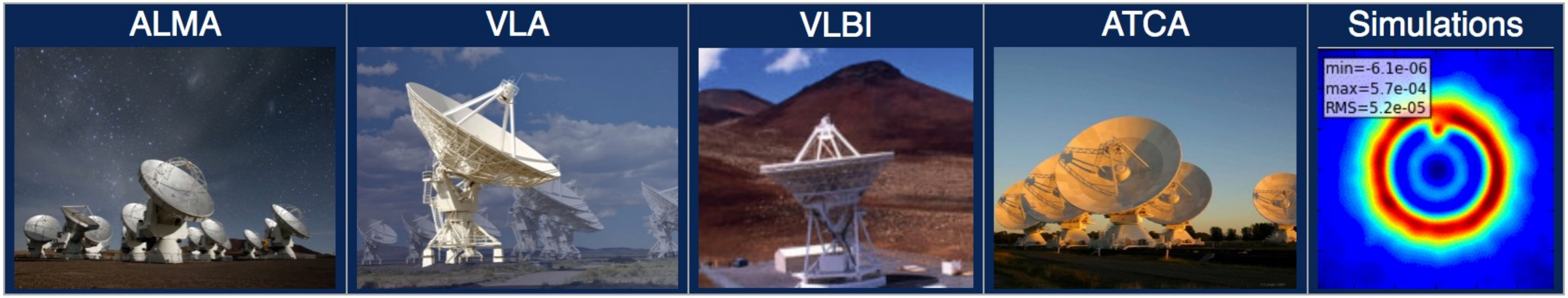
Consortium:
ASIAA
IDIA
NRAO
Univ. Alberta

CARTA = recommended alternative for CASA Viewer!

Information Summary

More information: CASA Guides

Tutorials / examples



More information: CASA Docs (code documentation)

CASA

Search docs

Open in Colab: https://colab.research.google.com/github/casangi/casadocs/blob/3a0ffee/docs/notebooks/simulation.ipynb

Open in Colab

Simulations

The capability of simulating observations and data sets from VLA, ALMA, and other existing and future observatories is an important use-case for CASA. This not only allows the user to get an idea of the capabilities of these instruments for doing science, but also provides benchmarks for the performance and utility of the software to process "realistic" data sets (with atmospheric and instrumental effects). Simulations can also be used to tune parameters of the data reduction and therefore help to optimize the process. CASA can calculate visibilities (create a MeasurementSet) for any interferometric array, and calculate and apply calibration tables representing some of the most important corrupting effects.

Tasks available for simulating observations are:

- **simobserve** - simulate and create custom synthetic MeasurementSets for an interferometric or total power observation
- **simanalyze** - image and analyze simulated data set, including diagnostic images and plots
- **simalma** - simulate an ALMA observation including multiple configurations of the 12-m interferometric array, the 7-m ACA, and total power measurements by streamlining the capabilities of both **simobserve** and **simanalyze**

Inside the Toolkit: The simulator methods are in the **simulator** tool **sm**. Many of the other CASA tools are helpful when constructing and analyzing simulations. Following general CASA practice, the greatest flexibility and functionality is available in the Toolkit, and the most commonly used procedures are bundled for convenience into the tasks.

Utility functions: The **simutil** python class contains numerous utility methods which can be used to facilitate simulations, especially when using the Toolkit.

Simulating interferometric observations using the **simobserve** and **simanalyze** tasks proceeds in the following steps:

1. Make a model image or component list. The model is a representation of the sky brightness distribution that you would like to simulate observing (details on model specification in the **simobserve** documentation).
2. Use the **simobserve** task to create a MeasurementSet (uv data) that would be measured by a telescope observing the specified input model

More information: CASA Reference Paper

Publications of the Astronomical Society of the Pacific, 134:114501 (17pp), 2022 November https://doi.org/10.1088/1538-3873/ac9642

OPEN ACCESS

CrossMark

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[4,8], 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[2,7], 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 Radio Astronomy Observatory, P.O. Box O, Socorro, NM 87801, USA
[3] European Southern Observatory, Karl Schwarzschild Strasse 2, D-85748 Garching, Germany
[4] National Radio Astronomy Observatory, 520 Edgemont Road, Charlottesville, VA 22903, USA; casa-feedback@nrao.edu, bemonts@nrao.edu
[5] National Astronomical Observatory of Japan, 2-21-1 Osawa, Mitaka, Tokyo 181-8588, Japan
[6] Joint Institute for VLBI ERIC, Oude Hoogeveensedijk 4, 7991 PD Dwingeloo, The Netherlands
[7] Inter-University Institute for Data Intensive Astronomy, University of Cape Town, Rondebosch, Cape Town, 7701, South Africa
Received 2022 June 14; accepted 2022 September 27; published 2022 November 15

Abstract

CASA, the Common Astronomy Software Applications, is the primary data processing software for the Atacama Large Millimeter/submillimeter Array (ALMA) and the Karl G. Jansky Very Large Array (VLA), and is frequently used also for other radio telescopes. The CASA software can handle data from single-dish, aperture-synthesis, and Very Long Baseline Interferometery (VLBI) telescopes. One of its core functionalities is to support the calibration and imaging pipelines for ALMA, VLA, VLA Sky Survey, and the Nobeyama 45 m telescope. This paper presents a high-level overview of the basic structure of the CASA software, as well as procedures for calibrating and imaging astronomical radio data in CASA. CASA is being developed by an international consortium of scientists and software engineers based at the National Radio Astronomy Observatory (NRAO), the European Southern Observatory, the National Astronomical Observatory of Japan, and the Joint Institute for VLBI European Research Infrastructure Consortium (JIV-ERIC), under the guidance of NRAO.

More information: Helpdesk & CASA contact e-mail

ALMA Helpdesk: *https://help.almascience.org/*

- ALMA Regional Centers (NAASC)
- One-on-one help

CASA contact: *casa-feedback@nrao.edu*

- Direct contact CASA Team
- Feedback: bugs, feature requests, general comments (questions)
- Best-effort (not a Helpdesk)

Questions?