

Introduction to CASA

J. Ott, C. Brogan, S. Myers

CASA

COMMON
ASTRONOMY
SOFTWARE
APPLICATIONS

CASA (Common Astronomy Software Applications)

- CASA is the offline data reduction package for ALMA and EVLA
- Current version: 3.0.1:
 - New releases about every 6 months.
 - “release”, “test” and “stable” versions available at NRAO
 - > **casapy** – latest release: underwent lots of testing, updated documentation
 - > **casapy-test** – cutting edge capabilities, no documentation, bugs
 - > **casapy-stable** – less bugs but also less features, could be a release
- For download: my.nrao.edu Linux, Mac OS X

Outline

- IPython & Python
- CASA help
- CASA task interface
- MS and data selection
- Documentation

CASA Interface

• IPython

- shell access
- autoparenthesis (autocall)
- command history
- session logging
 - ipython.log – ipython command history
 - casapy.log – casa messages
- numbered input/output
- history/searching

• Python

- Cookbook Appendix D

Python Pointers

- to run a .py script:
 - `execfile('<scriptname>')`
 - example: `execfile('ngc5921_demo.py')`
- indentation matters!
 - be careful when doing cut-and-paste to Python
 - cut a few (4-6) lines at a time
- Python counts from 0 to n-1!
- variables are global when using task interface
- tasknames are objects (not variables)

Tasks and tools in CASA

- Tasks – high-level functionality
 - function call or parameter handling interface
 - these are what you should use in tutorial
- Tools – complete functionality
 - `tool.method` calls, used by tasks
 - sometimes shown in tutorial scripts

Key Tasks

- To see list of tasks organized by type:

>tasklist

```
Default
New Info : Customize Bookmarks Close :
CASA <2>: tasklist
-----> tasklist()
Available tasks, organized by category (experimental tasks in parenthesis):

Import/Export      Information      Data Editing      Display/Plotting
-----
importvla          imhead          concat            clearplot
importfits         imstat          fixvis            plotants
importuvfits       listcal         flagautocorr     plotcal
exportfits         listhistory     flagdata          plotms
exportuvfits       listobs         flagmanager       plotxy
(importasdm)       listvis         plotms            viewer
(importgmrt)       vishead        plotxy            (viewerconnection)
visstat

Data Manipulation  Calibration      Imaging            Modelling
-----
concat            accum           clean             setjy
cvel              applycal       deconvolve        uvcontsub
fixvis            bandpass       feather           uvmodelfit
hanningsmooth    blcal          ft                uvsub
split             calstat        makemask          (uvcontsub2)
uvcontsub         clearcal       (autoclean)
uvsub             cvel           (boxit)
(uvcontsub2)     fluxscale
(msmoments)      fixvis
                  gaincal
                  gencal
                  listcal
                  polcal
                  setjy
                  smoothcal
                  (fringecal)
                  (peel)

Image Analysis     Simulation      Utilities           Single Dish
-----
imcontsub         simdata        browsetable       (after running asap_init())
imhead            (simdata2)    casalogger
imfit             clearplot
immath            clearstat
immoments         csvclean
imregrid          filecatalog
imsmooth          find
imstat            help par.parameter
imval             help task
(specfit)         rmtables
                  startup
                  taskhelp
                  tasklist
                  toolhelp

sdaverage
sdbaseline
sdcal
sdcoadd
sdffit
sdflag
sdimaging
sdimprocess
sdlist
sdmath
sdplot
sdsave
sdscale
sdsmooth
sdstat
sdtpimaging
(sdsim)
(msmoments)

User defined tasks
-----
CASA <3>: █
```

CASA T

Task Execution

- two ways to invoke:
 - call from Python as functions with arguments
 - `taskname(arg1=val1, arg2=val2, ...)`
 - unspecified parameters will be defaulted (globals not used)
 - use standard tasking interface
 - use global variables for task parameters
- see Chapter 1.3 in Cookbook

Task Interface

- standard tasking interface
 - use parameters set as global Python variables
 - `set <param> = <value>` (e.g. `vis = 'ngc5921.demo.ms'`)
 - parameter manipulation commands
 - using inp , default , saveinputs , tget , tput
 - execute
 - `<taskname> or go` (e.g. `clean()`)
 - return values
 - some tasks return Python dictionaries, e.g. `myval=imval()`

Task Interface

- examine task parameters with inp :

```
IPy:Jupiter
CASA <1>: default('clean')
CASA <2>: inp('clean')
# clean :: Deconvolve an image with selected algorithm
vis                =          ''          # name of input visibility file
imagename          =          ''          # Pre-name of output images
field              =          ''          # Field Name
spw                =          ''          # Spectral windows;channels: '' is all
selectdata         =          False       # Other data selection parameters
mode               =          'mfs'       # Type of selection (mfs, channel, velocity, frequency)
niter              =          500         # Maximum number of iterations
gain               =          0.1         # Loop gain for cleaning
threshold          =          '0.0mJy'   # Flux level to stop cleaning. Must include units
psfmode            =          'clark'     # method of PSF calculation to use during minor cycles
imagermode         =          ''         # Use csclean or mosaic. If '', use psfmode
multiscale         =          []         # set deconvolution scales (pixels), default: multiscale=[] (standard CLEAN)
interactive        =          False       # use interactive clean (with GUI viewer)
mask               =          []         # cleanbox(es), mask image(s), and/or region(s) used in cleaning
imsize             =          [256, 256]  # x and y image size in pixels, symmetric for single value
cell               =          ['1.0arcsec', '1.0arcsec'] # x and y cell size, default unit arcsec
phasecenter        =          ''         # Image phase center; position or field index
restfreq           =          ''         # rest frequency to assign to image (see help)
stokes             =          'I'        # Stokes params to image (eg I,IV, QU,IQUV)
weighting          =          'natural'   # Weighting to apply to visibilities
uvtaper            =          False       # Apply additional uv tapering of visibilities.
modelimage         =          ''         # Name of model image(s) to initialize cleaning
restoringbeam      =          []         # Output Gaussian restoring beam for CLEAN image
pbcor              =          False       # Output primary beam-corrected image
minpb              =          0.1        # Minimum PB level to use
async              =          False       # If true the taskname must be started using clean(...)
CASA <3>: █
```

Expandable Parameters

- boldface parameter are expandable

IPy:Jupiter

```
CASA <3>: tget('clean')
```

```
Restored parameters from file clean.last
```

```
CASA <4>: inp()
```

```
# clean :: Deconvolve an image with selected algorithm
```

```
vis                = 'ngc5921.usecase.ms.contsub' # name of input visibility file
imagename          = 'ngc5921.usecase.clean' # Pre-name of output images
field              = '0' # Field Name
spw                = '' # Spectral windows;channels: '' is all
selectdata       = False # Other data selection parameters
mode             = 'channel' # Type of selection (mfs, channel, velocity, frequency)
  nchan            = 46 # Number of channels (planes) in output image
  start            = 5 # first input channel to use
  width            = 1 # Number of input channels to average

niter              = 6000 # Maximum number of iterations
gain               = 0.1 # Loop gain for cleaning
threshold          = 8.0 # Flux level to stop cleaning. Must include units
psfmode           = 'clark' # method of PSF calculation to use during minor cycles
imagermode       = '' # Use csclean or mosaic. If '', use psfmode
multiscale      = [] # set deconvolution scales (pixels), default: multiscale=[] (standard CLEAN)
interactive     = False # use interactive clean (with GUI viewer)
mask               = [108, 108, 148, 148] # cleanbox(es), mask image(s), and/or region(s) used in cleaning
imsize             = [256, 256] # x and y image size in pixels, symmetric for single value
cell               = [15.0, 15.0] # x and y cell size, default unit arcsec
phasecenter        = '' # Image phase center; position or field index
restfreq           = '' # rest frequency to assign to image (see help)
stokes             = 'I' # Stokes params to image (eg I,IV, QU,IQUV)
weighting       = 'briggs' # Weighting to apply to visibilities
  robust           = 0.5 # Briggs robustness parameter
  npixels          = 0 # number of pixels to determine uv-cell size 0=> field of view

uvtaper         = False # Apply additional uv tapering of visibilities.
modelimage         = '' # Name of model image(s) to initialize cleaning
```

Parameter Checking

- sanity checks of parameters in `inp` :

```
IPy:Jupiter
CASA <5>: psfmode='hogwarts'
CASA <6>: inp()
# clean :: Deconvolve an image with selected data
vis = 'ngc5921.usecase.ms.corr' # visibility file
imagename = 'ngc5921.usecase.clean' # cleaned image name
field = '0' # field name
spw = '' # spectral window
selectdata = False # select data by channel, velocity, frequency
mode = 'channel' # mode of operation
  nchan = 46 # Number of channels (planes) in output image
  start = 5 # first input channel to use
  width = 1 # Number of input channels to average

niter = 6000 # Maximum number of iterations
gain = 0.1 # Loop gain for cleaning
threshold = 8.0 # Flux level to stop cleaning. Must include units
psfmode = 'hogwarts' # method of PSF calculation to use during minor cycles
imagermode = '' # Use csclean or mosaic. If '', use psfmode
multiscale = [] # set deconvolution scales (pixels), default: multiscale=[] (standard CLEAN)
interactive = False # use interactive clean (with GUI viewer)
mask = [108, 108, 148, 148] # cleanbox(es), mask image(s), and/or region(s) used in cleaning
imsize = [256, 256] # x and y image size in pixels, symmetric for single value
cell = [15.0, 15.0] # x and y cell size, default unit arcsec
phasecenter = '' # Image phase center: position or field index
restfreq = '' # rest frequency to assign to image (see help)
stokes = 'I' # Stokes params to image (eg I,IV, QU,IQUV)
weighting = 'briggs' # Weighting to apply to visibilities
  robust = 0.5 # Briggs robustness parameter
  npixels = 0 # number of pixels to determine uv-cell size 0=> field of view

uvtaper = False # Apply additional uv tapering of visibilities.
modelimage = '' # Name of model image(s) to initialize cleaning
restoringbeam = [] # Output Gaussian restoring beam for CLEAN image
```

erroneous values in red

Help on Tasks

- In-line help:
 - `tasklist` - Task list organized by category
 - `taskhelp` - One line summary of available tasks
 - `help taskname` - Full help for task
 - `toolhelp` - One line summary of available tools
 - `help par.parametername` - Full help for parameter name

Help on Tasks

• In-line help:

>help 'clean' OR >pdoc clean

```
IPy:Jupyter
CASA <7>: help('clean')
Help on module clean:

NAME
    clean

FILE
    /usr/lib/casapy/20.0.5444test-001/lib/python2.5/clean.py

DESCRIPTION
    # This file was generated using xslt from its XML file
    #
    # Copyright 2007, Associated Universities Inc., Washington DC
    #

FUNCTIONS
    clean_imp(vis=None, imagename=None, field=None, spw=None, selectdata=
gain=None, threshold=None, psfmode=None, imagermode=None, ftmachine=None
one, mask=None, nchan=None, start=None, width=None, imsize=None, cell=Non
er=None, outertaper=None, innertaper=None, modelimage=None, restoringbeam
r=None, cyclespeedup=None, async=None)
        Deconvolve an image with selected algorithm

        The main clean deconvolution task. It contains many functio

        1) Make 'dirty' image and 'dirty' beam (psf)
        2) Multi-frequency-continuum images or spectral channel im
        3) Full Stokes imaging
        4) Mosaicking of several pointings
        5) Multi-scale cleaning
        6) Interactive clean boxing
        7) Initial starting model

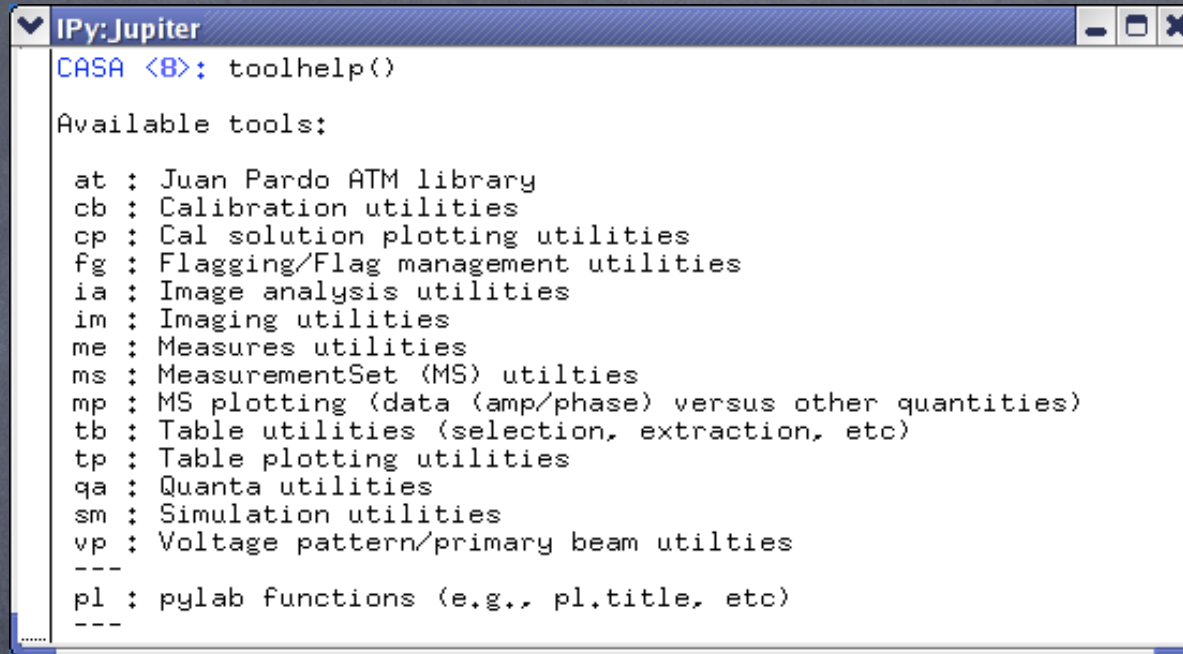
    vis -- Name of input visibility file
           default: none; example: vis='ngc5921.ms'
    imagename -- Pre-name of output images:
                 default: none; example: imagename='m2'
                 output images are:
                 m2.image: cleaned and restored image
                    With or without primary beam correction
                 m2.psf: point-spread function (dirty beam)
                 m2.flux: relative sky sensitivity over field
                 m2.model: image of clean components
                 m2.residual: image of residuals
```

Tools in CASA

- CASA Toolkit underneath tasks
 - core AIPS++ code (mostly in C++)
- tools are functions
 - call from casapy as `<tool>.<method>()`
 - default tool objects are pre-constructed
 - e.g. imager (im) , calibrator (cb), ms (ms) , etc. (see toolhelp)

CASA Tool List

- list of default tools from toolhelp :



```
IPy:Jupyter
CASA <8>: toolhelp()

Available tools:

at : Juan Pardo ATM library
cb : Calibration utilities
cp : Cal solution plotting utilities
fg : Flagging/Flag management utilities
ia : Image analysis utilities
im : Imaging utilities
me : Measures utilities
ms : MeasurementSet (MS) utilities
mp : MS plotting (data (amp/phase) versus other quantities)
tb : Table utilities (selection, extraction, etc)
tp : Table plotting utilities
qa : Quanta utilities
sm : Simulation utilities
vp : Voltage pattern/primary beam utilities
---
pl : pylab functions (e.g., pl.title, etc)
---
```

- tools described in the CASA Toolkit Reference:

- <http://casa.nrao.edu/docs/casaref/CasaRef.html>

The Measurement Set

- the MS is a directory on disk
 - the MAIN table in `table.*` files
 - also contains sub-tables
 - e.g. FIELD, SOURCE, ANTENNA, etc.
 - sub-tables are sub-directories
 - to copy must `cp -rf` to get contents
 - Best to remove ms with `rmtables('filename')`
 - WARNING: moving a MS can break cal-table dependencies

Example MS

- Example: `ls`
`ngc5921.usecase.ms`

```
smyers@olorin ~/CASA/Test $ ls ngc5921.usecase.ms
ANTENNA                POLARIZATION          table.f1              table.f3_TSM1        table.f8
DATA_DESCRIPTION       PROCESSOR             table.f10            table.f4             table.f8_TSM1
FEED                   SORTED_TABLE         table.f10_TSM1       table.f5             table.f9
FIELD                  SOURCE                table.f11            table.f5_TSM1        table.f9_TSM1
FLAG_CMD               SPECTRAL_WINDOW     table.f11_TSM1       table.f6             table.info
HISTORY                STATE                table.f2             table.f6_TSM0        table.lock
OBSERVATION            table.dat            table.f2_TSM1        table.f7
POINTING               table.f0             table.f3             table.f7_TSM1
```

- `ls ngc5921.usecase.ms/FIELD`

```
smyers@olorin ~/CASA/Test $ ls ngc5921.usecase.ms/FIELD
table.dat      table.f0      table.f0i      table.info      table.lock
```

MAIN Table Contents

Example using task browsetable:

Table Browser

File Edit View Tools Export Help

ngc5921.usecase.ms

	UVW	FLAG	LAG_CATEGOR	WEIGHT	SIGMA	ANTENNA1	ANTENNA2	ARRAY_ID	DATA_DESC_ID	EXPOSURE
0	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	1	1	0	0	30
1	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	27	27	0	0	30
2	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	7	7	0	0	30
3	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	2	2	0	0	30
4	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	11	11	0	0	30
5	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	17	17	0	0	30
6	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	9	9	0	0	30
7	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	19	19	0	0	30
8	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	20	20	0	0	30
9	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	18	18	0	0	30
10	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	3	3	0	0	30
11	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	15	15	0	0	30
12	[0, 0, 0]	[2, 63] Boolean	[0, 0, 0] Boolean	[23814, 23814]	[0.0514344, 0....	21	21	0	0	30

Restore Columns Resize Headers

PAGE NAVIGATION First << [1 / 23] >> Last 1 Go Loading 1000 rows.

Browsing table: ngc5921.usecase.ms

Data Selection Example

- standard selection parameters

- e.g. for task gaincal:

```
CASA <14>: inp
-----> inp()
# gaincal :: Determine temporal gains from calibrator observations:

vis                = 'ngc5921.ms'      # Name of input visibility file
caltable           = 'ngc5921.gcal'    # Name of output calibration table
field              = '0,1'            # field names or index of calibrators ''=>all
spw                = '0:2~56'        # spectral window:channels: ''=>all
selectdata       = True              # Other data selection parameters
  timerange        = ''              # time range: ''=>all
  uvrange          = ''              # uv range''=all
  antenna          = ''              # antenna/baselines: ''=>all
  scan             = ''              # scan numbers
  msselect         = ''              # Optional data selection (Specialized. but see help)
```

- field and spw common standard selections
 - expandable selectdata with other selections as sub-parameters

Data Selection Syntax

- see Chapter 2.5 of Cookbook
 - field – string with source name or field ID
 - can use '*' as wildcard, first checks for name, then ID
 - example: field = '1331+305' ; field = '3C*' ; field = '0,1,4~5'
 - spw – string with specwindow ID plus channels
 - use ':' as separator of spw from optional channelization
 - use '^' as separator of channels from step/width
 - example: spw = '0~2' ; spw = '1:10~30' ; spw = '2~5:5~54^5'

Selection Syntax

- see Chapter 2.5 of Cookbook
 - antenna - string with antenna name or ID
 - first check for name, then ID (beware VLA name 1-27, ID 0-26)
 - example: antenna = '1~5,11' ; antenna = 'VA*'
 - timerange - string with date/time range
 - specify 'T0~T1' , missing parts of T1 default to T0, can give 'T0+dT'
 - example: timerange = '2007/10/16/01:00:00~06:30:00'

Getting User Support

- First stop:

- CASA Home: <http://casa.nrao.edu>
- Cookbook, and “help” within CASA
- CASAguides.nrao.edu
- User’s forum in the future

- CASA Helpdesk & Support

- “Helpdesk” at <http://my.nrao.edu>
- You will need to register first
 - Submit questions, suggestions, bugs

CASA Documentation

- CASA Analysis cookbook:
 - http://casa.nrao.edu/Doc/Cookbook/casa_cookbook.pdf
- CASA User Reference Manual:
 - <http://casa.nrao.edu/docs/casaref/CasaRef.html>
- CASAguides Wiki:
 - <http://casaguides.nrao.edu>
- Python:
 - <http://python.org/doc> (e.g., see Tutorial for novices)
- IPython:
 - <http://ipython.scipy.org/moin/Documentation>