Simulations and Imaging in CASA

Statia Cook
Content from M. MacGregor!

Atacama Large Millimeter/submillimeter Array
Karl G. Jansky Very Large Array

What Is CASA?

CASA, the Common Astronomy Software Applications package, is for post-processing data from the next generation of radio astronomical telescopes such as ALMA and VLA.

For you: CASA is your go-to tool for simulations, data analysis, and imaging

Important note: CASA has an iPython interface

CASA
Common Astronomy Software Applications

NRAO

Where Do You Get CASA?

If you haven't downloaded in advance, you need it to participate in the hands-on session!

https://casa.nrao.edu/casa_obtaining.shtml

Download most recent version 5.1.2

Some Helpful Resources

CASA has lots of online guides to help get you started:

http://casaguides.nrao.edu/index.php?title=Main_Page

The full reference/cookbook is available here:

http://casa.nrao.edu/Doc/Cookbook/casa_cookbook.pdf

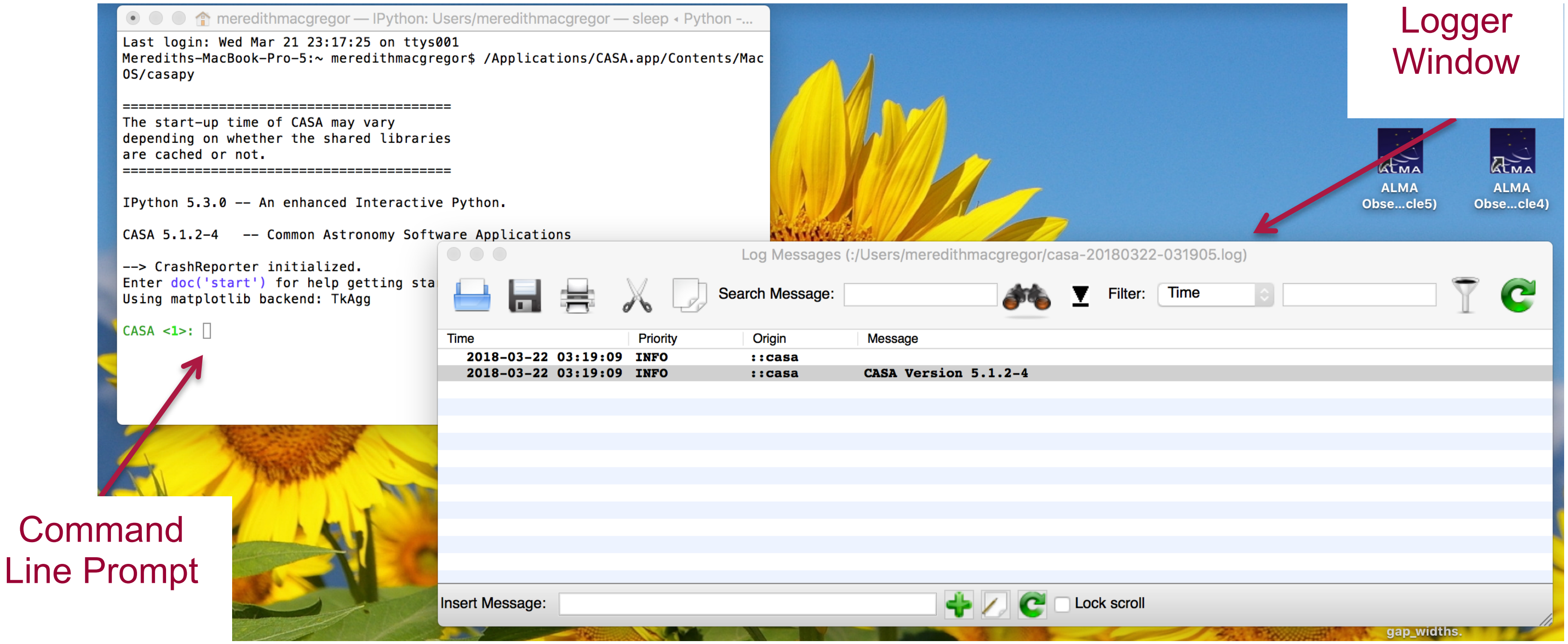
ALMA-specific tutorials are available here:

http://casaguides.nrao.edu/index.php?title=ALMAguides

NRAO

Starting CASA

When CASA first opens, you will see a Python command line prompt and a separate logger window



Pay attention to the logger window! Most tasks write important information to this window. All logger messages are also saved into a file labeled 'casapy.log'

Calling CASA Tasks

If you want to know what a task's parameters are, type:

tget <taskname>

```
CASA <2>: tget clean
--------> tget(clean)
Restored parameters from file clean.last

CASA <3>:
```

Then, type:

inp

This will bring up a list of all possible parameters for that task. It will also retrieve any previously used parameter values. If you want to restore the default parameters, instead use:

default('<taskname>')

To run a task type:

go

You can always get help on a task by typing:

help <taskname>

Simulating Observations

Running a simulation can help convince the TAC that your proposed observations are feasible.

It is also a great way to check that your uv-coverage/spatial scale needs are met.

Steps for simulating observations:

1. Use the ALMA sensitivity calculator to determine the necessary observing time for your science goals
2. Generate simulated visibilities using the 'simobserve' task in CASA (takes FITS input)
3. Image, analyze, and evaluate the resulting visibilities

Repeat for different antenna configurations, observing times, etc.

Simulating Observations

Sensitivity calculator available here: https://almascience.eso.org/proposing/sensitivity-calculator

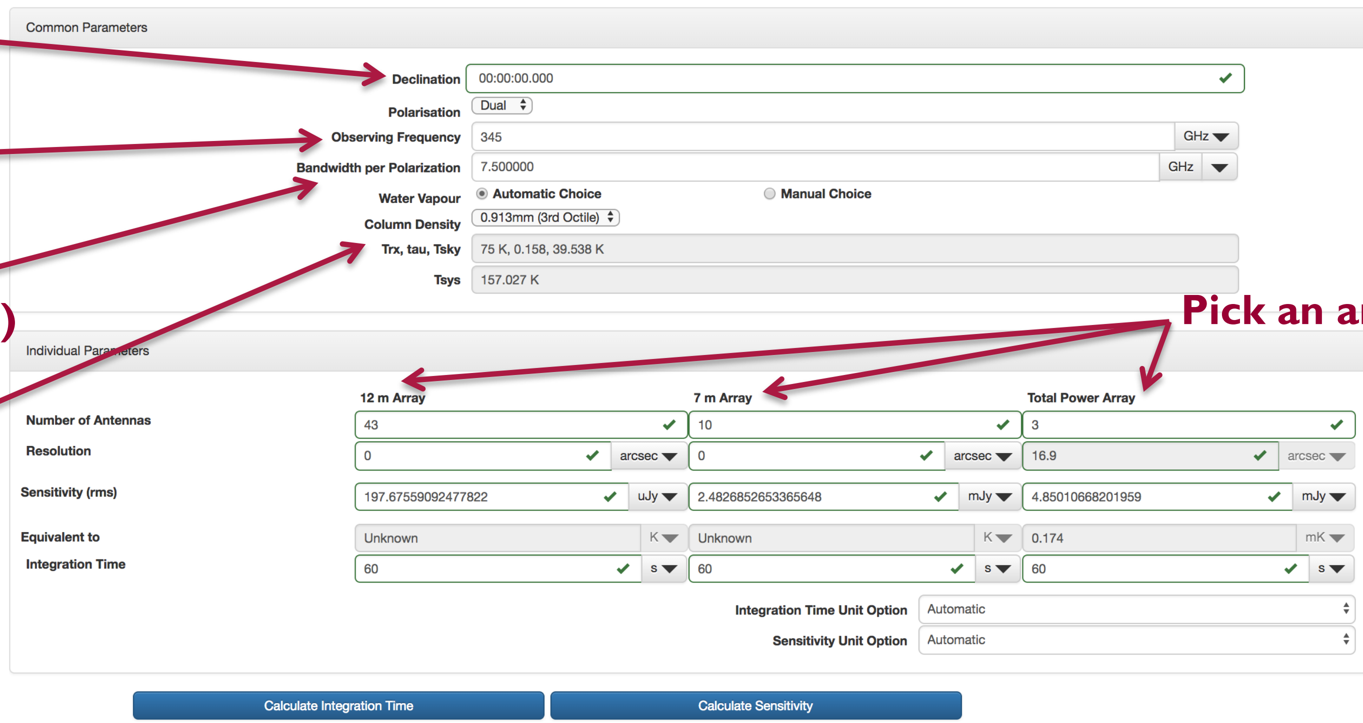
Source DEC

Frequency

Bandwidth (7.5 GHz default)

PWV (automatically chosen)

Pick an array



You can also access the sensitivity calculator through the OT!

Simulating Observations

All of the 'simobserve' parameters in CASA

'simobserve' takes FITS images as inputs and generates simulated visibilities for given antenna configurations, observing time, and PWV

A CASA guide on simulating observations is available here: https://casaguides.nrao.edu/index.php/Simulating_Observations_in_CASA_5.1

```
#  simobserve :: visibility simulation task
project             =  'hd10647'        #  root prefix for output file names
skymodel            = 'hd10647_model.fits' #  model image to observe
     inbright       =          ''       #  scale surface brightness of brightest pixel
                                        #   e.g. "1.2Jy/pixel"
     indirection    =          ''       #  set new direction e.g. "J2000 19h00m00
                                        #   -40d00m00"
     incell         =          ''       #  set new cell/pixel size e.g. "0.1arcsec"
     incenter       =          ''       #  set new frequency of center channel e.g.
                                        #   "89GHz" (required even for 2D model)
     inwidth        =          ''       #  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        =    ['', '']       #  angular size of map or "" to cover model
     maptype        =      'ALMA'       #  hexagonal, square (raster), ALMA, etc
     pointingspacing =         ''       #  spacing in between pointings or "0.25PB" or ""
                                        #   for ALMA default INT=lambda/D/sqrt(3),
                                        #   SD=lambda/D/3

obsmode             =       'int'       #  observation mode to simulate
                                        #   [int(interferometer)|sd(singledish)|""(none)]
     antennalist    = 'alma.out10.cfg'  #  interferometer antenna position file
     refdate        = '2014/05/21'      #  date of observation - not critical unless
                                        #   concatting simulations
     hourangle      =   'transit'       #  hour angle of observation center e.g.
                                        #   "-3:00:00", "5h", "-4.5" (a number without
                                        #   units will be interpreted as hours), or
                                        #   "transit"
     totaltime      =     '7200s'       #  total time of observation or number of
                                        #   repetitions
     caldirection   =          ''       #  pt source calibrator [experimental]
     calflux        =       '1Jy'

outframe            =      'LSRK'       #  spectral frame of MS to create
thermalnoise        =  'tsys-atm'       #  add thermal noise: [tsys-atm|tsys-manual|""]
     user_pwv       =         0.5       #  Precipitable Water Vapor in mm
     t_ground       =       269.0       #  ambient temperature
     seed           =       11111       #  random number seed

leakage             =         0.0       #  cross polarization (interferometer only)
graphics            =      'both'       #  display graphics at each stage to
                                        #   [screen|file|both|none]

verbose             =       False
overwrite           =        True       #  overwrite files starting with $project
```

Simulating Observations

General Parameters

```
#  simobserve :: visibility simulation task
project             =  'hd10647'           #  root prefix for output file names
skymodel            = 'hd10647_model.fits' #  model image to observe
     inbright       =          ''          #  scale surface brightness of brightest pixel
                                           #   e.g. "1.2Jy/pixel"
     indirection    =          ''          #  set new direction e.g. "J2000 19h00m00
                                           #   -40d00m00"
     incell         =          ''          #  set new cell/pixel size e.g. "0.1arcsec"
     incenter       =          ''          #  set new frequency of center channel e.g.
                                           #   "89GHz" (required even for 2D model)
     inwidth        =          ''          #  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        =    ['', '']          #  angular size of map or "" to cover model
     maptype        =      'ALMA'          #  hexagonal, square (raster), ALMA, etc
     pointingspacing =          ''         #  spacing in between pointings or "0.25PB" or ""
```

project : name of folder for simulation output

skymodel: input FITS image for simulation

incenter: center frequency for observations

inwidth: channel width (set to 7.5 GHz for continuum)

setpointings: sets up pointings for simulation, can also set to 'False' and provide a list of pointings (useful for setting up custom mosaics)

A Note on Pointings

Sometimes it can be helpful to define the pointings for your simulation using a pointing file:

Pointing RA

```
mmacgreg cfa0 7> more ptgfile_mosaic_new.txt
J2000    22h57m39.0462   -29d37m20.0530
J2000    22h57m39.6394   -29d37m40.0392
J2000    22h57m38.4529   -29d37m00.0668
J2000    22h57m38.3648   -29d37m12.6263
J2000    22h57m39.1343   -29d37m07.4872
J2000    22h57m38.9581   -29d37m30.9918
J2000    22h57m39.7276   -29d37m27.4797
mmacgreg cfa0 8>
```

Pointing DEC

This is a must if you want to design a custom mosaic

Simulating Observations

Observing Parameters

```
obsmode            =      'int'        #  observation mode to simulate [int(int
                                       #   erferometer)|sd(singledish)|""(none)
                                       #   ]
    antennalist    = 'alma.out10.cfg'  #  interferometer antenna position file
    refdate        = '2014/05/21'      #  date of observation - not critical
                                       #   unless concatting simulations
    hourangle      =  'transit'        #  hour angle of observation center e.g.
                                       #   "-3:00:00", "5h", "-4.5" (a number
                                       #   without units will be interpreted as
                                       #   hours), or "transit"
    totaltime      =    '7200s'        #  total time of observation or number
                                       #   of repetitions
    caldirection   =         ''        #  pt source calibrator [experimental]
    calflux        =      '1Jy'
```

antennalist : sample configuration file, full list of configuration files available here:

https://casaguides.nrao.edu/index.php/Antenna_Configurations_Models_in_CASA

refdate: date for simulation (not critical to change)

totaltime: total time for simulated observations

Simulating Observations

Noise Parameters

```
thermalnoise        = 'tsys-atm'        #  add thermal noise: [tsys-atm|tsys-
                                        #   manual|""]
     user_pwv       =        0.5        #  Precipitable Water Vapor in mm
     t_ground       =      269.0        #  ambient temperature
     seed           =      11111        #  random number seed

leakage             =        0.0        #  cross polarization (interferometer
                                        #   only)
graphics            =     'both'        #  display graphics at each stage to
                                        #   [screen|file|both|none]
verbose             =      False
overwrite           =       True        #  overwrite files starting with
                                        #   $project
```

thermalnoise : thermal noise model to use, 'tsys-atm' uses a model for the ALMA site, 'tsys-manual' allows the user to specify the zenith sky brightness and opacity manually

user_pwv: precipitable water vapor in mm (usually take from sensitivity calculator)

NRAO

Output from 'simobserve'

```
CASA <34>: ls
hd10647.alma.cycle5.3.ms/              hd10647.alma.cycle5.3.simobserve.last
hd10647.alma.cycle5.3.noisy.ms/        hd10647.alma.cycle5.3.skymodel/
hd10647.alma.cycle5.3.observe.png      hd10647.alma.cycle5.3.skymodel.flat/
hd10647.alma.cycle5.3.ptg.txt          hd10647.alma.cycle5.3.skymodel.png
hd10647.alma.cycle5.3.quick.psf/
```

What do you do next?

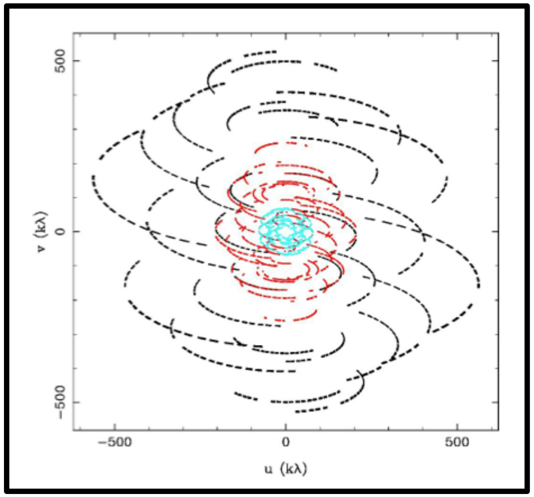
- Use 'tclean' to image the resulting visibilities *–OR–*
- Use 'simanalyze' in CASA, which creates images using 'clean'

Another approach to simulations is to use the 'simalma' task:
https://casaguides.nrao.edu/index.php/Simalma
But, 'simobserve' is more generalized and provides more capability (e.g. multiple configurations, pointings, etc.)

NRAO

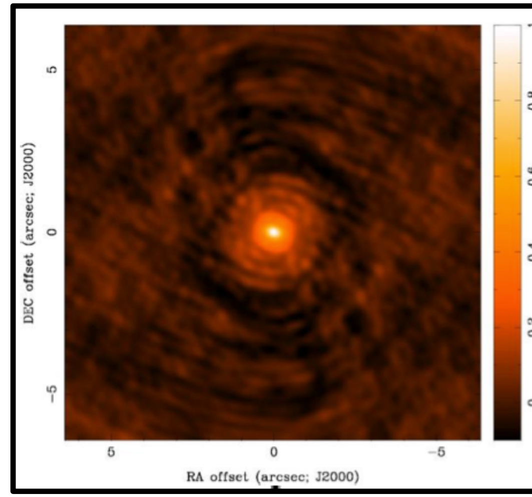
The Dirty Beam

S(u,v)

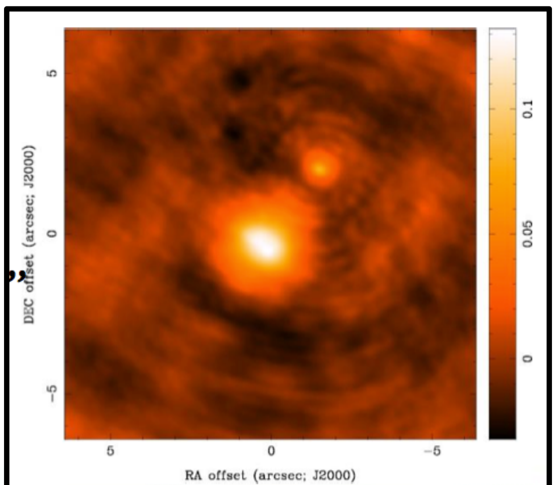


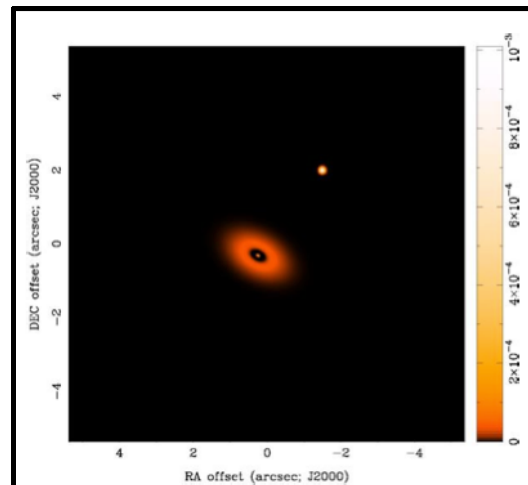
FT

s(l,m)
"Dirty Beam"



* (Convolution)





T_D(l,m)
"Dirty Image"

T(l,m)

Making an Image

Assumption: Image T(l,m) is a collection of point sources

Steps to Clean:

Initialize: residual map to dirty image and empty "clean component list"

1. Start by identifying the highest peak in the residual map as a point source
2. Subtract a fraction of this peak from the residual map using a scaled dirty beam: s(l,m) x gain
3. Add this point source location and amplitude to the "clean component list"
4. Go back to step 1 (complete an iteration) unless stopping criterion reached

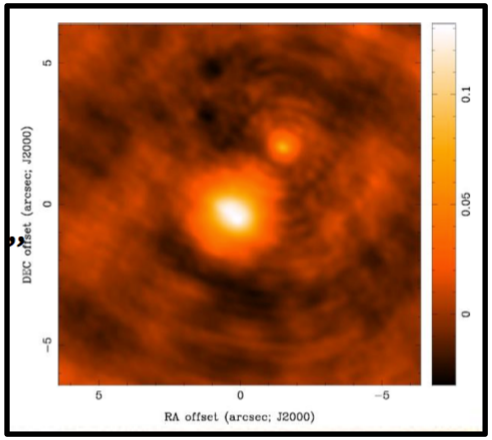
Stopping Criteria? Usually max(residual map) < multiple of rms noise

Making an Image

1. Make a model image with all point sources from the "clean component list"
2. Convolve point sources with an elliptical Gaussian, fit to the main lobe of the dirty beam ("clean beam")
3. Add residual map of noise and source structure below the set threshold

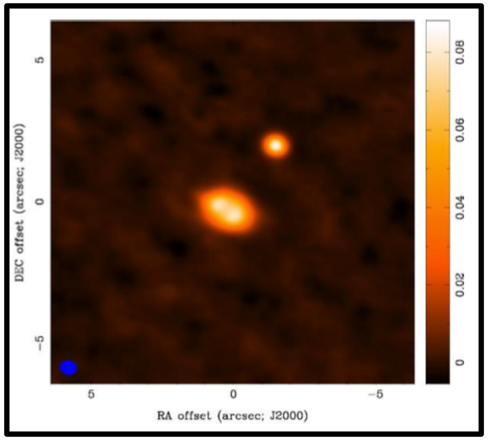
Result: A final "restored image" that is an estimate of the true sky brightness T(l,m)

Units of the restored image are (mostly) Jy per clean beam area = intensity



$T_D(l,m)$
"Dirty Image"

Clean



T(l,m)
"Restored Image"

Imaging in CASA

Imaging capabilities of 'clean' have been refactored and improved in 'tclean' in the current version of CASA

The ALMA pipeline now uses 'tclean' instead of 'clean' for imaging

Major syntax changes are summarized here: https://casaguides.nrao.edu/index.php/TCLEAN_and_ALMA

NRAO

All of the 'tclean' parameters in CASA

```
#  tclean :: Radio Interferometric Image Reconstruction
vis                 =         ''        #  Name of input visibility file(s)
selectdata          =      False        #  Enable data selection parameters
datacolumn          = 'corrected'       #  Data column to image(data,corrected)
imagename           =         ''        #  Pre-name of output images
imsize              =      [100]        #  Number of pixels
cell                = ['1arcsec']       #  Cell size
phasecenter         =         ''        #  Phase center of the image
stokes              =        'I'        #  Stokes Planes to make
projection          =      'SIN'        #  Coordinate projection (SIN, HPX)
startmodel          =         ''        #  Name of starting model image
specmode            =      'mfs'        #  Spectral definition mode
                                        #   (mfs,cube,cubedata)
     reffreq        =         ''        #  Reference frequency

gridder             = 'standard'        #  Gridding options (standard, wproject,
                                        #   widefield, mosaic, awproject)
     vptable        =         ''        #  Name of Voltage Pattern table
     pblimit        =        0.2        #  >PB gain level at which to cut off
                                        #   normalizations

deconvolver         =   'hogbom'        #  Minor cycle algorithm (hogbom,clark,m
                                        #   ultiscale,mtmfs,mem,clarkstokes)
restoration         =       True        #  Do restoration steps (or not)
     restoringbeam  =         []        #  Restoring beam shape to use. Default
                                        #   is the PSF main lobe
     pbcor          =      False        #  Apply PB correction on the output
                                        #   restored image

outlierfile         =         ''        #  Name of outlier-field image
                                        #   definitions
weighting           =  'natural'        #  Weighting scheme
                                        #   (natural,uniform,briggs)
     uvtaper        =         []        #  uv-taper on outer baselines in uv-
                                        #   plane

niter               =          0        #  Maximum number of iterations
usemask             =     'user'        #  Type of mask(s) for deconvolution
                                        #   (user, pb, auto-thresh, auto-
                                        #   thresh2, or auto-multithresh)
     mask           =         ''        #  Mask (a list of image name(s) or
                                        #   region file(s) or region string(s) )
     pbmask         =        0.0        #  primary beam mask

restart             =       True        #  True : Re-use existing images. False
                                        #   : Increment imagename
savemodel           =     'none'        #  Options to save model visibilities
                                        #   (none, virtual, modelcolumn)
calcres             =       True        #  Calculate initial residual image
calcpsf             =       True        #  Calculate PSF
parallel            =      False        #  Run major cycles in parallel
```

Imaging in CASA

General Parameters

```
#  tclean :: Radio Interferometric Image Reconstruction
vis                 =         ''        #  Name of input visibility file(s)
selectdata          =      False        #  Enable data selection parameters
datacolumn          = 'corrected'       #  Data column to image(data,corrected)
imagename           =         ''        #  Pre-name of output images
imsize              =      [100]        #  Number of pixels
cell                = ['1arcsec']       #  Cell size
phasecenter         =         ''        #  Phase center of the image
stokes              =        'I'        #  Stokes Planes to make
projection          =      'SIN'        #  Coordinate projection (SIN, HPX)
startmodel          =         ''        #  Name of starting model image
```

vis : the name of the visibility file

selectdata: allows you to select a chunk of data to image

imagename: what you want your image to be called

imsize : size of image in pixels

cell: size of each pixel in arcsec (make sure you cover the primary beam!)

Imaging in CASA

Key Clean Parameters

```
specmode          =      'mfs'        #  Spectral definition mode
                                      #   (mfs,cube,cubedata)
     reffreq      =         ''        #  Reference frequency

gridder           = 'standard'        #  Gridding options (standard, wproject,
                                      #   widefield, mosaic, awproject)
     vptable      =         ''        #  Name of Voltage Pattern table
     pblimit      =        0.2        #  >PB gain level at which to cut off
                                      #   normalizations

deconvolver       =   'hogbom'        #  Minor cycle algorithm (hogbom,clark,m
                                      #   ultiscale,mtmfs,mem,clarkstokes)
```

specmode: use 'mfs' for continuum images and 'channel/ velocity/frequency' for spectral line imaging

*For line imaging, you will also need to set the dimensions of the cube, rest frequency, velocity frame, and Doppler definition

gridder: 'standard' and 'mosaic' most common for ALMA

deconvolver: allows for different deconvolution options (hogbom, clark, mtmfs, multiscale, clarkstokes)

Imaging in CASA

Weighting and Stopping

```
                                     #   definitions
weighting          =  'natural'      #  Weighting scheme
                                     #   (natural,uniform,briggs)
     uvtaper       =          []     #  uv-taper on outer baselines in uv-
                                     #   plane

niter              =         100     #  Maximum number of iterations
     gain          =         0.1     #  Loop gain
     threshold     =         0.0     #  Stopping threshold
     cycleniter    =          -1     #  Maximum number of minor-cycle
                                     #   iterations
     cyclefactor   =         1.0     #  Scaling on PSF sidelobe level to
                                     #   compute the minor-cycle stopping
                                     #   threshold.
     minpsffraction =       0.05     #  PSF fraction that marks the max depth
                                     #   of cleaning in the minor cycle
     maxpsffraction =        0.8     #  PSF fraction that marks the minimum
                                     #   depth of cleaning in the minor cycle
     interactive   =        True     #  Modify masks and parameters at
                                     #   runtime
```

weighting: natural, uniform or robust

uvtaper: apply Gaussian uv taper to visibilities

niter: number of iterations you want clean to do

threshold: flux level you want clean to stop at

interactive: run clean interactively

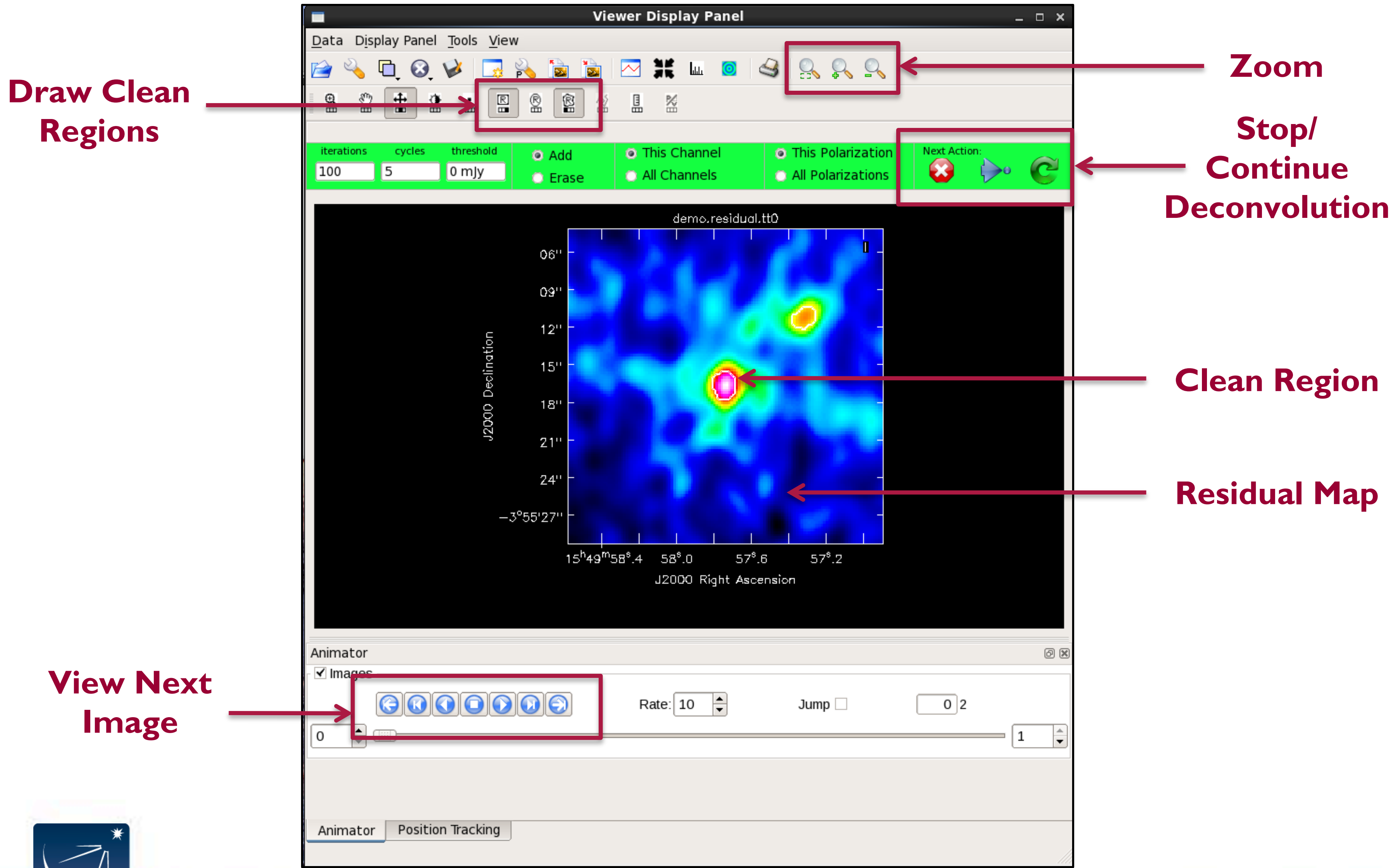
A Note On Weighting

By weighting, you are multiplying your uv distribution, S(u,v), by a weighting function, W(u,v), and changing your dirty beam shape.

	Natural	**Robust/Uniform**	**Taper**
Resolution	Medium	**Higher**	Lower
Sidelobes	Higher	Lower	Depends
Point Source Sensitivity	**Maximum**	Lower	Lower
Extended Source Sensitivity	Medium	Lower	**Higher**

There are trade-offs with all weighting schemes. Make sure to start conservative and adjust to get the best image to achieve your particular science goals!

Running 'tclean'



Export Your Final Image

In order to work with your image outside of CASA, you'll need to export it as a FITS file using the **'exportfits'** task in CASA:

```
#  exportfits :: Convert a CASA image to a FITS file
imagename           = 'HD141569_natural.image.tt0' #  Name of input CASA image
fitsimage           =         ''        #  Name of output image FITS file
velocity            =      False        #  Use velocity (rather than frequency) as spectral axis
optical             =      False        #  Use the optical (rather than radio) velocity convention
bitpix              =        -32        #  Bits per pixel
minpix              =          0        #  Minimum pixel value (if minpix > maxpix, value is
                                        #   automatically determined)
maxpix              =         -1        #  Maximum pixel value (if minpix > maxpix, value is
                                        #   automatically determined)
overwrite           =       True        #  Overwrite pre-existing imagename
dropstokes          =      False        #  Drop the Stokes axis?
stokeslast          =       True        #  Put Stokes axis last in header?
history             =       True        #  Write history to the FITS image?
async               =      False        #  If true the taskname must be started using
                                        #   exportfits(...)
```

You can also export your CASA measurement set (the original visibilities) as a FITS file using the **'exportuvfits'** task:

```
#  exportuvfits :: Convert a CASA visibility data set to a UVFITS file:
vis                 = 'field3.ms'       #  Name of input visibility file
fitsfile            =         ''        #  Name of output UV FITS file
datacolumn          = 'corrected'       #  Visibility file data column
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
antenna             =         ''        #  Select data based on antenna/baseline
timerange           =         ''        #  Select data based on time range
avgchan             =          1        #  Channel averaging width (value > 1 indicates averaging)
writesyscal         =      False        #  Write GC and TY tables, (Not yet available)
multisource         =       True        #  Write in multi-source format
combinespw          =       True        #  Export the spectral windows as IFs
     padwithflags   =       True        #  Fill in missing data with flags to fit IFs

writestation        =       True        #  Write station name instead of antenna name
async               =      False        #  If true the taskname must be started using
                                        #   exportuvfits(...)
```

A Few Things to Remember

1. Use your online resources. There are lots of them!
2. While CASA is a very powerful tool, it can also be very buggy. Don't get frustrated. Try quitting and restarting.
3. The tasks you are more likely to use are…

   simobserve: simulating observations
   tclean: deconvolving your image
   viewer: displaying your image
   exportfits: exporting your image
   exportuvfits: exporting your visibilities
   plotms: examining your visibilities

4. CASA is built in iPython, so use your python intuition. Variables and lists are all defined using Python syntax.

Try all of this yourself NOW in the hands-on tutorial!

NRAO

www.nrao.edu
science.nrao.edu

The National Radio Astronomy Observatory is a facility of the National Science Foundation operated under cooperative agreement by Associated Universities, Inc.

NRAO