# A Crash Course in CASA

Amy Kimball (NRAO)

ALMA Data Reduction Workshop

1 Dec 2011

Atacama Large Millimeter/submillimeter Array
Expanded Very Large Array
Robert C. Byrd Green Bank Telescope
Very Long Baseline Array

NRAO

# Outline

- What CASA is and where to get it
- CASA documentation and resources
- CASA interface: Python, tools, and tasks
- Key CASA tasks for data reduction/calibration
- CASA tasks for examining your data

# Introduction

- Transformational leap in hardware capability with ALMA needs equally transformational capabilities for data reduction and analysis.
- ***CASA (Common Astronomy Software Applications)*** is the offline data reduction package for ALMA (both interferometric and single-dish)
- CASA infrastructure is a set of C++ *tools* bundled together under an *iPython* interface as a set of data reduction *tasks*
- Used daily in Chile for ALMA Commissioning and Science Verification
- ALMA pipeline is being built from the CASA *toolkit*

# CASA (Common Astronomy Software Applications)

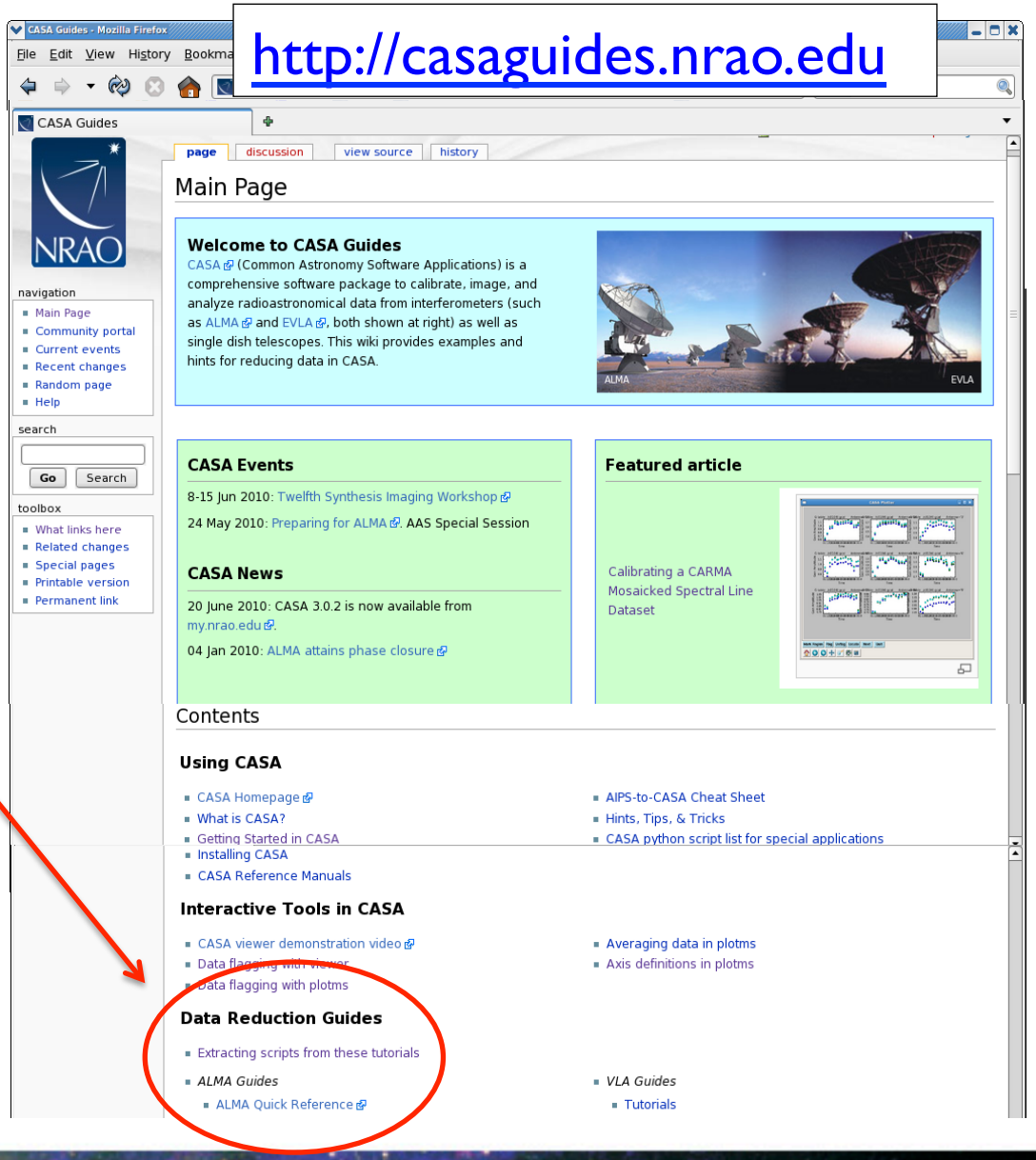
- ***CASA (Common Astronomy Software Applications)*** is the offline data reduction package for ALMA (both interferometric and single-dish)
- Current version: 3.3.0 (released 16 Nov 2011)
  - New releases about every 6 months
- Download from: http://casa.nrao.edu/
- Linux and Mac OS (≥10.6)
- Two versions available: *Release* and *Stable*
  - *Release*: current release with vigorous testing
  - *Stable*: more functionality; testing for a future release
  - Available documentation is for the *Release* version

# CASA documentation and user support

- CASA home: http://casa.nrao.edu → 'Using CASA'
  CASA Cookbook, online reference, example scripts
- Subscribe via: http://casa.nrao.edu → Getting Help → Mailing lists
  *casa-announce*: info about new releases, workshops, etc.
  *casa-users*: critical bugs and code updates
- Training material on "CASAguides" wiki: http://casaguides.nrao.edu
- NRAO user's forum: https://science.nrao.edu/forums/
- ALMA helpdesk: http://alma-help.nrao.edu
  NRAO helpdesk: http://help.nrao.edu

# CASAguides online resource

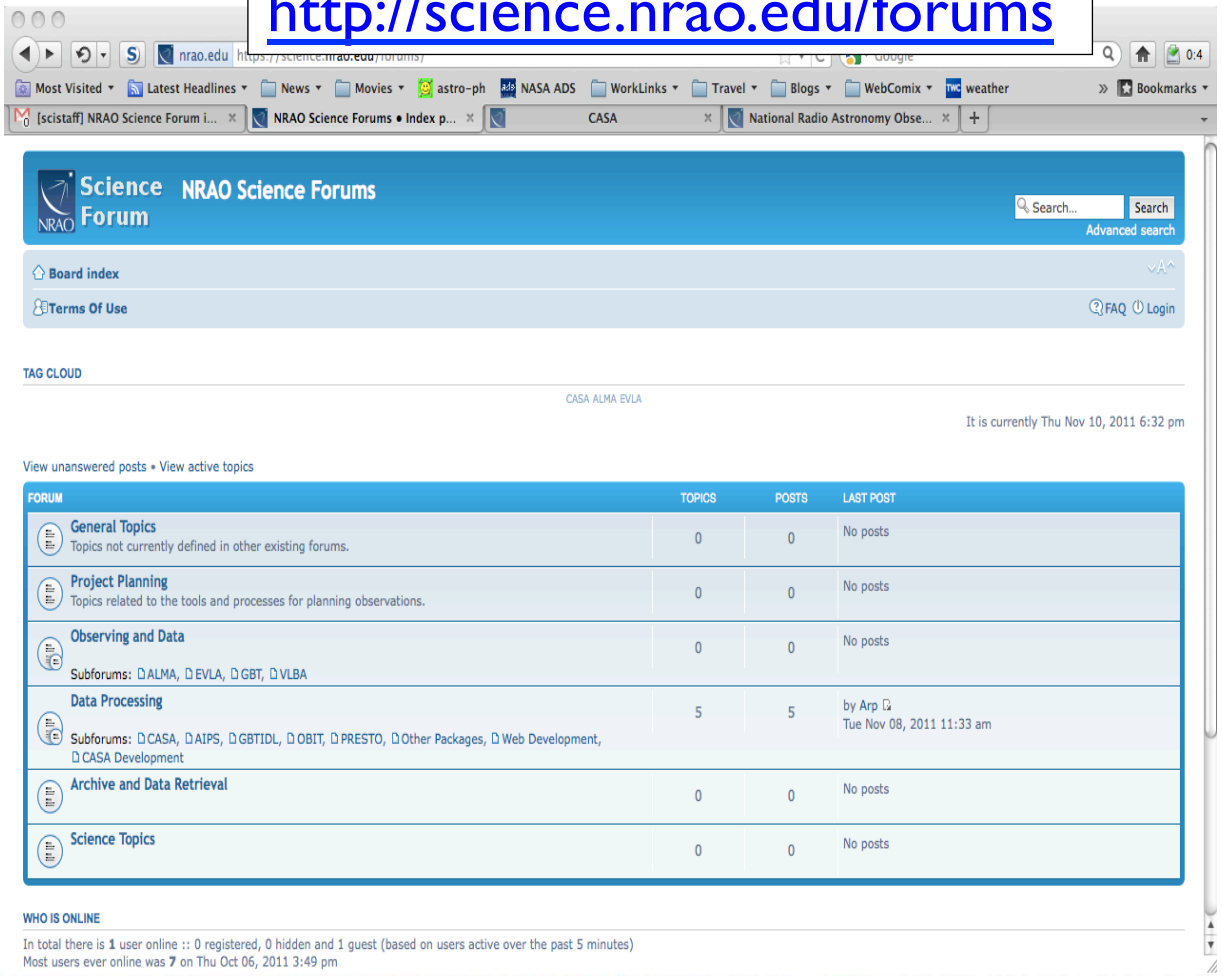
- Casaguides has fully annotated scripts including screen shots
- CASA user manual
- CASAguides for ALMA science verification data
- (Millimeter guides for EVLA, CARMA, SMA can also serve as important learning tools for ALMA data)



# NRAO user's forum

- Launched October 2011
- Discuss topics across NRAO services, including data processing/CASA
- Not officially staffed, but monitored by NRAO for accuracy

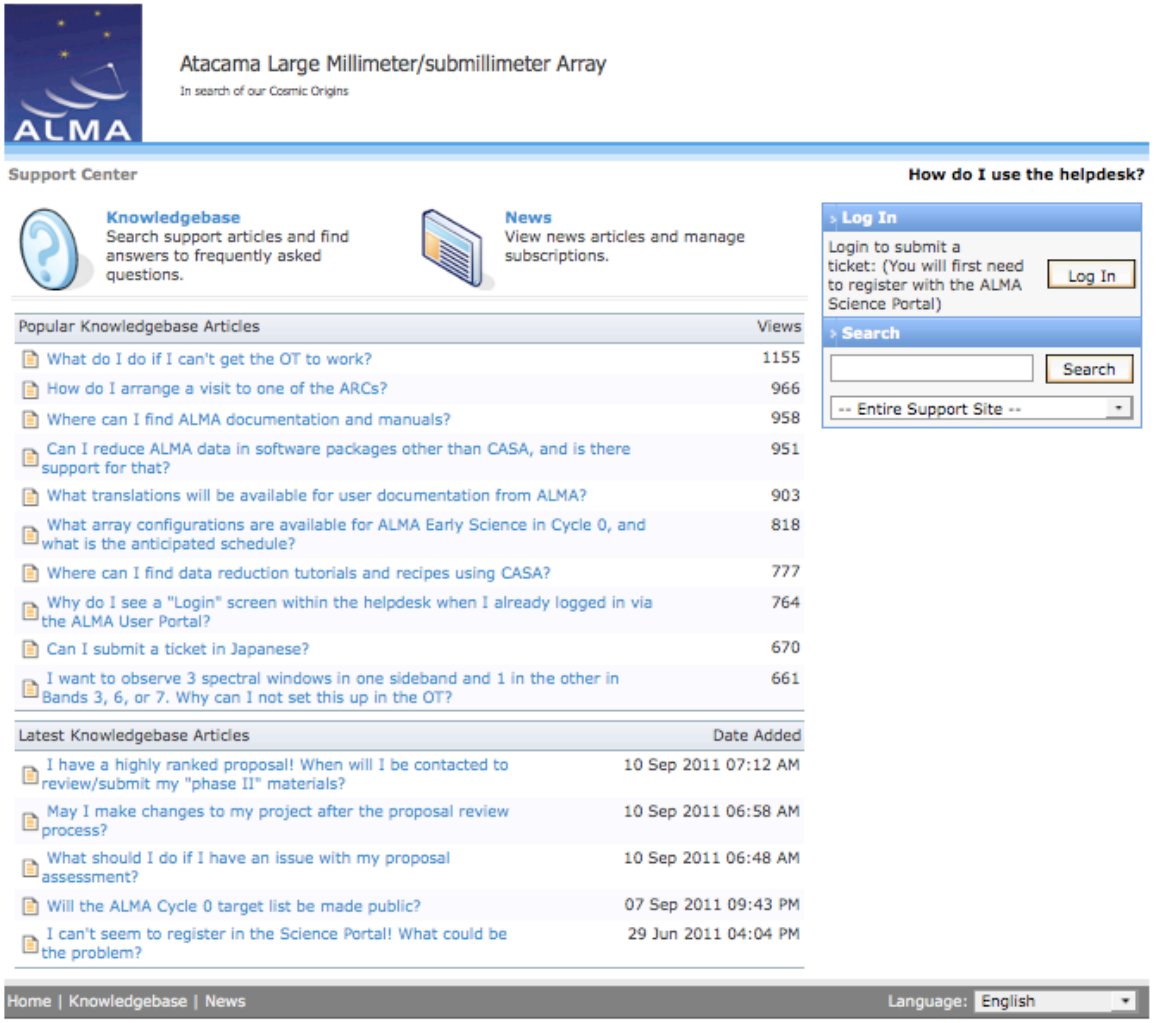
http://science.nrao.edu/forums



# ALMA helpdesk

- ALMA helpdesk: combines utilities of managing tickets/user support with knowledge base
- Best place to go with ALMA specific data-reduction questions

http://alma-help.nrao.edu

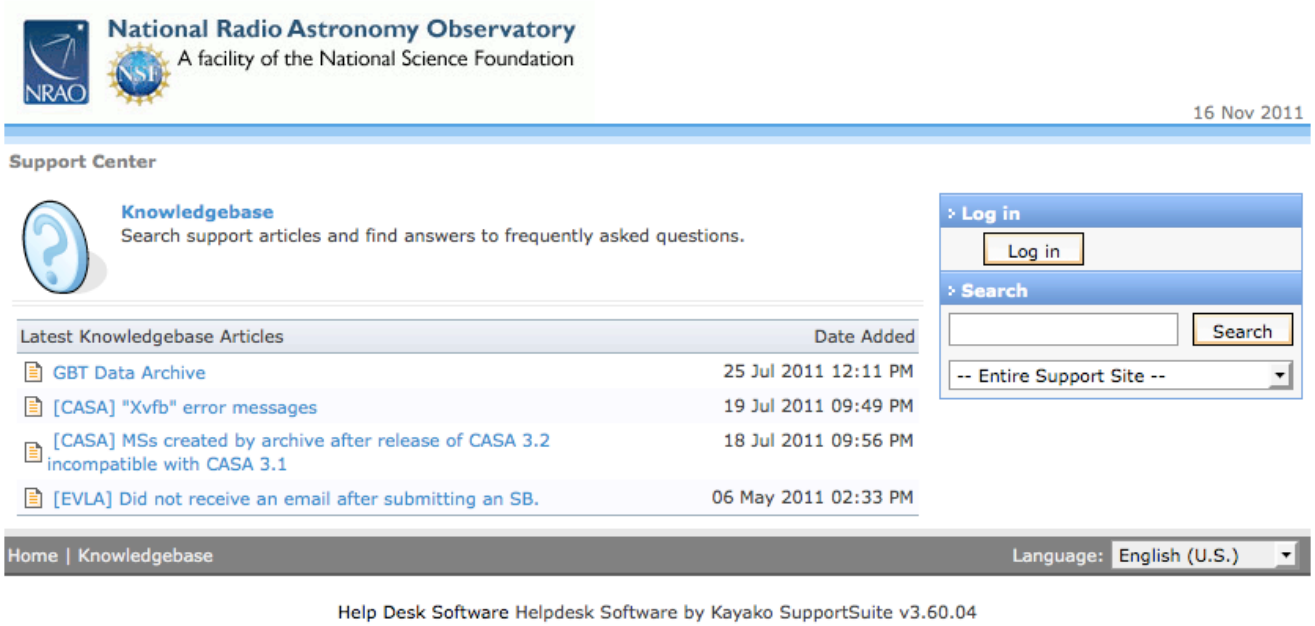



NRAO

# NRAO helpdesk

- Best place for non-ALMA data-reduction questions
- NRAO and ALMA helpdesks will be merged into one, around June 2012

http://help.nrao.edu



# Outline

- What CASA is and where to get it
- CASA documentation and resources
- **CASA interface: Python, tools, and tasks**
- Key CASA tasks for data reduction/calibration
- CASA tasks for examining your data

# CASA startup

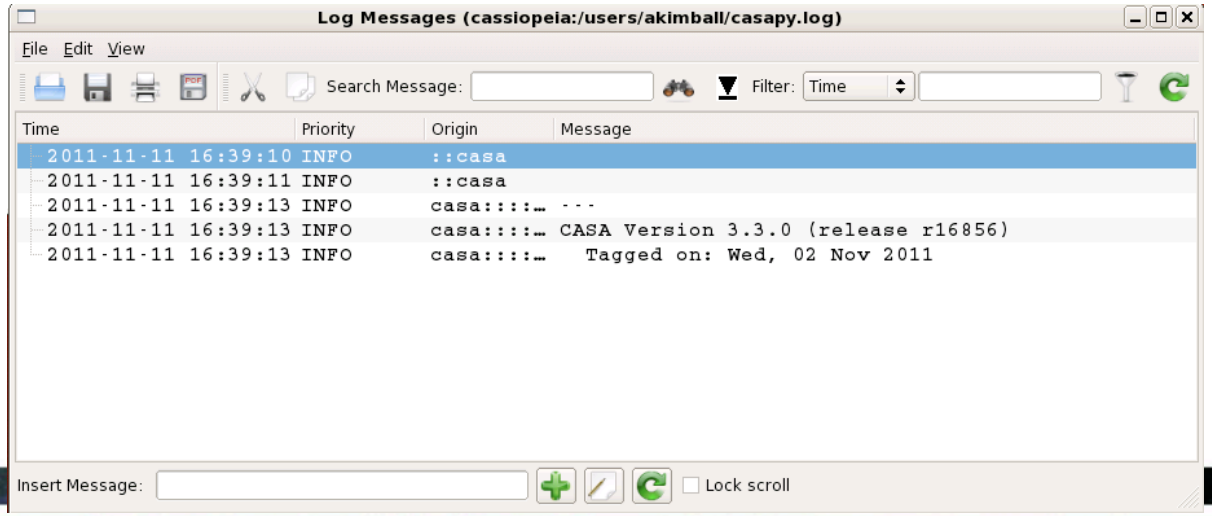
## MyComputer$ casapy

```
CASA Version 3.3.0 (r16856)
  Compiled on: Thu 2011/11/03 18:24:40 UTC
Initializing CASA python path in ~/.casa/init.py

--------------------------------------------------------------------------------
    For help use the following commands:
    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
    Single Dish sd* tasks are available after asap_init() is run

--------------------------------------------------------------------------------
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython.log
Mode           : backup
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active

CASA <3>:
```



# CASA uses iPython for an interactive environment

- Shell access
- Auto-parenthesis, tab completion
- Command history/searching
- Session logging
  - ipython.log – ipython command history
  - casapy.log – casa logger messages
- Line numbered input/output

# iPython pointers

- To run a .py script

  execfile('scriptname.py')
- Indentation matters!
  - careful with cut/paste to python (a few lines at a time)
  - or use command *cpaste*
- Run shell commands with leading exclamation mark: !du -hc
- Python starts counting/indexing from 0

- Python tutorials: http://python.org/doc
- iPython: http://ipython.org

# CASA has tasks and tools

- Tools – complete functionality
  - Interface to underlying C++ code
  - Objects: call with <tool>.<method>
  - List available tools with command toolhelp

- Tasks – high-level functionality
  - Python wrapper around the toolkit and pythoncode
  - Functions: normal function call or parameter setting interface
  - List CASA tasks with command tasklist or taskhelp

# CASA tasks

- List of tasks organized by type: tasklist
- Import/export, Information, Editing, Manipulation, Calibration, Modeling, Imaging, Analysis, Visualization, Simulation, Single disk, Utility

| Import/export | Information | Editing | Manipulation |
|---|---|---|---|
| exportfits | imhead | fixplanets | concat |
| exportuvfits | imstat | fixvis | conjugatevis |
| importaipscaltable | imval | flagautocorr | cvel |
| importasdm | listcal | flagcmd | fixvis |
| importfits | listhistory | flagdata | hanningsmooth |
| importfitsidi | listobs | flagmanager | imhead |
| importuvfits | listvis | msview | msmoments |
| importvla | plotms | plotms | plotms |
| (exportasdm) | plotuv | plotxy | plotxy |
| (importevla) | plotxy | (flagdata2) | split |
| (importgmrt) | vishead | (testautoflag) | testconcat |
| {importoldasdm} | visstat | | uvcontsub |
| | (listsdm) | | vishead |
| | | | {uvcontsub2} |

| Calibration | Modeling | Imaging | Analysis |
|---|---|---|---|
| accum | setjy | clean | imcollapse |
| applycal | uvcontsub | deconvolve | imcontsub |
| bandpass | uvmodelfit | feather | imfit |
| blcal | uvsub | ft | imhead |
| calstat | {uvcontsub2} | imcontsub | immath |
| clearcal | | (boxit) | immoments |
| fixplanets | | (csvclean) | impbcor |
| fluxscale | | {mosaic} | imregrid |
| ft | | {widefield} | imsmooth |
| gaincal | | | imstat |
| gencal | | | imtrans |
| listcal | | | imval |
| plotants | | | listvis |
| plotcal | | | slsearch |
| polcal | | | splattotable |
| setjy | | | (specfit) |
| smoothcal | | | |
| uvmodelfit | | | |
| uvsub | | | |

| Visualization | Simulation | Single dish | Utility |
|---|---|---|---|
| clearplot | sim_analyze | asap_init | browsetable |
| imview | sim_observe | sdaverage | clearplot |
| msview | simdata | sdbaseline | clearstat |
| plotants | | sdcal | concat |

# CASA tasks

- List of tasks with short help: taskhelp
- AIPS – CASA dictionary is available at https://safe.nrao.edu/wiki/bin/view/Software/CASA-AIPSDictionary
- (Historic) MIRIAD – CASA dictionary available in the CASA cookbook

```
accum              : Accumulate incremental calibration solutions into a calibration table
applycal           : Apply calibrations solutions(s) to data
autoclean          : CLEAN an image with automatically-chosen clean regions.
bandpass           : Calculates a bandpass calibration solution
blcal              : Calculate a baseline-based calibration solution (gain or bandpass)
boxit              : Box regions in image above given threshold value.
browsetable        : Browse a table (MS, calibration table, image)
calstat            : Displays statistical information on a calibration table
clean              : Invert and deconvolve images with selected algorithm
clearcal           : Re-initializes the calibration for a visibility data set
clearplot          : Clear the matplotlib plotter and all layers
clearstat          : Clear all autolock locks
concat             : Concatenate several visibility data sets.
conjugatevis       : Change the sign of the phases in all visibility columns.
csvclean           : This task does an invert of the visibilities and deconvolve in the image plane
.
cvel               : regrid an MS to a new spectral window / channel structure or frame
deconvolve         : Image based deconvolver
exportasdm         : Convert a CASA visibility file (MS) into an ALMA Science Data Model
exportfits         : Convert a CASA image to a FITS file
exportuvfits       : Convert a CASA visibility data set to a UVFITS file:
feather            : Combine two images using their Fourier transforms
find               : Find string in tasks, task names, parameter names:
fixplanets         : Changes FIELD and SOURCE table entries based on user given direction or POINTI
NG table, optionally fixes the UVW coordinates
fixvis             : Recalculates (u, v, w) and/or changes Phase Center
flagautocorr       : Flag autocorrelations
flagcmd            : Flagging task based on flagging commands
flagdata           :  All purpose flagging task based on selections
flagdata2          :  All purpose flagging task based on selections. It allows the combination of s
everal modes.
flagmanager        : Enable list, save, restore, delete and rename flag version files.
fluxscale          : Bootstrap the flux density scale from standard calibrators
ft                 : Insert a source model into the MODEL_DATA column of a visibility set:
gaincal            : Determine temporal gains from calibrator observations
gencal             : Specify Calibration Values of Various Types
hanningsmooth      : Hanning smooth frequency channel data to remove Gibbs ringing
imcollapse         : Collapse image along one axis, aggregating pixel values along that axis.
imcontsub          : Estimates and subtracts continuum emission from an image cube
imfit              : Fit one or more elliptical Gaussian components on an image region(s)
imhead             : List, get and put image header parameters
immath             : Perform math operations on images
immoments          : Compute moments from an image
impbcor            : Construct a primary beam corrected image from an image and a primary beam patt
ern.
importaipscaltable : Convert a AIPS calibration table (FITS format) to a CASA calibration table
importasdm         : Convert an ALMA Science Data Model observation into a CASA visibility file
importevla         : Convert an Science Data Model observation into a CASA Measurement Set
importfits         : Convert an image FITS file into a CASA image
importfitsidi      : Convert a FITS-IDI file to a CASA visibility data set
```

# Basic task commands

- parameter manipulation commands:

  default(<taskname>) : sets task's parameters to default values

  inp(<taskname>) : see task's parameter settings (input values)

  saveinputs(<taskname>) : saves parameters to <taskname>.saved

  tget(<taskname>) : retrieves parameters (<taskname>.last)

  shortcut: use simply default, inp, saveinputs, tget (without taskname) to manipulate current task

- get help on any task with help(<taskname>)

# Task interface

Examine task parameters (inputs) with *inp* :

```
CASA <49>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'mydata.ms'       #  Name of input visibility file
caltable            = 'mytable.bandpass.bpcal' #  Name of output gain calibration table
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
intent              =         ''        #  Select observing intent
selectdata          =       True        #  Other data selection parameters
     timerange      =         ''        #  Select data based on time range
     uvrange        =         ''        #  Select data within uvrange (default units meters)
     antenna        =         ''        #  Select data based on antenna/baseline
     scan           =         ''        #  Scan number range
     observation    =         ''        #  Select by observation ID(s)
     msselect       =         ''        #  Optional complex data selection (ignore for now)

solint              =      'inf'        #  Solution interval: egs. 'inf', '60s' (see help)
combine             =         ''        #  Data axes which to combine for solve (scan, spw, and/or
                                        #   field)
preavg              = 'hogwarts'        #  Pre-averaging interval (sec) (rarely needed)
refant              =         ''        #  Reference antenna name(s)
minblperant         =          4        #  Minimum baselines _per antenna_ required for solve
minsnr              =        3.0        #  Reject solutions below this SNR
solnorm             =      False        #  Normalize average solution amplitudes to 1.0 (G, T only)
```

# Task parameter checking

Default values in BLACK

```
CASA <49>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'mydata.ms'       #  Name of input visibility file
caltable            = 'mytable.bandpass.bpcal' #  Name of output gain calibration table
field               =           ''      #  Select field using field id(s) or field name(s)
spw                 =           ''      #  Select spectral window/channels
intent              =           ''      #  Select observing intent
selectdata          =         True      #  Other data selection parameters
     timerange      =           ''      #  Select data based on time range
     uvrange        =           ''      #  Select data within uvrange (default units meters)
     antenna        =           ''      #  Select data based on antenna/baseline
     scan           =           ''      #  Scan number range
     observation    =           ''      #  Select by observation ID(s)
     msselect       =           ''      #  Optional complex data selection (ignore for now)

solint              =        'inf'      #  Solution interval: egs. 'inf', '60s' (see help)
combine             =           ''      #  Data axes which to combine for solve (scan, spw, and/or
                                        #   field)
preavg              =  'hogwarts'       #  Pre-averaging interval (sec) (rarely needed)
refant              =           ''      #  Reference antenna name(s)
minblperant         =            4      #  Minimum baselines _per antenna_ required for solve
minsnr              =          3.0      #  Reject solutions below this SNR
solnorm             =        False      #  Normalize average solution amplitudes to 1.0 (G, T only)
```

# Task parameter checking

Expandable parameters are highlighted; sub-parameters one level lower

```
CASA <49>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'mydata.ms'        #  Name of input visibility file
caltable            = 'mytable.bandpass.bpcal' #  Name of output gain calibration table
field               =         ''         #  Select field using field id(s) or field name(s)
spw                 =         ''         #  Select spectral window/channels
intent              =         ''         #  Select observing intent
selectdata          =       True         #  Other data selection parameters
     timerange      =         ''         #  Select data based on time range
     uvrange        =         ''         #  Select data within uvrange (default units meters)
     antenna        =         ''         #  Select data based on antenna/baseline
     scan           =         ''         #  Scan number range
     observation    =         ''         #  Select by observation ID(s)
     msselect       =         ''         #  Optional complex data selection (ignore for now)

solint              =      'inf'         #  Solution interval: egs. 'inf', '60s' (see help)
combine             =         ''         #  Data axes which to combine for solve (scan, spw, and/or
                                         #   field)
preavg              = 'hogwarts'         #  Pre-averaging interval (sec) (rarely needed)
refant              =         ''         #  Reference antenna name(s)
minblperant         =          4         #  Minimum baselines _per antenna_ required for solve
minsnr              =        3.0         #  Reject solutions below this SNR
solnorm             =      False         #  Normalize average solution amplitudes to 1.0 (G, T only)
```

# Task parameter checking

User set values in BLUE

```
CASA <49>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'mydata.ms'       #  Name of input visibility file
caltable            = 'mytable.bandpass.bpcal' #  Name of output gain calibration table
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
intent              =         ''        #  Select observing intent
selectdata          =       True        #  Other data selection parameters
     timerange      =         ''        #  Select data based on time range
     uvrange        =         ''        #  Select data within uvrange (default units meters)
     antenna        =         ''        #  Select data based on antenna/baseline
     scan           =         ''        #  Scan number range
     observation    =         ''        #  Select by observation ID(s)
     msselect       =         ''        #  Optional complex data selection (ignore for now)

solint              =      'inf'        #  Solution interval: egs. 'inf', '60s' (see help)
combine             =         ''        #  Data axes which to combine for solve (scan, spw, and/or
                                        #   field)
preavg              = 'hogwarts'        #  Pre-averaging interval (sec) (rarely needed)
refant              =         ''        #  Reference antenna name(s)
minblperant         =          4        #  Minimum baselines _per antenna_ required for solve
minsnr              =        3.0        #  Reject solutions below this SNR
solnorm             =      False        #  Normalize average solution amplitudes to 1.0 (G, T only)
```

# Task parameter checking

Erroneous values in RED

```
CASA <49>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'mydata.ms'       #  Name of input visibility file
caltable            = 'mytable.bandpass.bpcal' #  Name of output gain calibration table
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
intent              =         ''        #  Select observing intent
selectdata          =       True        #  Other data selection parameters
     timerange      =         ''        #  Select data based on time range
     uvrange        =         ''        #  Select data within uvrange (default units meters)
     antenna        =         ''        #  Select data based on antenna/baseline
     scan           =         ''        #  Scan number range
     observation    =         ''        #  Select by observation ID(s)
     msselect       =         ''        #  Optional complex data selection (ignore for now)

solint              =      'inf'        #  Solution interval: egs. 'inf', '60s' (see help)
combine             =         ''        #  Data axes which to combine for solve (scan, spw, and/or
                                        #   field)
preavg              = 'hogwarts'        #  Pre-averaging interval (sec) (rarely needed)
refant              =         ''        #  Reference antenna name(s)
minblperant         =          4        #  Minimum baselines _per antenna_ required for solve
minsnr              =        3.0        #  Reject solutions below this SNR
solnorm             =      False        #  Normalize average solution amplitudes to 1.0 (G, T only)
```

# Task execution

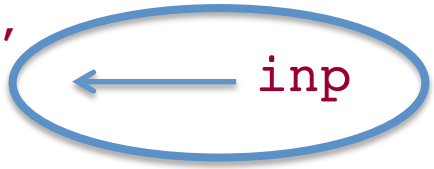
- Two ways to run a task:
  - call from Python as a function with arguments

```
gaincal(vis='mydata.ms', caltable='caltable.cal', field='2')
```

unspecified parameters will use default value

  - standard task interface: use global variables to set parameters

```
default(gaincal)
vis = 'mydata.ms'
caltable = 'mycaltable.cal'
field = '2'
gaincal()
```




- some tasks return Python dictionaries, e.g., `myval = imval()`

# Task help

In-line help:

help gaincal

```
CASA <50>: help gaincal
---------> help(gaincal)
Help on gaincal task:

Determine temporal gains from calibrator observations

      The complex gains for each antenna/spwid are determined from the
      data column (raw data) divided by the model column.  The gains can
      be obtained for a specified solution interval, spw combination and
      field combination.  The GSPLINE spline (smooth) option is still under
      development.

      Previous calibrations (egs, bandpass, opacity, parallactic angle) can
      be applied on the fly.  At present with dual-polarized data, both
      polarizations must be unflagged for any solution to be obtained.

      Keyword arguments:
      vis -- Name of input visibility file
              default: none; example: vis='ngc5921.ms'
      caltable -- Name of output gain calibration table
              default: none; example: caltable='ngc5921.gcal'

      --- Data Selection (see help par.selectdata for more detailed information)

      field -- Select field using field id(s) or field name(s).
                 ['go listobs' to obtain the list id's or names]
              default: ''=all fields
              If field string is a non-negative integer, it is assumed a
                field index,  otherwise, it is assumed a field name
              field='0~2'; field ids 0,1,2
              field='0,4,5~7'; field ids 0,4,5,6,7
              field='3C286,3C295'; field named 3C286 and 3C295
              field = '3,4C*'; field id 3, all names starting with 4C
          DON'T FORGET TO INCLUDE THE FLUX DENSITY CALIBRATOR IF YOU HAVE ONE
      spw -- Select spectral window/channels
                type 'help par.selection' for more examples.
              spw='0~2,4'; spectral windows 0,1,2,4 (all channels)
              spw='<2';  spectral windows less than 2 (i.e. 0,1)
              spw='0:5~61'; spw 0, channels 5 to 61, INCLUSIVE
              spw='*:5~61'; all spw with channels 5 to 61
```

# Tools in CASA

- What if there is no task?

➔ use CASA tools! (tasks are built upon tools)

mostly written in C++ (with some Python)

- tools objects with functions/methods
  - call from CASA as `toolname.methodname()`
  - default tool objects are pre-constructed
    - e.g., imager (`im`) , calibrater (`cb`), measurement set (`ms`) , etc. (see command `toolhelp`)

- tools described (sorta) in the CASA Toolkit Reference Manual:

http://casa.nrao.edu/docs/CasaRef/CasaRef.html

# Outline

- What CASA is and where to get it
- CASA documentation and resources
- CASA interface: Python, tools, and tasks
- **Key CASA tasks for data reduction/calibration**
- CASA tasks for examining your data

# Key tasks for data calibration

- flagdata/flagcmd/flagmanager: flag (remove) bad data
- setjy: set "model" column using known model for a calibrator
- bandpass: calculate bandpass calibration (amp/phase vs frequency)
- gaincal: calculate temporal gain calibration (amp/phase vs time)
- fluxscale: apply absolute flux scaling from standard calibrators
- plotcal: examine a calibration table
- applycal: apply calibration table(s) from previous steps
- split: split off calibrated data from your ms (for imaging!)

# Selecting subsets of data

- Many calibration (and other) tasks have a selectdata parameter

```
selectdata          =        True       #  Data selection parameters, which will affect all modes (antenna, timerange etc)
     field          =          ''       #  Field names or field index numbers: ''==>all, field='0~2,3C286'
     spw            =          ''       #  spectral-window/frequency/channel
     antenna        =          ''       #  antenna/baselines: ''==>all, antenna = '3,VA04'
     timerange      =          ''       #  time range: ''==>all, timerange='09:14:0~09:54:0'
     correlation    =          ''       #  Select data based on correlation
     scan           =          ''       #  scan numbers: ''==>all
     intent         =          ''       #  Select data based on observation intent: ''==>all
     feed           =          ''       #  multi-feed numbers: Not yet implemented
     array          =          ''       #  (sub)array numbers: ''==>all
     uvrange        =          ''       #  uv range: ''==>all; uvrange = '0~100klambda', default units=meters
     observation    =          ''       #  Select data based on observation ID: ''==>all
```

- See help for each task, especially `help(gaincal)`

# Flagging (or unflagging) your data
## a few important notes

1. Data in CASA are either flagged or not flagged.
   - Every bit of data has its own flag (set to True or False).
   - Flags for '`mydata.ms`' stored in '`mydata.ms.flagversions`'

2. Most flagging tasks have the option of creating a flag backup: store the state of the flags before running a flagging task.

3. Using *flagmanager*, flag states can be saved or restored.

# Flagging with *flagdata*

All purpose flagging task based on data selection.

- `vis = 'inputms.ms'` # (all these tasks have parameter vis)
- `flagbackup = True` or `False` # backup flags before running?
- `selectdata = True` or `False` # expandable parameter
- `mode = 'manualflag'` or `'shadow'` or `'quack'`

`manualflag` allows clipping, unflagging, averaging over channels
`shadow` for removing shadowed antennas (parameter `diameter`)
`quack` for flagging time ranges near beginning/ending of scans

# Flagging with *flagcmd*

All purpose flagging task based on commands. Many calls to *flagdata* can equal a single call to *flagcmd*.

- `flagmode = 'cmd' or 'file'`
  `flagfile or flagcmd =`
  `'filename' or ['list of commands']`
- `optype = 'apply'`
  `flagbackup = True or False`

# Example: *flagcmd* vs *flagdata*

```
flagdata(vis='X5d8.ms',antenna='DV04',timerange='01:50:00~02:50:00',flagbackup=F)
flagdata(vis='X5d8.ms',antenna='PM01',timerange='01:50:00~03:20:00',spw='1~8',flagbackup=F)
flagdata(vis='X5d8.ms',clipminmax=[0,1E-10],clipexpr='ABS XX',clipoutside=F,flagbackup=F)
flagdata(vis='X5d8.ms',clipminmax=[0,1E-10],clipexpr='ABS YY',clipoutside=F,flagbackup=F)
```

```
flagcmd(vis='X5d8.ms',flagmode='cmd',flagbackup=F,
        command=["antenna='DV04' timerange='01:50:00~02:50:00' ",
                 "antenna='PM01' timerange='01:50:00~03:20:00',spw='1~8' ",
                 "cliprange='0~1E-10' clipexpr='ABS_XX' ",
                 "cliprange='0~1E-10' clipexpr='ABS_YY' "]
```

# Flagmanager

Gives control to user for managing flag states.

- `vis = 'inputms.ms'`
- `mode = 'list' 'save' 'restore' 'delete' 'rename'`
  `versionname = 'name of flag state'`
  `comment = 'user comment'`

# Set up flux calibrator model: *setjy*

- `field = '<field name or number>'`
- `scalebychan = True` **OR** `False`
- `standard = 'Butler-JPL-Horizons 2010'` # solar system
  `'Perley-Butler 2010'` # extragalactic

*Setjy* sets the values in the "model" column of the ms for the selected field.

If source has non-zero spectral index, you MUST set `scalebychan=True`

Models exist for several quasars, planets, and moons.

(See '`help setjy`' for specifics.)

# Calibrating amp/phase vs frequency: *bandpass*

- caltable = 'bandpass.bcal'
- field = '<field name or number>'
- solint = 'inf' **or** 'int' **or** '10s'
- combine = '' **or** 'scan' **or** 'spw'
- refant = 'DV09' # or your favorite antenna
- bandtype = 'B' **or** 'BPOLY'
- gaintable = **previous calibration tables?**
- gainfield = **fields for previous tables**

# Calibrating amp/phase vs time: *gaincal*

- `caltable` = `'mycaltable.gcal'`
- `field` = `'<field name or #>'`
- `solint` = `'inf'` **or** `'int'` **or** `'10s'`
- `combine` = `''` **or** `'scan'` **or** `'spw'`
- `refant` = `'DV09'` # or your favorite antenna
- `calmode` = `'p'` **or** `'ap'`
- `gaintable` = **previous calibration tables?**
- `gainfield` = **fields for previous tables**

Common to run a `solint='int'` gaincal before running `bandpass`, to correct phase vs time behavior, before calibrating phase vs frequency.

For calibrator sources, common to run phase-only gaincal (`calmode='p'`), then phase and amplitude (`calmode='ap'`).

# Setting the absolute flux scale of calibrated data: *fluxscale*

- `caltable = 'input amp/phase table'`
- `fluxtable = 'output amp/phase table'`
- `reference = 'field of flux cal source'`
- `transfer = ['fields to transfer scaling to']`
- `refspwmap = ['vector list of spws']*`

Applies flux scaling of the reference field (with known flux density) to a transfer field (gain calibrator source).

Output table *replaces* input table (unlike gaincal, table is not incremental).

* If an spw is bad, use refspwmap to tie transfer field's spw to a good spw of reference field: e.g., if spw 2 of Titan is contaminated by Saturn:

```
refspwmap = [0,1,3,3]
```

# Applying the calibration tables: *applycal*

Example below for a science target in field 2, with bandpass calibrator in field 0 and gain calibrator in field 1.

```
• field       =   '2'
• interp      =   'linear' or 'nearest'
• gaintable   =  ['bandpass.bpcal',
                  'gaincal.gcal', 'flux.fcal']
• gainfield   =  ['0', '1', '1']
• flagbackup  =   False  # use flagmanager instead!
```

*Applycal* applies the given calibrations to the "data" column in the ms and stores the results in the "corrected" column in the ms.

Running *applycal* a second time OVERWRITES the corrected column (but flagged data stays flagged!).

# Select out a subset of your MS: *split*

- e.g., write out the "corrected" column (the calibrated data) of your target source as a small MS of its own, in preparation for imaging
- optionally apply time averaging, frequency averaging
- can choose certain spws, UV ranges, antennas, scans, etc.

```
vis           =  'inputvis.ms'
outputvis     =  'outputvis.ms'
datacolumn    =  'corrected'  # becomes data column in new MS
field         =  '<field of science target>'
timebin       =  '10s'        # to time average in 10s intervals
width         =  '32'         # to average together 32 channels
```

# Key tasks for data calibration

- flagdata/flagcmd/flagmanager: flag (r… …a
- setjy: set "model" column usin… …r a calibrator
- bandpass: calculate bandp… …mp/phase vs frequency)
- gaincal: calculate tem… …tion (amp/phase vs time)
- fluxscale: apply f… …rom standard calibrators
- plotcal: ex… …n table
- applycal: …on table(s) from previous steps
- split: split o… …ated data from your ms (for imaging!);
  split can do averaging

Write a script!

# Outline

- What CASA is and where to get it
- CASA documentation and resources
- CASA interface: Python, tools, and tasks
- Key CASA tasks for data reduction/calibration
- **CASA tasks for examining your data**

# Examining your data

- Observing summary (sources, scans, spectral windows, antennas, etc…): *listobs*
- Plotting the antennas: *plotants*
- Plotting/displaying data: *plotcal, plotms*
  (more tomorrow in imaging talk)

**Examine your data carefully before flagging**

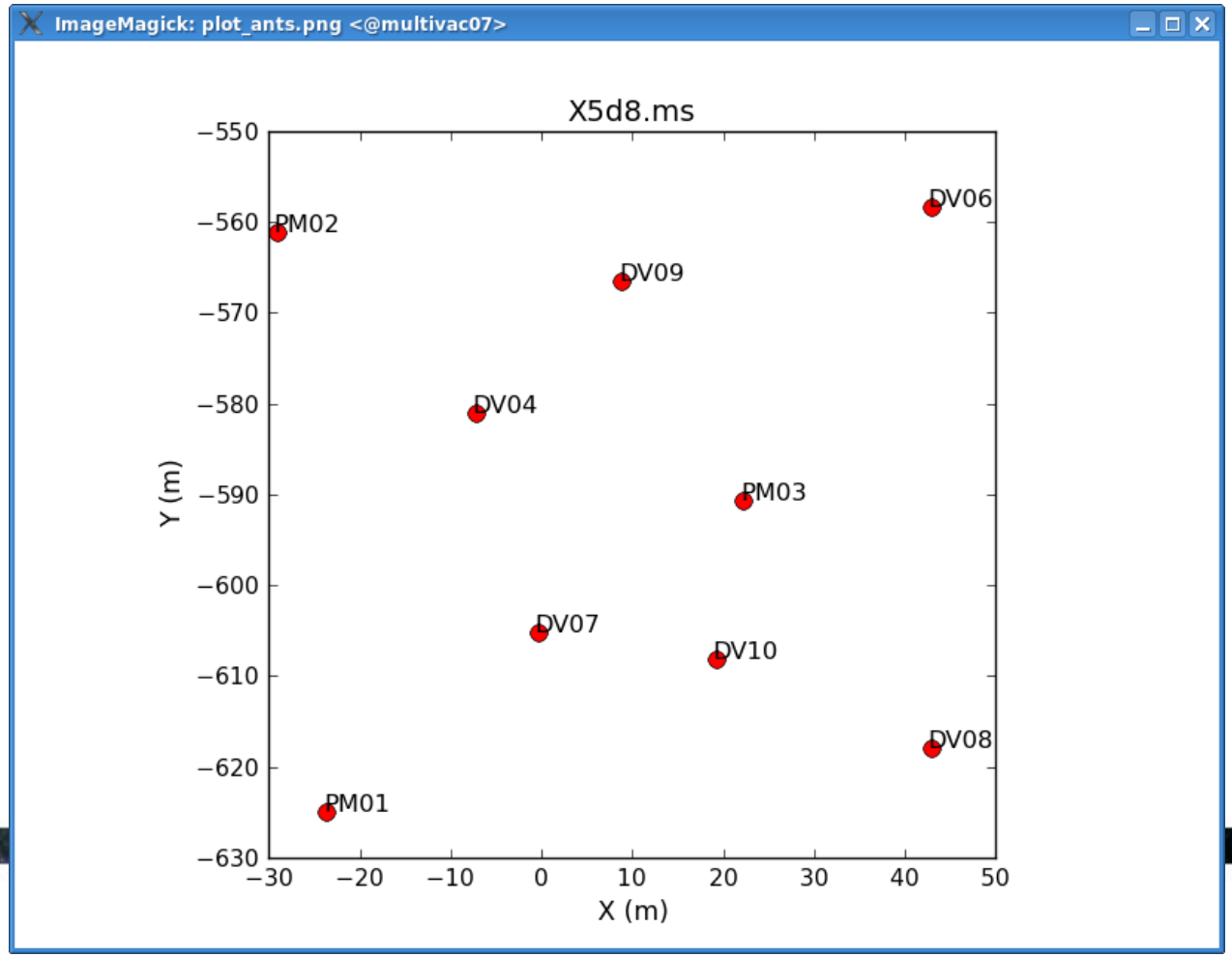
# Observing summary: *listobs*

```
vis                 = 'inputms.ms'
verbose             =  True or False
listfile            = 'outputfile.txt'
```

```
listobs:… ##########################################
listobs:… ##### Begin Task: listobs            #####
listobs:…
listobs:… ================================================================================
listobs:…            MeasurementSet Name:  /export/lustre/akimball/NGC3256/NGC3256_Band3_UnCalibratedMSandTablesForReduction/uid___A002_X1d54a1_X5.ms      MS Version 2
listobs:… ================================================================================
listobs:…    Observer: Unknown     Project: T.B.D.
listobs:… Observation: ALMA
listobs:… Data records: 206024       Total integration time = 3801.6 seconds
listobs:…    Observed from   16-Apr-2011/02:59:18.5   to   16-Apr-2011/04:02:40.1 (UTC)
listobs:…
listobs:…    ObservationID = 0         ArrayID = 0
listobs:…   Date        Timerange (UTC)          Scan  FldId FieldName           nRows   Int(s)   SpwIds       ScanIntent
listobs:…   16-Apr-2011/02:59:21.2 - 03:00:00.0     1      0 1037-295           1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:00:23.8 - 03:01:16.0     2      0 1037-295           2415   2.89     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_POINTING#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:02:19.3 - 03:02:58.6     3      1 Titan              1463   2.87     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:03:21.6 - 03:06:52.4     4      1 Titan              14553  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_AMPLI#ON_SOURCE,CALIBRATE_PHASE#ON_SOURCE,CALIBR
listobs:…                03:07:35.4 - 03:11:05.9     5      0 1037-295           14532  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_BANDPASS#ON_SOURCE,CALIBRATE_PHASE#ON_SOURCE,CAL
listobs:…                03:11:25.1 - 03:12:04.6     6      0 1037-295           1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:12:23.5 - 03:13:09.1     7      0 1037-295           2912   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_PHASE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:13:50.2 - 03:14:28.6     8      2 NGC3256            1449   2.89     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:14:53.7 - 03:24:33.4     9      2 NGC3256            38794  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] OBSERVE_TARGET#ON_SOURCE
listobs:…                03:24:52.2 - 03:25:31.0    10      0 1037-295           1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:25:50.4 - 03:26:35.5    11      0 1037-295           2905   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_PHASE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:26:54.7 - 03:27:33.1    12      2 NGC3256            1449   2.89     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:27:53.6 - 03:37:33.3    13      2 NGC3256            38794  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] OBSERVE_TARGET#ON_SOURCE
listobs:…                03:38:00.8 - 03:38:39.0    14      0 1037-295           1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:39:02.5 - 03:39:48.1    15      0 1037-295           2912   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_PHASE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:40:06.9 - 03:40:45.7    16      2 NGC3256            1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:41:06.4 - 03:50:45.9    17      2 NGC3256            38773  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] OBSERVE_TARGET#ON_SOURCE
listobs:…                03:51:04.8 - 03:51:43.5    18      0 1037-295           1449   2.89     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:52:03.4 - 03:52:49.2    19      0 1037-295           2905   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_PHASE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:53:16.4 - 03:53:54.8    20      2 NGC3256            1456   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_ATMOSPHERE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…                03:54:19.5 - 04:01:32.2    21      2 NGC3256            29078  2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] OBSERVE_TARGET#ON_SOURCE
listobs:…                04:01:54.2 - 04:02:40.1    22      0 1037-295           2905   2.88     [1, 3, 5, 7, 2, 4, 6, 8, 0] CALIBRATE_PHASE#ON_SOURCE,CALIBRATE_WVR#ON_SOURCE
listobs:…              (nVis = Total number of time/baseline visibilities per scan)
listobs:… Fields: 3
listobs:…   ID   Code Name                RA               Decl           Epoch   SrcId nVis
listobs:…   0    none 1037-295            10:37:16.07900 -29.34.02.8130 J2000   0     38759
listobs:…   1    none Titan               00:00:00.00000 +00.00.00.0000 J2000   1     16016
listobs:…   2    none NGC3256             10:27:51.60000 -43.54.18.0000 J2000   2     151249
listobs:…    (nVis = Total number of time/baseline visibilities per field)
listobs:… Spectral Windows:  (9 unique spectral windows and 2 unique polarization setups)
listobs:…   SpwID  #Chans Frame Ch1(MHz)    ChanWid(kHz)  TotBW(kHz)  Corrs
listobs:…   0           4 TOPO  184550      1500000       7500000     I
```

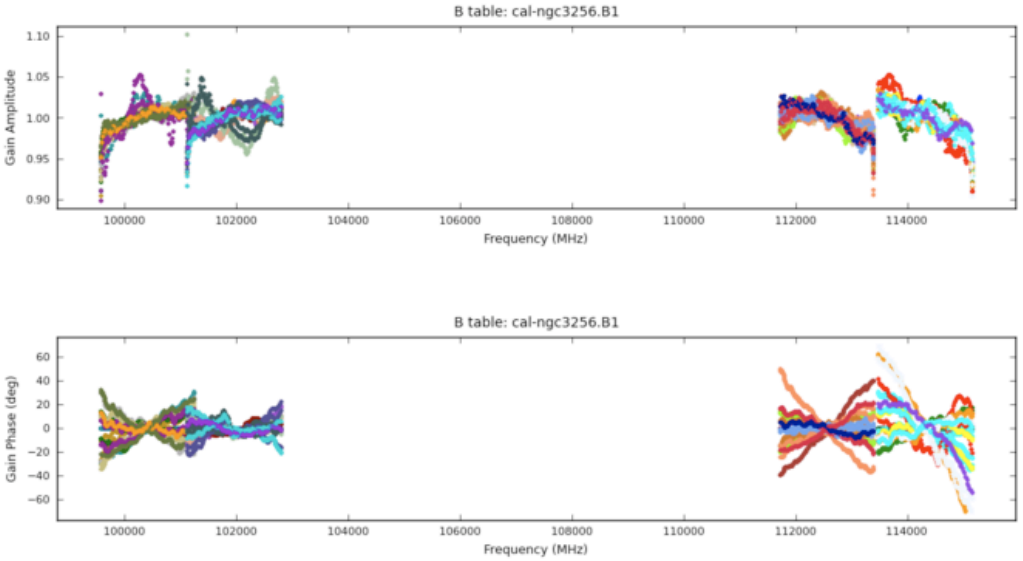
# Antenna positions: *plotants*

```
vis       = 'inputms.ms'
figfile   = 'antpos.png'
```



# Examining calibration tables: *plotcal*

- `caltable = 'bandpass.bpcal'` # table to plot
- `xaxis = 'frequency'` # or time, chan, etc.
- `yaxis = 'amp'` # or phase, etc.
- `iteration = 'antenna'` # or spw, baseline, etc.
- `subplot = 221` # number plots in xyz




Bandpass amplitude/phase plots

Antennae Band3 ALMA CASA Guide

# Data view: *plotms* (command line *casaplotms)*

- `xaxis = 'frequency'` # time, chan, uvdist, etc.
- `yaxis = 'amp'` # phase, antenna2, real, etc.
  `ydatacolumn = 'data'` # corrected, model, etc.
- `selectdata = True` # expandable parameter
- `averagedata = True`
  `avgchannel = '128'`
  `avgtime = '300s'`
  `avgscan = True`
  `avgbaseline = True`
- `iteraxis = 'antenna'` # scan, field, antenna, etc.
  `xselfscale/yselfscale = True`
- `coloraxis = 'baseline'` # scan, field, baseline, corr, ...

# Data Review: *plotms* (unix command line ***casaplotms***)