Introduction to CASA

Christopher Hales (NRAO Socorro)
on behalf of the CASA team

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

5th VLA Data Reduction Workshop
Socorro NM
14 March 2016

Overview of this talk

- General introduction to CASA
- Documentation and web resources
- Starting CASA
- Tasks, tools, and applications
- Structure of measurement sets and associated data
- CASA data selection syntax

General description

- CASA: Common Astronomy Software Applications
  - Post-processing package for next gen facilities like ALMA and VLA, and both interferometric and single dish
  - Data from other telescopes also usually work
  - Developed at NRAO (lead), ESO, NAOJ, CSIRO/ATNF and ASTRON
  - Active in community since October 2007
- Code is C++ (fast) underneath iPython interface (easy access & scripting)
- Latest CASA release is **version 4.5.2**
  - Many tasks and a lot of tools
  - Contains automated calibration pipeline

http://casa.nrao.edu

CASA releases

- New releases about every 6 months (typically May and November)
  - Also "monthly" versions that are markers on path to next release with more functionality, but likely contain unfinished developments, less tested code, and no up-to-date documentation

- Latest version 4.5.2 runs on:
  - Red Hat Linux 6 and 5 (64-bit)
  - Mac OS X 10.10 (Yosemite 64-bit) and 10.9 (Mavericks 64-bit)
  - May also work on other systems, see website for details
  - 4.6.0 is planned to run on Mac OS X 10.11 (El Capitan)

Why use CASA?

- Import data, inspect, edit, calibrate, image, view, analyze
  - CASA has some of the most sophisticated imaging algorithms (multi-scale clean, Taylor term expansion for wide bandwidths, W-term projection, OTF mosaicking, etc.)

- We have an active Algorithm Research Group, so expect more features in future versions…

CASA documentation and web resources

http://casa.nrao.edu

- About CASA
- CASA Releases
- Obtaining CASA
- Hardware Requirements
- Using CASA
- Getting Help
- Tutorials and Training
- CASA at NRAO

Search CASA

Go

CASA

About CASA

CASA, the *Common Astronomy Software Applications* package, is being developed with the primary goal of supporting the data post-processing needs of the next generation of radio astronomical telescopes such as ALMA and VLA. The package can process both interferometric and single dish data, and is developed by an international consortium of scientists based at the National Radio Astronomical Observatory (NRAO), the European Southern Observatory (ESO), the National Astronomical Observatory of Japan (NAOJ), the CSIRO Australia Telescope National Facility (CSIRO/ATNF), and the Netherlands Institute for Radio Astronomy (ASTRON) under the guidance of NRAO.

CASA
Common Astronomy Software Applications

The CASA infrastructure consists of a set of C++ tools bundled together under an iPython interface as a set of data reduction tasks. This structure provides flexibility to process the data via task interface or as a python script. In addition to the data reduction tasks, many post-processing tools are available for even more flexibility and special purpose reduction needs.

The latest CASA release is 4.5.2

For announcements and critical bug reports, consider to subscribe to the CASA mailing lists.
For CASA enquiries, please use the NRAO helpdesk

- Newsletter
- User Reference and Cookbook
- CASAguides
- NRAO Science Forums
- Old CASA Web Page

CASA documentation and web resources

http://casa.nrao.edu

- About CASA
- CASA Releases
- Obtaining CASA
- Hardware Requirements
- Using CASA
- Getting Help
  - CASA Helpdesk
  - NRAO Science User Forum
  - Mailing Lists
- Tutorials and Training
- CASA at NRAO

Search CASA

Go

CASA

About CASA

CASA, the *Common Astronomy Software Applications* package, is being developed with the primary goal of supporting the data post-processing needs of the next generation of radio astronomical and VLA. The interferometric is developed by m of scientists io Astronomical European O), the National Astronomical Observatory of Japan (NAOJ), the CSIRO Australia Telescope National Facility (CSIRO/ATNF), and the Netherlands Institute for Radio Astronomy (ASTRON) under the guidance of NRAO.

The CASA infrastructure consists of a set of C++ tools bundled together under an iPython interface as a set of data reduction tasks. This structure provides flexibility to process the data via task interface or as a python script. In addition to the data reduction tasks, many post-processing tools are available for even more flexibility and special purpose reduction needs.

The latest CASA release is 4.5.2

For announcements and critical bug reports, consider to subscribe to the CASA mailing lists.
For CASA enquiries, please use the NRAO helpdesk

- Newsletter
- User Reference and Cookbook
- CASAguides
- NRAO Science Forums
- Old CASA Web Page

CASA documentation and web resources

http://casa.nrao.edu

CASA documentation and web resources

- CASAguides Wiki contains fully annotated scripts & screen shots

Welcome to CASA Guides

CASA (Common Astronomy Software Applications) is a comprehensive software package to calibrate, image, and analyze radioastronomical data from interferometers (such as ALMA and EVLA, both shown below) as well as single dish telescopes. This wiki provides examples and hints for reducing data in CASA.


CASA News

http://casaguides.nrao.edu

- 29 October 2015: CASA 4.5.0 is now available
- 19 October 2015: CASA Newsletter #2

Events

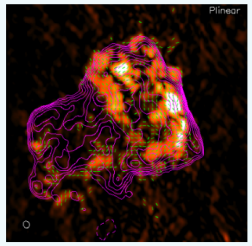
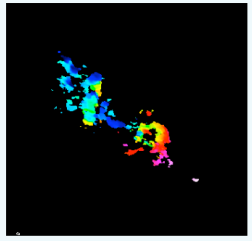
- 27-29 January 2016 ALMA Data Reduction Party
- 14-18 March 2016 VLA Data Reduction Workshop

Using CASA

- CASA Basics
  - CASA Homepage Information on the latest releases, documentation, and support
  - CASA mailing lists Please subscribe to receive information on releases, critical bugs, etc.
  - Installing CASA Where to obtain CASA, and how to install it in different operating systems
  - Overviews
    - Guide to CASA syntax, task execution, and scripting
    - CASA calibration, imaging, and a description of basic tasks
    - CASA Python Overview Includes basics of python, and guides to arrays and plotting
- CASA Documentation
  - CASA Reference Manual & Cookbook HTML and the PDF Version
  - CASA Task Reference
  - CASA Toolkit Manual
- Hints, Tips, & Tricks Task-specific tutorials to use CASA like a pro
  - Selecting Spectral Windows and Channels
  - CASA Region Format (CASA 3.3+)
  - Data flagging with plotms
  - Data flagging with viewer

CASA Tutorials

- **ALMA Guides/Tutorials**
- **Karl G. Jansky VLA Tutorials**
- **Simulating Observations**
- **'Old' VLA Tutorials**
- **CARMA Tutorials**
- **SMA Tutorials**
- **Extracting Scripts from Tutorials**

Useful Links

Starting CASA

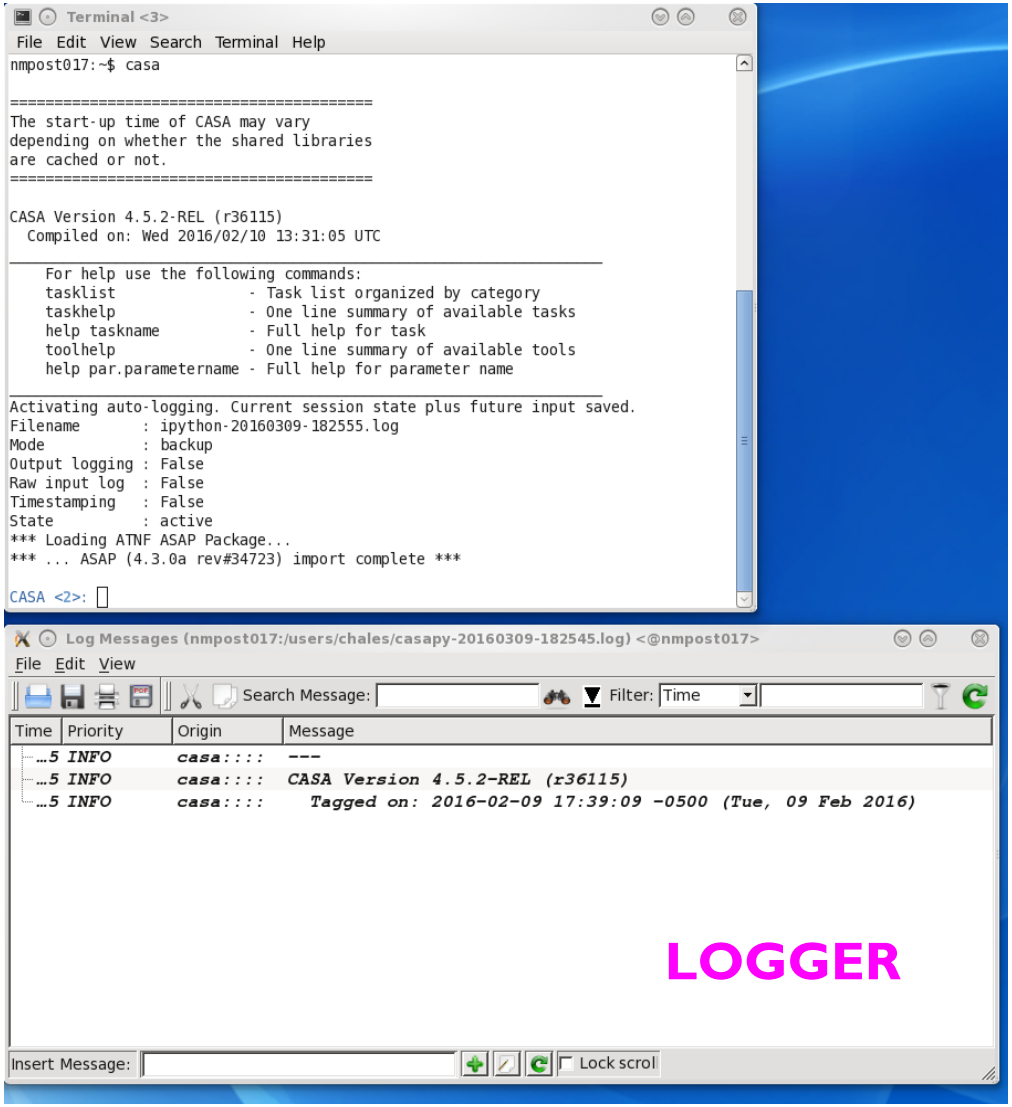
- Start CASA from the UNIX shell:

  % casa

- Session logging:
  - **ipython-TIMESTAMP.log**

    iPython command history
  - **casapy-TIMESTAMP.py**

    CASA logger messages



CASA interactive interface

- iPython interface (ipython.org) provides:
  - Numbered input/output
  - Shell access with leading exclamation mark, e.g. **!pwd** (or **os.system**)
  - Tab auto-completion
  - Auto-parenthesis
  - Command history (up-arrow or **hist [-n]**)
  - History/searching (start typing then use up-arrow, or use Ctrl-r)

- Some python tips:
  - Indentation matters, used for loops & conditions (beware copy paste)
  - Indices start from 0 and run to n-1

CASA tasks, tools, and applications

TASKS

- High-level functionality (set parameters and press go)
- These are what you will probably use most

TOOLS

- Provide access to complete functionality of CASA
- Used internally by tasks
- Sometimes shown in tutorial scripts

APPLICATIONS

- Typically used to view data (tables, images)
- Can be invoked inside CASA or as standalone programs

Find the right task

To see an organized list, type:

> tasklist

```
Terminal <3>
File  Edit  View  Search  Terminal  Help
CASA <18>: tasklist
---------> tasklist()
Available tasks, organized by category (experimental tasks in parenthesis ()
  deprecated tasks in curly brackets {}).


Import/export       Information         Editing             Manipulation
------------------  ------------------  ------------------  ------------------
exportasdm          imhead              fixplanets          concat
exportfits          imreframe           fixvis              conjugatevis
exportuvfits        imstat              flagcmd             cvel
importasdm          imval               flagdata            fixvis
importfits          listcal             flagmanager         hanningsmooth
importfitsidi       listfits            msview              imhead
importmiriad        listhistory         plotms              msmoments
importuvfits        listobs                                 mstransform
importvla           listpartition                           partition
(importevla)        listvis                                 plotms
(importgmrt)        plotms                                  split
                    plotuv                                  testconcat
                    vishead                                 uvcontsub
                    visstat                                 virtualconcat
                    (asdmsummary)                           vishead
                    (listsdm)                               (cvel2)
                    (makemask)                              (hanningsmooth2)
                                                            (split2)
                                                            (statwt)
                                                            (uvcontsub3)

Calibration         Modeling            Imaging             Analysis
------------------  ------------------  ------------------  ------------------
accum               predictcomp         clean               imcollapse
applycal            setjy               deconvolve          imcontsub
bandpass            uvcontsub           feather             imfit
blcal               uvmodelfit          ft                  imhead
calstat             uvsub               imcontsub           immath
clearcal            (uvcontsub3)        (boxit)             immoments
delmod                                  (csvclean)          impbcor
fixplanets                              (tclean)            impv
fluxscale                               (widebandpbcor)     imrebin
ft                                      {mosaic}            imreframe
gaincal                                 {widefield}         imregrid
gencal                                                      imsmooth
initweights                                                 imstat
listcal                                                     imsubimage
plotants                                                    imtrans
plotbandpass                                                imval
```

Find the right task

To see short summaries, type:

> taskhelp

Terminal <3>

File Edit View Search Terminal Help

```
CASA <12>: taskhelp
---------> taskhelp()
Available tasks:

accum          : Accumulate incremental calibration solutions into a calibration table
applycal       : Apply calibrations solutions(s) to data
asdmsummary    : Summarized description of an ASDM dataset.
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
caltabconvert  : Convert old-style caltables into new-style caltables.
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
cvel2          : Regrid an MS or MMS to a new spectral window, channel structure or frame
deconvolve     : Image based deconvolver
delmod         : Deletes model representations in the MS
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:
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-provided direction or POINTING ta
fixvis         : Recalculates (u, v, w) and/or changes Phase Center
flagcmd        : Flagging task based on batches of flag-commands
flagdata       : All-purpose flagging task based on data-selections and flagging modes/algorithms.
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  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
hanningsmooth2 : 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 pattern.
importasdm     : Convert an ALMA Science Data Model observation into a CASA visibility file (MS) or sin
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
importgmrt     : Convert a UVFITS file to a CASA visibility data set
importmiriad   : Convert a Miriad visibility file into a CASA MeasurementSet
importuvfits   : Convert a UVFITS file to a CASA visibility data set
```

Task interface

Inspect task inputs:

> inp clean

Black: default value

Red: invalid value

Blue: non-default value

Reset defaults:

> default clean

Grey: expandable

Terminal <3>

File Edit View Search Terminal Help

```
CASA <67>: inp clean
---------> inp(clean)
#  clean :: Invert and deconvolve images with selected algorithm
vis                 =          ''        #  Name of input visibility file
imagename           =          ''        #  Pre-name of output images
outlierfile         =          ''        #  Text file with image names, sizes, centers for outliers
field               =          ''        #  Field Name or id
spw                 =          ''        #  Spectral windows e.g. '0~3', '' is all
selectdata          =        True        #  Other data selection parameters
     timerange      =          ''        #  Range of time to select from data
     uvrange        =          ''        #  Select data within uvrange
     antenna        =          ''        #  Select data based on antenna/baseline
     scan           =          ''        #  Scan number range
     observation    =          ''        #  Observation ID range
     intent         =          ''        #  Scan Intent(s)

mode                =   'burrito'        #  Spectral gridding type (mfs, channel, velocity, frequency)
gridmode            =          ''        #  Gridding kernel for FFT-based transforms, default='' None
niter               =         500        #  Maximum number of iterations
gain                =         0.1        #  Loop gain for cleaning
threshold           =    '5.0mJy'        #  Flux level to stop cleaning, must include units: '1.0mJy'
psfmode             =     'clark'        #  Method of PSF calculation to use during minor cycles
imagermode          =   'csclean'        #  Options: 'csclean' or 'mosaic', '', uses psfmode
     cyclefactor    =         1.5        #  Controls how often major cycles are done. (e.g. 5 for frequently)
     cyclespeedup   =          -1        #  Cycle threshold doubles in this number of iterations

multiscale          =          []        #  Deconvolution scales (pixels); [] = standard clean
interactive         =       False        #  Use interactive clean (with GUI viewer)
mask                =          []        #  Cleanbox(es), mask image(s), region(s), or a level
imsize              = [256, 256]         #  x and y image size in pixels.  Single value: same for both
cell                = ['1.0arcsec']      #  x and y cell size(s). Default unit arcsec.
phasecenter         =          ''        #  Image center: direction or field index
restfreq            =          ''        #  Rest frequency to assign to image (see help)
stokes              =         'I'        #  Stokes params to image (eg I,IV,IQ,IQUV)
weighting           =   'natural'        #  Weighting of uv (natural, uniform, briggs, ...)
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               =         0.2        #  Minimum PB level to use
usescratch          =       False        #  True if to save model visibilities in MODEL_DATA column
allowchunk          =       False        #  Divide large image cubes into channel chunks for deconvolution

CASA <68>:
```

Task interface

Grey: expandable

Green: sub-parameter

```
Terminal <3>
File  Edit  View  Search  Terminal  Help
CASA <70>: inp
---------> inp()
#  clean :: Invert and deconvolve images with selected algorithm
vis                 =          ''        #  Name of input visibility file
imagename           =          ''        #  Pre-name of output images
outlierfile         =          ''        #  Text file with image names, sizes, centers for outliers
field               =          ''        #  Field Name or id
spw                 =          ''        #  Spectral windows e.g. '0~3', '' is all
selectdata          =        True        #  Other data selection parameters
     timerange      =          ''        #  Range of time to select from data
     uvrange        =          ''        #  Select data within uvrange
     antenna        =          ''        #  Select data based on antenna/baseline
     scan           =          ''        #  Scan number range
     observation    =          ''        #  Observation ID range
     intent         =          ''        #  Scan Intent(s)

mode                =  'burrito'         #  Spectral gridding type (mfs, channel, velocity, frequency)
gridmode            =          ''        #  Gridding kernel for FFT-based transforms, default='' None
niter               =         500        #  Maximum number of iterations
gain                =         0.1        #  Loop gain for cleaning
threshold           =    '5.0mJy'        #  Flux level to stop cleaning, must include units: '1.0mJy'
psfmode             =     'clark'        #  Method of PSF calculation to use during minor cycles
imagermode          =   'csclean'        #  Options: 'csclean' or 'mosaic', '', uses psfmode
     cyclefactor    =         1.5        #  Controls how often major cycles are done. (e.g. 5 for frequently)
     cyclespeedup   =          -1        #  Cycle threshold doubles in this number of iterations

multiscale          =          []        #  Deconvolution scales (pixels); [] = standard clean
interactive         =       False        #  Use interactive clean (with GUI viewer)
mask                =          []        #  Cleanbox(es), mask image(s), region(s), or a level
imsize              = [256, 256]         #  x and y image size in pixels.  Single value: same for both
cell                = ['1.0arcsec']      #  x and y cell size(s). Default unit arcsec.
phasecenter         =          ''        #  Image center: direction or field index
restfreq            =          ''        #  Rest frequency to assign to image (see help)
stokes              =         'I'        #  Stokes params to image (eg I,IV,IQ,IQUV)
weighting           =    'briggs'        #  Weighting of uv (natural, uniform, briggs, ...)
     robust         =         0.0        #  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
pbcor               =       False        #  Output primary beam-corrected image
minpb               =         0.2        #  Minimum PB level to use
usescratch          =       False        #  True if to save model visibilities in MODEL_DATA column
allowchunk          =       False        #  Divide large image cubes into channel chunks for deconvolution

CASA <71>:
```

Task help

Type:

> help clean

Terminal <3>

File Edit View Search Terminal Help

```
 Example :

The clean task has many options:

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


vis -- Name(s) of input visibility file(s)
        default: none;
        example: vis='ngc5921.ms'
                 vis=['ngc5921a.ms','ngc5921b.ms']; multiple MSes
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.flux.pbcoverage;  relative pb coverage over field
                               (gets created only for ft='mosaic')
          m2.model; image of clean components
          m2.residual; image of residuals
          m2.interactive.mask; image containing clean regions
        To include outlier fields:
          imagename=['n5921','outlier1','outlier2']
outlierfile --- Text file name which contains image names, sizes, field
                centers (See 'HINTS ON CLEAN WITH FLANKING FIELDS' below
                for the format of this outlier file.)
field -- Select fields to image or mosaic.  Use field id(s) or 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 to
          be a field index otherwise, it is assumed to be 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
          For multiple MS input, a list of field strings can be used:
          field = ['0~2','0~4']; field ids 0-2 for the first MS and 0-4
lines 301-348
```

How to run a task

```
> inp listobs
> vis = 'mydata.ms'
> listfile = 'outfile.txt'
> inp
> go
```

- Task interface
  - Use **inp taskname** to see list of parameters
  - Set (global) parameters one at a time
  - Useful for interactive work, exploring parameters
  - Recover previous parameters using **tget taskname** (antonym **tput**)

- iPython command line

```
> listobs(vis='mydata.ms',listfile='outfile.txt')
```

  - Set all parameters at once
  - Useful for scripting
    - Copy-paste into a text or .py file to keep record of processing

```
> execfile('commands.py')
```

Some things to note about running tasks

- Some tasks return a dictionary

```
> results = imstat()
```

- Command line execution will ignore global parameters
  - Unspecified parameters will be set to their default values

```
> field='3C286'
> listobs(vis='mydata.ms',listfile='outfile.txt')
```

field will be ignored here (set to default)

  - Exception is if global parameter is explicitly referenced

```
> field='3C286'
> listobs(vis='mydata.ms',listfile='outfile.txt',field=field)
```

Tools

- What if there's no task?

  → Use CASA tools!

- Tools (and their methods) are the building blocks of tasks
  - Contain full functionality of CASA
  - Used internally by tasks
  - E.g. imaging utilities (im), table utilities (tb), …

Find the right tool

To see short summaries, type:

> toolhelp

```
Terminal <3>
File  Edit  View  Search  Terminal  Help
CASA <74>: toolhelp
---------> toolhelp()

Available tools:

 af : Agent flagger utilities
 at : Juan Pardo ATM library
 ca : Calibration analysis utilities
 cb : Calibration utilities
 cl : Component list utilities
 cp : Cal solution plotting utilities
 cs : Coordinate system utilities
 cu : Class utilities
 dc : Deconvolver utilities
 fi : Fitting utilities
 fn : Functional utilities
 ia : Image analysis utilities
 im : Imaging utilities
lm: linear mosaic
 me : Measures utilities
 ms : MeasurementSet (MS) utilities
 msmd : MS metadata accessors
 mt : MS transformer utilities
 qa : Quanta utilities
 pm : PlotMS utilities
 po : Imagepol utilities
 rg : Region manipulation utilities
 sl : Spectral line import and search
 sm : Simulation utilities
 tb : Table utilities (selection, extraction, etc)
 tp : Table plotting utilities
 vp : Voltage pattern/primary beam utilities
 ---
 pl : pylab functions (e.g., pl.title, etc)
 sd : Single dish utilities
 ---

CASA <75>:
```

Using tools and their methods

- Tools contain a number of methods
  - Access using **tool.method()**
  - Use tab-completion to see listing

- Typically, data must be opened and closed (unlike tasks)
  - Failure to close may block other tasks and clutter memory

```
> ia.open('image.im')
> ia.regrid(outfile='out.im',…)
> ia.close()
```

CASA toolkit reference manual

- There's a good chance your problem can be solved on the tool level, don't be afraid to use this resource!

- >1000 tool methods available

- See CASA toolkit reference manual:

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

Still searching for functionality?

- Look through contributed scripts and tasks at:

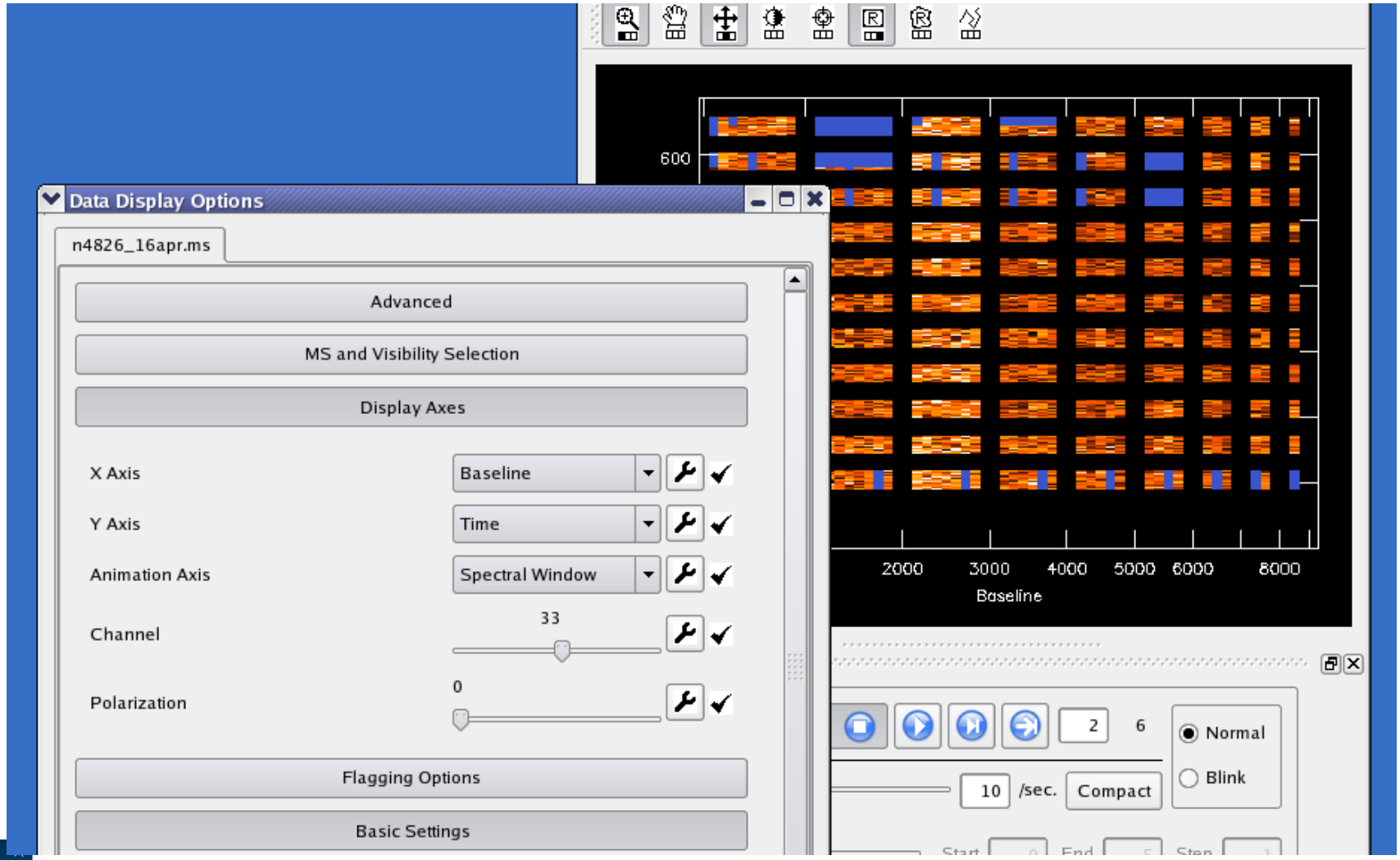
  http://casaguides.nrao.edu/

- If you still can't find what you need, write your own task!
  - Combination of Python plus CASA toolkit is very powerful
  - We encourage you to submit your own scripts to us
  - See instructions at link above

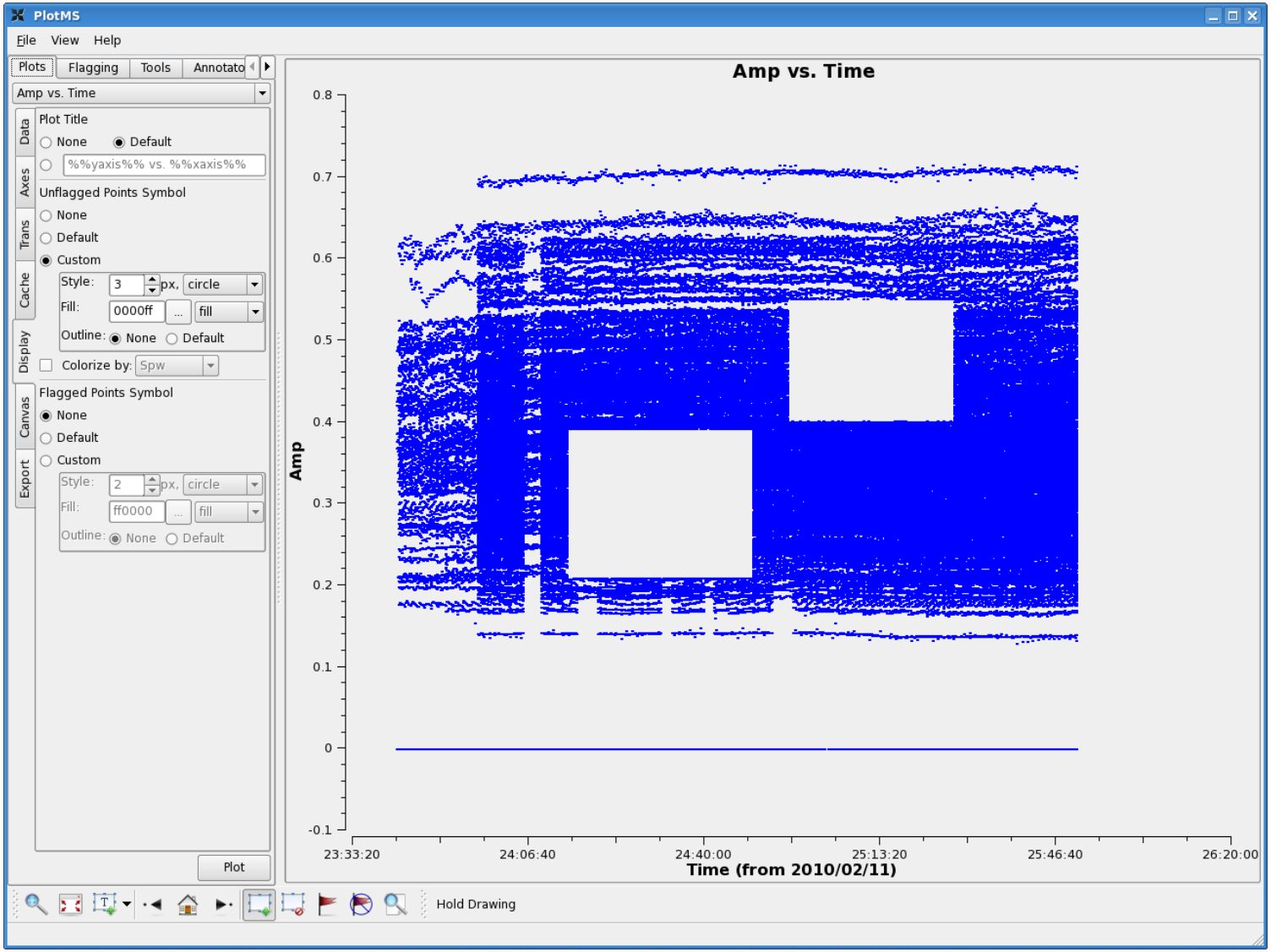
Applications

- Typically used to display data (tables, images)
  - Some editing capabilities
- Can be invoked inside CASA or as standalone programs from shell
- Visibilities: **msview**, **plotms** (standalone **% casaplotms**)
- Any table data: **browsetable** (standalone **% casabrowser**)
- Calibration tables: **plotcal** (eventually **plotms**)
- Images: **imview**, **viewer** (standalone **% casaviewer**)
- Single dish: **sdplot**
- Don't forget about full functionality of python! e.g. matplotlib

MSView


Data Display Options
n4826_16apr.ms
Advanced
MS and Visibility Selection
Display Axes
X Axis
Baseline
Y Axis
Time
Animation Axis
Spectral Window
Channel
33
Polarization
0
Flagging Options
Basic Settings
600
2000
3000
4000
5000
6000
8000
Baseline
2
6
Normal
Blink
10
/sec.
Compact

PlotMS


PlotMS
File View Help
Plots Flagging Tools Annotato
Amp vs. Time
Data
Axes
Trans
Cache
Display
Canvas
Export
Plot Title
None Default
%%yaxis%% vs. %%xaxis%%
Unflagged Points Symbol
None
Default
Custom
Style: 3 px, circle
Fill: 0000ff ... fill
Outline: None Default
Colorize by: Spw
Flagged Points Symbol
None
Default
Custom
Style: 2 px, circle
Fill: ff0000 ... fill
Outline: None Default
Plot
Amp vs. Time
Amp
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
23:33:20
24:06:40
24:40:00
25:13:20
25:46:40
26:20:00
Time (from 2010/02/11)
Hold Drawing

Plotcal

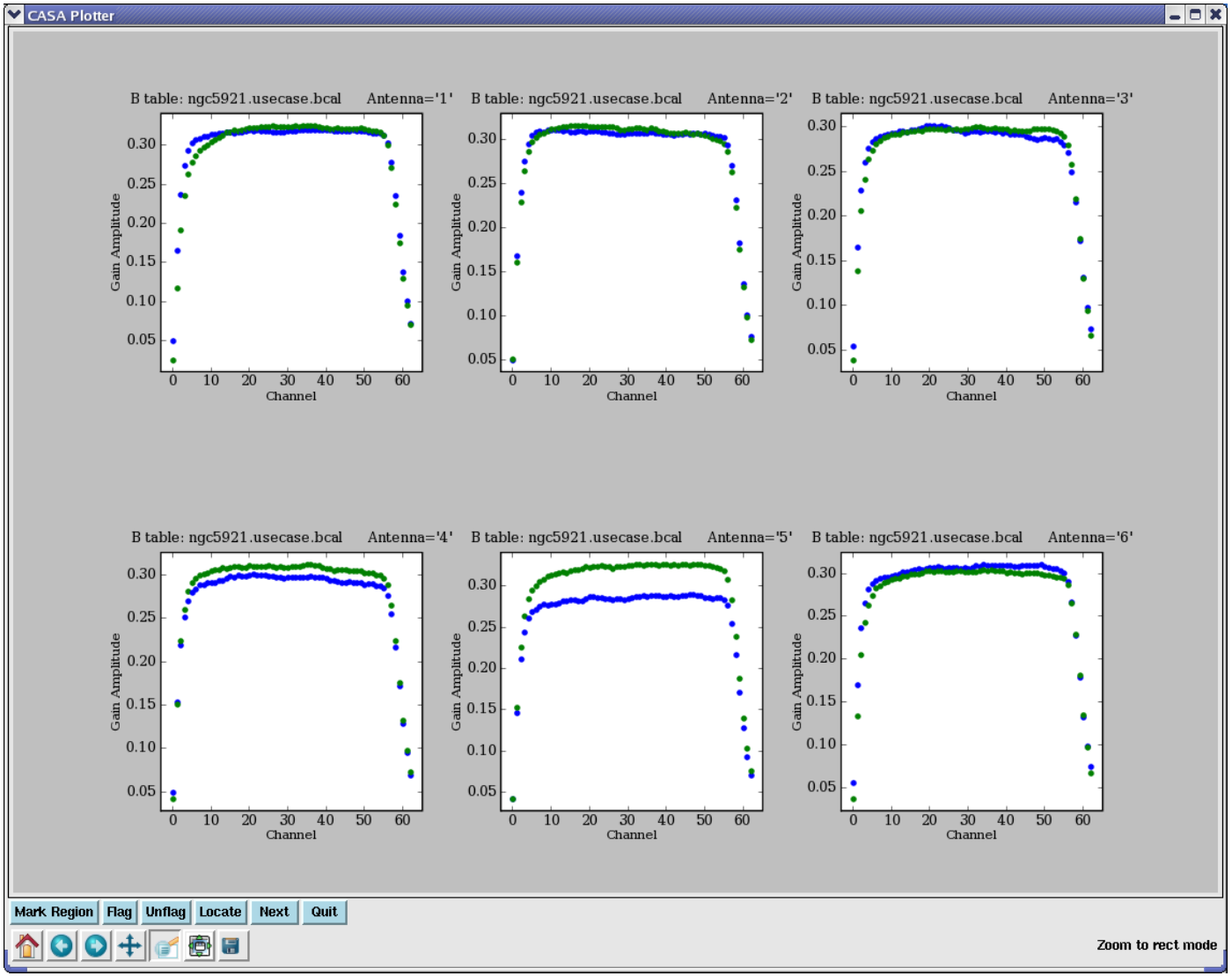

CASA Plotter
B table: ngc5921.usecase.bcal Antenna='1'
B table: ngc5921.usecase.bcal Antenna='2'
B table: ngc5921.usecase.bcal Antenna='3'
B table: ngc5921.usecase.bcal Antenna='4'
B table: ngc5921.usecase.bcal Antenna='5'
B table: ngc5921.usecase.bcal Antenna='6'
Gain Amplitude
Channel
Mark Region
Flag
Unflag
Locate
Next
Quit
Zoom to rect mode

Image Viewer

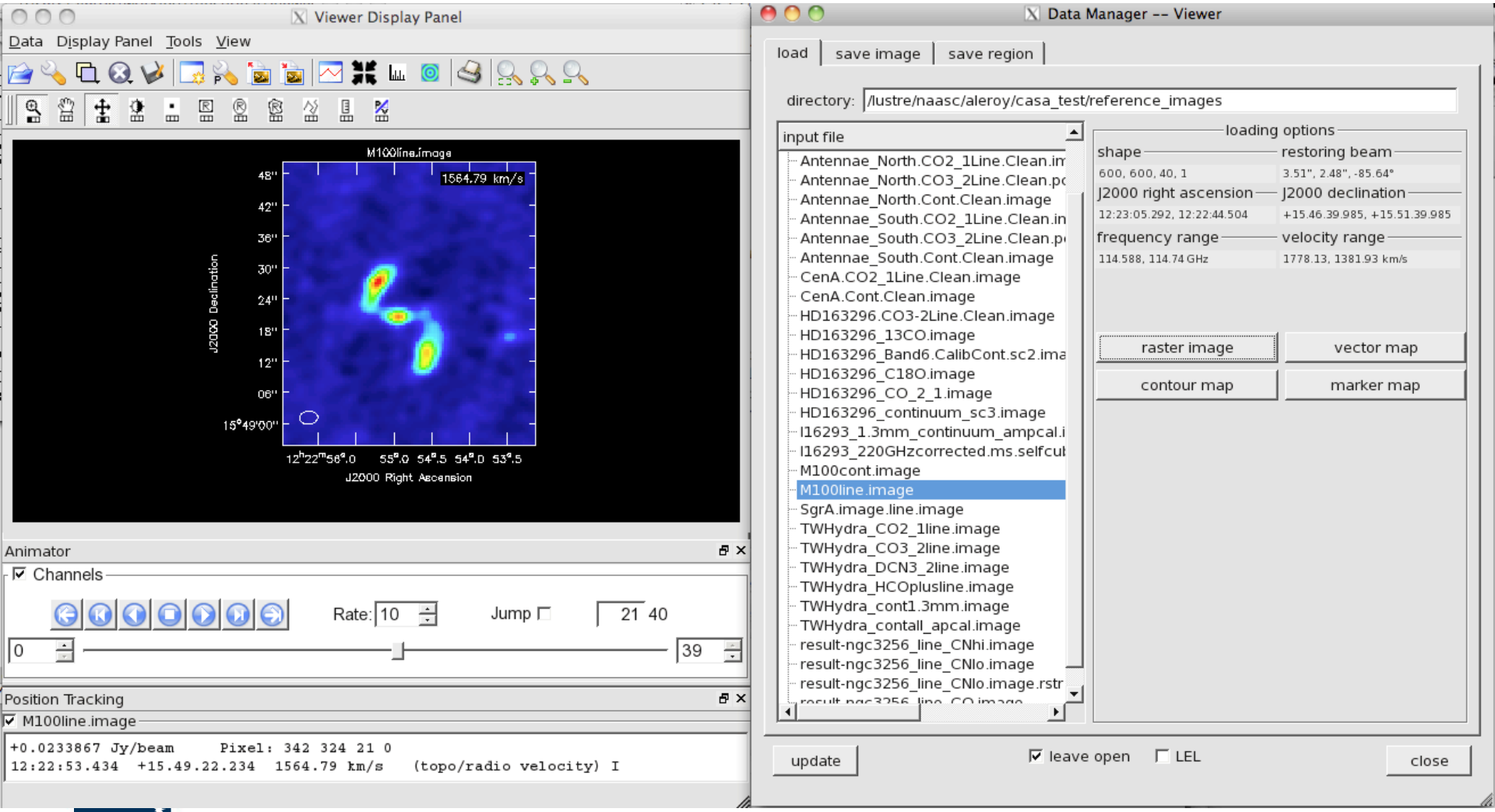
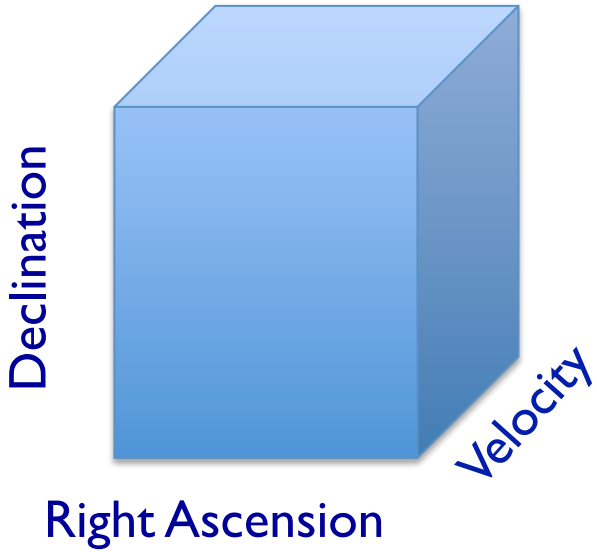


Image Viewer

- Cubes
- Movies
- Channel maps



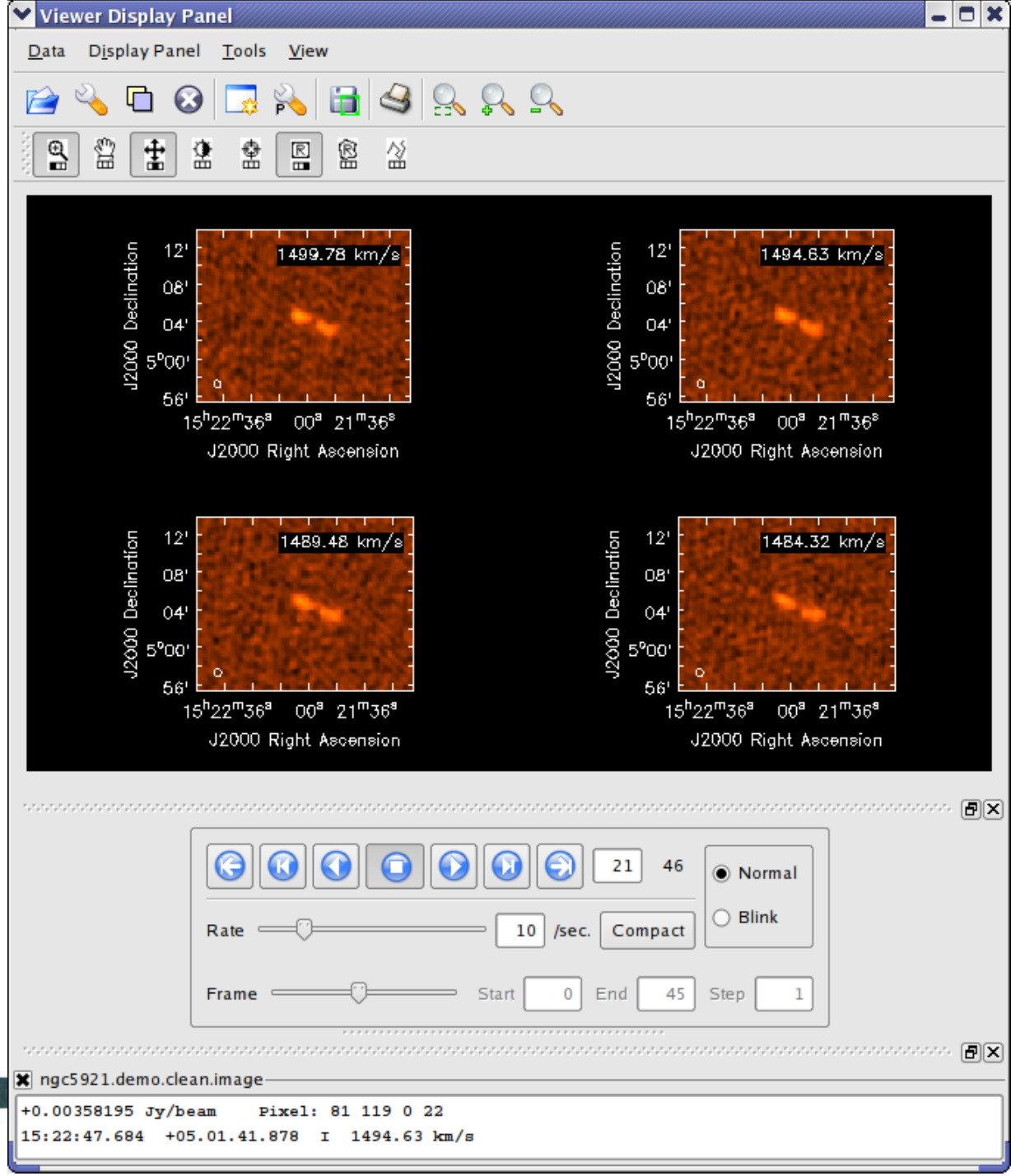
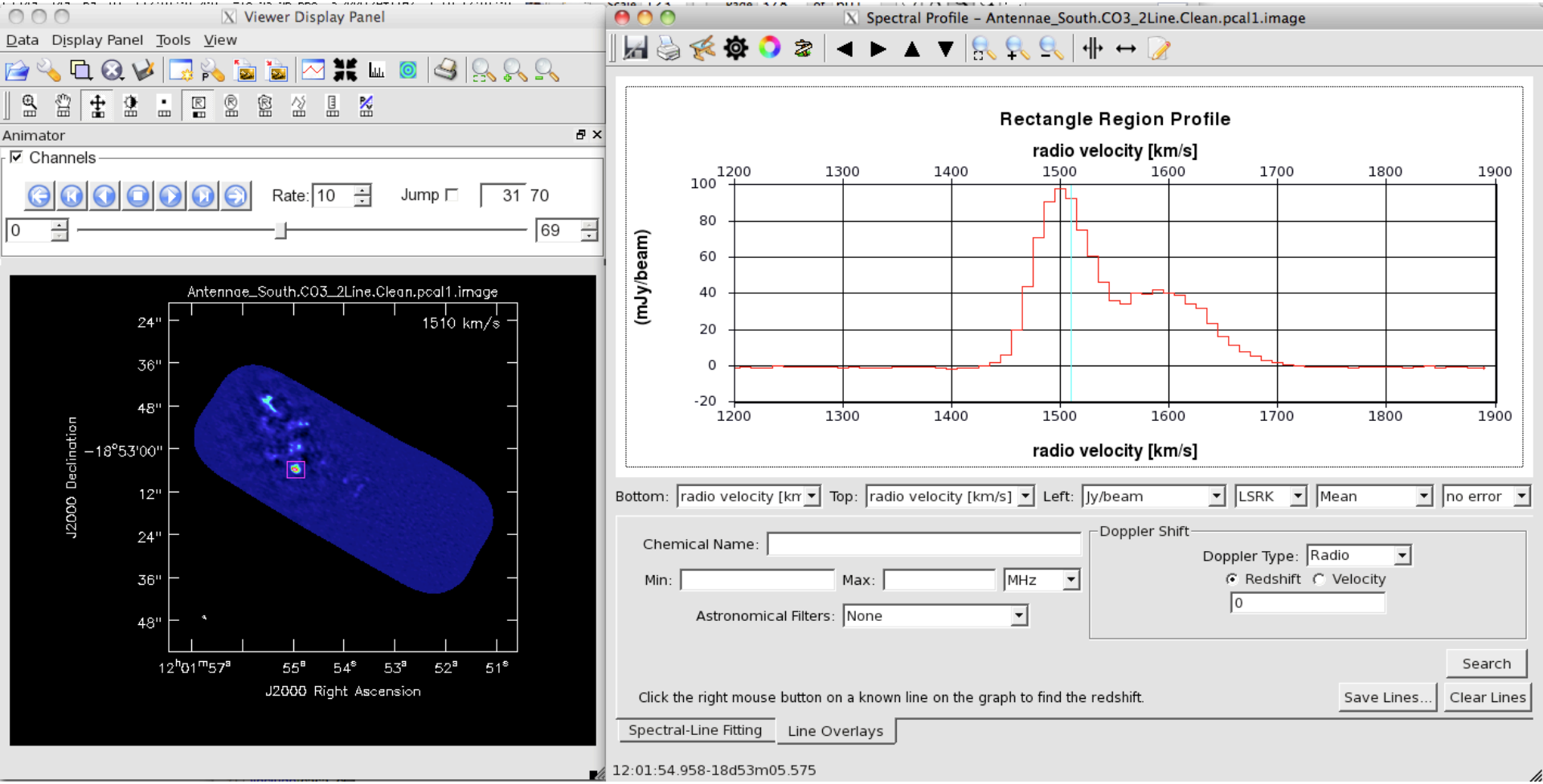
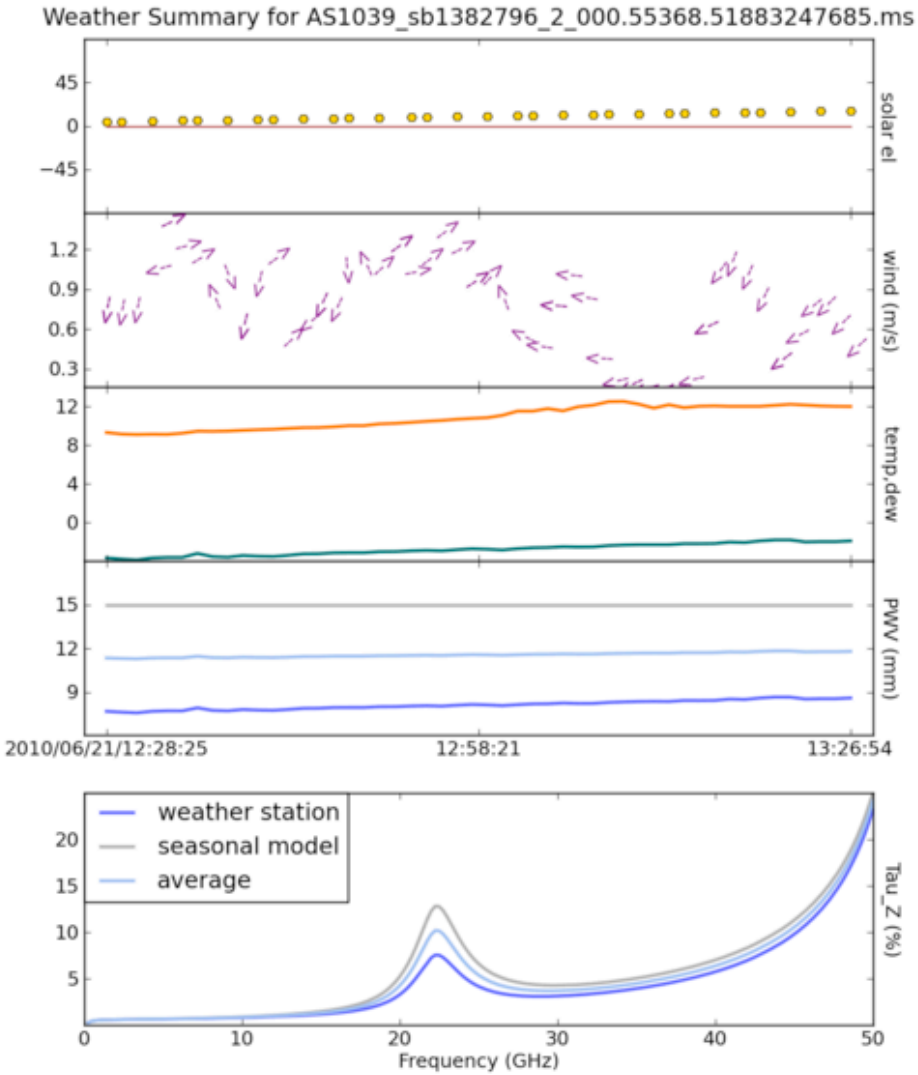


Image Viewer



Plot anything - matplotlib



Data structures

- Observatory data stored in (A)SDM format, CASA uses measurement sets
  - Use **importasdm** for ALMA, **importevla** for EVLA/JVLA, etc.

- Measurement set contains visibilities
  Calibration information stored in calibration tables (not in ms)
  Images are in CASA image format

- All of these are *directories* which contain the necessary information
  - Copying requires recursive option (**!cp –r**)

- Delete tables using **rmtables('mydata.ms')**
  !rm –rf or **os.system('rm –rf mydata.ms')** may also work,
  but could leave traces in the cache

Inspect a measurement set (ms)

- Contains visibilities (and flags) stored in MAIN table within table.* files

```
CASA <80>: !ls amazing_data.ms
ANTENNA           POINTING         SYSPOWER   table.f15       table.f20_TSM0  table.f24_TSM1  table.f8
CALDEVICE         POLARIZATION     table.dat  table.f16       table.f21       table.f25       table.f9
DATA_DESCRIPTION  PROCESSOR        table.f1   table.f17       table.f21_TSM1  table.f25_TSM1  table.info
FEED              SORTED_TABLE     table.f10  table.f17_TSM1  table.f22       table.f3        table.lock
FIELD             SOURCE           table.f11  table.f18       table.f22_TSM1  table.f4        WEATHER
FLAG_CMD          SPECTRAL_WINDOW  table.f12  table.f19       table.f23       table.f5
HISTORY           STATE            table.f13  table.f2        table.f23_TSM1  table.f6
OBSERVATION       SYSCAL           table.f14  table.f20       table.f24       table.f7
```

- Also contains sub-tables, e.g. FIELD, SOURCE, WEATHER, …

```
CASA <81>: !ls amazing_data.ms/FIELD
table.dat  table.f0  table.f0i  table.info  table.lock
```

MS MAIN table contents

Inspect with task **browsetable** (or standalone **% casabrowser**)

Table Browser <@nmpost017>

File Edit View Tools Export Help About

amazing_data.ms

table data | table keywords | field keywords

	UVW	FLAG	FLAG_CATEGORY	WEIGHT	SIGMA	ANTENNA1	ANTENNA2	ARRAY_ID	DA
0	[-278.403, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	1	0	0
1	[2810.11, 2...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	2	0	0
2	[426.12, 71...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	3	0	0
3	[270.096, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	4	0	0
4	[-610.975, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	5	0	0
5	[-1908.48, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	6	0	0
6	[141.022, 1...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	7	0	0
7	[712.44, 94...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	8	0	0
8	[-20.6492, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	9	0	0
9	[989.709, ...	[4, 64] Boo...	[0, 0, 0] Boolean	[8.53333e...	[0.000176...	0	10	0	0

Restore Columns | Resize Headers

PAGE NAVIGATION First << [1 / 3633] >> Last 1 Go Loading 1000 rows.

MS MAIN table contents

Inspect with task **browsetable** (or standalone **% casabrowser**)

Table Browser <@nmpost017>

File Edit View Tools Export Help About

amazing_data.ms

table data | table keywords | field keywords

	TIME_CENTROID	DATA	WEIGHT_SPECTRUM	MODEL_DATA	CORRECTED_DATA
0	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
1	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
2	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
3	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
4	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
5	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
6	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
7	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
8	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex
9	2015-04-19-22:...	[4, 64] Complex	[0, 0] Float	[4, 64] Complex	[4, 64] Complex

Restore Columns | Resize Headers

amazing_data.ms[0, 21] = Complex Array of size [4 64].

	0	
0	(1.38879e-05,0.00067147)	(-6.54117e-0
1	(3.43195e-05,0.000646329)	(0.00045081
2	(1.56872e-05,-0.000113082)	(0.00013204,
3	(-0.000342448,0.000368312)	(-0.00042331

Close | Close All

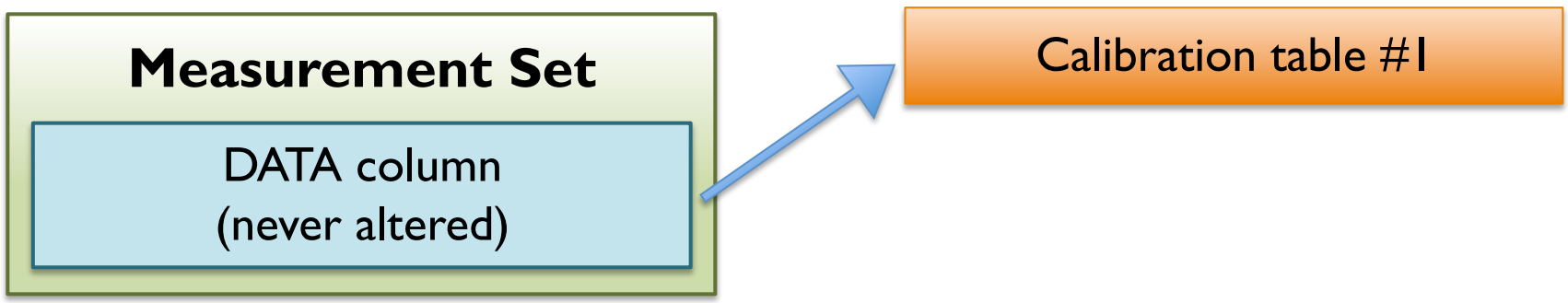
PAGE NAVIGATION First << [1 / 3633] >> Last 1 Go Loading 1000 rows.

MS columns & calibration tables

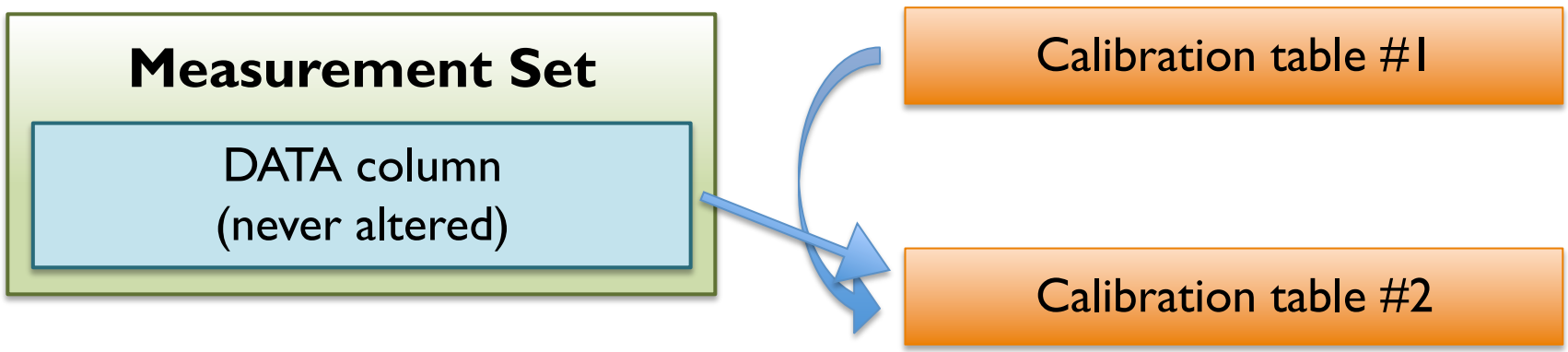
Measurement Set

DATA column
(never altered)

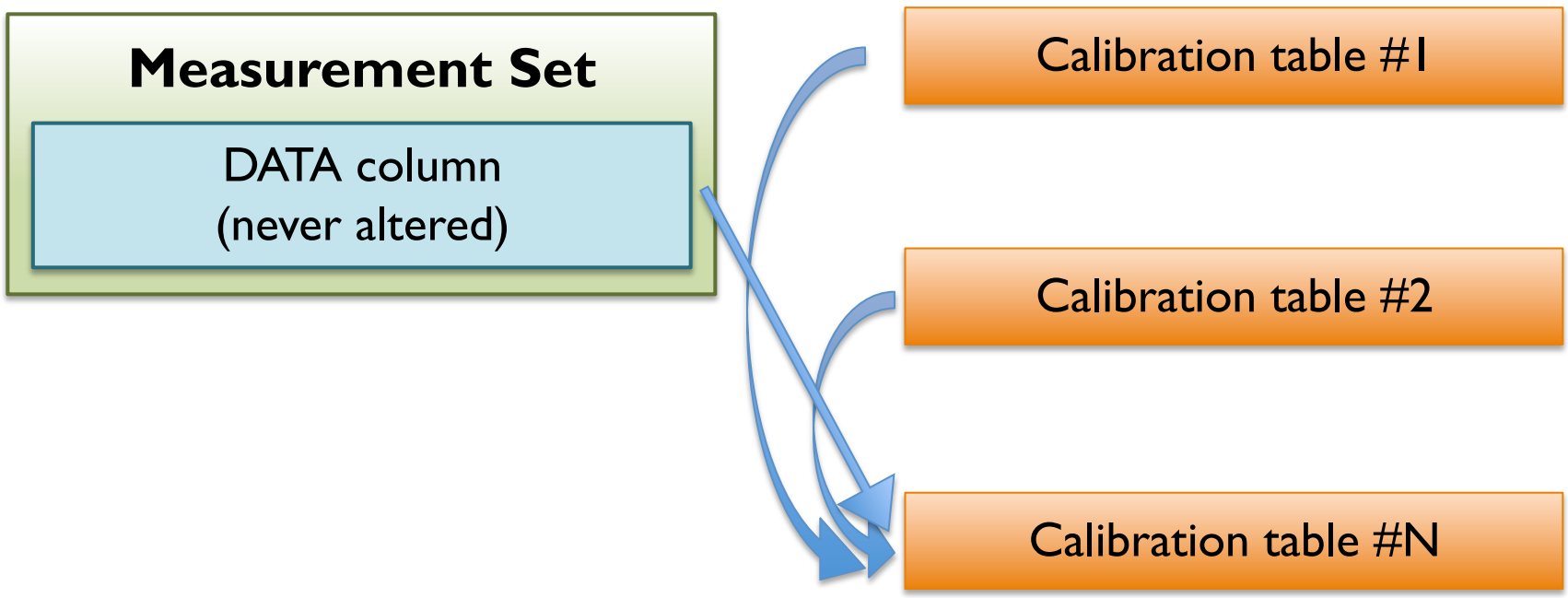
MS columns & calibration tables



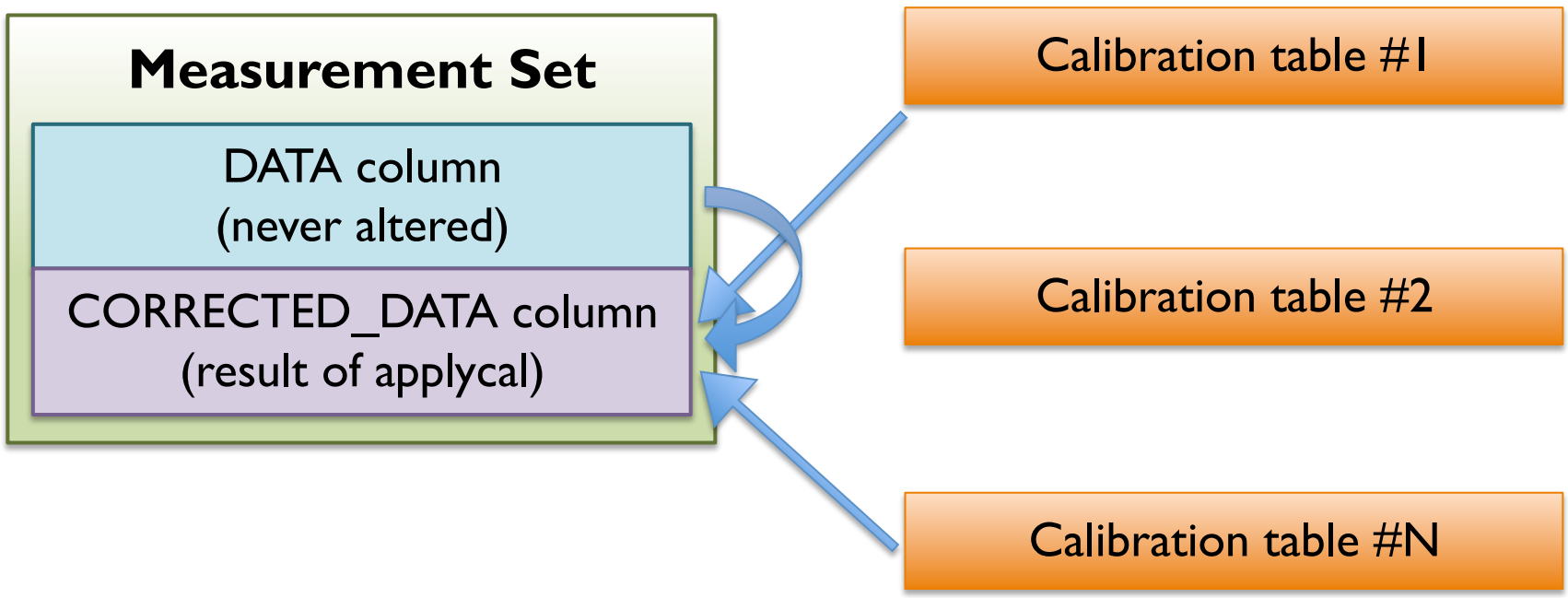
MS columns & calibration tables



MS columns & calibration tables



MS columns & calibration tables



MS columns & calibration tables

Measurement Set

DATA column
(never altered)

MODEL_DATA column
(result of fluxscale)

CORRECTED_DATA column
(result of applycal)

(note: data size tripled)

Calibration table #1

Calibration table #2

Calibration table #N

MS data selection syntax

- You can select subset of visibilities to perform actions on:
  - Antennas, baselines, frequencies, time, polarization, etc.

- The standard CASA selection syntax is the following:
  - Use tilde (~) for inclusive range, e.g. **spw='0~3'**
  - Use comma (,) for separator, e.g. **spw='0~3,7,11'**
  - Use colon (:) for spw channelization, e,g. **spw='0:0~40,3:20~40'**
  - Use semicolon (;) for spw channel separator, e.g. **spw='0:0~10;20;25'**
  - Use asterisk (*) for wildcard, e.g. **field='3C*'**
  - Use exclamation mark (!) for omission, e.g. **antenna='!ea05'**
  - Use less than (<) or greater than (>) for selection, e.g. **uvrange='<1000m'**

- For full syntax details type **help par.selectdata** or see:

  http://www.nrao.edu/~sbhatnag/misc/msselection/msselection.html

MS data selection syntax: Examples

- **field** (*spatial*)
  - String with source name or field ID (checks former 1st)
    - Beware field names that are integers
  - Examples: **field='1331+305'** ; **field='3C*'** ; **field='0,1,4~5'**

- **spw** (*spectral*)
  - String with spectral window ID plus channels
  - Examples: **spw='0:10~20;45,4~5:35~45;50~70'** ; **spw='*:10~80'** ;
    spw='1421Mhz:10~20;50,5:1.6~1.7GHz'

MS data selection syntax: Examples

- **timerange** (*spatial*)
  - String with date/time range in format T0~T1
  - Can give T0+dT, where missing parts of T1 default to T0
  - Example: **timerange = '2014/10/21/01:00:00~06:30:00'**

- **antenna**
  - String with antenna name or ID (checks former 1st)
    - Beware VLA name 1-27, these have ID's 0-26
  - & = CC only , && = CC+AC , &&& = AC only
  - Examples: **antenna = '1~5,8'** ; **antenna='!ea01&ea10'** ;
    antenna='ea05&&&'

MS data selection syntax: Examples

- **scan** – the scan numbers (an execution sequence)
  - e.g. **scan='3~14'**
- **correlation** – polarization products
  - e.g. **correlation='RR,LL,LR'**
- **uvrange** – select on uv range
  - e.g. **uvrange='30~3000m'** ; **uvrange='<1000m'**

Need help?

- Helpdesk: https://help.nrao.edu
- Questions can cover:
  - CASA problems
  - Calibration/imaging questions
  - Submit bug reports and suggestions
  - (Other issues, e.g. account/log-in info, proposal submission, etc.)
- When submitting a ticket, provide as much detail as possible:
  - CASA version
  - Operating system
  - Commands entered
  - Project ID (if relevant)
  - Scripts you followed from CASAguides (if relevant)

CASA documentation and web resources

http://casa.nrao.edu

- Installation details
- CASA reference manual and cookbook (large but detailed!)
- CASAguides (tutorials)
- Task and tool reference manuals
- Links to Python tutorials and references
- Helpdesk
- NRAO science user forum
- Sign up to mailing lists to receive updates

CASA
Common Astronomy Software Applications

NRAO