Introduction to CASA

Juergen Ott (CASA project scientist)
Crystal Brogan (CASA ALMA subsystem scientist)
Steven Myers (CASA EVLA subsystem scientist)
Jeff Kern (CASA manager)

CASA (Common Astronomy Software Applications)

- CASA is the offline data reduction package for ALMA and the EVLA (data from other telescopes usually work, too, but not primary goal of CASA)
- C++ bound to Python (plus some Qt or other apps)
- Import/export data, edit, calibrate, image, analyze
- Also supports single dish (based on ASAP)
- CASA has many tasks and a LOT of tool methods
- Easy to write scripts and tasks, including contributed scripts/tasks
- We have a lot of documentation, reduction tutorials, helpdesk
- CASA has some of the most sophisticated algorithms implemented (multi-scale clean, Taylor term expansion for wide bands, W-term projection, OTF mosaicing, etc.)
- We have a active Algorithm Research Group, so more goodness to come

Outline

- CASA startup
- CASA basic python interface
- Tasks and tools
- The Measurement Set
- Data selection syntax
- Visualization tools
- Data analysis
- User support/Documentation

CASA (Common Astronomy Software Applications)

- Current version: 3.3.0 (release r16856 built 2 Nov 2011)

  New releases about every 6 months (around 4/15 and 10/15).

  For download: casa.nrao.edu Linux, Mac OS X

  "release", "test" and "stable" versions available at NRAO/ESO/ALMA and via download

  > casapy - latest release: underwent lots of testing, updated documentation

  > casapy-test - cutting edge capabilities, no documentation, bugs

  > casapy-stable - less bugs but also less features, could be a release

  For the workshop we will use casapy 3.3.0 r16856

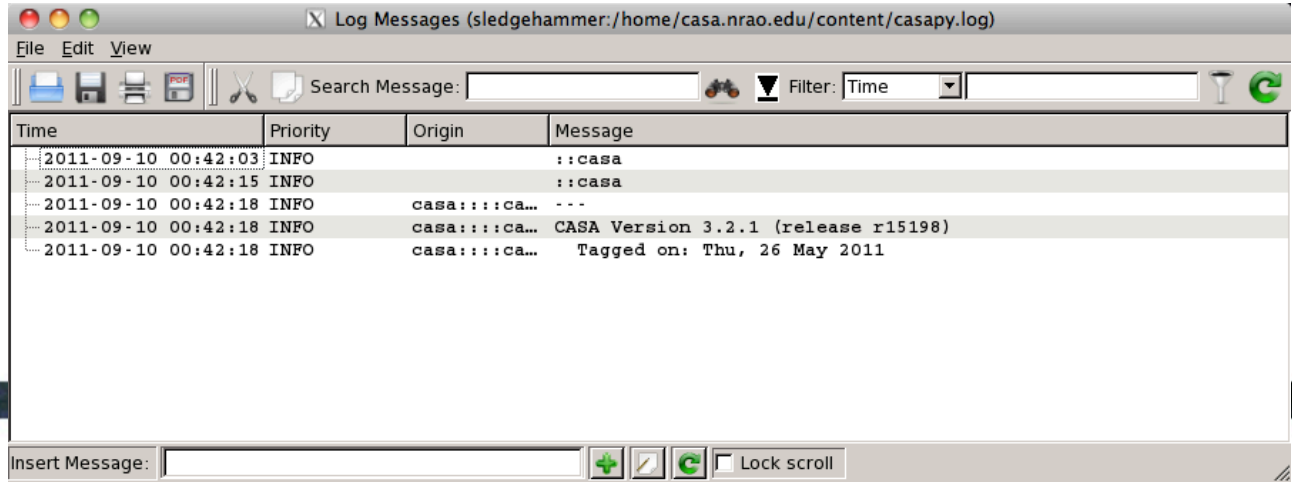
CASA Startup

$ casapy

CASA Version 3.2.1 (r15198)
Compiled on: Fri 2011/05/27 02:52:18 UTC

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 <2>:



NRAO
CASA
Common Astronomy Software Applications

CASA Interactive Interface

- IPython (ipython.org)
- Features:
  - shell access
  - auto-parenthesis (autocall)
  - command history
  - session logging
    - ipython.log – ipython command history
    - casapy.log – casa messages
  - numbered input/output
  - history/searching

Basic Python tips

- to run a .py script:

  execfile('<scriptname>')

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

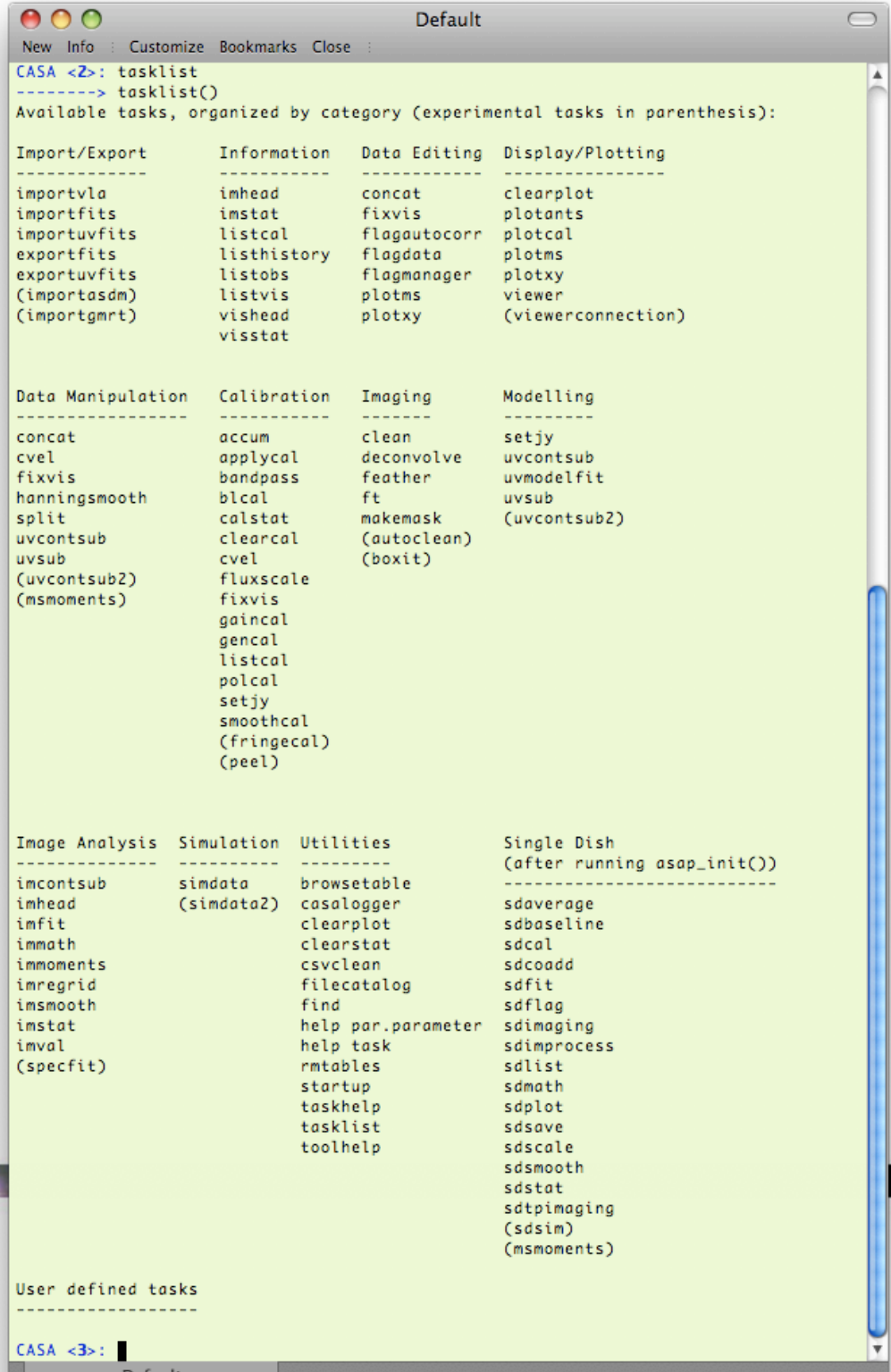
Tasks and tools in CASA

- Tasks - high-level functionality
  - function call or parameter handling interface
  - these are what you should use in tutorial
- Tools - complete functionality
  - tool.method calls, used by tasks
  - sometimes shown in tutorial scripts
- Applications – some tasks/tools invoke standalone apps
  - e.g. casaviewer, casaplotms, casabrowser, asdm2MS
- Shell commands can be run with a leading exclamation mark !du –hs

Key Tasks

- To see list of tasks organized by type:

tasklist



Key Tasks

- To see list of tasks with short help:

taskhelp

Default

New Info Customize Bookmarks Close

```
CASA <15>: taskhelp
---------> taskhelp()
Available tasks:

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:
fixvis              : Recalculates or converts (u, v, w)
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 se
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           : Subtracts specified continuum channels from a spectral line data set
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
```

Default | Default | Default | Default | Default

NRAO CASA Common Astronomy Software Applications

Task Interface

- examine task parameters with inp :

Default

New Info Customize Bookmarks Close

```
CASA <12>: 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          =      False        #  Other data selection parameters
mode                =  'channel'        #  Spectral gridding type (mfs, channel, velocity, frequency)
     nchan          =         -1        #  Number of channels (planes) in output image; -1 = all
     start          =          0        #  Begin the output cube at the frequency of this channel in the MS
     width          =          1        #  Width of output channel relative to MS channel (# to average)
     interpolation  =   'linear'        #  Spectral interpolation (nearest, linear, cubic). Use nearest for
                                        #   mode=channel
     chaniter       =      False        #  Clean each channel to completion (True), or all channels each cycle (False)
     outframe       =         ''        #  velocity frame of output image

gridmode            =         ''        #  Gridding kernel for FFT-based transforms, default='' None
niter               =        500        #  Maximum number of iterations
gain                =        0.1        #  Loop gain for cleaning
threshold           =   '0.0mJy'        #  Flux level to stop cleaning, must include units: '1.0mJy'
psfmode             =    'clark'        #  Method of PSF calculation to use during minor cycles
imagermode          =         ''        #  Options: 'csclean' or 'mosaic', '', uses psfmode
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
calready            =       True        #  True required for self-calibration
async               =      False        #  If true the taskname must be started using clean(...)

CASA <13>:
```

Default | Default | Default | Default | Default

Task Interface

- standard tasking interface
- use parameters set as global Python variables

  <param> = <value>

  (e.g. vis = 'ngc5921.demo.ms')
- parameter manipulation commands
  - inp, default, saveinputs, tget, tput
- execute

  <taskname> or go (e.g. clean())
- return values (except when using "go")
  - some tasks return Python dictionaries, e.g. myval=imval()

Task Execution

- two ways to invoke:
  - call from Python as functions with arguments

    taskname(arg1=val1, arg2=val2, ...), like

    clean(vis='input.ms', imagename='galaxy',selectvis=T, robust=0.5, imsize=[200,200])

    unspecified parameters will be defaulted (globals not used)
  - use standard tasking interface

    use global variables for task parameters
  - see Chapter 1.3 in Cookbook

Expandable Parameters

IPy:Jupiter

```
CASA <3>: tget('clean')
Restored parameters from file clean.last

CASA <4>: inp()
#  clean :: Deconvolve an image with selected algorithm
vis                 = 'ngc5921.usecase.ms.contsub' #  name of input visibility file
imagename           = 'ngc5921.usecase.clean' #  Pre-name of output images
field               =         '0'           #  Field Name
spw                 =          ''           #  Spectral windows:channels: '' is all
selectdata          =       False           #  Other data selection parameters
mode                =   'channel'           #  Type of selection (mfs, channel, velocity, frequency)
     nchan          =          46           #  Number of channels (planes) in output image
     start          =           5           #  first input channel to use
     width          =           1           #  Number of input channels to average

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

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

Parameter Checking

sanity checks of parameters in inp :

IPy:Jupiter

```
CASA <5>: psfmode='hogwarts'

CASA <6>: inp()
#  clean :: Deconvolve an image with selected algorithm
vis                 = 'ngc5921.usecase.ms.cont                sibility file
imagename           = 'ngc5921.usecase.clean' #               ages
field               =         '0'           #  Fiel
spw                 =         ''            #  Spec                '' is all
selectdata          =      False            #  Othe              eters
mode                =  'channel'            #  Type of selection (mfs, channel, velocity, frequency)
     nchan          =         46            #  Number of channels (planes) in output image
     start          =          5            #  First input channel to use
     width          =          1            #  Number of input channels to average

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

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

erroneous values in red

Help on Tasks

- In-line help:

>help clean OR >pdoc clean

IPy:Jupiter

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

NAME
    clean

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

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

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

             The main clean deconvolution task.  It contains many functio

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


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

Tools in CASA

- What if there's no task?

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

- CASA Toolkit underneath tasks
- core AIPS++ code (mostly in C++)

- tools are functions.methods
  - call from casapy as `<tool>.<method>()`
  - default tool objects are pre-constructed
    - e.g. imager (im) , calibrater (cb), ms (ms) , etc. (see toolhelp)

See Miriam's talk on Toolkit tomorrow!

The Measurement Set

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

Example MS

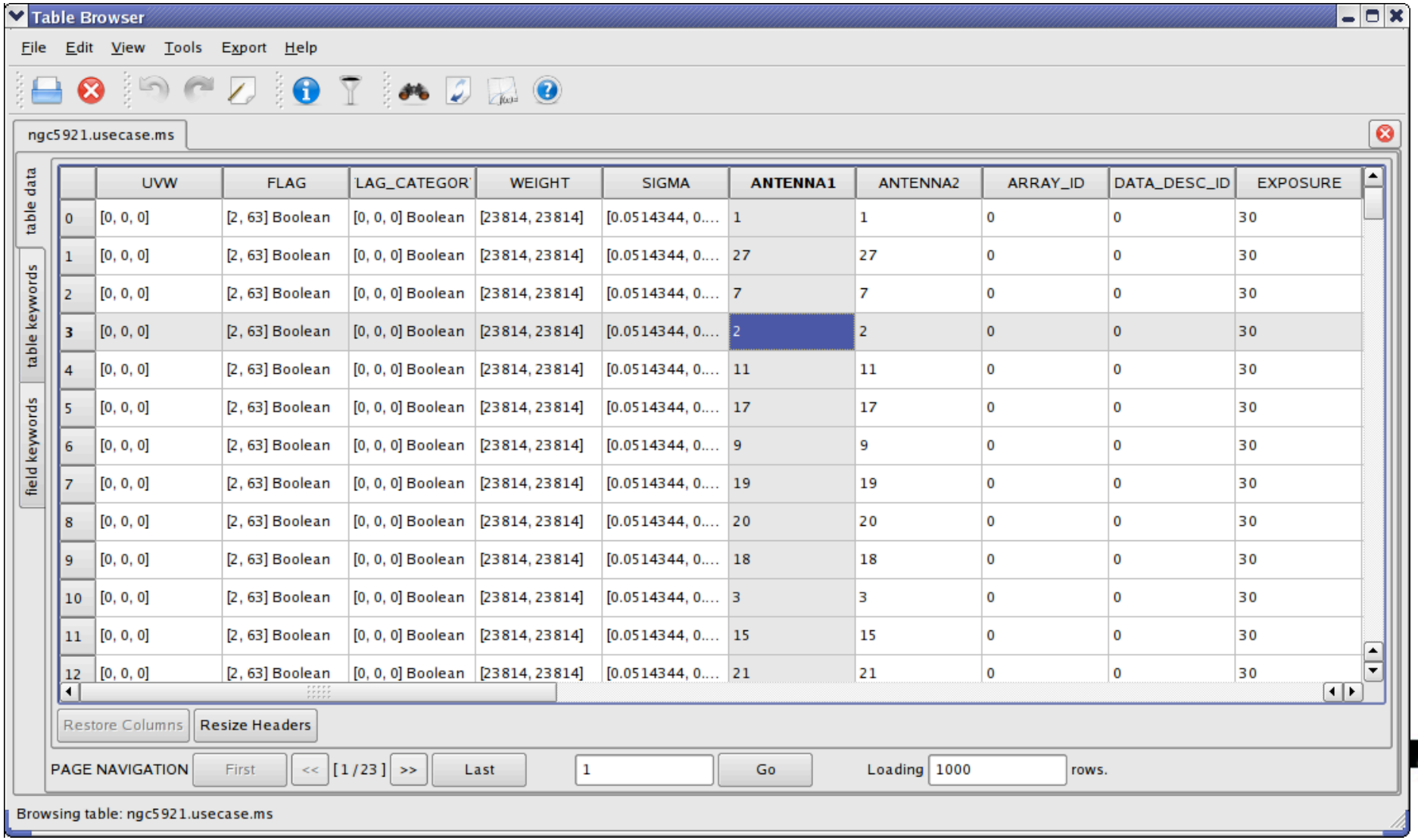
- **Example:** `ls ngc5921.usecase.ms`

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

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

MAIN Table Contents

- Example using task browsetable: (application casabrowser)



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

Data Selection Example

- standard selection parameters
  - e.g. for task gaincal:

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

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

Data Selection Syntax

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

Selection Syntax

- see Chapter 2.5 of Cookbook
  - antenna - string with antenna name or ID
    - first check for name, then ID (beware VLA name 1-27, ID 0-26)
    - example: antenna = '1~5,11' ; antenna = 'EA*', '!VA'
    - Baselines: 'EA01&EA10'
  - timerange - string with date/time range
    - specify 'T0~T1' , missing parts of T1 default to T0, can give 'T0+dT'
    - example: timerange = '2007/10/16/01:00:00~06:30:00'

Calibration

- Data structure: 3 columns (data + 2 scratch columns):
- DATA column (raw data)
- MODEL (Fourier transform of source model onto data)
- CORRECTED_DATA (calibrated data)
- Columns created when needed, this may take some time
- Sets of calibration tables applied incrementally (apply all previous calibration tables before solving/application)
- Applycal changes CORRECTED_DATA (can split to DATA)
- Refactoring underway to work without scratch columns

Calibration continued

- Solvers (e.g. bandpass, gaincal, polcal, blcal)
- Based on data x calibration - model
- Uses Hamaker-Bregman-Sault Measurement Equation formalism
- Generate calibration tables by type, e.g. bandpass (B), gain (G,T), pol leakage (D), pol angle (X), place into equation
- Some types have channel dependencies (Df,Xf) or polynomial (BPOLY) or spline (GSPLINE) representations
- Working on making caltables applicable across different MS

Imaging

- Deconvolution using clean task
- Grid data onto uv-plane, transform to residual image, find model components (minor cycles), transform back to data and subtract to form residual data (major cycles), repeat [Cotton-Schwab clean]
- Control of algorithms used (e.g. csclean, mosaic), mapping to output cube planes (mfs, channel, velocity, frequency)
- Multi-frequency synthesis (mfs) for continuum, including higher order Taylor terms (intensity,alpha,…)
- Mosaicing using convolutional gridding to single uv-plane, plus uv-faceting

Visualization Tools

- Data needs to be displayed to understand it!
  - Can be a challenge for large datasets
- Visibilities: plotms, msview
- Images: viewer, imview
- Calibration tables: plotcal (soon plotms)
- Any table values: browsetable
- Single dish: sdplot

- Plot anything: use Python's matplotlib

PlotMS

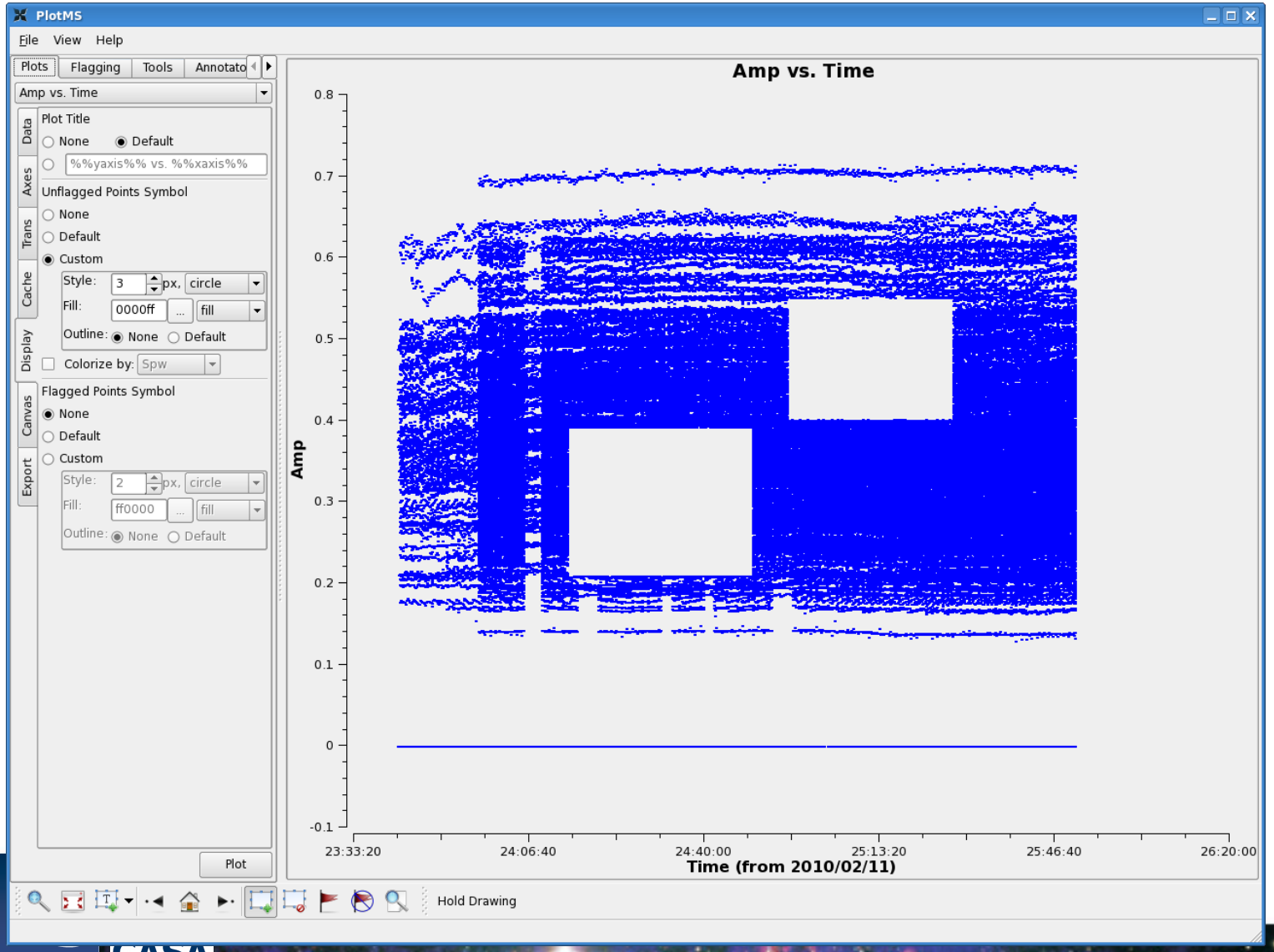

PlotMS
File View Help
Plots Flagging Tools Annotato
Amp vs. Time
Data Axes Trans Cache Display Canvas Export
Plot Title
None Default
%%yaxis%% vs. %%xaxis%%
Unflagged Points Symbol
None
Default
Custom
Style: 3 px, circle
Fill: 0000ff ... fill
Outline: None Default
Colorize by: Spw
Flagged Points Symbol
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

Image Viewer

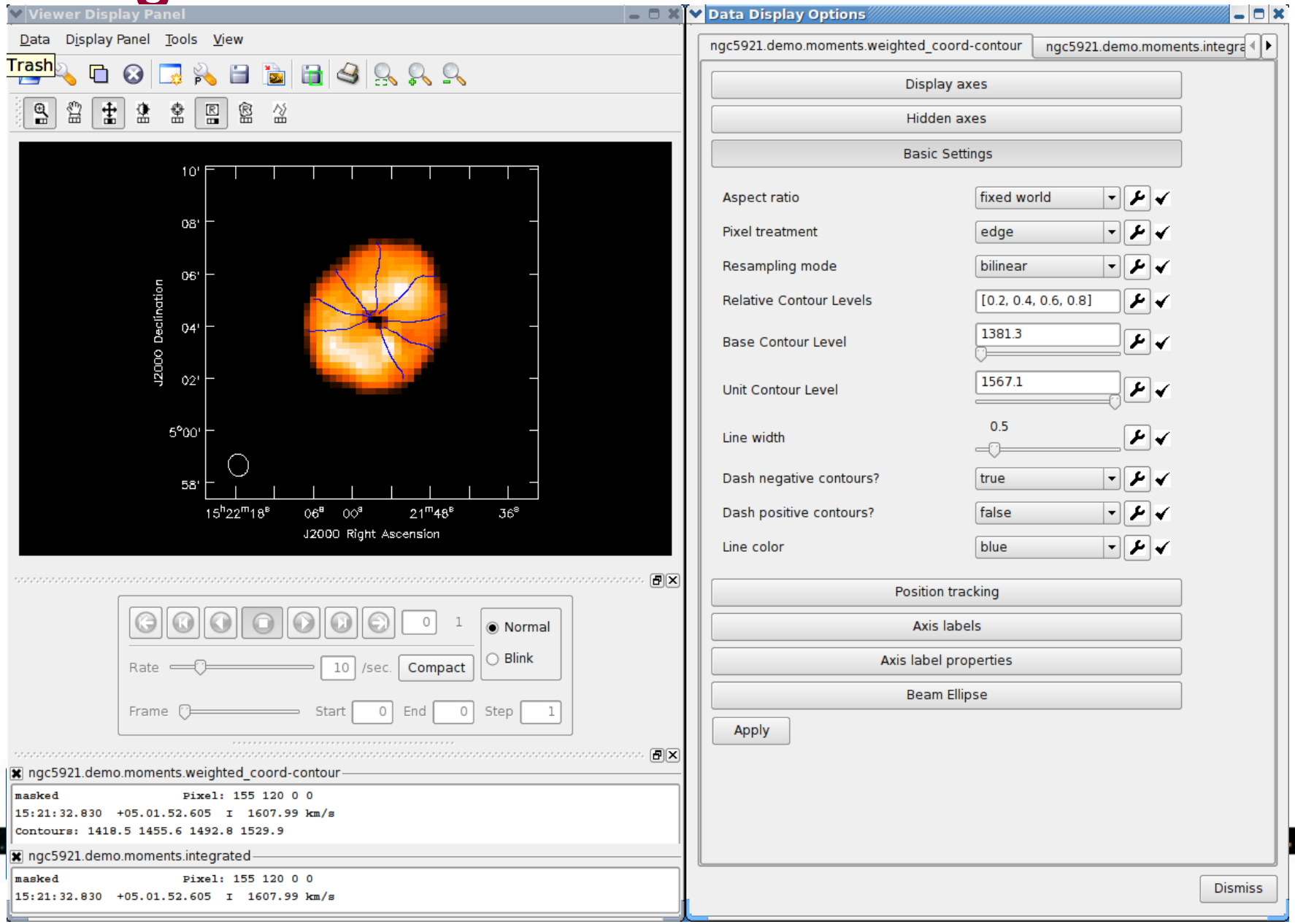
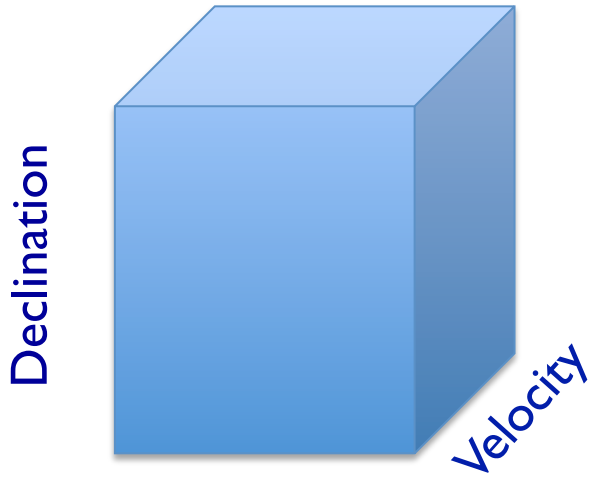
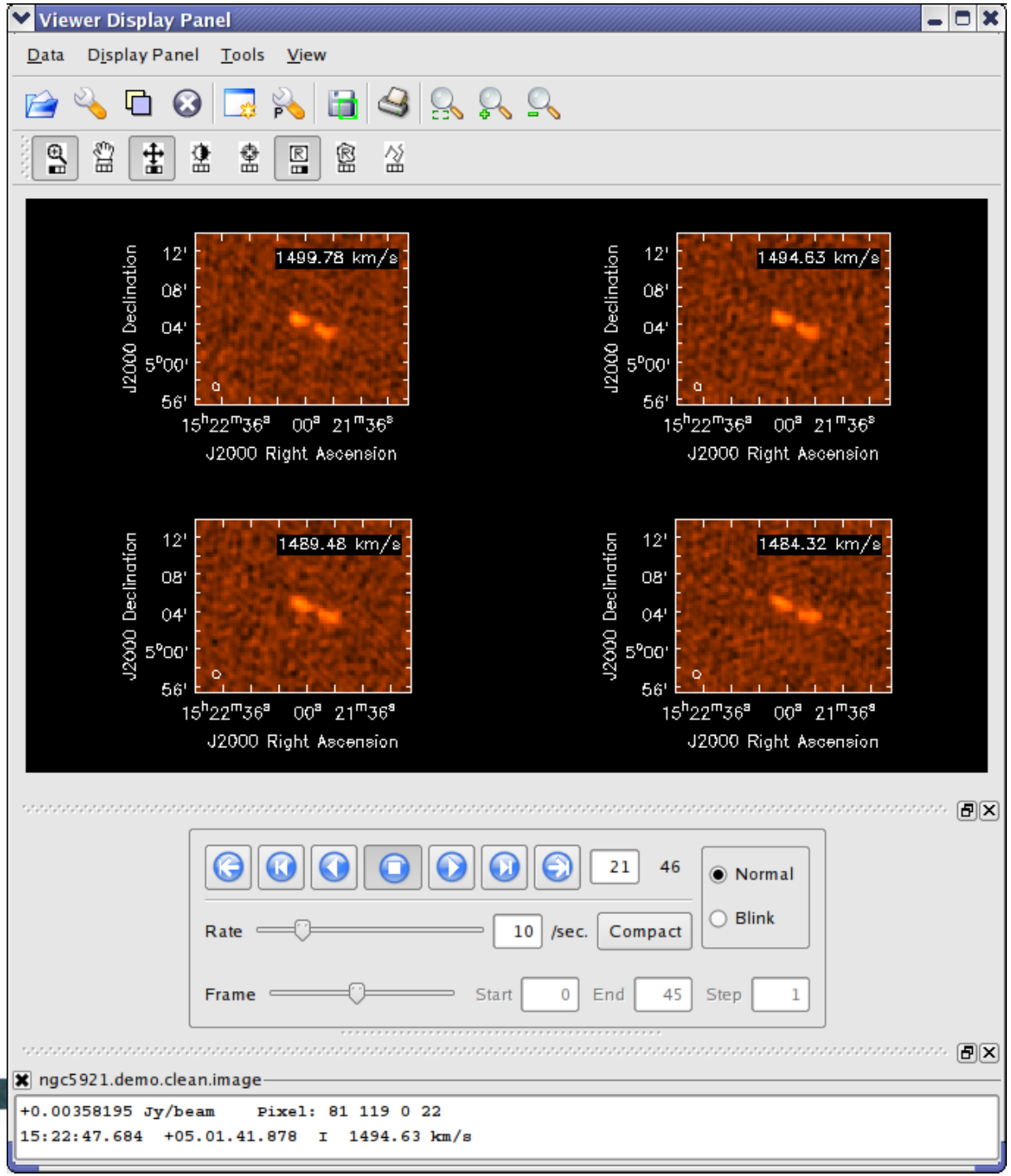


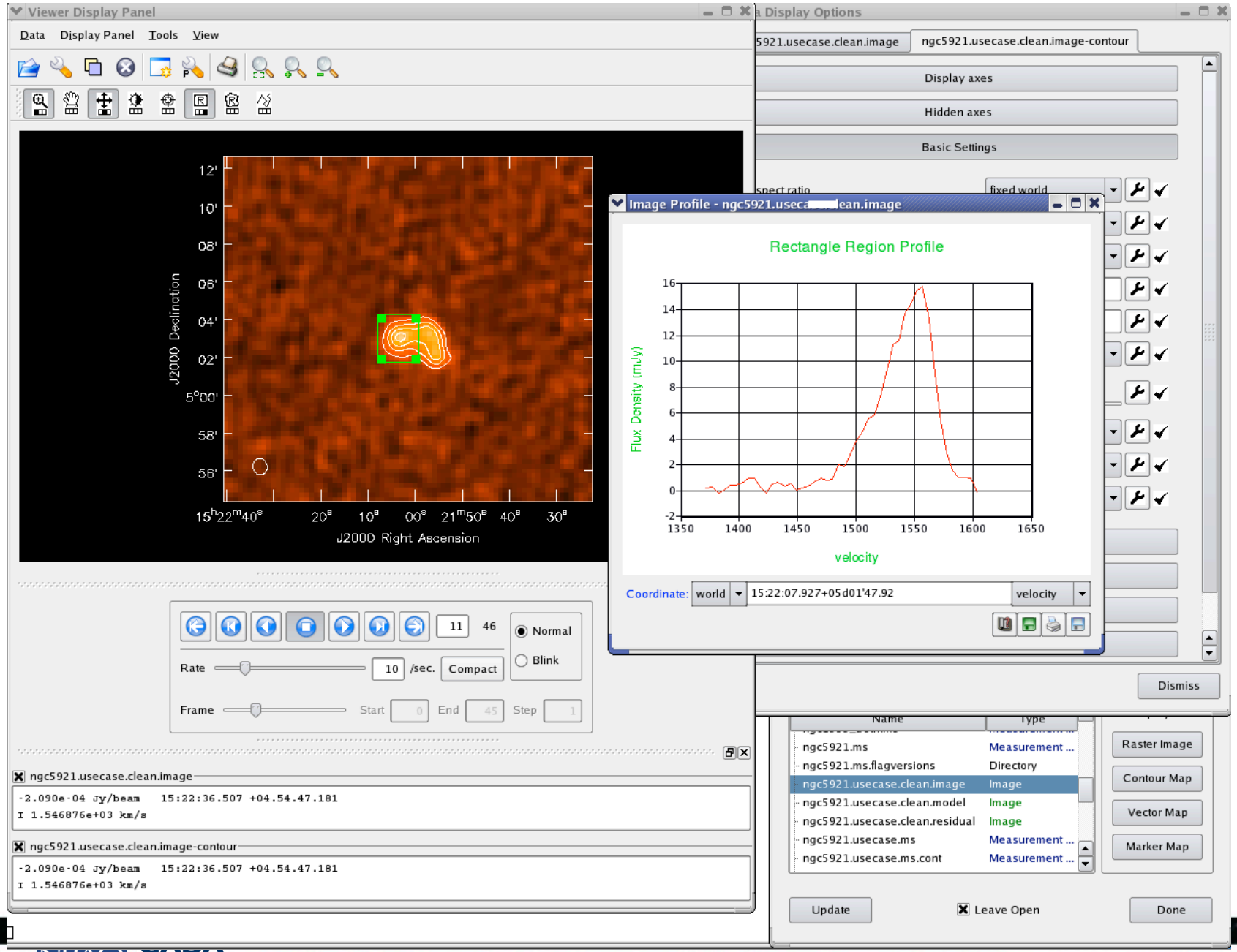
Image Viewer

- Displaying cubes
- Movies
- Channel maps





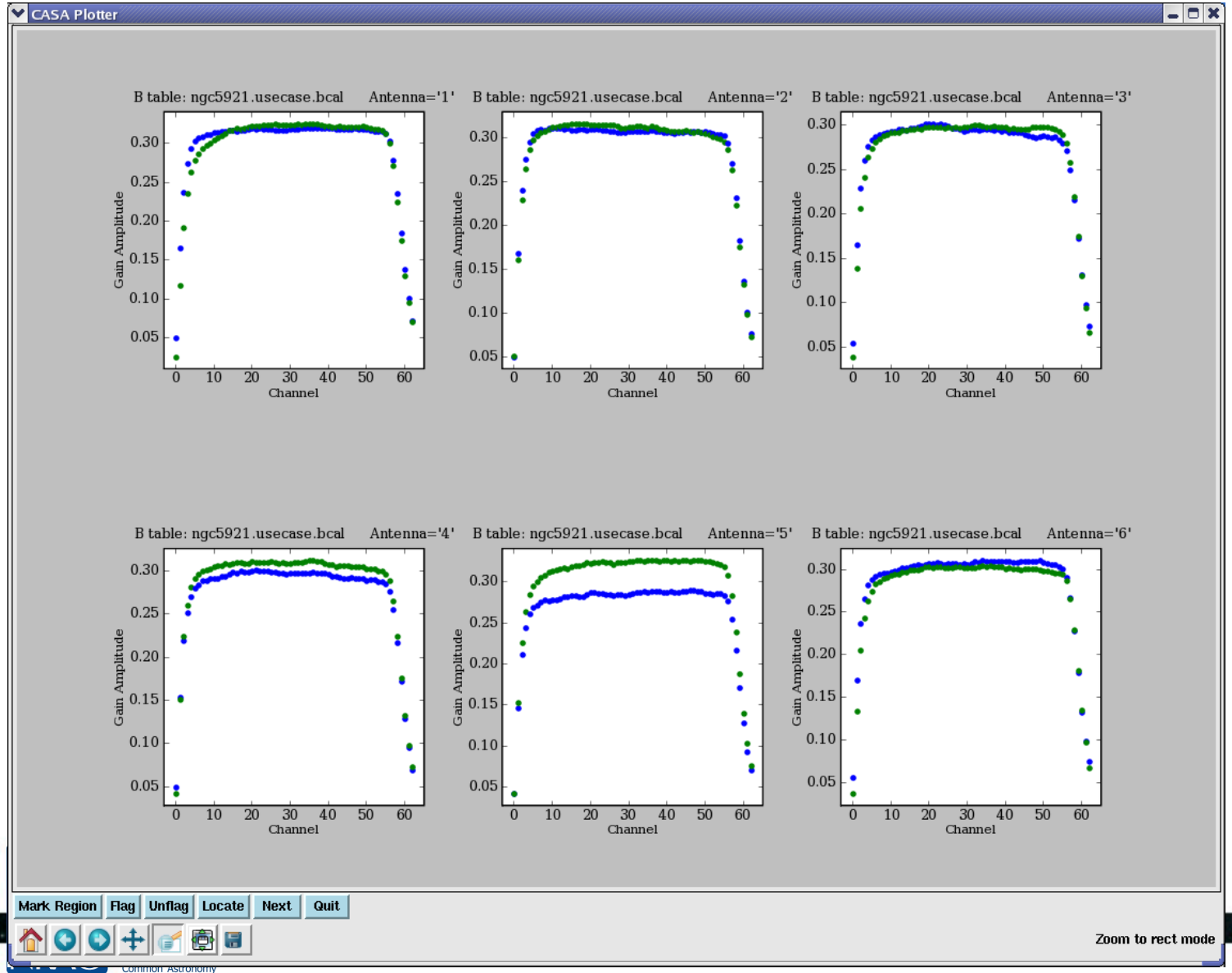
NRAO CASA Common Astronomy Software Applications


Viewer Display Panel
Data
Display Panel
Tools
View
J2000 Declination
J2000 Right Ascension
Image Profile - ngc5921.usecase.clean.image
Rectangle Region Profile
Flux Density (mJy)
velocity
Coordinate:
world
15:22:07.927+05d01'47.92
velocity
Display Options
ngc5921.usecase.clean.image
ngc5921.usecase.clean.image-contour
Display axes
Hidden axes
Basic Settings
fixed world
Normal
Blink
Rate
/sec.
Compact
Frame
Start
End
Step
ngc5921.usecase.clean.image
-2.090e-04 Jy/beam 15:22:36.507 +04.54.47.181
I 1.546876e+03 km/s
ngc5921.usecase.clean.image-contour
-2.090e-04 Jy/beam 15:22:36.507 +04.54.47.181
I 1.546876e+03 km/s
Name
Type
ngc5921.ms
Measurement ...
ngc5921.ms.flagversions
Directory
ngc5921.usecase.clean.image
Image
ngc5921.usecase.clean.model
Image
ngc5921.usecase.clean.residual
Image
ngc5921.usecase.ms
Measurement ...
ngc5921.usecase.ms.cont
Measurement ...
Raster Image
Contour Map
Vector Map
Marker Map
Dismiss
Update
Leave Open
Done

MSViewer


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

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

Plot Anything - matplotlib

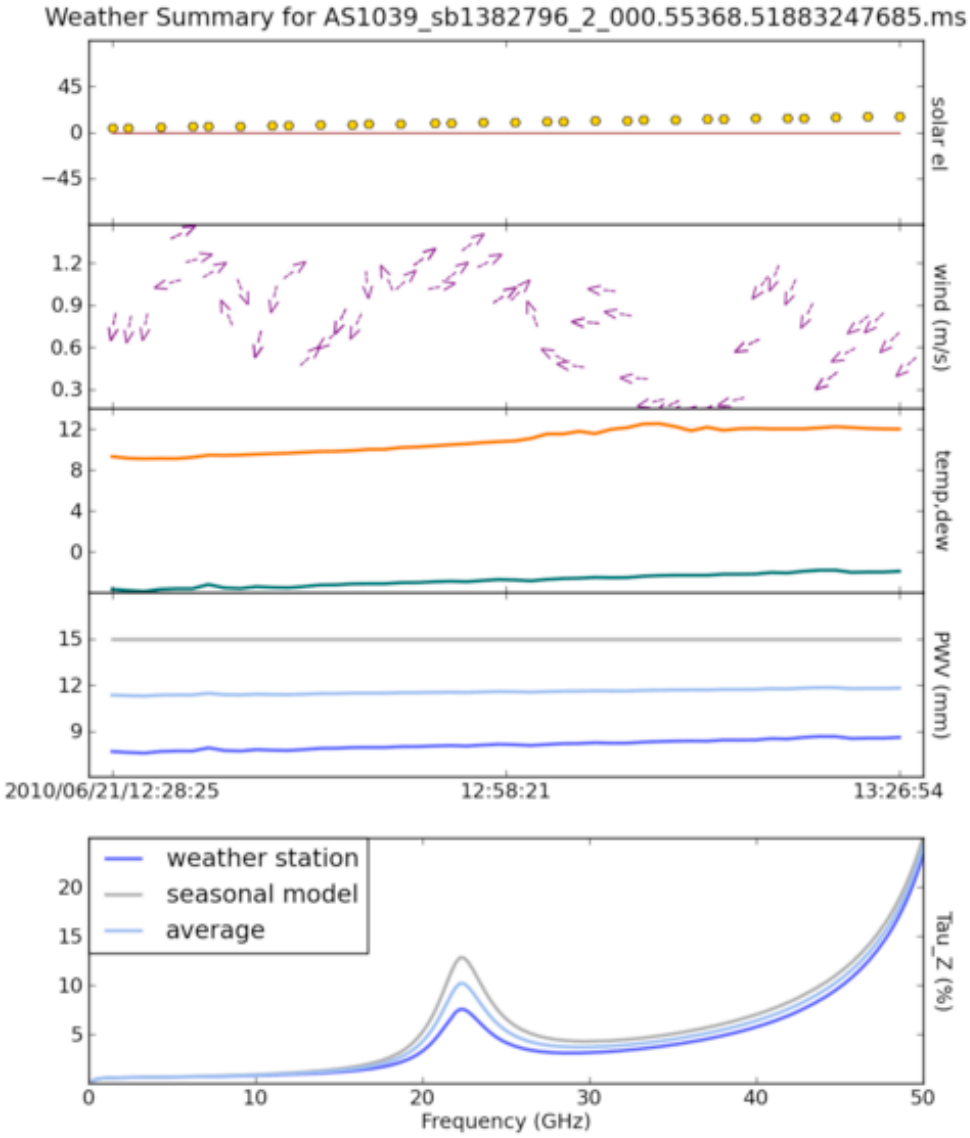


Image analysis

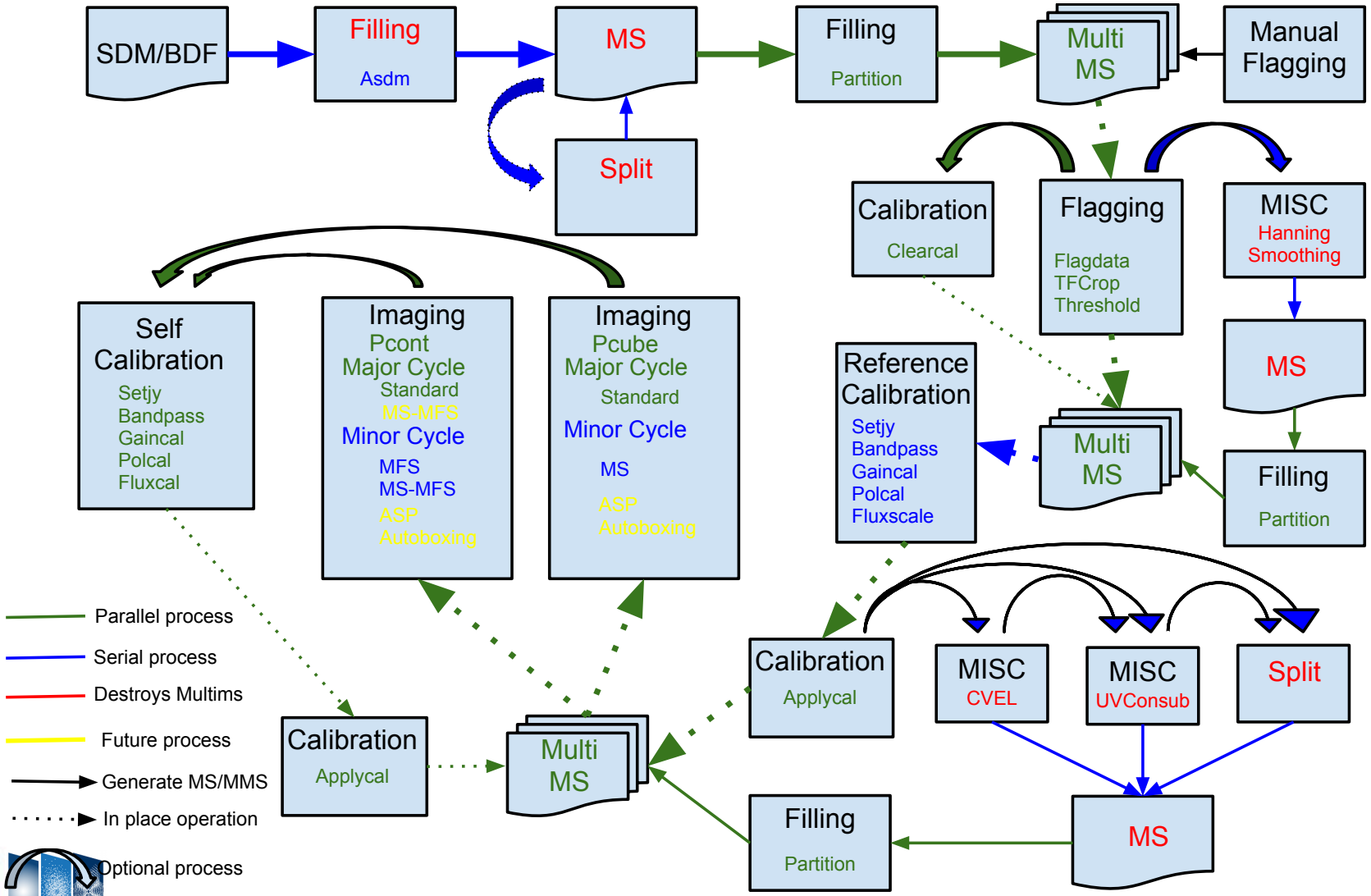
- specfit: to fit 1-dimensional gaussians and/or polynomial models to an image or image region.
- imfit : fit one or more elliptical Gaussian components on an image region(s).
- Also immath, imstat, imval
- Currently many gaps, use Python plus toolkit
- Contributed scripts can be used (and submitted by you).
- Contributed scripts are currently available at:

  http://casaguides.nrao.edu/ → Data Reduction Guides → EVLA Guides → Contributed Scripts

Ahead to the Future - Parallelization

- Large ALMA & EVLA datasets are challenging workstations
  - Large data volumes = expensive I/O
  - High sensitivity = expensive CPU
  - Want these balanced! (maybe use GPUs also eventually)
- CASA High Performance Computing Initiative
  - Parallelize code at all levels for use on cluster
  - Parallelize data so I/O can be easily parallelized
  - Process control
  - Make this all available to users as part of casa
    - Not a special purpose build
    - Some capabilities available now, you can use our cluster too!

Parallelized Data Flow (courtesy J.Robnett)



Getting User Support

- CASA Home: http://casa.nrao.edu
  - Cookbook, online reference, download, example scripts
- CASAguides.nrao.edu
  - For data reduction tutorials, tips, tricks, …
- "Helpdesk" at help.nrao.edu
  - Submit questions, suggestions, bugs (needs my.nrao.edu registration)
- CASA mailing lists: casa-announce, casa-users
- CASA topic in NRAO Science Forum

CASA Documentation

- Homepage: http://casa.nrao.edu → Using CASA
- CASA Reference Manual & Cookbook:
  - http://casa.nrao.edu/Doc/Cookbook/casa_cookbook.pdf
  - http://casa.nrao.edu/docs/UserMan/UserMan.html
- CASA Task Reference (same as inline help):
  - http://casa.nrao.edu/docs/TaskRef/TaskRef.html
- CASA Toolkit Manual:
  - http://casa.nrao.edu/docs/casaref/CasaRef.html
- CASAguides Wiki:
  - http://casaguides.nrao.edu
- Python:
  - http://python.org/doc (e.g., see Tutorial for novices)
- IPython:
  - http://ipython.org
- matplotlib:
  - http://matplotlib.sourceforge.net/

Large but detailed!