CASA
Juergen Ott (NRAO)

National Radio Astronomy Observatory

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

Juergen Ott  (CASA project scientist)
Crystal Brogan (CASA ALMA subsystem scientist)
Miriam Krauss  (CASA EVLA subsystem scientist)
Jeff Kern          (CASA manager)
Introduction to CASA

Juergen Ott (CASA project scientist)
Crystal Brogan (CASA ALMA subsystem scientist)
Miriam Krauss (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)
- Import/export data, edit, calibrate, image, analyze
- Also supports single dish! (based on ASAP)
- CASA has many tasks and a LOT of tools
- It is very versatile on how you run it
- Easy to write scripts and we would like to build up repository for contributed scripts and 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 coolness to come
CASA (Common Astronomy Software Applications)

Current version: 3.2.1
- New releases about every 6 months.

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

“release”, “test” and “stable” versions available at NRAO/ESO/ALMA

> 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
Outline

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

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

• to run a .py script:
  ```python
  execfile('<scriptname>')
  ```
  example: `execfile('ngc5921_demo.py')`

• indentation matters!
  – be careful when doing cut-and-paste to Python
  – cut a few (4-6) lines at a time

• Python counts from 0 to n-1!

• variables are global when using task interface

• tasknames are objects (not variables)
Tasks and tools in CASA

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

To see list of tasks organized by type:

> tasklist
Key Tasks

To see list of tasks with short help:

>taskhelp
Task Interface

examine task parameters with `inp`:

```bash
CASA <12>: inp
----------> inp()
# clean :: Invert and deconvolve images with selected algorithm
vis = ''           # Name of input visibility file
imagername = ''    # 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)
channel = -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'
gain = 0.1         # Maximum number of iterations
threshold = '1.0mJy' # Loop gain for cleaning
psfmode = 'clark'   # Flux level to stop cleaning, must include units: '1.0mJy'
psfmode = 'cfrac'   # Method of PSF calculation to use during minor cycles
options = 'csclen' or 'mosaic', '' uses psfmode
imagemode = []     # 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 values: same for both
cell = ['1.0arcsec'] # Phasecenter: direction or field index
phasenr = ''       # Rest Frequency to assign to image (see help)
restfreq = 'I'     # Stokes params to image (eg I,IV,IQ,IOUV)
imagetype = 'I'    # Weighting of uv (natural, uniform, briggs, ...)
weighting = 'natural' # Apply additional uv tapering of visibilities
uv tapering = False # Name of model image(s) to initialize cleaning
modelimage = [ ]   # Output Gaussian restoring beam for CLEAN image
probor = False     # Minimum PB level to use
minpb = 0.2        # True required for self-calibration
calready = True    # If true the taskname must be started using clean(...)
async = False
```

CASA <13>:
Task Interface

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

**casa**

### Restore parameters from file clean.last

```plaintext
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 = 'O' # Field Name
spw = ' ' # Spectral windows: channels: '' is all
selectdata = False # Other data selection parameters
mode = 'channel' # Type of selection (mfs, channel, velocity, frequency)
chan = 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
imagemode = ' ' # Use cs clean or mosaic. If '', use psfmode
multiscale = [] # set deconvolution scales (pixels), default: multiscale=[] (standard CLEAN)
interactive = False # use interactive clean (with GUI viewer)
mask = [100, 100, 140, 140] # cleanbox(es), mask image(s), and/or region(s) used in cleaning
imsize = [256, 256] # x and y image size in pixels. symmetric for single value
cell = [1.0, 1.0] # 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,UVV)
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
uv taper = False # Apply additional uv tapering of visibilities.
modelimage = ' ' # Name of model image(s) to initialize cleaning
```
# Parameter Checking

sanity checks of parameters in inp:

```python
IPy:Jupiter

CASAO<5>: psfmode='hogwarts'

CASAO<6>: inp()

# clean :: Deconvolve an image with selected algorithm
vis = 'ngc5921.usecase.ms.cont' # Visibilities file
imname = 'ngc5921.usecase.clean' # Images
field = '0' # Field
spw = '' # Spectral window ' ' is all
selectdata = False # Other parameters
mode = 'channel' # Type of selection (ims, 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
imagemode = 'usecscan 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
imsizex = 256 # x and y image size in pixels, symmetric for single value
imsizey = [156, 156] # x and y cell size, default unit arcsec
phasecenter = '' # Image phase center: position or field index
restfreq = 'I' # Rest frequency to assign to image (see help)
stokes = ['I', 'V', 'U', 'Q', 'UQ', 'UV'] # Stokes params to image (eg I, IV, QU, IQUV)
weighting = 'briggs' # Weighting to apply to visibilities
    robust = 0.5 # Briggs robustness parameter
    nux = 0 # number of pixels to determine uv-cell size 0=> field of view

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

In-line help:

> help 'clean' OR > pdoc clean

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(vis=None, imagename=None, field=None, spw=None, selectdata=
      gain=None, threshold=None, psfmode=None, imagermode=None, ftmode=None
      one, mask=None, nchan=None, start=None, width=None, imsizemode=None, cell=None
      er=None, cutertaper=None, inner taper=None, modelimage=None, restoringbeam
      r=None, cyclensmodeup=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
   m2.psf: point-spread function (dirty beam)
   m2.flux: relative sky sensitivity over field
   m2.model: image of clean components
   m2.residual: mass 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)
CASA Tool List

- list of default tools from `toolhelp`:

```plaintext
CASA <16>: toolhelp

Available tools:

- at: Juan Pardo ATM library
- cb: Calibration utilities
- cp: Cal solution plotting utilities
- fg: Flagging/Flag management utilities
- ho: Image analysis utilities

```ia.open('rotated.image')
```ia.rebin(region=mybox,outfile='slice.image',bin=[1,33,1,1],dropdeg=True)
```ia.close('rotated.image')

```plaintext
- tpb: Table utilities (selection, extraction, etc)
- tp: Table plotting utilities
- qa: Quanta utilities
- sl: Spectral line import and search
- sm: Simulation utilities
- vp: Voltage pattern/primary beam utilities

```

CASA <17>: }
```
There’s a good chance that your problem can be solved on the tool level, don’t be afraid! ~1000 tools available!

Tools described in the CASA Toolkit Reference:

http://casa.nrao.edu/docs/CasaRef/CasaRef.html
The Measurement Set

- the MS is a directory on disk
- the MAIN table in `table.*` files

- also contains sub-tables
  - e.g. FIELD, SOURCE, ANTENNA, etc.

- sub-tables are sub-directories
- to copy must `cp -rf` to get contents
- Best to remove ms with `rmtables('filename')`
- WARNING: moving a MS can break cal-table dependencies
Example MS

Example: `ls ngc5921.usecase.ms`

```plaintext
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
FIELD SOURCE table.f11 table.f5_TSM1 table.f9
FLAG_CMD SPECTRAL_WINDOW table.f11_TSM1 table.f6_TSM1
HISTORY STATE table.f2 table.f6_TSM0 table.info
OBSERVATION table.dat table.f2_TSM1 table.f7
POINTING table.f0 table.f3 table.f7_TSM1
```
**MAIN Table Contents**

Example using task **browsetable:**

![Table browser interface](image)
Visualization Tools

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

- Plot anything: use python’s matplotlib
PlotMS
Image Viewer
Image Viewer

- Displaying cubes
- Movies
- Channel maps
MSViewer
Plotcal
Anything - matplotlib
Data Selection Example

- standard selection parameters

  e.g. for task gaincal:

```python
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

- **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 (scratch columns):
  - **DATA** column (raw data)
  - **MODEL** (source model, e.g. extended flux calibrators, selfcal model)
  - **CORRECTED_DATA** calibrated data
- Columns are created when needed, this may take some time
- Calibration is with calibration tables, e.g. bandpass, gain, pol, antenna offset etc., and they are applied multiplicative (apply all previous calibration tables to create new one or to be applied to data)
Getting User Support

- **CASA Home:** [http://casa.nrao.edu](http://casa.nrao.edu)
  Cookbook, online reference, download, example scripts

- **CASAguides.nrao.edu**
  For data reduction tutorials, tips, tricks, …

- “Helpdesk” at [help.nrao.edu](http://help.nrao.edu)
  Submit questions, suggestions, bugs (needs my.nrao.edu registration)

- **CASA mailing lists:** casa-announce, casa-users

- User’s forum in the future
CASA Documentation

- CASA Analysis cookbook:

- CASA User Reference Manual:
  [http://casa.nrao.edu/docs/casaref/CasaRef.html](http://casa.nrao.edu/docs/casaref/CasaRef.html)

- CASAguides Wiki:
  [http://casaguides.nrao.edu](http://casaguides.nrao.edu)

- Python:
  [http://python.org/doc](http://python.org/doc) (e.g., see Tutorial for novices)

- IPython:

- matplotlib:
Single Dish Data Reduction with CASA

Kana Sugimoto (ALMA project, NAOJ)

Give a first touch to CASA single dish data reduction package!

Contents

✓ Brief summary of single dish observation and data reduction
✓ Single dish data reduction with CASA
Brief summary of single dish observations

Single dish telescopes measure a total flux in their beam for each observation point of sky.

- continuum observation
- spectral observation
- polarization

**Imaging**
change antenna directions over the map region and measure flux of each direction ... pointing observation, on-the-fly observation
Single dish Calibration Basics

Radiation don't come only form sources (atmosphere, CMB, system...) → need **calibration**

**Example: position switching**

- **ON source signal**
  \[ V_{ON} = V_{source} e^{-\tau} + V_{CMB} e^{-\tau} + V_{atm} (1 - e^{-\tau}) + V_{sys} \]

- **OFF source signal**
  \[ V_{OFF} = V_{CMB} e^{-\tau} + V_{atm} (1 - e^{-\tau}) + V_{sys} \]

- position switching, frequency switching, nutator switching, etc.

**The brightness temperature of a source**

\[ T_A = T_{CAL} \frac{V_{ON} - V_{OFF}}{V_{CAL}} \]

\[ T_A^* = \frac{T_A}{\eta e^{-\tau}} \]

**GBT:** \( T_{cal} / V_{cal} = T_{sys} / V_{OFF} \)
Single dish data reduction steps example

1. data import
   - ... convert data (internal format)

2. view data
   - ... observation summary, view data and spectra

3. calibration
   - ... according to observation strategy

4. flag
   - ... eliminate bad data

5. data manipulation
   - ... sum up spectra (S/N ↑)

6. baseline subtraction
   - ... line observation

7. spectral data analysis
   - ... line width, intensity, statistics

8. imaging
   - ... create image

9. view image & analyze
   - ... momentum, etc.
Single dish data reduction with CASA

1. **data import**
   - ... sdsave
2. **view data**
   - ... sdlist, sdplot, browsetable
3. **calibration**
   - ... sdaverage
4. **flag**
   - ... sdflag
5. **data manipulation**
   - ... sdaverage
6. **baseline subtraction**
   - ... sdbaseline
7. **spectral data analysis**
   - ... sdfit, sdstat
8. **imaging**
   - ... sdimaging
9. **view image & analyze**
   - ... viewer, etc.

---

**Orion SiO (reduction/)**

**FLS3a HI (imaging/)**
Tutorial data

Spectral line data reduction (reduction/)
- Data file: OrionS_rawACSmod
- Target: Orion S
- GBT: 45GHz (Spws=8, 8192 channels, XX YY)
- Observation: position-switching (4 scans/spw: ON-OFF-ON-OFF)
- Analysis script: orion_SiO.py (reduce IF15 = SiO line)

Spectral Imaging (imaging/)
- Data file: fls3a_HI.asap (calibrated data)
- Field: FLS3a (Galactic HI mapping)
- GBT: HI line @1.4GHz (Spws=2, 1024 channels, XX YY)
- Observation: 6 x 3deg (grid: 0.1x 0.05 deg)
- Analysis script: fls3a.py (generate image of IF0)
Step 0! Import single dish package

First of all, import **ASAP**, the single dish reduction package in CASA

```
CASA<>: asap_init()
```

Single dish spectral analysis functions in CASA are based on ASAP (ATNF Spectral Analysis Package) originally developed in Australia Telescope National Facility.

+ Many enhancements to analyze data from **ALMA** and the other telescopes such as **NRO 45m**, **ASTE**, **GBT**, and **ATNF**.

So far, **you should explicitly import the package to CASA**.
Single dish data reduction with CASA

- **data import**
  - view data
  - calibration
  - flag
  - data manipulation
  - baseline subtraction
  - spectral data analysis
  - imaging

- **... sdsave**
  - ... sdlist, sdplot, browsetable
  - ... sdaverage
  - ... sdflag
  - ... sdaverage
  - ... sdbaseline
  - ... sdfit, sdstat
  - ... sdimaging
  - ... viewer, etc.
**sdsave** - import / export data

Transform data format
- input: SDFITS, rpfits, NOSTAR, NEWSTAR , MS, scantable
- output: SDFITS, ASCII , MS, scantable

**scantable** ... internal data format of ASAP

---

```python
# sdsave :: ASAP SD task: save the sd spectra in various format
sdfile = 'OrionS_rawACSmod'  # name of input SD dataset
antenna = 0  # antenna name or id (only effective for MS
            # input)
rowlist = []  # list of row numbers to process (e.g. [0,2,4,6])
scanlist = []  # list of scan to process (e.g. [20,21,22,23])
field = ''  # string for selection by source name
iflist = []  # list of IF ids to select (e.g. [0,1])
pollist = []  # list of polarization ids to select (e.g. [0,1])
scanaverage = False  # average integs within scans (True,False)
timeaverage = False  # average scans over time (True,False)
polaverage = False  # average over polarizations (True,False)
outfile = 'orion_Si0.asap'  # output file name
outform = 'ASAP'  # output file format (ASCII,MS2,SDFITS,ASAP)
overwrite = True  # overwrite the output file if already exists
async = False  # If true the taskname must be started using
               # sdsave(...)```
Single dish data reduction with CASA

- data import
  - ... sdsave

- view data
  - ... sdlist, sdplot, browsetable

- calibration
  - ... sdaverage

- flag
  - ... sdflag

- data manipulation
  - ... sdaverage

- baseline subtraction
  - ... sdbaseline

- spectral data analysis
  - ... sdfit, sdstat

- imaging
  - ... sdimaging, sdtpimaging

- view image & analyze
  - ... viewer, etc.
sdlist – list observation

View scantable summary

```
# sdlist :: ASAP SD task: list summary of single dish data
sdfile = 'orion_SiO.asap' # name of input SD dataset
antenna = 0 # antenna name or id (only effective for MS input)
scanaverage = False # average ints within scans (True,False)
listfile = '' # Name of output file for summary list
overwrite = False # overwrite the output file if already exists
async = False # If true the taskname must be started using
             #    sdlist(...)
```

Output → CASA logger
<table>
<thead>
<tr>
<th>Priority</th>
<th>Origin</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Scan Table Summary</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Beams: 1</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>IFs: 26</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Polarisations: 2 (linear)</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Channels: 8192</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Observer: Joseph McMullin</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Obs Date: 2006/01/19/01:45:58</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Project: AGBT06A_018_01</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Obs. Type: OffOn:PSWITCHOFF:TPWCAL</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Antenna Name: GBT</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Flux Unit:</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Rest Freqs: [4.5490258e+10] [Hz]</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Absissa: Channel</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Selection: none</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Scan Source Time Integration</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>Beam Position (J2000)</td>
</tr>
<tr>
<td>INFO</td>
<td>sdlist</td>
<td>IF Frame RefVal RefPix Increment Channels</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scan Source</th>
<th>Time</th>
<th>Integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>OrionS</td>
<td>01:45:58</td>
<td>4 x 30.0s</td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
<tr>
<td>OrionS</td>
<td>05:15:13.5 - 05:24:08.2</td>
<td></td>
</tr>
</tbody>
</table>

Insert Message | + | / | | Lock scroll |
### browsetable – view scantable (table data)

**Table Browser**

<table>
<thead>
<tr>
<th>SRC_TYPE</th>
<th>FIELD_NAME</th>
<th>SPECTRA</th>
<th>FLAGTRA</th>
<th>TSYS</th>
<th>DIRECTION</th>
<th>AZIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>OrionS [28.1654, 0.185437, 0.185668, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>OrionS [6.6194, 0.30304, 0.302895, 0.3...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>OrionS [33.93, 0.153334, 0.175013, 0.1...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>OrionS [12.9472, 0.0422599, 0.0421586...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>OrionS [7.73297, 0.240423, 0.239547, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>OrionS [33.5171, 0.257387, 0.257224, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>OrionS [4.91839, 0.221327, 0.221545, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>OrionS [14.2414, 0.264974, 0.264911, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>11</td>
<td>OrionS [28.3547, 0.198971, 0.198763, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>OrionS [6.37802, 0.319975, 0.319337, 0...</td>
<td>[0, 0, 0, 0...</td>
<td>[1.37542, -0.0942875]</td>
<td>2.7994</td>
<td></td>
</tr>
</tbody>
</table>
# Table Browser

**browsetable** – view scantable (table data)

<table>
<thead>
<tr>
<th>Table Keywords</th>
<th>Field Keywords</th>
<th>Keyword</th>
<th>Type</th>
<th>Value</th>
<th>Extra Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>VERSION</td>
<td>Unsigned Integer</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>POLTYPE</td>
<td>String</td>
<td>linear</td>
<td></td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>DIRECTIONREF</td>
<td>String</td>
<td>J2000</td>
<td></td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>FREQUENCIES</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/FREQUENCY...</td>
<td>Subtable has 8 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>WEATHER</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/WEATHER</td>
<td>Subtable has 1 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>FOCUS</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/FOCUS</td>
<td>Subtable has 1 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>TCAL</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/TCAL</td>
<td>Subtable has 256 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>MOLECULES</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/MOLECULES</td>
<td>Subtable has 1 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>HISTORY</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/HISTORY</td>
<td>Subtable has 4 rows.</td>
</tr>
<tr>
<td>Handle</td>
<td>Handle</td>
<td>FIT</td>
<td>Table</td>
<td>/home/sugimtken/workcasa/singledish/reduction/orion_SiO.asap/FIT</td>
<td>Subtable has no rows.</td>
</tr>
<tr>
<td>HANDLE</td>
<td>HANDLE</td>
<td>nIF</td>
<td>Integer</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>HANDLE</td>
<td>HANDLE</td>
<td>nBeam</td>
<td>Integer</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>HANDLE</td>
<td>HANDLE</td>
<td>nPol</td>
<td>Integer</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>HANDLE</td>
<td>HANDLE</td>
<td>nChan</td>
<td>Integer</td>
<td>8192</td>
<td></td>
</tr>
</tbody>
</table>

**Browsing table: orion_SiO.asap**
sdplot – plot scantable

Orion spectra (calibrated)

Data File: orion_S10.cal.asap
Obs Date: 2006/01/19/01:45:58
Obs Type: Orion:J160:TPW:CAL
Observer: Joseph McMailin
Antenna Name: GB1

Beam: 1
IFS: 26
Polarisation: 2 (linear)
Channels: 8192
Rest Freq: [4.5490258e+10] Hz

Brightness Temperature (K)

LSRK Frequency (GHz)

Scan 25 (Orions)
Scan 27 (Orions)
Scan 26 (Orions)

SDSPlot
Graphical User Interface for Data Analysis

NAOJ
National Astronomical Observatory of Japan
\textbf{sdplot} – plot scantable

\texttt{plottype = \{\textquote{totalpower}, \textquote{azel}, \textquote{pointing}\}}

- \texttt{row \#} – total power
- \texttt{R.A. \textendash Dec (Orion)}
- \texttt{R.A. \textendash Dec (FLS3a)}
- \texttt{time \textendash azimuth}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{example.png}
\end{figure}
Single dish data reduction with CASA

- Data import
- View data
- Calibration
  - Flag
  - Data manipulation
  - Baseline subtraction
  - Spectral data analysis
- Imaging
- View image & analyze

... sdsave
... sdlist, sdplot, browsetable
... sdaverage
... sdflag
... sdaverage
... sdbaseline
... sdfit, sdstat
... sdimaging
... viewer, etc.
# sdaverage :: ASAP SD task: do data selection, calibration, and averaging
sdfile = 'orion_SiO.asap'  # name of input SD dataset
antenna = 0  # antenna name or id (only effective for MS input)
fluxunit = 'K'  # units for line flux (K,Jy) (=current)
telescopeparm = ''  # param of telescope for flux conversion

specunit = ''  # units for spectral axis (channel,km/s,GHz,=current)
frame = ''  # frequency reference frame, e.g. LSRK (=current)
doppler = ''  # doppler convention, e.g. RADIO (=current)
calmode = 'ps'  # SD calibration mode (ps,nod,fs,fsotf,none)
scanlist = [24, 25, 26, 27]  # list of scans to use (e.g. [1,2,3,4])
field = ''  # string for selection by source name
iflist = [15]  # list of IF ids to select (e.g. [0,1])
pollist = []  # list of polarization ids to select (e.g. [0,1])
channelrange = []  # channel range selection (e.g. [0,5000])

scanaverage = False  # average integers within scans (True,False)
timeaverage = False  # average scans over time (True,False)
polaverage = False  # average over polarizations (True,False)
tau = 0.09  # atmospheric optical depth for correction
verify = False  # verify the results of calibration
outfile = 'orion_SiO.cal.asap'  # output file name
outform = 'ASAP'  # output file format (ASCII,MS,SDFITS,ASAP)
overwrite = True  # overwrite the output file if already exists
plotlevel = 0  # plot results (0=none,1=some,<0=hardcopy)
async = False  # If true the taskname must be started using
              # sdaverage(...)
sdaverage – calibration

Observation

Calibrated
Single dish data reduction with CASA

- **Data Import**
  - `... sdsave`

- **View Data**
  - `... sdlist, sdplot, browsetable`

- **Calibration**
  - `... sdaverage`

- **Flag**
  - `... sdflag`

- **Data Manipulation**
  - `... sdaverage`

- **Baseline Subtraction**
  - `... sdbaseline`

- **Spectral Data Analysis**
  - `... sdfit, sdstat`

- **Imaging**
  - `... sdimaging`

- **View Image & Analyze**
  - `... viewer, etc.`
sdflag – flag out bad data

# sdflag :: ASAP SD spectral flagging task

sdfile = 'orion_SiO.cal.asap' # name of input SD dataset
antenna = '0' # antenna name or id (only effective for MS input)
scanlist = [] # list of scans to use (e.g. [1,2,3,4])
field = '' # string for selection by source name
iflist = [] # list of IF ids to select (e.g. [0,1])
pollist = [] # in by ids or names to select

maskflag = [[0, 10], [8000, 8191]] # list of mask regions to flag/unflag
flagrow = [] # to apply row-based

clip = False # flag mode (flag, unflag)
flagmode = 'flag' # flag mode (flag, unflag)

outfile = 'orion_SiO.fl.asap' # name of output file
outform = 'ASAP' # output file format (ASCII, MS, SDFITS, ASAP)
overwrite = True # overwrite the output file if already exists
plotlevel = 0 # control for plotting of results
async = False # If true the taskname must be started using

by channel
by row ID
by flux
Single dish data reduction with CASA

- data import
- view data
- calibration
- flag
- **data manipulation**
- baseline subtraction
- spectral data analysis
- imaging
- view image & analyze

... sdsave
... sdlist, sdplot, browsetable
... sdaverage
... sdflag
... sdaverage
... sdbaseline
... sdfit, sdstat
... sdimaging
... viewer, etc.
Single dish data reduction with CASA

- Data import
  - ... sdsave

- View data
  - ... sdlist, sdplot, browsetable

- Calibration
  - ... sdaverage

- Flag
  - ... sdflag

- Data manipulation
  - ... sdaverage

- Baseline subtraction
  - ... sdbaseline

- Spectral data analysis
  - ... sdfit, sdamplitude

- Imaging
  - ... sdimaging

- View image & analyze
  - ... viewer, etc.
**sdbaseline** – baseline subtraction

polyominal baseline fitting and subtraction

```plaintext
# sdbaseline :: ASAP SD task: fit/remove a spectral baseline
sdbase = 'orion_SiO.ave.asap' # name of input SD dataset
antenna = 0 # antenna name or id (only effective for MS
      # input)
fluxunit = '' # units for line flux (K,Jy) (=current)
spectenit = '' # units for spectral axis (channel,km/s,GHz)
frame = '' # frequency reference frame, e.g. LSRK
      # (=current)
doppler = '' # doppler convention, e.g. RADIO (=current)
scanlist = [] # list of scans to use (e.g. [1,2,3,4])
field = '' # string for selection by source name
iflist = [] # F ids to select (e.g. [0,1])
pollist = [] # polarization ids to select (e.g. [0,1])
tau = 0.0 # atmospheric optical depth for correction

blmode = 'interact' # mode for baseline fitting
blpoly = 5 # order of baseline polynomial
verify = False # verify the results of baseline fitting
masklist = [] # list of mask regions to include in
      # BASELINE fit

outfile = 'orion_SiO.bl.asap' # output file name
outform = 'ASAP' # output file format (ASCII,MS,SDFITS,ASAP)
overwrite = True # overwrite the output file if already
      # exists
plotlevel = 0 # plot results (0=none,1=some,<0=hardcopy)
async = False # If true the taskname must be started using
      # sdbaseline(...)```
**sdbaseline** – baseline subtraction

**interactive mask selection**

- **draw rectangle**
  - **add** mask chans: LEFT-mouse
  - **delete** mask chans: RIGHT-mouse

current mask in yellow

press “Enter” to finish selection
Single dish data reduction with CASA

- **data import**
  - sdsave

- **view data**
  - sdlist, sdplot, browsetable

- **calibration**
  - sdaverage

- **flag**
  - sdflag

- **data manipulation**
  - sdaverage

- **baseline subtraction**
  - sdbaseline

- **spectral data analysis**
  - sdfit, sdstat

- **imaging**
  - sdimaging

- **view image & analyze**
  - viewer, etc.
### sdfit: fit spectral line

#### Gaussian or Lorentzian fitting

```python
# sdfit: ASAP SD task: fit a spectral line
sdffile = 'orion_SiO.bl.asap'  # name of input SD dataset
antenna = 0  # antenna name or id (only effective for MS)
fluxunit = ''  # units for line flux (K, Jy) (=current)
specunit = ''  # units for spectral axis (channel, km/s, GHz)
frame = ''  # frequency reference frame, e.g. LSRK

# (=current)

doppler = ''  # doppler convention, e.g. RADIO (=current)
scanlist = []  # list of scans to use (e.g. [1, 2, 3, 4])
field = ''  # for selection by source name
iflist = []  # ids to select (e.g. [0, 1])
pollist = []  # polarization ids to select (e.g. [1])

fitfunc = 'gauss'  # function for fitting
fitmode = 'list'  # mode for fitting
maskline = [3000, 5000]  # list of mask regions to INCLUDE in LINE fitting
invertmask = False  # invert mask (EXCLUDE masklist instead)

# lines to fit in in maskline region

nfit = [1]  # list of number of gaussian/lorentzian

fitfile = 'orion_SiO.fit.txt'  # name of output file for fit results
overwrite = True  # overwrite the fitfile if already exists
plotlevel = 1  # control for plotting of results
async = False  # If true the taskname must be started using
               # sdfit(...)"
# sdfit :: ASAP SD task

sdfile
antenna

fluxunit
specunit
frame

doppler
scanlist
field
iflist
pollist

fitfunc
fitmode
maskline

invertmask
nfit

fitfile = 'orion_SiO.fit.txt'  # name of output file for fit results
overwrite = True  # overwrite the fitfile if already exists
plotlevel = 1  # control for plotting of results
async = False  # If true the taskname must be started using sdfit(...)
sdstat – calculate statistics

Min (channel), max (channel), mean, median, sum, rms, stddev, integrated intensity, and equivalent width

Output → CASA logger
sdstat – calculate statistics

INFO sdstat...  #################################################################
INFO sdstat...  ####### Begin Task: sdstat #######
INFO sdstat...  sdstat:::casa
INFO sdstat...  Current fluxunit = K
INFO sdstat...  No need to convert fluxunits
INFO sdstat...  Using current frequency frame
INFO sdstat...  Using current doppler convention
INFO sdstat...  Using full region
INFO sdstat...  -----------------------------
INFO sdstat...  max
INFO sdstat...  Scan[0] (OrionS) Time[2005/01/19/02:07:54]:
INFO sdstat...  IF[15] = 0.469  (@ 4109.000 [channel])
INFO sdstat...  -----------------------------
INFO sdstat...  min
INFO sdstat...  Scan[0] (OrionS) Time[2005/01/19/02:07:54]:
INFO sdstat...  IF[15] = -0.286  (@ 7980.000 [channel])
INFO sdstat...  -----------------------------
Single dish data reduction with CASA

- data import
  - ... sdsave
- view data
  - ... sdlist, sdplot, browsetable
- calibration
  - ... sdaverage
- flag
  - ... sdflag
- data manipulation
  - ... sdaverage
- baseline subtraction
  - ... sdbaseline
- spectral data analysis
  - ... sdfit, sdstat
- imaging
  - ... sdimaging
- view image & analyze
  - ... viewer, etc.
Important!
Convert data to **MeasurementSet** before imaging.

```bash
# sdsave : ASAP SD task: save the sd spectra in various format
sdfile   = 'fls3a_HI.asap'   # name of input SD dataset
antenna  = 0               # antenna name or id (only effective for MS input)
rowlist  = []              # list of row numbers to process (e.g. [0,2,4,6])
scanlist = []              # list of scan to process (e.g. [20,21,22,23])
field    = ''              # string for selection by source name
iflist   = []              # list of IF ids to select (e.g. [0,1])
pollist  = []              # list of polarization ids to select (e.g. [0,1])
sconaverage = False  # average integs within scans (True,False)
timeaverage = False   # average scans over time (True,False)
polaverage = False   # average over polarizations (True,False)
outfile  = 'fls3a_HI.ms'  # output file name
outform  = 'MS2'         # output file format (ASCII,MS2,SDFITS,ASAP)
overwrite = True        # overwrite the output file if already exists
async    = False        # If true the taskname must be started using
                      #  sdsave(...)
**Single dish data** is stored in the **FLOAT_DATA** column.
**sdimaging** – single dish imaging

**Input:** MeasurementSet  **Output:** CASA image

```python
# sdimaging :: SD task: imaging for total power and spectral data
sdfile = 'fis3a_HI.ms'  # name of input SD dataset
      # for this task)
specunit = 'channel'  # units for spectral axis
scanlist =  []  # list of scans to use (e.g., [1,2])
field = 0  # string for selection by field name
spw = 0  # spectral window id for each scan
antenna = 0  # antenna name(s) or id(s)
stokes = 'I'  # stokes or correlation name
gridfunction = 'PB'  # gridding function for im
      # ("BOX","SF","PB")
imagename = 'fis3a_HI.full.im'  # output image name
overwrite = True  # overwrite option
imsise = [150, 150]  # x and y image size in pixels
      # single value

cell = ['1.5arcmin', '1.5arcmin']  # x and y cell size. default unit
      # arcmin
dochannelmap = True  # True for channel map, False for total power
      # imaging
nchan = 100  # number of spectral channel for created image
start = 30  # reference value of start channel (in units
      # of specunit)
step = 9  # width of each spectral channel (in units of
      # specunit)
phasecenter = 'J2000 17:18:29 +59.31.23'  # Image phase center: position or
      # field id
ephemsrname =  ''  # ephemeris source name
pointingcolumn = 'direction'  # pointing data column to use
async = False  # If true the taskname must be started using
      # sdimageing(...)
```
The other single dish tasks

- `sdsmooth` ... channel average spectra
- `sdcal = sdaverage + sdbaseline + sdsmooth`
- `sdflagmanager` ... save, restore, and list flag versions
- `sdtpimaging` ... total power imaging (incl. scan noise subtraction)
- `sdimprocess` ... subtract scan noise
- `sdscale` ... scale spectra
- `sdmath` ... various arithmetic (+, -, *, /) operations on spectra
- `sdcoadd` ... merge scantables
Step 1: run the reduction scripts (orion_SiO.py & fls3a.py) by `execfile`, e.g.,
```
CASA<>: execfile('orion_SiO.py')
```
to understand single dish data reduction steps

Step 2: open the scripts to see how parameters are set. Copy and paste each line to set parameters and run tasks. You can also change parameters as you like to see the changes.

- CASA task reference
  [http://casa.nrao.edu/docs/TaskRef/TaskRef.html](http://casa.nrao.edu/docs/TaskRef/TaskRef.html)
- CASA cookbook