Data Handling Tools

Ger van Diepen
ASTRON

Tools

python/ipython

- pylab www.scipy.org
  numpy, scipy, matplotlib
- pyqt http://wiki.python.org/moin/PyQt
- pyfits http://www.stsci.edu/resources/software_hardware/pyfits
- pydal usg.lofar.org/wiki/doku.php NOT pydal.sourceforge.net
- pyrap pyrap.googlecode.com
- CASA http://casa.nrao.edu

C++

- casacore casacore.googlecode.com
- HDF5 www.hdfgroup.org
- Root http://root.cern.ch/root
- cfitsio http://heasarc.gsfc.nasa.gov/fitsio

Image viewers

- casaviewer (CASA)
- kview http://www.atnf.csiro.au/computing/software/karma
- SAOimage/DS9 http://hea-www.harvard.edu/RD/ds9

casacore

C++ library for astronomical data handling

- Arrays — templated N-dim arrays (STL-conforming)
- Tables — storage mechanism with TaQL query language
- MeasurementSet — visibility data storage and access (using Tables)
- Measures — values in frame (direction, position, epoch, ...)
- Coordinates — world coordinates for images
- Images — N-dimensional image cubes with 0 or more masks (supports HDF5, FITS, Miriad, expressions (LEL))

- Used by LOFAR, ASKAP, ALMA, MeqTrees, pyrap, pydal

- See

Download	casacore.googlecode.com
Classes	www.astron.nl/casacore/trunk/casacore/doc/html
TaQL	www.astron.nl/casacore/trunk/casacore/doc/notes/199.html
LEL	www.astron.nl/casacore/trunk/casacore/doc/notes/223.html
MS definition	www.astron.nl/casacore/trunk/casacore/doc/notes/229.html

Tables

- Relational DBMS-like storage mechanism
  - A column can contain:
    - Scalar bool, integer, real, complex, string
    - N-dim array bool, integer, real, complex, string (column major)
    - Record (struct)
  - Keywords can be attached to table and each column
    - Used for subtables
    - Define unit and reference frame of a column
      (module TableMeasures)
- TaQL is SQL-like query language
  results in reference table
- Various storage managers
  - StandardStMan columnar storage; for scalars or small fixed arrays
  - IncrStMan rather constant data
  - TiledStMan bulk data arrays (tiling gives fast access for all axes)
- A table is a directory containing several files and subtables
- Concurrent access (one writer, multiple readers)

MeasurementSets

Collection of Tables containing visibility data

- Described in note 229 (www.astron.nl/casacore/trunk/casacore/doc/notes/229.html)
- Predefined structure of subtables and columns
- Instrument-specific subtables and columns can be added

- Main table has data columns DATA and FLAG
  - Per row 2-dim array with axes frequency and polarisation
  - Indexed by columns containing baseline (antenna1 and antenna2), time, band, subarray, etc.

- Subtables define meta data

  ANTENNA, ARRAY, FEED, FIELD, SPECTRAL_WINDOW, ...

  Some subtables are optional

- C++ headers in ms/MeasurementSets
- No specific Python code; use pyrap.tables

Images

- N-dim cube (ra, dec, freq, pol, ...)
  with world coordinates

- Two native formats
  - Tables, HDF5
  - Tiled storage
  - data type float, double, complex, or dcomplex
  - zero or more masks
- Two external formats (readonly)
  - FITS and Miriad
  - Non-tiled storage (FITS proposal for tiling)
  - data type float only

- Virtual concatenation of images (e.g. to combine subbands)
- Image expressions using LEL (e.g. difference of images)

  See www.astron.nl/casacore/trunk/casacore/doc/notes/223.html

TaQL

Table Query Language www.astron.nl/casacore/trunk/casacore/doc/notes/199.html

- SQL-like with support for arrays and units
- Subqueries
- SELECT, UPDATE, INSERT, DELETE, CREATE TABLE, CALC
- No join, GROUPBY, HAVING
- Uses Python style (0-based indexing, C array order, end exclusive)

```
Select from my.ms orderby unique TIME
Select unique TIME from my.ms
     get unique times

Select from my.ms where ntrue(FLAG) == 0
Select from my.ms where not any(FLAG)        # faster way
     find rows without flagged data

Update my.ms set FLAG[,0]=FLAG[,0]||amplitude(DATA[,0]) > 3*median(amplitude(DATA[,0]))
     flag XX data based on a simple median filter (per row)

Update my.img set map = map - runningmedian(map, 25, 25)
     subtract background noise from image using median in a 51x51 box around each pixel

Insert into my.ms/HISTORY (TIME,MESSAGE) values (mjd(), "historystring")
     add a line to the HISTORY table of a MeasurementSet (converts automatically to sec)

Calc date()-"1Jan2000"
     how many days since 1-1-2000 (3328)
```

TaQL (cont'd)

Can be used from:

- C++

```
#include <tables/Tables/TableParse.h>
Table tab = tableCommand ('command');
```

- Python

```
import pyrap.tables as pt
tab = pt.taql ('select col1, col2 from sometable where col1=1
                orderby col1,col2 giving outname')
```

or

```
t = pt.table('sometable')
t1 = t.query ('col1=1', sortlist='col1, col2', columns='col1, col2',
               name='outname')
```

In C++ also

```
#include <tables/Tables/ExprNode.h>
Table t('sometable')
Table t1 = t(t.col('col1') == 1);
```

pyrap

python interface to casacore

- Manipulate tables, images from python
- data as numpy arrays in row major order (reverses axes!)
  - pyfits does the same

pyrap.tables
: create, read, write, selection, sort, iteration, display

pyrap.images
: read, write, tofits, statistics, regrid, display

pyrap.measures
: measures and frames conversions

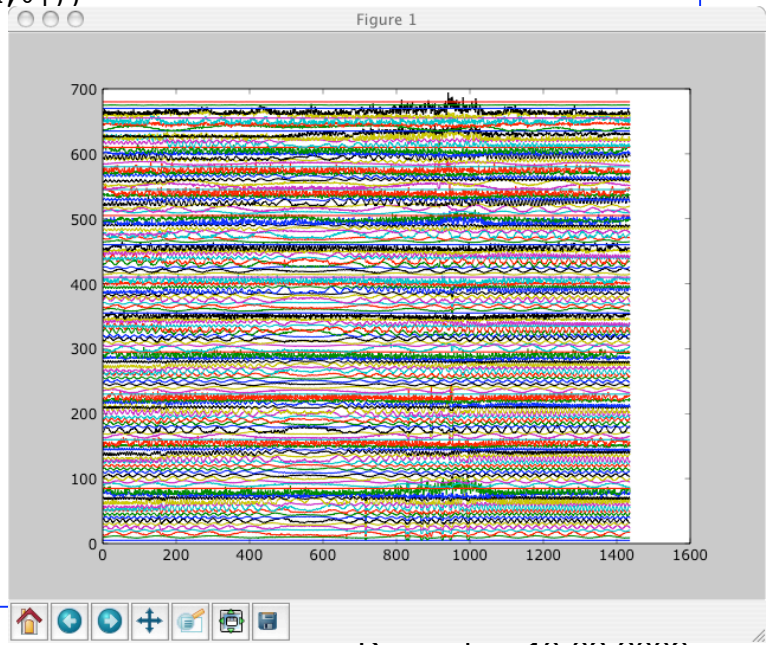
pyrap.functionals and pyrap.fitting
: fitting of data to function parameters (1-dim or more)

See

Download	pyrap.googlecode.com
Modules	www.astron.nl/casacore/trunk/pyrap/docs

pyrap.tables

```
import numpy
import pylab
import pyrap.tables as pt

def plotbl (ms, band=0, ch=0, sep=5.0, fig=None):
    t  = pt.table(ms);
    t1 = t.query ('DATA_DESC_ID=%d' % band)
    pylab.figure(fig)
    pylab.clf()
    offset = 0.0
    for t2 in t1.iter(["ANTENNA1","ANTENNA2"]):
        # Get XX data of given channel
        ampl = numpy.absolute (t2.getcolslice("DATA", [ch,0], [ch,0]))
        sc = sep/numpy.mean(ampl)
        ampl = ampl.reshape(ampl.shape[0])
        pylab.plot(sc*ampl + offset)
        offset += sep
    pylab.show()
```



pyrap.tables

```
>>> from pyrap.tables import *
>>> t=table('~/3C343.MS')
>>> t.getcell('DATA',0).shape
(64, 4)
# Determine percentage of flagged data
>>> a=tablecommand('calc sum([select nelements(FLAG) from ~/3C343.MS])')
>>> print a
[ 50030592.]
>>> b=tablecommand('calc sum([select ntrue(FLAG) from ~/3C343.MS])')
>>> print b/a
[ 0.33072053]
# Find out which baselines are heavily flagged
>>> for t2 in t.iter(["ANTENNA1","ANTENNA2"]):
...     a=tablecommand('calc sum([select ntrue(FLAG) from $t2])')                # note use of $t2
...     b=tablecommand('calc sum([select nelements(FLAG) from $t2])')
...     if a/b > 0.5:
...         print t2.getcell('ANTENNA1',0), t2.getcell('ANTENNA2',0),a/b
...
0 14 [ 1.]
0 15 [ 1.]
# Do it again without antenna 14 and 15
>>> a=tablecommand('calc sum([select ntrue(FLAG) from $t where ANTENNA2 not in [14,15]])')
>>> b=tablecommand('calc sum([select nelements(FLAG) from $t where ANTENNA2 not in [14,15]])')
>>> a/a
array([ 0.13312374])
# Show antenna info of 14
anttab = table(t.getkeyword('ANTENNA')           # or anttab=table('~3C343/ANTENNA')
anttab[14]                                        # contents of row 14 as a dict
{'NAME': 'RTE', 'MOUNT': 'equatorial', 'OFFSET': array([ 0.,  0.,  0.]), 'PHASED_ARRAY_ID': -1,
    '_OBSOLETE_ARRAY_ID': 0, 'STATION': 'station', 'DISH_DIAMETER': 25.0, 'ORBIT_ID': -1, 'POSITION':
    array([ 0.,  0.,  0.]), 'TYPE': 'GROUND-BASED', '_OBSOLETE_ANTENNA_ID': 14, 'FLAG_ROW': False}
```

pyrap.images

Uses numpy masked arrays

- for image mask=True means bad pixel; numpy opposite
- hence, getmask and putmask negate mask automatically

```
    import pyrap.images as pim
    im = pim.image('my.img')
    print im.statistics()
    npmaskedarray = im.get()
    nparray = im.getdata()
    im.tofits ('fitsfile')
    im.view()

# image expression
    im = pim.image("(image1 + image2 + image3) / 3")
    im.saveas ('avgimage')

# image concatenation
    im = pim.image(['image1', 'image2', 'image3'])
    im.saveas ('concimage')
```

CASA

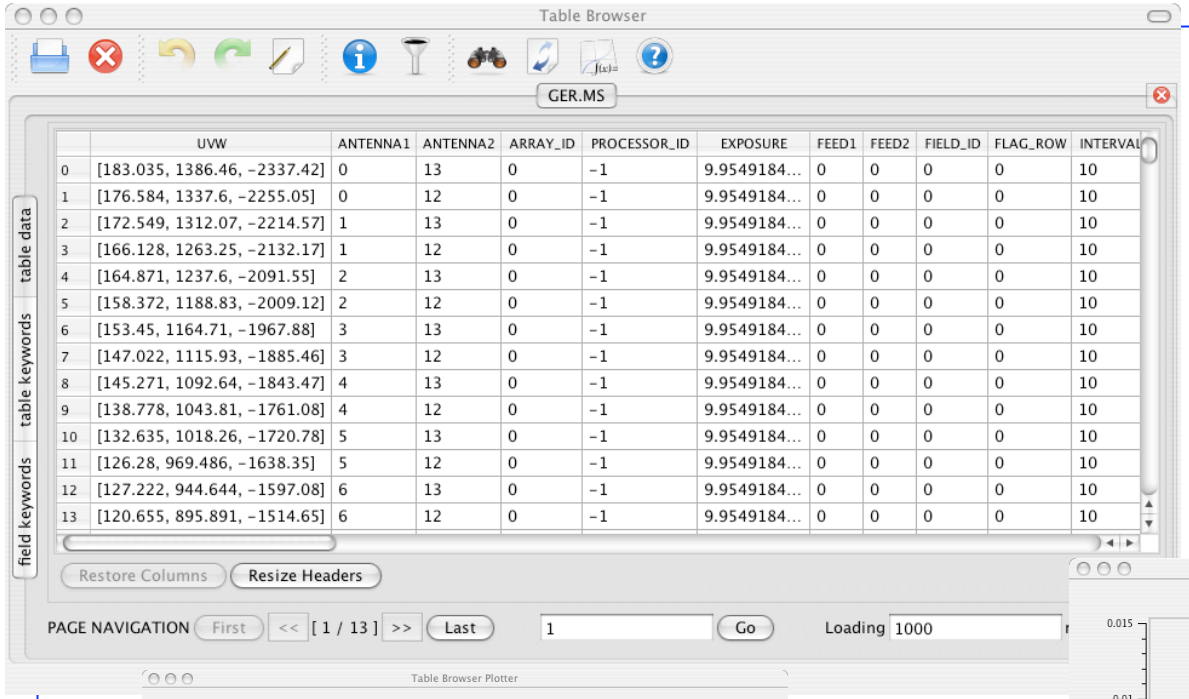
- ALMA data processing package
  - Based on casacore
- Has some useful tools:
  - casabrowser
    - view data in tabular way
    - editing, plotting, querying
  - casaviewer
    - viewing images (also FITS, Miriad, HDF5, expressions)
    - viewing and flagging MeasurementSets
  - plotms (in transition)
- Can be used from command line:

```
casaviewer ('my.img')
```

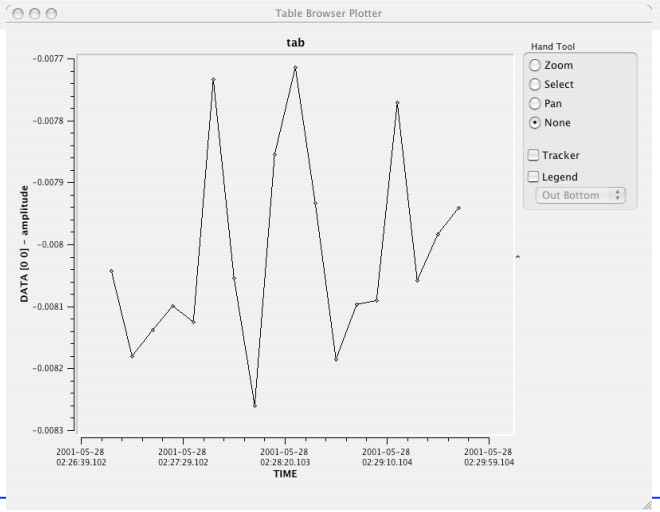
- Can be used from python (pyrap):

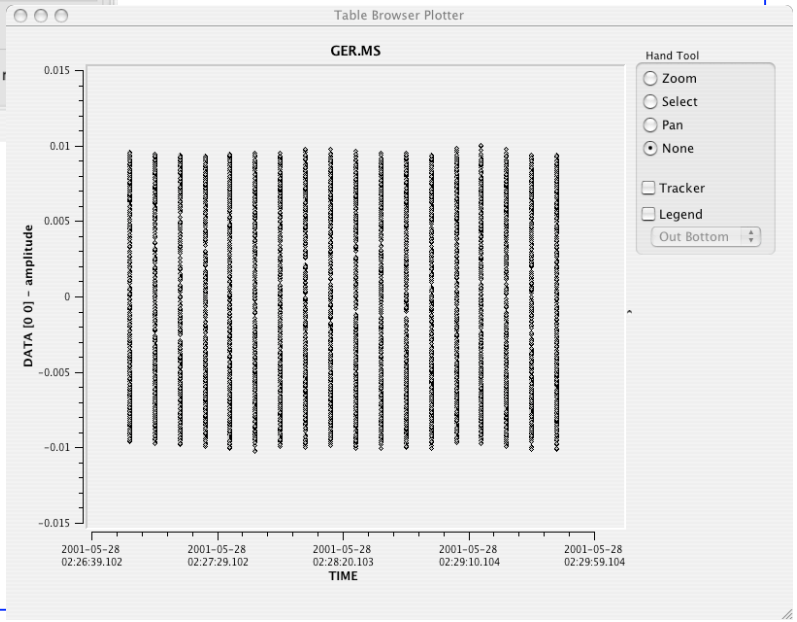
```
im = image('my.img')
im.view()
```

casabrowser




```
python
    from pyrap.tables import *
    t = table('GER.MS')
    t1 = t.query('DATA_DESC_ID=0')
    t1.browse()


casabrowser ('GER.MS')
```





casaviewer (ms)

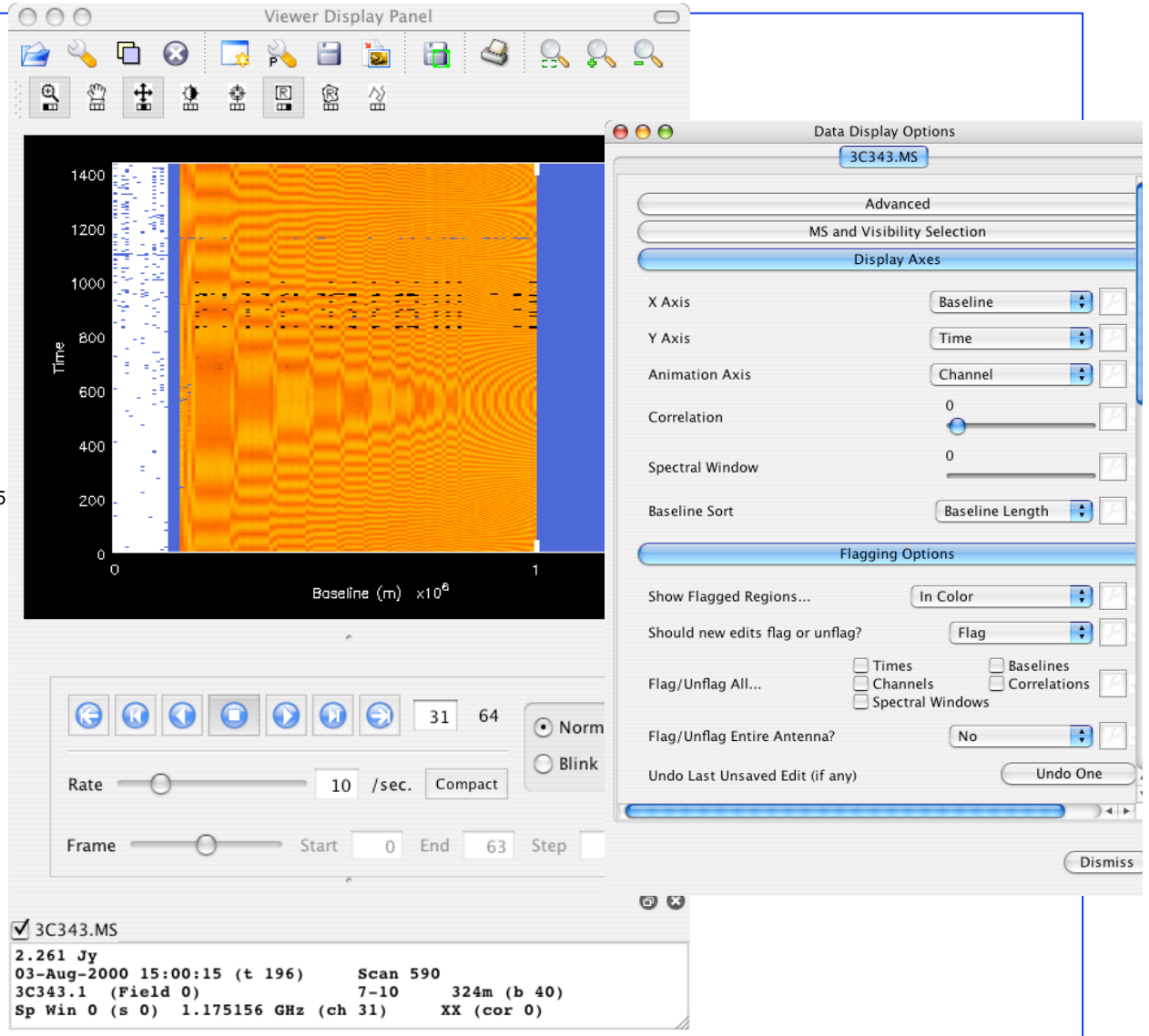
```
python
     from pyrap.tables import *
     t = table('GER.MS')
     t1 = t.query('DATA_DESC_ID=0')
     t1.view()


casaviewer ('GER.MS')
```

Sorting... Done.

/Users/diepen/3C343.MS
  Selected MS: Time slots: 1437 Baselines (incl. gaps): 15
  Correlations: 4 Channels: 64 Spectral Windows: 1

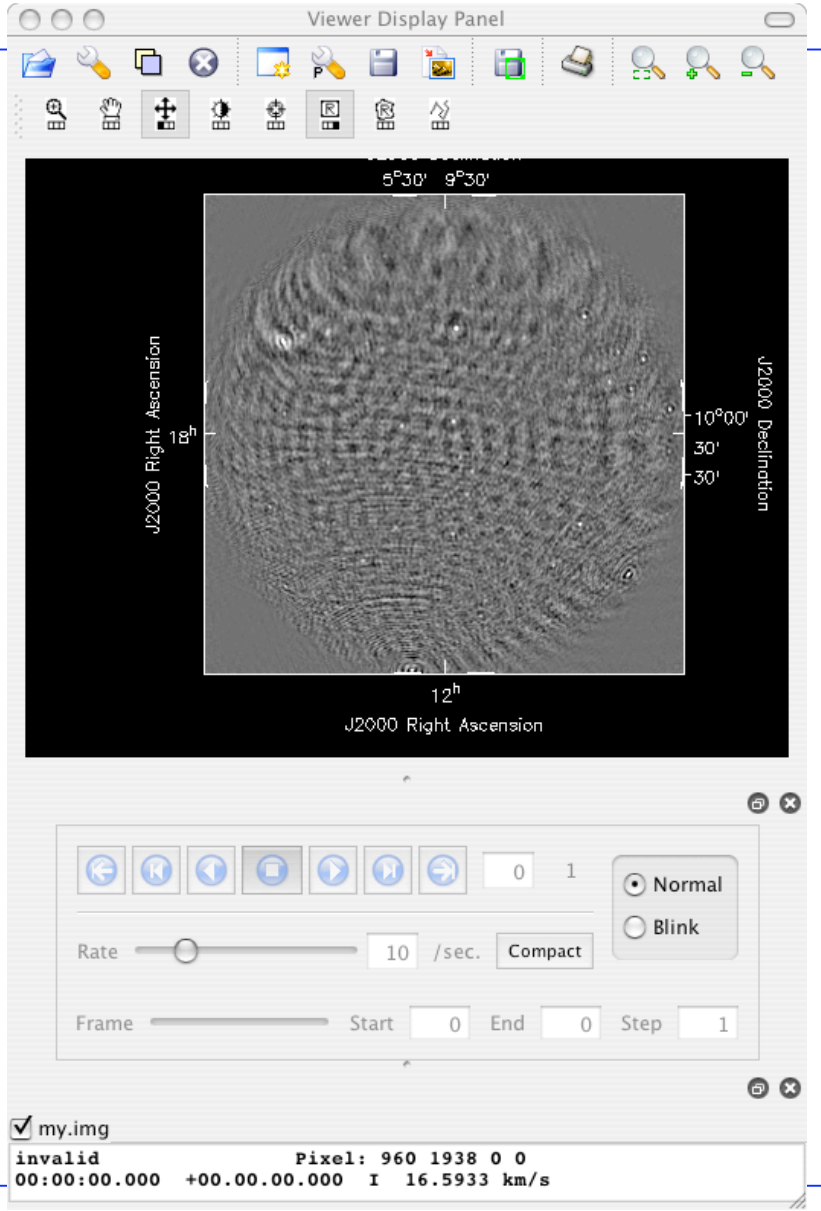
Loading MS vis. data: 28% 47% 80% 89% Done.

Resorting MS vis. data: 38% 95% Done.



casaviewer (image)

```
python
    from pyrap.images import *
    im = image('my.img')
    im.view()


casaviewer ('my.img')
```



MS exercises

For these exercises use MeasurementSet ex1.ms.

1. Find out the spectral info (#bands, #channels, freqs) (use subtable SPECTRAL_WINDOW).
2. Create a subset excluding antennae 14 and 15.
3. Determine in the subset how many data points exceed 5 times the median of the spectrum per timestamp per baseline.
4. Determine how many of these data points are already flagged.
5. Make a plot of the XX amplitude (averaged spectrum) vs time for all baselines.
6. Make a plot of each channel for a given baseline.
7. Start casabrowser on the subset.
8. Start casaviewer on the subset.

Image exercises

For all exercises use images sb[0369].img.

These are 4 Stokes I images of LOFAR subbands 0,3,6,9.

1. Concatenate these images into a single cube and print the coordinate info.
2. Create a new image spec-cont.img where the continuum data is subtracted from the spectral line data. Calculate the continuum by averaging the 4 bands.
3. View the difference of the two cubes.