International Centre for Radio Astronomy Research

EVLA software developments

James Miller-Jones

ICRAR – Curtin University

james.miller-jones@curtin.edu.au

Drawing on material from D. Petry, S. Myers and A. Richards

Curtin University

The Government of Western Australia

The University of Western Australia
Achieve International Excellence

Overview

- The EVLA: new capabilities and challenges
- EVLA software developments: CASA
    - Philosophy and architecture
    - Look and feel
    - Data reduction pathways
- New algorithms and development

The Expanded Very Large Array

Upgrading the existing VLA with 21st century technology

- Operation at any frequency from 1-50 GHz
- Up to 8GHz bandwidth per polarization
- Improvement in continuum sensitivity by a factor 5-20 (rms noise 2-6 μJy in 1h)
- New WIDAR correlator
- Frequency resolution 0.2 Hz – 2 MHz
- Between 16,384 and 4,194,304 spectral channels
- Dynamic range $>10^6$
- Dynamically scheduled observing
- Automatic processing to produce default images
- Estimated completion date 2012
- First science data taken 2010 March 2nd

- Image courtesy of NRAO/AUI

EVLA: New challenges

- **Enhanced bandwidth**
  - RFI flagging
  - Frequency-dependent source spectra
  - Frequency-dependent primary beams
- **Enhanced sensitivity**
  - Sources visible further down the primary beam
  - Source confusion important at low frequencies
  - Previously unimportant effects noticeable
- **Data volumes**
  - Data transfer becomes an issue
  - Increased processing power required
  - Manual inspection difficult or impossible
- **Requires pipelining capabilities**
  - AIPS (POPS, ParselTongue)
  - **CASA**

Common Astronomy Software Applications

CASA: the EVLA data analysis package

- Refocussed (and funded) in 2003 from AIPS++
- Implements the Measurement Equation
- C++ with Python wrappers and an iPython interface
- Works with Measurement Sets (MSs)
- In public release since Dec 2009
  - Linux (RHEL)
  - Mac OS X
- Two parts:
  - *casacore* (general utilities, infrastructure)
  - *casa non-core* (user interface, high-level analysis routines, viewers)

http://casa.nrao.edu

CASA Architecture

FEATURES:

1. A data structure
   Tables, Calibration tables, Images, MS
2. Data import/export facilities
   SDM to MS, MS to UVFITS, FITS to Image
3. Data access, display, editing tools
   Table browser, Viewer, Plotter
4. Science analysis tools (tasks)
   C++ classes for calibration and imaging
5. High-level analysis procedures
   Special procedures for tasks, e.g. CLEAN
6. Programmable command-line interface with scripting
   Python/IPython
7. Documentation
   Cookbook, web guides and tutorials

The Measurement Equation

$$\vec{V}_{ij} = J_{ij}\vec{V}_{ij}^{IDEAL}$$

Hamaker, Bregman & Sault (1996)

$$\vec{V}_{ij} = M_{ij}B_{ij}G_{ij}D_{ij}E_{ij}P_{ij}T_{ij}\vec{V}_{ij}^{IDEAL}$$

- Ideal visibilities on each baseline corrupted by a sequence of effects
- Factor these effects into 4x4 matrices
- Use observations of calibrators to solve for these:
  - M_{ij} = baseline-based correlator errors
  - B_{ij} = bandpass
  - G_{ij} = electronic gain due to signal path between feed and correlator
  - D_{ij} = instrumental polarization
  - E_{ij} = effects introduced by telescope optics (elevation etc.)
  - P_{ij} = parallactic angle
  - T_{ij} = tropospheric effects (e.g. opacity)
- Solution of linear equations via χ^2 minimization

The Measurement Set

Accessible directories of tables

- Developed by Cornwell, Kemball & Wieringa between 1996 and 2000
- Designed to store visibility and single-dish data
- *Supports any setup of radio telescopes*
- Supports data processing using Measurement Equation
- Storage in **tables**

- **MAIN**
  - Original visibility data
  - One row per integration per baseline per spectral window
- **MODEL**
  - FT of model image or calibrator flux density
- **CORRECTED**
  - Copy of visibilities with applied calibration tables

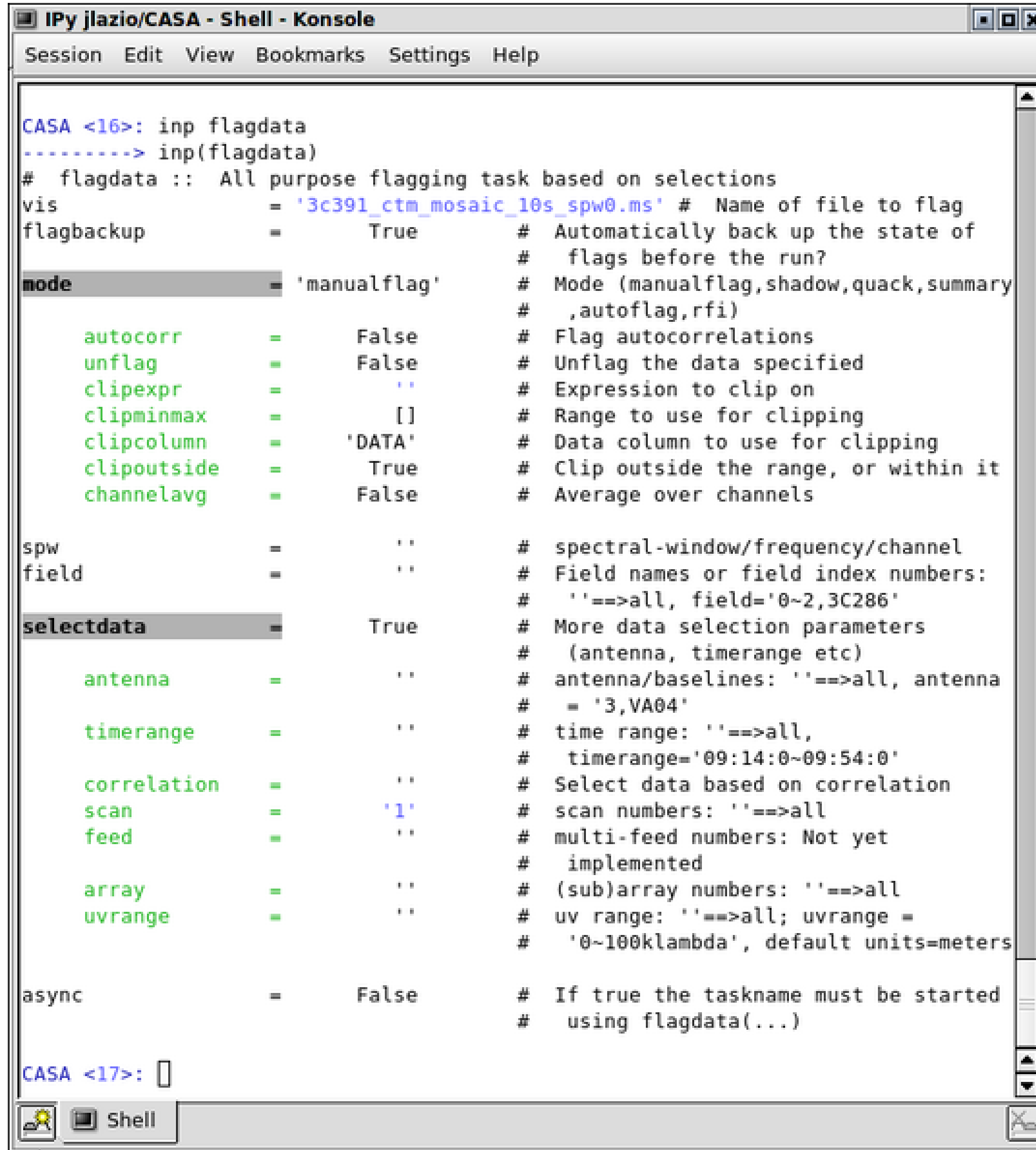
- Flag tables
- Extra administrative tables

- Size triples when you begin processing

CASA interface

Command-line interface

- Run in the shell
- Command-line with scripting
- iPython
- Task interface (similar to AIPS, MIRIAD)
- Expandable menus
- Validation checking
- Tab completion
- Auto-paretheses
- Access to shell commands (!)
- Command-line numbering
- History
- execfile for script execution

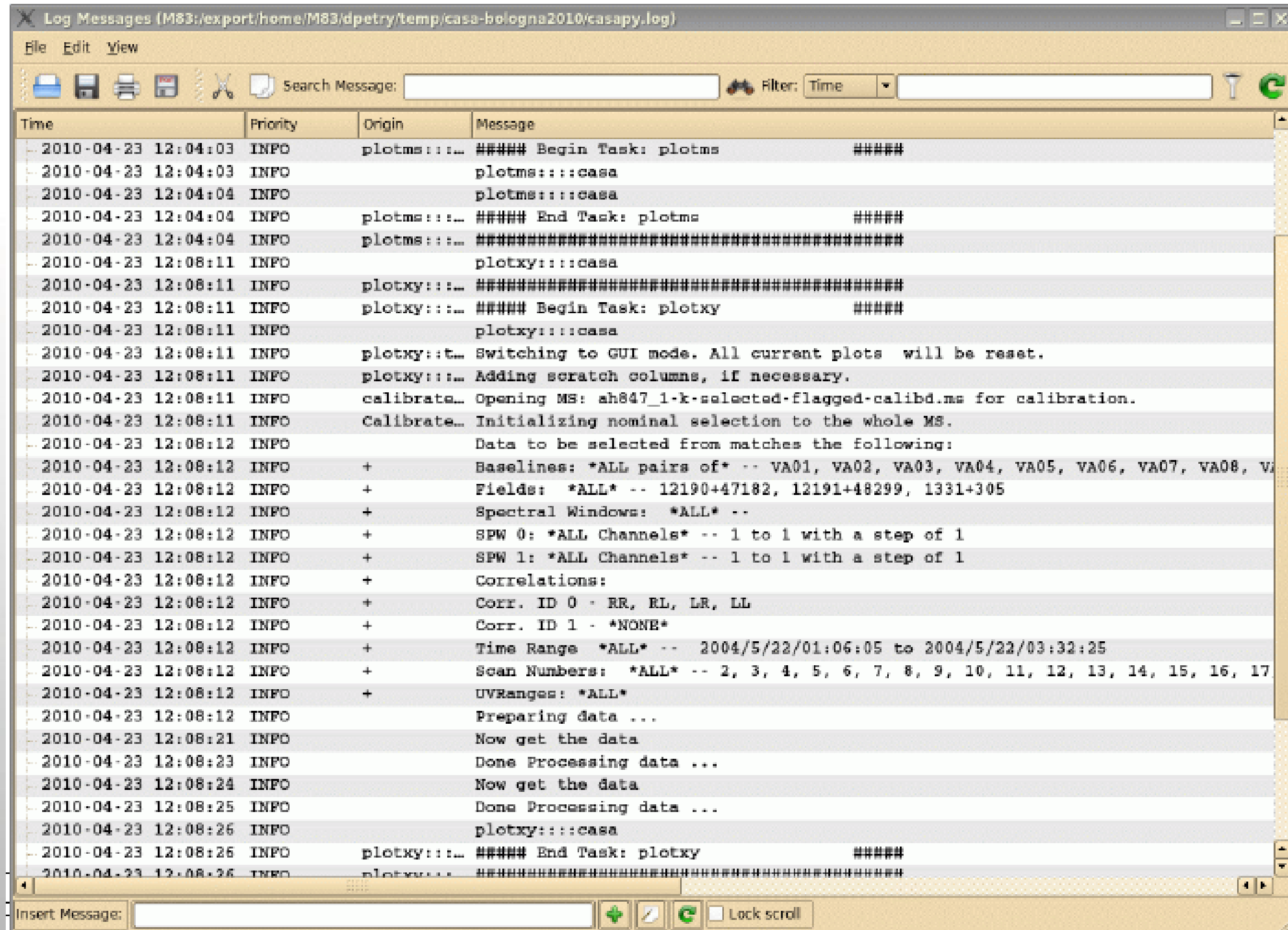


```
IPy jlazio/CASA - Shell - Konsole
Session  Edit  View  Bookmarks  Settings  Help

CASA <16>: inp flagdata
---------> inp(flagdata)
#  flagdata :: All purpose flagging task based on selections
vis                 = '3c391_ctm_mosaic_10s_spw0.ms' #  Name of file to flag
flagbackup          =       True        #  Automatically back up the state of
                                        #   flags before the run?
mode                = 'manualflag'      #  Mode (manualflag,shadow,quack,summary
                                        #   ,autoflag,rfi)
     autocorr       =      False        #  Flag autocorrelations
     unflag         =      False        #  Unflag the data specified
     clipexpr       =         ''        #  Expression to clip on
     clipminmax     =         []        #  Range to use for clipping
     clipcolumn     =     'DATA'        #  Data column to use for clipping
     clipoutside    =       True        #  Clip outside the range, or within it
     channelavg     =      False        #  Average over channels

spw                 =         ''        #  spectral-window/frequency/channel
field               =         ''        #  Field names or field index numbers:
                                        #   ''==>all, field='0~2,3C286'
selectdata          =       True        #  More data selection parameters
                                        #   (antenna, timerange etc)
     antenna        =         ''        #  antenna/baselines: ''==>all, antenna
                                        #   = '3,VA04'
     timerange      =         ''        #  time range: ''==>all,
                                        #   timerange='09:14:0~09:54:0'
     correlation    =         ''        #  Select data based on correlation
     scan           =        '1'        #  scan numbers: ''==>all
     feed           =         ''        #  multi-feed numbers: Not yet
                                        #   implemented
     array          =         ''        #  (sub)array numbers: ''==>all
     uvrange        =         ''        #  uv range: ''==>all; uvrange =
                                        #   '0~100klambda', default units=meters

async               =      False        #  If true the taskname must be started
                                        #   using flagdata(...)

CASA <17>: 
Shell
```

CASA interface

Logger

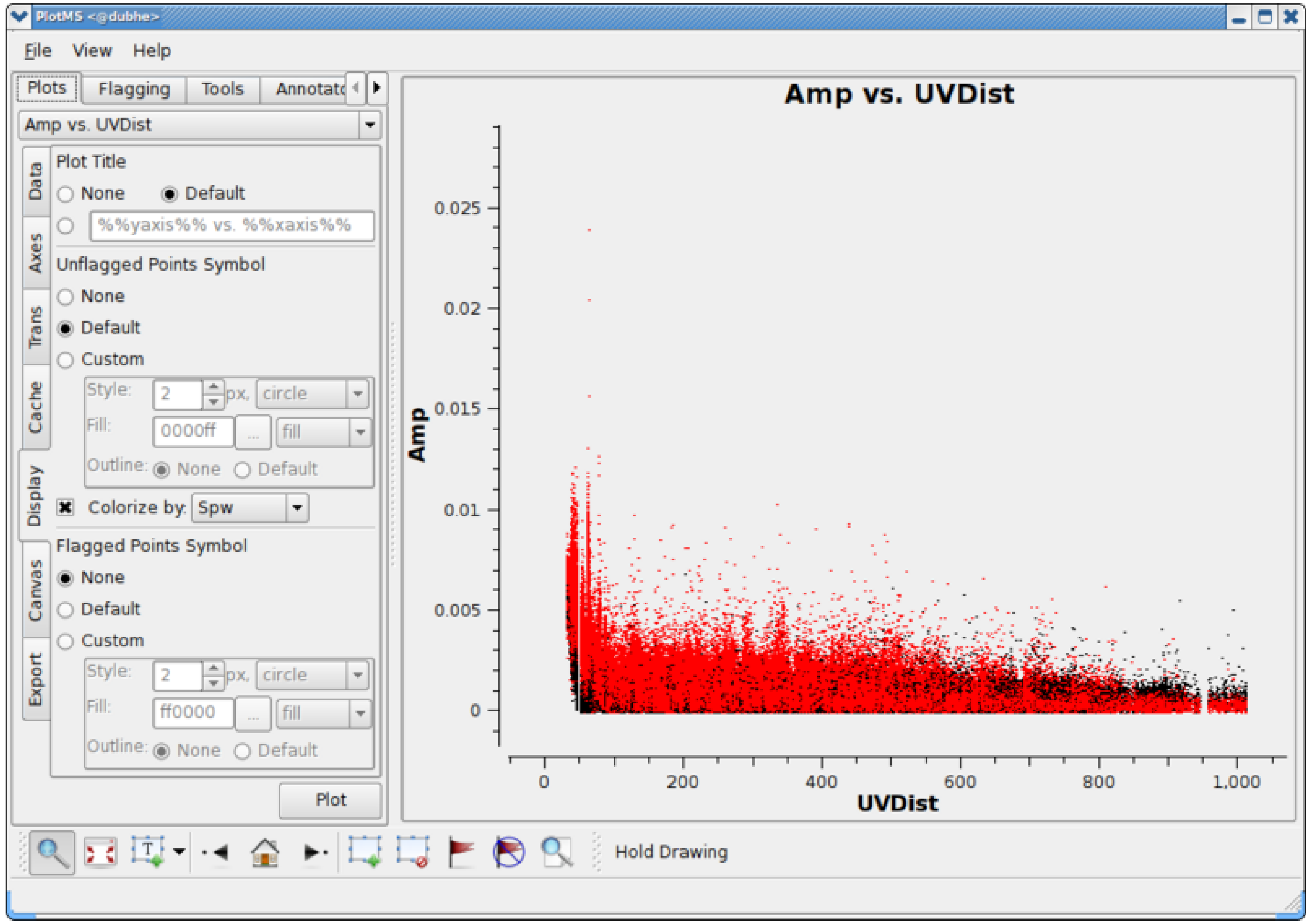
- Monitoring
- Debugging
- Automatic output to log file
- Results of task execution



CASA interface

Plotting

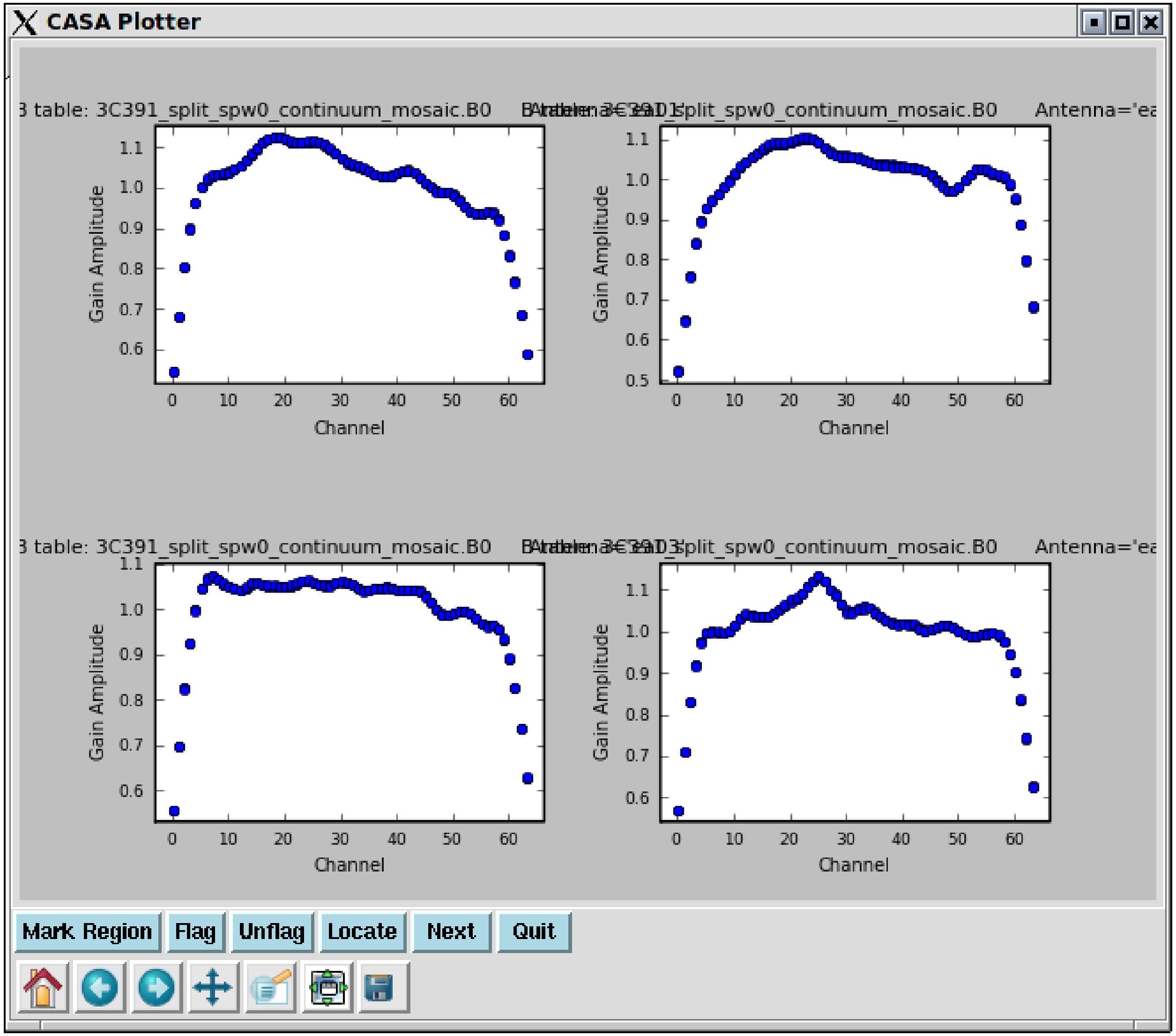
- Visibility data
- Range of variables
- Averaging
- Color-coding
- Interactive flagging
- Qt-based
- Uses generic plotting class which uses Qwt



CASA interface

Plotting of tables

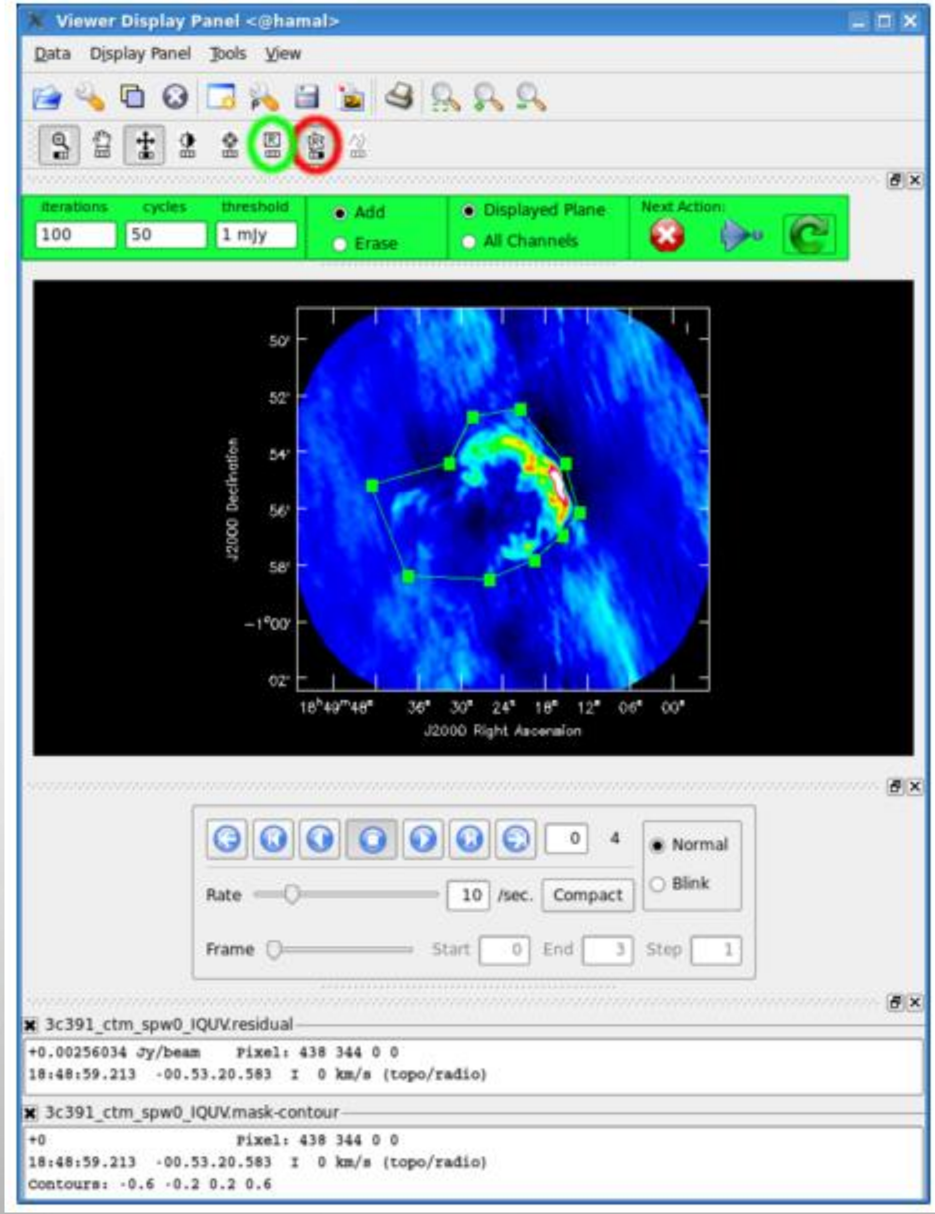
- Inspection of calibration tables
- Control over variables
- Iteration
- Subplot support



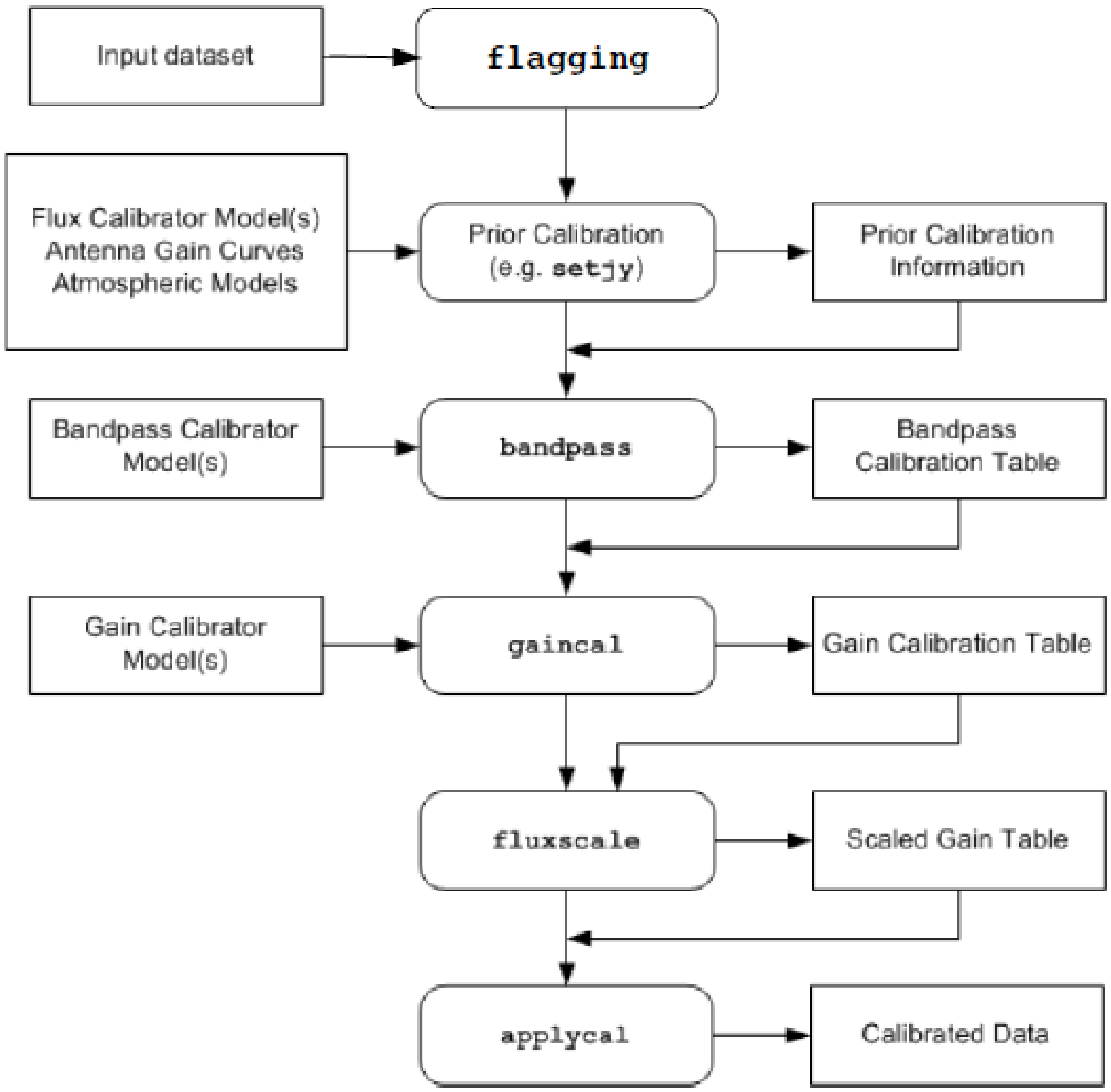
CASA interface

Imaging and display

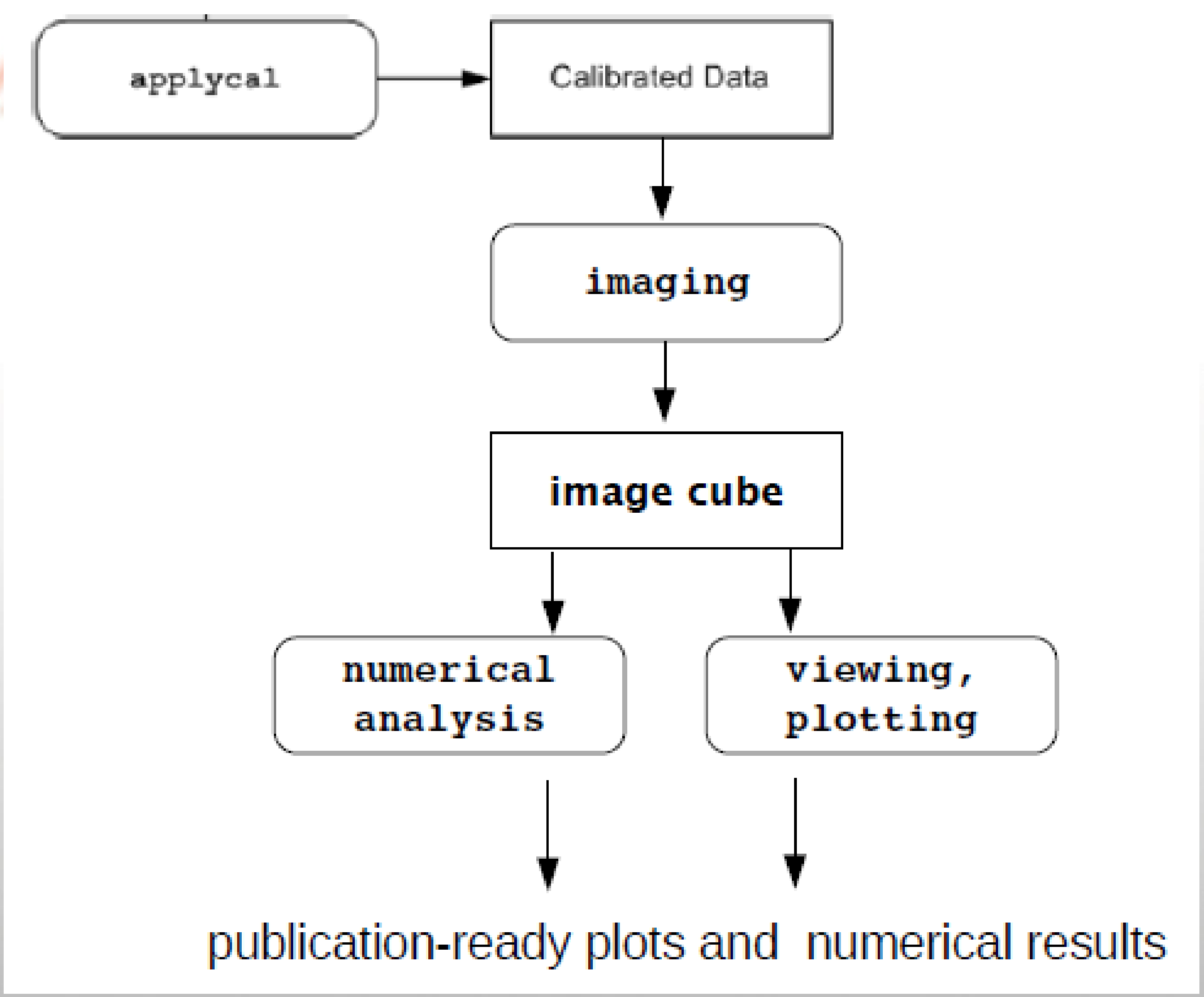
- Viewer
  - Uses Qt widget set
  - Rendering based on pgplot
  - Step through images planes (frequencies, polarizations)
  - Support for ds9 region files
  - Box settings (clean/image analysis)
  - Zoom/contrast enhancements
  - Overlays



Data analysis in CASA



Data analysis in CASA



Scripting

execfile

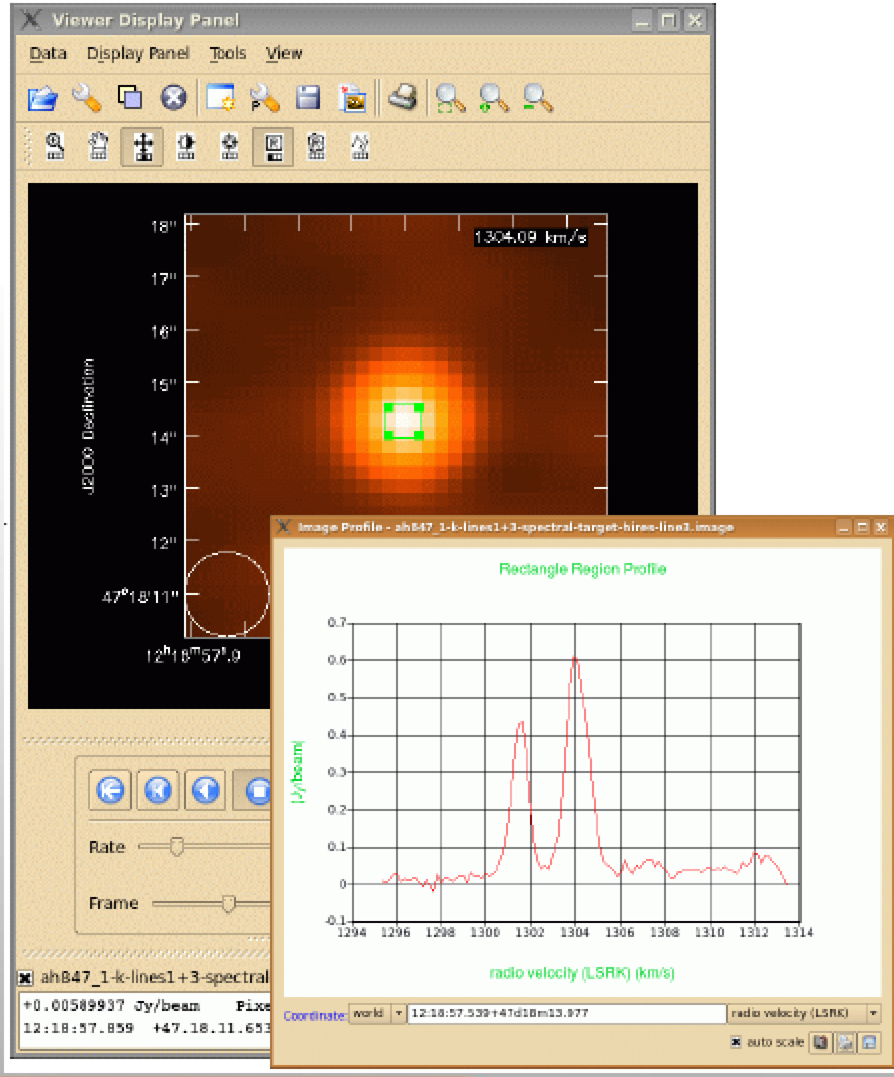
- Flexible use:
  - Task-based command line interface
  - Python function calls

- Scripts written as a sequence of function calls
- Run with execfile

```
setjy(vis='3c391_ctm_mosaic_10s_spw0.ms',field='J1331+3030',
modimage='/home/casa/data/nrao/VLA/CalModels/3C286_C.im', standard='Perley-Taylor 99',
fluxdensity=-1)

gaincal(vis='3c391_ctm_mosaic_10s_spw0.ms', caltable='3c391_ctm_mosaic_10s_spw0.G0',
field='J1331+3030', refant='ea21', spw='0:27~36', calmode='p', solint='int', minsnr=5, solnorm=T,
gaintable=['3c391_ctm_mosaic_10s_spw0.antpos'])

bandpass(vis='3c391_ctm_mosaic_10s_spw0.ms',caltable='3c391_ctm_mosaic_10s_spw0.B0',fi
eld='J1331+3030',spw='',refant='ea21',solnorm=True,combine='scan', solint='inf',bandtype='B',
gaintable=['3c391_ctm_mosaic_10s_spw0.antpos','3c391_ctm_mosaic_10s_spw0.G0'])
```

Key imaging algorithms and functionality

- Full polarization handling
- Spectral line cubes
- Large scale structure: Mosaicking and joint deconvolution
- Diffuse emission: Multi-scale CLEAN
- Wide field imaging: *w*-projection and outlier fields
- Multi-frequency synthesis (Sault-Wieringa algorithm)
- (*To come:*) Direction-dependent primary beam correction

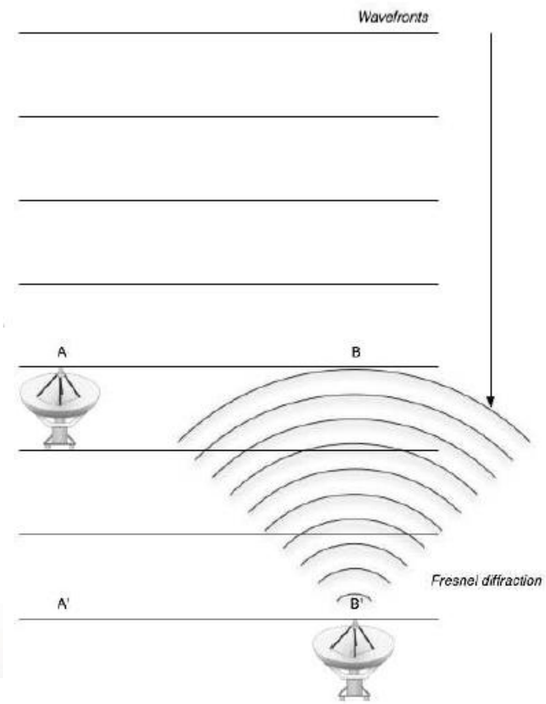


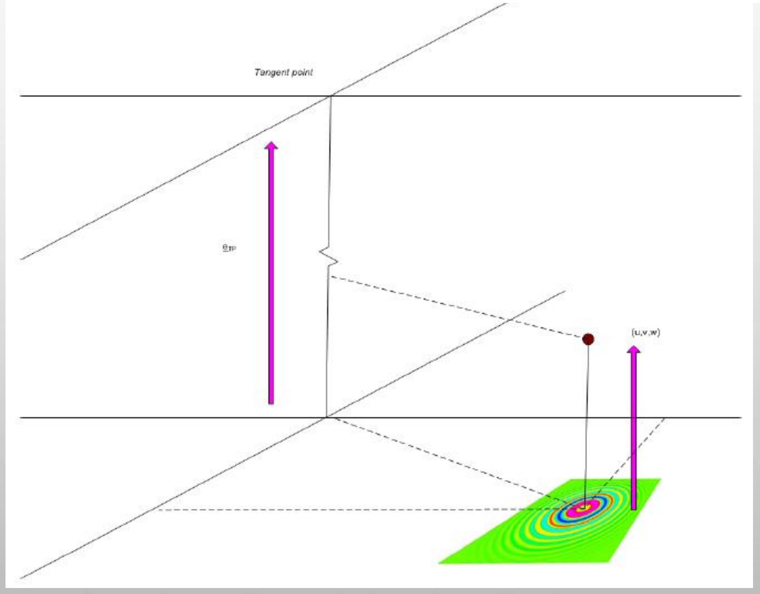
w-projection

- 2D-array has non-coplanar baselines
- Relationship between visibility *V(u,v,w)* and sky brightness *I(l,m)* is no longer a 2-D FT

$$V(u,v,w) = \int\int \frac{I(\ell,m)}{\sqrt{1-\ell^2-m^2}} e^{-2\pi i[u\ell+vm+w(\sqrt{1-\ell^2-m^2}-1)]} d\ell dm$$

- Reproject from *(u,v,w)* to *(u,v,w=0)* plane
- Convolution relation between *V(u,v,w)* and *V(u,v,w=0)*
- Sample spread over plane with footprint scaling with *w*
- Single visibility now sensitive to a range of spatial frequencies
- Better performance in both speed and error control compared to facetting





Cornwell, Golap & Bhatnagar

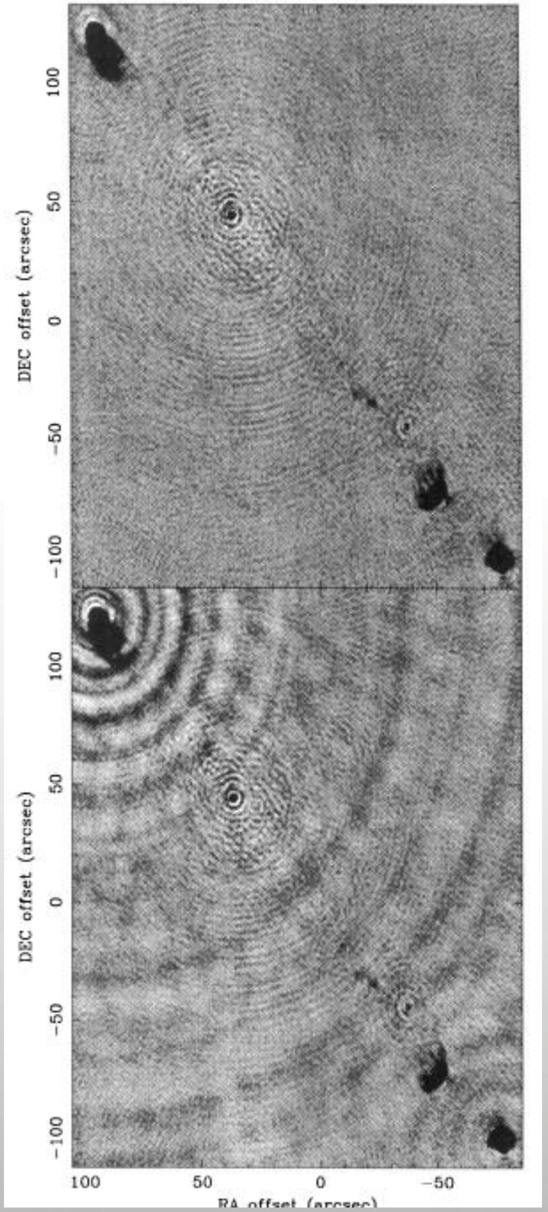
Sault-Wieringa algorithm

Frequency-dependent fluxes

- Source spectral indices differ across the field
- Frequency-dependent primary beam causes apparently steep spectra near field edges
- Model frequency dependence of sky emission with a Taylor series about a reference frequency

$$I_{\mathrm{D}}(\ell, m) = I(\nu_0) B_0(\ell, m) + \nu_0 \frac{\partial I}{\partial \nu} B_1(\ell, m).$$

- Choose number of Taylor terms
- Make use of full bandwidth without amplitude errors

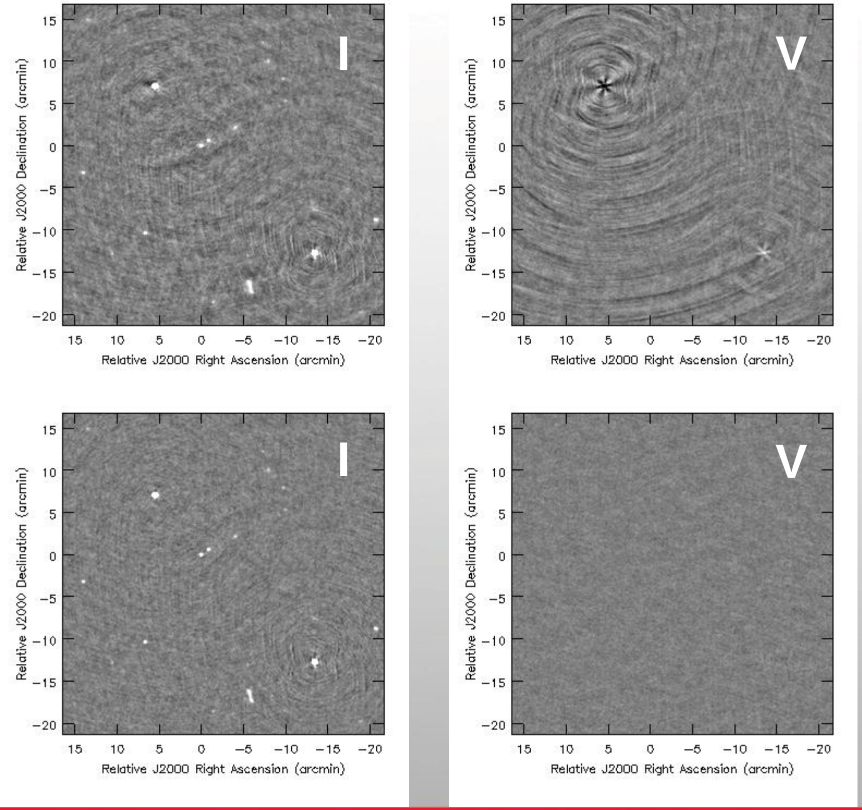


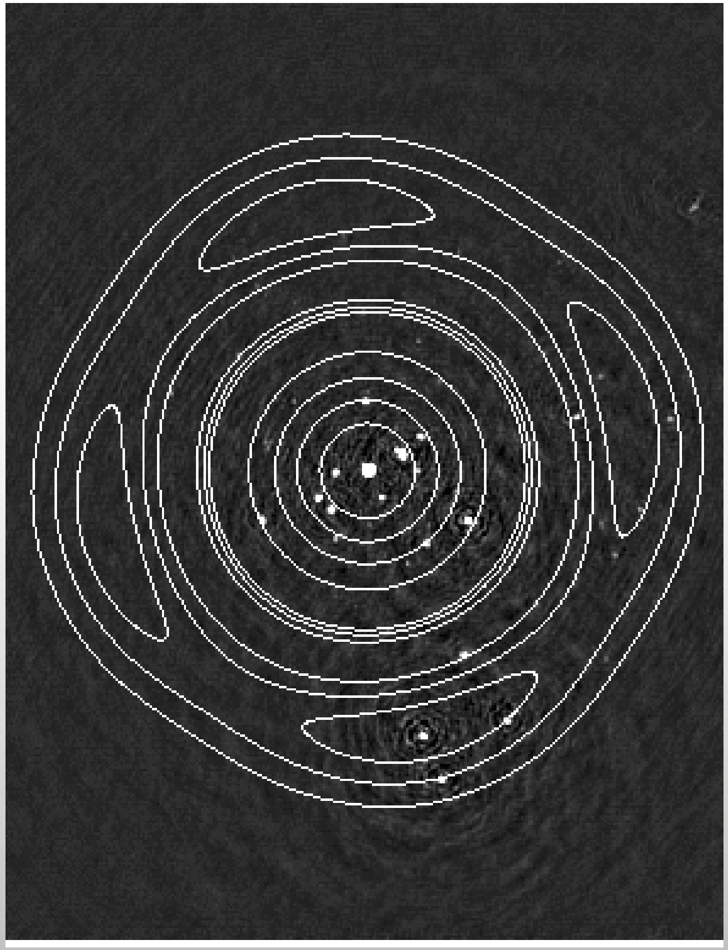
Sault & Wieringa (1994)

A-projection

Direction-dependent errors with wide bandwidths

- Primary beam rotates on the sky
- Creates low-level direction-dependent errors in the images
- AWProject algorithm corrects for rotating azimuthally asymmetric primary beams with polarization squint





Bhatnagar et al. (2008)

Hardware recommendations

- Imaging is most time-consuming step
- Gridding within imaging dominates processing time
- I/O limited process
- Parallelization still under development

- Dual core processor
- 12-24GB of memory per node
- 2-3 TB hard drive space

- See: http://science.nrao.edu/evla/postproc/hardware.shtml

Summary

- The EVLA is already operational
- Its new sensitivity is requiring new processing capabilities and algorithms
- CASA is the designated software package for EVLA data processing
- Based on the Measurement Equation
- Wide range of capabilities
  - Scripting
  - New algorithms
- Development still ongoing

Useful links

1. CASA homepage: http://casa.nrao.edu/
2. Casaguides wiki: http://casaguides.nrao.edu