



# The MeqTree module working prototype for LOFAR calibration

Jan Noordam (noordam@astron.nl)

URSI GA New Delhi  
Oct 24, 2005

# The LOFAR challenge

- ◆ Pathological ionosphere
- ◆ Very crowded fields (confusion, flops)
- ◆ Very bright sources (dynamic range)
- ◆ Variable (f,t,k) station beamshapes
- ◆ Instrumental polarisation (X/Y voltage beams)
- ◆ High station beam sidelobes (all-sky imaging)
- ◆ New types of instrumental effects (e.g. BSR)
- ◆ Huge data volume (Tbytes/hr)



# Ionosphere movie (GMRT)



# The bottom line

- ◆ The new generation of large radio telescopes (LOFAR, ALMA, SKA) needs software around a more complex **Measurement Equation (M.E.)** than the existing packages can offer
- ◆ A user should not depend on package builders to implement a particular M.E.
- ◆ MeqTree is an attempt to do this (the only one!)
- ◆ The existing telescopes need something also...



# The Measurement Equation

$$\vec{V}_{ij}(u, v) = \sum_k \int df \int dt \left( J_{ik} \otimes J_{jk}^* \right) S \vec{I}_k$$

See also:

AIPS++  
note 185

Hamaker-Bregman-Sault formalism

Station-pair (i,j): Vector of 4 ‘correlations’

Matrix equation: full polarization

Instrumental Jones matrices  $J$  (2x2)

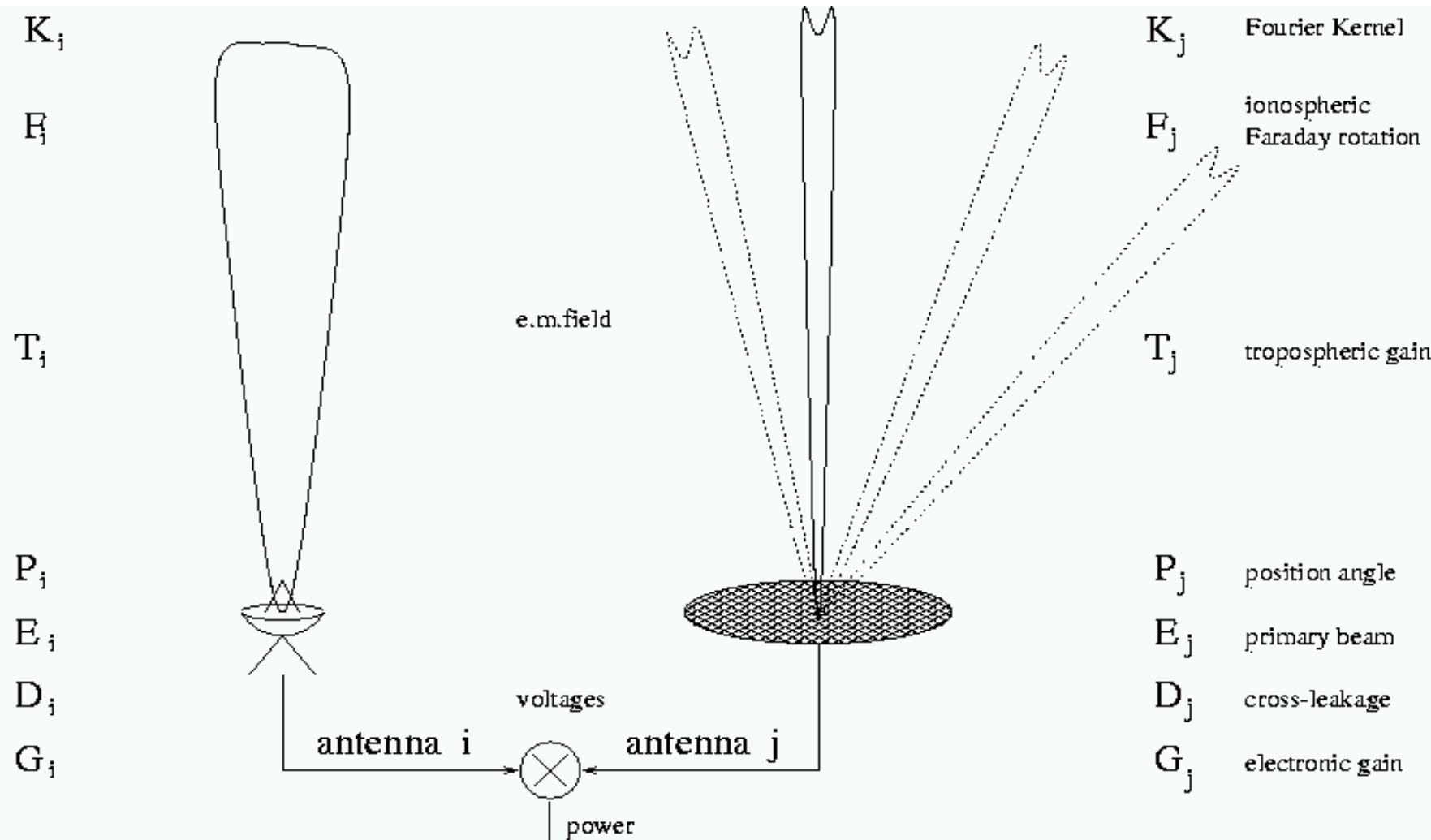
Stokes matrix  $S$  (4x4)

Sum over sources ( $k$ ): Flux [I,Q,U,V]

Integral over time-freq domain



# M.E. instrumental effects

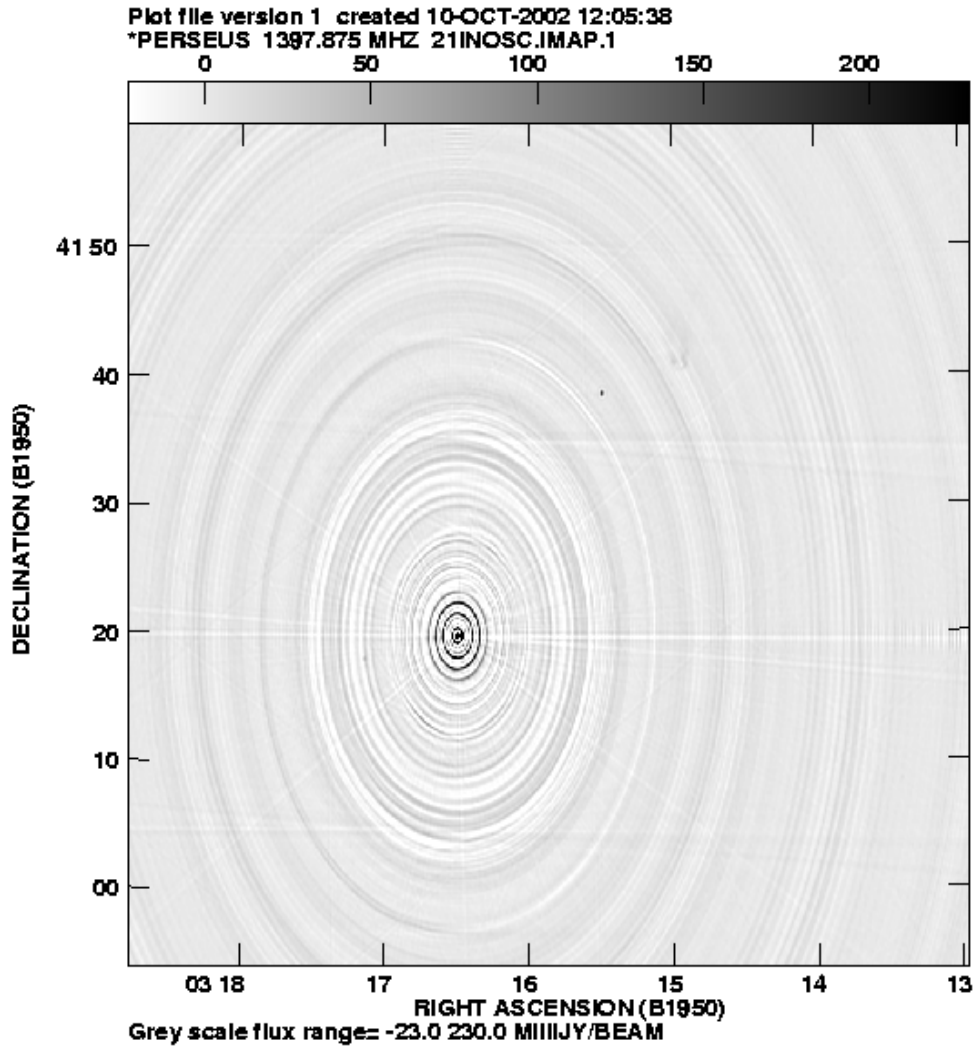


$$M_{ij} = G_i D_i E_i P_i T_i F_i K_i$$

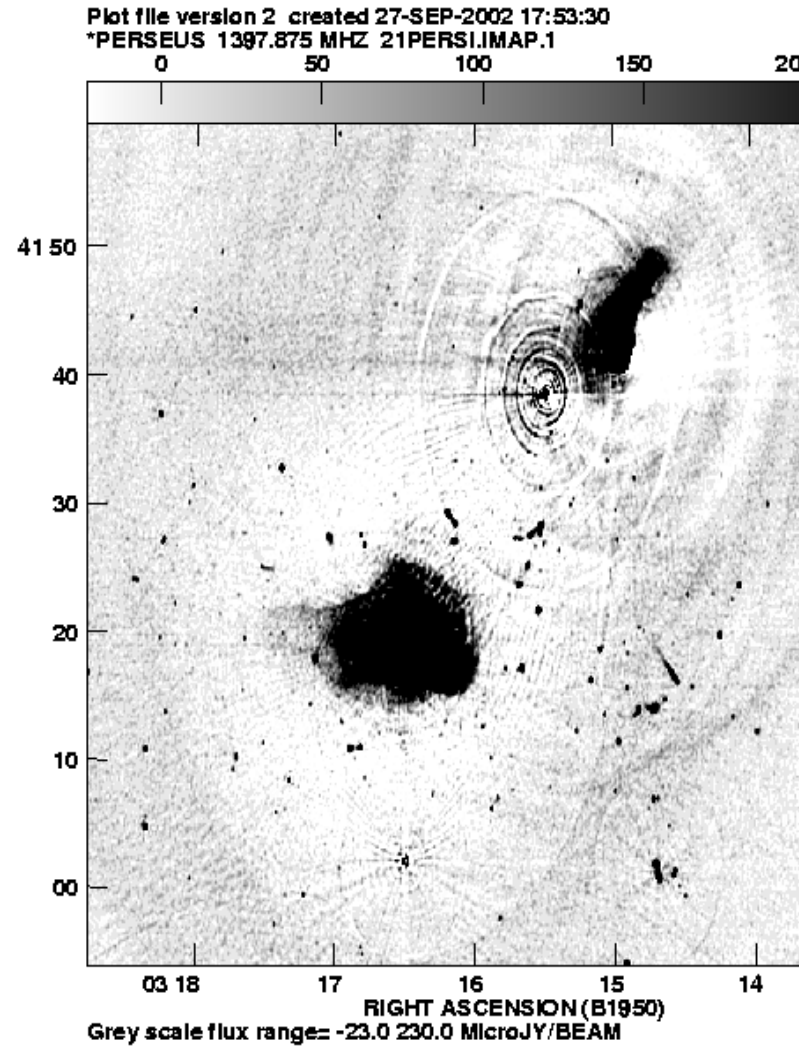
$$M_{ij} = G_j D_j E_j P_j T_j F_j K_j$$

$M_{ij}$  multiplicative interferometer-based effects  
 $A_{ij}$  additive

# 3C84, WSRT, 21cm



Dynamic range: 100 : 1



Dynamic range: 500.000

# The big issues

- ◆ Image-plane effects
  - ⊕ Extra processing
  - ⊕ Apply to extended sources
- ◆ Minimise the number of parameters
  - ⊕ Minimum Ionospheric Model (MIM)
  - ⊕ Beamshapes
- ◆ Data volume (performance)



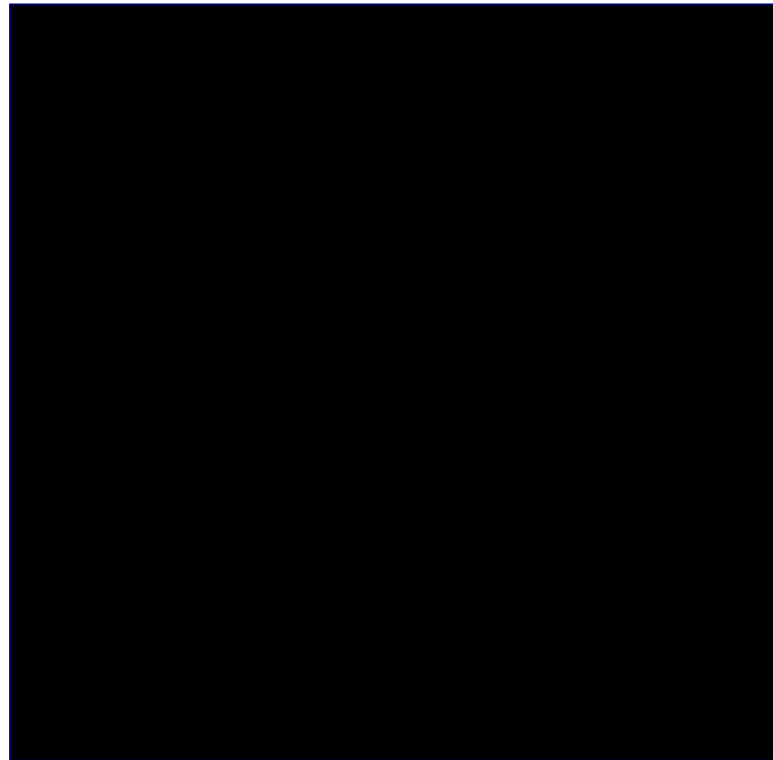
# Path to LOFAR calibration software

- ◆ Two separate teams:
  - ⊕ Development: MeqTree module and AIPS++
  - ⊕ Implementation: Black Board Selfcal (BBS)
- ◆ MeqTree is a tool for M.E. development
  - ⊕ Reduce the number of parameters
  - ⊕ Low threshold
  - ⊕ Exercise on real data (WSRT/WHAT, VLA74 HMz ...)
- ◆ Involve lots of people (especially next generation)
- ◆ Continue with MeqTree module afterwards

# Alternate between two domains

- ◆ Instrumental errors are in Fourier domain
  - ⊕ They vary in time and frequency
  - ⊕ They vary over the Field of View (!)
- ◆ Bright sources are subtracted in Fourier domain
- ◆ Faint sources are found in image plane
  - ⊕ But subtracted in Fourier plane (PSF)
- ◆ Residual images must be deconvolved

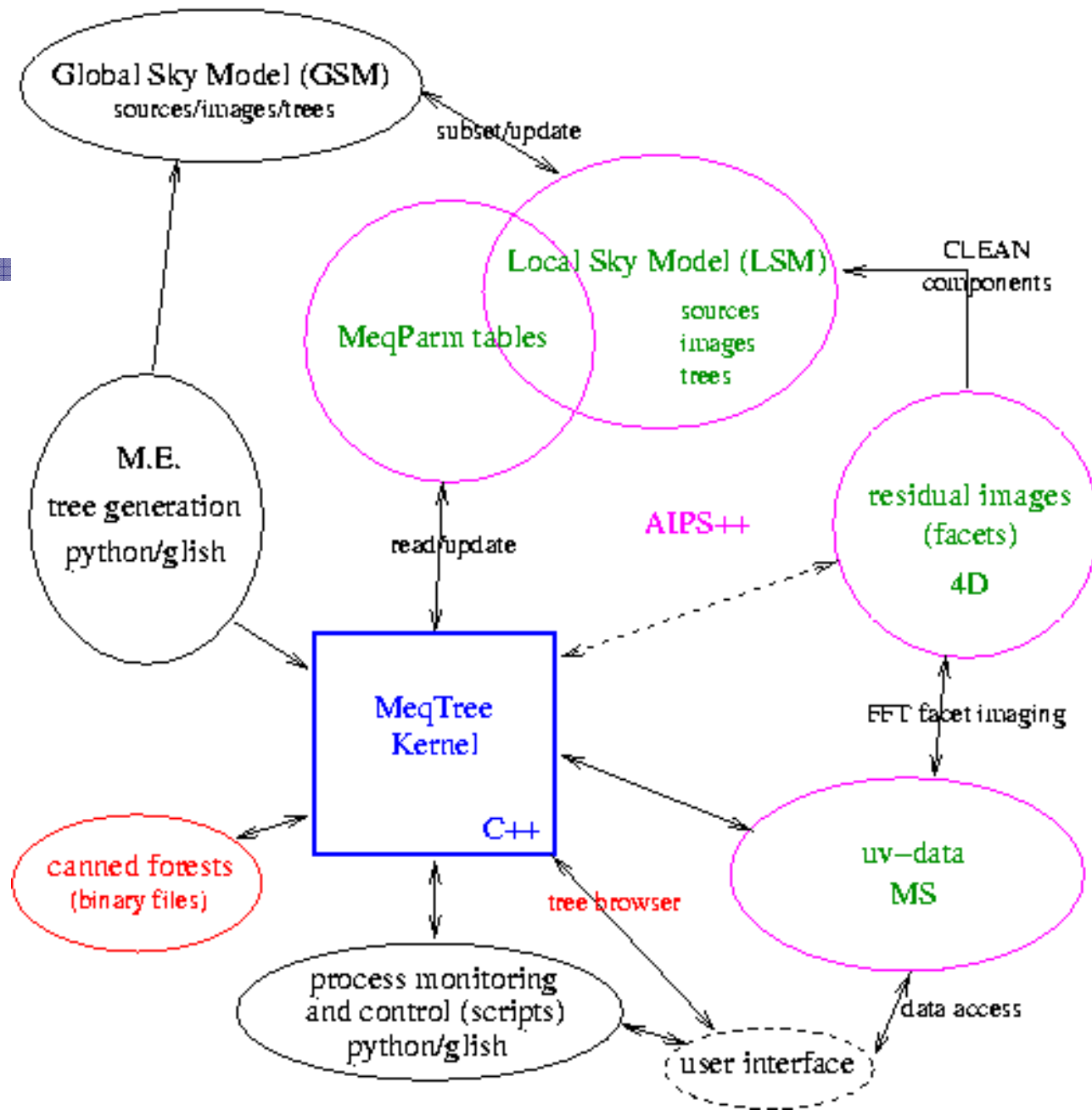
# The result of successful calibration





# Part II: MeqTree Module Overview

# MeqTree Environment



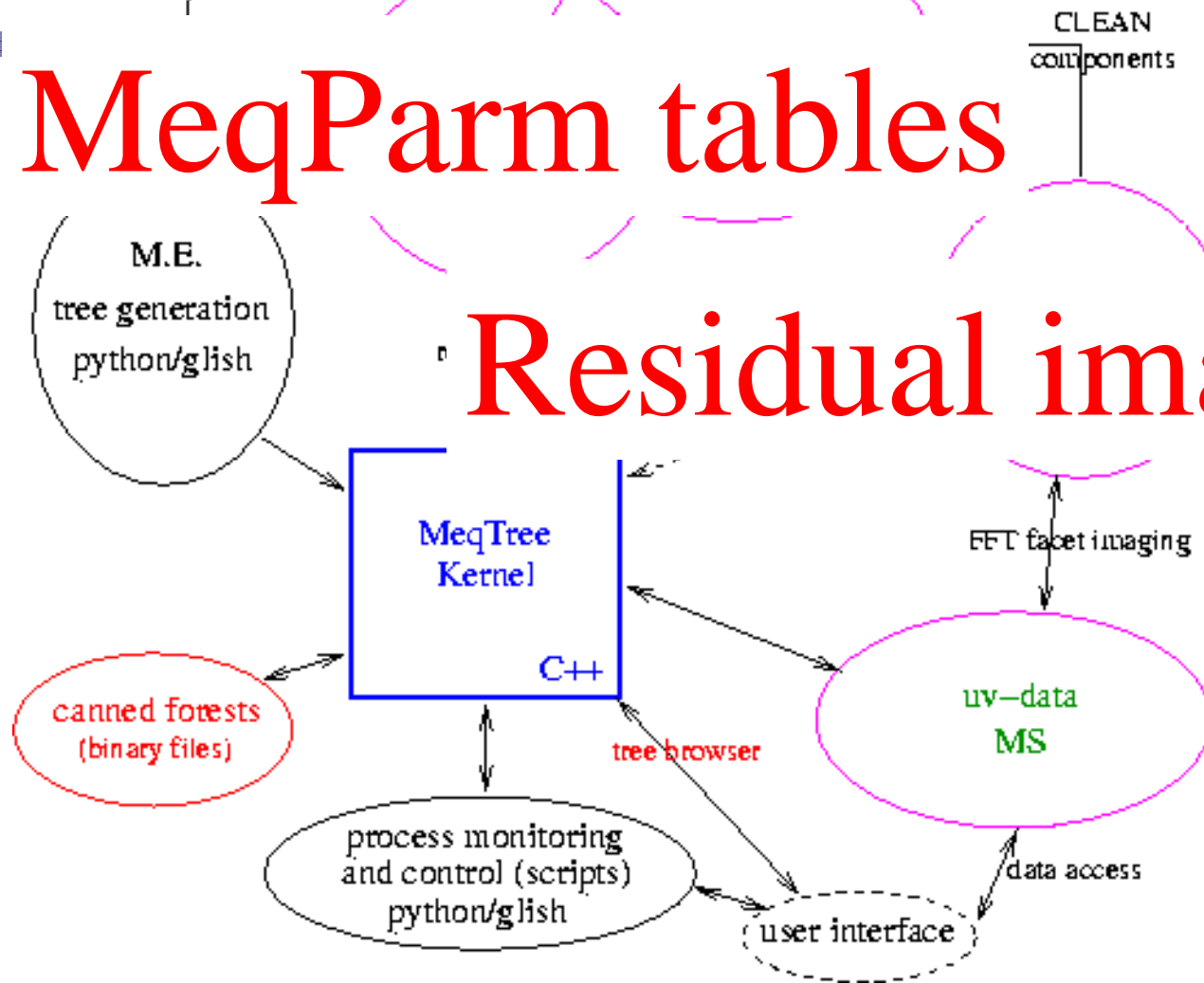
UNSC (Oct 2005)

# Global Sky Model

## MeqParm tables

## Residual images

LOFAR deliverables



UNSW (Oct 2005)

Tom Oosterloo

Michiel Brentjens

Sarod Yatawatta

Maaijke Mevius

Ronald Nijboer

Jan Noordam

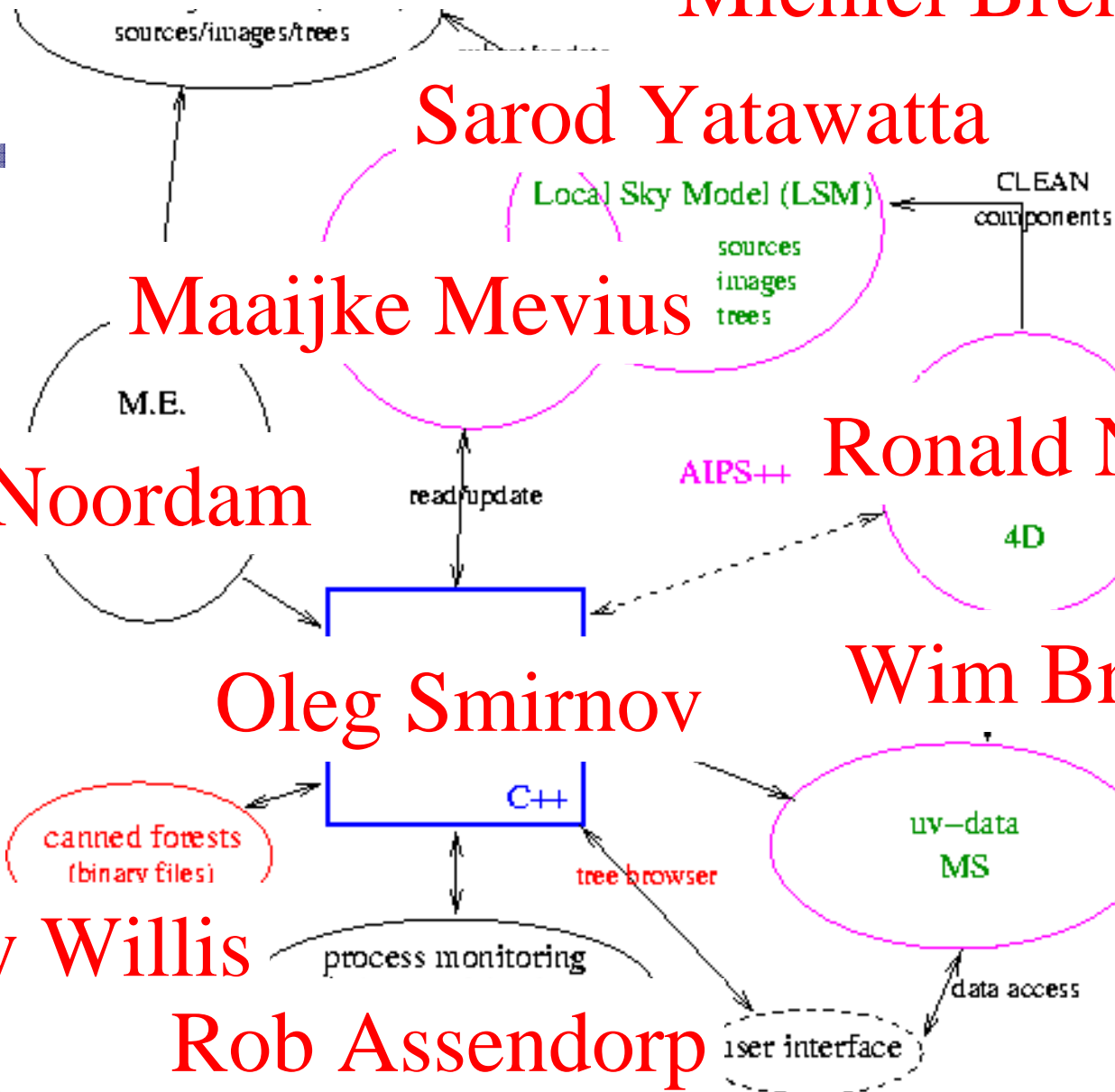
Oleg Smirnov

Wim Brouw

Tony Willis

Rob Assendorp

The MeqTree Team

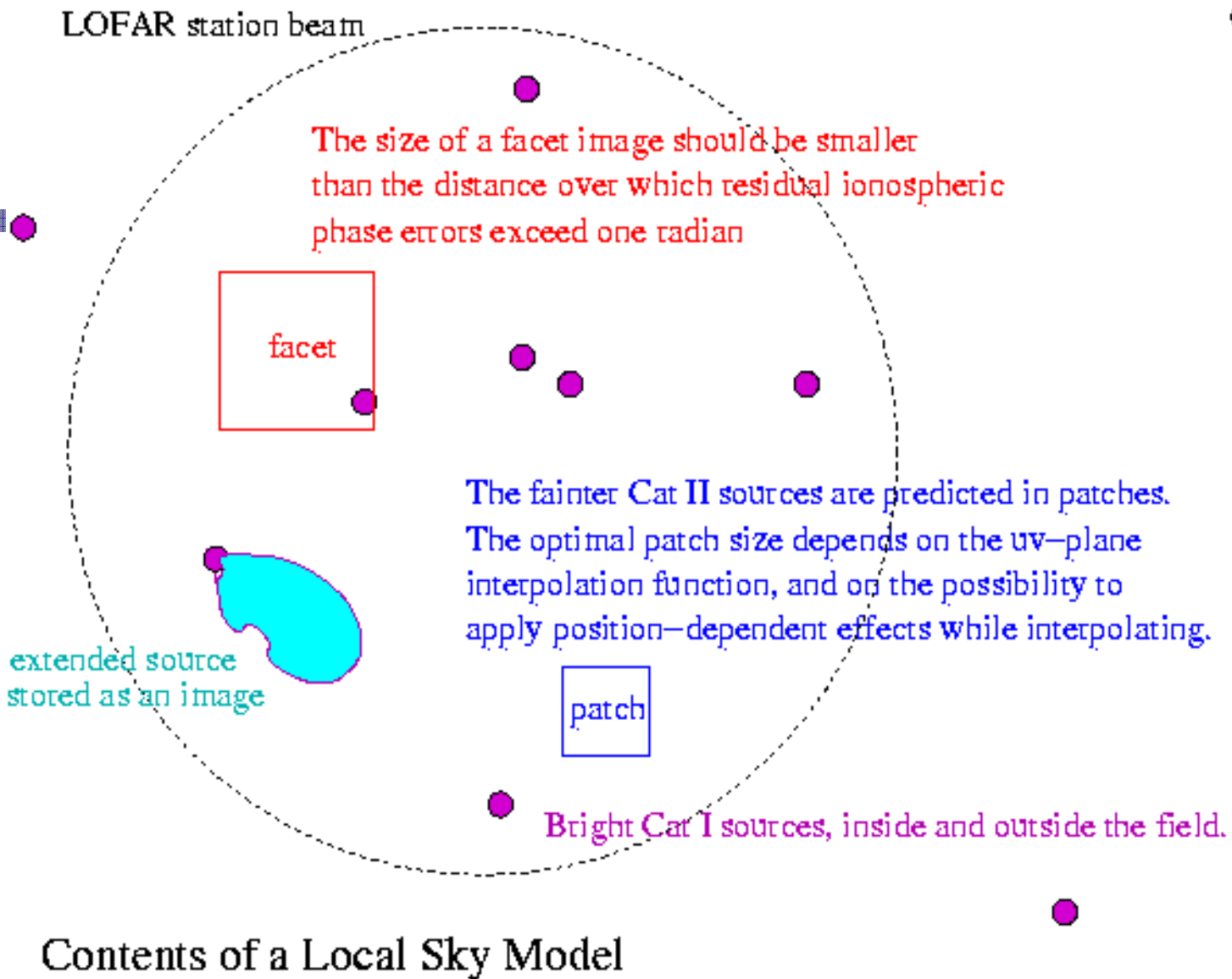


# The MeqTree Kernel

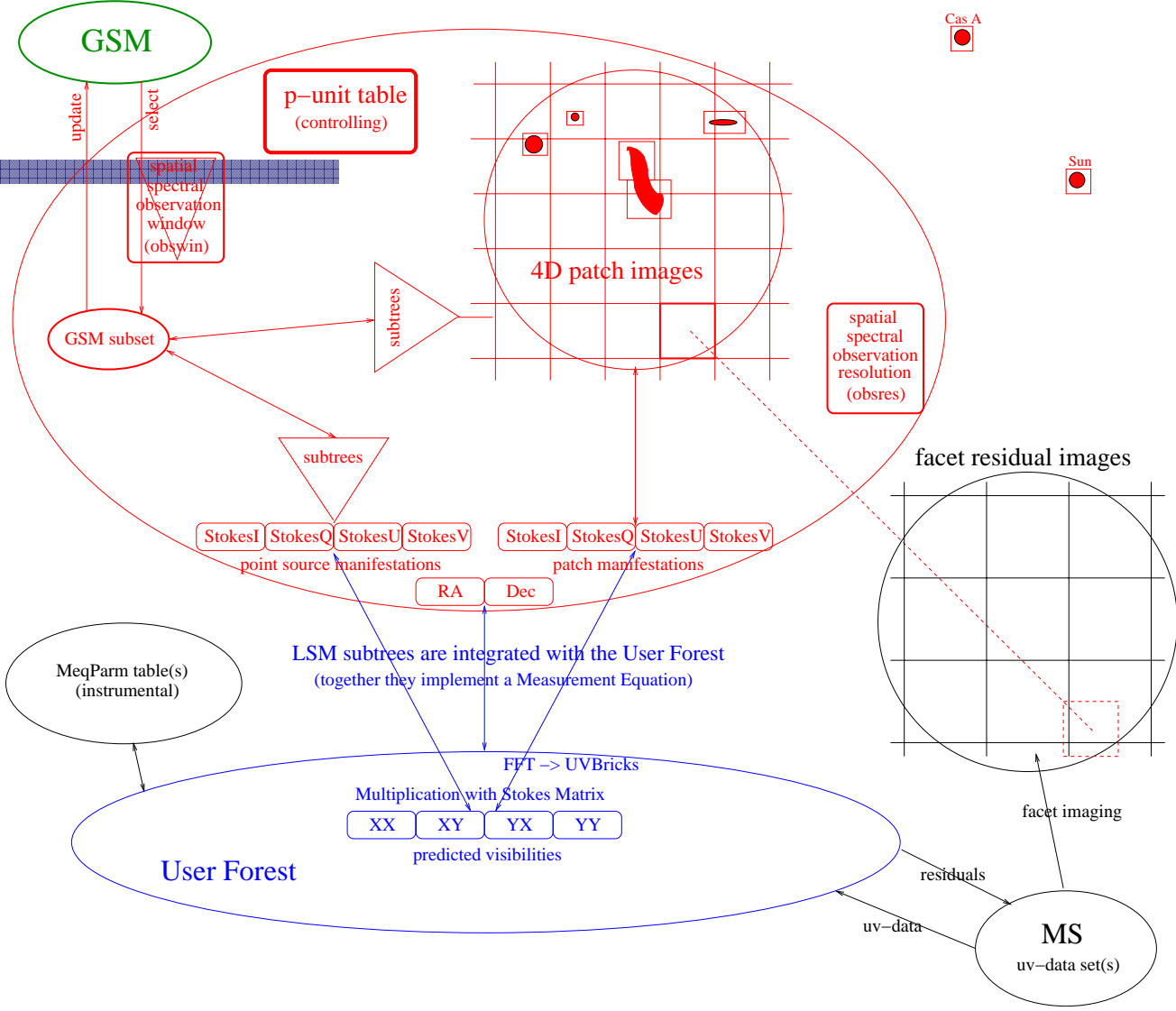
- ◆ Implements arbitrary M.E.
- ◆ Solves for subsets of M.E. parameters
- ◆ Policy Free (!)
  - ⊕ Will be applied to other areas
- ◆ Data interface (MS)
- ◆ MeqParm interface (AIPS++ tables)
- ◆ Tree Definition Language (TDL)
- ◆ User interaction via the MeqBrowser



# Local Sky Model (LSM)



# Local Sky Model (red) and user tree (blue)



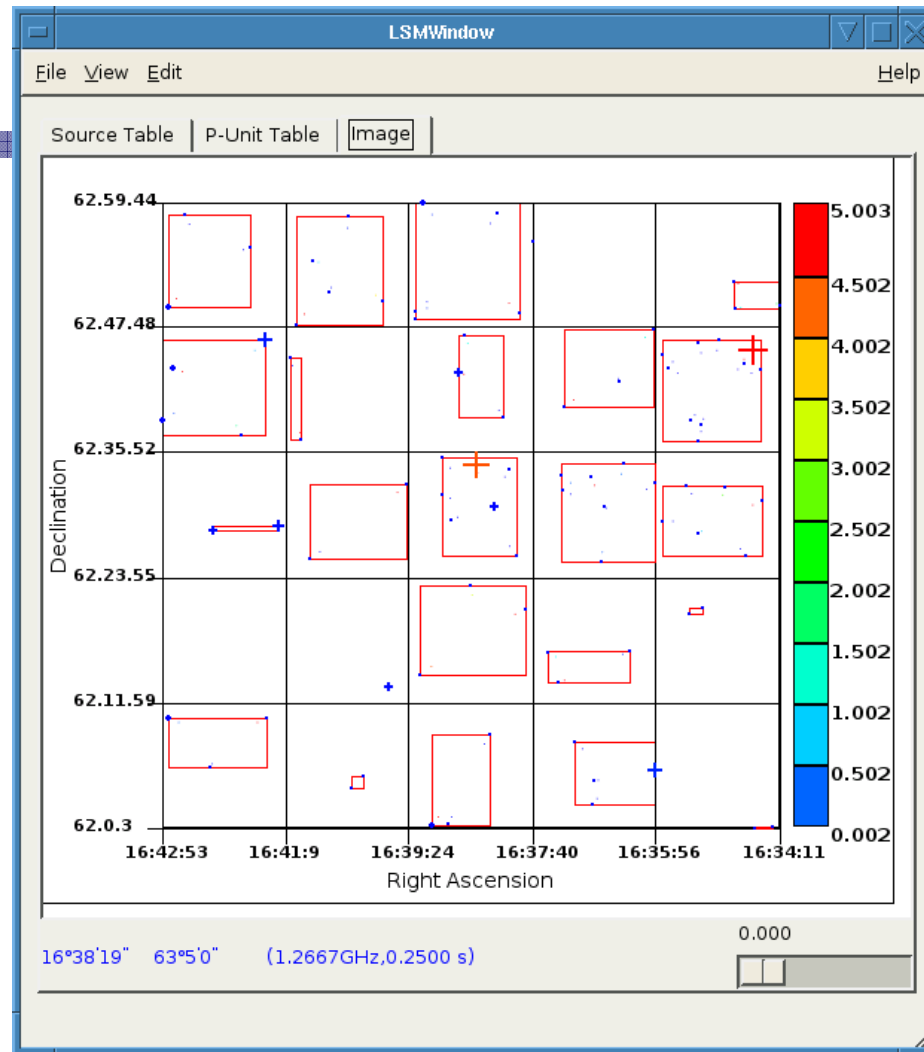
Cas A

Sun

URSI (Oct 2005)



# LSM Interface GUI



URSI (Oct 2005)



# Three categories of sources

- ◆ Cat I: Bright enough to be used for Selfcal
  - ⊕ Subtracted individually from the uv-data
- ◆ Cat II: Bright enough to be identified in a residual image, and put into the LSM
  - ⊕ Subtracted in groups (patches) from the uv-data
- ◆ Too faint to be in the LSM, so not subtracted in the uv-plane
  - ⊕ Imperfectly imaged...

# Peeling

- ◆ Deal with sources/patches sequentially
  - ⊕ Large potential saving in processing
- ◆ Selfcal: Solve for instrumental parameters in the direction of bright sources, in order of brightness
  - ⊕ Potential problem: contamination of fainter sources
- ◆ Subtraction: All LSM sources (Cat I and Cat II)
- ◆ Each time, the phase centre is shifted to the position of the peeling source or patch

# Bottleneck: Source subtraction

- ◆ All LSM sources are subtracted from the uv-data
- ◆ Requires uv-model derived from LSM
- ◆ But: We are not interested in the details (flux, position) of individual foreground sources
- ◆ Are there ways to ‘filter’ them out?
- ◆ The PAST team may have the beginning of such an algorithm...

# Residual (facet) imaging

facet



- ◆ Split the field-of-view up in facets
- ◆ Correct residual uv-data for facet centre
  - ⊕ Errors increase towards the edge
- ◆ Integrate (f,t) to decrease field-of-view
- ◆ FFT: Make a residual image of each facet
- ◆ Analyse residual images:
  - ⊕ Find/update Cat II GSM sources
  - ⊕ Deconvolve Cat III sources (if possible)
- ◆ Combine with GSM: LOFAR deliverables

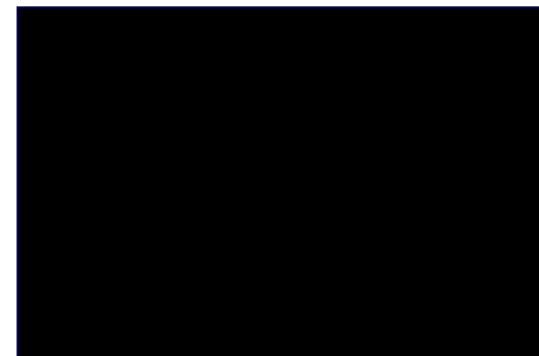
# Source extraction from residual images

- ◆ Need a PSF
  - ⊕ Strictly valid for centre of field only
  - ⊕ But may degrade benignly towards the edge
- ◆ First check positions where Cat II sources have been subtracted: update LSM source parameters
- ◆ Then look for new (Cat III) sources for LSM
- ◆ Iterate until diminishing returns
- ◆ Ultimately, use the LSM to update the GSM



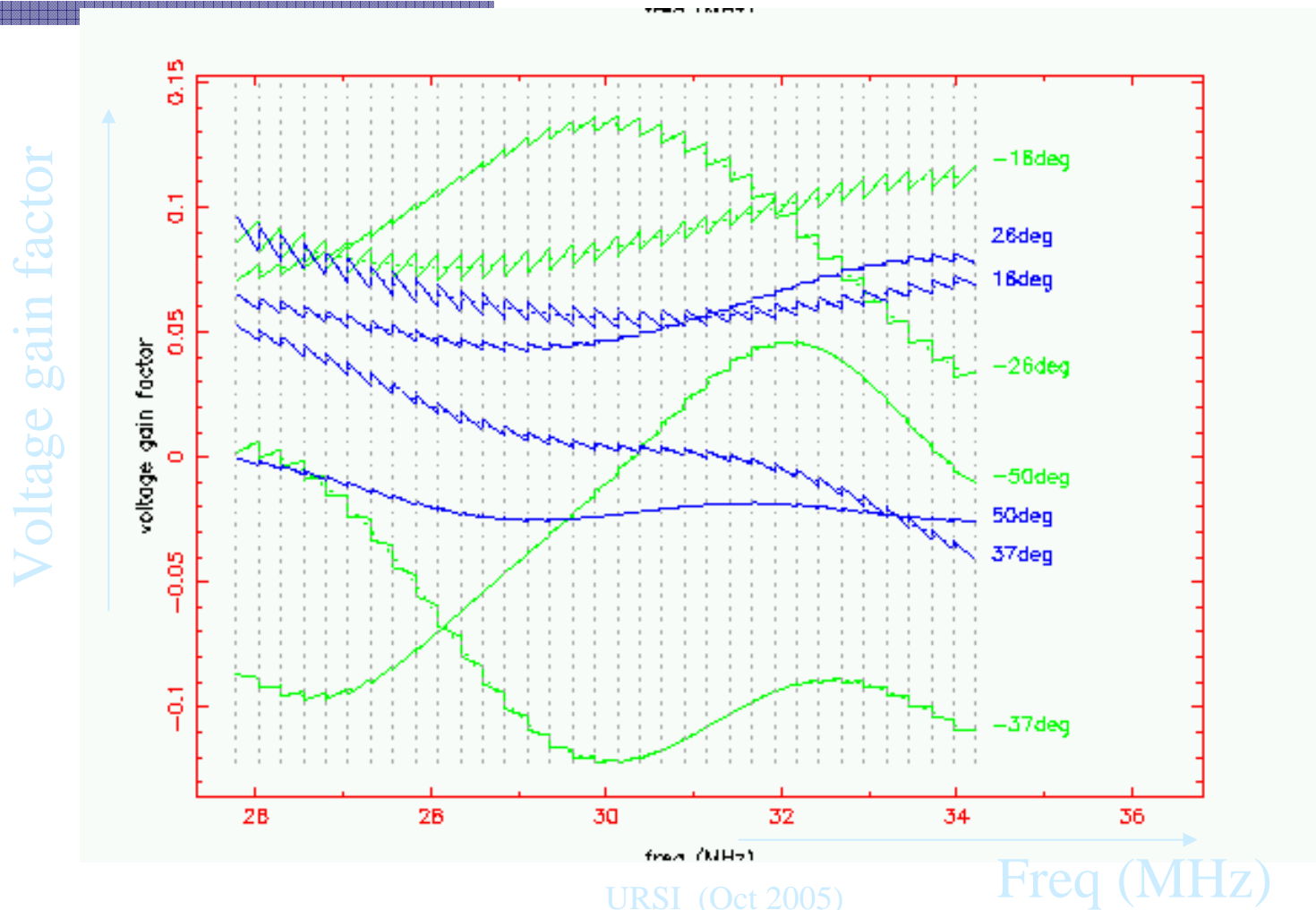


# Some imaging artefacts



- ◆ Remnants of Cat I subtraction (peeling)
- ◆ Remnants of Radio Frequency Interference
- ◆ BSR effect (far from bright sources...?)
- ◆ Pattern of facet edges (cat III imaging)
- ◆ Sidelobe confusion (PSF sidelobe level)
- ◆ Pattern of *p-unit* edges (cat II subtraction)
- ◆ Etc, etc

# Bandpass Sawtooth Ripple (BSR effect)



URSI (Oct 2005)

Freq (MHz)



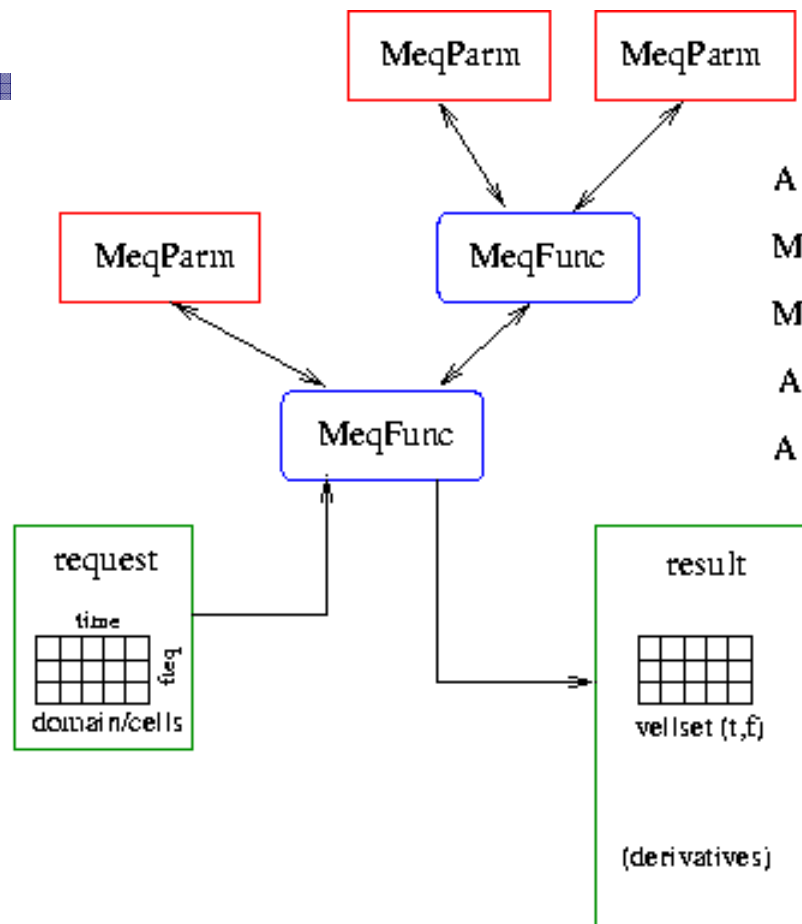
# Global Sky Model (GSM)

- ◆ Very large number of LOFAR sources
- ◆ Inhomogeneous parametrisation
  - ⊕ Includes shapelets, pixons and actual images
  - ⊕ Use MeqTrees in GSM/LSM
- ◆ Freq/time dependencies
  - ⊕ Use funklets in GSM/LSM
- ◆ Should be compatible with Virtual Observatory

---

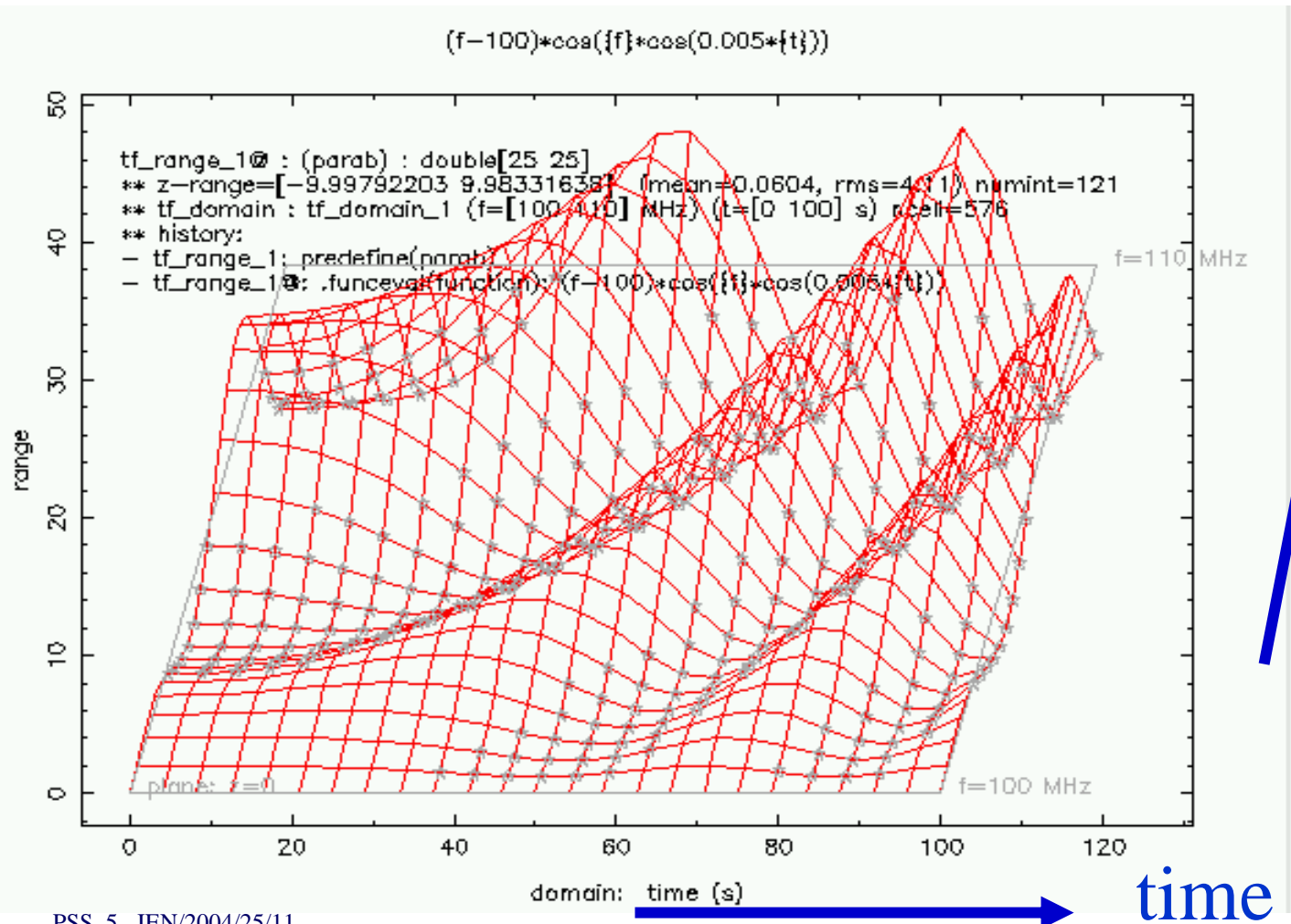
# PART III: MeqTree details

# My first MeqTree



- A MeqTree implements a (part of) a M.E.
- MeqParm nodes represent M.E. parameters
- MeqFunc nodes combine child results
- All nodes have the same interface
- A node deals with its direct family only

A node result contains vells, i.e. an array of values for the cells of a given domain (t,f):

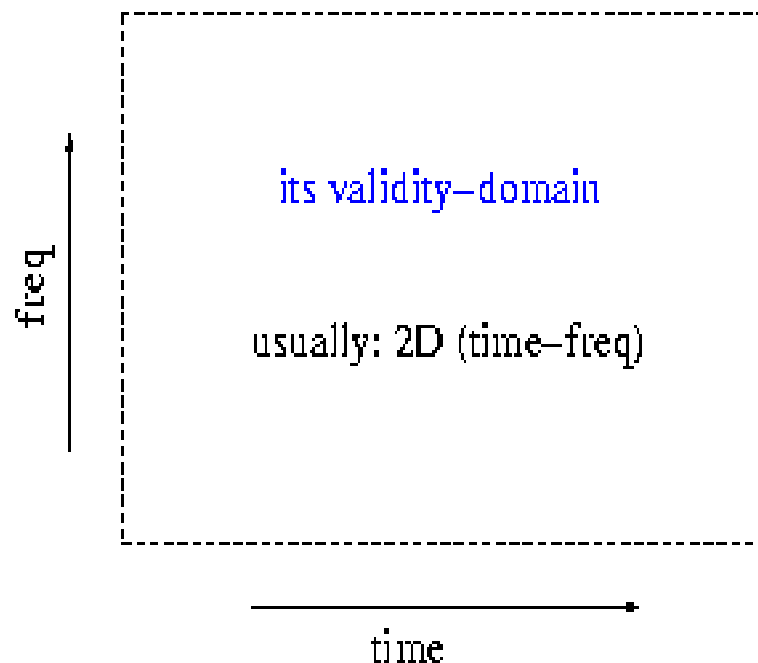


# MeqParm values are 'funklets'

c00	c10	c20	c30
c01	c11	c21	...
c02	c12	...	...

A polc: an N-dim array of  
polynomial coefficients

(or coeff of any other smooth function)

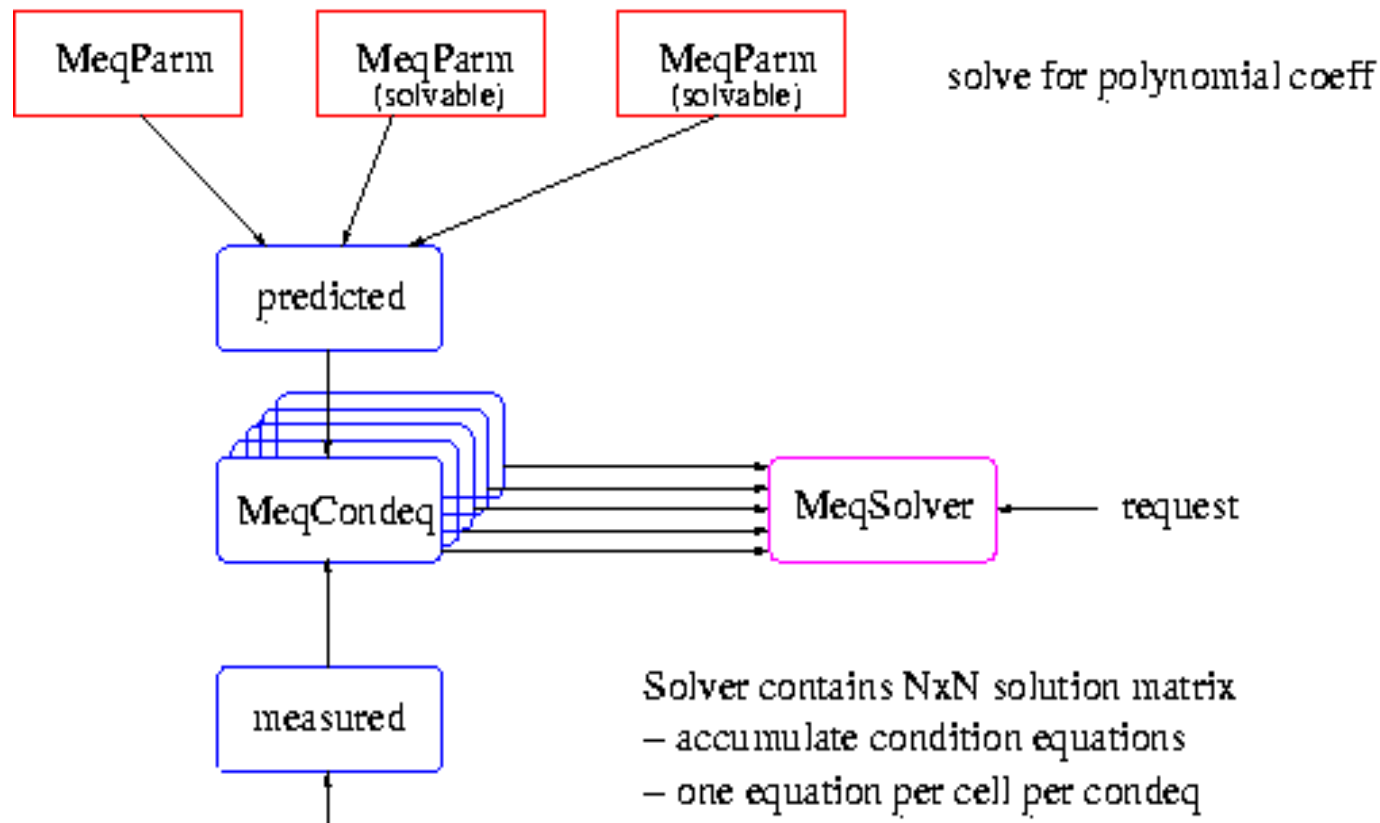


# Funklets (cont'd)

- ◆ Most M.E. parameters have 2D (f,t) polynomial funklets
- ◆ But: funklets can have any number of dimensions, with different base 'functionals' along each axis
- ◆ Examples of complex funklets:
  - ⊕ The Minimum Ionospheric Model (MIM)
  - ⊕ Station voltage beam shapes



# A solving tree



- Solver contains  $N \times N$  solution matrix
- accumulate condition equations
  - one equation per cell per condeq
  - solve: invert the matrix
  - update the **MeqParm** values (via the tree)

URSI (Oct 2005)

# MeqSolver issues

- ◆ Based on AIPS++ fitting (WNB)
- ◆ SVD (always inverts)
- ◆ Non-linear (requires iteration)
- ◆ Use solver metrics to monitor quality
- ◆ Local intelligence to minimise iteration
- ◆ Resample data to minimise cells (=equations)
- ◆ Use tiled solutions to minimise node overhead
- ◆ Optimize inversion where possible
- ◆ Alternatives?

# The many windows into MeqTrees

- ◆ Every node may be inspected individually
  - ⊕ Contents of its state record
  - ⊕ Plot of its result
- ◆ Built-in views of the results of groups of nodes
- ◆ History nodes collect intermediate results
- ◆ Debugger functionality (pause, resume, brktpnts)
- ◆ The meqbrower itself is a new kind of user interface (incl. editing of script hierarchy)



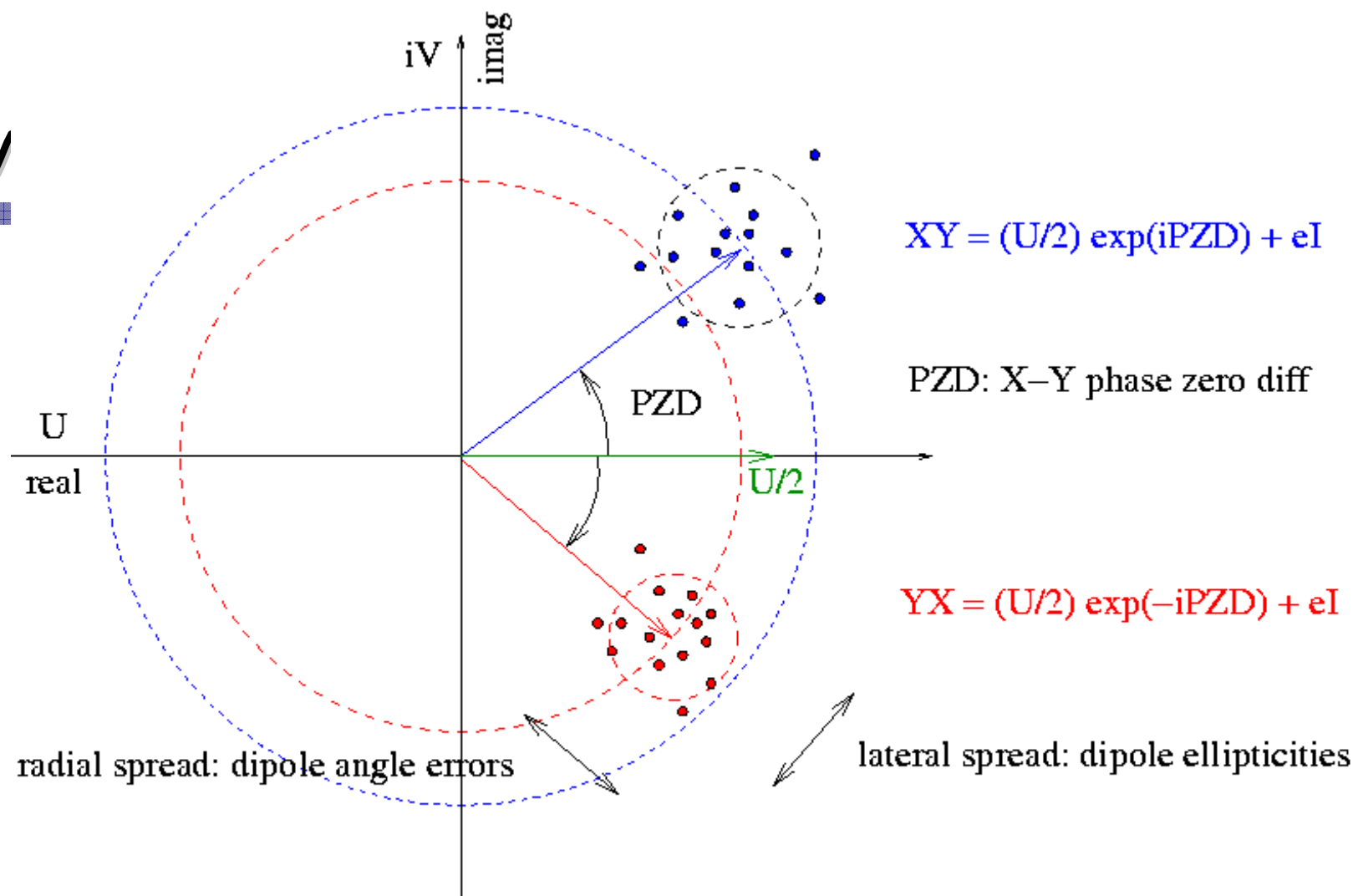
# MeqBrowser with tree, script and views

The screenshot displays the MeqBrowser application window with the following components:

- Trees:** A hierarchical tree view on the left showing the project structure, including 'Forest state', 'All nodes (1822)', 'By class (30)', and 'Root nodes (5)'. The tree is expanded to show 'MG\_JEN\_cps\_GDJones.py' and its sub-nodes.
- Tabbed Tools:** A central panel showing the script 'MG\_JEN\_cps\_GDJones.py' with 2 errors. The script contains configuration parameters for a solver, such as 'punit = 'Sitest'', 'num\_iter=10', and 'epsilon=1e-4'. A status bar at the bottom indicates 'Script executed successfully. 1822 node definitions (of which 5 are root nodes) sent to the kernel. 1 predefined function(s) available.'
- Gridded Viewers:** Four plots in the bottom right quadrant showing complex plane results:
  - dconc\_spigots\_uvp\_allcorr:** A plot with Real Axis from -0.08 to 0.08 and Imaginary Axis from -0.08 to 0.08. It shows several overlapping circles and points in red, blue, and green.
  - dconc\_spigots\_uvp\_allcorr:** A zoomed-in plot with Real Axis from -0.004 to 0.003 and Imaginary Axis from -0.001 to 0.002. It shows points marked with 'x' in various colors.
  - dconc\_corrected\_JGD\_QU\_al:** A plot with Real Axis from -0.6 to 0.6 and Imaginary Axis from -0.6 to 0.6. It shows concentric circles and points.
  - dconc\_corrected\_JGD\_QU\_:** A plot with Real Axis from -0.1 to 0.05 and Imaginary Axis from -0.06 to 0.08. It shows a single circle and points.

XY

WSRT polarization calibration



$$XY = (U/2) \exp(iPZD) + eI$$

PZD: X-Y phase zero diff

$$YX = (U/2) \exp(-iPZD) + eI$$

WSRT XY and YX data from a polarised (but:  $V=0$ ) central point source, e.g. 3c286

# MeqBrowser with parm fiddler

meqserver.0.3306.10 - idle

MeqTimba View Bookmarks Debugger Help

node /

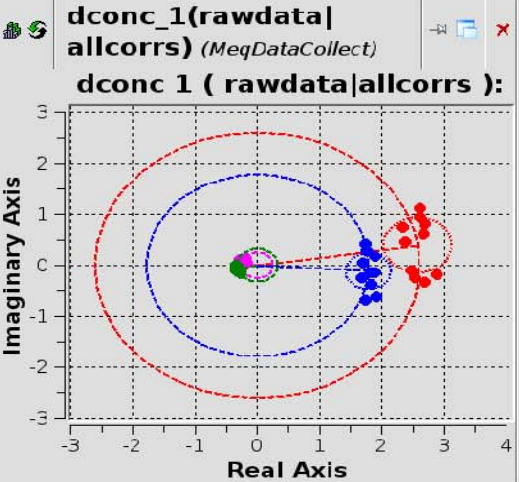
- Forest state
- All nodes (1374)
- By class (22)
- Root nodes (3)
  - \_ROOT\_rqg\_solver
  - \_ROOT\_dataCollect
  - noMS\_root
    - 09: noMS\_root[s1=4][s2=5]
    - 08: noMS\_root[s1=3][s2=5]
    - 07: noMS\_root[s1=3][s2=4]
    - 06: noMS\_root[s1=2][s2=5]
    - 05: noMS\_root[s1=2][s2=4]
    - 04: noMS\_root[s1=2][s2=3]
    - 03: noMS\_root[s1=1][s2=5]
    - 02: noMS\_root[s1=1][s2=4]
    - 01: noMS\_root[s1=1][s2=3]
    - 00: noMS\_root[s1=1][s2=2]
    - 0: sc\_25\_dconc\_result[s1=1][s2=2]
      - (dconc\_9(result|cross))
      - (dconc\_8(result|para))
      - (dconc\_7(result|allcorrs))
      - 0: visu\_3\_result[s1=1][s2=2]
        - 0: rqg\_solver\_GD
          - 3: sc\_17\_dconc\_corrected\_GD[s1=1][s2=5]
          - 2: dcoll\_6(realvsimag:(d)parms\_of\_solve)
          - 1: dcoll\_5(realvsimag:condeqs\_of\_solve)
          - 0: solver\_GD
            - 39: condeq\_4[s1=4][s2=5][c=YY]

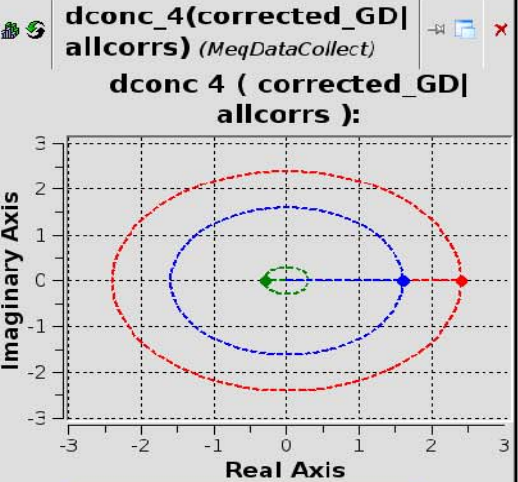
Messages Trees Snapshots Events

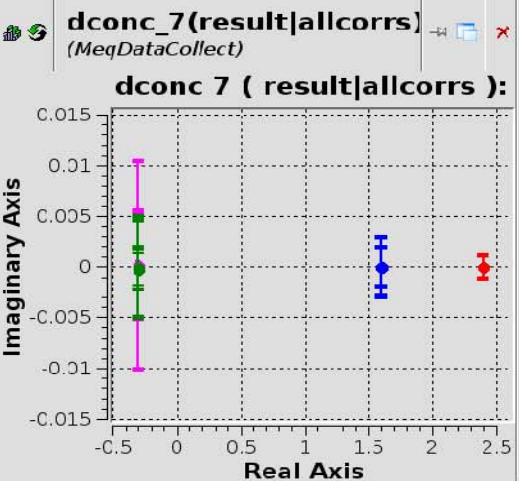
idle

VSZ:209/211M RSS:109/110M DS:188/189M

★ solver\_GD
★ uvdata\_allcorrs
★ uvdata\_cross

**dconc\_1(rawdata|allcorrs)** (MeqDataCollect)  
**dconc 1 ( rawdata|allcorrs ):**


**dconc\_4(corrected\_GD|allcorrs)** (MeqDataCollect)  
**dconc 4 ( corrected\_GD|allcorrs ):**


**dconc\_7(result|allcorrs)** (MeqDataCollect)  
**dconc 7 ( result|allcorrs ):**


★ Fiddle from noMS\_root

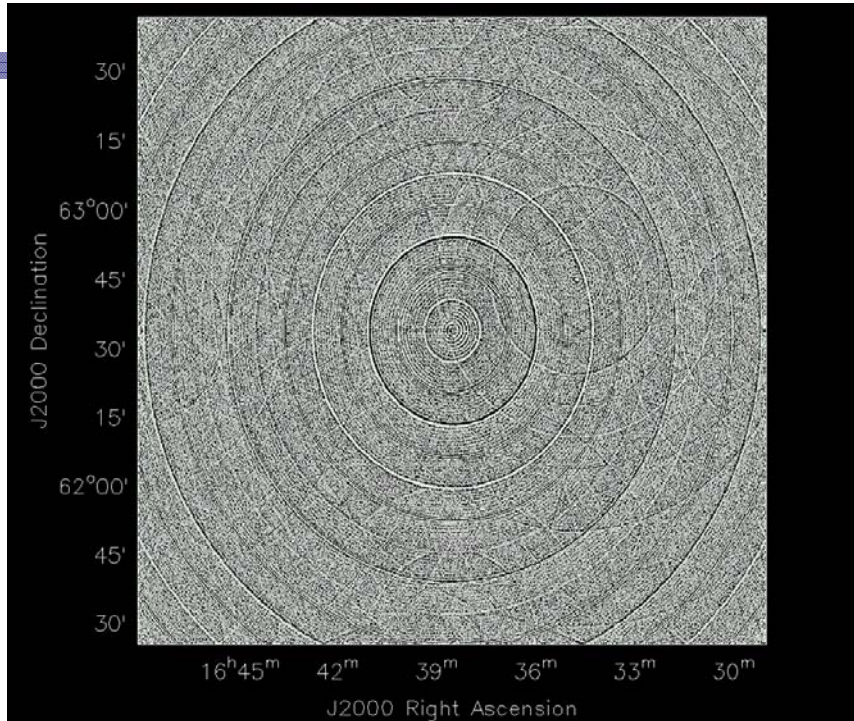
name	no parm selected
53	gain[s=2]   p=
54	gain[s=3]   p=
55	gain[s=3]   p=
56	gain[s=4]   p=
57	gain[s=4]   p=
58	gain[s=5]   p=
59	gain[s=5]   p=
60	phase[s=1]   t=
61	phase[s=1]   t=
62	phase[s=2]   t=
63	phase[s=2]   t=
64	phase[s=3]   t=
65	phase[s=3]   t=
66	phase[s=4]   t=
67	phase[s=4]   t=
68	phase[s=5]   t=
69	phase[s=5]   t=
70	uvp PZD

stepsize: 1.0  
 reexecute  
 Execute Zero

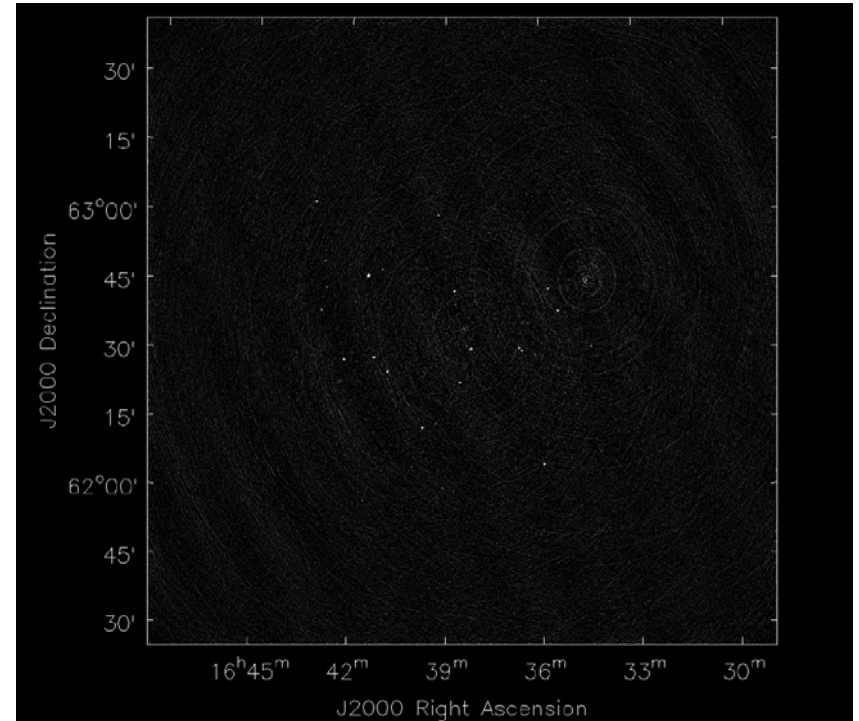
# Results so far

- ◆ Stable kernel and browser
- ◆ 3c343 (Michiel Brentjens)
  - ⊕ Also: Miriad, NEWSTAR, AIPS++
- ◆ M81/82 (Tom Oosterloo): Peeling in Miriad
- ◆ Minimum Ionospheric Model (MIM)
  - ⊕ URSI poster by Jim Anderson
- ◆ Model of CLAR beam (Tony Willis)
- ◆ Model of LOFAR WHAT station beam

# MeqTree: 3C343.1 & 3C343



Uncalibrated Image



Calibrated Image  
2 sources subtracted

Result by Michiel Brüggeles

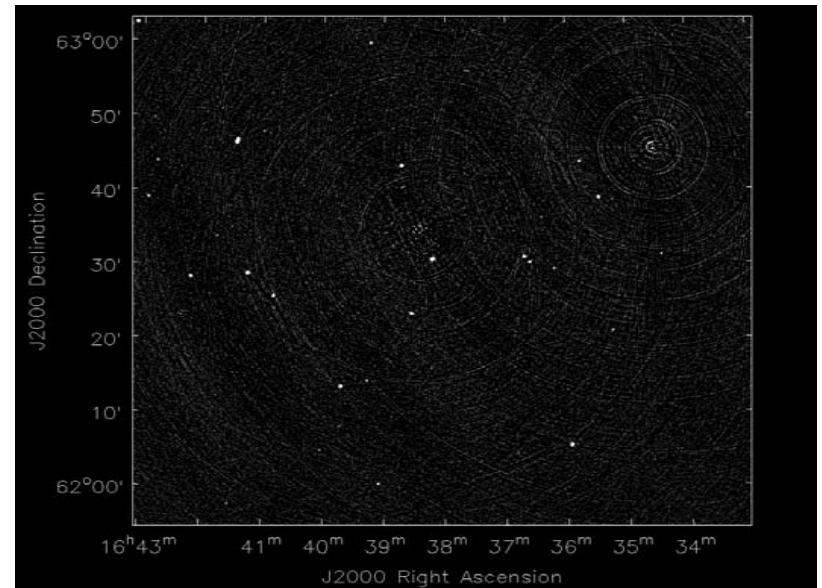
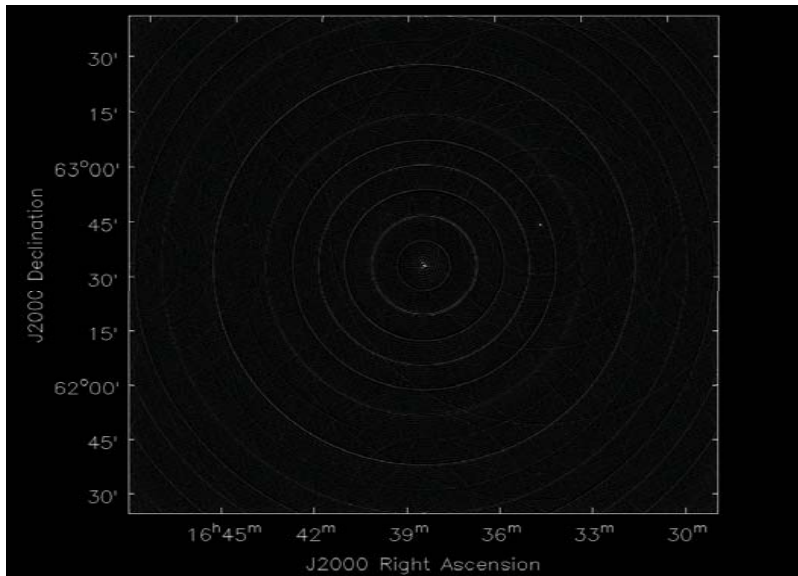


URSI (Oct 2005)



The MeqTree System is a reduction module that implements  
**Generalized Self-Cal and automated Peeling**

# MeqTree: 3C343.1 & 3C343



Result by Michiel Brentjes

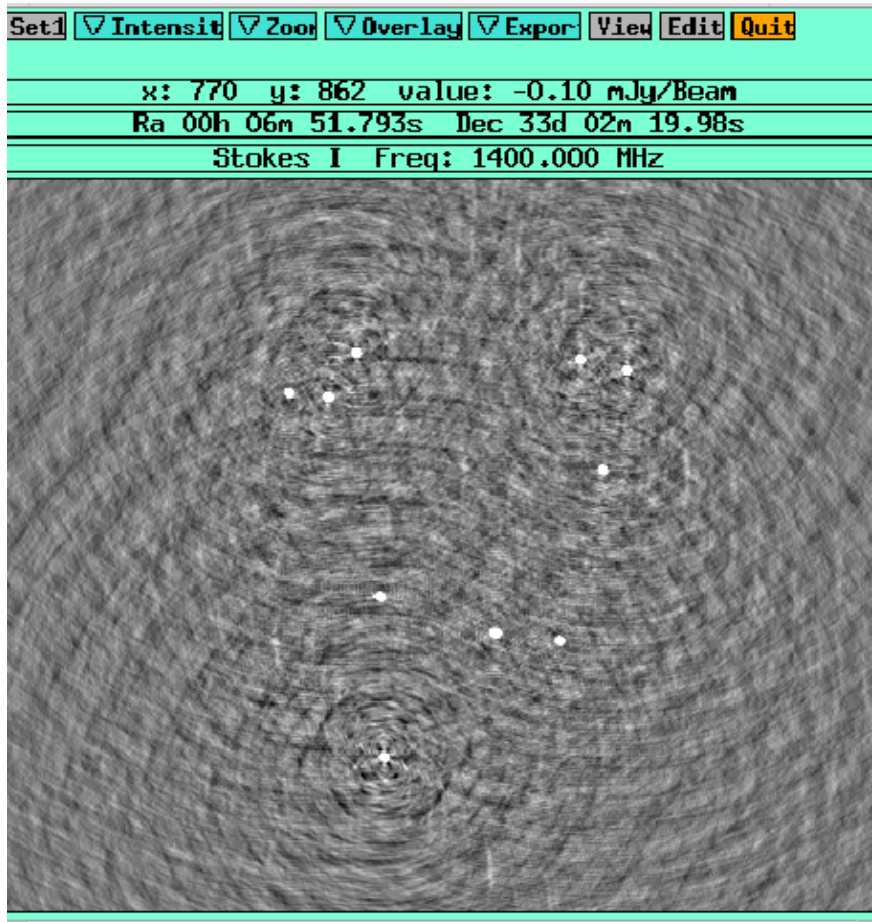
LOFAR Self-Cal will be based on experience with the MeqTree System

URSI (Oct 2005)

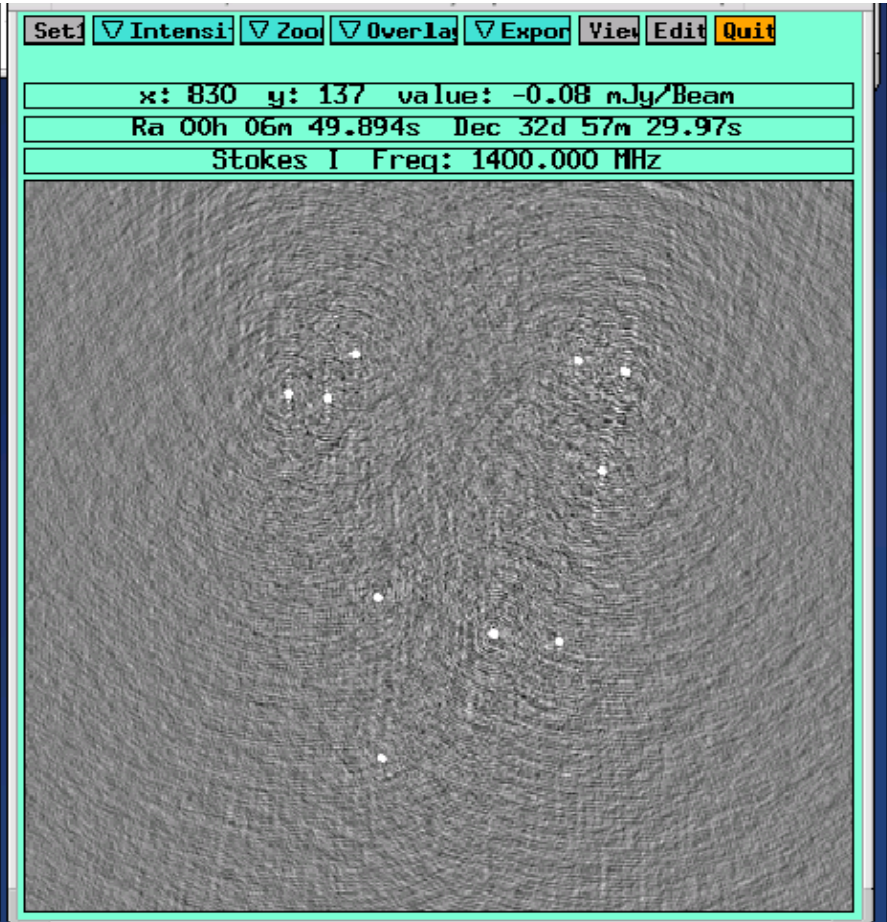


# First external user (Tony Willis, Penticton, Canada): CLAR simulations

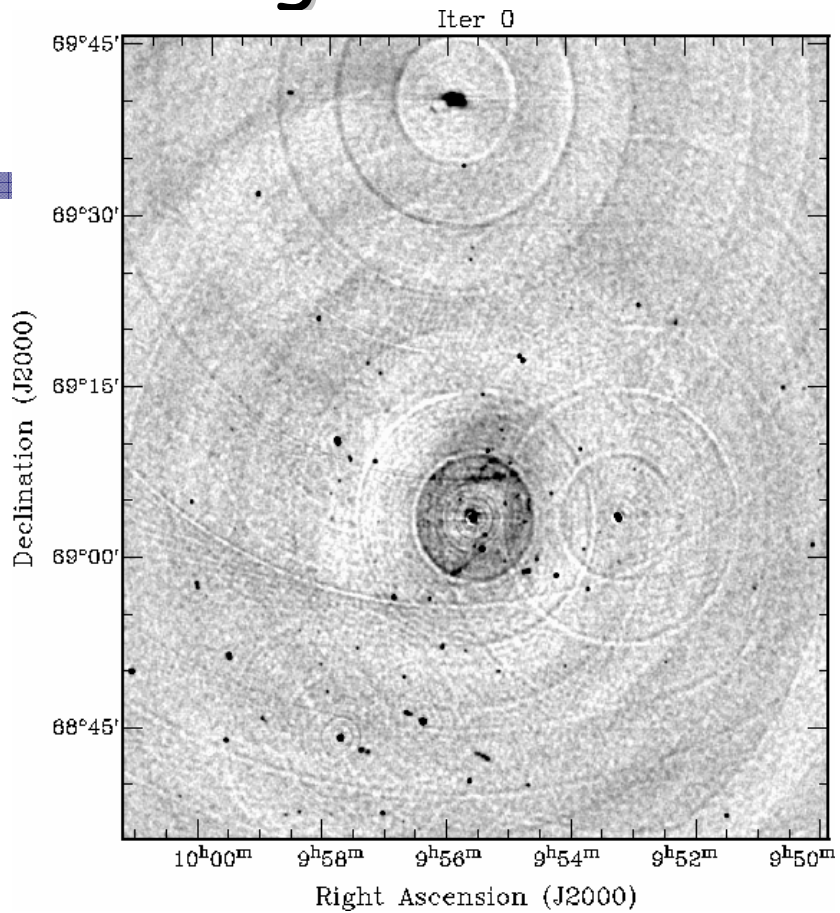
AIPS++ (20 min chunks, image-plane)



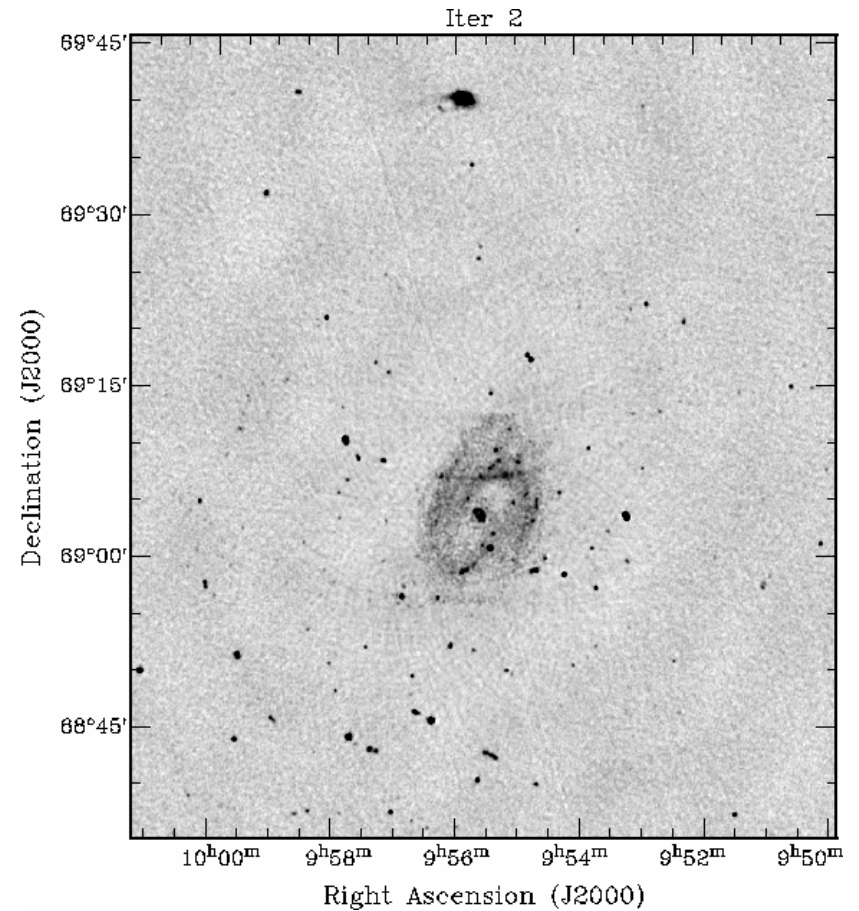
MeqTree (continuous, station-specific)



# Peeling on M81 & M82



standard selfcal entire field



2-patch peeling (miriad)

(Result by Tom Oosterloo)



URSI (Oct 2005)

# Conclusions

---

- ◆ LOFAR calibration will not be easy
- ◆ We have a lot of new ideas
- ◆ But more are needed
- ◆ The stage is set for M.E. development