IONOSFERIC REFRACTION AND FARADAY ROTATION

ASTRON-JIVE SUMMER SCHOOL 2010

Supervisors:
GER DE BRUYN
RONALD NIJBOER

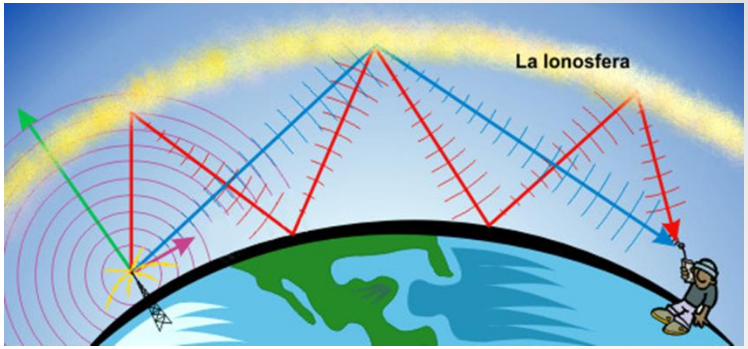
Student: **MONICA TRASATTI**

OUTLINE:

- Introduction
- Goals
- Data used
- Lofar imaging pipeline
- Results
- Conclusions

IONOSPHERE

It's a dynamic inhomogeneous medium, with electron density varying as a function of position and time.



The spatial variations in electron density cause local refraction of the wave as it travels through the ionosphere, causing:

- **Propagation delay**, that integrated along the LoS results in a phase rotation given by

$$\theta_{ion} \propto \int n(e)dl$$

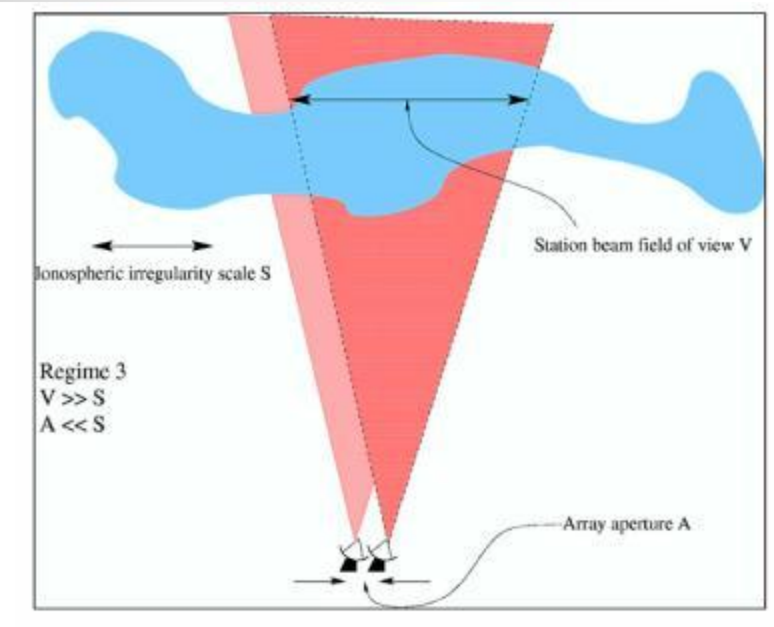
Depend on **absolute TEC** along the LoS

- **Faraday rotation** (rotation of the plane of polarization)

$$\theta_{far} = \mathrm{RM}\,\lambda^2$$

The effects are function of <u>position</u>, <u>time</u> and <u>frequency</u>.

CALIBRATION SCENARIOS

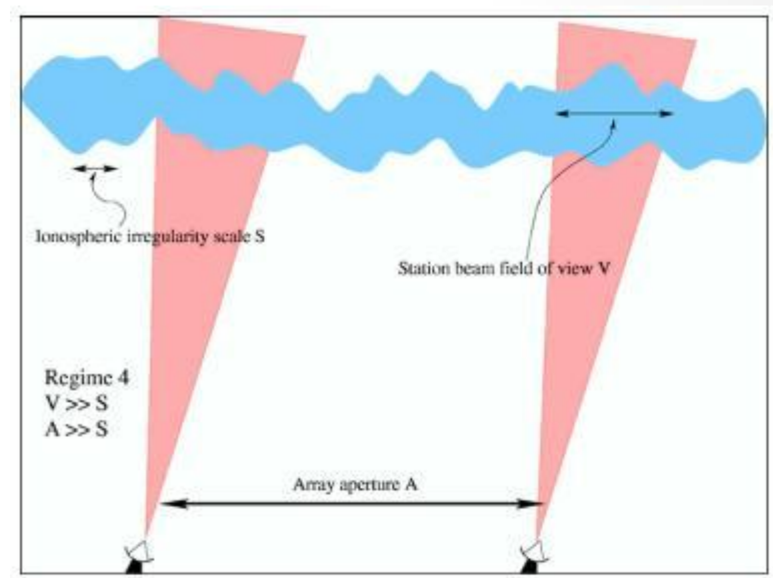


Large FOV of elements + small array: All lines of sight go through the same propagation path but there may be considerable differences in propagation conditions towards distinct sources within the FOV.

→ Same direction dependent complex gain for all elements.

Large FOV of elements + large array: The LoS towards each source may experience propagation conditions that differ for different elements in the array.

→ Distinct complex gain corrections for each source and each receiving element.



Lonsdale 2004

Selfcalibration/imaging depends on **RELATIVE TEC** which varies rapidly.

GOALS:

- Determine the properties of the large scale ionosferic TEC distribution by looking in two different directions

- Measure the ionospheric RM using an intrinsically polarized source in the field (PSR J0218+423)

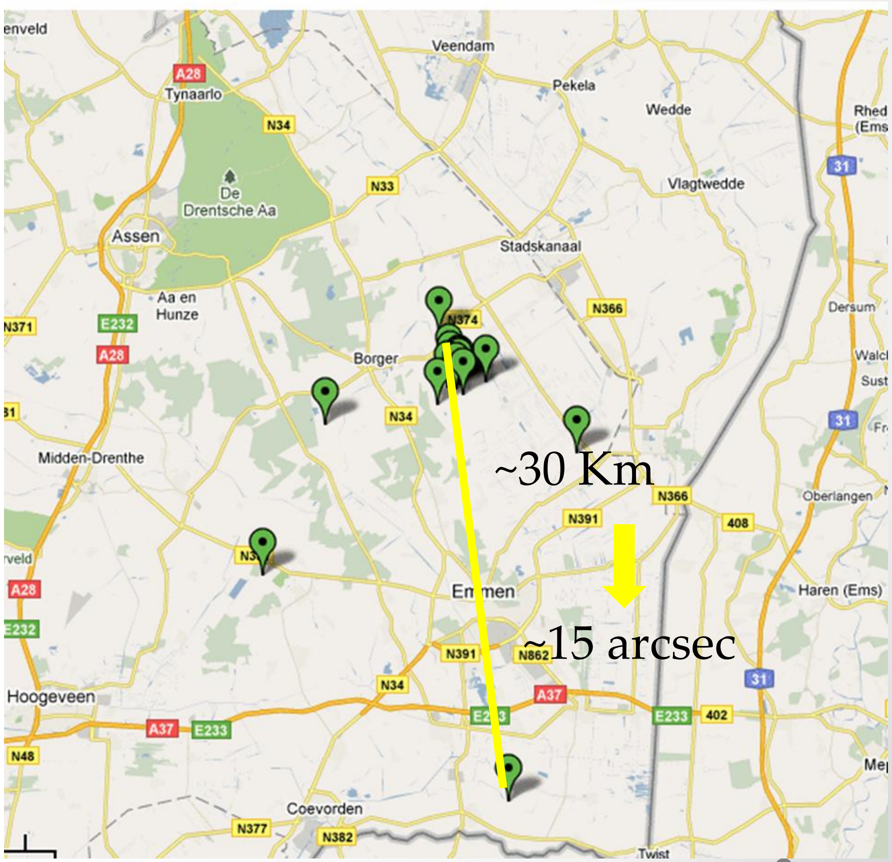
LOFAR DATA

L2010_07096:

- Phase center : +02:22:39.6 +43:02:07.7 (3C66 FIELD)
- Duration (hrs) : 6.00
- Observation time span : 2010-04-24 T08:40:00 – 2010-04-24 T14:39:58
- Integration time : 3 sec
- Numbers of antenna : 27 (HBA)
- Frequency range : 126 – 174 MHz

247 subbands → SB10.MS
SB11.MS
..
..
..
SB17.MS

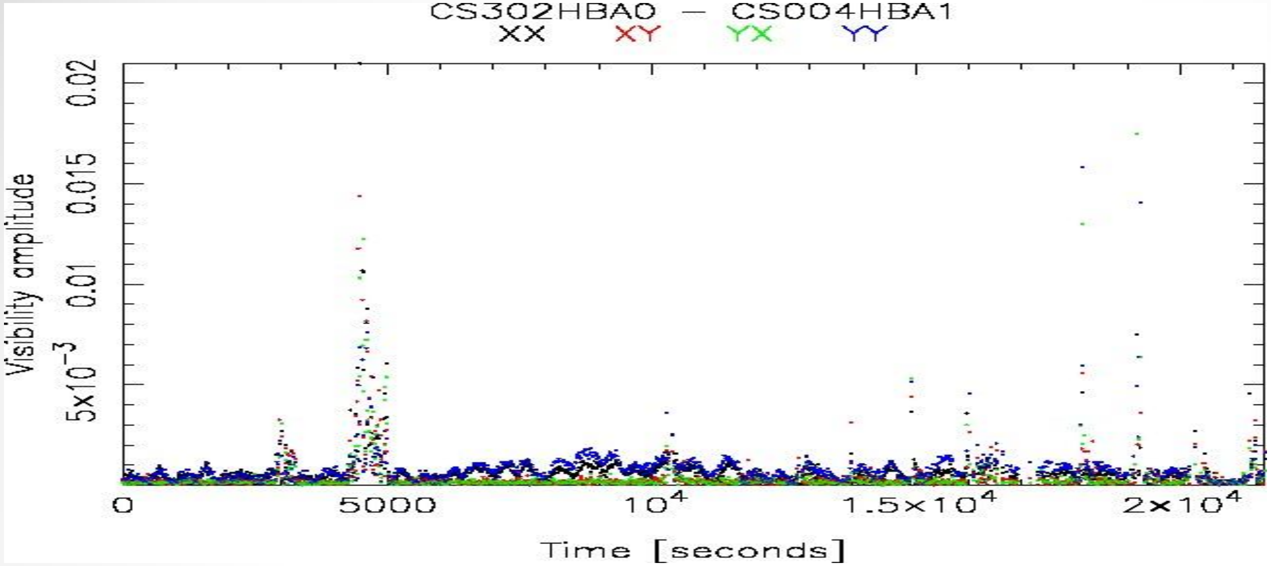
Frequency range: 127.8 – 129.38 MHz



LOFAR imaging pipeline:

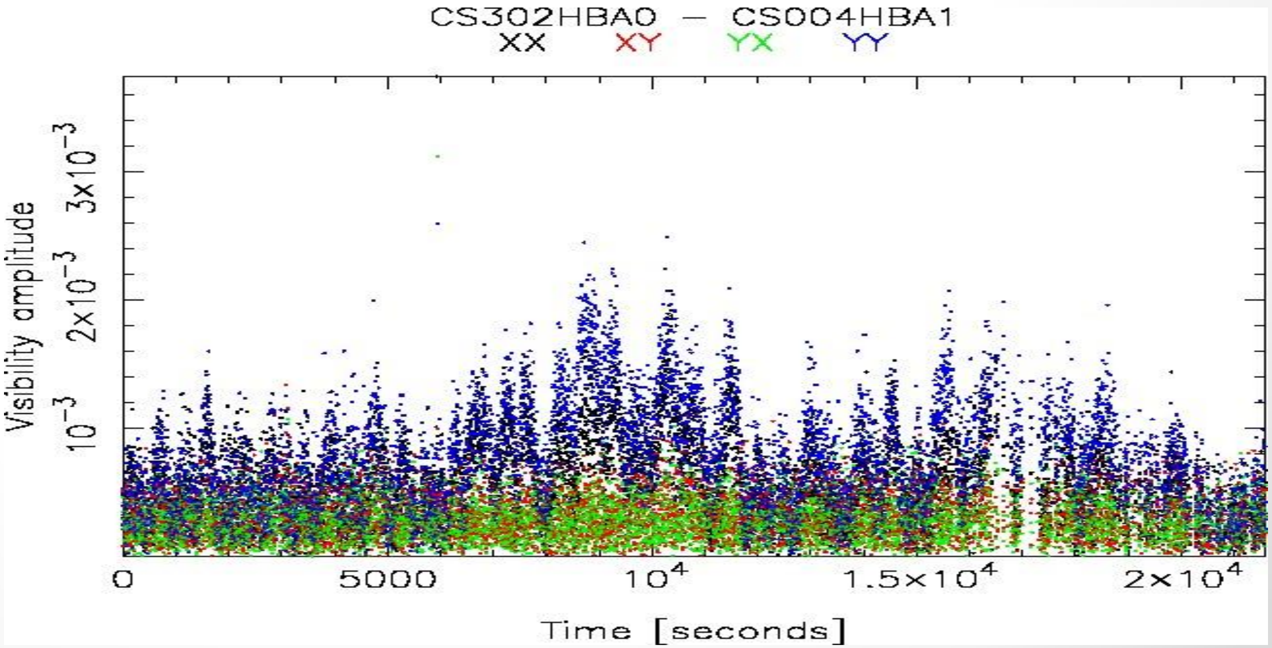
- Initial flag and compression
- BlackBoard Selfcal
- Imaging with casa
- Clean components as new model

FLAGGING AND COMPRESSION



NDPPP

AO flagger (André Offringa)



BlackBoard Selfcal (BBS)

BBS requires:

1. A file that describes the layout of the cluster nodes (eg. Sub4.clusterdesc)
2. A .vds/.gds file that describes the contents and location of the MSs
3. A sky model that defines what BBS will attempt to replicate during calibration
4. A parset file that defines the series of operations that BBS has to perform

It is possible to run BBS

- on a single SB
- distributed way on several SBs →

Strategy.UseSolver = T
….
Step.solve.Solve.CalibrationGroups = [n]

IMAGING : CASA SOFTWARE

Task : CLEAN

Widefield mode with 64 wprojection planes

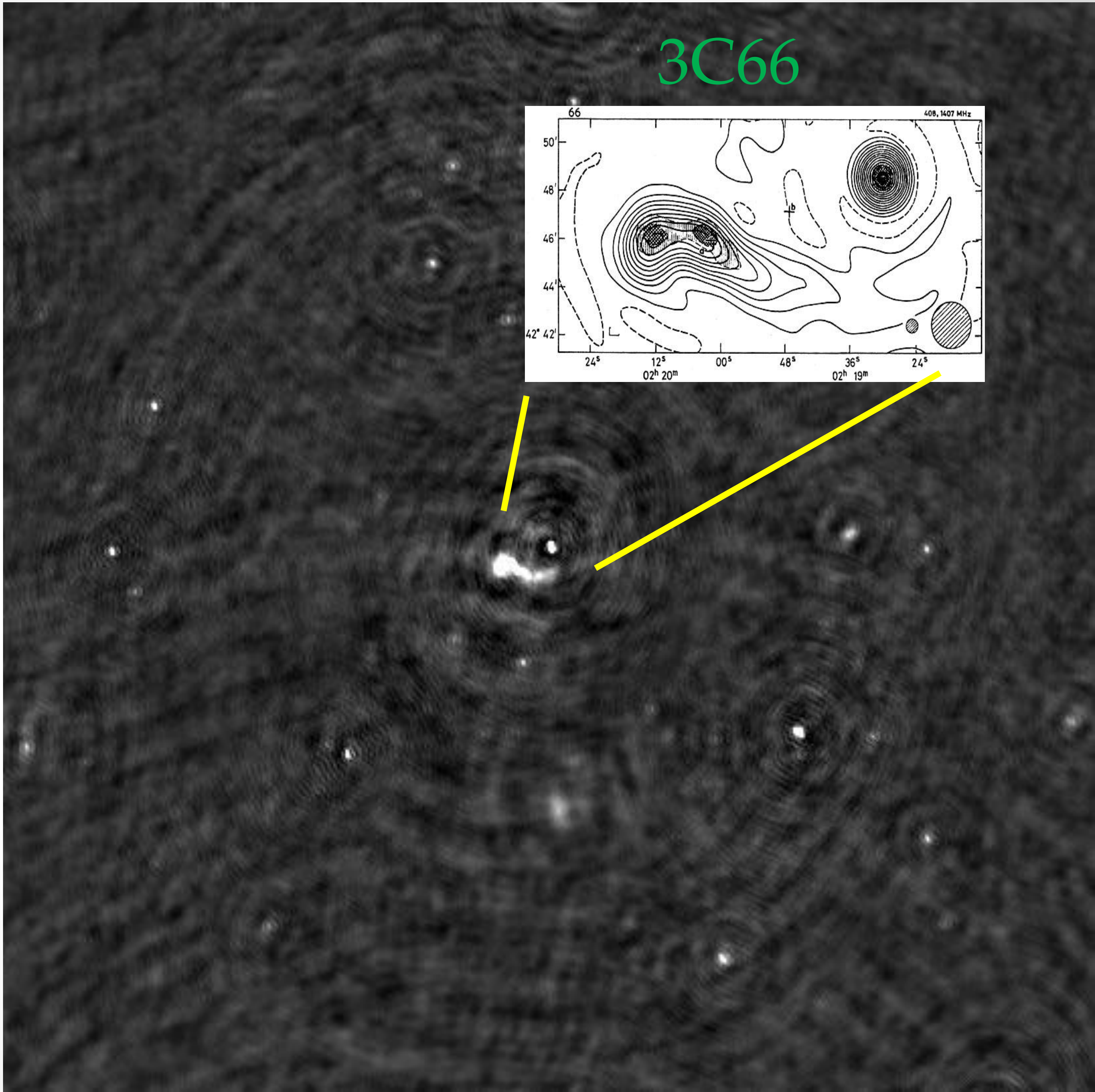
Briggs weighting with robust parameter -2

```
portal : trassati <2>
File  Edit  View  Scrollback  Bookmarks  Settings  Help
CASA <65>: niter=50000

CASA <66>: imagename='sb10_3C66CLEAN'

CASA <67>: inp
---------> inp()
# clean :: Invert and deconvolve images with selected algorithm
vis                 = 'SB10_A0_1_5bis.MS' #  Name of input visibility file
imagename           = 'sb10_3C66CLEAN'  #  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                =      'mfs'        #  Spectral gridding type (mfs, channel, velocity, frequency)
     nterms         =          1        #  Number of terms used to model the sky frequency dependence
                                        #   (Note: nterms>1 is under development)
     reffreq        =         ''        #  Reference frequency for MFS (relevant only if nterms > 1)

gridmode            = 'widefield'       #  Gridding kernel for FFT-based transforms, default='' None
     wprojplanes    =         64        #  Number of w-projection planes for convolution
     facets         =          1        #  Number of facets along each axis (main image only)

niter               =      50000        #  Maximum number of iterations
gain                =        0.1        #  Loop gain for cleaning
threshold           =   '0.02Jy'        #  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), and/or mask region(s)
imsize              =       3600        #  x and y image size in pixels. Single value: same for both
cell                =          8        #  x & 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, QU,IQUV)
weighting           =   'briggs'        #  Weighting of uv (natural, uniform, briggs, ...)
     robust         =         -2        #  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
pbcor               =      False        #  Output primary beam-corrected image
minpb               =        0.2        #  Minimum PB level to use
calready            =      False        #  True required for self-calibration
async               =      False        #  If true the taskname must be started using clean(...)

CASA <68>: go
---------> go()
Executing:  clean()
```

FIRST IMAGE

Peak: 3.17 Jy
noise~8.4 mJy

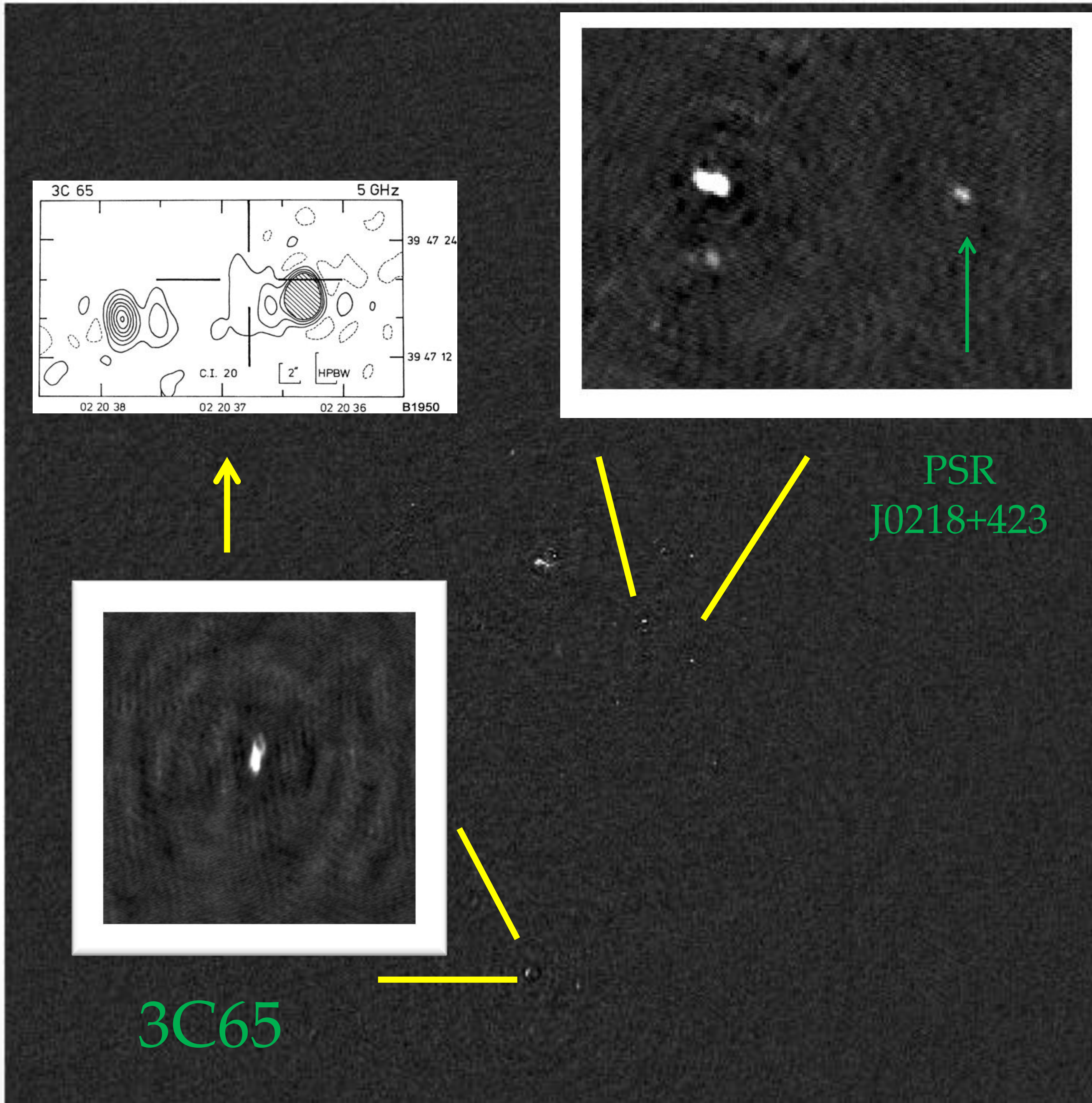
DR~ 377.4



BETTER IMAGE

Peak: 5.22 Jy
Noise ~ 5.2 mJy

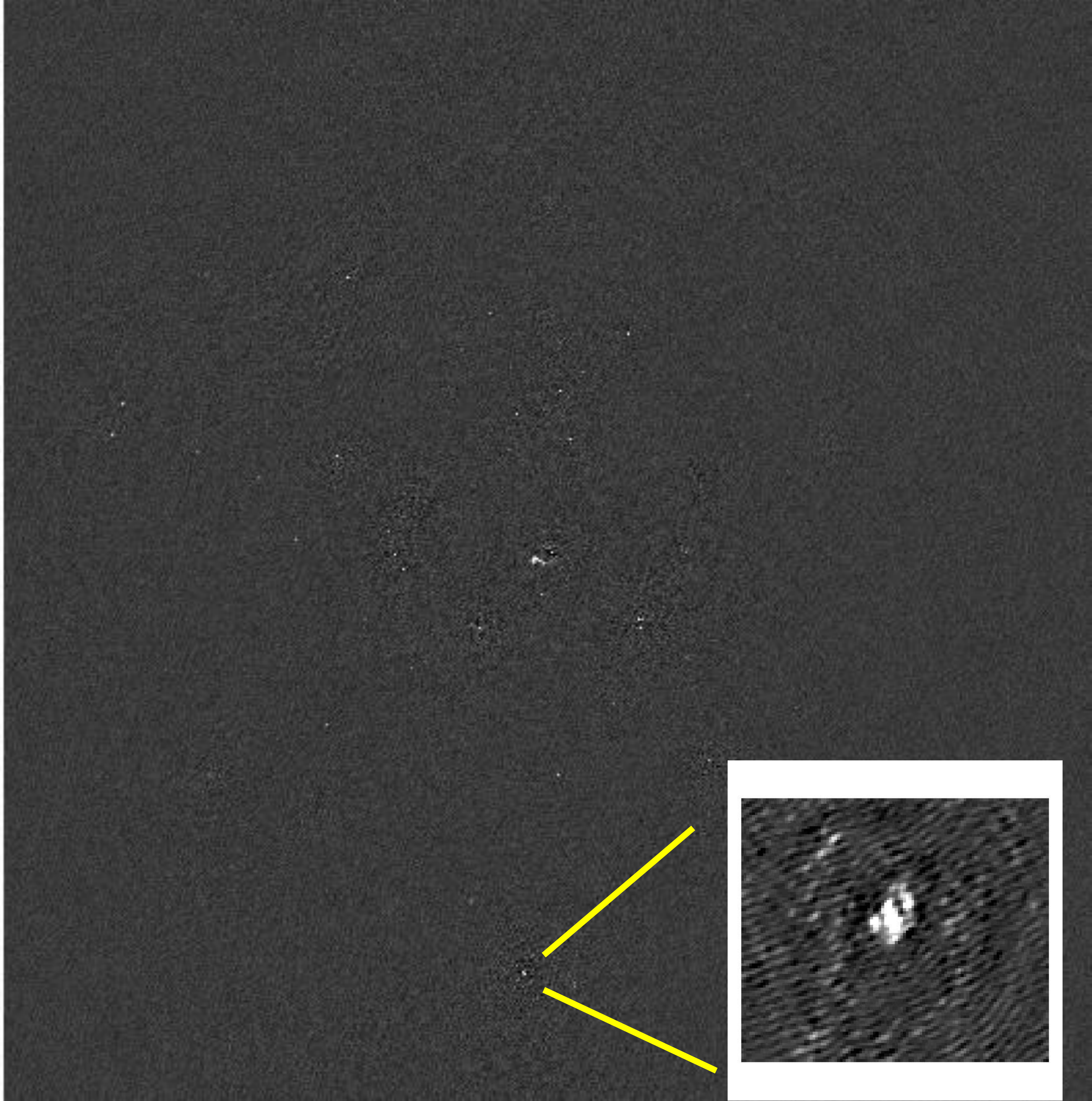
DR = 1005



LAST IMAGE

peak = 4.9 Jy
noise~ 6 mJy/Beam

DR = 816



DIRECTIONAL DEPENDENT BBS

- Changes in the catalog file:

```
# (Name, Type, PATCH, Ra, Dec, I)
# Define source patch (patch position is ignored)
 , , 3C66, 02:22:25.5, +43.00.48
 , , 3C65, 02:23:43.6, +40.01.16

#Define CLEAN components for the patch
Pixel…., POINT, 3C66, …, …, …
…
Pixel…., POINT, 3C65, … , …, …
```

- Changes in the parset file:

```
Strategy.Steps = [solve, correct]

Step.solve.Model.DirectionalGain.Enable = T
…
Step.correct.Model.Sources = [3C66] (or [3C65])
Step.correct.Model.DirectionalGain.Enable = T
```

DIRECTIONAL DEPENDENT BBS

NEXT STEP:

New step in the BBS strategy:

Strategy.Steps = [solve, subtract, correct]
…..
Step.subtract.Operation = SUBTRACT
Step.subtract.Model.Sources = […]
Step.subtract.Model.DirectionalGain.Enable = T

CONCLUSIONS

- Software still needs improvements
- We can go on testing the directional dependent BBS
- 3C66 field is a good field to study the ionospheric RM

Thank you!