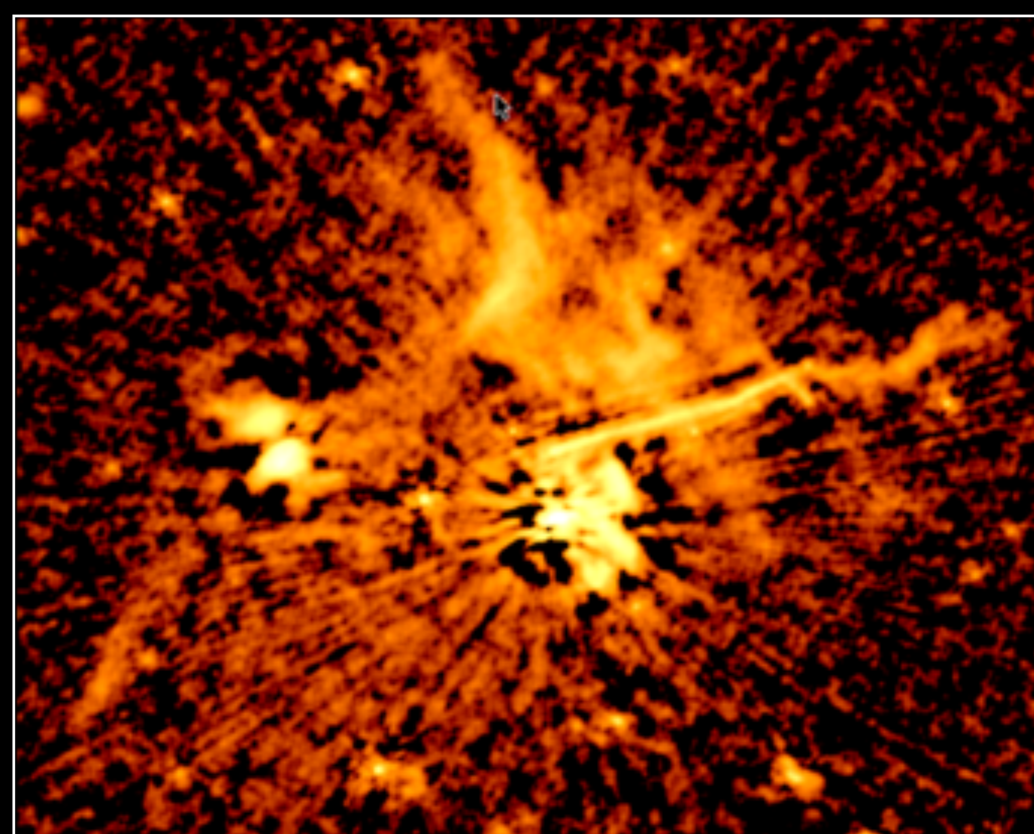
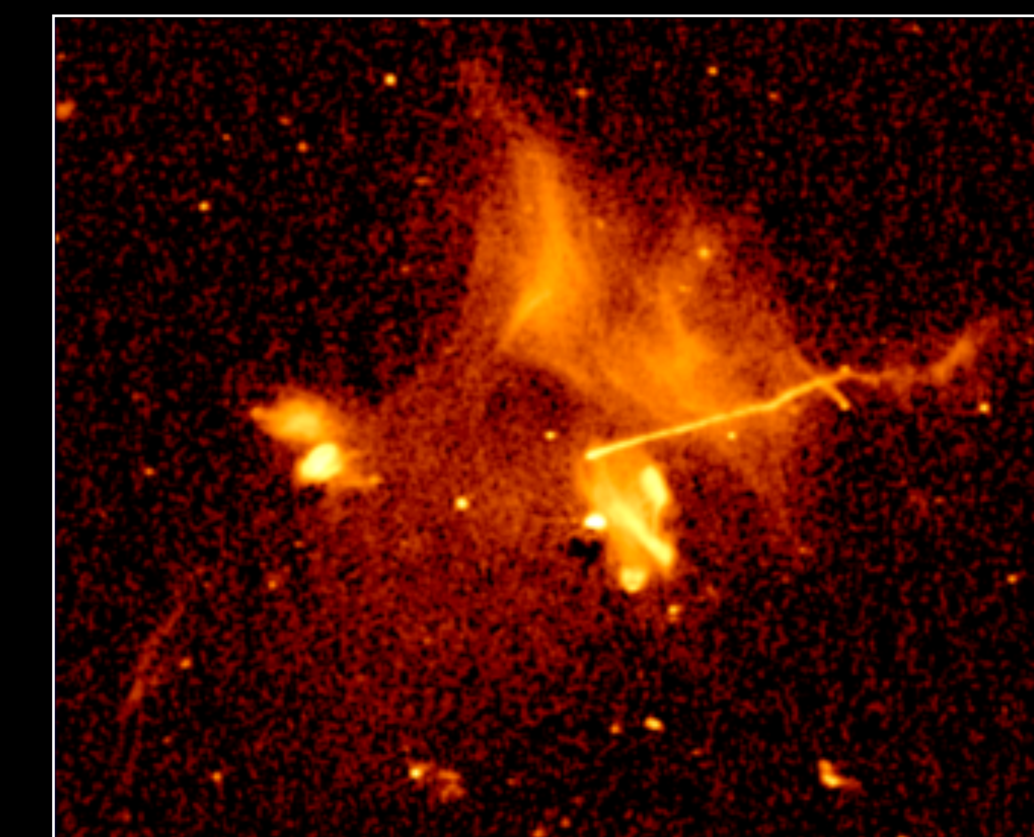
# LOFAR self-calibration and facetselfcal






Reinout van Weeren

Roland Timmerman, Frits Sweijen, Jurjen de Jong, Christian Groeneveld, and many others

*Leiden Observatory, Leiden University*



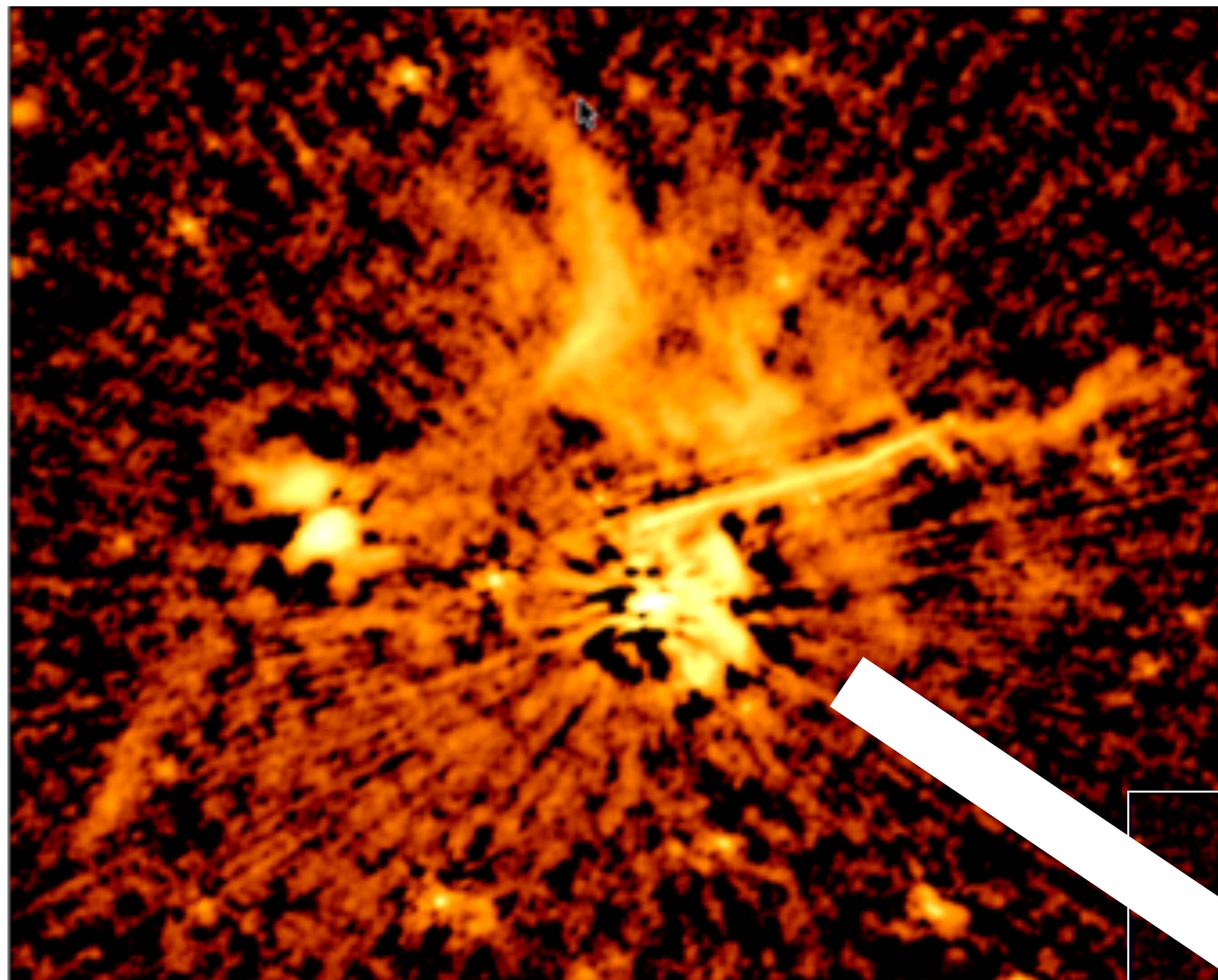
European Research Council

NWO Vidi scheme

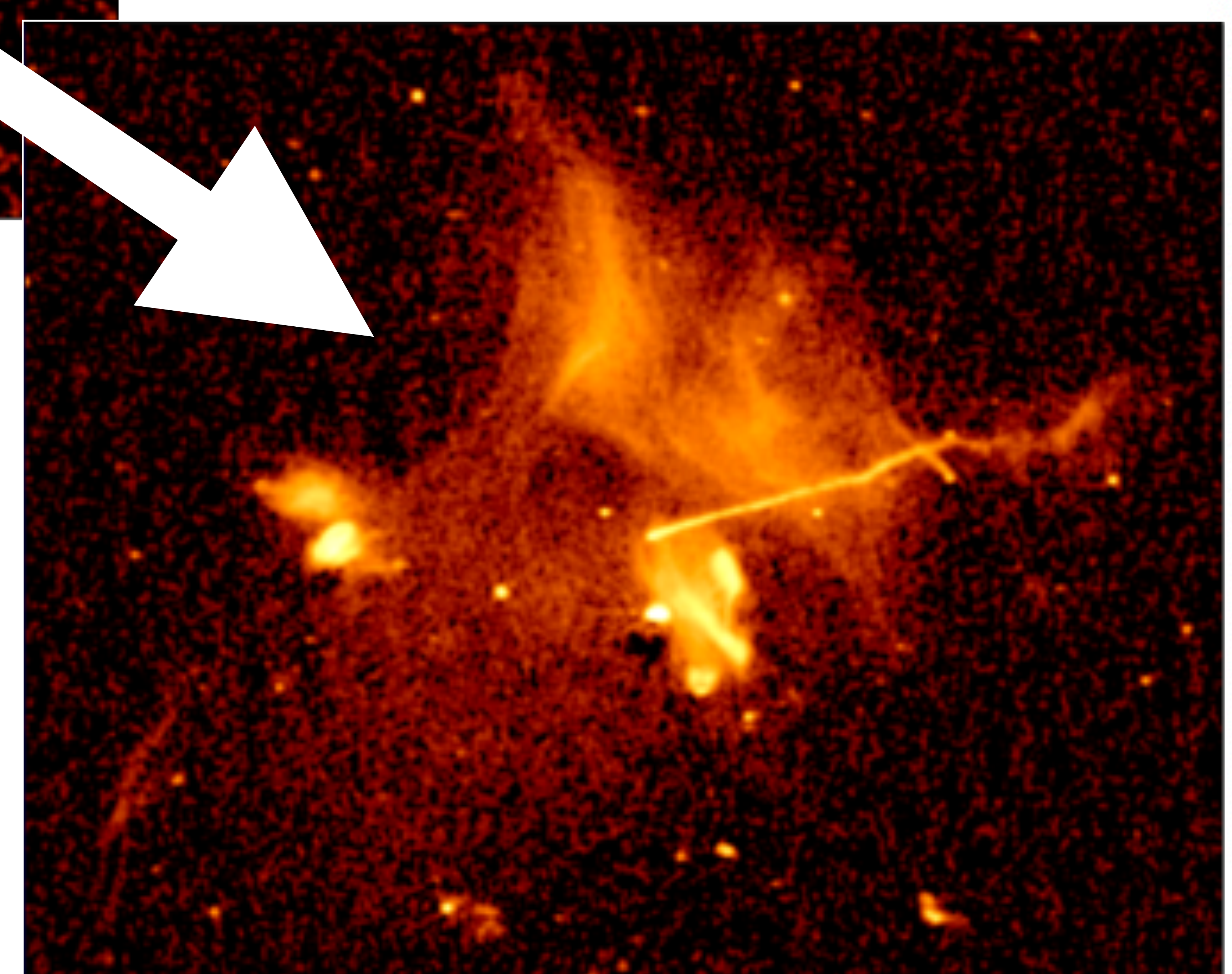
# Outline

- Introduction
- self-calibration
- facetcalibration

# THE MAIN IDEA



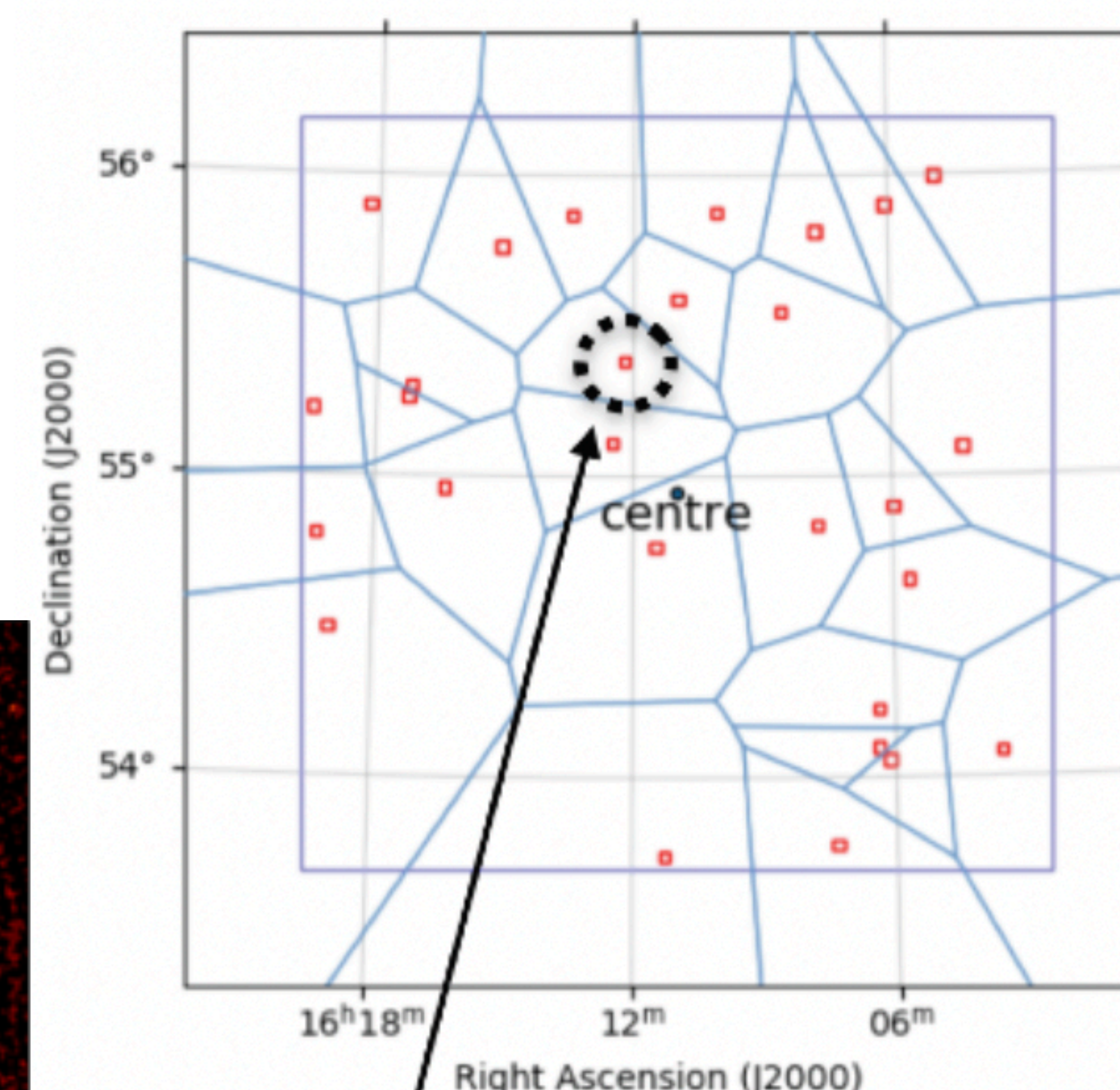
Classic calibration (DI)



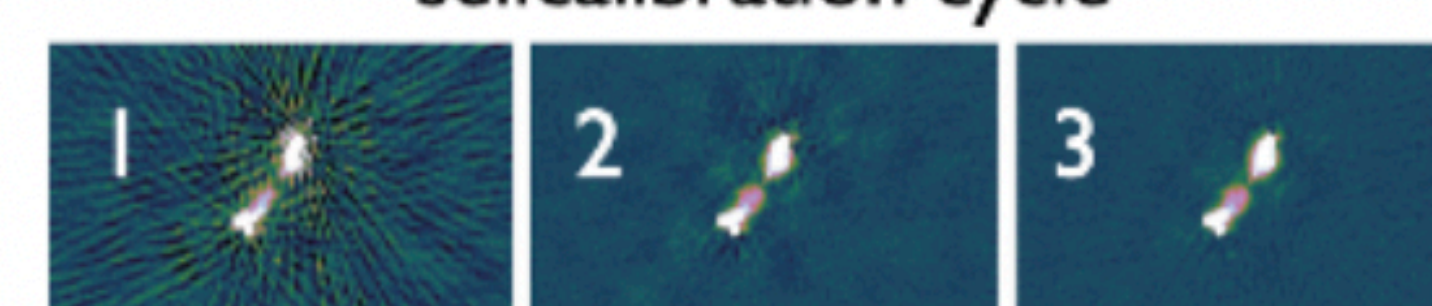
Facet calibration (DD)

van Weeren (2016, 2021)

facet imaging




selfcalibration cycle



# (self-)calibration: time

$$V_{ij}(t) = g_i(t)g_j^*(t)V_{ij,true}(t) + \varepsilon_{ij}(t)$$

$V_{ij}(t)$ = Visibility measured between antennas i and j

$g_i(t)$ = Complex gain corrections of antenna i

$V_{ij,true}(t)$ = True visibility (in practice we want to find that)

$\varepsilon_{ij}(t)$ = Noise

* = conjugation

# (self-)calibration: time

$$V_{ij}(t) = g_i(t)g_j^*(t)V_{ij,true}(t) + \varepsilon_{ij}(t)$$

$V_{ij}(t)$ = Visibility measured between antennas i and j

$g_i(t)$ = Complex gain corrections of antenna i

$V_{ij,true}(t)$ = True visibility (in practice we want to find that)

$\varepsilon_{ij}(t)$ = Noise

* = conjugation

- Number $g_i$ at a given time = $N_{antenna}$
- Number $V_{ij}$ at a given time = $N(N-1)/2$ = $N_{baselines}$
- Solving for $g_i(t)$ can be done because there are more measurements $V_{ij}(t)$ than unknown gain corrections $g_i(t)$
- We want: $N_{baselines} >> N_{antenna}$

# (self-)calibration

Cornwell & Fomalont (1989)

$$\mathcal{S} = \sum_k \sum_{\substack{i,j \\ i \neq j}} w_{ij}(t_k) \left| \widetilde{V}_{ij}(t_k) - g_i(t_k) g_j^*(t_k) \widehat{V}_{ij}(t_k) \right|^2$$

Data

Model

Each visibility has a weight

Minimize sum over all baselines

gains (what we want)

# (self-)calibration: time, frequency

$$V_{ij}(t) = g_i(t,\nu)g_j^*(t,\nu)V_{ij,true}(t,\nu) + \varepsilon_{ij}(t,\nu)$$

$V_{ij}(t,\nu)$ = Visibility measured between antennas i and j

$g_i(t,\nu)$ = Complex gain corrections of antenna i

$V_{ij,true}(t,\nu)$ = True visibility (in practice we want to find that)

$\varepsilon_{ij}(t,\nu)$ = Noise

* = conjugation

- Number $g_i$ at a given time = $N_{antenna}$
- Number $V_{ij}$ at a given time = N(N-1)/2 = $N_{baselines}$
- Solving for $g_i(t,\nu)$ can be done because there are more measurements $V_{ij}(t,\nu)$ than unknown gains $g_i(t,\nu)$
- We want: $N_{baselines} >> N_{antenna}$

# self-calibration: iteration

$$V_{ij}(t,\nu) = g_i(t,\nu)g_j^*(t,\nu)V_{ij,model}(t,\nu) + \varepsilon_{ij}(t,\nu)$$

$V_{ij}(t,\nu)$ = Visibility measured between antennas i and j

$g_i(t,\nu)$ = Complex gain of antenna i

$V_{ij,model}(t,\nu)$ = Model visibility ("close" to truth)

$\varepsilon_{ij}(t,\nu)$ = Noise

# self-calibration: iteration

$$V_{ij}(t,\nu) = g_i(t,\nu)g_j^*(t,\nu)V_{ij,model}(t,\nu) + \varepsilon_{ij}(t,\nu)$$

$V_{ij}(t,\nu)$ = Visibility measured between antennas i and j

$g_i(t,\nu)$ = Complex gain of antenna i

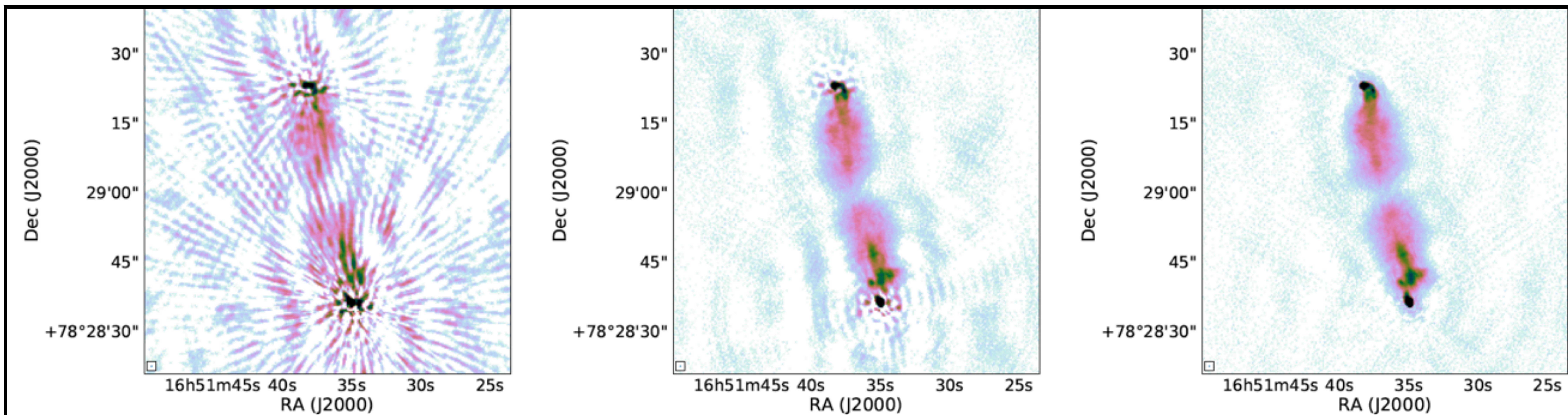
$V_{ij,model}(t,\nu)$ = Model visibility ("close" to truth)

$\varepsilon_{ij}(t,\nu)$ = Noise

- Make image + deconvolve image ⇛ produces a model ($V_{ij,model}$)
- Solve for $g_i(t,\nu)$
- Remake image with corrected data $g_i(t,\nu)g_j^*(t,\nu)V_{ij}$ + deconvolve
- Solve for $g_i(t,\nu)$
- Remake image with corrected data $g_i(t,\nu)g_j^*(t,\nu)V_{ij}$ + deconvolve
- Repeat until image does not change….

# self-calibration: iteration

iteration 0 | iteration 1 | iteration 2



# Ionosphere & station beam

Antenna corrections are also direction dependent: 😱

$g_i(t, \nu, \text{direction})$

Ionosphere and beam also effect polarization (Faraday Rotation):

😱 😱

$g_i(t, \nu, \text{direction}, \text{polarization})$

What's the problem? We have computers….?

# Challenges

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

4. Starting models (initial guesses) are (sometimes) not good, not guaranteed calibration converges

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

4. Starting models (initial guesses) are (sometimes) not good, not guaranteed calibration converges

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

4. Starting models (initial guesses) are (sometimes) not good, not guaranteed calibration converges

5. Data size is large: computationally challenging

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

4. Starting models (initial guesses) are (sometimes) not good, not guaranteed calibration converges

5. Data size is large: computationally challenging

# Challenges

1. Data is noisy. Finding the gains is an optimization problem: we need enough signal/ measurements

2. Number of fitting parameters can be too large for the number of measurements: Overfitting

3. Long baselines and low frequencies: number of gains we need to find is large (gains vary as a function of time, frequency, polarization, direction)

4. Starting models (initial guesses) are (sometimes) not good, not guaranteed calibration converges

5. Data size is large: computationally challenging

LOFAR: We are affected by all of these problems

# LOFAR calibration errors

- Ionospheric TEC: phase effect
- Ionospheric Faraday Rotation: phase difference between RR and LL polarizations
- Clock (±15 ns variations)

*phase error* $\propto$ TEC/$\nu$

*phase error RR-LL* $\propto$ RM/$\nu^2$

*phase error* $\propto$ clock x $\nu$

- Beam: LOFAR stations have beams that differ from our used beam model
- At lower level all polarizations are affected: fulljones

*amplitude errors: slowly varying*

*small complex valued (amplitude & phase) errors for all four polarizations: slowly varying*

# LOFAR calibration errors

- Ionospheric TEC: phase effect

*phase error* $\propto$ TEC/$\nu$

Make use of all this information to reduce the number of parameters we need to solve for.

- At lower level all polarizations are affected: fulljones

*small complex valued (amplitude & phase) errors for all four polarizations: slowly varying*

# Errors in images

# Errors in images

# Errors in images

# LOFAR DI/DD calibration & imaging

- Two main calibration programs: DP3 & killMS (van Diepen et al. 2018; Tasse et al. 2014ab)

- Two main imaging programs: WSClean & DDFacet (Offringa et al. 2014, 2017; Tasse et al. 2018)

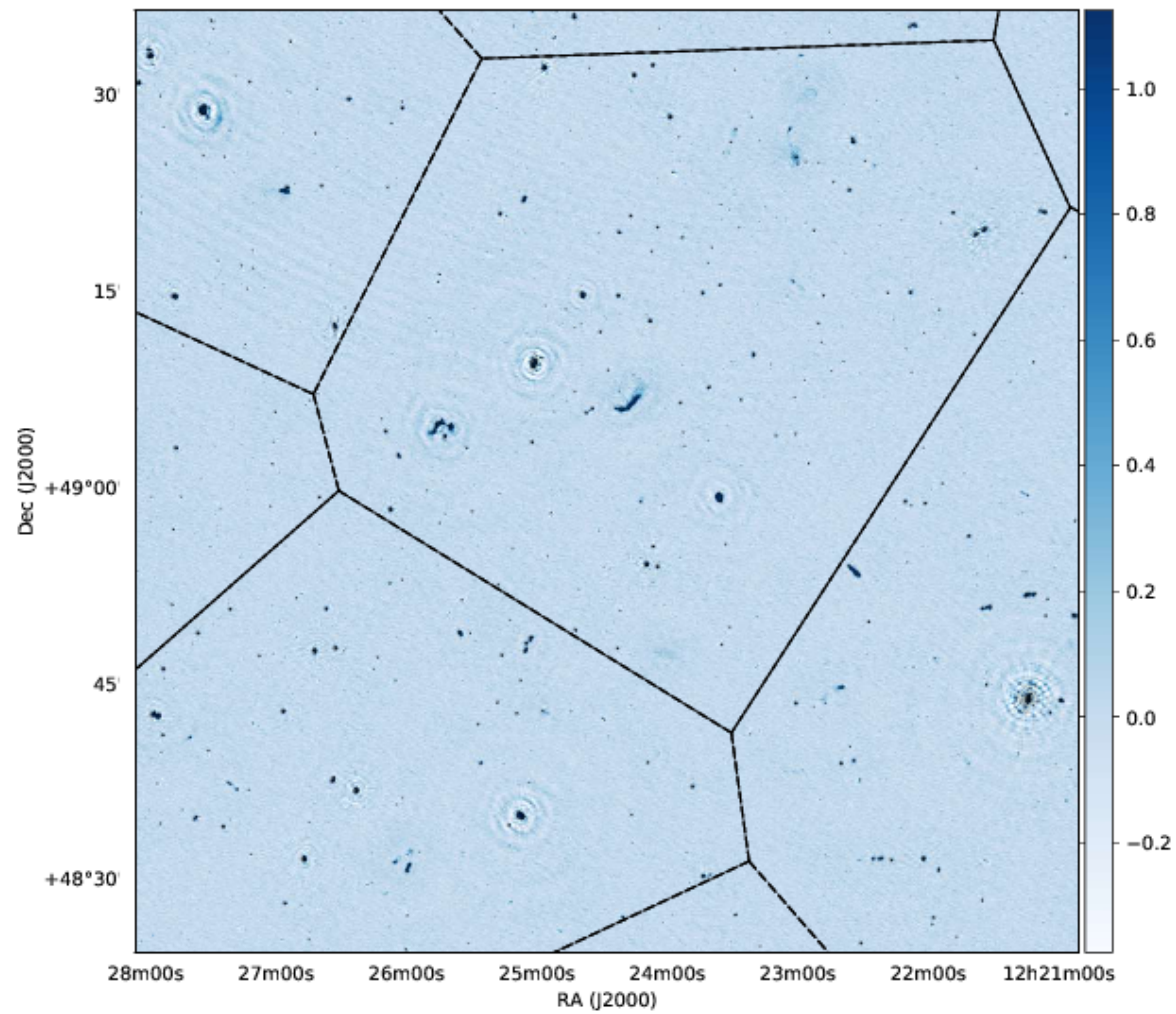
- Integrated into pipelines: DDF-pipeline (Tasse et al. 2021) & Rapthor

# Background

Why was facetselfcal developed?

# LoTSS & LoLSS processing

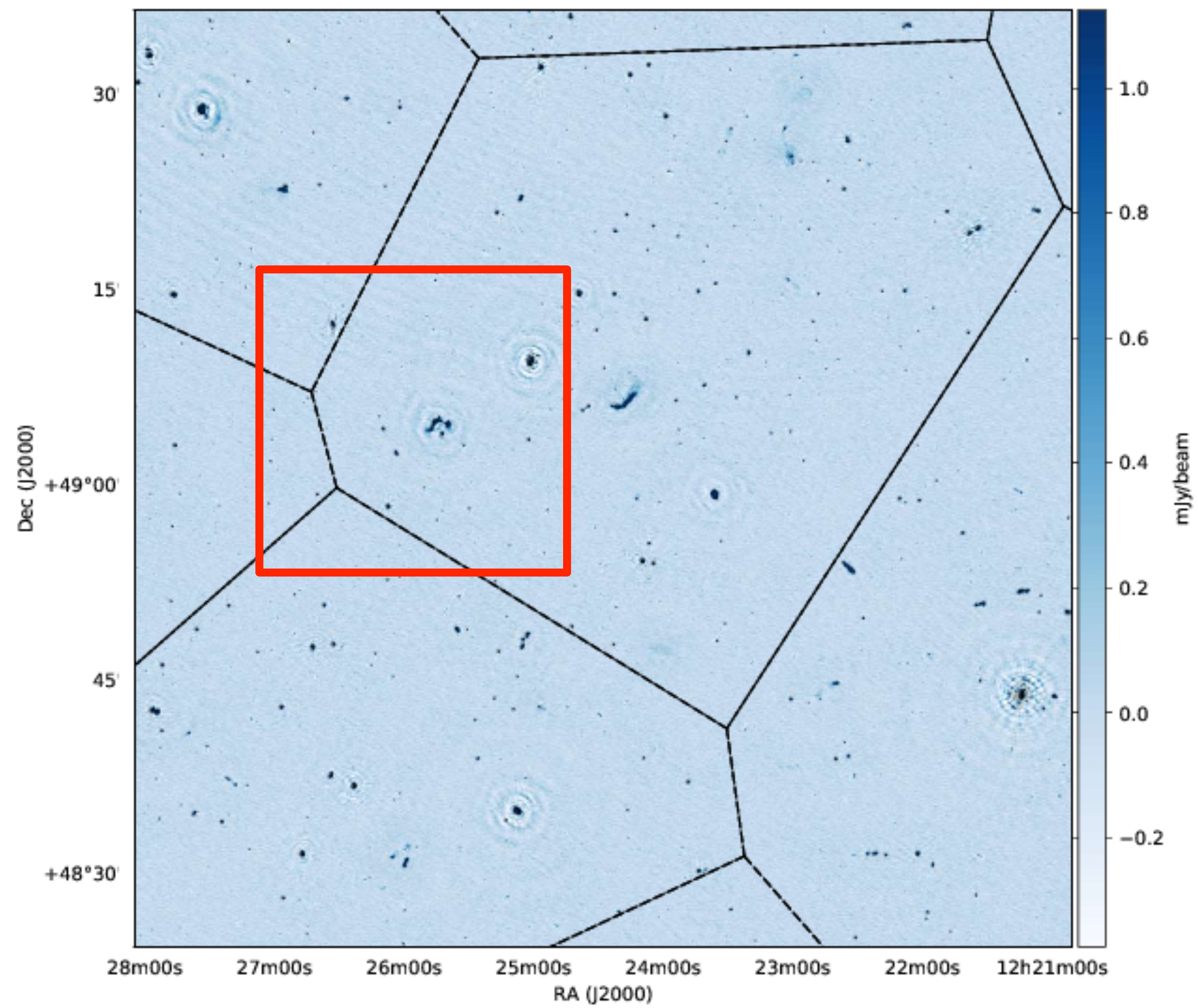
- Facet layout can be non-optimal for target-of-interest given that DDE corrections work on a per-facet basis
- Target can be located in two or more overlapping pointings
- Weightings scheme and uv-cuts might not be ideal for science case
- Re-imaging is expensive (uv-tapers, weightings, different deconvolution)




DDF-pipeline (Tasse+ 2021) makes use of DDFacet and kMS for calibration and imaging (Tasse+ 2014; Smirnov+ 2015; Tasse+ 2017). LoLSS pipeline (de Gasperin+2019,2020,2021) makes use of DDFacet, WSClean (Offringa+ 2014; Offringa & Smirnov 2017) and DP3 (van Diepen+ 2018).

# LoTSS & LoLSS processing

- Facet layout can be non-optimal for target-of-interest given that DDE corrections work on a per-facet basis
- Target can be located in two or more overlapping pointings
- Weightings scheme and uv-cuts might not be ideal for science case
- Re-imaging is expensive (uv-tapers, weightings, different deconvolution)




DDF-pipeline (Tasse+ 2021) makes use of DDFacet and kMS for calibration and imaging (Tasse+ 2014; Smirnov+ 2015; Tasse+ 2017). LoLSS pipeline (de Gasperin+2019,2020,2021) makes use of DDFacet, WSClean (Offringa+ 2014; Offringa & Smirnov 2017) and DP3 (van Diepen+ 2018).

# facetselfcal

van Weeren+ (2021)

https://github.com/rvweeren/lofar_facet_selfcal

Calibration: DP3 and (van Diepen et al. 2018)

Imaging: WSClean (Offringa et al. 2014, 2017)

+ own Python code

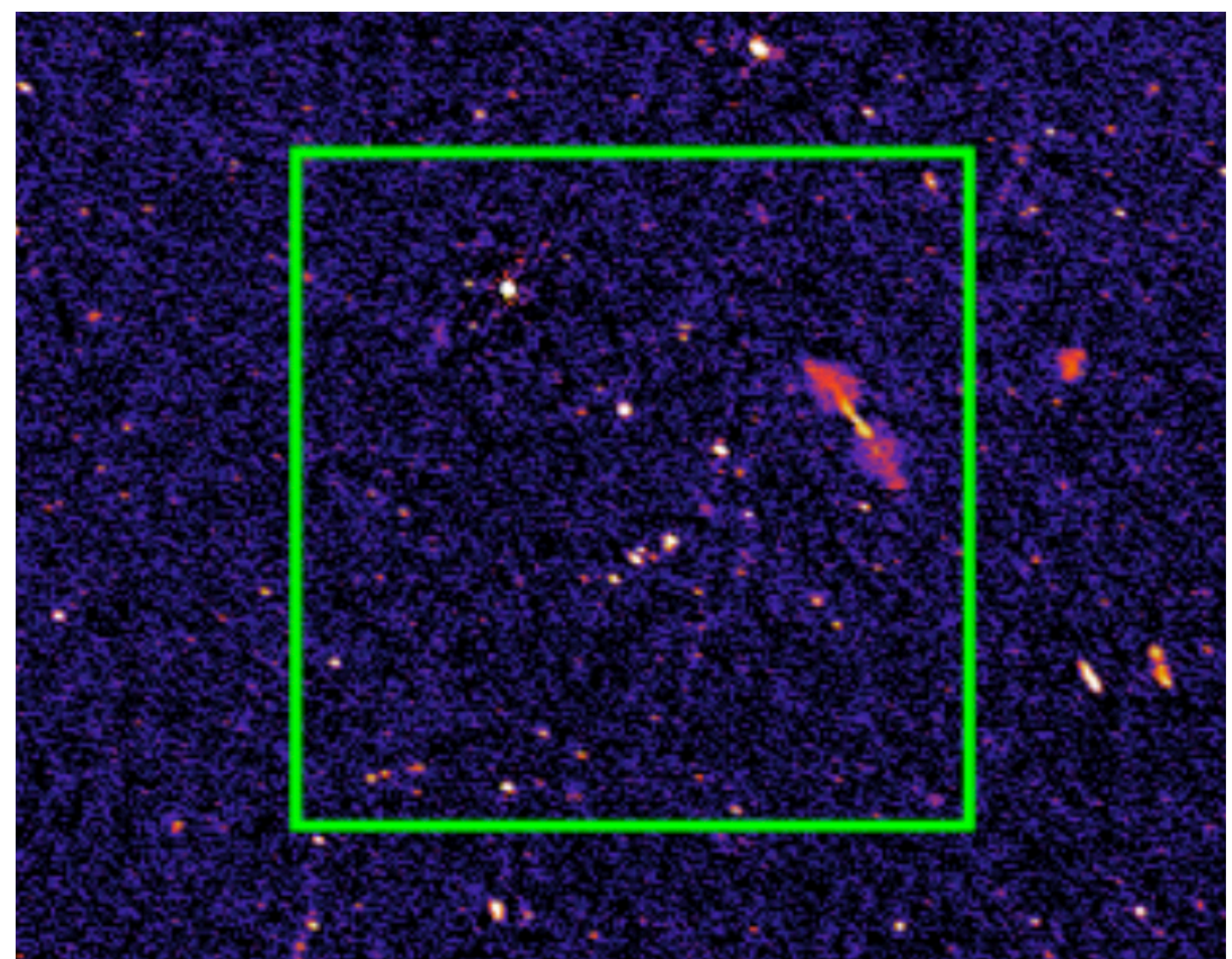
# "Extract"

van Weeren+ (2021); DDF-pipeline (Tasse+2021)

Subtract all sources, except those in the box, with their DD solutions from the visibilities using DDFacet

Requirements:

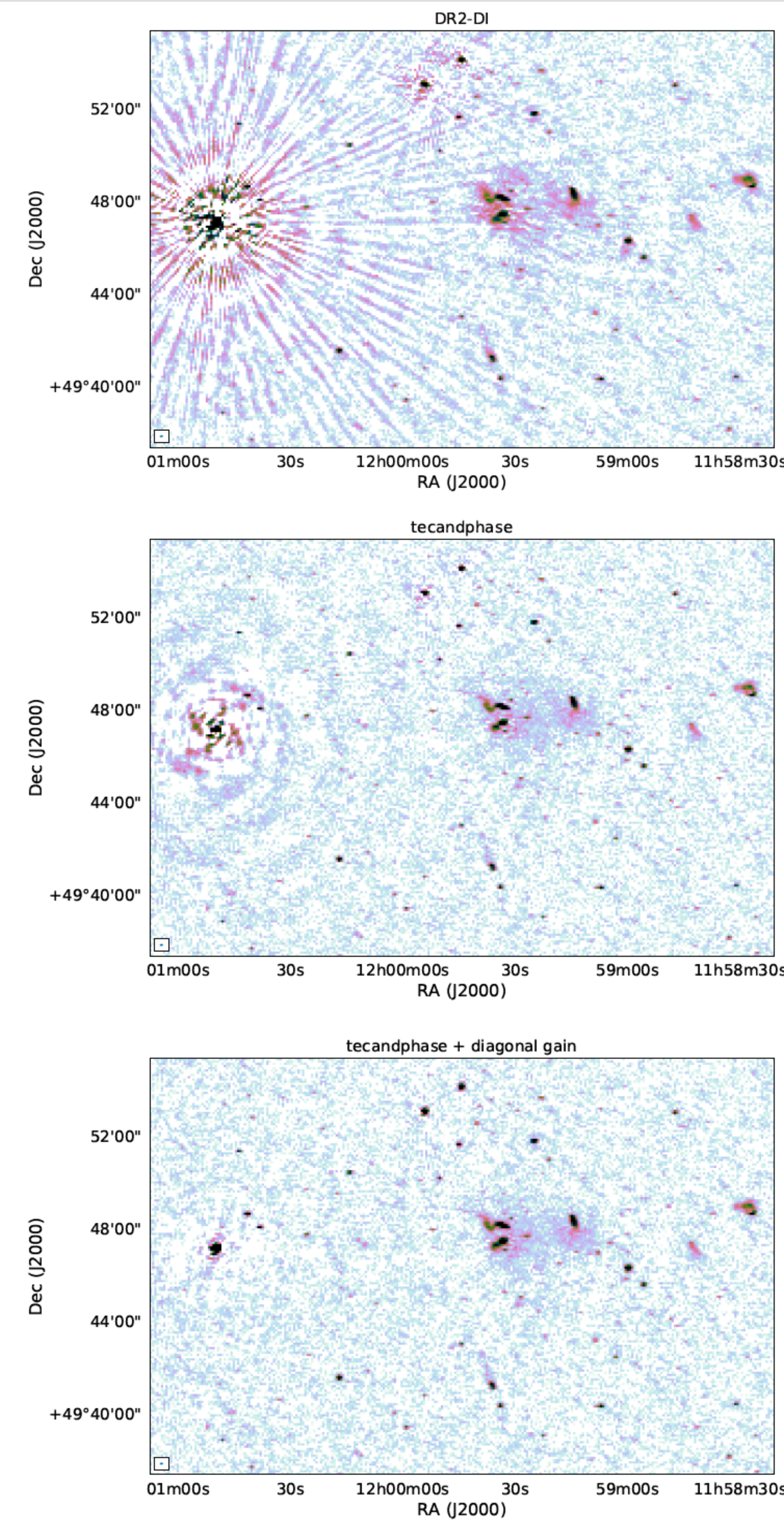
1. Box not too large, avoid DDE effects across the box (≲1.0°)
2. Enough flux in the box for calibration (≳0.2 Jy source Dutch-HBA)



# “Selfcal the facet box”

van Weeren+ (2021)

1. Perturbative solves with automated selfcal
2. Start with biggest effect first
3. Continue with next biggest effect
4. Solution interval provided by user or computed based on visibility noise and model flux
5. Arbitrary number of perturbative steps possible without needing to write code
6. Options 5 makes it a powerful tool to hunt down calibration limitations and test ideas

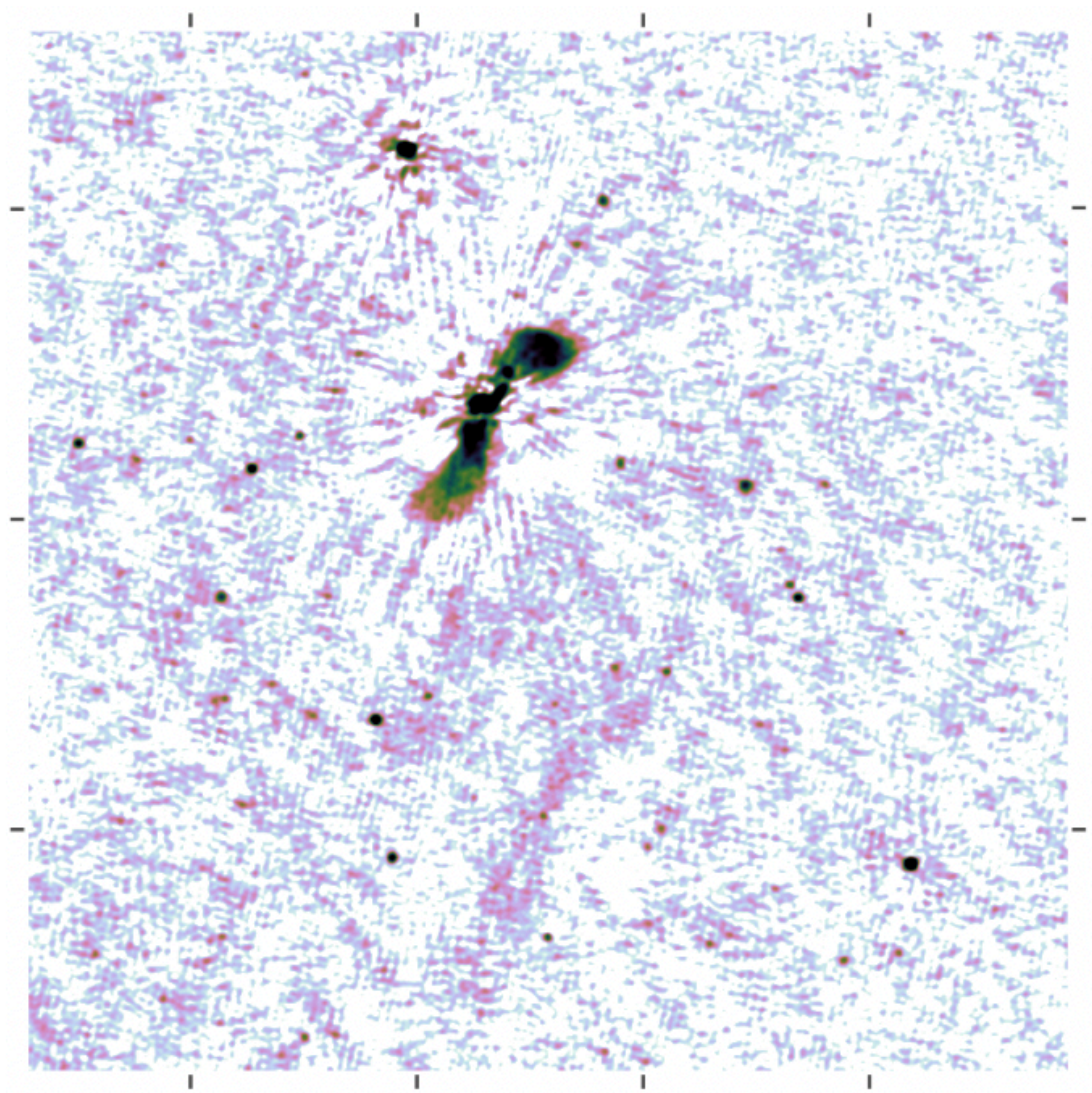


# “Selfcal the facet box”

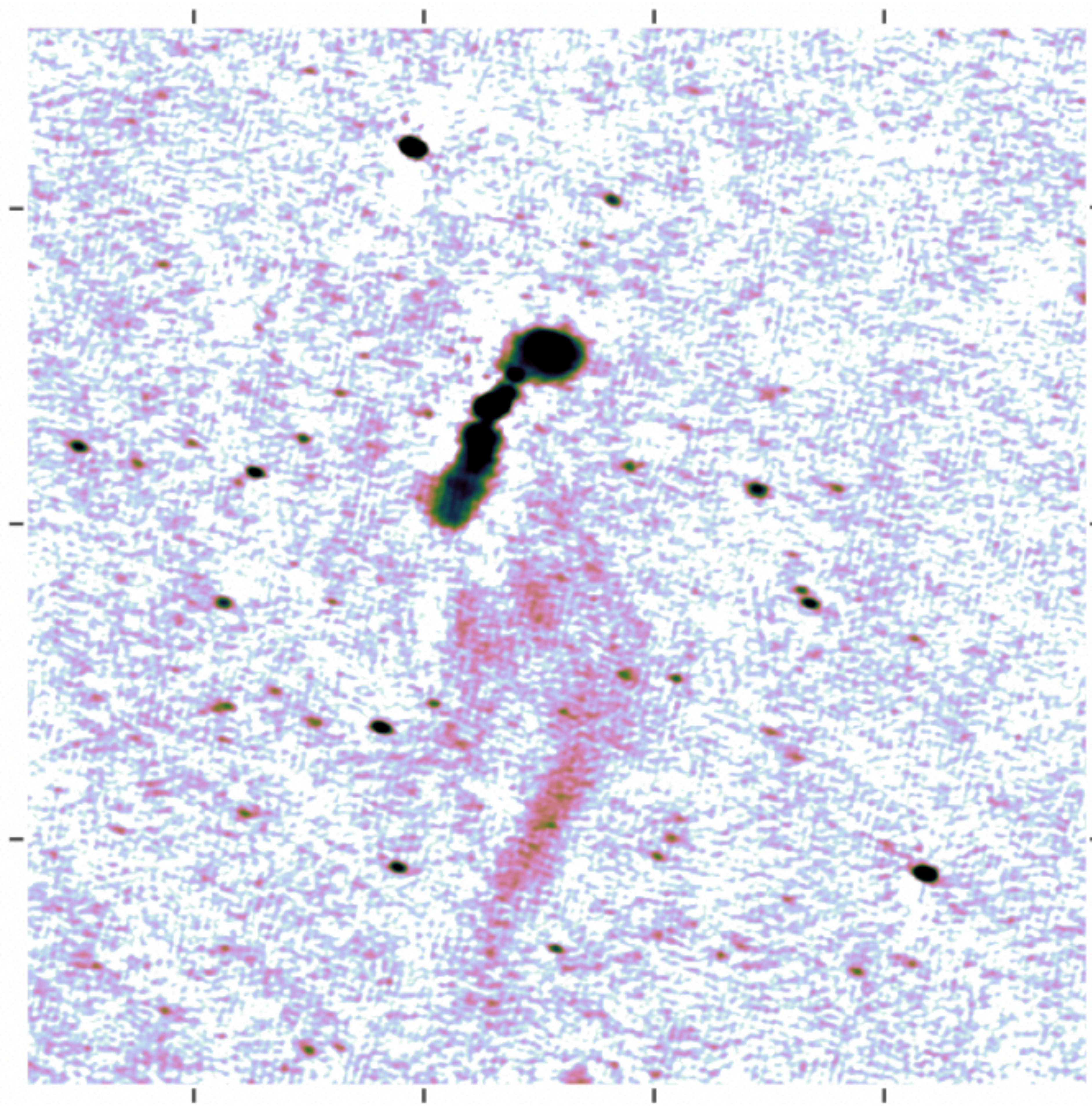
van Weeren+ 2021

- tweaked calibration
- multiscale clean
- no uvmin cut

**LoTSS DR2**

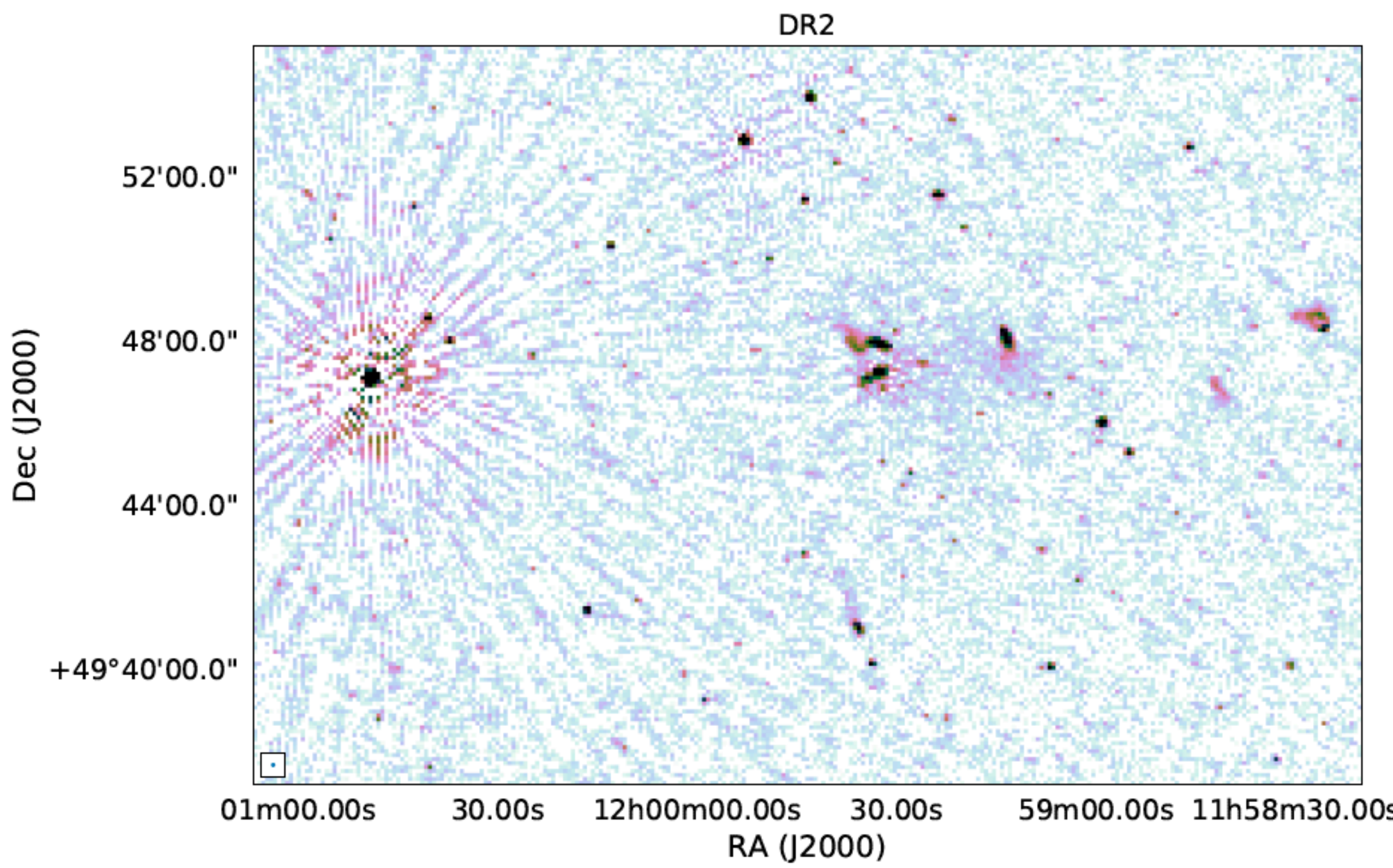


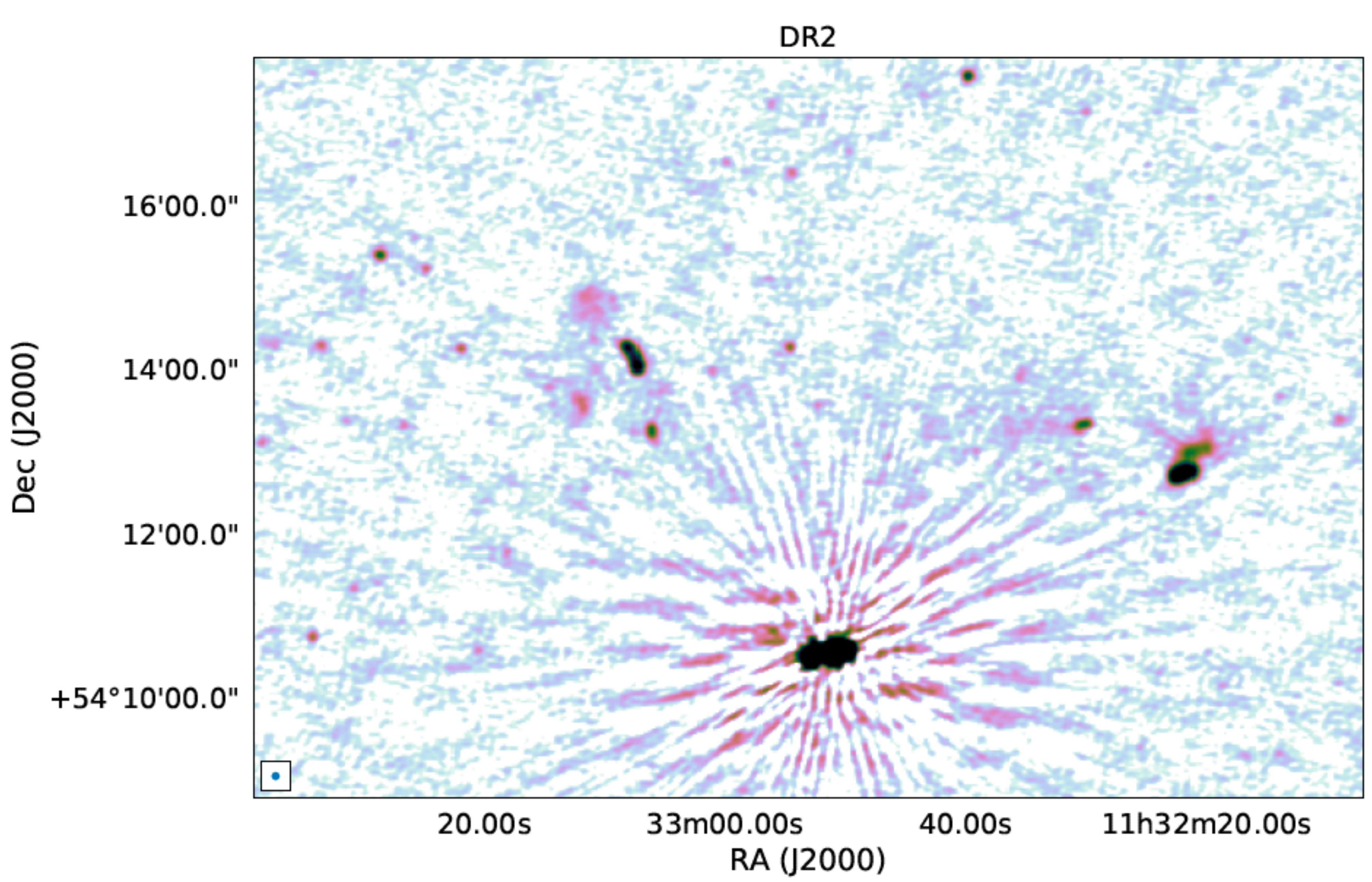
**LoTSS DR2+facetselfcal**



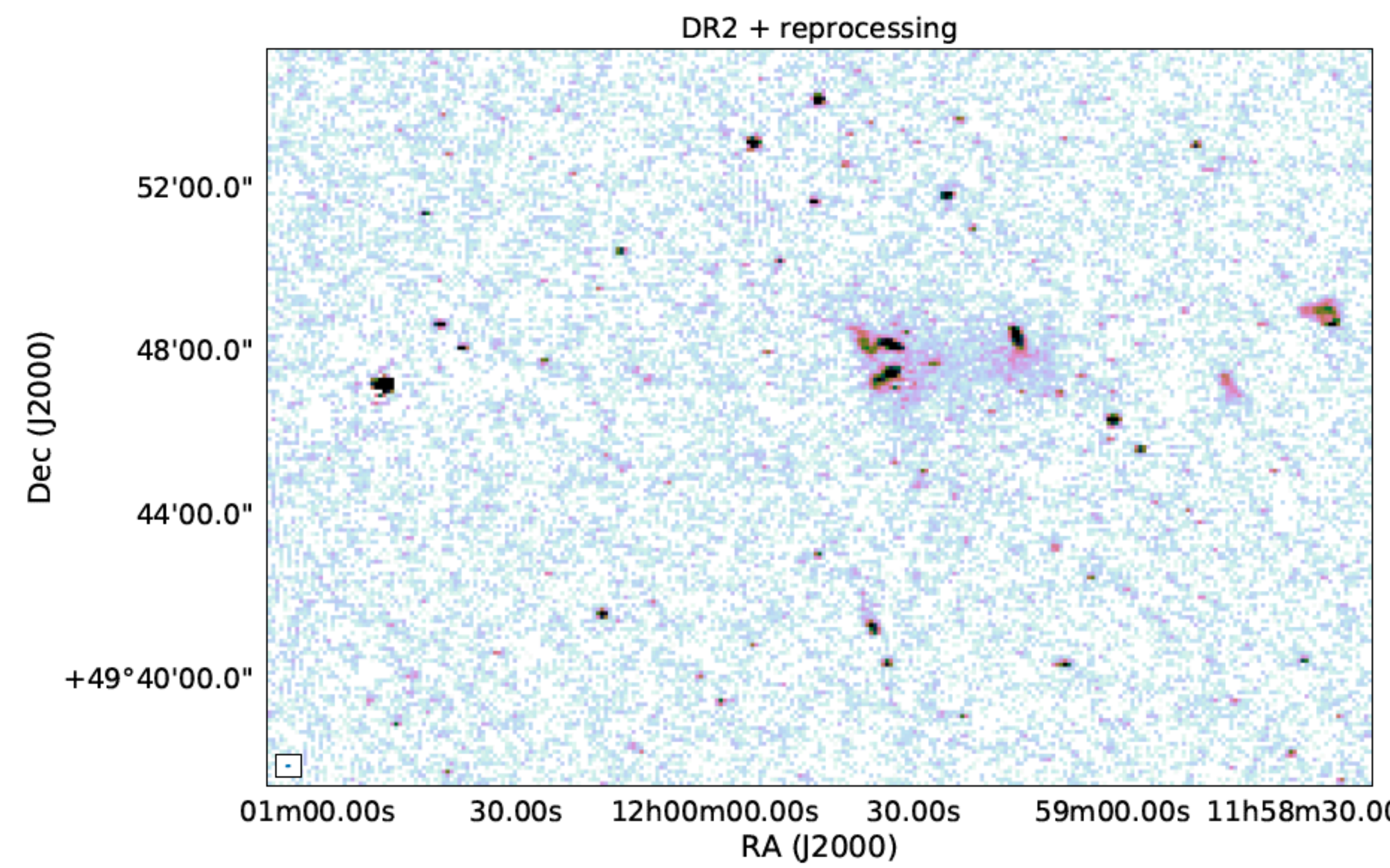
# LoTSS reprocessing

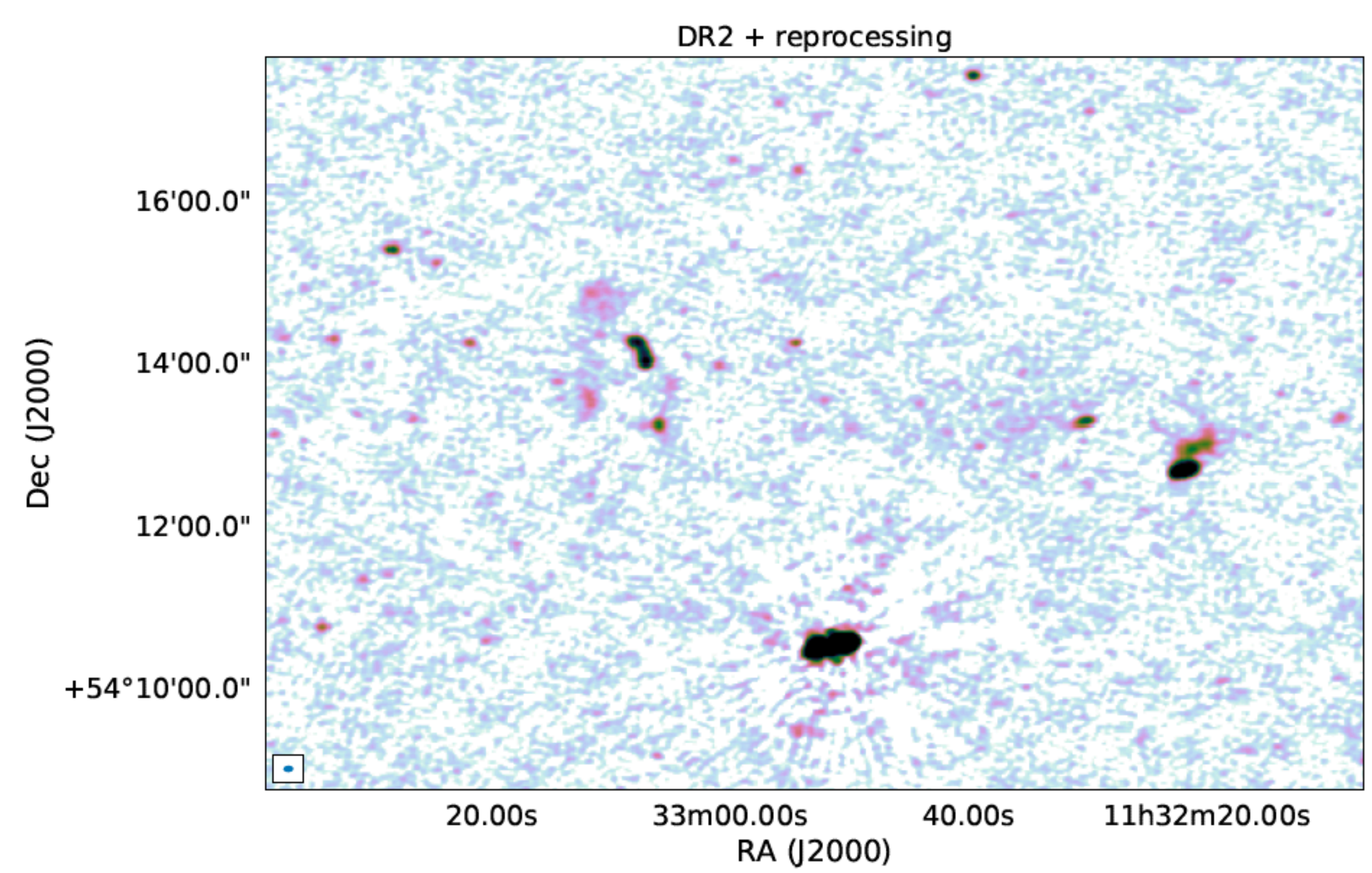
## DR2



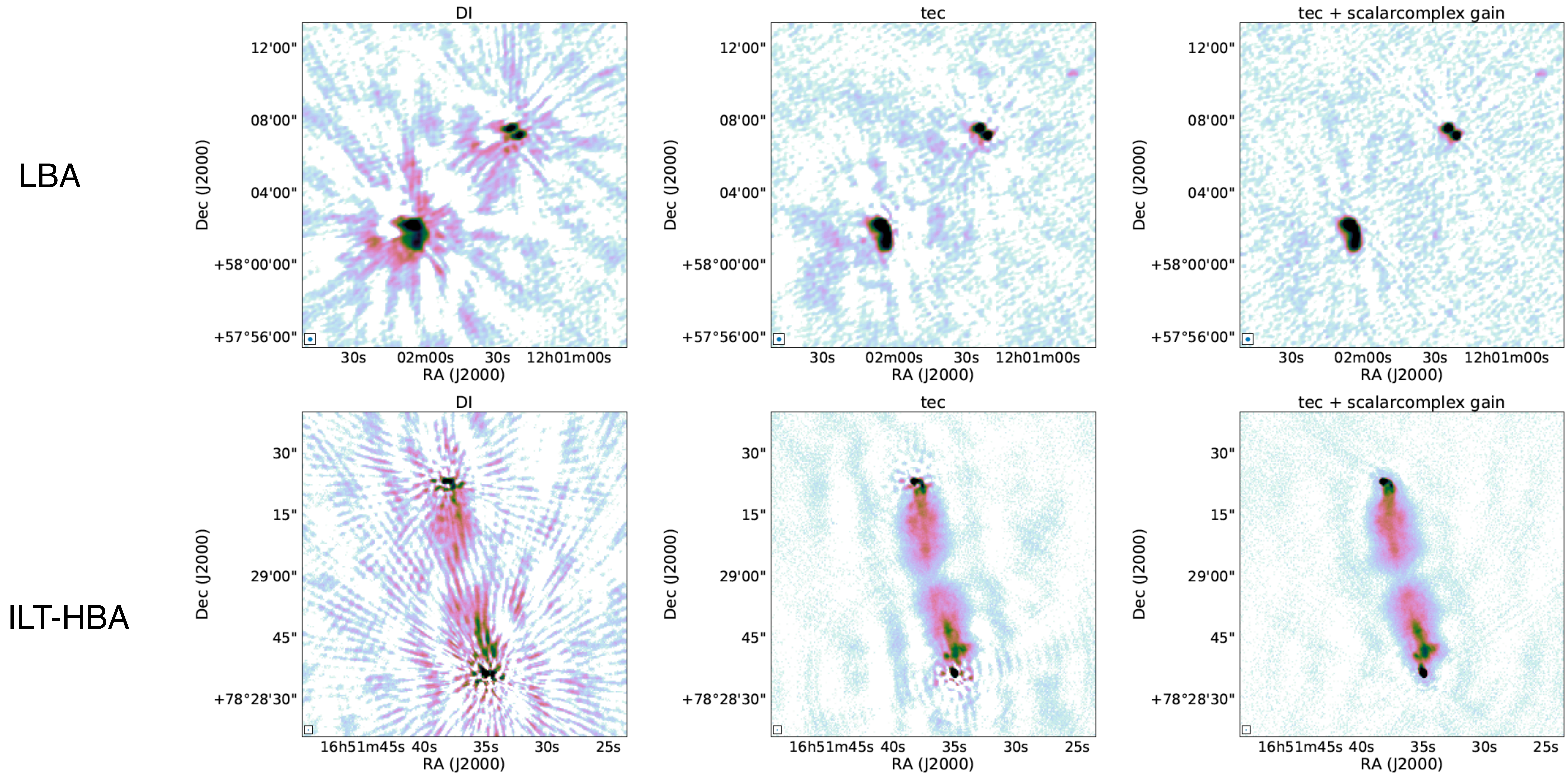




## DR2+facetselfcal



# facetselfcal



# ILT calibration

Jurjen de Jong

Morabito et al. (2022)

Sweijen et al. (2022)

ILT calibration: two-step calibration approach

1. *facetselfcal*: in-field calibrator (corrects bulk of the ionosphere and clock)
2. *facetselfcal*: dozens of facet/target calibrators (DDE ionosphere+beam)

# Facetselfcal

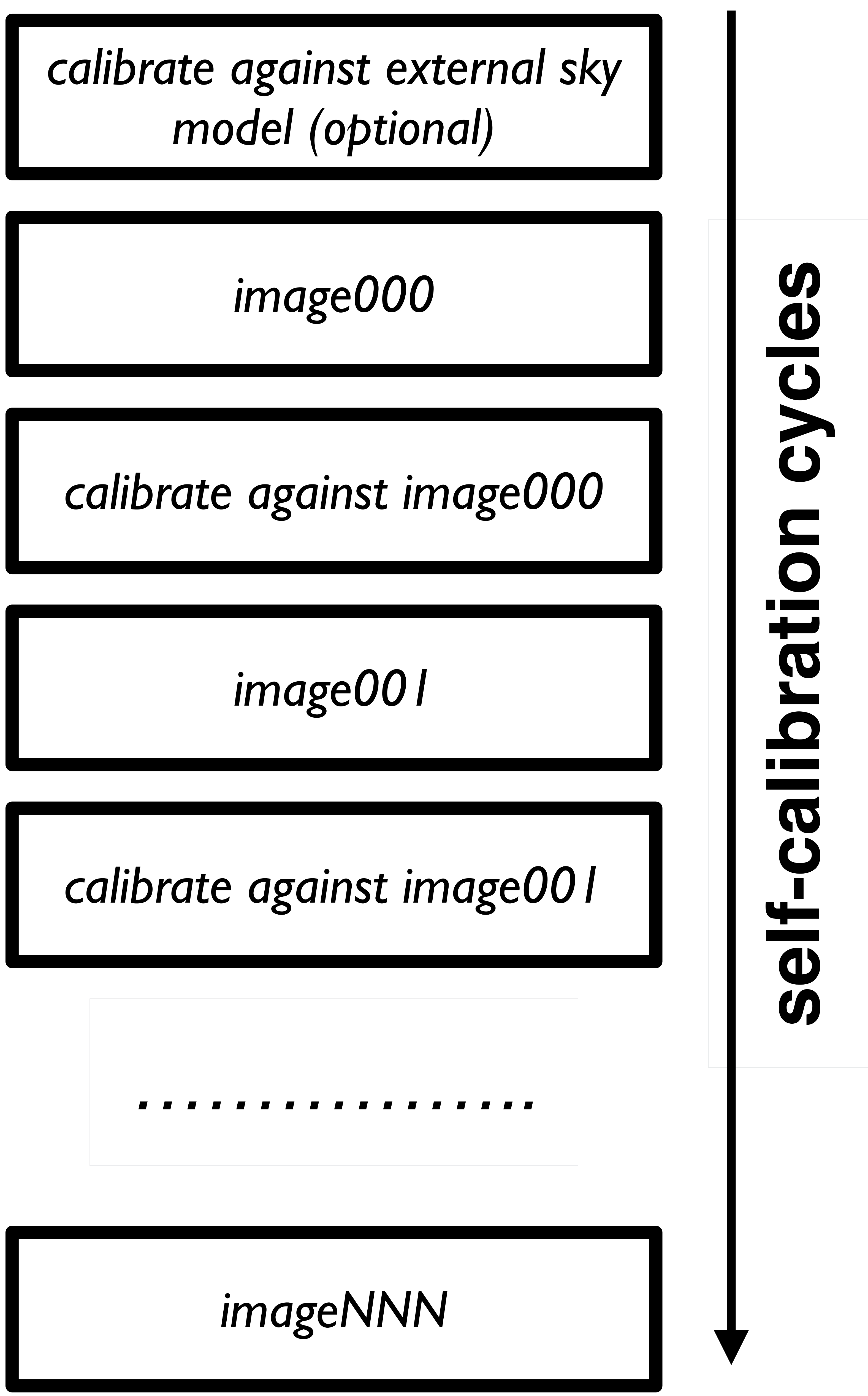
- *two polarizations=XX, YY (RR,LL)*
- *four polarizations=XX,YX,YX,YY (RR,RL,LR,LL)*

Most commonly used solve types:

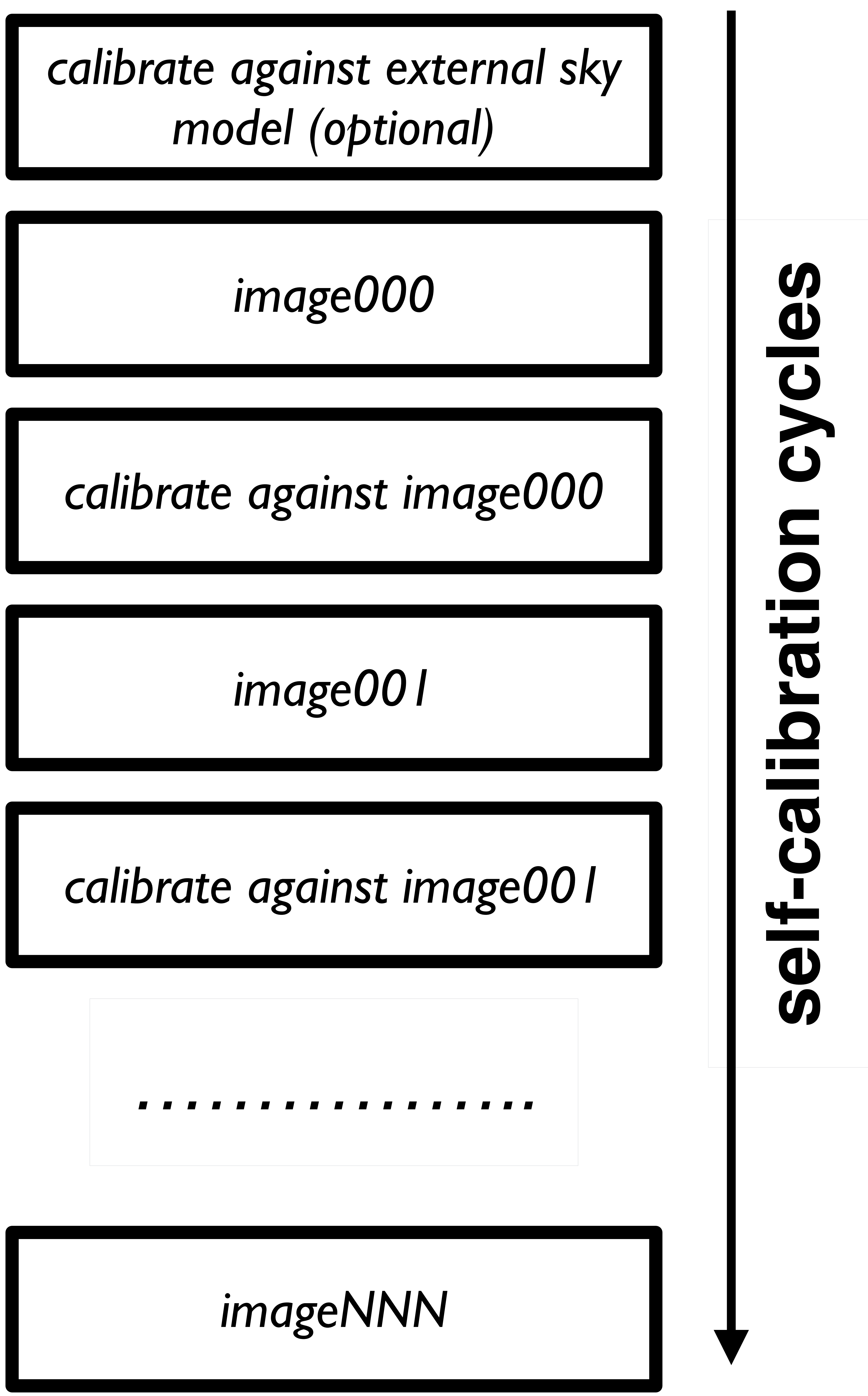
- **scalarphasediff**: solve for the difference between RR and LL
- **scalarphase**: combine the two polarizations and solve for the phases
- **scalarcomplexgain**: combine the two polarizations and solve for amplitude&phase
- **scalaramplitude:** combine the two polarizations and solve for the amplitudes
- **amplitudeonly:** solve for two polarization amplitudes
- **phaseonly:** solve for two polarization phases
- **fulljones:** solve for all four polarizations, amplitude & phase
- *All of the above can handle frequency smoothness (DP3 constrains the solution to be "smooth" along the frequency axis with a user specified value)*

- **tec:** combine the two polarizations and enforce *phase* $\propto \nu^{-1}$
- **tecandphase:** combine the two polarizations and enforce *phase* $\propto \nu^{-1}$ + const

# perturbations

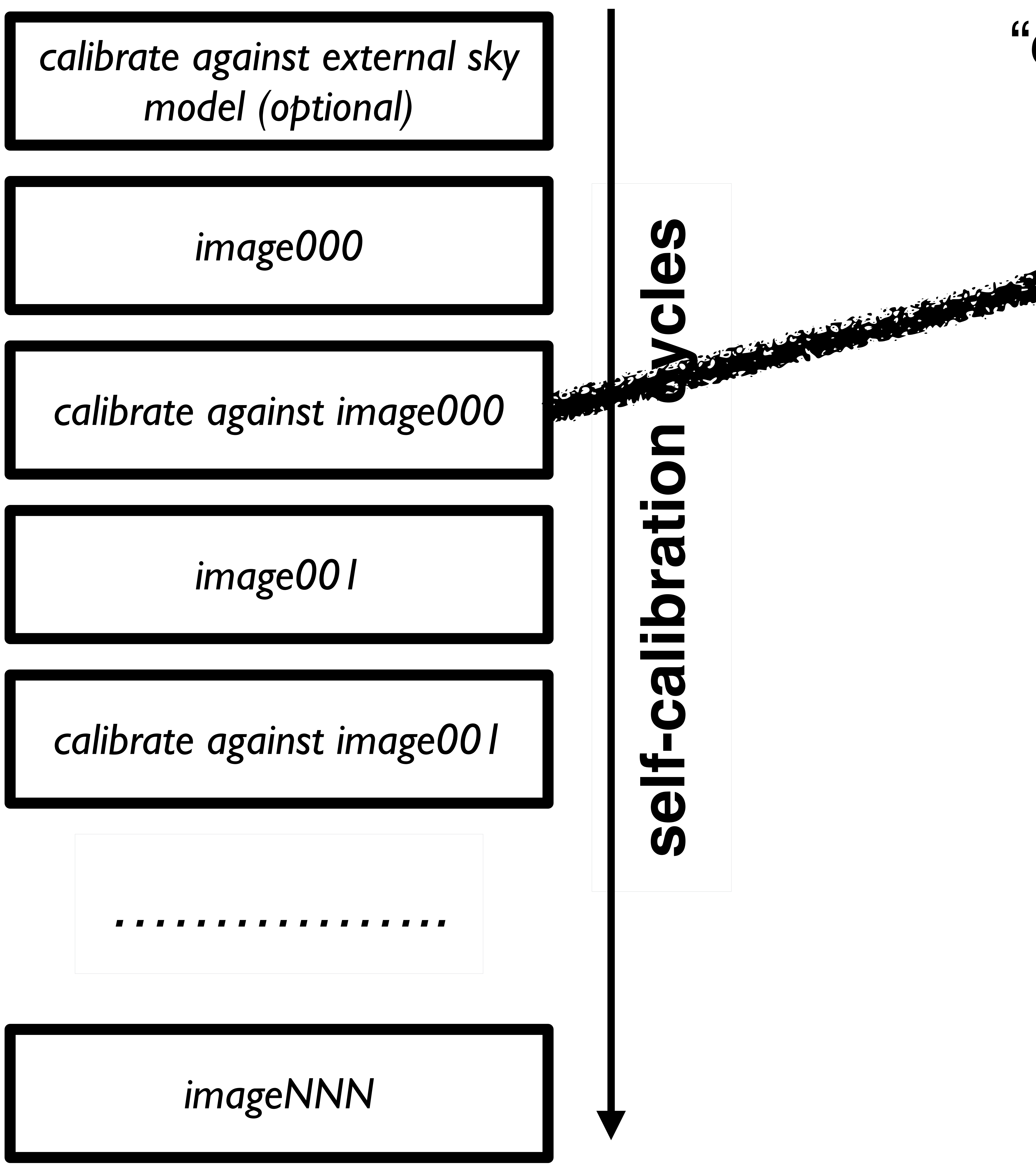


# perturbations

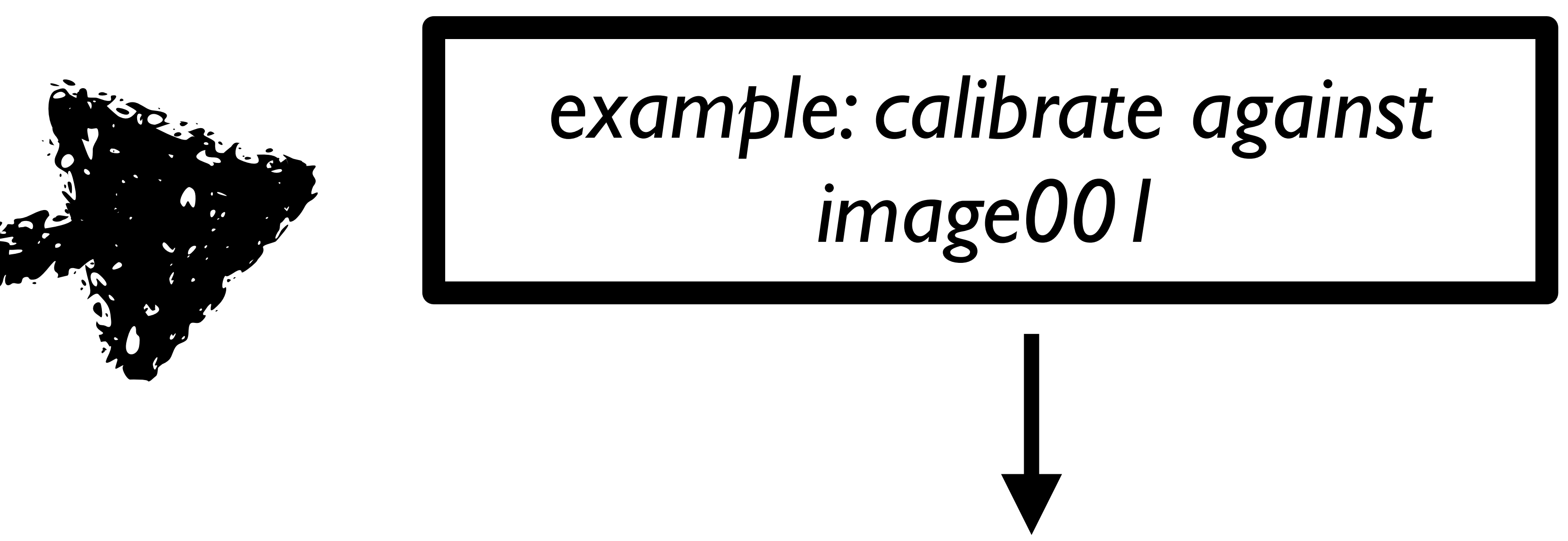



“calibratie step” can involve multiple calibrations: **perturbative** solves

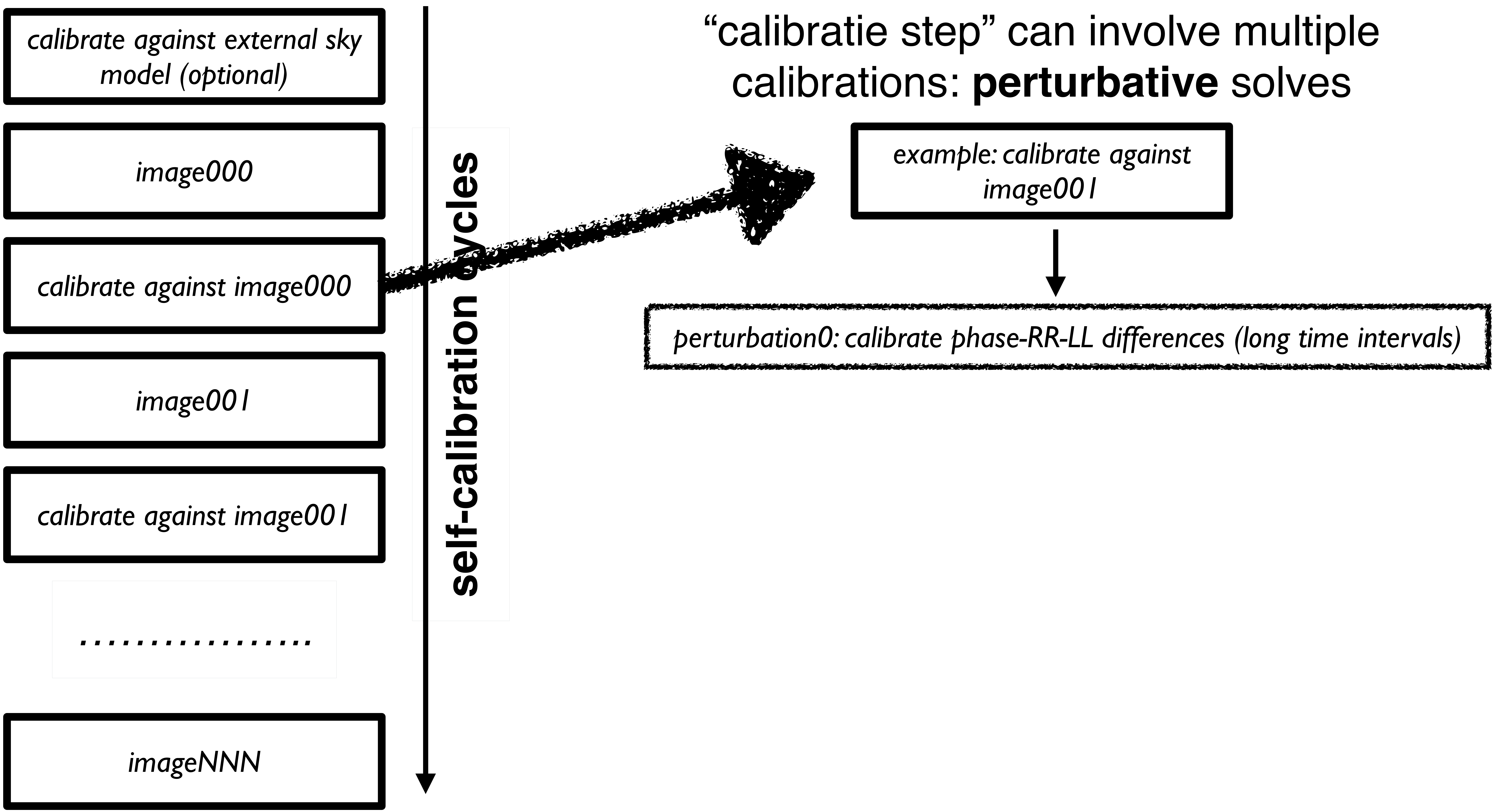
# perturbations




"calibratie step" can involve multiple calibrations: **perturbative** solves



# perturbations



# perturbations

*calibrate against external sky model (optional)*

*image000*

*calibrate against image000*

*image001*

*calibrate against image001*

..................

*imageNNN*

**self-calibration cycles**

“calibratie step” can involve multiple calibrations: **perturbative** solves

*example: calibrate against image001*

*perturbation0: calibrate phase-RR-LL differences (long time intervals)*

*corrupt model with found solutions*

# perturbations

*calibrate against external sky model (optional)*

*image000*

*calibrate against image000*

*image001*

*calibrate against image001*

..................

*imageNNN*

**self-calibration cycles**

"calibratie step" can involve multiple calibrations: **perturbative** solves

*example: calibrate against image001*

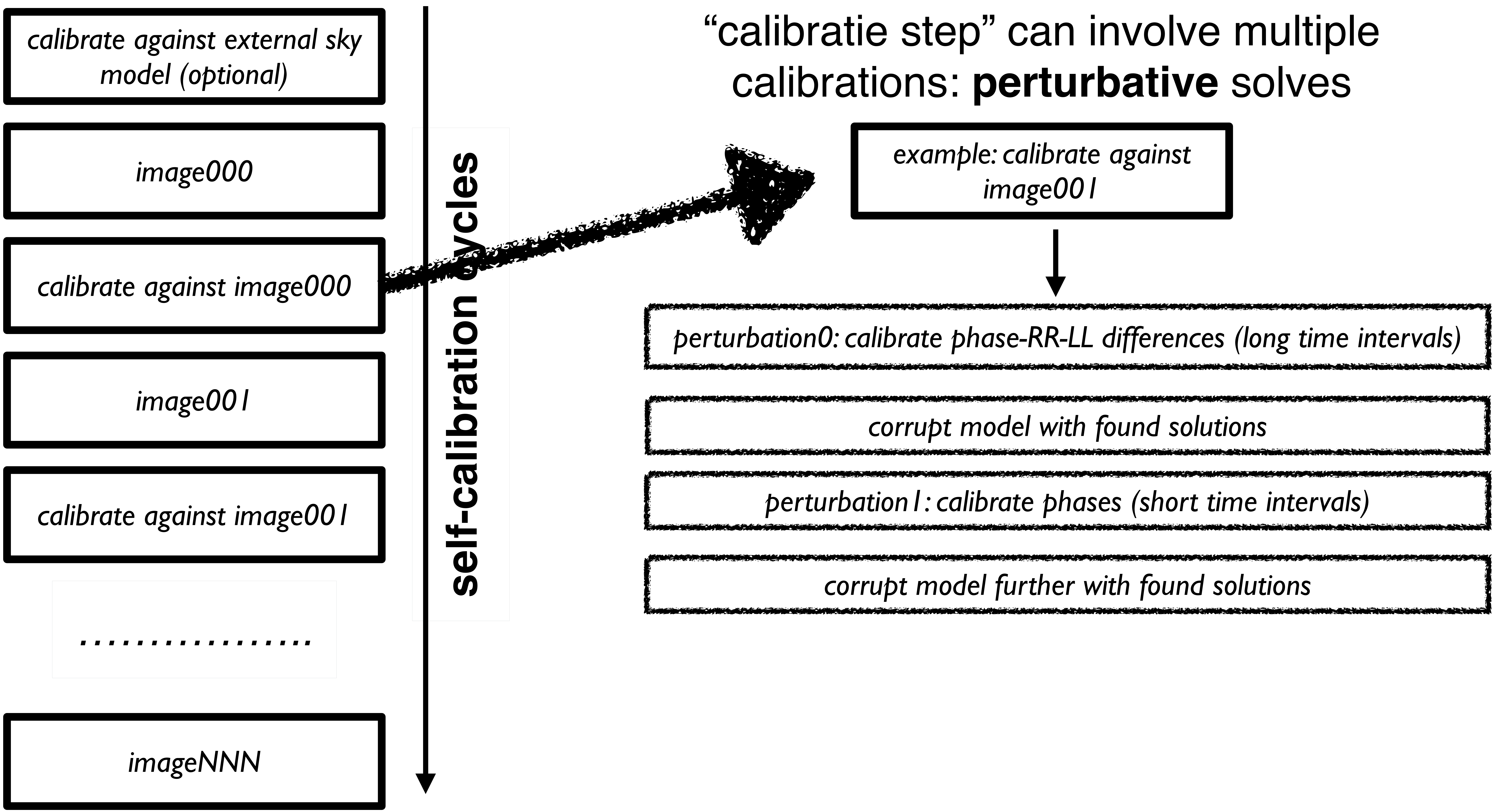
*perturbation0: calibrate phase-RR-LL differences (long time intervals)*

*corrupt model with found solutions*

*perturbation1: calibrate phases (short time intervals)*

# perturbations

“calibratie step” can involve multiple calibrations: **perturbative** solves



# perturbations

*calibrate against external sky model (optional)*

*image000*

*calibrate against image000*

*image001*

*calibrate against image001*

.................

*imageNNN*

**self-calibration cycles**

"calibratie step" can involve multiple calibrations: **perturbative** solves

*example: calibrate against image001*

*perturbation0: calibrate phase-RR-LL differences (long time intervals)*

*corrupt model with found solutions*

*perturbation1: calibrate phases (short time intervals)*

*corrupt model further with found solutions*

*perturbation 2: calibrate phases and amplitudes (long time intervals)*

# perturbations

calibrate against external sky model (optional)

image000

calibrate against image000

image001

calibrate against image001

...................

imageNNN

self-calibration cycles

"calibratie step" can involve multiple calibrations: **perturbative** solves

*example: calibrate against image001*

*perturbation0: calibrate phase-RR-LL differences (long time intervals)*

*corrupt model with found solutions*

*perturbation1: calibrate phases (short time intervals)*

*corrupt model further with found solutions*

*perturbation 2: calibrate phases and amplitudes (long time intervals)*

*correct visibilities for all perturbation solutions*

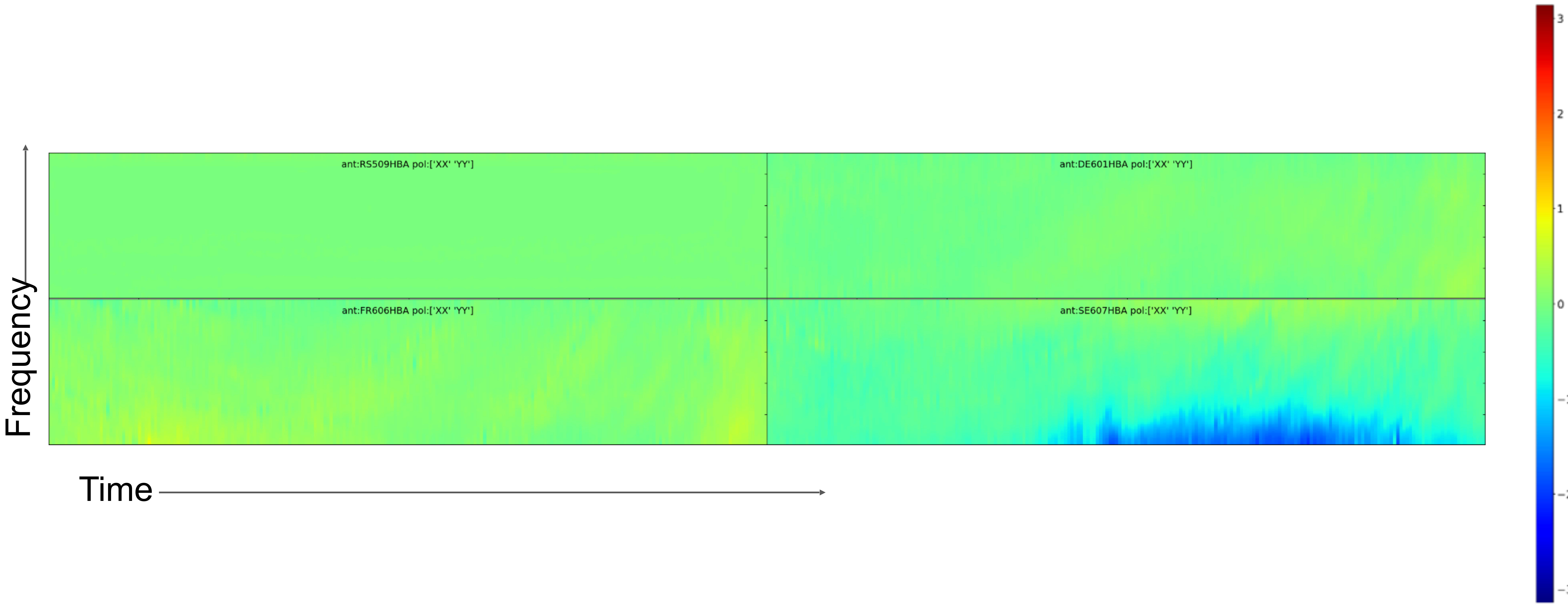
# Facetselfcal for a delay/infield calibrator

Run the script:

```
python facetselfcal.py --imsize=1024 -i selfcalimage_lbcs --
pixelscale=0.075 --robust=-0.5 --skymodelpointsource=1 --
uvmin=20000 --soltype-
list="['scalarphasediff','scalarphase','scalarcomplexgain']"
--soltypecycles-list="[0,0,2]" --solint-
list="['3min','32sec','40min']" --nchan-list="[1,1,1]" --
smoothnessconstraint-list="[10.0,1.5,10.0]" --docircular --
antennaconstraint-list="['alldutch',None,None]" --
forwidefield --stop=12 --avgfreqstep=4 --avgtimestep=4 --
phaseupstations='core' mydata.ms
```

# Facetselfcal

**scalarphasediff**: solve for the difference between RR and LL

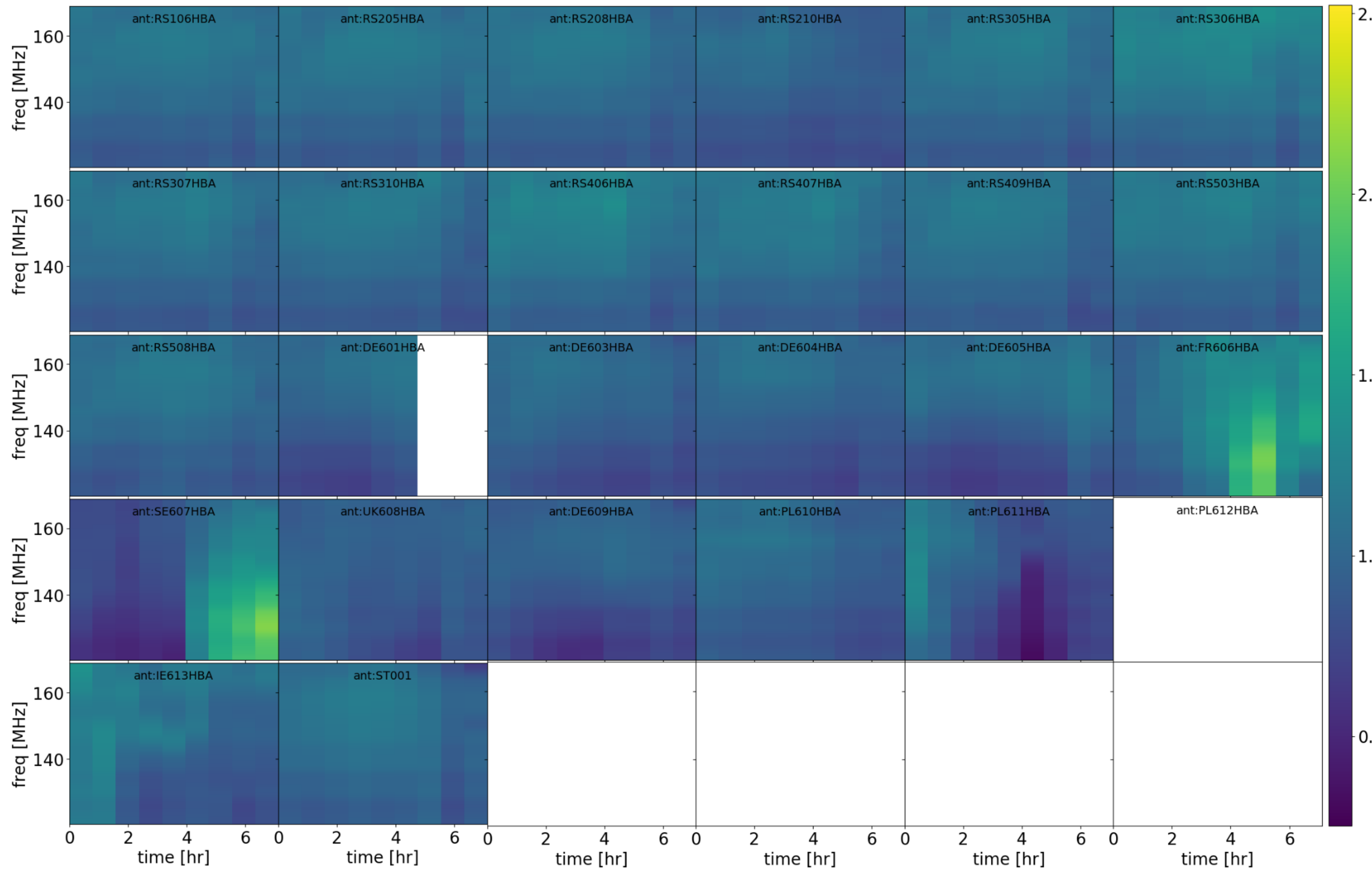


# Facetselfcal

**scalarphase**: combine the two polarizations and solve for the phases as a function of time

# Facetselfcal

**(scalar)complexgain**: slow solve for the complex gains



# Many tuning and advanced options

- Options to handle ILT data (e.g., automated starting models from VLASS, visibility stacking)
- Automated “box/phase up” selfcal, with solution interval and frequency smoothness tuning for HBA-Dutch (LoTSS extract), LBA-Dutch (incl. decameter band), HBA-ILT-target
- DD capability with wide(r)field mode (HBA, LBA, MeerKAT)

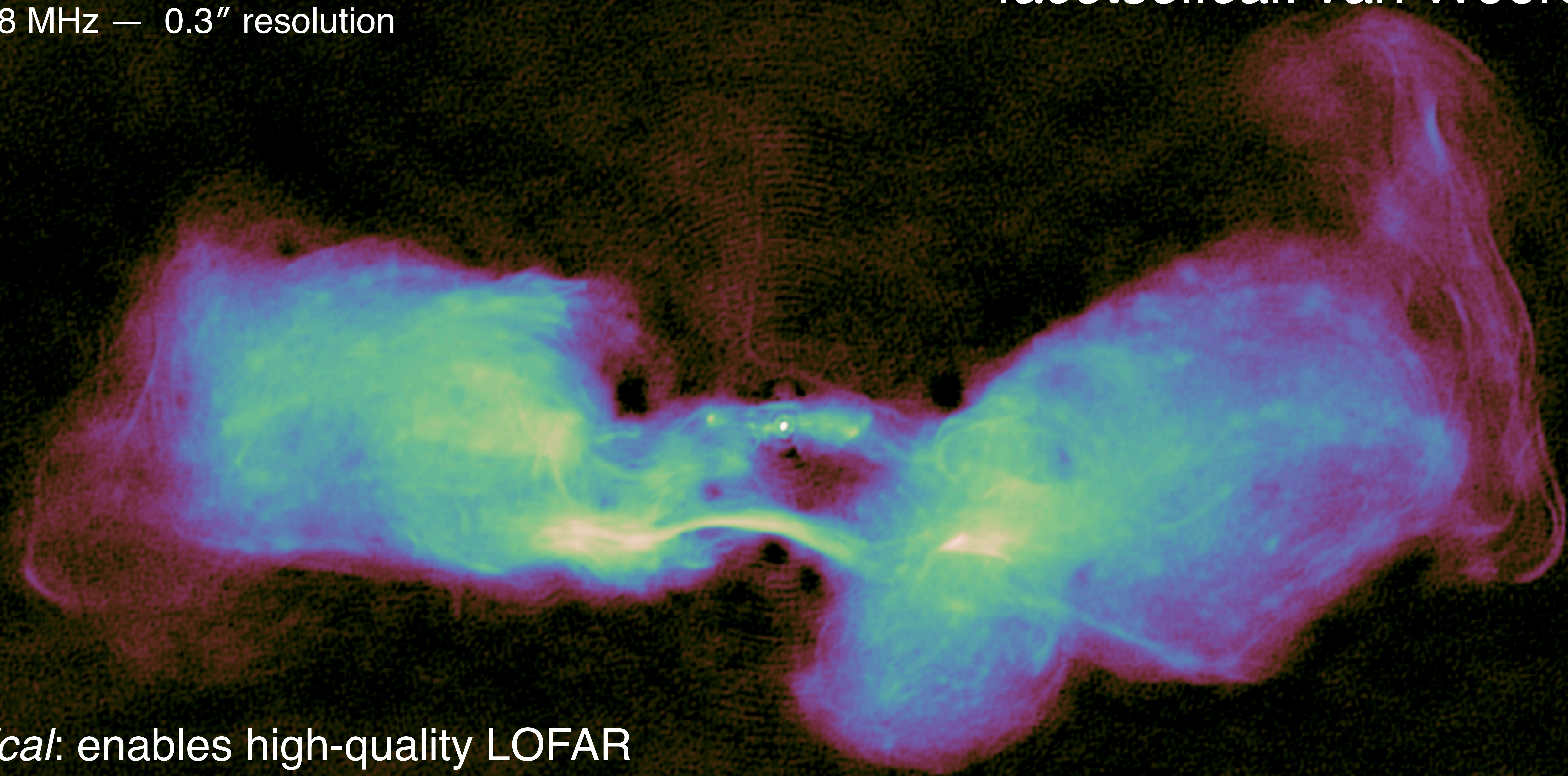
# facetselfcal -h

```
Self-Calibrate a facet from a LOFAR observation

positional arguments:
  ms                    msfile(s)

optional arguments:
  -h, --help            show this help message and exit
  --imager IMAGER       Imager to use WSClean or DDFACET. The default is
                        WSCLEAN.
  -i IMAGENAME, --imagename IMAGENAME
                        Prefix name for image. This is by default "image".
  --imsize IMSIZE       Image size, required if boxfile is not used. The
                        default is None.
  -n NITER, --niter NITER
                        Number of iterations. This is computed automatically
                        if None.
  --maskthreshold MASKTHRESHOLD
                        Mask noise thresholds used from image 1 to 10 made by
                        MakeMask.py. This is by default [5.0,4.5,4.5,4.5,4.0].
  --removenegativefrommodel REMOVENEGATIVEFROMMODEL
                        Remove negative clean components in model predict.
                        This is by default turned off at selfcalcycle 2. See
                        also option autoupdate-removenegativefrommodel.
  --autoupdate-removenegativefrommodel AUTOUPDATE_REMOVENEGATIVEFROMMODEL
                        Turn off removing negative clean components at
                        selfcalcycle 2 (for high dynamic range imaging it is
                        better to keep all clean components). The default is
                        True.
  --fitsmask FITSMASK   Fits mask for deconvolution (needs to match image
                        size). If this is not provided automasking is used.
  --robust ROBUST       Briggs robust parameter. The default is -0.5.
  --multiscale-start MULTISCALE_START
                        Start multiscale deconvolution at this selfcal cycle.
                        This is by default 1.
  --deepmultiscale      Do extra multiscale deconvolution on the residual.
  --uvminim UVMINIM     Inner uv-cut for imaging in lambda. The default is 80.
  --pixelscale PIXELSCALE, --pixelsize PIXELSCALE
                        Pixels size in arcsec. Typically, 3.0 for LBA and 1.5
                        for HBA (these are also the default values).
  --channelsout CHANNELSOUT
                        Number of channels out during imaging (see WSClean
                        documentation). This is by default 6.
  --multiscale          Use multiscale deconvolution (see WSClean
                        documentation).
  --multiscalescalebias MULTISCALESCALEBIAS
                        Multiscalescale bias scale parameter for WSClean (see
                        WSClean documentation). This is by default 0.8.
  --usewgridder USEWGRIDDER
                        Use wgridder from WSClean, mainly useful for very
                        large images. This is by default True.
  --paralleldeconvolution PARALLELDECONVOLUTION
                        Parallel-deconvolution size for WSCLean (see WSClean
                        documentation). This is by default 0 (no parallel
                        deconvolution). Suggested value for very large images
                        is about 2000.
  --parallelgridding PARALLELGRIDDING
                        Parallel-gridding for WSClean (see WSClean
                        documentation). This is by default 1.
  --deconvolutionchannels DECONVOLUTIONCHANNELS
                        Deconvolution channels value for WSClean (see WSClean
                        documentation). This is by default 0 (means
                        deconvolution-channels equals channels-out).
  --idg                 Use the Image Domain gridder (see WSClean
                        documentation).
  --fitspectralpol FITSPECTRALPOL
                        Use fit-spectral-pol in WSClean (see WSClean
                        documentation). The default is True.
  --fitspectralpolorder FITSPECTRALPOLORDER
                        fit-spectral-pol order for WSClean (see WSClean
                        documentation). The default is 3.
  --gapchanneldivision  Use the -gap-channel-division option in wsclean
                        imaging and predicts (default is not to use it)
  --taperinnertukey TAPERINNERTUKEY
                        Value for taper-inner-tukey in WSClean (see WSClean
                        documentation), useful to supress negative bowls when
                        using --uvminim. Typically values between 1.5 and 4.0
                        give good results. The default is None.
  --wscleanskymodel WSCLEANSKYMODEL
                        WSclean basename for model images (for a WSClean
                        predict). The default is None.
  --avgfreqstep AVGFREQSTEP
                        Extra DP3 frequency averaging to speed up a solve.
                        This is done before any other correction and could be
                        useful for long baseline infield calibrators. Allowed
                        are integer values or for example '195.3125kHz';
                        options for units: 'Hz', 'kHz', or 'MHz'. The default
                        is None.
  --avgtimestep AVGTIMESTEP
                        Extra DP3 time averaging to speed up a solve. This is
                        done before any other correction and could be useful
                        for long baseline infield calibrators. Allowed are
                        integer values or for example '16.1s'; options for
                        units: 's' or 'sec'. The default is None.
  --msinnchan MSINNCHAN
                        Before averaging, only take this number of input
                        channels. The default is None.
  --msinntimes MSINNTIMES
                        DP3 msin.ntimes setting. This is mainly used for
                        testing purposes. The default is None.
  --autofrequencyaverage-calspeedup
                        Try extra averaging during some selfcalcycles to speed
                        up calibration.
  --autofrequencyaverage
                        Try frequency averaging if it does not result in
                        bandwidth smearing
  --phaseupstations PHASEUPSTATIONS
                        Phase up to a superstation. Possible input: 'core' or
                        'superterp'. The default is None.
  --phaseshiftbox PHASESHIFTBOX
                        DS9 region file to shift the phasecenter to. This is
                        by default None.
  --weightspectrum-clipvalue WEIGHTSPECTRUM_CLIPVALUE
                        Extra option to clip WEIGHT_SPECTRUM values above the
                        provided number. Use with care and test first manually
                        to see what is a fitting value. The default is None.
  -u UVMIN, --uvmin UVMIN
                        Inner uv-cut for calibration in lambda. The default is
                        80 for LBA and 350 for HBA.
  --uvminscalarphasediff UVMINSCALARPHASEDIFF
                        Inner uv-cut for scalarphasediff calibration in
                        lambda. The default is equal to input for --uvmin.
  --update-uvmin        Update uvmin automatically for the Dutch array.
  --update-multiscale   Switch to multiscale automatically if large islands of
                        emission are present.
  --soltype-list SOLTYPE_LIST
                        List with solution types. Possible input:
                        'complexgain', 'scalarcomplexgain', 'scalaramplitude',
                        'amplitudeonly', 'phaseonly', 'fulljones', 'rotation',
                        'rotation+diagonal', 'tec', 'tecandphase',
                        'scalarphase', 'scalarphasediff', 'scalarphasediffFR',
                        'phaseonly_phmin', 'rotation_phmin', 'tec_phmin',
                        'tecandphase_phmin', 'scalarphase_phmin',
                        'scalarphase_slope', 'phaseonly_slope'. The default is
                        [tecandphase,tecandphase,scalarcomplexgain].
  --solint-list SOLINT_LIST
                        Solution interval corresponding to solution types (in
                        same order as soltype-list input). The default is
                        [1,1,120].
  --nchan-list NCHAN_LIST
                        Number of channels corresponding to solution types (in
                        same order as soltype-list input). The default is
                        [1,1,10].
  --smoothnessconstraint-list SMOOTHNESSCONSTRAINT_LIST
                        List with frequency smoothness values (in same order
                        as soltype-list input). The default is [0.,0.,5.].
  --smoothnessreffrequency-list SMOOTHNESSREFFREQUENCY_LIST
                        List with optional reference frequencies (in MHz) for
                        the smoothness constraint (in same order as soltype-
                        list input). When unequal to 0, the size of the
                        smoothing kernel will vary over frequency by a factor
                        of smoothnessreffrequency*(frequency^smoothnessspectra
                        lexponent). The default is [0.,0.,0.].
  --smoothnessspectralexponent-list SMOOTHNESSSPECTRALEXPONENT_LIST
                        If smoothnessreffrequency is not equal to zero then
                        this parameter determines the frequency scaling law.
                        It is typically useful to take -2 for scalarphasediff,
                        otherwise -1 (1/nu). The default is [-1.,-1.,-1.].
  --smoothnessrefdistance-list SMOOTHNESSREFDISTANCE_LIST
                        If smoothnessrefdistance is not equal to zero then
                        this parameter determines the freqeuency smoothness
                        reference distance in units of km, with the smoothness
                        scaling with distance. See DP3 documentation. The
                        default is [0.,0.,0.].
  --antennaconstraint-list ANTENNACONSTRAINT_LIST
                        List with constraints on the antennas used (in same
                        order as soltype-list input). Possible input:
                        'superterp', 'coreandfirstremotes', 'core', 'remote',
                        'all', 'international', 'alldutch', 'core-remote',
                        'coreandallbutmostdistantremotes',
                        'alldutchbutnoST001'. The default is [None,None,None].
  --resetsols-list RESETSOLS_LIST
                        Values of these stations will be rest to 0.0 (phases),
                        or 1.0 (amplitudes), default None, possible settings
                        are the same as for antennaconstraint-list (alldutch,
                        core, etc)). The default is [None,None,None].
  --soltypecycles-list SOLTYPECYCLES_LIST
                        Selfcalcycle where step from soltype-list starts. The
                        default is [0,999,3].
  --BLsmooth            Employ BLsmooth for low S/N data.
  --dejumpFR            Dejump Faraday solutions when using scalarphasediffFR.
  --usemodeldataforsolints
                        Determine solints from MODEL_DATA.
  --preapplyH5-list PREAPPLYH5_LIST
                        List of H5 files to preapply (one for each MS). The
                        default is [None].
  --iontimefactor IONTIMEFACTOR
                        BLsmooth ionfactor. The default is 0.01. Larger is
                        more smoothing (see BLsmooth documentation).
  --ionfreqfactor IONFREQFACTOR
                        BLsmooth tecfactor. The default is 1.0. Larger is more
                        smoothing (see BLsmooth documentation).
  --blscalefactor BLSCALEFACTOR
                        BLsmooth blscalefactor. The default is 1.0 (see
                        BLsmooth documentation).
  -b BOXFILE, --boxfile BOXFILE
                        DS9 box file. You need to provide a boxfile to use
                        --startfromtgss. The default is None.
  --skymodel SKYMODEL   Skymodel for first selfcalcycle. The default is None.
  --skymodelsource SKYMODELSOURCE
                        Source name in skymodel. The default is None (means
                        the skymodel only contains one source/patch).
  --skymodelpointsource SKYMODELPOINTSOURCE
                        If set, start from a point source in the phase center
                        with the flux density given by this parameter. The
                        default is None (means do not use this option).
  --predictskywithbeam  Predict the skymodel with the beam array factor.
  --startfromtgss       Start from TGSS skymodel for positions (boxfile
                        required).
  --startfromvlass      Start from VLASS skymodel for ILT phase-up core data
                        (not yet implemented).
  --tgssfitsimage TGSSFITSIMAGE
                        Start TGSS fits image for model (if not provided use
                        SkyView). The default is None.
  --beamcor BEAMCOR     Correct the visibilities for beam in the phase center,
                        options: yes, no, auto (default is auto, auto means
                        beam is taken out in the curent phase center,
                        tolerance for that is 10 arcsec)
  --losotobeamcor-beamlib LOSOTOBEAMCOR_BEAMLIB
                        Beam library to use when not using DP3 for the beam
                        correction. Possible input: 'stationreponse',
                        'lofarbeam' (identical and deprecated). The default is
                        'stationresponse'.
  --docircular          Convert linear to circular correlations.
  --dolinear            Convert circular to linear correlations.
  --forwidefield        Keep solutions such that they can be used for
                        widefield imaging/screens.
  --doflagging DOFLAGGING
                        Flag on complexgain solutions via rms outlier
                        detection (True/False, default=True). The default is
                        True (will be set to False if --forwidefield is set).
  --clipsolutions       Flag amplitude solutions above --clipsolhigh and below
                        --clipsollow (will be set to False if --forwidefield
                        is set).
  --clipsolhigh CLIPSOLHIGH
                        Flag amplitude solutions above this value, only done
                        if --clipsolutions is set.
  --clipsollow CLIPSOLLOW
                        Flag amplitude solutions below this value, only done
                        if --clipsolutions is set.
  --dysco DYSCO         Use Dysco compression. The default is True.
  --restoreflags        Restore flagging column after each selfcal cycle, only
                        relevant if --doflagging=True.
  --remove-flagged-from-startend
                        Remove flagged time slots at the start and end of an
                        observations. Do not use if you want to combine DD
                        solutions later for widefield imaging.
  --flagslowamprms FLAGSLOWAMPRMS
                        RMS outlier value to flag on slow amplitudes. The
                        default is 7.0.
  --flagslowphaserms FLAGSLOWPHASERMS
                        RMS outlier value to flag on slow phases. The default
                        7.0.
  --doflagslowphases DOFLAGSLOWPHASES
                        If solution flagging is done also flag outliers phases
                        in the slow phase solutions. The default is True.
  --useaoflagger        Run AOflagger on input data.
  --useaoflaggerbeforeavg USEAOFLAGGERBEFOREAVG
                        Flag with AOflagger before (True) or after averaging
                        (False). The default is True.
  --normamps NORMAMPS   Normalize global amplitudes to 1.0. The default is
                        True (False if fulljones is used).
  --normampsskymodel NORMAMPSSKYMODEL
                        Normalize global amplitudes to 1.0 when solving
                        against an external skymodel. The default is False
                        (turned off if fulljones is used).
  --resetweights        If you want to ignore weight_spectrum_solve.
  --start START         Start selfcal cycle at this iteration number. The
                        default is 0.
  --stop STOP           Stop selfcal cycle at this iteration number. The
                        default is 10.
  --stopafterskysolve   Stop calibration after solving against external
                        skymodel.
  --noarchive           Do not archive the data.
  --skipbackup          Leave the original MS intact and work and always work
                        on a DP3 copied dataset.
  --helperscriptspath HELPERSCRIPTSPATH
                        Path to file location pulled from
                        https://github.com/rvweeren/lofar_facet_selfcal.
  --helperscriptspathh5merge HELPERSCRIPTSPATHH5MERGE
                        Path to file location pulled from
                        https://github.com/jurjen93/lofar_helpers.
  --auto                Trigger fully automated processing (HBA only for now).
  --delaycal            Trigger settings suitable for ILT delay calibration,
                        HBA-ILT only - still under construction.
  --targetcalILT TARGETCALILT
                        Type of automated target calibration for HBA
                        international baseline data when --auto is used.
                        Options are: 'tec', 'tecandphase', 'scalarphase'. The
                        default is 'tec'.
  --makeimage-ILTlowres-HBA
                        Make 1.2 arcsec tapered image as quality check of ILT
                        1 arcsec imaging.
  --makeimage-fullpol   Under development, make Stokes IQUV version for
                        quality checking.
  --blsmooth_chunking_size BLSMOOTH_CHUNKING_SIZE
                        Chunking size for blsmooth. Larger values are slower
                        but save on memory, lower values are faster. The
                        default is 8.
  --tecfactorsolint TECFACTORSOLINT
                        Experts only.
  --gainfactorsolint GAINFACTORSOLINT
                        Experts only.
  --phasefactorsolint PHASEFACTORSOLINT
                        Experts only.
```

# Summary

Roland Timmerman
3C338: 120-168 MHz —  0.3″ resolution

*facetselfcal*: van Weeren+ (2021)



- *facetselfcal*: enables high-quality LOFAR imaging (HBA, LBA, long baselies, MeerKAT)
- *facetselfcal:* tackle calibration challenges and develop new ideas