Tutorial: Calibration

E. Orru’

working dir

ssh -YX lodsXX@portal.lofar.eu
ssh -YX lhd002
screen
Inside the screen:

```
use Slurm; srun -A tutorials --reservation=tutorials_287 -N1 -w lof00X -u bash -i
```

It is good if it says: bash: no job control in this shell

Detach: Ctrl-a, then d

Now in the original terminal:
ssh -YX lhd002
ssh -YX lof00X

cd /data/scratch/

mkdir lods##

mkdir T3

cd T3

MS

ONE COMMAND PER COUPLE!!!!

L456102_calibrator

cp -rf ../DATASCHOOL2016_T2/L456102_calibrator/ .

mkdir L456106_target/

cd L456106_target/

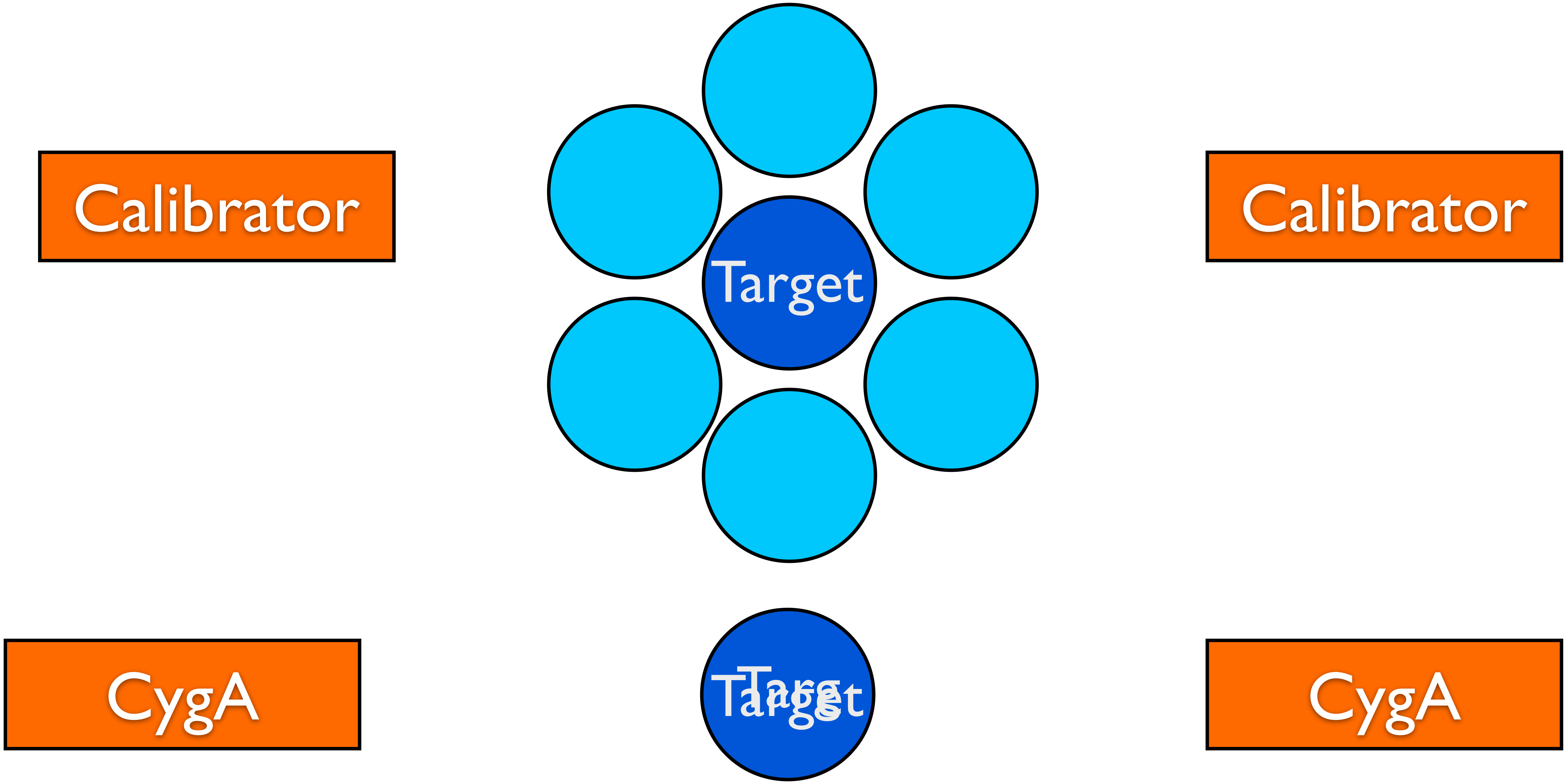
ls /data/scratch/DATASCHOOL2016_T2/L456106_target/

PICK 5SBs

cp -r /data/scratch/DATASCHOOL2016_T2/L456106_target/L456106_SB*_uv.dppp.MS.flg .

cp -r /data/scratch/DATASCHOOL2016_T2/A-Team_lowres.skymodel .

Observation set up



Calibration

CygA

model

we know how that source looks like at that frequency. Both the flux and morphology can be parametrized.

predict

we use the parametrized model to predict how our telescope would see the calibrator source.

solve Gains

calculate the *gains* in order to minimize the difference between the observed data and the model

correct

the gains are multiplied to the data in order to correct the fluxes and the phases.

*gains*do you remember the measurement equation?

$$\|\mathbf{V}_{pq} - \mathbf{G}_p \mathbf{M}_{pq} \mathbf{G}_p^H\|$$

$$G_p = \begin{bmatrix} A_{xx}e^{\phi_{xx}} & 0 \\ 0 & A_{yy}e^{\phi_{yy}} \end{bmatrix}$$

Calibrator model

open the calibrator model with your favorite editor

How does it look?

```
# (Name, Type, Patch, Ra, Dec, I, Q, U, V, ReferenceFrequency='74e6', SpectralIndex='[]', MajorAxis, MinorAxis, Orientation) = format

# The above line defines the field order and is required.

# models for the A-Team members

, , CygA, 19:59:26, +40.44.00
CygA_4_2, POINT,    CygA, 19:59:30.433, +40.43.56.221, 4.827e+03, 0.0, 0.0, 0.0, 7.38000e+07, [-0.8], 7.63889e-03, 6.94444e-03, 1.00637e+02
CygA_4_1, GAUSSIAN, CygA, 19:59:28.476, +40.44.01.804, 4.165e+03, 0.0, 0.0, 0.0, 7.38000e+07, [-0.8], 1.20516e-02, 5.34011e-03, 1.03230e+02
CygA_4_3, POINT,    CygA, 19:59:22.168, +40.44.28.591, 3.896e+03, 0.0, 0.0, 0.0, 7.38000e+07, [-0.8], 7.63889e-03, 6.94444e-03, 9.21061e+01
CygA_4_4, POINT,    CygA, 19:59:24.413, +40.44.18.617, 2.798e+03, 0.0, 0.0, 0.0, 7.38000e+07, [-0.8], 7.63889e-03, 6.94444e-03, 6.74922e+01
CygA_4_5, POINT,    CygA, 19:59:25.555, +40.43.52.813, 1.560e+02, 0.0, 0.0, 0.0, 7.38000e+07, [-0.8], 7.63889e-03, 6.94444e-03, 0.00000e+00
```

Run the comand:

```
use Lofar

makesourcedb in=A-Team_lowres.skymodel out=A-Team_lowres.sourcedb format="<"
```

predict

we use the parametrized model to predict how our telescope would see the calibrator source.

- In /data/scratch/lods##/T2/`L456102_calibrator/` using your favorite editor open a file called predict_model.parset where you copy and paste the content of the yellow window

```
msin =
msin.datacolumn = DATA
msin.baseline = [CR]S*&
msout = .
msout.datacolumn = MODEL_DATA
numthreads = 5

steps = [predict]

predict.type=predict
predict.sourcedb=A-Team_lowres.sourcedb
predict.sources=CygA
predict.usebeammodel=True
```

- Run the command for one SB

```
NDPPP predict_model.parset
msin=L456102_SB000_uv.dppp.MS.flg msout=
L456102_SB000_uv.dppp.MS.flg
```

- Now we run the same command for all the SBs of the calibrator

```
cd L456102_calibrator
cp A-Team_lowres.sourcedb

for i in *MS.flg; do NDPPP predict.parset
msin=$i msout=$i; done
```

calibrate: gaincal

In /data/scratch/lods##/T2/L456102_calibrator/, using your favorite editor open a file called gaincal.parset where you copy and paste the content of the yellow window

```
msin=
msout=.
msin.datacolumn = DATA
msin.baseline = [CR]S*&
msout.datacolumn = CORRECTED_DATA
numthreads = 5
steps=[gaincal]

#gaincal.sourcedb=A-Team_lowres.sourcedb
#gaincal.sources = CygA
#gaincal.usebeammodel=true
gaincal.usemodelcolumn=true
gaincal.parmdb=
gaincal.type=gaincal
gaincal.caltype=diagonal
```

- Run the command for one SB

```
NDPPP gaincal.parset
msin=L456102_SB000_uv.dppp.MS.flg
gaincal.parmdb=L456102_SB000_uv.dppp.MS.flg/
instrument
msout=L456102_SB000_uv.dppp.MS.flg
```

- Now we run the same command for all the SBs of the calibrator

cd L456102_calibrator

```
for i in *MS.flg; do NDPPP
gaincal.parset msin=$i
gaincal.parmdb=$i/instrument msout=
$i; done
```

Inspect solutions: LoSoTo

- `/data/scratch/T2/L456102_calibrator`

- Open you favorite editor and copy this bash script to create the globaldb and copy all the instrument tables naming them instrument1, instrument2,...

- In globaldb copy the ANTENNA, FIELD, sky

```
#!/bin/bash
# To copy the instrument tables in a globaldb

mkdir globaldb
i=0
for ms in `ls -d *MS.flg`; do
    echo "Copying ${ms}/instrument"
    cd $ms
    # copy other tables
    if [ $i == 0 ]; then
        cp -r ANTENNA ../globaldb
        cp -r FIELD ../globaldb
        cp -r sky ../globaldb
    fi
    cp -r instrument ../globaldb/instrument-$i
    cd ..
    i=$((i + 1))
done
```

Datafile standard in HDF5 (H5parm)

Size: down to 1/2 of parmdb
Read speed: 1/2 to 1/5 of parmdb
Easy access with python libs

H5parm

__sol000__
amplitude
phase
phase_smooth

__sol001__
phase
clock
tec

time (100)
freq (50)
ant (28)
pol (2)

data (100x50x28x2)
weights (100x50x28x2)

LoSoTo

Lofar Solutions Tool

Library: read and write H5parm

Program: run a sequence of operations

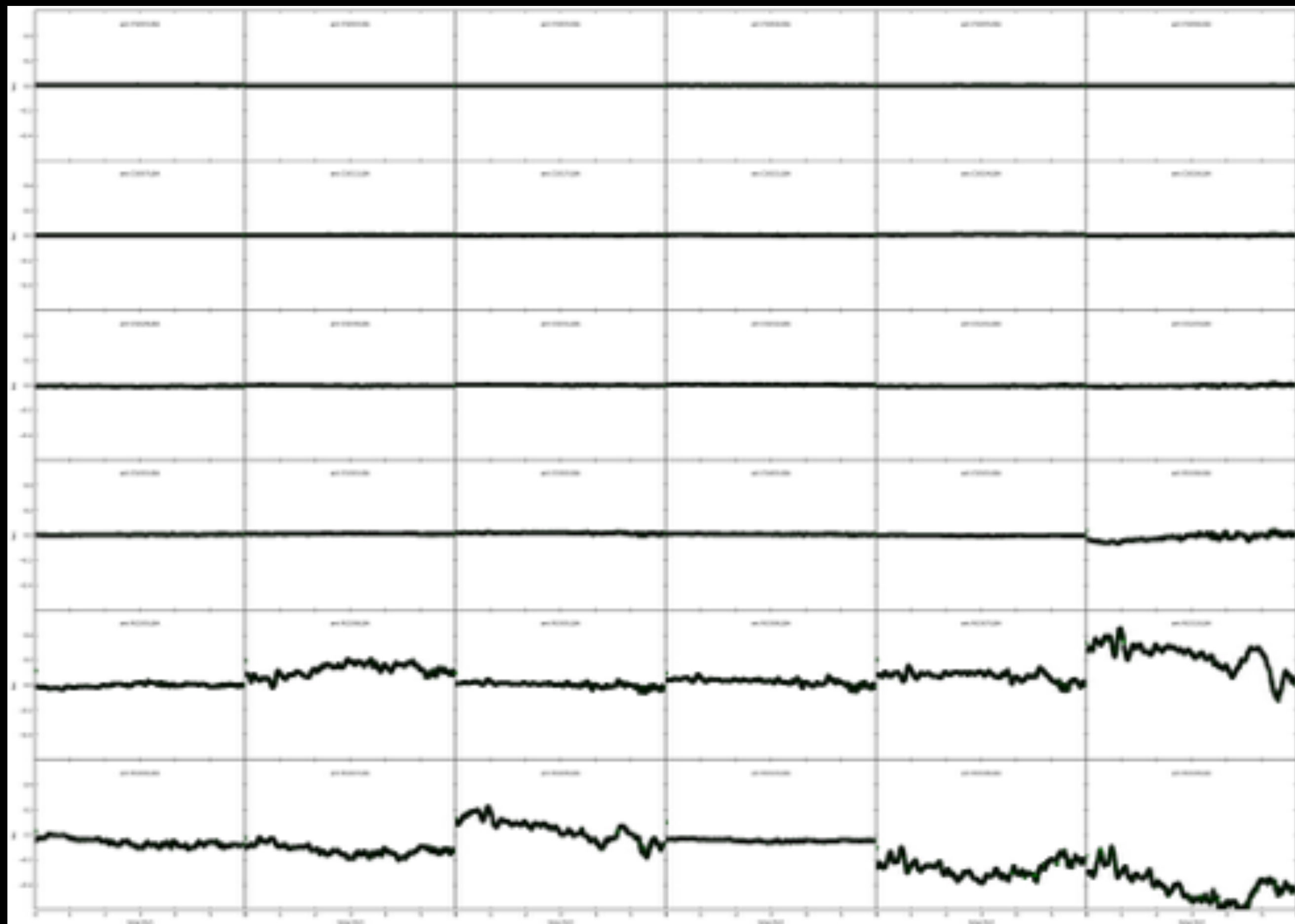
Tools: to convert parmdb from/to H5parm

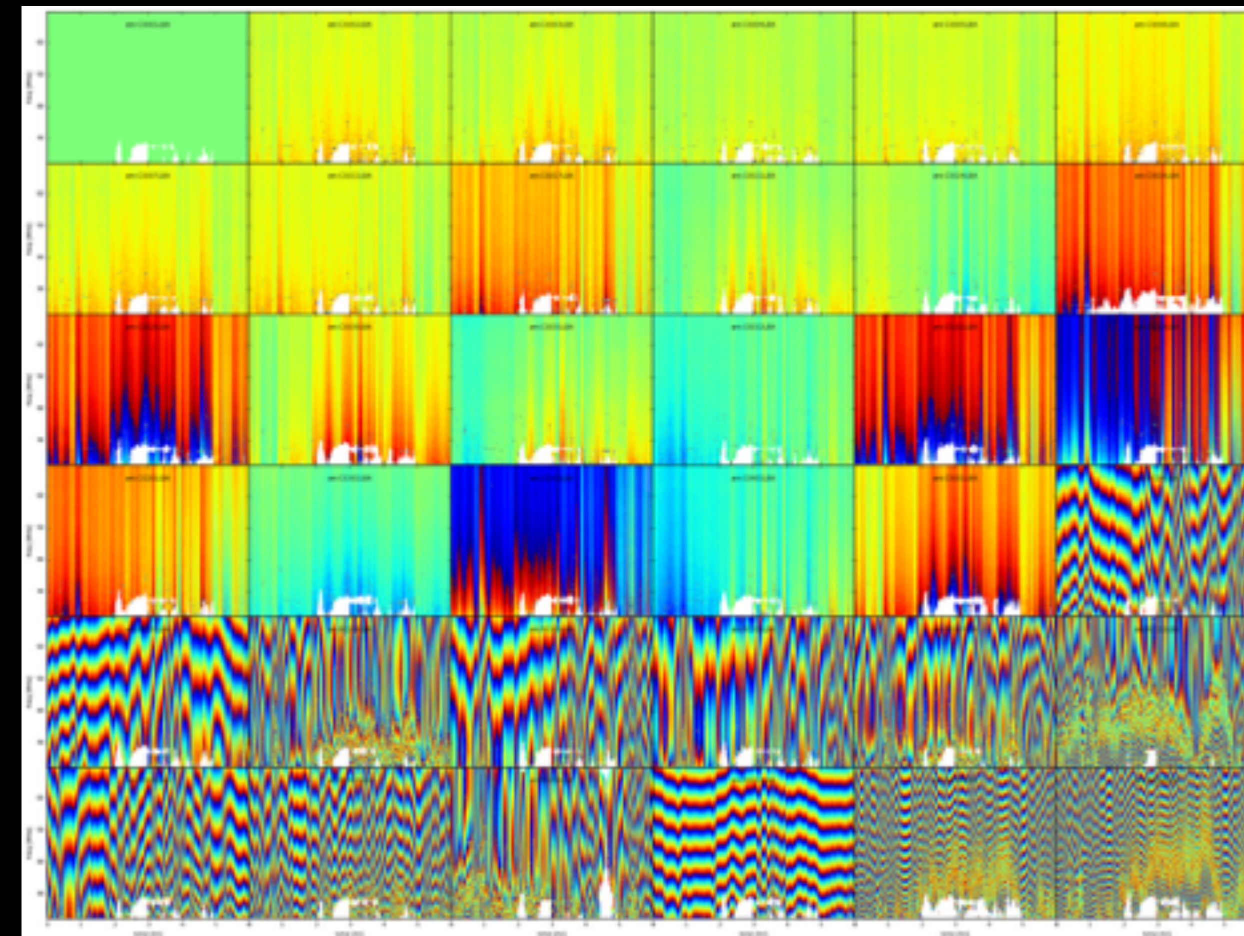
F. de Gasperin
D. Rafferty

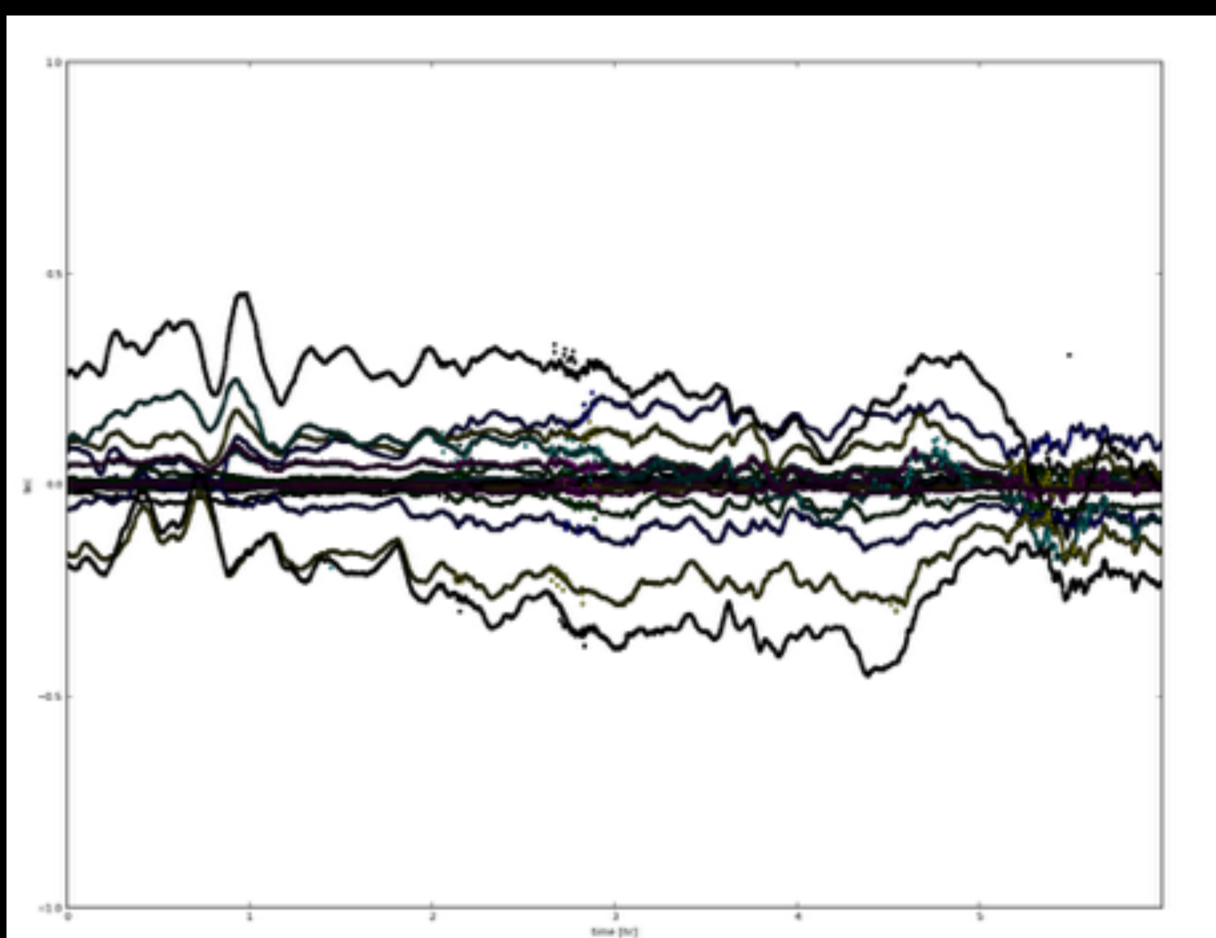
Operations

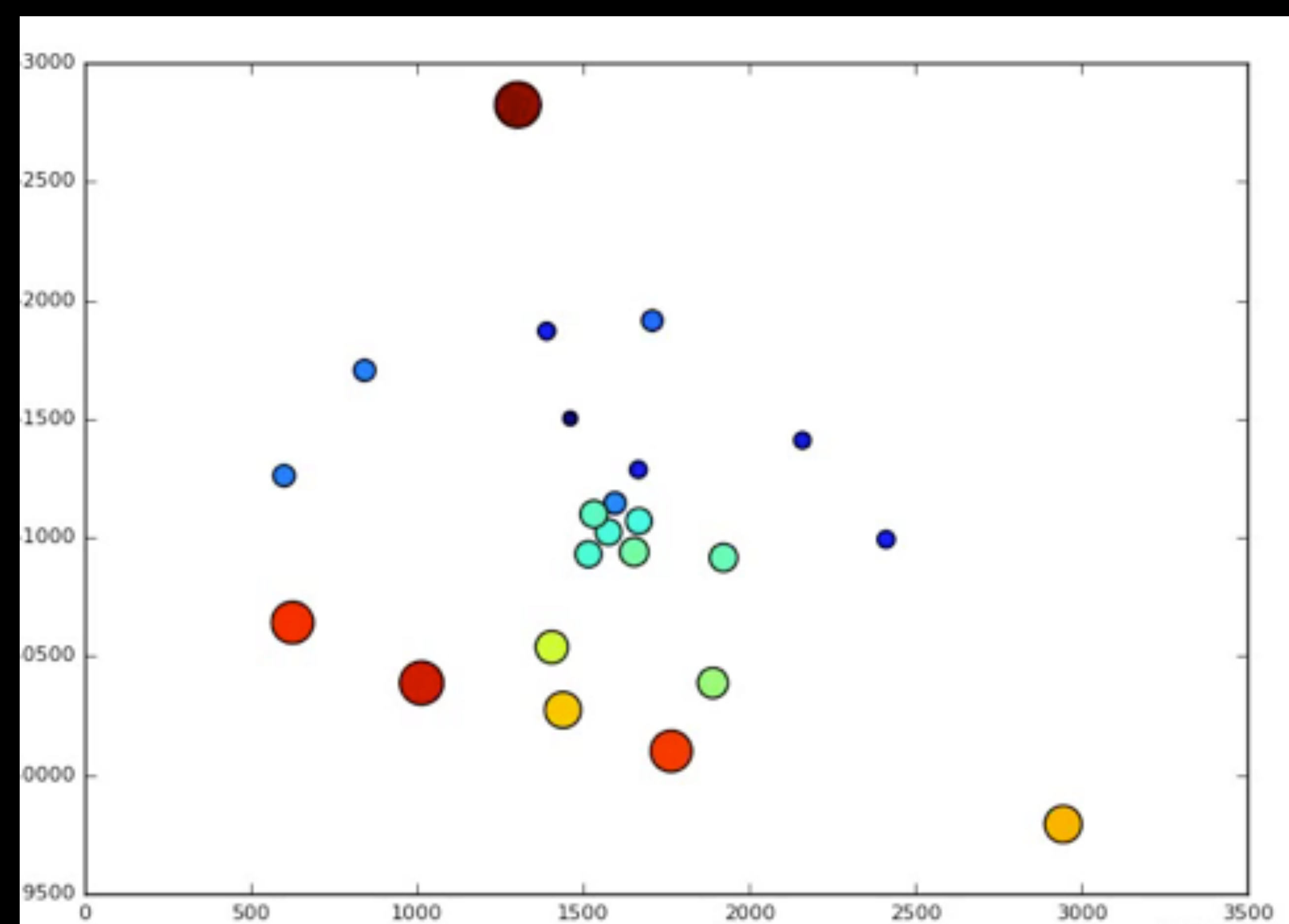
ABS: absolute value
CLIP: flag above/below a certain sigma
CLOCKTEC: clock/tec separation
DUPLICATE: duplicate a solution table
FLAG: remove trends and flag outliers
FLAGEXTEND: extend flags along any (even multiple) axis
INTERP: interpolate solutions along (even multiple) axis
NORM: normalize solutions
PLOT: 1D, 2D, movie
RESIDUALS: subtract a table from another
REWEIGHT: used to flag antennas, channels…
RESET: to default values
SMOOTH: smooth along (even multiple) axis
TECFIT: fit TEC values to phase solutions
TECSCREEN: fit TEC screens to TEC values

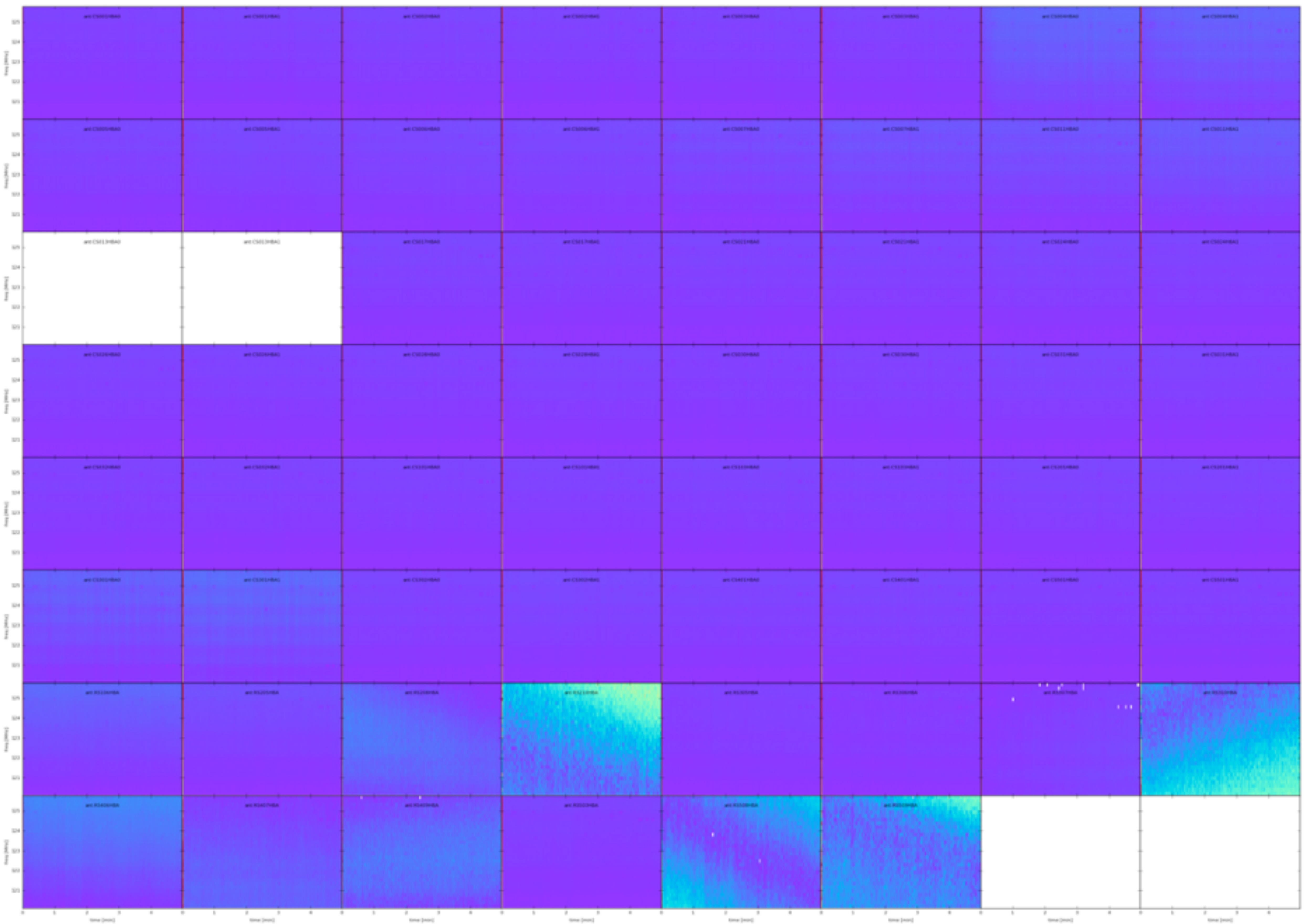
https://github.com/revoltek/losoto

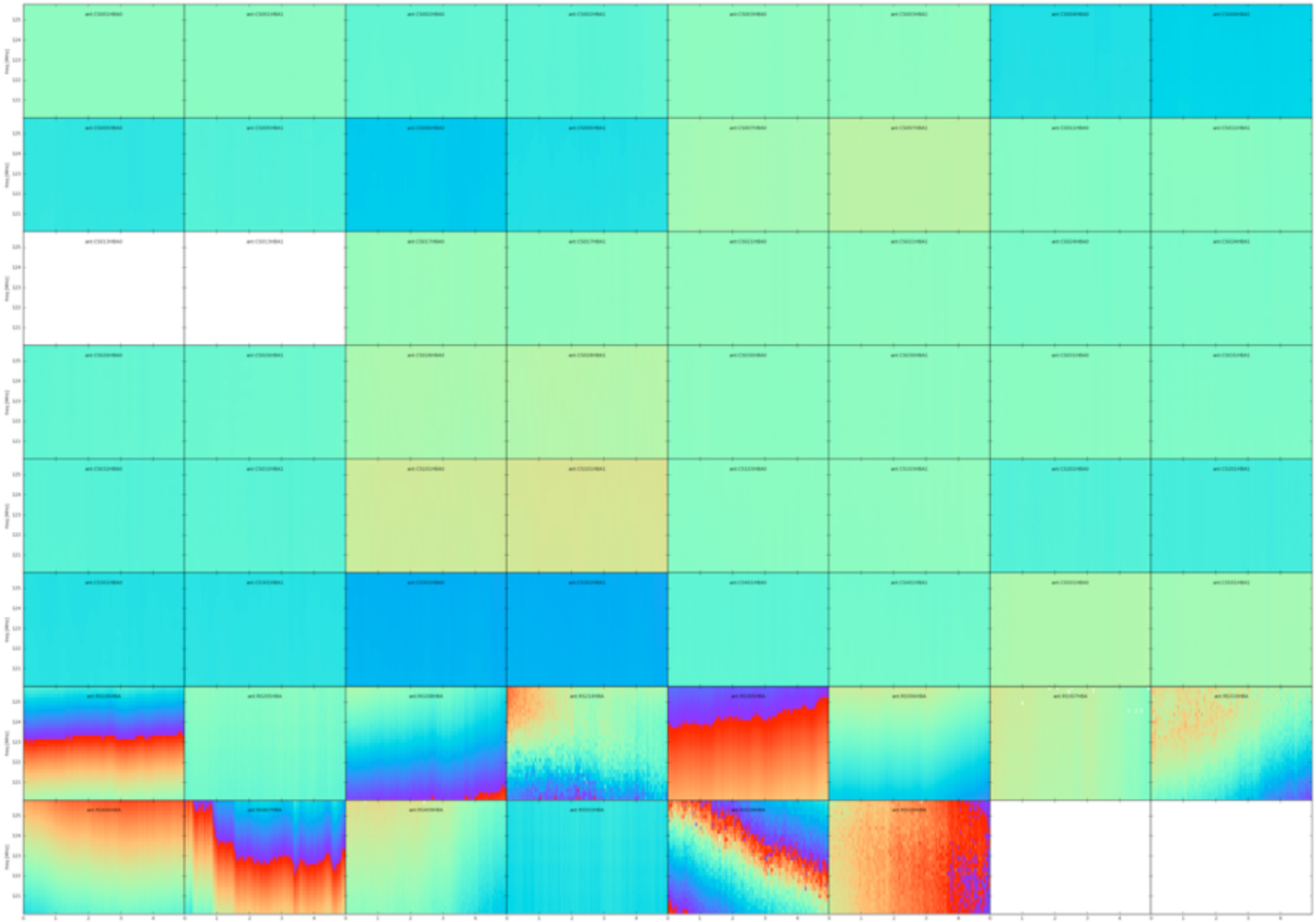


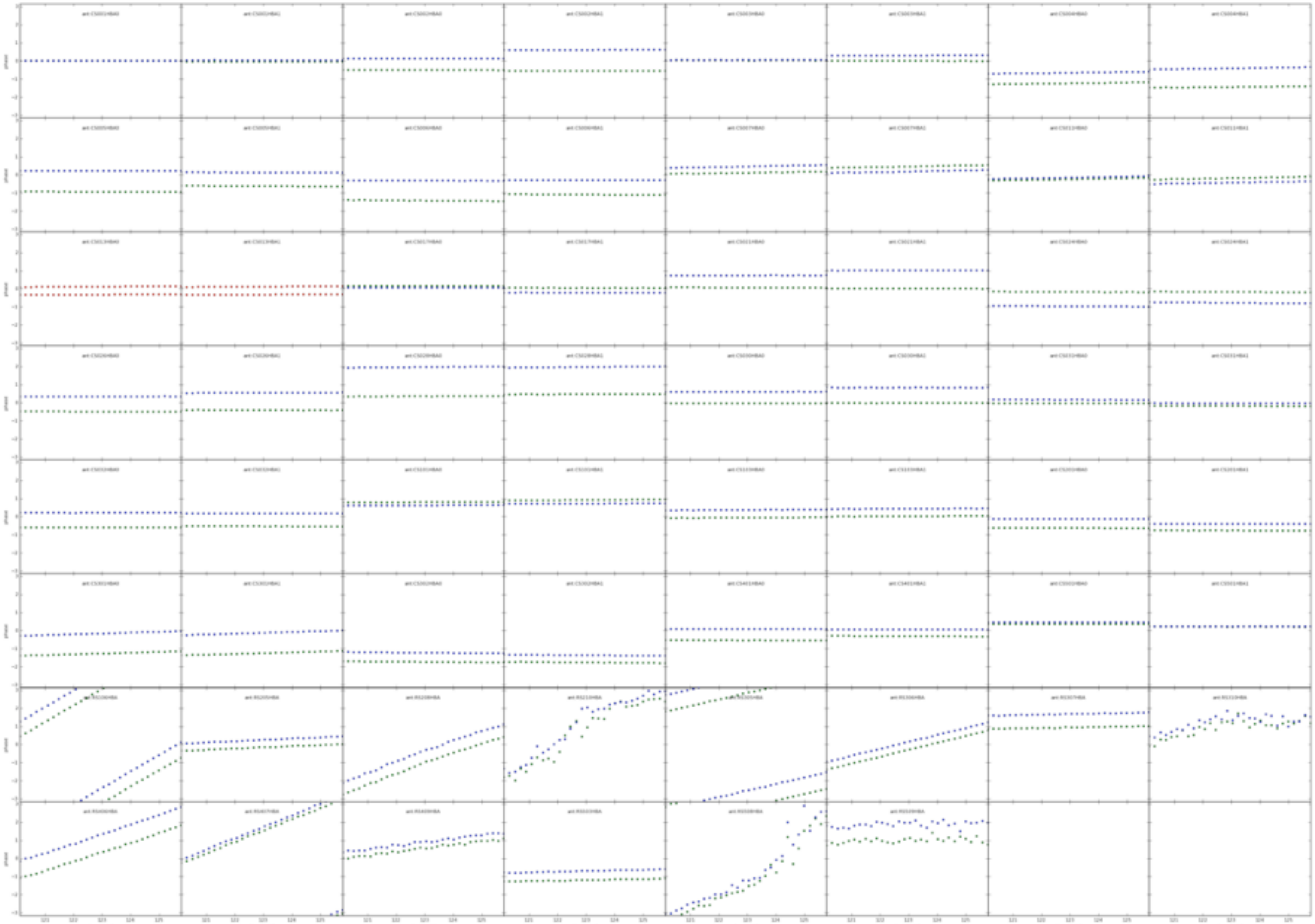





3000
2500
2000
1500
1000
0500
0000
9500
0
500
1000
1500
2000
2500
3000
3500
+3.825e6







Inspect solutions: LoSoTo

use Losoto

```
H5parm_importer.py -v cal.h5 globaldb

cp /data/scratch/DATASCHOOL2016_T2/
losoto.parset .

losoto -v cal.h5 losoto.parset

H5parm_exporter.py -v cal.h5 globaldb
```

```
#losoto parset
LoSoTo.Steps = [plotP1, plotP2, plotP3, plotA1,
plotA2, plotA3, flag, flagextend, merge]
```

```
#!/bin/bash
# copy back the instrument tables from a globaldb
# to be run after H5parm_exporter.py

i=0
for ms in `ls -d *MS.flg`; do
    echo "Copying back ${ms}/instrument"
    rm -r ${ms}/instrument
    cp -r globaldb/sol000_instrument-$i ${ms}/instrument
    i=$((i + 1))
done
```

Apply calibrator solutions to the target: Applycal

- The goal is to transfer the solutions found for calibrator to the target.

- We need to make or solution time independent because the calibrator was observed at a different time with respect to the target. This can be done with parmexportcal

```
parmexportcal in=L456102_SB000_uv.dppp.MS.flg/
instrument/ out=L456102_SB000_uv.dppp.MS.flg/
instrument_tind
```

- In /data/scratch/lods##/T2, using your favorite editor open a file called applycal.parset where you copy and paste the content of the yellow window

- Run the command for one SB

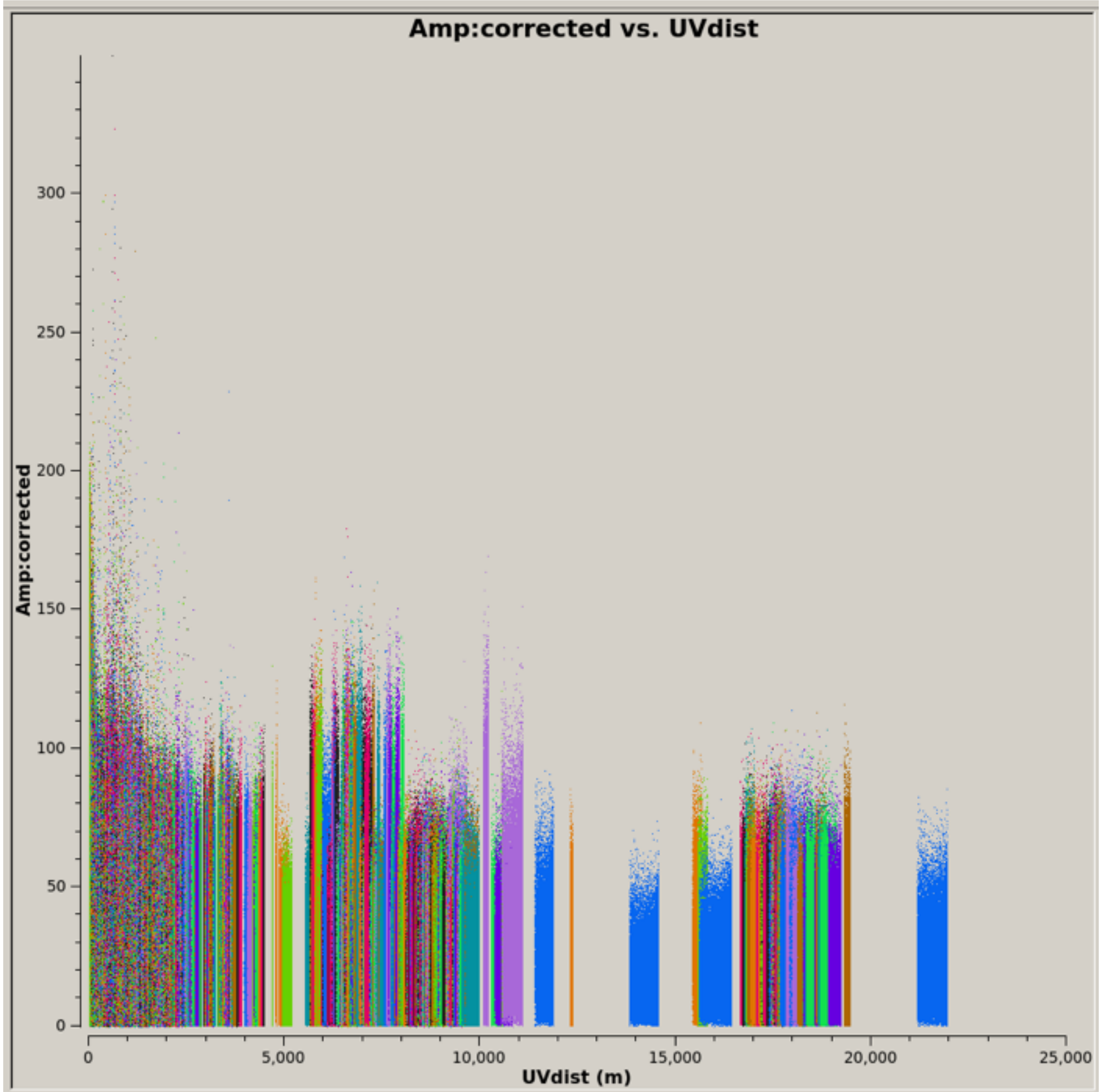
```
NDPPP applycal.parset msin=L456106_target/
L456106_SB000_uv.dppp.MS.flg
applycal.parmdb=L456102_calibrator/
L456102_SB000_uv.dppp.MS.flg/instrument_tind/ msout=
L456106_target/L456106_SB000_uv.dppp.MS.flg
```

```
msin =
msin.datacolumn = DATA
msin.baseline = [CR]S*&
msout = .
msout.datacolumn =
CORRECTED_DATA
numthreads = 5

steps = [applycal, applybeam]

applycal.type = applycal
applycal.correction = gain
applycal.parmdb =

applybeam.type = applybeam
applybeam.invert = True
```



Amp:corrected vs. UVdist
Amp:corrected
UVdist (m)
0
50
100
150
200
250
300
5,000
10,000
15,000
20,000
25,000

Apply calibrator solutions to the target: Applycal

- Now we run the same command for all the SBs of the calibrator

```
cd L456102_calibrator

for i in *MS.flg; do parmexportcal in=$i/instrument out=$i/instrument_tind; done
```

- Run the command for all SBs
- we write a small python script to do that:

```
import os
for i in range(0,29,1):
    if i <10:
        calib_instrument = 'L456102_calibrator/L456102_SB00'+str(i)+'_uv.dppp.MS.flg/instrument_tind'
        targetMS = 'L456106_target/L456106_SB00'+str(i)+'_uv.dppp.MS.flg'

    else:
        calib_instrument= 'L456102_calibrator/L456102_SB0'+str(i)+'_uv.dppp.MS.flg/instrument_tind'
        targetMS = 'L456106_target/L456106_SB0'+str(i)+'_uv.dppp.MS.flg'

    #
    print 'NDPPP appycal.parset msin='+str(targetMS)+' applycal.parmdb='+str(calib_instrument)+''
    os.system('NDPPP applycal.parset msin='+str(targetMS)+' applycal.parmdb='+str(calib_instrument)+'')
```

- python applysol.py

Phase calibration

- create a model of the field

```
use Lofar

cd L456106_target/

gsm.py [-p patchname] outfile RA DEC radius
[vlssFluxCutoff [assocTheta]]

gsm.py -p P1 P1.sky 315.00000000  52.90000000 5

makesourcedb in=P1.sky out=P1.sourcedb format="<"
```

- calibrate using the parset in yellow

```
 NDPPP ../phaseonly.parset
msin=L456106_SB000_uv.dppp.MS.flg
msout=L456106_SB000_uv.dppp.MS.flg.ph
gaincal.parmdb=L456106_SB000_uv.dppp.MS.flg.ph/
instrument
```

For all SBs:

```
for i in *flg; do NDPPP ../phaseonly.parset msin=
$i msout=$i.ph gaincal.parmdb=$i.ph/instrument;
done
```

```
msin=
msout=
msin.datacolumn =
CORRECTED_DATA
msin.baseline = [CR]S*&
msout.datacolumn = DATA
numthreads = 5
steps=[gaincal]

gaincal.sourcedb=P1.sourcedb
gaincal.sources =
gaincal.parmdb=
gaincal.type=gaincal
gaincal.caltype=phaseonly
gaincal.usebeammodel=false
gaincal.applysolution=True
```

OR use applycal to apply solutions

```
msin =
msin.datacolumn = DATA
msin.baseline = [CR]S*&
msout = .
msout.datacolumn =
CORRECTED_DATA

steps = [applycal]

applycal.type = applycal
applycal.parmdb =
```