

## Data inspection and editing (Flagging, demixing & averaging)

Tammo Jan Dijkema

LOFAR Data Processing School, 18 September 2018

# Outline: tools used in this tutorial

Tools for data inspection:

- MSOverview
- casabrowser
- RFIGui
- PlotMS
- TaQL ( [taql.astron.nl](http://taql.astron.nl) or dop341:8000 from within astron)
- Python + python-casacore

Tool for reduction of UV data:

- DPPP

Steps performed on raw correlated data:

1. Add weights
2. Flag and remove RFI (radio frequency interference)
3. Remove (“demix”) contribution of off-axis bright sources
4. Average / compress the data to lower time and freq. resolution

Compression is critical to decreasing processing time, which is necessary given the data volumes.

# Overview of data preprocessing steps

Steps performed on raw correlated data:

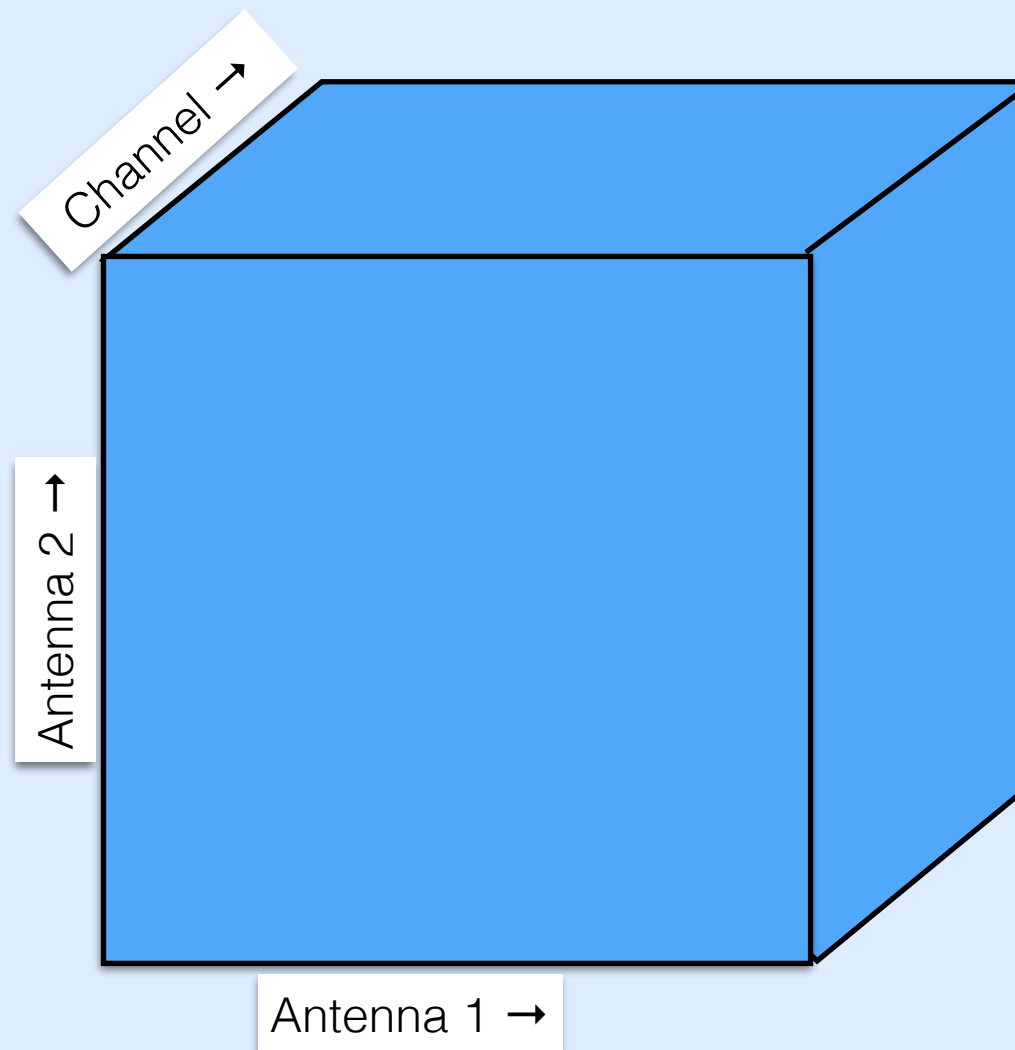
1. Add weights
2. Flag and remove RFI (radio frequency interference)
3. Remove (“demix”) contribution of off-axis bright sources
4. Average / compress the data to lower time and freq. resolution

Compression is critical to decreasing processing time, which is necessary given the data volumes.

# Visibilities, output of correlator

Visibilities vary along:

- Antenna 1
- Antenna 2
- Time
- Channel
- Polarization 1 (X,Y)
- Polarization 2 (X,Y)
- ...



# Visibilities, output of correlator

Correlator outputs a measurement set (MS) per subband, containing visibilities: a complex number for each

- Timeslot
- Baseline (combination of two antennas)
- Channel
- Correlation (XX, XY, YX, YY)

The screenshot shows a 'Table Browser' window with a table of visibility data. The table has columns for UVW, ANTENNA1, ANTENNA2, TIME, and DATA. The DATA column contains complex numbers. A callout box labeled 'channel' points to the 'DATA' column header. Another callout box labeled 'correlation' points to the 'DATA' column header. The detailed view on the right shows a complex array of size [4 64] with columns 1, 2, 3, 4, 5 and rows 0, 1, 2, 3. The array contains complex numbers.

UVW	ANTENNA1	ANTENNA2	TIME	DATA
[0, 0, 0]	0	0	2013-03-29 13:59:49.00	[4, 64] Complex
[-61.8924, -100.236, 52.5616]	0	1	2013-03-29-13:59:49.00	[4, 64] Complex
[0, 0, 0]	1	1	2013-03-29 13:59:49.00	[4, 64] Complex
[-09.4601, 341.317, -114.298]	0	2	2013-03-29-13:59:49.00	[4, 64] Complex
[-27.5678, 441.551, -166.869]	1	2	2013-03-29-13:59:49.00	[4, 64] Complex
[0, 0, 0]	2	2	2013-03-29 13:59:49.00	[4, 64] Complex
[-92.5321, 382.646, -129.721]	0	3	2013-03-29-13:59:49.00	[4, 64] Complex
[30.6398, 482.882, 182.286]	1	3	2013-03-29 13:59:49.00	[4, 64] Complex
[-3.07190, 41.3283, -15.4234]	2	3	2013-03-29-13:59:49.00	[4, 64] Complex
[0, 0, 0]	3	3	2013-03-29-13:59:49.00	[4, 64] Complex
[71.9337, 494.055, 177.613]	0	4	2013-03-29 13:59:49.00	[4, 64] Complex
[-10.0413, 594.292, -230.177]	1	4	2013-03-29-13:59:49.00	[4, 64] Complex
[7.75266, 152.738, -61.3146]	2	4	2013-03-29-13:59:49.00	[4, 64] Complex

channel →

← correlation

L114220\_SAP000\_SD040\_uv\_1.MS[3, 20] - Complex Array of size [4 64].

	1	2	3	4	5	
0	(1.17e-04, 1.72e-04)	(1.63e-04, 3.59e-04)	(2.85e-04, 5.35e-05)	(3.19e-04, 9.45e-05)	(4.02e-04, 2.59e-06)	[1.
1	(-1.25e-04, -1.57e-05)	(-1.61e-05, 2.60e-06)	(-1.79e-05, -6.01e-06)	(1.30e-05, -1.13e-04)	(-6.21e-05, 1.31e-05)	[-3
2	(8.50e-05, -7.75e-05)	(4.61e-05, 4.30e-05)	(4.04e-05, 8.34e-05)	(-7.04e-05, -8.13e-05)	(3.61e-05, -1.22e-05)	[-3
3	(6.62e-05, 6.70e-05)	(2.62e-04, 4.35e-05)	(2.04e-04, 5.93e-05)	(2.43e-04, 5.40e-05)	(1.20e-04, 2.17e-04)	[1.

Close

Close All

PAGE NAVIGATION First << [1 / 52] >> Last 1 Go Loading 1000 rows.

# Overview of DPPP (or DP<sup>3</sup>)

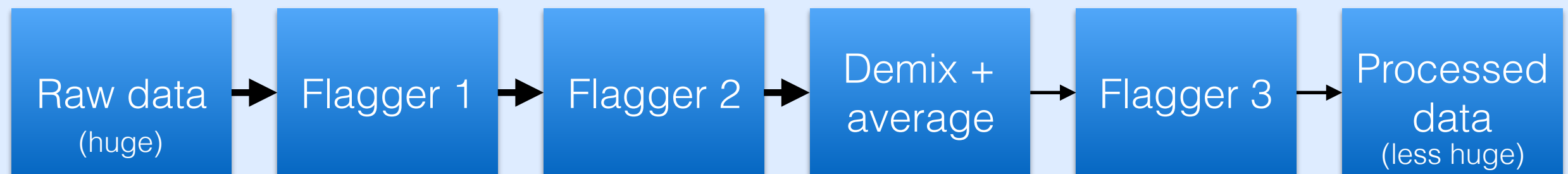
Default Preprocessing Pipeline: perform some operations on data.

It's a **pipeline**, so only read and written once. Data is piped through all steps as soon as possible: first data can be written to disk when last data has not been read yet.

DPPP is the only program that can read raw correlated LOFAR data and writes it out as the standard (CASA) MS-format.

A **step** operates on the output of the previous step.

Typical pipeline:



# DPPP: user interface

Command-line tool, input as a parset,  
output as feedback on screen.

```
> DPPP myreduction.parset
```

Overriding some parameters  
from the command line:

```
> DPPP myreduction.parset msin=L91.MS
```

Documentation:

[http://www.lofar.org/wiki/doku.php?id=public:user\\_software:ndppp](http://www.lofar.org/wiki/doku.php?id=public:user_software:ndppp)

```
msin=L123.MS
msout=L123_dppp.MS
steps=[flagger1, flagger2, demix, flagger3]

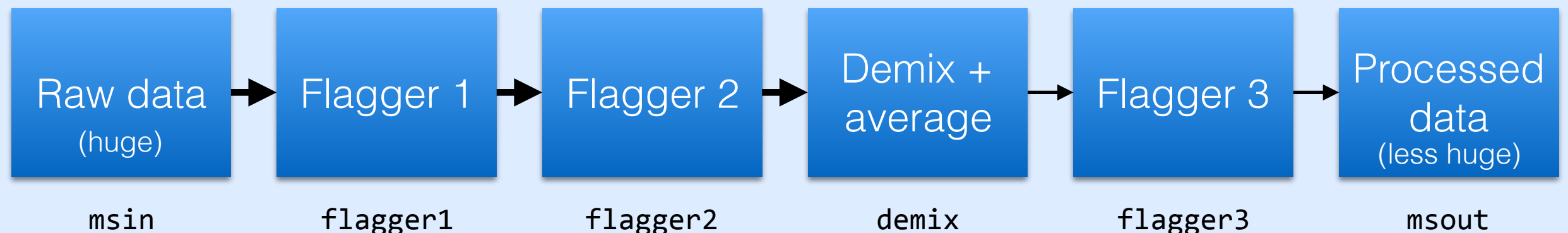
flagger1.type=aoflagger

flagger2.type=preflagger
flagger2.baseline=*&&& # autocorrelations

demix.type=demixer
demix.subtractsources=[CygA,CasA]
demix.skymodel=Ateam.sourcedb

flagger3.type=aoflagger
```

Typical pipeline:



# Weights, autoweight

Associated to each visibility  $v_{i,j} = \hat{v}_{i,j} + n_{i,j}$  (between station  $i$  and  $j$ ) is a weight  $w_{i,j}$ .

To exploit the data as much as possible, weights should be set such that noisy visibilities get down-weighted.

$$w_{i,j} = \frac{N_{\text{samples}}}{\sigma_i^2 \sigma_j^2}$$

Variance of noise of one station  $\sigma_i$  estimated from autocorrelation  $v_{i,i}$ .

Weights are computed when DPPP reads raw data and `msin.autoweight=true`.

Autoweight should be performed only once on the data.

Weights are stored in the column `WEIGHT_SPECTRUM`.



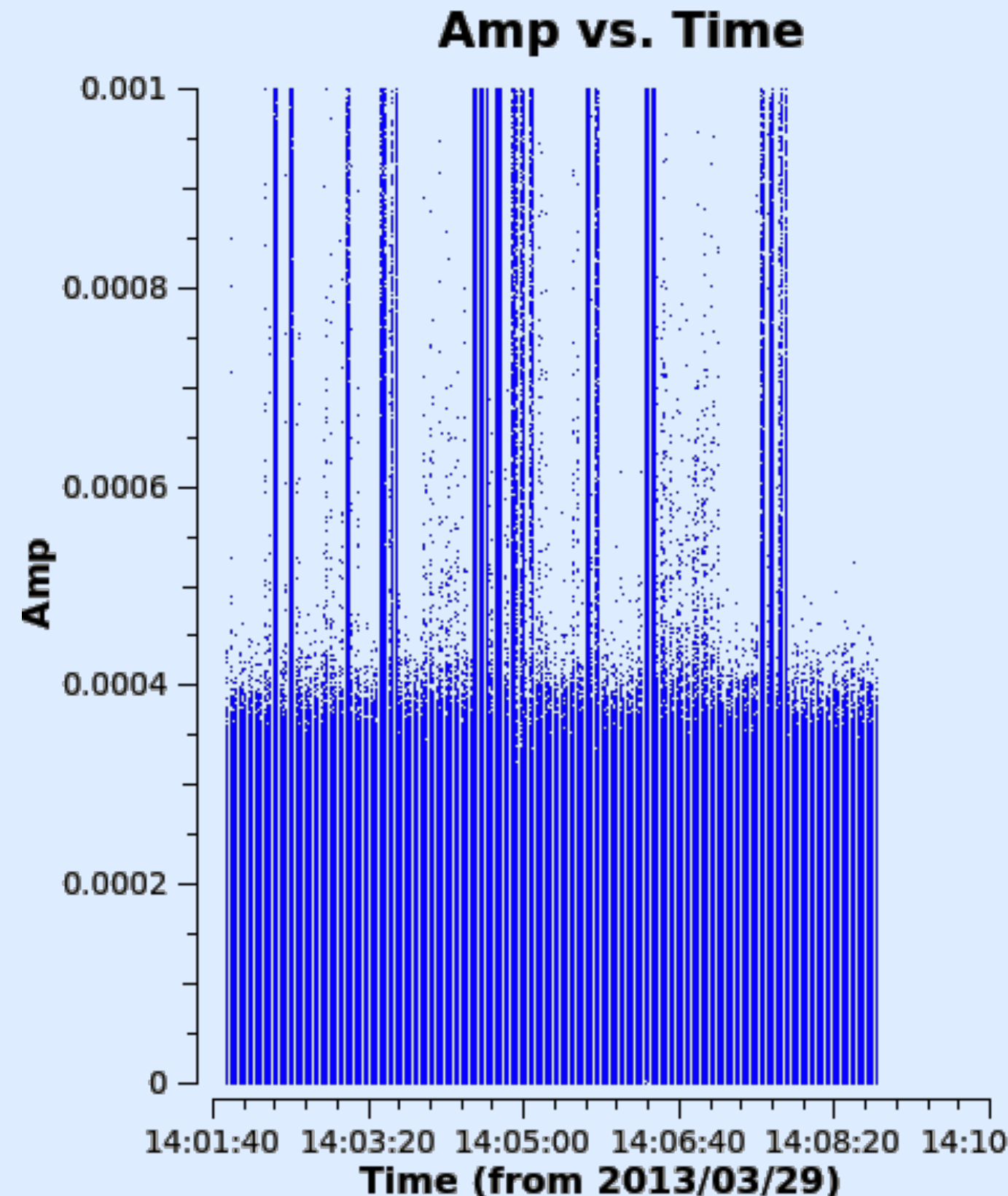
# Flagging

Some samples affected by radio frequency interference (RFI).

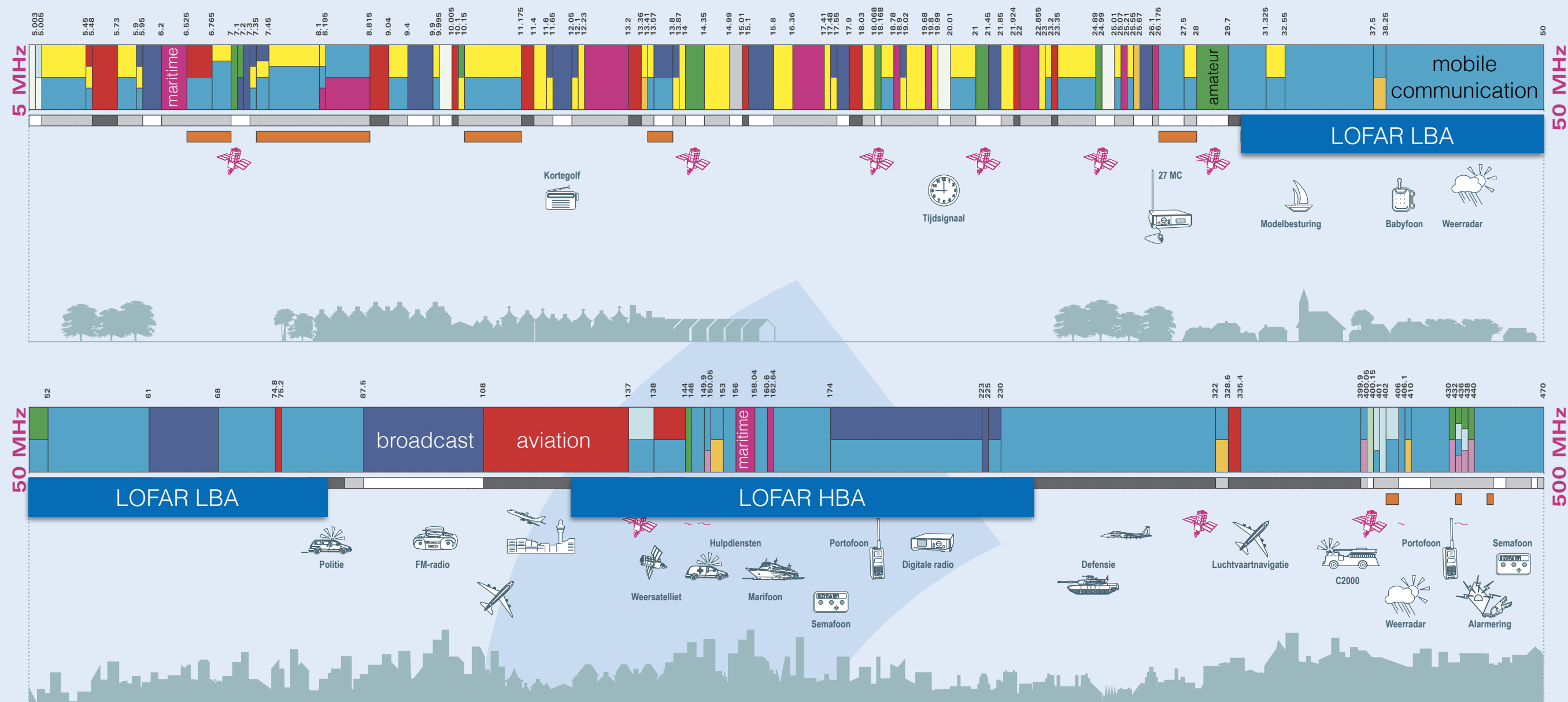
This makes these samples unsuitable for further processing.

We will flag them, and pretend they were never there.

Data is not deleted, just a check is put in **FLAG** column in MS.



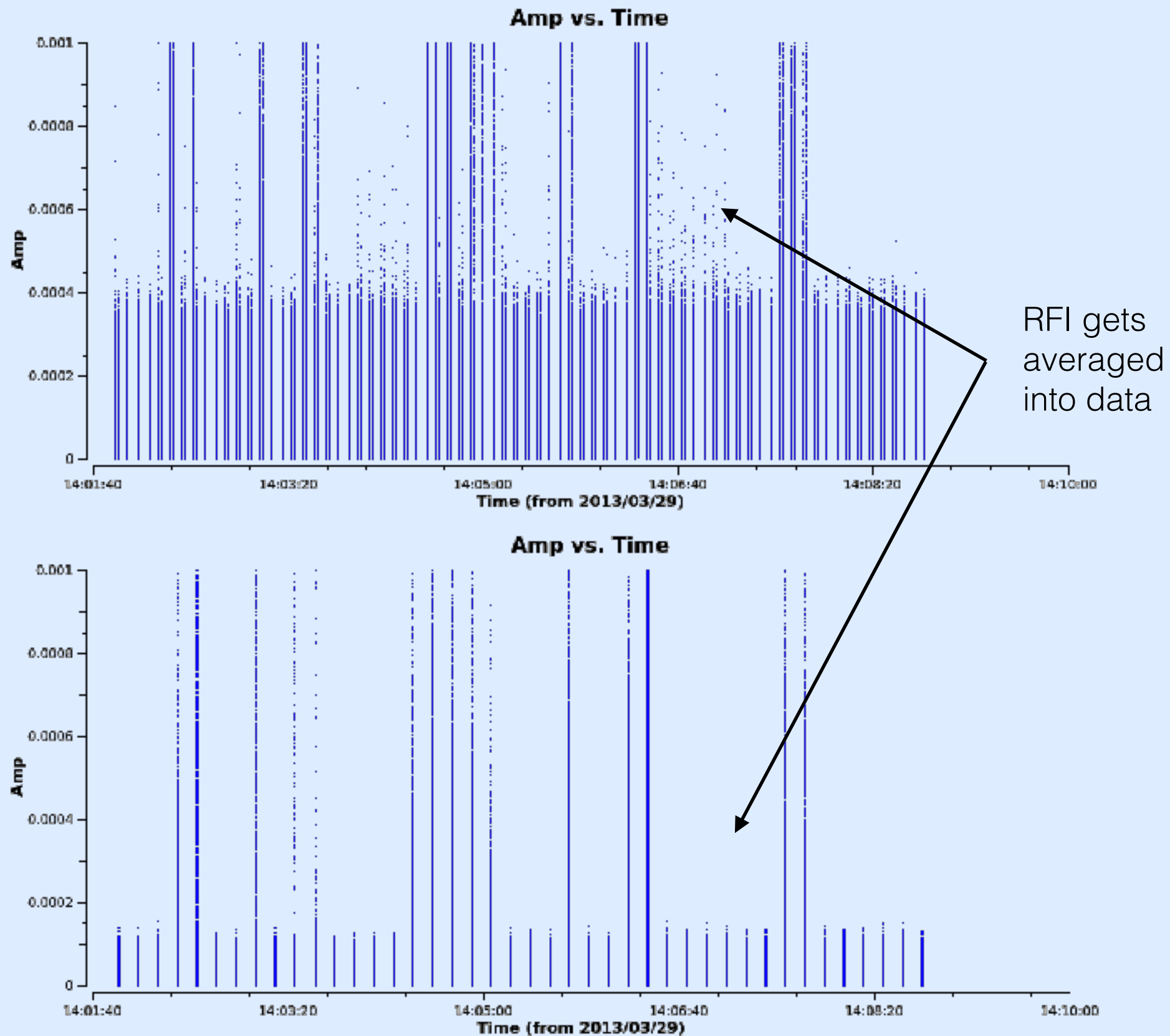
# Frequency allocations, interfering signals



Source: frequentiespectrumkaart 2005

Most RFI near LOFAR is narrowband and/or short duration.

# Data should be flagged at high resolution

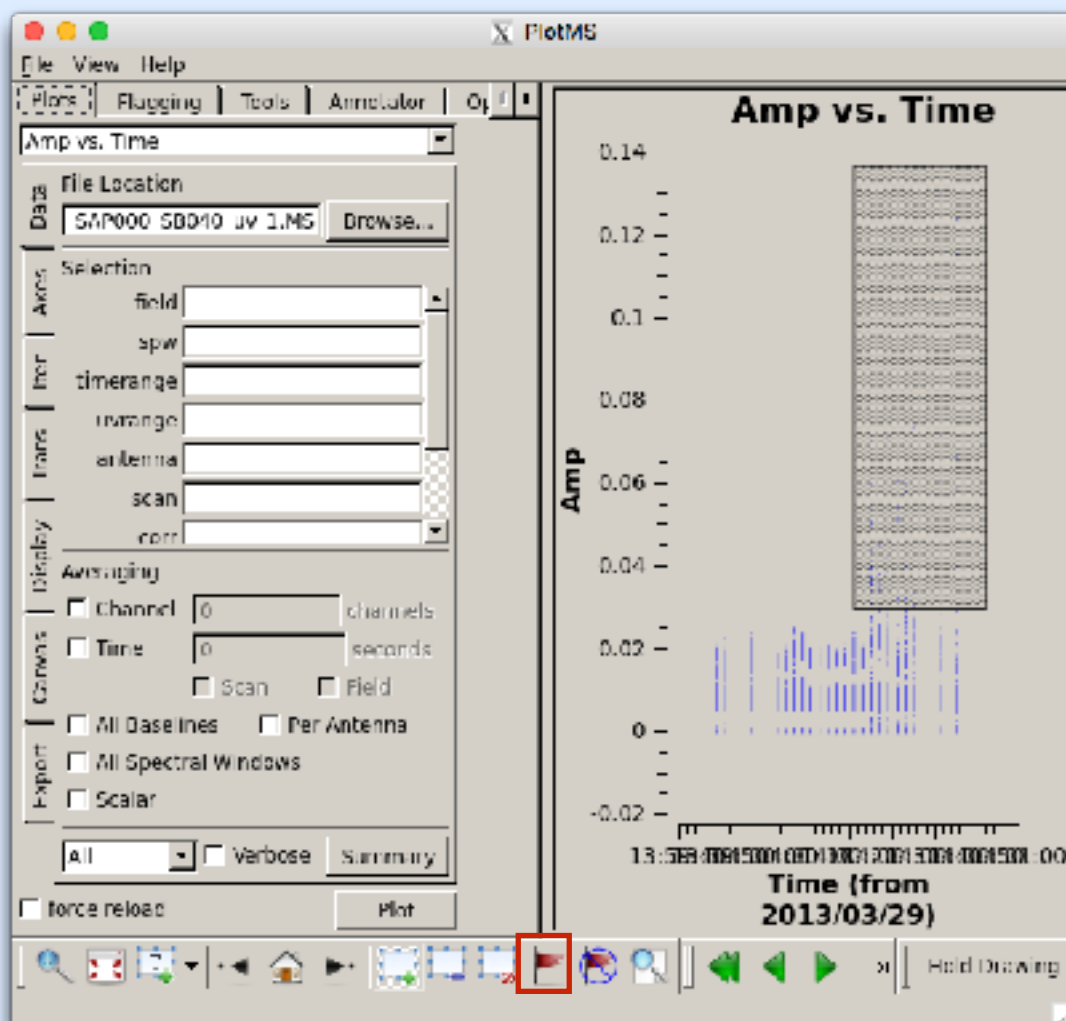


# Three methods of flagging

## 1. Manual Flagging

Inspect data, select visibilities to flag

Can be done with `casaplotms`



## 2. Semi-automatic flagging

For example:

- Flag all autocorrelations
- Flag all signal stronger than 100 Jy
- Flag the first channel
- Flag station CS013

Can be done with DPPP, step `preflagger`

## 3. Automatic flagging

For example using AOFlagger

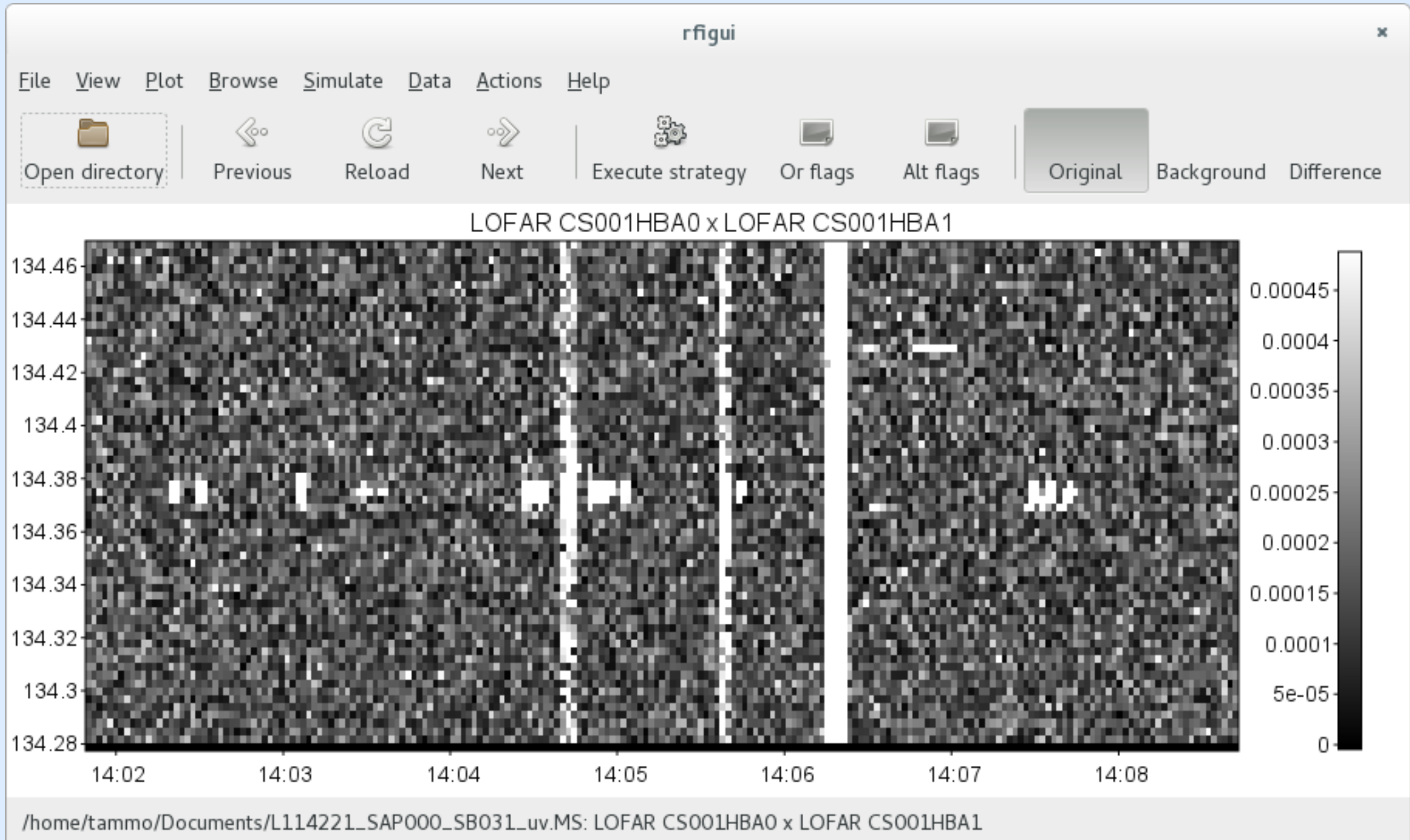
Flags based on time-freq statistics (per bl).  
Performs best on long time ranges!

Can be called from DPPP, step `aoflagger`

Interactive counterpart: `rfigui / aoflagger`

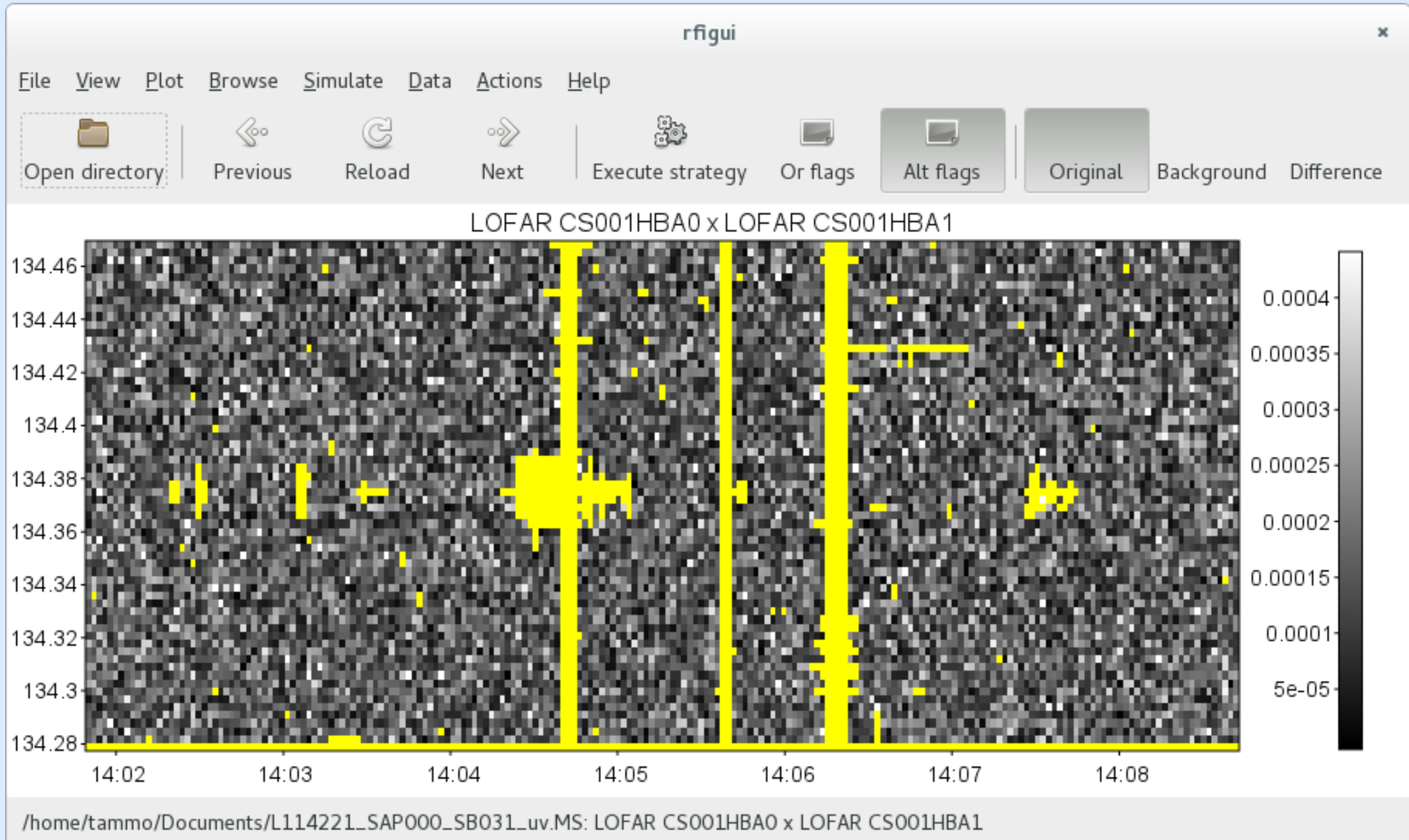
# AOFlagger, rfigui

AOFlagger (André Offringa) flags data based on statistics:



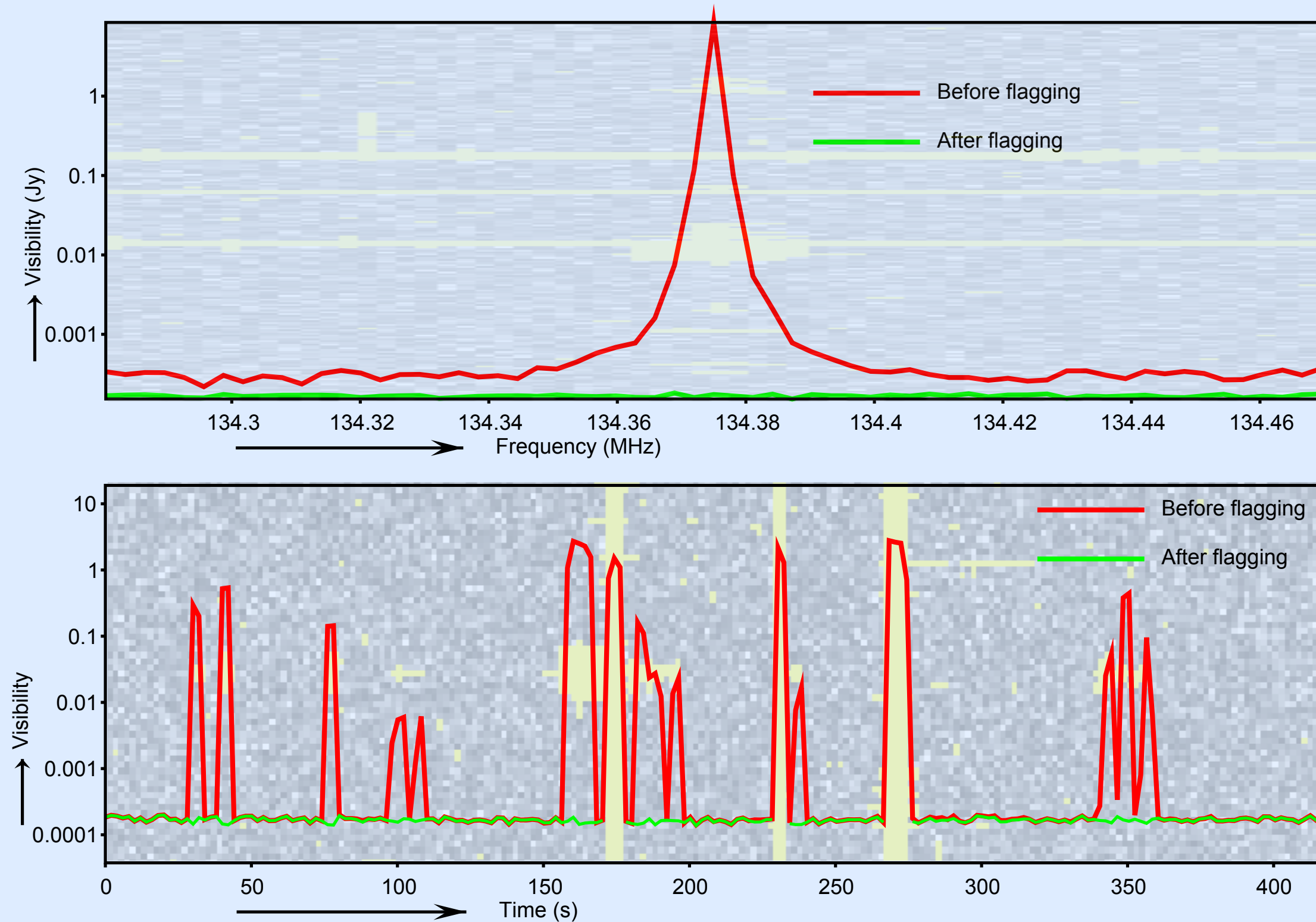
# AOFlagger, rfigui

AOFlagger (André Offringa) flags data based on statistics:





AOFlagger (André Offringa) flags data based on statistics:



# A flagging strategy

1. Flag data with preflagger, flag misbehaving stations.
2. AOFlagger on data on high resolution.
3. Inspect results, maybe some manual flagging.
4. Demix and average data.
5. Run AOFlagger on averaged data.
6. (optional) Inspect results before calibration
7. Calibrate
8. (optional) Run AOFlagger again



# Direction-independent calibration

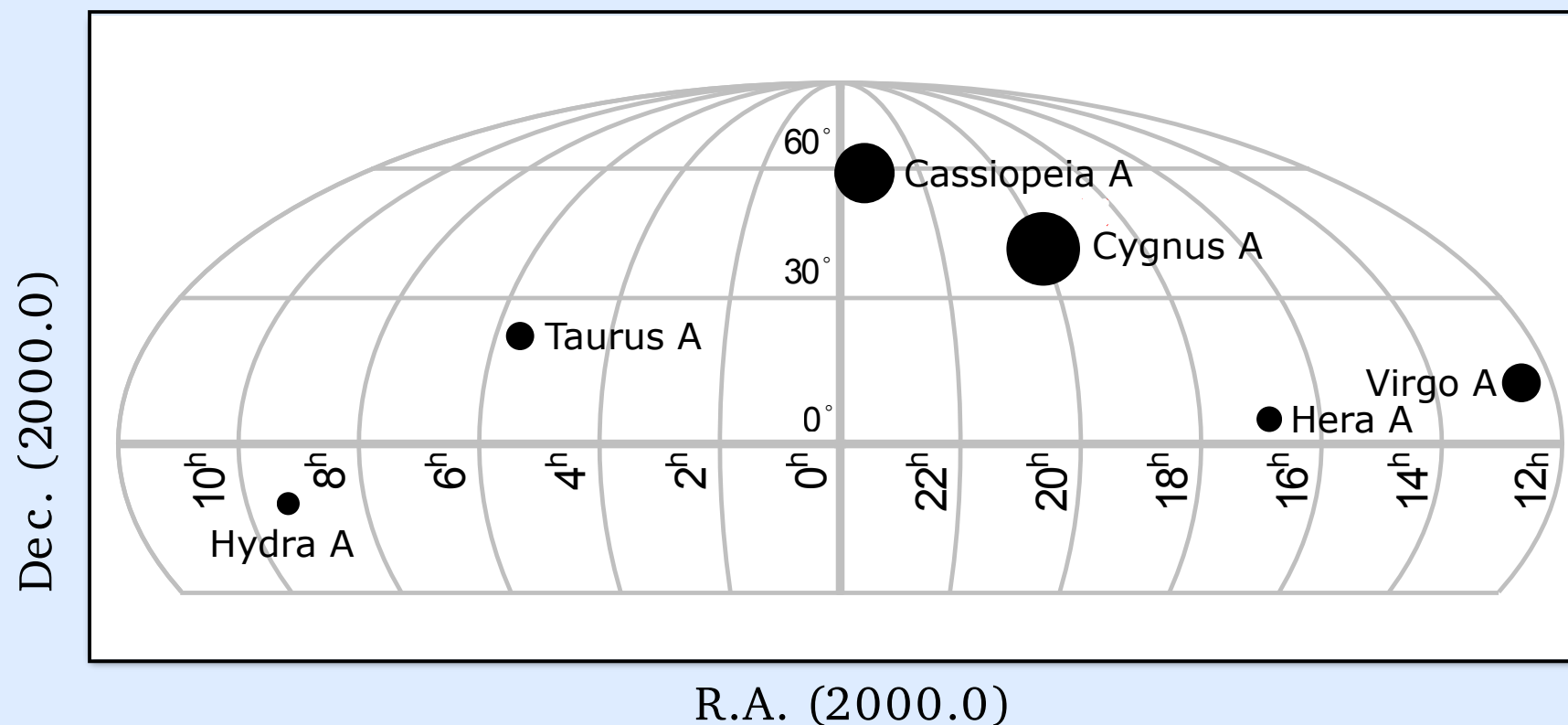
The signal you have measured has been altered by ‘unwanted’ station-, time and frequency dependent effects.

These effects are not known accurately beforehand, so let’s fit them afterwards, by fitting the data to a model sky.

Direction independent calibration in DPPP:



Sky at low frequencies is dominated by a few sources, together called **A-team** sources.



If A-team source is affecting signal, its signal needs to be subtracted. To subtract, the data must be calibrated against a model of the A-team source.

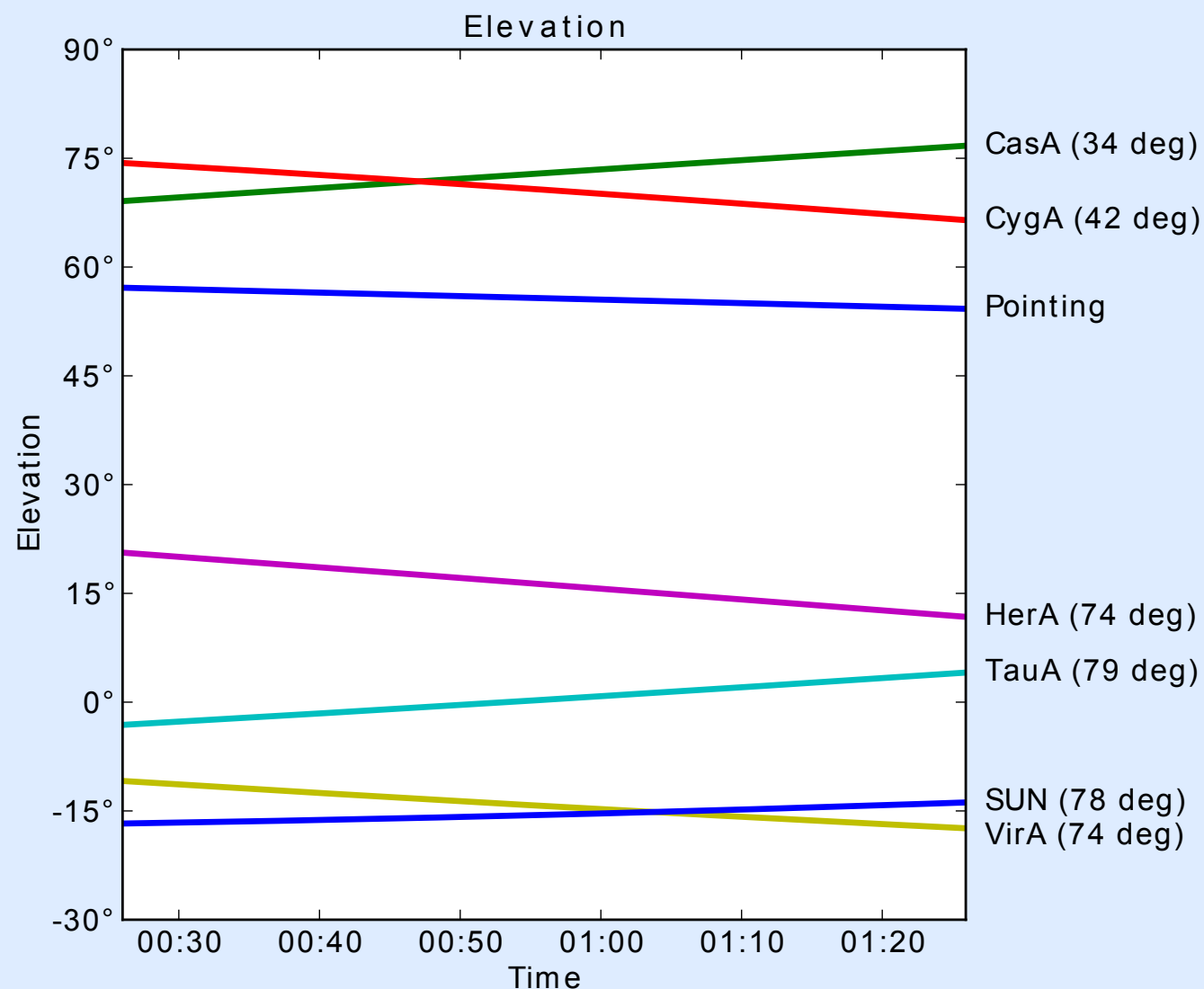
Time and frequency resolution needs to be such that signal from A-team sources is not too much affected by time and frequency smearing.

# Demixing: is your data affected by A-team? **ASTRON**

**LBA:** yes, your data is affected by CygA and CasA and perhaps more

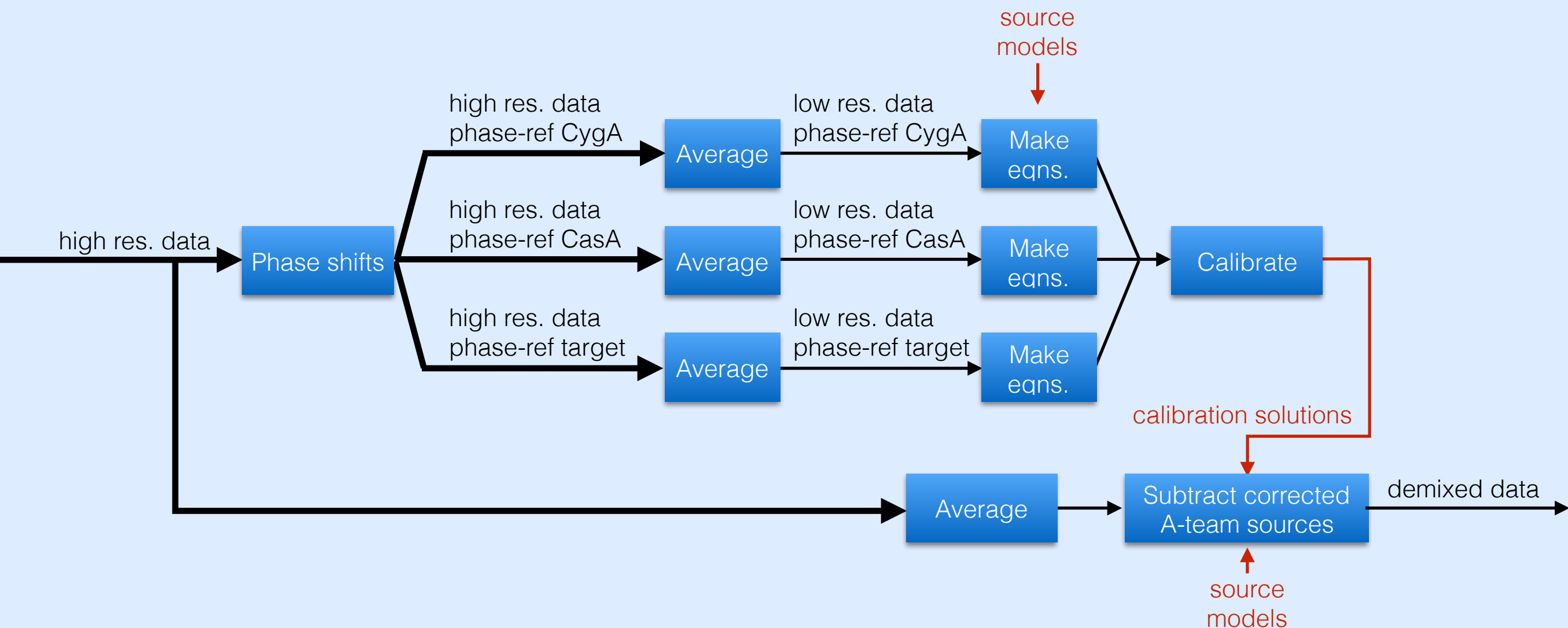
**HBA:** your data might be affected by A-team:

- If target is within  $30^\circ$  separation of A-team source
- If A-team elevation is high during observation



# How demixing works

- Demixing: subtracting calibrated model visibilities of bright sources
- Calibration is expensive: should be done on averaged data.
- Data can only be averaged near phase center.
- Idea: phase shift (high-res) data to the bright source, then average.



# Dysco compression

Dysco (Offringa 2016) is a technique to compress visibilities, in a lossy way, by cleverly quantizing numbers.

This can save a factor of 4 in visibility size and thus transfer time. The cost is an increase in the noise.

Important: compress visibilities only once! Recompression will introduce more noise.

The observatory has started dysco-compressing raw visibilities.

Dysco is loaded as a plug-in. If you see

```
Error: Shared library dyscostman not found in CASACORE_LDPATH or (DY)LD_LIBRARY_PATH
```

then you need to install dysco. On CEP3:

```
module load dysco
```

Steps performed on raw correlated data:

1. Add weights
2. Flag and remove RFI (radio frequency interference)
3. Remove (“demix”) contribution of off-axis bright sources
4. Average / compress the data to lower time and freq. resolution

Compression is critical to decreasing processing time, which is necessary given the data volumes.

Details can be found in the [cookbook](#) and [DPPP documentation](#).

# Log in to CEP3

Log in to the LOFAR portal:

```
> ssh lodsXX@portal.lofar.eu
```

Go on to the head node op CEP3:

```
> ssh lhd002
```

Now activate a dummy session using your reservation:

```
> screen
```

```
> srun -A lofarschool2018 --reservation=lofarschool2018_114 -N 1 -w  
lof0YY -u bash -i
```

Detach this screen (Ctrl-A).

Open a new terminal to do your actual work

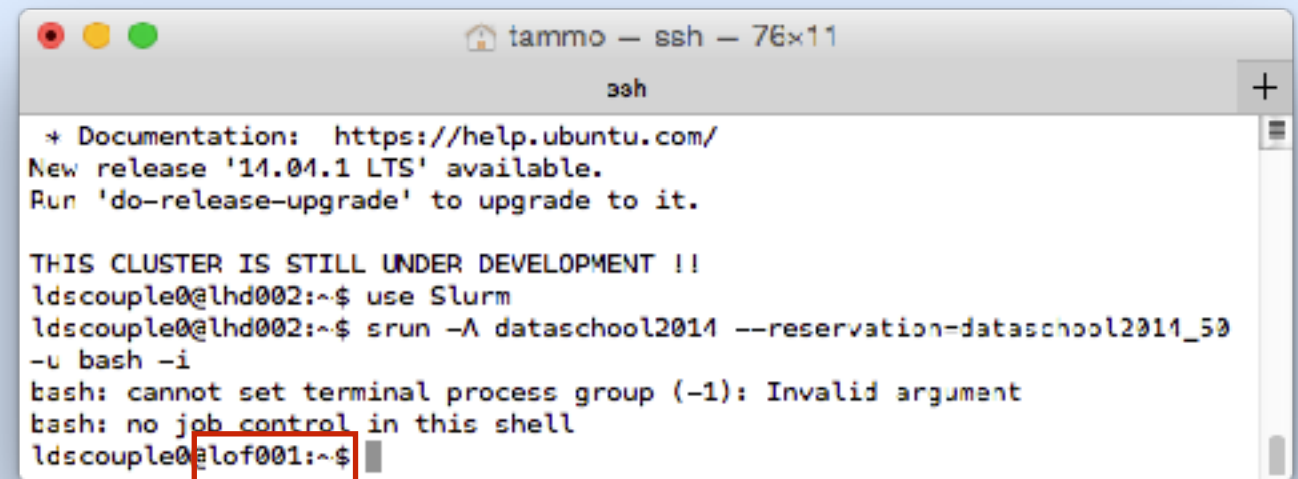
```
> ssh -Y portal.lofar.org
```

```
> ssh -Y lhd002
```

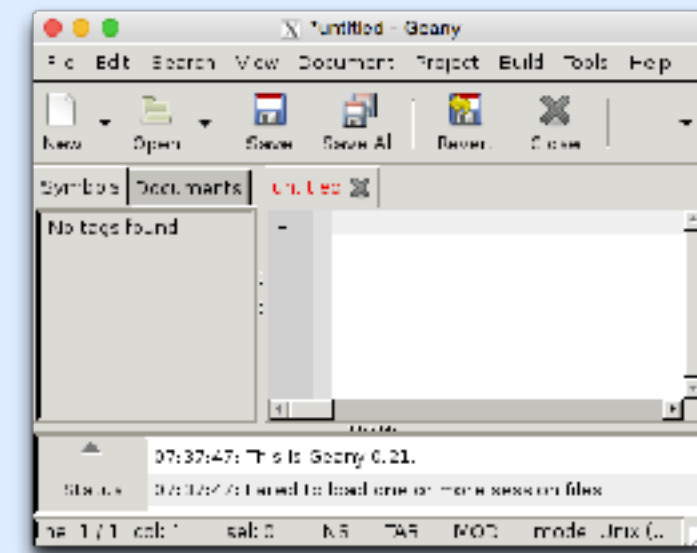
```
> ssh -Y lof0YY
```

Verify that graphics forwarding works:

```
> geany
```



```
tammo - ssh - 76x11  
ssh  
+ Documentation: https://help.ubuntu.com/  
New release '14.04.1 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
THIS CLUSTER IS STILL UNDER DEVELOPMENT !!  
ldscouple@lhd002:~$ use Slurm  
ldscouple@lhd002:~$ srun -A dataschool2014 --reservation=dataschool2014_50  
-u bash -i  
bash: cannot set terminal process group (-1): Invalid argument  
bash: no job control in this shell  
ldscouple@lof001:~$
```



# Explore data

The dataset for this exercise is on your node in `/data/scratch/DATASCHOOL2018_T1/`  
Have a look at this data and get an idea about the size of the measurement set

```
> cd /data/scratch/DATASCHOOL2018_T1/  
> du -hs .
```

★ How large is this subband?

To have a closer look, we need some astronomical tools.

```
> module load casa  
> module load lofar
```

Now we can see some real information:

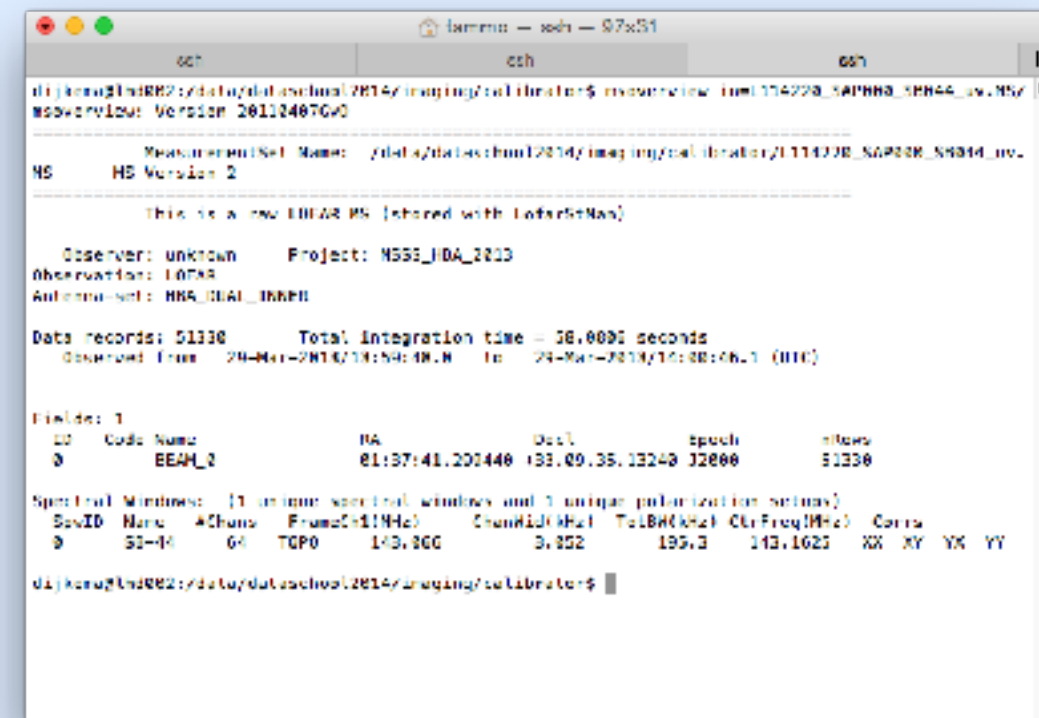
```
> msoverview in=L114221_SAP000_SB031_uv.MS
```

- ★ Which field was observed?
- ★ What was the duration of this observation?
- ★ What was the center frequency of this subband?
- ★ How many channels (frequencies) are in there?

More verbose information:

```
> msoverview in=L114221_SAP000_SB031_uv.MS verbose=true
```

- ★ What is the number of time slots? What is the integration time?
- ★ How many stations, how many baselines? (And what is the relation?)





# Get rid of LofarStMan

msoverview mentions: This is a raw LOFAR MS (stored with LofarStMan)

This means the data cannot be handled with casa. The same is currently true if the data has been compressed with Dysco. Let's create a copy in plain casa format.

First, go to the compute node, and make a directory to do your work in.

```
> ssh -Y lof0YY
> mkdir /data/scratch/lods0XX
> cd /data/scratch/lods0XX
```

Now we should convert the data to something in casa format.

```
> geany DPPP-makeplain.parset # or vim, emacs, nano, ...
```

Put the following commands in the parset and save it:

```
msin=L114221_SAP000_SB031_uv.MS
msout=L114221_SAP000_SB031_uv_plain.MS
msin.autoweight=true
steps=[]
```

Now run DPPP on this parset:

```
> DPPP DPPP-makeplain.parset
```

Now we have our own copy which can be opened in casa tools

# casaplotms

Try to open the data in casa tools:

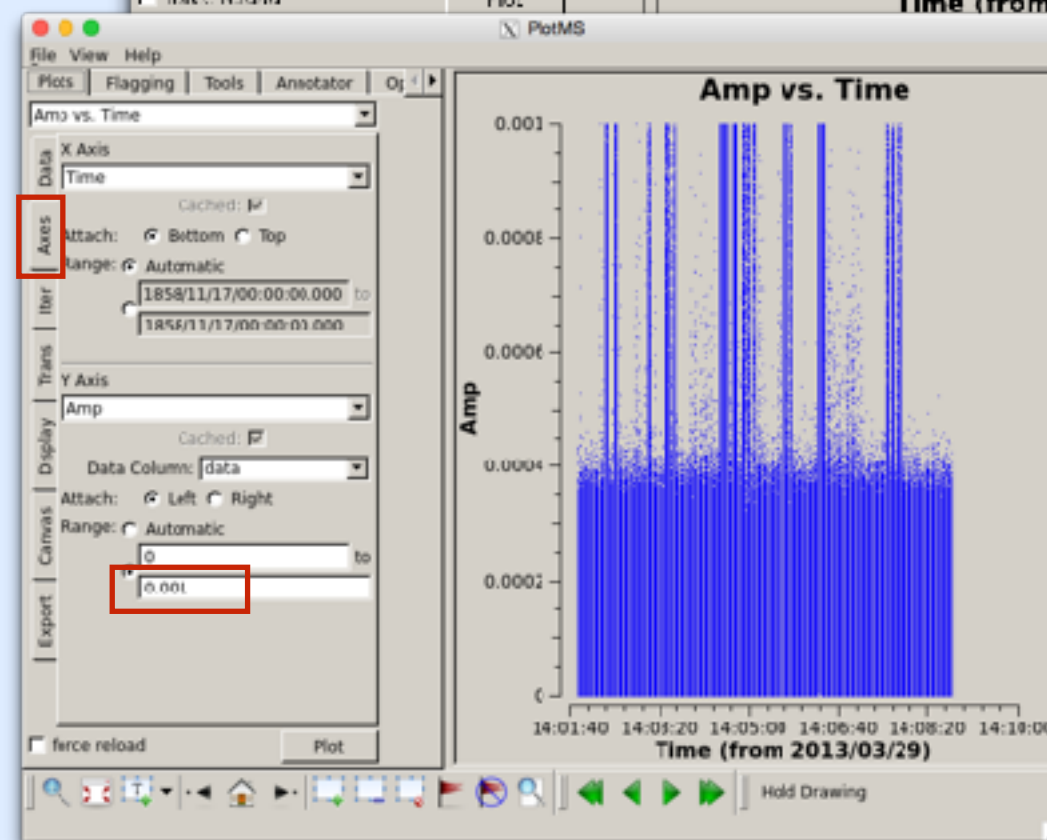
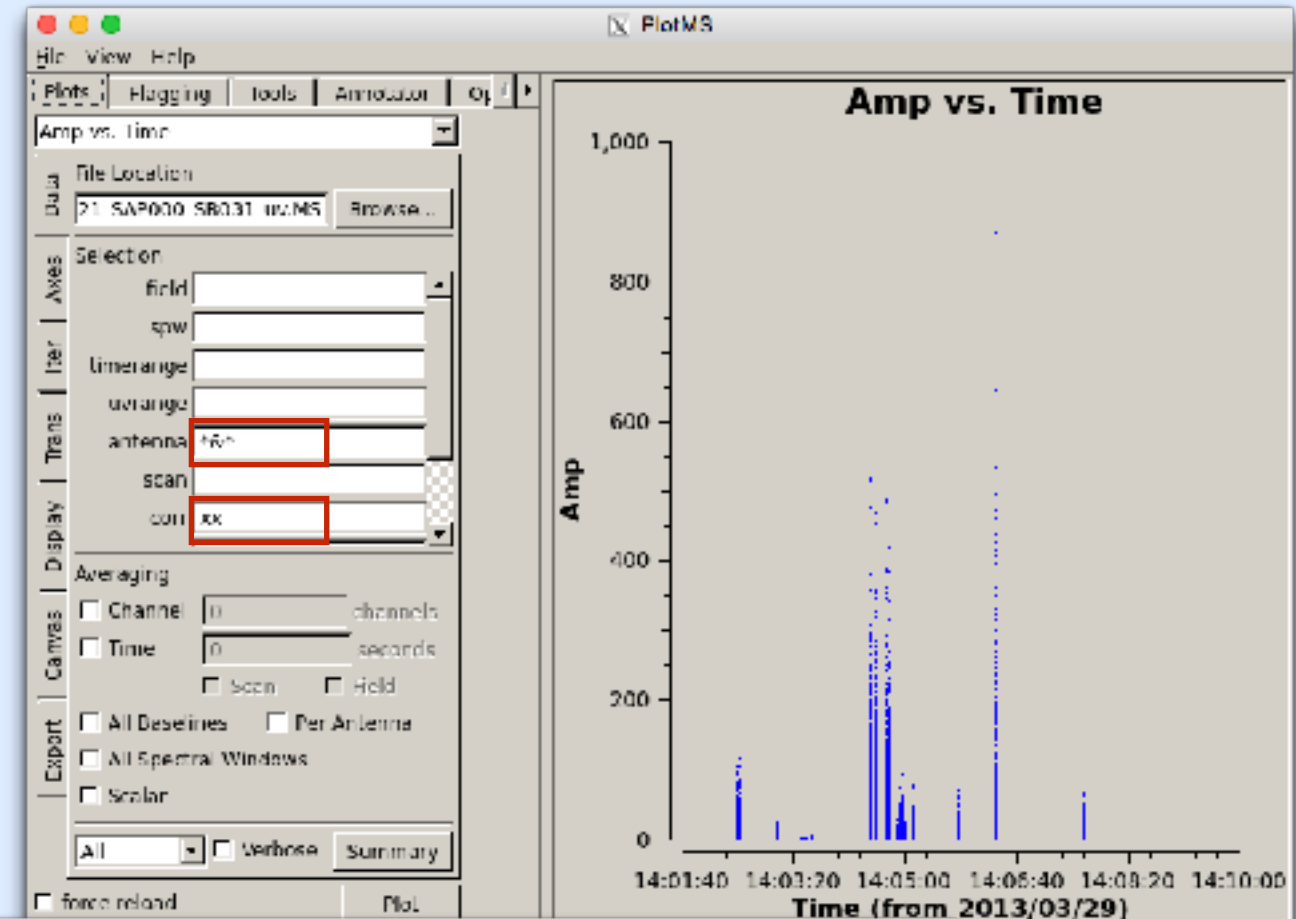
> `casaplotms`

Open your MS from the GUI. To speed up plotting, only plot the xx correlation. Select only cross correlations by typing `*&*` in the antenna field

(\*: any antenna, &: cross correlations)

Note the large spikes: probably RFI. Adjust the y-scale to find any real signal. In the Axes tab, set max y-scale to something sensible.

★ Does the scale of the signal make sense to you?  
(answer: no)



# rfigui


rfigui makes plots to detect RFI

```
> rfigui  
L114221_SAP000_SB031_uv.MS
```

In the dialog, just choose 'Open'  
Press 'Forward' to see data for the first  
baseline: LOFAR CS001HBA0 × LOFAR CS001HBA1

The spikes we spotted are clearly RFI.  
Some are broadband, some are not.

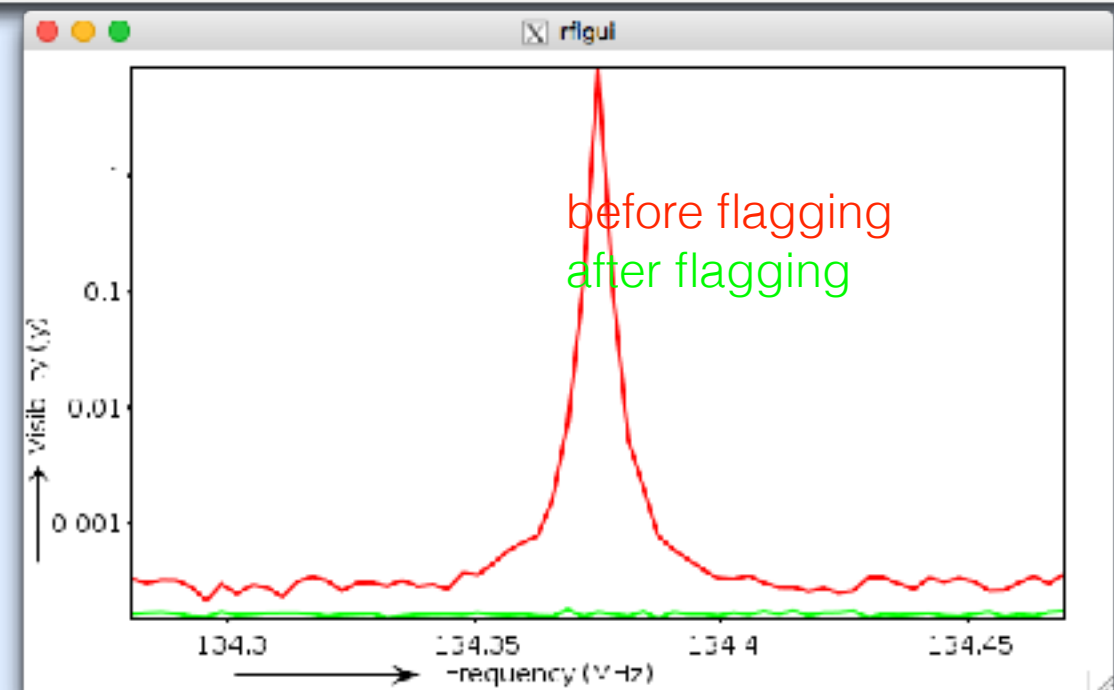
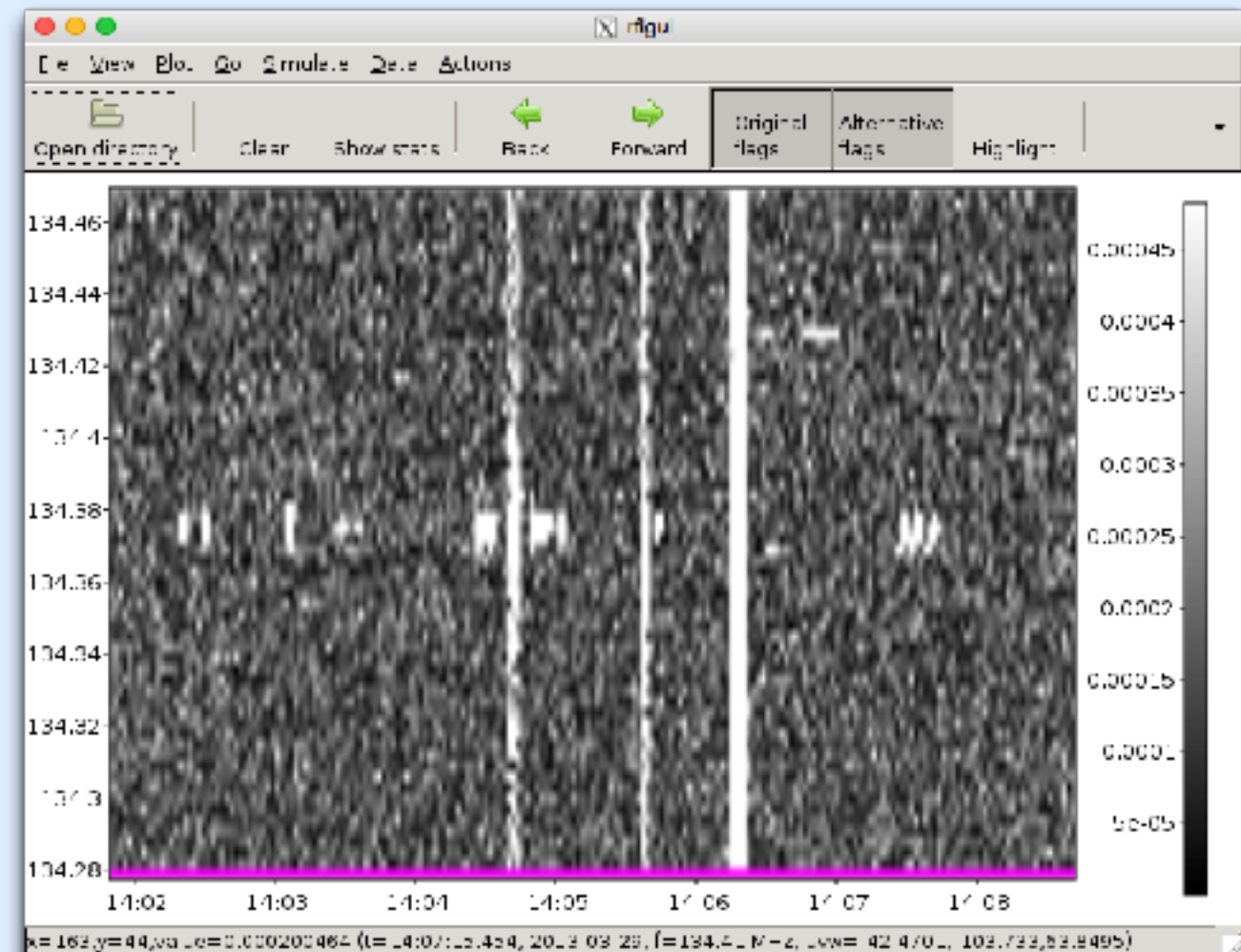
Create a power spectrum plot (plot menu).

 What is the interference at 134.375  
MHz?

rfigui (aoflagger) can also flag the data:  
Actions, Execute Strategy

Make a power spectrum plot (Plot menu).

Find other useful plots in the plot menu.



# aoflagger

The AOfFlagger can be called with DPPP.

- > `cd /data/scratch/lods0XX`
- > `geany DPPP-flag.parset`  
*# or vim, emacs, nano, ...*

Enter the following contents in the parset:  
`msin=L114221_SAP000_SB031_uv_plain.MS`  
`msout=.`

`steps=[preflagger, aoflagger]`  
`preflagger.baseline=*&&& # autocorr's`

Now run your first real action:

- > `DPPP DPPP-flag.parset`

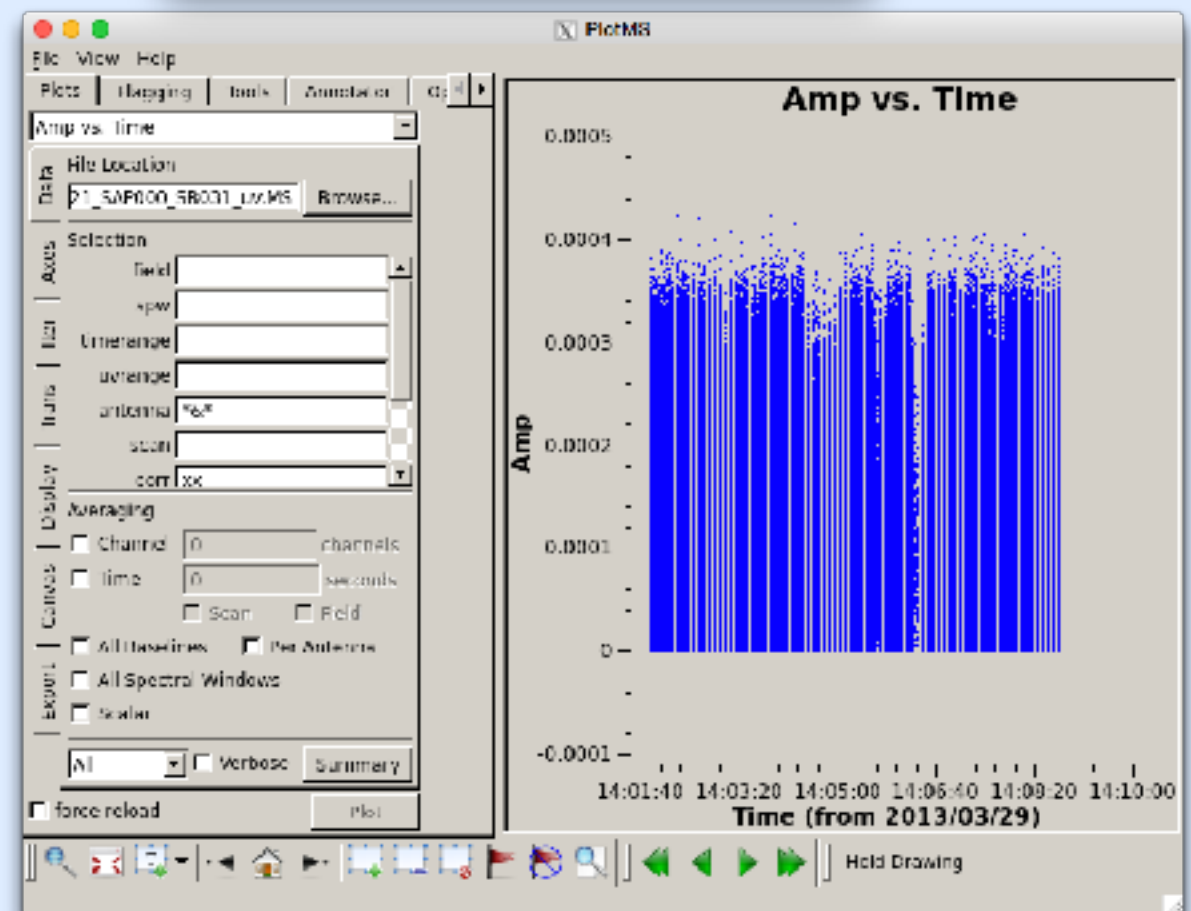
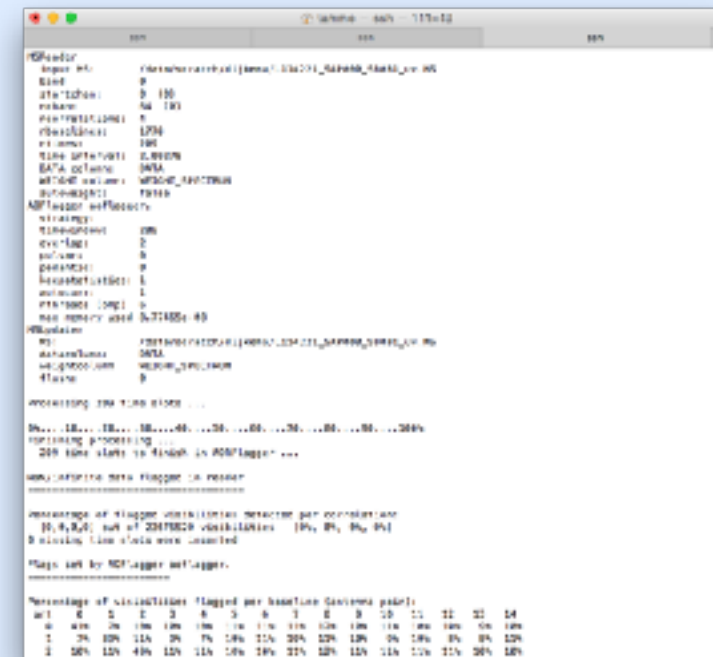
From the output:

- ★ Which station/ant nr was most affected?
- ★ Which channel was most affected?

Re-examine the data with casaplotms

- > `casaplotms`

Does the data look reasonable now?



# averaging

The data has been flagged, so now we can average it down in time and frequency.

```
msin=L114221_SAP000_SB031_uv_plain.MS # The 'plain' data
```

```
msout=L114221_SAP000_SB031_uv_plain_avg.MS
```

```
steps=[average]
```

```
average.timestep=5
```

```
average.freqstep=8
```

Run this parset through DPPP.

★ Do you get the compression you expected?

As the name suggests, DPPP (Default PreProcessing Pipeline) was designed do pipelines of steps. We could have done our reduction in one go:

```
msin=L114221_SAP000_SB031_uv.MS # The raw data
```

```
msin.autoweight=true
```

```
msout=L114221_SAP000_SB031_uv_avg.MS
```

```
steps=[preflagger,aoflagger,averager]
```

```
preflagger.baseline=*&&&
```

```
averager.timestep=5
```

```
averager.freqstep=8
```

★ Have another look in rfigui. Does the data look ok?