

Obtain a login sheet

Just for reference:

Username	Password	Reservation	Node
ldscouple0	30ce4bda00	dataschool2014_50	lof001
ldscouple1	30ce4bda01	dataschool2014_50	lof001
ldscouple2	30ce4bda02	dataschool2014_50	lof001
ldscouple3	30ce4bda03	dataschool2014_51	lof002
ldscouple4	30ce4bda04	dataschool2014_51	lof002
ldscouple5	30ce4bda05	dataschool2014_51	lof002
ldscouple6	30ce4bda06	dataschool2014_52	lof003
ldscouple7	30ce4bda07	dataschool2014_52	lof003
ldscouple8	30ce4bda08	dataschool2014_52	lof003
ldscouple9	30ce4bda09	dataschool2014_53	lof004
ldscouple10	30ce4bda10	dataschool2014_53	lof004
ldscouple11	30ce4bda11	dataschool2014_53	lof004
ldscouple12	30ce4bda12	dataschool2014_54	lof005
ldscouple13	30ce4bda13	dataschool2014_54	lof005

Username	Password	Reservation	Node
ldscouple14	30ce4bda14	dataschool2014_54	lof005
ldscouple15	30ce4bda15	dataschool2014_55	lof006
ldscouple16	30ce4bda16	dataschool2014_55	lof006
ldscouple17	30ce4bda17	dataschool2014_55	lof006
ldscouple18	30ce4bda18	dataschool2014_56	lof007
ldscouple19	30ce4bda19	dataschool2014_56	lof007
ldscouple20	30ce4bda20	dataschool2014_56	lof007
ldscouple21	30ce4bda21	dataschool2014_57	lof008
ldscouple22	30ce4bda22	dataschool2014_57	lof008
ldscouple23	30ce4bda23	dataschool2014_57	lof008
ldscouple24	30ce4bda24	dataschool2014_58	lof009
ldscouple25	30ce4bda25	dataschool2014_58	lof009
ldscouple26	30ce4bda26	dataschool2014_58	lof009

Log in to CEP3

Log in to the LOFAR portal:

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

Go on to the head node op CEP3:

```
> ssh lhd002
```

Now activate a dummy session using your reservation:

```
> use Slurm
```

```
> srun -A dataschool2014 --reservation=dataschool2014_XX -u bash -i
```

Note the hostname of your assigned node, something like l0f0XX

Open a new terminal to do your actual work (keep previous terminal open)

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

```
> ssh -Y lhd002
```

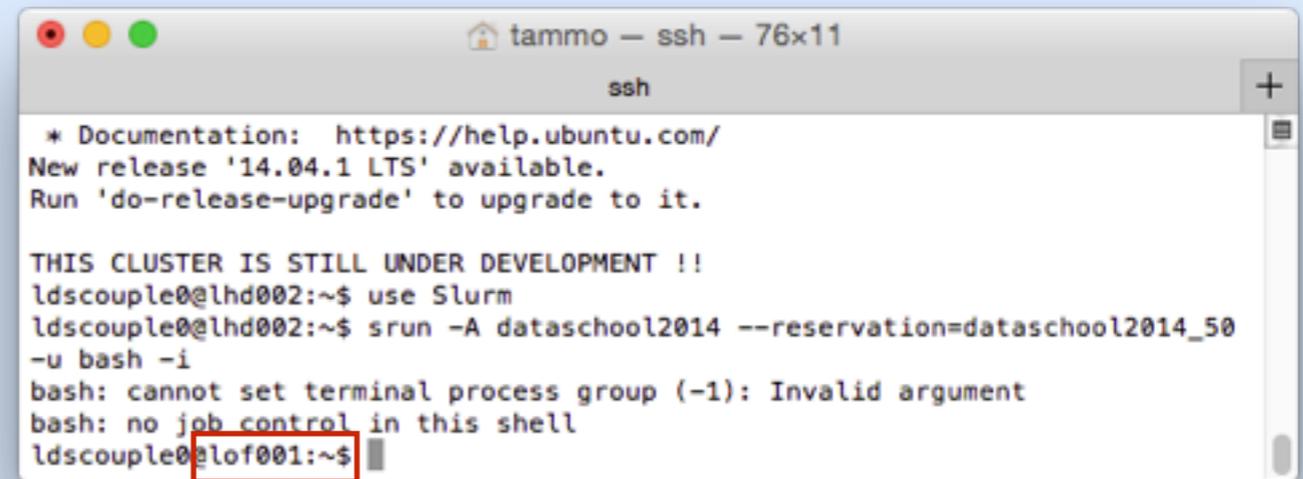
```
> ssh -Y l0f0XX
```

Verify that graphics forwarding works:

```
> geany
```

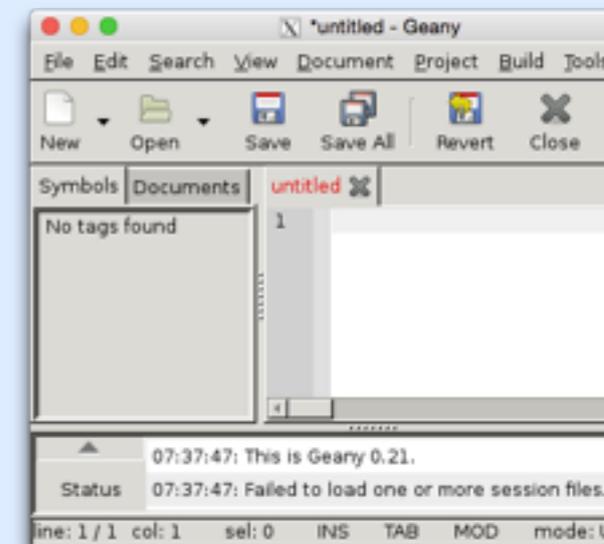
We'll start working on the head node (no serious processing), so exit the l0f0XX-node for now to come back to lhd002

```
> exit
```



```
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 !!
ldscouple0@lhd002:~$ use Slurm
ldscouple0@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
ldscouple0@l0f001:~$
```



Explore data

The full data for this tutorial is stored in `lhd002:/data/scratch/dataschool/imaging`

Have a look at this data and get an idea about the size of the measurement sets

```
> cd /data/scratch/dataschool/imaging
```

```
> du -hs .
```

★ How much disk space does the total observation take?

★ How many subbands were recorded for the calibrator, how many for the target?

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

```
> use Lofar
```

Now we can see some real information:

```
> cd /data/dataschool2014/imaging/target
```

```
> 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?

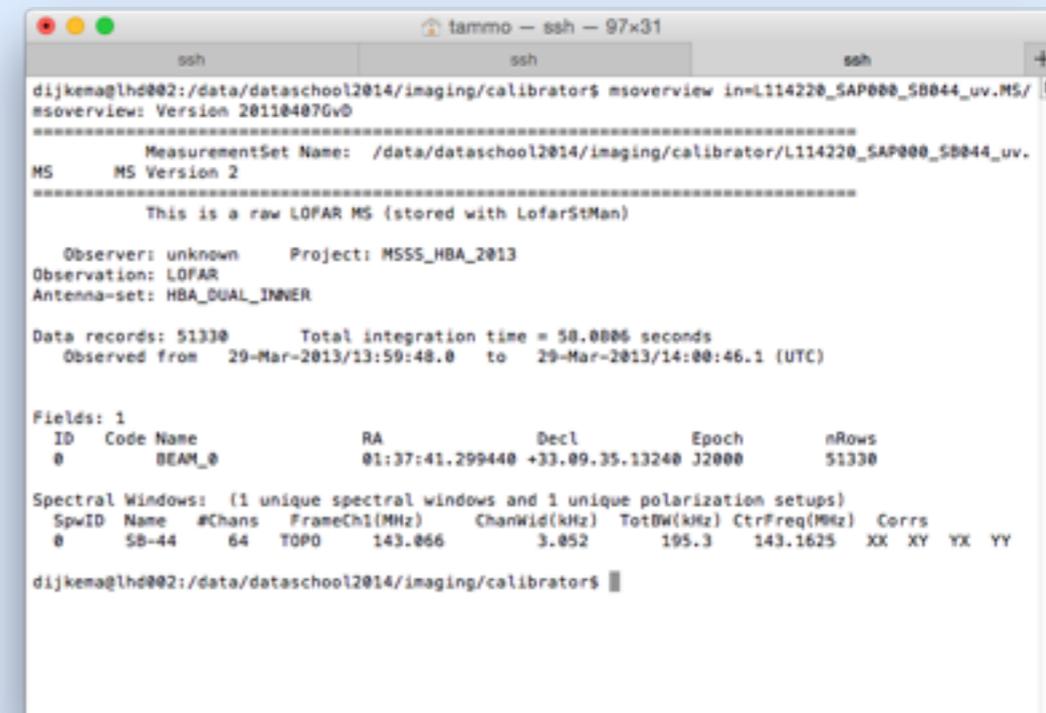
★ How many channels (frequencies) are in there?

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?)



```
tammo - ssh - 97x31
ssh ssh ssh
dijkema@lhd002:/data/dataschool2014/imaging/calibrators$ msoverview in=L114220_SAP000_SB044_uv.MS/
msoverview: Version 20110407Gv0
=====
MeasurementSet Name: /data/dataschool2014/imaging/calibrator/L114220_SAP000_SB044_uv.
MS MS Version 2
=====
This is a raw LOFAR MS (stored with LofarStMan)

Observer: unknown Project: MSSS_HBA_2013
Observation: LOFAR
Antenna-set: HBA_DUAL_INNER

Data records: 51330 Total integration time = 50.0006 seconds
Observed from 20-Mar-2013/13:59:48.0 to 20-Mar-2013/14:00:46.1 (UTC)

Fields: 1
ID Code Name RA Decl Epoch nRows
0 BEAM_0 01:37:41.299440 +33.09.35.13240 J2000 51330

Spectral Windows: (1 unique spectral windows and 1 unique polarization setups)
SpwID Name #Chans FrameCh1(MHz) ChanWid(kHz) TotBW(kHz) CtrFreq(MHz) Corrs
0 SB-44 64 TOP0 143.066 3.052 195.3 143.1625 XX XY YX YY

dijkema@lhd002:/data/dataschool2014/imaging/calibrators$
```

Get rid of LofarStMan

msoverview mentions: This is a raw LOFAR MS (stored with LofarStMan)
This means the data cannot be handled with casa. Let's grab a copy in casa format.
First, go to the compute node, and make a directory to do your work in.

```
> ssh -Y lof0XX  
> cd /data/scratch/school_tmp  
> mkdir yourname  
> cd yourname
```

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=/data/dataschool2014/imaging/target/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:

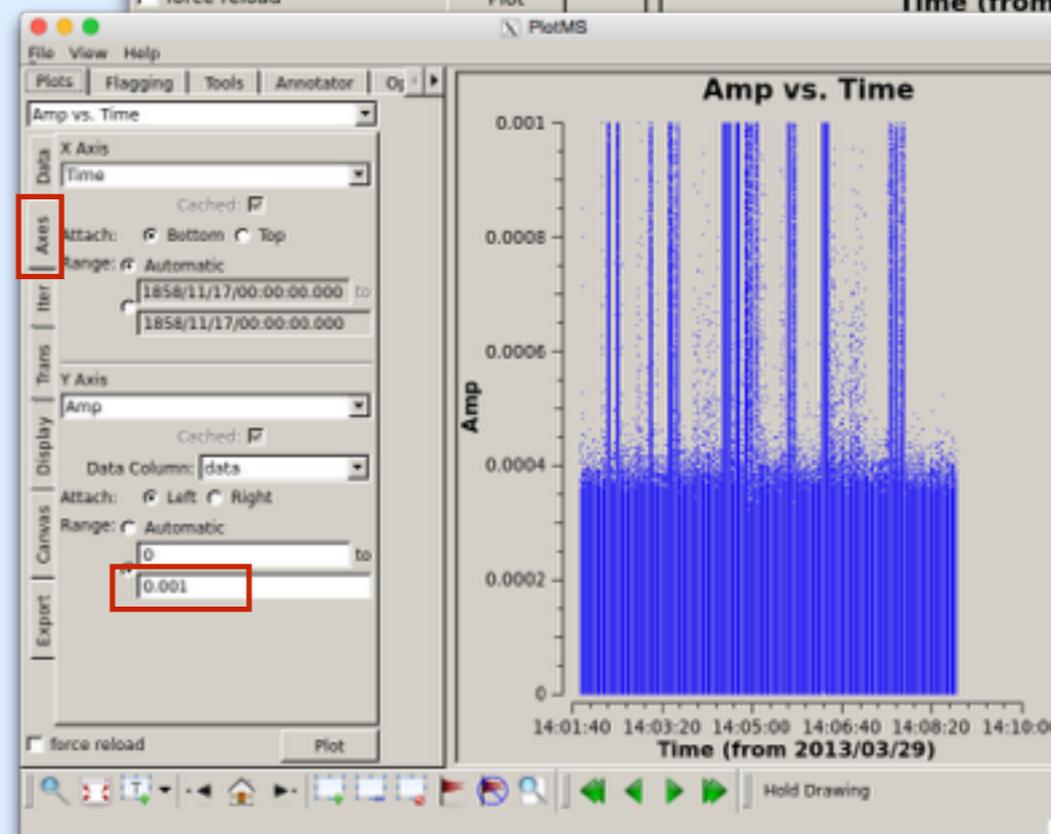
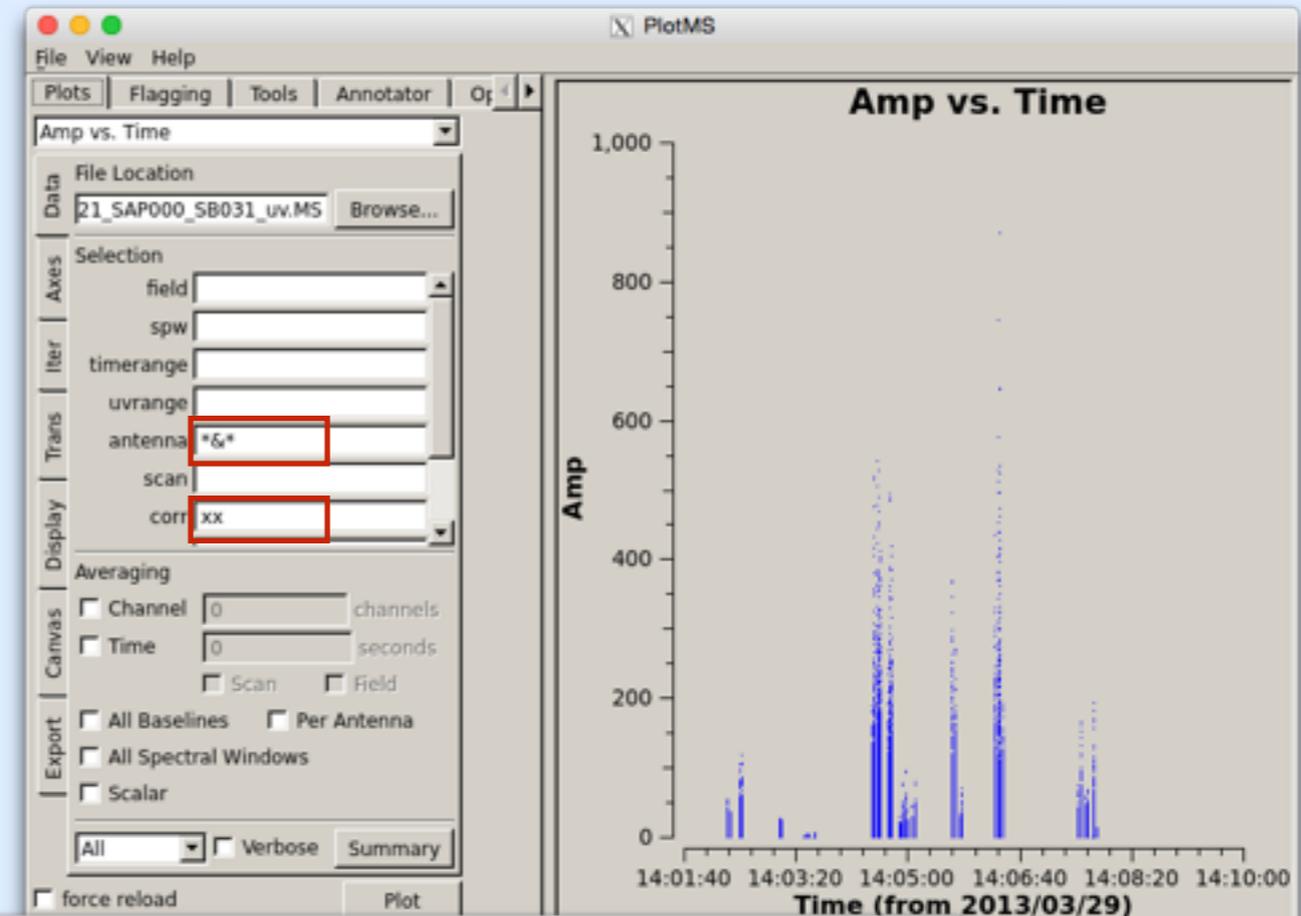
- > use Casa
- > casaplotms &

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.

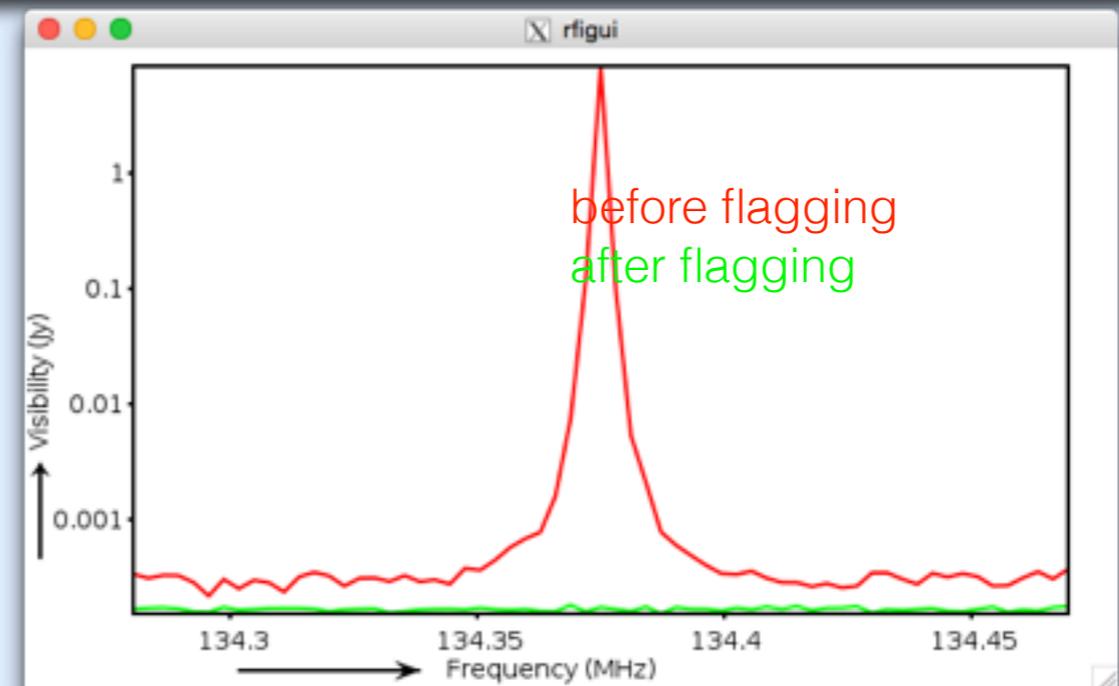
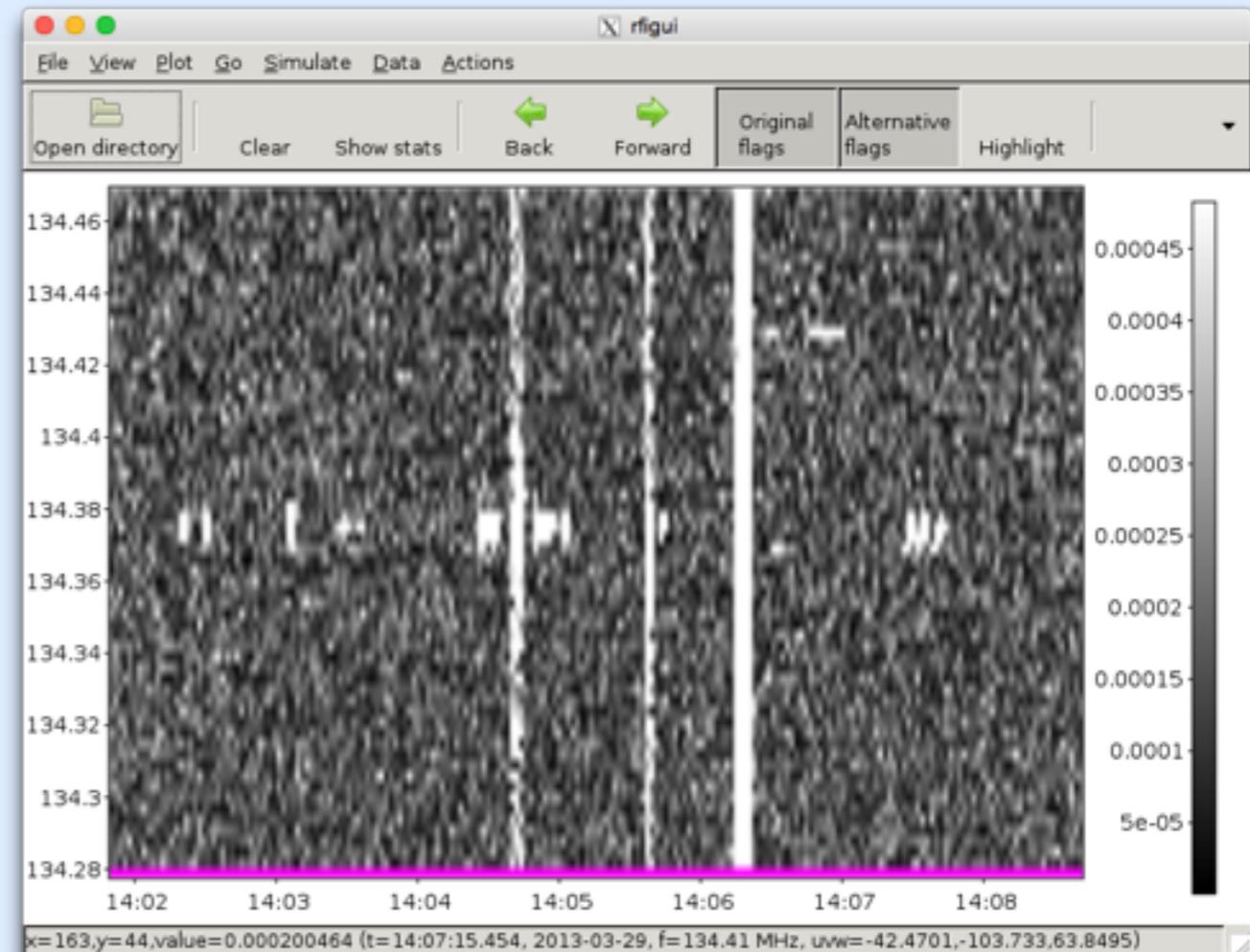
Create a power spectrum plot (plot menu).

 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 AOFlagger can be called with DPPP.

- > `cd /data/scratch/yourname`
- > `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`

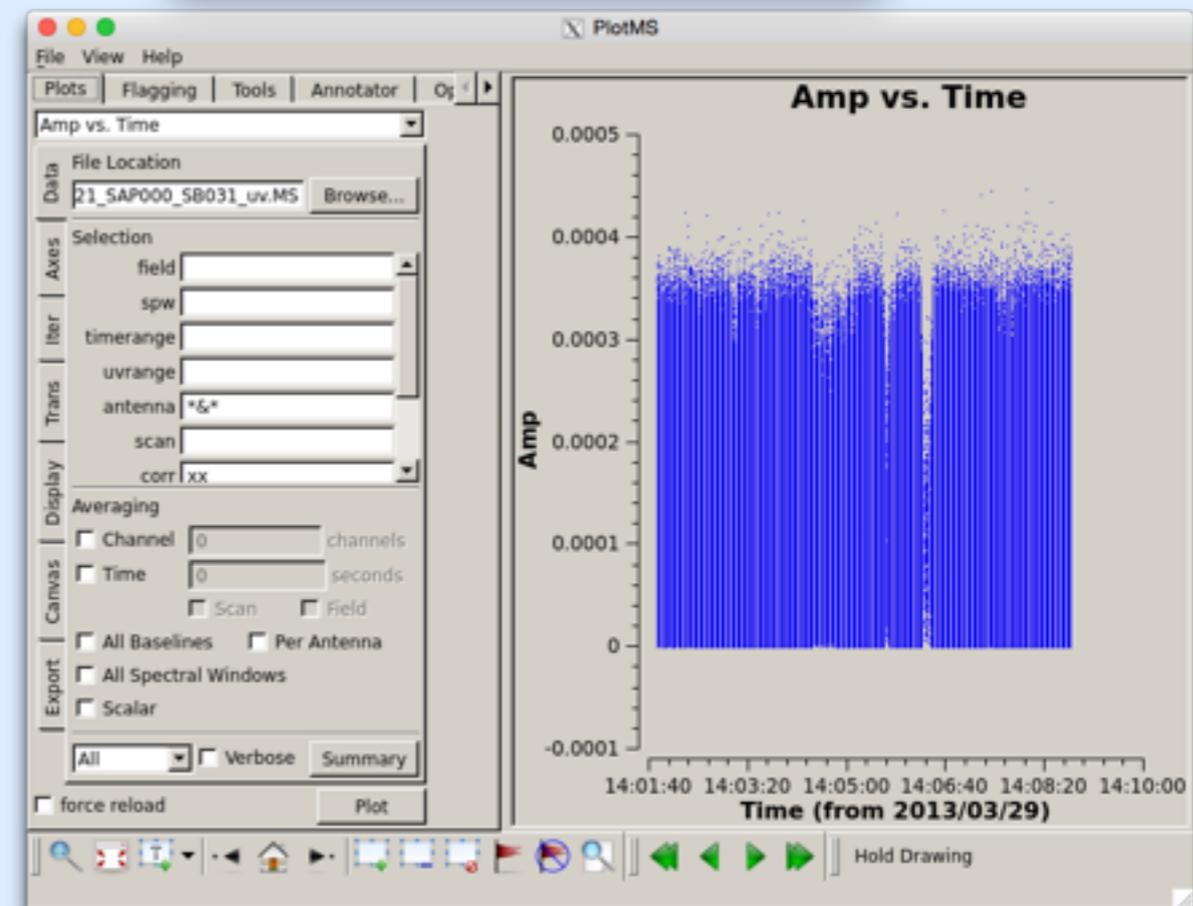
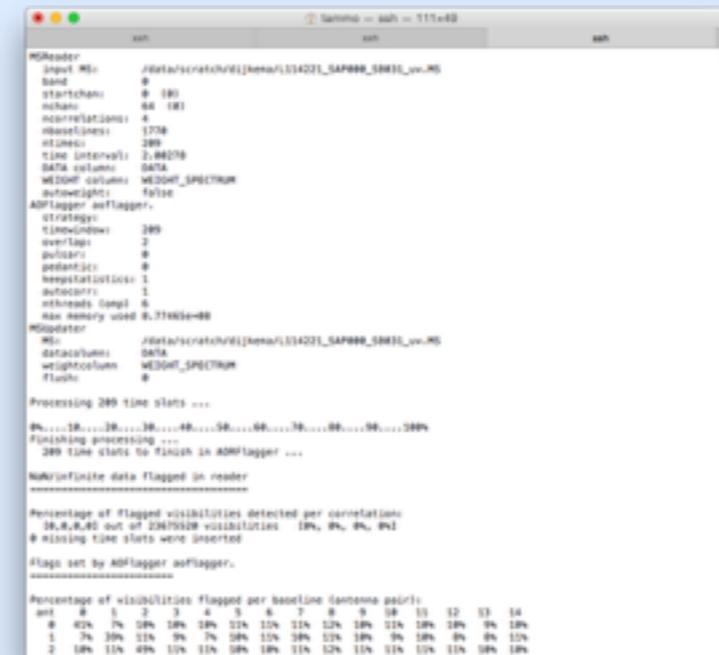
From the output:

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

Re-examine the data with casaplotms

- > `casaplotms`

Does it look reasonable now?



demixing

The observation we were working on was far from bright sources, so demixing is not necessary. Let's work on another one for now, which has been flagged already.

The data is in

`/data/dataschool2014/imaging/demix/`

- ★ Find out the pointing with `msoverview`
- ★ Which A-team sources would interfere?

We will demix this data and average it in time and frequency. We need a model of the A-team sources:

```
scp -r lhd002:/data/dataschool2014/imaging/demix/Ateam.sourcedb .
```

You can verify the contents of the sourcedb with `showsourcedb`:

```
showsourcedb in=Ateam.sourcedb mode=patch
```

Use the following parset for DPPP:

```
msin=/data/dataschool2014/imaging/demix/HBA_L249986_SAP000_SB226_uv.MS
```

```
msout=HBA_L249986_SAP000_SB226_uv_demixed.MS
```

```
steps=[demix]
```

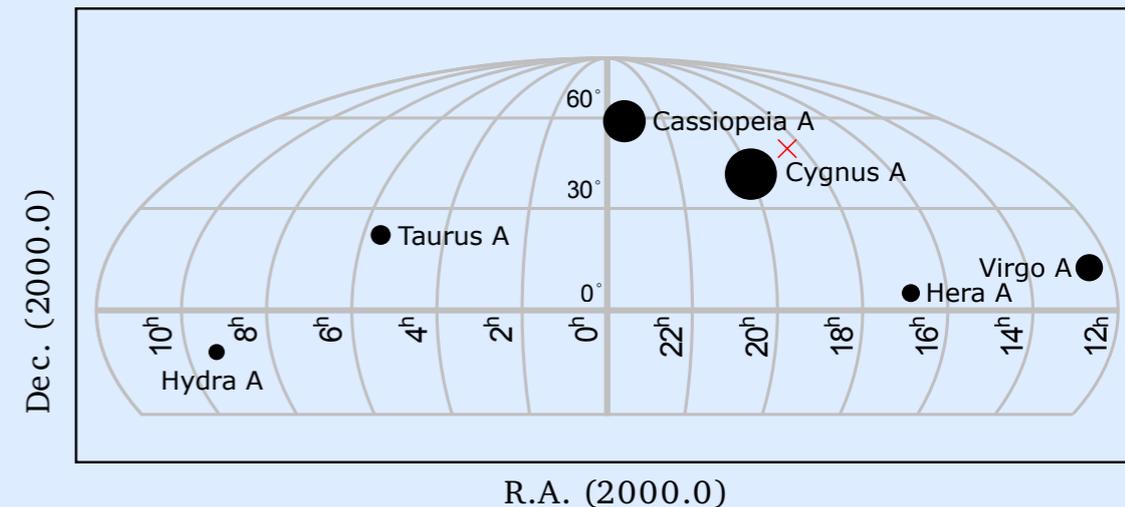
```
demix.subtractsources=[CygA]
```

```
demix.skymodel=Ateam.sourcedb
```

```
demix.timestep=5
```

```
demix.freqstep=8
```

- ★ How much did we decrease the data in size? What did you expect?
- ★ Have a look with `casaplotms` at data before and after. Did we succeed in demixing?



averaging

Back to our original reduction (no demix).

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=[averager]
```

```
averager.timestep=5
```

```
averager.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
```

Congratulations on reducing your first data set! Only 179 to go :-)

We are not going to do this by hand.

Write a script in your favorite language (which should be bash or python) that processes a number of subbands.

Not all data are available on the compute nodes (1of0XX) yet, you can copy them from the head node as 1hd002 follows:

```
> scp -r 1hd002:/data/dataschool2014/imaging/target/L114221_SAP000_SB031_uv.MS .
```

Tip: you can override keys in your parset by specifying them on the command line:

```
DPPP reduction.parset msin=another.MS msout=another_avg.MS
```

A possible solution in bash:

```
for inputname in $(ssh 1hd002 'ls /data/dataschool2014/imaging/target'); do
  echo "Copying ${inputname} from head node to working node"
  scp -rq 1hd002:/data/dataschool2014/imaging/target/${inputname} .
  outputname=$(echo ${inputname} | sed "s/\.MS/_processed.MS/")
  echo ${outputname}
  DPPP DPPP-all.parset msin=${inputname} msout=${outputname}
  echo "Removing raw data ${inputname} from working node"
  rm -rf ${inputname}
done
```