Tutorial T2:
Data Inspection and Flagging

Andy Biggs

ALMA Regional Centre,

European Southern Observatory

Preliminaries

- Move to a directory where you intend to work
  - Should have plenty of space
  - Dataset we'll be working on is 1.6 GB in size
- Download data, script and final list of flags
  - You've already done this, right...?
- Tutorial script contains skeletal CASA commands
  - Tips and hints as well
  - Goal is to complete script

Start CASA

- Current data-reduction package for radio astronomy
  - Common Astronomy Software Applications
- On the command line type:
  - **`casa`** (previously was **`casapy`**)
- After a short while, this brings up:
  - An iPython prompt
    - All Python functionality is available in CASA
    - e.g. **`range`** command, numpy, loops, etc.
  - A logger window
    - Useful information is reported here by each task

Locate the data

- List directory from within CASA
  - Can use usual unix commands i.e. '**ls**'
  - Often requires a '**!**' e.g. '**!echo $HOME**'
- The dataset we will use is **`all_avg.ms/`**
  - This is a directory!
    - **`.ms`** = Measurement Set (MS)
    - Contains tables and visibility data
    - Rarely have to worry about contents of MS
- CASA works directly on this MS
  - One doesn't load the data into CASA

Running CASA tasks (1)

- Set inputs individually
  - **`inp taskname`**
  - **`parameter1=x`**
  - **`parameter2=y`**
  - **`inp`** (to review)
  - **`go`**
- Parameter names can be tab-completed
  - Avoids typos and creation of useless variables
- **`tget taskname`**
  - Returns previous inputs (stored in **`.last`** file)

Running CASA tasks (2)

- Use a “one-liner”
  - **`taskname(parameter1=x,parameter2=y)`** or…
  - **`taskname(x,y)`** (if x and y are the first two inputs)
- Assemble one-liners in a python script
  - **`execfile('t2_flagging.py')`**
  - It is generally a good idea to do this
    - Allows all processing steps to be easily repeated
    - Reminder of what you did

Time to start looking at the data!

View summary of the data

- Relevant task is **`listobs`**
- Summary information includes:
  - Which sources are included in the file
  - Spectral window (spw) properties
    - Bandwidth, central frequency, number of channels, correlations
  - Which antennas were in the array
  - Observation sequence
    - Which source with which spectral windows when?
- Spectral windows
  - Spectral windows are selected via a (0-based) index
  - Non-science spws often included e.g. ALMA

`listobs` output

```
Fields: 5
  ID   Code Name                RA               Decl           Epoch        nRows
  0         1302+5748           13:02:52.465277 +57.48.37.60932 J2000       204900
  1         1252+5634           12:52:26.285900 +56.34.19.48800 J2000       409200
  2    ACAL 1331+305            13:31:08.287300 +30.30.32.95900 J2000        40500
  3         1407+284            14:07:00.394410 +28.27.14.68990 J2000        38700
  4         0319+415            03:19:48.160110 +41.30.42.10330 J2000        54000
Spectral Windows:  (4 unique spectral windows and 1 unique polarization setups)
  SpwID  Name   #Chans   Frame   Ch0(MHz)  ChanWid(kHz) TotBW(kHz) CtrFreq(MHz)  Corrs
  0      none       64   TOPO    4817.000      2000.000    128000.0   4880.0000   RR  RL  LR  LL
  1      none       64   TOPO    4945.000      2000.000    128000.0   5008.0000   RR  RL  LR  LL
  2      none       64   TOPO    5073.000      2000.000    128000.0   5136.0000   RR  RL  LR  LL
  3      none       64   TOPO    5201.000      2000.000    128000.0   5264.0000   RR  RL  LR  LL
```

3C277.1

`listobs` output

```
Fields: 5
  ID   Code Name                RA               Decl           Epoch        nRows
  0         1302+5748           13:02:52.465277 +57.48.37.60932 J2000       204900
  1         1252+5634           12:52:26.285900 +56.34.19.48800 J2000       409200
  2    ACAL 1331+305            13:31:08.287300 +30.30.32.95900 J2000        40500
  3         1407+284            14:07:00.394410 +28.27.14.68990 J2000        38700
  4         0319+415            03:19:48.160110 +41.30.42.10330 J2000        54000
Spectral Windows:  (4 unique spectral windows and 1 unique polarization setups)
  SpwID  Name   #Chans   Frame   Ch0(MHz)  ChanWid(kHz)  TotBW(kHz) CtrFreq(MHz)  Corrs
  0      none       64   TOPO    4817.000       2000.000    128000.0   4880.0000   RR  RL  LR  LL
  1      none       64   TOPO    4945.000       2000.000    128000.0   5008.0000   RR  RL  LR  LL
  2      none       64   TOPO    5073.000       2000.000    128000.0   5136.0000   RR  RL  LR  LL
  3      none       64   TOPO    5201.000       2000.000    128000.0   5264.0000   RR  RL  LR  LL
```

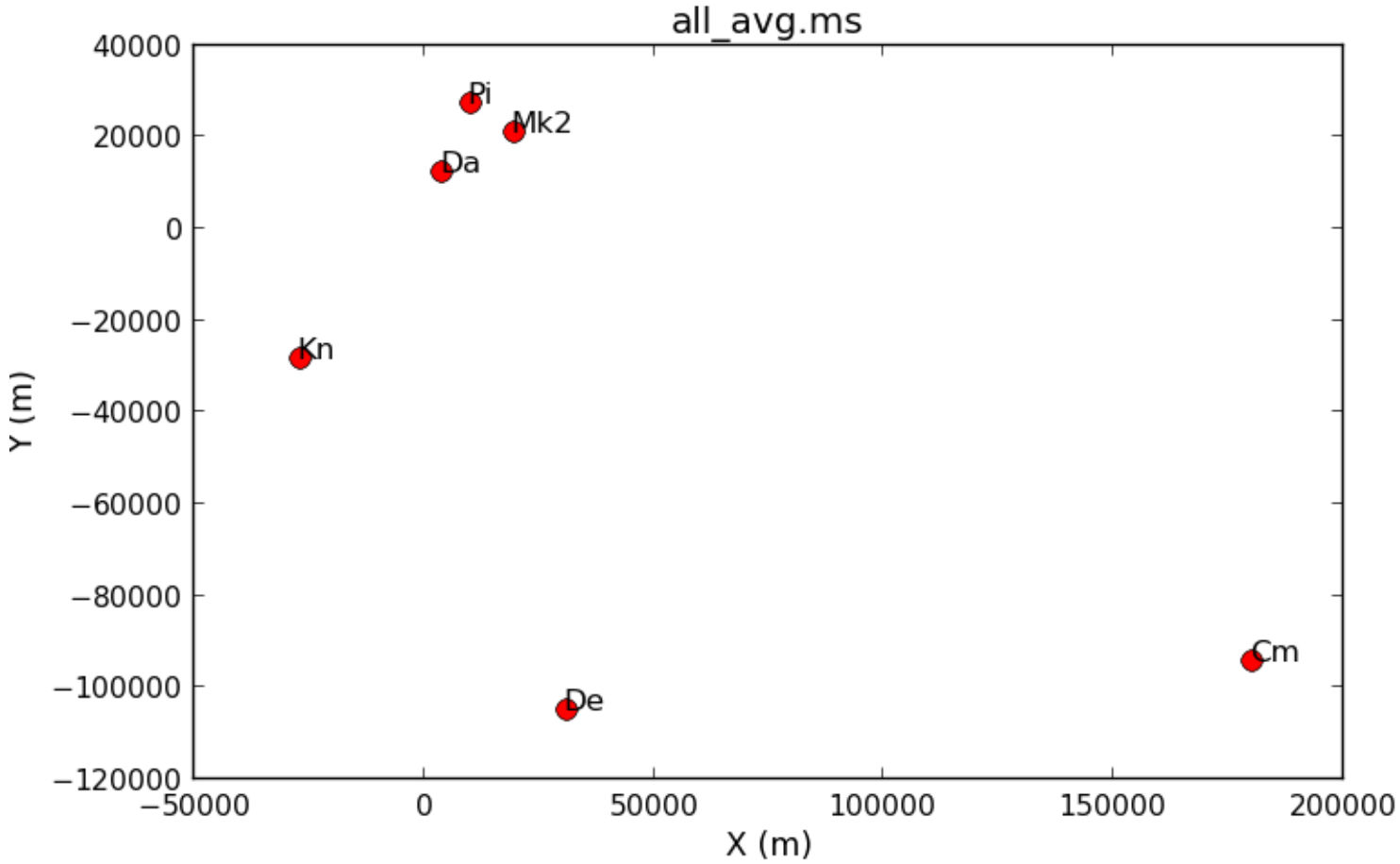
`listobs` output

```
Fields: 5
  ID   Code Name                RA               Decl           Epoch        nRows
  0         1302+5748           13:02:52.465277 +57.48.37.60932 J2000       204900
  1         1252+5634           12:52:26.285900 +56.34.19.48800 J2000       409200
  2    ACAL 1331+305            13:31:08.287300 +30.30.32.95900 J2000        40500
  3         1407+284            14:07:00.394410 +28.27.14.68990 J2000        38700
  4         0319+415            03:19:48.160110 +41.30.42.10330 J2000        54000
Spectral Windows:  (4 unique spectral windows and 1 unique polarization setups)
  SpwID  Name   #Chans   Frame   Ch0(MHz)  ChanWid(kHz) TotBW(kHz) CtrFreq(MHz)  Corrs
  0      none      64   TOPO    4817.000      2000.000    128000.0   4880.0000   RR  RL  LR  LL
  1      none      64   TOPO    4945.000      2000.000    128000.0   5008.0000   RR  RL  LR  LL
  2      none      64   TOPO    5073.000      2000.000    128000.0   5136.0000   RR  RL  LR  LL
  3      none      64   TOPO    5201.000      2000.000    128000.0   5264.0000   RR  RL  LR  LL
```

View antenna locations

- Relevant task is **`plotants`**
- Shows 2-D map of antenna positions
- Interesting information includes:
  - Size of the array
  - Antenna distribution
- Often used to select reference antenna
  - Choose one with a large range of baseline lengths
  - For e-MERLIN, ‘Mk2’ is usually used

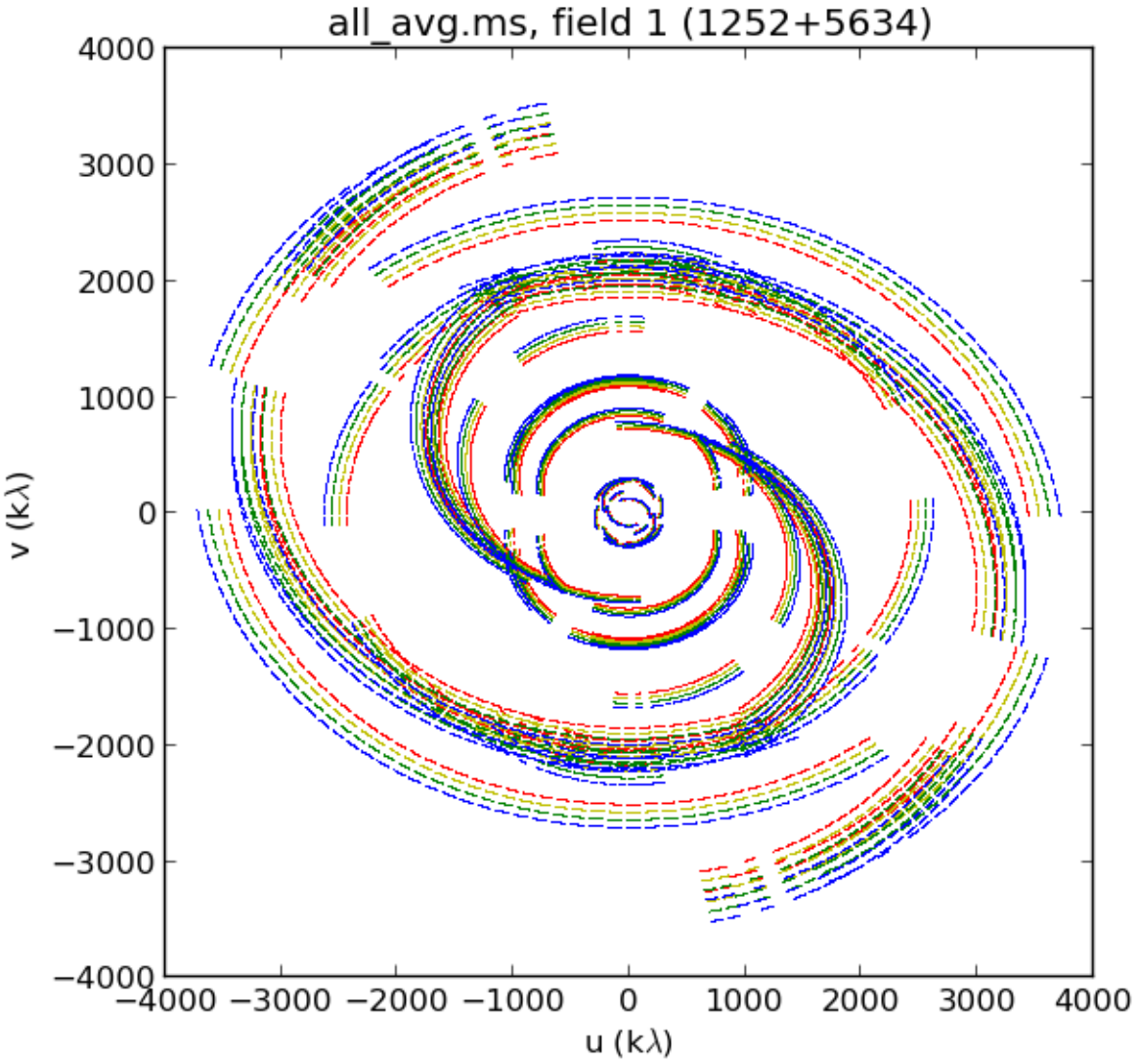
`plotants` output



View (*u,v*) coverage

- Relevant task is **`plotuv`**
- Shows projected baseline for each visibility
  - *w* coordinate is not shown (obviously)
- Complete coverage (no gaps) would be nice...
  - Incomplete coverage produces image sidelobes
  - The bigger the gaps, the bigger the sidelobes
- Wide-bandwidths help a lot
  - Separate track for each frequency point
  - $(u, v) = B / \lambda$

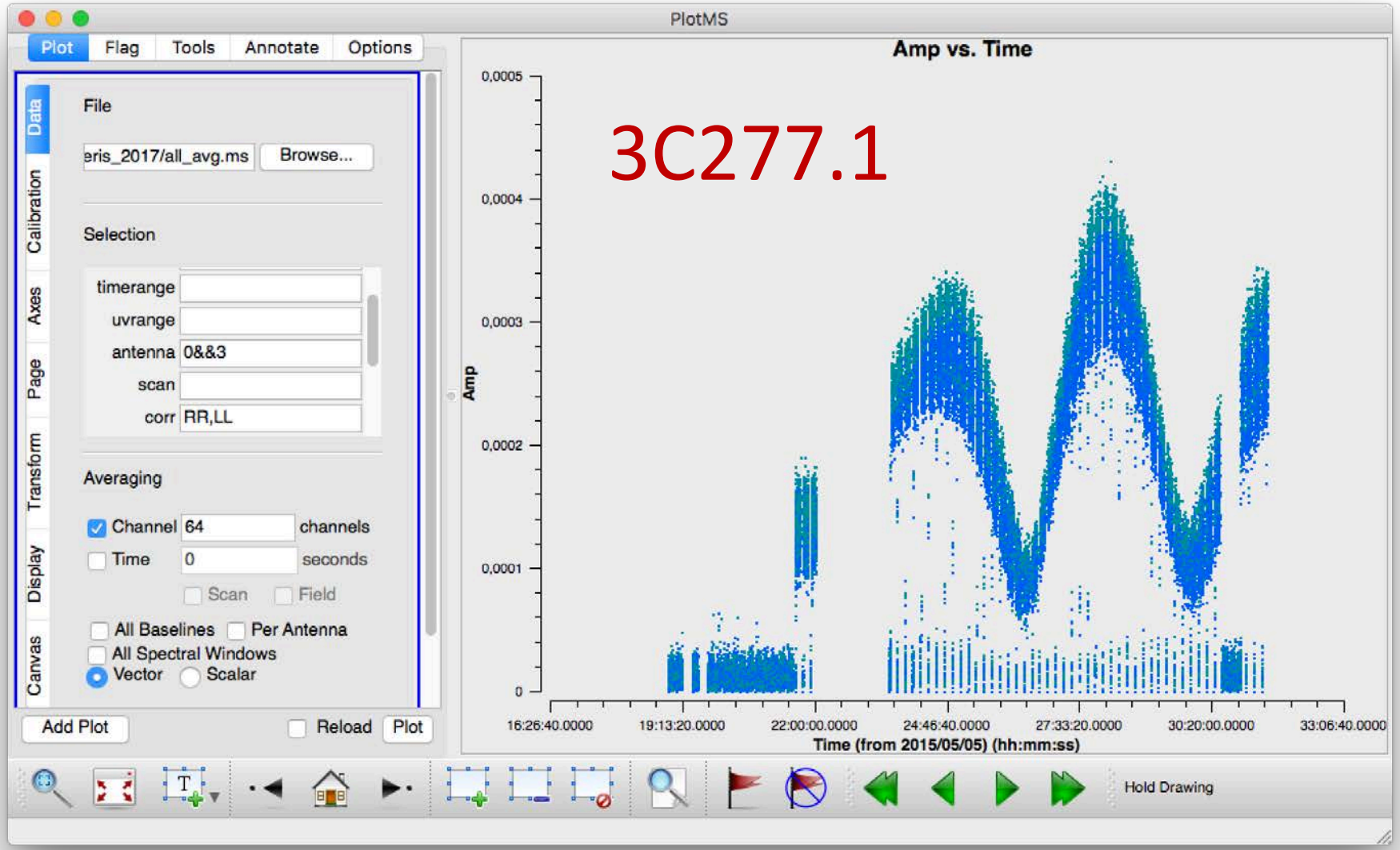
plotuv output



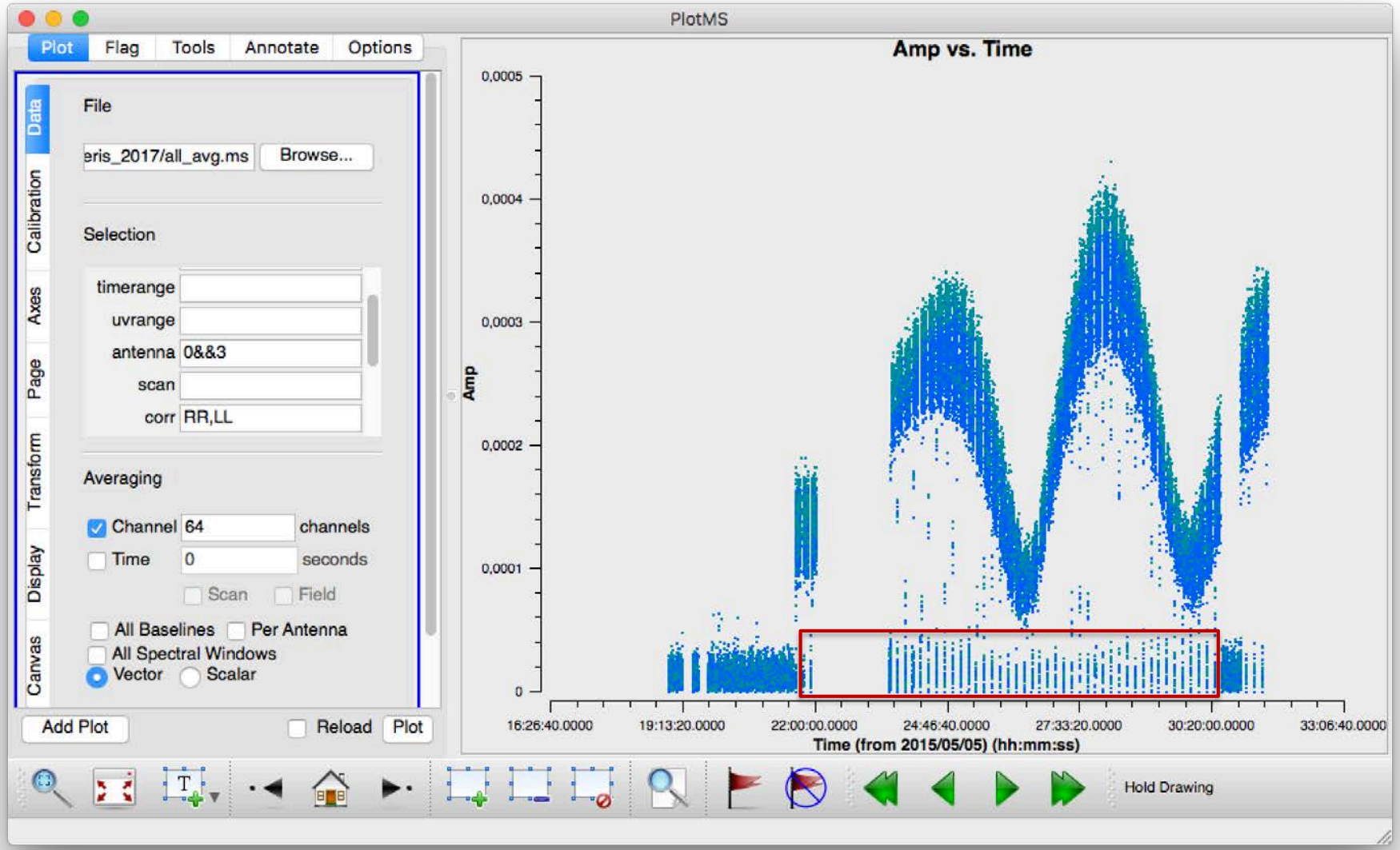
View data

- Relevant task is **`plotms`**
  - 2-D data plotter e.g. amplitude against frequency
- **`plotms`** is a very powerful task
  - Data can be selected on almost every property
    - Antenna, baseline, timerange, spw, correlation, ...
  - Data can be displayed in many different ways
    - >50 axis possibilities (time, frequency, phase, amplitude, ...)
- Very useful for:
  - Visualising data
  - Finding bad data

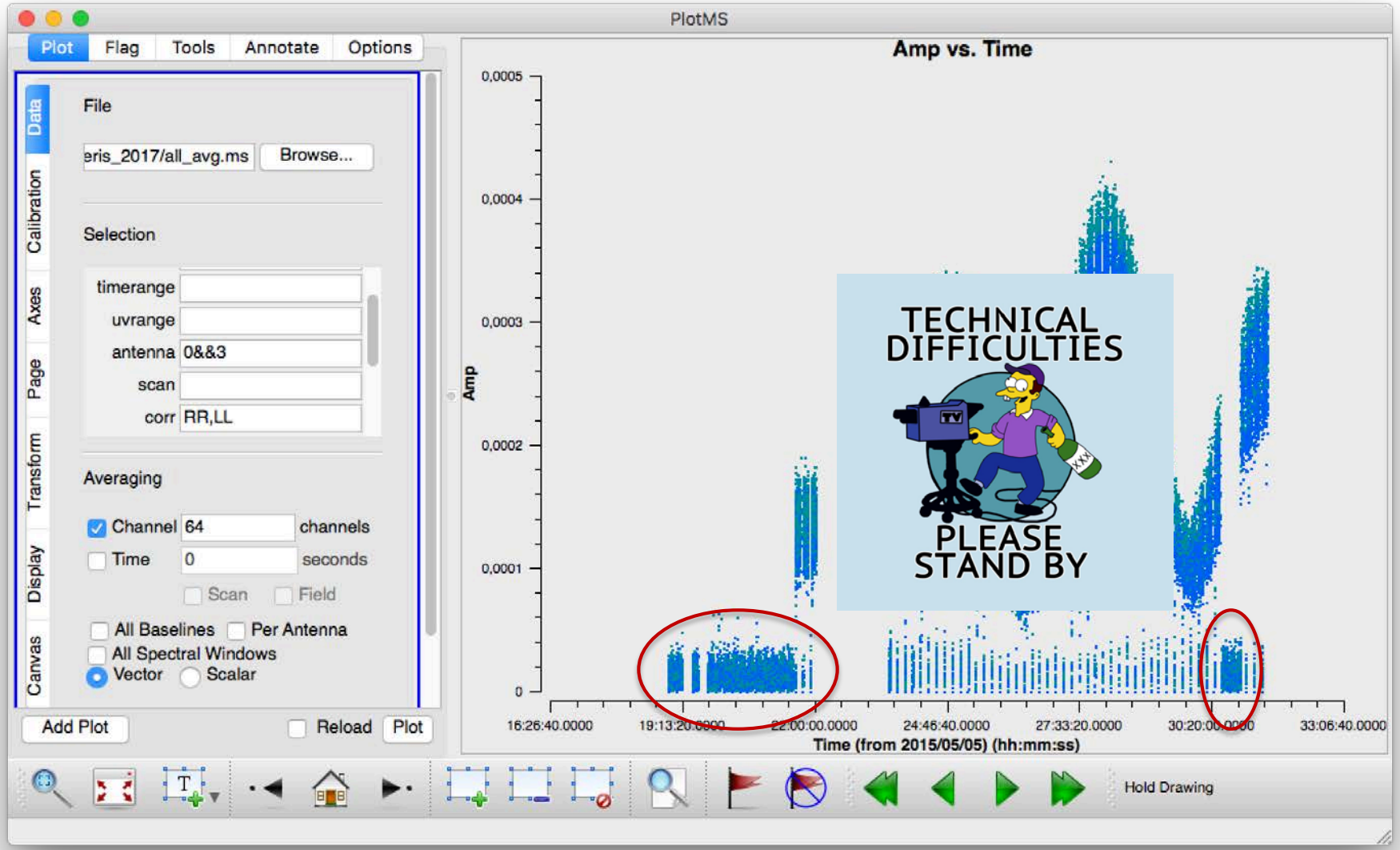
Example **plotms** output



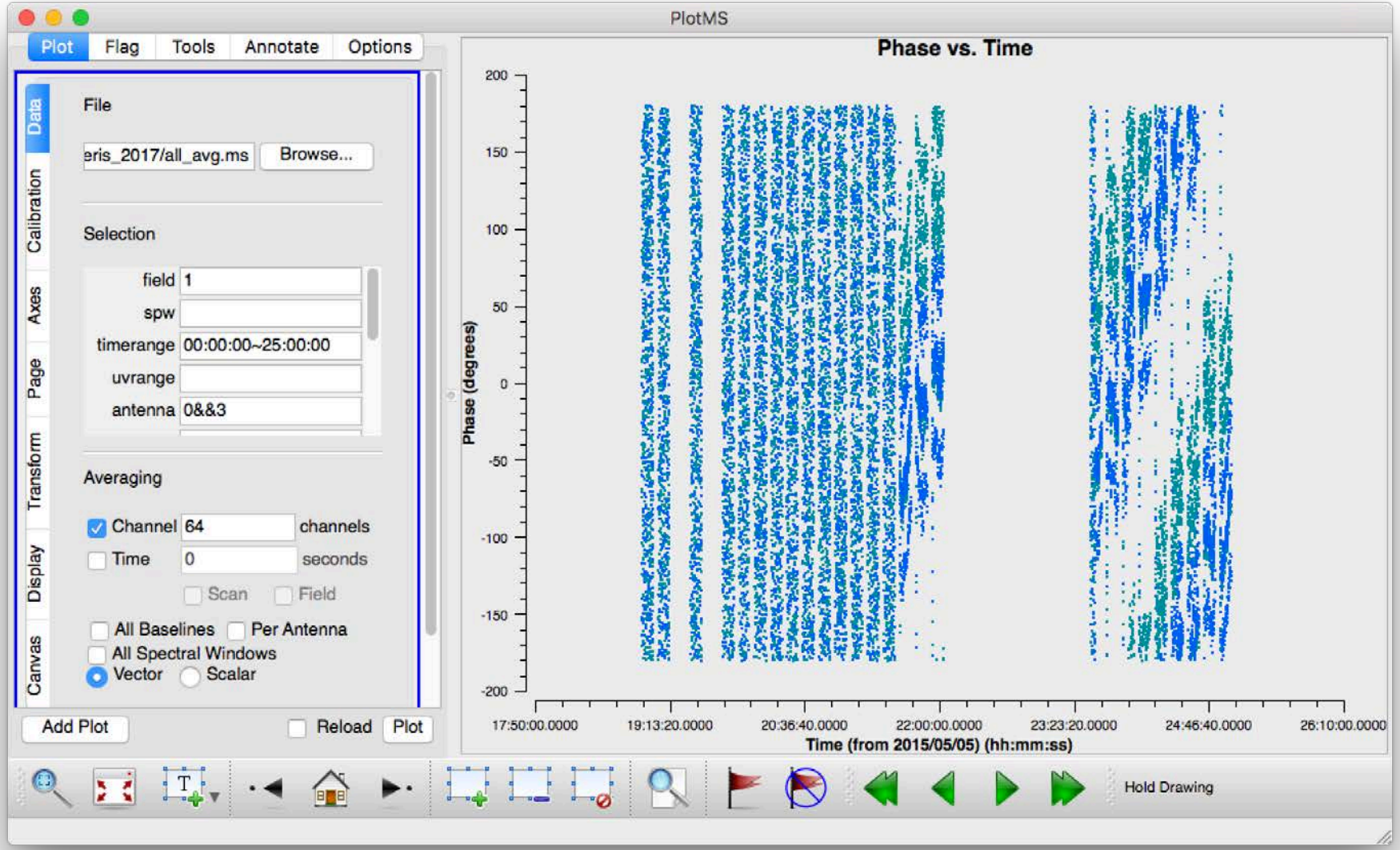
Data to be ‘quacked’



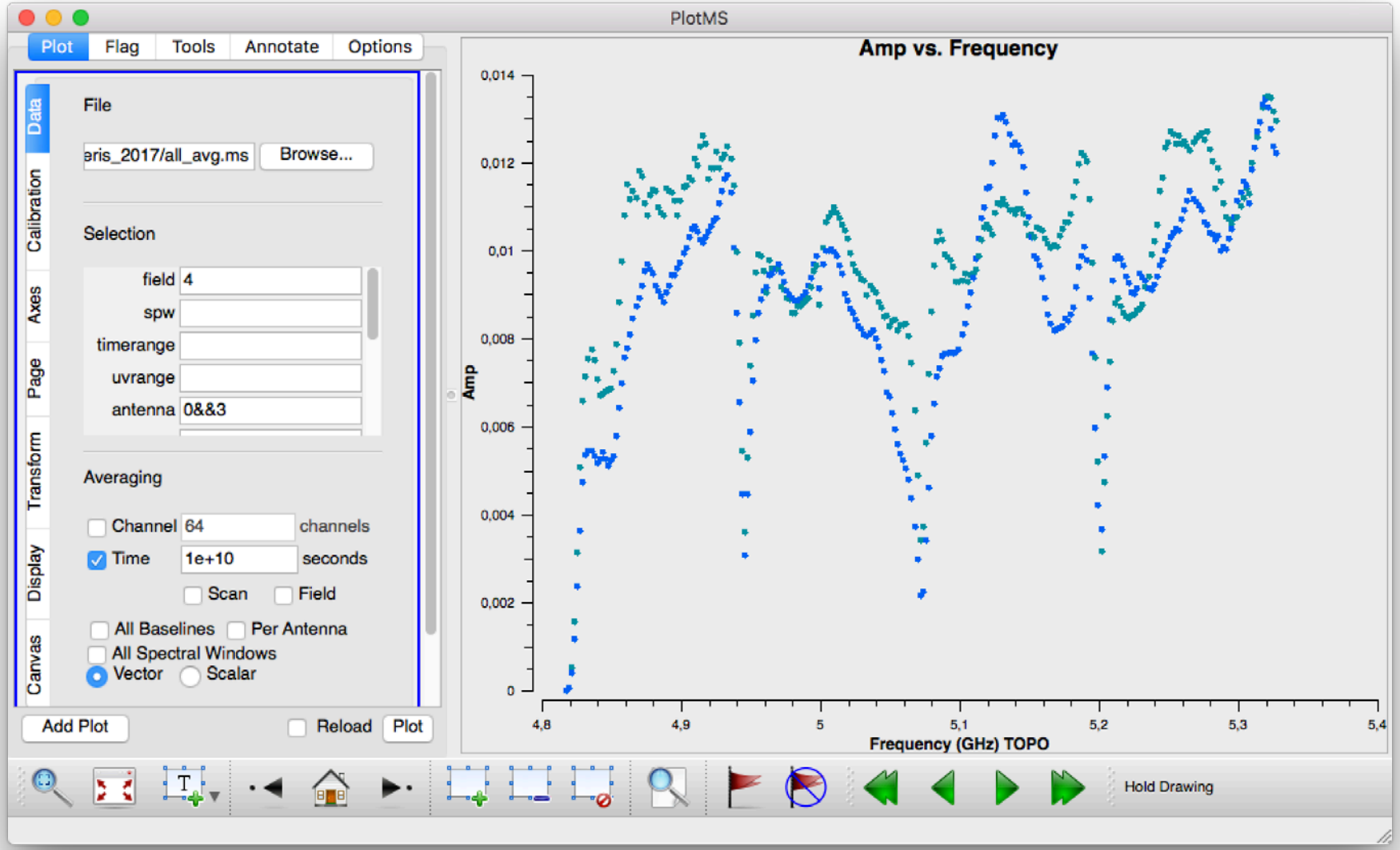
When radio telescopes go bad!



Noisy phase of bad data



Noisy channels



Flagging

- Some data normally needs to be deleted ("flagged")
- Typical reasons include:
  - Data taken whilst telescopes are off-source (slewing)
    - Many telescopes now flag this automatically
  - Edge channels of the spectral windows
    - Amplitude of these is usually lower
  - An antenna is not working very well or at all
    - Problem may be time-variable
    - May only affect some spectral windows
- Each problem is usually antenna-based and affects all correlations

Flag versions

- Relevant task is **`flagmanager`**
- Two main modes of operation:
  - Save current list of flags to a named version
    - Do this regularly!
  - Replace the current flags with those from a named version
    - In the unlikely event that you screw up!
- Versions are stored in a **`.flagversion`** directory
- Note the appearance of the '**mode**' parameter
  - Inputs change depend on the selected mode

Flag data

- Relevant task is **`flagdata`**
- In ‘manual’ mode can flag extremely flexibly i.e. by
  - Field (source)
  - Spectral window (including sub-channels)
  - Antenna
  - Timerange
- Follow the guidelines in the script and flag the bad data
- We will not be able to do all the necessary flagging
  - The aim is to familiarise you with the process

Final flagging

- All the flags are included in ‘all_avg_1.flags’
  - You should download this (see script)
- Apply these using **`flagdata`**
  - Use **`mode = ‘list’`**

The data are now ready to be calibrated!
(Tutorial 4 on Tuesday)

RadioNet has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 730562