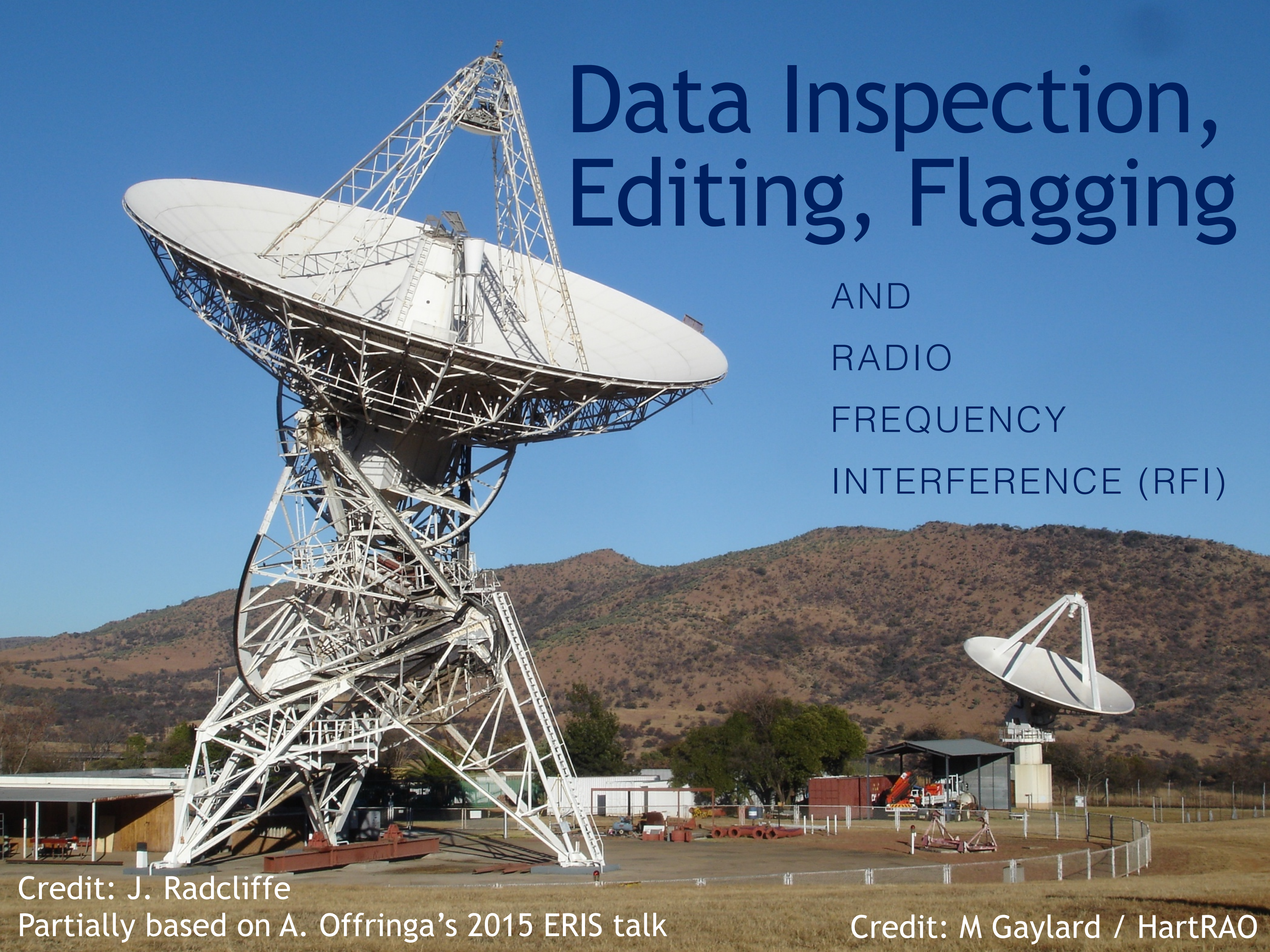
Data Inspection, Editing, Flagging

AND

RADIO

FREQUENCY

INTERFERENCE (RFI)



Credit: J. Radcliffe

Partially based on A. Offringa's 2015 ERIS talk

Credit: M Gaylard / HartRAO

Introduction

Why do we need to edit data??

Radio sources are extremely weak!

Remember 1Jy = 10^{-26} Watts per sq. metre per Hz !

- Brightest sources are ~few 10s Jansky
- Mobile phones at 1km away are 10^8 brighter…
- Protected radio bands are exceeded by our bandwidths!
- Satellites love to operate at useful radio frequencies.

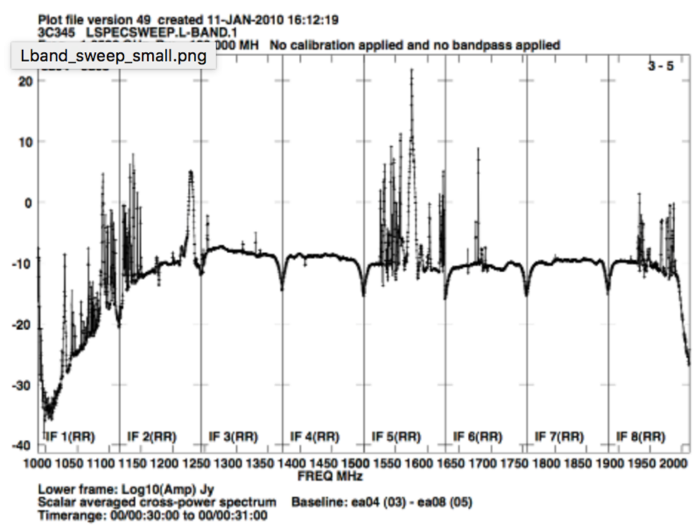
This is known as **R**adio **F**requency **I**nterference (RFI) and we need to remove it

Jansky (Jy)	Source
110,000,000	Radio-frequency interference from a GSM phone transmitting 0.5 W at 1800 MHz at a distance of 1 km (RSSI of -70 dBm)[8]
20,000,000	Disturbed Sun at 20 MHz (Karl Guthe Jansky's initial discovery, published in 1933)
4,000,000	Sun at 10 GHz
1,000,000	Milky Way at 20 MHz
10,000	1 Solar flux unit
2,000	Milky Way at 10 GHz
1,000	Quiet Sun at 20 MHz

Introduction

But that’s not all!

- Broken elements (antennas/stations)
- Correlator malfunctions
- Shadowing
- Initial pointing delay
- Bandpass issues
- Low elevation
- Correlated noise on some baselines (e.g. LOFAR split stations)

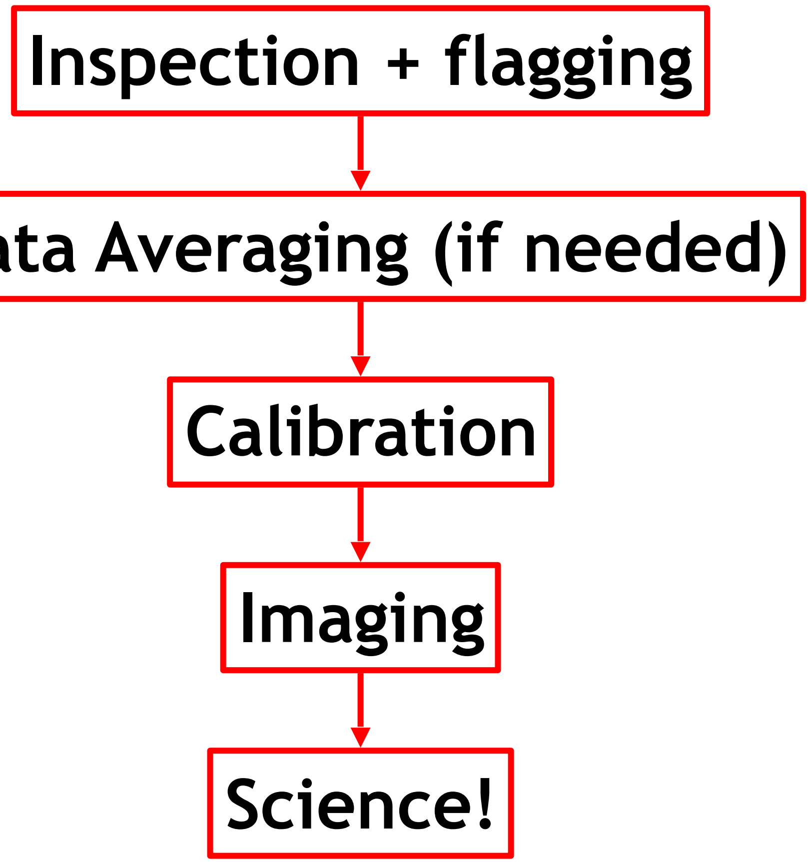


Introduction

Your source is literally a needle in a haystack!

Introduction

Data **cannot** be (self-)calibrated when any of these issues are still in the data.



Introduction

This talk:

- Radio Frequency Interference (RFI)
- Data inspection
- Manual data flagging (how to remove sources of bad data)
- Automatic RFI flagging algorithms
- Data averaging
- Sources of imaging errors

Radio Frequency Interference

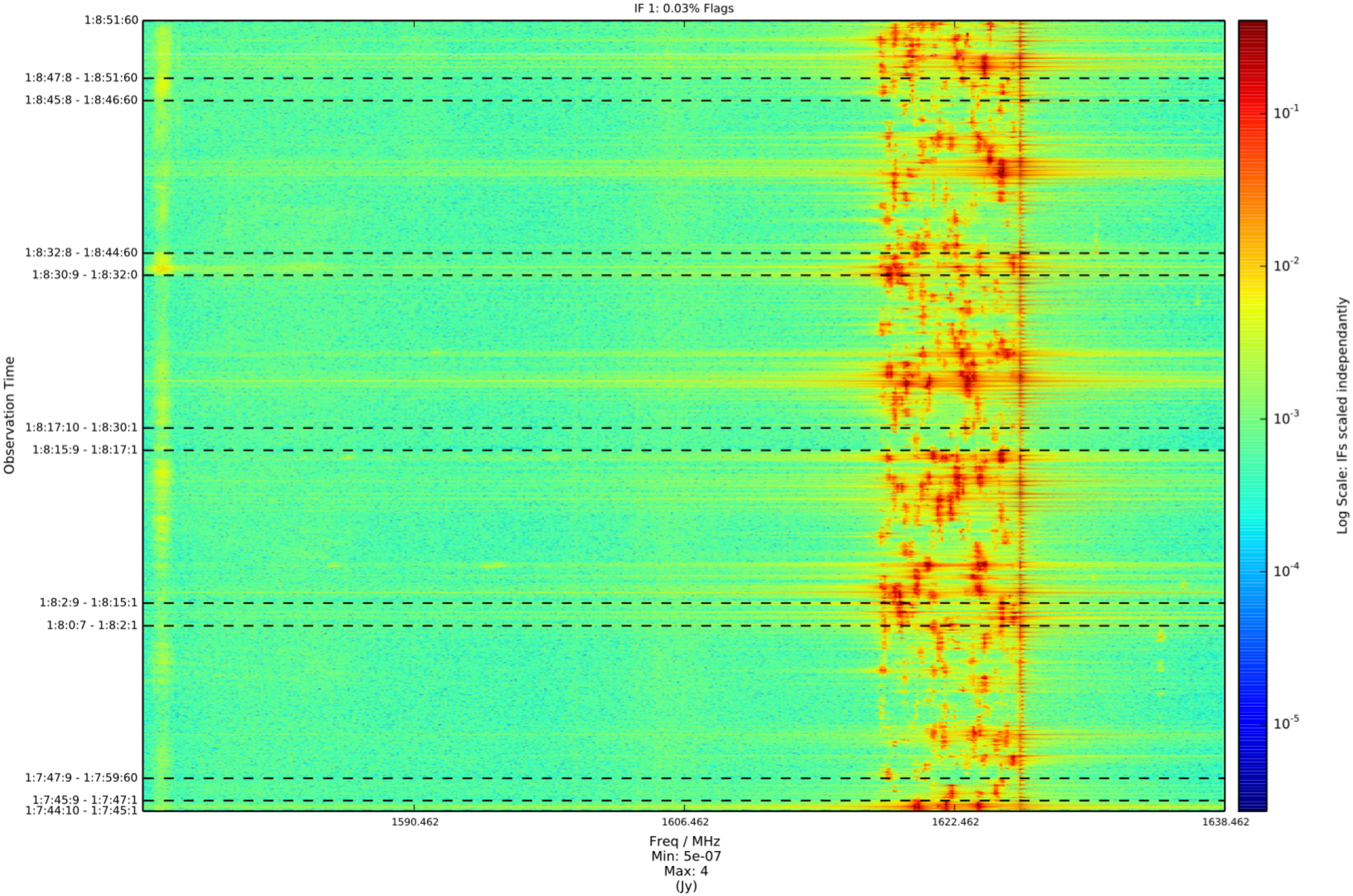
- Radio Frequency Interference (RFI) is a major problem for nearly all radio telescopes.
- Advantage: target source is often much much lower in flux than the RFI source
- Can be generated internally or externally.
- Lots of interference at low frequencies (<1.5 GHz, e.g. LOFAR, GMRT, WSRT, EVLA, MWA, ...)
- Less of an issue for higher frequencies (ALMA); or VLBI but mitigation still required in most cases.

Table 10: Identified VLA RFI Between 1 and 4 GHz

Frequency (MHz)	Source	Comments
1025-1150	Aircraft navigation	Very strong
1200.0	VLA modem	
1217-1237	GPS L2	Very strong
1243-1251	GLONASS L2	
1254	Aeronautical radar	
1263	Aeronautical radar	
1268	COMPASS E6	
1310	Aeronautical radar	
1317	Aeronautical radar	
1330	Aeronautical radar	
1337	Aeronautical radar	
1376-1386	GPS L3	Intermittent
1525-1564	INMARSAT satellites	
1564-1584	GPS L1	Very strong
1598-1609	GLONASS L1	
1618-1627	IRIDIUM satellites	
1642	2nd harmonic VLA radios	Sporadic
1683-1687	GOES weather satellite	
1689-1693	GOES weather satellite	
1700-1702	NOAA weather satellite	
1705-1709	NOAA weather satellite	
1930-1990	PCS cell phone base stations	
2178-2195	Satellite Downlink	very strong*
2320-2350	Sirius/XM Satellite radio	very strong*
3700-4200	Satellite Downlinks	very strong*

Radio Frequency Interference

This is what a satellite (IRIDIUM) looks like in data:



Radio Frequency Interference

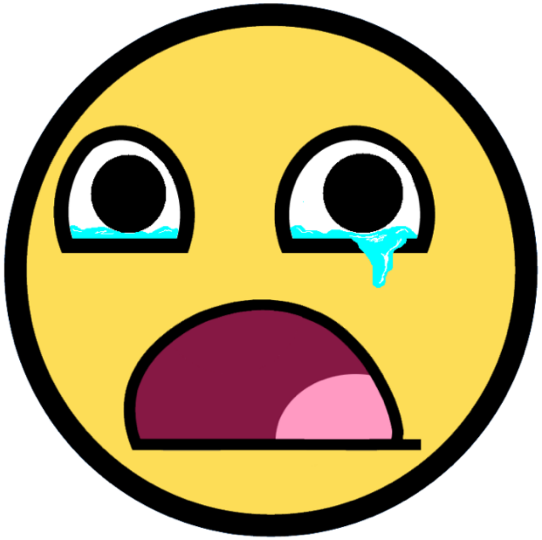
- Many arrays apply flag masks to data **automatically** to remove known RFI affected channels

- Does not catch time varying RFI, therefore manual identification is almost always required (or with RFI detecting algorithm)!

Radio Frequency Interference

- Many arrays apply flag masks to data **automatically** to remove known RFI affected channels

- Does not catch time varying RFI, therefore manual identification is almost always required (or with RFI detecting algorithm)!



Data Inspection

There are many tasks in CASA which help you inspect & search for bad data:

plotms --- create X-Y plots of data in MS and calibration tables, flag data

plotxy --- older X-Y plotter with some functionalities not yet implemented in plotms

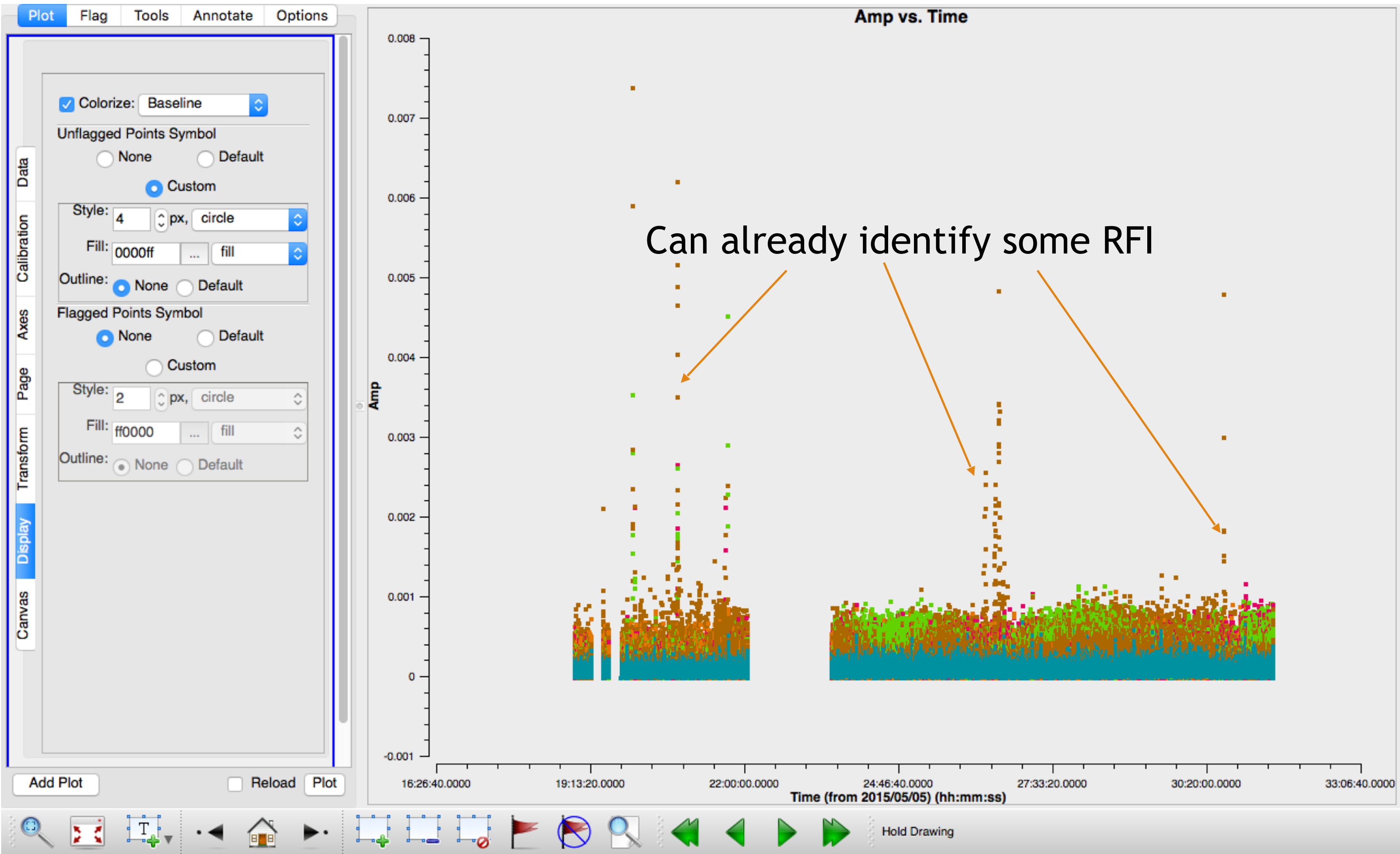
browsetable --- browse data in any CASA table (including a MS)

plotants --- create simple plots of antenna positions (useful only to check shadowing)

casaviewer --- can provide detailed view of the measurement set (fiddly!)

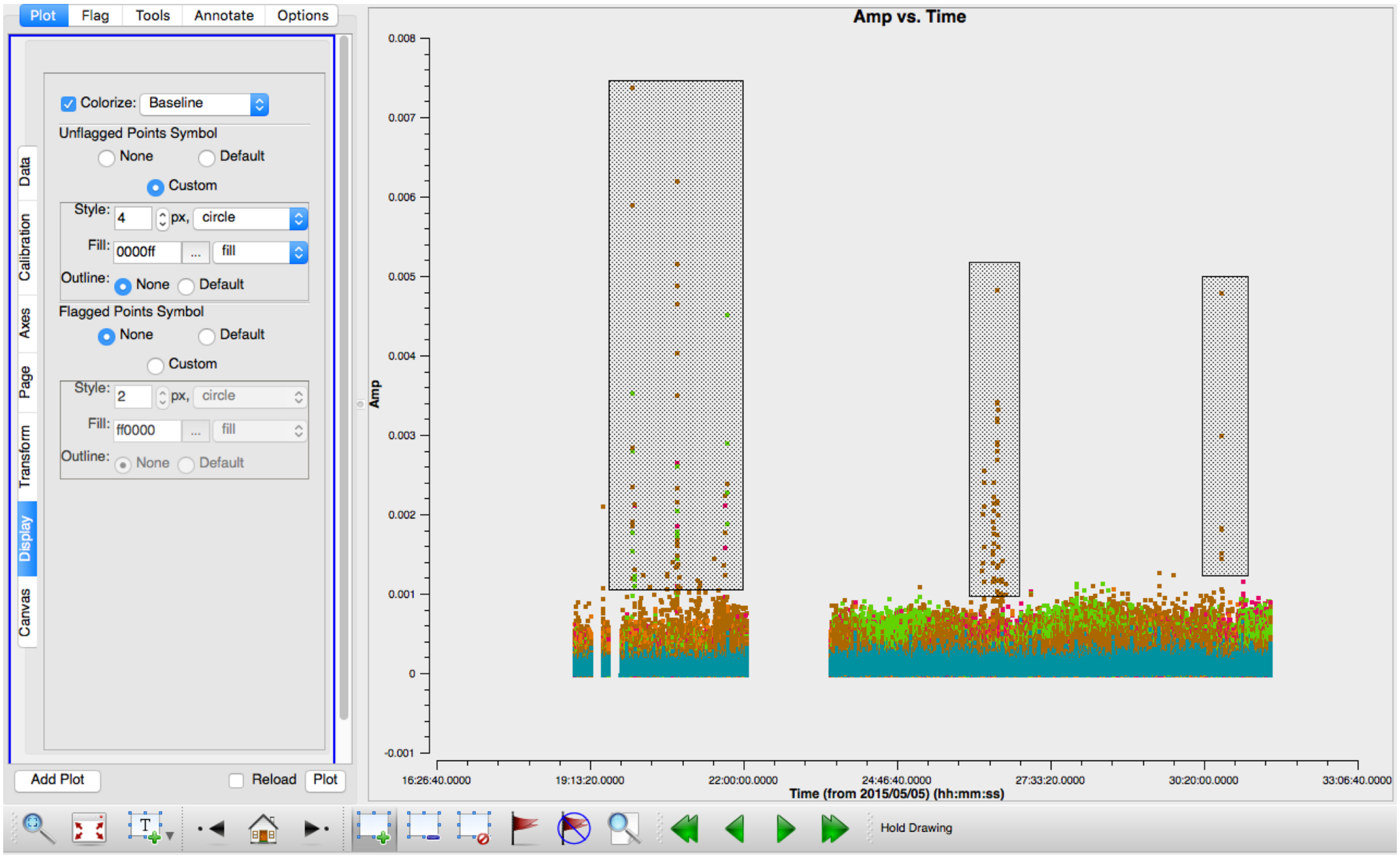
Data Inspection: plotms

plotms - is a very useful tool to identify bad data, gui interface is easy to use!



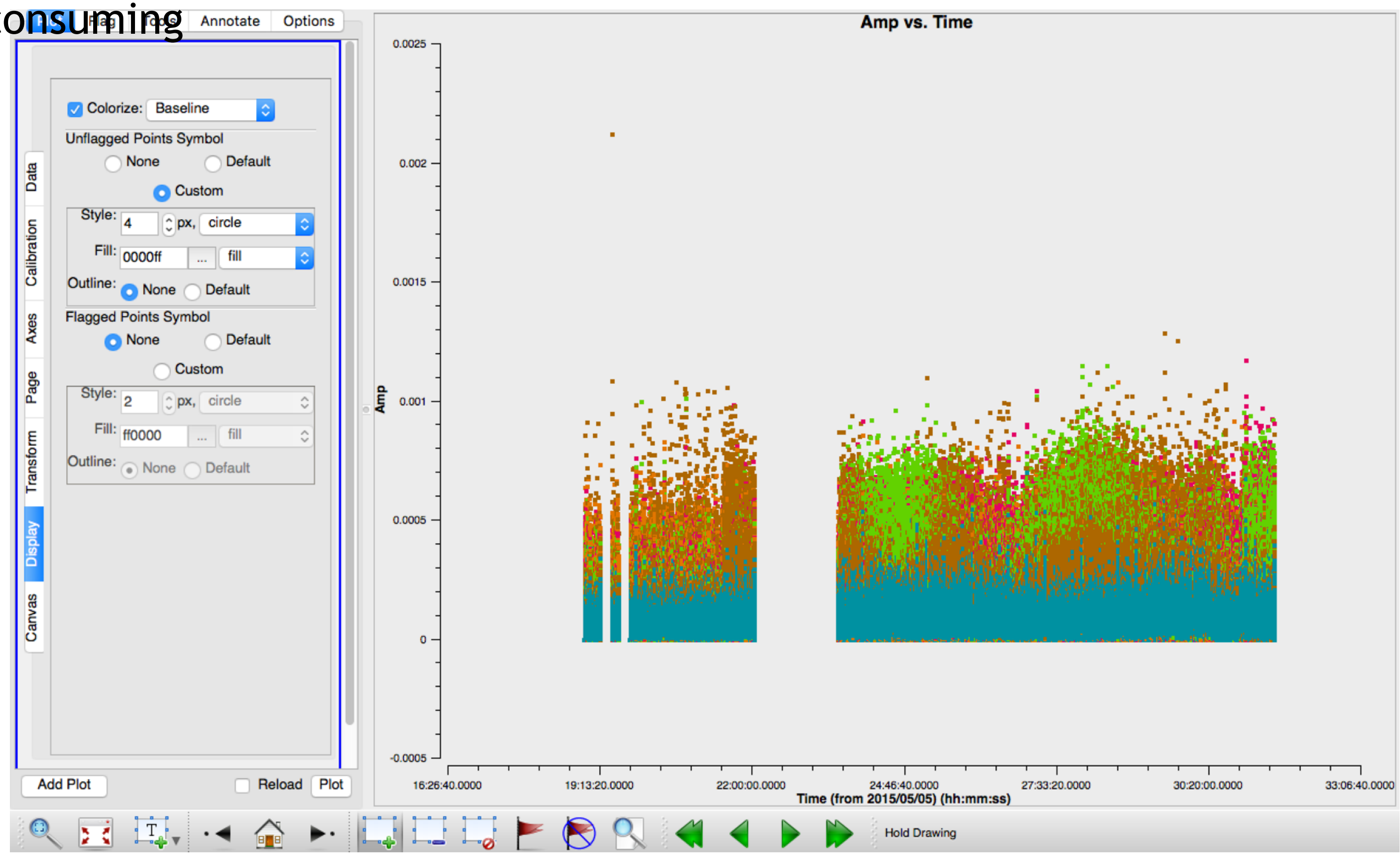
Data Inspection: plotms

Can flag interactively in plotms by using the region + flag buttons



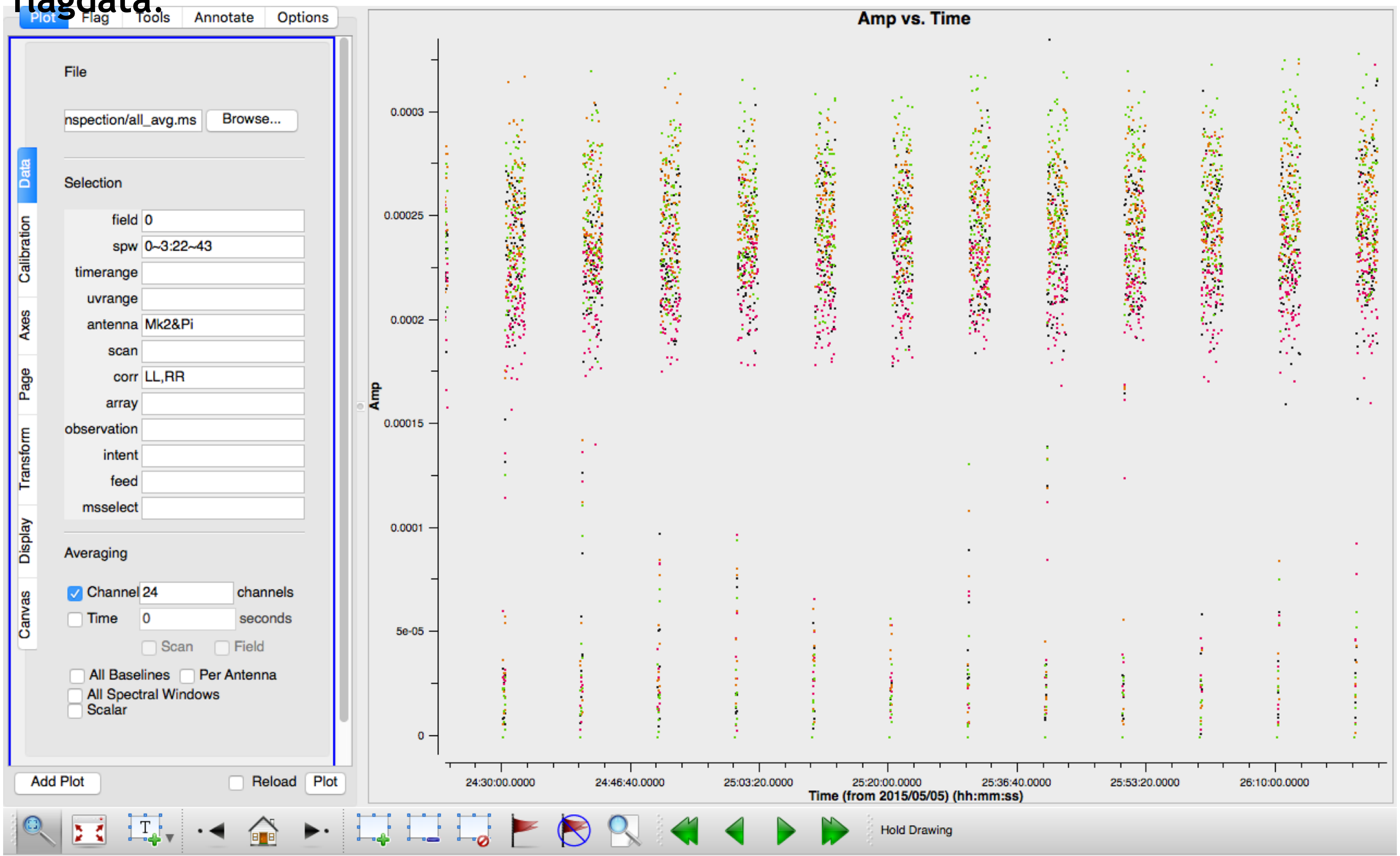
Data Inspection: plotms

- Most of the high amplitude RFI is gone however this is not the most efficient way to flag.
- There is more than just very high flux RFI to remove and can be time consuming



Data Inspection

See a repeating pattern of low fluxes at the start of each scan? We could do this interactively (it could take a while) or we could script it and use **flagdata.**



Data Flagging

flagdata is a task used to flag bad channels, scans, antennas etc etc. in a scriptable way! Has many modes to efficiently flag bad data.

```
#  flagdata :: All-purpose flagging task based on data-selections and flagging modes/algorithms.
vis                 =         ''        #  Name of MS file or calibration table
                                        #   to flag
mode                =   'manual'        #  Flagging mode
     field          =         ''        #  Field names or field index numbers:
                                        #   '' ==> all, field='0~2,3C286'
     spw            =         ''        #  Spectral-window/frequency/channel: ''
                                        #   ==> all, spw='0:17~19'
     antenna        =         ''        #  Antenna/baselines: '' ==> all,
                                        #   antenna ='3,VA04'
     timerange      =         ''        #  Time range: '' ==>
                                        #   all,timerange='09:14:0~09:54:0'
     correlation    =         ''        #  Correlation: '' ==> all,
                                        #   correlation='XX,YY'
     scan           =         ''        #  Scan numbers: '' ==> all
     intent         =         ''        #  Scan intent: '' ==> all,
                                        #   intent='CAL*POINT*'
     array          =         ''        #  (Sub)array numbers: '' ==> all
     uvrange        =         ''        #  UV range: '' ==> all; uvrange
                                        #   ='0~100klambda', default
                                        #   units=meters
     observation    =         ''        #  Observation ID: '' ==> all
     feed           =         ''        #  Multi-feed numbers: Not yet
                                        #   implemented
     autocorr       =      False        #  Flag only the auto-correlations

action              =    'apply'        #  Action to perform in MS and/or in
                                        #   inpfile (none/apply/calculate)
     display        =         ''        #  Display data and/or end-of-MS reports
                                        #   at runtime (data/report/both).
     flagbackup     =       True        #  Back up the state of flags before the
                                        #   run

savepars            =      False        #  Save the current parameters to the
                                        #   FLAG_CMD table or to a file
```

Data Flagging

Two most important parameters here are `mode` and `action`:

`Mode` determines what sort of flagging needs to be performed (explained later):

`Action` has two options ‘apply’ & ‘calculate’:

- If you use apply, the flags will be applied to the MS.
  - Setting display = ‘data’ launches an interactive GUI (v. helpful) and the data flags can be inspected and if unsatisfactory one can exit without applying the flags
  - display = ‘report’ prints flagging statistics (can also set display=‘both’ and it doesn both GUI + statistics

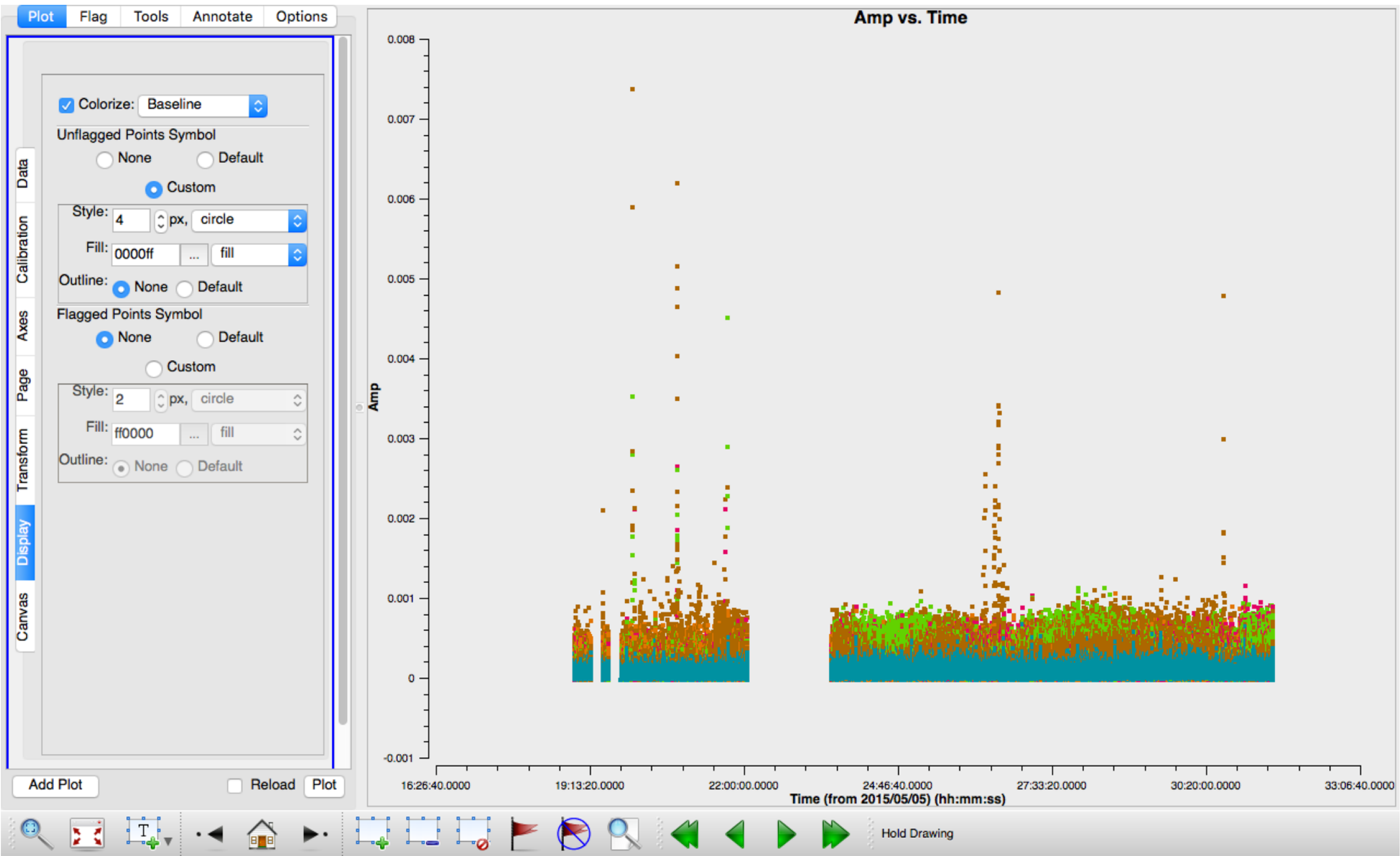
- Use ’calculate’ and the flags are calculated but not written to the MS. Useful if display = ‘data’ so flags can be inspected without straight application

Nb: set flagbackup=True so that the flags are saved to .flagversions file before applying!

```
action          =    'apply'        #  Action to perform in MS and/or in inpfile
                                    #   (none/apply/calculate)
     display    =         ''        #  Display data and/or end-of-MS reports at runtime
                                    #   (data/report/both).
     flagbackup =       True        #  Back up the state of flags before the run
```

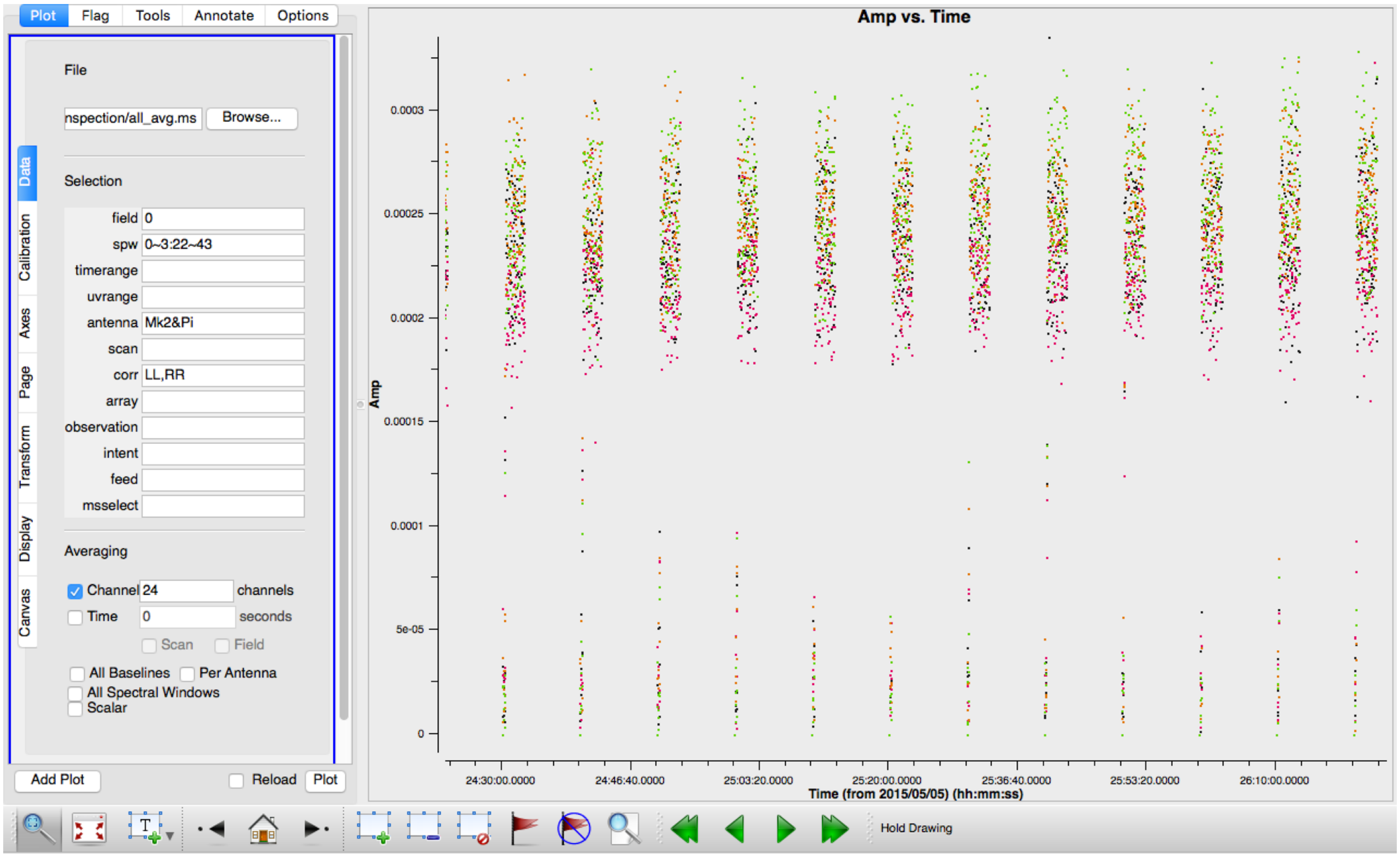
Data Flagging

Returning to this data we can find where the bad data is contained and manually flag it using **flagdata**. This way we ensure that if something should happen to the data we can recover the flags!



Data Flagging: flagdata

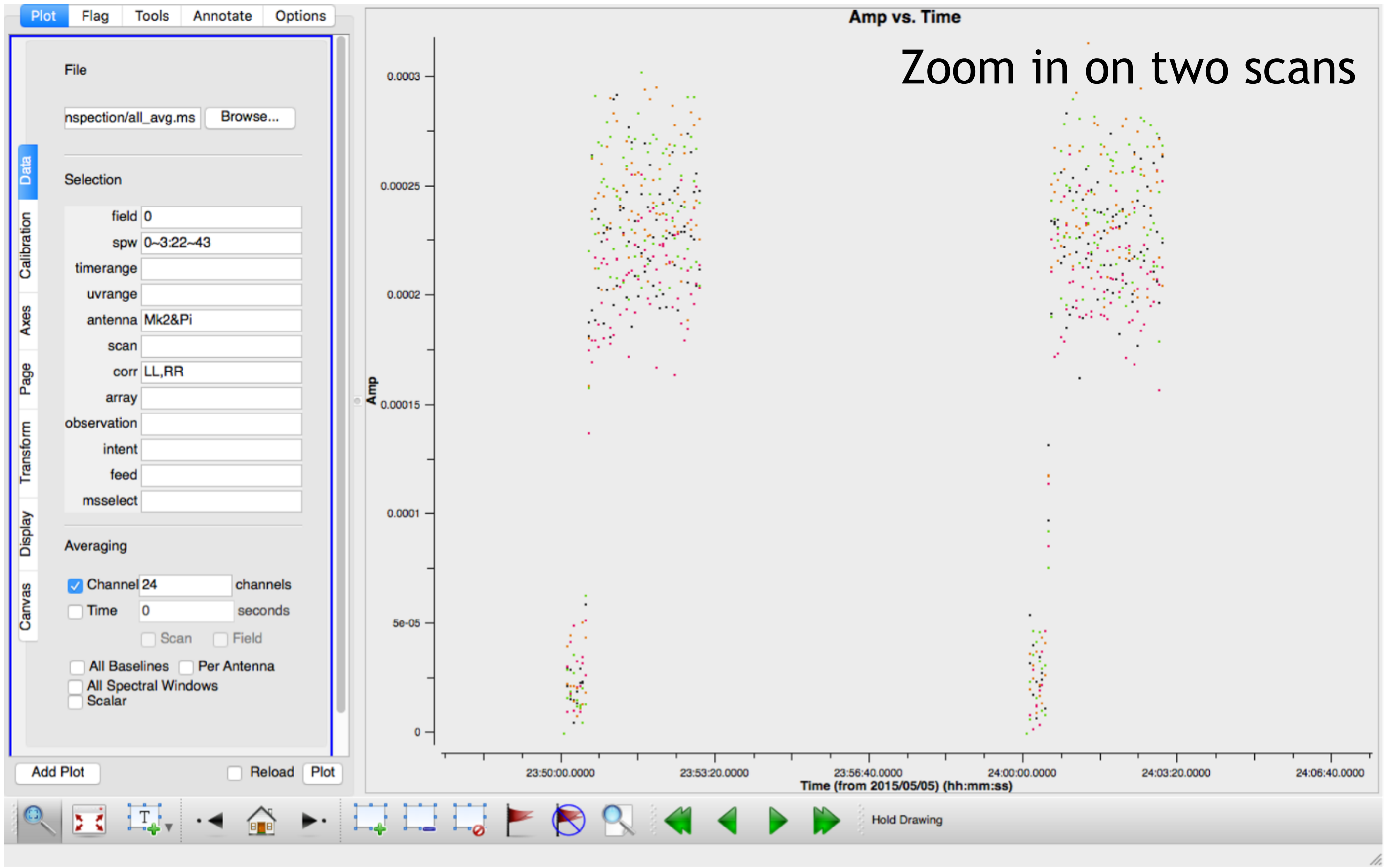
Remember this effect. This requires a mode in **flagdata** called ‘**quacking**’.



Data Flagging: flagdata

The low fluxes at the start (and end) are caused by data being recorded when the telescope is being moved between sources.

- Use plotms to approximate the time off source. e.g. here it is 40s
- Note that slewing times will be different for non-heterogenous arrays!



Data Flagging: flagdata

To flag these scans, these inputs are required!

```
mode                =    'quack'        #   Flagging mode
     field          =        '0'        #   Field names or field index numbers:
                                        #    '' ==> all, field='0~2,3C286'
     spw            =         ''        #   Spectral-window/frequency/channel: ''
                                        #    ==> all, spw='0:17~19'
     antenna        =     'Pi&*'        #   Antenna/baselines: '' ==> all,
                                        #    antenna ='3,VA04'
     timerange      =         ''        #   Time range: '' ==>
                                        #    all,timerange='09:14:0~09:54:0'
     correlation    =         ''        #   Correlation: '' ==> all,
                                        #    correlation='XX,YY'
     scan           =         ''        #   Scan numbers: '' ==> all
     intent         =         ''        #   Scan intent: '' ==> all,
                                        #    intent='CAL*POINT*'
     array          =         ''        #   (Sub)array numbers: '' ==> all
     uvrange        =         ''        #   UV range: '' ==> all; uvrange
                                        #    ='0~100klambda', default
                                        #    units=meters
     observation    =         ''        #   Observation ID: '' ==> all
     feed           =         ''        #   Multi-feed numbers: Not yet
                                        #    implemented
     quackinterval  =         40        #   Quack n seconds from scan beginning
                                        #    or end
     quackmode      =      'beg'        #   Quack mode. beg: first n seconds of
                                        #    scan; endb: last n seconds of scan;
                                        #    end: all but first n seconds of
                                        #    scan; tail: all but last n seconds
                                        #    of scan.
     quackincrement =       True        #   Flag incrementally in time?
```

Pickmere baselines only (e-MERLIN is heterogeneous) — antenna

40s as found in plotms — quackinterval

Low flux only found at start of scan — quackmode

Occurs on all scans! — quackincrement

Data Flagging: flagdata

- Broken elements → remove antennas **(flagdata mode = manual)**
- Correlator malfunctions → remove timesteps **(manual)**
- Shadowing → remove antennas in time range **(shadow)**
- Initial pointing delay → remove first timesteps **(quack)**
- Bandpass issues → remove channels **(manual)**
  Low elevation → remove antennas with low elevation **(elevation)**
- Correlated noise on some baselines (e.g. LOFAR split stations) → Flag baselines **(manual)**
- Interference → remove antennas, timestep, frequencies or baselines... **(manual,clip)**

If all this is done, then calibrating the data is a whole lot easier!

Automated Data Flagging

We are really good at picking up patterns which allows easy identification of RFI, however computers are not!

Common manual excision methods:

- Manual selection by data reducing astronomer

- Thresholding / specialized project pipelines (e.g. Baan et al. 2004, Winkel et al. 2007)

- **Manual selection is not practical for modern observatories:**
  - Enormous data volumes, computationally fast algorithms required.
  - Needs to be more accurate than thresholding

We need fast automated RFI excision routines!

Automated Data Flagging

Two classes of RFI excision methods:

- Detection: find & throw away affected data
- Filtering or subtracting: estimate RFI contribution and restore affected data

Detection methods ("flagging") commonly used

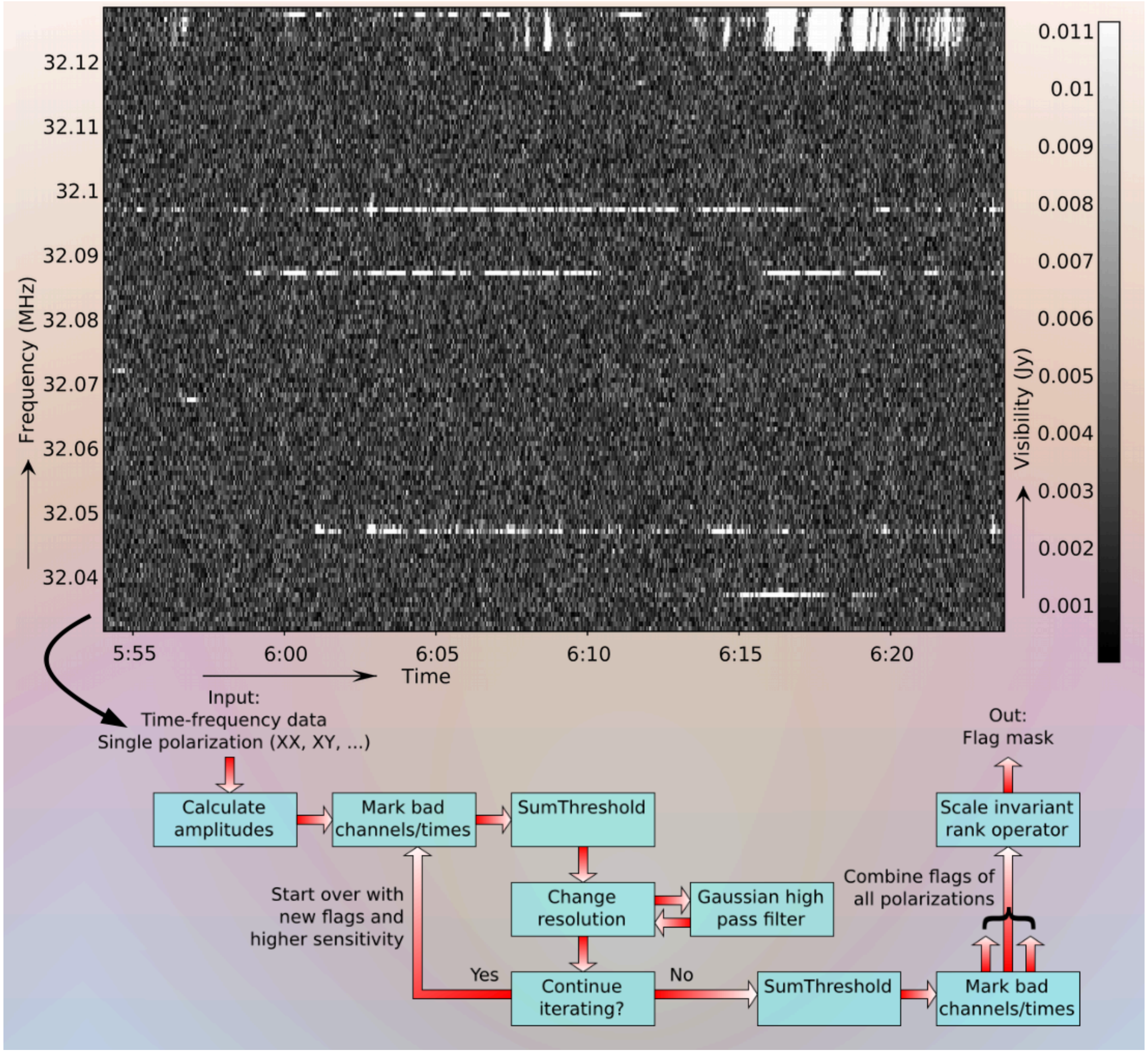
- Some specialized pipelines for surveys or instruments

Filtering RFI is harder

- Resulting data quality is not well understood
- Requires more resources
- Lack of full (automated) filtering pipelines

Automated Data Flagging Example

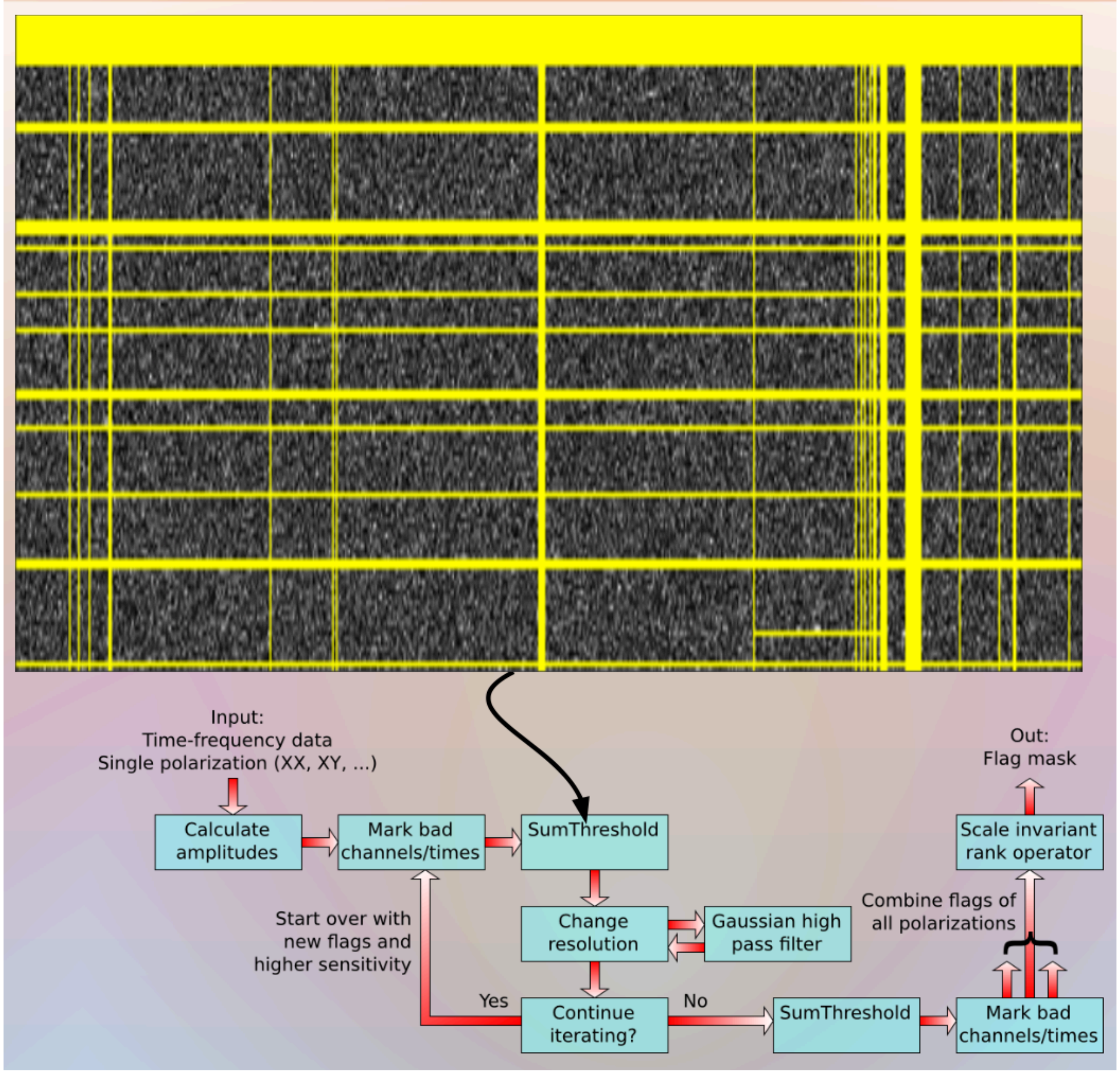
How AOFlagger works:



Automated Data Flagging Example

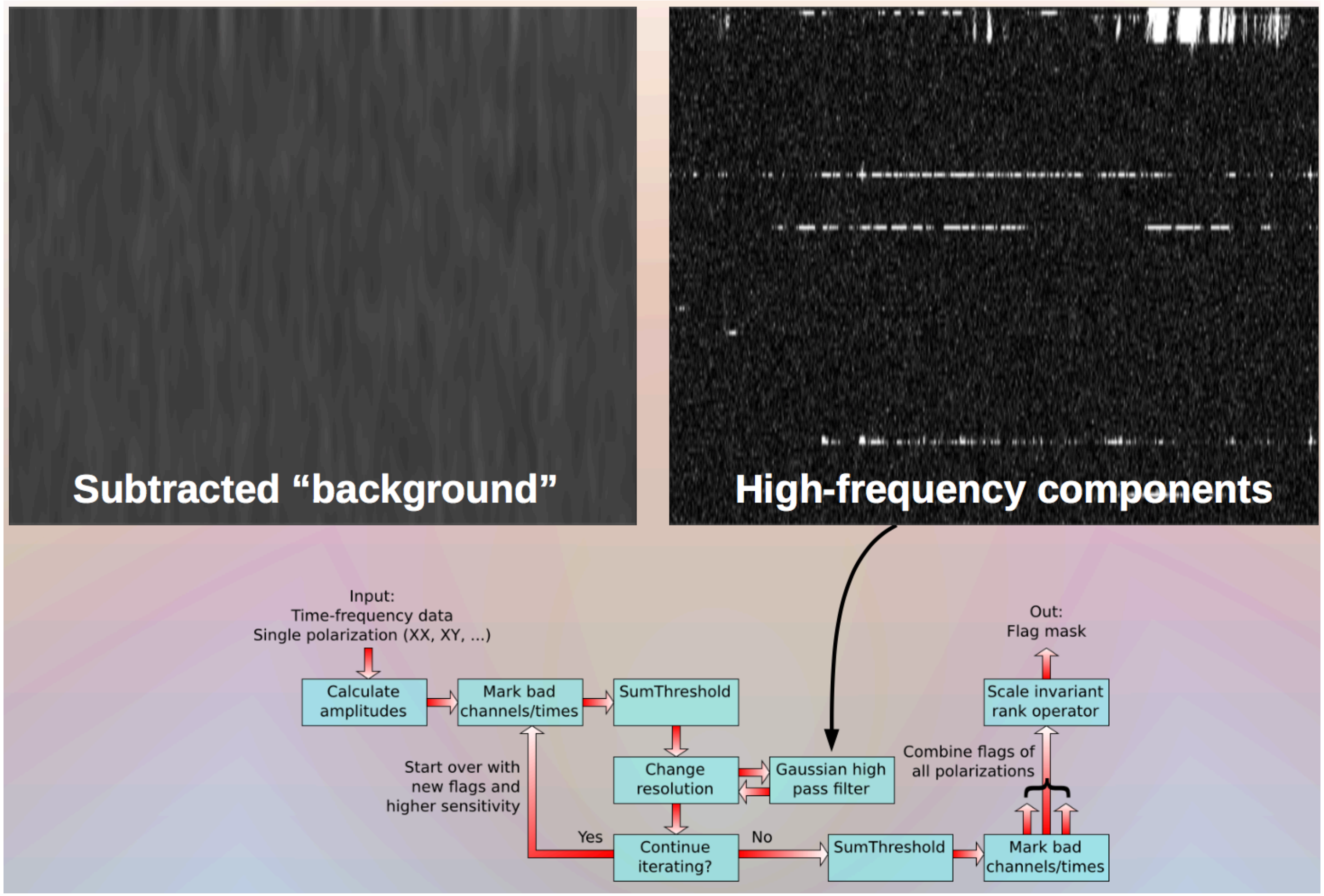
How AOFlagger works:

Initial sum threshold



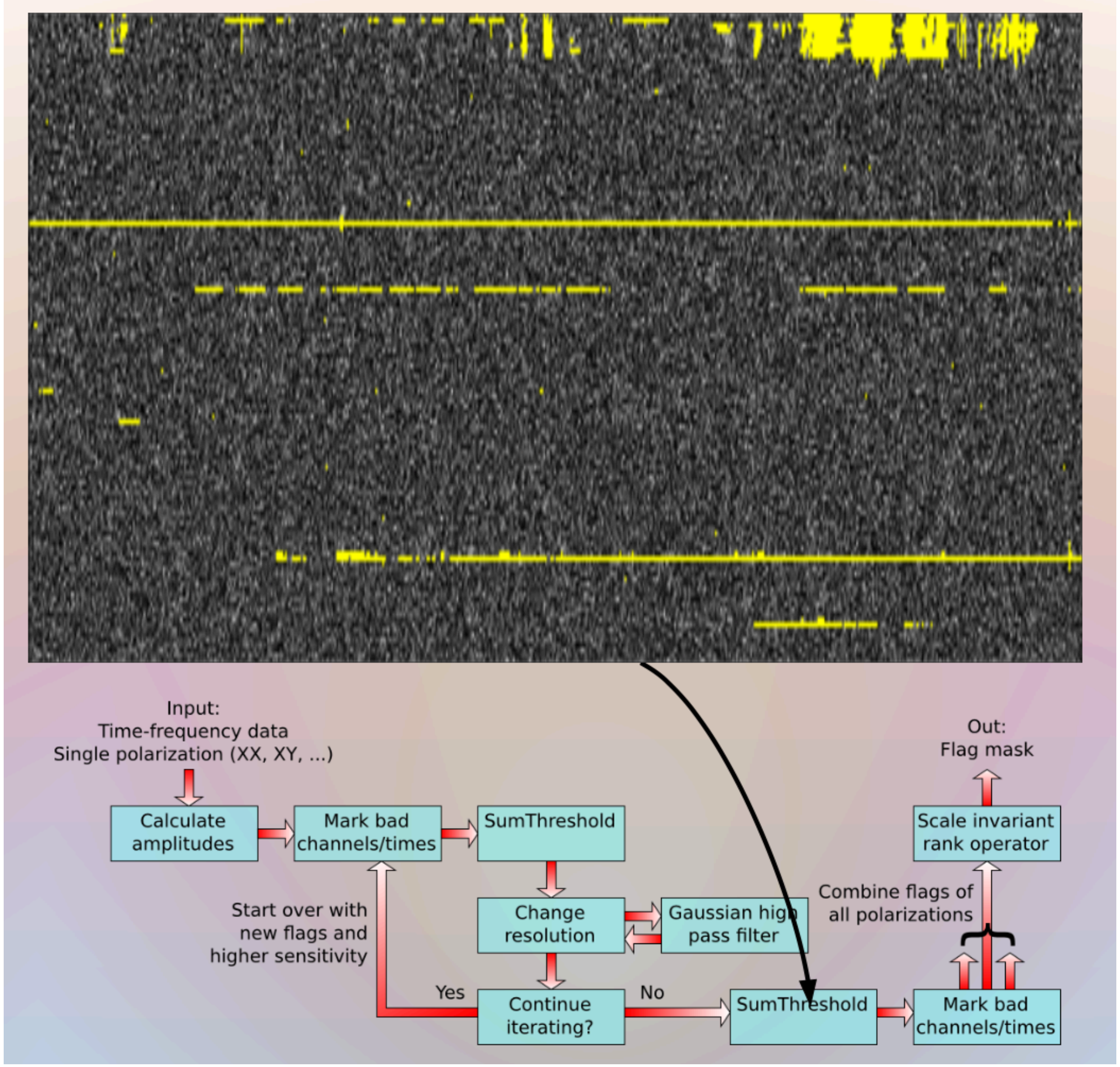
Automated Data Flagging Example

- Change time & frequency resolution then remove components using high pass filter
- Iterate, changing resolution each time



Automated Data Flagging Example

Finish iterating & do final sum threshold



Automated Data Flagging Example

What could go wrong?

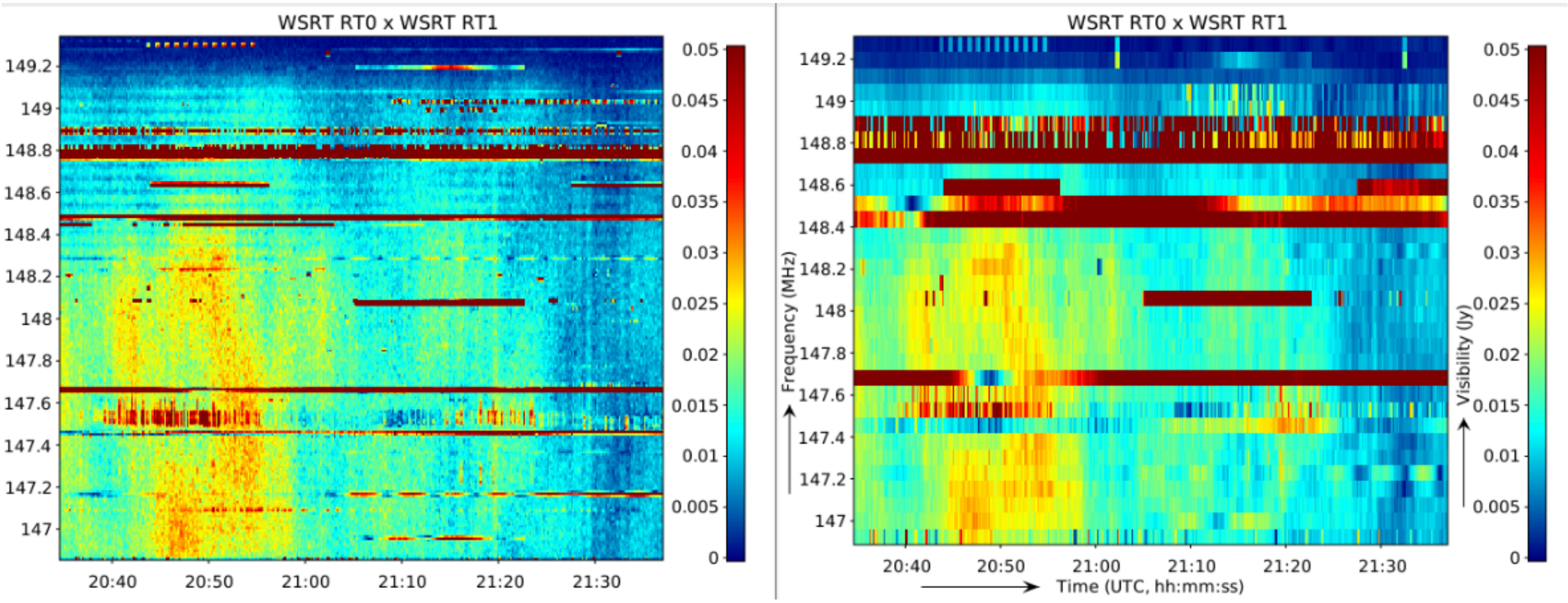
- Some astronomical sources vary quickly in time (sun, pulsars, ...)
- Quick fringes are line-like patterns
- Spectral line observations

Mostly not an issue – sources are mostly much weaker than RFI, and invisible in single correlations.

Data Averaging

HEALTH WARNING: Always flag (first) at highest possible resolution!

Highest resolution: Averaged without RFI detection:



- Flagging is incremental: don't reset flags!
- Correlator might have set flags. These will be lost. To undo flagging, use backup (column).

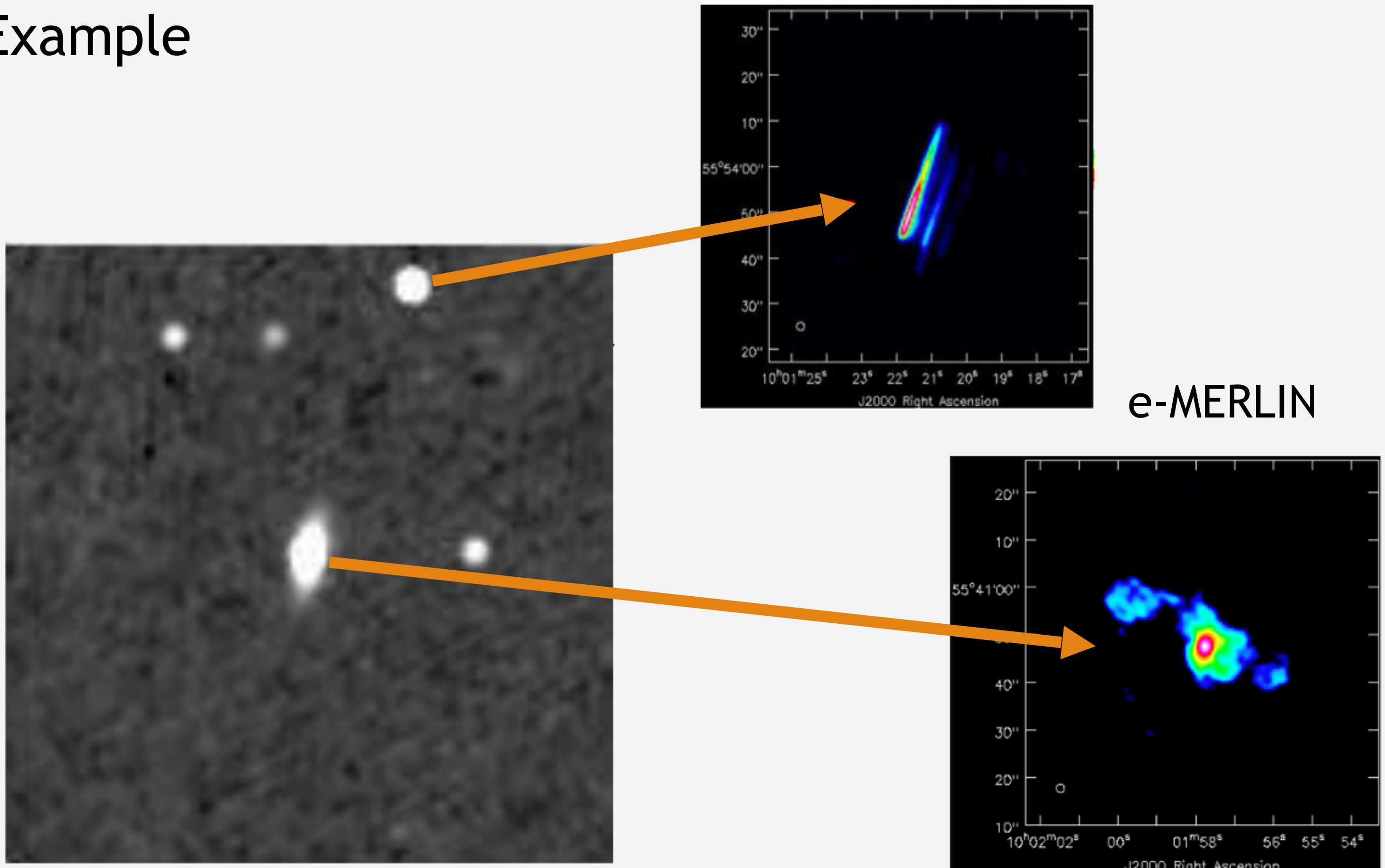
Data Averaging

- Data size can be reduced by averaging data in time and/or frequency direction
- Only average after RFI detection
- Over-averaging causes smearing
    - Time-smearing: in tangential direction
    - Frequency-smearing: in radial direction
- Calibration might also constrain averaging factor

BANDWIDTH SMEARING

(from Basic Imaging Lecture)

Example



NVSS image

Effect is radial smearing, corresponding to radial extent of measurements in uv plane

TIME SMEARING

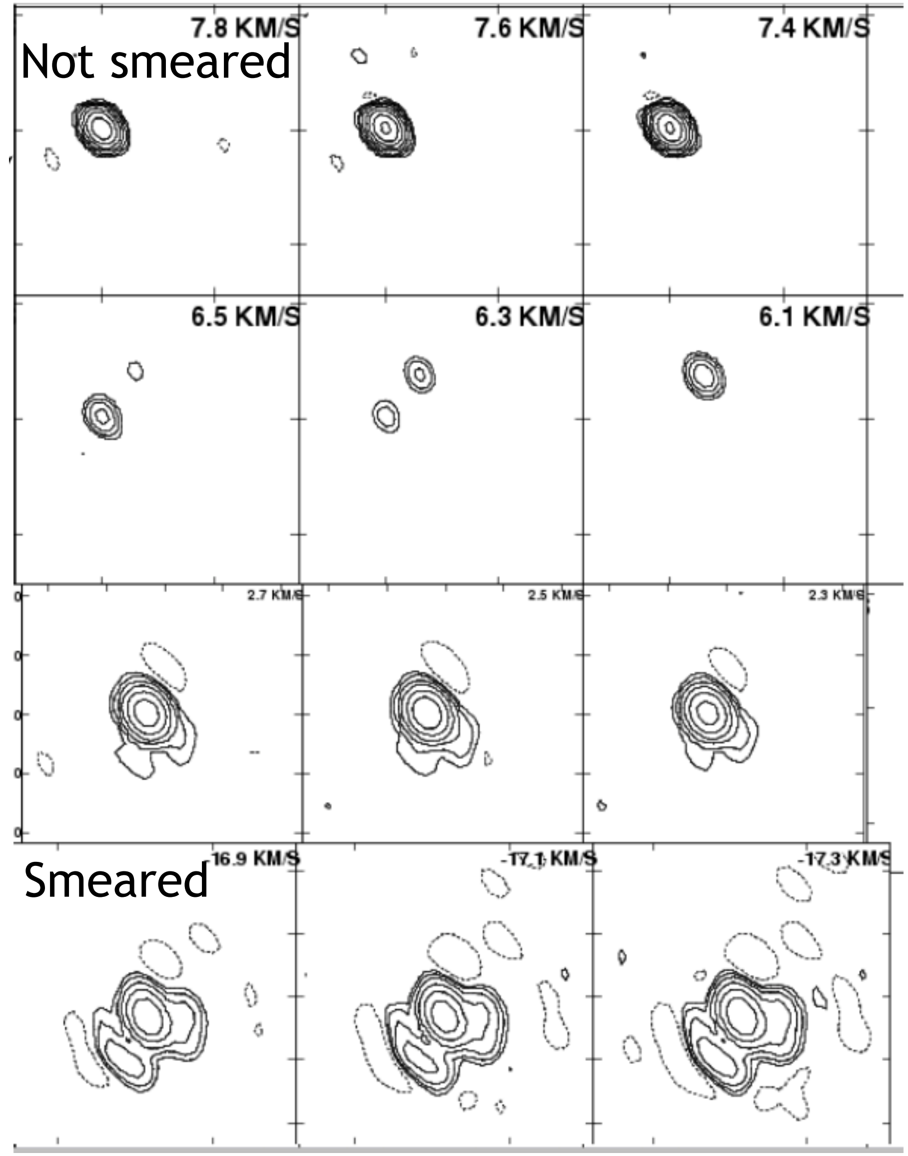
(from Basic Imaging Lecture)

- Time-average smearing (de-correlation) produces tangential smearing

- Not easily parameterized. At declination +90° a simple case exists where percentage time smearing is given by:

$$\omega_e \delta t_{\text{int}} \frac{\theta}{\theta_{\text{HPBW}}}$$

- At other declinations, the effects are more complicated.



Data Averaging in CASA

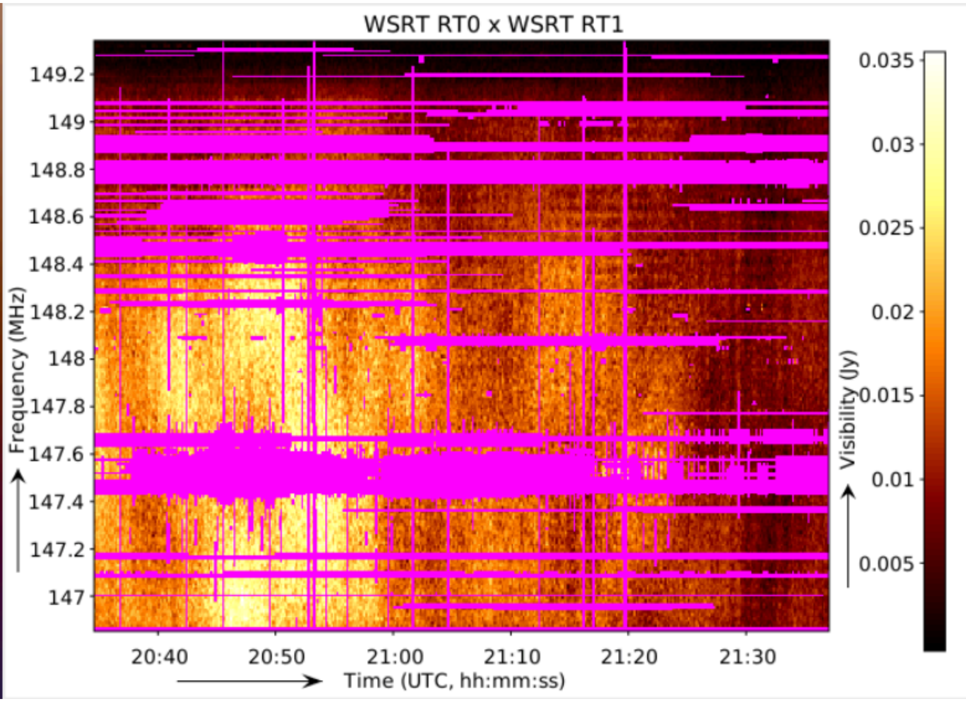
If smearing calculated to be at acceptable levels (see Basic Imaging), use CASA task split to average the data.

Example input:

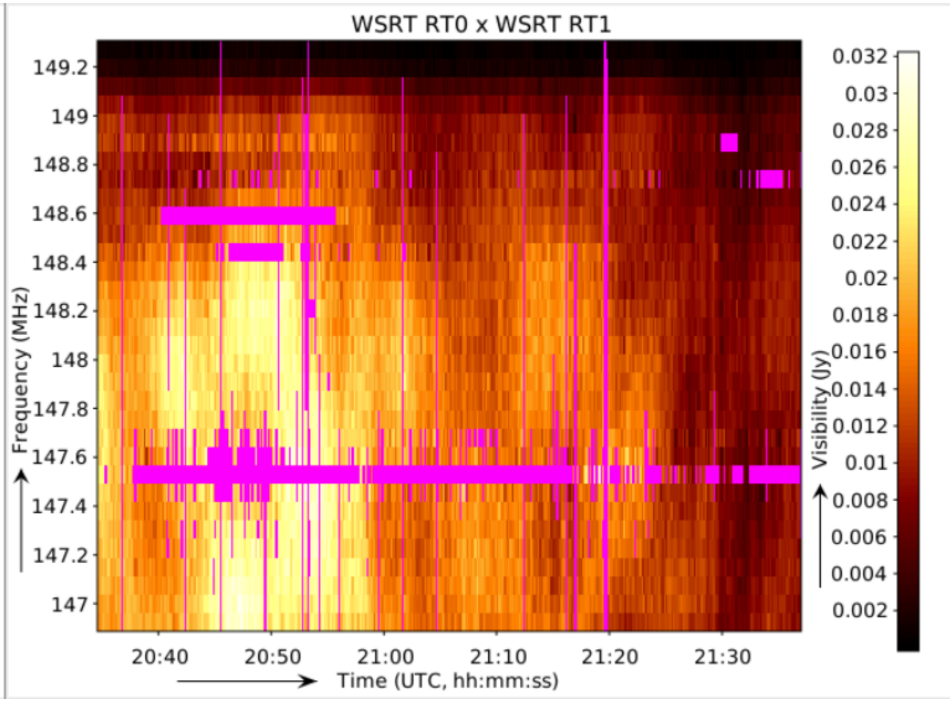
```
inp split
vis='3C196_spw5_sub1.MS'  (input)
outputvis='averaged.MS'
width=8  (Average over 8 channels)
timebin='60s'  (Average over 60 s)
go
```

Data Averaging in CASA

Original resolution:



After averaging



- Processing data can be very time expensive, but almost all steps scale linear with number of visibilities.
- Work on averaged data (and/or subset) while experimenting with settings

Summary

- First step in data processing is data inspection
- Second step is BACKUP YOUR DATA
- Third step is data flagging and RFI detection
- Calibration, imaging, … to be discussed!