RFI and Compression

David Rafferty

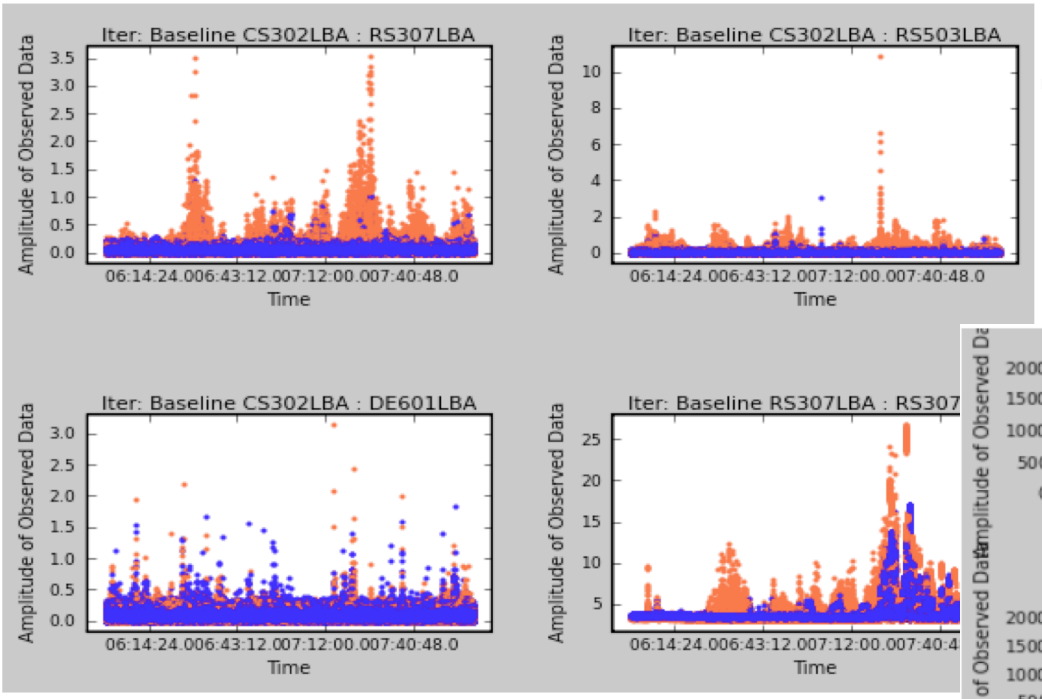
Introduction

- Given the “raw” data (i.e., from the online processing pipeline), the first two steps that are typically done are:
  - Find and remove RFI (Radio Frequency Interference)
  - Compress data in time and frequency
- Removal of RFI is critical to good calibration and imaging
- Compression is critical to decreasing processing time, which, given LOFAR data volumes, is an important concern

RFI

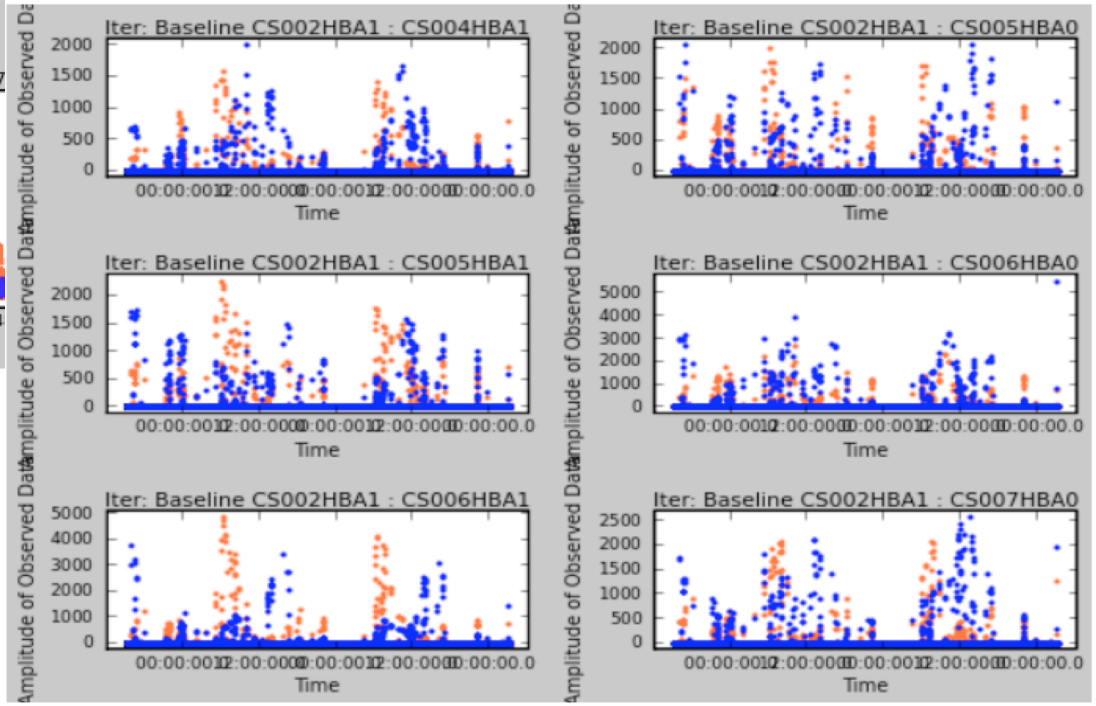
- RFI comes from a variety of terrestrial and non-terrestrial sources:
  - Satellites - communication, GPS, etc.
  - Aircraft - can emit and reflect RFI from distant sources
  - Ground-based - radio stations, electrical fences, etc.
  - Observatory-based - computer hardware, etc.
- Can extend over wide range of frequencies and over long periods of time, but for LOFAR is mostly narrowband and short duration

RFI in LOFAR Data

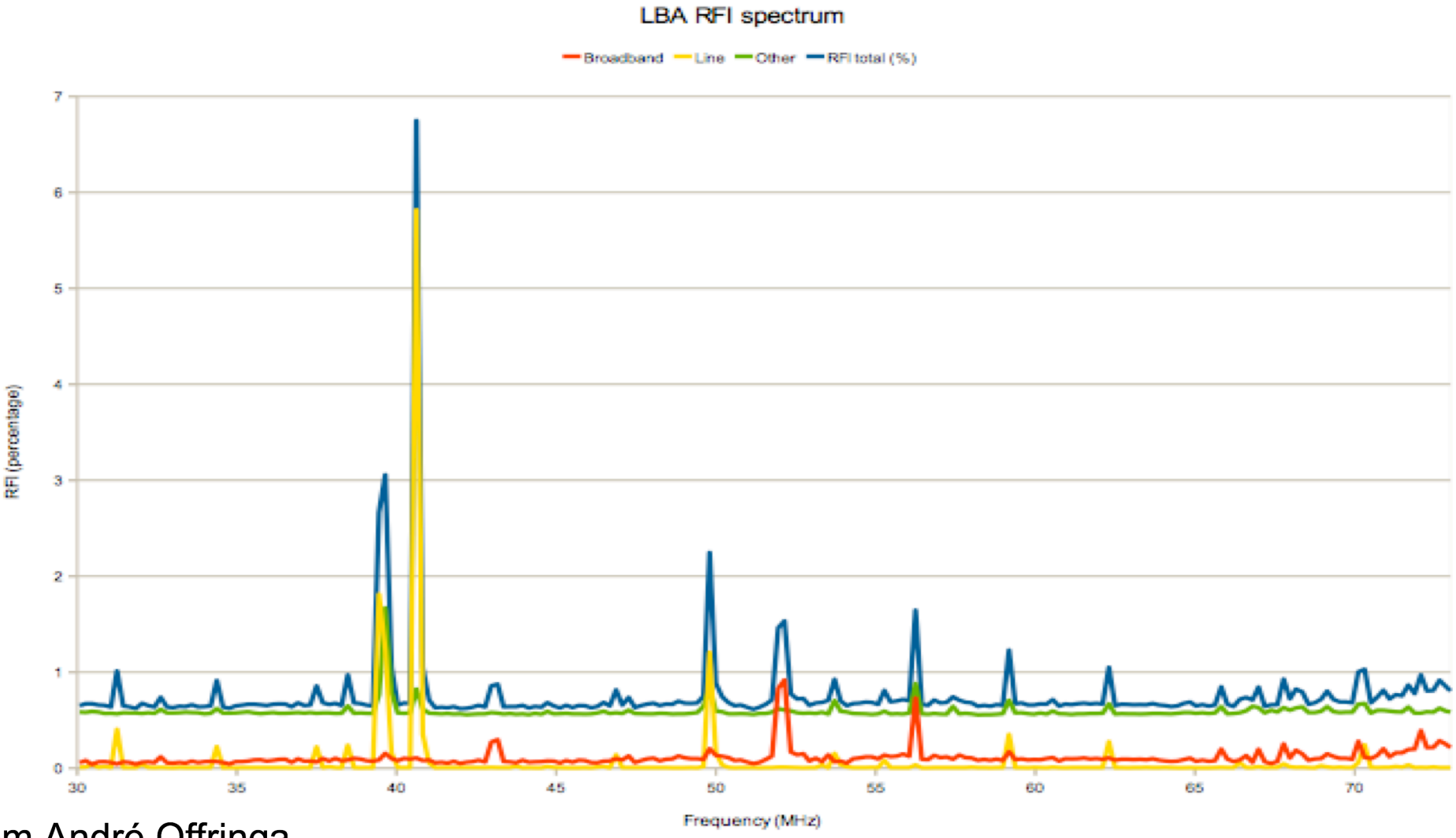


LBA data (SB64 of L2009_14319):
No flagging or compression

HBA data (SB48 of L2009_16167):
No flagging or compression

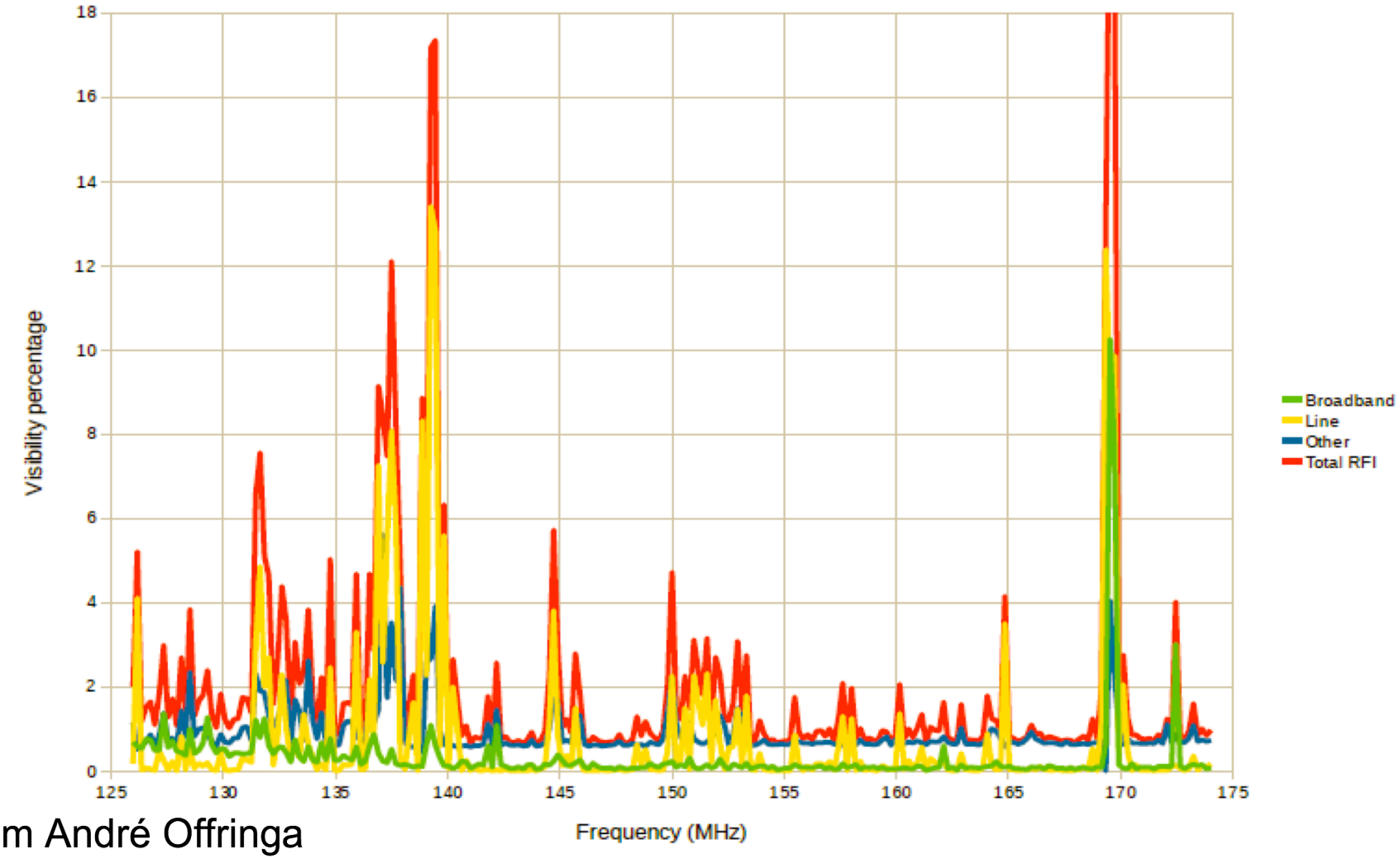


RFI in LOFAR Data



from André Offringa

RFI in LOFAR Data



from André Offringa

RFI Flagging

- Several different (and generally complementary) approaches:
  - Search for anomalously high visibility amplitudes and flag
  - Search for RFI and subtract
  - Eliminate RFI pre-correlator (filters, nulls, etc.)
- Post-correlator methods work best on data with high frequency and time resolution
  - Compression obscures RFI

Current LOFAR RFI Flaggers

- Current flaggers are amplitude-only (I.e., no phase information is used)
- MADFlagger:
  - Compares median-subtracted visibility amplitudes to the median absolute deviation (MAD) of the data in a given window
  - Implemented in NDPPP (the New Default Pre-Processing Pipeline)
- rficonsole:
  - Fits a surface to the visibilities
  - Much more sophisticated treatment of points near strong RFI that are affected but not strongly so.
  - Not part of NDPPP - must be run separately

The MADFlagger

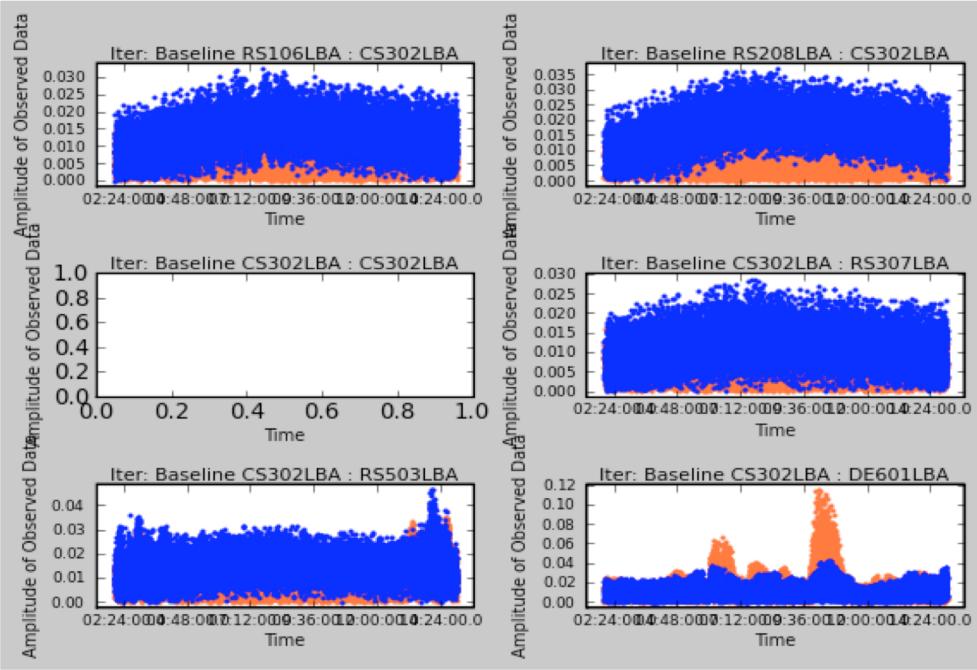
- Calculates MAD $= \text{median}_i\left(\left|X_i - \text{median}_j\left(X_j\right)\right|\right)$ within specified window in time and frequency
- Flags if:

  $\text{threshold} \times 1.48 \times \text{MAD} < \left|\text{X}_i - \text{median}\right|$
- Parameters are set in NDPPP.parset. E.g.,

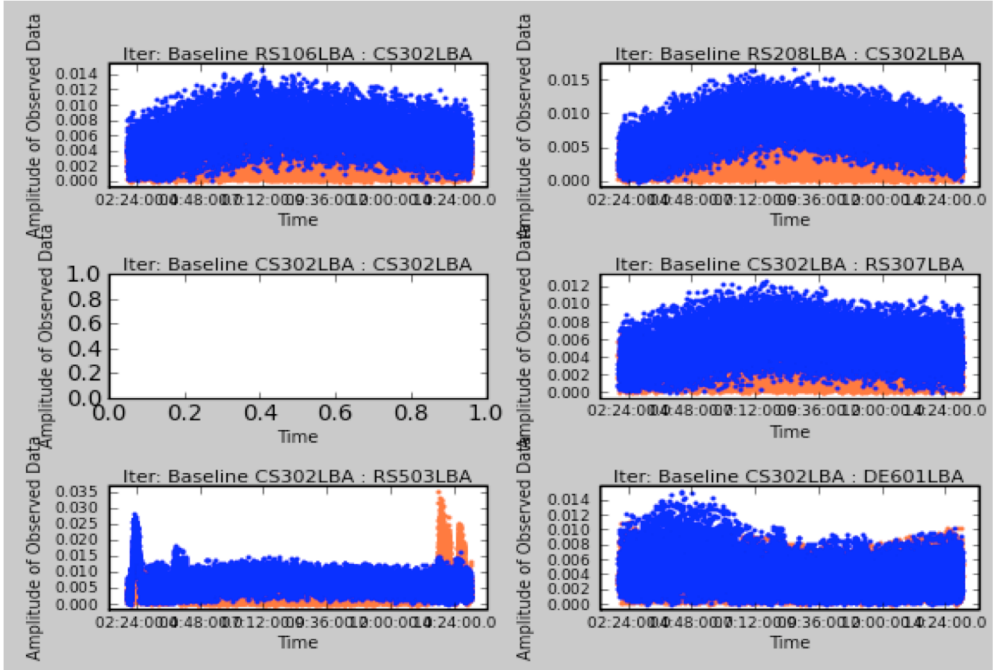
```
flag1.type = madflagger
flag1.threshold = 4
flag1.freqwindow = 31
flag1.timewindow = 5
flag1.correlations = [0,3]
```

- see the Cookbook for details

LBA Data after MADFlagger

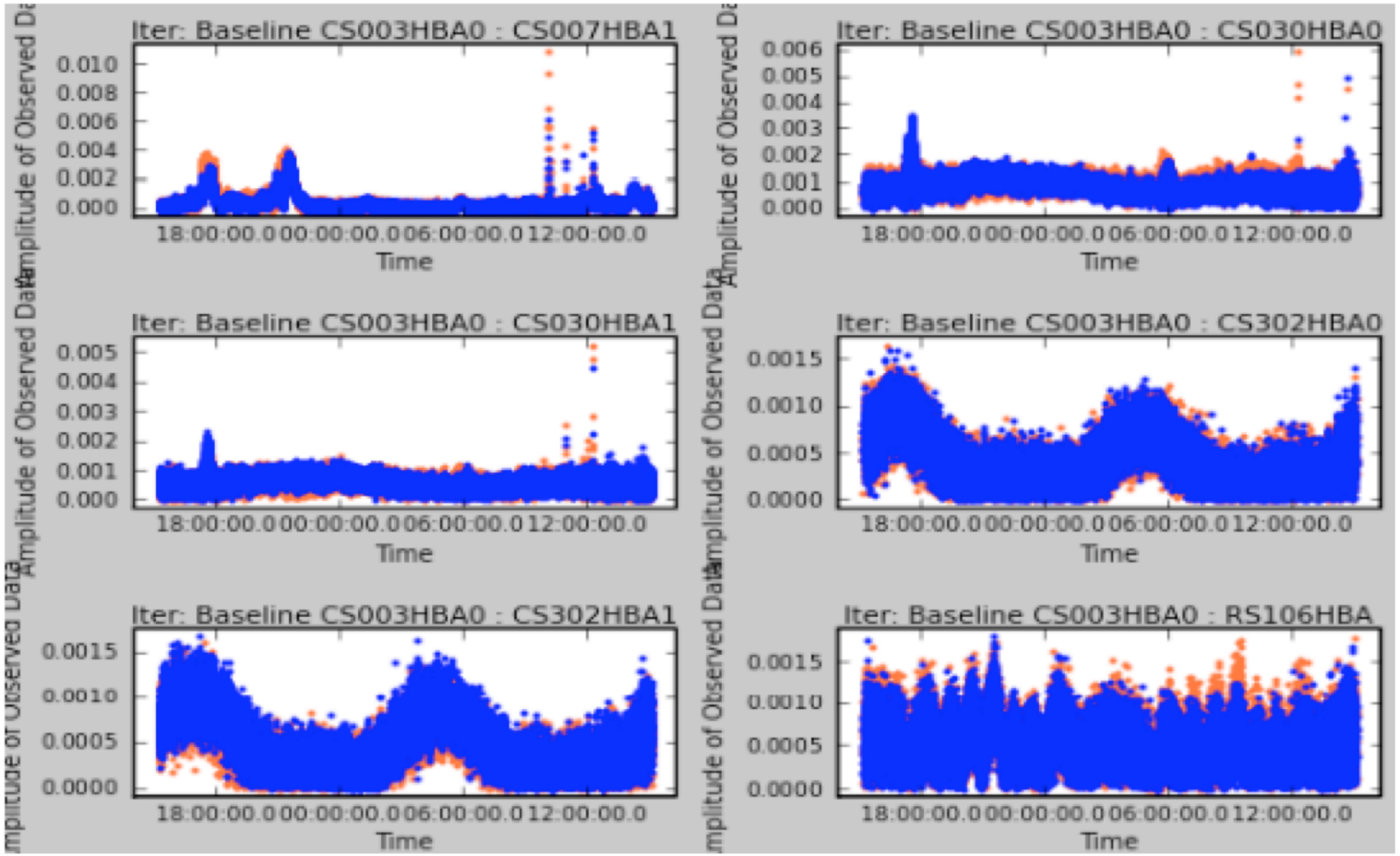


SB72 of L2009_14319



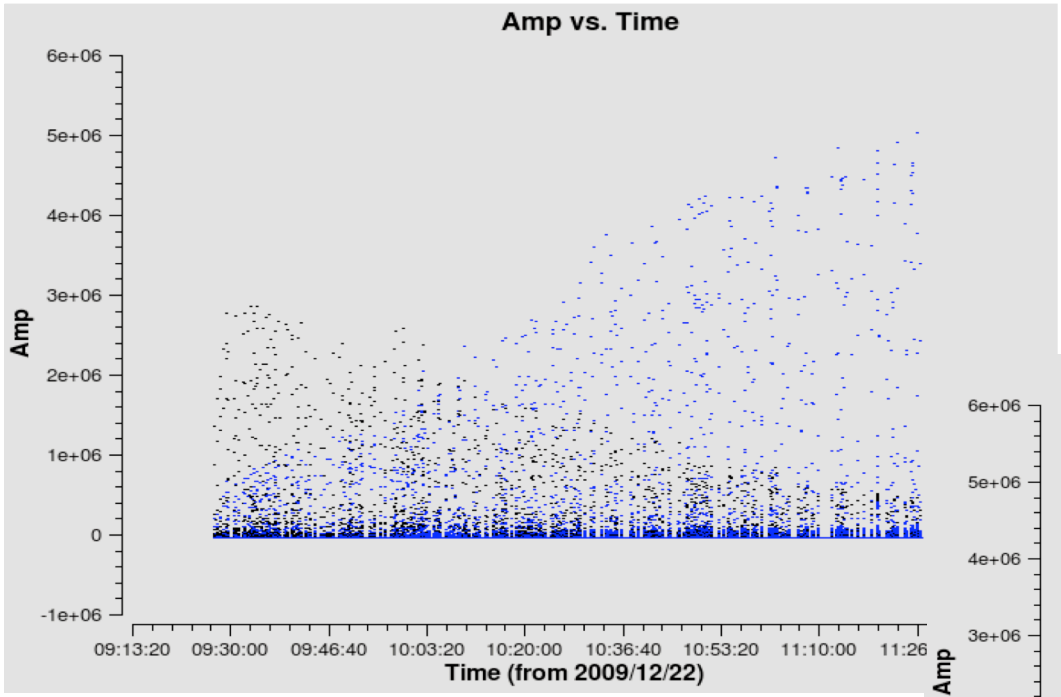
SB57 of L2009_14319

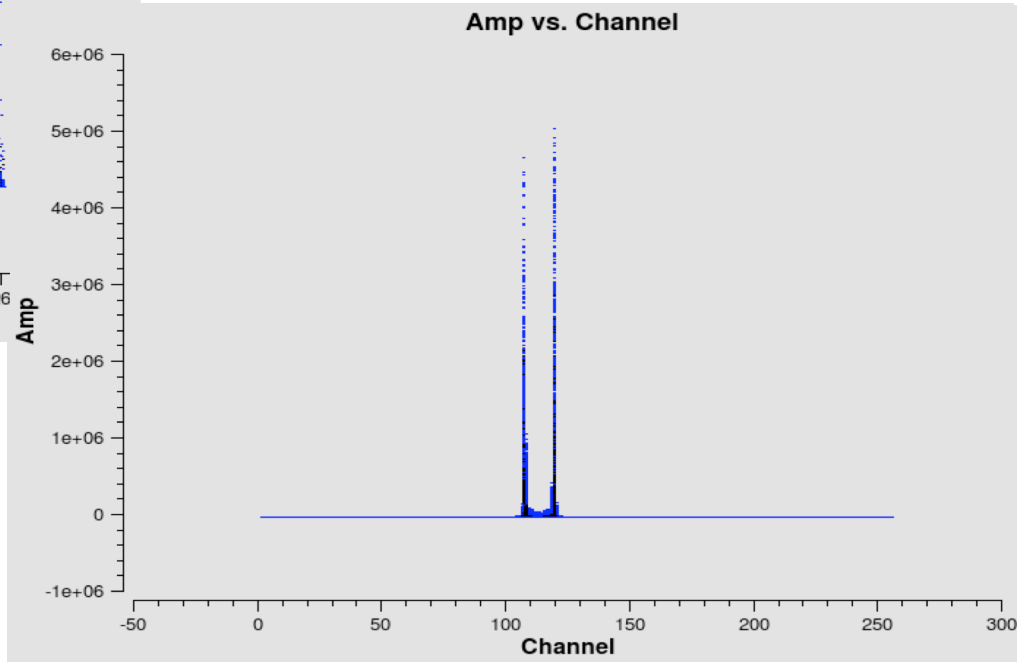
HBA Data after MADFlagger



SB38 of L2009_16167

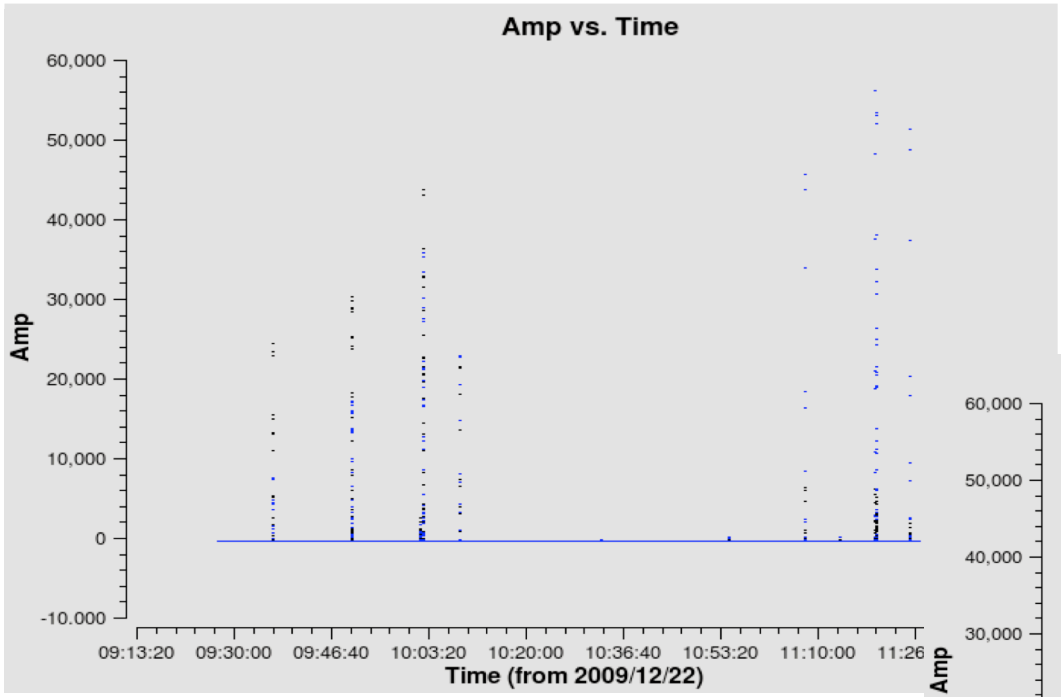
HBA Data after MADFlagger



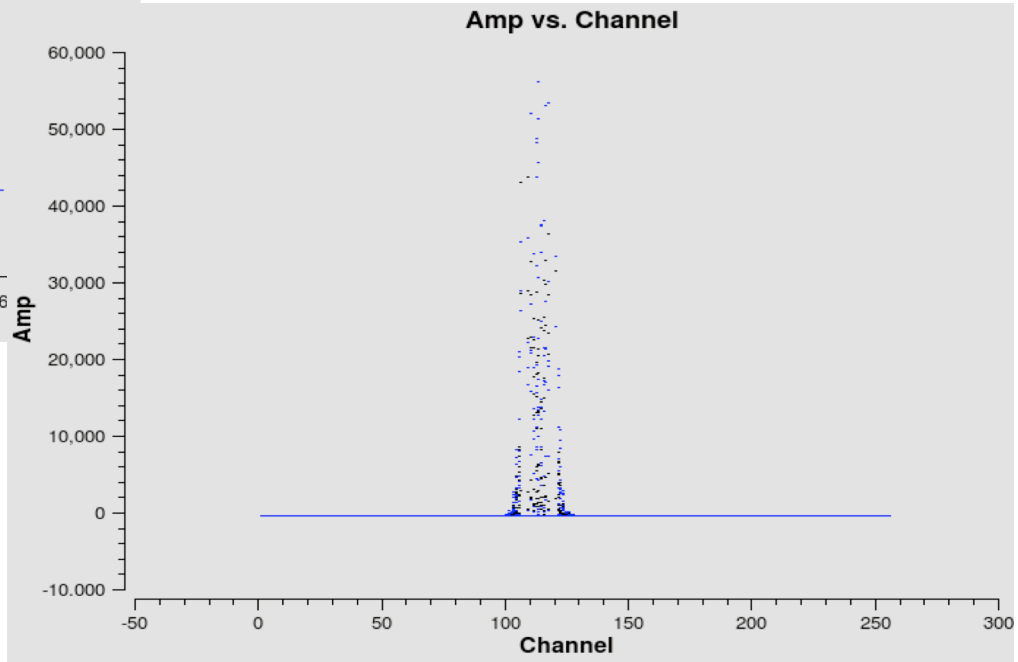


SB50 of L2009_16167

HBA Data after MADFlagger



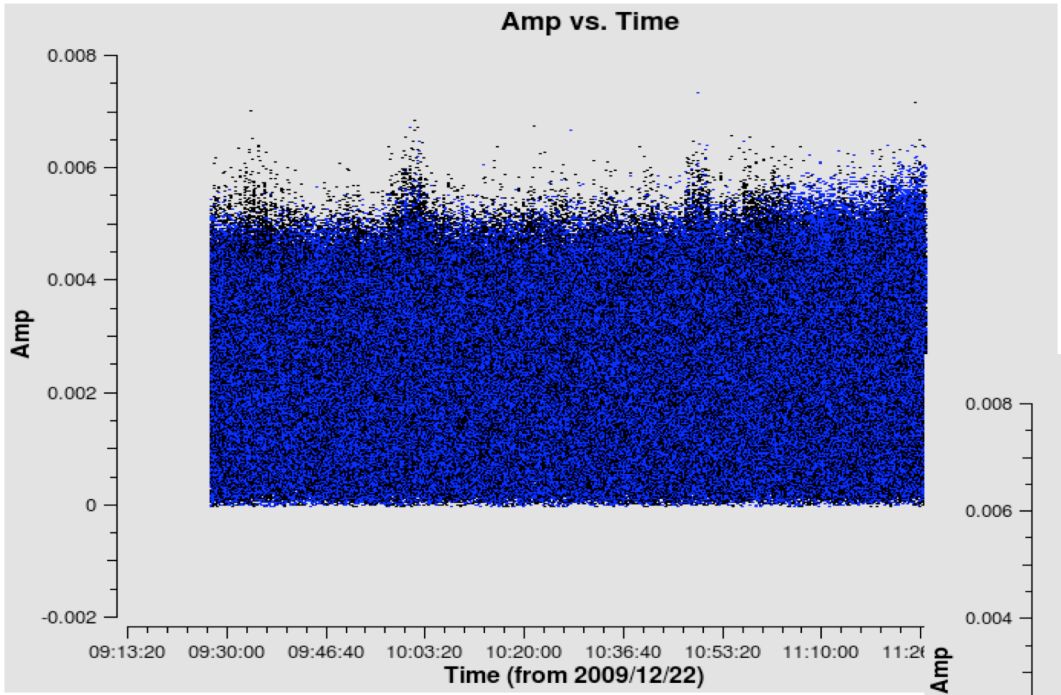

```
flag1.threshold = 4
flag1.freqwindow = 31
flag1.timewindow = 5
```

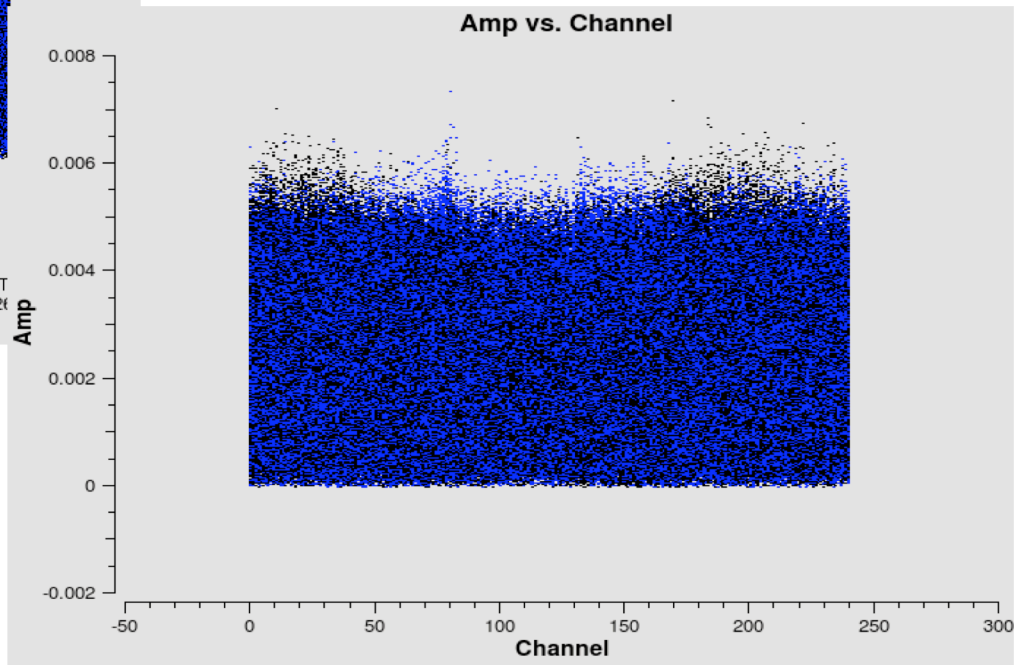


SB50 of L2009_16167

HBA Data after MADFlagger

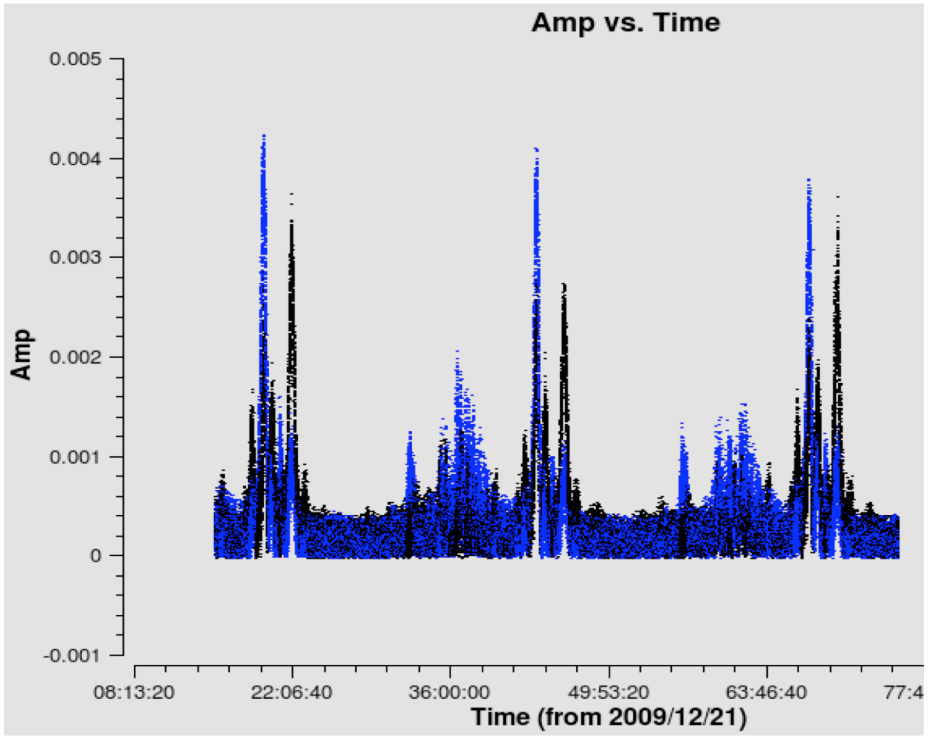
```
flag1.threshold = 2
flag1.freqwindow = 101
flag1.timewindow = 1
flag2.threshold = 2
flag2.freqwindow = 1
flag2.timewindow = 101
```





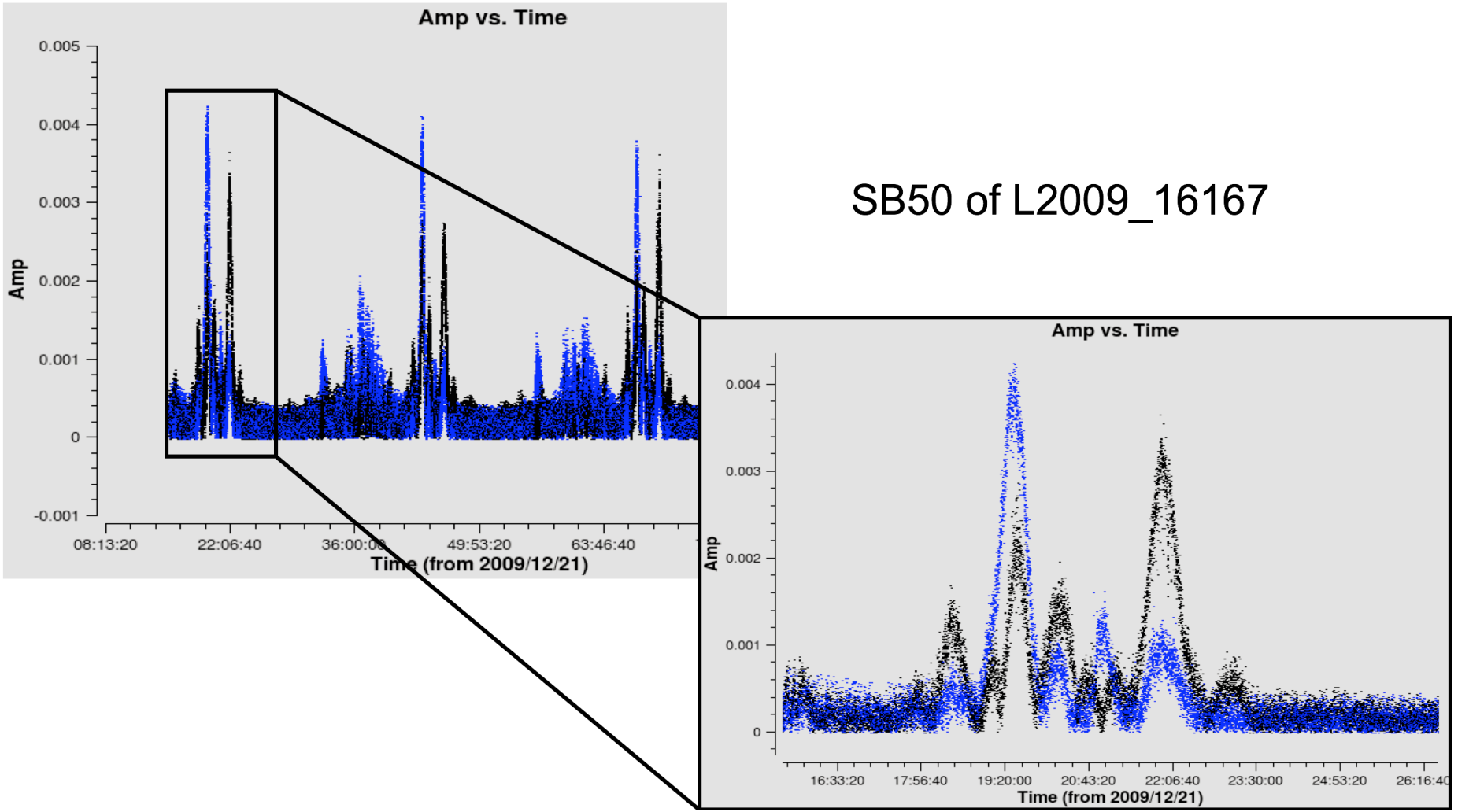
SB50 of L2009_16167

HBA Data after MADFlagger



SB50 of L2009_16167

HBA Data after MADFlagger



SB50 of L2009_16167

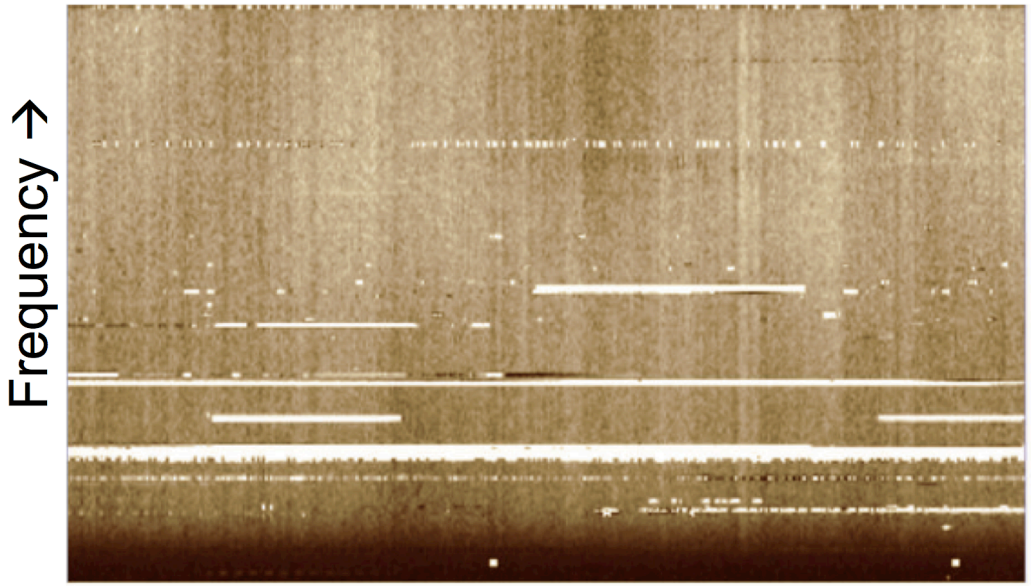
AOFlagger (rficonsole)

- Flagger developed by André Offringa (see Offringa et al. 2010, MNRAS, 405, 155 for details)
- Not part of NDPPP
  - Run with command "`rficonsole <input.ms>`"
- Default strategy appears to work very well
- Has two main advantages over MADFlagger:
  - Uses a more sophisticated fit to visibilities
  - Flags neighboring points that don't meet flagging criteria on their own but do when summed

rficonsole: Surface Fitting

- Estimate astronomical signal by fitting a surface to the smoothed visibilities:

Offringa et al. (2010)



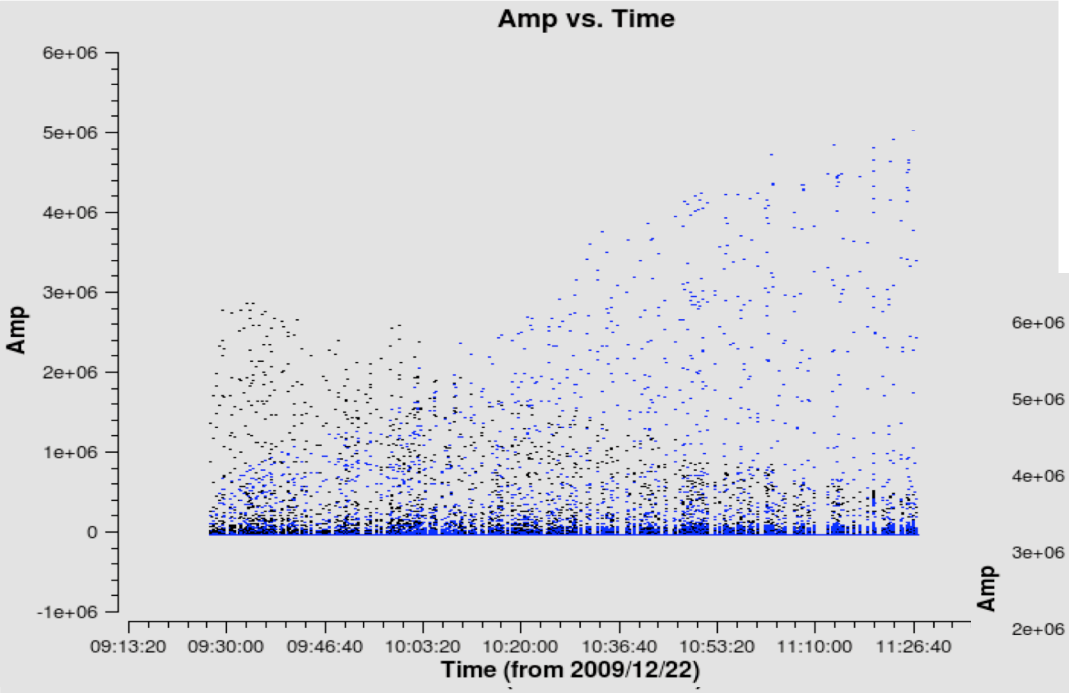
raw visibility amplitudes

fitted surface

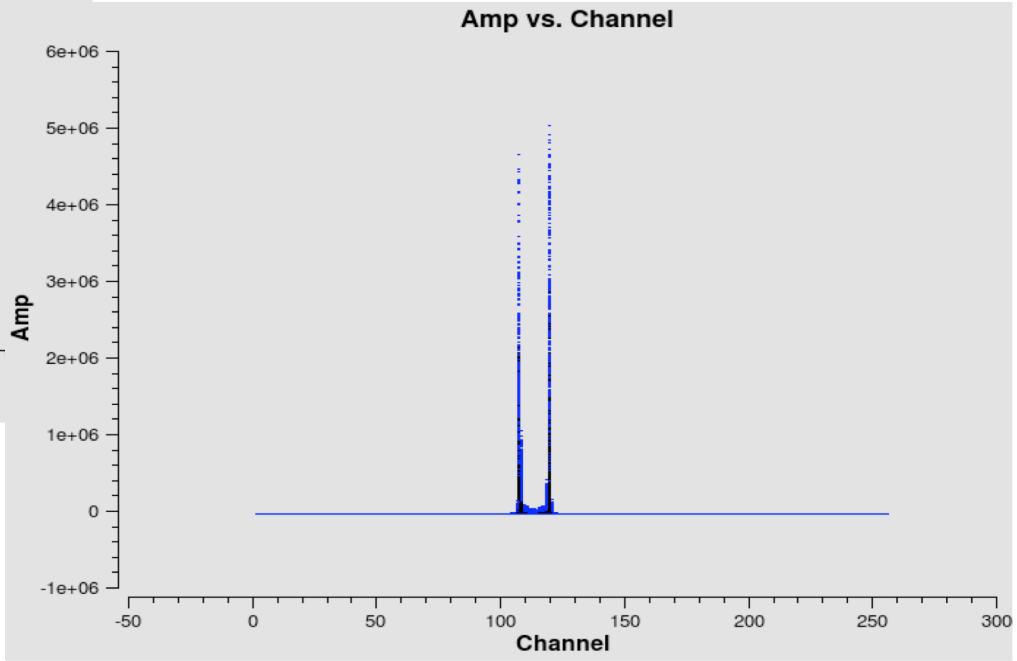
rficonsole: Thresholding

- Often, points near strong RFI are also affected, but only moderately ➔ these points often fall below the threshold for flagging
- Use combinatorial thresholding:
  - sum amplitudes of connected points and compare to threshold
  - threshold can vary depending on number of points summed
  - RFI that is missed by “single-pixel” thresholding can be caught by combinatorial thresholding

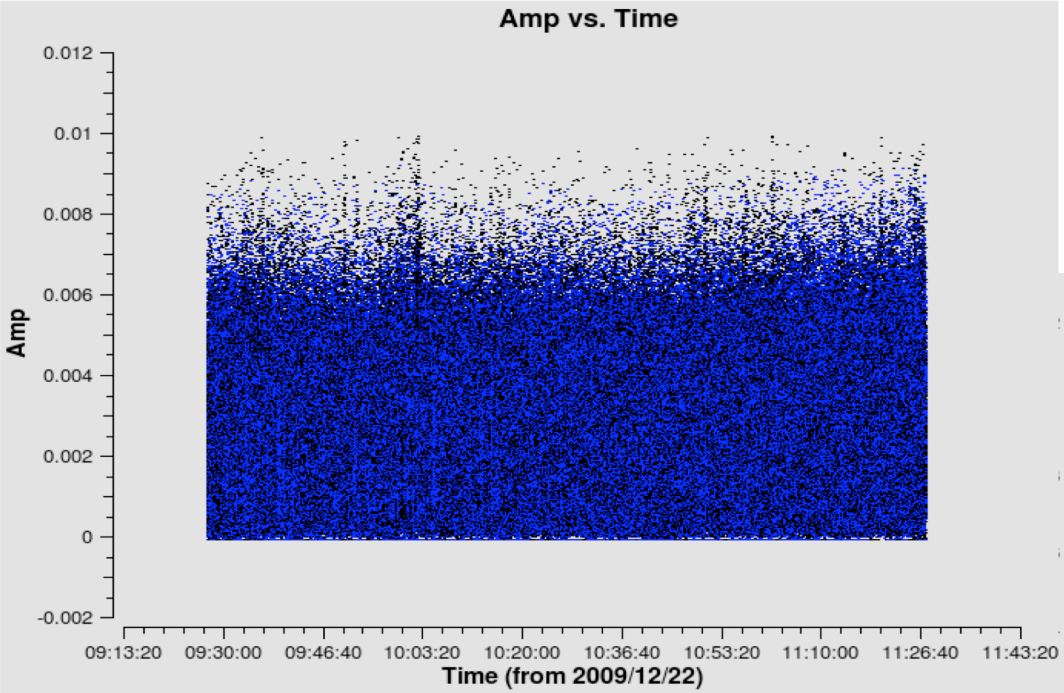
HBA Data after rficonsole



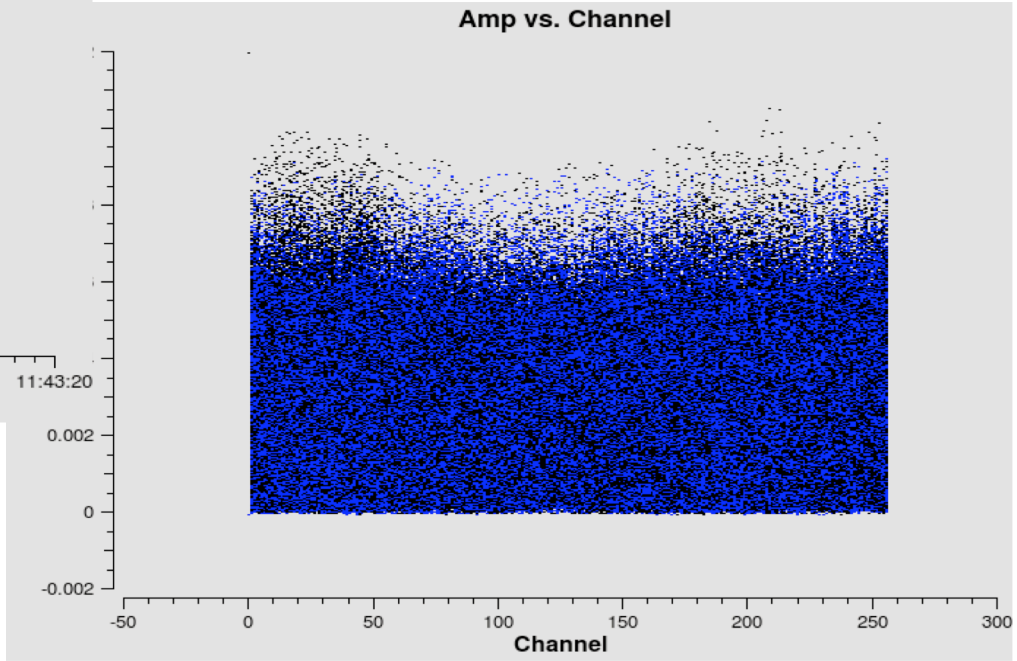
SB50 of L2009_16167



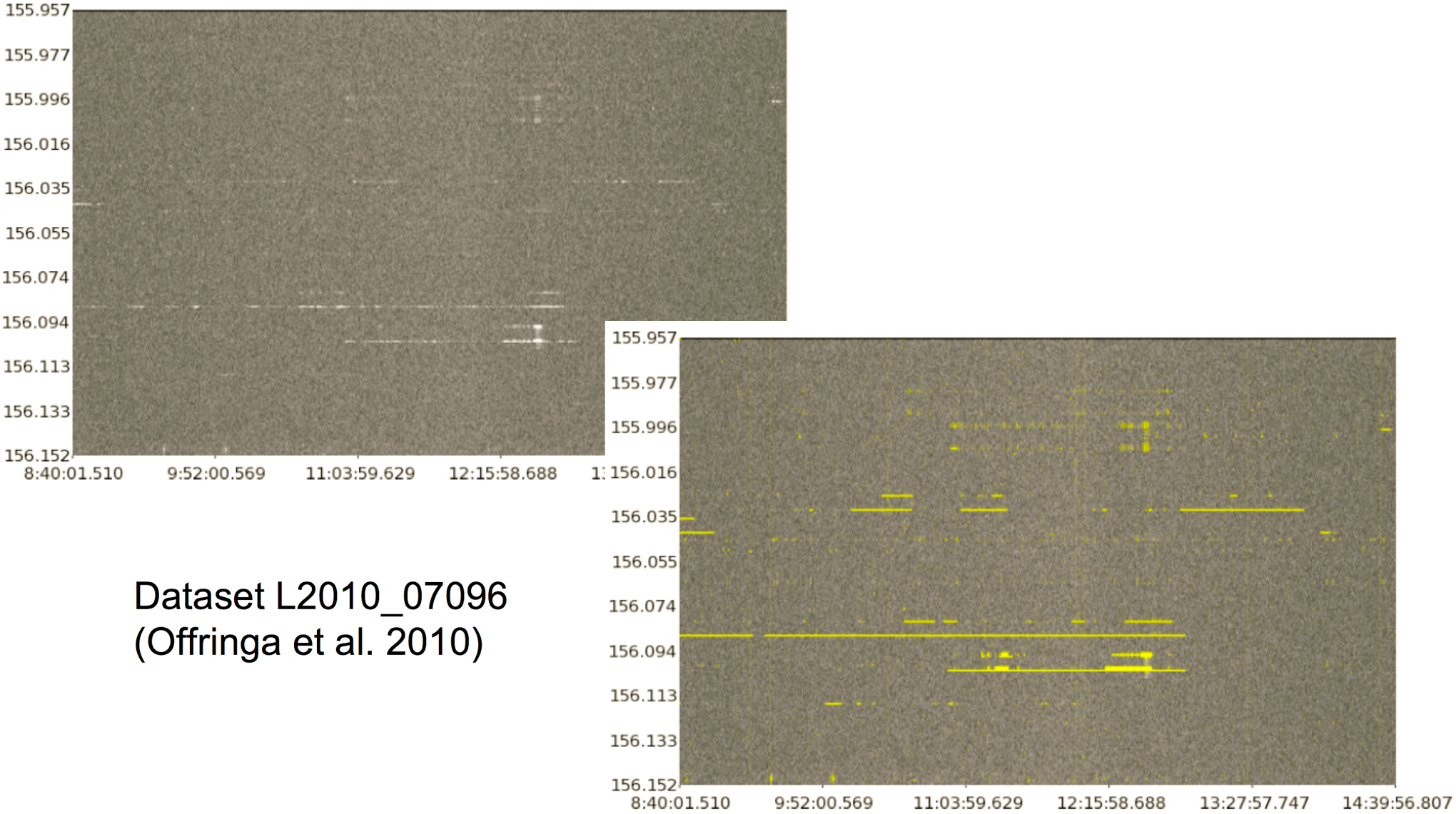
HBA Data after rficonsole



SB50 of L2009_16167

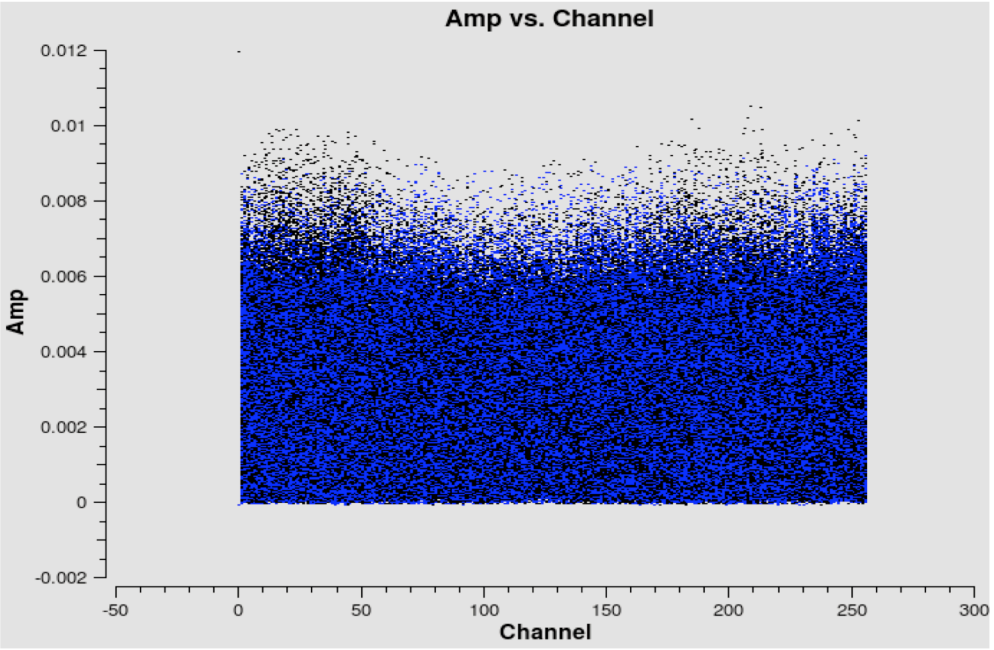


HBA Data after rficonsole

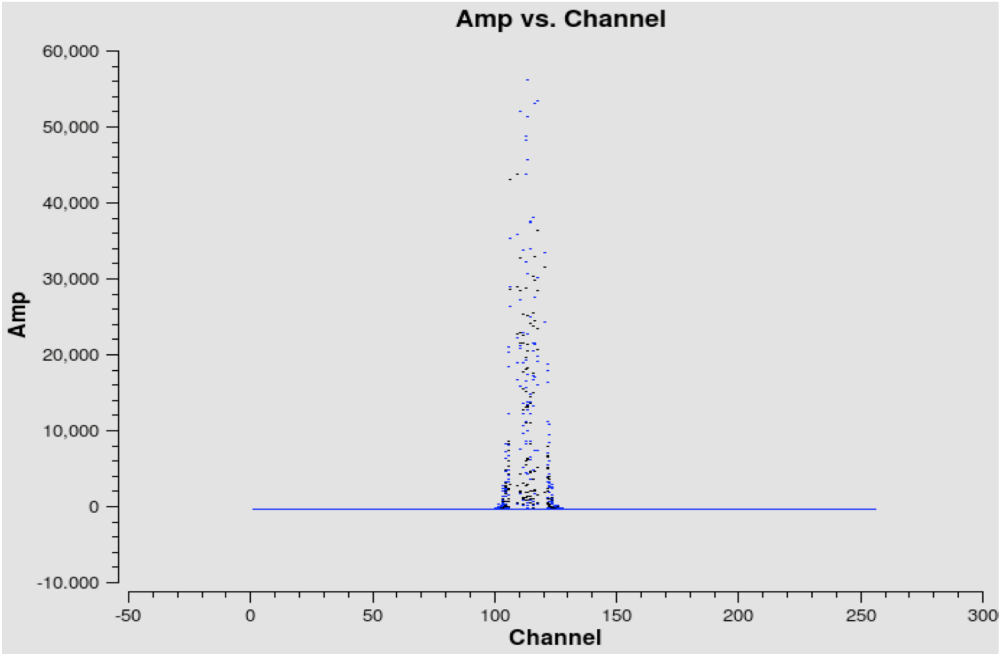


Dataset L2010_07096
(Offringa et al. 2010)

rficonsole vs. MADFlagger



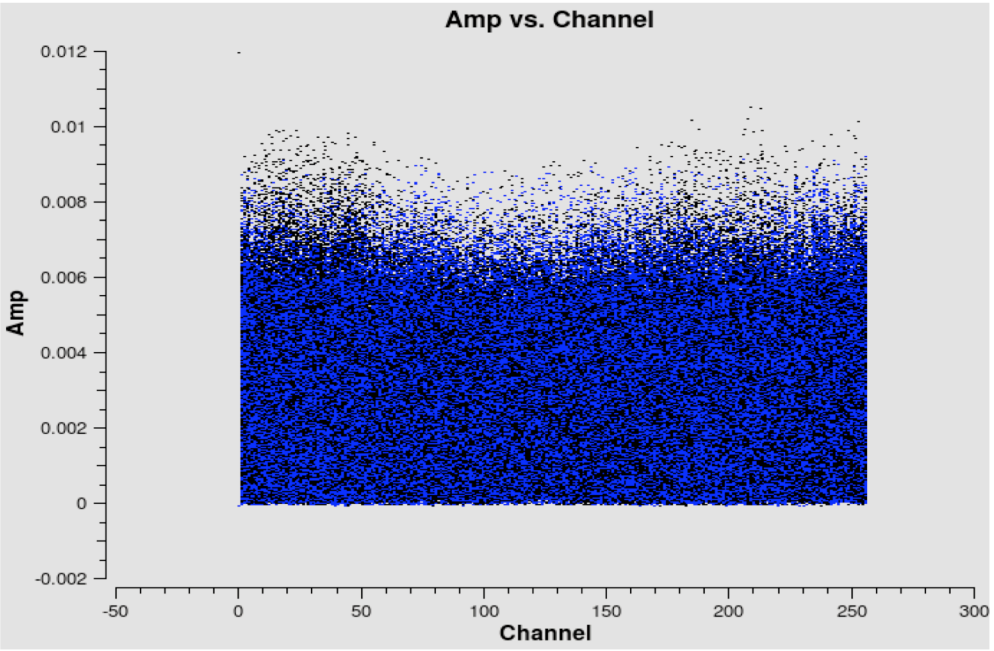
rficonsole



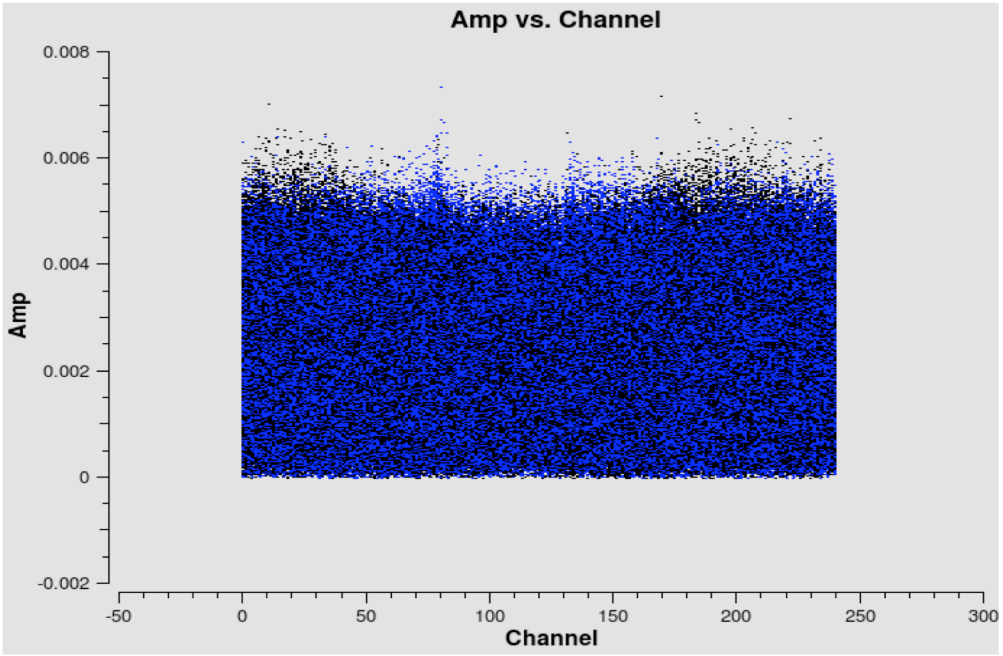
MADFlagger

SB50 of L2009_16167

rficonsole vs. MADFlagger



rficonsole



MADFlagger

SB50 of L2009_16167

rficonsole vs. MADFlagger

- Currently, rficonsole is slower (by a factor of ~2.5) than the MADFlagger in the imaging pipeline
- rficonsole speed is limited by i/o
  - best performance achieved by running one subband with 7 threads/instances per compute node. Use:

```
> rficonsole -j 7 <input.ms>
```

Remaining RFI Issues

- rficonsole works very well, but can still miss:
  - long-duration, broadband RFI
  - RFI with low amplitude
- Other RFI detection strategies may be implemented at a later date. E.g.,
  - fringe fitting: allows RFI that is constant in time and location to be subtracted from visibility data
  - factor analysis: uses multiple stations simultaneously to detect weak RFI (current methods flag baseline by baseline)

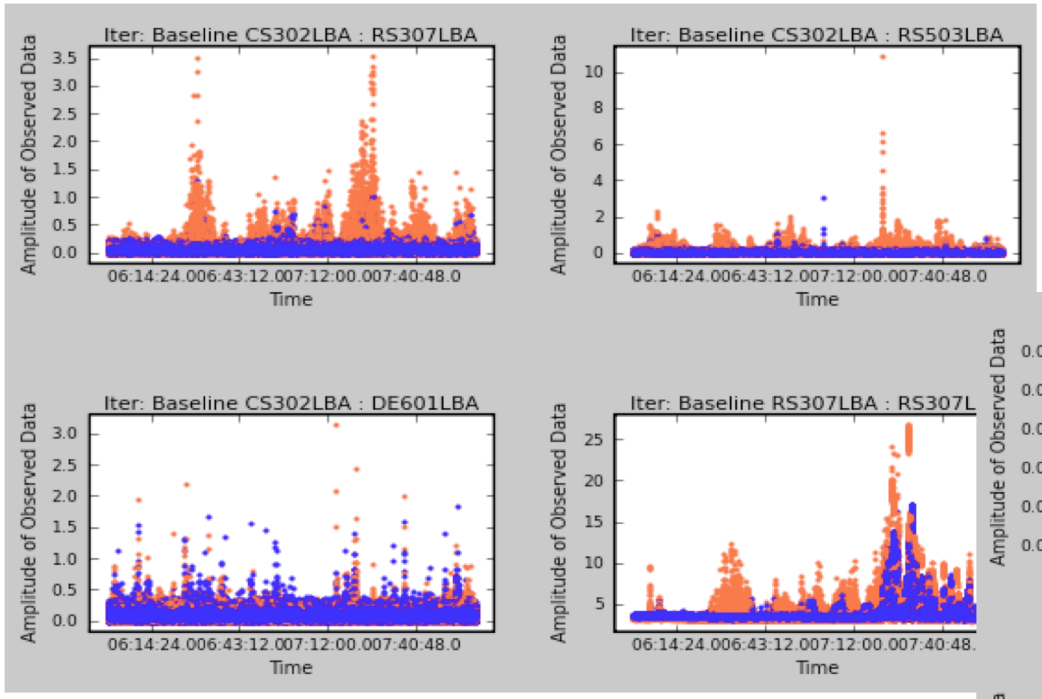
Data Visualization

- Plots of visibility amplitudes vs. time or channel are useful for checking RFI flagging
  - `casaplotms`: the CASA plotter
- Often, it is helpful to see “images” of amplitude as a function of channel and time:
  - “`>python /home/rafferty/plot_flags.py <MS>`” simple plotting script using pyrap (run “`use Pythonlibs`” first to initialize python libraries)

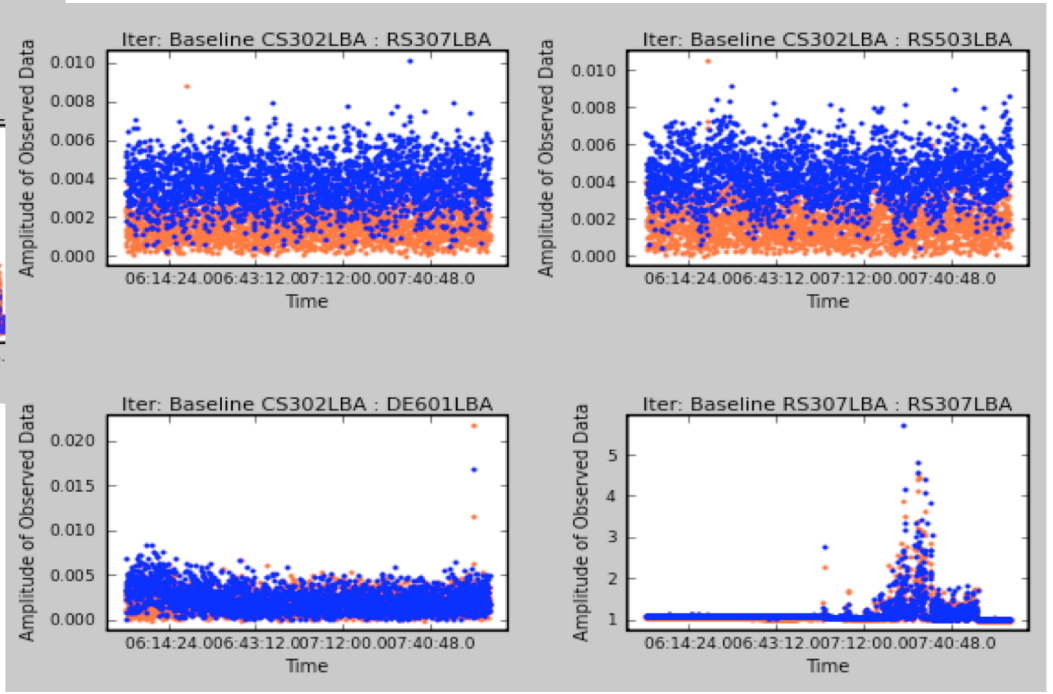
Compression

- Compression is simply an averaging of the visibility data in time and/or frequency
- It is useful for
  - reducing data volume
  - speeding up calibration and imaging
- However, compression tends to severely hinder RFI identification

Example of RFI and Compression



No flagging or compression



Compression, but no flagging

Basic Compression Guides

- Some limits to compression are set by bandwidth and time smearing
  - For a single subband of LOFAR data, bandwidth smearing generally does not produce a large effect when imaging sources in primary beam
  - But, can have large effect on bright sources in the sidelobes (e.g., Cygnus A)
- Limits on time averaging are set primarily by the need to track ionospheric changes on times scales of a few minutes or less

Compression in NDPPP

- NDPPP is used for compression (can be done in CASA using “split”)
- Parameters are set in the parset file. E.g.:

```
avg1.type = squash
avg1.freqstep = 256
avg1.timestep = 1
```

  – this will compress all 256 channels of the given subband into one channel, but will not compress in time

Summary

- RFI in post-correlation LOFAR data is a serious but manageable problem
- Two tools are available for flagging:
  - NDPPP:
    - Uses the MADFlagger
    - Faster for pipeline, but poorer performance overall
  - AOFlagger
    - run with “> rficonsole <ms>”
    - Slower for many subbands but better performance
- NDPPP is used for compression, preferably after flagging
- See Cookbook for details of all tasks