



# Radio Frequency Interference (RFI): Identification and Excision

Anna D. Kapińska (NRAO)



# What is RFI?

## RFI – Radio Frequency Interference

A disturbance caused by various sources emitting around our targeted frequencies that affect our data, introducing noise. Often RFI is stronger than the science data, and hinders signal we are after.

# UNITED STATES FREQUENCY ALLOCATIONS

## THE RADIO SPECTRUM

**RADIO SERVICES COLOR LEGEND**

	AERONAUTICAL MOBILE		INTER-SATELLITE		RADIO ASTRONOMY
	AERONAUTICAL MOBILE SATELLITE		LAND MOBILE		RAILROAD OPERATIONAL SATELLITE
	AERONAUTICAL RADIONAVIGATION		LAND MOBILE SATELLITE		RADIOLOCATION
	AMATEUR		MARITIME MOBILE		RADIOLOCATION SATELLITE
	AMATEUR SATELLITE		MARITIME MOBILE SATELLITE		RADIONAVIGATION
	BROADCASTING		MARITIME RADIONAVIGATION		RADIONAVIGATION SATELLITE
	BROADCASTING SATELLITE		METEOROLOGICAL		SPACE OPERATION
	EARLY RESEARCH SATELLITE		METEOROLOGICAL SATELLITE		SPACE RESEARCH
	FIXED		MOBILE		STANDARD FREQUENCY AND TIME SIGNAL
	FIXED SATELLITE		MOBILE SATELLITE		STANDARD FREQUENCY AND TIME SIGNAL SATELLITE

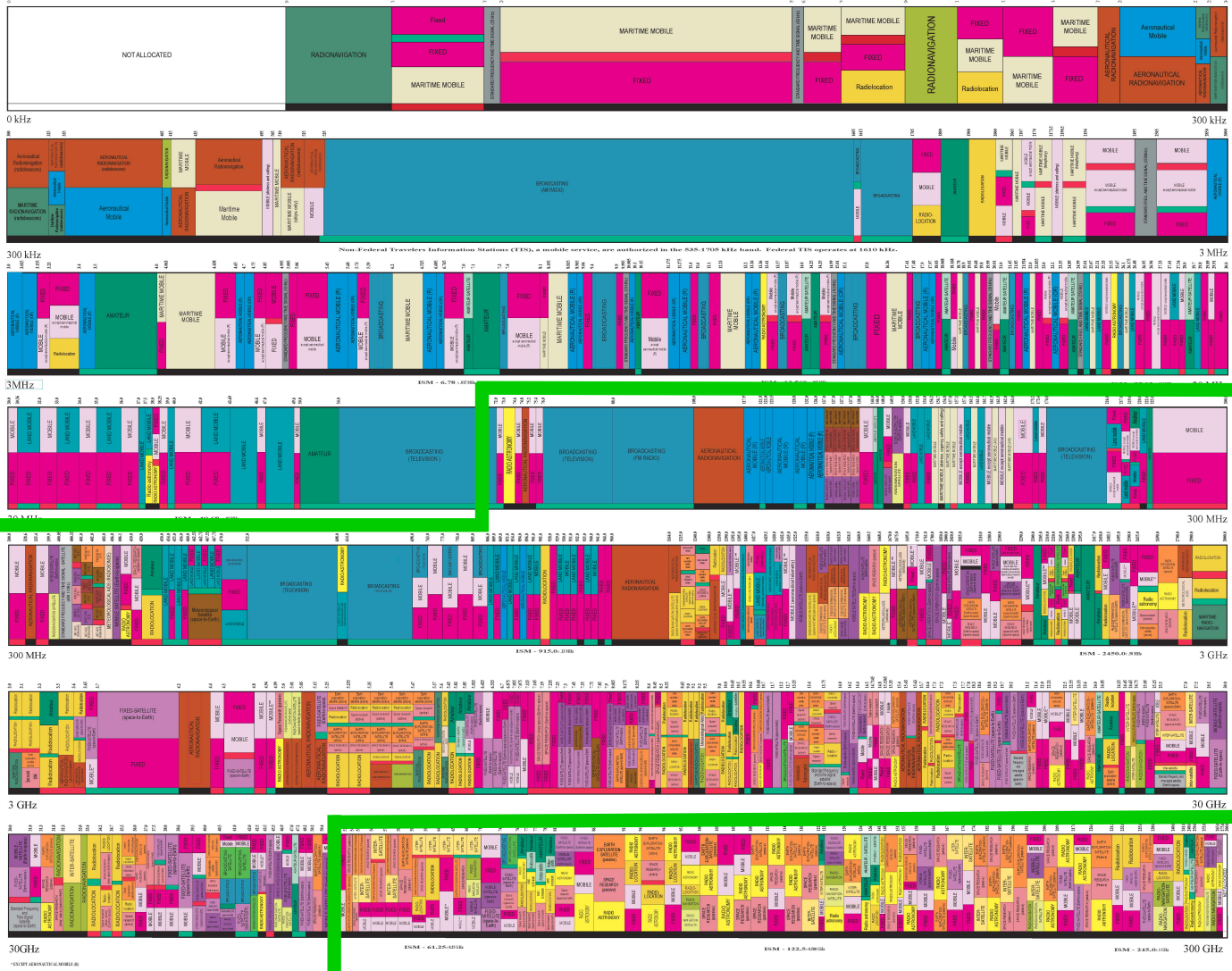
**ACTIVITY CODE**

	FEDERAL EXCLUSIVE		FEDERAL/STATE/LOCAL SHARED
	NON-FEDERAL EXCLUSIVE		

**ALLOCATION USAGE DESIGNATION**

SERVICE	EXAMPLE	DESCRIPTION
Primary	FIXED	Cable Lines
Secondary	Mobile	Land Mobile and Land Mobile Satellite

U.S. DEPARTMENT OF COMMERCE  
National Telecommunications and Information Administration  
Office of Spectrum Management  
JANUARY 2016



VLA frequency range

PLEASE NOTE: THIS SERVICE LIST IS FOR INFORMATIONAL PURPOSES ONLY. IT DOES NOT REPRESENT THE ACTUAL ALLOCATION OF SPECTRUM TO USERS.





# UNITED STATES FREQUENCY ALLOCATIONS

## THE RADIO SPECTRUM

INFORMATION

**RADIO SERVICES COLOR LEGEND**

AIR NAVIGATION MOBILE	INTER-SATELLITE	RADIO ASTRONOMY
AIR NAVIGATION MOBILE SATELLITE	LAND MOBILE	RADIO DETERMINATION SATELLITE
AIR NAVIGATION RADIO NAVIGATION	LAND MOBILE SATELLITE	RADIO LOCALIZATION
AMATEUR	MARITIME MOBILE	RADIOLOCATION SATELLITE
AMATEUR SATELLITE	MARITIME MOBILE SATELLITE	RADIONAVIGATION
BROADCASTING	MARITIME RADIONAVIGATION	RADIONAVIGATION SATELLITE
BROADCASTING SATELLITE	METEOROLOGICAL	SPACE OPERATION
EARLY RESEARCH SATELLITE	METEOROLOGICAL SATELLITE	SPACE RESEARCH
FIXED	MOBILE	STANDARD FREQUENCY AND TIME SIGNAL
FIXED SATELLITE	MOBILE SATELLITE	STANDARD FREQUENCY AND TIME SIGNAL SATELLITE

**ACTIVITY CODE**

FEDERAL EXCLUSIVE    FEDERAL/STATE SHARED

NON-FEDERAL EXCLUSIVE

**ALLOCATION USAGE DESIGNATION**

SERVICE	EXAMPLE	DESCRIPTION
Primary	FIXED	Capital Letter
Secondary	MOBILE	1st Capital and lower case letters

This chart is a graphic representation in part of the Table of Frequency Allocations published by the FCC and ITU. It does not purport to reflect all aspects of U.S. law and may change from time to time. It is not a legal document. Details for specific allocations can be found in the FCC's Commission Decision on Frequency Allocations.

**U.S. DEPARTMENT OF COMMERCE**  
National Telecommunications and Information Administration  
Office of Spectrum Management  
JANUARY 2016



50 GHz

PLEASE NOTE: THIS CHART IS LIMITED TO THE FREQUENCY BANDS LISTED. BANDS NOT LISTED ARE NOT NECESSARILY UNAVAILABLE TO THE FEDERAL GOVERNMENT.





# RFI at the VLA

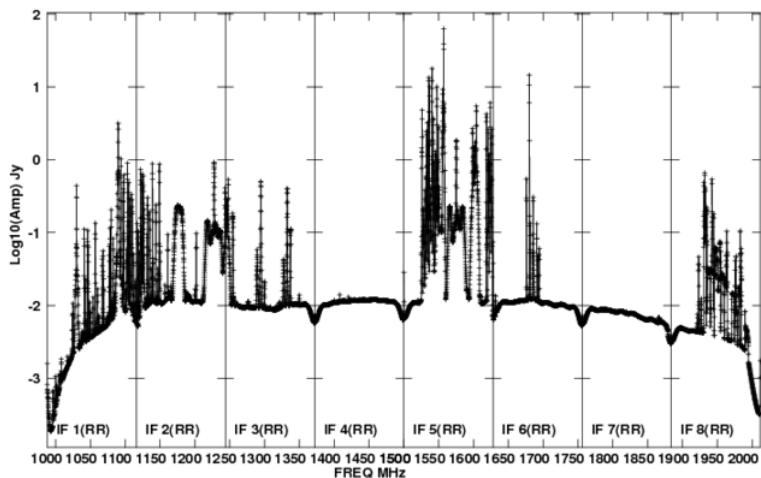
<https://science.nrao.edu/facilities/vla/docs/manuals/obsguide/rfi>

Biggest issue at lower frequency bands: 4, P, L, S, C, and D-configuration.  
But it does not mean it doesn't exist at higher frequency bands.

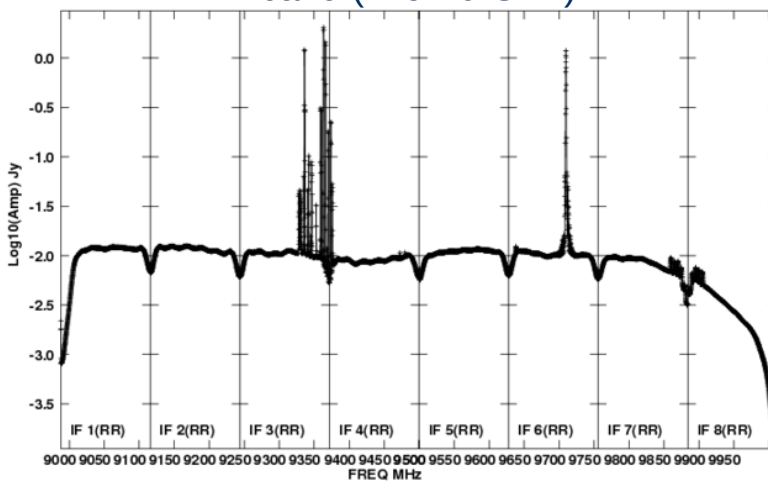
Can be internal or external. Internally-generated RFI is minimised, your NRAO staff working hard to eliminate these!

Examples of RFI in the VLA wide band observations (July 2014, D configuration)

L band



X band (in 9-10 GHz)



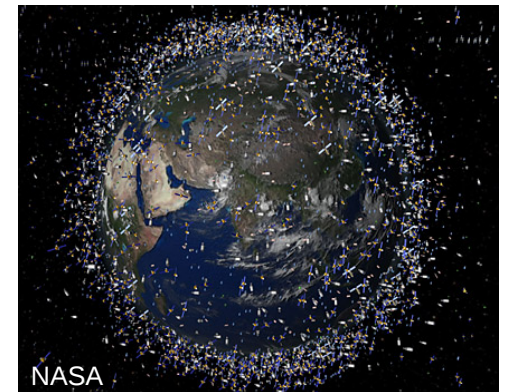
# Satellite Transmissions & Clarke Belt

Earth is now heavily surrounded by the satellites.

2000+ satellites orbiting Earth, hundreds along the *Clarke Belt* (zone of geosynchronous / geostationary satellites).

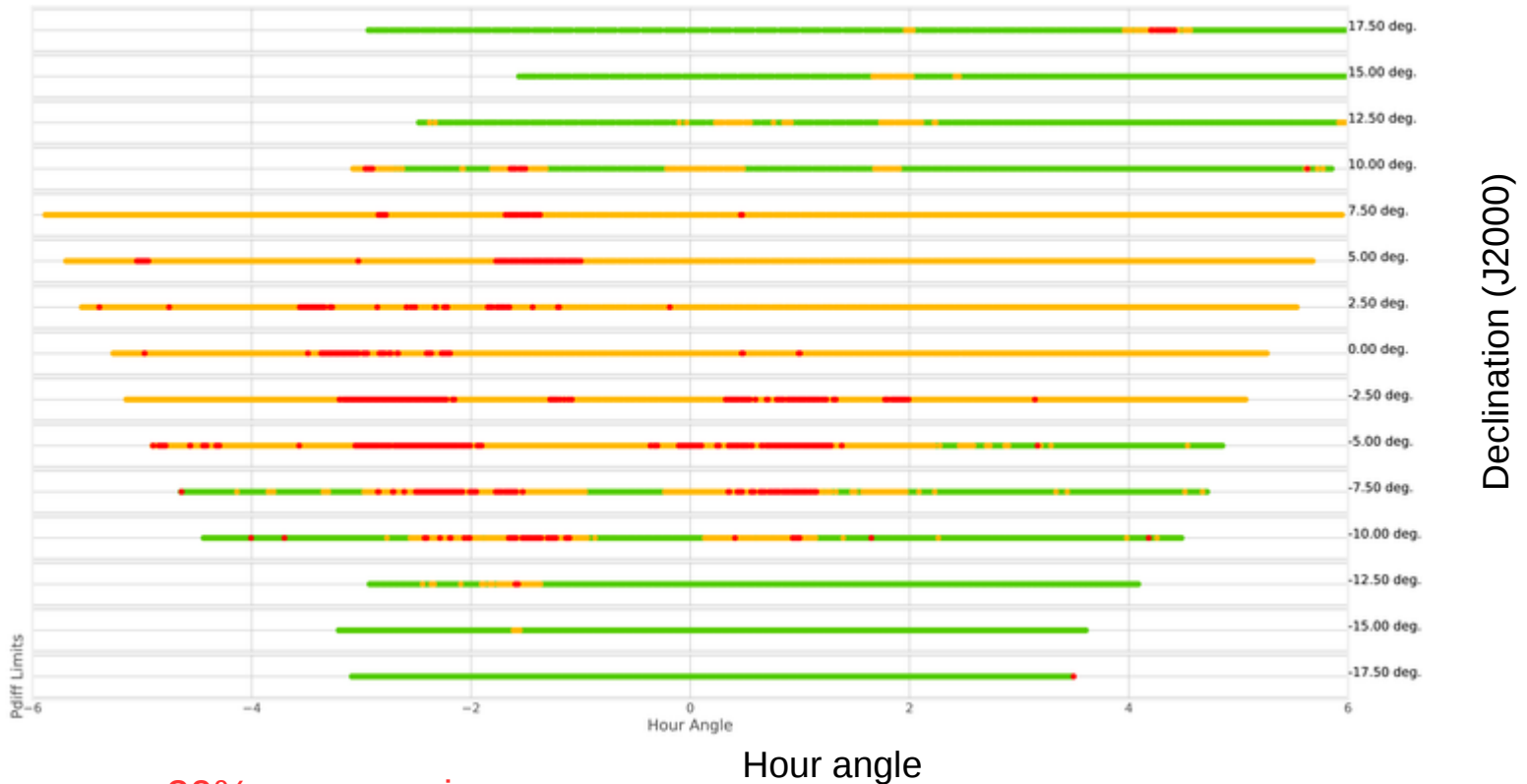
How does it affect VLA data?

- significant degradation of data can occur if VLA antenna observes within  $10^\circ$  of the satellite, @VLA Clarke Belt at declinations  $-15^\circ$  to  $+5^\circ$
- mainly S, C, Ku, K and Ka bands
- in C, X and Ku bands satellites can also saturate the 3-bit samplers (special 8/3-bit set up required)



# Satellite Transmissions & Clarke Belt

S band (2 – 4 GHz) survey of satellite interference at VLA (conducted in 2016/2017)





# Finding RFI in your data

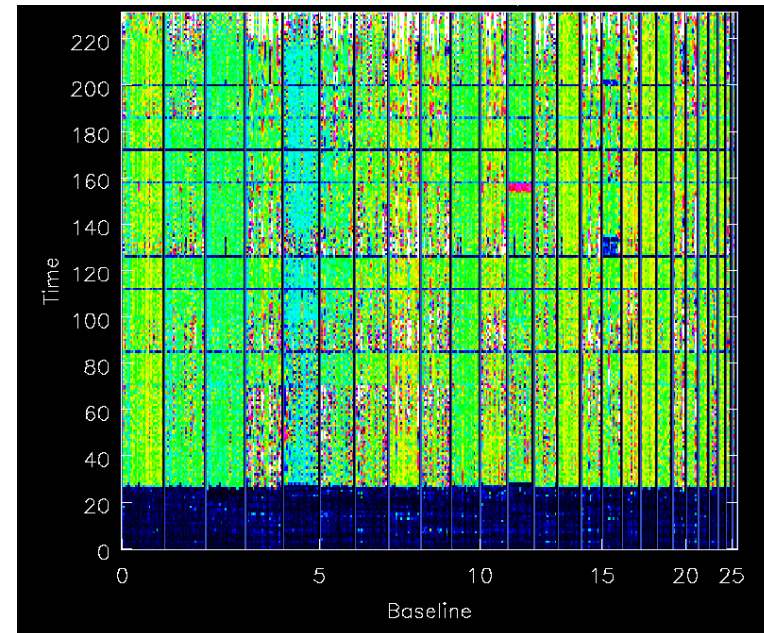
Two tasks allow you to visually inspect the data: `plotms()` and `msview()`

`plotms()` → Lorant's talk this morning (+ will do some interactively soon)

`msview()` → displays data in 3D fashion, where *intensity* of the plotted data represents *amplitude*, *phase*, etc, and x/y axes are time/baseline

```
>> msview() demonstration <<
```

Remember: if you leave off strong RFI in your data, the images will have issues such as 'ripples', high noise etc. But this is also how you can check for the remaining, offending RFI in your data: **image!**



# CASA Flagging tasks

`flagcmd()` → apply flags info on which stored in external file

Example: Online Flags

→ issues recorded by the operators such as *slew*, *subreflector*, *focus errors*

During downloading data from NRAO archive → apply online flags

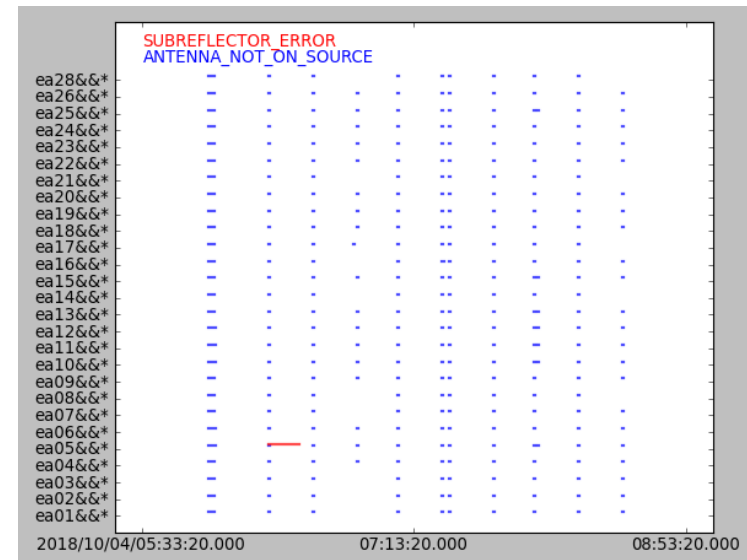
Apply telescope flags:

Apply flags generated during observing

But if you choose not to do that before getting archive data, these flags can be applied afterwards e.g. with `flagcmd()`.

Or you may want to plot the online flags to inspect them

`flagcmd(vis='drwRFI.ms', action='plot')`



# CASA Flagging tasks

**flagdata ()** → most often used task to flag data

<i>Deterministic</i>	<b>mode = 'manual'</b>	#Flag subset of data based on MS-selection syntax
	<b>mode = 'unflag'</b>	#Unflag data with MS-selection syntax ( <b>careful!</b> )
	<b>mode = 'quack'</b>	#Flag data at the beginning/end of a scan
	<b>mode = 'shadow'</b>	#Flag baselines with shadowed antennas
	<b>mode = 'elevation'</b>	#Flag data between specified elevation limits
	<b>mode = 'clip'</b>	#Threshold-based flagging on data expressions (incl. flagging zero-amplitude data)
<i>Autoflag</i>	<b>mode = 'tfcrop'</b>	#Find and flag outliers on the 2D time-frequency plane
	<b>mode = 'rflag'</b>	#Find and flag outliers via sliding-window RMS filters
	<b>mode = 'extend'</b>	#Extend flags around existing ones
<i>Operational</i>	<b>mode = 'summary'</b>	#Count existing flags and return statistics
	<b>mode = 'list'</b>	#Apply flags from external list



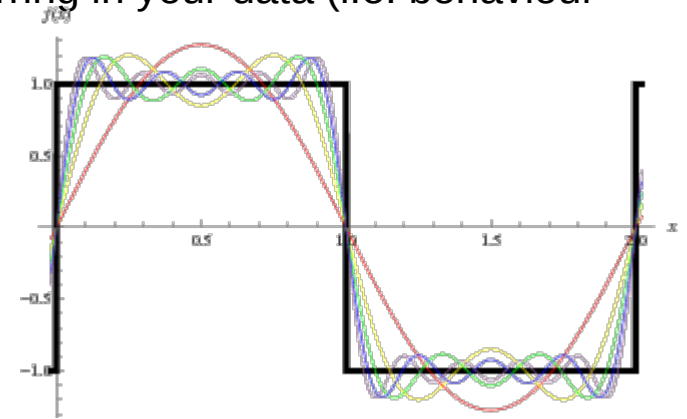
# Little help first: Hanning smoothing

Do not use this on spectral line data!

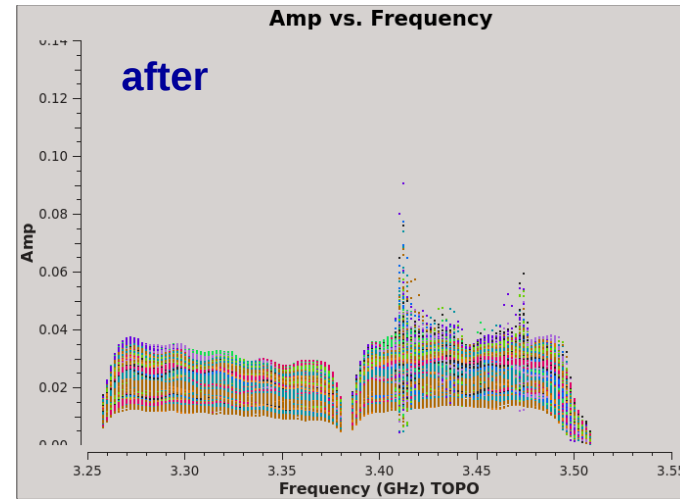
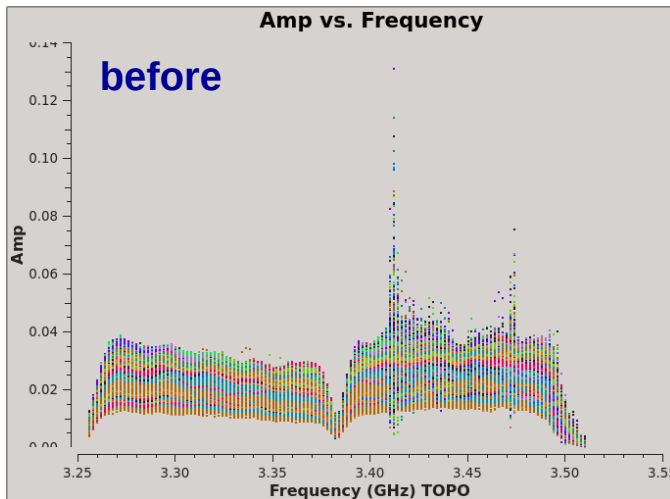
Strong RFI can give rise to the Gibbs phenomenon occurring in your data (i.e. behaviour of Fourier series at a discontinuity).

This is seen as 'ringing' → a pattern spreading to channels neighbouring the strong narrow RFI spike.

Hanning smoothing algorithm, `hanningsmooth()`, will:  
→ remove amplitude spikes and reduce the ringing,  
reducing number of affected channels  
→ but also reduce spectral resolution by a factor of 2



Our data  
→



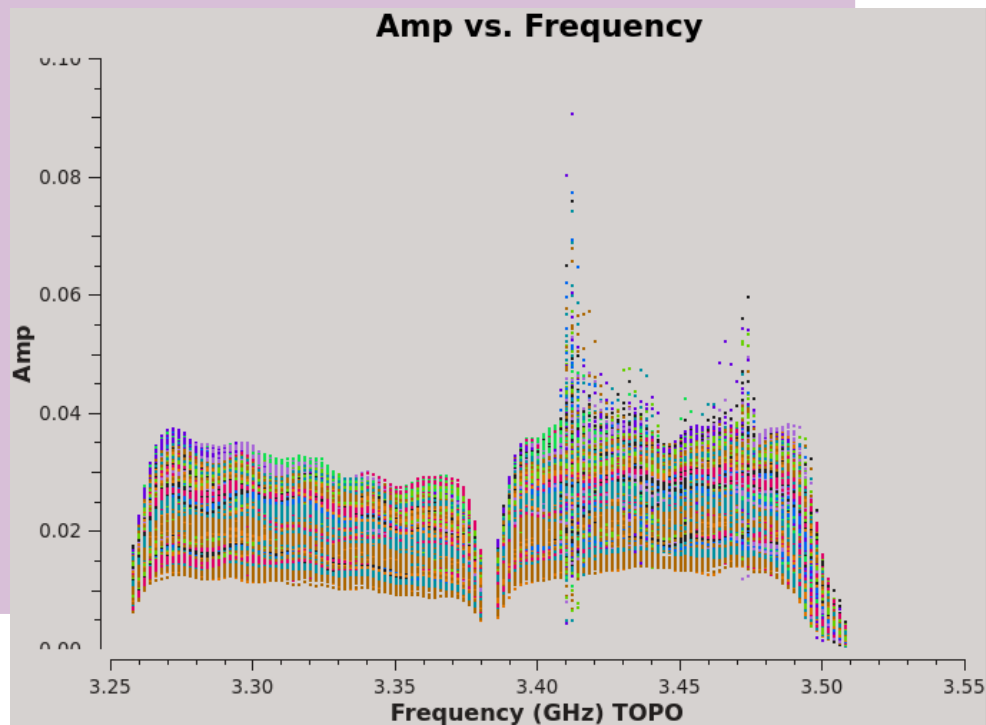
Wolfram

# Manual flagging in action

Let us first use `plotms()` to see where the RFI is

```
# In CASA
default plotms
vis='drwRFI.ms'
xaxis='frequency'
yaxis='amp'
avgtime='1e4'
coloraxis='Antenna1'
customsymbol=True
symbolshape='circle'
symbolsize=3
inp

go
```



# Manual flagging in action

Let us first use `plotms ()` to see where the RFI is

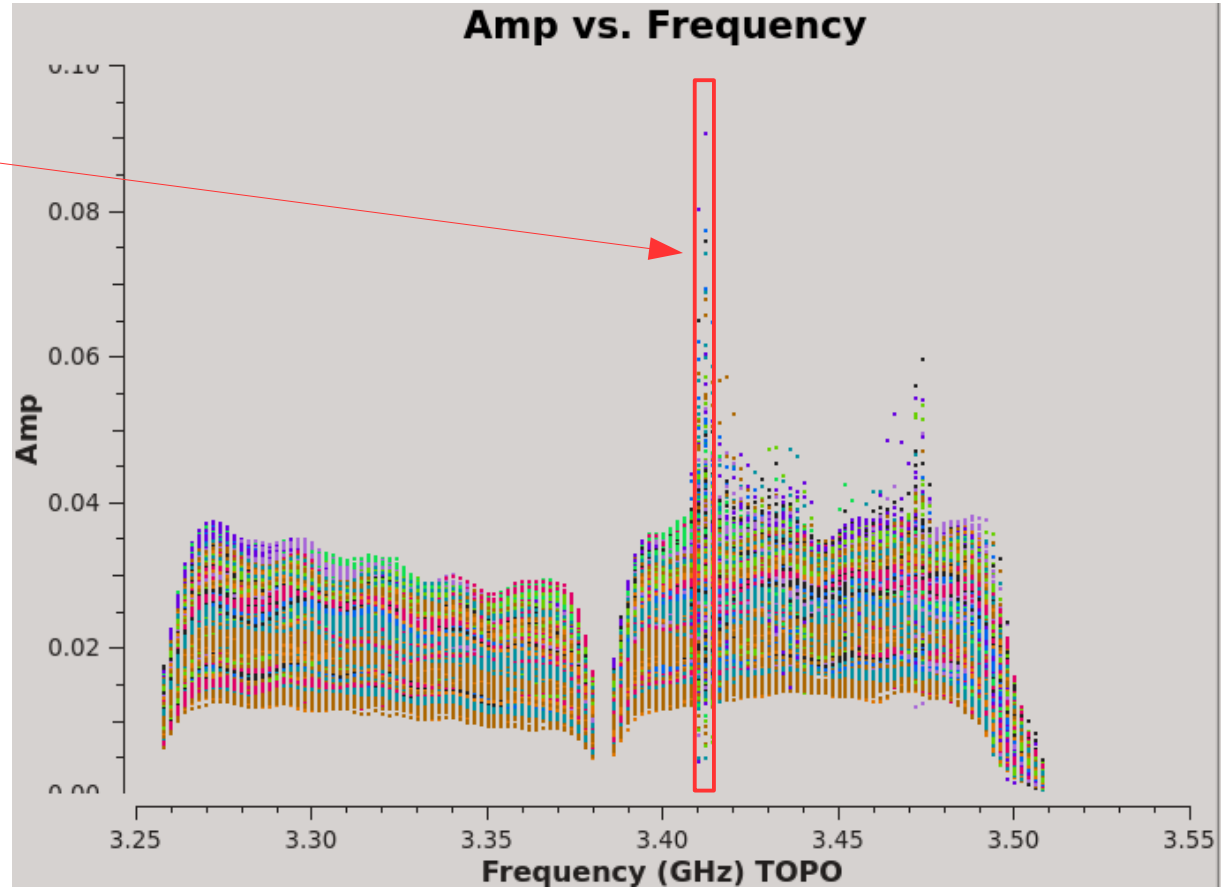
Let's remove this spike

It's in:

`spw=1, channels=13~15`

After inspecting (paging through) all the antennas I know that almost half of them are affected, so decided to fully remove those channels.

[ **Note:** See Lorant's talk for how to *Locate* the data you may want to flag ]





# Manual flagging in action

`flagdata ()` task with `mode = 'manual'`

```
# In CASA
default flagdata
vis='drwRFI.ms'
mode='manual'
spw='1:13~15'
flagbackup=True           # required if to restore previous flagging versions
inp

go
```

# Manual flagging in action

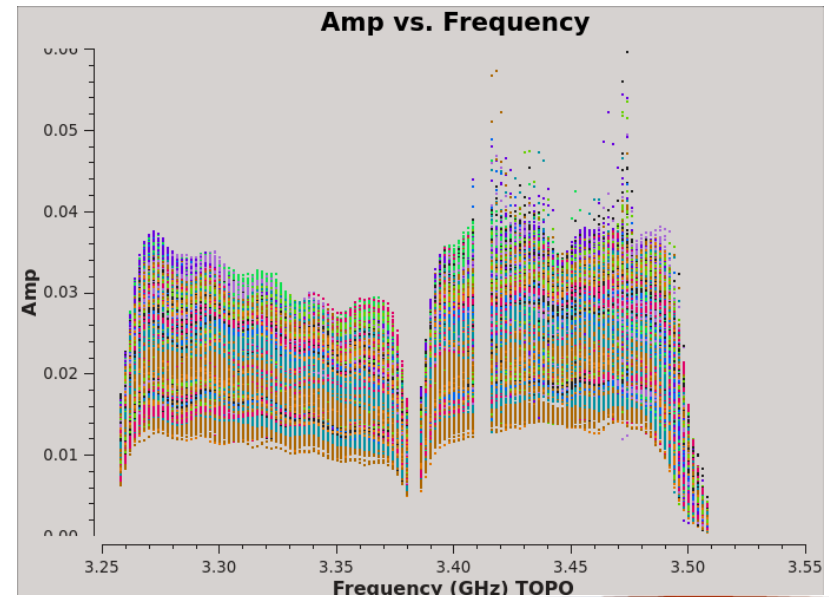
Let's now get back to `plotms()` to see the effect of our flagging

→ if yours is still up, check the *Reload* box and click *Plot*



→ if you closed yours then just type

```
tget plotms ()  
go
```



# Flagging Manager

```
# In CASA  
ls -l drwRFI.ms.flagversions
```

Terminal output:

```
drwxr-x--- 2 akapinsk nmstaff 4096 Sep 29 15:24 flags.flagdata_1/  
-rw-r----- 1 akapinsk nmstaff  51 Sep 29 15:24 FLAG_VERSION_LIST
```

# Flagging Manager

```
# In CASA
ls -l drwRFI.ms.flagversions
```

Terminal output:

```
drwxr-x--- 2 akapinsk nmstaff 4096 Sep 29 15:24 flags.flagdata_1/
-rw-r----- 1 akapinsk nmstaff   51 Sep 29 15:24 FLAG_VERSION_LIST
```

Of course you can also save the state of your flags manually at any time. So, let's save the flags after our manual flagging.

```
# In CASA
default flagmanager
vis='drwRFI.ms'
mode='save'
versionname='after_manual_1'
comment='after manual flagging'
inp

go
```

# Flagging Manager

Oh, oops, I made a mistake! How to restore data before the `flagdata()` run? Help!!



# Flagging Manager

Oh, oops, I made a mistake! How to restore data before the `flagdata()` run? Help!!

Yes, `flagdata()` has a mode you can use, but this can unleash a dragon!

```
mode = 'unflag'           # will unflag everything, not just  
                           your last flagdata() execution
```



Better to use → `flagmanager()`

# Flagging Manager

Oh, oops, I made a mistake! How to restore data before the `flagdata()` run? Help!!

```
# In CASA
default flagmanager
vis='drwRFI.ms'
mode='restore'
versionname='flagdata_1'
inp

go
```

Now reload your `plotms()` to see that we undid our flagging. If you closed your `plotms()` just run the following:

```
tget plotms
inp

go
```

# Flagging Manager

Let's again have a look at all the flag versions we have now for our ms file

```
# In CASA
tget flagmanager
mode='list'
inp

go
```

Logger output:

```

) ...ager:::+ #####
) ...ager:::+ ##### Begin Task: flagmanager #####
) ...nager::: flagmanager (vis="drwRFI.ms",mode="list",versionname="",oldname="",comment="",
) ...ager:::+ merge="replace")
) ...ger::open Table type is Measurement Set
) ...lagger::
) ...ager:: + MS : /lustre/aoc/sciops/akapinsk/DRW/presentation/drwRFI.ms
) ...lagger:: main : working copy in main table
) ...lagger:: flagdata_1 : Flags autosave on 2019-10-07 11:21:55
) ...lagger:: after_manual_1 : after manual flagging
) ...nager::: ##### End Task: flagmanager #####
) ...ager:::+ #####

```

*Note:* Restoring will not remove previous flag tables

# Keeping track of the amount of flagging

It's always good to keep track of how much you have flagged. Sometime you may even need that info for the publication (depending in the reviewer...)

```
# In CASA
default flagdata
vis='drwRFI.ms'
mode='summary'
inp
go
```

Logger output:



```
...etResult antenna ea12 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea13 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea14 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea15 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea16 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea17 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea18 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea19 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea20 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea21 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea22 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea23 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea24 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea25 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea26 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult antenna ea28 flagged: 107648 total: 1.60358e+06 (6.71%)
...etResult array 0 flagged: 2.25504e+06 total: 2.24502e+07 (10%)
...etResult correlation LL flagged: 1.12752e+06 total: 1.12251e+07 (10%)
...etResult correlation RR flagged: 1.12752e+06 total: 1.12251e+07 (10%)
...etResult field J0259+0747 flagged: 2.25504e+06 total: 2.24502e+07 (10%)
...etResult observation 0 flagged: 2.25504e+06 total: 2.24502e+07 (10%)
...etResult scan 11 flagged: 136080 total: 1.35475e+06 (10%)
...etResult scan 13 flagged: 126360 total: 1.25798e+06 (10%)
...etResult scan 15 flagged: 136080 total: 1.35475e+06 (10%)
...etResult scan 17 flagged: 311040 total: 3.09658e+06 (10%)
...etResult scan 19 flagged: 136080 total: 1.35475e+06 (10%)
...etResult scan 21 flagged: 136080 total: 1.35475e+06 (10%)
...etResult scan 23 flagged: 136080 total: 1.35475e+06 (10%)
...etResult scan 25 flagged: 311040 total: 3.09658e+06 (10%)
...etResult scan 7 flagged: 690120 total: 6.87053e+06 (10%)
...etResult scan 9 flagged: 136080 total: 1.35475e+06 (10%)
...etResult spw 0 flagged: 1.12752e+06 total: 1.12251e+07 (10%)
...etResult spw 1 flagged: 1.12752e+06 total: 1.12251e+07 (10%)
...etResult Total Flagged: 2.25504e+06 Total Counts: 2.24502e+07 (10%)
...gdata::: Flags are not written to the MS. (action='calculate')
...gdata::: ##### End Task: flagdata #####
...data::: + #####
```

# Flagdata quack mode

It's always a good idea to remove the first few seconds of each of your scans.

Why?

→ antennas have just been slewing to new source, the slewing is flagged (online flags), but often they need 'settling' time, the 'setting' time may result is some bad data

Easy to do, there is a special mode in the `flagdata ()` task:

```
mode = 'quack'  
quackinterval = 5.0      # in seconds  
quackmode = 'beg'       # this one means beginning of each scan,  
                          but there are also other modes: endb, end, etc.
```





# Auto-flagging: TFCrop

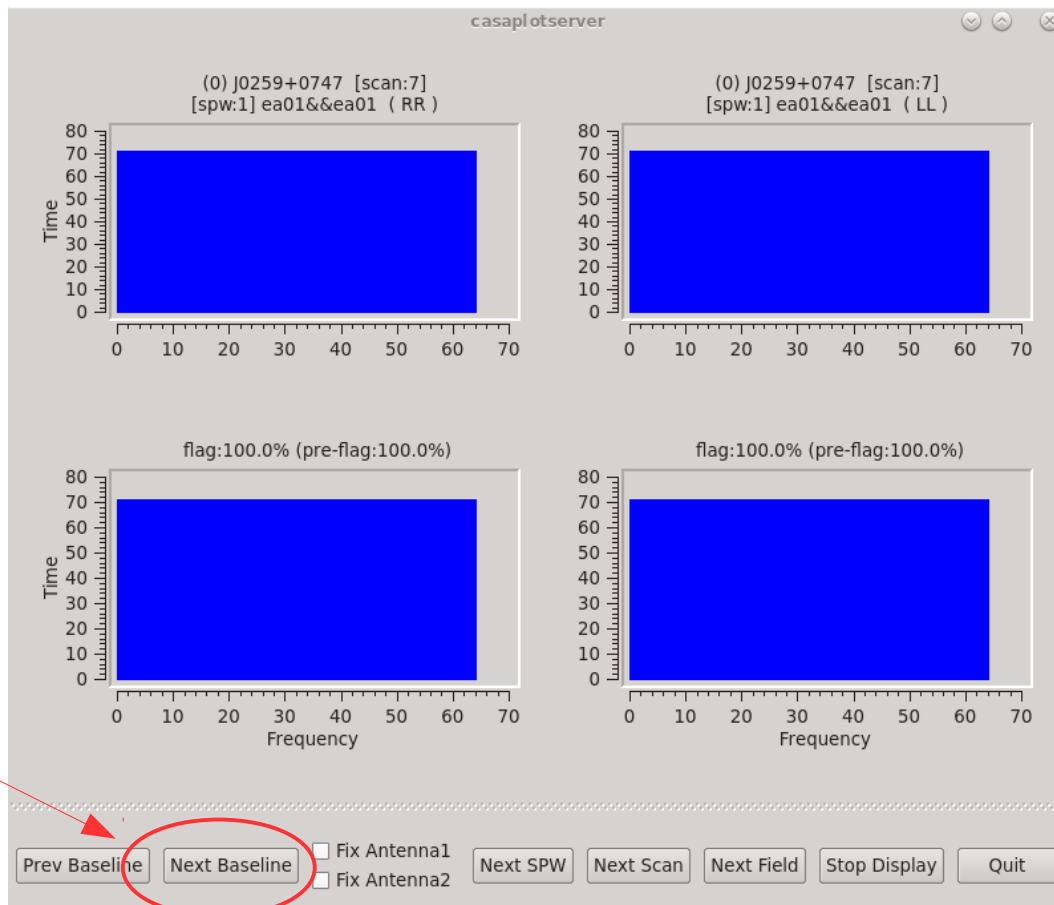
The auto-flagging algorithm `tfcrop` detects outliers in your data on the 2D time-frequency plane. Its statistics is based on each baseline independently.

```
# In CASA
tget flagdata
mode='tfcrop'
spw='1'
timecutoff=3.0
action='calculate'
display='both'
flagbackup=True      # required if to restore previous flagging versions
inp                  # stop here for a moment to see what other options you can set

go
```

# Auto-flagging: TFCrop

It's BLUE because all is flagged

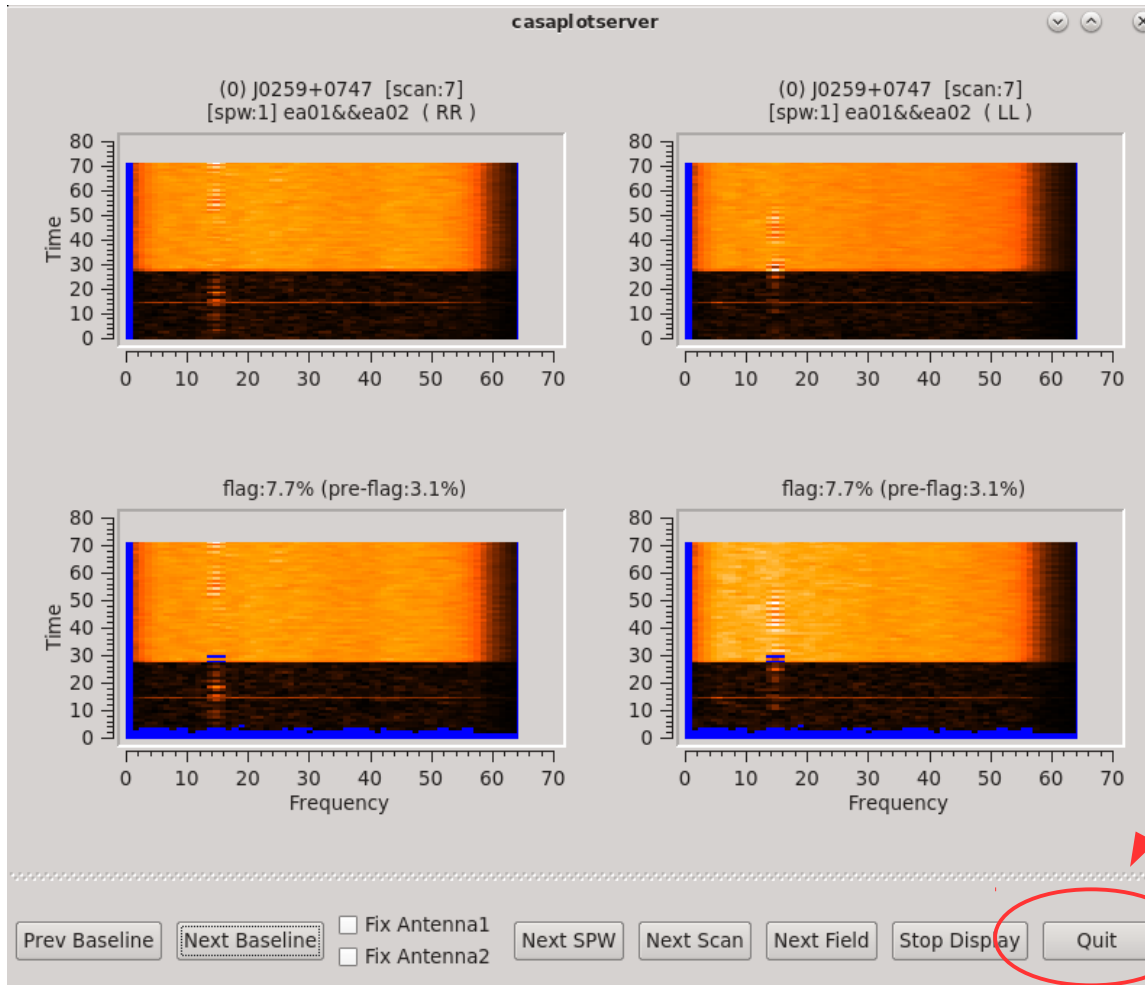


Navigate to the next baseline

# Auto-flagging: TFCrop

Current flagging status

After TFCrop flagging



Careful! *Quit* will kill the task!

*Stop Display* will just turn off display and let `flagdata()` to finish.

Since here we are just calculating flags so far, you can click either *Stop Display* or *Quit* to exit.

# Auto-flagging: TFCrop *cont.*

For best results you need to tune the `tfcrop` parameters, e.g.

```
timecutoff, freqcutoff      # threshold for finding outliers, in units of fit st.dev.  
flagdimension = 'freqtime' # direction(s) in which to calculate statistics  
channelavg, timeavg        # pre-average the data  
timefit = 'line'          # fitting function along time axis, line is default (ok: poly/line)  
freqfit = 'poly'         # fitting function along freq axis, poly is default (ok: poly/line)  
maxnpieces = n           # n order of polynomial in fitting functions above
```

Sometime you may also need to vary the parameters for e.g. spws or bands within the same data set.

Each data set is different and may need different parameter set up for best results

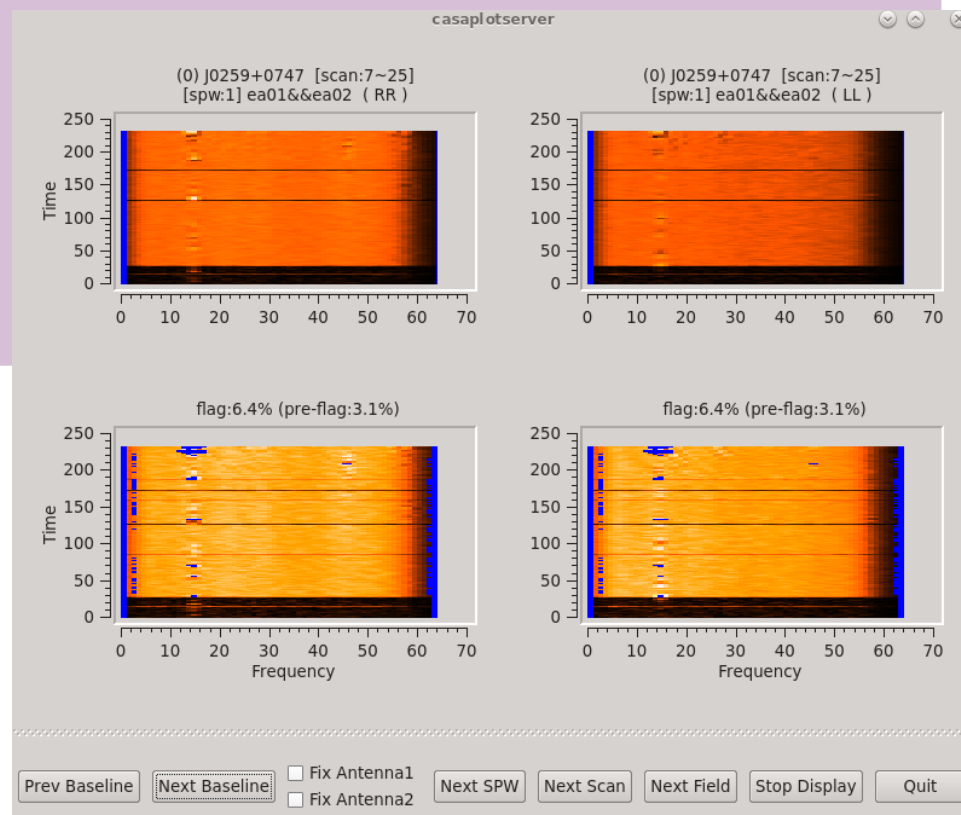
- make sure you inspect your data well
- know what you are dealing with, and
- choose the parameters accordingly.

And yes, you can run `tfcrop` on your data multiple times !

# Auto-flagging: TFCrop

Let's try some tuning to see if we can remove the most offending RFI

```
# In CASA
tget flagdata
maxnpieces = 4
usewindowstats = 'std'
halfwin = 2
inp
go
```





# Auto-flagging: TFCrop

If you are happy with the amount of flagging, then let us now apply it

```
# In CASA
tget flagdata
action='apply'
display=''
inp

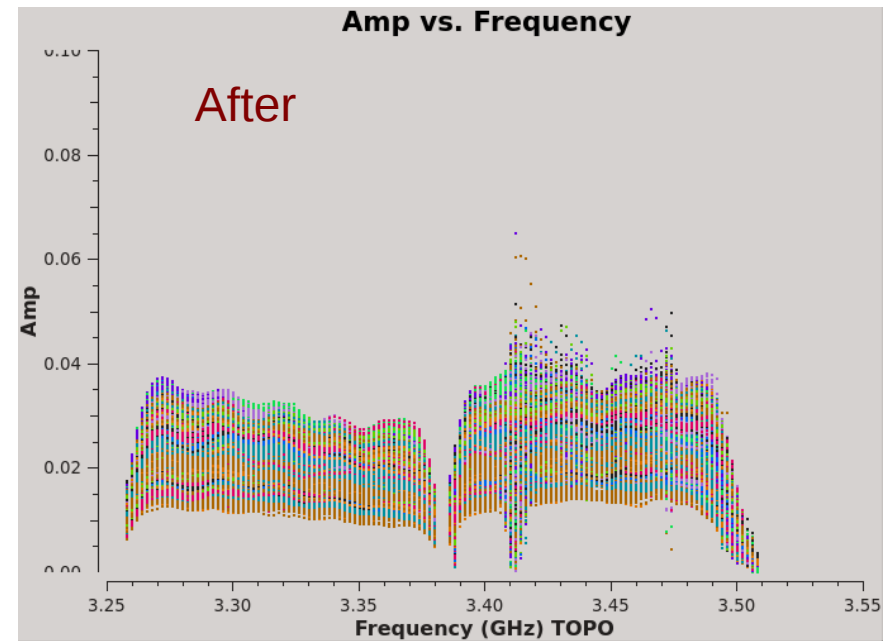
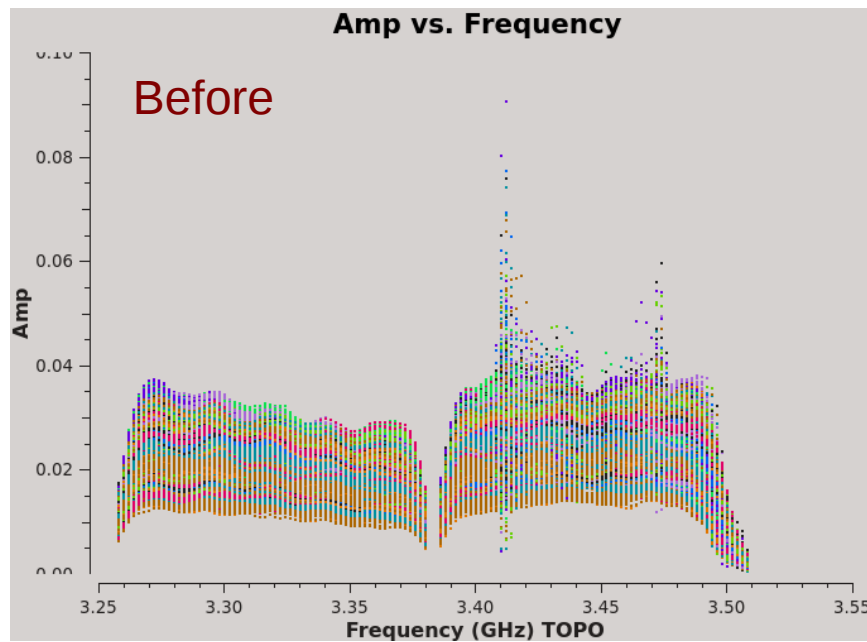
go
```

# Auto-flagging: Inspect results

```
# In CASA
tget plotms
inp

go
```

Or just check *Reload* box and click *Plot* if you still have `plotms()` open



# Extending your flags

Available as a standalone mode in `flagdata()`  
or as an `extendflags` parameter within `tfcrop` and `rflag` modes

→ will extend or grow flags accumulated in the MS file along time, frequency, polarisation, baseline etc

Flag extension:

→ e.g. if you applied flag only to RR product, you can extend that afterwards also to LL.

Flag growing, example parameters:

```
growtime=80.0 # for each channel flag entire timerange if >80% data flagged
growfreq=92.0 # for each time flag selected chans if >92% data flagged
growaround    # flag a data point if >4 neighbouring points are flagged
flagneartime  # flag a data point before and after a flagged one
```

Note: It is recommended flag extension is used when executing auto-flagging modes.

# Extending your flags

There is still yucky RFI there so let's try more

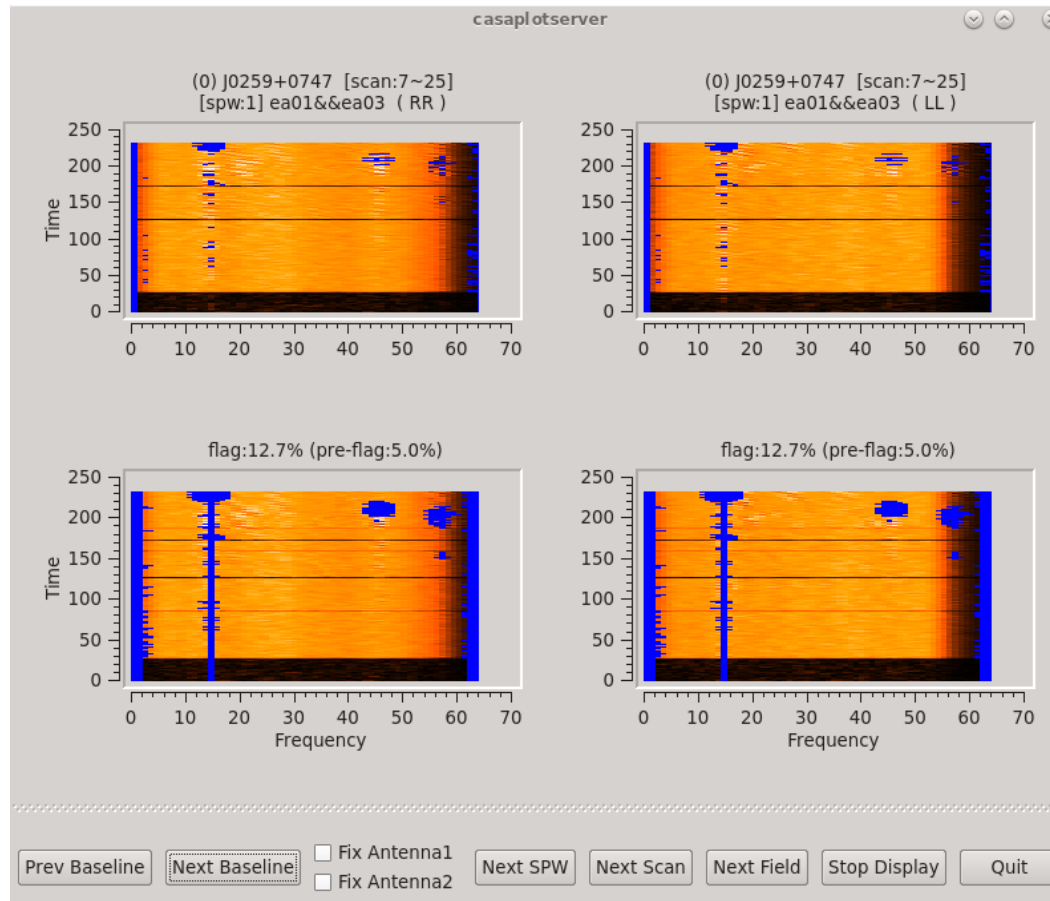
```
# In CASA
default flagdata
mode = 'extend'
combinescans = True
growtime = 30.0
growfreq = 30.0
growaround = True
flagneartime = True
flagnearfreq = True
action = 'calculate'
display = 'both'
inp

go
```

# Extending your flags

Woohoo, finally getting the remaining RFI removed!

3<sup>rd</sup> baseline plotted here





# Extending your flags

So, if you are now happy with the extra flagging, click *Quit* on the display, and let's apply these flags

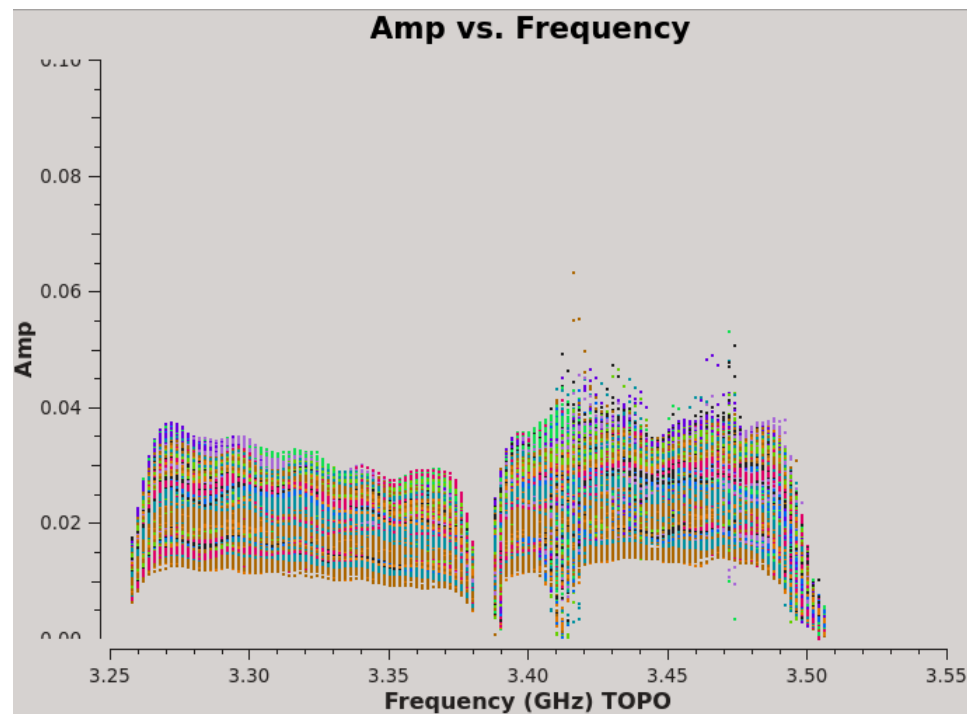
```
# In CASA
tget flagdata
action = 'apply'
display = ''
inp

go
```

# Extending flags: Inspect results

```
# In CASA  
tget plotms  
inp  
  
go
```

Current result:



# Auto-flagging: Rflag

The `rflag` algorithm detects outliers in your data based on local rms statistics.

Requires calibrated data!

The calibration pipeline uses Rflag

Step 1. `rflag` iterates through time chunks calculating local rms of *imag* and *real* visibilities within a sliding time window, and deriving a median rms across given time window

Step 2. `rflag` iterates through frequency chunks (channels) for each time chunk calculating rms of avg of *imag* and *real* visibilities

---

Again, for best results you need to tune the `rflag` parameters, e.g.

<code>winsize</code>	# number of timesteps in the sliding time window
<code>timedevscale, freqdevscale</code>	# st.dev. threshold for outlier flagging
<code>channelavg, timeavg</code>	# pre-average the data

Again, can be run multiple times on the same data set.

# TFCrop vs Rflag: which to use when?

Due to differences in the algorithms, worth executing both on the same data set.

	TFCrop	Rflag
How does it work?	→ search for RFI spikes above smooth base, per baseline	→ use local vs global stats to find outliers
Strong, spiky RFI	Great!	Good, but continuous RFI in time/freq needs tuning
Noisy RFI	Good only for bright RFI, won't work well for low noisy RFI spikes	Great!
Broadband RFI	Not robust, but possible with some tuning	Good for noisy RFI. Continuous RFI needs tuning.
Extended emission	Great! [each baseline treated separately]	Not great, biased by high flux density on short baselines
Raw, uncalibrated data	Yes	No
Calibrated data	Yes	Yes

# Auto-flagging: Spectral lines

Be thoughtful of your precious spectral lines! If you just run `tfcrop` or `rflag` **without adjusting** for the location of your spectral line(s), it/they will likely **be removed** during auto-flagging.

Solution:

1. You know exactly where your line is, note its location in `spw` and channel(s)
2. In the `flagdata()` task exclude that location with the `spw` parameter

Example:

- you have 2 spectral windows (0,1), each with 64 channels
- your line is in `spw=1`, channels=21~22
- exclude that location with the following format of the `spw` parameter

```
spw = '0, 1:0~20; 23~63'
```

Pipeline: not optimised for dealing with spectral lines

If you do not know where your lines are (e.g. you are searching for some) **DO NOT** run pipeline or auto-flagging on your data!

# A note on `statwt()`

DATA

MODEL

CORRECTED

WEIGHT

SIGMA

Typically the WEIGHT and SIGMA columns are set to some arbitrary values (e.g. 1), or are theoretically estimated from poorly known antenna and receiver properties.

`statwt()` will empirically measure the visibility scatter (e.g. as a function of time, antenna, and/or baseline) and use it to set WEIGHT and SIGMA

- it may be beneficial sometimes to weight down any remaining RFI in your data with `statwt()` prior to imaging

Requires calibrated data!

# Final points to remember

Online CASA flagging guide

[https://casaguides.nrao.edu/index.php/VLA\\_CASA\\_Flagging-CASA5.5.0](https://casaguides.nrao.edu/index.php/VLA_CASA_Flagging-CASA5.5.0)

- All data have some level of RFI, and with progressing technological evolution it will only get worse for us
- Inspect your data thoroughly, and remove the most obstructive RFI as this is what will affect the quality and noise of your images the most.
- Use combination of auto-flagging algorithms and manual flagging if required (but careful with spectral lines!!). Execute auto-flagging multiple times → statistics these algorithms rely on will change each time!

- You can flag all types of data:
  - \* the visibilities,
  - \* the weights, and
  - \* solutions in calibration tables





**[www.nrao.edu](http://www.nrao.edu)**  
**[science.nrao.edu](http://science.nrao.edu)**  
**[public.nrao.edu](http://public.nrao.edu)**

*The National Radio Astronomy Observatory is a facility of the  
National Science Foundation  
operated under cooperative agreement by Associated Universities,  
Inc.*