RFI Identification and Automatic Flagging

Urvashi Rau & Emmanuel Momjian

NRAO

VLA Data Reduction Workshop

27 – 31 October 2014

Outline

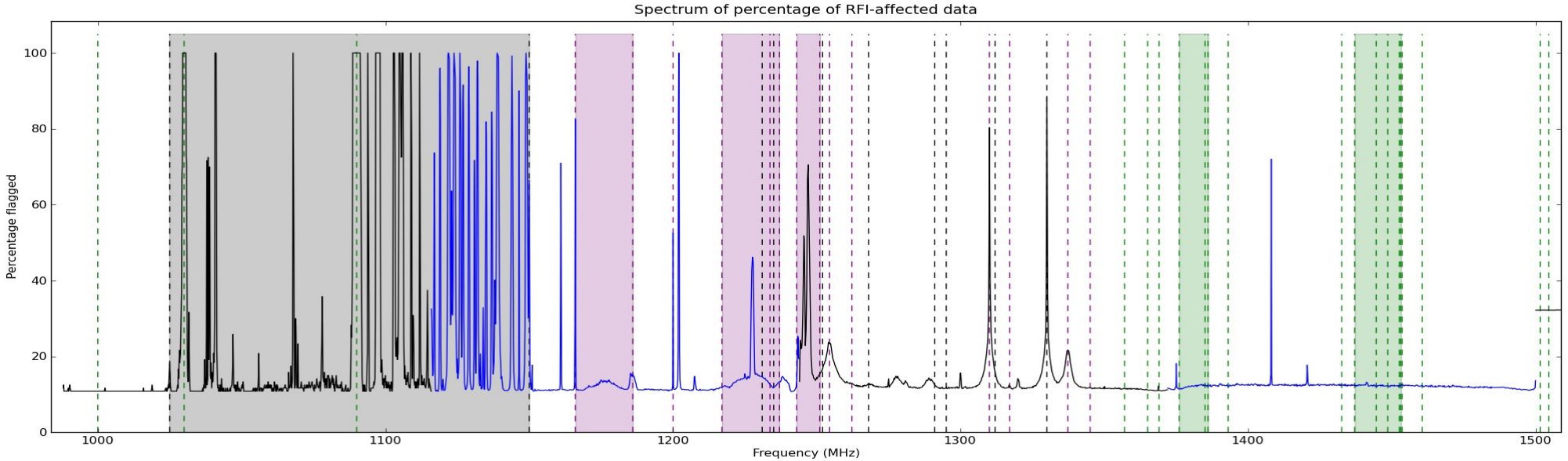
- RFI at the VLA + Online Flags
- Automatic Flagging Options and Strategies
- Some examples of what works when....

(31 slides)

RFI at the VLA + Online flags

Radio Frequency Interference : Terrestrial, Satellites, etc..

At L-Band, can use ~500 MHz with very rough flagging, ~800 MHz if done carefully.
RFI is less at higher frequencies. At X-band and beyond, RFI is minimal



https://science.nrao.edu/facilities/vla/ -- Guide to VLA Observing -- Radio Frequency Interference

Online + Deterministic Flags :

– slew, subreflector error, focus error
– operator logs of known bad antennas and time-ranges
– shadowing between antennas (elevation-dependent)
– exact zeros (from the correlator)
– known frequency ranges with bad RFI

Flagging Tasks - flagdata + flagcmd

Deterministic

mode='manual' or mode='unflag' # Use MS-selection syntax to pick subsets to flag/unflag

mode = 'quack' # Flag data at the beginning or end of a scan (operate on selected data).

mode = 'elevation' # Flag data between specified elevation limits (operate on selected data)

mode='shadow' # Flag baselines with shadowed-antennas

Autoflag

mode = 'clip' # Threshold-based flagging on data-expressions (ABS_RR, ABS_I, etc)

mode='tfcrop' # Automatic flagging : Find outliers on the 2D time-frequency plane

mode='rflag' # Find outliers based on sliding-window RMS filters

mode='extend' # Grow/extend flags around existing ones.

Operational

mode='summary' # Count existing flags and return a python-dictionary of counts per antenna, spw, etc

mode = 'list' # Supply a list of flag commands, built out of parameters of any other mode.

Run-modes : " apply / calculate " + runtime "display" of data before and after flagging, and reports.

Managing the Flags - Flagmanager

Flagging tasks (flagdata, flagcmd), applycal, flagmanager can make flag backup versions
These are the state of the flag of the data before the new flags are applied on the data.

Flag backups are stored in the directory: mydata.ms.flagversions

Flag backups are named as 'flags.the task creating it_version#'
flags.flagcmd_1
flags.flagdata_1
flags.before_applycal_1
flags.flagcmd_2
flags.flagdata_2
flags.flagdata_3
flags.mybackup (flagmanager backups get named by you, e.g. mybackup)

Flagmanager is the task to operate on flag backups
mode= 'list' to list existing flagtables
= 'save' to save flag column from vis to a specified flag file
= 'restore' to place the specified flag file into vis
= 'delete' to delete specified flag file
= 'rename' to rename a specified flag file

To restore flags:
versionname='flagcmd_1' restores flags.flagcmd_1

Flagging Modes : tfcrop

Detect outliers on the 2D time-freq plane.

For each baseline,

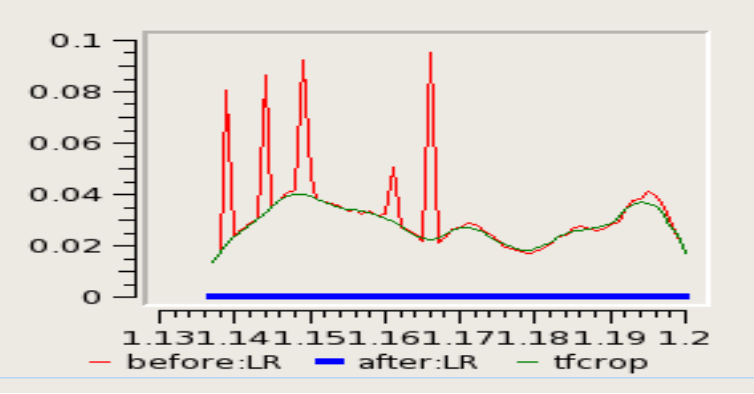
(1) Average visibility amplitudes along time dimension to form an average spectrum

(2) Calculate a robust piece-wise polynomial fit for the band-shape at the base of RFI spikes

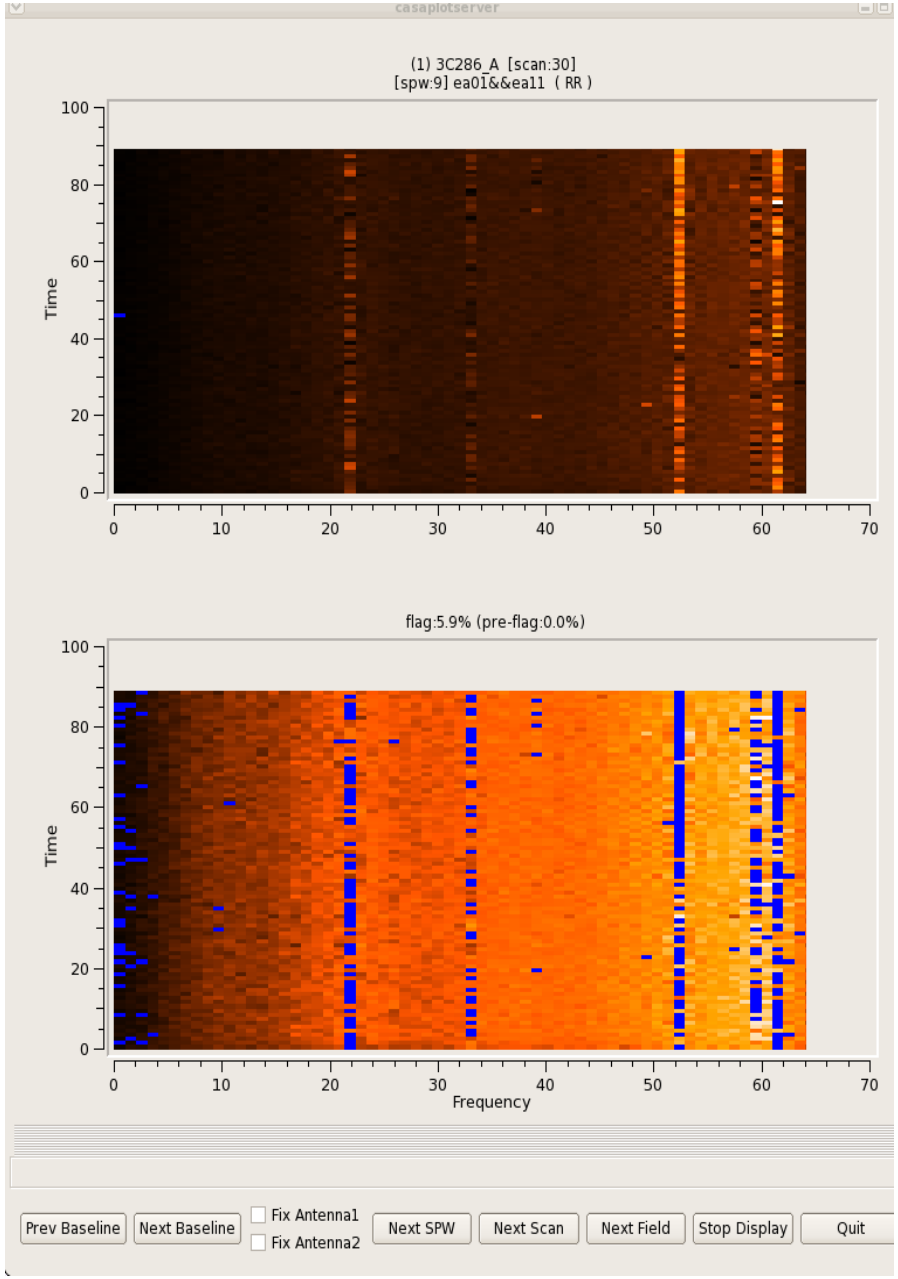
– Calculate 'sigma' of (data - fit).

(3) Flag points deviating from the fit by more than N-sigma

(4) Repeat (1-3) along the other dimension.



- Calculates and applies flags in one pass through the MS
- Can operate on un-calibrated data and bandshapes
- Flag extension can be done via mode='extend'



Flagging Modes : tfcrop

ntime = 'scan' or '2.0min' or 120.0 # Time range to use for each chunk

combinescans = True/False # if ntime > scan, join multiple scans

datacolumn = 'data', 'corrected', 'model', 'residual'

correlation = 'ABS_ALL' # Correlation expression. Augmented MS-selection syntax.
Expressions : 'ALL', 'RR','LL', 'I', 'Q','U','V', 'WVR'
Functions : 'ABS', 'ARG', 'RE', 'IM', 'NORM'
Ex : 'ABS_RR,LL,V'

timecutoff, freqcutoff = 3.0 # Scale-factor for deviation from fit, to compute thresholds

timefit, freqfit = 'line' or 'poly' # Fit a straight line, or piecewise polynomial

maxnpieces = 7 # maximum number of pieces in the polynomial fit (only for 'poly')

flagdimension = 'freqtime' # Directions in which to calculate stats 'timefreq','time','freq'

usewindowstats , halfwin # Options to use sliding-window statistics 'std' and 'sum'

Flagging Modes : rflag

(A close copy of RFLAG from AIPS (E.Greisen, 2011))

Detect outliers based on sliding-window RMS filters.

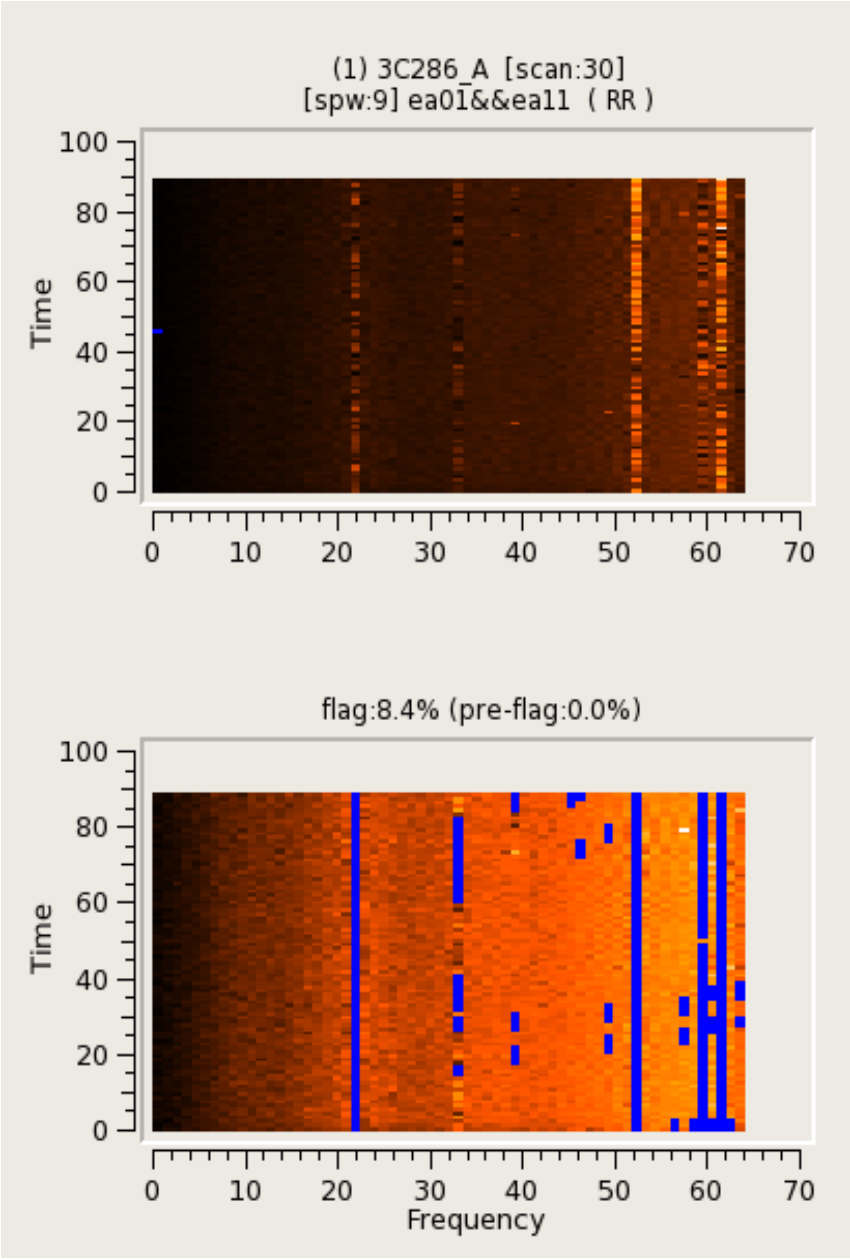
(1) Time-Analysis (for each channel)

(1a) Calculate local RMS of real and imag parts of visibilities within a sliding time window.
(1b) Calculate the median RMS across time windows, deviations of local RMS from this median, and the median deviation
(1c) Flag if local RMS > N x (medianRMS + medianDev)

(2) Spectral-Analysis (for each timestep)

(2a) Calculate Avg of real and imag parts of visibilities and their RMS across channels.
(2b) Calculate the deviation of each channel from this Avg, and the median-deviation.
(2d) Flag if deviation > N x medianDev

- Can be run with 1 or 2 passes through the data
- Requires calibrated data.
- Flag extension can be done via mode='extend'
- Can flag short baselines if there is extended emission
- Good for low level noisy RFI



Flagging Modes : rflag

ntime = 'scan' or '2.0min' or 120.0 # Time range to use for each chunk

combinescans = True/False # if ntime > scan, join multiple scans

datacolumn = 'data', 'corrected', 'model', 'residual'

correlation = 'ABS_ALL' # Correlation expression. Augmented MS-selection syntax.
Expressions : 'ALL', 'RR','LL', 'I', 'Q','U','V', 'WVR'
Functions : 'ABS', 'ARG', 'RE', 'IM', 'NORM'
Ex : 'ABS_RR,LL,V'

winsize = 3 # size of sliding time window

timedev = [[0, 2, 0.0534]] # time-series noise estimate [field, spw, noise-estimate]
freqdev = [ ] # spectral noise estimate

timedevscale = 5 # threshold scaling for timedev
freqdevscale = 5 # threshold scaling for freqdev

spectralmin = 1e+6 # flag whole spectrum if freqdev > spectralmin
spectralmax = 0.0 # flag whole spectrum if freqdev < spectralmax

outfile = 'thresholds.txt' # text file to hold output thresholds from rflag.

Flagging Modes : extend

Example :

Flag only RR,

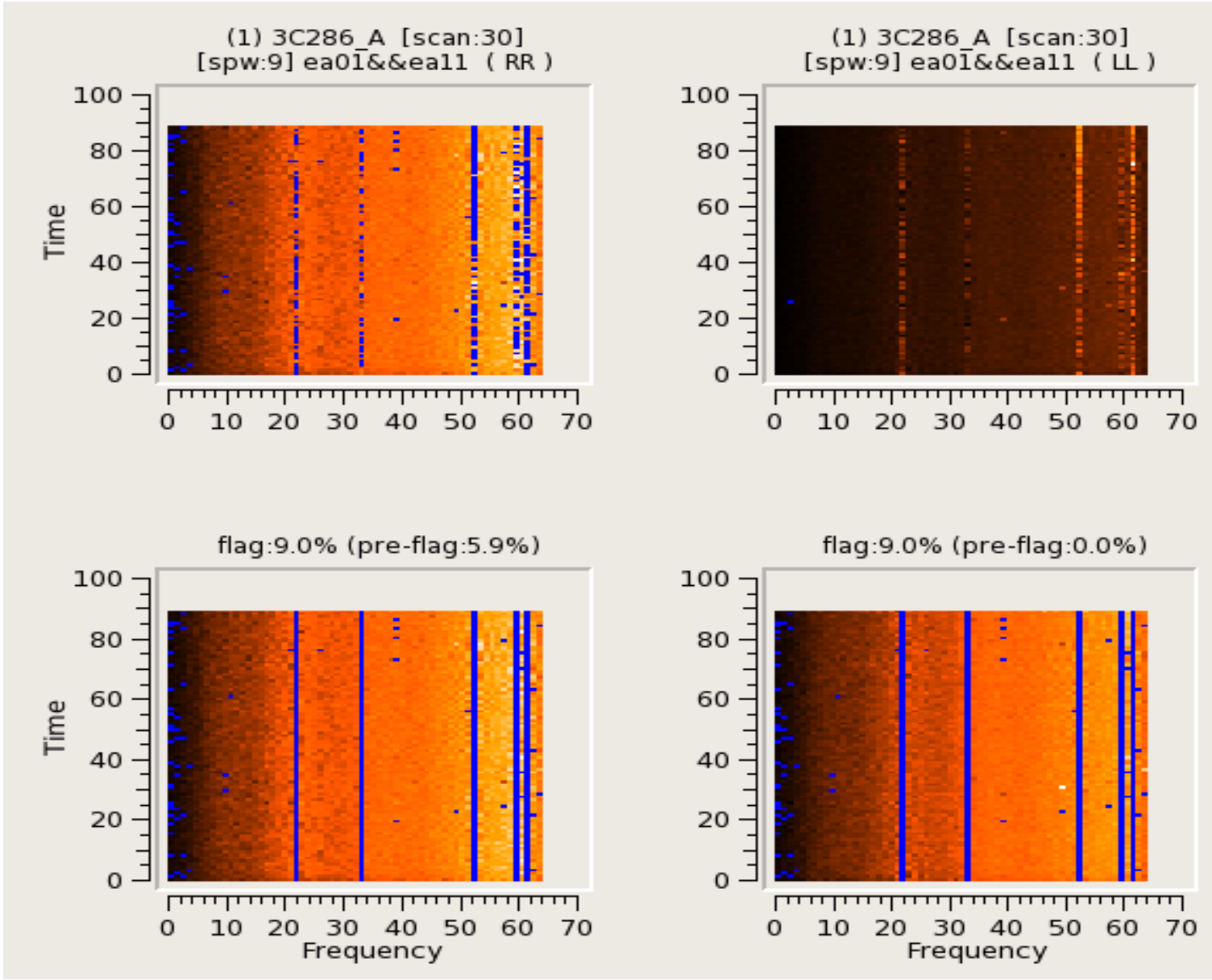
+

Growtime = 30.0

+

Extendpols = True

Useful as a follow-up option after rflag or tfcrop.



Flagging Modes : extend

Extend flags along time, frequency, polarizations (and baselines). Operates on selected data

```
ntime  = 'scan' or '2.0min' or 120.0   #   Time range to use for each chunk

combinescans = True/False              #    if ntime > scan, join multiple scans

extendpols = True/False                #    Extend across all selected correlations

growtime = 80.0                        #   For each channel, flag entire timerange (ntime)
                                             if more than 80% of the data is flagged.

growfreq = 80.0                        #   For each time, flag selected channels (in current
                                             SPW)  if more than 80% of the data is flagged.

growaround = True/False                #   On the 2D time/freq plane, if more than 4
                                             surrounding points are flagged, flag it.

flagneartime = True/False              #   Flag points before after every flagged point in time
flagnearfreq  = True/False             #   Flag channels before/after every flagged one.
```

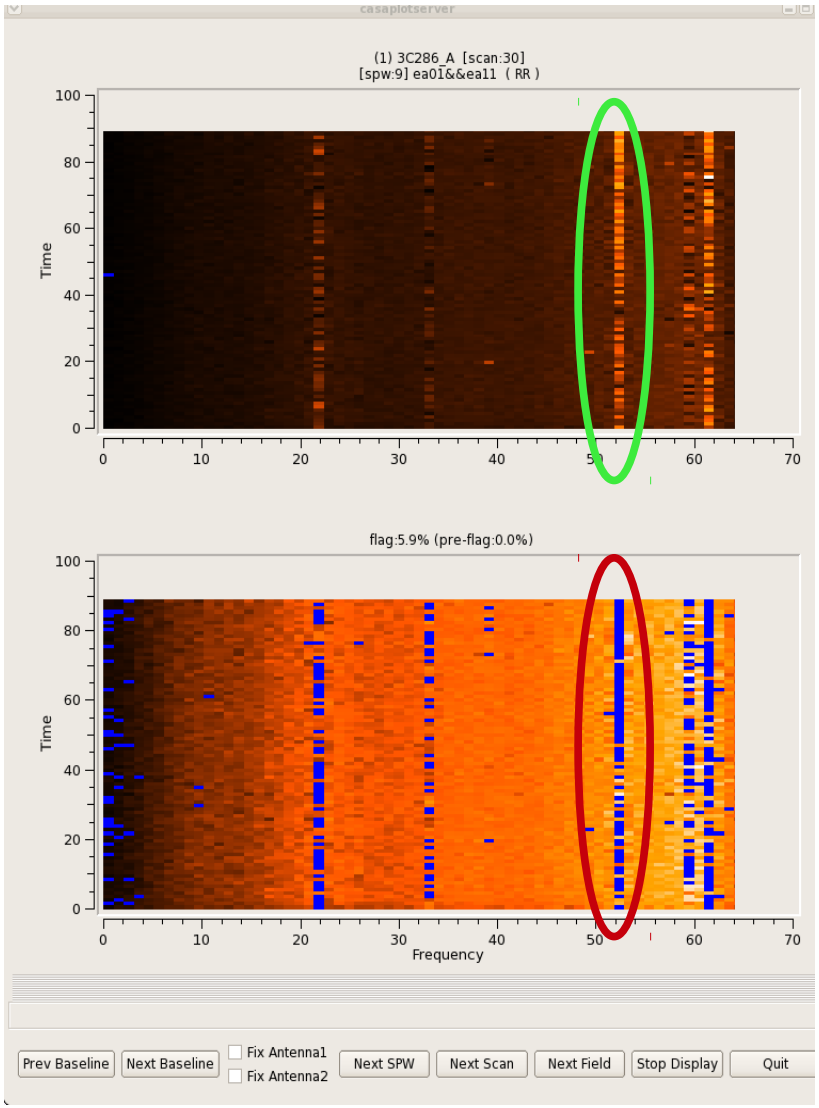
Protecting spectral-lines, during automatic flagging

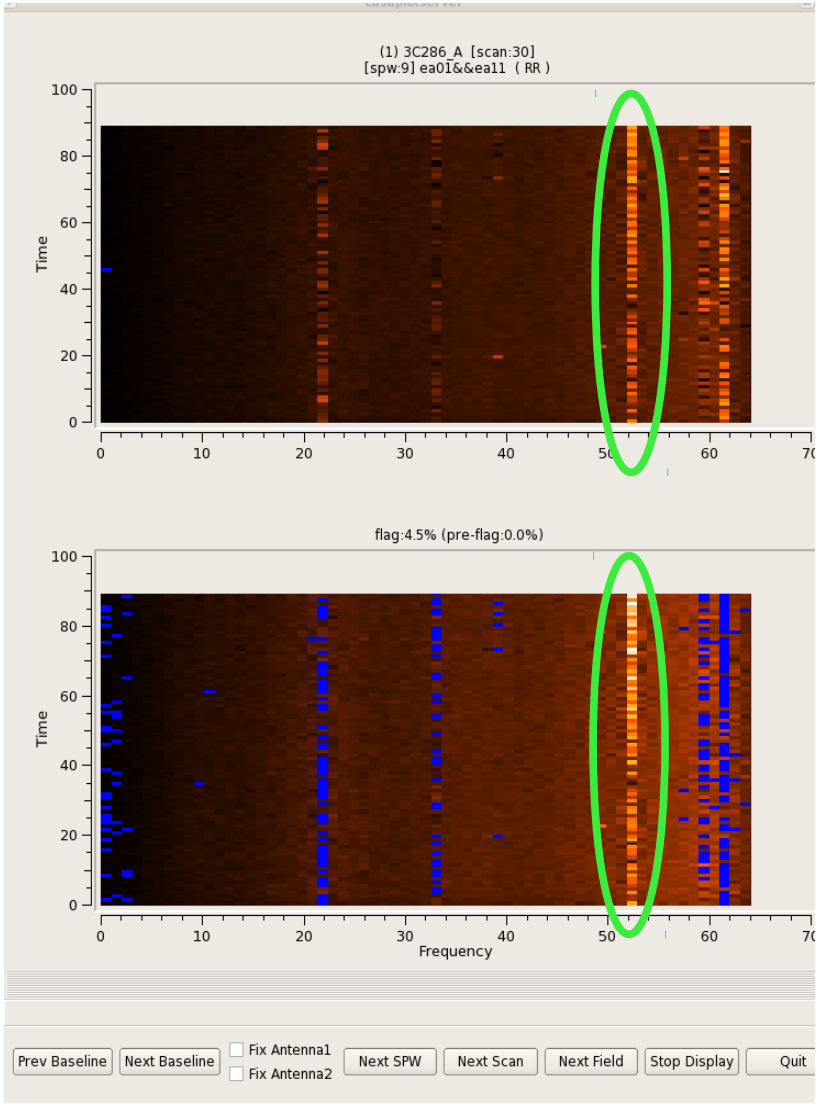
spw = ' 9 '

spw = ' 9 : 0~45; 53~63 '

before

after





This is relevant to 'clip' , 'extend' , 'tfcrop' , 'rflag'

Flagging Modes : list (flagdata, flagcmd)

Supply a list of flag commands, to be run in one single pass through the data.

- Apply large numbers of online flags (all mode='manual') together.
- Combine modes that read the same data, to save on I/O
- flagcmd can read flags from XML files and read/write to the FLAG_CMD MS subtable

Example list of commands (in a text file) :

flagdata(vis='xxx.ms', mode='list', inpfile='cmds.txt')

```
mode='manual'   antenna='1&2'    spw='3'    timerange='12:23:02.3~12.24:45.2'
mode='manual'   antenna='ea05'    spw='5:50~60'   scan='2~10'
mode='manual'   antenna='ea01,ea24'    spw='9'    correlation='RR,RL'
mode='quack'    quackinterval=5.0
mode='shadow'  tolerance=5.0
```

Example command list (interactive input) :

flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist)

cmdlist = [“mode='manual' antenna='1&2' spw='3'” , “mode='shadow' tolerance=5.0”]

Summary mode - output python dictionary of flag counts

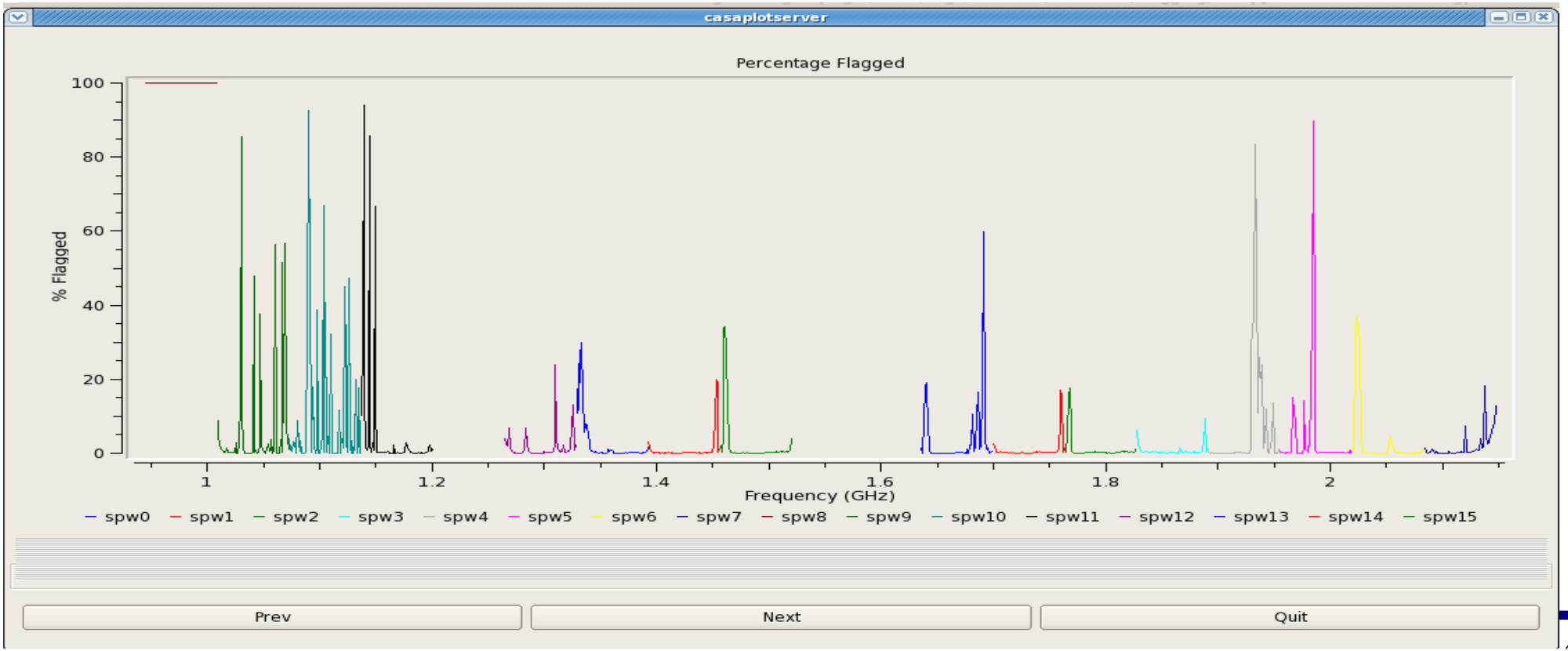
Output : Dictionary of flag counts (vs) antenna, field, spw, channel, correlation, etc...

```
counts = flagdata(vis='xxx.ms' , mode='summary' , display='report' , spwchan=True );

print “Counts per spw : “, counts['spw'];

print “Total % flagged : “ , 100.0 * counts['flagged'] / counts['total']

print “Number of unflagged visibilities in spw 9 : “, counts['spw']['9']['total'] – counts['spw']['9']['flagged']
```



Auto-flagging strategies – all methods need tuning

In general, examine pieces of your data visually, to decide auto-flagging strategy

--- one scan --- few baselines --- all spws ---

All current RFI flagging algorithms require tuning for best results.

– Rflag calculates statistics across all times, channels, baselines and pols

Tunable parameters : N-sigma thresholds, sliding-window sizes

– TFCrop operates on the time-freq plane, separately per baseline and correlation.

Tunable parameters : N-sigma thresholds, window sizes, types of time/freq fits.

– Flag Extension

Tunable parameters : Grow flags in time and frequency or both,
extend across correlations, etc....

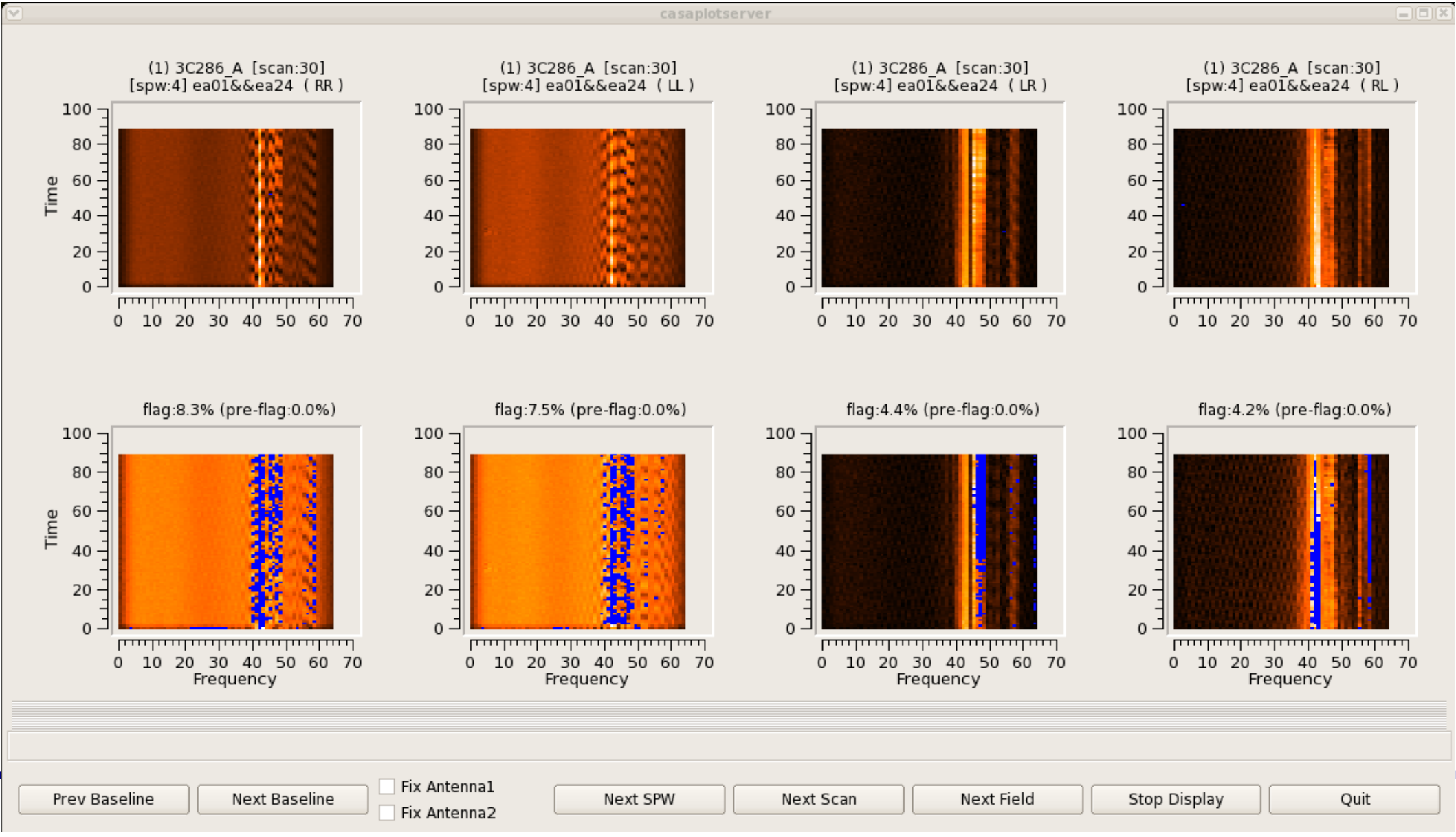
– AO-Flagger from LOFAR : http://sourceforge.net/p/aoflagger/wiki/Home/

Operates on the 2D time-freq plane, based on a robust 2D rubber-sheet fit

Examples

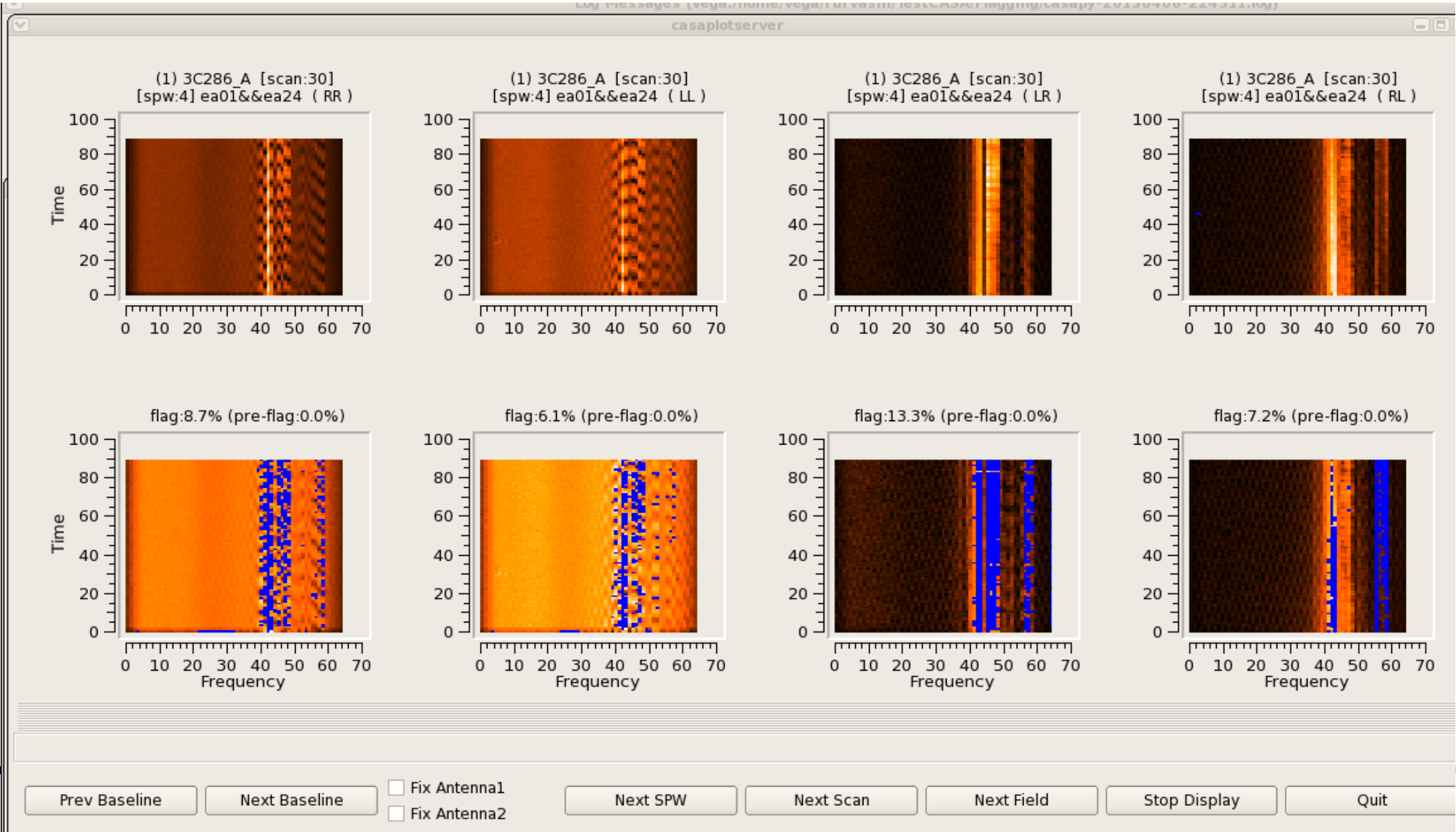
TFCrop: defaults, no H-S

flagdata(vis='xxx.ms', spw='4' mode='tfcrop', action='calculate', display='data', extendflags=F)
 (or)
cmdlist = [" spw='4' mode='tfcrop' extendflags=F "]
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop: reduced poly-fit pieces, no H-S

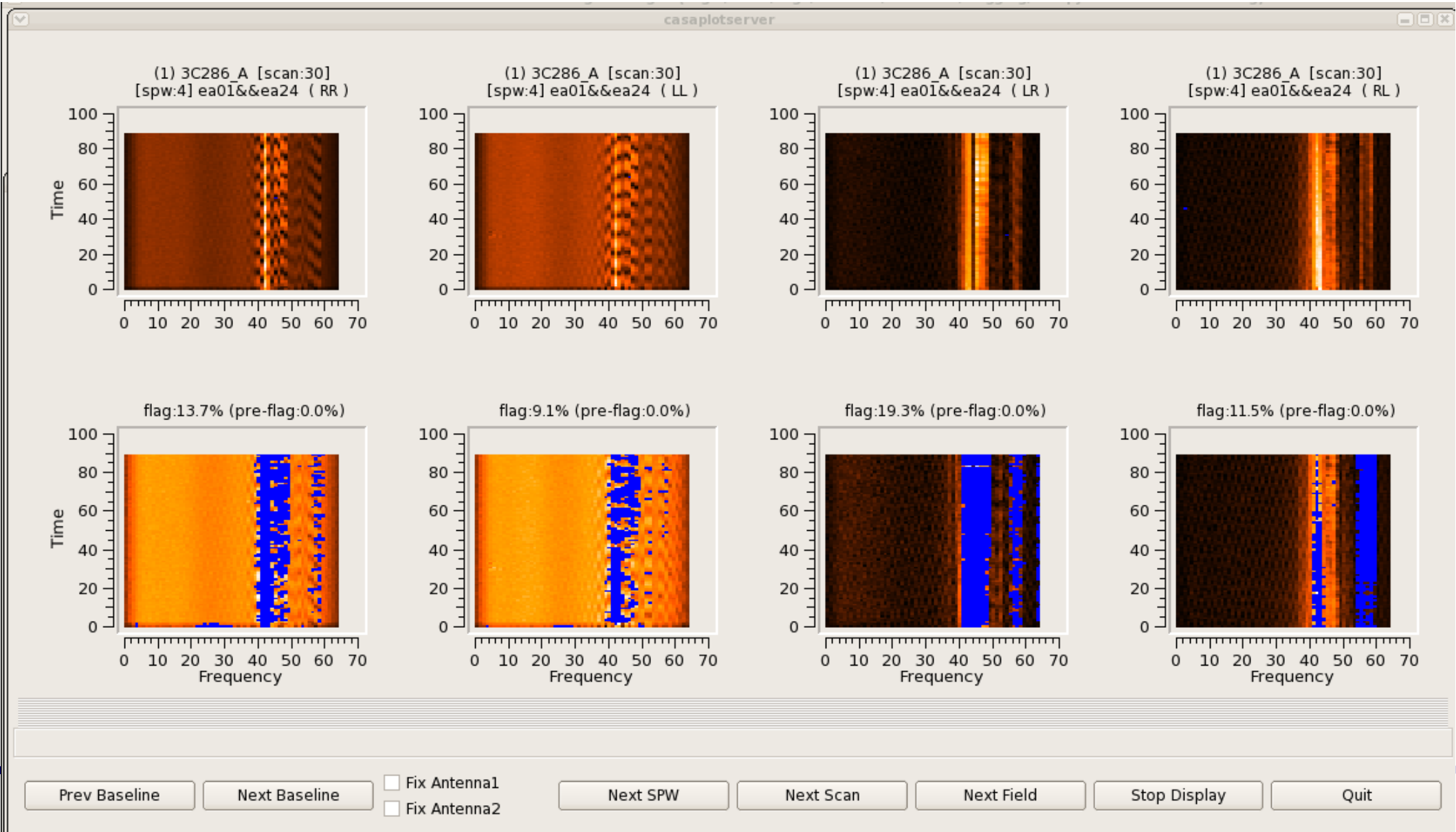
flagdata(vis='xxx.ms', spw='4' mode='tfcrop', **maxnpieces=4**, action='calculate', extendflags=F, display='data')
cmdlist = [" spw='4' mode='tfcrop' maxnpieces=4 extendflags=F"]
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop: reduced poly-fit pieces + sliding-win stat, no H-S

cmdlist = [“ spw='4' mode='tfcrop' **maxnpieces=4** **usewindowstats='std' halfwin=2**, extendflags=F “]

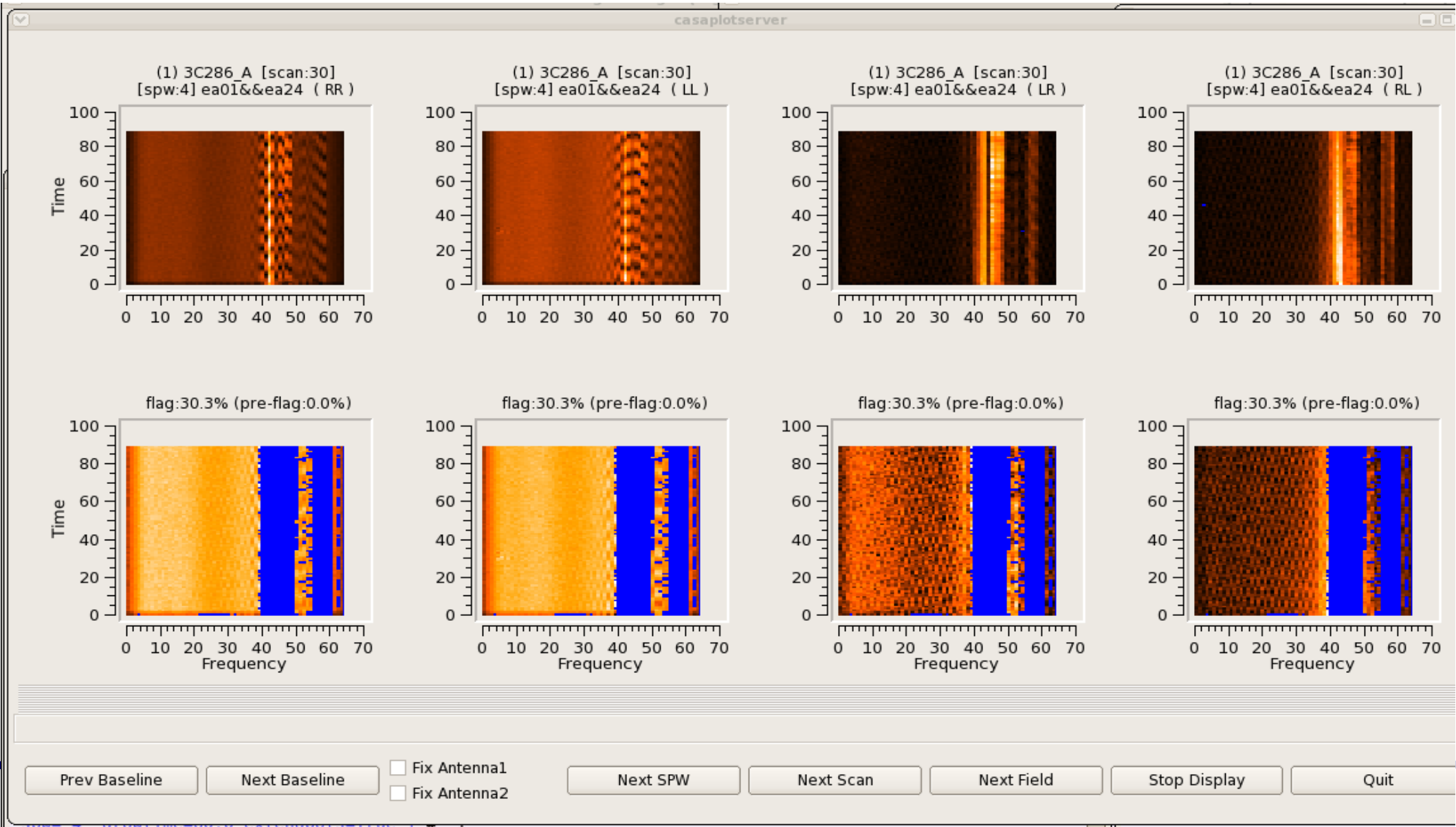
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop: reduced poly-fit pieces + sliding-win stat + extend, no H-S

cmdlist = [“ spw='4' mode='tfcrop' **maxnpieces=4 usewindowstats='std' halfwin=2** extendflags=F “, **“ spw='4' mode='extend' growtime=60.0 extendpols=T “**]

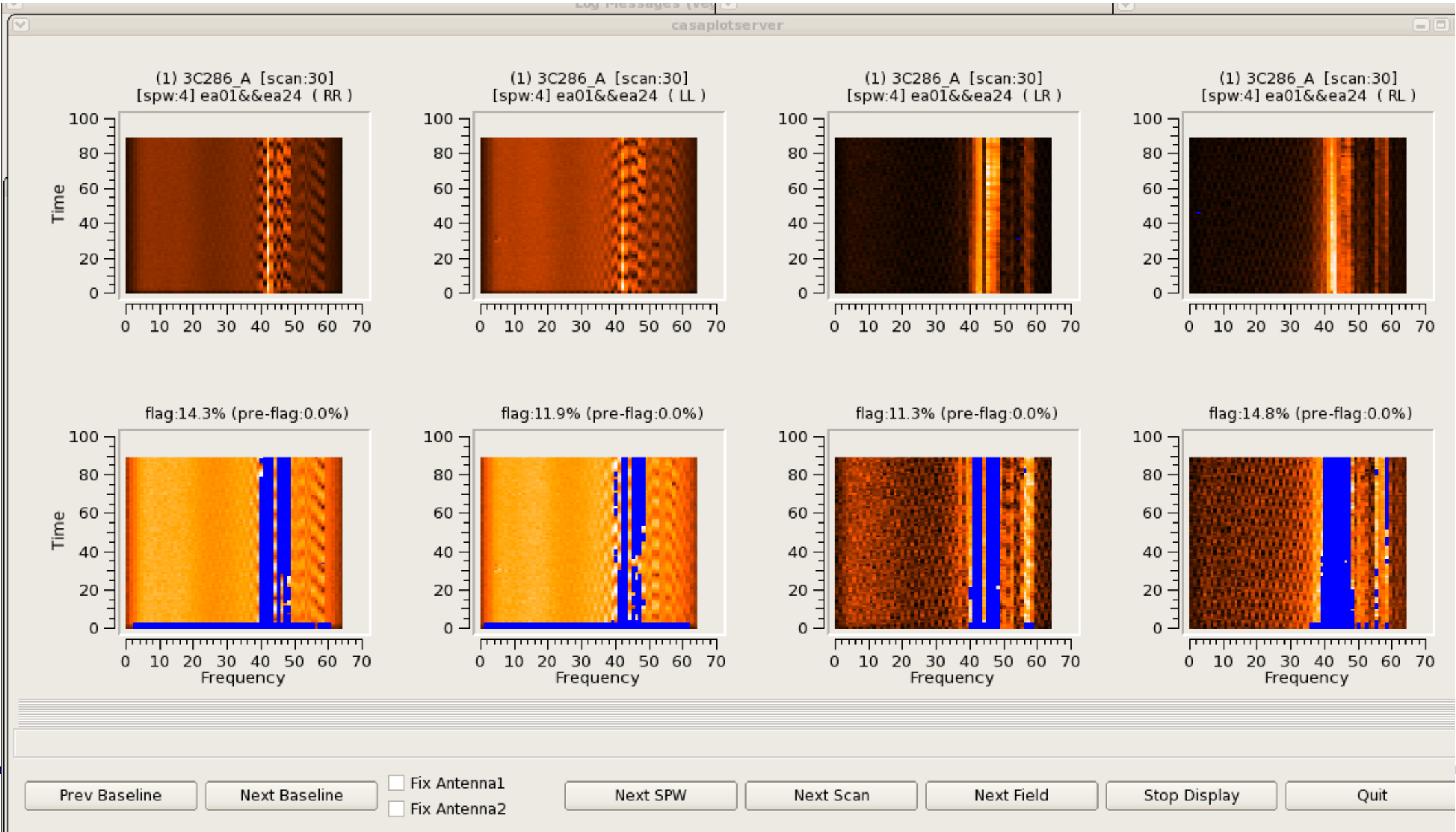
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: defaults, no H-S

cmdlist = [“ spw='4' mode='rflag' extendflags=F“]

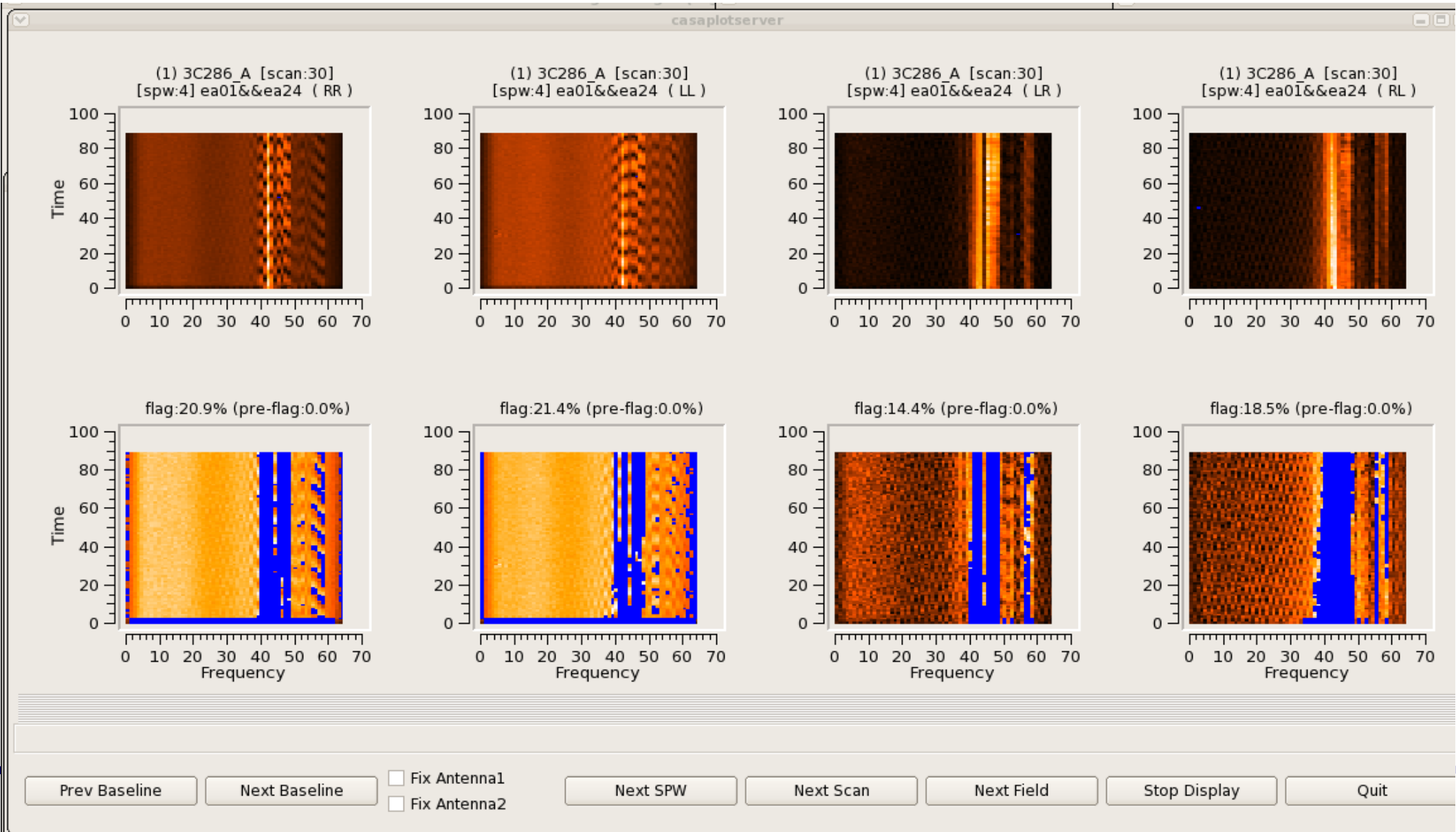
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: lower threshold, no H-S

cmdlist = [“ spw='4' mode='rflag' **freqdevscale=4.0** extendflags=F“]

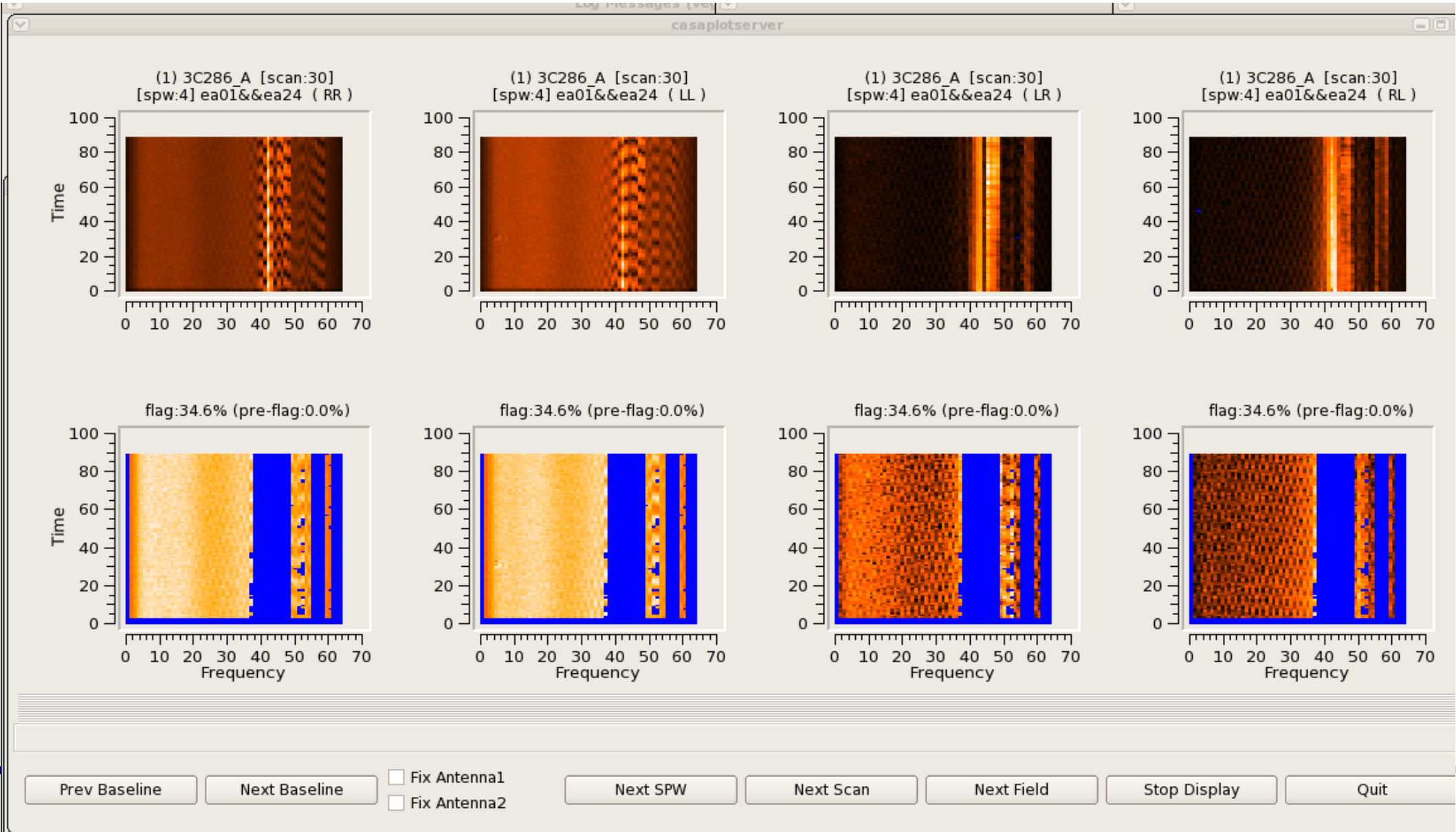
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: lower threshold + extend, no H-S

cmdlist = [“ spw='4' mode='rflag' **freqdevscale=4.0** extendflags=F“,
“ **spw='4' mode='extend' growtime=30.0 extendpols=T**]

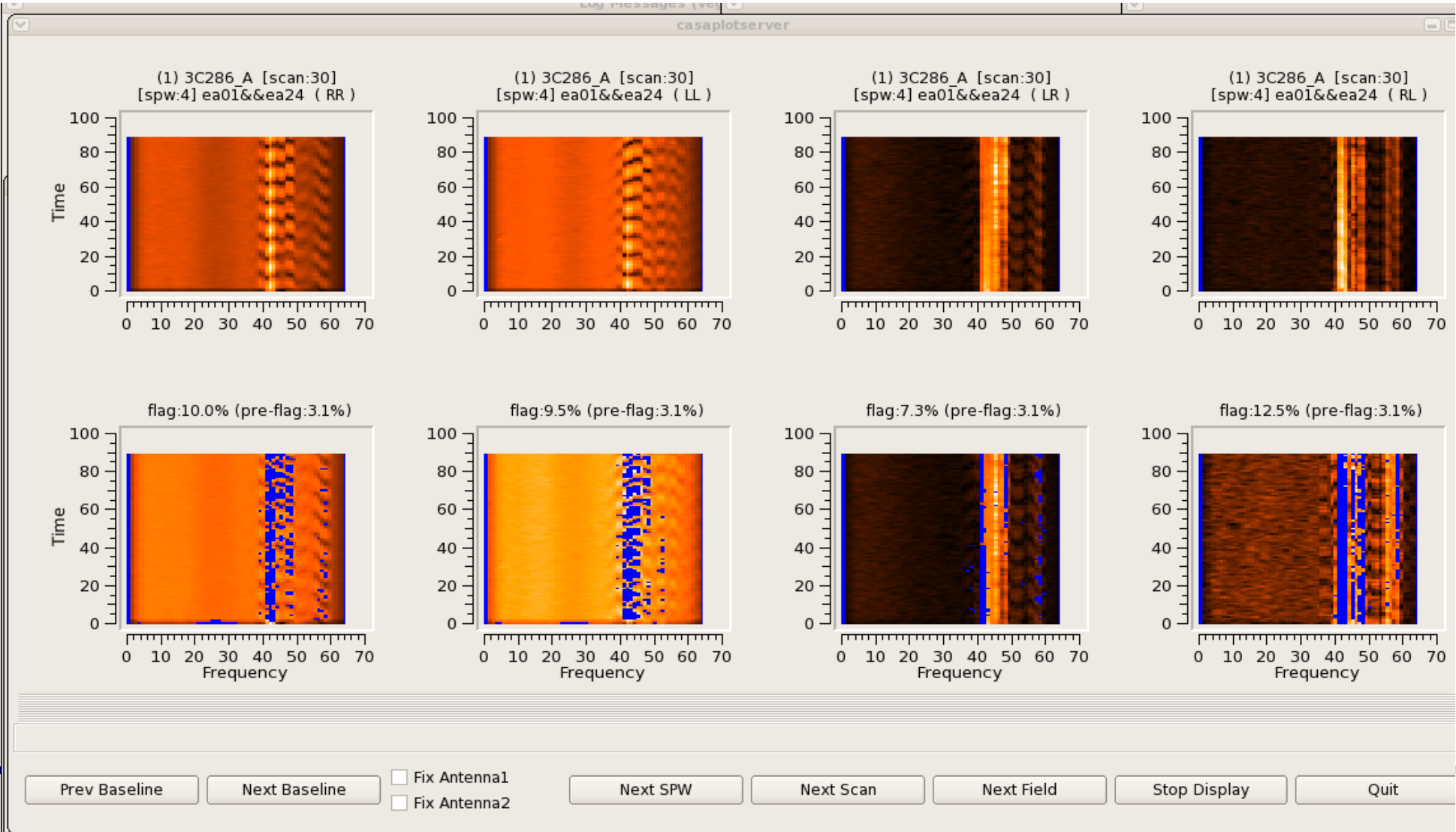
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFcrop: defaults, H-S

cmdlist = [“ spw='4' mode='tfcrop' extendflags=F“]

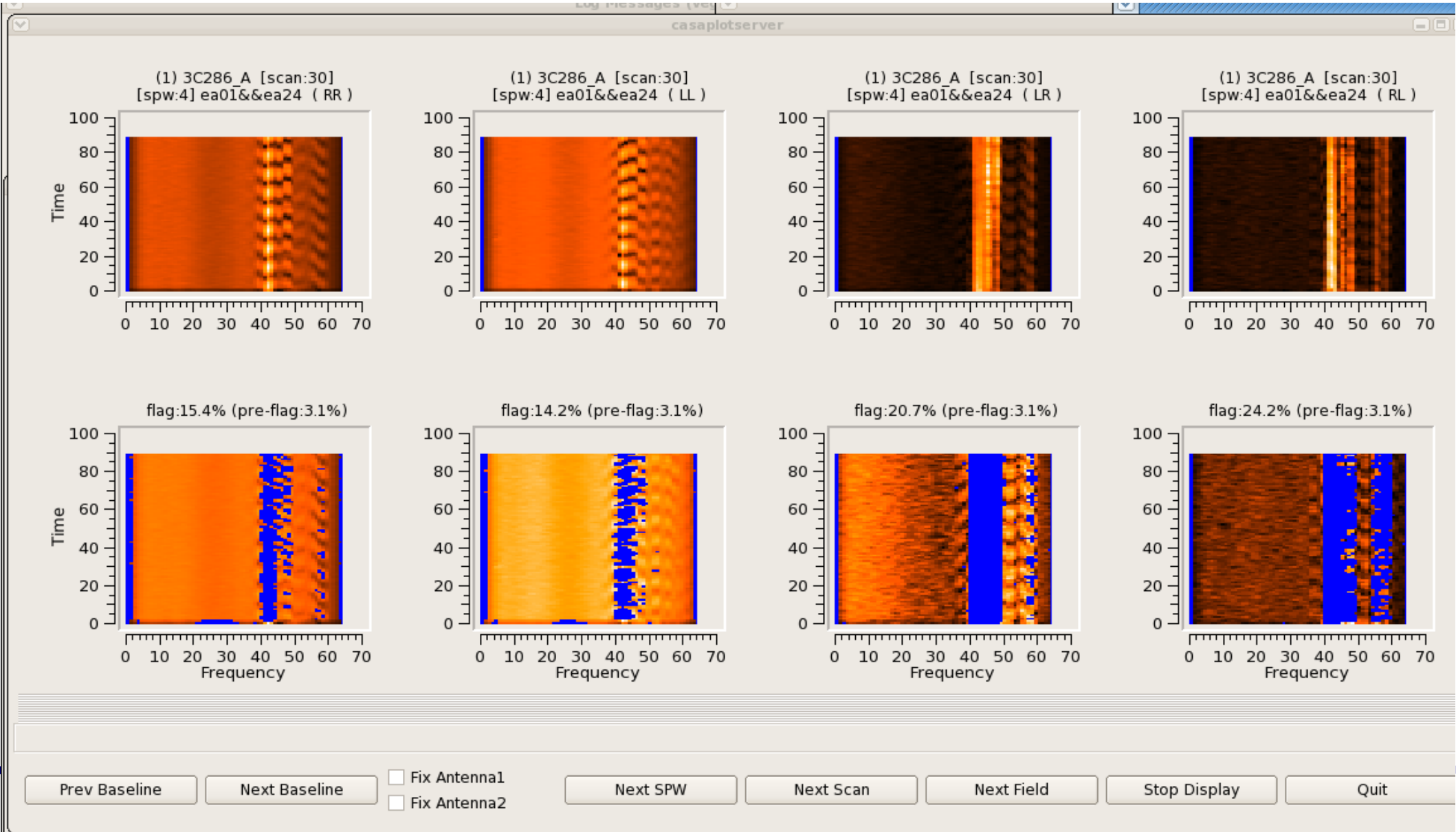
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop: sliding-win stat, H-S

cmdlist = [“ spw='4' mode='tfcrop' **usewindowstats='sum'** extendflags=F “]

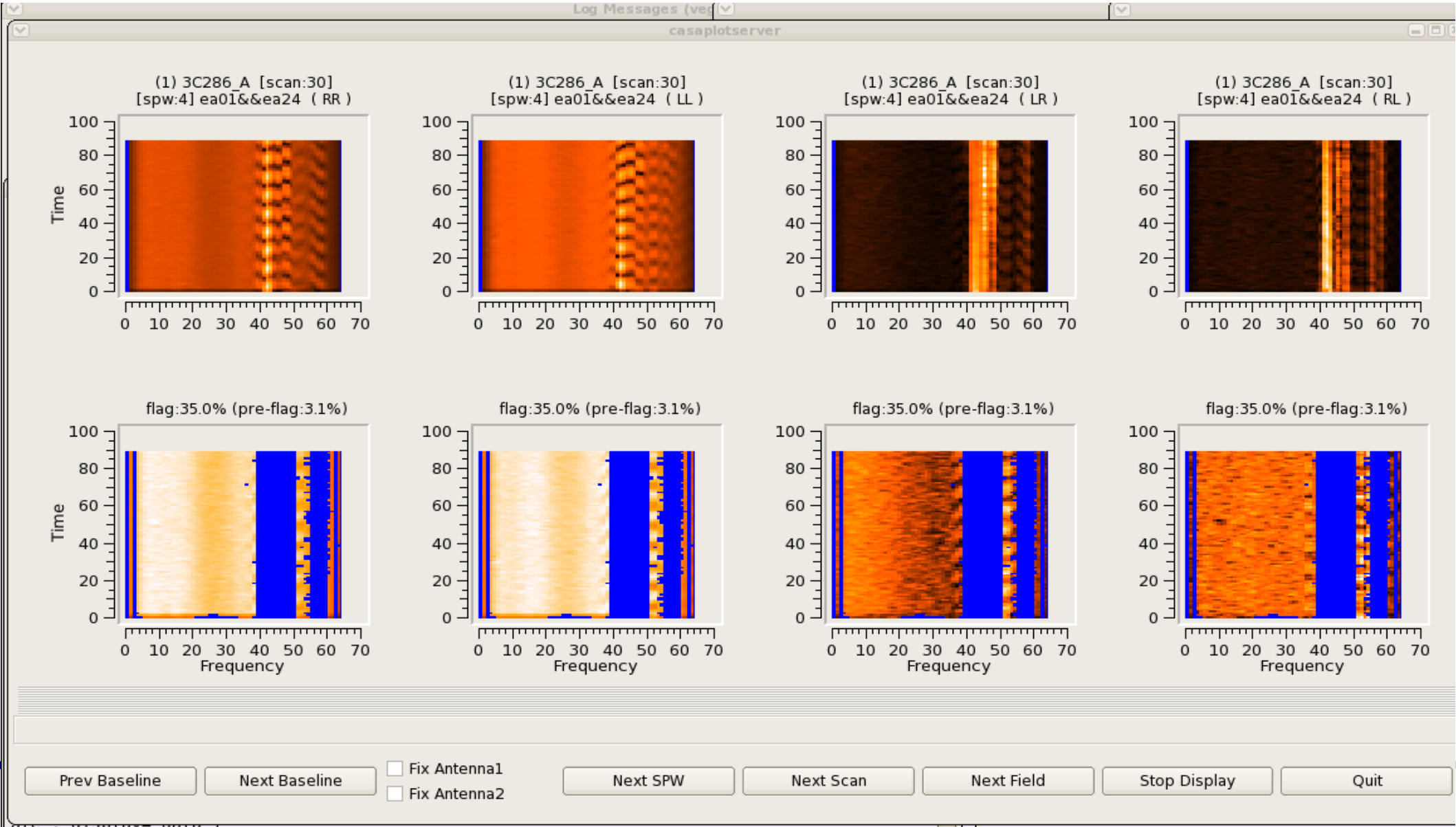
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop: sliding-win stat + extend, H-S

cmdlist = [“ spw='4' mode='tfcrop' **usewindowstats='sum'** extendflags=F “,
“ **spw='4' mode='extend' growtime=30.0 extendpols=T** “]

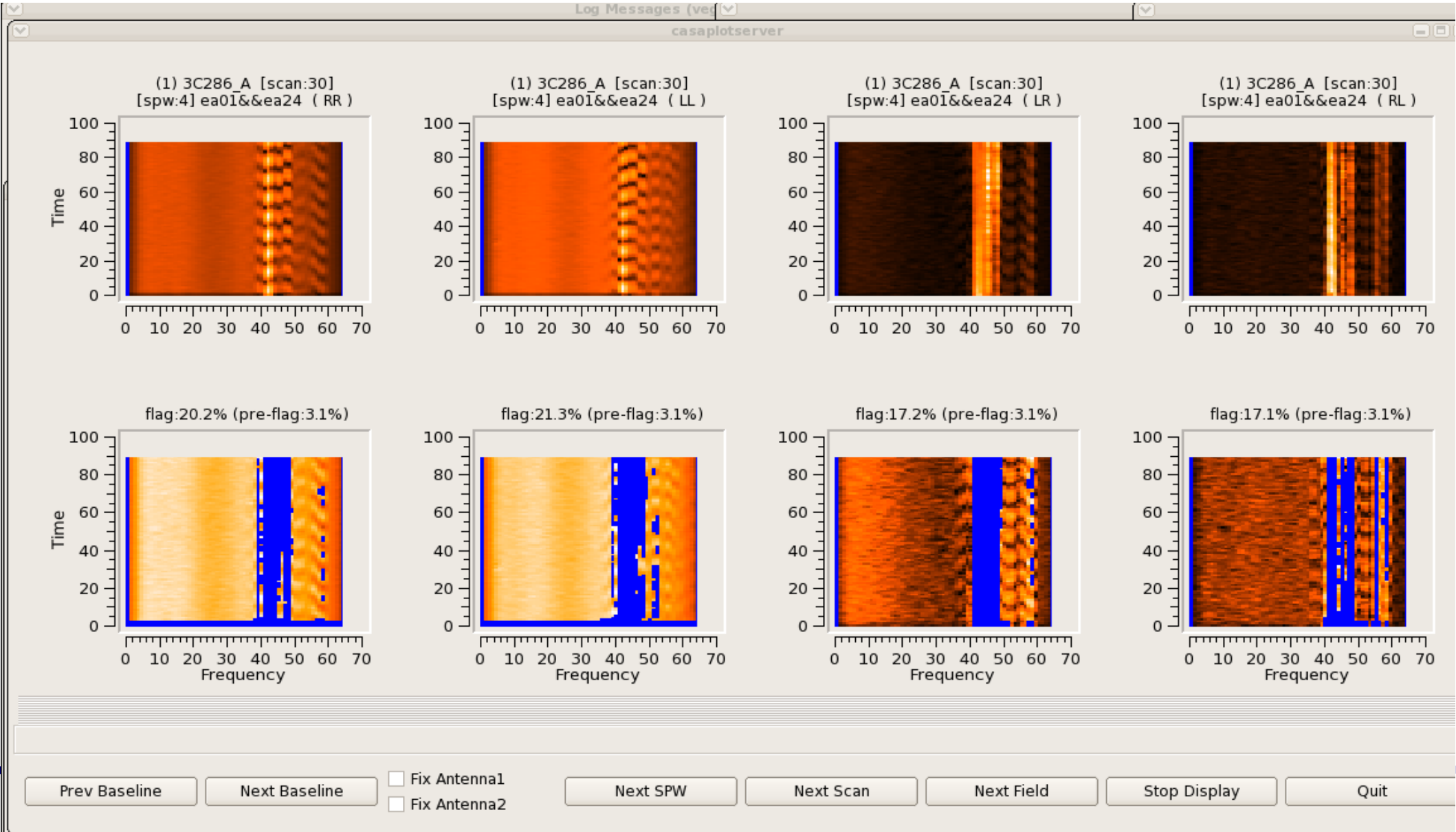
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: defaults, H-S

cmdlist = [“ spw='4' mode='rflag' extendflags=F“]

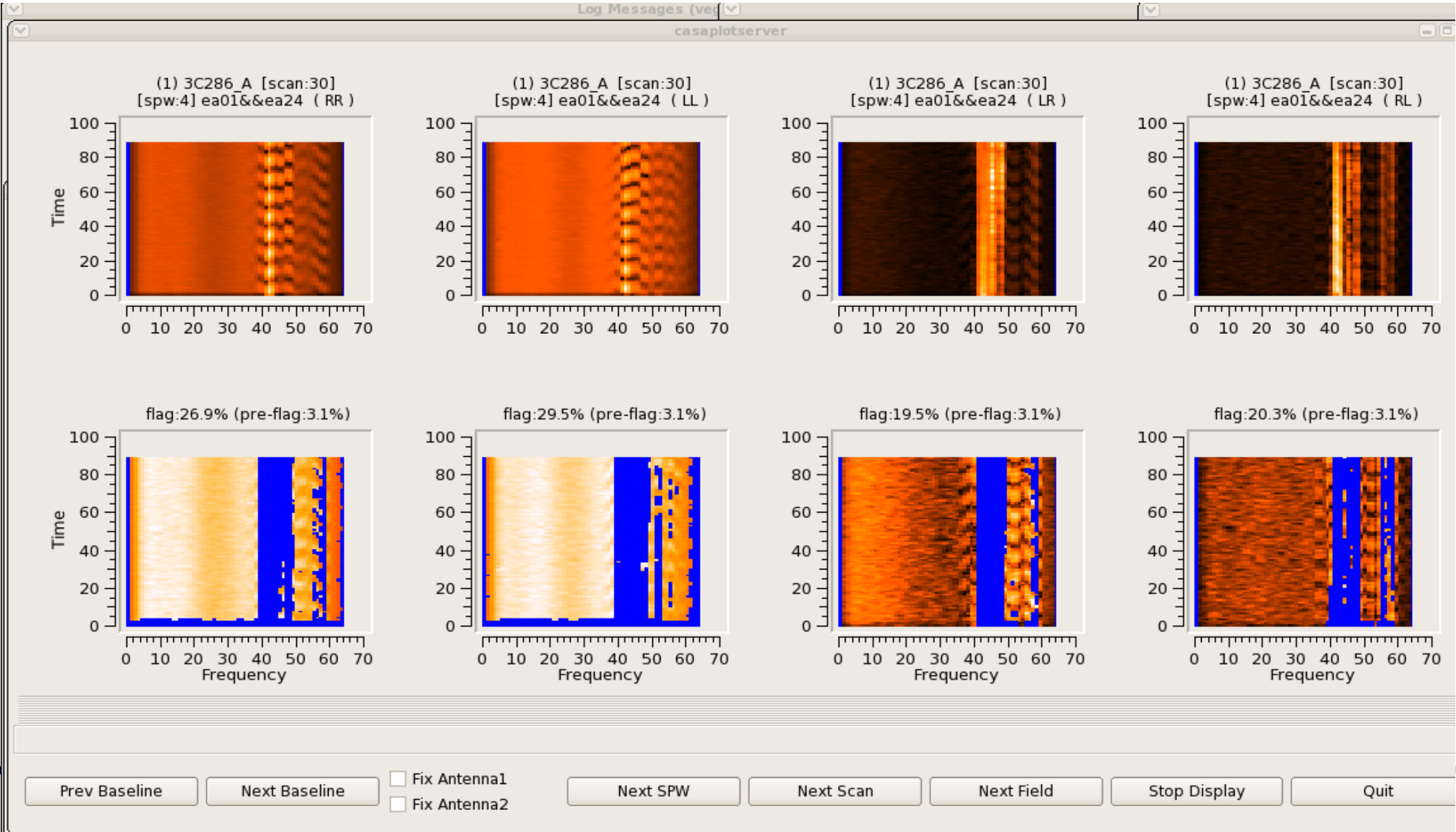
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: lower threshold, H-S

cmdlist = [“ spw='4' mode='rflag' **freqdevscale=3.0** extendflags=F“]

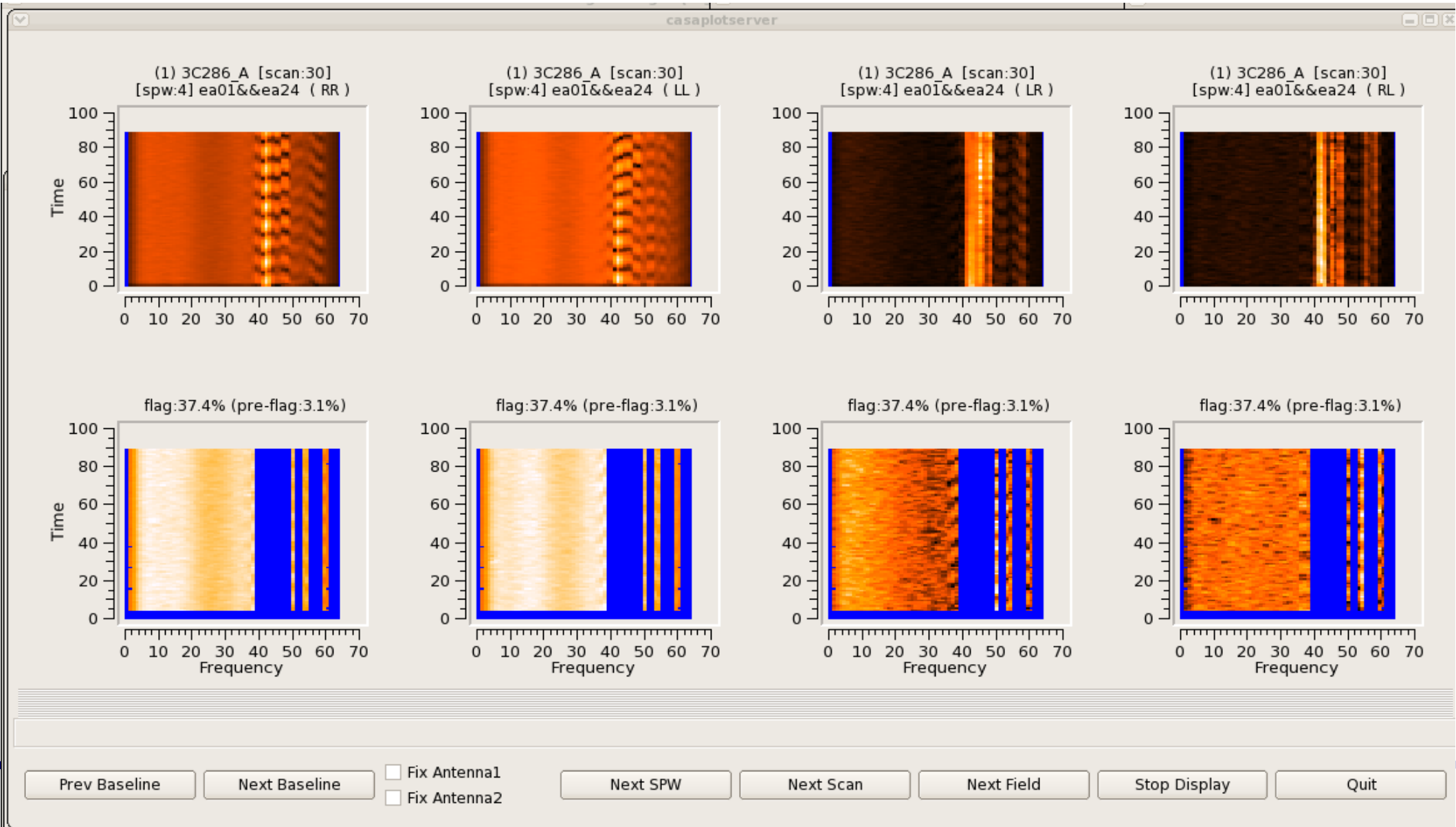
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: lower threshold + extend, H-S

cmdlist = [" spw='4' mode='rflag' **freqdevscale=3.0** extendflags=F ",
" **spw='4' mode='extend' growtime=30.0 extendpols=T** "]

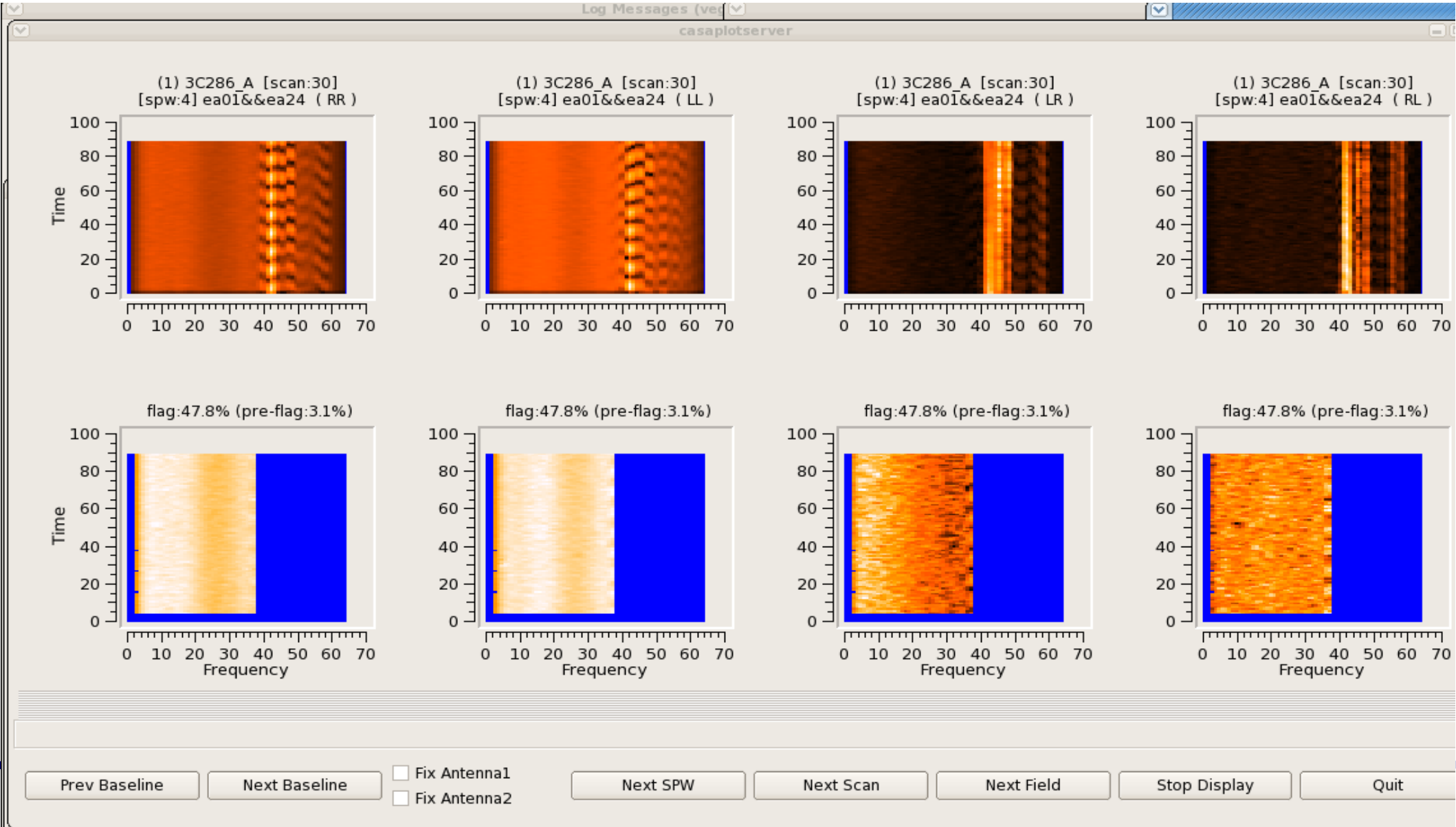
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag: lower threshold + extend + extend, H-S

cmdlist = [“ spw='4' mode='rflag' **freqdevscale=3.0** extendflags=F “,
“ **spw='4' mode='extend' growtime=30.0 extendpols=T flagnearfreq=T** “]

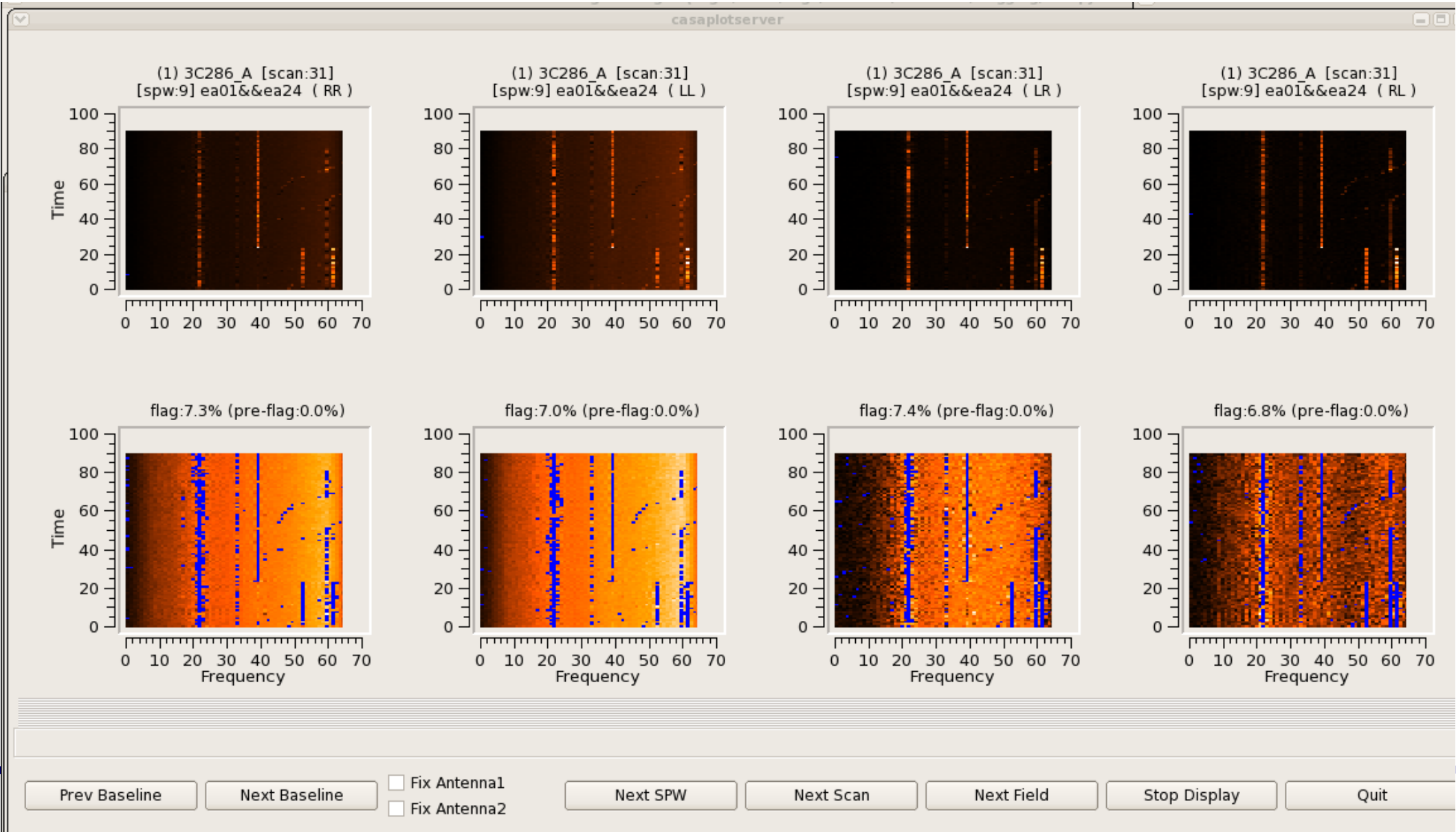
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (narrow spiky RFI): defaults, no H-S

cmdlist = [“ spw='9' mode='tfcrop' extendflags=F“]

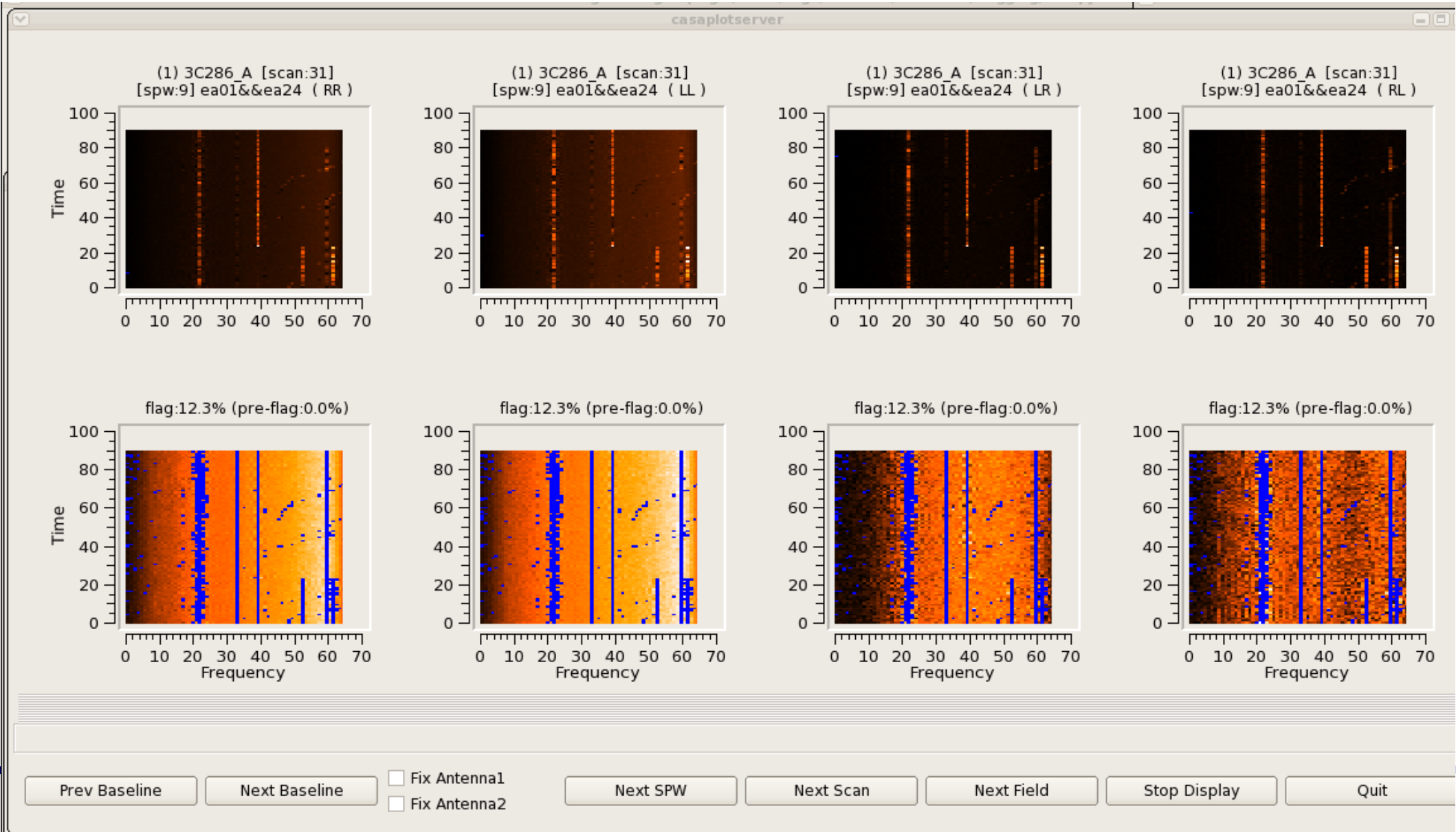
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (narrow spiky RFI): extend, no H-S

cmdlist = [“ spw='9' mode='tfcrop' extendflags=F “,
“ **spw='9' mode='extend' growtime=50.0 extendpols=T**]

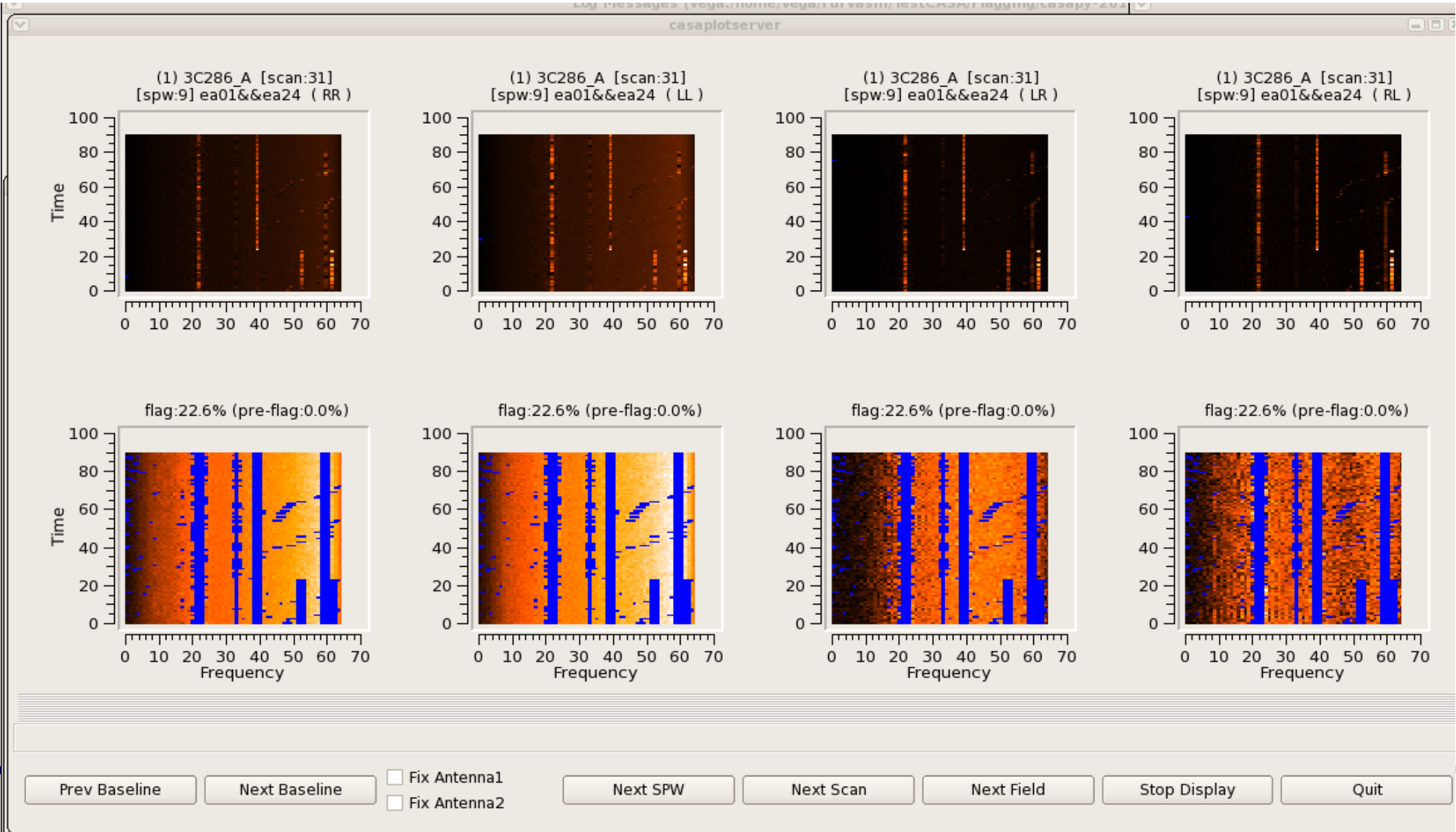
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (narrow spiky RFI): sliding-win stat + extend, no H-S

cmdlist = [“ spw='9' mode='tfcrop' **usewindowstats='sum'** extendflags=F “ ,
“ **spw='9' mode='extend' growtime=50.0 extendpols=T** “]

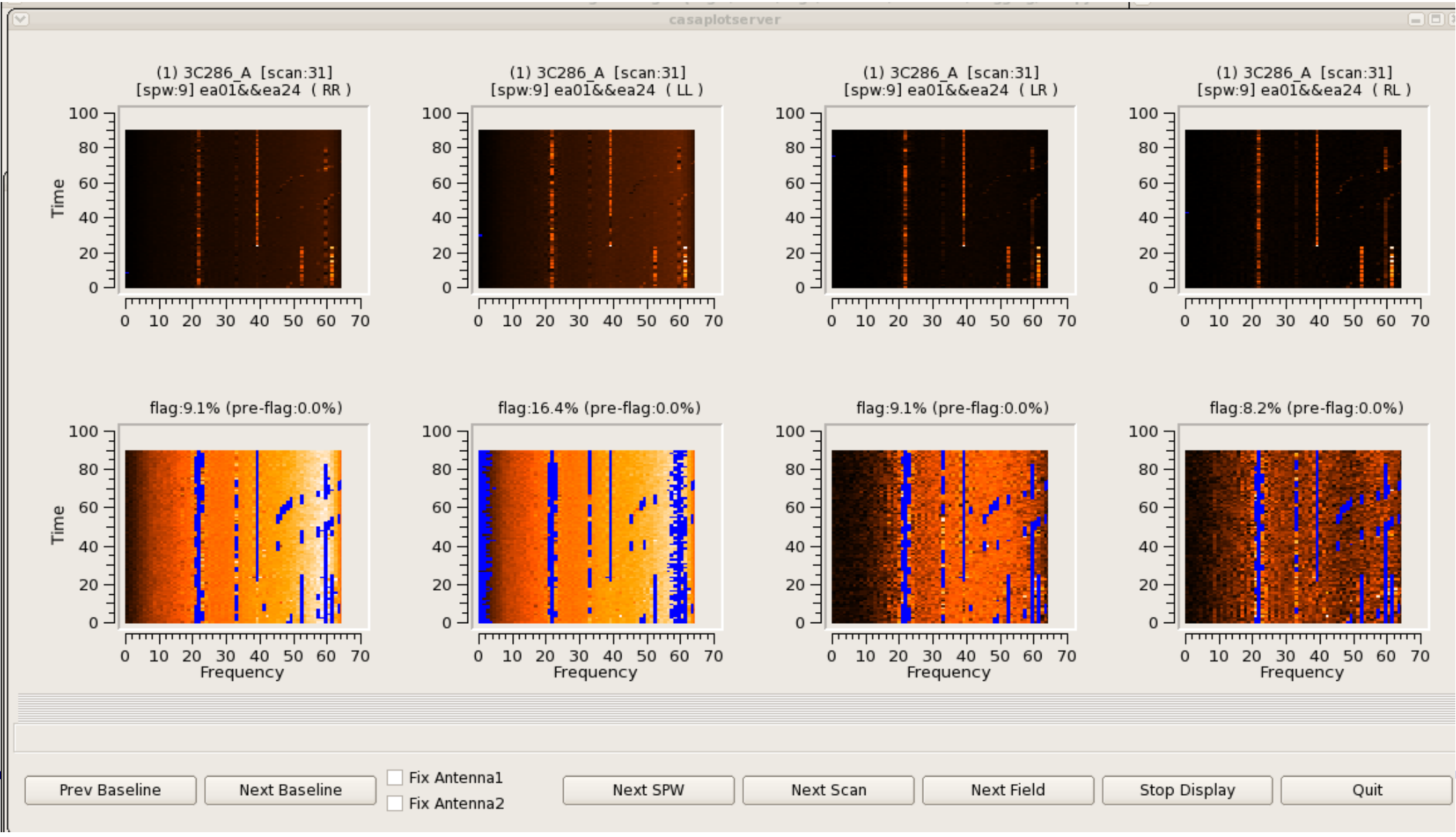
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (narrow spiky RFI): defaults, no H-S

cmdlist = [“ spw='9' mode='rflag' extendflags=F “]

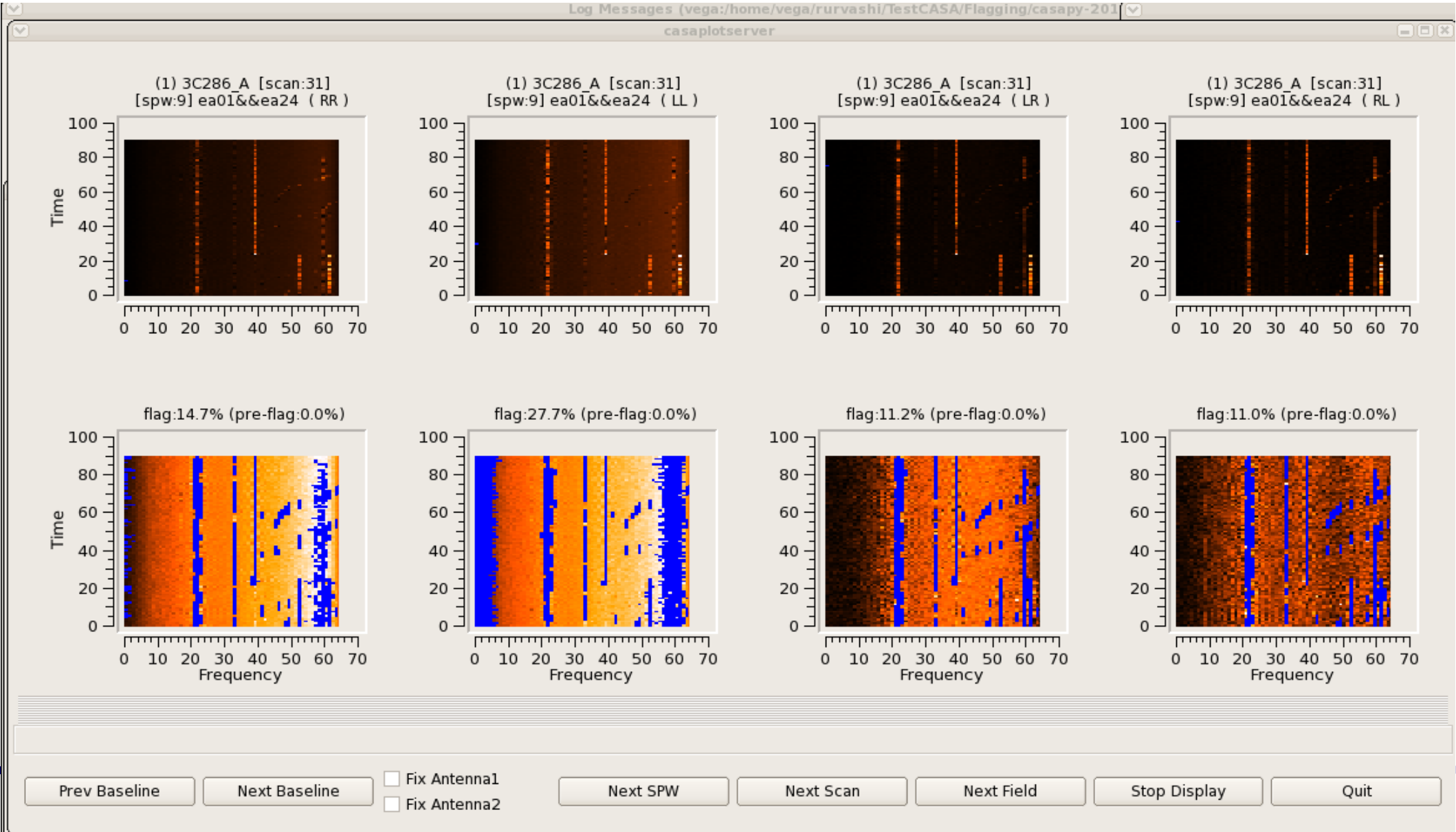
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (narrow spiky RFI): lower threshold, no H-S

cmdlist = [“ spw='9' mode='rflag' **freqdevscale=4.0** extendflags=F “]

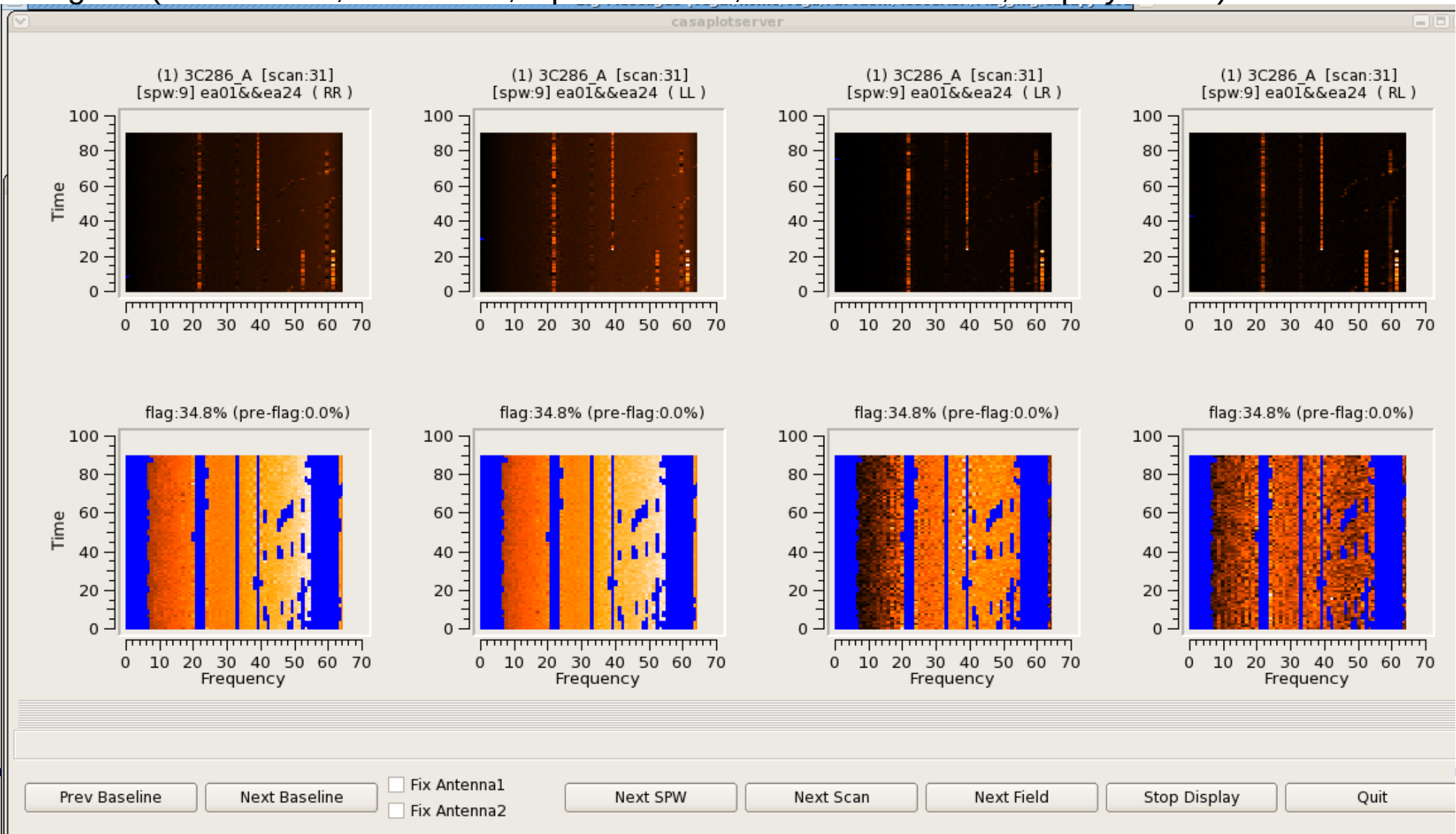
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (narrow spiky RFI): lower threshold + extend, no H-S

cmdlist = [“ spw='9' mode='rflag' **freqdevscale=4.0** extendflags=F “ ,
“ **spw='9' mode='extend' growtime=50.0 flagneartime=T extendpols=T**]

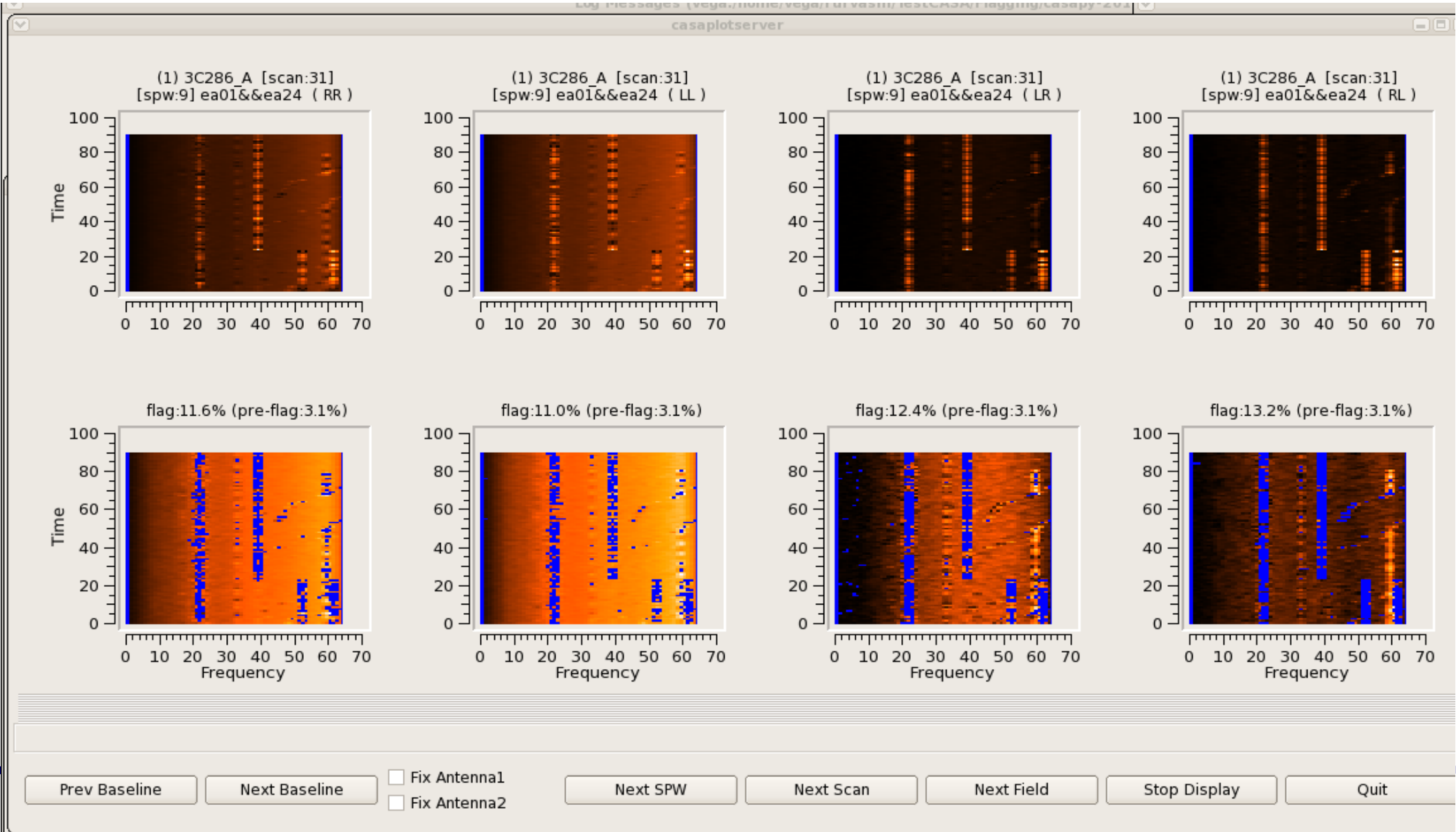
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (narrow spiky RFI): defaults, H-S

cmdlist = [“ spw='9' mode='tfcrop' extendflags=F“]

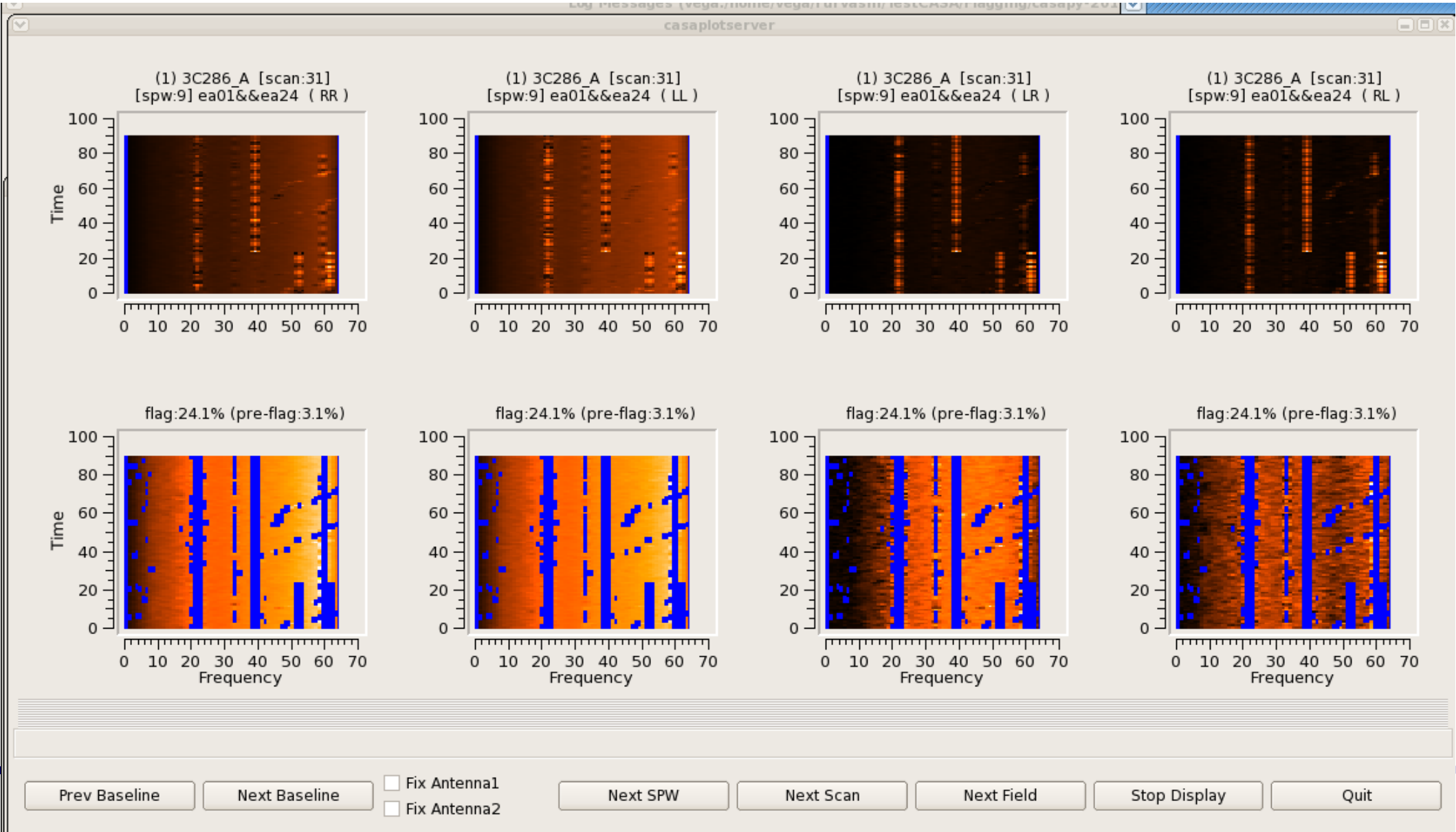
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (narrow spiky RFI): extend, H-S

cmdlist = [“ spw='9' mode='tfcrop' extendflags=F “ ,
“ **spw='9' mode='extend' growtime=50.0 extendpols=T flagneartime=T**“]

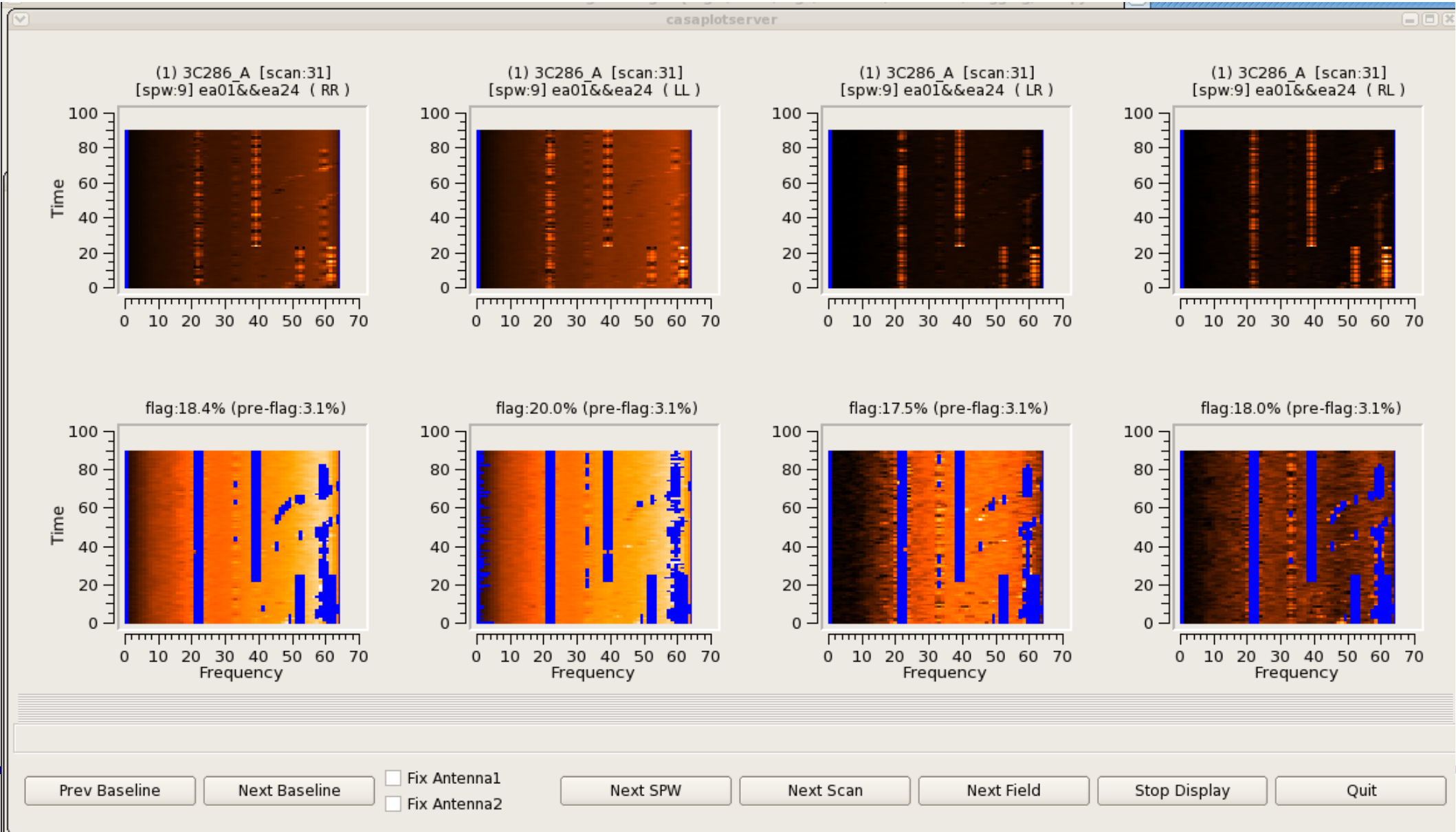
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (narrow spiky RFI): defaults, H-S

cmdlist = [“ spw='9' mode='rflag' extendflags=F “]

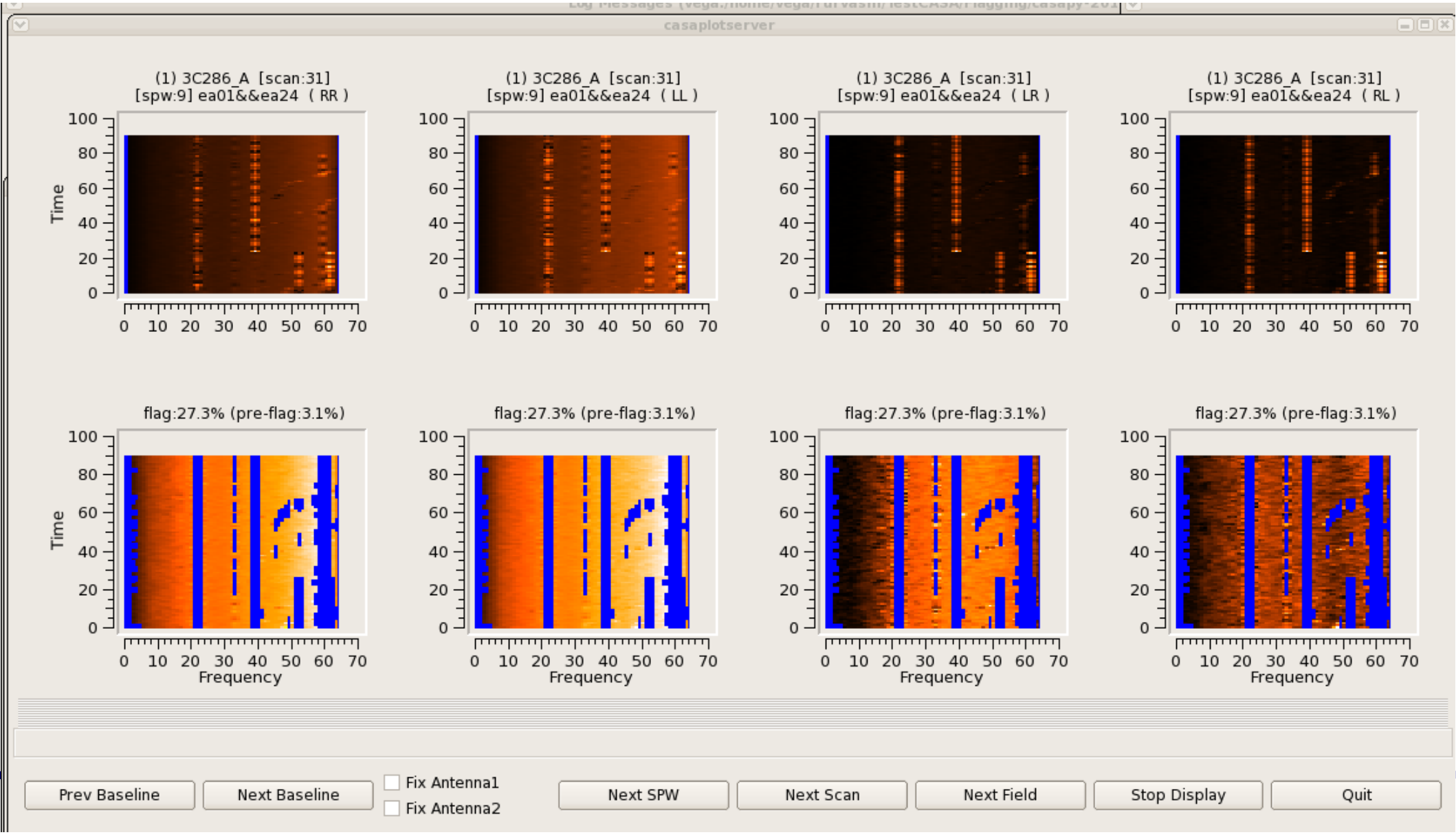
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (narrow spiky RFI): extend, H-S

cmdlist = [“ spw='9' mode='rflag' extendflags=F “ ,
“ **spw='9' mode='extend' growtime=50.0 extendpols=T flagneartime=T**“]

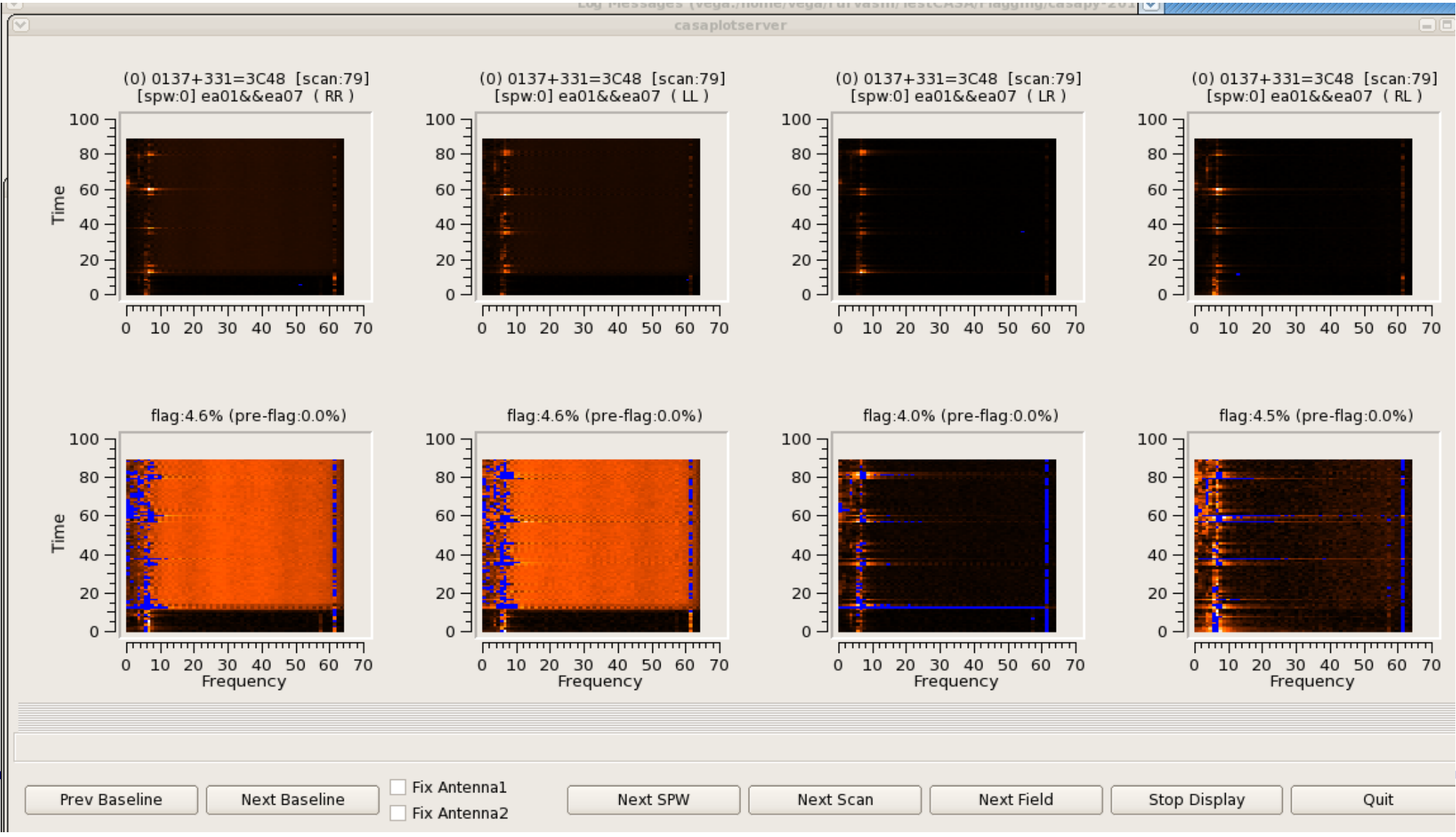
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (Broadband Intermittent RFI): defaults, no H-S

cmdlist = [“ spw='5' mode='tfcrop' extendflags=F“]

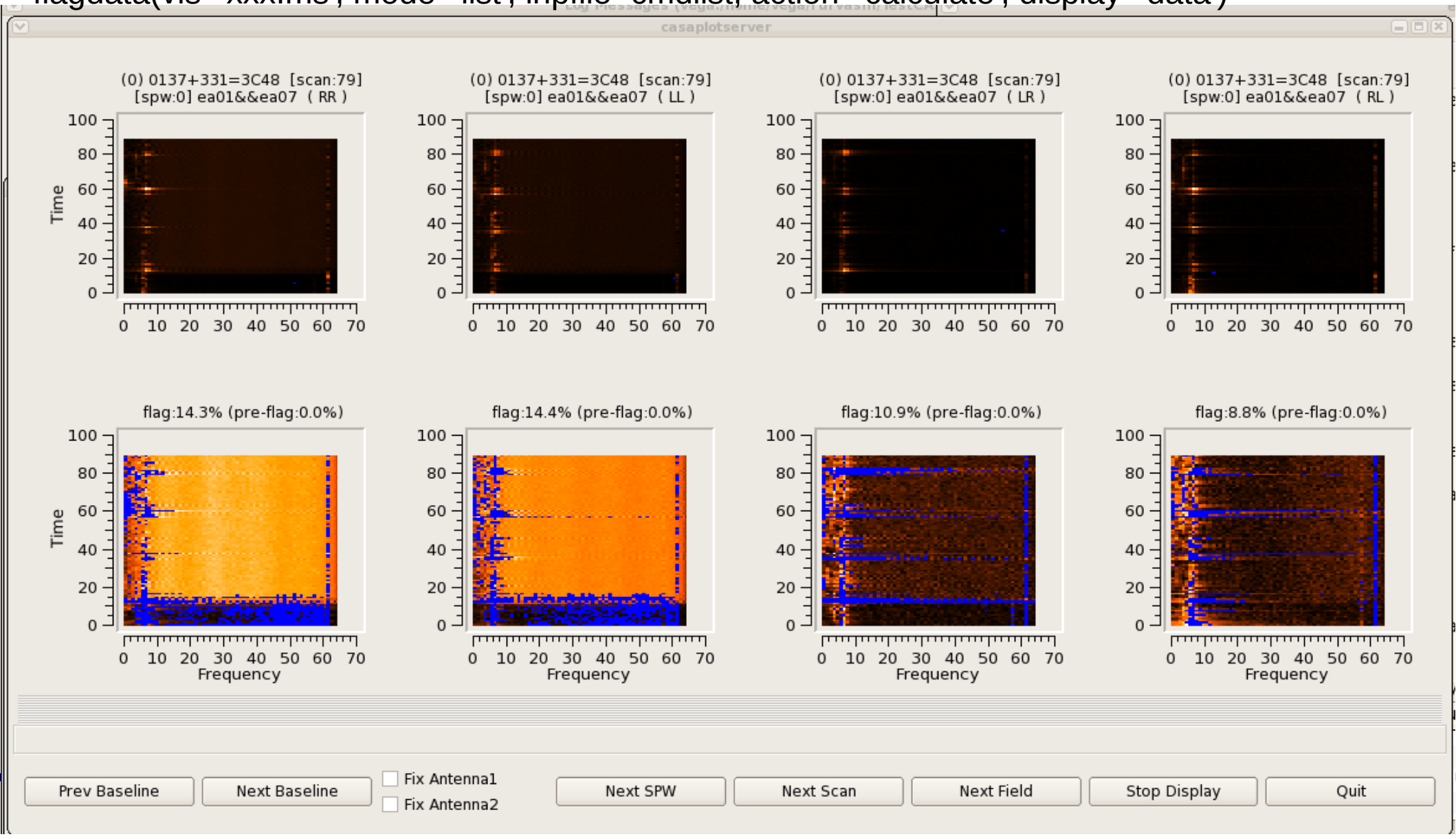
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (B.I. RFI): reduced poly-fit pieces + lower thresholds + poly-fit in time, no H-S

cmdlist = [“ spw='5' mode='tfcrop' **maxnpieces=4 timecutoff=2.5 freqcutoff=3.0 timefit='poly'** extendflags=F “]

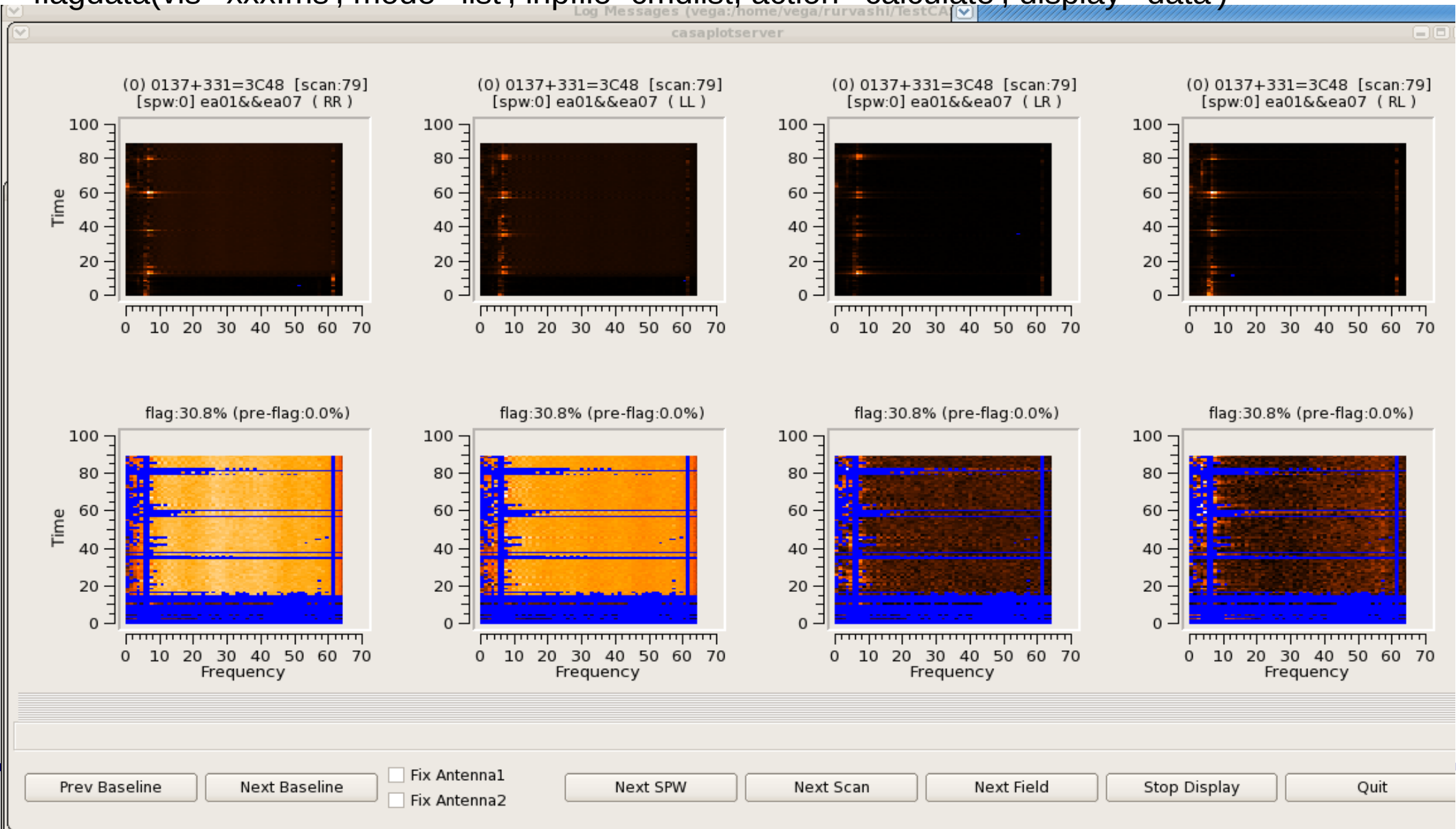
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (B. I. RFI): reduced poly-fit pieces + lower thresholds + poly-fit in time + extend, no H-S

cmdlist = [“ spw='5' mode='tfcrop' **maxnpieces=4 timecutoff=2.5 freqcutoff=3.0 timefit='poly'** extendflags=F“, “ **spw='5' mode='extend' growtime=50.0 extendpols=T growfreq=50.0** “]

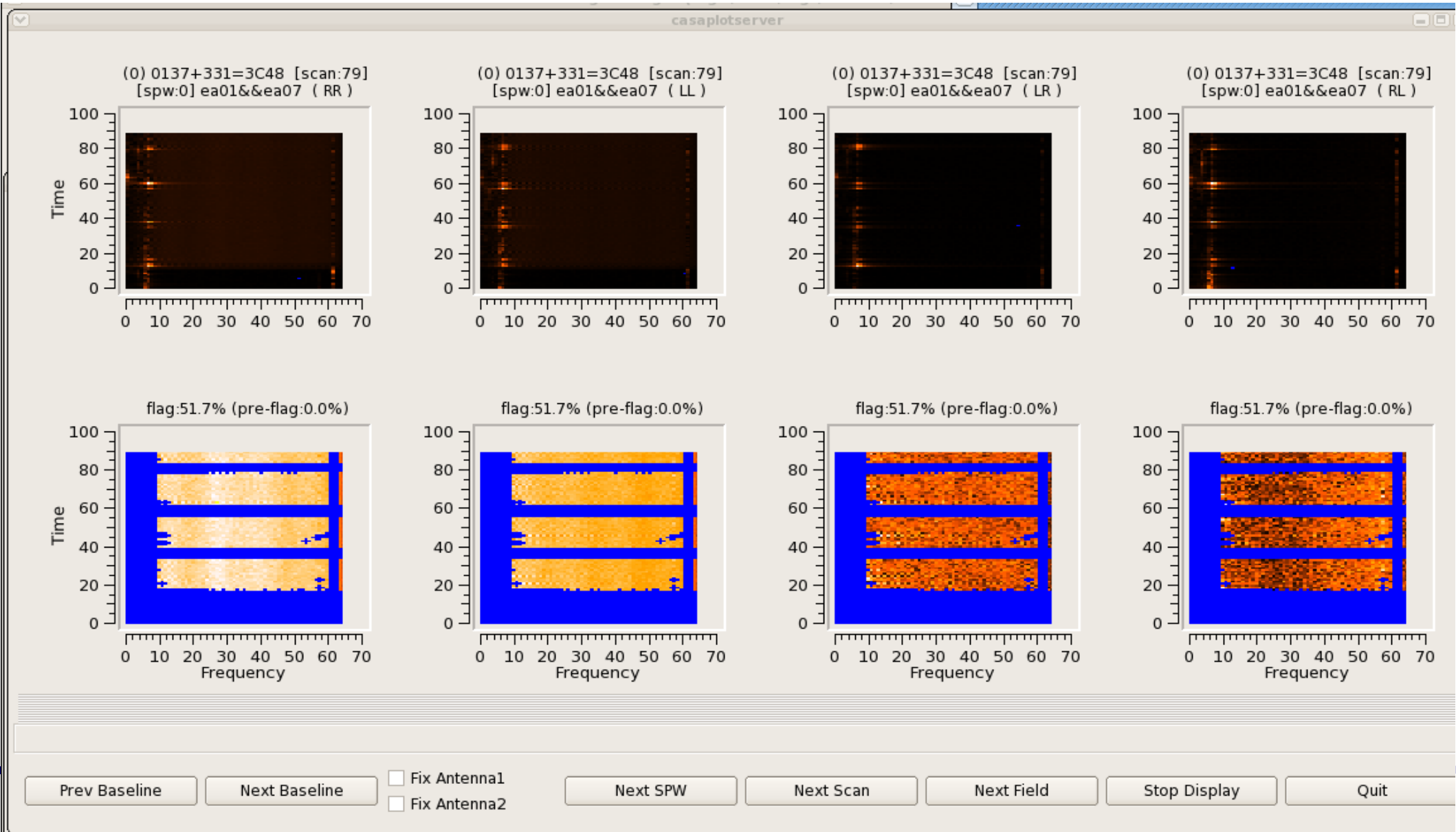
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



TFCrop (B. I. RFI): reduced poly-fit pieces + lower thresholds + poly-fit in time + extend + extend, no H-S

cmdlist = [“ spw='5' mode='tfcrop' **maxnpieces=4 timecutoff=2.5 freqcutoff=3.0 timefit='poly'** extendflags=F“, “ **spw='5' mode='extend' growtime=50.0 extendpols=T growfreq=50.0 flagnearfreq=T flagneartime=T** “]

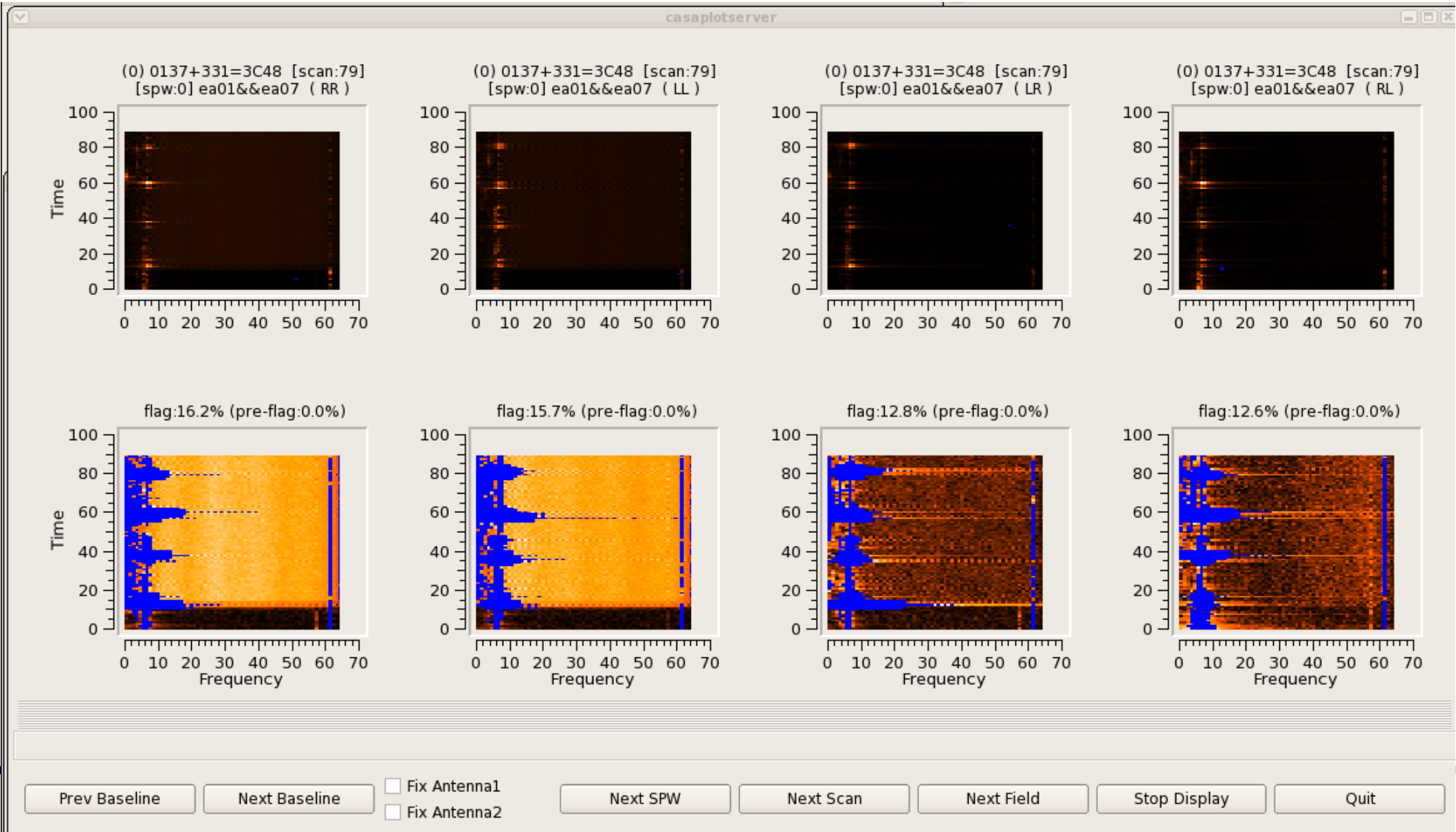
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (Broadband Intermittent RFI): defaults, no H-S

cmdlist = [“ spw='5' mode='rflag' extendflag=F “]

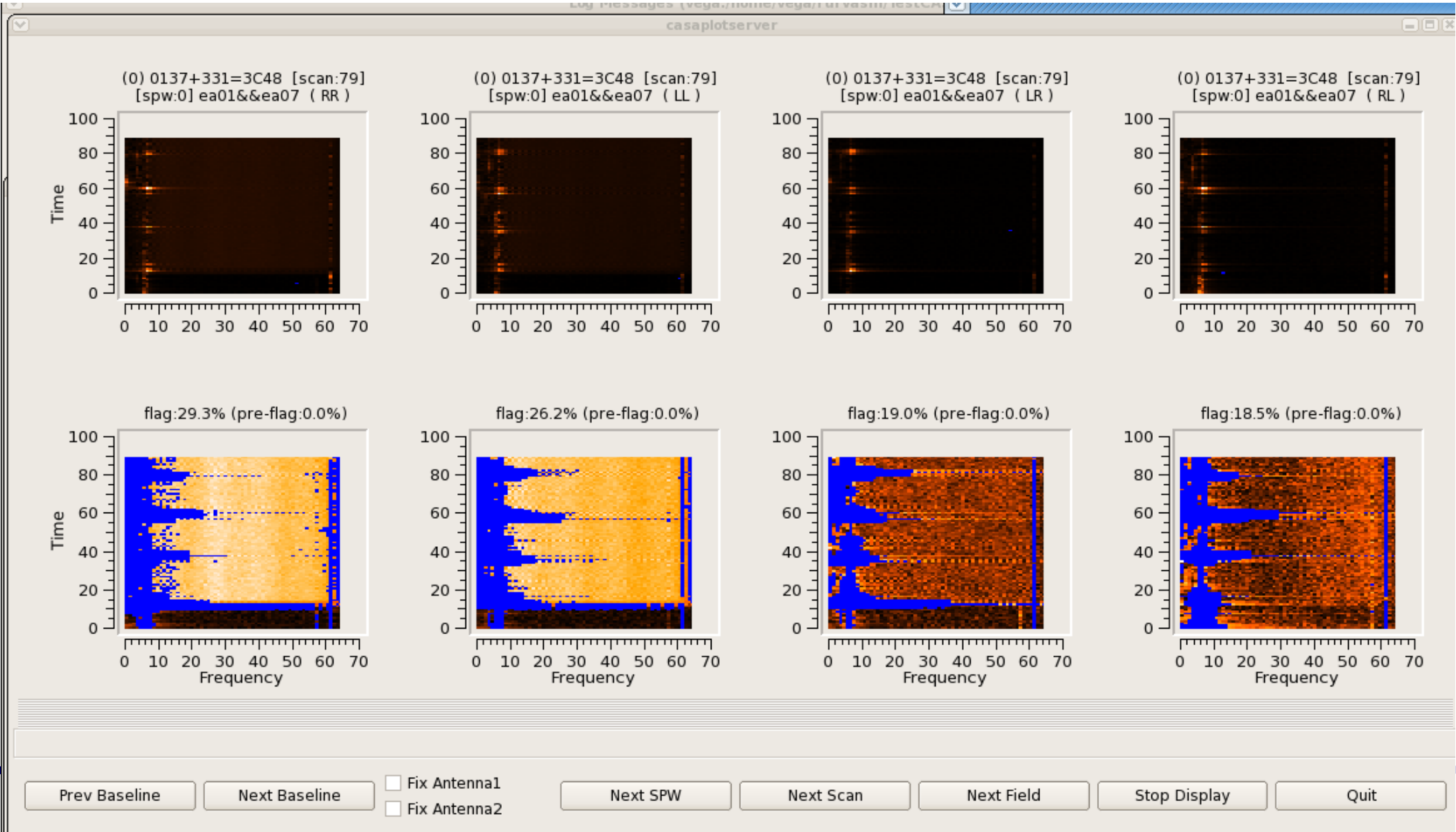
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (B.I. RFI): lower thresholds, no H-S

cmdlist = [“ spw='5' mode='rflag' **freqdevscale=3.0 timedevscale=3.0** extendflag=F “]

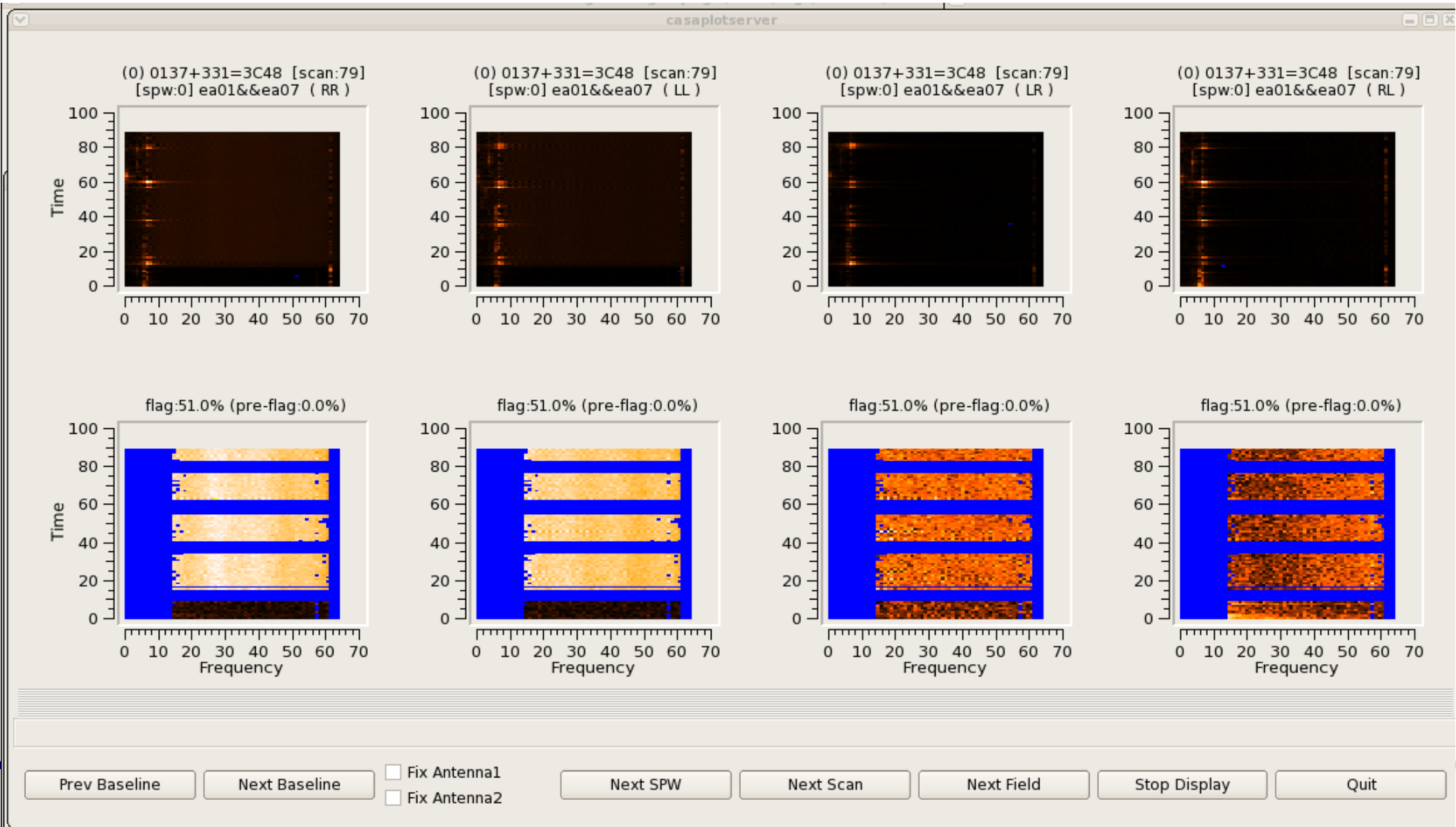
flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Rflag (B.I. RFI): lower thresholds + extend, no H-S

cmdlist = [“ spw='5' mode='rflag' **freqdevscale=3.0 timedevscale=3.0** extendflags=F“ ,
“ **spw='5' mode='extend' growtime=50.0 growfreq=30.0 extendpols=T** “]

flagdata(vis='xxx.ms', mode='list', inpfile=cmdlist, action='calculate', display='data')



Choosing what to do.... and when...

	TFCrop (search for spikes above smooth base, per baseline)	**Rflag** (use local stats vs global stats to find outliers)
Strong spiky RFI	Good	Good, but continuous RFI (time or freq) needs tuning.
Noisy RFI	Good if spikes are bright enough. Not good for low noisy RFI	Good
Broadband RFI	Not robust, but possible with tuning of polynomial fit (maxnpieces)	Good if RFI looks noisy. Continuous RFI needs tuning.
Un-calibrated data	Yes, can fit underlying bandshape	No, it needs a flat base
Extended emission	No problem, since each baseline is treated separately.	Stats are biased by high flux on short baselines (use uvrange, or operate on residual data)

One suggested usage : TFCrop on uncalibrated data (to catch brightest RFI) + Rflag on calibrated target data (to catch rest of RFI)

Hanning Smoothing : Helps for strong RFI where ripples are seen in surrounding channels. But, widens RFI and can cause overflagging if there is an 'RFI forest'.

Summary

– Automatic Flagging options exist.

– They all need tuning. Usually, one setup per SPW or band

=> Look at small pieces of your data, and decide flagging strategy

=> Use plotms or viewer or flagdata(action='calculate', display='data') and try different flagging setups.

=> Defaults will not suffice for all cases, experiment with various parameters.

– Use batch-modes in flagdata/flagcmd when relevant.

– Documentation : CASA – Using Casa – Other Documentation – Flagging
AIPS – rflag