Automated Tuning of RFI Identification and Flagging Algorithms

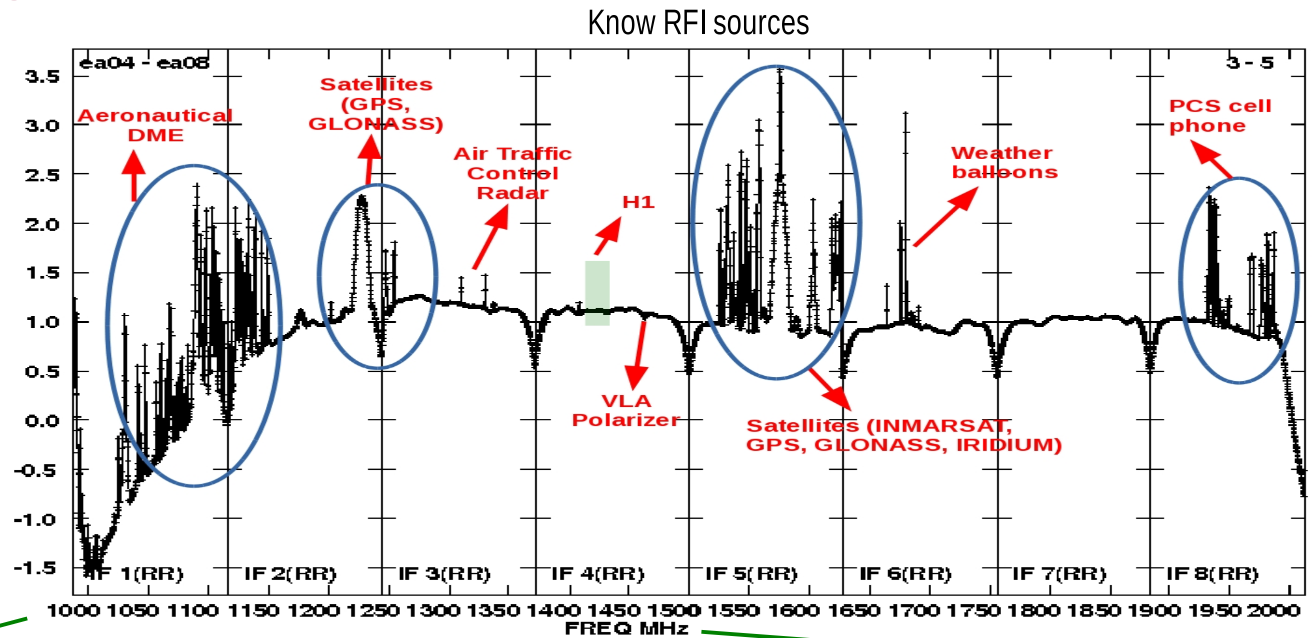
Bruno Martins & **Urvashi Rau**

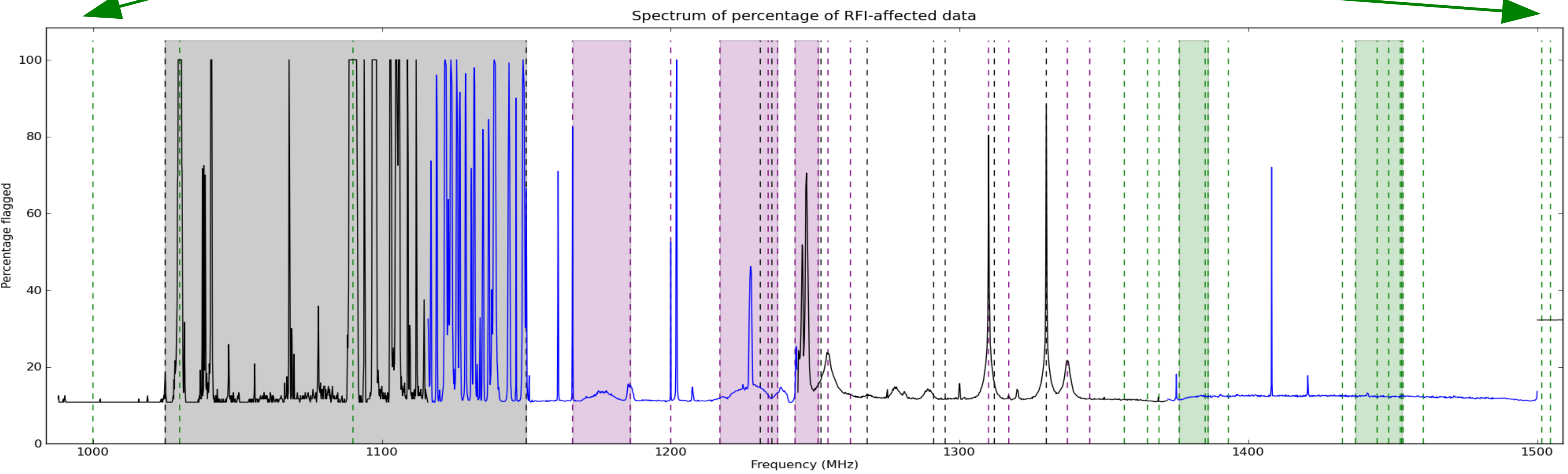
UNISUL / UVA
Brazil / USA
(2016 Summer Intern at NRAO)

NRAO
USA

RFI2016 : Coexisting with Radio Frequency Interference
17 – 20 October 2016

RFI at the VLA





Problem : All our algorithms need manual tuning !

- Autoflag Algorithms Exist

  tfcrop : Outlier detection on 2D time-freq slices [NRAO - CASA]

  rflag : Heirarchical statistical approach [NRAO – AIPS,CASA]

  aoflagger : Outlier detection on 2D time-freq slices [ASTRON]

- They all need tuning for different types of RFI and telescopes

- Goal : automate this tuning via a genetic algorithm that simulates parameter evolution to optimize a “flagging quality” metric

- Intended use : Tune parameters on a small fraction of the data and then apply the results to the entire dataset (or class of datasets)

TFCrop

Adapted from U.Rau, A.P.Rao,
GMRT Tech Rep 2003

For each 2D time-freq plane (per antenna pair)

(1) Form an average along one dimension

(2) Calculate a robust piece-wise polynomial fit across the base of RFI spikes

(3) Flag un-averaged values deviating from the fit by > N-sigma

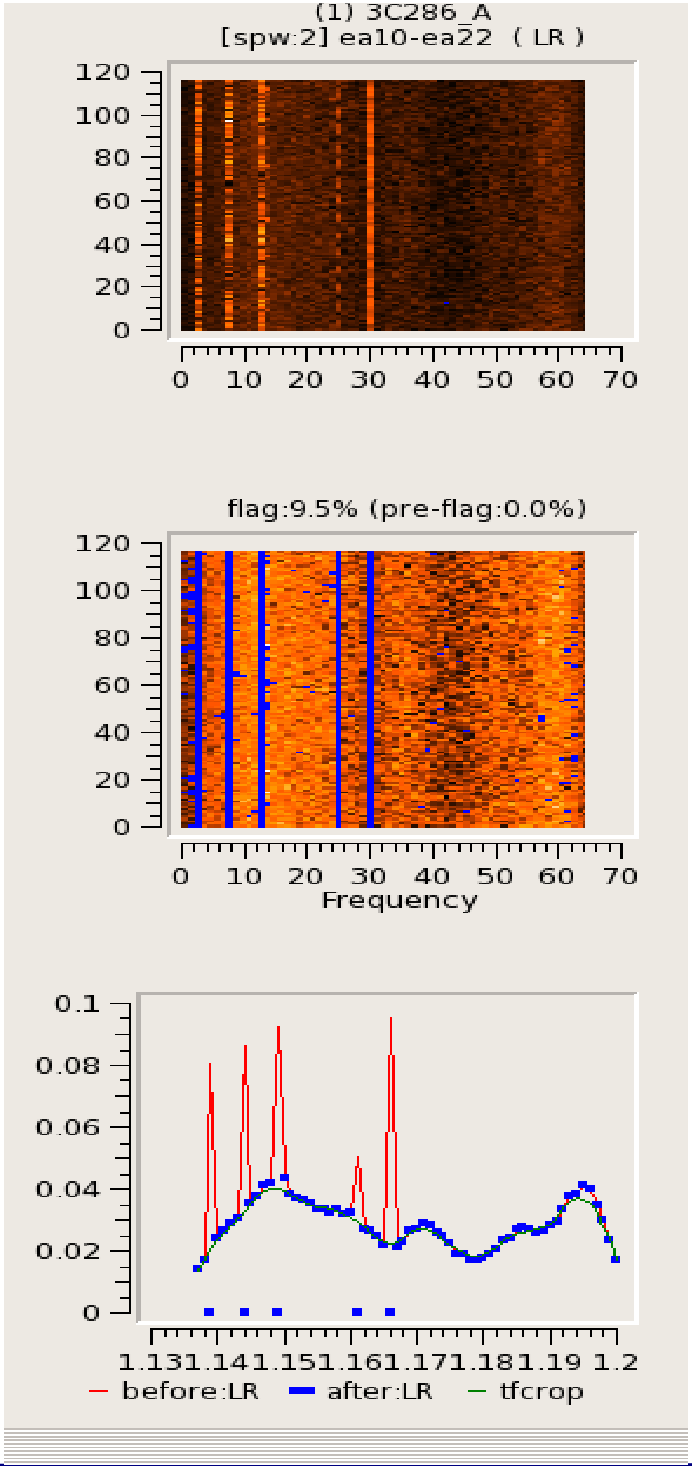
(4) Repeat (1-3) along the other dimension.

Relevant Parameters :

timecutoff, freqcutoff : N-sigma thresholds

usewindowstats : Ways to detect deviation from the fit

maxnpieces : Tuning the robust polynomial fits



TFCrop

Adapted from U.Rau, A.P.Rao,
GMRT Tech Rep 2003

For each 2D time-freq plane (per antenna pair)

(1) Form an average along one dimension

(2) Calculate a robust piece-wise polynomial fit across the base of RFI spikes

(3) Flag un-averaged values deviating from the fit by > N-sigma

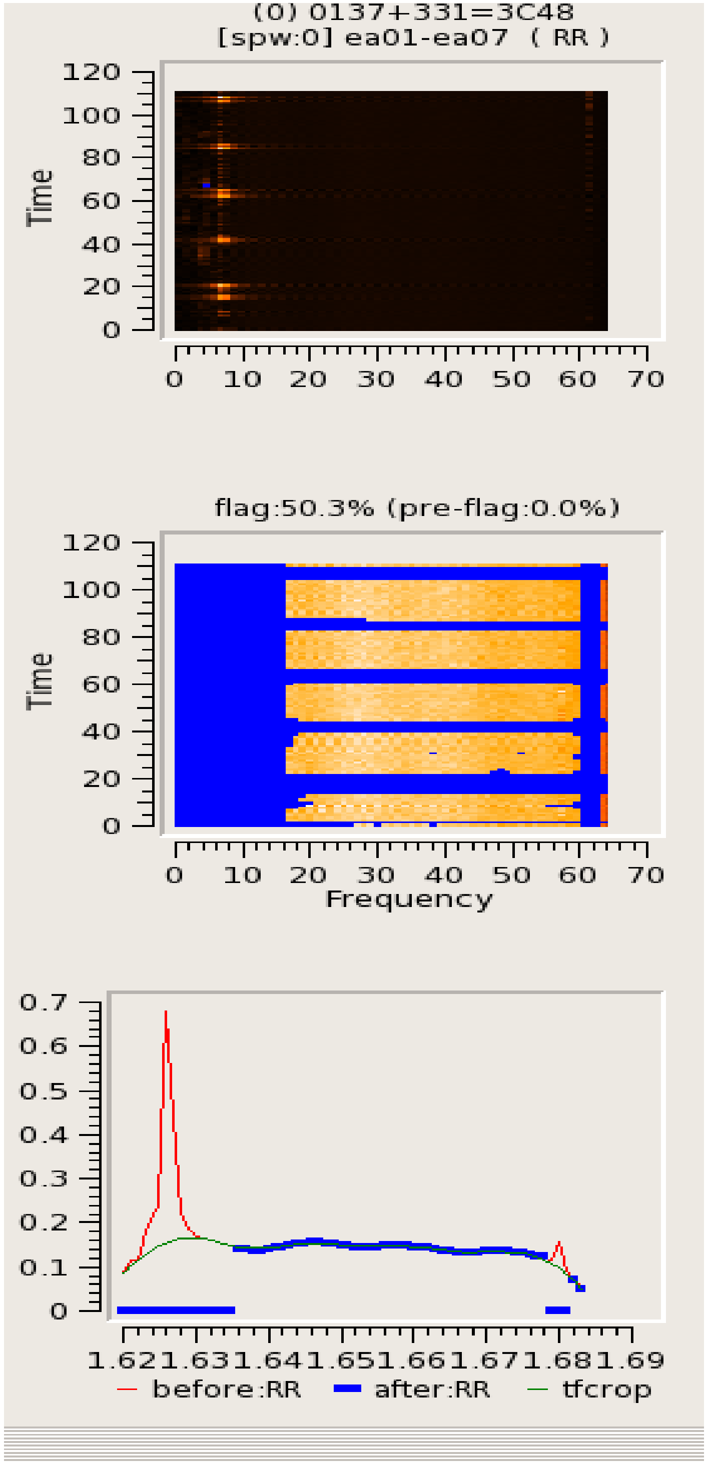
(4) Repeat (1-3) along the other dimension.

Relevant Parameters :

timecutoff, freqcutoff : N-sigma thresholds

usewindowstats : Ways to detect deviation from the fit

maxnpieces : Tuning the robust polynomial fits



RFlag

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

Repeat* along time and frequency axes :

(1) Calculate local RMS of real and imag parts of visibilities within a sliding window.

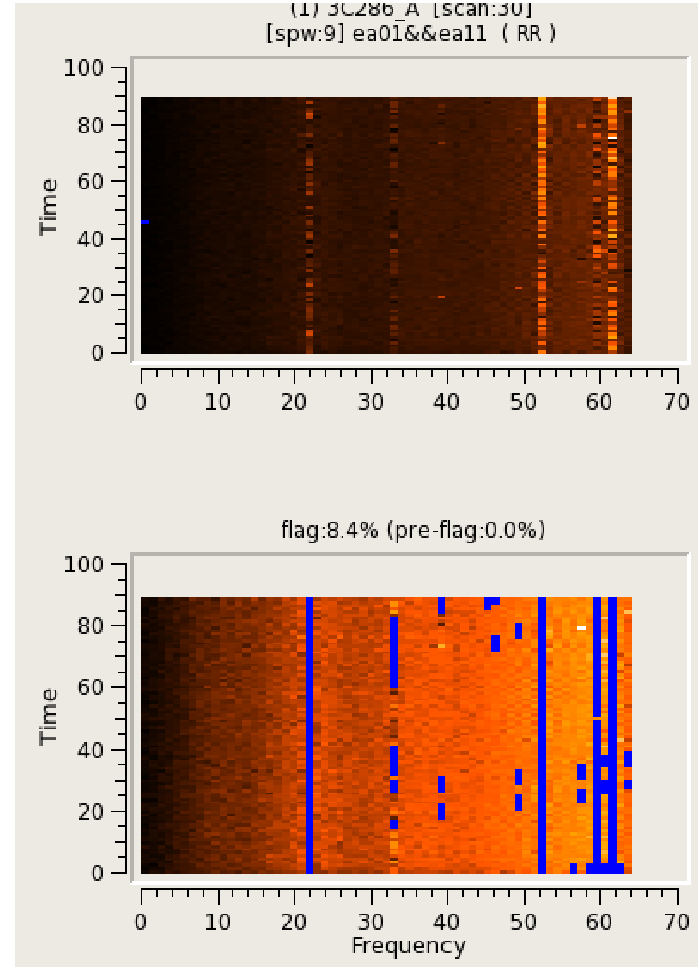
(2) Calculate the median RMS across windows, deviations of local RMS from this median, and the median deviation

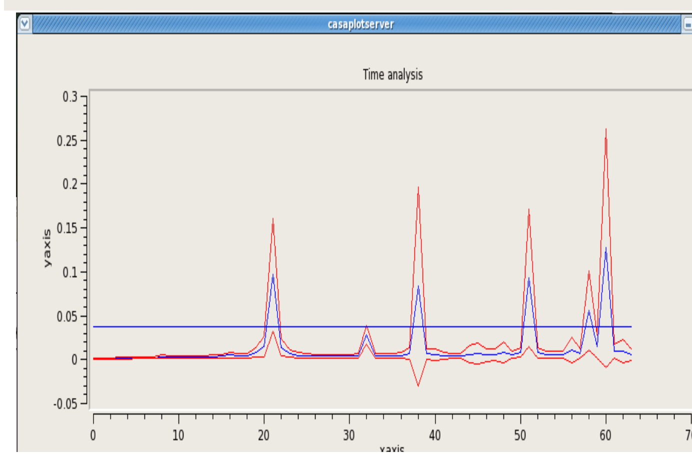
(3) Flag if
local RMS > N x (medianRMS + medianDev)

(Most) Relevant Parameters :

timedevscale, freqdevscale : Threshold scale factors

winsize : Sliding window size



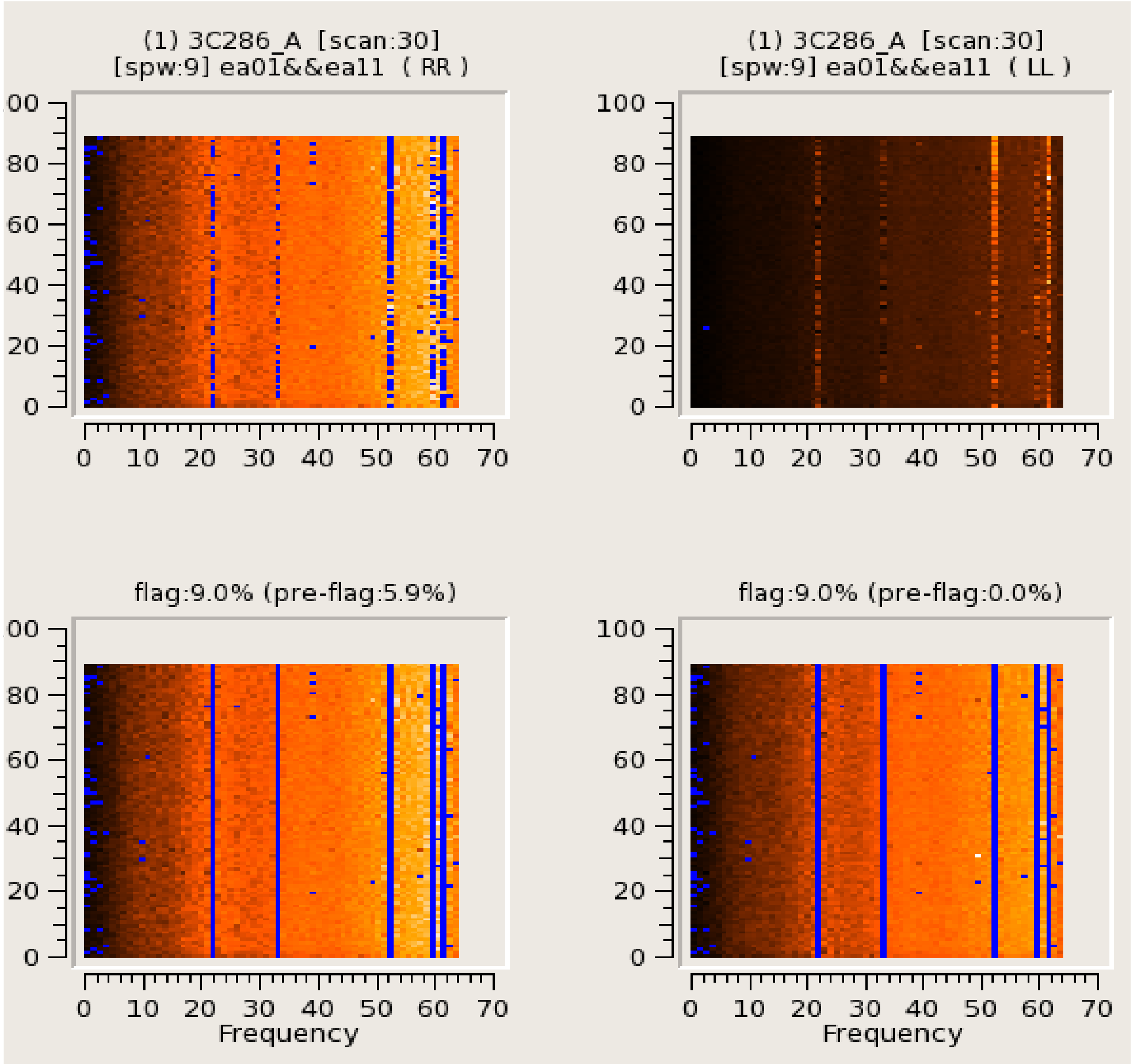


Extend Flags

(1) Fill flags if more than X% is already flagged along time(freq)

(2) Merge flags across pols

(3) Flag (based on) surrounding cells



Relevant Parameters :

growtime, growfreq

extendpols

flagneartime, flagnearfreq

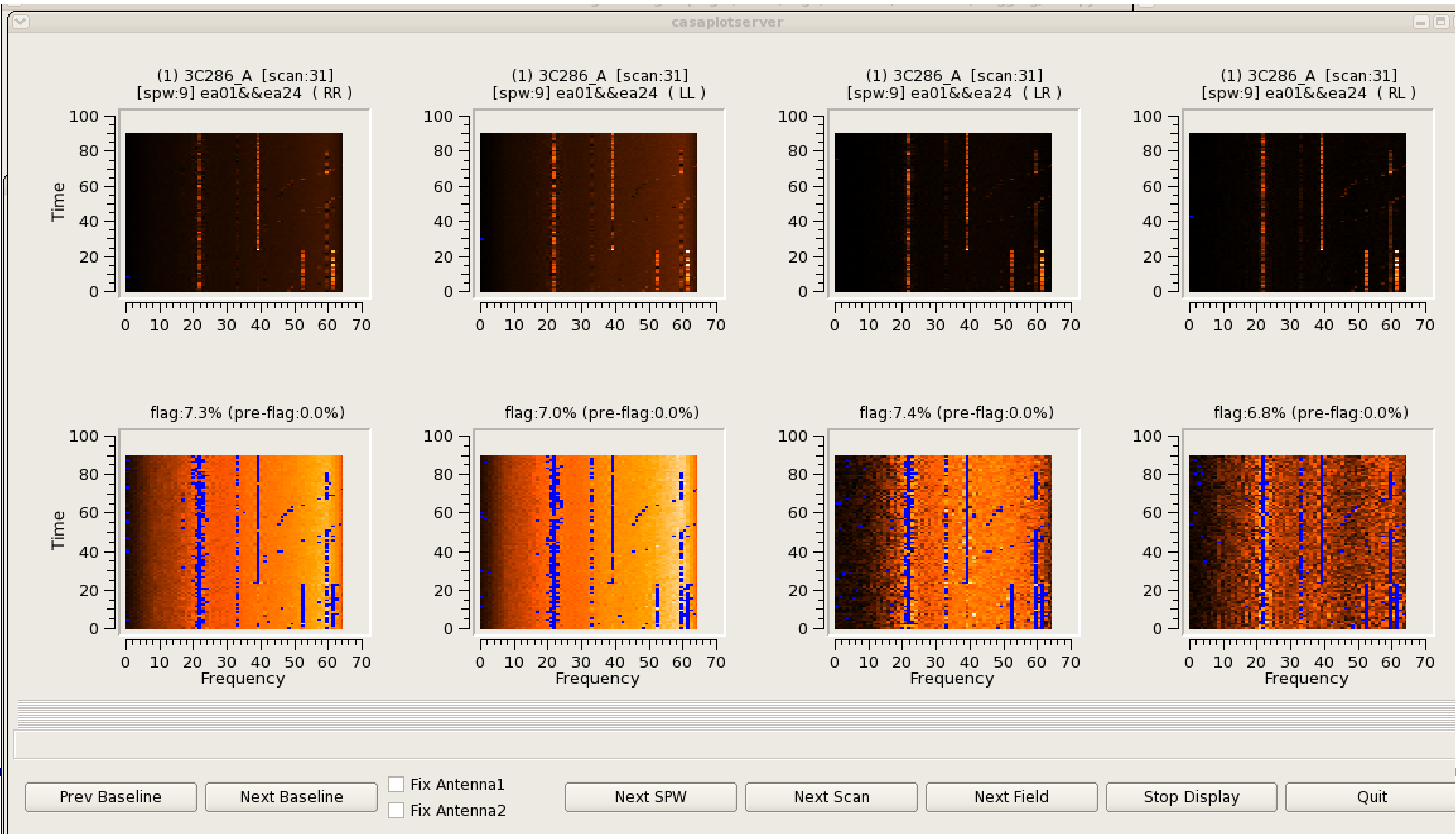
growaround

Example : Flag only RR (left panels) + growtime=30.0 + extendpol=True

Examples of Manual Tuning

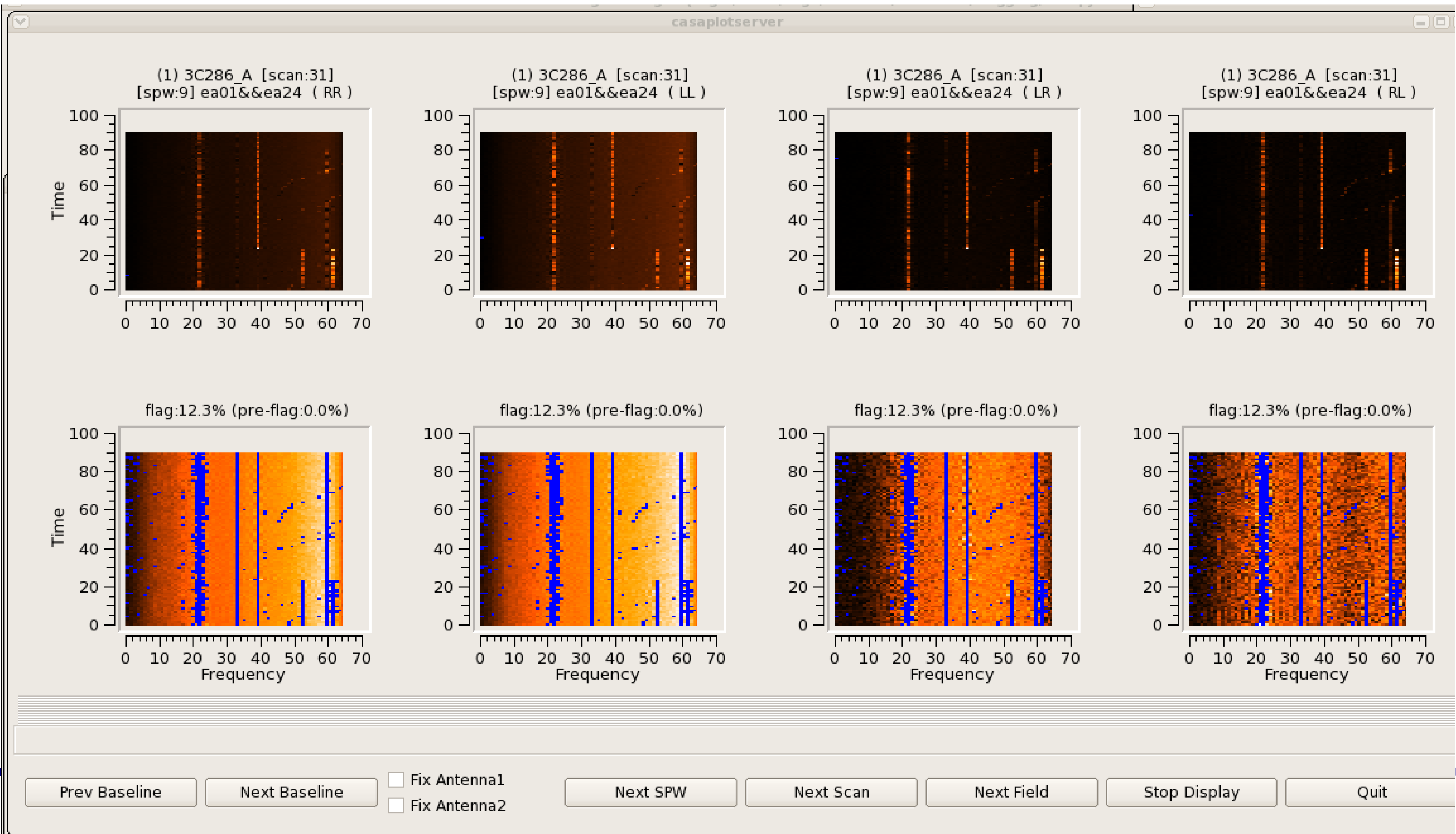
Examples of Manual Tuning

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



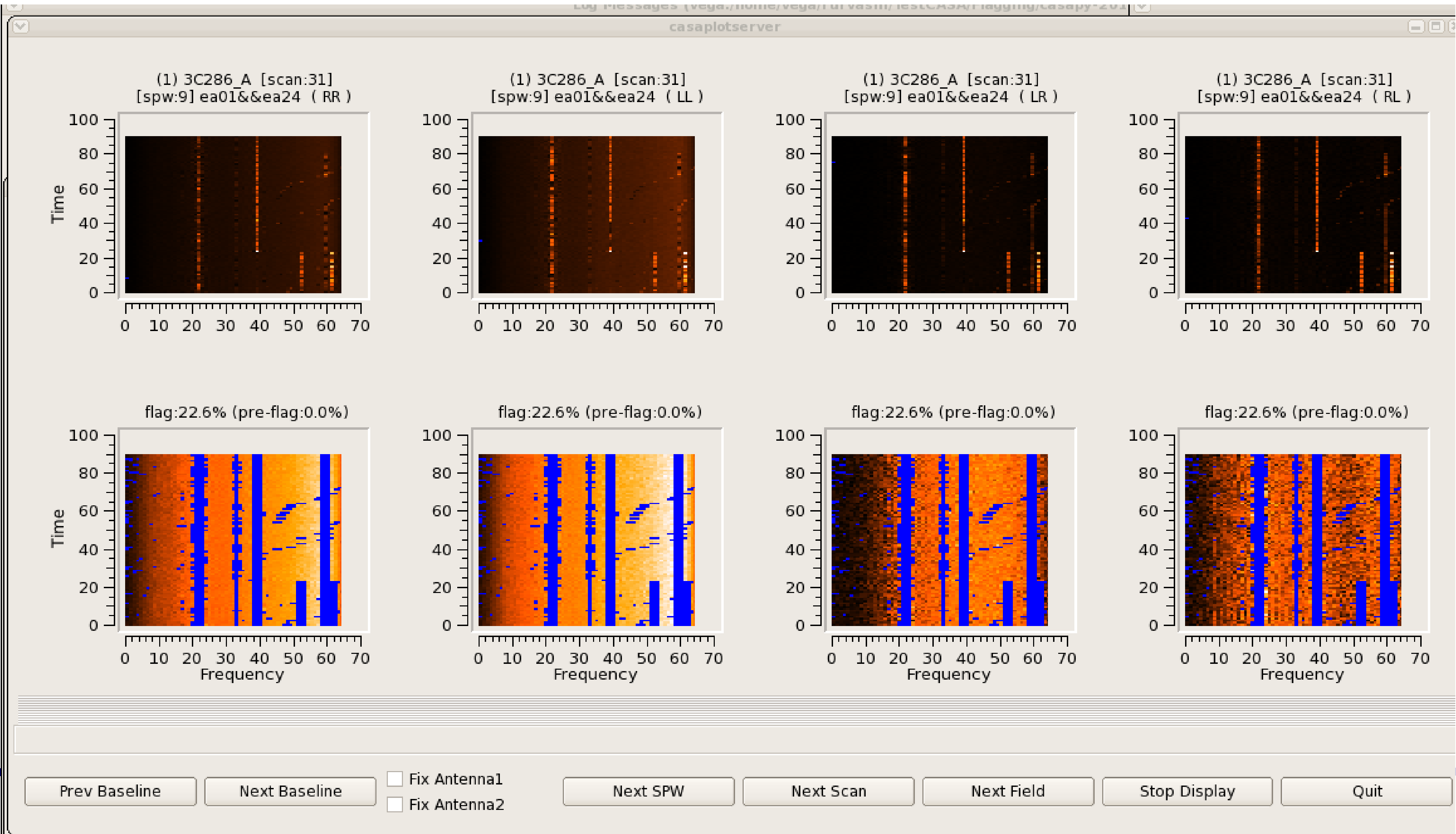
Examples of Manual Tuning

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



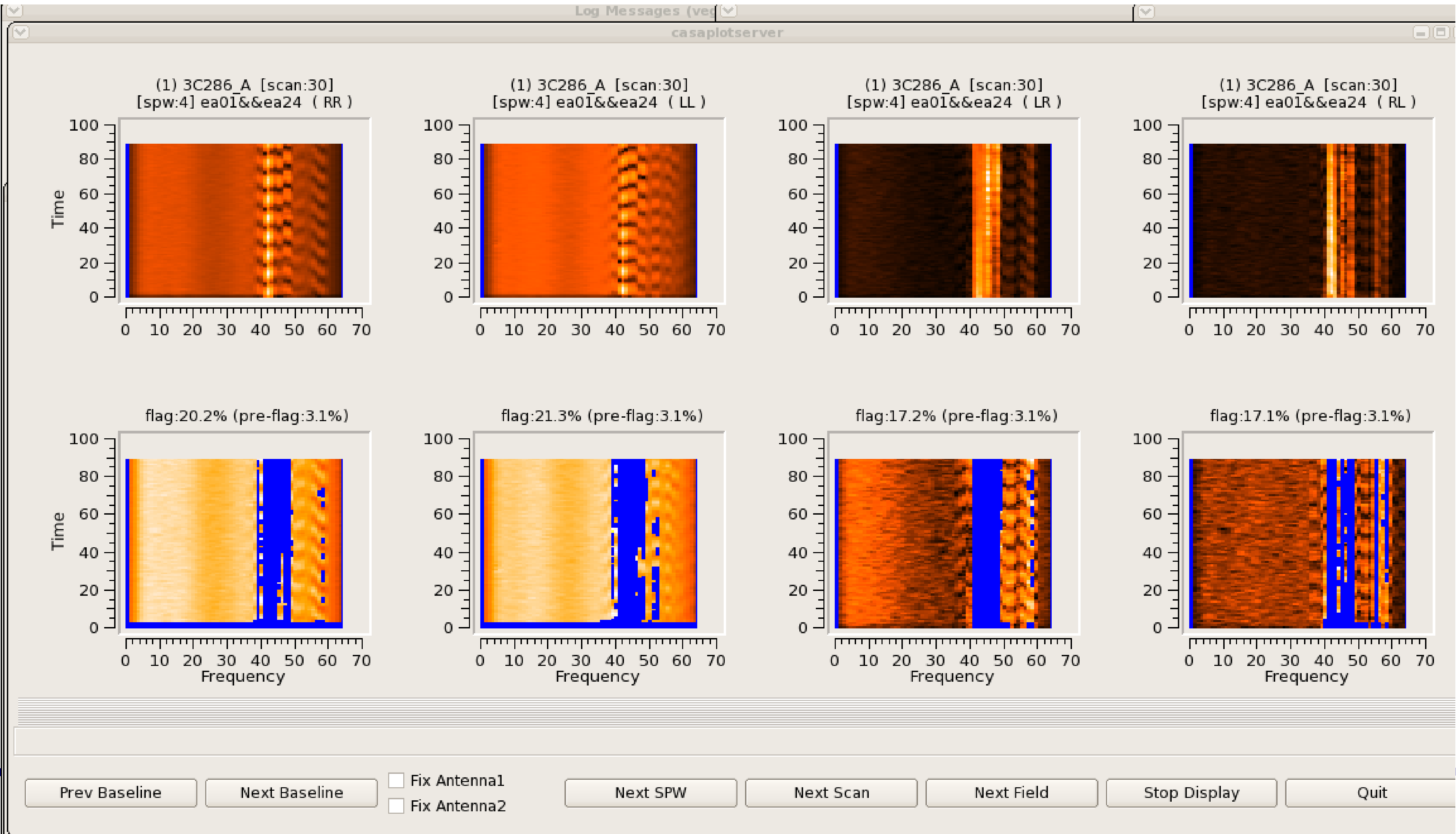
Examples of Manual Tuning

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



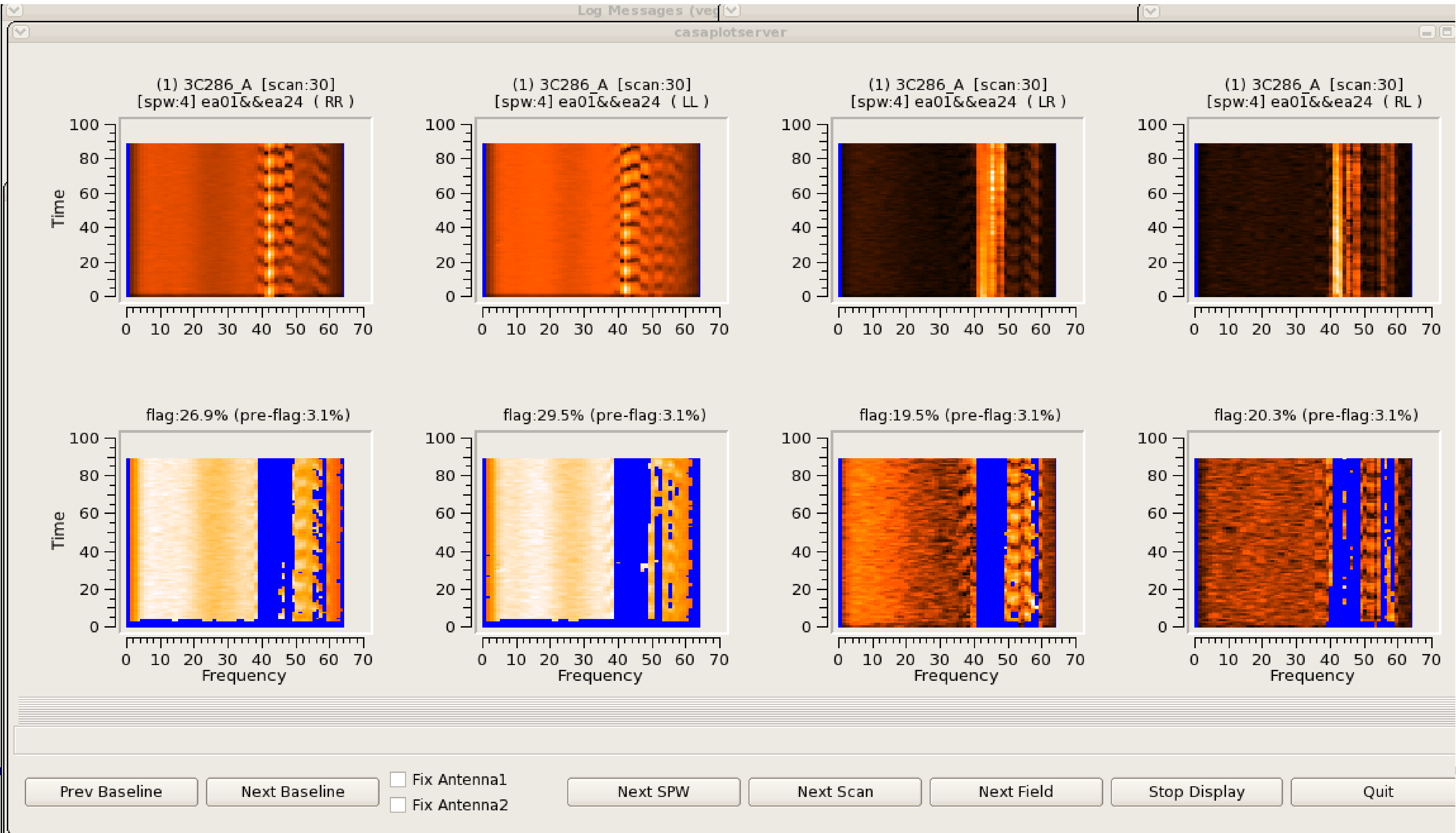
Examples of Manual Tuning

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



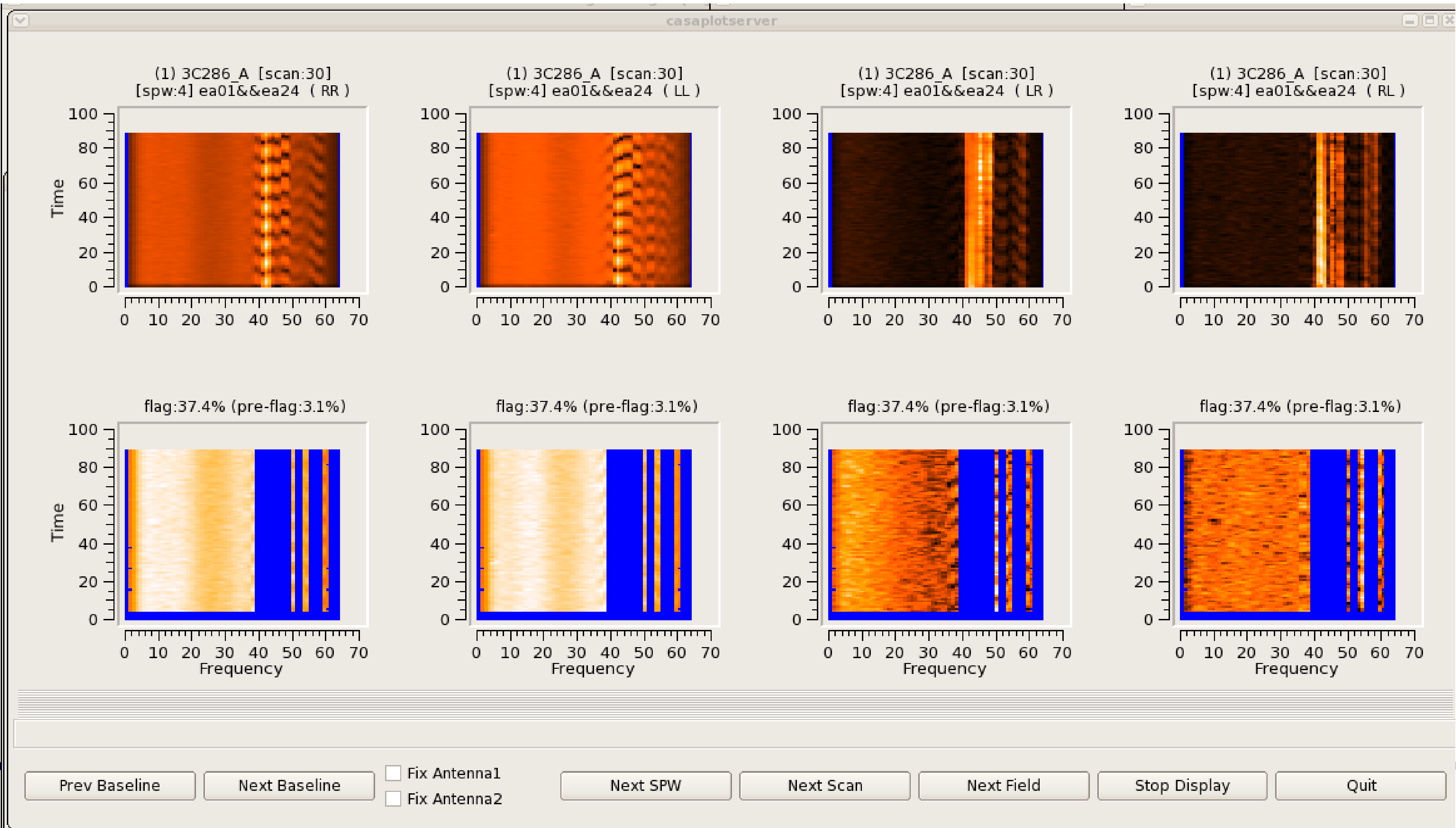
Examples of Manual Tuning

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



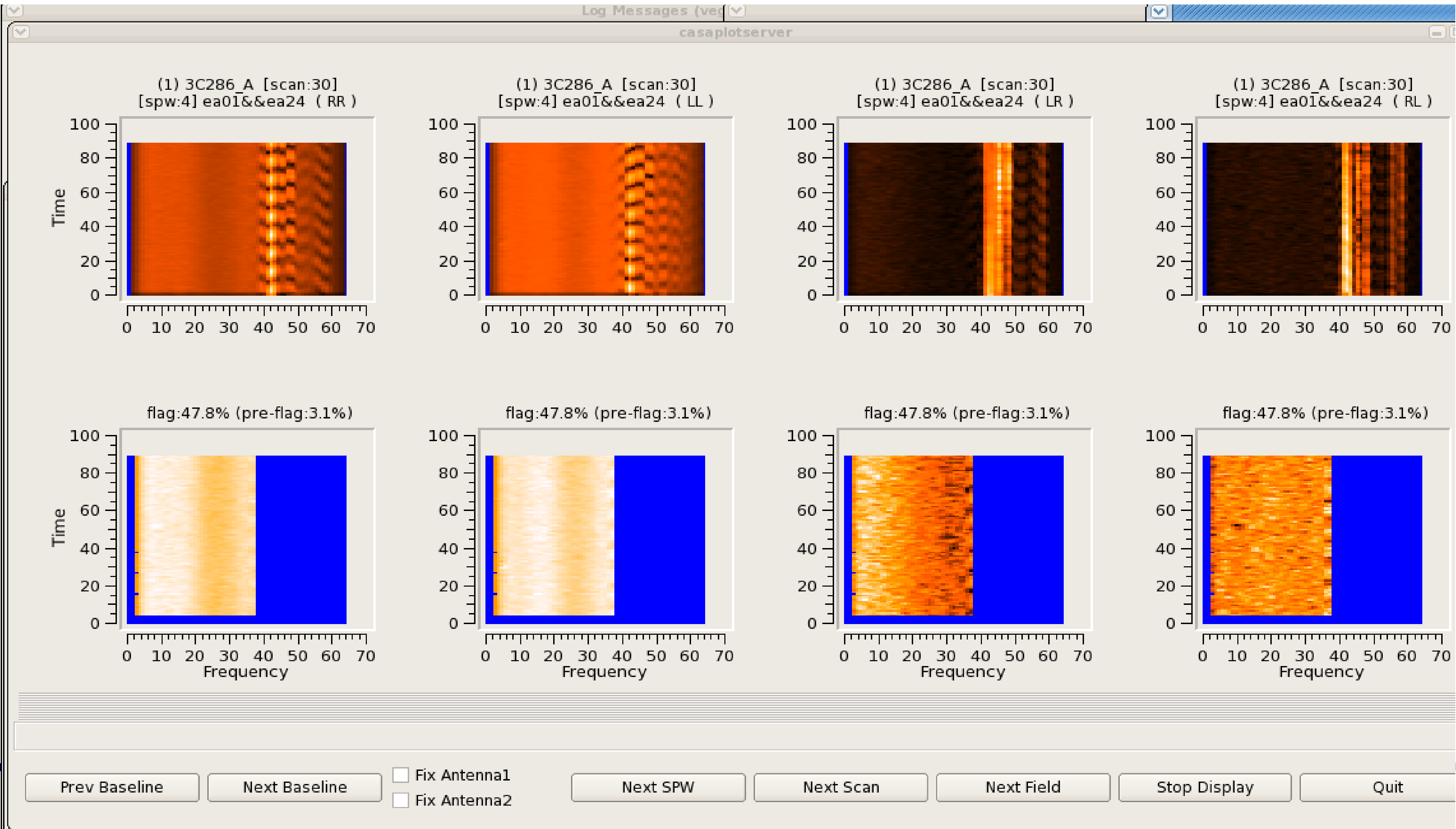
Examples of Manual Tuning

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



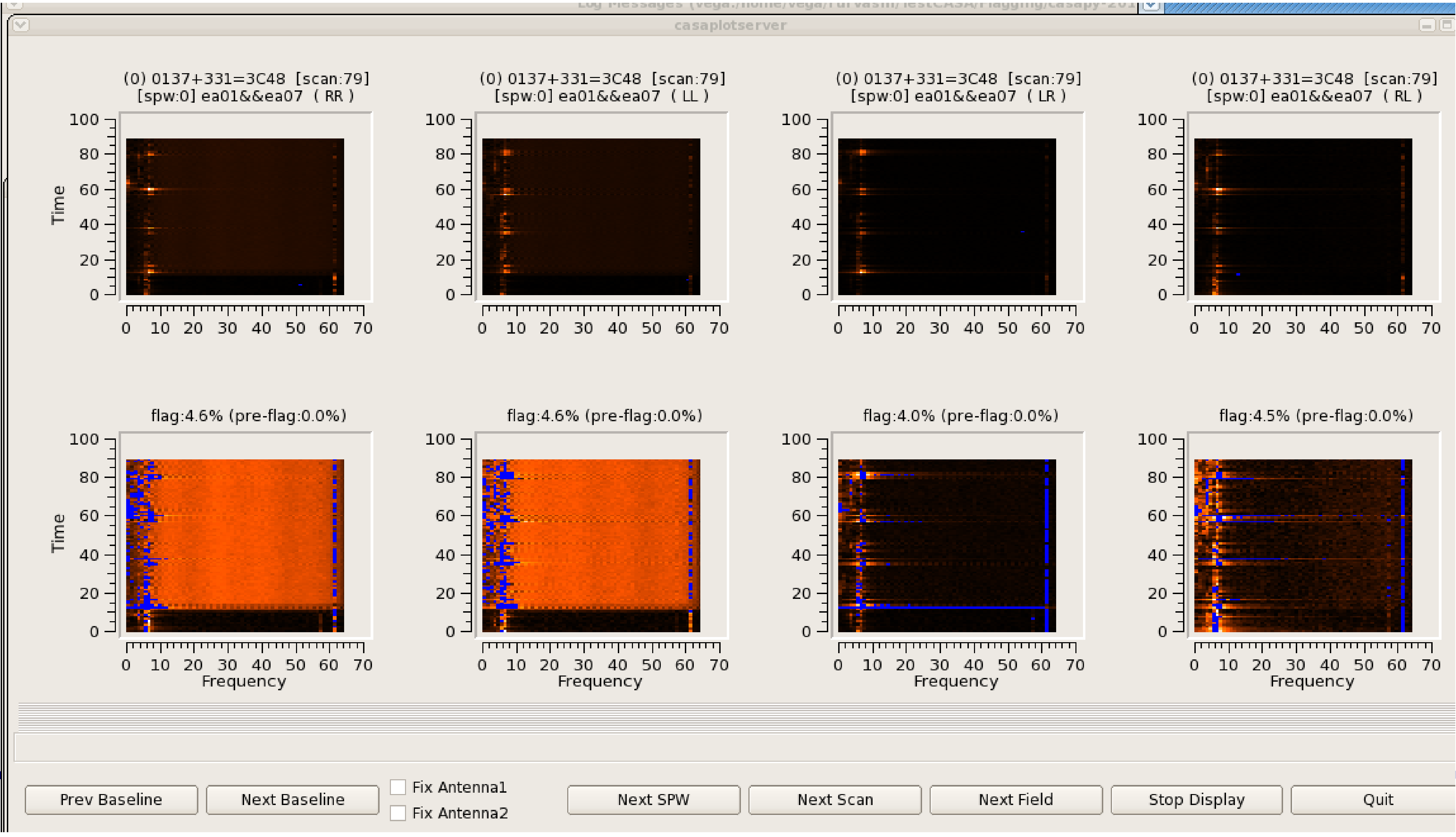
Examples of Manual Tuning

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



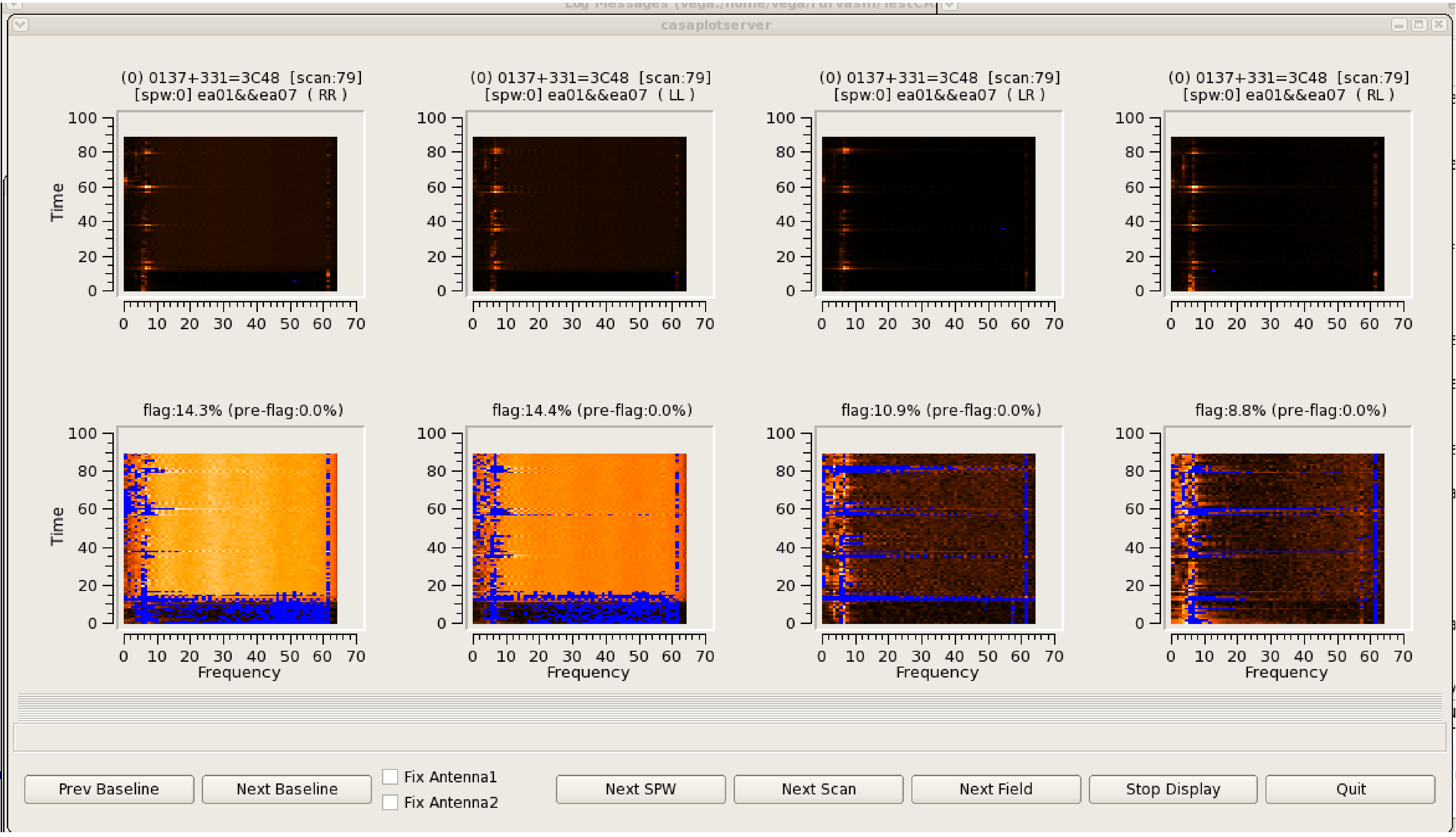
Examples of Manual Tuning

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



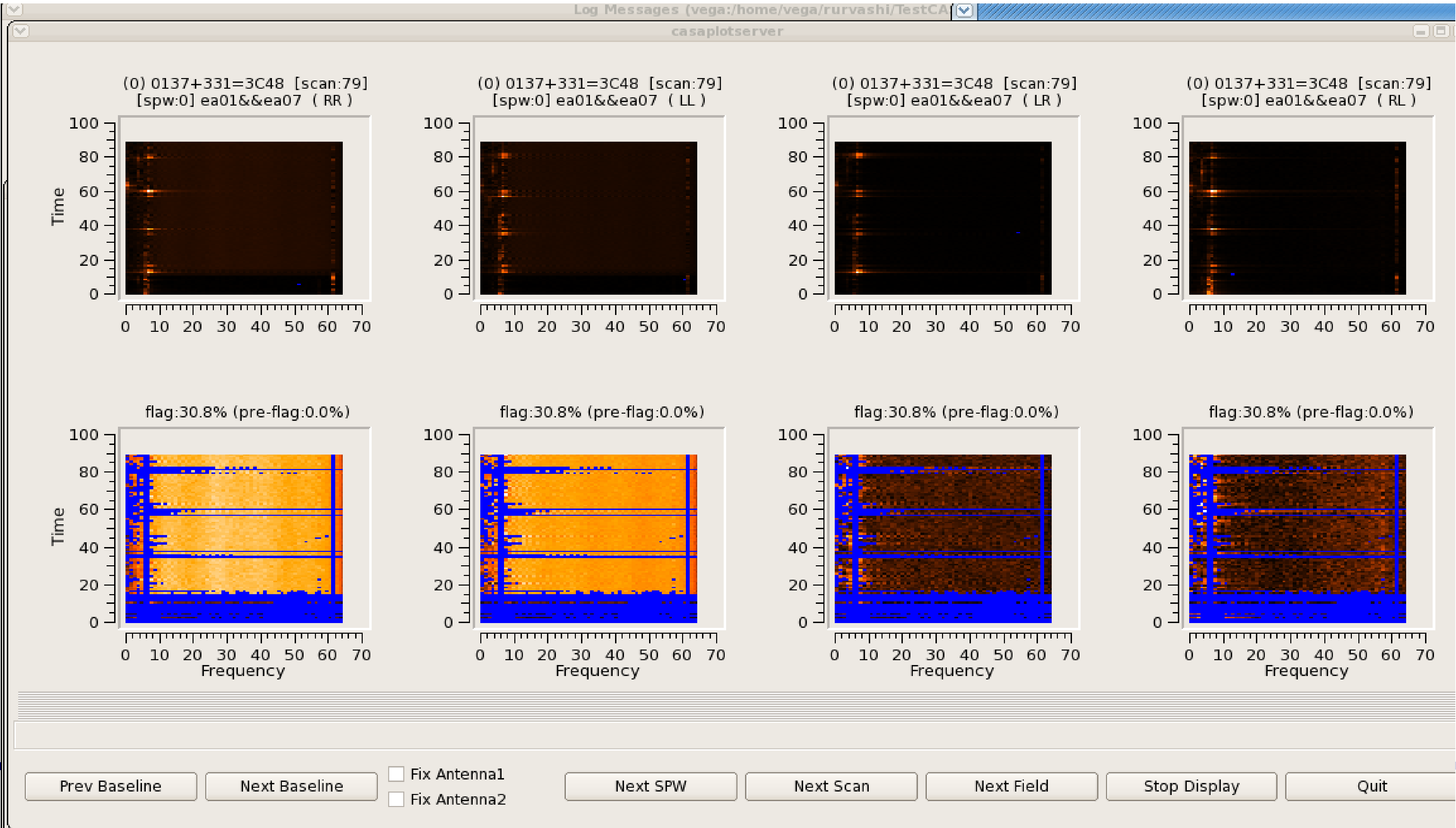
Examples of Manual Tuning

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



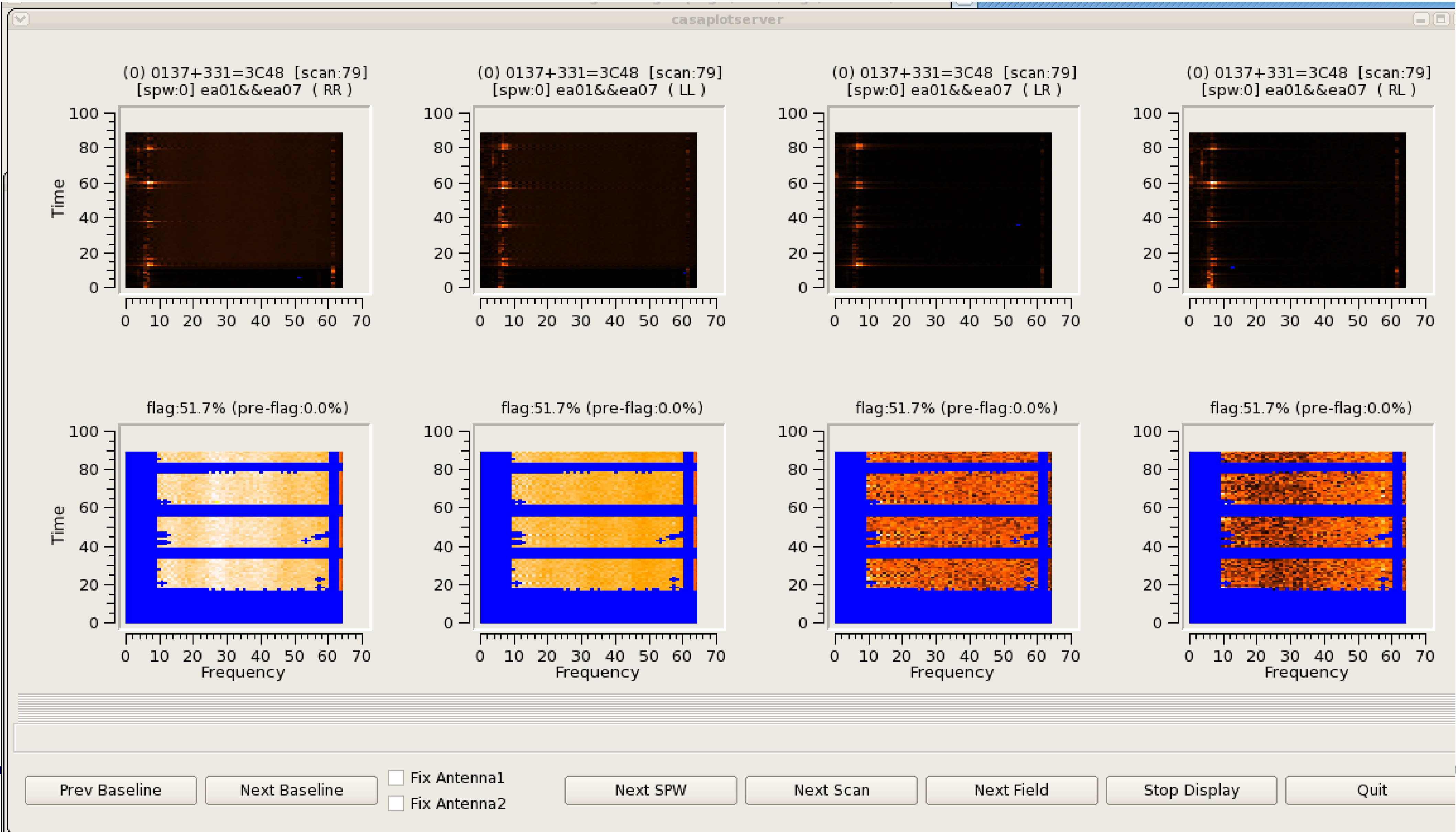
Examples of Manual Tuning

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** “]



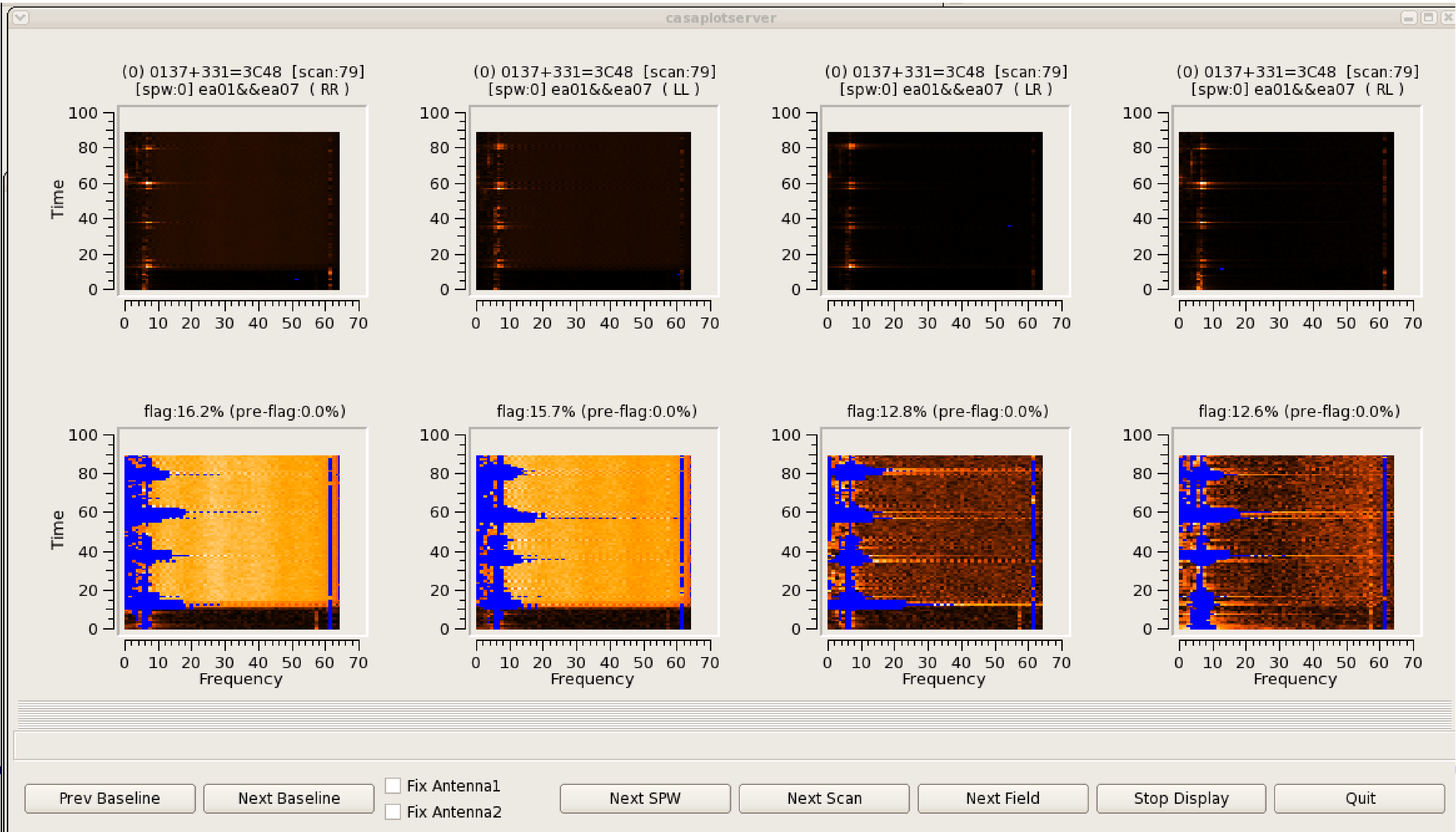
Examples of Manual Tuning

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** “]



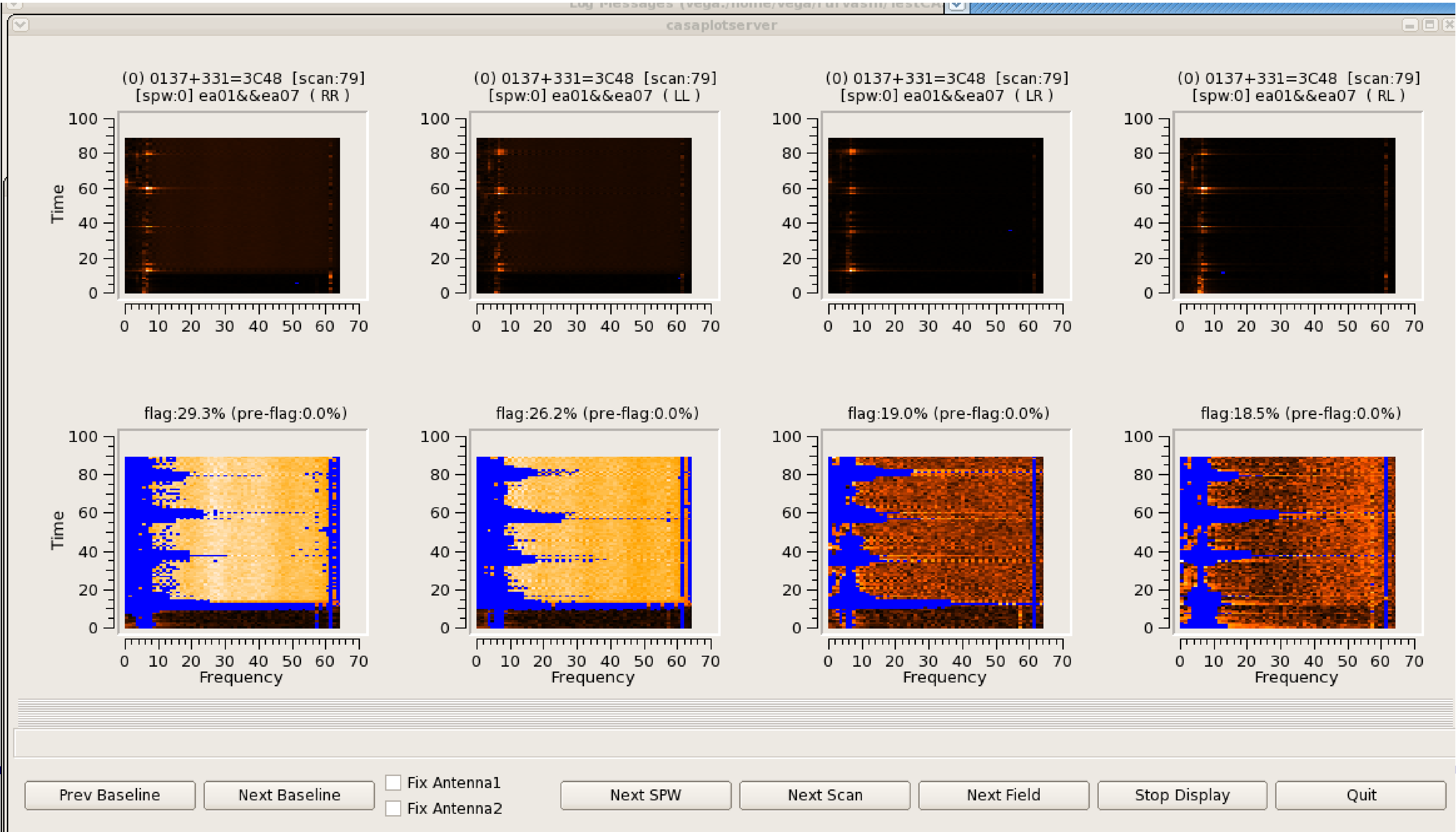
Examples of Manual Tuning

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



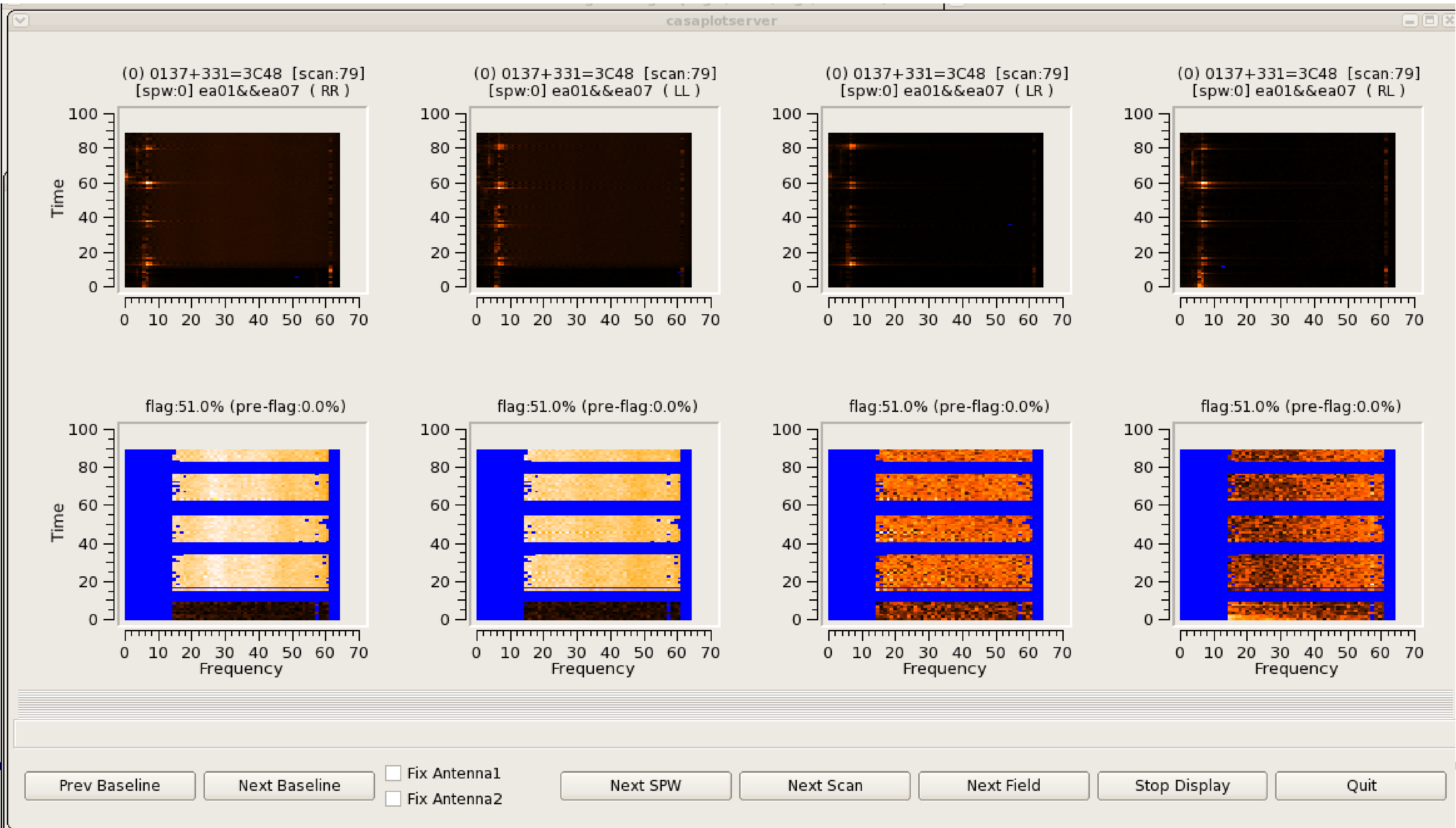
Examples of Manual Tuning

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



Examples of Manual Tuning

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** “]



Can we automate all this tuning ?

- Examples show that the algorithms CAN be tuned for optimal results.
- A human tweaks parameters and visually checks flagging quality on a small fraction of the dataset before letting it run on all the data.
- Each algorithm has a small set of really relevant parameters.

General approach :

=> Quantify 'flagging quality'

=> Apply an algorithm that optimizes it to find best-fit parameters

=> Auto-tune parameters on a subset of the data and apply to the rest

A Genetic Algorithm for Parameter Evolution

- A search heuristic that mimics the process of natural selection, used to solve optimization problems
- Random guided search

- Steps :

    (1) Generate initial population randomly

    (2) Breed new individuals through crossover and mutation

    (3) Evaluate the population using a fitness metric

    (4) Replace the least-fit population with new individuals

    (5) Repeat from (2) and continue through several generations

    (6) Pick the parameters of the best-fit individual

Characteristics of an Individual (and the population)

- Candidate solutions (individuals) are represented by parameter sets

  Individuals : TFCrop + Extend (or) RFlag + Extend

  tfcrop : timecutoff, freqcutoff, maxnpieces, usewindowstats

  rflag : timedevscale, freqdevscale, winsize

  extend : growtime, growfreq, flagneartime, flagnearfreq, growaround

- Choices : Population size, Number of generations,

  Dropout rate, mutation rate,

  Allowed ranges for parameter values (from prior knowledge)

Reproduction (crossover)

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround

cut

parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

Left

cut

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

Left | cut | Right

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

Left | cut | Right

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE
child1 =								

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround

Left | cut | Right

parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

child1 =	0.5	3.0	4	70.0

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE
child1 =	0.5	3.0	4	70.0	90.0	TRUE	FALSE	TRUE

Left

cut

Right

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover)

Left | cut | Right

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

child1 =	0.5	3.0	4	70.0	90.0	TRUE	FALSE	TRUE
child1 =	1.5	5.0	2	50.0	80.0	FALSE	FALSE	FALSE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover) + Mutation

Left cut Right

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

child1 =	0.5	3.0	4	70.0	90.0	TRUE	FALSE	TRUE
child1 =	1.5	5.0	2	**60.0***	80.0	**TRUE***	FALSE	FALSE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals

Reproduction (crossover) + Mutation + Dropout

rflag =	timedevscale	freqdevscale	winsize	growtime	growfreq	flagneartime	flagnearfreq	growaround

Left | cut | Right

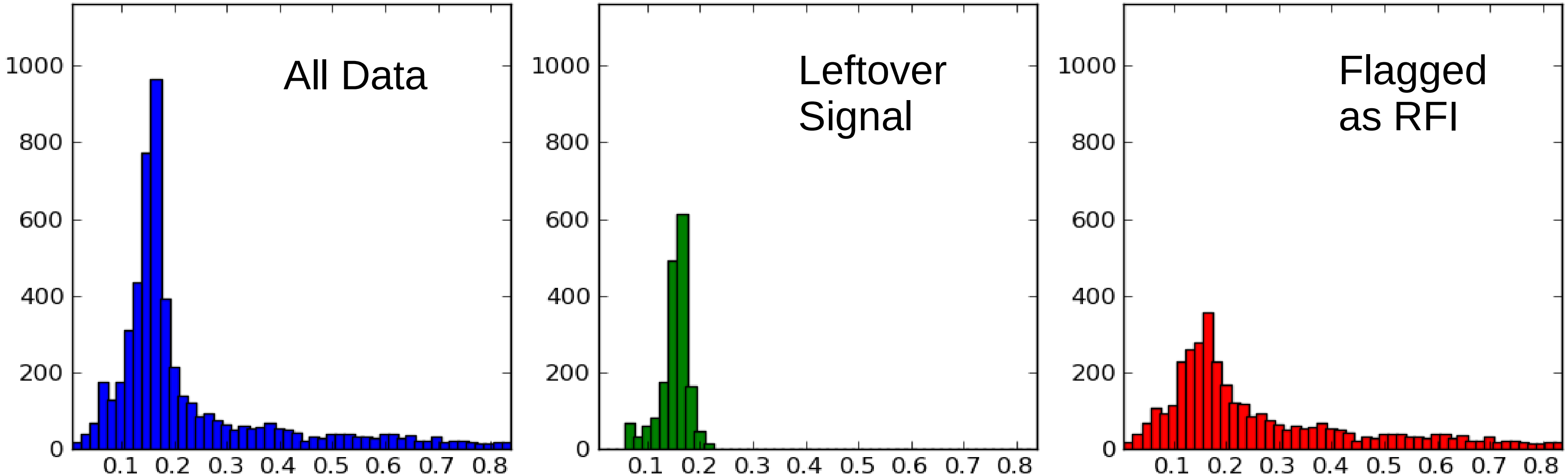
parent1 =	0.5	3.0	4	70.0	80.0	FALSE	FALSE	FALSE
parent2 =	1.5	5.0	2	50.0	90.0	TRUE	FALSE	TRUE

child1 =	0.5	3.0	4	70.0	90.0	TRUE	FALSE	TRUE
child1 =	1.5	5.0	2	**60.0***	80.0	**TRUE***	FALSE	FALSE

- Breed each new generation of individuals by mixing the characteristics of all pairs of parent individuals
- Replace least-fit individuals with new ones

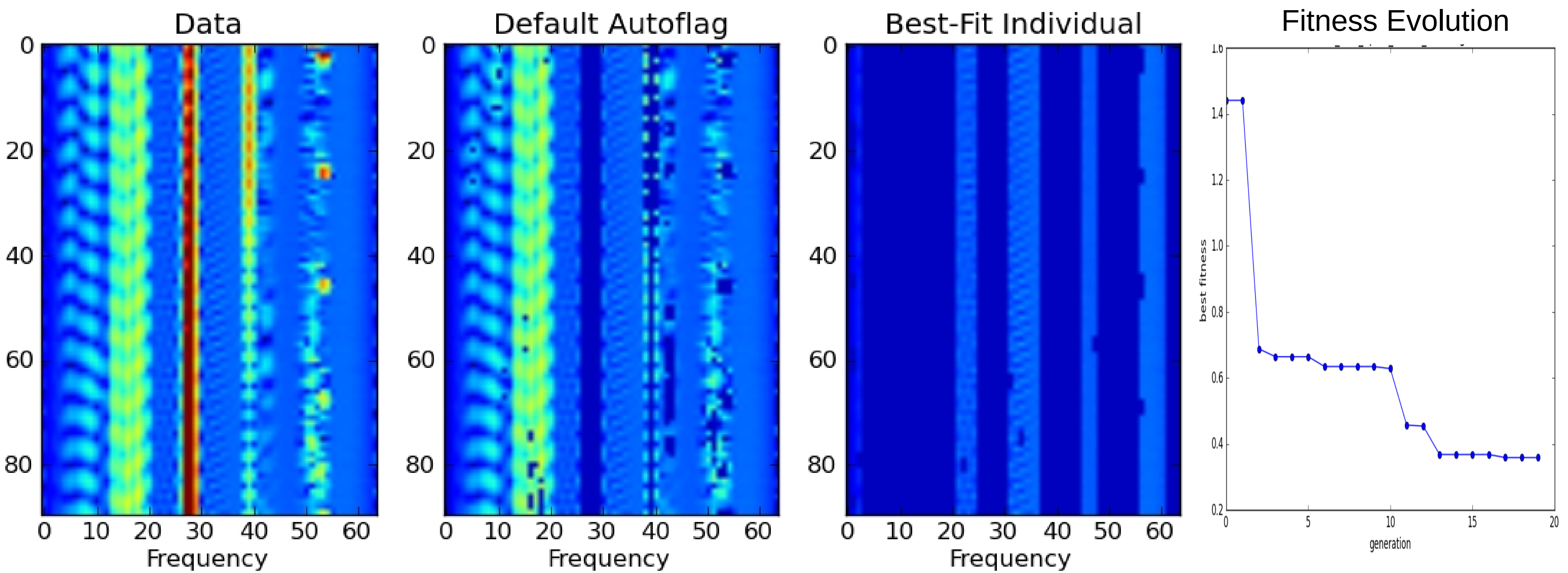
Fitness metric to evaluate each individual

- Compare statistics of flagged vs unflagged data
  - Flagged data should have a higher mean (or median) than unflagged data
  - Unflagged data should look Gaussian (max ~ 3-sigma)
  - Protect against over (or under) flagging (>70% flagged or 0% flagged)
- Use these criteria to compute a score that must be optimized.



Results

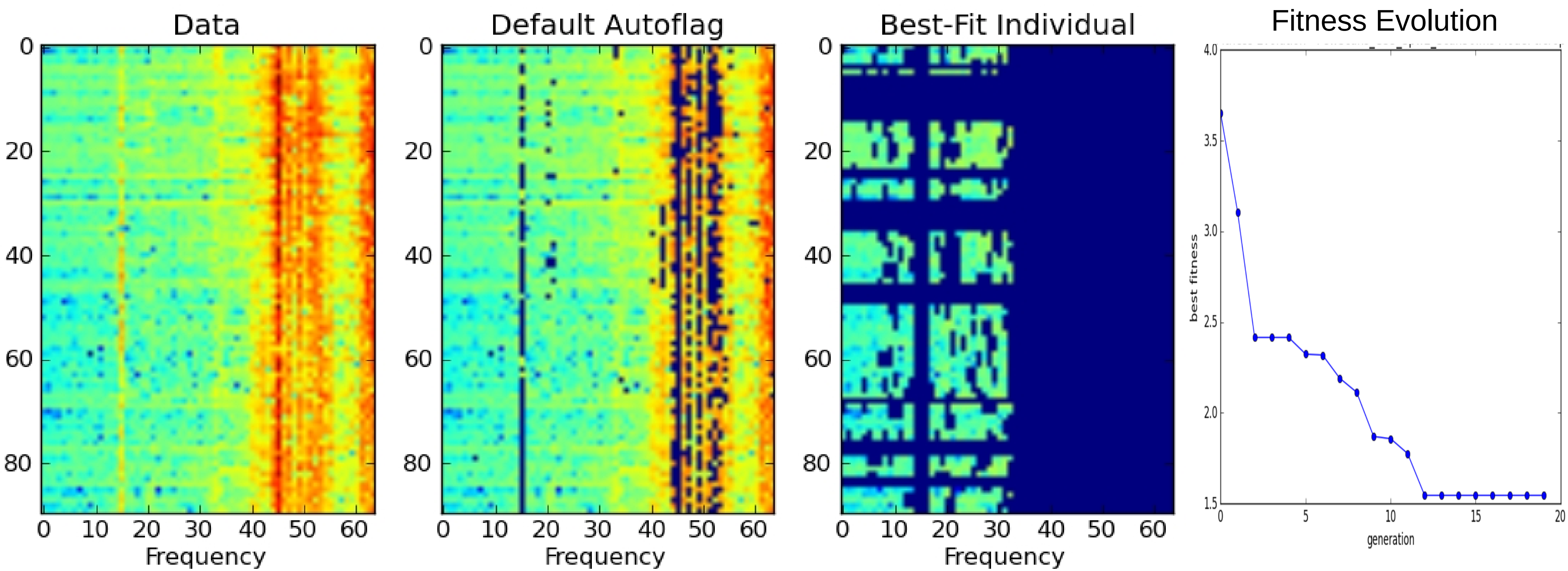
Results



Example : A mix of bright narrowband RFI and lower level broader RFI...

cmdlist=[" mode='tfcrop' timecutoff=5.0 freqcutoff=2.0 maxnpieces=3 usewindowstats='sum' ", " mode='extend' growtime=40.0 growfreq=90.0 flagneartime=True flagnearfreq=False growaround=False"]

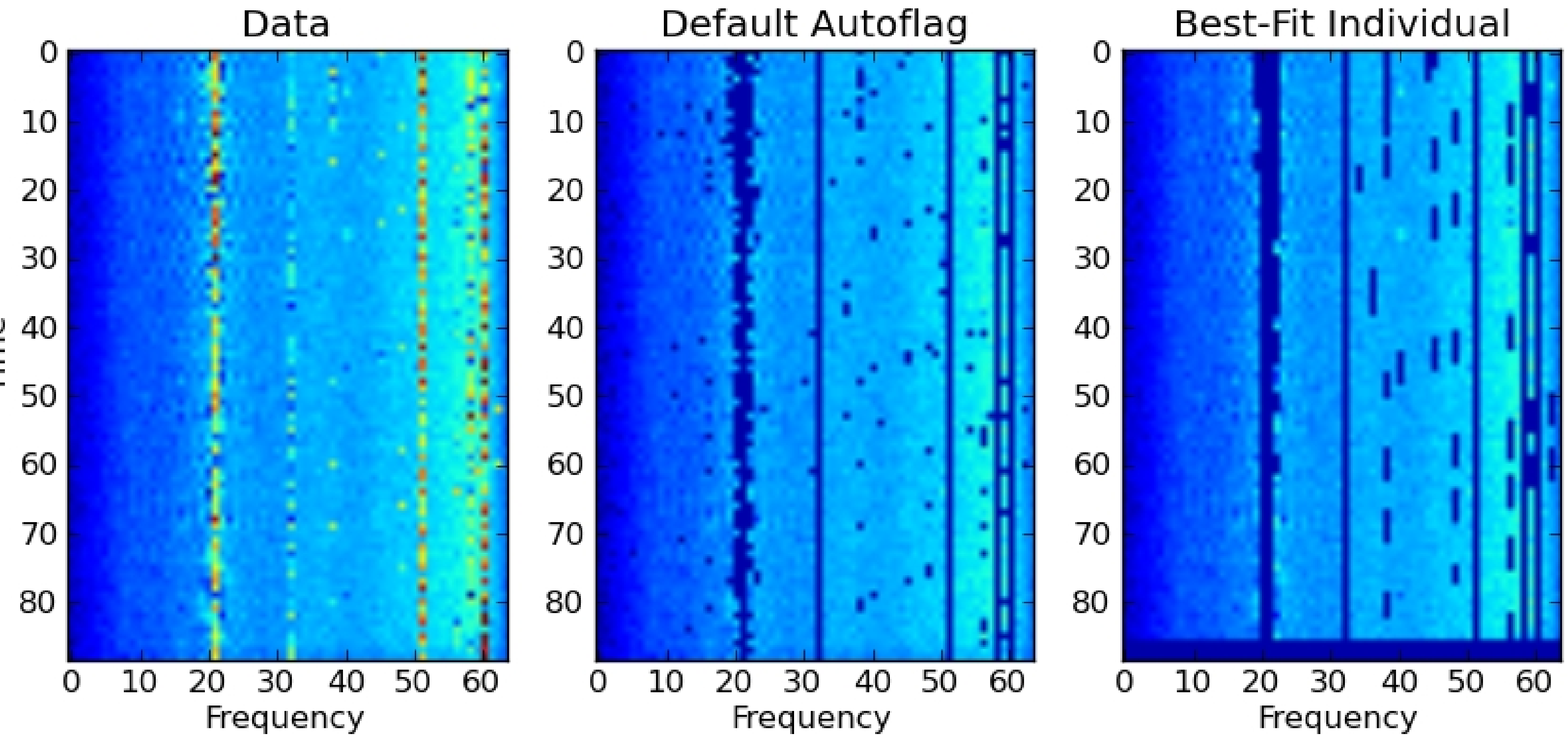
Results



Example : Most of the data are contaminated with RFI....

cmdlist=[" mode='tfcrop' timecutoff=1.5 freqcutoff=2.0 maxnpieces=7 usewindowstats='std'", " mode='extend' growtime=80.0 growfreq=70.0 flagneartime=True flagnearfreq=False growaround=False"]

Results

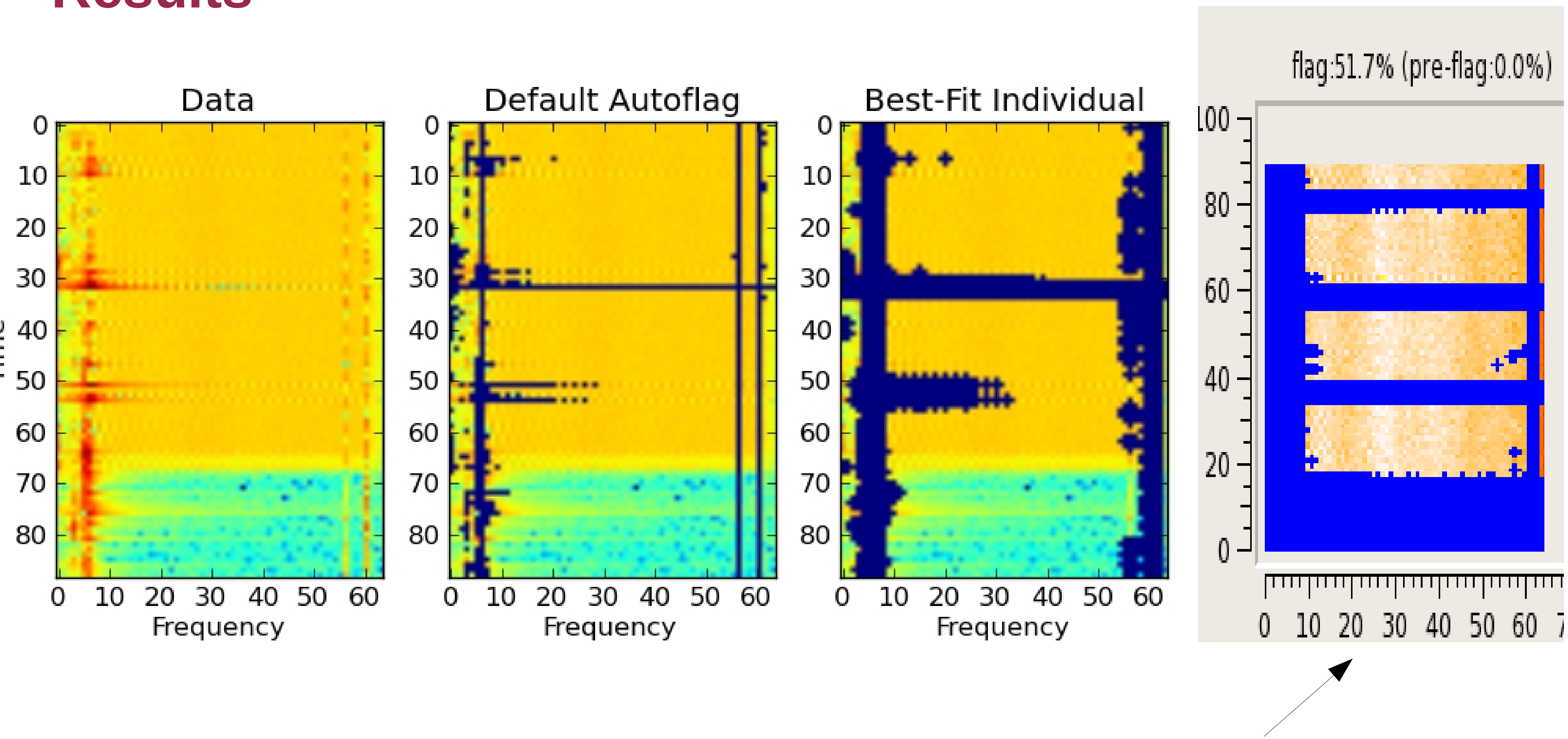


First generation produced adequate results

Example : “Easy” RFI that our algorithms are already tuned for...

cmdlist=[" mode='rflag' timedevscale=3.0 freqdevscale=5.0 winsize=3 ", " mode='extend' growtime=70.0 growfreq=40.0 flagneartime=False flagnearfreq=False growaround=False"]

Results



Example : Auto-tuning did not achieve results that matched hand-tuning
– Statistics didn't match what the fitness function was designed for ?

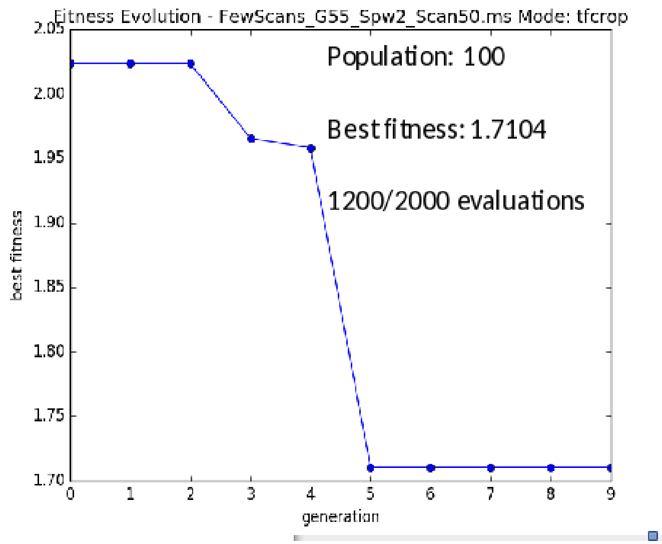
cmdlist = [" mode='tfcrop' timecutoff=4.0 freqcutoff=5.0 maxnpieces=1 usewindowstats='std'", " mode='extend' growtime=80.0 growfreq=70.0 flagneartime=True flagnearfreq=True growaround=False"]

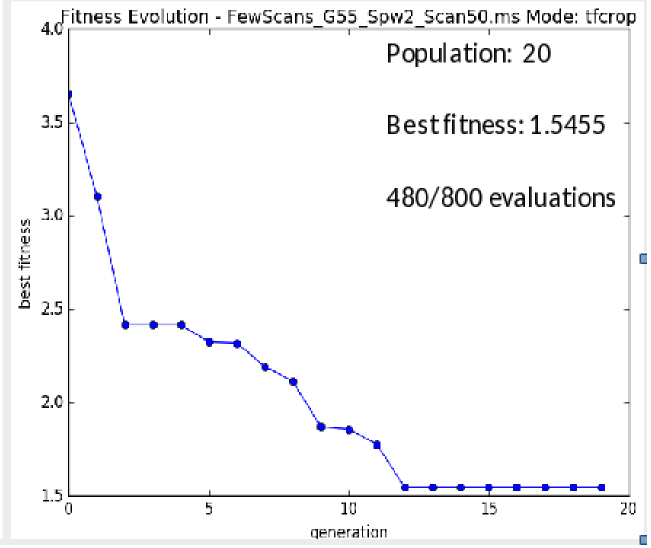
Tuning the tuner...

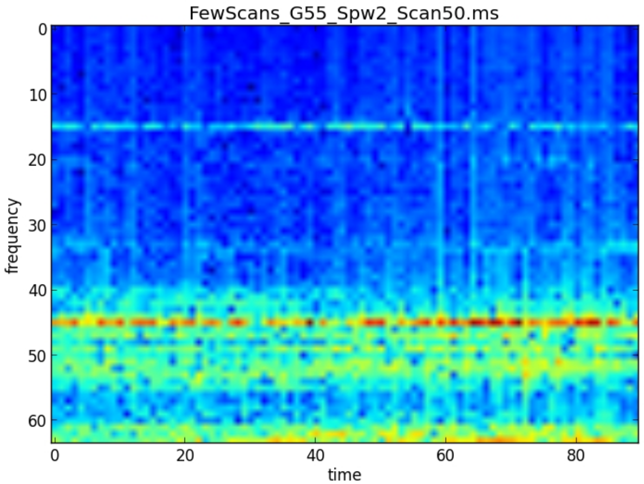
Initial population count, number of generations, dropout rate, mutation rate, etc...

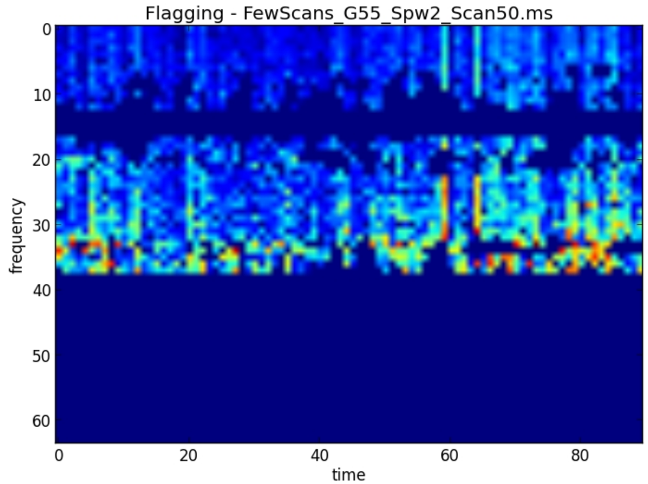
- Healthy convergence
- Minimum number of required evaluations
- Diversity vs Evolution
- Robust fitness function

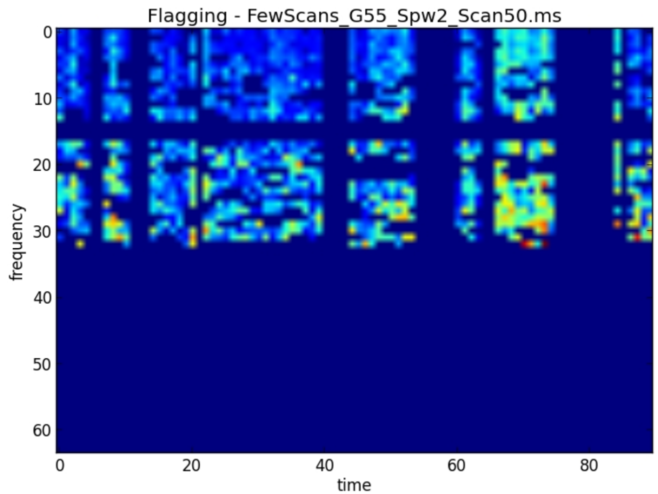
Might need tuning once per algorithm...
… but perhaps not.











Summary + Future Work

- Prototype looks promising. Can apply to any parameterized auto-flagger

- Algorithmic improvements :

  - Reproduction control, Improve population control (ageing), better termination criteria, better evaluation function

  - Include parameter evolution into the autoflag algorithms themselves

- Develop practical usage pattern

  - For each spectral window, tune on 1 scan, 1 ant-pair, 1 pol

  - A few thousand evaluations of autoflag (~10min. Easy to parallelize)

    …. Eventually, deploy on data reduction pipelines

Acknowledgements