EVLA Memo 161
The Effects of Automated RFI Mitigation on Source Intensity

Joe Helmboldt
Naval Research Laboratory
May 8 2012

1 Introduction

In this informal write-up, I would like to describe in some detail the work I've been doing on automated flagging and subtraction of RFI. I will outline each algorithm and have included some figures to illustrate some of the issues being discussed. I will start with my simple flagging routine and then move on to the subtraction routines which use Ramana Athreya's RFIfix algorithm as a starting point.

2 Flagging

The flagging algorithm I am currently using is a basic sigma-clipping routine that runs on visibilities binned by uv-radius. I have plans to alter it so that the data is also binned by azimuthal angle within each uv-radius bin, but for now, this routine seems to work well. I have it running in IDL and python. Here are the details:

1. The data are binned by uv-radius. These are not equally spaced bins, but are set up so that each bin has roughly the same number of data per bin. The default is to set up the bins so that for each channel and polarization, each bin has 1000 visibilities per bin. There is a free parameter that allows the user to change this, but the algorithm does not allow it to go below 10.

2. For each radial bin, the average amplitude is computed for each channel. For 10 or more channels, the half with the weakest amplitudes (i.e., all those below the median among all channels) are used for the sigma-clipping portion. If there are less than 10 channels, only the weakest channel is used; there is an option to do this even with more than 10 channels if the user wishes.

3. Once the channels to be used are identified, the real and imaginary parts of the visibilities are concatenated and the standard sigma-clipping is applied. The default is to do 10 iterations, each time clipping data beyond the mean plus or minus 2.576 times the standard deviation (i.e., the 99th percentile, assuming a Gaussian distribution). I stress that this is done on the visibilities and not the amplitudes, which is how I get away with assuming Gaussian distributions. The number of iterations and the number of "sigmas" used can be changed by the user. The final limits determined by the last iteration of the clipping portion are then applied to all the data in the radial bin (i.e., all channels) to flag.

To illustrate the performance of this algorithm, and for the remainder of this document, I will use the VLSS field 1330-253 (for my own selfish reasons; the field contains M83, for which I would

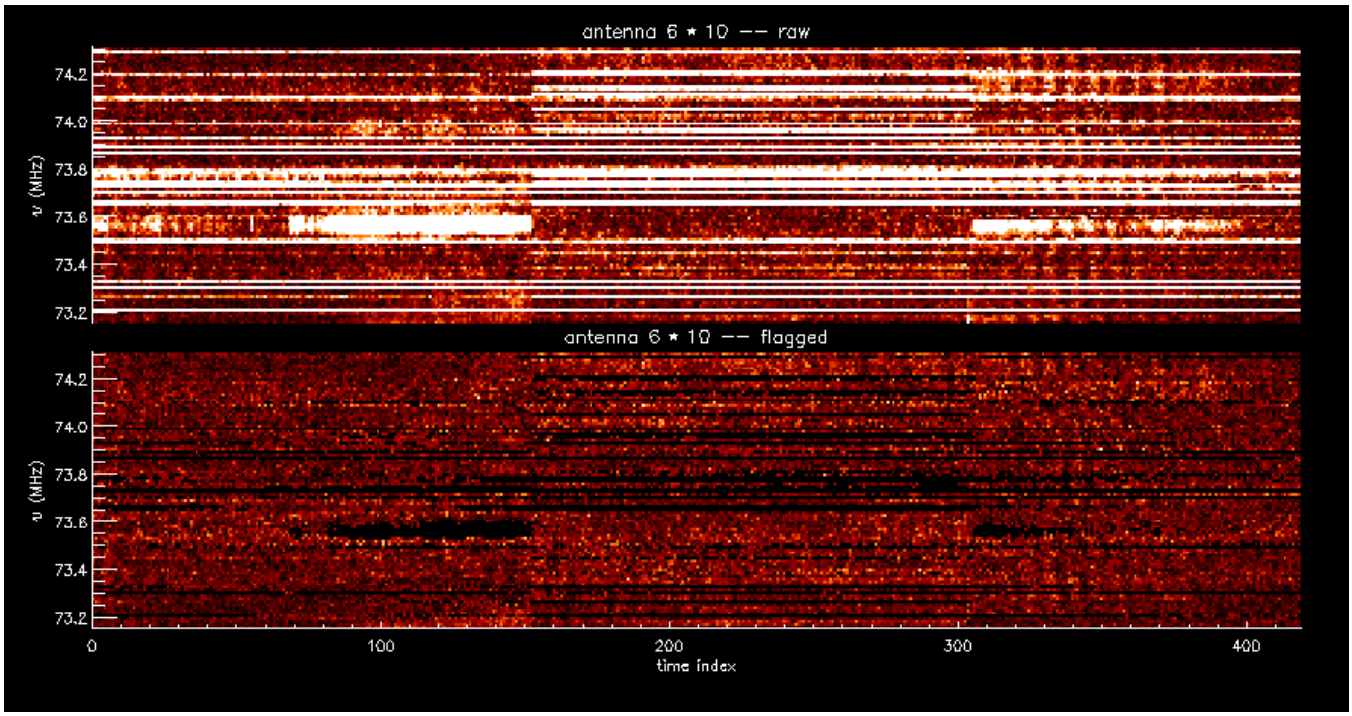

Figure 1: For the shortest baseline for the VLSS 1330-253 field residual data, images of the Stokes I amplitude with frequency on the y-axis and time index on the x-axis before (upper) and after (lower) automated flagging. The display stretch goes from 0 to 700 Jy.

like a low frequency image). I ran the flagging algorithm on this field, then ran IonImage on it within Obit. I then un-flagged the residual data produced by IonImage and will use that residual data for testing the rest of the algorithms described here. I re-ran the flagging routine on the residual data to help illustrate its effectiveness. In Fig. 1, I have displayed the Stokes I amplitudes before and after flagging for the shortest baseline for clipping at the 3-sigma level. As you can see, the algorithm flags most of the RFI you can see by eye, but some lower level RFI is probably missed.

3 RFI Model Subtraction

3.1 RFIfix

The first RFI subtraction routine I implemented was (essentially) Ramana's RFIfix. I based my implementation on our conversations in Charlottesville. While it may differ somewhat from Ramana's algorithm, it is based on the same assumptions that (1) the RFI will oscillate at the fringe rate of the field center and that (2) over a relatively short period of time, the combined source visibilities can be approximated by a constant. For sources that are relatively far from the center of the field of view (FOV), there are some problems with this, but I will expand on this later. So, the model fit to the data is

$$V = A + Be^{-2i\pi w} \tag{1}$$

The complex constant A approximates the source visibility and the complex constant B gives the amplitude and phase term for the RFI. Since there can be more that one source of RFI, even if the

phase terms of these sources are stable with time, the phase for the combined RFI signal can vary with time as can either the amplitude of the combined signal or the amplitudes of the individual RFI sources. Therefore, the data should be fit over a relatively short time interval so that the time dependence of B can be accounted for. My implementation of the algorithm is as follows:

1. For one baseline, channel, and polarization, I fit the model in equation (1) (which is a linear fit) to the visibilities as a function of w separately for each time step over the span of one RFI period, each time centering the model on that time step. If there are fewer than 3 time steps across one period, the fit is not done because the RFI is undersampled. If the total time covered by the data does not span more than 1/3 of a period, the fit is not done because the model will be too poorly constrained (I am fairly certain Ramana applies this same criterion). Think of Ramana's diagrams of the data as a circle in the real and imaginary plane centered at the real and imaginary components of the source visibility with a radius equal to the amplitude of the RFI. The center and size of the circle can't be constrained well enough if only a small portion of the circle is measured.

2. Once all the model fits are done, I go back to each time step and compare the χ^2 difference between it and the model to that of other time steps. For any of the other time steps which have a lower χ^2, its model fit is used to compute the model value at the current time step. If this produces a better match to the data, this fit is used instead. The purpose of this is to obtain better fits for time steps that are near the edges of bursts of RFI. For these time steps, the original fit will contain some time steps that are beyond the extent of the RFI burst. I believe this is also similar to something Ramana does for the same purpose.

3. After this is finished, for all time steps where subtracting the RFI portion of the model [i.e., $B\exp(-2i\pi w)$] lowers the rms/amplitude, the RFI model is subtracted. This criterion prevents the subtraction from making things worse either by increasing the noise or artificially increasing the mean (i.e., A may be way off from the true value). There may be problems with this criterion; I have written more on this below.

Fig. 2 shows a result from a test I ran on the residual data for 1330-253. I ran my implementation of RFIfix on the data, then ran my flagging routine to flag any remaining RFI and Fig. 2 shows the amplitudes for the shortest baseline for the RFIfix-subtracted data before and after flagging. Comparing this to Fig. 1, you can see that even for this RFI-plagued baseline, RFIfix gets rid of most of the RFI, and the flagging gets rid of the rest.

I have found two main problems with this approach. The first we have discussed in Charlottesville before. For a source that is not at the center of the FOV, the source will have a non-zero fringe rate that can be similar to the RFI fringe rate for a period of time. To illustrate this, I used UVSUB within AIPS to construct a model data set using the u, v, w coordinates of the 1330-253 field for a point source offset from the FOV center and ran RFIfix on it. In the upper left panel of Fig. 3, I have plotted the ratio of the visibility amplitudes after RFIfix to those before it was run as a function of the ratio of the simulated source's fringe rate to that of the RFI (see the Appendix for a description of how the source fringe rate is computed). You can see in this plot a systematic trend toward a source amplitude of zero starting at a source fringe rate of zero (and also at twice the RFI fringe rate), reaching zero amplitude when the source fringe rate equals that of the RFI. The data points also seem to cluster around a fringe rate ratio of -1, but this has more to do with uv-coverage. This can be seen in the upper right panel of Fig. 3 where I have plotted the fraction of

data whose amplitudes where changed by RFIfix within bins of fringe rate ratio. There is a definite peak at a fringe rate ratio of 1, but not one at -1. There is also a noticeable dip near zero because this is where the RFI fringe rate is relatively large (so the ratio is small) and RFIfix does nothing because there are two or fewer time steps across one RFI period (i.e., it is sampled at or below the Nyquist rate). You can also see that the source is more frequently affected when the baseline u is small by looking at the difference between the dirty image of the simulated source after RFIfix and the dirty image before RFIfix in the right panels of Fig. 4. The main result is a dark horizontal line in the difference between the two images (a narrow range in u translating to a wide range in l in the image plane).

The solution to this appears to be to use residual data, as Bill suggested. I have illustrated this by adding noise to the simulated data to "bury" it in the noise within the visibility data for each channel, but not so much that it cannot be detected within the image plane. After doing this, I reran RFIfix and have plotted in the lower panels of Fig. 3 the same things as in the upper panels of Fig. 3, this time for the simulated data plus noise. Clearly the prominent dip at a fringe rate ratio of unity has been eliminated.

The plots in Fig. 3 also illustrate the second problem. The simulated data has no RFI, so ideally RFIfix should do nothing to it, with or without noise. Subtracting an RFI model where there is no RFI is analogous to cleaning the noise during the image deconvolution process. Subtracting a model where there no source, if done often enough, can create a bias. Fig. 3 shows that this does happen quite often with RFIfix. This can be seen by examining the images in the right column of Fig. 4, dirty images made using the source-plus-noise simulated data before and after RFIfix. The difference between these two images shows a negative residual at the center. From Fig. 3, we can see that this is not the result of the algorithm mistaking the source for RFI, but from the general bias created by subtracting the RFI model where there is no RFI, producing a bias. I initially thought that this bias might be caused by my criterion that the subtraction of the RFI model only reduce the amplitude of the data. However, I tested versions of the code where I allowed the subtraction of the RFI model to increase the data amplitude by as much as a factor of two, and then by a larger factor. But, in both cases, I still got results similar to those seen in the right panels of Fig. 4. I therefore concluded that the problem really lies with allowing RFIfix to "run amok", so to speak, letting it subtract RFI where there is no RFI to subtract.

3.2 Modified RFIfix

To address the biassing problem, I have developed a modified version of RFIfix that attempts to only subtract RFI where it is actually present in the data. I also made a couple of other modifications to how RFIfix works. Here are the specifics:

1. For each polarization, baseline and channel, I break the data up into scans. For a single scan, I compute the discrete Fourier transform (DFT) of the visibilities as a function of w. In this DFT, RFI will show up as a large peak at a frequency of -1. I use the average w spacing for the scan to compute the expected spacing between the clones of this peak, Δf, that will appear as a comb-like function in frequency space that results from the DFT of the signal being convolved with the Fourier transform of the sampling function. I then compute the DFT for frequencies from $-1-\Delta f/4$ to $-1+\Delta f/4$ to make sure I do not run into the wings of the nearest clones of the potential RFI peak at $f=-1$. I then use the cumulative distribution of the amplitudes of the DFT for the first and last quarters of this frequency range to compute

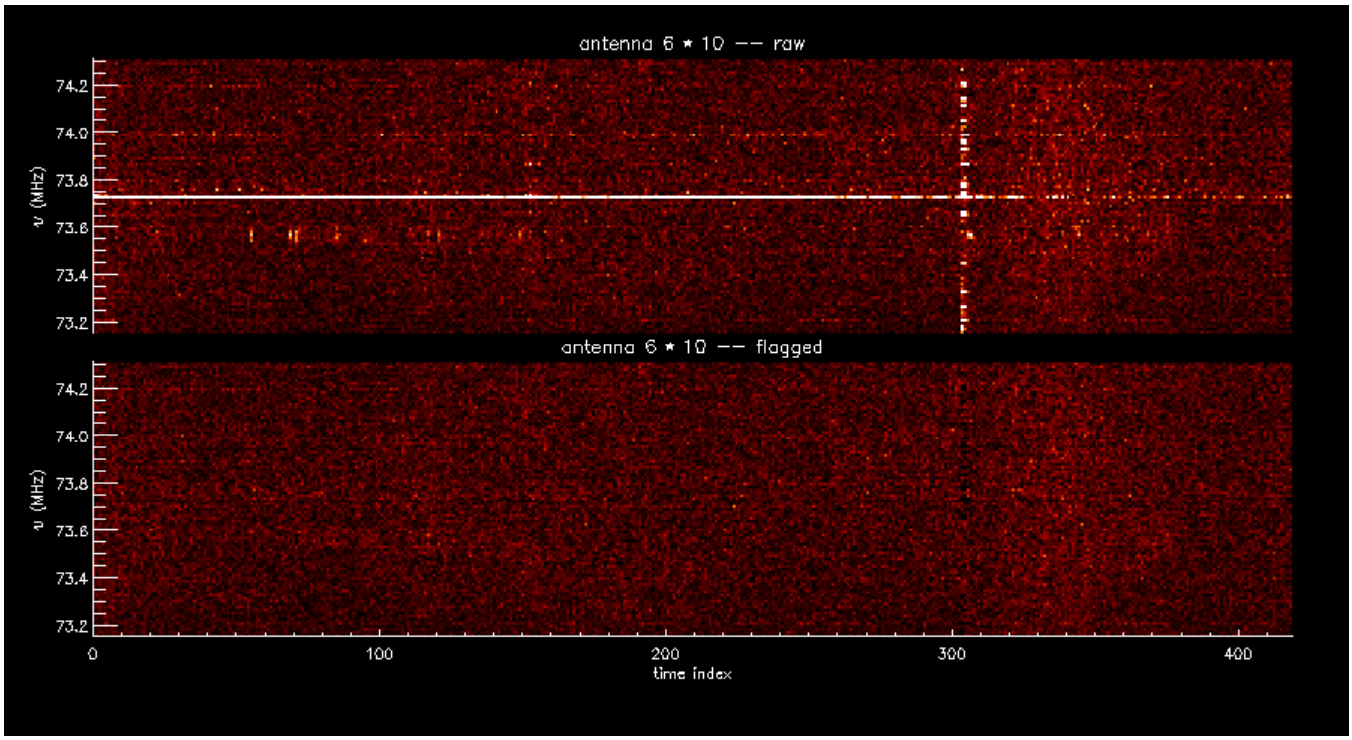

Figure 2: For the shortest baseline for the VLSS 1330-253 field residual data *after* RFIfix *was applied*, images of the Stokes I amplitude with frequency on the y-axis and time index on the x-axis before (upper) and after (lower) automated flagging. The display stretch goes from 0 to 700 Jy.

the 99th percentile for amplitudes that I assume are completely the result of noise (+ residual source flux). I also compute the 90th percentile to be used at a later step. I then locate the maximum amplitude near $f = -1$ (allowing for the fact that noise can change the actual location of the RFI peak from the expected location) and compare it to the 99th percentile threshold. If it is above this threshold, I proceed to the next step. If not, I do nothing to this scan and move on to the next one. Note that this may miss relatively strong but short bursts of RFI. However, these can be flagged relatively easily later and will not cause you to loose that much data. If you really want to subtract such bursts out, the time interval used can be changed from one scan to some arbitrary value of your choosing.

2. If RFI was detected using the above scheme, RFIfix is run on the scan with some modifications. First, given the previous results, I assume that this is being run on residual data in order to protect bright sources which still may be "detected" as RFI by my detection scheme and that any residual source flux is negligible when looking at single-channel visibilities. This allows me to adjust the model and only fit $B\exp(-2i\pi w)$, i.e., the mean visibility is assumed to be zero. This also allows me to throw out the requirements that (1) the RFI be Nyquist sampled or better and (2) at least 1/3 of a period needs to be used for the fit. For a single plane wave of a known frequency with no added constant(s), you need not sample it at intervals $< 1/(2f)$ to fit for its complex amplitude. For example, if you know its frequency, sampling the plane wave at intervals of $0.3/f$, $0.6/f$, and $0.9/f$ is identical to sampling at intervals of $0.3/f$, $1.6/f$, and $2.9/f$. The Nyquist theorem really only applies to a signal with a distribution of frequencies, not one single frequency. The requirement of fitting over at least 1/3 of a period can also be

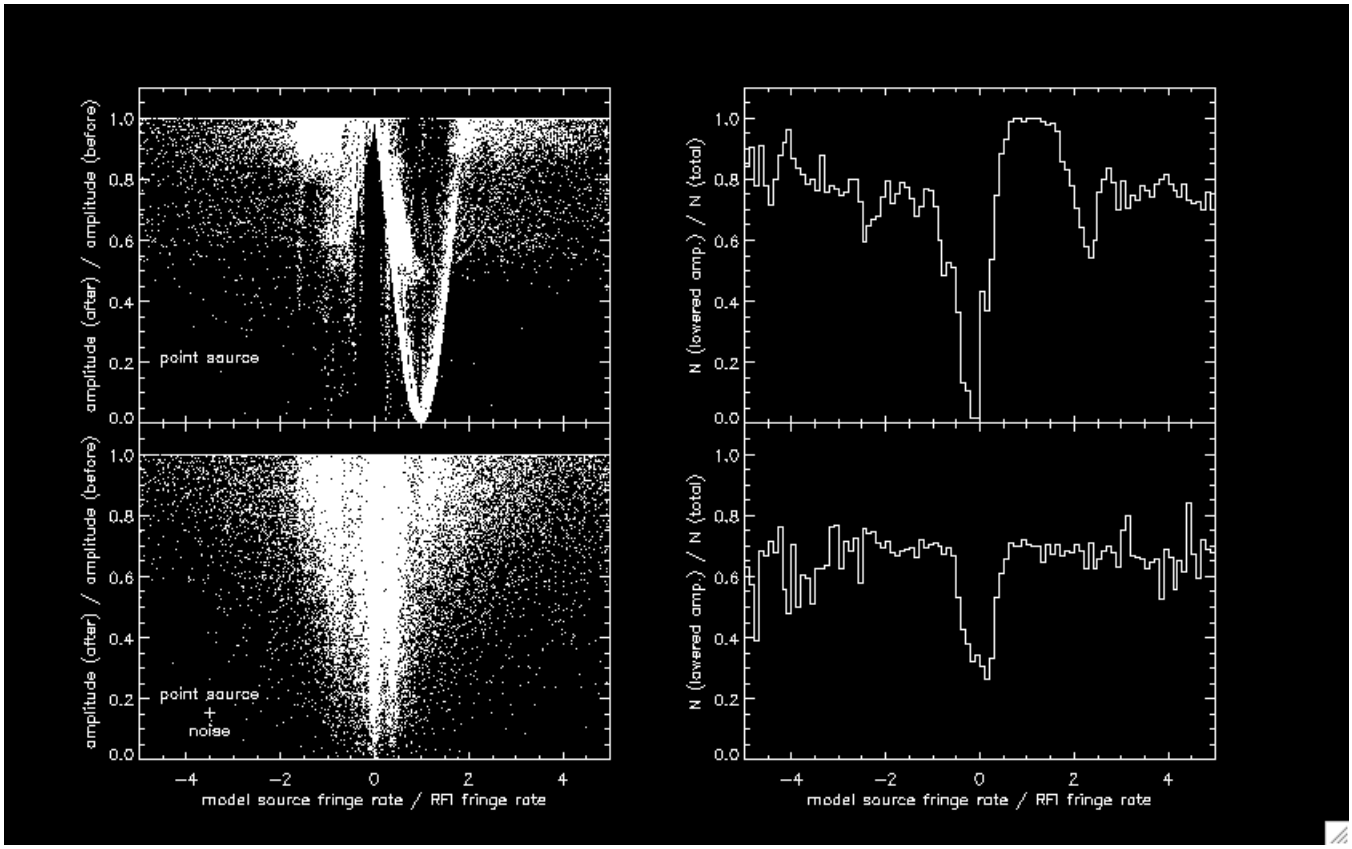

Figure 3: Upper left: for a single, off-center point source, the ratio of the visibility amplitude after RFIfix to the actual amplitude as a function of the ratio of the source's fringe rate to the RFI fringe rate. Upper right: the fraction of visibilities where RFIfix lowered the amplitude within bins of the ratio of the source's fringe rate to the RFI fringe rate. Lower panels: the same as the upper panels, but with Gaussian noise added to the simulated visibility data.

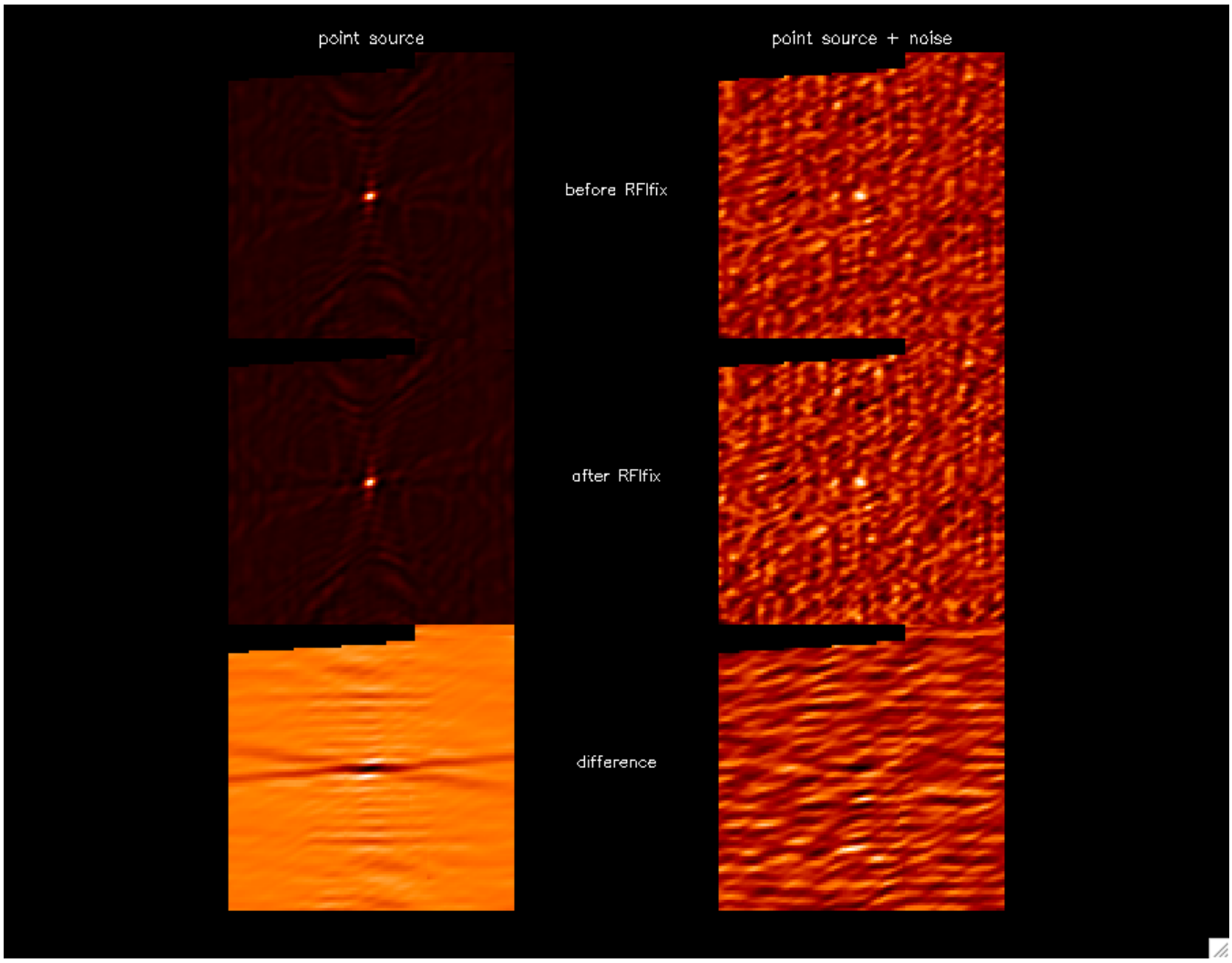

Figure 4: For the point source data presented in Fig. 3 (with noise on the right, without on the left), the dirty images before and after RFIfix and the difference between the two.

eliminated because the mean is set at zero. Think again of Ramana's representation of the RFI in the complex plane as a circle centered at the coordinates of the mean source visibility. If the mean is set at zero, the center of the circle is also set at the origin, and you do not need 1/3 of the circle to determine its radius. For this version of RFIfix, I do the fit for each time step using its visibility and those of the three closest time steps so that four time steps are used for each fit would should be enough to constrain well the two free parameters, the real and imaginary parts of the constant B. Following this, I perform step 2 of the description of RFIfix given above just as I did before to improve the fits near the boundaries of any bursts of RFI.

3. The next step is to determine where the RFI actually is within the scan. To do this, I first median smooth the visibility amplitudes so that I can identify the largest amplitudes with little influence from noise. Starting with the time step with the largest median-smoothed amplitude, I subtract the RFI model determined in step 2. I then compute the amplitude of the DFT computed at a single frequency, $f = -1$ and compare it to the 90th percentile threshold determined in step 1. If this amplitude is now below the 90th percentile threshold, I stop here. Otherwise, I proceed to the time step with the next largest median-smoothed amplitude, subtract its RFI model, compute the power at $f = -1$, and compare it to the 90th percentile threshold. I keep doing this until the power at $f = -1$ is below this threshold or until I run out of time steps; after this, the RFI model for any of the remaining time steps is set to zero.

4. The final step is to do a little smoothing of the RFI model. Since the fits are done over relatively short periods of time, the fitted values for B may be fairly noisy, and I would like to avoid artificially reducing the noise in the data. To do this, I basically median smooth the real and imaginary parts of B as a function of time for all time steps where the RFI model was not set to zero. I then multiply the smoothed values for B times $\exp(-2i\pi w)$. After this, I subtract this final RFI model from the original visibility data for the current scan and proceed to the next one.

In Fig. 5, I have displayed the amplitudes for the modified RFIfix-subtracted data for the shortest baseline for 1330-253 before and after flagging. If you compare this to Fig. 3, you can see that while it gets rid of a lot of visible RFI, it does an incomplete job with some of the stronger parts of the comb during the third and final scan. However, when I ran this on the simulated data that includes noise, it did nothing to the visibilities, as should be the case. To test this further, I added a 100 Jy point source to the residual data for 1330-253 at the same location as the simulated point source and ran the modified RFIfix. The resulting dirty images before and after are shown in the left panels of Fig. 6. From these you can see that with this bright source, we still have the problem of losing flux that we had before. However, I also redid this test using an added 1 Jy source in the same location. The resulting images are shown in the right panels of Fig. 6. From the difference between the images after modified RFIfix and before, there appears to be no evidence of source flux subtraction due to bias. This difference image also shows the telltale bands produced by RFI within images, implying that this version of RFIfix did a fairly decent job of removing RFI, but it may not have removed a lot of low-level RFI. In principle, the 99th and 90th percentile thresholds in steps 1 and 3 are fairly arbitrary and are based on some tests I did on a few baselines and channels. These could be adjusted to go deeper after weaker RFI, but you have to be careful not to go so deep as to produce a bias by subtracting RFI models from noise.

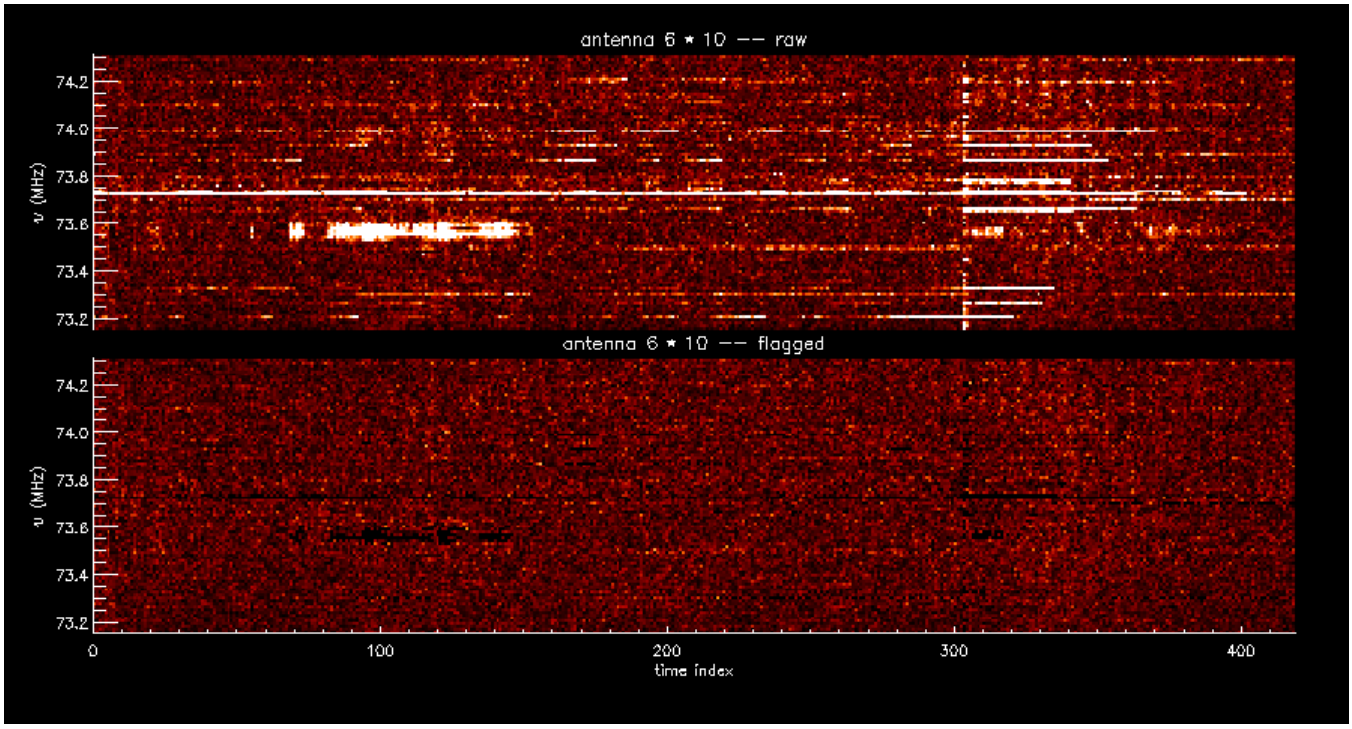

Figure 5: For the shortest baseline for the VLSS 1330-253 field residual data *after my modified* RFIfix *was applied*, images of the Stokes I amplitude with frequency on the y-axis and time index on the x-axis before (upper) and after (lower) automated flagging. The display stretch goes from 0 to 700 Jy.

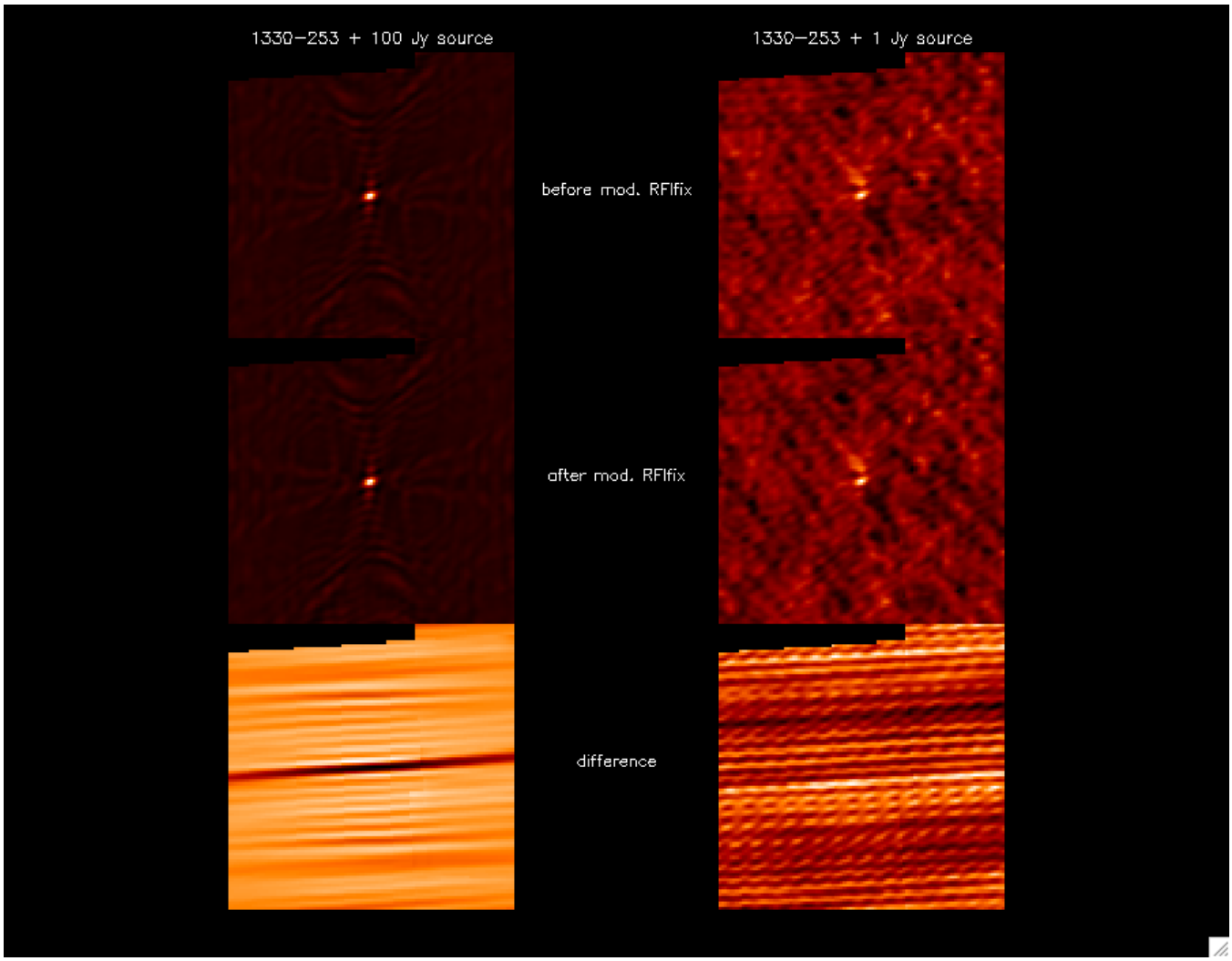

Figure 6: For a 100 Jy source (left) and a 1 Jy source (right) added to residual data for the VLSS field 1330-253 made using IonImage, the dirty images before and after my modified version of RFIfix and the difference between the two.

3.3 RFIsmooth: A Slightly Different Approach

After working on the modified version of RFIfix, I had an idea for a similar but different approach. For now, I am calling it RFIsmooth, which is not a great name, but is useful for now for comparisons with RFIfix. I am open to any suggestions for a new name. It is based on the same principal that the RFI can be modeled by a plane wave of the form $B\exp(-2i\pi w)$ and that we are using residual data where the source flux is negligible. The algorithm then proceeds as follows:

1. For one polarization, baseline, channel, and scan, instead of fitting the RFI model to the data, I first multiply the data by $\exp(2i\pi w)$. I then median smooth this "un-fringe-stopped" data with a filter of a given width in time (I have used a 2 minute wide filter for the VLSS data). In principle, the width of the filter should be roughly the width of time over which the complex amplitude of the RFI, B, is stable. In practice, it should be the smallest width that still gets rid of visible noise fluctuations. I am also assuming that over this chosen time window, the noise multiplied by $\exp(2i\pi w)$ averages out to zero. Mathematically speaking, I am not sure that this is completely valid, but in practice it seems to work. After median smoothing, I do a regular boxcar smooth to smooth out the rough edges a bit, and what we are left with is essentially mean values for B as a function of time.

2. Ideally, the next step would be to simple multiply these smoothed values for B times $\exp(-2i\pi w)$ and subtract this from the unaltered data. However, I found that doing this produces a bias, so I developed a thresholding scheme. First, I recognized that the smoothing that I am doing is quite similar to computing the Fourier transform at a single frequency ($f = -1$) over a short time period (here, 2 minutes). I therefore compute the DFT for the entire scan at frequencies within the same noise windows as I used in step 1 of the modified RFIfix described above. I then used the cumulative distribution of the amplitudes of this DFT to compute the 68th percentile of the noise amplitudes. I then multiplied this value by a scaling factor to correct for the fact that the boxcar smoothing and the DFT are slightly different operations (essentially differing by a factor of δw, the mean w spacing) and for the fact that the DFT was computed over the entire scan while the smoothing is done over a smaller time window (i.e., features will be sharper in the DFT over the entire scan than they will be over my two-minute window). I then use this as my threshold value. Any time steps where the amplitudes of the smoothed values for B are larger than this threshold have the RFI model subtracted from them; the other time steps are left alone. The choice of using the 68th percentile is arbitrary and based on some tests I did with a few baselines and channels. For 1330-253, this seemed to get rid of RFI while not producing a bias. You can change this value to go deeper or less deep depending on what the situation calls for.

This algorithm has the advantage of being a bit simpler than my modified version of RFIfix and runs much faster. It also seems to perform better as you can see by comparing Fig. 7 to Fig. 5 where you can see that much of the RFI missed in Fig. 5 is successfully subtracted out by RFIsmooth. RFIsmooth seems to do roughly as good a job removing RFI and RFIfix without the bias problem.

To illustrate this further and to provide potential users with a tool for checking for the presence of bias, I developed a relatively simple routine (in IDL) that adds a chosen number of point sources at a given flux level to the visibility data, randomly positioned with a circular FOV of specified width. The routine then flags the data and does a DFT for each source to compute the resulting peak intensity of the source in the image plane. I have been using 100 sources over a 15 degree

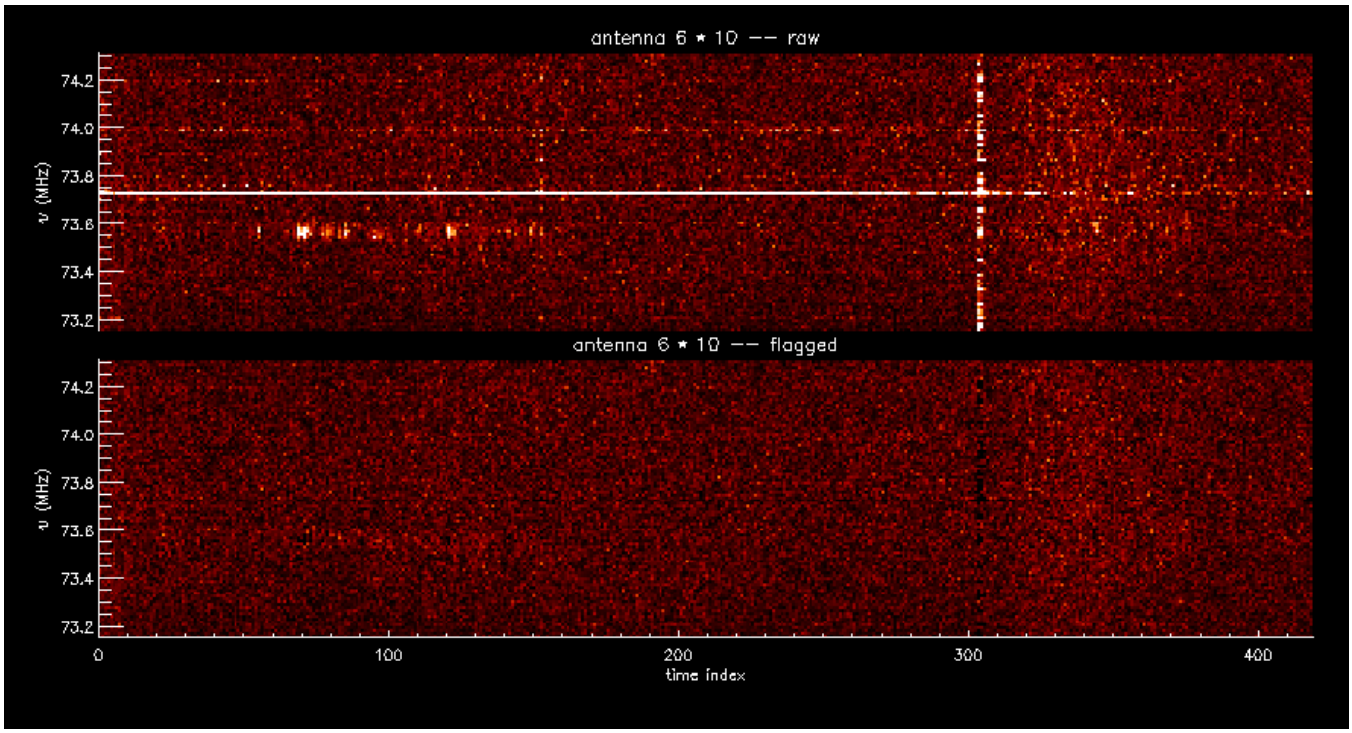

Figure 7: For the shortest baseline for the VLSS 1330-253 field residual data *after* RFIsmooth *was applied*, images of the Stokes I amplitude with frequency on the y-axis and time index on the x-axis before (upper) and after (lower) automated flagging. The display stretch goes from 0 to 700 Jy.

FOV, so this series of DFTs actually takes much less time than making a dirty image of the entire FOV within AIPS. The routine then un-flags the data, runs your choice of subtraction routine, does some flagging, then recomputes the intensities. You can then compare the intensities before and after subtraction. I show such a comparison using 100 randomly placed 1 Jy point sources in Fig. 8 where I have plotted the intensities of the simulated sources after subtraction versus those before subtraction for RFIfix (upper left), my modified RFIfix (upper right), and RFIsmooth (lower left). I have also plotted a comparison of the modified RFIfix to RFIsmooth in the lower right panel. The bias from RFIfix is fairly obvious. There also seems to be a little bit of a bias for the modified RFIfix, but it seems quite small and could be eliminated by using a higher threshold in step 3 in §3.2. RFIsmooth produces no apparent bias. These plots also illustrate the way in which you can choose your threshold level for something like RFIsmooth. Basically, you should use the lowest threshold that does not produce a bias. My choice of using the 68th percentile threshold worked for this VLSS field, but it may not be appropriate for all data sets.

4 Discussion

After all this, for the 1330-253 field, the subtraction routines did not really help out that much. None of them really improved the source detection rate beyond what could be done with just flagging. RFIfix did reduce the rms, but that is only because of the bias it produces which lowers the intensity scale of the entire image; I actually detected fewer sources with SAD within AIPS after running RFIfix. However, after looking at the results a little more, only 6% of the visibilities

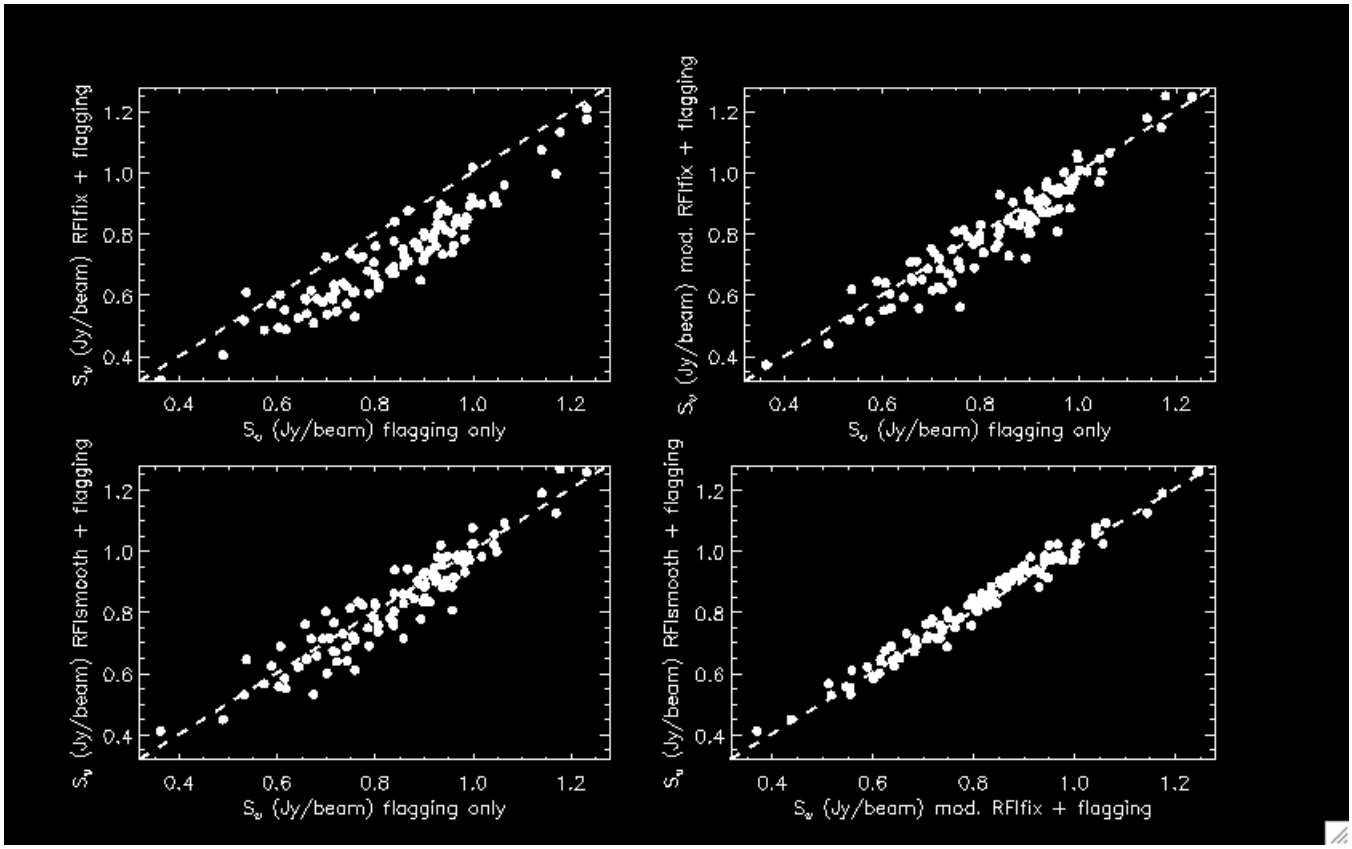

Figure 8: For 100 randomly placed 1 Jy sources added to the residual data for 1330-253, the measured peak intensities (no cleaning) for RFIfix plus automated flagging versus just automated flagging (upper left), my modified RFIfix plus automated flagging versus just automated flagging (upper right), RFIsmooth plus automated flagging versus just automated flagging (lower left), and RFIsmooth plus automated flagging versus my modified RFIfix plus automated flagging.

were flagged when no subtraction was done, which is not a huge fraction. All of the subtraction routines reduced the fraction of flagged visibilities down to about 3%. I think it is fairly impressive that they are able to "rescue" about half of the flagged data. However, going from 6% flagged to 3% flagged does not do much to improve the image rms. And, any low-level RFI that was removed did not make a whole lot of difference either. Therefore, I think these subtraction routines will be much more valuable for data that has a lot more RFI and/or has a substantial amount of low-level RFI. But, when done carefully, it does not hurt to run the subtraction.

I also discovered that the subtraction routines really help to rescue the shorter baselines as you might have inferred from Fig. 1, 2, 5, and 7. In Fig. 9, for each polarization, I plotted the fraction of visibilities flagged within bins of *uv*-radius when no RFI subtraction was done along with the same for the data after each of the three subtraction routines was run. As expected, the fraction of flagged data is much higher at the shortest spacings when no subtraction is done, reaching as high as 15%. However, when any of the subtraction routines is applied first, this trend virtually disappears and the fraction of flagged data stays roughly flat over all *uv*-radii. This implies that the RFI subtraction routines are especially useful for the shortest spacings which should be of great interest to anyone interested in extended emission from things like galaxy disks or cluster halos/relics. This may also prevent you from having to virtually throw away all the the shortest baselines when the RFI is fairly bad (which is maybe more relevant for GMRT). At the moment, my favorite scheme is RFIsmooth since it appears to perform well and runs significantly faster than either RFIfix routines. My plans for the near future are to (1) talk to Wendy and possibly Aaron to see if there are any VLSS fields where the RFI was particularly terrible where RFIsmooth may be a much bigger help, and (2) run some simulations to see how much sensitivity you can gain for extended sources by running RFIsmooth on some RFI-plagued data where the shortest spacings are virtually unusable. Any comments and suggestions on improvements or other tests you would like to see would be greatly appreciated.

5 Appendix: Fringe Rates

For a point source at some position in the sky with direction cosines l, m, and n has a fringe-stopped visibility at a given frequency, ν, of

$$V_\nu = F_\nu e^{-2\pi i[lu+mv+(n-1)w]} \tag{2}$$

where F_ν is the source's total flux density and u, v, and w are the baseline coordinates defined as

$$u = \sin(h)X+ \quad \cos(h)Y \tag{3}$$

$$v = -\sin(\delta)\cos(h)X+ \quad \sin(\delta)\sin(h)Y+ \quad \cos(\delta)Z \tag{4}$$

$$w = \cos(\delta)\cos(h)X- \quad \cos(\delta)\sin(h)Y+ \quad \sin(\delta)Z \tag{5}$$

where X, Y, and Z represent the antennas' physical positions (in units of wavelengths) and h and δ are the hour angle and declination of the field center, respectively. We can also express the visibility of the source as a function of time as

$$V_\nu = F_\nu e^{2\pi i(\nu_f t+\phi)} \tag{6}$$

where ν_f is the fringe rate of the source, and ϕ is a generic phase term. Equations (1) and (5) show that we can relate the fringe rate to the baseline coordinates and the source position via the time

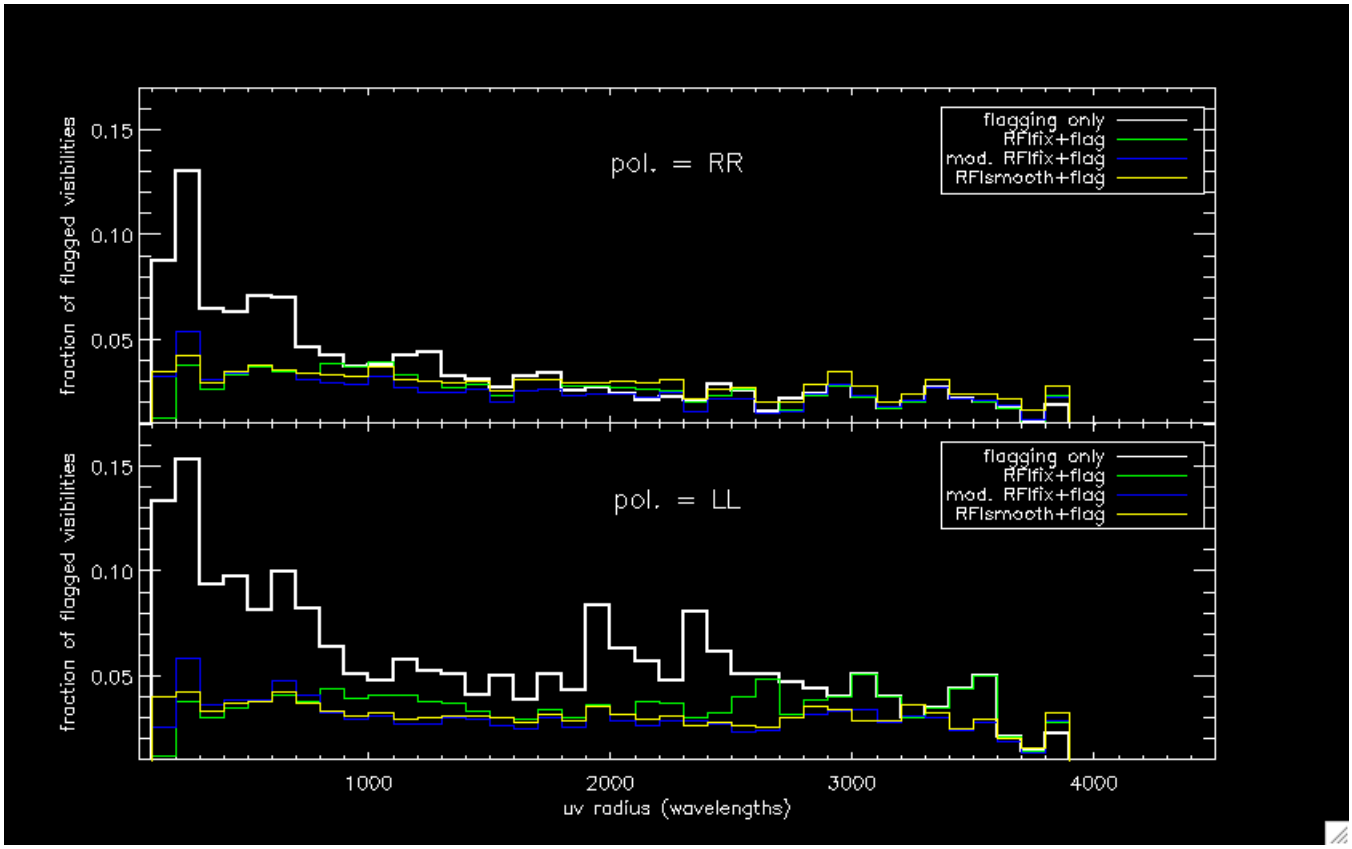

Figure 9: For RR (upper) and LL (lower) polarizations, the fraction of data flagged by automated flagging with no RFI subtraction (white), after RFIfix (green), after my modified RFIfix (blue), and after RFIsmooth (yellow).

derivatives of the baseline coordinates

$$\nu_f = -[l\dot{u} + m\dot{v} + (n-1)\dot{w}] \tag{7}$$

where the dot, as usual, denotes a time derivative. Using the fact that the time derivatives of X, Y, Z, and δ in equations (2), (3), and (4) are all zero and that the time derivative of the hour angle, h, is the Earth's rotation rate, $\omega_\oplus$, we can solve for the following

$$\dot{u} = \omega_\oplus[\cos(\delta)w - \sin(\delta)v] \tag{8}$$

$$\dot{v} = \omega_\oplus \sin(\delta)u \tag{9}$$

$$\dot{w} = -\omega_\oplus \cos(\delta)u \tag{10}$$

Non-moving RFI has a fringe rate equal to the non-fringe stopped fringe rate of the field center $\nu_{RFI} = \dot{w} = -\omega_\oplus \cos(\delta)u$. Therefore we can express the ratio of the source's fringe rate to that of the RFI as

$$\frac{\nu_f}{\nu_{RFI}} = \left[\frac{w}{u} - \frac{v}{u}\tan(\delta)\right] l + \tan(\delta)m - (n-1) \tag{11}$$

which makes it useful to define a quantity, $\mu = [w - \tan(\delta)v]/u$. Finally, if you establish a field-of view with an angular radius of θ, you can see that the fringe rate will vary with azimuthal angle, φ, around the circular boundary of the FOV by expanding the direction cosines l, m, and n in terms of θ and φ according to

$$\frac{\nu_f}{\nu_{RFI}} = \sin(\theta)\cos(\varphi)\mu + \sin(\theta)\sin(\varphi)\tan(\delta) - [\cos(\theta) - 1] \tag{12}$$

We can then use this equation to determine the value of φ where the fringe rate reaches its extrema $= \tan^{-1}[\tan(\delta)/\mu]$. Plugging this value for φ into equation (11) then gives you the maximum(minimum) fringe rate for a given field of view for a particular baseline at a particular time (i.e., at a given set of u, v, w coordinates).