L Special Considerations for EVLA P-band Data Processing in *AIPS*

The EVLA has been equipped with new, wide-band receivers in the range 230 MHz to 470 MHz, known as P-band. Most of the data reduction and imaging at this band are similar to those at other bands, so the following will assume some degree of familiarity with the rest of the *CookBook*, particularly Chapter 4. This guide was initially prepared by Minnie Mao from an earlier guide written by Susan Neff. Enough has changed with the P-band system that a thorough revision of the Appendix has been done.

Note that the choice of file names, disk assignments, catalog numbers, and the like depends on your circumstances and will almost certainly not be the same as those shown in the examples below. Note also that experienced (but still reasonable) EVLA users disagree on the details of data reduction. Some compromise in these details has been attempted below.

RFI, especially in spectral-line observations, has tended to accumulate at the phase center for reasons not well understood. It may be best to observe your sources somewhat offset from the phase center.

L.1 P-band calibration and editing in *AIPS*

1. Download your data set from the archive found at
   `https://data.nrao.edu/`
   making sure that you select SDM-BDF data set (all files) as the download data format.

2. Before starting AIPS and while in the terminal window you intend to use for AIPS, enter

> cd *my_data_directory* CR	to change to the data directory containing the top-level directory of your data set.
> export MYAREA=‘pwd‘ CR	for bash shells, or
> setenv MYAREA ‘pwd‘ CR	for tcsh and c shells.

   Note the back ticks around `pwd` which cause the output of `pwd` to appear in the typed line.

3. Start AIPS with, for example,

> aips tv=local pr=3 CR	to use Unix sockets for the TV and printer 3

   and enter your user number when prompted.

4. Look at the SDM-BDF file contents with BDFLIST — make sure you do *not* have a quote at the end of the file name, or else the lower case characters get changed into upper case and the file won't be found.

> default bdflist CR	initialize all adverbs
> dowait 1; docrt 1 CR	to get output on terminal as well as log file.
> asdmfile(1) = ’MYAREA: CR	two parts are concatenated.
> asdmfile(2) = ’TSUB0001.sb23643640.eb23666175.56454.14034020834 CR	
> bdflist CR	to examine file contents.

   Pay particular attention to the configuration numbers and number of spectral channels. These refer to each unique correlator *and* antenna setup, not just the size of the current VLA. Each configuration and number of channels that you want will need to be loaded individually.

5. Load data into *AIPS* with `BDF2AIPS` — We put everything on disk d because that's where there's more room. Note, do not use `DEFAULT` if you wish to retain the adverbs from the previous step. Loading data may take a while depending on the size of your data.

> outn 'my-name' C_R	to select a meaningful file name.
> config c C_R	to select configuration c.
> outdi d C_R	to select the disk.
> bdf2aips C_R	to translate the data into *AIPS*.
> nchan N C_R	to limit to data with N spectral channels; some spectral-line configurations can contain more than one number of channels but `AIPS` can only load one number at a time.

Do this as often as needed to load all desired configurations. Spectral-line set-ups can have spectral windows with different numbers of channels within the same configuration. *AIPS* can only handle one at a time, so specify `nchan` N C_R to load the data with N channels and repeat as needed.

6. Look at header of the *AIPS* file written by `BDF2AIPS`

> indi d; getn 1 C_R	to select disk and catalog number, assuming you had no files on disk d previously.
> imh C_R	to see the details of the uv-data set header.

7. look at where the telescopes were during your observation with `PRTAN`

> default prtan C_R	
> indi d; getn 1 C_R	
> docrt = -1 C_R	to make a printer listing to keep at your terminal.
> go prtan C_R	

8. Look at the observation summary with `LISTR`

> default listr C_R	
> indi d; getn 1 C_R	
> opty 'scan' C_R	to list scans and sources.
> docrt = -1 C_R	to make a printer listing to keep at your terminal.
> go C_R	

At this stage, Minnie's summary recommends separating P-band from 4-band data. At present, only P-band is recorded so that step is not needed. Then, Minnie recommends using `POSSM` to find dead antennas plus those with crossed polarization. Then she recommends `UVFLG` to flag the dead ones and `FIXRL` to uncross the mis-wired ones. These problems have been essentially eliminated. In addition, P-band data are now delivered by default in a number of relatively narrow spectral windows (called IFs in *AIPS*) which helps minimize data lost to RFI and eliminates the `NOIFS` and `MORIF` steps from Minnie's guide.

9. However, the requantizer gains are now allowed to vary and must be corrected.

> default tyapl C_R	to select the needed task
> indi d; getn 1 C_R	to select the input file.
> optype 'PGN' C_R	to select requantizer gain correction.
> inext 'SY' ; in2ver = 1 C_R	to apply the SysPower table, the version number is not defaulted.
> outdis d ; go C_R	to stay on one disk.

10. Narrow-band RFI will normally cause ringing in P-band data. Use `POSSM` to examine some baselines from a calibration source to check this if you want to be careful. The default setup delivers data from outer IFs for which the sensitivity of the P-band receiver is poor. Use SPLAT to drop the insensitive IFs (those centered on 232, 248, 264, and 472 MHz) and to do Hanning smoothing to reduce the ringing.

> default splat C_R	to select the needed task
> indi *d*; getn 2 C_R	to select the (new) input file.
> flagver 1 C_R	to apply the on-line flags.
> smooth = 1, 0 C_R	to do Hanning smoothing once and for all.
> bif = 4; eif = 15 C_R	to select the more sensitive IFs.
> outdis *d* ; go C_R	to put the output on the same disk.

11. There is also a Navy satellite that makes a mess of data from 360 MHz to 380 MHz. Delete this with `UVFLG` in one of 2 ways. For the standard setup, you can specify the channels as

> default uvflg C_R
> indi *d*; getn 3 C_R
> bif 6 ; eif 6; bchan 65; echan 128; C_R
> go C_R
> bif 7 ; eif 7; bchan 1; echan 97; C_R
> go C_R

For any setup, you may make a text file containing the line

```
BFREQ = 360 EFREQ = 380 /
```

and then use it with `UVFLG` as

> default uvflg C_R	
> indi *d*; getn 3 C_R	
> intext = '*MYWORK:pband.flg* C_R	to give file name
> go C_R	

12. Edit UV data with `EDITA` and the `SY` table. *The* `SY` *table* P_{sum} *parameter is sensitive to the occasional bursts of RFI which render the data useless. Flag all seriously discrepant points but do not flag really tightly since noise and even abnormalities in this parameter are "normal."*

> default edita C_R
> indi *d*; getn 3 C_R
> inext 'SY' C_R
> go C_R

13. At this time, it is best to identify those spectral channels affected by RFI throughout the run and to flag them once and for all. `FTFLG` will display the data combining all included baselines in a single display (per polarization), while the very similar `SPFLG` allows much better discrimination (and more work) by displaying the baselines separately. Be sure to flag all sources if you flag channels over all times.

> default ftflg C_R	
> indi *d*; getn 3 C_R	
> calcode '*' C_R	to examine calibrators only
> dparm(6) = x C_R	to specify the basic integration time in the data in seconds.
> go C_R	

If you use `SPFLG`, you should limit the times, sources, and/or baselines examined. The flagging then is best done with `UVFLG` from notes made while you study the data with `SPFLG`.

> default uvflg CR	
> indi d; getn 3 CR	
> bif = bi ; eif = ei CR	to set range of spectral windows.
> bchan = bc ; echan = ec CR	to set range of spectral channels.
> antennas = $a1, a2, a3$ CR	to select antennas to flag.
> baseline = $b1, b2, b3, b4$ CR	to limit flags to the baselines from the pairs $a1$ through $a3$ with $b1$ through $b4$.
> reason 'bad channels' CR	to set a reason in the flag table.
> go CR	and repeat as needed with different IFs, channels, antennas, etc.

14. This step is optional and should be considered only if you have several P-band data sets to reduce. There is a special capability in *AIPS* developed for lowband data. The task `FGTAB` was written to read a flag table and write to a text file those flags that apply to all times, sources, antennas, subarrays, and frequency IDs. The channel specifications are written in frequency units which enables `UVFLG` to read the text file and apply the same flags to the channels to which they pertain in the second data set. Of course, if the two data sets have identical frequency structure, then one could simply copy the `FG` table from one data set to the other. Task `TACOP` would be used if you want the flag file by itself. `TAPPE` would be used if you want to append the flag table of the first file to one of the second. Note that these two tasks transfer all flags in the input table, not just those that flag the general RFI environment.

15. Calibrate for the delay errors with `FRING`. Choose a time range in the middle of the observation where you know the antennas have settled down. A short time range is all you need — look at `LISTR` — because you don't want time variations to come into play. This is also why you should choose something bright — like 3C286, which is used here as the name for the generic fringe- and bandpass-fitting calibration source. It would be wise to check these data for RFI with `POSSM` before running `FRING`.

> default fring CR	
> indi d; getn 3 CR	
> calso '3C286' ' CR	to select the primary calibration source.
> timerang 0 3 40 0 0 3 40 55 CR	to select $<$ 1 minute of good data from the calibration source.
> solint 1 CR	to average over all included times.
> aparm(5) 1 ; aparm(6) 1 CR	to fit a single delay over all IFs and to print some extra detail in the fitting.
> dparm(9) 1 CR	to avoid any fitting for rates — important.
> refant 22 CR	to set a reference antenna, if one is known to be better than others.
> go CR	

This print level will display the delays found, which should all be a few nano-seconds although larger (correct) delays sometimes arise.

16. Apply these delay solutions to the calibration (`CL`) table with `CLCAL`

> default clcal CR	
> indi d; getn 3 CR	
> calso '3C286',' ' CR	
> snver 1 CR	to specify the `SN` table version written by `FRING`.
> go CR	to copy `CL` table 1 to version 2 adding the delay correction, taken to be constant in time.

17. You should flag out any bad data on your bandpass calibrator. Some users feel that a quick-and-dirty flagging is enough at this stage. Use `CLIP` with a carefully chosen, perhaps IF-dependent, cutoff level. A more detailed editing is done by other users. Here are options to use `RFLAG` and `SPFLG`. Maybe you'd like to `RFLAG` first and then eyeball with `SPFLG`. The options are endless! The first execution of `RFLAG` determines flagging level parameters and makes plots on the TV for you to be sure that they are reasonable, The second execution applies these levels and determines new levels in case you want to run `RFLAG` even more times.

> default rflag C_R	
> indi d; getn 3 C_R	
> sources(1) '3C286' C_R	
> docal 1 C_R	to correct delays and a-priori gains.
> avgchan 11 C_R	to set the width of the median-window filter run across the spectral channels in each IF
> fparm = 3, x, -1, -1 C_R	to do a small rolling time buffer, set the sample interval, and to have `NOISE` and `SCUTOFF` set IF-dependent cutoff levels.
> fparm(13) = 1000 C_R	to clip amplitudes at 1000 Jy.
> doplot 12 ; dotv 1 C_R	to do the 2 most useful plots on the TV.
> go C_R	

`RFLAG` will return cutoff levels to the appropriate adverbs. Therefore, to apply the flags using the flagging levels found above, do *not* do a `tget rflag` which will reset the levels to zero. Instead, simply do

> doplot = -12 C_R	to apply the cutoff levels and determine new ones, plotting on the TV.
> go C_R	

`RFLAG` will make a new flag table each time it determines flags (when `DOPLOT` $\leq$ 0). `RFLAG` has numerous `FPARM` options you may choose. `FPARM(9)` and `FPARM(10)` scale the rms for `NOISE` and `SCUTOFF` that `RFLAG` computes. If the fraction of a channel that is flagged exceed `FPARM(7)`, the entire channel is flagged. `FPARM(6)` is especially useful at P-band as it allows channels adjacent to an RFI peak to be flagged. Similarly, `FPARM(8)` allows channels between strong RFI peaks to be flagged. Read the help file for detailed information. Flagging calibrator scans aggressively is recommended especially for spectral-line observations.

Check the results with

> default spflg C_R	
> indi d; getn 3 C_R	
> sources(1) '3C286' C_R	to examine only the bandpass calibrator.
> dparm(6) = x C_R	to set sampling time in seconds
> docal 1 C_R	to correct delays and a-priori gains.
> go C_R	

It is at this point that experienced users differ significantly in their approach. If you have only one bandpass-calibration scan and the ionosphere was reasonably behaved (phases nearly constant through that scan), then you should run `BPASS` on the calibration source in place. However, when one has multiple bandpass scans and/or a badly behaved ionosphere, it may be best to split the calibration scans out and self-calibrate them on a short time interval. The addition of multiple scans where the source has been phased up, but the remaining RFI has not, will reduce the effect of that RFI substantially. This scheme is described next.

18. Split off your calibrator for bandpass calibration — this is only necessary for self-calibrating the BP calibrator.

> default split CR	
> indi d; getn 3 CR	
> outdi indi ; source '3C286' ' CR	to select calibrator and stay on the same disk.
> outcl 'bpspl' CR	to indicate the use for this file.
> docalib 1 CR	to apply delay and a-priori calibration.
> go CR	To make a single-source file.

19. Self-cal on your bandpass cal with `CALIB` to solve for wiggles in phase as a function of time.

> default calib CR	
> indi d; getn 4 CR	
> refant 22 CR	to stay with the chosen reference antenna.
> solint y CR	to average n records in each solution interval, set $y = nx$.
> solty 'L1R' ; solmode 'P' CR	to do phase-only solutions.
> aparm(6) 1 CR	to get some diagnostic messages.
> go CR	to write a new `SN` table.

Eyeball this new table with `SNPLT` — should be flat, not all over the place. But `EDITA` lets you look at more things at once.

> default edita CR	
> indi d; getn 4 CR	
> inext 'sn' ; dotwo 1 CR	to plot the phases in the new table.
> crowded 1 ; do3col 1 CR	to combine all IFs in each colorful plot.
> go CR	

If these plots are too crowded, you can interactively select individual IFs and polarizations. This task will let you flag data if desired. If you do, then you should re-run `CALIB`.

20. Use `BPASS` to calibrate the bandpass shape — corrects for wiggles in phase and amplitude as a function of frequency

> default bpass CR	
> indi d; getn 4 CR	
> docalib 1 CR	to apply the new `SN` table phases.
> refant 22 CR	
> solty 'l1r' ; solint = -1 CR	to average the entire time range.
> bpassprm(5) = 1; bpassprm(10) = 3 CR	to normalize the solutions at the end rather than the data record-by-record.
> go CR	To generate a new `BP` table containing one record per antenna.

21. Eyeball the resulting bandpass (`BP`) table — `APARM(8) = 2` plots the `BP` table in `POSSM`. If you see odd bandpass shapes or spikes in the bandpass, you may need to edit your data more extensively and repeat steps 17 through 20.

> tget possm CR	
> indi d; getn 4 CR	
> solint = -1 CR	
> aparm(8) 2 ; aparm(9) 1 CR	to plot all IFs together for each bandpass.

> nplots 2 C_R	2 per page is enough here.
> go C_R	

Alternatively, BPEDT lets you use the bandpass table to guide additional flagging of the bandpass calibrator(s). It is similar to EDITA in that it lets you look at multiple antennas at the same time for comparison.

> default bpedt C_R
> indi d; getn 4 C_R
> inext 'bp' ; GO C_R

22. Copy the bandpass table back to your main data with TACOP.

> default tacop C_R	
> indi d; getn 4 C_R	
> inext 'bp' C_R	
> outdi indi ; geton 3 C_R	to select output name parameters
> go C_R	

23. Since RFLAG probably made a large number of flags, let us copy the data file with UVCOP and apply the flags.

> default uvcop C_R	
> indi d; getn 3 C_R	
> outdi indi ; outn inna C_R	to stay on d.
> outcl 'fixcp' C_R	with a meaningful class name
> flagver n C_R	to apply the highest flag table (n); use IMHEAD if unsure. Version 0 means no flagging here.
> go C_R	

24. Use SETJY to compute the flux for the primary flux calibrator (3C286), storing it in the SU table.

> default setjy C_R	
> indi d; getn 5 C_R	
> source '3C286' ' C_R	specify all "known" calibrator sources.
> opty 'calc' C_R	to calculate and display fluxes from the frequencies plus tables of known sources.
> go C_R	

25. Go through all of your calibration sources, one at a time, to check for RFI with POSSM and, if present, to address it with SPFLG and/or RFLAG

> tget possm C_R	
> source 'calsrc_i' ' C_R	to examine the i^{th} calibration source.
> indi d; getn 5 C_R	
> docal 1 ; doband 1 C_R	apply our hard-won calibration.
> solint 0 C_R	to look at the average over all time.
> aparm(9) 1 C_R	to plot all IFs together.
> nplots 2 C_R	2 per page is enough here.
> go C_R	

If RFI is present (it almost certainly will be), two rounds or more of RFLAG may be needed.

> default rflag C_R
> indi d; getn 5 C_R
> source 'calsrc$_i$' ' C_R — to examine the $i^{\rm th}$ calibration source.
> docal 1 ; doband 1 C_R — apply the calibration.
> avgchan 11 C_R — to set the width of the median-window filter run across the spectral channels in each IF
> fparm = 3, x, -1, -1 C_R — to do a small rolling time buffer, set the sample interval, and to have NOISE and SCUTOFF set IF-dependent cutoff levels.
> fparm(13) = 1000 C_R — to clip amplitudes at 1000 Jy. There are many more FPARM options you may choose also.
> doplot 12 ; dotv 1 C_R — to do the 2 most useful plots on the TV.
> go C_R

RFLAG will return cutoff levels to the appropriate adverbs. Therefore, to apply the flags using the flagging levels found above, do *not* do a `tget rflag` which will reset the levels to zero. Instead, simply do

> doplot = -12 C_R — to apply the cutoff levels and determine new ones, plotting on the TV.
> go C_R

RFLAG will make a new flag table each time it determines flags (when DOPLOT ≤ 0). It may be good to check what RFLAG has done with POSSM. If more edits are needed, use SPFLG

> default spflg C_R
> indi d; getn 5 C_R
> source 'calsrc$_i$' ' C_R — to examine the $i^{\rm th}$ calibration source.
> dparm(6) x C_R
> docal 1 ; doband 1 C_R — apply the calibration.
> go C_R

26. Since RFLAG probably made a large number of flags, let us copy the data file once again with UVCOP and apply the flags.

> default uvcop C_R
> indi d; getn 5 C_R
> outdi indi ; outn inna C_R — to stay on d.
> outcl 'flaged' C_R — with a meaningful class name
> flagver n C_R — to apply the highest flag table (n); use IMHEAD if unsure. Version 0 means no flagging here.
> go C_R

27. Since all flag tables $< n$ were already merged into table n which we have applied, delete all flag tables in the output file.

> default extdest C_R
> indi d; getn 6 C_R
> inext 'fg' ; invers = -1 C_R — to delete all remaining versions of flag table.
> extdest C_R

28. Find the amplitude and phase calibration as a function of *time* using `CALIB`

> default calib CR	
> indi d; getn 6 CR	
> calcode '*' CR	to select all calibration sources
> docalib 1 ; doband 1 CR	to apply all existing calibration.
> refant 22 CR	to keep our reference antenna.
> solint 0 CR	One solution per calibrator scan is okay with good short-term phase stability. Otherwise, set to the longest time in minutes over which phase is stable.
> solty 'L1R' ; solmode 'a&p' CR	to solve for amplitude and phase with an L1 method iterated robustly.
> aparm(6) 1 CR	to print closure error statistics.
> go CR	

Pay attention to the fraction of failed solutions. If the fraction is large, try adjusting parameters such as `SOLINT`, etc.

29. The solution table (`SN`) produced by the previous step will contain amplitude gains at multiple levels, correct ones for those sources with fluxes in the source table (3C286) and incorrect ones for those which `CALIB` was forced to call 1 Jy. To solve for the unknown fluxes of the secondary calibrators

> default getjy CR	
> indi d; getn 6 CR	
> source 'calsrc$_1$', 'calsrc$_2$', 'calsrc$_3$', . . . CR	to list the secondary calibrators.
> calsour = '3c286' ' CR	to list the known calibrator(s).
> snver = n CR	to select only the `SN` table just written by `CALIB`. Use `IMHEADER` to find the maximum version number n.
> go CR	to solve for the secondary fluxes as functions of IF.

Check the values displayed carefully to make sure that they are sensible. `SOUSP` offers tools to examine and adjust the fluxes and `SN` table n.

30. Look at both amplitude and phase solutions with `EDITA` or `SNPLT`.

> tvini CR	
> default snplt CR	
> indi d; getn 6 CR	
> inext 'sn' ; opty 'amp' CR	to plot amplitudes.
> nplots 4 ; dotv 1 CR	to plot 4 panels on each TV page.
> opco 'alif' ; do3col 1 CR	to plot all IFs in each panel using color to distinguish them.
> go CR	

One can also plot IFs in separate panels or one at a time if these plots are too crowded. Then

> opty 'phas'; go CR	To check the phases.

To use `EDITA` to look at both amplitude and phase at the same time for multiple antennas

> default edita CR	
> indi d; getn 6 CR	
> inext 'sn' ; dotwo 1 CR	to plot the phases in the new table.
> crowded 1 ; do3col 1 CR	to combine all IFs in each colorful plot.
> go CR	

If these plots are too crowded, you can interactively select individual IFs and polarizations. This task will let you flag data if desired. If you do, you should then re-run CALIB.

31. Apply the solutions to the calibration table with CLCAL

> default clcal C_R	
> indi d; getn 6 C_R	
> calcode '*' C_R	to select all calibration sources
> sampty 'box' ; bparm 0.3, 0.3 C_R	To smooth the SN table if you used SOLINT less than full scans. Leave this out if you used scan averages.
> refant 22 C_R	
> go C_R	

32. Use SNPLT to check that the solutions have been interpolated between calibrators.

> tget snplt C_R
> inext 'cl' C_R
> go C_R

33. Make an image of 3C286 as a sanity check using IMAGR

> default imagr C_R	
> indi d; getn 6 ; outdi indi C_R	
> docal 1 ; doband 1 C_R	to apply the calibration.
> source = '3c286' ' C_R	to select only the primary calibrator.
> bchan 5; echan = $n - 4$ C_R	to omit edge channels in each IF, where n is the number of channels in an IF, now usually 128.
> nchav = echan-bchan+1; chinc nchav C_R	to average all spectral channels into the image.
> cellsize 1. C_R	to set the image cell spacing — depends on configuration.
> imsi 1024 C_R	to make a largish image to see sources around 3C286.
> outn '3C286_quick' C_R	
> niter 100 ; dotv 1 C_R	to do some Cleaning and guide the progress with the TV
> go C_R	

34. If you wish to image Stokes Q and U, it is now necessary to calibrate the polarization. This is still somewhat of a research question in AIPS, but the advice in § 4.3.16 may work. You will need to run procedure VLATECR to correct for Faraday rotation and dispersion, task RLDLY to correct for the V minus H delay offset, task PCAL with SPECTRAL 1 to calibrate the polarization D terms, and then task VHDIF to calibrate the V minus H phase offset which is also a function of spectral channel. PCAL is best run with an unpolarized calibration source which is common at P band. However, VHDIF requires a calibration source with a useful flux in Stokes U. 3C345 has this, but, due to the rotation measure, not at all frequencies. Significant frequency smoothing may help. Task VHMOD models a point source with linear polarization which may help you model your source.

35. At this point, it may be simpler to SPLIT the data set and apply calibration to the target sources which we then can work with one at a time.

> defsult split C_R
> indi d; getn 6 C_R
> outdi indi C_R
> docal 1 ; doband 1 C_R

> dopol 1 C_R	if you calibrated polarization
> sources = 'target$_1$', 'target$_2$', . . . C_R	to select only target sources.
> go C_R	

To create a number of separate, single-source uv data sets each with separate target source.

36. Flag RFI and bad data out of the target data set with two or more rounds of `RFLAG`

> default rflag C_R	
> indi d; getn 9 C_R	
> avgchan 11 C_R	Median window width on spectra.
> fparm = 3, x, -1, -1 C_R	to do a small rolling time buffer, set the sample interval, and to have `NOISE` and `SCUTOFF` set IF-dependent cutoff levels.
> doplot 12 ; dotv 1 C_R	To examine the two important plots on the TV.
> go C_R	

and then apply the flags levels and determine new ones

> doplot -12 ; go C_R

and perhaps repeat this line for more flagging. *NOTE: for flagging spectral-line target sources, you should set* `FPARM(4)=0` *since you should not flag in the spectral domain.* Check what `RFLAG` has done with `SPFLG`, which can be tedious.

> default spflg C_R	
> indi d; getn 9 C_R	to select the first of the split target data sets.
> dparm(6) x C_R	to set integration time.
> go C_R	

37. Because we used `TYAPL` only to apply the re-quantizer gains, your data weights simply reflect integration time. For better imaging, weights should reflect the actual data uncertainties. So re-weight your data with `REWAY`

> default reway C_R	
> indi d; getn 9 C_R	
> outdi indi C_R	
> aparm 31,0,500 C_R	to find rmses from rolling buffers 31 time intervals long on a baseline basis and then smooth them over 500 seconds.
> go C_R	

Note that this applies the final `SPFLG+RFLAG` flag table. Delete the other flag table versions with

> default extdest C_R	
> indi d; getn 10 C_R	
> inext 'fg' ; invers = -1 C_R	to delete all remaining versions of flag table.
> extdest C_R	

L.2 P-band imaging in *AIPS*

38. The following steps relate to imaging your calibrated data. As the aim of this appendix is calibration, the imaging will be discussed only briefly here. The following items refer to continuum imaging. Make a dirty image of the center with `IMAGR`

> default imagr C_R	
> indi d; getn 10; outdi indi C_R	
> bchan 5; echan = $n-4$ C_R	to omit edge channels in each IF, where n is the number of channels in an IF.
> nchav = echan-bchan+1; chinc nchav C_R	to average all spectral channels into the image.
> cellsize 10 C_R	to set a cell size which will depend on the VLA configuration.
> imsi 1024,1024 C_R	
> niter 0 C_R	to do no Cleaning
> go C_R	

39. Clean image with faceting, auto-boxing, and TV display; we use `DEFAULT IMAGR` here to initialize all of `IMAGR`'s many parameters

> default imagr C_R	
> indi d; getn 10; outdi indi C_R	
> task 'setfc' C_R	
> cellsize 0 ; imsize 0 C_R	to have `SETFC` determine the correct cell and image size for these data.
> bparm 2.2, 5, 2, 8, 0.3 C_R	to image fully out to a radius of 2 degrees and to include 0.3 Jy NVSS sources out to a radius of 8 degrees.
> flux 1 C_R	to include exterior sources above 1 Jy only.
> boxfile 'MYAREA:Target1.box C_R	to list the facets found.
> go C_R	

The following does not use default since we need adverbs set for, and by, `SETFC`, such as inname, imsize, and cellsize

> task 'IMAGR' C_R	
> flux 0 C_R	
> bchan 5; echan = $n-4$ C_R	to omit edge channels in each IF, where n is the number of channels in an IF.
> nchav = echan-bchan+1; chinc nchav C_R	to average all spectral channels into the image.
> niter 5000 C_R	to Clean 5000 iterations.
> do3dim true ; overlap 2 C_R	to use true 3-D geometry, Cleaning one facet at a time in each cycle (recommended).
> dotv true C_R	to control the imaging with the TV.
> oboxfile boxfile C_R	to save all changes to Clean boxes.
> im2parm=2,0 C_R	to allow up to 2 Clean boxes to be created automatically at each cycle.
> go C_R	

Note, the field may have enough spectral index variation to require imaging (and self-calibration) of each spectral window (IF) individually.

40. There is normally enough flux in a P-band field to do phase self-calibration and we recommend that you now do that. Use `CALIB` with the output of step 39 as the model including all `NFIELD` facets as the `NMAPS` images.

> default calib C_R	
> indi d; getn 10; outdi indi C_R	
> nmaps = nfield C_R	to include all facets
> in2d = indisk; get2n M C_R	to set the 2^{nd} file name to the output image of class `ICL001` (catalog number M).
> soltyp 'l1r' ; solmode 'p' C_R	to start with phase-only solutions.
> solint = nx C_R	to set a short integration time as n times the sampling interval.
> go C_R	

41. Examine the output `SN` table with `EDITA` to see if there are times with bad solutions. If not, repeat steps 39 and 40 using

> tget imagr; docal = 1; go C_R	to make new images.
> tget calib; in2seq = 2 C_R	to point at the second set of images.
> solmode 'a&p' C_R	to do amplitude as well *if there is enough flux in the field*.
> go C_R	

Re-examine the new `SN` table with `EDITA`. In particular, watch for periods of time when the amplitude gain of an antenna gets rather larger than normal. It has been found that the ionosphere can defocus sources on scales as small as the antenna diameter. It is best to delete such data.

42. Spectral-line imaging is both simpler and more complicated. One may image each spectral channel individually. However, even for absorption experiments, it may be best to image all channels free of the spectral line together to make a continuum image and model. This will almost certainly require multiple facets and extensive cleaning but it will produce a single continuum model that can be applied to all spectral channels with `UVSUB`. Spectral index in the continuum may force you to do this one spectral window at a time. You can add the continuum image back to the cube with `COMB`.

    At this point, run `IMAGR` on the output of `UVSUB` one spectral window at a time over all spectral channels of interest with `NCHAV=1; CHINC=1`. You may be able to limit the field of view imaged to one facet surrounding the target source.

43. The task `TVSPC` is a lovely way to explore the output image cube. To run this task, we must first transpose the cube

> default trans C_R	
> indi d; getn C; outdi indi C_R	where C is the catalog number of the output cube from `IMAGR`.
> transcod = '312' C_R	to put frequency first.
> go C_R	to make a transposed cube.

One also needs a "meaningful" single-plane image, perhaps the central facet of the continuum image made above. Or we can compute the first moment image.

> default xmom C_R	
> indi d; getn T; outdi indi C_R	where T is the catalog number of the output cube from `TRANS`.
> flux = f C_R	to compute only with brightness $> f$ Jy/beam.
> go C_R	to amke a first moment image.

Finally

> default tvspc C_R	
> indi d; getn M; outdi indi C_R	where M is the catalog number of the meaningful image, perhaps the output plane from `XMOM`.
> in2di d; get2n T C_R	where T is the catalog number of the output cube from `TRANS`.
> in4di d; get4n C C_R	where C is the catalog number of the output cube from `IMAGR`.
> go C_R	Have fun!

This allows you to look at spectra from the cube at individual spatial pixels as well as the cube as an image at individual frequencies. Enter `EXPLAIN TVSPC` or see *AIPS*Memo 120 for details.

44. Make image(s) and final calibrated data sets into FITS files with `FITTP`. Exit `AIPS` and

> setenv FIT /lustre/mmao/Pband C_R	for tcsh and c shells
> export FIT=/lustre/mmao/Pband C_R	for bash shell

Then start `AIPS` again

> default fittp C_R	
> indi d; getn M C_R	to select an image or data set
> dataout 'FIT:VirA.FITS C_R	to make an appropriate output name
> go C_R	

And repeat with new M and `DATAOUT` values.

L.3 Additional Recipes

L.3.1 Banana storage

Bananas ripen after harvesting. They do it best at room temperature. Because of this there are three stages to banana storage.

1. **On the counter:** When you buy a bunch of bananas that are not exactly at the ripeness you want, you can keep them at room temperature until they are just right for you. Be sure to keep them out of any plastic bags or containers.
2. **In the refrigerator:** If there are any bananas left, and they are at the ripeness you like, you can put them in the refrigerator. The peel will get dusty brown and speckled, but the fruit inside will stay clear and fresh and at that stage of ripeness for 3 to 6 days.
3. **In the freezer:** If you want to keep your bananas even longer, you can freeze them. Mash the bananas with a little lemon juice, put them in an air tight freezer container and freeze. Once they're defrosted, you'll go bananas baking bread, muffins and a world of other banana yummies. Or, you can freeze a whole banana on a Popsicle stick. When it is frozen, dip it in chocolate sauce, maybe even roll it in nuts, then wrap it in aluminum foil and put it back in the freezer. Talk about a scrumptious snack.