10 SINGLE-DISH DATA IN *AIPS*

AIPS was not originally intended as a reduction package for single-dish data and cannot be considered as such today. However, because of the similarity of single-dish data taken at "random" pointings on the sky to interferometric data taken at "random" locations in the *uv* plane, *AIPS* was seen as a system to be used to solve large imaging problems arising from single-dish observations. Many of the *AIPS* *uv*-data tasks are able to do something sensible — or even desirable — with single-dish data and a few special tasks to process single-dish data have been written. The present chapter contains a discussion of the representation of single-dish data in *AIPS* followed by a description of how such data may be calibrated, corrected, converted into images, and analyzed by *AIPS*. A final section on using single-dish observations to improve the imaging of interferometric data represents what little we now know about this potentially important process.

10.1 *AIPS* format for single-dish data

Single-dish data in *AIPS* is treated as *uv* data with different, but related random parameters and with the imaginary part of the visibility replaced by an additive calibration or offset. The *u* and *v* random parameters are replaced by parameters labeled `RA` and `DEC`, although other labels such as `ELON`, `ELAT`, `GLON`, and `GLAT` are also recognized. (Conversion between these coordinate systems is not provided in the "*uv*" plane although some conversion can be done on images.) The random parameter data are the sample coordinates in degrees. The `TIME1` random parameter is the time (IAT) since midnight on the reference date in days as with real *uv* data. The `BEAM` random parameter corresponds to `BASELINE` and is used to separate data which should be edited and calibrated separately (*e.g.*, separate beams of a multi-feed system, different polarizations or observing runs of a multi-polarization system). The actual beam number is recorded as 257 times the desired number so that visibility-data tasks will recognize the "baseline" as auto-correlation data. Two other random parameters, `SCAN` and `SAMPLE`, have no relation to any visibility parameters and are simply used to retain the "scan" number and sample number within the scan which are traditional in single-dish observations. Very little is made of these, but `INDXR` will make a new index entry when the scan number changes and `PRTSD` will display the scan and sample numbers. `SDVEL` uses the scan numbers to determine when to update the reference velocity in some observing modes. Single-dish data may be stored in compressed form, in which the weight and compression scale are stored as random parameters exactly as in true visibility data. This should *not* be done if the applied offset is large as in beam-switched continuum observations.

The measured single-dish flux, usually in units of degrees Kelvin, appears in the real part of the "complex visibility." The imaginary part of the visibility is sometimes used to hold an offset which can be applied to the data to remove, for example, a time-variable bias. The data weight is used to weight the data and should be proportional to σ^{-2}, where σ is the uncertainty in the flux. The visibility sample can contain multiple polarizations, described with the `STOKES` axis (values 1 through 4 for I, Q, U, and V, respectively). The sample can also contain multiple spectral-line frequencies, described with a `FREQ` axis giving the observed reference frequency and increment in Hz.

The *uv*-data header in an *AIPS* data set is expected to contain the reference (usually central) longitude and latitude given either as 1-pixel coordinate axes and/or in the "observed" coordinate location. The convolution size is usually used to hold the single-dish beam width (fwhm) and rest frequency and velocity information should also appear with spectral-line data. Many of the parameters can be added to the header by the user if they are missing and needed. Verbs `ADDBEAM`, `ALTDEF`, and `PUTHEAD` are useful for this purpose. A complete data set will also have an antenna extension file giving the location of the antenna. This allows tasks to compute things like zenith angles and Doppler corrections when needed. (`OTFUV` began making antenna tables in the `15JAN96` release.)

10.1.1 On-the-fly data from the 12m

At the present time, the only reliable routes for single-dish data into *AIPS* are provided by the tasks `OTFBS` and `OTFUV`. These tasks work only on beam-switched continuum and on spectral-line observations, respectively, from the NRAO 12m telescope. They use files in the UniPops native format and do not read the FITS table format written by UniPops. Both programs are designed for "on-the-fly" or "OTF" observing modes in which the telescope takes data rapidly while continuously changing its pointing position.

10.1.1.1 Listing OTF input files

To read OTF files, you must first define an environment variable to point to the disk area in which your data resides. This environment variable and your file names should be in upper case letters, but there is an `AIPS` "feature" which allows you to use lower case. On Unix systems, you may set the environment variable and rename the files to upper case with

% cd /my/disk/directory CR	to switch to the disk directory containing your data.
% setenv MYAREA ‘pwd‘ CR	to define `$MYAREA` under c shell, or
% export MYAREA=‘pwd‘ CR	to define `$MYAREA` under Bourne, bash, korn shells.
% mv mysdd.file MYSDD.FILE CR	to rename the data file to upper case letters.
% mv mygsdd.file MYGSDD.FILE CR	to rename the gain file to upper case letters.

Then start your `AIPS` session.

To review the contents of your data set, use the task `OTFIN` which will list SDD modes, IF and scan numbers, times, coordinates, velocities, and number of samples. This output should help in setting the range of scan numbers to be loaded by `OTFUV` or `OTFBS`. Type:

> TASK 'OTFIN' ; INP CR	to list the required inputs on your screen.
> INFILE 'MYAREA:MYSDD.FILE' CR	to specify the name of the 12m raw data file, where `MYAREA` is an environment variable which points at a disk data area and `MYSDD.FILE` is the name of your file in that area. See § 3.10. If your environment variable and/or your file name contain lower case letters, type the name carefully with the correct case for all letters and leave off the second (close) quote mark. When you use this "feature" of the `AIPS` compiler, you cannot type anything following the `INFILE` name (or other string adverb) on that line.
> BCOUNT 0 ; ECOUNT 0 CR	to include all 12m scans in the file.
> BIF 0 CR	to include all SDD "IFs."
> DOCRT -1 CR	to print the listing on the line printer, or
> DOCRT 132 CR	to view the listing, one page at a time, on your terminal window. The width given should match the width of your window; a width of 132, as given here, maximizes the information per line.
> INP CR	to review the parameters.
> GO CR	to run the task.

10.1.1.2 Reading spectral-line OTF files into *AIPS*

To run OTFUV after running OTFIN, type

> TASK 'OTFUV' ; INP C_R — to list the required inputs on your screen.

> INFILE 'MYAREA:MYSDD.FILE' C_R — to specify the name of the 12m raw data file, where MYAREA is an environment variable which points at a disk data area and MYSDD.FILE is the name of your file in that area. See § 3.10.

> IN2FILE 'MYAREA:MYGSDD.FILE' C_R — to specify the name of the 12m gain file corresponding to the file specified with INFILE.

> BCOUNT n_1 ; ECOUNT n_2 C_R — to include 12m scans n_1 through n_2 in the output file.

> BIF 0 ; EIF 0 C_R — to include all SDD "IFs" matching the lowest numbered one found. IFs which do not match in central frequency or channel width are skipped.

> DOUVCOMP TRUE C_R — to write the data in a compressed format. This reduces the size of the file by nearly a factor of 3 with no significant loss of information in this case.

> XINC 1 ; YINC 1 C_R — to write out all data samples with no time averaging. One can smooth by YINC samples and write out the data every XINC sample times in order to reduce the size of the output data set and improve the signal-to-noise of the individual samples with only a minor loss of information..

> DOWEIGHT 1 C_R — to use offs and gains interpolated to the time of each observation. This seems to produce better results.

> DETIME 0 C_R — to add no offset to the actual observation times.

> BCHAN 0; ECHAN 0 C_R — to include all spectral channels.

> CHANSEL 0 C_R — to flag no channels. CHANSEL 31,34,3 C_R, for example, would mark channels 31 and 34 as bad. Data may be edited later more selectively.

> INP C_R — to review the parameters.

> GO C_R — to run the task.

While OTFUV runs, it will show you (on the message monitor or your window) the name and location of the output *AIPS* file created and then provide a list of the scans and IFs read and the gain scans used upon them.

In many cases, the 12m in OTF mode observes two separate polarizations using the same center frequency and spectral resolution. In the UniPops/12m nomenclature, these are separate "IFs." A similar nomenclature is used to distinguish the feeds in the multi-feed system. OTFUV can now read up to eight IFs at the same time, avoiding the necessity of multiple runs of OTFUV, followed by a data sort to restore time order. OTFUV will distinguish the IFs not by an *AIPS* "IF axis," but by assigning them beam numbers equal to the SDD IF number (or autocorrelator baseline number equal to the SDD IF number with itself).

You may append data from another IF in the first input data set or data from another OTF pass on the source to the *AIPS* data set created above, by entering new INFILE and IN2FILE names and new BCOUNT and ECOUNT ranges, if needed and

> BIF m_1 ; EIF m_2 C_R — to load IFs m_1 through m_2.

> DOCONCAT TRUE C_R — to enable the concatenation mode.

> OUTDISK n ; GETONAME m C_R — to select the output file, where n and m are the output disk and catalog slot number used by the first run of OTFUV.

> FQTOL *ff* CR	to allow data sets within *ff* MHz of each other to be concatenated. Doppler tracking will cause two OTF passes to appear to be at separate frequencies. Narrow-band, wide-field observations should not be concatenated in this way; see the discussion of `SDVEL` below (§ 10.2.4).
> GO CR	to run the task appending the additional data.

Another way to concatenate two 12m IFs — or multiple observing runs — is to create two output files with `OTFUV` and then concatenate them with `DBCON`. If the two `OTFUV` files are in time order, then `DBCON` will actually merge the two data sets, retaining the time order. Avoid the use of multiple sub-arrays, which are a useless complication in this case, by setting `DOARRAY = 0`. To have the most "complete" antenna file, put the data set with the higher 12m IF in the first input name set (`INNAME`, `INCLASS` etc.)

10.1.1.3 Reading continuum OTF files into *AIPS*

The NRAO 12m telescope can observe in a beam-switched continuum on-the-fly mapping mode. Such data may be read into *AIPS* and reduced, in a somewhat experimental fashion, into images. To read in the data (after using `OTFIN`), enter

> TASK 'OTFBS' ; INP CR	to list the required inputs on your screen.
> INFILE 'MYAREA:MYSDD.FILE' CR	to specify the name of the 12m raw data file, where `MYAREA` is an environment variable which points at a disk data area and `MYSDD.FILE` is the name of your file in that area. See § 3.10.
> BCOUNT n_1 ; ECOUNT n_2 CR	to include 12m scans n_1 through n_2 in the output file.
> BIF 0 ; EIF 0 CR	to include all SDD "IFs" matching the lowest numbered one found. IFs which do not match in central frequency or channel width are skipped.
> INP CR	to review the parameters.
> GO CR	to run the task.

While `OTFBS` runs, it will show you (on the message monitor or your window) the name and location of the two output *AIPS* files created (one for "plus" and one for "minus" beam throws) and then provide a list of the scans and IFs read with the number of samples. The two output files will have the same names except for a "+" and a "–" as the sixth character of the output class.

10.1.2 Other input data formats

Another method for getting single-dish data into *AIPS* is through the use of FITS-format binary tables. If the data are able to be put in a usable table, then the *AIPS* FITS reading tasks such as `FITLD` (see § 5.1.2) can be used to read them into a disk table attached to a cataloged file. Then `SDTUV` can be used to convert the table into the *uv* format described above applying a variety of calibrations along the way. Unfortunately, the non-*AIPS* program that did the UniPops to FITS conversion has been lost and the *AIPS* FITS readers cannot handle the FITS tables written by UniPops. There are two problems with the latter: *AIPS* is unable to handle tables with more than 128 columns while UniPops writes tables with around 200 columns. Even if *AIPS* could be extended in some special task, it would be unable to handle the current UniPops tables since the parameters given do not correctly describe the contents. Specialized unpublished knowledge about each receiver is required to disentangle the coordinate information and data structure.

The task `SDTUV` expects a sequence of related tables each with a number of keywords giving useful information such as scan, observer, telescope, object, scan start UT date and time, sample rate, velocity, and the like. The data are then a regular time sequence with each row of the table containing the right ascension, declination, and data for N receivers. Breaks in the time sequence are assumed to be new scans found in the next table. SDTUV has the ability to apply receiver position offsets and pointing corrections and to fit and remove receiver baselines using a sliding median window and spline fit. Interference rejection, lateral defocusing corrections, and a priori baseline removal are also offered. At present SDTUV is an example of what can be done rather than a directly usable task. It is limited to continuum problems currently and is moderately restricted in the number of data samples that can be read in any one scan.

Therefore, it will be necessary to write some sort of program in addition to those in the standard *AIPS* release to get single-dish data into *AIPS*. We encourage anyone who develops such a program to provide it to the *AIPS* group so that we may offer it to other single-dish users.

10.2 Single-dish data in the "uv" domain

Once you have gotten your data into *AIPS*, a wide range of tasks become available to you. In addition to the single-dish specific tasks discussed below, these include data movement tasks (UVCOP, UVSRT, DBCON), data averaging (AVER, UVAVG, AVSPC), non-interactive editing (CLIP, UVFLG), interactive editing (SPFLG, EDITR, TVFLG), data backup and restore (FITTP, FITLD), and data display (PRTAN, PRTUV, UVPRT, UVPLT).

10.2.1 Looking at your data: PRTSD, UVPLT, POSSM

In the process of calibrating, modeling, editing, and imaging of single-dish data, there are occasionally problems that seem to arise because users are not aware of the data that they actually have. PRTSD is the task for such users. It displays the data with or without calibration for selected portions of your data set. This will help you identify what pointing positions actually occur in your data, which channels are highly variable or bad, and the like. SPFLG, UVPLT, and others are good for looking at the data set as a whole, but PRTSD really shows you what you have.

To run it, type:

> TASK 'PRTSD' ; INP C_R to list the required inputs on your screen.

> INDISK *n* ; GETN *ctn* C_R to select the single-dish "*uv*" file to be displayed.

> DOCRT 1 C_R to select the on-screen display at its current width; make sure your window is at least 132 characters across for the best results.

> DOCELL -1 C_R to look at the data values; DOCELL > 0 causes the offsets that have been removed (usually 0) to be displayed.

> CHANNEL m C_R to display channels m through $m + 5$.

> DOCAL FALSE C_R to apply no calibration. Note that the 12m off scans and instrumental gains are applied by OTFUV; this parameter applies only to any additional calibration contained in CS files. See § 10.2.3.

> TIMERANG 0 C_R to look at all times.

> ANTENNAS $a1, a2, \ldots$ C_R to look at beams/IFs $a1, a2, \ldots$ only.

> BPRINT bb C_R to begin the display with the bb^{th} sample in the data set *before* application of the other selection criteria (TIMERANG, ANTENNAS, etc.)

> NPRINT 2000 C_R to shut off the display interactively or after a lot of lines.

> XINC x C_R to display only every x^{th} sample of those selected by the other criteria.

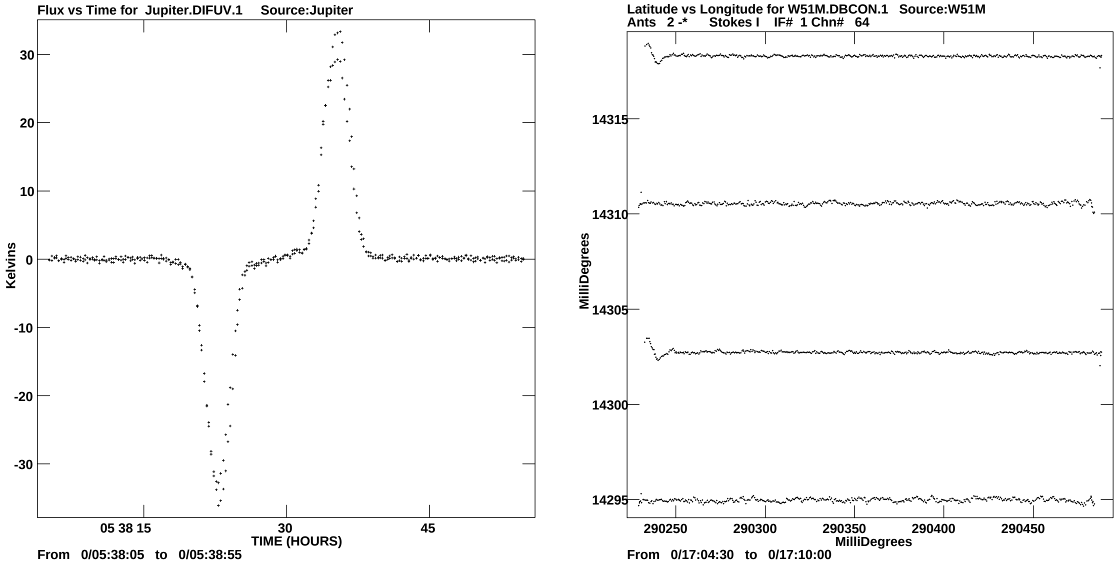


Figure 10.1: *left:* `UVPLT` display of 12m beam-switched continuum data on Jupiter. The time range is set to display one row of the OTF observation and the "minus" beam throw data have been subtracted from the "plus" throw. *right:* `UVPLT` display of the right ascension and declination of each sample in spectral-line OTF data set over a limited time range.

> INP CR	to review the inputs.
> GO CR	to start the task.

`PRTSD` will start and, after a pause to get through any data not included at the start of the file, will begin to display lines on your terminal showing the scan number, time, coordinates, and data for six spectral channels. After 20 or so lines, it will pause and ask if you want to continue. Hit CR to continue or type `Q` CR or q CR to quit. If you decide to get hard copy, set `DOCRT = -1` and the output will be printed. To save the display in a text file, without printing, set `DOCRT = -1` and give the name of the file in the `OUTPRINT` adverb. See § 3.2 and § 3.10.1 for more information on printing.

There are a number of tasks which plot *uv* visibility data; see § 6.3.1. The most basic of these is `UVPLT`, which can be useful for single-dish data sets. For example, to generate the plot of flux versus time in 12m OTF beam-switched continuum differenced data seen in the accompanying figure (Figure 10.1), the parameters given below were used:

> TASK 'UVPLT' ; INP CR	to review the inputs.
> INDI *n* ; GETN *ctn* CR	to select the disk and catalog entry of the data set.
> DOCALIB FALSE CR	to apply no calibration; `UVPLT` does not understand single-dish calibration.
> BPARM = 11,9,0 CR	to plot time in hours on the x axis and flux in Kelvins on the y axis. The other parameters can be used to specify fixed scales on one or both axes, but are just self-scaled in this example.
> XINC 1 CR	to plot every selected sample.
> BCHAN 1 ; ECHAN 1 CR	to plot only "spectral channel" 1, the actual data values.
> ANTENNA 1,0 ; BASELINE 0 CR	to do all baselines with antenna 1, namely 1–1 or, in 12m nomenclature, IF 1..
> TIMER = 0, 5, 38, 5, 0, 5, 38, 55 CR	to restrict the times to a single scan.
> DOCRT = -1 ; GO CR	to make a plot file of these data.

After `UVPLT` is running, or better, after it has finished:

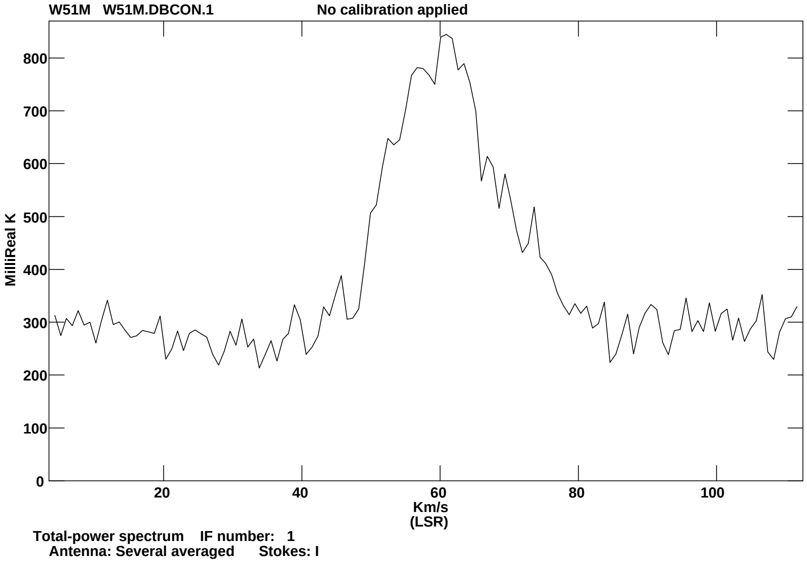


Figure 10.2: POSSM display of all of a 12m observation taken on W51. All samples (on and off the actual source) and both "antennas" are averaged together.

```
> PLVER 0 ; GO LWPLA CR              to plot the latest version on a PostScript printer/plotter.
```

The second plot in Figure 10.1 was generated with BPARM = 6, 7 and shows where samples occur on the sky in a different data set.

With spectral-line data, POSSM will plot observed spectra averaged over selected "antennas," time ranges, and the like. Thus,

```
> TASK 'POSSM' ; INP CR              to review the inputs.
> INDI n ; GETN ctn CR               to select the disk and catalog entry of the data set.
> BCHAN 0 ; ECHAN 0 CR               to plot all channels.
> ANTENNAS 0 ; BASELINE 0 CR         to average all 12m IFs.
> TIMERA 0 ; SOLINT 0 CR             to average all times into one plot.
> APARM(7) = 2 CR                    to have velocity labels on the x axis.
> GO CR                              to run the task.
```

LWPLA was then used to make a PostScript version of the plot seen in Figure 10.2.

10.2.2 Editing your data: UVFLG, SPFLG, EDITR

Editing is the process by which you mark data samples as "unreliable" or "bad." In AIPS, there are two methods for doing this. The simplest is to have the editing software alter the weight of the sample to indicate that it is flagged. If the data are not compressed, this is a reversible operation. If the data are compressed, however, then the data themselves are marked as "indefinite" and the operation is not reversible. The second method is the use of a flag (FG) extension table attached to your *uv* data set. This method requires that the data be sorted into time order and is supported by most, but not all, tasks. If the task does not have the FLAGVER adverb, then it does not support flag tables. However, since flag tables can be applied to the data by SPLIT, we use them in the recipes below.

To sort the data into "time-baseline" (TB) order,

```
> TASK 'UVSRT' ; INP CR              to review the inputs.
> INDI n ; GETN ctn CR               to select the disk and catalog entry of the data set.
> SORT 'TB' CR                       to sort into time-baseline order.
> ROTATE 0 CR                        to avoid damage to the coordinates.
> INP CR                             to check the parameters, e.g., the output name.
> GO CR                              to run the task.
```

The most direct flagging task is UVFLG, which puts commands into the flag table one at a time (or more than one when read from a disk text file). To use this task to flag channel 31 from 7 to 8 hours on the first day of observation from the second input (single-dish nomenclature) IF:

```
> TASK 'UVFLG' ; INP CR              to review the inputs.
> INDI n ; GETN ctn CR               to select the disk and catalog entry of the sorted data set.
> FLAGVER 1 CR                       to select the use of a flag table.
> TIMERANG 0, 7, 0, 0, 0, 8, 0, 0 CR to set the time range from 7 to 8 hours.
> BCHAN 31 ; ECHAN 31 CR             to flag only channel 31.
> BIF 0 ; EIF 0 CR                   to do all AIPS IFs.
> ANTEN 2, 0 ; BASELIN 2, 0 CR       to select "baseline" 2-2, the 2nd IF in 12m nomenclature.
> APARM 0 CR                         to ignore amplitude in flagging.
> OPCODE 'FLAG' CR                   to flag the data.
> REASON 'Bad channel' CR            to store away a reason.
> INP CR                             to check the full set of adverbs.
> GO CR                              to add one line to the flag table, creating one if needed.
```

Multiple runs of UVFLG may be done to incorporate what you know about your data into the flagging table. Use PRTSD and the plot programs to help you find the bad data. If you have a long list of flagging commands, you may find it easier to use the INFILE option of UVFLG to read in up to 100 flagging instructions at a time from a free-format text file.

The task CLIP is popular on interferometer data sets since it automatically flags all samples outside a specified flux range without interaction with the user. This blind flagging is often acceptable for interferometer data since each *uv* sample affects all image cells so that the damage done by a few remaining bad samples is attenuated by all the good samples. However, a bad sample in single-dish data affects only a few image cells and is hence not attenuated. Thus it is important to find and remove samples that are too small as well as those that are too large. For this reason, we do not recommend CLIP, but suggest that you look at your data and make more informed flagging decisions.

The best known of the interactive editing tasks is TVFLG (§ 4.4.3). This task is not suitable for single-dish data since it displays multiple baselines along the horizontal axis. The data on these baselines are related in interferometry, but, in single dish, they are from separate feeds or polarizations and hence neither numerous nor necessarily related. For spectral-line single-dish data, the task SPFLG is an ideal task to examine your data and to edit portions if needed. SPFLG is a menu-driven, TV display editing task in which spectral channel varies along the horizontal axis of the TV display and time along the vertical. (The spectral channels for each interferometer IF are displayed on the horizintal axis, but single-dish data in *AIPS* has only 1 of this sort of IF.) The data may be displayed with as much or as little time averaging as desired and is very useful for examining your data even if you do not think that editing is needed.

To run SPFLG, type

```
> TASK 'SPFLG' ; INP CR              to review the inputs.
> INDI n ; GETN ctn CR               to select the disk and catalog entry of the sorted data set.
> FLAGVER 1 CR                       to select the use of a flag table.
> BCHAN 0 ; ECHAN 0 CR               to view all spectral channels.
```

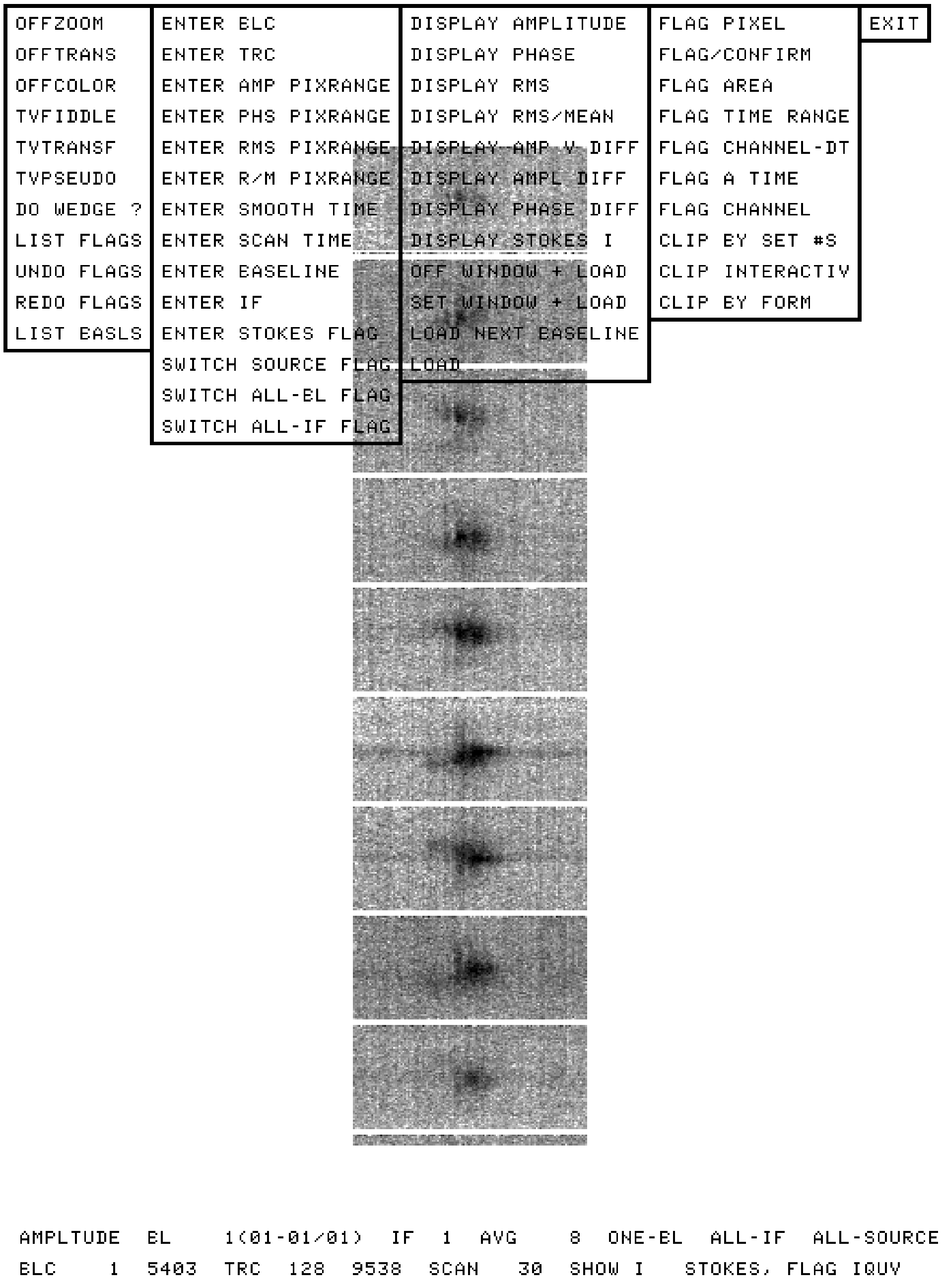
> DOCALIB FALSE CR	to inhibit *interferometer* calibration of your data.
> IN2SEQ 0 ; DOCAT FALSE CR	to create a new, but temporary "master file" each time.
> ANTEN 0 ; BASEL 0 CR	to include all "baselines."
> DPARM 0, 1, 0, 0, 0, 0.1 CR	to include autocorrelation data and to set the fundamental interval used to average data into the master file. The defaults for these parameters are not suitable for single-dish data. The other DPARM parameters may be ignored since they can be altered during the interactive session.
> INP CR	to review the inputs.
> GO CR	to begin the interactive display and editing.

The task will then read your data to determine which times occur in the included portions (you may set TIMERANG, restrict autocorrelations, etc.) and then construct a master grid file with spectral channel as the first axis, pseudo-regular times on the second axis (gaps are mostly suppressed), and, if needed, baseline number on the third axis. SPFLG tells you the size of the resulting file, *e.g.*, `SPFLG1: Basic UV image is 128 14079 pixels in X,Y (Ch,T)`.

At this point, SPFLG selects an initial *display* smoothing time long enough to fit all of the master grid onto your TV window. It then averages the data to this interval and creates a display not unlike that seen in Figure 10.3. Move the TV cursor to any menu item (it will change color to show which has been selected) and press button D for on-line help information or press buttons A, B, or C to select the operation. Normally, you will probably begin by reducing the smoothing time (ENTER SMOOTH TIME menu option followed by typing in the new smoothing multiple on your AIPS window). Note that the display does not change other than to add an asterisk after the smoothing time to indicate that that will change on the next image load. This behavior is to allow you to alter a number of choices before doing the potentially expensive TV display. In this typical example, you would either ENTER BLC and ENTER TRC by hand and finally LOAD the sub-image or you can do this interactively with SET WINDOW + LOAD. You may examine data values (like CURVAL) and flag data with the options in the fourth column. Flagged data are removed from the display. You may review the flags you have prepared, undo any that you dislike, re-apply the remaining ones to make sure the display is correct, and modify the appearance of the display with the options in the first column. The image may be shown in zoom only during editing in order to give you greater accuracy in examining the data values and locations. If you are doing some time smoothing within SPFLG, the DISPLAY RMS option allows you to view images of the rms rather than the value of the time average. Such a display allows you to find excessively noisy portions of the data quickly.

Finally, when you are done, select EXIT. If you have prepared any flagging commands, SPFLG will ask you if you wish to enter them into your input data set. Answer yes unless you want to discard them or you have set DOCAT TRUE to catalog the master file in order to use it for multiple sessions. If you set FLAGVER to one, then the flag commands are put into a flag table and can be deleted later if you wish. If no flag table is specified, then the data themselves are flagged and SPFLG will take a long time to finish after you tell it to apply the flags.

SPFLG is not useful on continuum data; the interactive editor of choice for such data used to be the task IBLED, but is now EDITR. These tasks are also useful for spectral-line data in that they can display the average (and rms) of a selected range of channels. The spectral averaging should let you see more subtle level problems than can be seen on individual channels (*i.e.*, in SPFLG). EDITR is a menu-driven, TV display editing task, but it does not use grey-scales to show data values. Instead, it plots time on the horizontal axis and data value on the vertical axis. The full data set for the chosen baseline is displayed initially in a potentially crowded area at the bottom of the TV window. This area is available for editing. If DOTWO is true, then it also displays above the edit area a second observable (initially the difference between the amplitude and a running mean of the amplitude) for the primary baseline. EDITR allows you to display up to ten other "baselines" (*e.g.*, 12m-antenna IFs) in frames above the active editing frame. These should speed the process of editing and guide you in the choice of flagging one or all baselines at the time of the observation. A smaller time range or window into these full data sets may be selected interactively to enable



```
AMPLTUDE  BL    1(01-01/01)  IF  1  AVG    8  ONE-BL  ALL-IF  ALL-SOURCE
BLC    1  5403  TRC  128  9538  SCAN   30  SHOW I   STOKES, FLAG IQUV
```

Figure 10.3: A display of a sample TV screen from SPFLG on single-dish data, made using the AIPS task TVCPS to produce a negative black-and-white display. The SPFLG menu (in the boxes) and status lines (at the bottom) are displayed in a graphics plane which is normally colored light green. The data are grey scales in a TV memory and may be enhanced in black-and-white or pseudo-colored. The data actually displayed range in intensity from -1.7 to 5.2 Kelvins (as stated during the image loading) and have been averaged to 0.8 seconds. The entire master grid contains 14079 times, but the current window includes only times 5403 through 9538. Flag commands generated at the moment illustrated will flag all source names, all IFs (in the AIPS sense), only the displayed baseline, and all Stokes.

more detailed editing. Be sure to set SOLINT to specify an appropriate averaging interval. Unlike SPFLG, no further time averaging is possible. The menu options allow you to work your way through all of your data, selecting time windows and baselines as desired. Consult § 5.5.2 for more details about EDITR.

EDITR has the ability to display a second data set for reference in parallel with the one being edited. This option is likely to prove useful for beam-switched continuum observations. Select one of the beam throws for editing and the other for reference display. Then, if editing is required, reverse the roles. It may also be useful to look at your beam-switched data in its differenced form. The task DIFUV may be used to difference the plus and minus throws, followed by EDITR (or any other *AIPS* uv-data task) to look at the differences. Be sure to tell DIFUV that the time difference between the plus and the minus beam throws should not be considered significant, *i.e.*, SOLINT = 1 / 8 / 60 or a little bit more to avoid round-off effects.

10.2.3 Calibrating your data: CSCOR, SDCAL

The current calibration routines for single-dish data in *AIPS* are fairly rudimentary. The concept is similar to that used for interferometers. Corrections are developed in an extension table (called CS in single-dish, CL in interferometry) which can be applied to the data by some tasks. In particular, the single-dish tasks PRTSD and SDGRD are able to apply the CS table to the data without modifying the data as stored on disk. They do this using the DOCAL and GAINUSE adverbs. Other *uv* tasks, designed primarily for interferometry, also use these adverbs, but do not understand or apply CS tables. For such tasks, you should carefully turn off the calibration option. If you do not, such tasks will fail.

There are two tasks which can create CS tables: SDTUV discussed above and INDXR. To use the latter, enter

> TASK 'INDXR' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry of the data set.
> CPARM T_1, T_2, ΔT C_R	to set the largest gap (T_1) and longest scan (T_2) times expected in the data set (for the index table) and to set the time interval (ΔT) in the CS table, all in minutes.
> GO C_R	to run the task to create an index (NX) and a calibration (CS) table attached to the main data set.

Note that this task requires the data to be in time order and expects an antenna (AN) table. You may set CPARM(5) to the maximum antenna number (beam number) in your data set and, with a few grumbles, INDXR will still create and initialize a CS table when you do not have an antenna table.

At this writing, the CS table may be used to correct the recorded right ascension and declination (*i.e.*, the pointing) and to correct the amplitudes for atmospheric opacity and other gain as a function of zenith angle effects. To add an atmospheric opacity correction to the CS table produced by INDXR, type:

> TASK 'CSCOR' ; INP C_R	to review the inputs.
> TIMERAN 0 ; ANTENN 0 C_R	to do all times and antennas.
> GAINVER 1 ; GAINUSE 2 C_R	to modify the base table, producing a new table.
> OPCODE 'OPAC' C_R	to do the opacity correction.
> BPARM O_z, 0 C_R	to specify the zenith opacity in nepers.
> GO C_R	to run the task.

Note that CSCOR only writes those records in the output file that you have selected via TIMERANG, ANTENNAS, etc. To make a new CS table to work for the full data set, you should first use TACOP to write the new table and then set GAINVER and GAINUSE to both point at the new table. CSCOR needs to compute the zenith angle and therefore needs to have an antennas file. If your data set does not have one, you may give the antenna longitude and latitude in the CPARM adverb. The other operations offered by CSCOR are GAIN, PTRA, and PTDC which apply as second-order polynomial functions of zenith angle corrections to the gain, right ascension, and declination, respectively. The format of the `CS` table allows for an additive flux correction as well. There are no tasks at this time to determine such a correction.

The basic single-dish tasks `PRTSD` and `SDGRD` can apply the `CS` table to the data as they read them in. Other *uv* tasks which are more directed toward interferometry data cannot do this. If you need to use such tasks with corrected data, then you must apply the corrections with `SDCAL` and write a new "calibrated" data set. To do this:

> TASK 'SDCAL' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry of the data set.
> TIMERA 0 ; FLAGVER 1 C_R	to do all times and apply any flagging.
> BCHAN 1 ; ECHAN 0 C_R	to get all channels.
> DOCAL TRUE ; GAINUSE 0 C_R	to apply the highest numbered `CS` table.
> APARM 0 C_R	to do no averaging of spectral channels.
> GO C_R	to run the task.

The output file from `SDCAL` can then be fed to `UVPLT`, `SPFLG`, or any other *uv*-data task including of course `PRTSD` and `SDGRD`.

10.2.4 Correcting your spectral-line data: `SDLSF` and `SDVEL`

It may be convenient to remove a spectral baseline from each sample before the imaging step. Doing so may allow you to skip the removal of a spectral baseline from the image cubes (as described in § 10.4.1). To do this, type:

> TASK 'SDLSF' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry of the data set.
> NCOUNT 1 C_R	to solve for a slope as well as a constant in the baselines.
> DOALL 1 C_R	to fit a single baseline to all samples taken at a particular time. This is useful for single-beam, multi-polarization data, but, for multi-beam data, it is found that instrumental problems dominate weather and require `DOALL = -1` C_R instead.
> DOOUT -1 C_R	to avoid writing a continuum data set.
> FLUX 0 ; CUTOFF 0 C_R	to write all data with no flagging.
> CHANSEL $s_1, e_1, i_1, s_2, e_2, i_2 \ldots$ C_R	to use every i_1 channel from s_1 through e_1, every i_2 channel from s_2 through e_2, and so forth to fit the baseline. Be sure to avoid dubious channels, if any, at the ends and any channels with real line signal. It is important to have regions at both ends of the spectrum to fit the slope.
> INP C_R	to review the inputs.
> GO C_R	to run the task.

You may, and probably should, use `FLUX` and `CUTOFF` to flag those data having excessive noise or excessive signals in individual channels. These "excesses" are measured only in the channels selected by `CHANSEL` for fitting the baseline.

If you have observed a wide field with relatively narrow spectral channels, there is an effect which you should consider. The "velocity" corresponding to a particular frequency of observation depends on the velocity definition (*e.g.*, LSR or heliocentric), the direction at which the telescope pointed, the time of year, the time of day, and the location of the telescope. Most telescopes adjust the observing frequency to achieve the desired velocity for some reference time and position and many adjust the frequency periodically to account for time changes. However, few, if any, can adjust the observing frequency for every pointing direction and time in a rapidly scanned on-the-fly observing mode. The 12m telescope now sets the frequency once per image with respect to the reference coordinate (usually the image center). In this mode, the maximum velocity error in a 2 degree by 2 degree image is about 1.16 km/s (in LSR velocities) and 0.79 km/s (heliocentric). Since mm lines are often narrow, this can be a significant effect. Fortunately, single-dish OTF data may be fully corrected for this effect so long as your spectra are fully sampled in frequency. The task `SDVEL` shifts each spectrum so that the reference channel has the reference velocity for its pointing position. The `DPARM` adverb array is used to tell the task how the telescope set reference velocities and to ask the task to report any excessive shifts and even flag data having really excessive shifts. The latter are to detect and/or remove times in which the telescope pointing was significantly in error (*i.e.*, high winds). `DPARM(1)` should be set to 0 for 12m data taken after 5 May 1997 and to 2 for data taken before that date. The task `VTEST` was written to help you evaluate the magnitude of this effect.

10.2.5 Modeling your data: `SDMOD` and `BSMOD`

It is sometimes useful to replace your actual data with a source model or, if your continuum levels are well calibrated, to add or subtract a model from your data. The task to do this is called `SDMOD` and allows up to four spatially elliptical Gaussians (or an image) to replace the data or to be added to the data in either with either a Gaussian or no frequency dependence. When the data are replaced, a random noise may also be added. `SDMOD` has options for modeling beam-switched continuum data (set `BPARM(1) = 1`) as well as for spectral-line data. For example, to see what a modestly noisy point source at the origin would look like after all of the imaging steps:

> TASK 'SDMOD' ; INP CR	to review the inputs.
> INDI *n* ; GETN *ctn* CR	to select the disk and catalog entry of the data set.
> BCHAN *n* ; ECHAN *n* CR	to get one channel only.
> NGAUSS 1 ; APARM 0 CR	to get one Gaussian with no frequency dependence.
> GWIDTH 0 ; GPOS 0 CR	to do a point source (convolved with the single-dish beamwidth in the header) at the coordinate center.
> GMAX 1, 0 ; FLUX 0.05 CR	to do a 1 K object with rms noise of 0.05 K.
> GO CR	to run the task.

The output file from `SDMOD` can then be fed to `SDGRD`, `BSGRD`, or any other appropriate task as if it were regular data. The input model is convolved with the single-dish beamwidth given in the *uv* data header before being used to replace or add to the input data. The history file will show in detail what was done.

Beam-switched observations may be modeled with task `BSMOD`. No input data set is needed. Instead two regular grids of switched data are constructed from a specified model plus noise and a variety of instrumental defects.

10.3 Imaging single-dish data in AIPS

10.3.1 Normal single-dish imaging

The process of imaging in single-dish is a process of convolving the "randomly distributed" observations with some convolving function and then resampling the result on a regular image grid. This process used to be done in AIPS with tasks `SELSD` and `GRIDR`, which will probably still work. However, the tasks `SDGRD` and `SDIMG` combine the data calibration, selection, projection, sorting, and gridding in one task capable of imaging all spectral channels into one output data "cube." They are relatively easy to run, but selecting the correct input adverb values is more difficult. Choose `SDGRD` for most single-dish applications; `SDIMG` is very similar but can handle larger output images at the cost of making a sorted copy of the entire input data set (which can be very large). Type:

`> TASK 'SDGRD' ; INP CR` to review the inputs.

`> INDI` *n* `; GETN` *ctn* `CR` to select the disk and catalog entry of the data set.

`> TIMERA 0 ; FLAGVER 1 CR` to do all times and apply any flagging.

`> BCHAN 1 ; ECHAN 0 CR` to get all channels.

`> DOCAL FALSE CR` to apply no calibration.

`> OPTYPE '-GLS' CR` to make the image on a "global sinusoidal" kind of projection.

`> APARM 0 CR` to use the observed right ascension and declination given in the header as the center of the image. For concatenated data sets, use `APARM` to specify a more appropriate center.

`> REWEIGHT 0, 0.05 CR` to have an "interpolated" or best-estimate image for output, cutting off any cells with convolved weight < 0.05 of the maximum convolved weight.

`> CELLSIZE` *c* `CR` to set the image cells to be *c* arc seconds on a side.

`> IMSIZE` N_x , N_y `CR` to make the image of each channel be N_x by N_y pixels centered on the coordinate selected by `APARM`.

`> XTYPE 16 ; XPARM 0,0,0,0,50 CR` to select convolution function type 16 (a round Bessel function times Gaussian) with default parameters and 50 samples of the function per pixel. The default of 20 samples/cell is probablt adequate.

`> INP CR` to review the inputs.

`> GO CR` to run the task.

`SDGRD` begins by reading the data selecting only those samples which will fit fully on the image grid. It reports how many were read and how many selected. If you have made the image too small, with `IMSIZE` or `CELLSIZE`, then data will be discarded. Use `PRTSD` with a substantial `XINC` to determine the full spatial distribution of your data. It does not hurt to have the output image be a bit bigger than absolutely necessary. If you are uncertain about the parameters to use, try running `SDGRD` on a single channel to begin with since it will be much faster.

A number of these parameters require more discussion. `REWEIGHT(1)` selects the type of output image. The data are multiplied by their weights (which depend on the system temperature), convolved by the sampled convolving function and then summed at each image pixel. `REWEIGHT(1) = 1` selects the result, which is not calibrated in any way since its scaling depends on the scaling of the data weights and the convolving function and on the distribution of data. While the program "grids" the actual data it also does the same process on the data replaced by 1.0. That result, the convolved weights may be obtained with `REWEIGHT(1) = 2`. The most meaningful image, which is obtained with `REWEIGHT(1) = 0`, is the ratio of the former to the latter. This is the interpolated or best-estimate image and will be similar to the convolved image in well-sampled regions except for having retained the calibration. `REWEIGHT(1) = 3` tells `SDGRD` to compute an image of the expected noise (actually $1/\sigma^2$) in the output image of type 0; see `WTSUM` below (§ 10.4.2) for its use.

`REWEIGHT(2)` controls which pixels are retained in the output image and which are blanked by specifying a cutoff as a fraction of the maximum convolved weight. It is important to blank pixels which are either simple extrapolations of single samples or, worse, extrapolations of only a couple noisy samples. In the latter case, it is possible to get very large image values. Thus, if the output is $(W_1D_1+W_2D_2)/(W_1+W_2)$ where the D's are data and the W's are convolved weights and if, say $W_1 = 0.1001, D_1 = 1.0, W_2 = -0.1000, D_2 = -1.0$, then the output would be 2001. Such large and erroneous values will be obvious, but will confuse software which must deal with the whole image and will also confuse people to whom you may show the image. In simple cases, in which all data have roughly the same data weights (system temperatures), setting `REWEIGHT(2) = 0.2` or even more is probably wise. However, if some portions of the data have significantly lower weights than others, then you may have to set a lower value in order to keep the low-weight regions from being completely blanked.

The choices of `CELLSIZE` and the widths of the convolving function are related to the spatial resolution inherent in your data, *i.e.*, to the single-dish beamwidth. If the pixels and function are too small, then data samples which are really from the same point in the sky will appear as if different in the output image. If, however, they are too large, then too much data will be smoothed together and spatial resolution will be lost. The latter may be desirable to improve signal-to-noise, but image smoothing can be done at a later stage as well. You may wish to experiment with these parameters, but it is usually good to start with a `CELLSIZE` about one-third of the beamwidth (fwhm) of your telescope. The default parameters (`XPARM`s) of all convolving functions may be used with this cell size. You may vary these parameters in units of cells or in units of arc seconds; enter HELP UVnTYPE C_R to look at the parameters for type n ($n = 1$ through 6). If you give `XTYPE =` $n + 10$, then you get a round rather than square function which is perhaps better suited to this type of data. If you wish to change the cell size, but retain the same convolving function in angular measure on the sky, you may give `XTYPE` < 0 and specify the `XPARM`s in arc seconds rather than cells.

The choice of convolving function affects the noise levels and actual spatial resolution in the output image. In effect, the Fourier transform of the convolving function acts to modify the illumination pattern of the feed horn onto the aperture. Figure 10.4 shows slices through the Fourier transforms of six of the available convolving functions. The ideal function would be flat all the way across and then suddenly zero at the edges. Type 14 is the widest, but has a deep dip in the middle. This leaves out the center portion of the dish and illuminates the outer portions, effectively improving the spatial resolution of the image over that of the normal telescope, but with a noticeable loss of signal-to-noise ratio. The spheroidal functions, on the other hand, illuminate the center fully and leave out the outer portions. This degrades the spatial resolution, but noticeably improves the noise levels. Types 4 and 16 seem to be the best compromise. Type 16 is preferred since it is zero at the edges. Round functions require more computer memory than square ones, so type 4 would be preferred on computers with small memories.

Images may be built up from observations taken at significantly different times. The simplest way to do this is to concatenate the two "*uv*" data sets on disk with `OTFUV` or `DBCON` (§ 10.1.1.2) and then use `SDGRD` once to make the image. Some single-dish data sets are so large — or the time interval so great — that this is not practical. `SDGRD` combines observations taking into account the data weights which are based on the measured system temperatures. You can get the same weighted averaging in the image plane if you first compute a "weight" image and then use the task `WTSUM` to do the averaging. To get a weight image:

> TGET SDGRD C_R — to get the inputs used for the actual image cube.

> REWEIGHT(1) 3 C_R — to get the weight image which is proportional to $1/\sigma^2$ expected from the actual gridding done on the data whose weights are assumed proportional to their $1/\sigma^2$.

> BCHAN n ; ECHAN BCHAN C_R — to image a single channel when there is no channel-dependent data weights and flagging.

> GO C_R — to get the weight image.

See § 10.4.2 for details about `WTSUM`.

10.3.2 Beam-switched continuum imaging

The construction of images from beam-switched on-the-fly continuum observations is more properly a research question than one of production software. Observers in this mode should be aware that the optimal methods of data reduction are probably not yet known and that the methods currently provided require the user to determine three critical correction parameters. In this mode, the telescope is moved in a raster of offsets in azimuth and elevation with respect to the central coordinate. The beam is switched rapidly from a "plus" position to a "minus" position at constant elevation. On the 12m, there are four plus samples and four minus samples taken each second, all taken while the telescope is being driven rapidly in azimuth at a constant (relative to the central source) elevation. In principle, each pair of plus and minus points contain

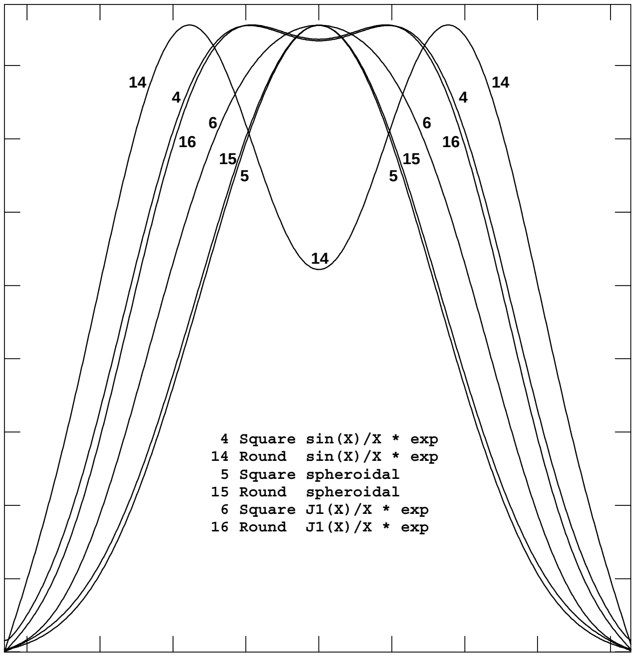


Figure 10.4: Slices through the Fourier transforms of six convolving functions using the XTYPE numbers shown on the plot with default values of the XPARMs.

the same instrumental bias but different celestial signals. It is then the job of the software to disentangle the time variable bias from the two beams' estimates of the sky brightness.

A technique for doing the disentangling was first described by Emerson, Klein, and Haslam (Astronomy and Astrophysics, 76, 92–105, 1979). The plus and minus samples are differenced removing the instrumental bias and creating two images of the sky, one positive and one negative. Problems arise because the two images potentially can overlap and because, in the OTF mode of observing, the telescope positioning is not exactly along rows of the output image and the relative positioning of the plus and minus beams varies both due to the wobbles in the telescope pointing and due to the reversing of the direction of telescope movement. The Emerson *et al.* technique involves a convolution of each row in the differenced image with a function which is a set of positive and negative delta functions (or $\frac{\sin x}{x}$ functions when the total beam throw is not an integer number of image cells). It turns out that the problem of image overlap is largely solved by this technique. Unfortunately, differences in the position of the plus and minus beam with respect to the source and to the image cells appear to limit the quality of the images produced with this technique.

The principal task used to produce images from data is called BSGRD. It makes two images from the two "uv" data sets written by OTFBS, gridding each sample at the coordinate at which it was observed (neglecting the throw but not the telescope movement between plus and minus). If the beam throw was not exactly along constant elevation, it then shifts the two images. Then it applies the Emerson *et al.* technique, fitting and removing baselines, differencing the two images, and convolving the difference image with an appropriate $\frac{\sin x}{x}$ function. Finally, BSGRD regrids the data from relative azimuth-elevation coordinates onto a grid in normal celestial coordinates. This task is a combination of four tasks, SDGRD described above to make the images, OGEOM to do the rotation correction, BSCOR to apply the Emerson *et al.* technique, and BSGEO to regrid the data onto normal celestial coordinates.

To use BSGRD, type

```
> TASK 'BSGRD' ; INP C_R                    to review the inputs.
```

> INDI n ; GETN ctn C_R	to select the disk and catalog entry of the data set. Note that the class name is assumed to have a plus sign in the sixth character for the plus throw data set and a minus sign in that character for the minus throw data set.
> TIMERA 0 ; FLAGVER 1 C_R	to do all times and apply any flagging.
> DOCAL FALSE C_R	to apply no calibration.
> OPTYPE '-GLS' C_R	to make the image on a "global sinusoidal" kind of projection.
> APARM 0 C_R	to use the observed right ascension and declination given in the header as the center of the image. For concatenated data sets, use APARM to specify a more appropriate center.
> REWEIGHT 0, 0.05 C_R	to have an "interpolated" or best-estimate image for output, cutting off any cells with convolved weight $<$ 0.05 of the maximum convolved weight.
> CELLSIZE c C_R	to set the image cells to be c arc seconds on a side.
> IMSIZE N_x , N_y C_R	to make the image of each throw be N_x by N_y pixels centered on the coordinate selected by APARM.
> XTYPE 16 ; XPARM 0,0,0,0,50 C_R	to select convolution function type 16 (a round Bessel function times Gaussian) with default parameters and 50 samples of the function per pixel. The same function is used in both convolutions.
> FACTOR f C_R	to multiply the recorded throw lengths by f in doing the Emerson *et al.* correction.
> ROTATE ρ C_R	to correct the throws for being ρ degrees off from horizontal.
> DPARM $1, 1, x_1, x_2, x_3, x_4$ C_R	to specify that the two beams have the same relative amplitude and to give the pixel numbers to be used to fit baselines in *both* images.
> ORDER 1 C_R	to fit a slope as well as a constant in the horizontal baseline in each row.
> DOCAT -1 C_R	to delete the intermediate images created by BSGRD.
> INP C_R	to review the inputs.
> GO C_R	to run the task.

BSGRD takes three correction parameters which you must supply: the throw length error FACTOR, the throw angle error ROTATE, and the relative beam gain error DPARM(1). To estimate these, you will need data on a relatively strong point source. Use SDGRD to make an image of each throw of these data, setting ROTATE = 0 since rotation *must* be done later and setting ECHAN = 1 to eliminate the coordinate information which is confusing to MCUBE and used only by BSGEO. The tasks IMFIT and/or JMFIT (§ 7.5.2) may be useful in fitting the location and peak of the two beams. Since there is likely to be a significant offset from zero in these images, be sure to fit for the offset using a second component of CTYPE = 4. For reasons that are not clear, these tasks may not provide sufficiently accurate positions. Another approach then is to take the two images produced by SDGRD and then run OGEOM and BSCOR for a range of rotations and factors. Find the image that is most pleasing and put its parameters into BSGRD for the program source. Of course, it is not clear that these correction factors are constant with time or pointing, so this could all be bologna.

For example,

> TASK 'OGEOM' ; INP C_R	to review the inputs.
> INDI n ; GETN ctn_+ C_R	to select the disk and catalog entry of the plus image.
> APARM 0 C_R	to do no shifts or rescaling.
> DOWAIT 1 C_R	to wait for a task to finish before resuming AIPS.
> OUTCLA 'OGEOM+' C_R	to set the output class to show the throw sign.

> FOR APARM(3) = -2 , 2.01 BY 0.1 ; GO; END C_R — to produce 41 plus images each with a slightly different rotation.

> GETN ctn_- ; OUTCLA 'OGEOM-' C_R — to select the minus image as input and specify the output class.

> FOR APARM(3) = -2 , 2.01 BY 0.1 ; GO; END C_R — to produce 41 minus images each with a slightly different rotation.

Then apply the Emerson *et al.* corrections to each of the 41 with

> TASK 'BSCOR' ; INP C_R — to review the inputs.

> INDI n ; GETN ctn_+ C_R — to select the disk and catalog entry of one of the rotated plus images.

> IN2DI n ; GET2N ctn_- C_R — to select the disk and catalog entry of one of the rotated minus images.

> FACTOR f C_R — to multiply the recorded throw lengths by f in doing the Emerson *et al.* correction. Use 1.0 as an initial guess.

> DPARM $1, 1, x_1, x_2, x_3, x_4$ C_R — to specify that the two beams have the same relative amplitude and to give the pixel numbers to be used to fit baselines in *both* images. The choice of the x_n is significant.

> ORDER 1 C_R — to fit a slope as well as a constant in the horizontal baseline in each row.

> FOR INSEQ = 1 : 41; IN2SEQ = INSEQ ; GO ; END C_R — to produce 41 “corrected” images.

It is convenient to look at the images with tools such as `TVMOVIE` (§ 8.5.4) and `KNTR` (§ 10.4.5). To build the “cube”, use `MCUBE` as:

> TASK 'MCUBE' ; INP C_R — to review the inputs.

> INDI n ; GETN ctn C_R — to select the disk and catalog entry of the first of the corrected images.

> IN2SEQ 41 ; IN3SEQ 1 C_R — to set the sequence number loop limit and increment.

> AXREF 0 ; AX2REF 41 ; NPOINTS 41 C_R — to set the locations of the images in the cube explicitly.

> DOALIGN -2 C_R — to have `MCUBE` ignore the differing image rotations.

> DOWAIT -1 ; GO C_R — to resume normal task functioning and to build the data cube.

Examine the cube to find the “best” plane and use the rotation of that plane to run similar tests varying the throw length correction factor. One of these cubes, testing rotation, is illustrated in Figure 10.5.

The determination of throw length is similar:

> TASK 'BSCOR' ; INP C_R — to review the inputs.

> INDI n ; GETN ctn_+ C_R — to select the disk and catalog entry of the plus image at the best rotation.

> IN2DI n ; GET2N ctn_- C_R — to select the disk and catalog entry of one of the corresponding rotated minus image.

> OUTNA 'ROTATE TEST' C_R — to assign a new output name.

> DPARM $1, 1, x_1, x_2, x_3, x_4$ C_R — to specify that the two beams have the same relative amplitude and to give the pixel numbers to be used to fit baselines in *both* images. The choice of the x_n is significant.

> ORDER 1 C_R — to fit a slope as well as a constant in the horizontal baseline in each row.

> DOWAIT 1 C_R — to run the task in wait mode.

> FOR FACTOR = 0.9 ; 1.101 BY 0.005; GO ; END C_R — to produce 41 “corrected” images.

It is convenient to look at the images with tools such as `TVMOVIE` (§ 8.5.4) and `KNTR` (§ 10.4.5). To build the “cube”, use `MCUBE` as:

> TASK 'MCUBE' ; INP C_R — to review the inputs.

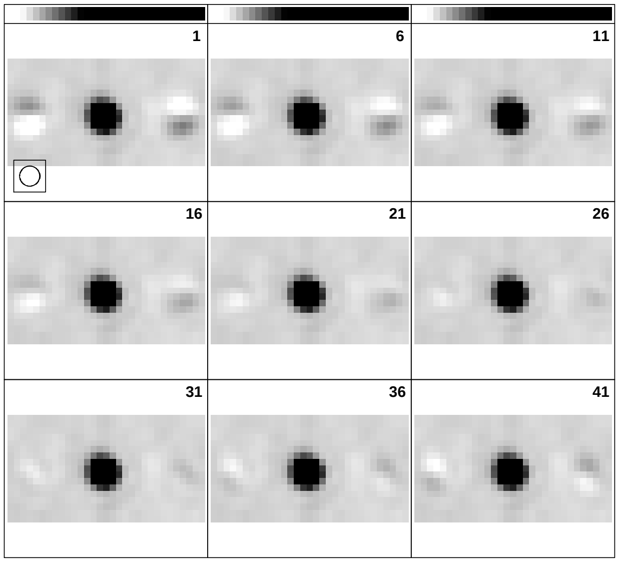


Figure 10.5: Images at selected rotations from 2.0 to -2.0 by -0.5 degrees. Rotations between -0.5 and -1.0 appear to minimize the artifacts due to the incomplete cancelation of the plus and minus beams.

> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry of the first of the new corrected images.
> IN2SEQ 41 ; IN3SEQ 1 C_R	to set the sequence number loop limit and increment.
> AXREF 0 ; AX2REF 41 ; NPOINTS 41 C_R	to set the locations of the images in the cube explicitly.
> DOWAIT -1 ; GO C_R	to resume normal task functioning and to build the data cube.

Examine the cube to find the “best” plane and use the scaling factor of that plane in later imaging. One of these cubes, testing throw length, is illustrated in Figure 10.6.

BSGRD is in fact a deconvolution algorithm to remove the plus-minus beam from the difference image. An experimental Clean algorithm has been made available in BSCLN. Although initial tests seemed promising, it appears to converge to the EKH solution after very many iterations and to have systematic problems before that point. The one-dimensional diaplay task BSTST will allow you to evaluate and compare the two algorithms on model (one-dimensional) data.

10.4 Analysis and display of single-dish data

The analysis and display of images produces from single-dish data are not, in general, different from those produced by interferometers. See Chapter 6 for a discussion of display tools, Chapter 7 for a variety of analysis tasks, and § 8.5 and § 8.6 for spectral-line analysis and display. Some matters of particular interest to single-dish users will be discussed below.

10.4.1 Spectral baseline removal

As the SPFLG display in § 10.2.2 shows, one of the first things most users will want to do is remove a spectral baseline at each pixel in the their image. This is frequently done with SDLSF (§ 10.2.4). To do this in the image plane, you must first transpose the data cube to make the frequency axis be first:

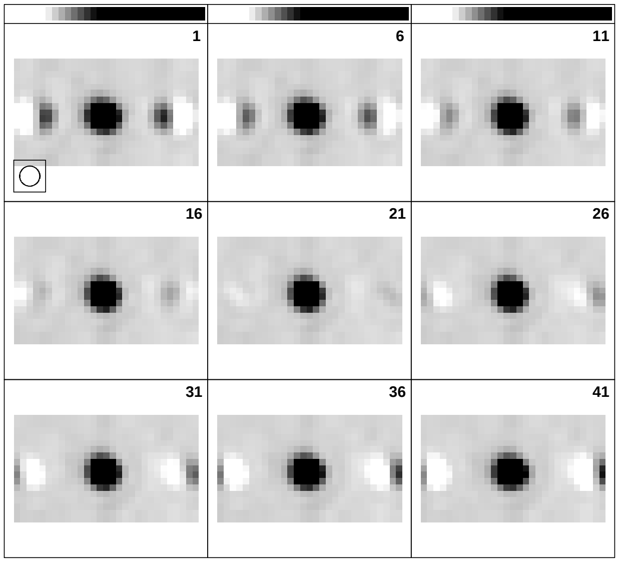


Figure 10.6: Images at selected beam throw corrections from 0.9 to 1.1 by -0.5. Note that rotation errors cause vertical separations of the plus and minus images while throw length errors cause horizontal separations (and hence incomplete cancelation) of the beams.

> TASK 'TRANS' ; INP C_R	to review the task's parameters.
> INDI n_1 ; GETN ctn_1 C_R	to select the input image from disk n_1 catalog slot ctn_1.
> TRANSCOD '312' ; OUTCL 'VXY' C_R	to move the frequency axis from 3$^{\rm rd}$ to 1$^{\rm st}$.
> GO C_R	to transpose the cube.

You must also determine, using this input image if needed, which spectral channels are completely free of real emission or absorption. `TVMOVIE` is often useful; see § 8.5.4. Then:

> TASK 'IMLIN' ; INP C_R	to review the task's parameters.
> INDI n_2 ; GETN ctn_2 C_R	to select the input image (output from `TRANS`) from disk n_2 catalog slot ctn_2.
> ORDER 1 C_R	to subtract linear baselines; up to 4$^{\rm th}$ are allowed.
> NBOXES n C_R	to select n contiguous regions along the spectral axis to be used in fitting the channels.
> BOX c_{11}, c_{12}, c_{21}, c_{22}, ... c_{n1}, c_{n2} C_R	to use spectral channels $c_{11} - c_{12}$, $c_{21} - c_{22}$, up to $c_{n1} - c_{n2}$ to fit the baselines at each pixel.
> INP C_R	to review the inputs.
> GO C_R	to fit the baselines, writing a new data cube.

It is sometimes useful to specify `DOOUT TRUE` to obtain images of the fit parameters and of their uncertainties. The uncertainty in the DC offset is a good measure of the uncertainty in the image.

The output from `IMLIN` is the baseline-corrected image in the familiar position-velocity form, with a third axis giving multiple positions on the second celestial coordinate. To go back to sky images as a function of frequency:

> TASK 'TRANS' ; INP C_R	to review the task's parameters.
> INDI n_3 ; GETN ctn_3 C_R	to select the input image (output from `IMLIN`) from disk n_3 catalog slot ctn_3.
> TRANSCOD '231' ; OUTCL 'XYV' C_R	to move the frequency axis from 1$^{\rm st}$ to 3$^{\rm rd}$.
> GO C_R	to transpose the cube back again.

10.4.2 Combining images: WTSUM, BSAVG

To do a weighted average of multiple images of the same field, be sure to make all images with the same geometry type, the same cell size, and the same center coordinate. If you have two images,

> TASK 'WTSUM' ; INP C$_R$	to review the inputs.
> INDI n_1 ; GETN ctn_1 C$_R$	to select the first input image from disk n_1 catalog slot ctn_1.
> INDI n_2 ; GET2N ctn_2 C$_R$	to select the second input image from disk n_1 catalog slot ctn_2.
> INDI n_3 ; GET3N ctn_3 C$_R$	to select the first weight image from disk n_3 catalog slot ctn_3.
> INDI n_4 ; GET4N ctn_4 C$_R$	to select the second weight image from disk n_4 catalog slot ctn_4.
> DOINVER FALSE C$_R$	to state that the weight images are weights rather than rms's.
> GO C$_R$	to compute an averaged image cube and a new weight image.

The weight images can be either a single plane or a cube that matches the corresponding image cube. All must be on the same spectral and celestial coordinate system.

If you have more than two images of the same field, then all images must have the same name parameters, differing only by having consecutive sequence numbers. All weight images must have the same name parameters with corresponding consecutive sequence numbers. The verb RENAME may be used to correct problems in naming. Then

> TASK 'WTSUM' ; INP C$_R$	to review the inputs.
> INDI n_1 ; GETN ctn_1 C$_R$	to select the first input image from disk n_1 catalog slot ctn_1.
> CLR2NAME ; IN2SEQ m_2 C$_R$	to select the looping mode and set the highest image sequence number.
> INDI n_3 ; GET3N ctn_3 C$_R$	to select the first weight image from disk n_3 catalog slot ctn_3.
> CLR4NAME C$_R$	to clear the unused fourth name set.
> DOINVER FALSE C$_R$	to state that the weight images are weights rather than rms's.
> GO C$_R$	to compute an averaged image cube and a new weight image.

If m_1 is the sequence number of the first image (in ctn_1) and w_1 is the sequence number of the first weight image (in ctn_3), then images of sequence numbers m_1 through m_2 will be weighted with corresponding weight images of sequence number w_1 through $w_1 + m_2 - m_1$. All weight images must be a single plane or all weight images must be a full cube matching the images.

BSAVG is a special task written to average beam-switched continuum images. Each image is Fourier transformed and weighted to give no weight to Fourier components at the beam switching spatial frequency and direction (since the images lack any non-noise information at these lines in the Fourier domain). Images made at different paralactic angles (*i.e.*, different hour angles) have these zero-weight lines at different angles while images made with different throw lengths have these zero-weight lines at different spatial frequencies. Thus, averaging images in this way (and Fourier transforming them back) should produce images with less noise and more information content. This algorithm works only on images that are made very quickly. If there is a significant rotation of the paralactic angle during the observation of one image, then the zero-weight "line" is actually curved and smeared away from the center (in Fourier space). The failure of this algorithm when observations are made with constant-elevation throws is one reason why some telescopes are designed to beam-switch in celestial coordinates.

10.4.3 Spectral moment analysis

A data cube may be reduced to a line-sum and a predominant-velocity image when the spectral shape is fairly simple at all points of the image. The simplest task to do this is:

> TASK 'XMOM' ; INP C_R	to review the inputs.
> INDI n ; GETN *ctn* C_R	to select the input image from disk n catalog slot *ctn* — use the output from `IMLIN` with velocity as the first axis.
> FLUX x C_R	to include only pixels $> x$ in brightness when computing the moments.
> GO C_R	to compute images of the o^{th} through 3^{rd} moments plus an image of the number of pixels used at each position.

This simple prescription will produce a result which should tell you whether this mode of analysis is interesting. If it is, then the regions of signal should be separated from regions of no signal so that the latter do not contribute to the noise in the moment images. See the discussions in § 7.4 and § 8.6 for methods of doing this. After the non-signal regions are blanked, the moments should be recomputed.

10.4.4 Source modeling and fitting

Gaussian fitting of images is discussed in some detail in § 7.5 while source modeling may be done in the "*uv*" data domain with `SDMOD` (§ 10.2.5) and in the image domain with `IMMOD`. The task `SAD` will find, and fit Gaussians to, sources in your image. Although it works on a plane of the image at a time, it records the plane number in its output model-fit (`MF`) table. This will allow you to examine the fits to your sources as a function of frequency. To run `SAD` on a number of image planes:

> TASK 'SAD' ; INP C_R	to review the inputs.
> INDI n ; GETN *ctn* C_R	to select the image cube from disk n catalog slot *ctn*
> BLC 0 ; TRC 0 C_R	to search for sources over the full plane.
> DORESID FALSE C_R	to delete the residual image after fitting; the fit results are kept in an `MF` file attached to the input image.
> NGAUSS 10 C_R	to allow up to 10 possible sources to be fit; make this enough to allow for a noise spike or two.
> CUTOFF x C_R	to fit "islands" of flux $> x$ only — this is probably the most important parameter.
> DOCRT 132 C_R	to display results on your workstation rather than the line printer.
> DOALL 1 ; DOWIDTH 1 C_R	to allow the task to fit multiple sources to an island and to fit the source widths.
> OUTVERS -1 C_R	to suppress writing of `CC` files.
> INVERS 1 C_R	to use one `MF` file for all fits.
> DOWAIT TRUE C_R	to resume `AIPS` only when the task finishes; this allows looping without tripping over ourselves.
> INP C_R	to recheck the inputs.
> FOR BLC(3) = c_1 TO c_2 ; GO; END C_R	to fit channels c_1 through c_2.

`SAD` will reject dubious solutions for a variety of reasons. The `DPARM` adverb allows you to control these reasons and `PRTLEV` controls how much of an explanation you get.

`SAD` offers a printer option to provide a detailed account of each execution. To view a simpler summary of the current contents of one or more `MF` files, use

> TASK 'MFPRT' ; INP C_R	to review the inputs.
> INDI n ; GETN *ctn* C_R	to select the image cube from disk n catalog slot *ctn* as input to `SAD`.
> INVER n_1 ; IN2VER n_2 ; XINC 1 C_R	to view `MF` file versions n_1 through n_2.
> DOCRT 132 C_R	to see the display on your monitor.

> FLUX 0 ; IMSIZE 0 C_R	to see all components.
> SORT 'C' C_R	to see the file in channel number order.
> GO C_R	to run the task.

Setting DOCRT FALSE and specifying OUTPRINT will produce a file suitable for some non-*AIPS* modeling programs.

10.4.5 Image displays

The subject of displays in *AIPS* has been treated extensively in earlier chapters. To make a printer representation of your image, see § 6.2.2 for a discussion of PRTIM. See § 6.3.2 for a discussion of plotter displays of images including tasks CNTR, PCNTR, GREYS, PLROW, PROFL, IMVIM, and IMEAN. Spectral-line displays are described in some detail in § 8.5.4 including tasks KNTR and PLCUB and the TV=movie display verbs TVMOVI and TVCUBE. The use of the TV for display, image enhancement, parameter setting, data examination, image comparison, and the like is described in detail in §§ 6.4.

For tutorial purposes, we will include one example here. The contouring task of choice is now KNTR since it can display images in grey-scales and/or contours with one or more planes per display and with an optional beam display. For example, to display several spectral channels as contours with the 0^{th}-moment (total CO) image as a grey scale on each display, enter

> TASK 'KNTR' ; INP C_R	to review the inputs.
> INDI n_1 ; GETN ctn_1 C_R	to select the image cube from disk n_1 catalog slot ctn_1.
> IN2DI n_2 ; GET2N ctn_2 C_R	to select the 0^{th}-moment image plane from disk n_2 catalog slot ctn_2.
> DOCONT 1 ; DOGREY 2 C_R	to have contours drawn of the first image and grey-scale of the second image.
> BLC 0 , 0 , c_1 ; TRC 0 , 0 , c_1 C_R	to draw the full plane from channels c_1 through c_2.
> ZINC Δc C_R	to display every $\Delta c^{\,\mathrm{th}}$ channel.
> PIXRANGE B_1, B_2 C_R	to do grey scales from B_1 through B_2 only, clipping the most negative and positive values if desired. The default is the full range of image DOGREY.
> CLEV 0.1 C_R	to plot 0.1 K as the basic contour level.
> LEVS 2.7, 7.4, 20.1, 54.6, 148.4, 403.4 C_R	to do logarithmic contours, starting at 0.27 K.
> CBPLOT 18 C_R	to plot a half-power beam contour in the upper right corner and fill it in.
> LABEL 1 C_R	to label each pane with its coordinate (velocity usually).
> DOTV -1 ; INP	to make a plot file and to review the inputs.
> GO C_R	to run the task.

When KNTR has finished:

> PLVER 0 C_R	to plot the most recent plot file for the image.
> OUTFILE ' ' C_R	to print the plot immediately rather than saving it in a file.
> GO LWPLA C_R	to translate the plot file into PostScript on a suitable printer.

LWPLA offers additional control over fonts, paper size, line width, the grey-scale plotting (if PIXRANGE was not quite right), and number of copies. It can make an "encapsulated PostScript file for inclusion in other documents, such as this *CookBook*.

10.4.6 Backing up your data

The next chapter describes how to help the *AIPS* programming team (with "GRIPEs"), to exit AIPS (with EXIT), to delete your data (with ZAP and ALLDEST), and, most importantly, to back up your data to magnetic tape. Do not assume that data on disk is permanent. Many single-dish data sets are very large even by modern computer standards and only 1–3 can be on disk at a time. Disks can fail and users can make mistakes, so it it is wise to make backups to tape. Read § 3.9 for details on mounting and positioning tapes. Run FITTP or FITAB (§ 11.2.1) on all *uv* data and image files that you wish to keep. Then run PRTTP on your tape to make a record of —and double check — its contents.

10.5 Combining single-dish and interferometer data

We add this section to this chapter with some trepidation since the combination of single-dish data into interferometric imaging is still an area more suited to research than to production. In principle, the problem is fairly simple. You begin by observing a region of sky with a single-dish telescope rather larger than the individual telescopes of the interferometer. From these observations, you make an image which you correct if necessary (*e.g.*, by removing spectral baselines). Then you deconvolve the image removing the convolution of the sky with the beam of the large single-dish telescope. The "sky" observed with the interferometer is the product of the real sky (estimated by your deconvolved image) and the beam of the individual telescopes of the interferometer. Therefore, you multiply your deconvolved image with an image of the single-dish beam and Fourier transform the result. Adjusting the flux scales (usually of the single-dish data), you append or "feather in" the "visibilities" produced by the Fourier transform.

This is a lot of steps and contains several dangers, namely pointing, image alignment, the deconvolution, and the flux re-calibration. *AIPS* can provide you with some help. The imaging and image correction software is described earlier in this chapter. The deconvolution is tricky. Try DCONV first. It attempts an iterative solution of the deconvolution problem in the image plane. If that is not acceptable, try CONVL with OPCODE 'DCON' (in 15JAN96 and later releases). This is a brute force deconvolution that will be very noisy at high spatial frequencies, but these frequencies will be tapered or truncated away later. A third approach is to use PATGN (OPCODE 'GAUS') to make an image of the single-dish beam of the large telescope. APCLN (§ 5.3.7) can then be persuaded to do a Clark image-based Clean; use a small restoring beam. Remember that this image will (must) be tapered in the *uv* plane. It does not have to be beautiful in detail in the image plane.

The next step is to make an image of the interferometer single-dish beam on the same cell size and center as your deconvolved image. Use PATGN with OPCODE 'BEAM' for this. Then multiply the result by the deconvolved image with COMB using OPCODE 'MULT' (§ 7.1). If this produces an image with any blanked pixels, run REMAG to convert the blanks to zeros. Then start trying IM2UV to produce a *uv* data set. Use UVTAPER to weight down longer spacings, FLUX to scale the visibilities, and UVRANGE to omit the outer spacings. (The first two options appear only in 15JAN96 and later releases.) You should use PRTUV, UVPLT, and even UVFLG on the output of IM2UV to make sure that the visibility phases and amplitudes of your single-dish and interferometer data are in reasonable agreement. Finally, combine the two data sets with DBCON and have fun with IMAGR (Chapter 5).