3 BASIC *AIPS* UTILITIES

This chapter reviews some basic *AIPS* utilities with which you should be familiar before you start calibrating data or processing images in *AIPS*. Many of these utilities will appear in later chapters on calibration, image making, and so on. However, in those chapters, these utilities will be explained only briefly.

3.1 Talking to *AIPS*

3.1.1 *POPS* and *AIPS* utilities

When using the *AIPS* system, you talk to your computer through a command processor called *POPS* (for People Oriented Parsing System) that lives in the program `AIPS`. The steps needed to start this basic program are discussed in § 2.2.3. The copy of this program that you get will be called `AIPS`n where n is often referred to as the "*POPS* number" of your session.

The *POPS* command processor is not unique to *AIPS*. It has been present in other programs at the NRAO for many years, and will be familiar to users of the NRAO single-dish telescopes. Chapters 4 to 11 of this *CookBook* give explicit examples of most of the *POPS* commands that a new *AIPS* user needs to know, so we will not give a separate *POPS* tutorial here. The command HELP POPSYM $\subset_R$ will list the major *POPS* language features on your terminal, and Chapter 12 below reviews some advanced features of *POPS*.

As well as providing a command processor, *AIPS* replaces many features of your computer's operating system with its own utilities. This may seem inconvenient at first — you will have to learn the *AIPS* utilities as you go along. You will see the advantage of this approach when you use *AIPS* in a computer that has a different operating system. Your interface to *AIPS* will be almost identical on a VAX, or a Convex C-1, or a Unix-based workstation, or a Cray X-MP. Once learned, your *AIPS* skills will therefore be highly portable.

Lists of the important *AIPS* utilities can be obtained at your terminal by typing ABOUT CATALOG $\subset_R$ and ABOUT GENERAL $\subset_R$. See also Chapter 13 for a relatively recent version of all such category lists.

3.1.2 Tasks

AIPS provides a way for you to set up the parameters for, and then execute, many applications programs sequentially or in parallel. The more computationally intensive programs may take many minutes, hours (or even days) of CPU time to run to completion. They are therefore embodied in *AIPS* "tasks" — programs that are spawned by the `AIPS` program to execute independently and asynchronously (unless you choose to synchronize them). This lets you get on with other work in `AIPS`, while one or more tasks are running. You may spawn, however, only one copy at a time of each task from a given *AIPS* session (*i.e.*, *POPS* number.)

A typical task setup will look like:

> TASK *'task_name'* $\subset_R$	to make *task_name* the default for later commands; note the quote ' marks.
> HELP $\subset_R$	to write helpful text on your terminal about the purpose of the task and about its input parameters.

You will then spend some time setting up parameter values, as in § 3.1.4 below. Then, type

> INP $\subset_R$	to review the parameter values that you have set and
> GO $\subset_R$	to send the task into execution.

You may also specify which task you want to execute by an immediate argument, *e.g.*, GO UVSRT ↵ to execute the task UVSRT. After the GO step, you will watch for messages saying that the task has started executing normally, has found your data, etc., while you get on with other work in *AIPS*.

If you discover that you have started a task erroneously, you may stop it abruptly with

> ABORT ↵	to kill the task named by TASK, or
> ABORT *task_name* ↵	to kill *task_name*.

This will stop the job quickly and delete any standard scratch files produced by it. However, input data files — and output data files that are probably useless — may be left in a "busy" state in your data catalog. The catalog file is described in § 3.3, including methods to clear the "busy" states and to delete unwanted files.

The current full list of tasks may be obtained on your terminal (or workstation window) by typing ABOUT TASKS ↵. Since this list runs for many pages, you may wish to direct the output to the line printer (with DOCRT = -2 ↵) or to consult the list in Chapter 13 of this *CookBook*.

3.1.3 Verbs

Some of the smaller *AIPS* utilities run quickly enough to be run inside the AIPS program rather than being spawned. These "verbs" include simple arithmetic and *POPS* operations, the HELP, ABOUT, INP, and GO commands mentioned already, interactive manipulations of the TV-like display, and many more. Verbs are sent into action simply by setting their input parameters and typing the name of the verb followed by ↵. (The sequence GO *verb_name* ↵ will also work, but a bit more slowly since it also saves the input parameters of the verb for you; see § 3.5 for a further discussion of saved parameters. The sequence TASK *'verb_name'* ; GO ↵ will not work, however.) While a verb is executing, AIPS will not respond to anything you type on the terminal (but it will remember what you type for later use). Just watch out for messages and do what is called for with the TV cursor or terminal. You may, of course, think about what you will do next.

You can list all the verbs in *AIPS* on your terminal by typing HELP VERBS ↵, but the output lists only the names. To find out more, type ABOUT VERBS ↵ which describes what the verbs do. Since this output fills several pages, you may wish to direct it to the line printer (set parameter DOCRT to -2), or to consult the (perhaps dated) list printed in Chapter 13 of this *CookBook*.

3.1.4 Adverbs

AIPS uses "adverbs" (which may be real numbers or character strings, scalars or arrays) to pass parameters to both "verbs" and "tasks." A significant part of your personal time during an *AIPS* session will be spent setting adverbs to appropriate values, then executing the appropriate verbs or tasks. Examples of adverb-setting commands in *AIPS* are:

> CELL 0.5 ↵	to set a single scalar CELL to 0.5
> CELL 1/2 ↵	alternate for above with *POPS* in-line arithmetic
> IMSIZE 512,256 ↵	to set a two-element array IMSIZE to IMSIZE(1)=512, IMSIZE(2)=256
> IMSIZE 512 256 ↵	an alternate for the above if both values are positive
> IMSIZE 256+256,256 ↵	an alternate for the above using *POPS* in-line arithmetic
> UVWTFN 'NA' ↵	to set a string variable UVWTFN to the value NA
> LEVS 0 ↵	to set all elements of the 30-element array LEVS to zero
> LEVS = -2,-1,1,2,3,4,5 ↵	to set LEVS(1)=-2, LEVS(2)=-1, LEVS(3)=1, etc. The = avoids in-line arithmetic that would otherwise subtract 2 from LEVS(1)

> LEVS = -2,-1 1 2 3 4 5 ↵ — an alternate for the above; the comma avoids in-line arithmetic that would otherwise set `LEVS(1) = -3`

Many *AIPS* tasks will assume sensible "default" values for adverbs that you choose not to (or forget to) specify. Some adverbs cannot be sensibly defaulted; these should be clearly indicated in the appropriate help information. You may review the current input parameters for any *AIPS* task or verb on your terminal by typing

> INP ↵ — to review the parameters for task `TASK`, or

> INP *task_name* ↵ — to review the adverbs for task *task_name*.

Any adverbs which you have set to *à priori* unusable values will be followed on the next line by a row of asterisks and an informative message. Details of the input parameters used by any *AIPS* verb or task can be obtained on your terminal by typing:

> HELP ↵ — to review the parameters for task `TASK`, or

> HELP *task_name* ↵ — for task *task_name*

> HELP *verb_name* ↵ — for verb *verb_name*

> HELP *adverb_name* ↵ — for adverb *adverb_name*

See § 3.8 below for more methods of obtaining on-line help with *AIPS*.

You can list all the adverbs in *AIPS* on your terminal by typing HELP ADVERBS ↵, but the output lists only the names. To find out more, type ABOUT ADVERBS ↵ which describes what the adverbs do. Since this output fills several pages, you may wish to direct it to the line printer (set parameter `DOCRT` to -2), or to consult the (perhaps dated) list printed in Chapter 13 of this *CookBook*.

3.2 Your *AIPS* message file

`AIPS` and all tasks talk back to you by writing messages to a disk file called the "message file" and/or by sending them to you on the appropriate "message monitor." Simple instructions and progress messages usually go only to the monitor; very few (if any) messages go only to the file. For `AIPS` itself, the message monitor is always the workstation window or terminal into which you are typing your commands. For the tasks, the monitor can also be a separate terminal (on well-equipped, but old, systems) or a second workstation window under control of the *AIPS* dæmon process `MSGSRV`. You can control whether or not you get the message server window by the setting of a Unix environment variable. Enter

> HELP MSGSRV ↵ — for details.

In the Charlottesville and most non-NRAO *AIPS* installations, you get a message server by default. The AOC has chosen — wrongly — to make the default be no message server. You may also control the size and appearance of the message server with parameters in the X-Windows `.Xdefaults` file. These parameters are also listed in by HELP MSGSRV ↵.

You may review the contents of the message file by typing PRTMSG ↵ at the > prompt at your terminal. `PRTMSG` is an example of an *AIPS* "verb" — it does not need a GO from you to execute, and it is *not* shed from your terminal. Each message in the file has, associated with the text, the time, task name, *POPS* number, and the priority of the message. The priority codes range from 0 for user input to 2 for "unimportant" messages to 5 for "answers" and other significant normal messages to 8 for serious error messages. The `PRTMSG` verb has adverbs to let you select either the printer or your window or terminal for the display and to let you control which messages will be displayed. For example, to set the minimum priority level for messages to be displayed, type:

> PRIORITY *np* ↵ — where *np* is the desired minimum level,

before running `PRTMSG`; then only messages at this level or above will be listed on the printer or terminal. If *np* is ≤ 5, then messages at level 0 are also shown. `PRTMSG` has further adverbs to limit the output by program

name (PRTASK, uses minimum match), message age (PRTIME as upper limit to the age), and AIPS number (PRNUM). Note that PRNUM must be your AIPS, *i.e.*, POPS, session number, not your user identification number. The choice of the output device is made with

> DOCRT -1 ↵ to select the line printer

> DOCRT 1 ↵ to select the terminal at its current width ≥ 72 characters

> DOCRT *nc* ↵ to select the terminal at width *nc* characters: $72 \leq nc \leq 132$.

The wider you can make your window display, up to 132 characters, the more information AIPS can put on a line. You may change the line printer selection with PRINTER.

PRTMSG does not delete messages from your message file. Use:

> CLRMSG ↵ to delete messages and to compress the message file.

CLRMSG supports adverbs like those of PRTMSG, except that the deletion is of messages older than PRTIME and the printing is of messages younger than PRTIME seconds ago. Old messages are automatically deleted from your message file when you EXIT from AIPS. (The time limit for "old" messages is set by your local AIPS Manager. Usually, it is about 3 days.)

3.3 Your AIPS data catalog files

Your *uv* data sets and images are your largest inputs to, and outputs from, AIPS. A summary record of all your disk data sets (*uv* data, images, beams and temporary "scratch" data created by active tasks) is kept in your disk catalog files (one per disk). To interrogate this catalog file, use:

> INDI 0 ; MCAT ↵ to list all images on all disks, or

> INDI 0 ; UCAT ↵ to list all *uv* data sets on all disks.

A complete listing of the catalog file, which may be printed with PRTMSG, can be generated by:

> CLRNAME ↵ to reset INNAME, INCLASS, INSEQ, INTYPE, and INDISK,

> CATALOG ↵ to generate the listing.

which will list all of your disk data sets. To limit the listing to a particular name, class, sequence number, type, and/or disk, use a combination of the adverbs INNAME, INCLASS, INSEQ, INTYPE, and INDISK. The INNAME and INCLASS adverbs allow a rather powerful wild-card grammar; type HELP INNAME ↵ for details. Unless you want a hard copy, it is faster to use MCAT and UCAT, although they respond only to the INDISK adverb. A typical listing looks like:

```
CATALOG ON DISK  1
CAT USID MAPNAME      CLASS   SEQ  PT     LAST ACCESS      STAT
 18   76 3C166L5OK   .IIM001.    1 MA 27-OCT-1996 22:30:18
 19   76 3C166L5OK   .IBM001.    1 MA 27-OCT-1996 23:02:14
 22   76 3C166L5OK   .IIM001.    2 MA 28-OCT-1996 15:30:45
CATALOG ON DISK  2
CAT USID MAPNAME      CLASS   SEQ  PT     LAST ACCESS      STAT
 22   76 1200+519    .IIM001.    1 MA 01-NOV-1996 23:50:10
 23   76 1200+519    .IBM001.    1 MA 01-NOV-1996 23:59:58
 24   76 1200+519    .QIM001.    1 MA 28-OCT-1996 00:10:10
 25   76 1200+519    .UIM001.    1 MA 28-OCT-1996 00:19:19
 28   76 1200+519    .ICL001.    1 MA 02-NOV-1996 00:35:20 WRIT
 31   76 SCRATCH FILE.IMAGR1.    1 SC 02-NOV-1996 00:35:37 WRIT
 32   76 SCRATCH FILE.IMAGR1.    2 SC 02-NOV-1996 00:35:39 WRIT
CATALOG ON DISK  3
CAT USID MAPNAME      CLASS   SEQ  PT     LAST ACCESS      STAT
  2   76 3C138 A C   .UVSRT .    1 UV 22-OCT-1996 12:56:50
 36   76 1200+519    .UVXY  .    1 UV 02-NOV-1996 00:32:50 READ
```

```
 37   76 1200+519     .IMAGR .     1 UV 02-NOV-1996 00:34:25 WRIT
```

This user (identification number 76) has eight image files, three on disk 1 and six on disk 2. He also has two sorted *uv* data sets and an IMAGR *uv* work file on disk 3. There are two scratch (temporary) files on disk 2 which were created by IMAGR running out of AIPS1 (this determines their IMAGR1 classname). Image data files (images and beams) are distinguished by the type code MA. The *uv* data files are distinguished by the type code UV and scratch files by type SC.

Note that this user has encoded useful information other than the source name into the image file names on disk 1. These images were of 3C166 at L band with 50 kilo-wavelength (*uv*) taper. Such information is also carried in AIPS history files (see § 3.4 below), but it is often useful to place it at a level where CAT can see it. The user also gave the UVSRT file in slot 2 on disk 3 a name that encodes the source name (3C138), the VLA configuration (A), and the observing band (C). Careful choice of AIPS filenames can save much other bookkeeping. The file name can be any valid string up to 12 characters long. Also note how SEQ numbers distinguish different versions of a file with the same name; this and the global variables in AIPS are helpful features when doing iterative computations such as self-calibration.

3.3.1 Speedy data file selection

Each catalog entry has an identification number called the "catalog slot number". The CAT column at the left of the listing above shows these catalog numbers. They can be used to set up inputs quickly for AIPS programs that read cataloged disk data sets. Use:

> INDI *n1* ; GETN *ctn1* ↵ where *n1* selects the disk and *ctn1* is the catalog slot number.

The verb GETNAME (abbreviated through minimum match as GETN above) sets the adverbs INNAME, INCLASS, INSEQ, and INTYPE used by many tasks and verbs. Some tasks require a second, a third and even a fourth set of input image name adverbs. For these, use:

> IN2D *n2* ; GET2N *ctn2* ↵ to set the second set, and

> IN3D *n3* ; GET3N *ctn3* ↵ to set the third set.

> IN4D *n4* ; GET4N *ctn4* ↵ to set the fourth set.

The verb GETONAME (GETO for minimum match) sets the adverbs OUTNAME, OUTCLASS and OUTSEQ to those of a pre-existing output file. GETO is particularly useful with calibration tasks that copy extension tables (*e.g.*, CL or FG tables) from one database to another or for restarting an image deconvolution.

3.3.2 Catalog entry status

Note that several catalog slots on disks 2 and 3 in our sample catalog listing above do not have blank entries in the STAT column. This listing was made while the user was running a Clean deconvolution with IMAGR on the sorted *uv* data set in slot 36 — this *uv* data file is opened for READing. The Clean image file, ICL001 in slot 28, and the scratch and IMAGR files are opened for WRITing. Procedures that attempt to read files which are opened for writing, or vice versa, will be rejected with appropriate error messages. You must therefore note any non-blank entries in the STAT column carefully. In some situations (mainly involving system crashes or abortion of tasks [§ 3.1.2]) files may be left in READ or WRIT status indefinitely. If this happens, you may reset the file status with CLRSTAT ↵ after issuing the appropriate INDISK and GETNAME. Note that a WRIT status on a file which is not, in fact, being used at present probably indicates that the data in the file have been corrupted. Such files should usually be removed from your catalog by first clearing the file status with GETN *nn*; CLRST ↵ then deleting them with ZAP ↵.

Before using a data set as input to an AIPS task, check that the data set has a clear status. (It is possible to let two tasks read the same data at the same time, but this is not recommended as it will usually slow execution.) Also note the data set's disk number and its ordinal number in the catalog, as these are useful for GETN, GET2N, etc.

3.3.3 Renaming data files

Files may be renamed, after they have been cataloged, using the *AIPS* verb RENAME. Typical inputs might be:

> INDI 2 ; INNA '1200+519' ↵	to select disk 2 and set the input (old) name.
> INCL 'IIM001' ; INSEQ 1 ↵	to set the rest of the input name adverbs, *i.e.*, to select the file in slot 22 on disk 2 in the example above.
> OUTN '1200+51 15K' ; OUTSEQ 2 ↵	to set desired output name and sequence number.
> INP RENAME ↵	to review the inputs.
> RENAME ↵	to rename the I image to '1200+51 15K' and reset its sequence number to 2.

Two verbs can be used to alter the catalog numbers of files. RENUMBER moves a file to an empty, user-specified slot; a one-line command to do this would be SLOT *n*; RENUM ↵ where *n* is the new slot number. RECAT compresses the catalog (*i.e.*, it removes gaps in the catalog numbers) without changing the order of the entries in the catalog.

3.3.4 Header listings

Every image or *uv* data set in *AIPS* has an associated header file that contains information needed to describe the data set in detail.

The header also contains information on the number of extension files of each type that have been associated with the data set. The most important file extensions that can be associated with *AIPS* image data are the HIstory file described below, the CC or Clean component files (see Chapter 5) and the PLot files and SLice files (see Chapter 6).

Multi-source *uv* data files may have many extensions (see Chapter 4). The most important are the HIstory file, the ANtennas file (subarray geometric data, date, frequency and polarization information, *etc.*), the BP (bandpass) file for bandpass calibration data, the CL (calibration) file for calibration and model information, the FQ (frequency) file for frequency offsets of the different IFs, the FG (flag) file for editing information, the NX (index) file (which assists rapid access to the data), the SN (solution) file for gain solutions from *AIPS* calibration routines, and the SU (source) file with source-specific information such as name, position, and velocity. Chapter 4 describes the use of these extensions in some detail.

You can list the header file of any catalog entry on your terminal by following the GETNAME step above with

> IMHEAD ↵	for a detailed listing, or
> QHEAD ↵	for a shorter listing.

The output of IMHEAD and QHEAD can also be printed using PRTMSG (at PRIORITY 2).

Output from IMHEAD on a multi-source *uv* data set might look like:

```
Image=3C345     (UV)         Filename=Z17G1_A     .MULTI .   1
Telescope=SBLNKGYO           Receiver=VLBI
Observer=FAP                 User #= 1353
Observ. date=27-FEB-1991     Map date=13-JUN-1995
# visibilities    112813     Sort order  TB
Rand axes: UU-L  VV-L  WW-L  BASELINE  TIME1  WEIGHT  SCALE
           SOURCE
-------------------------------------------------------------
Type    Pixels   Coord value  at Pixel    Coord incr   Rotat
```

```
COMPLEX      1   1.0000000E+00    1.00 1.0000000E+00    0.00
STOKES       4  -1.0000000E+00    1.00-1.0000000E+00    0.00
FREQ       128   2.2228990E+10   63.50 5.0000000E+05    0.00
RA           1    16 41 17.608    1.00      3600.000    0.00
DEC          1    39 54 10.820    1.00      3600.000    0.00
------------------------------------------------------------
Maximum version number of extension files of type SU is   1
Maximum version number of extension files of type CL is   3
Maximum version number of extension files of type HI is   1
Maximum version number of extension files of type AN is   1
Maximum version number of extension files of type NX is   1
Maximum version number of extension files of type FG is   1
Maximum version number of extension files of type SN is   1
```

Output from IMHEAD on an image file might look like:

```
Image=3C219     (MA)         Filename=3C219-BC-6  .ICL001.   1
Telescope=VLA                Receiver=
Observer=BRID                User #=   76
Observ. date=06-SEP-1992     Map date=18-APR-1994
Minimum=-1.89720898E-04      Maximum= 5.05501366E-02 JY/BEAM
------------------------------------------------------------
Type    Pixels   Coord value  at Pixel    Coord incr   Rotat
RA---SIN   510    09 17 50.662  263.00     -0.300000    0.00
DEC--SIN   640    45 51 43.555  294.00      0.300000    0.00
FREQ         1   4.8726000E+09    1.00 2.5000000E+07    0.00
STOKES       1   1.0000000E+00    1.00 1.0000000E+00    0.00
------------------------------------------------------------
Map type=NORMAL              Number of iterations=   50000
Conv size=   1.40 X   1.40   Position angle=   0.00
Observed RA   09 17 50.600    DEC  45 51 44.00
Maximum version number of extension files of type HI is   1
Maximum version number of extension files of type PL is   5
Maximum version number of extension files of type SL is   1
```

Both QHEAD and IMHEAD list the maximum version numbers of the table extension files associated with a data set. Because you may acquire many versions of such tables during calibration, these verbs are often invoked during calibration in *AIPS*.

3.4 Your *AIPS* history files

Every *uv* and image file has an associated "history", or HI, file. This HI "extension" of the data set stores important information about the processing done so far on the data in the file. Every *AIPS* task and verb that alters either the data or the file header will record its key parameters in the history file. The history file is written to tape when you use FITS format, so you can preserve it for reference in later *AIPS* sessions or when sending data to colleagues.

In general, each "card" in the history file begins with the task or verb name. It then gives one or more of the input adverb values it used (*i.e.*, the defaults are filled in). All or parts of the file may be displayed on your terminal or printed on the line printer. For example, use:

> INDISK *n* ; GETN *ctn* ↵ to select the file to be displayed.

> PRTASK 'UVMAP' ↵ to examine only history information from UVMAP.

> DOCRT 1 ↵	to direct the display to your terminal, using its full width.
> PRTHI ↵	to print the UVMAP history.
> PRTASK ' ' ; DOCRT FALSE ↵	to select all history cards and direct the output to the line printer.
> PRTHI ↵	to print the full history file.

There are several (legitimate) reasons why you might wish to edit your history files. Repetitive self-calibration cycles, or image combinations, can lead to very long and very repetitive histories which could be substantially shortened with no real loss of information. Also some entries in the history file may become obsolete by, say, the deletion of plot files. The verb STALIN allows you to send a range of history lines to Siberian salt mines (*i.e.*, delete) by number with some selectivity and, optionally, interactive confirmation of each deletion. You may, of course, simply wish to add information to the history file. The verb HINOTE can be used to append one line, given by the adverb COMMENT, or many lines, typed in interactively, to the history file. Even more powerfully, the verb HITEXT allows you to write your history file to an external text file (see § 3.10.1). You may edit that file with your favorite Unix file editor and then read it back, writing your edited file into any *AIPS* history file you want (with verb HINOTE).

3.5 Saving and restoring inputs

All input and output parameters ("adverbs") are global throughout AIPS. When an adverb value is specified for, or set by, a task or verb, it remains at that value for any other task or verb that uses an adverb of the same name (until you change it). This global nature of the *AIPS* adverbs is useful in most cases. It can, however, be inconvenient — especially if you are taken by surprise because you have not reviewed the adverb values before running a task. Before running any task or verb, check your current input adverbs carefully with:

> INP *name* ↵	where *name* is the program name, or
> INPUTS *name* ↵	to write the input values to the message file.
> QINP ↵	to resume the previous INP or INPUTS with the page last displayed.

Some tasks have multiple pages of input parameters. QINP allows you to change a parameter on a page, review that page and then go on to the next page without having to view the first pages over again. Some verbs and a few tasks have *output* adverbs. Unless they are also used on input, they will not appear when you do INP or INPUTS. After running such verbs and tasks, do

> OUTPUTS *name* ↵	to view the output values and write them to the message file.

To reset all adverbs for a particular task or verb to their initial values, without changing any other adverbs or procedures, enter

> DEFAULT *name* ↵	to reset the values for *name*.
> DEFAULT ↵	to reset the values for the verb or task named in the TASK adverb.

You can save all adverbs you have specified for *AIPS* to disk at any time by typing:

> SAVE *aaaaa* ↵	where *aaaaa* is any string of up to 12 characters.
> GET *aaaaa* ↵	will restore these inputs later.

These commands save or restore your entire AIPS "environment". For this reason, GET must be the only command on the input line; SAVE may appear with other commands, but will be executed before *any* of the other commands on the line. Thus, the sequence INNAME '3C123' ↵ INNAME 'BLLAC' ; SAVE BLLAC ↵ will save a 3C123 environment, not a BLLAC one. AIPS automatically saves your environment in a disk area called LASTEXIT whenever you use the EXIT, KLEENEX, or RESTART commands. The command GET LASTEXIT

is automatically executed whenever you start up the AIPS program again on the same machine. Thus, you retain your own AIPS environment from one use of AIPS to the next. To obtain a null version of the adverb values and of the rest of the AIPS environment, type:

> RESTORE 0 ↵

There is also one temporary area for saving your AIPS environment. To save your inputs temporarily, type:

> STORE 1 ↵ to save your inputs in area 1, and

> RESTORE 1 ↵ to recover the inputs you previously stored in area 1.

When new verbs and adverbs are created at your site, your old SAVE files will not know about them. Beginning with the 15JAN96 release, you may update the old files with the sequence:

> GET *aaaaa* ↵ to recover the old SAVE area.

> COMPRESS ↵ to get the new basic vocabularies without losing your adverb values and procedures.

> SAVE *aaaaa* ↵ to save the updated area for later; use the full name of the SAVE area here.

The list of SAVE areas may be reviewed with the verb SGINDEX. In 31DEC02, a SAVE area may be written as a RUN file (§ 3.10.2) if you first GET the area and then use SG2RUN.

The input adverb values associated with a task or a verb can be stored by the command:

> TPUT *name* ↵ where *name* is the verb or task name.

and retrieved by the command:

> TGET *name* ↵

TPUT and TGET allow you to avoid, to some extent, the global nature of the adverb values in AIPS. This is sometimes advantageous. Whenever a task (or a verb, for that matter) is executed by the verb GO, TPUT runs automatically. TGET will therefore recover the last set of input adverbs used to execute the task, unless you deliberately overwrite them with a TPUT of your own. Note that AIPS will complain if you try to TGET input adverbs for a task for which no TPUT has previously been run (either manually or automatically). You must "put" before you can "get." TGINDEX will show you what tasks have been TPUT and when. In 31DEC00, VPUT, VGET, and VGINDEX allow you to save, recover, and list task-specific adverbs from up to 35 completely user-controlled storage areas.

You can change between versions of AIPS software once you are inside AIPS by typing

> VERSION '*version*' ↵ where *version* is one of OLD, NEW or TST

Alternatively, you may use this command to access a private version of a program in some other area — see § 12.2.2. Note that toggling between different versions of AIPS is possible only when the data formats are the same. Unfortunately, 15APR98 is no longer compatible with previous versions of AIPS. Note also, that you are toggling between different versions of tasks, not the verbs within the AIPS program. That version is selected when you start the program (§ 2.2.3) and can be changed only by exiting and start anew.

3.6 Monitoring disk space

Since the 15APR92 release of AIPS, the availability of data areas via NFS has vastly increased the amount of disk space accessible from a given AIPS session. The da= command line option to the aips command allows you to specify "disks" (data areas) from many hosts in addition to the current host, subject to a maximum of 15 disks per session. Note, however, that the BADDISK adverb has a limit of 10 disks. Thus, if more than 10 disks are accessed via NFS, you will not be able to prevent one or more from being used for scratch files. This can be important. Reading data over NFS is relatively efficient, but writing data is not. Even file creations (under Unix) require the writing of zeros to the whole file in order to guarantee later access to the requested space. Over NFS, this can be a slow process. For example, if user disk 1 is accessed

via NFS, then every line of the message file must be written with NFS, a process which has been observed to require about one second of real time per message!

Another aspect of the new disk allocation system is a scheme by which the local AIPS Manager may restrict the availability of some disk areas to a set of user numbers, specified on a disk-by-disk basis. Managers usually use this tool to set aside most disks on a staff member's workstation for his/her sole use and to reserve space for visitors or other special projects on "public" workstations on a case-by-case basis. Use the FREE verb within AIPS to show you the space used and available on all disks for your session and also to show whether or not that space is reserved. The right-most column of FREE's output will show Alluser if the space is not reserved, Resrved if you are one of the users for which the space is reserved, Not you if you are not allowed to use the space, and Scratch if the space is to be used only for scratch files. Use FREE often to keep track of how much space is available and where the space can be found.

Disk space is still generally at a premium. If more than one user has access to the disk areas you are using, then another useful tool for monitoring disks is the AIPS task called DISKU. To run it, type

> USER 32000 ; INDISK 0 ↵ to get all disks and users.

> GO DISKU ↵ to run the AIPS disk user task.

This will (eventually) list on the AIPS monitor (and the message file) the amount of data space in use by each user for all AIPS disks. Identify the worst disk hogs and apply appropriate peer pressure. If you are, mysteriously, the culprit on some disk, then

> USER 0 ; INDISK *n* ↵ where *n* is the mysteriously eaten disk

> DOALL 1 ; GO DISKU ↵ to run the job

will give you the size of every one of your files on the specified disk. Armed with this information, you may be able to take appropriate action upon your own data.

Sometimes the available disk space has been eaten up by AIPS scratch files that are no longer in use. Tasks that abort while executing (and other mysterious events) may produce this situation. To delete all your scratch files, except those for tasks which are still running, type:

> SPY ↵ to see which tasks are running.

> SCRD ↵ to delete the files.

SCRDEST is run automatically whenever EXIT, KLEENEX, RESTART, or ABORT *task_name* are executed. Note that the imaging and deconvolution tasks IMAGR, MX, UVMAP, APCLN, HORUS and VTESS, the data editor TVFLG and the sorter UVSRT may create large scratch and "work" files, so you should watch for "dead" copies of scratch and work files from these programs in your disk catalog. Both MCAT and UCAT will show scratch files as well as the requested file type. Note too that, if you are using more than one computer on a given disk area, only those scratch files created by your current computer will be deleted when you run the SCRD verb. Work files have to be deleted individually since they can be still of use after the task which created them has finished.

The verb TIMDEST destroys all user data sets that have not been used in some minimum time interval. In unmodified versions of AIPS, this time interval is 14 days. TIMDEST also deletes messages over 3 days old from all users' message files. The adverbs of TIMDEST allow you to request less stringent cutoffs. Your local AIPS Manager may set other limits on the time ranges. TIMDEST may take a long time to run if disk usage on your computer is not well policed. This is a design "feature" intended to promote regular use of TIMDEST by authorized AIPS Managers rather than by individual users. However, you are welcome to use it. Be aware, however, that "all" in the sentences above includes you.

Chapter 11 of this *CookBook* tells you how to backup or delete your own data to relieve disk crowding. At present, all other methods for managing disk space involve system-dependent commands of one sort or another. Since these may have unexpected consequences they are not recommended.

3.7 Moving and compressing files

Two AIPS tasks are frequently used to move files from one disk to another with options to reduce the file size. They are SUBIM, used on images, and UVCOP, used on *uv* data sets. SUBIM uses the adverbs BLC and TRC to select a portion of the input image and XINC and YINC to select a pixel increment through the portion. If these adverbs are defaulted (set to 0), the entire image is copied. Clean component, history, and other table extension files are copied as well, but plot and slice extensions are not. Similarly, UVCOP uses a wide range of adverbs to select which IFs, channels, frequency IDs, times, antennas, and sources are to be copied. If all of these adverbs are defaulted (set to 0 or blank), then all data are copied except (optionally) for completely flagged records. A flag table may also be applied to the data. including flag tables too large to be handled by most tasks. With extensive data editing, UVCOP may produce a rather smaller data set even with no other selection criteria. Antenna, gain, and other table extension files are copied, but plot files are not. The task MOVE may be used to copy all files associated with a catalog number (without modification) to another disk or to another user number.

3.8 Finding helpful information in AIPS

Much AIPS documentation can be displayed on your terminal by typing HELP *word* ↵, where *word* is the name of an AIPS verb, task or adverb. The information given will supplement that given in the INPUTS for a verb or task. It is the only source of information on the adverbs. Type XHELP *word* ↵ to display the help file in your WWW browser with links to adverbs from task help files.

To print the HELP information on your line printer, set DOCRT = -1 and enter EXPLAIN *word* ↵ instead. (Using DOCRT = 1 with EXPLAIN will send the output to your terminal screen.) For the more important verbs and tasks, EXPLAIN will print extra information, not shown by HELP about the use of the program, with detailed explanations, hints, cautions and examples.

HELP may also be used to list the names of all POPS symbols known to AIPS by category, an operation helpful when you can't remember the name of something. Type:

> HELP ADVERBS ↵	to get a list of all adverbs in the symbol table
> HELP ARRAYS ↵	to get a list of all array adverbs in the symbol table
> HELP REALS ↵	to get a list of all real adverbs in the symbol table
> HELP STRINGS ↵	to get a list of all character string adverbs in the symbol table
> HELP VERBS ↵	to get a list of all verbs, pseudoverbs, and procedures in the symbol table
> HELP PSEUDOS ↵	to get a list of all pseudo verbs in the symbol table
> HELP PROCS ↵	to get a list of all procedures in the symbol table

In the past, AIPS contained a range of general HELP files which purported to list all verbs and tasks in various categories. Since these were maintained by hand, they were essentially never current and complete. That entire system has been replaced by the verbs ABOUT and APROPOS to be discussed below. A few general help files do remain, and they may even be relatively current. A list of these may be found by typing:

> HELP HELP ↵	for help on HELP.

A few general help files remain. They are POPSYM (symbols used in POPS interpretive language), WHATSNEW (major changes in AIPS since the last update — actually maintained in 31DEC00), NEWTASK (writing and incorporating a new task into AIPS), and PANIC (solutions to common problems). Relatively recent versions of these files are listed in Chapter 13 of this *CookBook*.

The HELP verb is very useful, but only if you know that the function you want exists in AIPS and know its name. Two new functions have appeared in AIPS to assist you in this search. The first of these, APROPOS,

searches all of the one-line summaries and keywords of all AIPS help files for matches to one or more user-specified words. For example, type

> APROPOS CLEAN ↵ — to display all keyword and 1-line summaries of help files containing words beginning with "clean" (in upper and/or lower case), and

> APROPOS 'UV PLOT' ↵ — note the quote marks which are required if there are embedded blanks, or

> APROPOS UV,PLOT ↵ — to display all keyword and 1-line summaries of help files containing *both* words beginning with "uv" and words beginning with "plot."

The text files used by APROPOS are maintained by the AIPS source code maintenance (check-out) system itself. As a result, they should always be current. Of course, the quality of the results depends on the quality of the programmer-typed one-line and keyword descriptions in the help files. These were not regarded previously as important, and hence are of variable quality.

The second new method for finding things in AIPS is the verb ABOUT. Type

> ABOUT *keyword* ↵ — to see a list of all AIPS tasks, verbs, adverbs, etc. which mention *keyword* as one of their "keywords."

You need only type as many letters of *keyword* as are needed for a unique match. The source-code maintenance system is used to force all help files to use only a limited list of primary and secondary keywords. Software tools to update the list files have also been written, and are used at least once with every AIPS release. The list of categories recognized is as follows (where only the upper-case letters shown in the name are actually used):

```
ADVERB        POPS symbol holding real or character data
ANALYSIS      Image processing, analysis, combination
AP            Tasks using the "array processor"
ASTROMETry    Accurate position and baseline measurements
BATCH         Running AIPS tasks in AIPS batch queues
CALIBRATion   Calibration of interferometer uv data
CATALOG       Dealing with the AIPS catalog file
COORDINAtes   Handling image coordinates, conversions
EDITING       Editing tables, uv and image data.
EXT-APPL      Access to extension files (tables)
FITS          FITS format for data interchange
GENERAL       General AIPS utilities
HARDCOPY      Creating listings and displays on paper
IMAGE-UTil    Utilities for handling images
IMAGE         Transforming of images
IMAGING       Creation of images: FFT, Clean, ...
INFORMATion   General lists and user help functions
INTERACTive   Functions requiring user interaction
MODELING      Model fitting to uv or image data
OBSOLETE      Functions slated for removal
ONED          Functions for one-dimensional image slices
OOP           Tasks coded with object oriented principles
OPTICAL       Functions of interest for optical astronomy data
PARAFORM      Skeleton tasks for use in building new tasks
PLOT          Displays of image and uv data
POLARIZAtion  Calibration, analysis, display of polarization
POPS          Aspects of the AIPS' user language POPS
PROCEDURe     Creation of and available procedures
PSEUDOVErb    Pseudoverbs in the POPS language and AIPS
```

```
RUN           Creation of and available RUN files
SINGLEDIsh    Functions of interest for single-disk radio data
SPECTRAL      Functions for spectra-line and other 3D data
TABLE         AIPS table extension files
TAPE          Use of magnetic tapes
TASK          AIPS tasks - available asynchronous functions
TV-APPL       Tasks using the TV display
TV            Basic functions on the TV display
UTILITY       Basic functions on tables, uv and image data
UV            Functions dealing with interferometer uv data
VERB          Synchronous functions inside the AIPS program
VLA           Functions of particular interest for the VLA
VLBI          Functions of particular interest for very long
              baseline data.
```

A variety of synonyms are also recognized. Besides those that are merely spelling variants, the currently accepted synonyms are

```
FILES       ->  CATALOG          POSITION   ->  COORDINATES
FLAGGING    ->  EDITING          EXTENSION  ->  EXT-APPL
PRINTING    ->  HARDCOPY         PRINTER    ->  HARDCOPY
MAP         ->  IMAGE            MAP-UTIL   ->  IMAGE-UTIL
MAPPING     ->  IMAGING          LANGUAGE   ->  POPS
CUBE        ->  SPECTRAL         LINE       ->  SPECTRAL
VISIBILITY  ->  UV               VLBA       ->  VLBI
PARAMETERS  ->  ADVERB           HELPS      ->  INFORMATION
SLICE       ->  ONED
```

Even veteran *AIPS* users should use HELP WHATSNEW C_R when a new release of *AIPS* is installed on their computer. When it is current, this file provides brief descriptions of recent developments in *AIPS*. Reading it may bring pleasant surprises and avoid unpleasant ones! More detailed descriptions of new developments in *AIPS* can be found in the *AIPSLetter* published by the NRAO with each *AIPS* software release. An *AIPS* Memo series is published by the NRAO with details of various aspects of the implementation of, and planning for, *AIPS*. Advanced users may also wish to receive, and contribute to, the *AIPS* electronic mail forum — BANANAS. There is also an electronic news group called `alt.sci.astro.aips` devoted to *AIPS* matters. This *AIPS CookBook*, many of the *AIPS* Memos, and various other publications of the *AIPS* group are available via anonymous `ftp` (at `baboon.cv.nrao.edu`) and via the Internet and the "World-Wide Web" starting with "URL" (Universal Resource Location) `http://www.cv.nrao.edu/aips/aips-home.html`).

Your local *AIPS* Manager probably receives the *AIPSLetter*, *AIPS* Memos, and BANANAS and can make information from them available at your site. He/she should also be aware of the electronic means of information retrieval, and be able to help you use them. If this is not the case, write to the *AIPS* Group (at NRAO, 520 Edgemont Road, Charlottesville, VA 22903–2475) or send electronic mail to **aipsmail@nrao.edu** for further information about these services.

3.9 Magnetic tapes

Large volumes of data are usually brought into, and taken away from, *AIPS* using magnetic tape. The tape drives assigned to you are displayed as you start up AIPS, *e.g.*,

```
Tape assignments:
Tape 1 is IBM 9-track model 9348-012 on LEMUR
```

```
Tape 2 is HP 9-track model 88780B on LEMUR
Tape 3 is IBM 7208/001 Exabyte 8200 (external) on LEMUR
Tape 4 is ZZYZX 1.3Gb DAT (left, Model# ZW/HT1420T-CC6) on LEMUR
Tape 5 is ZZYZX 1.3Gb DAT (right; both 150mb personality) on LEMUR
Tape 6 is IBM Exabyte 8200 (internal) on LEMUR
Tape 7 is REMOTE
Tape 8 is REMOTE
```

for the heavily loaded, and now obsolete, IBM called `lemur`. The tape numbers you see above correspond to `AIPS` adverb `INTAPE` values of 1, 2, 3, and so on. The description is meant to give you some idea of which box or slot is to receive your tape. Most of the drives will have a label on them identifying their *AIPS* tape number. If in doubt, ask a local guru for help. The last two tape "drives," called `REMOTE`, will be discussed separately below.

In case you forget this list, the verb `TAPES` will show it to you. `TAPES` is even capable of going out on the Internet and asking what devices are available to an *AIPS* user at the computer specified by the `REMHOST` adverb (if it is running `TPMON`)!

3.9.1 Hardware tape mount

On some *AIPS* systems, tapes are handled by designated operators. Before mounting tapes, read Appendix Z (for NRAO sites) or obtain directions from your local *AIPS* Manager or operators for methods by which tapes are to be handled. Most *AIPS* systems, however, are on the self-service plan. In that case, the simplest thing to do is to find a drive of the required type without a tape in it. There is no way in most Unix systems (certainly not in AIX or SunOS) of reserving a tape drive globally for your exclusive use, though once you have it `MOUNT`ed from within `AIPS`, no other `AIPS` user can access it. It is most efficient to use a tape drive directly connected to your computer (and hence listed as you started up `AIPS`). However, any "AIPSable" drive will do. Mount the tape physically on the drive following the mounting instructions in Appendix Z or those posted at your installation for the particular kind of tape drive. For half-inch (nine-track) tapes, don't forget to insert a write ring if you intend to write on the tape or to remove any write ring if you intend only to read the tape. Exabyte and DAT tapes have a small slide in the edge of the tape which faces out which takes the place of the write ring of 9-track tapes. For 8mm (Exabyte) tapes push the slide to the right (color black shows) for writing and to the left (red or white shows) for reading. With 4mm DAT tapes, the slide also goes to the right for writing (but white or red shows) and to the left for reading (black shows). Note the identification number m marked on the drive you are using, as you will need to provide that number to the software for mounting and dismounting the tape and for executing *AIPS* tasks which read or write tape.

3.9.2 Software mounting local tapes

After you have the tape physically mounted on the tape drive, *AIPS* must also be told that you have done this and which tape drive you have chosen. This step is called a "software tape mount." It is necessary to wait until the mechanism in the drive has "settled down", *i.e.*, when the noises and flashing lights have stopped, before you can do the software mount. This operation is done from inside `AIPS` by typing:

> INTAPE *m* CR — to specify the drive labeled *m*.
> DENSITY *dddd* CR — to set the density to *dddd* bpi if needed.
> MOUNT CR — to mount the tape in software.

Read any messages which appear on your terminal carefully since they report the success, failure, and/or limitations of the operation. The meaning of "density" with modern magnetic tape devices is mostly a

matter of convention. With half-inch, 9-track tapes, AIPS understands the usual 800, 1600, and 6250 bytes per inch densities. A special value for density, 22500, is taken to mean high density (5-Gbyte) mode on 8mm (Exabyte) tapes. You must set the DENSITY adverb to one of these magic values, but in many cases it does not matter which one you use.

Please dismount the tape as soon as you are finished with it, using:

> INTAPE *n* ; DISMO ↵ to dismount a tape from the drive labeled *n*.

The dismount verb should cause the tape to be rewound and, in most cases, ejected from the drive. Please remove the tape from the tape drive promptly so that others may use the drive. Note that exiting AIPS under most circumstances — even with CTRL C — will cause your mounted tapes to be dismounted automatically.

3.9.3 Software mounting REMOTE tapes

On all AIPS systems, the last two tape drives are indicated as REMOTE. This means you can use two additional adverbs in AIPS to access tape drives on other computers. It doesn't matter where the computer is, as long as it's connected via Internet and has AIPS installed on it in the conventional way. For example, if you wanted to use AIPS tape drive 2 on remote host rhesus, you would type:

> REMHOST 'RHESUS' ; REMTAPE 2 ↵

> DENSITY *dddd* ↵ to set the density to *dddd* bpi if needed.

> INTAPE *n* ; MOUNT ↵ set local "tape" number and software mount

where *n* is the number of one of the REMOTE tape assignments in the list of tape drives you see on AIPS startup. If you know which computers are to provide remote tape services for you, it is a good idea to specify them when you start AIPS using the tp=*hostname* option (see § 2.2.3). In this way, you make certain that the AIPS dæmon tasks TPMON*n* which provide the remote service are running where they are needed.

3.9.4 Using tapes in AIPS

AIPS provides a number of basic tools for managing magnetic tapes. It is very helpful to have a list of the contents of magnetic tapes you intend to read. To list the contents of a tape on the line printer:

> TASK 'PRTTP' ; INP ↵ to review the inputs.

> NFILES 0 ↵ to list all files on the tape.

> PRTLEV 0 ↵ to list the image headers but not the details — both more and less detailed listings are available.

> DOCRT FALSE ↵ to print on the line printer.

> GO ↵ to run the task.

It is also a good idea to run PRTTP on your data tapes after you have written them, but before you have deleted the data from disk. PRTTP reads the the tape record by record to test for tape errors as well as to check the data format.

The AIPS program has a number of verbs to position and check magnetic tapes. These include

> REWIND ↵ to rewind the tape, *e.g.*, after running PRTTP.

> NFILES *n* ; AVFILE ↵ to advance the tape $n > 0$ file marks.

> NFILES -*n* ; AVFILE ↵ to move the tape backwards to the n^{th} previous file.

> NFILES 0 ; AVFILE ↵ to position the tape at the start of the current file.

> AVEOT ↵ to advance the tape to the end of information, usually for the purpose of adding more data at the end.

> TPHEAD ↵ to display the contents of the data file at the current tape position.

Users are encouraged to treat magnetic tapes with some caution. The tapes themselves can have — or develop — errors which render the data in the file unavailable. Furthermore, there are no generally accepted standards governing magnetic tape software in the industry. As a consequence, each Unix operating system handles them differently and each can change over time. This creates great difficulties in AIPS and may cause your version not to handle all tape devices in a fully compatible manner.

3.10 AIPS external disk files

AIPS maintains a wide range of disk files for its own use internally. Unless you intend to write programs for AIPS you need not be concerned about their formats or, in many cases, even their existence. However, recent versions of AIPS also support "external" disk files to be read from and written to disk directories controlled by you. You may read and write from/to binary "FITS-disk" files with `TPHEAD`, `UVLOD`, `IMLOD`, `FITLD`, `FITTP`, and `FITAB`.. `AIPS` and some tasks also allow text files to be read or written from/to disk. For example, all print tasks can be instructed to append their output to user-specified text files. These can be examined later with an editor or written to tape with standard tape utilities. The two PostScript tasks, `LWPLA` and `TVCPS`, can be instructed to write their output plots in user-specified text files for later processing and, for example, inclusion in manuscripts. And `AIPS` itself can be instructed to take its input commands from user-created text files.

3.10.1 Disk text files

The most significant user control over external files is the specification of the file's full name, *i.e.*, its directory path and its name in that path. You specify the directory path by creating an environment variable ("logical name" in AIPSpeak) *before* starting `AIPS`. The simplest way is to change directory (`cd` Unix utility) to the area you wish to use and enter

```
% setenv MYAREA `pwd` ↵
```

where *MYAREA* is a logical name of your choosing (but all in *upper case*). Note that the `pwd` is surrounded by backward single quote marks. The grammar above is for users of c-shell and tc-shell. Users of korn, bourne, and bash shells would type:

```
$ MYAREA=`pwd`; export MYAREA ↵
```

also with backward single quote marks. If you are going to read a text file into AIPS, its name must also be in upper-case letters. Finally, inside `AIPS`, you specify the file with, *e.g.*,

```
> OUTPRINT = 'MYAREA:3C123.PRT' ↵
```

where `3C123.PRT` is any all upper-case file name of your choosing. Note the surrounding quote marks and the colon that separates the logical name and the file name portions. You may put the file anywhere under any name you choose, but we request that you put it an area owned by you, if you have one, or that you use an identifying name and a standard AIPS area set aside for the purpose. Files left around in the AIPS directories are subject to summary deletion. Be sure that AIPS has the privilege to write into your directory; use `chmod` to allow appropriate write privilege on the directory file (try to avoid world write!). On Unix systems, duplicate file names are not allowed and AIPS tasks will usually die when trying to write a file name that already exists. Print tasks will append to pre-existing files, however.

In `31DEC02`, file names may also be entered as complete path names, so long as they do not require more than 48 characters, the length of the adverb data values. Thus

```
INFILE = '/home/primate2/egreisen/AIPS/Text.prt
```

Note that the trailing quote mark is left off and this is the last command on the input line so that the case is preserved.

Ordinary text files are used in AIPS for a variety of purposes. Every print task offers the option of saving the output in a file specified by OUTPRINT rather than immediately printing and discarding it. Similarly, output PostScript files from LWPLA and TVCPS may be saved in files specified by OUTFILE rather than immediately printing and discarding them. They may be used later in larger displays, or even enclosed as figures in a TEX document such as this *CookBook*. OUTFILE is used by numerous other tasks, such as SLICE and IMEAN, to write output specific to the tasks which may be of use to other programs. AIPS tables may even be written as text files by task TBOUT, edited by the user, and then read back in by task TBIN. History files may be revised in a similar manner. The adverb INFILE may be used by a number of tasks to specify source models, lists of "star" positions, holography data, and the like. Television color tables are read from and written to disk text files specified with the OFMFILE adverb.

3.10.2 RUN files

RUN files are ordinary text files containing AIPS commands to be executed in sequence in a batch-like manner. They are often used to define procedures which you save in your own area or in an AIPS-provided public area with the logical name $RUNFIL. The name of the file must be all upper case letters, followed by a period, followed by your user number as a three-digit "extended-hexadecimal" number with leading zeros. (To translate between decimal and extended hexadecimal, use the AIPS procedures or the AIPS verbs called EHEX and REHEX.) The files are edited from Unix level using emacs, vi, textedit or your other preferred text editor. For example, log in to the aips (or your own) account. From Unix level, type:

% cd $RUNFIL — to change to RUN area.

% emacs MAPIT.03D ↵

to edit with emacs a file called MAPIT for user 121. You may now also use any area of your choosing instead of the public $RUNFIL area. For instructions on the individual editors, consult the appropriate Unix Manuals. Instruction manuals for the GNU emacs editor are available from local computer staff. In 31DEC02, a SAVE area (§ 3.5) may be written as a RUN file if you first GET the area and then use SG2RUN.

To use the RUN file, define a logical name as in the previous Section. Then start up AIPS under your user number and enter

> VERSION = 'MYAREA' ↵ — where MYAREA is your disk area, or

> VERSION = ' ' ↵ — if $RUNFIL is to be used

> RUN *FILE* ↵ — to execute the file named *FILE.uuu*

where *uuu* is your user number if extended hexadecimal with leading zeros to make three digits.

3.10.3 FITS-disk files

FITS is an IAU-endorsed binary format standard for astronomical data heavily used by AIPS for almost all of its data on magnetic tape. In fact, it is the only format written by AIPS except for simple tape copying. The basic FITS paper (by Wells, Greisen, and Harten) appeared in *Astronomy & Astrophysics Supplement Series*, Volume 44, pages 363–374, 1981. The newsgroup sci.astro.fits is devoted to discussion of FITS. World-wide web users can access the FITS home page at

http://www.cv.nrao.edu/fits/

AIPS also supports the FITS format written to disk in exactly the same form as it is written to magnetic tape. The tasks FITTP and FITAB may be instructed to write their output files on disk rather than on tape. Likewise, TPHEAD, FITLD, UVLOD, IMLOD, and PRTTP can read from disk. To write to a FITS-disk file, specify:

> OUTFILE *'filename'* ↵ where *filename* is the name of the desired output file.

and to read from a FITS-disk file, you specify:

> INFILE *'filename'* ↵

where you must specify *filename* with environment variables ("logical names" in *AIPS*peak), *e.g.*,

> OUTFILE = 'MYDATA:3C123.FIT' ↵

in exactly the same way as described for text files in § 3.10.1. There is a standard public area, called logically FITS, which you may use for reading and writing FITS-disk files. FITTP will use this area if you do not specify a logical name. Be aware that older files will be purged from this public area when space is needed. Note too that FITTP will write only one disk file per execution; the DOALL option is disabled when writing to disk.

In the 31DEC02 release of *AIPS*, there is a package of procedures to assist in writing and reading more than one FITS-disk file at a time. Enter RUN WRTPROCS to define the procedures. The procedure FITDISK will write a single disk catalog file to a disk file using a name based on the *AIPS* file name parameters. You may then construct loops invoking FITDISK to write multiple files. For example:

> FOR I=1:10; GETN(I); FITDISK; END ↵

Such file names are useful for their mnemonic content, but must be read back one at a time. The procedure WRTDISK will dump a range of catalog numbers to disk unders names that allow the procedure READISK to read them back as a group. These two procedures are particularly useful when moving your data between computer architectures (*e.g.*, from a Solaris to a Linux computer).

Beginning with the 31DEC03 release, FITLD can read multiple disk files in either the normal FITS format (as written by FITTP) or the special FITS format written by the VLBA correlator. The only requirement for this operation is that file names end in sequential numbers beginning with 1. FITAB has the ability to write special FITS files with visibility data in tables. These files may be broken up into multiple files, called "pieces," for size and reliability considerations. These pieces, when written to disk, have names ending in sequential numbers. Special code in FITLD and UVLOD recognize these pieces and read the requested number of them as if they were in one file.

Remote FITS-disk files may be read in much the same manner as remote magnetic tapes. Type HELP INFILE ↵ or HELP OUTFILE ↵ for details.

FITS-disk files are written as Fortran files and hence are available also to user-coded programs. The Fortran specifications for the file are ACCESS='DIRECT', RECL=2880, FORM='UNFORMATTED' in the OPEN statement for Unix systems. Most Fortrans cannot read or write files larger than 2 Gigabytes, so *AIPS* now reads and writes these files with C subroutines. Users may also, of course, code programs to create such files to be read by FITLD, IMLOD or UVLOD. Consult *GOING AIPS*, Volume 2, Chapter 13 for details on how to do this.

One of the main uses for FITS-disk files is to transfer data over the Internet between computers. For example, to transfer a file from **rhesus** (in Charlottesville) to **kiowa** (at the AOC), log in to **rhesus**, change to the directory in which you wish to store the file (for example, cd $FITS ↵), and enter:

% ftp kiowa ↵	to start ftp to the remote system.
Name (kiowa:...): *loginame* ↵	to log in to account *loginame*.
Password: *password* ↵	to give the account's password.
ftp> cd *directory* ↵	to change to the *directory* name containing the file.
ftp> binary ↵	to allow reading of a binary file.
ftp> hash ↵	to get progress symbols as the copy proceeds.
ftp> put *filename* ↵	to send the file
ftp> quit ↵	to exit from ftp.

The file should then be in the desired directory. You may have to rename it, however, to a name in all upper-case letters since that may be required by *AIPS*. (See § 3.10.1 for a trick that allows you to use

lower-case letters in file names.) The file format will be correct. In general it is better to use the ftp program to "get" files instead of "put"ting them; things tend to go faster that way.

An alternative to using `ftp` is to use the `rcp` (remote copy) Unix utility or to write the output file directory in the appropriate area on the other computer. In order to do this, you have to have accounts on both machines, and you should have set up a `.rhosts` file (see the Unix manual page on `rhosts` for instructions). Once you know this works (test it via, *e.g.*, `rsh rhesus whoami`), the syntax for the remote copy is:

```
% rcp $FITS/MYFILE.FITS kiowa:/AIPS/FITS/MYFILE  CR
```

(this shows how you would copy it from `rhesus` to `kiowa`). A secure copy (`scp`) would be better if you have set up the secure connection capability.

If you wish to copy a FITS-disk file from one machine to another within a site, check if you can just use the unix `cp` command; this is often possible if the remote disk is mounted (or can be automounted) via NFS (the Network File System).

FITS files may be compressed with standard utility programs such as `gzip`. This does not produce much compression for files written with full dynamic range and floating-point format. However, `FITAB` offers the option of writing images (not *uv* data) which are quantized at some suitable level. These are capable of significant compression even if they are in floating-point format.

3.10.4 Other binary data disk files

Data written by the on-line system of the VLA are now often found in disk files rather than on tape. These data are available from an archive of all VLA data. See

`http://e2e.aoc.nrao.edu/archive/e2earchive.html`

for information on how to access your current data and all data for which the proprietary period has expired. `FILLM` and `PRTTP` can read the disk files produced from the archive, including reading more than one such file in a single execution. In this case, the file names must end in consecutive numbers beginning with `NFILES`+1.

3.11 Additional recipes

3.11.1 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.

3.11.2 Cream of banana soup

1. Cook 1 quart green **banana** pulp, 1 1/2 quarts **chicken stock**, 1 small **celery stalk**, 1/2 **onion**, 1 **carrot**, 1 small **bay leaf**, 5 **peppercorns**, and **salt** to taste together for about 30 minutes until the mixture thickens.
2. Strain over 1/4 cup **flour** and 1/4 cup **butter** which have been combined as for a white sauce. Cook until thickened.
3. Just before serving, add 2 cups **cream** or **milk** and heat.
4. Serve with a slice of lemon on each plate as a garnish.

3.11.3 Banana curried chicken

1. Fry 2 chopped **onions** in 50 ml **cooking oil** until light brown.
2. Add 1/4 cup **cake flour** and mix well. Add 1 (cup?) **chicken stock** gradually while stirring.
3. Add 1 cup **raisins**, 1 teaspoon **salt**, 2 pounds cooked, boned **chicken**, 5 sliced **bananas**, 2 grated **apples**, 2 tablespoons grated **lemon rind**, 1 tablespoon **sugar**, 1 1/2 tablespoons **curry powder**, 1 **bay leaf**, 4 **peppercorns**.
4. Cover saucepan and simmer for 20 minutes.
5. Remove bay leaf. Add 1 cup **cream** and heat just before serving.
6. Serve on a bed of rice. Decorate with pineapples if preferred.

Thanks to Turbana Corporation (`www.turbana.com`).

3.11.4 Banana July cocktail

1. Sprinkle 3 sliced **bananas** with 1 tablespoon **lemon juice**.
2. Mix with 1 1/4 cans drained and flaked **tuna**, 1/2 **onion** chopped, and 2 tablespoons chopped **gherkins** or **olives**.
3. Spoon into 7 cocktail shells.
4. Melt 2 tablespoons **butter** in a saucepan. Add 2 tablespoon **cake flour** and salt and pepper to taste.
5. Add 1/4 cup **chicken stock** and 1/4 cup dry **white wine**. Simmer for one minute stirring constantly.
6. Add 1/3 cup grated **cheddar cheese** and allow to cool.
7. Add 1/4 cup fresh cream to sauce and pour over banana-tuna mixture.
8. Sprinkle with 1 tablespoon grated **cheese** and **paprika**. Decorate with a slice of **gherkin** pr **olive**.
9. Bake 15–20 minutes at 350° F; serve warm.

Thanks to Turbana Corporation (`www.turbana.com`).

4 CALIBRATING INTERFEROMETER DATA

This chapter focuses on ways to do the initial calibration of interferometric fringe-visibility data in *AIPS*. The sections which follow concentrate primarily on continuum calibration for connected-element interferometers, especially the VLA. However, the information in these sections is useful to spectral-line, solar, and VLBI observers as well. For additional advice on spectral-line calibration, see § 4.7; for advice on calibrating observations of the Sun, see § 4.8; and for the gory details of VLBI, read Chapter 9. After the initial calibration has been completed, data for sources with good signal-to-noise are often taken through a number of cycles of imaging with self-calibration. See § 5.4 for information on these later stages of the reduction process. For accurate calibration, you must have accurate *a priori* positions and structural information for all your calibration sources and accurate flux densities for at least one of them. It is best if the calibration sources are unresolved "point" sources, but it is not required.

For the basic calibrations, visibility ("*uv*") data are kept in "multi-source data sets," each of which contains, in time order, visibility data for one or more "unknown" sources and one or more calibration sources. Associated with these data are "extension" files containing tables describing these data. When VLA archive data are first read into *AIPS* a number of basic tables are created and filled with information describing the data set. These are

1. `AN` (antennas) for sub-array geometric data, date, frequency, polarization information, *etc.*,
2. `FQ` (frequency) for frequency offsets of the different IFs (IF pairs in VLA nomenclature),
3. `NX` (index) to assist rapid access to the data,
4. `SU` (source) for source specific information such as name, position, velocity, and
5. `TY` (temperature) for measured system temperatures.

A null `CL` table is also created at this time. VLBI, and especially VLBA, data sets will end up with even more table files. Calibration and editing tasks then create, as needed, other tables including

6. `BL` (baseline) for baseline-, or correlator-, dependent corrections,
7. `BP` (bandpass) for bandpass calibration,
8. `CL` (calibration) for calibration and model information,
9. `FG` (flag) for flagging (editing) information, and
10. `SN` (solution) for gain solutions from the calibration routines.

All of these tables can be written to, and read back from, FITS files along with the visibility data. These, and any other, *AIPS* tables can be manipulated and examined using the general tasks `PRTAB`, `TACOP`, `TABED`, `TAMRG`, `TASRT` and `TAFLG`.

The visibility data within the multi-source data set are not normally altered by the calibration tasks. Instead, these tasks manipulate the tabular information to describe the calibration corrections to be applied to the data and any flagging (deletion) of the data.

The *AIPS* programs discussed in this chapter are part of a package that has been developed to calibrate interferometer data from a wide range of connected-element and VLB arrays, especially the VLA and VLBA. These programs therefore support many functions (and inputs) that are not required when calibrating normal VLA data. The examples given below show only the essential parameters for the operation being described, but, to get the results described, it is essential that you check *all* the input parameters before running any task. Remember that *AIPS* adverbs are global and will be "remembered" as you proceed. A list of calibration-related symbols is given in § 13.6, but a possibly more up-to-date list can be obtained by typing ABOUT CALIBRAT in your *AIPS* session. More general information on calibration can be routed to your printer by typing DOCRT FALSE ; EXPLAIN CALIBRAT ↵, while deeper information on a specific task is obtained with EXPLAIN *taskname* ↵.

When you are satisfied with the calibration and editing (or are simply exhausted), the task SPLIT is used to apply the calibration and editing tables and to write *uv* files, each containing the data for only one source. These "single-source" *uv* files are used by imaging and deconvolution tasks that work with only one source at a time. Many of the tasks described in this chapter will also work on single-source files. For VLA calibration, there are several useful procedures described in this chapter and contained in the RUN file called VLAPROCS. Each of these procedures has an associated HELP file and inputs. Before any of these procedures can be used, this RUN file must be invoked with:

> RUN VLAPROCS ↵ to compile the procedures.

Beginning with the 31DEC03 version, there is a "pipeline" procedure designed to do a preliminary calibration and imaging of ordinary VLA data sets. This provides a good first look at the data but should never be used for published results. To run the pipeline, enter

> RUN VLARUN ↵ to compile the procedures.
> INP VLARUN ↵ to review the input adverbs and, when ready,
> VLARUN ↵ to execute the pipeline.

4.1 Copying data into *AIPS* multi-source disk files

There are several ways to write VLA data to *AIPS* multi-source *uv* data sets on disk. They include:

1. For VLA observations on or after January 1, 1988, use FILLM to read the VLA archive tape (or a copy thereof) directly.

2. For VLA observations before January 1, 1988, use FILLM on a translation of the original archive tape. All VLA archive data have been copied to Exabyte tapes, while being translated to the modern format. Contact the VLA data analysts (phone 505–835–7359, e-mail analysts@nrao.edu) to obtain a translated copy of any old observing files.

3. For an *AIPS* multi-source data set written to a FITS tape or FITS disk during an earlier *AIPS* session, use UVLOD or FITLD to read the tape.

4. For VLA data from the archive, use FILLM to read one or more disk files; see § 3.10.4.

5. For single-source data sets that are already on disk and are very similar in structure, use UV2MS on one of them to create a multi-source data set, and then on each of the others to append them to that multi-source data set. Each of the input data sets should have the same number of polarizations, IFs, spectral channels, and "random parameters." UV2MS also makes no corrections for differences in observed source positions or frequencies. After all are appended, use UVSRT to put the data in time-baseline order and INDXR to make an index and initial (null) calibration file.

6. For single-source data sets that are already on disk and are not sufficiently similar in structure for the method above, use MULTI on each single-source file to convert to multi-source format. Then

use DBCON to concatenate the individual multi-source files into one big multi-source file. Finally use UVSRT, if needed, to put the data in time-baseline order and INDXR to make an index and initial (null) calibration file.

Data from other telescopes can be read into *AIPS* only if they are written in *AIPS*-like FITS files already or if you have a special format-translation program for that telescope. The VLBA correlator produces a format which is translated by the standard *AIPS* task FITLD; see § 4.1.2. Translation tasks for the Westerbork Synthesis Telescope (WSLOD) and the Australia Telescope (ATLOD) are available from the Dutch and Australians, respectively, but are not distributed by the NRAO with the normal *AIPS* system.

4.1.1 Reading from a VLA archive tape using FILLM

To load a *uv* data file to disk from a VLA archive tape, you must (hardware) mount the tape on a tape drive and then (software) mount the tape inside the AIPS program. See § 3.9 for a discussion of this process. It is strongly recommended that you begin by obtaining an index of the contents of your data tape. Reference dates, time ranges, file numbers, frequencies observed, and the like are reported in the index and are needed to guide the actual loading of the data. To print an index of the archive tape, use task PRTTP:

> TASK 'PRTTP' ; INP CR	to review the inputs needed.
> NFILES 0 CR	to start at the beginning of tape.
> PRTLEV 0 CR	to give complete summaries; only PRTLEV = -3 actually affects the output (adversely).
> DOCRT FALSE CR	to send output to the line printer.
> INFILE ' ' CR	to read from tape not disk. Multiple VLA archive files may be read from disk beginning with 31DEC02.
> GO CR	to index the tape.

Typical inputs to FILLM might be:

> TASK 'FILLM' ; INP CR	to review the inputs needed.
> INFILE ' ' CR	to read from tape not disk. Multiple VLA archive files may be read from disk beginning with 31DEC02.
> OUTNA ' ' CR	to take the default output file name.
> OUTDI 3 CR	to write the data to disk 3 (one with enough space).
> DOUVCOMP TRUE CR	to write visibilities in compressed format to save disk space.
> DOCONCAT TRUE CR	to concatenate files if this is second tape.
> DOALL TRUE CR	to include data from all frequency bands, source qualifiers, and numbers of spectral channels, writing as many output data sets as needed.
> VLAOBS 'AC238' CR	to select only data from observing program AC238. The default is to load data from *all* programs.
> NFILES 4 CR	to skip the first 4 files on the archive tape.
> DOWEIGHT 1 CR	Data weights will depend on the "nominal sensitivity" and should be calibrated along with the visibility amplitudes (DOCALIB = 2).
> CPARM 30, 0 CR	to average the data for 30 seconds; default is no averaging.
> CPARM(6) 1 CR	to select VLA sub-array 1.
> CPARM(7) 2000 CR	to have observations within 2 MHz be regarded as being at the same frequency.
> CPARM(8) 2 CR	to use a 2-minute interval for the CL table; default is 5 min.

> CPARM(9) 0.5 ↵	to use a 30-second interval for the TY table; default is the input data interval.
> DPARM 0 ↵	to have no selection by specific frequency.
> REFDATE '*yyyymmdd*' ↵	to specify the year, month, and day of the reference date. This should be the first date in the data set (or earlier). All times in *AIPS* will be measured with respect to that date and must be positive. The default is the first date included by the data selection adverbs, which may not be the desired one. Note that REFDATE is only a reference point; it does not affect which data are loaded from the tape.
> TIMERANG *db* , *hb* , *mb* , *sb* , *de*, *he* , *me* , *se* ↵	to specify the beginning day, hour, minute, and second and ending day, hour, minute, and second (wrt REFDATE) of the data to be included. The default is to include all times.
> INP ↵	to review the inputs.
> GO ↵	to run the program when you're satisfied with inputs.

Be careful when choosing the averaging time with CPARM(1). If you have a large data set, setting this time too *low* will make an unnecessarily large output file; this may waste disk space and slow the execution of subsequent programs. Setting it too *high* can, however, (1) smear bad data into good, limiting the ability to recognize and precisely remove bad data, (2) smear features of the image that are far from the phase center, and (3) limit the dynamic range that can be obtained using self-calibration. If you need a different (usually shorter) averaging time for the calibrator sources than for your program sources, use CPARM(10) to specify the averaging time for calibrators. See Lectures 12 and 13 in *Synthesis Imaging in Radio Astronomy*[1] for general guidance about the choice of averaging time given the size of the required field of view and the observing bandwidth.

CPARM(2) controls a number of mostly esoteric options. If your data include the Sun or planets, you must set CPARM(2) = 16 to avoid having each scan on the moving source assigned a different name. The adverb DOWEIGHT = 1 has the same affect as CPARM(2) = 8 and both select the use of the nominal sensitivity to scale the data weights. When this is done, the weights will be $1/\sigma^2$ as they should for imaging, with σ in "Jy" in the same uncalibrated scale as the fringe visibilities. Having selected this option, you should apply any amplitude calibration to the weights as well as the visibilities; use DOCAL=2 rather than DOCAL=1. If you store the data in compressed form, only one weight may be retained with each sample. Any differences between polarizations and/or IFs in that sample will be lost. Uncompressed data also require less computer time to read but 2 to 3 times as much disk space to store.

FILLM was changed September 21, 2001 in the 31DEC01 release to write a weather (WX) table to the output file. At the same time, it was changed to use "canned" VLA antenna gain curves and a balance of the current with a seasonal model weather data to estimate opacity and gain corrections to be written into the first calibration (CL) table. These functions are controlled by adverbs IN2FILE and BPARM and may be turned off, although the default is to make the corrections. In subsequent tasks, set DOCALIB = 2 to use these initial calibration data.

Some words of warning about the use of NFILES are appropriate here. VLA archive tapes used to contain 3 tapes files for every actual data file. Most archive tapes today do not have these ANSI standard-label files, but still tend to begin with a header (non-data) tape file. If you set the NFILES adverb carefully based on the index printed by PRTTP, you should have no problem with these "excess" tape files.

FILLM is designed to read all your data from tape in one pass. All data meeting the selection criteria will be read from the input tape and filled into a *uv* multi-source file. Three selection criteria are always active:

[1] *Synthesis Imaging in Radio Astronomy*, Astronomical Society of the Pacific Conference Series, Volume 6 "A Collection of Lectures from the Third NRAO Synthesis Imaging Summer School" eds. R. A. Perley, F. R. Schwab and A. H. Bridle (1989)

(a) TIMERANG, if non-zero, will restrict processing of data to the specified range of times (with respect to REFDATE); (b) VLAOBS, will restrict processing to the specified VLA observing code (which will be set to the code in the first valid data record if you do not specify it); and (c) CPARM(6) will restrict processing to the specified VLA sub-array (which, if you do not specify it, will be set to 1 if VLAOBS is not specified or to the first sub-array belonging to VLAOBS). All other selection criteria may be overridden by setting DOALL to TRUE.

Where possible, FILLM will try to place all data in one file. However, in many cases this is not possible. For instance so-called "channel 0" data from a spectral-line observation will be placed in a separate file from its associated line data. Similarly, scans which have differing numbers of frequency channels will also be placed into separate files. Another case is observations made in mode LP, *i.e.*, one IF-pair is set to L band, the other to P band. In this case the two bands will be split into separate files. Yet another case arises when there are observations of different bandwidths. All of this should be relatively transparent to the user.

If your data are on multiple tapes, you can write them all into the same data file by specifying DOCONCAT on the second (and subsequent) runs of FILLM.

A significant reduction in the disk space used may be achieved using the compressed format invoked by adverb DOUVCOMP; this factor is 1.89 for 2-IF continuum data and approaches 3.0 for line data. Almost all tasks can process compressed *uv* data. The task UVCMP allows you to change the formats of *uv* data sets between *compressed* and *uncompressed*, if required to use one of the few aberrant tasks.

FILLM and many *AIPS* tasks are able to handle multiple, logically different, frequencies within a multi-source data set. FILLM does this by assigning an FQ number to each observation and associating a line of information about that frequency in the FQ file associated with the data set. Users should note that this concept can become quite complicated and that not all tasks can handle it in full generality. In fact, most tasks can only process one FQ number at a time. Polarization calibration works only on one FQ at a time since the antenna file format allows for only one set of instrumental polarization parameters. Therefore, it is *strongly* advised that you fill continuum experiments which involve multiple frequencies into separate data sets. FILLM will separate bands automatically, but you will have to force any remaining separation. To do this, (a) use the QUAL adverb in FILLM, assuming that you have used separate qualifiers in OBSERVE for each frequency pair; (b) use the DPARM adverb array in FILLM to specify the desired frequencies precisely; or (c) use the UVCOP task to separate a multiple FQ data set into its constituent parts. Note that the first two options require multiple executions of FILLM, while the third option requires more disk space.

Spectral-line users and continuum observers using different frequencies in the same band should be aware of the FQ entry tolerance. Each frequency in a *uv* file will be assigned an FQ number as it is read from tape by FILLM. For spectral-line users, the observing frequency will normally change as a function of time due to Doppler tracking of the Earth's rotation, or switching between sources or between spectral lines; in general, this will cause different scans to have different FQ numbers. FILLM assigns an FQ number to a scan based on the FQ tolerance adverb CPARM(7) which defines the maximum change of frequency allowed before a new FQ number is allocated. If CPARM(7) < 0, the the same FQ number is assigned to all data in spectral-line data sets. If CPARM(7) is positive, a scan will be assigned to an existing FQ number if

$$||\nu_{current} - \nu_{firstFQ}|| < \mathrm{CPARM}(7)$$

where $\nu_{firstFQ}$ is the frequency of the first sample to which the particular FQ number was assigned. If no match is found, then a new FQ number is created and assigned and another line added to the FQ table file. Alternatively, if CPARM(7) is zero, then the FQ tolerance is assumed to be half of the maximum frequency difference caused by observing in directions 180 degrees apart (*i.e.*, $\Delta\nu = 10^{-4} \times \nu$).

An example: if an observer observes the 1612, 1665 and 1667 MHz OH masers in VY CMa and NML Cygnus, then presumably he would like his data to have 3 FQ numbers, one associated with each OH transition. However, running FILLM with CPARM(7) set to 0 would produce 6 FQ numbers because the frequency difference between the masers in VY CMa and NML Cygnus is greater than the calculated tolerance of 160 kHz. Therefore, in order to ensure that only 3 FQ numbers are assigned, he should set

CPARM(7) to 1000 kHz. Setting CPARM(7) < 0 would result in all data having the same FQ number, which is clearly undesirable.

For most continuum experiments the FQ number will be constant throughout the database. Normally any change in frequency should be given a new FQ number. To achieve this, FILLM treats CPARM(7) differently for continuum. If CPARM(7) $\leq$ 0.0, then FILLM assumes a value of 100 kHz. A positive value of CPARM(7) is treated as a tolerance in kHz as in the spectral line case.

Note: *If your uv database contains several frequency identifiers, you should go through the calibration steps for each* FQ *code separately.*

FILLM is prepared to try to read past up to 50 parity or other tape errors. Do not be alarmed by a few warning messages, especially at the end of tape on old 9-track, half-inch tapes. These are relatively normal and will cause no harm. If FILLM is executing correctly, your message terminal will report the number of your observing program, the VLA archive tape format revision number, and then the names of the sources as they are found on the tape. Once FILLM has completed, you can find the database on disk using:

> INDI 0 ; UCAT ↵

This should produce a listing such as:

```
Catalog on disk  3
Cat Usid Mapname      Class  Seq  Pt     Last access      Stat
  1  103 25/11/88    .X BAND.   1 UV 05-FEB-1994 12:34:16
```

You might then examine the header information for the disk data set by:

> INDI 3 ; GETN 1 ; IMH ↵

This should produce a listing like:

```
 Image=MULTI     (UV)         Filename=25/11/88    .X BAND.   1
 Telescope=VLA                Receiver=VLA
 Observer=AC238               User #=  103
 Observ. date=25-NOV-1988     Map date=05-FEB-1994
 # visibilities    191317     Sort order  TB
 Rand axes: UU-L-SIN  VV-L-SIN  WW-L-SIN  BASELINE  TIME1
            SOURCE  FREQSEL WEIGHT  SCALE
 -----------------------------------------------------------------
 Type    Pixels   Coord value  at Pixel    Coord incr   Rotat
 COMPLEX      1   1.0000000E+00    1.00 1.0000000E+00    0.00
 STOKES       4  -1.0000000E+00    1.00-1.0000000E+00    0.00
 IF           2   1.0000000E+00    1.00 1.0000000E+00    0.00
 FREQ         1   8.4110000E+09    1.00 1.2500000E+07    0.00
 RA           1    00 00 00.000    1.00      3600.000    0.00
 DEC          1    00 00 00.000    1.00      3600.000    0.00
 -----------------------------------------------------------------
 Maximum version number of extension files of type HI is   1
 Maximum version number of extension files of type AN is   1
 Maximum version number of extension files of type NX is   1
 Maximum version number of extension files of type SU is   1
 Maximum version number of extension files of type FQ is   1
 Maximum version number of extension files of type WX is   1
 Maximum version number of extension files of type CL is   1
 Keyword = 'CORRMODE'  value = '        '
 Keyword = 'VLAIFS  '  value = 'ABCD    '
```

This header identifies the file as a multi-source file (Image=MULTI) with 191317 floating-point visibilities in time-baseline (TB) order. There are two entries on the IF axis. These correspond to the VLA's "AC" and

"BD" IF-pairs respectively. The description of the frequency (FREQ) axis shows that the first IF ("AC") is at 8411 MHz and has 12.5 MHz bandwidth. The parameters of the second IF-pair ("BD") are determined from the data in the FQ table file and cannot be read directly from this header; these values are shown in the 'SCAN' listing from LISTR. The header shown above indicates that the data are in compressed format since the number of pixels on the COMPLEX axis is 1 and the WEIGHT and SCALE random parameters are present. Uncompressed data does not use these random parameters and has 3 pixels on the COMPLEX axis.

The term "IF" can be confusing. At the VLA, IFs "A" and "C" correspond to right-hand and left-hand circularly polarized (RHC and LHC) signals, respectively, and are normally for the same frequency in an observing band. Such pairs, if at the same frequency, are considered to be one "IF" in *AIPS*. An observation which was made in spectral line mode "2AC" is considered at the VLA to have two "IFs" whereas within *AIPS* this would be filled as one "IF" with two polarizations if they were both observed with the same frequency, the same number of channels, and the same channel separation. If these conditions do not hold, then they are filled into separate *uv* files, each with a single IF and a single polarization. The term "sub-array" is also confusing. At the VLA — and in task FILLM — sub-array means the subset of the 27 antennas actually used to observe your sources. (The VLA allows up to 5 simultaneous sub-arrays in this sense.) In the rest of *AIPS*, sub-array refers to sets of antennas used together at the same time. If observations from separate times (*e.g.*, separate array configurations) are concatenated into the same file, then *AIPS* will regard the separate sets of antennas as different "sub-arrays" whether or not the same physical antennas occur within more than one of these sub-arrays.

If your experiment contains data from several bands FILLM will place the data from each band in separate data sets. Also, if you observed with several sets of frequencies or bandwidths in a given observing run these will be assigned different FQ numbers by FILLM. You can determine which frequencies correspond to which FQ numbers from the 'SCAN' listing provided by LISTR. Line data are divided into the "channel 0" (central 3/4 of the of the observing band averaged) and the spectra. Data observed in the "LP" mode (or any other two-band mode) will be broken into separate data sets, one for each band.

As a practical note, setting the start and stop times of the data for your experiment with TIMERANG will cause FILLM to read all valid data up to the stop time you have specified and then exit normally. This way, it will not read to the end of the VLA archive file. The default action of FILLM (to read to the end of the tape if TIMERANG = 0) can be particularly annoying if your data are at the very start of a large archive data file.

When the data are successfully loaded to disk, type DISMOUNT ↵ to dismount your input tape.

4.1.2 Reading from a FITS tape with FITLD

FITLD is used to read FITS-format tapes into *AIPS*. It recognizes images, single- and multi-source *uv* data sets, and the special FITS *uv*-data tables produced by the VLBA correlator. In particular, VLA data sets that have been read into *AIPS* previously with FILLM and then saved to tape (or pseudo-tape disk) files with FITTP can be recovered for further processing with task FITLD. (The older task UVLOD will also work with *uv* data sets in FITS format, but it cannot handle image or VLBA-format files.)

A multi-source data file with all of its tables can be read from a FITS tape by:

> TASK 'FITLD' ; INP ↵	to review the inputs needed.
> INTAPE *n* ↵	to specify the tape drive for input from tape.
> INFILE '*filename*' ↵	if the input is from a FITS disk file (see § 3.10.3).
> DOUVCOMP TRUE ↵	to write visibilities in compressed format.
> OUTNA '' ↵	take default (previous *AIPS*) name.
> OUTCL '' ↵	take default (previous *AIPS*) class.

> OUTSEQ 0 ↵	take default (previous *AIPS*) sequence #.
> OUTDI 3 ↵	to write the data to disk 3 (one with enough space).
> INP ↵	to review the inputs (several apply only to VLBA format files).
> GO ↵	to run the program when you're satisfied with inputs.

FITLD is the equivalent of FILLM, but for output from the VLBA, rather than the VLA, correlator. The data-selection adverbs SOURCES, QUAL, CALCODE, and TIMERANG and the table-control adverbs CLINT and FQTOL are used, for VLBA-format data only, in FITLD in ways similar to the data-selection and control adverbs of FILLM. See Chapter 9 for more specific information.

4.2 Record keeping and data management

4.2.1 Calibrating data with multiple FQ entries

In general an observing run with the VLA, especially a spectral-line run, will result in a *uv* data file containing multiple FQ entries. In early versions of the *AIPS* software, the different FQ entries would automatically have been placed in different physical files. Now, FILLM allows you to place all of them in the same file. This may be convenient, but it has a number of costs. If a file contains multiple, independent frequencies, then it occupies more disk space and costs time in every program to skip the currently unwanted data (either a small cost when the index file is used or a rather larger cost when the file must be read sequentially). Since multiple frequencies are still not handled correctly in all programs (*i.e.*, polarization calibration) and since it is not possible to calibrate all of the different FQ data in one pass, you might consider separating the multiple frequencies into separate files (as described in § 4.1.1). In either case, you must calibrate each frequency with a separate pass of the scheme outlined below. There are three adverbs to enable you to differentiate between the different FQ entries: FREQID enables the user to specify the FQ number directly (with -1 or 0 meaning to take the first found); SELFREQ and SELBAND enable the user to specify the observing frequency and bandwidth to be calibrated (the tasks then determine to which FQ number these adverbs correspond). If SELFREQ and SELBAND are specified they override the value of FREQID.

There are certain bookkeeping tasks that must be performed between calibrating each FQ set. First, you must ensure that you have reset the fluxes of your secondary calibrators by running SETJY with OPTYPE = 'REJY' — if not, this will cause the amplitudes of your data to be incorrect. Second, it is wise to remove the SN tables associated with any previous calibration using the verb EXTDEST. Although this is not strictly necessary, it will simplify your bookkeeping.

A practical note: it is often useful to have used different qualifiers for different frequencies. This gives you another "handle" on the data. Unfortunately, not all programs use the QUAL, or even the CALCODE, adverb.

4.2.2 Recommended record keeping

It is useful to print a summary of the time stamps and source names of the scans in your data set. This reminds you of the structure of your observing program when you decide on interpolation and editing strategies, and may help to clarify relationships between later, more detailed listings of parts of the data set. It is also useful to have a printed scan summary and a map of the antenna layout if you need to return to processing the data months or years later. Finally, it is also making sure that all *AIPS* input parameters have their null (default) values before invoking the parts of the calibration package, such as CALIB, that have many inputs. The null settings of most parameters are arranged to be sensible ones so that basic VLA calibration can be done with a minimum of specific inputs; but some inputs may lose their default values if

you interleave other *AIPS* tasks with the calibration pattern recommended below. Therefore, you should *always* review the input parameters with INP *taskname* ↵ before running task *taskname*.

We suggest that you begin a calibration session with the following inputs:

> DEFAULT LISTR ↵	to set all LISTR's inputs to null (default) values.
> TASK 'LISTR' ; INP ↵	to review the inputs needed.
> INDI n; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ in FILLM example above.
> TPUT CALIB ↵	to store null values for later use with CALIB.
> OPTYP 'SCAN' ↵	to select scan summary listing.
> DOCRT -1 ↵	to send the output to the printer.
> INP ↵	to review the inputs for LISTR.
> GO ↵	to run the program when the inputs are set correctly.

Note that the DEFAULT LISTR sets the adverbs to select all sources and all times and to send printed output to the terminal rather than the printer. It is also very useful to have a printed summary of your antenna locations, especially a list of which ones you actually ended up using. To do this, enter

> NPRINT 0 ↵	to do all antennas
> INVERS n ↵	to do sub-array n
> GO PRTAN ↵	to print the list and a map of antenna locations.

In looking over the output from LISTR, you may notice that some of the sources you wish to use as calibrators have a blank "Calcode". To mark them as calibrators, use:

> TASK 'SETJY' ; INP ↵	to select the task and review its inputs.
> SOURCES '*sor1*' , '*sor2*' , '*sor3*' , ... ↵	to select the unmarked calibrator sources.
> OPTYPE 'RESE' ↵	to reset fluxes and velocities.
> CALCODE 'C' ↵	to mark the sources as "C" calibrators.
> GO ↵	to run the task.

This operation will let you select the calibrators by their Calcodes rather than having to spell out their names over and over again. You may wish to consider separate calibrator codes for primary and secondary gain calibrators to make them easier to separate. You may reset a calibrator code to blank by specifying CALCODE = '----'.

4.3 Beginning the calibration

After loading the data to disk, it has been traditional to begin with a substantial session of data checking and editing. With data from the VLA, this is always time consuming and often not necessary. Nonetheless, it is probably a good idea to check for two specific kinds of problems before beginning the actual calibration. These are corrupted data in the first record of most scans and totally dead antennas. Many other problems in the data are quickly and easily diagnosed by carefully inspecting the solution tables produced from the calibrators on un-edited data. Missing antennas and erratic amplitudes due to sampling problems and RF interference can be spotted from the SN tables and the closure-error messages produced by CALIB. If you *can't* spot errors from these, you may not need to edit the calibrator data. If the SN tables have well-behaved phases for most antennas and rapidly rotating phases for one or two, then you may need to apply baseline corrections rather than editing. See § 4.4.4 for details of how to make antenna-position corrections.

The next section tells how to detect simple problems in the data and eliminate them to reduce the warnings from the calibration tasks. The following sections tell you how to enter fluxes for the primary calibrator sources and do a preliminary calibration for all calibrators. In so doing, you should generate one or more

solution (SN) tables containing the complex gains at the times of the calibration observations. These tables may be examined for problems with the observations. If you find problems, then you need to edit the data or apply baseline corrections and should consult § 4.4. If you do not find problems, you may proceed directly to § 4.5. (Of course, you may decide to edit the data from your program sources at a later stage of the data reduction and have to return to § 4.4 then.)

4.3.1 Initial editing

The warning messages from the calibrations described in the next sections may be reduced by flagging those antennas which were not actually working, but which were not flagged by the on-line system. Another problem that has plagued the VLA (and other interferometers) persistently is that the first record in scans can be corrupted; usually its amplitudes are lower than they should be. These data can be flagged using TVFLG or UVFLG, but this can be time consuming. The task EDITA described in § 4.4.2 is now likely to be the best initial (and perhaps only) editing tool which you need. For a more traditional approach, we recommend that you do the following before beginning your regular data editing. Use the task LISTR on your terminal (to save time and paper) to see if you have the problem:

> TASK 'LISTR' ↵	to set the data listing task
> INDI n ; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> OPTYPE 'LIST' ↵	to select column listing format
> ANTEN *a1* , 0 ↵	to select one reliable antenna to display.
> BASEL 0 ↵	to select all baselines to this antenna.
> SOURCES ' ' ; CALCODE '*' ↵	to select all calibrator sources only.
> TIMER 0 ↵	to select all times.
> STOKES 'RR' ↵	to examine only one Stokes at a time.
> BIF 1 ; EIF BIF ↵	to specify the "AC" IFs only; it is quicker to look at only 1 IF at a time although more than one can be listed in sequence.
> FREQID 1 ↵	to select FQ number 1 (note that FQ numbers must also be done separately).
> DOCRT 132 ↵	to see full width display on the terminal. Use your window manager to stretch the window to ≥ 132 characters width.
> DOCALIB -1 ↵	to turn off calibration.
> DPARM 0 ↵	to select amplitudes with no averaging.
> INP ↵	to re-check *all* the inputs parameters.
> GO ↵	to start the task.

The task will prompt you for a ↵ after each "page full" of output. When you have seen enough, enter Q. This display will let you determine whether the start-of-scan problem infects your data and, if so, how badly. If it is rare, forget it for now and use manual flagging methods later if needed. If it is widespread, use the *AIPS* task QUACK:

> TASK 'QUACK' ↵	
> SOURCES ' ' ↵	to select all sources.
> TIMER 0 ↵	to select all times.
> ANTENNAS 0 ↵	to select all antennas.
> FLAGVER 1 ↵	to insert flagging information in FG table 1.
> OPCODE 'BEG' ↵	flag first APARM(2) min of each scan.
> REASON 'BAD START OF SCAN' ↵	reason for the flagging.
> APARM 0 , 1/6 , 0 ↵	flag first 10 seconds of each scan.
> GO ↵	

The display generated above will also allow you to determine quickly which antennas are absent, which antennas are present but dead, and, with more careful examination, which antennas are flaky and may need special consideration. "Dead" antennas are visible in this display as columns with small numbers — columns that differ by factors of two or so from the others are generally fine. To be thorough, it is probably best to check the other IF:

```
> BIF 2 ; EIF 2 ↵R          to specify the "BD" IFs.
> GO ↵R                     to run the program again.
```

as well as `STOKES = 'LL'`.

To remove the dead antennas, run `UVFLG`. For example, if antennas 6, 9, and 22 were bad for the full run in both IFs and Stokes, they could be deleted with

```
> TASK 'UVFLG' ; INP ↵R          to select the editor and check its inputs.
> TIMER 0 ↵R                     to select all times.
> BIF 1 ; EIF 2 ↵R               to specify the "AC" and "BD" IFs.
> BCHAN 0 ; ECHAN 0 ↵R           to flag all channels.
> FREQID 1 ↵R                    to flag only the present FQ number.
> ANTEN 6 , 9, 22 ↵R             to select the antennas.
> BASEL 0 ↵R                     to select all baselines to these antennas.
> STOKES '' ↵R                   to select all Stokes.
> REASON = 'ZOMBIE ANTENNA' ↵R   to set a reason.
> FLAGVER 1 ↵R                   to select the first (only) flag table.
> INP ↵R                         be careful with the inputs here!
> GO ↵R                          to run the task when ready.
```

4.3.2 Primary flux density calibrators

The flux densities of 3C286 (1328+307) and 3C48 (0134+329) on the scale of Baars *et al.* (Astr. & Ap., **61**, 99 (1977)) are given in the 1990 VLA Calibrator Manual as:

3C286:

$$\log S = 1.480 + 0.292 \log \nu - 0.124 (\log \nu)^2$$

3C48:

$$\log S = 2.345 + 0.071 \log \nu - 0.138 (\log \nu)^2$$

where S = flux density in Jy and ν is Frequency in MHz. These values are, at a few selected frequencies:

Frequency (MHz)	S 3C286 (Jy)	S 3C48 (Jy)
1465	14.51	15.37
1680	13.55	13.76
4885	7.41	5.36
8415	5.20	3.15
14765	3.48	1.75
15035	3.44	1.71
22485	2.53	1.09

Careful measurements made with the D array of the VLA have shown that the Baars *et al.* (1977) coefficients are in error slightly, based on the assumption that the Baars' expression for 3C295 is correct; see the VLA

Calibrator Manual. Revised values of the coefficients have been derived by Rick Perley. Task SETJY has these formulae built into it, giving you the option (OPTYPE 'CALC') of letting it calculate the fluxes for primary calibrator sources 3C295, 3C48, 3C286, 3C147, 3C138, and 1934-638. The default setting of APARM(2) = 0) will calculate the flux densities of 3C48, 3C147, and 3C286 according to the 1999.2 Perley coefficients, while APARM(2) = 1 will calculate the flux densities using the original Baars *et al.* coefficients. Earlier (1990, 1995) Perley coefficients may also be selected with higher values of APARM(2). SETJY will recognize both the 3C and IAU designations (B1950 and J2000) for these sources. You may insert your own favorite values for these sources instead (OPTYPE = ' ') and you will have to insert values for any other gain calibrators you intend to use.

Unfortunately, since both 3C48 and 3C286 are resolved by the VLA in most configurations and at most frequencies, they cannot be used directly to determine the amplitude calibration of the antennas without a detailed model of the source structure. Beginning in April 2004, model images for the calibrators at some frequencies are included with *AIPS*. As of November 2005, models of 3C286 and 3C48 are available for all bands *except* 90 cm. Type CALDIR ↵ to see a list of the currently available calibrator models. Sources which are small enough to be substantially unresolved by the VLA have variable flux densities which must be determined in each observing session. A common method used to determine the flux densities of the secondary calibrators from the primary calibrator(s) is to compare the amplitudes of the gain solutions from the procedure described below.

Use SETJY to enter/calculate the flux density of each primary flux density calibrator. The ultimate reference for the VLA is 3C295, but 3C286 (1328+307), which is slightly resolved in most configurations at most frequencies, is the most useful primary calibrator. CALIB has an option that will allow you to make use of Clean component models for calibrator sources. You are strongly encouraged to use the existing models. If you follow past practice at the VLA, you may have to restrict the *uv* range over which you compute antenna gain solutions for 3C286, and may therefore insert a "phony" flux density appropriate only for that *uv* range at this point. In both cases, the following step should be done. CALIB will scale the total flux of the model to match the total flux of the source recorded by SETJY in the source table. This corrects for the model being taken at a somewhat different frequency than your observations and for the model containing most, but not all, of the total flux. An example of the inputs would be:

```
> TASK 'SETJY' ; INP ↵
> SOURCES '3C286' , ' ' ↵          if you used 3C286 as the source name.
> ZEROSP 7.41 , 0 ↵                I flux 7.41 Jy, Q, U, V fluxes 0.
> BIF 1 ; EIF 1 ↵                  selects "AC" IF.
> INP ↵                            to review inputs.
> OPTYPE ' '                       use values given in ZEROSP.
> GO ↵                             when inputs okay.
> BIF 2 ; EIF 2 ↵                  selects "BD" IF.
> ZEROSP 7.46, 0 ↵                 I flux 7.46 Jy at the 2nd IF, Q, U, V fluxes 0.
> GO ↵
```

Note that, although SOURCES can accept a source list, ZEROSP has room for only one set of I, Q, U, V flux densities. To set the flux densities for several different sources or IFs, you must therefore rerun SETJY for each source and each IF, changing the SOURCES, BIF, EIF, and ZEROSP inputs each time.

If you wish, you can let SETJY calculate the fluxes, in which case it is able to do both IFs together.

```
> TASK 'SETJY' ; INP ↵
> SOURCES '3C286' , ' ' ↵          if you used 3C286 as the source name.
> BIF 1 ; EIF 2 ↵                  will calculate for both "AC" and "BD" IFs.
> OPTYPE 'CALC' ↵                  perform the calculation.
> APARM(2) = 0 ↵                   to use the VLA "1990" coefficients.
> INP ↵                            to review inputs.
```

> GO ↵ when inputs okay.

CALIB will use the V polarization flux in the source table if one has been entered. The the RR polarization will be calibrated to I+V and the LL to I-V. While this has little practical use with circular poolarizations because V is almost always negligible, it can be used for linearly polarized data from the WSRT. That telescope has equatorially mounted dishes, so the XX polarization is I-Q and the YY is I+Q independent of parallactic angle. For WSRT data, you should relabel the polarizations to RR/LL and enter `I, 0, 0, -Q` for ZEROSP, since Q is not negligible in standard calibrators.

4.3.3 First pass of the gain calibration

4.3.3.1 Using calibrator models

Since April 2004, source models have been shipped with *AIPS* as FITS files. Initially, they are available for 3C48, 3C138, 3C147, and 3C286 at K, Q, and U bands. As of November 2005, models for 3C48 and 3C286 are available for all bands except 90 cm. Additional models are in the works, so you should always check to see what is available:

> CALDIR ↵ to list the available models by source name and band code.

Then to load a model:

> TASK 'CALRD' ↵ to select the calibrator source reading task.
> OBJECT '3C286' ↵ to load a model of 3C286.
> BAND 'K' ↵ to select the available model at K band.
> OUTDISK n ↵ to write the model image and Clean components to disk n.
> GO ↵ to run the task and load the model.

Then you may select the model image with GET2N for use in CALIB. Note that the procedure VLACALIB does not allow you to use a source model for the calibrator. Example inputs for CALIB are:

> TASK 'CALIB'; INP ↵ to select task and review inputs.
> INDI n ; GETN m ↵ to select the data set, $n = 3$ and $m = 1$ above.
> CALSOUR = '*Cala*' , ' ' ↵ flux calibrator for which you have a model.
> UVRANGE 0 ↵ no *uv* limits needed.
> ANTENNAS 0 ↵ antenna selection not needed.
> REFANT n ↵ reference antenna number — use a reliable antenna located near the center of the array.
> WEIGHTIT 1 ↵ select $1/\sigma$ weights, may be more stable.
> IN2DI o ; GET2N p ↵ to select the model.
> NCOMP 0 ↵ use all components.
> SOLMODE 'A&P' ↵ do amplitude and phase solutions.
> APARM(6) 2 ↵ print closure failures.
> MINAMPER 10 ↵ display warning if baseline disagrees in amplitude by more than *10%* from the model.
> MINPHSER 10 ↵ display warning if baseline disagrees by more than 10° of phase from the model.
> CPARM(5) 1 ↵ scalar average amplitudes before determining solution.
> FREQID 1 ↵ use FQ number 1.
> INP CALIB ↵ to review inputs.
> GO ↵ to make the solution.

CALIB will use the clean components table attached to the model to find antenna gain solutions. It will sum the clean components within a certain radius of the center of the map (so that confusing sources that are part of the model do not influence the gain) and scale them to the flux in the SU table. Therefore, you must still run SETJY before running CALIB.

After running CALIB check the solutions for all antennas with SNPLT or LISTR (OPTYPE='GAIN'). If you have multiple primary or secondary calibrators you will have to run CALIB separately for each, using models where they are available and restricting the UVRANGE and ANTENNAS where they are not. You can either write into the same SN table by setting SNVER to a table number or to different SN tables by setting SNVER = 0. Then you you can proceed as normal flagging and editing your data and proceed to final calibration as described in § 4.5.

4.3.3.2 Not using calibrator models

We strongly encourage you to use the available models. If you must do it the old-fashioned way, then you have to limit the *uv* ranges and antennas to where the calibrator is a point source. The range of baseline length used can be controlled by the adverb UVRANGE. If there are too few baselines to a given antenna, accurate solutions may not be possible; therefore, it is frequently necessary to limit the antennas used to the inner antennas on each arm. (The antenna pad numbers which include the order number from the array center on each arm can be determined by running PRTAN; see § 4.2.2.) CALIB may fail to produce any valid solutions if antennas with no data are included in the solution. The VLA Calibrator Manual suggests the following sets of UVRANGE (in kiloλ) and inner number of antennas. Most of the complexity of CALIB can be hidden using the procedure VLACALIB. Before attempting to invoke this procedure, you must first

> RUN VLAPROCS ↵ to compile the procedures.

Both 3C286 and 3C48 are resolved by the VLA in some configurations and frequencies. Good models for these sources are available at a few frequencies; see CALRD below. Point models for these sources are only accurate over a limited range of baseline length. The range of baseline length used can be controlled by the adverb UVRANGE. Better models are now available for some sources and bands; see below. If there are too few baselines to a given antenna, accurate solutions may not be possible; therefore, it is frequently necessary to limit the antennas used to the inner antennas on each arm. (The antenna pad numbers which include the order number from the array center on each arm can be determined by running PRTAN; see § 4.2.2.) CALIB may fail to produce any valid solutions if antennas with no data are included in the solution. The VLA Calibrator Manual suggests the following sets of UVRANGE (in kiloλ) and inner number of antennas.

3C48, 3C147, 3C138:

Band	UVRANGE	Array	No. ant. per arm	Notes
90cm	0- 40	All	All	
20cm	0- 40	A	7	
	"	B,C,D	All	
6cm	0- 40	A	3	
	"	B,C,D	All	
3.6cm	0- 40	A	2	
	"	B	6	
	"	C,D	All	
2cm	0- 40	A	1	Not recommended
	"	B	4	

	"	C,D	All	
1.3cm	0- 40	A	1	Not recommended
	"	B	3	
	"	C,D	All	

3C286:

Band	UVRANGE	Array	No. ant. per arm	Notes
90cm	0- 18	A	7	
	"	B,C,D	All	
20cm	0- 18	A	4	
	"	B,C,D	All	
	90-180	A	All	Reduce flux 6%
6cm	0- 25	A	1	Not recommended
	"	B	4	
	"	C,D	All	
	150-300	A	All	Reduce flux 2%
3.6cm	50-300	A	3	Reduce flux 1%
	"	B	7	Reduce flux 1%
	"	C	All	Reduce flux 1%
	0- 15	D	All	
2cm	0-150	A	3	
	"	B,C,D	All	
1.3cm	0-185	A	2	
	"	B	7	
	"	C,D	All	

The values of UVRANGE for each secondary calibrator may be determined from the VLA Calibrator manual or by using UVPLT to plot the amplitudes as a function of baseline length. Since the latter works correctly only after a complete calibration has been done, it is often reasonable to use the 3C286/3C48 restrictions for all calibrator sources (at this stage). If your secondary calibrators are point sources over most baselines, then it may save you time to do the full calibration now. Not only will it save you, possibly, from re-running CALIB at a later time with a wider UVRANGE, but it will provide information on the data quality from the longer baselines.

Once you have read in procedure VLACALIB, you may use it to invoke CALIB. You will have to do this once for each calibrator, unless you can use the same UVRANGE for more than one of them. Thus,

> INDI n ; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> CALSOUR = '*Cala*' , '*Calx*' ↵	to name two calibrators using the same UVRANGE and other adverb values.
> UVRANGE *uvmin uvmax* ↵	*uv* limits, if any, in kiloλ.
> ANTENNAS *list of antennas* ↵	antennas to use for the solutions, see discussion above.
> REFANT n ↵	reference antenna number — use a reliable antenna located near the center of the array.

> MINAMPER 10 ↵	display warning if baseline disagrees in amplitude by more than *10%* from the model.
> MINPHSER 10 ↵	display warning if baseline disagrees by more than 10° of phase from the model.
> DOPRINT 1 ; OUTPRINT ' ' ↵	to generate significant printed output on the line printer.
> FREQID 1 ↵	use FQ number 1.
> INP VLACALIB ↵	to review inputs.
> VLACALIB ↵	to make the solution and print results.

This procedure will first run CALIB, then print any messages from CALIB about closure errors on the line printer, and finally run LISTR to print the amplitudes and phases of the derived solutions. Plots of these values may be obtained using task SNPLT.

If the secondary calibrators require different values of UVRANGE, then CALIB must be run until it has run for all calibration sources. Attached to your input data set is a solution SN table. Each run of CALIB writes in this table (if SNVER = 1, for the times of the included calibration scans, the solutions for both the "AC" and "BD" IFs using the flux densities you set for your calibrators with SETJY or GETJY. (CALIB assumes a flux density of 1 Jy if no flux density is given in the SU table.) If a solution fails, however, the whole SN table can be compromised, forcing you to start over. It is possible to write multiple SN tables with SNVER = 0. Later programs such as GETJY and CLCAL will merge all SN tables which they find (if told to do so). Tables with failed solutions must be deleted.

The LISTR outputs provided by VLACALIB should be examined carefully to check on the calibration; amplitudes should be consistent (both among antennas and among time stamps) and phases should vary smoothly. If you decide that the solutions are not acceptable (*e.g.*, there are no valid solutions) *and* you are creating a new SN table on each run of CALIB, then delete that SN table using EXTDEST before proceeding. The later stages of processing assume that all extant SN tables are valid. Note that re-running CALIB on the same SN table simply over-writes the old solutions with new ones. CALIB gives messages which indicate the number of valid and invalid solutions which should help you evaluate the results. If VLACALIB is run using the values of MINAMPER and MINPHSER shown above, it will print a list of baselines and times which show substantial "closure" errors. (If you use CALIB directly rather than VLACALIB, you may use these adverbs plus CPARM(2-4) to get additional reports and statistics on closure errors.) It is important to remember that normal thermal noise and, at longer wavelengths, background confusion cause closure errors too. Thus, some closure error on weaker calibrators is to be expected and may be ignored. Interpreting closure errors is a real art, but a couple of generalizations are possible. If the same closure error shows up in both polarizations and both IFs, then you have probably got a resolved object. If one antenna dominates the closure list, especially if it is at only one IF and/or one polarization, then you have got a bad antenna. If the errors are uniformly small, distributed amongst all antennas, and not correlated between IFs or polarizations, then you have simply noise and/or background confusion. In tis case, do *not* edit the data — the randomness of the "errors" nearly always averages out nicely and the solution is just fine. Large or systematic errors indicate either that the calibrator source is resolved or that there are problems with the data requiring editing. If a calibrator is being resolved, delete the bad SN table and re-run VLACALIB with an appropriate UVRANGE. In the 31DEC04 release, one can actually flag data based on closure errors using the DOFLAG option. This should be used carefully, if at all and then only if the model of the calibration source contains all of its flux. Modern versions of CALIB display some statistics of closure problems automatically and allow the use of "robust" solution methods.

4.4 Assessing the data quality and initial editing

At each stage in the data calibration process, it is a good idea to take a look at the data to determine their quality and then to "flag" (edit, delete) those that are suspect or clearly bad. Having begun the actual calibration, it is important to get an impression of the overall quality of the data and to edit out any obviously corrupted data, (*e.g.*, bad integrations that were not detected and expunged by the on-line monitoring system, high amplitudes due to interference, unstable amplitudes due to undetected equipment problems, *etc*). During the initial calibration, you need to do this only on the observations of calibration sources. However, at a later stage, you may also need to apply techniques similar to those described below to your program sources. If you do edit any calibration data at this point, you must re-run CALIB following the instructions given above for the affected sources.

The philosophy of editing and the choice of methods are matters of personal taste and the advice given below should, therefore, be taken with a few grains of salt. When interferometers consisted of only a couple of movable antennas, there was very little data and it was sparsely sampled. At that time, careful editing to delete all suspect samples, but to preserve all samples which can be calibrated, was probably justified. But modern instruments produce a flood of data, with the substantial redundancy that allows for self-calibration on strong sources. Devoting the same care today to editing is therefore very expensive in your time, while the loss of data needlessly flagged is rarely significant. A couple of guidelines you might consider are:

- Don't flag on the basis of phase. At least with the VLA, most phase fluctuations are due to the atmosphere rather than the instrument. Calibration can deal with these up to a point, and self-calibration (if you have enough signal) can refine the phases to levels that you would never reach by flagging. The exceptions are (1) IF phase jumps which still happen on rare occasions, and (2) RF interference which sometimes is seen as an excursion in phase rather than amplitude.
- Don't flag on minor amplitude errors, especially if they are not common. Except for very high dynamic range imaging, these will not be a problem, and in those cases, self-calibration always repairs or sufficiently represses the problem.
- Don't flag if CALIB reports few closure errors and the SN tables viewed with EDITA, SNPLT, and LISTR and the calibrator data viewed with the matrix format of LISTR show only a few problems.

There are two general methods of editing in *AIPS*. The "old-fashioned" route uses LISTR to print listings of the data on the printer or the user's terminal. The user scans these listings with his eyes and, upon finding a bad point, enters a specific flag command for the data set using UVFLG. While this may sound clumsy, it is in fact quite simple and by far the faster method when there are only a few problems. In a highly corrupted data set, it can use a lot of paper and may force you to run LISTR multiple times to pin down the exact problems. The "modern" route uses interactive ("TV"-based) tasks to display the data in a variety of ways and to allow you to delete sections of bad data simply by pointing at them with the TV cursor. These tasks are TVFLG (§ 4.4.3) for all baselines and times (but only one IF, one Stokes, and one spectral channel at a time), SPFLG (§ 10.2.2) for all spectral channels, IFs, and times (but only one baseline and one Stokes at a time), EDITA (§ 4.4.2) for editing based on TY (T_{ant}), SN or CL table values, EDITR (§ 5.5.2) for all times (but only a single antenna (1–11 baselines) and one channel average at a time) and WIPER for all types of data (but with the origin of the points not available while editing). TVFLG is the one used for continuum and channel-0 data from the VLA, while SPFLG is only used to check for channel-dependent interference. SPFLG is useful for spectral-line editing in smaller arrays, such as the Australia Telescope and the VLBA. (The redundancy in the spectral domain on calibrator sources helps the eyes to locate bad data.) EDITR is more useful for small arrays such as those common in VLBI experiments. EDITA has been found to be remarkably effective using VLA system temperature tables. All four tasks have the advantage of being very specific in displaying the bad data. Multiple executions should not be required. However, they may require you to look at each IF, Stokes, channel (or baseline) separately (unless you make certain broad assumptions); EDITA and EDITR do allow you to look at all polarizations and/or IFs at once if you want. They all require you

to develop special skills since they offer so many options and operations with the TV cursor (mouse these days). A couple of general statements can be made

- For highly corrupted data (say with considerable RF interference, significant cross-talk between antennas, or erratic antennas) TVFLG is definitely preferred. It gives an overall view of the data which is far superior to that given by LISTR. RFI and similar problems are more troublesome at lower frequencies, so TVFLG is probably preferred for L, P, and "4" bands.
- Most VLA data at higher frequencies are of good quality and the flexibility of TVFLG is not needed. In such cases, LISTR with OPCODE = 'MATX' can find scans with erroneous points efficiently.
- The displays given by TVFLG and, to a lesser extent, LISTR in its MATX mode are less useful when there are only a few baselines. Thus, for arrays smaller than the VLA, users may wish to use SPFLG on spectral-line data sets and EDITR on continuum data sets.
- A reasonable strategy to use is to run LISTR first. If there are only a few questionable points, use LISTR and UVFLG, otherwise switch to an interactive task, such as EDITA followed by TVFLG.
- Task FLAGR is a new, somewhat experimental task to measure the rms in the data on either a baseline or an antenna basis and then delete seriously discrepant points and times when many antennas/correlators are questionable. It also clips amplitudes and weights which are outside specified normal ranges. Task FINDR reports the rmses and excessive values to assis in running FLAGR.
- Task CLIPM makes entries in a flag table, applying calibration and then testing amplitudes for reasonableness on a source-by-source basis. It can be very useful for large data sets, but does not show you the bad data to evaluate yourself.
- Task DEFLG makes entries in a flag table whenever the phases are too variable as measured by too low a ratio of vector-averaged to scalar-averaged amplitudes. This may be useful when applied to the calibrator source in phase-referencing observations and for other data at the highest and lowest frequencies which are affected by atmospheric and ionospheric phase variability.
- Task SNFLG makes entries in a flag table whenever the phase solutions in an SN or CL table change excessively between samples on a baseline basis.
- Task WIPER makes entries in a flag table for all data samples wiped from a UVPLT-like display of any *uv* dataset parameter versus any other parameters. The source, Stokes, IF, time, baseline, etc. of the points are not known during the interactive editing phase.
- Task WETHR makes entries in a flag table whenever various weather parameters exceed specified limits. WETHR also plots the weather (WX) table contents.
- Task VPFLG flags all correlators in a sample whenever one is flagged. Observations of sources with circular polarization (Stokes V) require this operation to correct the flagging done on-line (which flags only known bad correlators).
- Task FGPLT plots the times of selected flag-table entries to provide you information on what these powerful tasks have done.

4.4.1 Editing with LISTR and UVFLG

Data may be flagged using task UVFLG based on listings from LISTR. To print out the scalar-averaged raw amplitude data for the calibrators, and their *rms* values, once per scan in a matrix format, the following inputs are suggested:

```
> TASK 'LISTR' ; INP ↵R          to review the inputs needed.
```

> INDI n; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> SOURCES ' ' ; CALCODE '*' ↵	to select calibrators.
> TIMER 0 ↵	to select all times.
> ANTENNAS 0 ↵	to list data for all antennas.
> OPTYPE 'MATX' ↵	to select matrix listing format.
> DOCRT FALSE ↵	to route the output to printer, not terminal.
> DPARM 3 , 1 , 0 ↵	amplitude and *rms*, scalar scan averaging.
> BIF 1 ↵	to specify the "AC" IFs (note that IFs must be listed separately).
> FQID 1 ↵	to select FQ number 1 (note that FQ numbers must also be done separately).
> INP ↵	to review the inputs.
> GO ↵	to run the program when inputs set correctly.
> BIF 2 ↵	to specify the "BD" IFs.
> WAIT ; GO ↵	to wait for the previous execution to finish and then run the program again.

For unresolved calibrators, the VLA on-line gain settings normally produce roughly the same values in all rows and columns within each matrix. At L, C, X, and U bands, these values should be approximately 0.1 of the expected source flux densities. At P band, the factor is about 0.01. The factors for other bands are unspecified. Any rows or columns with consistently high or low values in either the amplitude or the *rms* matrices should be noted, as they probably indicate flaky antennas. In particular, you should look for

- In the amp-scalar averages, look for *dead* antennas, which are easily visible as rows or columns with small numbers. Rows or columns that differ by factors of two or so from the others are generally fine. Such deviations mean only that the on-line gains were not set entirely correctly.

- In the *rms* listings, look for discrepant high values. Almost all problems are antenna based and will be seen as a row or column. Factors of 2 too high are normally okay, while factors of 5 high are almost certainly indicative of serious trouble.

The next step is to locate the bad data more precisely. Suppose that you have found a bad row for antenna 3 in right circular polarization in IF 2 between times (*d1*, *h1*, *m1*, *s1*) and (*d2*, *h2*, *m2*, *s2*). You might then rerun LISTR with the following new inputs:

> SOURCES ' ' ↵	to select all sources.
> TIMER *d1 h1 m1 s1 d2 h2 m2 s2* ↵	to select by time range.
> ANTENNAS 1 , 2 , 3 ↵	to list data for antenna 3 with two "control" antennas.
> BASEL 1 , 2 , 3 ↵	to list all baselines with these three antennas.
> OPTYPE 'LIST' ↵	to select column listing format.
> DOCRT 1 ↵	to route the output to terminal at its width.
> DPARM = 0 ↵	amplitude only, no averaging.
> STOKES 'RR' ↵	to select right circular.
> BIF 2 ↵	to specify the "BD" IFs.
> FLAGVER 1 ↵	to choose flag table 1.
> GO ↵	to run the program.

This produces a column listing on your terminal of the amplitude for baselines 1–2, 1–3 and 2–3 at every time stamp between the specified start and stop times. The '1–2" column provides a control for comparison with the two columns containing the suspicious antenna.

Note that "amp-scalar" averaging ignores phase entirely and is therefore not useful on weak sources, nor can it find jumps or other problems with the phases. To examine the data in a phase-sensitive way, repeat the above process, but set DPARM(2) = 0 rather than 1. Bad phases will show up as reduced amplitudes and increased *rms*'s.

Once bad data have been identified, they can be expunged using UVFLG. For example, if antenna 3 RR was bad for the full interval shown above, it could be deleted with

> TASK 'UVFLG' ; INP ↵	to select the editor and check its inputs.
> TIMER *d1 h1 m1 s1 d2 h2 m2 s2* ↵	to select by time range.
> BIF 2 ; EIF = BIF ↵	to specify the "BD" IFs.
> BCHAN 0 ; ECHAN 0 ↵	to flag all channels.
> FREQID 1 ↵	to flag only the present FQ number.
> ANTEN 3 , 0 ↵	to select antenna 3.
> BASEL 0 ↵	to select all baselines to antenna 3.
> STOKES 'RR' ↵	to select only the RR Stokes (LL was found to be okay in this example).
> REASON = 'BAD RMS WHOLE SCAN' ↵	to set a reason.
> FLAGVER 1 ↵	to select the first (only) flag table.
> INP ↵	be careful with the inputs here!
> GO ↵	to run the task when ready.

Continue the process until you have looked at all parts of the data set that seemed anomalous in the first matrix listing, then rerun that listing to be sure that the flagging has cleaned up the data set sufficiently. If there are lots of bad data, you may find that you have missed a few on the first pass. If you change your mind about a flagging entry, you can use UVFLG with OPCODE = 'UFLG' to remove entries from the flag table. (Note that, if you use different REASONs for your different flag entries, then you can also undo all flags with a given REASON using OPCODE = 'REAS' in UVFLG.) If the table becomes hopelessly messed up, use EXTDEST to delete the flag table and start over or use a higher numbered flag table. The contents of the flag table may be examined at any time with the general task PRTAB and entries in it may also be removed with TABED and/or TAFLG.

4.4.2 Editing with EDITA

The task EDITA uses the graphics planes on the *AIPS* TV display to plot data from tables and to offer options for editing (deleting, flagging) the associated *uv* data. At this time, only the TY (system temperature), SN (solution), and CL (calibration) tables may be used. We recommend using EDITA with the TY tables to do the initial editing of VLA data sets, probably before running the programs described in § 4.3. For accuracy in evaluating and flagging your data, it is a good idea to have the TY table filled with the same interval as the data themselves; see § 4.1.1. Try:

> TASK 'EDITA ; INP ↵	to review the inputs needed.
> INDI n ; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> INEXT 'TY' ↵	to use the system temperature table.
> INVERS 0 ↵	to use the highest numbered table, usually 1.
> TIMER 0 ↵	to select all times.
> FREQID 3 ↵	Select FQ entry 3.
> BIF 1 ; EIF 0 ↵	to specify all IFs; you can then toggle between them interactively and even display all at once.
> ANTENNAS 0 ↵	to display data for all antennas.

> ANTUSE 1 , 2 , 3 , 4 , 5 , 6 , 7 ↵	to display initially the first 7 antennæ, editing antenna 1. Others may be selected interactively.
> FLAGVER 1 ↵	to use flag (FG) table 1.
> SOLINT 0 ↵	to avoid averaging any samples.
> DOHIST FALSE ↵	to omit recording the flagging in the history file.
> DOTWO TRUE ↵	to view a 2nd observable for comparison
> CROWDED TRUE ↵	to allow plots with all polarizations and/or IFs simultaneously.
> INP ↵	to review the inputs.
> GO ↵	to run the program when inputs set correctly.

If you make multiple runs of EDITA, it is important to make sure that the flagging table entries are all in the same version of the FG table. To ensure this you should set FLAGVER to 1 and keep it that way for all runs of EDITA. A sample display from EDITA is shown on the next page.

The following discussion assumes that you have read § 2.3.2 and are familiar with using the *AIPS* TV display. An item in a menu such as that shown in the figure is selected by moving the TV cursor to the item (holding down or pressing the left mouse button). At this point, the menu item will change color. To obtain information about the item, press *AIPS* TV "button D" (usually the D key and also the F6 key on your keyboard). To tell the program to execute the menu item, press any of *AIPS* TV buttons A, B, or C. Status lines around the display indicate what is plotted and which data will be flagged by the next flagging command. In the figure below, only the displayed antenna (2), and time range will be flagged. You must display at least a few lines of the message window and your main AIPS window since the former will be used for instructions and reports and the latter will be needed for data entry (*e.g.*, antenna selection).

The first thing to do with EDITA is to look at all of the polarizations, IFs, and antennæ, in order to flag the obviously bad samples (if any). Use SWITCH POLARIZATION to switch between polarizations and ENTER IF to select the IF to edit. Alternatively, NEXT CORRELATOR will cycle through all polarizations and IFs. If CROWDED was set to true, SWITCH POLARIZATION will cycle through displaying both polarizations as well as each separately, and ENTER IF will accept 0 as indicating all. NEXT CORRELATOR shows only one correlator at a time, but can switch away from a multi-correlator display. These options appear only if there is more than one polarization and/or more than one IF in the loaded data. Use ENTER ANTENNA to select the antenna to be flagged and ENTER OTHER ANT to select secondary antennæ to be displayed around the editing area. If the secondary antennæ have no obvious problems, then they do not have to be selected for editing. EDITA will plot all of the times in the available area, potentially making a very crowded display. You may select interactively a smaller time range or "frame" in order to see the samples more clearly. It is necessary to select each frame in order to edit the data in that frame so it helps to make the TV screen as big as possible with the F2 button or your window manager. Note that the vertical scales used by EDITA are linear, but that the horizontal scale is irregular and potentially discontinuous. Integer hours are indicated by tick marks and the time range of the frame is indicated. Use FLAG TIME or FLAG TIME RANGE to delete data following instructions which will appear on the message window. While you are editing, the source name, sample time and sample value currently selected will be displayed in the upper left corner of the TV screen. This information can also be used to determine if QUACK is needed. In the case of the data displayed in the figure, it was found that the source name changed to the source of the new scan *before* the T_{sys} reached the value appropriate to the source. This is seen in the scan with higher T_{sys} on the good antennæ, while antenna 1 seems to be dubious over the last half of the observation. Therefore, besides heavy editing of antenna 1 polarization 2, QUACK was needed with these data (usually the case for spectral-line VLA observations).

Having flagged all obviously bad points, select SWITCH ALL IF, SWITCH ALL TIME, SWITCH ALL ANT, and SWITCH ALL POL so that the next flag command(s) apply to all of the data. Set the SCAN LENGTH long enough to include the shorter of the full scan and about 12 samples. Then display the difference between the current sample and the running mean by selecting SHOW TSYS - <T>. Use FLAG ABOVE and FLAG BELOW to flag all samples more than a few sigma away from the local mean. Finally, apply your flagging to your *uv* data set by selecting EXIT.

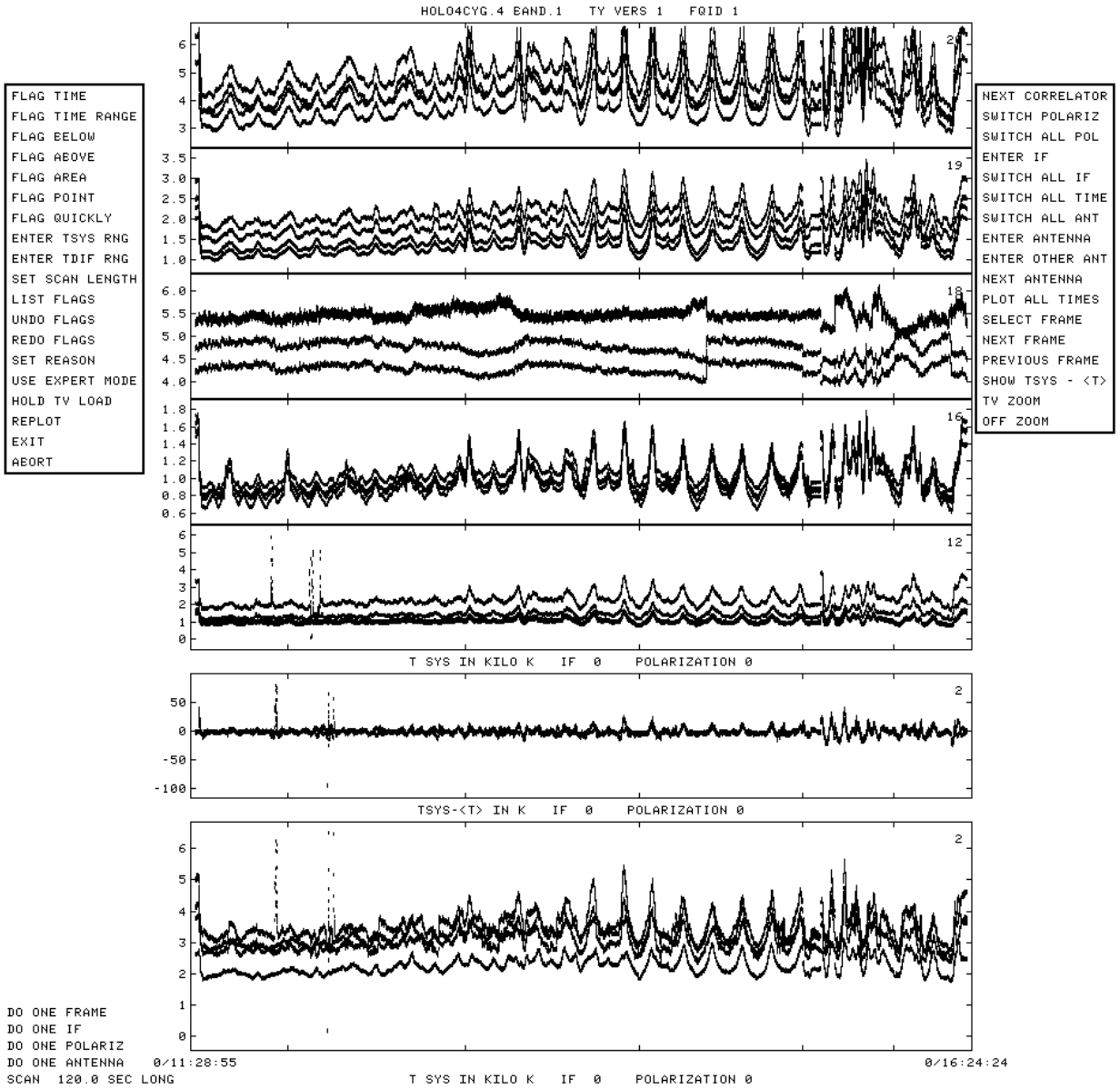

Figure 4.1: A display of a sample TV screen from EDITA, made using the *AIPS* task TVCPS to produce a negative black-and-white display. The EDITA menu (in the boxes), the status lines (at the bottom), the editing area (bottom) of a portion of the data from the selected antenna (2), the subsidiary plots of data from selected secondary antennæ (12, 16, 18, 19, 20), the edit tool (bar or box), and the edit location values are displayed in different graphics planes which normally appear in different colors. In this example, with CROWDED=TRUE, two IFs and two polarizations are displayed and may be edited simultaneously.

At this point, return to § 4.3.1 to run QUACK followed by the first pass of the gain calibration. Then run TVFLG below with DOCAL TRUE so that the data will be displayed on the same flux scale for all baselines.

4.4.3 Editing with TVFLG

If your data are seriously corrupted, contain numerous baselines, and you like video games, TVFLG is the visibility editor of choice. The following discussion assumes that you have read § 2.3.2 and are familiar with using the *AIPS* TV display. The following inputs are suggested:

> TASK 'TVFLG' ; INP ↵	to review the inputs needed.
> INDI n ; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> SOURCES ' ' ↵	to select all sources.
> TIMER 0 ↵	to select all times.
> STOKES 'RRLL' ↵	to select both right and left circular polarizations; you can then toggle between RR and LL interactively.
> FREQID 3 ↵	Select FQ entry 3.
> BIF 1 ; EIF 2 ↵	to specify both VLA IFs; you can then toggle between the two interactively.
> ANTENNAS 0 ↵	to display data for all antennas.
> BASELINE 0 ↵	to display data for all baselines.
> DOCALIB 2 ↵	to apply initial calibration to the data.
> FLAGVER 1 ↵	to use flag (FG) table 1.
> DPARM = 0 ↵	to use default initial displays and normal baseline ordering.
> DPARM(6) = 30 ↵	to declare that the input data are 30-second averages, or to have the data averaged to 30 seconds.
> DPARM(5) = 10 ↵	to expand the flagging time ranges by 10 seconds in each direction. The times in the master grid are average times and may not encompass the times of the samples entering the average without this expansion.
> DOCAT 1 ↵	to save the master grid file.
> INP ↵	to review the inputs.
> GO ↵	to run the program when inputs set correctly.

If you make multiple runs of TVFLG, it is important to make sure that the flagging table entries are all in the same version of the FG table. To ensure this you should set FLAGVER to 1 and keep it that way for all runs of TVFLG.

TVFLG begins by constructing a "master grid" file of all included data. This can be a long process if you include lots of data at once. It is probably better to use the channel selection, IF selection, source selection, and time range selection adverbs to build rather smaller master grid files and then to run TVFLG multiple times. It will work with all data included, allowing you to select interactively which data to edit at any one moment and allowing you to resume the editing as often as you like. But certain operations (such as undoing flags) have to read and process the entire grid, and will be slow if that grid is large. The master grid file is always cataloged (on IN2DISK with class TVFLGR), but is saved at the end of your session only if you set DOCAT = 1 (actually > 0) before starting the task. To resume TVFLG with a pre-existing master grid file, set the adverb IN2SEQ (and IN2DISK) to point at it. When resuming in this way, TVFLG ignores all of its data selection adverbs since they might result in a different master grid than the one it is going to use. If you wish to change any of the data selection parameters, *e.g.*, channels, IFs, sources, times, or time averaging, then you must use a new master grid.

Kept with the master grid file is a special file of TVFLG flagging commands. This file is updated as soon as

you enter a new flagging command, making the master grid and your long editing time virtually proof from power failures and other abrupt program terminations. These flagging commands are not entered into your actual *uv* data set's flagging (FG) table until you exit from TVFLG and tell it to do so. During editing, TVFLG does not delete data from its master grid; it just marks the flagged data so that they will not be displayed. This allows you to undo editing as needed during your TVFLG session(s). When the flags are transferred to the main *uv* data set, however, the flagged data in the master grid are fully deleted since undoing the flags at that point has no further meaning. When you are done with a master grid file, be sure to delete it (with ZAP) since it is likely to occupy a significant amount of disk.

TVFLG keeps track of the source name associated with each row of data. When averaging to build the master grid and to build the displayed grids, TVFLG will not average data from different sources and will inform you that it has omitted data if it has had to do so for this reason. For multi-source files, the source name is displayed during the CURVALUE-like sections. However, the flagging table is prepared to flag *all* sources for the specified antennas, times, *etc.* or just the displayed source. If you are flagging two calibrator scans, you may wish to do all source in between as well. Use the SWITCH SOURCE FLAG interactive option to make your selection before you create flagging commands. Similarly, you will need to decide whether flagging commands that you are about to prepare apply only to the displayed channel and/or IF, or to all possible channels and/or IFs. In particular, spectral-line observers often use TVFLG on the pseudo-continuum "channel-0" data set, but want the resulting flags to apply to all spectral channels when copied to the spectral-line data set. They should be careful to select all channels before generating any flagging commands. Each flagging command generated is applied to a list of Stokes parameters, which *does not have to include* the Stokes currently being displayed. When you begin TVFLG and whenever you switch displayed Stokes, you should use the ENTER STOKES FLAG option to select which Stokes are to be flagged by subsequent flagging commands.

If you get some of this wrong, you can use the UNDO FLAGS option in TVFLG if the flags have not yet been applied to the *uv* data set. Or you can use tasks UVFLG, TABED or TAFLG to correct errors written into the FG table of your multi-source *uv* data set. It is rather harder to undo errors if you use TVFLG on a single-source data set since the flagging commands are applied directly (and destructively) to the data. For this reason, we normally recommend that you use TVFLG on multi-source data sets, converting single-source ones with MULTI before running TVFLG.

TVFLG displays the data, for a single IF, channel, and Stokes, as a grey-scale display with time increasing up the screen and baseline number increasing to the right. Thus baselines for the VLA run from left to right as 1–1, 1–2, 1–3, ..., 2–2, 2–3, ..., 27–27, 27–28, and 28–28. An input parameter (DPARM(3) = 1 allows you to create a master grid and display baselines both as, say 1–2 and 2–1. An interactive (switchable) option allows you to order the baselines from shortest to longest (ignoring projection effects) along the horizontal axis.

The interactive session is driven by a menu which is displayed on a graphics overlay of the TV display. An example of this full display is shown on the next page. Move the cursor to the desired operation (noting that the currently selected one is highlighted in a different color on many TVs) and press button A, B, or C to select the desired operation; pressing button D produces on-line help for the selected operation. The first (left-most column) of choices is:

OFFZOOM	turn off any zoom magnification
OFFTRANS	turn off any black & white enhancement
OFFCOLOR	turn off any pseudo-coloring
TVFIDDLE	interactive zoom, black & and white enhancement, and pseudo-color contours as in AIPS
TVTRANSF	black & white enhancement as in AIPS
TVPSEUDO	many pseudo-colorings as in AIPS
DO WEDGE ?	switches choice of displaying a step wedge
LIST FLAGS	list selected range of flag commands
UNDO FLAGS	remove flags by number from the FC table master grid

REDO FLAGS	re-apply all remaining flags to master grid
SET REASON	set reason to be attached to flagging commands

Note: when a flag is undone, all cells in the master grid which were first flagged by that command are restored to use. Flag commands done after the one that was undone may also, however, have applied to some of those cells. To check this and correct any improperly un-flagged pixels, use the `REDO FLAGS` option. This option even re-does `CLIP` operations! After an `UNDO` or `REDO FLAGS` operation, the TV is automatically re-loaded if needed. Note that the `UNDO` operation is one that reads and writes the full master grid.

Column 2 offers type-in controls of the TV display and controls of which data are to be flagged. In general, the master grid will be too large to display on the TV screen in its entirety. The program begins by loading every $n^{\rm th}$ baseline and time smoothing by m time intervals in order to fit the full image on the screen. However, you may select a sub-window in order to see the data in more detail. You may also control the range of intensities displayed (like the adverb `PIXRANGE` in `TVLOD` inside `AIPS`). The averaging time to smooth the data for the TV display may be chosen, as may the averaging time for the "scan average" used in some of the displays. Which correlators are to be flagged by the next flagging command may be typed in. All of the standard Stokes values, plus any 4-bit mask may be entered. The spectral channel and IF may be typed in. Flagging may be done only for the current channel and IF and source, or it may be done for all channels and/or IFs and/or sources. Note that these controls affect the next `LOAD`s to the TV or the flagging commands prepared after the parameter is changed. When the menu of options is displayed at the top of the TV, the current selections are shown along the bottom. If some will change on the next load, they are shown with an asterisk following. Column 2 contains

ENTER BLC	Type in a bottom left corner pixel number on the terminal
ENTER TRC	Type in a top right corner pixel number on the terminal
ENTER AMP PIXRANGE	Type in the intensity range to be used for loading amplitude images to the TV
ENTER PHS PIXRANGE	Type in the phase range to be used for loading phase images to the TV
ENTER RMS PIXRANGE	Type in the intensity range to be used for loading images of the rms to the TV
ENTER R/M PIXRANGE	Type in the value range to be used for loading rms/mean images to the TV
ENTER SMOOTH TIME	Type in the time smoothing length in units of the master grid cell size
ENTER SCAN TIME	Type in the time averaging length for the "scan average" in units of the master grid cell size
ENTER CHANNEL	Type in the desired spectral channel number using the terminal
ENTER IF	Type in, on the terminal, the desired IF number
ENTER STOKES FLAG	To type in the 4-character string which will control which correlators (polarizations) are flagged. Note: this will apply only to subsequent flagging commands. It should be changed whenever a different Stokes is displayed.
SWITCH SOURCE FLAG	To switch between having all sources flagged by the current flag commands and having only those sources included in this execution of `TVFLG` flagged. The former is desirable when a time range encompasses all of 2 calibrator scans.
SWITCH ALL-CH FLAG	To reverse the flag all channel status; applies to subsequent flag commands
SWITCH ALL-IF FLAG	To reverse the flag all IFs status; applies to subsequent flag commands

The all-channel flag remains true if the input data set has only one channel and the all-IF flag remains true

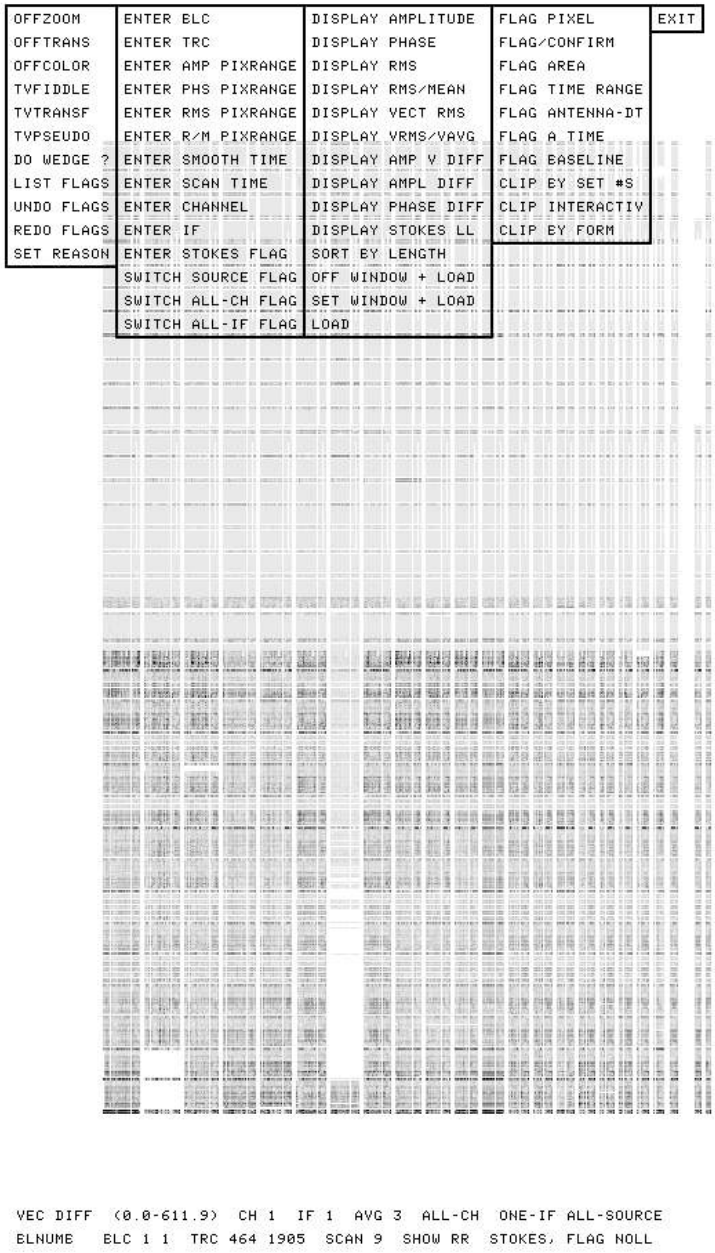

Figure 4.2: A display of a sample TV screen from `TVFLG`, made using the *AIPS* task `TVCPS` to produce a negative black-and-white display. The `TVFLG` menu (in the boxes) and status lines (at the bottom) are displayed in a graphics plane which is normally colored light green. The data are grey scales in a TV memory and may be enhanced in black-and-white or pseudo-colored. The particular display chosen is the amplitude of the vector difference between the sample and a running vector average of samples surrounding it. This particular parameter is sensitive to both phase and amplitude problems and may save you the extra time of looking at phase and amplitude separately. It requires that there be data to average, but does not blur the flagging by the averaging interval (as the RMS method does). The visibility data are from the VLA. All baselines are shown once only in baseline number order. Antenna 22 is missing for all times, while antenna 23 is missing toward the end of the run and and antennas 2 and 7 are missing for some times near the start of the observation. The displayed data are the RR Stokes samples and have been windowed to exclude some times. Flag commands generated at the moment illustrated will flag all source names, all spectral channels, only one IF, and all Stokes except LL.

if the input data set has no more than one IF.

An extra word should be said about the "scan average" to which reference was made above. This is used solely for displaying the difference of the data at time T and the average of the data at times near T. This average is computed with a "rolling buffer." Thus, for a scan average time of 30 seconds and data at 10-second intervals, the average for a set of 7 points is as follows:

time	average of times		
00	00	10	20
10	00	10	20
20	10	20	30
30	20	30	40
40	30	40	50
50	40	50	60
60	40	50	60

The third column of options is used to control which data are displayed and to cause the TV display to be updated. The master grid must be converted from complex to amplitude, phase, the rms of the amplitude, or the rms divided by the mean of the amplitude for display. It may also be converted to the amplitude of the vector difference between the current observation and the "scan average" as defined above or the absolute value of the difference in amplitude with the scalar-average amplitude or the absolute value of the difference in phase with the vector scan average. Furthermore, the baselines may be reordered in the TV display by their length rather than their numerical position. This column has the options:

DISPLAY AMPLITUDE	To display amplitudes on the TV
DISPLAY PHASE	To display phases on the TV
DISPLAY RMS	To display amplitude rms on the TV
DISPLAY RMS/MEAN	To display amplitude rms/mean on the TV
DISPLAY VECT RMS	To display vector amplitude rms on the TV
DISPLAY VRMS/VAVG	To display vector amplitude rms/mean on the TV
DISPLAY AMP V DIFF	To display the amplitude of the difference between the data and a running (vector) "scan average"
DISPLAY AMPL DIFF	To display the abs(difference) of the amplitude of the data and a running scalar average of the amplitudes in the "scan"
DISPLAY PHASE DIFF	To display the abs(difference) of the phase of the data and the phase of a running (vector) "scan average"
DISPLAY STOKES *xx*	To switch to Stokes type *xx* (where *xx* can be RR, LL, RL, LR, etc as chosen by the `STOKES` adverb).
SORT BY *xxxxxxxx*	To switch to a display with the x axis (baseline) sorted by ordered by `LENGTH` or by `BASELINE` number
OFF WINDOW + LOAD	Reset the window to the full image and reload the TV
SET WINDOW + LOAD	Interactive window setting (like `TVWINDOW`) followed by reloading the TV
LOAD	Reload TV with the current parameters

`SET WINDOW + LOAD` is "smarter" than `TVWINDOW` and will not let you set a window larger than the basic image. Therefore, if you wish to include all pixels on some axis, move the TV cursor outside the image in that direction. The selected window will be shown.

The fourth column is used to select the type of flagging to be done. During flagging, a TV graphics plane is used to display the current pixel much like `CURVALUE` in `AIPS`. Buttons A and B do the flagging (except A switches corners for the area and time-range modes). Button C also does the flagging, but the program then returns to the main menu rather than prompting for more flagging selections. Button D exits back to the menu without doing any additional flagging. Another graphics plane is used to show the current

area/time/baseline being flagged. All flagging commands can create zero, one, two, or more entries in the flagging list; hit button D at any time. There are also two clipping modes, an interactive one and one in which the user enters the clip limits from the terminal. In both, the current image computed for the TV (with user-set windows and data type, but not any other windows or alternate pixels etc. required to fit the image on the TV) is examined for pixels which fall outside the allowed intensity range. Flagging commands are prepared and the master file blanked for all such pixels. In the interactive mode, buttons A and B switch between setting the lower and upper clip limits, button C causes the clipping to occur followed by a return to the main menu, and button D exits to the menu with no flagging. The options are

FLAG PIXEL	To flag single pixels
FLAG/CONFIRM	To flag single pixels, but request a yes or no on the terminal before proceeding
FLAG AREA	To flag a rectangular area in baseline-time
FLAG TIME RANGE	To flag all baselines for a range of times
FLAG ANTENNA-DT	To flag all baselines to a specific antenna for a range of times
FLAG TIME	To flag all baselines for a specific time
FLAG BASELINE	To flag all times for a specific baseline
CLIP BY SET #S	To enter from the terminal a clipping range for the current mode and then clip high and low samples
CLIP INTERACTIV	To enter with the cursor and LUTs a clipping range for the current mode and then clip data outside the range.
CLIP BY FORM	To clip selected channels/IFs using the "method" and clipping range of some previous clip operation

The last operation allows you to apply a clipping method already used on one channel/IF to other channels and/or IFs. `CLIP BY FORM` asks for a command number (use `LIST FLAGS` to find it) and applies its display type (amp, phase, rms, rms/mean, differences), averaging and scan intervals and clip levels to a range of channels, IFs and Stokes (as entered from the terminal). To terminate the operation, doing nothing, enter a letter instead of one of the requested channel or IF numbers. To omit a Stokes, reply, if requested for a flag pattern, with a blank line. You may watch the operation being carried out on the TV as it proceeds.

The right-most column has only the option:

EXIT	Go resume `AIPS` and, optionally, enter the flags in the data

Before the flags are entered in the data, `TVFLG` asks you whether or not you actually wish to do this. You must respond yes or no. Note that, if the master grid is to remain cataloged, there is no need to enter the flagging commands every time you decide to exit the program for a while. In fact, if you do not enter the commands, you can still undo them later, giving you a reason not to enter them in the main *uv* data set too hastily.

The two most useful data modes for editing are probably amplitude and amplitude of the vector difference. The former is useful for spotting bad data over longer time intervals, such as whole scans. The latter is excellent for detecting short excursions from the norm. For editing uncalibrated data, rms of two time intervals is useful, but the rms modes require data to be averaged (inside `TVFLG`) and therefore reduce the time resolution accuracy of the flagging. If you edit by phase, consider using the pseudo-coloration scheme that is circular in color (option `TVPSEUDO` followed by button B) since your phases are also circular.

Using `TVFLG` on a workstation requires you to plan the real estate of your screen. We suggest that you place your message server window and your input window side-by-side at the bottom of the screen. Then put the TV window above them, occupying the upper 70–90% of the screen area. (Use your window manager's tools to move and stretch the TV window to fill this area.) Instructions and informative, warning and error messages will appear in the message server window. Prompts for data entry (and your data entry) appear in the input window. Remember to move the workstation cursor into the input window to enter data (such as IF, channel, antenna numbers, and the like) and then to move the cursor back into the TV area to select

options, mark regions to be flagged, adjust enhancements, and so on.

4.4.4 Baseline corrections

Sometimes, *e.g.*, during a VLA array re-configuration, your observations may have been made when one or more of the antennas had their positions poorly determined. The positional error is usually less than a centimeter at the VLA, but even this may affect your data significantly. The most important effect is a slow and erroneous phase wind which is a function of source position and time. Since this error is a function of source position, it cannot be removed exactly using observations of a nearby calibrator, although the error will be small if the target source is close to the calibrator. In many observations, the target sources and calibrators are sufficiently close to allow this phase error to be ignored. Self-calibration will remove this error completely *if* you have enough signal-to-noise to determine the correction during each integration.

The maximum phase error introduced into the calibrated visibility data by incorrect antenna coordinates $\Delta\phi_B$, in radians, by a baseline error of ΔB meters is given by

$$\Delta\phi_B \approx 2\pi\Delta\theta\Delta B/\lambda$$

where $\Delta\theta$ is the angular separation between the calibrator and the target source in radians and λ is the wavelength in meters.

Note, however, that the error due to the phase-wind is not the only error introduced by incorrect antenna positions. A further, but much smaller effect, will be incorrect gridding of the data due to the erroneous calculation of the baseline spatial frequency components u, v and w. This effect is important only for full primary beam observations in which the antenna position error is of the order of a meter. It is highly unlikely that such a condition will occur. Note too, that this error *cannot* be corrected by the use of self-calibration. However, after correcting the antenna position with CLCOR, you may run UVFIX to compute corrected values of u, v, and w. The maximum phase error in degrees, $\Delta\phi_G$, caused by incorrect gridding of the u,v,w data is

$$\Delta\phi_G \approx 360\Delta\epsilon\Delta\Theta$$

where $\Delta\epsilon$ is the antenna position error in antenna diameters and $\Delta\Theta$ is the angular offset in primary beams.

If baseline errors are significant they need to be removed from your data before calibration. For the VLA, go to web page

`http://www.vla.nrao.edu/astro/archive/baselines/`

for information on baseline corrections determined by the analysts that may affect your data. Do not expect to be informed of baseline corrections in any other way. The web page tells you how to determine the corrections that you should apply to your data. Note that NRAO's data analysts use the *AIPS* task LOCIT and procedure BASFIT to determine the antenna position corrections. These are available to the general user, but a data set designed to determine antenna corrections is normally required. Such data sets consist of about 100 observations of a wide range of phase calibrators taken as rapidly as possible.

To apply the antenna position corrections, first copy the first version of the CL table to version 2 with task TACOP.

> TASK 'TACOP' ↵	
> INDISK m ; GETN n ↵	to select the data set, $n = 3$ and $m = 1$ above.
> CLRONAME ↵	to copy to the input file by default.
> INEXT 'CL' ; INVERS 1 ; OUTVER 2 ↵	to copy CL table version 1 to version 2.
> NCOUNT 1 ↵	copy only one version.
> KEYWO ' ' ; KEYVAL 0 ; KEYSTR ' ' ↵	these adverbs not required here.
> GO	to run task TACOP.

This will create a higher version of the CL table 1 and append it to your multi-source data set. Next, use CLCOR to enter the antenna position corrections (in meters) in the new version of the CL table. This must be done for each affected antenna in turn.

> TASK 'CLCOR' ↵	
> INDISK m ; GETN n ↵	to get the correct data set. Note that you don't have to keep doing this unless you switch between different input data files.
> SOURCES ' ' ;STOKES ' ' ↵	to do all sources, all Stokes,
> BIF 0 ; EIF 0 ↵	and all IFs.
> SUBARRAY x ↵	to choose the correct sub-array.
> OPCODE 'ANTP' ↵	to select the antenna position correction mode.
> GAINVER 2 ↵	to choose the correct version of the CL table to read.
> GAINUSE 2 ↵	to force CLCOR to use the same table as output.
> ANTENNA k ↵	to select antenna.
> CLCORPRM $\Delta b_x, \Delta b_y, \Delta b_z, 0, 0, 0, 1$ ↵	to add the appropriate antenna corrections in meters; the 1 in CLCORPRM(7) indicates VLA phase conventions rather than VLB conventions.
> GO ↵	to run CLCOR.

The program will need to be run as many times as there are antennas for which positional corrections must be made. Note that subsequent calibration must be applied to CL table 2 to create higher versions of the calibration table. This new CL table (version 2) will replace version 1 in all of the subsequent sections on calibration. Thus, in subsequent executions of CALIB, you must apply these corrections by specifying DOCALIB TRUE ; GAINUSE 2 (or higher). Note too that CLCOR changes the antenna file for the changed antenna location(s). It might be best to use TASAV to make a pristine copy of your tables before starting the baseline correction step.

4.5 Antenna-based complex gain solutions

At this point, we assume that you have removed the worst of the bad calibrator data (if any) and have run CALIB over as large a UVRANGE as possible for each calibrator. The resulting gain tables can be brought to a consistent amplitude scale, bootstrapping the unknown fluxes of the secondary calibrators. Final pass(es) of CALIB are done if needed and then the solution tables are merged into a full calibration (CL) table.

4.5.1 Bootstrapping secondary flux-density calibrators

Task GETJY can be used to determine the flux density of the secondary flux calibrators from the primary flux calibrator based on the flux densities set in the SU table and the antenna gain solutions in the SN tables. The SU and SN tables will be updated by GETJY to reflect the calculated values of the secondary calibrators' flux densities. This procedure should also work if (incorrect) values of the secondary calibrators' flux densities were present in the SU table when CALIB was run. Bad or redundant SN tables should be deleted using EXTDEST before running GETJY, or avoided by selecting tables one at a time with adverb SNVER.

To use GETJY:

> TASK 'GETJY' ; INP ↵	
> SOURCES '*cal1*' , '*cal2*' , '*cal3*' ... ↵	to select secondary flux calibrators.
> CALSOU '3C286' , ' ' ↵	to specify primary flux calibrator(s).
> CALCODE ' ' ↵	to use all calibrator codes.

> BIF 1 ; EIF 2 ↵	to do both IFs.
> FREQID 1 ↵	to use FQ number 1.
> ANTENNAS 0 ↵	to include solutions for all antennas.
> TIMERANG 0 ↵	to include all times.
> SNVER 0 ↵	to use all SN tables.
> INP ↵	to review inputs.
> GO ↵	to run the task when the inputs are okay.

GETJY will give a list of the derived flux densities and estimates of their uncertainties. If any of the uncertainties are large, then reexamine the SN tables as described above and re-run CALIB and/or GETJY as necessary. Multiple executions of GETJY will not cause problems as previous solutions for the unknown flux densities are simply overwritten.

4.5.2 Full calibration

Once you have determined the flux densities of all your gain calibrators, you are ready to complete the first pass of the calibration. At this point, many observers take a conservative viewpoint and delete their existing SN table(s) with

> INEXT 'SN' ↵	to specify the SN table.
> INVERS -1 ↵	to delete all versions.
> EXTDEST ↵	to do the deletion.

This step forces you to re-run CALIB for all your gain calibration sources and is not required if the previous bootstrapping calibrations included all antennas and most correlators, for these calibrators.

Procedure VLACALIB may be used for your gain calibration sources as you did previously.

> INDI n ; GETN m ↵	to select the data set, $n = 3$ and $m = 1$ above.
> CALSOUR = '*aaaa*' , '*xxxx*' ↵	to name two calibration sources using the same UVRANGE.
> UVRANGE *uvmin uvmax* ↵	*uv* limits, if any, in kiloλ.
> ANTENNAS *list of antennas* ↵	antennas to use for the solutions, see discussion above.
> REFANT n ↵	reference antenna number.
> MINAMPER 10 ↵	display warning if baseline disagrees in amplitude by more than *10%* from the model.
> MINPHSER 10 ↵	display warning if baseline disagrees by more than 10° of phase from the model.
> DOPRINT -1 ↵	to dispense with all the print out this time.
> FREQID 1 ↵	use FQ number 1.
> INP VLACALIB ↵	to review inputs.
> VLACALIB ↵	to make the solution and print results.

If there are different *uv* ranges for different sources, then re-run the procedure with changed parameters, such as:

> CALSOUR = '*cal1*' , '*cal2*' , '*cal3*' ↵	to name secondary flux calibrator(s).
> ANTENNAS 0 ↵	solutions for all antennas.
> UVRANGE 0 ↵	no *uv* limits, or range if any, in kiloλ.
> INP VLACALIB ↵	to review inputs.
> VLACALIB ↵	to process the secondary calibrators.

At this time, you should use as many antennas and as large a UVRANGE as you can for each calibrator, consistent with its spatial structure.

4.5.3 Final (?) initial global calibration

At this point you should have gain and phase solutions for the times of all calibration scans, including the correct flux densities for the secondary calibrators. The next step is to interpolate the solutions derived from the calibrators into the CL table for all the sources. CLCAL may be run multiple times if subsets of the sources are to be calibrated by corresponding subsets of the calibrators, unless you limit it to one or more tables with SNVER and INVERS, CLCAL assumes that all SN tables contain only valid solutions and concatenates all of the SN tables with the highest numbered one. Therefore, any bad SN tables should be removed before using CLCAL. For polarization calibration, it is essential that you calibrate the primary flux calibrator (3C48 or 3C286) also so that you can solve for the left minus right phase offsets and apply PCAL.

To use CLCAL:

> TASK CLCAL ; INP ↵	to review the inputs.
> SOURCES *'sou1'* , *'sou2'* , *'sou3'* , ... ↵	sources to calibrate, ' ' means all.
> CALSOUR *'cal1'* , *'cal2'* , *'cal3'* , ... ↵	calibrators to use for SOURCES.
> FREQID *n* ↵	use FQ number *n*.
> OPCODE 'CALI' ↵	to combine SN tables into a CL table.
> GAINVER 1 ↵	to select the input CL table; 1 for first calibration, 2 if there are baseline corrections.
> GAINUSE 2 ↵	to select the output CL table; 2 is normal, 3 if there are baseline corrections.
> REFANT *m* ↵	to select the reference antenna; needed only if REFANT reset since CALIB was run.
> INTERP '2PT' ↵	to use linear interpolation of the possibly smoothed calibrations..
> SAMPTYPE ' ' ↵	to do no time-smoothing before the interpolation.
> SAMPTYPE 'BOX' ↵	to use boxcar smoothing, followed by interpolation.
> BPARM *n* , *n* ↵	to smooth, if BOX selected, with an *n*-hr long boxcar in amplitude and phase.
> DOBLANK 1 ↵	to replace failed solutions with smoothed ones but to use all previously good solutions without smoothing.
> INP ↵	to check inputs.
> GO ↵	to run CLCAL.

Calibrator sources may also be selected with the QUAL and CALCODE adverbs; QUAL also applies to the sources to be calibrated. Note that REFANT appears in the inputs because *AIPS* references all phases to those of the reference antenna. If none is given, it defaults to the one used in the most solutions.

Users should note that the inputs to CLCAL have changed for the 31DEC03 release. The smoothing and interpolation functions have been separated into two adverbs and the smoothing parameters are now conveyed with BPARM and ICUT. In smoothing, the DOBLANK adverb is particularly important; it controls whether good solutions are replaced with smoothed ones and whether previously failed solutions are replaced with smoothed ones. One can select either or both.

Note that CLCAL uses both the GAINUSE and GAINVER adverbs. This is to specify the input and output CL table versions, which should be different. CL table version 1 is intended to be a "virgin" table, free of all injury from any calibration you do using the *AIPS* package. It may not always be devoid of information, as "on-line" corrections may be made and recorded here by some telescope systems, *e.g.*, the VLBA. The VLA, through tasks FILLM or INDXR, now can put opacity and antenna gain information in this file. CLCAL and most other *AIPS* tasks are forbidden to over-write version 1 of the CL table. This protects it from modification, and keeps it around so that you may *reset* your calibration to the raw state by using EXTDEST

to destroy all CL table extensions with versions higher than 1. Be careful doing this, since you rarely want to delete CL version 1. (All past versions of AIPS have allowed you to do this, but, beginning with the 15JUL94 release, AIPS will ask for special confirmation before allowing you to delete CL version 1.) Should you destroy CL table version 1 accidentally, you may generate a *new* CL table version 1 with the task INDXR. This new CL table may contain the calibration generated from the weather and antenna gain files. If you have made baseline corrections — or any of the many other sorts of corrections allowed by CLCOR — then you will probably want to protect (and use for input) CL table version 2 as well. In that case, GAINVER = 3 is recommended.

If you have any reason to suspect that the calibration has gone wrong — or if you are calibrating data for the first time — you should examine the contents of the output CL table. LISTR with OPTYPE = 'GAIN' will print out the amplitudes and phases in the specified CL or SN table. Note that these tables can be vary large. Use the SOURCES and TIMERANG adverbs to limit the output, or look at it on your terminal (DOCRT = 1) so that you can stop the display whenever you have had enough. Task SNPLT will provide you with a graphical display which may be easier on the eye.

The most important step in the calibration is your verification that everything has gone according to plan. To check this, you should produce matrix listings for all your calibrator sources. For simplicity in interpretation, limit each listing to the UVRANGE to which you limited the calibrator during calibration. Thus:

> TASK 'LISTR' ↵	
> DOCRT -1 ↵	to direct output to the printer.
> SOURCES '*cal1*' , '*cal2*' , '*cal3*' , ... ↵	to list all selected calibrators by name.
> UVRANGE *uvmin uvmax* ↵	*uv* limits, if any, in kiloλ.
> OPTYP 'MATX' ↵	to get the matrix form of listing.
> DOCALIB TRUE ↵	to list with calibration applied.
> GAINUSE 2 ↵	or 3, to point to the *new* gain table.
> FREQID *n* ↵	list data for FQ *n*.
> DPARM = 5 , 1 , 0 ↵	to have amplitude and phase using scalar scan averaging.
> BIF 1 ↵	to specify the "AC" IFs; only one can be done at a time.
> INP ↵	to review the inputs.
> GO ↵	to run the program when inputs set correctly.
> BIF 2 ↵	to specify the "BD" IFs.
> WAIT ; GO ↵	to run the program again after the first job is done.

The matrix average amplitudes for the calibrators in this listing should be very close to the values that you entered with SETJY (or which were derived by GETJY) and the phases in all rows and columns for these sources should be very close to zero.

If some rows and columns of the amplitude matrices are systematically different from the mean, the amplitude calibration for the associated antennas is imperfect. The reasons for this should be investigated. More flagging of visibilities, scans, or antennas, may be indicated. If the phase matrices have all elements near zero, then the phase calibration is in good shape. If some calibrators have discrepant phases and others do not, the discrepant calibrators are probably resolved. Note that you will not be able to detect errors in the assumed positions of your calibrators at this stage if you have used the usual 2-point interpolation of the calibration. Position errors in the calibrators have now become phase and position errors in the target sources.

If the previous steps indicate serious problems and/or you are seriously confused about what you have done and you want to start the calibration again, you can use the procedure VLARESET from the RUN file VLAPROCS to reset the SN and CL tables.

> INP VLARESET ↵	to verify the data set to be reset.
> VLARESET ↵	to reset SN and CL tables.

A SUMMARY OF *AIPS* CONTINUUM UV-DATA CALIBRATION

From VLA Archive Tape to a UV FITS Tape
AIPS Memo No. 76 Updated
Glen Langston

A.1 Basic calibration

The Gentle User enters the Computer room with a VLA archive tape containing a scientific breakthrough. The user's sources are named SOURCE1 and SOURCE2. The interferometer phase is calibrated by observations of CAL1 and CAL2. The flux density scale is calibrated by observing 3C48 (=0137+331) and polarization is calibrated with observations of 3C286 and/or 3C138. Mount the tape on drive number n, log in and start *AIPS*. Example input: AIPS NEW. Mount the tape: INTAPE=n; DENS=6250; MOUNT.

PRTTP Find out what is on the tape, get project number and bands. TASK='PRTTP'; PRTLEV=-2; NFILES=0; INP; GO; WAIT; REWIND.

FILLM Load your data from tape. Select only one band at a time to process. TASK='FILLM'; VLAOBS='?'; BAND=' '; NFILES=?; DOWEIGHT 1; INP; GO (Replace all ?'s with appropriate values.) FILLM will load your visibilities (*uv*-data) with weights suitable for calibration into a large file for each band. It also creates 6 *AIPS* tables each; these tables have two letter names which are:

- HI Human readable history of things done to your data. Use PRTHI to read it.
- AN Antenna location and polarization tables. Antenna polarization calibration is placed here.
- NX Index into visibility file based source name and observation time. Not modified by calibration.
- SU Source table contains the list of sources observed and indexes into the frequency table. The flux densities of the calibration sources are entered into this table.
- FQ Frequencies of observation and bandwidth with index into visibility data. Not modified.
- CL Calibration table describing the antenna based gains. Version 1 should never be modified. The CL table contains entries at regular time intervals (*i.e.*, 2 minutes) for each antenna. **The ultimate goal of calibration is to create a good** CL **version 2.** Use PRTAB to read tables.

PRTAN Print out the antenna locations. TASK='PRTAN'; PRTLEV=0; INP; GO. Choose a good *Reference* antenna (called *R*) near the center of the array (REFANT=R). Check the VLA operator log to make sure the antenna was OK during the entire observation.

QUACK Flag the bad points at the beginning of each scan, even the ones with good amplitudes could have bad phases. Creates a Flag Table (FG). You want to use FG table version 1 for all tasks. TASK='QUACK'; FLAGV=1; OPCOD=' '; APARM=0; SOUR=' '; INP; GO deletes the first six seconds of each scan, which may not be enough.

- FG A flag table marks bad data. FG tables contain an index into the UV data based on time range, antenna number, frequency and IF number.

LISTR Lists your UV data in a variety of ways. Make a list of your observations. TASK='LISTR'; OPTYP='SCAN'; DOCRT=-1; SOUR=' '; CALC='*'; TIMER=0; INP; GO. NOTE: IF you

have observed in a such a way as to create more than one FREQID, you must run through the entire calibration once for EACH FREQID. For new users, it is better to use UVCOP to copy each FREQID into separate files and calibrate each file separately. This is required if you are doing polarization calibration.

UVCOP Skip this step if your data consists of only one FREQID. Copy different FREQIDs into separate files. TASK='UVCOP'; FREQID=?; CLRON; OUTDI=INDI; INP; GO. The result will be a ??.UVCOP file.

SETJY Sets the flux of your flux calibration source in the SU table. TASK='SETJY'; SOUR='3C48',' '; OPTYP='CALC'; FREQID=1; INP; GO. Adjust flux density for partial resolution following the rules in the VLA Calibration Source Manual or the *AIPS CookBook*.

TASAV As insurance, make a copy of all your tables. TASK='TASAV'; CLRON; OUTDI=INDI; INP; GO.

CALIB CALIB is the heart of the *AIPS* calibration package. RUN VLAPROCS, an *AIPS runfile*, to create procedures VLACALIB, VLACLCAL and VLARESET. The procedure VLACALIB runs CALIB. Set the UV and Antenna limits for 3C48. For L, C and X band 5% and 5 degree errors are OK; for other bands the limits are higher. CALIB places antenna amplitude and phase corrections into an SN table for the time of observation of phase calibration sources.

SN Solution table contains antenna based amplitude and phase corrections for the time of observations of the calibration sources. These SN table results are latter interpolated for all times of observation and placed in a CL table. Only the CL table corrections will be applied to the program sources.

TASK='VLACAL'; CALS='3C48',' '; CALCODE='*'; REFANT=*R*; UVRA=?; SNVER=1; DOCALIB=-1; DOPRINT=1; MINAMP=10; MINPH=10; INP; VLACAL. The task CALIB lists antenna pairs which deviate significantly from the solution. If you have lots of errors, then carefully examine your data using TVFLG or LISTR. (See *AIPS CookBook* for a lengthy discussion on flagging.) If one antenna is bad over a limited time range, use UVFLG to flag that antenna for the time from just after the previous good CAL observation to before the next good CAL observation.

UVFLG Flag bad UV-data. TASK='UVFLG'; ANTEN=?,0; BASELI=?,0; TIMER=?; FLAGV=1; SOUR=' '; OPCOD=' '; INP; GO. If in doubt about any data, FLAG THEM! If you have flagged the primary calibrator, return to CALIB above and try again.

CALIB Now calibrate the antenna gain based on the rest of the cal sources. Look in the Calibrator manual for UV limits; if there are limits, VLACAL must be run separately for these sources. TGET VLACAL; CALS='CAL1','CAL2',' '; ANTEN=0; BASELI=0; UVRANGE=?,?; INP; VLACAL. Flag bad antennas listed. Each execution of CALIB replaces previous corrections in the SN table or appends new corrections. If unsatisfied with a VLACAL execution, all effects of it are removed by running VLACAL again for the same sources (but different ADVERBS or after flagging bad data).

GETJY Sets the flux of phase calibration sources in the SU table. TASK 'GETJY'; SOUR='CAL1,'CAL2',' '; CALS='3C48',' '; BIF=0; EIF=0; INP; GO. GETJY over-writes existing SU table entries, and is not affected by previous executions.

TASAV Good time to save your tables. TGET TASAV; INP; GO.

CLCAL Read the antenna amplitude and phase corrections from the SN table and interpolate the corrections into a new CL table. CLCAL applies calibration source corrections to the program sources. Each execution of CLCAL adds to output CL table version 2. CLCAL is run using the procedure VLACLCAL. TASK='VLACLC'; SOUR='SOURCE1','CAL1',' '; CALS='CAL1',' '; OPCODE='CALI'; TIMER=0; INTERP='2PT'; INP; VLACLC. Run CLCAL for the

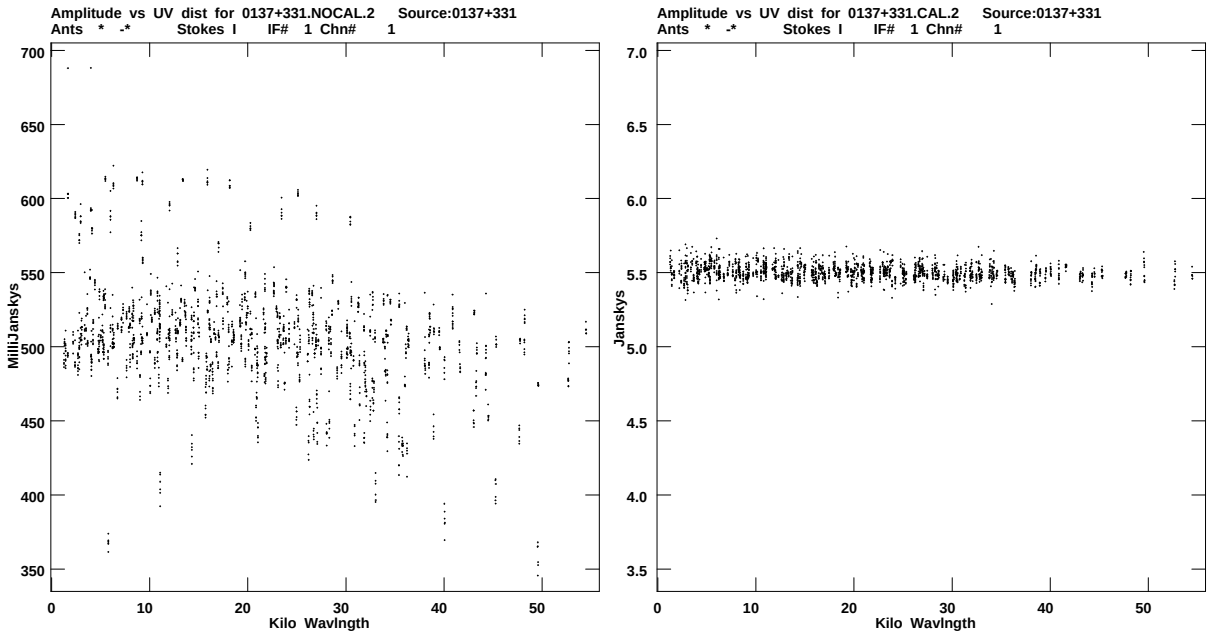

Figure A.1: *(left)* Un-calibrated *uv*-data and *(right)* calibrated *uv*-data from a C-band snapshot of 3C48. Default VLA gains are a tenth of the actual gains and can show significant scatter. Only wild *uv* points ∼50% greater than the average can be detected before calibration.

second source using the second calibrator. TGET VLACLC; SOUR='SOURCE2','CAL2',' '; CALS='CAL2',' '; INP; VLACLC. Move the SN table corrections for 3C48 into the CL table. TGET VLACLC; SOUR='3C48',' ';CALS='3C48',' '; INP; VLACLC. (3C48 could also be calibrated with CAL1 or CAL2.)

LISTR Make a matrix listing of the Amplitude and RMS of calibration sources with calibration applied. Look for wild points. TASK='LISTR'; OPTYP='MATX'; SOUR='CAL1','CAL2',' '; DOCAL=2; DOCRT=-1; DPARM=3,1,0; UVRA=0; ANTEN=0; BASELI=0; BIF=1; INP; GO. If only a few points are bad, flag them and continue. If too many are bad, delete CL table 2 and the SN tables using VLARESET. Then return to the first CALIB step. If the data look good, run LISTR again for IF two. TGET LISTR; BIF=2; INP; GO

UVPLT Plot the *uv*-data in a variety of ways. Make a Flux versus Time plot first. Choose XINC so the plot will have no more than 1000 points. TASK='UVPLT'; SOUR='SOURCE1',' '; XINC=10; BPARM(1)=11; DOCAL=2; BIF=1; INP; GO. Look at the plot with LWPLA, TKPL, TVPL or TXPL. Plot other IF . Flag wild points. Plot Flux versus baseline. TGET UVPLT; BPARM=0; INP; GO.

Calibration is now complete for continuum, un-polarized observations. Write the calibrated data to tape with FITTP if you don't want to calibrate the polarization. To create images from the *uv*-data use SPLIT to calibrate the multi-source data and create a single source *uv*-data set. (FITTP and SPLIT are described at the end of the polarization calibration process.)

A.2 Polarization calibration

For polarization observations, the following steps are required. For 21cm or longer wavelength observations, ionospheric Faraday rotation corrections may be needed. See FARAD in the *AIPS CookBook*, but don't expect much help anymore.

TASAV As added insurance, save your tables again. TGET TASAV; INP; GO.

LISTR Print the parallactic angles of the calibration sources. TGET LISTR; SOUR=' '; CALC='*'; OPTYP='GAIN'; DPARM=9,0; INP; GO

PCAL Intrinsic antenna polarization calculation. PCAL will be successful only if cal. sources are observed at several parallactic angles. PCAL will modify the AN and SU tables. TASK='PCAL'; CALS='CAL1','CAL2',' '; BIF=1; EIF=2; DOCAL=2; REFANT=*R*; INP; GO

RLDIF Now determine the absolute linear polarization angle. Make a matrix listing of the angle of 3C286. TASK 'RLDIF'; SOUR='3C286',' '; DOCAL=2; BIF=1; EIF=0; DOPOL=1; GAINUSE=2; DOCRT=-1; INP; GO. The observed angles are different for each frequency and IF . This task returns the average angles for all IFs in CLCORPRM.

CLCOR Now apply the angle corrections to CL table 2. The relative phase of Left and Right circular polarization produces the linear polarization angle and the phase correction is applied to L. The phase difference (twice the angle of linear polarization) for 3C286 is 66^o and for 3C138, $\phi = -18^o$ at L band, perhaps -24^o at higher frequencies. Change RLDIF's results to this form with FOR I = 1:20; CLCORP(I)=66-CLCORP(I); END. Then TASK='CLCOR'; STOKES='L'; SOUR=' '; OPCOD='POLR'; BIF=1; EIF=2; GAINVER=2; GAINUSE=0; INP; GO. Run RLDIF again to check the phases. TGET RLDIF; INP; GO. Note that CLCOR copies the CL table version 2 to 3 while applying the phase correction. If the phases are wrong, delete version 3, return to PCAL and RLDIF and then do another CLCOR.

A.3 Backup and imaging

FITTP Writes the output *uv*-data to tape. DISMOUNT your archive; MOUNT your output tape. TASK='FITTP'; DOEOT=1; OUTTAP=INTAP; INP; GO. Use DOEOT=-1 when at the beginning of a new tape.

SPLIT The *AIPS* calibration process only modifies the tables associated with the multi-source *uv*-data set. SPLIT selects individual sources, reads the CL table and multiplies the visibilities by the corrections to produce a calibrated single-source *uv*-data set. TASK='SPLIT'; SOUR=' '; CALC=' '; UVRA=0; TIMER=0; DOCAL=2; FLAGVER=1; GAINUSE=3; DOPOL=1; DOBAN=-1; BIF=0; EIF=0; STOKES=' '; BLVER=-1; APARM=0; DOUVCOM=1; ICHANSEL=0; INP; GO

Mapping Use your favorite Fourier Transform task (*e.g.*, IMAGR, HORUS, MX, WFCLN or UVMAP) to produce images from the calibrated data. A set of *AIPS* procedures (called MAPIT) has been developed to automatically Fourier Transform, deconvolve and self-calibrate the *uv*-data. See *AIPS* Memo No. 72. Procedure MAPPR provides a simplified interface to IMAGR.