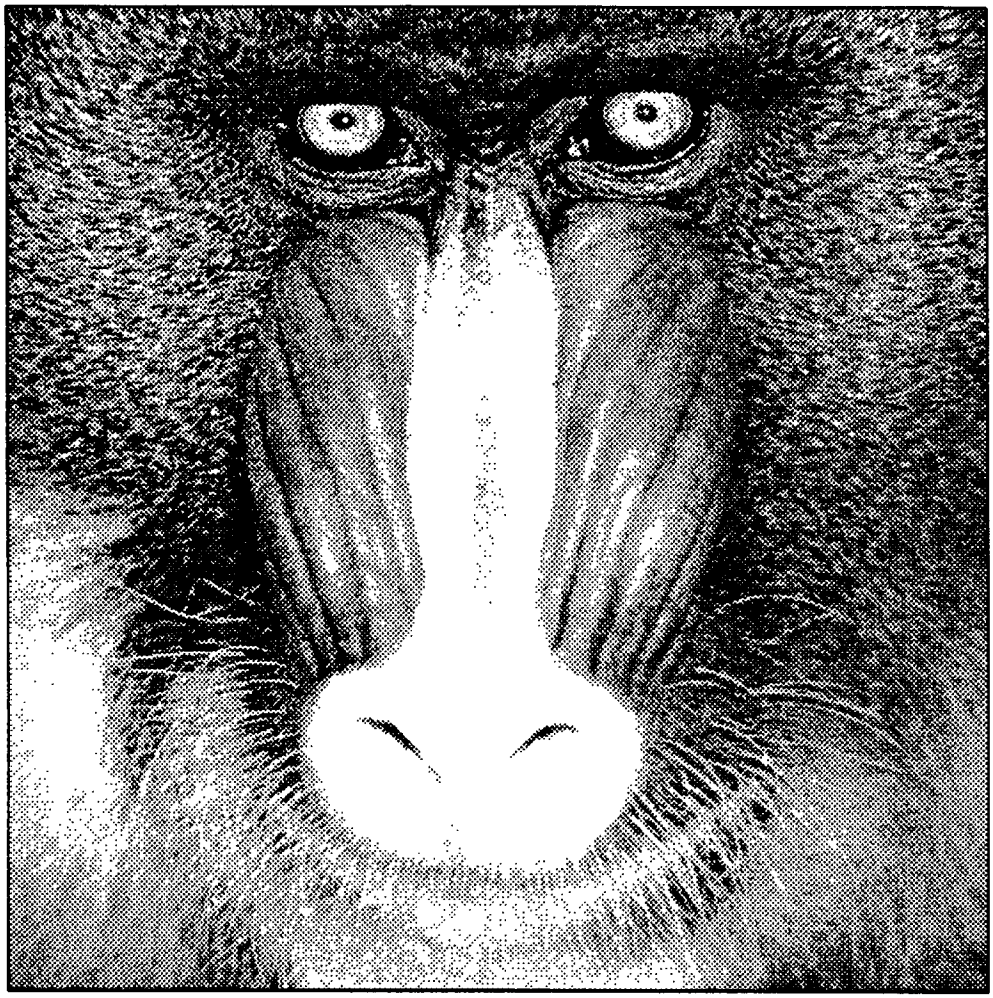
AIPS COOKBOOK

15-Oct-1985



The National Radio Astronomy Observatory
Edgemont Road
Charlottesville, VA 22903–2475
Operated by Associated Universities, Inc.
under contract with the National Science Foundation

ACKNOWLEDGEMENTS

The *COOKBOOK* cover design is by Pat Smiley of the NRAO Graphic Arts Department. It is based on a design suggested by John Bally of Bell Labs.

The image on the title page was plotted on a QMS Lasergrafix 800 by the *AIPS* tasks GREYS and QMSPL. It is the red portion of the digitized image of a Mandrill which has become a standard in the image-processing field.

The plots at the ends of Sections 7, 8 and 9 were also generated on a QMS Lasergrafix 800 using various *AIPS* plotting tasks and QMSPL. The data displayed in § 7 were provided by S. Baum, A. Bridle, T. Heckman, G. Miley and W. van Breugel. The data displayed in § 8 were provided by A. Bridle and R. Perley and those displayed in § 9 by D. Wells. The editors thank these people for providing their data prior to their publication.

TABLE OF CONTENTS

1. INTRODUCTION

This *COOKBOOK* is intended to help beginning users of the NRAO *A*stronomical *I*mage *P*rocessing *S*ystem (*AIPS*) by providing a recipe approach to the most basic *AIPS* operations. While it illustrates some aspects of *AIPS*, it does not pretend to be complete. However, it does include detailed instructions for running many important items of *AIPS* software. With these as a model, the user should be able to run other *AIPS* software aided by the EXPLAIN, HELP and INPUTS files and the complete index of software given in § 14 of the *COOKBOOK*. In this edition, we have expanded some sections in order to provide an overview of a few less basic, but nonetheless interesting, programs which are available and which often seem to be forgotten even by experienced *AIPS* users.

AIPS software is continually changing and growing. (This edition describes the 15 October 1985 release of *AIPS*, frequently referred to as 15OCT85.) Users are encouraged to recommend new and better analysis and display tools and to help debug the existing software by entering "Gripes" (see § 11.3). Please note that examples of bugs that are documented by printouts of inputs, message logs, etc. are most useful to the programmers. Also note that written bug reports are *much* more effective than verbal reports.

The original *COOKBOOK* was written in 1981 by Alan Bridle as he first encountered *AIPS*. It was substantially rewritten in 1982 by Alan Bridle and Ed Fomalont, with assistance from Tim Cornwell and Jacqueline van Gorkom. In 1983, it was TEXset and revised by Eric Greisen, with the assistance of Don Wells, Fred Schwab, and Alan Bridle. The present edition was revised and edited by Alan Bridle and Eric Greisen. It has been updated and corrected to reflect the state of *AIPS* as of 15 October 1985. It contains an expanded spectral-line section (§ 9) written by Jacqueline van Gorkom and a new VLBI section (§ 10) written by John Benson. Appendix Z contains instructions and advice peculiar to the individual *AIPS* sites of the NRAO. Please report any inadequacies or errors in the *COOKBOOK* to Alan Bridle in Charlottesville.

This edition contains a glossary of astronomical and computing terms written by Fred Schwab. This glossary is larger than before, but is still incomplete; suggestions for additions to it should be sent to Fred Schwab in Charlottesville.

As a result of typesetting, the *COOKBOOK* does not exist in a form which can be listed intelligibly on ordinary line printers. Copies may be obtained by writing to the NRAO Computer Division in Charlottesville or at the VLA. A faster method is to use your own copying machine. *AIPS* sites which have a QMS Lasergrafix printer, such as Charlottesville and the VLA, may have files on disk from which up-to-date copies of the *COOKBOOK* may be printed. Consult your local *AIPS* Manager for details on how to do this.

Additional written documentation on *AIPS* is available in several forms. A programmers' reference manual called *Going AIPS* is available in two volumes. The first volume is intended for applications programmers, while the second volume is needed by programmers developing *AIPS* for new peripheral devices or computers. Printed copies of *Going AIPS* may be obtained by writing to the NRAO Computer Division in Charlottesville. The printer files are provided in *AIPS* releases and should be available at your installation. In addition, there is a four-volume manual called *AIPS: A Manual*, **NRAO Computer Division Users' Manual Series**, Number 31. The original text files and a printed copy may be available at your installation.

AIPS provides run-time documentation in the form of HELP and EXPLAIN files which may be viewed at the terminal or printed. (See § 4.8 for explicit instructions.) Should these not suffice, consult your local *AIPS* Manager and then, if needed, call the *AIPS* programmers in Charlottesville. Although individual *AIPS* programs have often been written, and are best understood, by a single programmer, the *AIPS* group as a whole assumes responsibility for all released software. Anyone in the group will attempt to help you or, at least, to identify another member of the group better able to help you.

1.1. What *AIPS* can do

AIPS is a system of hardware and software for:

1. Processing *uv* data to make images and to improve their quality.
2. Displaying images in a variety of ways.
3. Analyzing and combining images to obtain useful parameters.

Input to and output from *AIPS* is by magnetic tape. The *uv* data can be read or written in the so-called EXPORT format which can be generated by the DEC-10 at the VLA, the IBM in Charlottesville, or an *AIPS* program. Visibility data can also be read and written in the "groups" extension of the FITS format (often called UV-FITS). Images can be read or written in the FITS format. A special Charlottesville IBM format for images is also supported.

1.2. Organization of the *COOKBOOK*

1.2.1. Contents

Section 2 of the *COOKBOOK* describes in general terms how to get started in *AIPS* — signing up, logging in, mounting tapes, etc. Appendix Z gives details of these operations specific to NRAO's *AIPS* sites. Your local *AIPS* Manager may be able to provide a version of this appendix appropriate to your system. Section 3 leads you through the basics of reading in your *uv* data and making images — the operations which beginning users are most likely to encounter first. Section 4 describes a few useful features of *AIPS* which you will need to use it efficiently. This section is best read after you have acquired some experience with *AIPS* either by following through § 3 or by some other means. Sections 5 through 8 introduce the basic *AIPS* tools for improving images, for making interactive and hard-copy displays of them, and for analyzing them. Section 9 contains hints and further *AIPS* tools of particular interest, but not restricted, to spectral-line users and other observers who have images of more than 2 dimensions. Similarly, § 10 is aimed primarily at users of VLB interferometers. Section 11 deals with exiting from *AIPS* and tidying up your disk areas to be polite to the other users. Section 12 suggests some cures for common hang-ups and miscellaneous "disasters" which seem to afflict *AIPS* users. No such list can be made comprehensive or sufficiently general to cover all the computer systems now running *AIPS*. You will need to consult with your local *AIPS* Manager or other users if you encounter an unlisted problem.

Section 13 is intended for the "mature" *AIPS* user who wishes to learn about data formats, procedures, RUN files, and various subtleties of AIPS syntax. We recommend that you read this after becoming familiar with the operations described in §§ 3 through 8. Section 14 contains lists of all available routines broken down by categories and in a complete, alphabetic list. Appendix G presents Fred Schwab's Glossary of radio astronomy data processing terminology.

1.2.2. Minimum match

In this *COOKBOOK*, we use the minimum-matching capability of *AIPS* to abbreviate the instructions needed to run the programs. This speeds up your activity at the terminal while working in *AIPS*. However, the full names of some of the AIPS instructions may be easier to learn and to remember. They are given in § 14.

1.2.3. Fonts and what they signify

Throughout this *COOKBOOK*, RESPONSES TO BE TYPED BY THE USER APPEAR IN THE PRESENT FONT. Prompts provided by the operating or *AIPS* systems are left-justified on the same line, *e.g.*, system prompt **$**, AIPS prompt **>**. THIS IS THE FONT USED FOR SAMPLE OUTPUTS FROM THE COMPUTER and for program names such as PRTUV. A lower-case italic font, *such as this*, is used for numeric and character parameter values which must be supplied by the user. The symbol AIPS refers to the program which you will use to communicate with the computer. The symbol *AIPS* refers to the full system, made up of the AIPS program, numerous other programs which may be run from AIPS, and the hardware configuration. The symbol C_R means "hit the RETURN key on the terminal".

In previous editions of the *COOKBOOK*, the words "map" and "image" were used more or less interchangeably as they often are in radio astronomy. In this edition, "map" has been replaced by "image" throughout in keeping with the modern trend in astronomical terminology. Note, however, that the term "map" still appears in some *AIPS* task names, disk file types, and other terms that are buried in the code itself. The symbol § means Section and refers to the various chapters and sub-chapters of this *COOKBOOK*.

1.3.1. Banana storage

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

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

1.3.2. Banana coffeelate

1. Peel and mash 2 ripe **bananas**.
2. Blend in 1/2 teaspoon **vanilla extract**, a few grains **salt**, 1/4 cup **chocolate syrup**, 2 teaspoons **sugar**, and 2 teaspoons instant powdered **coffee**.
3. Add $1\frac{1}{2}$ cups **milk**.
4. Beat with rotary beater or electric mixer until smooth and creamy. Chill.

2. STARTING UP *AIPS*

This section contains general information concerning the steps needed to obtain access to, and use, an *AIPS* system. It attempts (as does the design and coding of *AIPS* itself) to avoid specific references to particular computer devices and to the peculiarities of any one *AIPS* installation. However, some installation-specific practicalities remain. For the NRAO installations, these are described in Appendix Z.

2.1. Signing up for *AIPS* time

Most computer systems cannot support more than a few simultaneous users of *AIPS*. Thus, most locations are obliged to institute a mechanism for distributing the available *AIPS* time to the people desiring it. At NRAO, sign up sheets and rules for their use are normally posted in or near the principal "*AIPS* Caige" (user-terminal room). To promote fair and efficient use of the system, there are often restrictions on the amounts of time that any one user or user group may reserve.

AIPS can support several users which it calls AIPS1, AIPS2, etc. The user designated AIPS1 has software priority for the use of the array processor and moral priority for the use of the display devices and tape drives. In many cases, however, the user designated AIPS2 may carry out similar processing tasks by verbal negotiation with the AIPS1 user over the use of these resources. The facilities available through AIPS3 and higher are more limited and most users will require either AIPS1 or AIPS2.

Your installation may have separate sign-up sheets and terminals for AIPS1, AIPS2, etc. with rules reflecting these priorities. Consult your local *AIPS* Manager for details.

2.2. Using the terminals

The way that a terminal behaves is a function both of the type of terminal and of the computer system to which it is attached. It is important for beginning users to master the foibles of the terminal(s) they will be using. For example, on some terminals/systems, CTRL H is used to back over and delete characters which have been typed. Other systems require instead the use of the RUB OUT key or the BACK SPACE key for this purpose. Some systems erase deleted characters while others merely reposition the cursor so that deleted characters may be overwritten. Appendix Z describes these characteristics for the NRAO hardware.

Some computer systems are very friendly to the users. They allow, for example, the "XON / XOFF" protocol by which the user may temporarily suspend the execution of his programs while he examines the current contents of the terminal display and then resume the execution. They also recognize interrupt commands which, usually with little ceremony, will abort the current program. Such friendly systems also normally allow "type-ahead" by which the user may type the next command line(s) while the current line is being executed.

More expensive terminals are now available with more than one screen's worth of memory. Such terminals allow the user to scroll and/or page back to look at earlier output. We recommend such terminals for *AIPS* since its HELP and INPUTS files are sometimes longer than the usual number of lines (24) on a terminal screen.

2.3. Logging in

Logging in may be the most machine-dependent step in using *AIPS*. Most systems require you to signal your intention to use the computer from a particular terminal in order to assign that terminal to you and to begin any accounting processes. Then, normally, you must ask some batch or job control process in the system to run the AIPS program for you. Finally, you must tell the AIPS program who you are. Details for the VLA VAXes, the Charlottesville VAX, and the Charlottesville MODCOMP are given in Appendix Z. On other installations, consult your local *AIPS* Manager before logging in.

As you enter the commands needed to log in to your system, please read all messages which appear. They are often important and relate to current system, disk, and *AIPS* problems which may affect your reductions.

After the system messages, the statement

```
To start up AIPS type AIPS OLD or AIPS NEW or AIPS TST
```

will appear followed by a system prompt ($) on those systems which offer a choice. Type in your selection. The OLD version is likely to be relatively free of bugs (provided the *AIPS* version in NEW does not prescribe format changes which prevent OLD from working), but the NEW version will contain improvements and will be mostly debugged. The TST version is a debugging area recommended for NRAO staff and those few users who may require the most recent software. (Note that this choice affects only the version of the AIPS program itself. You may choose TST, NEW or OLD versions of the *AIPS* reduction programs at a later time — see §4.5.)

You should then see the message:

```
Starting 15OCT85 version of AIPS
BEGIN THE ONE TRUE AIPS NUMBER n
```

where 15OCT85 identifies the release of *AIPS* and *n* is a number between 1 and 6. Where appropriate, check that this "*AIPS* number" is indeed the one for which you signed up. If not, you are probably using the wrong terminal: on many systems AIPS1 and AIPS2 are assigned to specific terminals, labeled as such, while other *AIPS* numbers are available on a first-come-first-served basis. AIPS will now ask you for your user number and provide a ? prompt:

```
AIPS n:  ENTER USER ID NUMBER
? uuu ⏎
```

where *uuu* is the number assigned to you for the local *AIPS* system. The AIPS prompt > should now appear.

2.4. 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, go to the room in which the tape drives are located and find one which is not in use. 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. 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. 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.

2.5. Software tape mount

When you have the tape physically mounted on the tape drive, most computer systems must also be told that you have done this and which tape drive you have chosen. This step is called a "software tape mount." Most systems allow this operation to be done from inside AIPS by typing:

> INTAPE *m* $\subset_R$	to specify the drive labeled *m*.
> DENSITY *dddd* $\subset_R$	to set the density to *dddd* bpi if needed.
> MOUNT $\subset_R$	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. A few systems require the software tape mount to be done before actually beginning the AIPS program. Consult your local *AIPS* Manager for instructions.

2.6.1. Banana daiquiri

1. Combine in an electric blender: 2 oz. light rum, 0.5 oz. banana liqueur, 0.5 oz. lime juice, 1/2 small banana peeled and coarsely chopped, and 1/2 cup crushed ice.
2. Blend at high speed until smooth.
3. Pour into large saucer champagne (or similar) glass. Serves one.

2.6.2. Hot banana soufflé

1. Preheat oven to 375°.
2. Select a 6-cup soufflé dish or other mold and grease it liberally with 1 tablespoon butter.
3. Place 6 eggs, 1/2 cup cream, juice of 1/2 lemon, 1 tablespoon kirsch, and 1/4 cup sugar in blender. Blend until the batter is smooth.
4. Peel 2 large bananas, removing any fibers and break into chunks. With blender running, add the chunks one at a time.
5. Break 11 ounces cream cheese into chunks and add them to the blender.
6. When all the ingredients are thoroughly mixed, run the blender at high speed for a few seconds.
7. Pour batter into prepared dish and place it in the hot oven. Bake 45–50 minutes until the top is lightly browned and puffy. You may quit when the center is still a bit soft or continue baking until the center is firm.
8. Serve at once. A whipped cream flavored with Grand Marnier makes a nice topping.

3. MAKING IMAGES

A complete list of the software dealing with *uv* data can be obtained at your terminal by typing HELP UVPR ↵ or HELP MAPETC ↵. These lists are also given in § 14 of the *COOKBOOK*.

Images are made by *AIPS* from *uv* data in either of two possible formats. The first is that of the VLA EXPORT tapes prepared on the DEC-10 or by the task UVEXP in previous *AIPS* sessions. The second is that of FITS format tapes for *uv* data written by the VLA "Pipeline", the VLA DEC-10, the *AIPS* task FITTP, or other systems.

To make images and store the *uv* data on disk, the following sequence of *AIPS* programs is needed:

1. PRTTP or EXIND to identify and check the data on tape and to find the number of visibility data points to be processed.
2. UVLOD to copy data from tape to disk.
3. UVSRT to sort data into proper order for the FFT.
4. UVMAP or MX to make the image(s) and beam.

Image improvement — CLEANing or other deconvolution, self-calibration, editing, etc. — is discussed in § 5.

3.1. PRTTP and EXIND

Bring your data tape to the *AIPS* processor and follow the tape mounting instructions in §§ 2.4 and 2.5 above. The program PRTTP reads a full tape and prints out a summary of all the *uv* and image data on tapes written in any of the supported formats. To use this program, type:

> TASK 'PRTTP' ; INP ↵	to list the required inputs on the terminal.
> INTAPE *m* ↵	to specify the tape drive number (*m*).
> NFILES 0 ↵	to print information for all the files.
> GO ↵	to run the program.

The tape will be rewound if necessary and will then begin to move forward. All files will be read. A printout will appear on the system printer with a valuable summary of header information for each file on the data tape.

As PRTTP begins, you should see the message PRTTP BEGINS on the *AIPS* "monitor" (a terminal screen near your primary terminal or, in its absence, your own terminal). The AIPS prompt > should have already returned on your terminal, however, since PRTTP is running as a detached "task." Tasks are the more complicated *AIPS* programs, run by the GO command after setting the task name with TASK *'taskname'*. They are shed from the terminal, as PRTTP has been here, allowing you to use AIPS for any other processing except running the same task at the same time.

The program EXIND works on only one file at a time and works only on EXPORT format *uv* data tapes. The following might be used as a set of inputs to EXIND:

> TASK 'EXIND' ; INP ↵	to list the required inputs on the terminal.
> INTAPE *n* ↵	to specify the use of drive number *n* — make this specification compatible with your tape mount!
> NFILE 3 ↵	to skip 3 files before getting data (default: 0).
> GO ↵	to run EXIND with these inputs.

To run EXIND again, this time on the next file, enter:

> NFILE 0 ; GO ↵

once EXIND has finished on the first file. If you have reset the task name in order to run another task while EXIND was in progress, enter:

> NFILE 0 ; GO EXIND ↵

instead. To get hard copy of the EXIND output for later use, follow the instructions in § 4.1 dealing with the message file.

3.2. UVLOD

UVLOD copies the *uv* data from tape to disk. The data on tape can be in either EXPORT or FITS format. The following shows inputs to UVLOD for reading data on 3C138 at C band (6 cm) from the third file on a tape mounted on tape unit number 2:

> TASK 'UVLOD' ; INP ↵	to set the task name and review the required inputs.
> SOURCE '3C138' ↵	to load only 3C138 data.
> BAND 'C' ↵	to load only 6-cm data.
> INTAPE 2 ↵	to specify the tape drive number, which must be same as tape mount number.
> NFILES 2 ↵	to skip to the third file on the tape.
> OUTN '3C138 A C' ↵	to signify that this is A-array data at C band (for example).
> OUTC 'UVDATA' ↵	
> OUTDISK 3 ↵	to specify writing to disk 3, *e.g.*, to select a disk with sufficient free space. (See § 4.6 for help in monitoring free disk space).
> NPOINTS 17 ↵	to specify the number of visibility points in thousands. Get the number you need from the PRTTP or EXIND output.
> REWIND ↵	to rewind tape before skipping files.
> GO ↵	to run UVLOD.

The tape will begin to move and appropriate messages should appear on the *AIPS* monitor. When the prompt > appears on your terminal, you are free to use AIPS for other purposes. Only one source and band will be loaded with each run of UVLOD as set up here. However, more than one source or band may be loaded under control of additional parameters: type HELP UVLOD ↵ for details.

Once UVLOD has finished, check that your disk catalog now contains the *uv* data you have just tried to load by:

> INDI 0 ; UCAT ↵

which will list all *uv* data sets in your disk catalog. This list should look something like:

```
CATALOG ON DISK  3
CAT USID MAPNAME       CLASS   SEQ  PT     LAST ACCESS      STAT
  1   76 3C138 A C    .UVDATA .    1 UV 22-OCT-1985 12:33:34
```

Alternatively, get terminal *and* hard-copy listing of your catalog by:

> INDI 0 ; INTY 'UV' ↵	to list all disks, *uv* files only.
> CATALOG ↵	to put the catalog listing in the message file.
> PRTMSG ↵	to print the message file.

This sequence takes a little longer to execute, but the hard-copy list (sent to the appropriate printer) may be useful if your catalog is a long one.

Note that the catalog has assigned an ordinal number to the data set in the first (CAT) column of the listing. This number and the disk number (3) should be noted for future reference as they are useful when selecting this data set for further processing.

Please deassign the tape drive once you are completely done with tape data. Use:

> INTAPE *n* $\subset_R$	to specify the drive labeled *n*.
> DISMOUNT $\subset_R$	to dismount the tape in software.

Also, please remove your tape from the tape drive promptly so that other users will know that it is again available for use.

3.3. UVSRT

The next step in image making is to sort the data for the FFT. This is done with UVSRT:

> TASK 'UVSRT' ; INP $\subset_R$	to set the task name and list the input parameters.
> INDI *n* ; GETN *ctn* $\subset_R$	to select the input file, where *n* is the disk number with the *uv* data and *ctn* is its catalog number on that disk. ($n = 3$ and $ctn = 1$ from our UCAT example).
> OUTN INNA ; OUTCL 'UVSRT' $\subset_R$	to set the output file name to the same as the input file name and the output file class to UVSRT. These are actually the defaults.
> SORT 'XY' $\subset_R$	to select the "XY" sort type required for image making.
> INP $\subset_R$	to review the inputs you have selected. *N.B.*, check them carefully since the sort can be a very time-consuming step for large data sets.
> GO $\subset_R$	to run the task UVSRT.

On a VAX 780 under VMS, UVSRT will take about 15 minutes to sort a data set with 100,000 visibility points (longer if the machine is being heavily used) and will need two scratch files of about 15,000 blocks each. Sorting times increase as somewhere between $n\log(n)$ and n^2 and the scratch files go as n. If the disks are filled, the job may get terminated. Check in §4.6 for things to do if this happens.

Once UVSRT has finished, check that a *uv* data base with the "class" UVSRT has appeared in your disk catalog by:

> INDI 0 ; UCAT $\subset_R$

The catalog listing might now look like:

```
CATALOG ON DISK  3
CAT USID MAPNAME       CLASS   SEQ  PT     LAST ACCESS       STAT
  1   76 3C138 A C    .UVDATA .    1 UV 22-OCT-1985 12:33:34
  2   76 3C138 A C    .UVSRT  .    1 UV 22-OCT-1985 12:56:50
```

Note that the catalog number of the sorted file need not be contiguous with that of the unsorted file. On those *AIPS* systems which are configured to have "public" catalog files, the file numbers will often not be contiguous. Many *AIPS* installations, including the NRAO VAX systems at the VLA and Charlottesville, have "private" catalog files, in which your *uv* files will have contiguous catalog numbers starting from 1 when you first write *uv* data to disk.

Once the sorted data base is made, please delete the unsorted data unless you plan to self-calibrate your data. Data bases can be very large and there is no need to keep the unsorted data on disk if you do not intend to self-calibrate, as it can be reloaded quickly with UVLOD. To do the deletion:

> INDI *n* ; GETN *ctn* ⏎ where *n* and *ctn* select the data set to be deleted. (*n* = 3, *ctn* = 1 here.)

AIPS will return the name of the selected data set. Check this; then:

> ZAP ⏎ will do the deletion.

3.4. Image making (UVMAP and MX)

There are two tasks which may be used to grid and Fourier transform the sorted data. The simpler one, UVMAP, will be described first. The other, called MX, can also do deconvolution by the CLEAN method and will be described second. Matters of image making strategy to be considered in using these tasks are discussed in §§ 3.4.3 and 3.4.4.

Note that radio synthesis images are made in a rectangular coordinate system of direction cosines that represents a projection of angular coordinates onto a tangent plane. Over wide fields of view, the image coordinates are not simple scalings of right ascension and declination. For details of coordinate systems supported by *AIPS*, please consult *AIPS* Memo No. 27, "Nonlinear Coordinate Systems in *AIPS*," by Eric W. Greisen.

For many practical purposes, it is sufficiently accurate to suppose that parameters such as image shifts (*e.g.*, RASHIFT below) correspond to simple angular shifts of the image on the sky, and *AIPS* input terminology reflects this simple approximation. Actual coordinate shifts and transformations in *AIPS* are accomplished using the full nonlinear expressions, however. This should be borne in mind when relating shifts in pixels (image cells) to shifts in sky coordinates (α, δ) manually.

3.4.1. UVMAP

The task which simply does gridding and Fourier transformation of the sorted data is UVMAP. Examples of its inputs follow:

> TASK 'UVMAP' ; INP ⏎ to see you what you have to specify.

> INDI *n* ; GETN *ctn* to select the *sorted* data set (*n* = 3, *ctn* = 2 here).

> STOKES 'IQU' to make a Beam, I, Q, and U images of the source (the default 'I' makes only an I image and beam).

N.B., The default STOKES 'I' will select *uv* points where LL and RR data are present *and* points where only LL or only RR are present. STOKES 'IQU' requires both LR and RL to be present and either LL or RR or both. As these criteria are not identical, beams output with class IBEAM from STOKES 'I' may not be identical to those output as IBEAM from STOKES 'IQU', and you may wish to distinguish them later for CLEANing purposes, using the RENAME command (see § 4.2 on "The Catalog file").

> IMSIZE 512, 256 ⏎ to make a 512 by 256 image—IMSIZE parameters must be powers of 2 from 32 to 2048 in UVMAP.

> CELL 0.5 ⏎ to select 0.5 arc sec cell size (*i.e.*, 256 by 128 arcsec field of view with the 512 by 256 IMSIZE specified above).

> UVTAPER 50, 40 $\subset_R$	to specify the widths to 30% of the Gaussian taper in x and y, in units of kilowavelengths (*i.e.*, the UVTAPER specified here is 30% at 50,000 and 40,000 wavelengths). Default: no UVTAPERing.
> UVRANGE 10, 100 $\subset_R$	to restrict the projected baselines used to the range 10,000 to 100,000 wavelengths. Default: no baseline restriction.
> UVWTFN 'NA' $\subset_R$	to specify "natural" weighting. The default, ' ', specifies "uniform" weighting.

Other inputs, as well as the last 3, are defaulted sensibly. Use:

> HELP *xxx* $\subset_R$

where *xxx* is a parameter name, *e.g.*, IMSIZE, UVWTFN, etc., to get useful information on what they specify. The default *uv* convolution function is a spheroidal function (XTYPE, YTYPE = 5) that has very good characteristics for suppressing aliasing. Check that you are satisfied with the inputs by:

> INP $\subset_R$

then:

> GO $\subset_R$	will run UVMAP.

The *AIPS* monitor provides some important messages while UVMAP is running. When you see UVMAP: ENDS SUCCESSFULLY on this monitor, you should find the requested images in your catalog using:

> INDI 0 ; MCAT $\subset_R$

This would produce a listing such as:

```
CATALOG ON DISK  2
CAT USID MAPNAME       CLASS   SEQ  PT      LAST ACCESS      STAT
 42   76 3C138 A C    .IMAP    .    1 MA 22-OCT-1985 13:50:10
 43   76 3C138 A C    .IBEAM   .    1 MA 22-OCT-1985 13:59:58
 44   76 3C138 A C    .QMAP    .    1 MA 22-OCT-1985 14:10:10
 45   76 3C138 A C    .UMAP    .    1 MA 22-OCT-1985 14:19:19
```

To get a hard copy listing of the catalog type:

> INDI 0 ; CLRN $\subset_R$	to clear all the inputs that specify file names.
> INTY 'MA' ; CAT ; PRTM $\subset_R$	to catalog images and print the listing on the system printer.

The UVMAP messages which contain vital parameters of your imaging task are logged in a history file that is cataloged on disk as an "extension" of your output image file (see § 4.3). This history file can be printed out by:

> INDI *n* ; GETN *ctn* ; PRTHI $\subset_R$	where *n* and *ctn* are the disk and catalog number of the desired file ($n = 3$, $ctn = 42$ here would select the I image).

3.4.2. MX

Another option for image making is the task MX, which combines imaging and CLEANing in one more general, and often more accurate, package. MX permits simultaneous imaging of up to 16 fields of view and up to 255 frequency channels, but can image only one Stokes parameter at a time. We discuss its more sophisticated options in § 5.1.2 under CLEANing and in § 9.3 (Spectral Line).

The following inputs would make a single I image and beam using MX:

> TASK 'MX' CR	
> INDI *n* ; GETN *ctn* CR	to select the sorted data set ($n = 3$ and $ctn = 2$ in our previous example).
> NMAPS 1 ; NFIELD 1 CR	to make one image of one field.
> STOKES 'I' CR	to select Stokes I for imaging.
> IMSIZE 1024 CR	to make a 1024 by 1024 image. IMSIZE values must be a power of 2 from 32 to 4096 in MX.
> CELL 0.1 CR	to select 0.1 arcsec cell size, *i.e.*, 102.4 arcsec square field of view with the 1024 by 1024 IMSIZE specified above.
> UVTAP 10, 20 CR	to specify the widths to 30% of the Gaussian taper in x and y in kilowavelengths.
> RASHIFT 200 ; DECSHIFT -100 CR	to shift the image center $\sim 200/(15\cos(\delta))$ sec of RA East, 100 arcsec south from the tangent point of unrotated uv data. (*N.B.*, if the uv data were ROTATEd by UVSRT, the shifts apply to the new, rotated coordinates).
> UVRANGE 0, 50 CR	to restrict the projected baselines used to the range 0 to 50,000 wavelengths. Default: no restriction.
> UVWT ' ' CR	to use default (uniform) weighting; 'NA' would specify natural weighting.
> NITER 0 CR	to do no CLEANing once the image is made.
> INP CR	to review the inputs. (You will need to CTRL-S your terminal at some point as there are many more inputs to MX than we have listed here!)

Then:

> GO CR	to run MX.

Scan the *AIPS* monitor for messages while MX is running. After you see MX: ENDS SUCCESSFULLY on this monitor, you should find the requested images in your catalog using:

> INDI 0 ; MCAT CR

The image file classname produced by MX for STOKES 'I' and NITER 0 is .IMAP; the default beam file classname is .IBEAM. The vital imaging parameters are logged in a history file that is cataloged on disk as an extension of your image file; see § 4.3 and use:

> INDI *n* ; GETN *ctn* ; DOCRT 1 ; PRTHI CR	to view this on your terminal.
> INDI *n* ; GETN *ctn* ; DOCRT -1 ; PRTHI CR	to generate hard copy on your printer.

where *n* and *ctn* are the disk number and catalog slot number of the required image.

3.4.3. Which to use — UVMAP or MX?

MX is generally a little faster than UVMAP (the exceptions being uncommon combinations of small images made from large datasets). MX makes a *uv* "workfile" which is used in its CLEAN step, so you save some computation by using MX rather than UVMAP for imaging when you want to use MX later for CLEANing. MX has the following advantages over the combination of UVMAP and APCLN (the task which CLEANs images made by UVMAP):

1. MX's CLEANing step allows you to CLEAN almost to the edges of the imaged field, whereas APCLN only CLEANs properly over a quarter of the field. MX's CLEANing step also subtracts the components from the ungridded data, avoiding aliasing of sidelobes into the field of view. These properties of MX make it the best choice for snapshot data.
2. MX lets you image up to 16 rectangular subfields, then CLEAN them all simultaneously. It is therefore the method of choice for all observations where there may be strong but localized sources over most of the primary beam. It could be prohibitively expensive in time and disk space to image and subsequently CLEAN a single huge field with UVMAP and APCLN in such cases.
3. MX is the only imaging program in *AIPS* that permits images as large as 4096 by 4096 cells, as occasionally required for high-resolution imaging of very strong extended sources. Note, however, that you would need more than 525,000 blocks of disk space (400,000 with CLEANing) to make one such image on a VAX, and may then be unable to CLEAN it in an acceptable time. (The disk space requirements of MX are complicated and do not completely scale with image size. A 2048 by 2048 with CLEANing would require 300,000 blocks of disk *plus* enough disk to hold a copy of the input *uv* file.) We strongly recommend that you consult an *AIPS* advisor at your site before making images over 1024 by 1024 in size!
4. MX does not convert the images to integer format, and subtracts components from the gridded *uv* data, taking proper account of the *w* terms. These features all lead to more accurate final CLEAN images than could be obtained with UVMAP and APCLN. The increased accuracy can be important if you seek very high dynamic range.

However, UVMAP may be preferable if you want to make images of several Stokes parameters simultaneously (and with identical *uv* coverage), and if you will CLEAN an area no larger than a quarter of the field area and do not require the ultimate in final dynamic range. Multi-Stokes imaging with UVMAP involves fewer separate steps than with MX, and images are made by UVMAP in the integer format expected by image arithmetic programs such as COMB (§ 8.1), whereas images from MX must be converted to this format. UVMAP may have some advantages for spectral-line work as well (see § 9 of this *COOKBOOK*).

The main danger in using MX is one of observer psychology. Because MX combines imaging and CLEANing, some observers feel encouraged to CLEAN images without ever inspecting the dirty one. This is extremely bad practice. They can thereby CLEAN images that are dominated by noise (wasting both their own time and CPU cycles). Worse, they may fail to specify CLEAN boxes properly, leaving significant emission outside the CLEAN area, or CLEANing large areas of empty sky. These practices can lead to false results and/or needlessly increase the time to produce an image of optimum quality.

Do not use MX blindly. Combine the imaging and CLEANing steps only after you have diagnosed some elementary properties of your field — where the brightest emission is, what shape it is, and whether it is bright enough to justify CLEANing anyway. It is a good idea to make the first images of your field at the lowest resolution (heaviest taper) justified by your data. This will allow you to choose the input parameters for subsequent combined imaging and CLEANing steps optimally.

3.4.4. Helping the deconvolution when imaging

Here are a few points to bear in mind when setting up the imaging parameters in UVMAP or MX if you think you will need to deconvolve the synthesized images later.

Other things being equal, the accuracy of beam deconvolution algorithms (§ 5.1) generally inproves when the shape of the dirty beam is well sampled. When imaging complicated fields, it may be necessary to compromise between cell size and field of view. If you are going to CLEAN an image, you should set your imaging parameters so that there will be at least three or four cells across the main lobe of the dirty beam.

With UVMAP, you must image a large enough field that no strong sources whose sidelobes will affect your image have been aliased by the FFT. With MX, make a small image field around each confusing source.

You help CLEAN to guess what may have happened in the unsampled "hole" at the center of the *uv* plane by including a zero-spacing (usually single-dish) flux density when you make the image. This gives CLEAN a datum to "aim at" in the center of the *uv* plane. Extended structure can often be reconstructed by deep CLEANing when the zero-spacing flux density is between 100% and ~125% of the average visibility amplitude at the shortest spacings (run UVPLT to estimate this average for your data set). If your data do not meet this criterion, there may be no reliable way for you to image the extended structure of your source without adding further information to your observations (*e.g.*, by adding *uv* data from a more compact array, by Fourier transforming a suitably tapered and deconvolved single dish image of the VLA primary beam, or by using such an image as the default image for a maximum entropy deconvolution as in § 5.1.6).

If UVPLT shows a rapid increase in visibility amplitudes on the few shortest baselines in your data, but not to a value near the integrated flux density in your field, you may get better images of the *fine* structure in your source by excluding these short baselines with the UVRANGE parameter.

If your source has complicated fine structure and is at declinations south of +50°, there may be important visibility structure in the outer regions of the *uv* plane which the VLA samples sparsely at these declinations, even after "full synthesis" imaging. In such cases, CLEAN may give images of higher dynamic range if you are not too greedy for resolution at the imaging stage and use either UVTAPER or UVRANGE to down-weight or exclude poorly sampled outer segments of the *uv* plane.

3.5. Coriander banana nut bread

1. Blend together in a large bowl $1\frac{2}{3}$ cups sifted all-purpose **flour**, 3/4 cup **sugar**, 1 tablespoon **baking powder**, 1/2 teaspoon **baking soda**, 1/2 teaspoon **salt**, 2 teaspoons ground **coriander**.
2. Mix in $1\frac{1}{2}$ cups chopped unblanched **almonds** and set aside.
3. Melt 1/3 cup **shortening** and set aside to cool.
4. Mix until well blended 1 large well-beaten **egg**, 1/4 cup **buttermilk**, and 1 teaspoon **vanilla extract**.
5. Blend in $1\frac{1}{4}$ cups mashed ripe **bananas** and the shortening.
6. Make a well in center of dry ingredients and add banana mixture all at one time. Stir only enough to moisten dry ingredients.
7. Turn into greased 9 × 5 × 3-inch loaf pan and spread to corners.
8. Bake at 350° about 1 hour or until a wooden pick comes out clean when inserted in center of bread. Immediately remove from pan and set on rack to cool.

4. SOME IMPORTANT UTILITIES

Before going on to further image processing, you should become more familiar with a few *AIPS* and system utilities, some of which have already been encountered in the image-making steps of § 3 above. Lists of available utilities can be obtained at your terminal by typing HELP CATINFO CR_R and HELP GENERAL CR_R. See also § 14.

4.1. The Message file

Many *AIPS* programs will produce messages on the *AIPS* monitor — the black-and-white alphanumeric screen near your terminal or, in its absence, your terminal itself. Most of the messages that fly by on this monitor are also written to disk in a file called the message file. The message file can be printed out by typing PRTMSG CR_R following a > prompt at your terminal. PRTMSG is an example of an *AIPS* "verb." Unlike an *AIPS* "task," it does not need a GO in order to execute and it is *not* shed from your terminal. Because they tie up the terminal while executing, "verbs" are generally simple programs which do not take long to execute.

Each entry in the message file has an associated priority: 0 for user input, 2 for "unimportant" messages, 5 for general output messages, and 8 for serious error messages. To set the priority level for hard copy output of messages type:

> PRIORITY *np* CR_R where *np* is the desired minimum level,

before running PRTMSG; then only messages at this level or above will be listed on the printer. If *np* is ≤ 5, then messages at level 0 are also printed. PRTMSG has additional parameters which allow you to limit the output by program name (PRTASK), time (PRTIME), and *AIPS* number (PRNUM) and to have it displayed on your terminal (DOCRT 1) rather than on the printer (DOCRT -1). PRTMSG does not delete messages from your file. Use:

> CLRMSG CR_R to delete messages and compress the file.

Note that CLRMSG supports parameters like those of PRTMSG. Old messages are automatically deleted from your file when you EXIT from AIPS. (The time limit for "old" messages is set by your *AIPS* Manager and is typically about 3 days.)

4.2. The Catalog file

A summary record of your *AIPS* data sets (*uv* data, images, beams) is kept in a disk file called the catalog file. To interrogate your disk catalog, use:

> INDI 0 ; MCAT CR_R to list all images.

> INDI 0 ; UCAT CR_R to list all *uv* data sets.

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

> CLRNAME CR_R to reset adverbs INNAME, INCLASS, INSEQ, INTYPE, INDISK.

> CAT CR_R 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 an appropriate combination of the parameters INNAME, INCLASS, INSEQ, INTYPE, and INDISK, respectively. Unless a hard copy or a limited part of the catalog is desired, it is faster to use MCAT and UCAT. A typical listing looks like:

```
CATALOG ON DISK  1
CAT USID MAPNAME      CLASS   SEQ  PT     LAST ACCESS      STAT
 18   76 3C166L5OK   .IMAP   .   1 MA 27-OCT-1985 22:30:18
 19   76 3C166L5OK   .IBEAM  .   1 MA 27-OCT-1985 23:02:14
 22   76 3C166L5OK   .IMAP   .   2 MA 28-OCT-1985 15:30:45
CATALOG ON DISK  2
CAT USID MAPNAME      CLASS   SEQ  PT     LAST ACCESS      STAT
 22   76 1200+519    .IMAP   .   1 MA 01-NOV-1985 23:50:10
 23   76 1200+519    .IBEAM  .   1 MA 01-NOV-1985 23:59:58
 24   76 1200+519    .QMAP   .   1 MA 28-OCT-1985 00:10:10
 25   76 1200+519    .UMAP   .   1 MA 28-OCT-1985 00:19:19
 28   76 1200+519    .ICLN   .   1 MA 02-NOV-1985 00:35:20 WRIT
 31   76 SCRATCH FILE.MX1    .   1 SC 02-NOV-1985 00:35:37 WRIT
 32   76 SCRATCH FILE.MX1    .   2 SC 02-NOV-1985 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-1985 12:56:50
 36   76 1200+519    .UVXY   .   1 UV 02-NOV-1985 00:32:50 READ
 37   76 1200+519    .UVWORK .   1 UV 02-NOV-1985 00:34:25 WRIT
```

This user (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 MX *uv* work file on disk 3. There are two scratch (temporary) files on disk 2 which were created by MX running out of AIPS1 (hence their MX1 classname). Image data files (images and beams) are distinguished by the type code MA in this list. The *uv* data files are distinguished by the type code UV and the scratch files by type SC.

Note that this user has given the image files on disk 1 image names which encode useful information other than the source name. These were images of 3C166 at L band with 50 kilowavelength (*uv*) taper. Such information could be extracted from history files, but it is useful to carry it at a level where it can be seen by CAT. He has also given the UVSRT file in slot 2 on disk 3 a name which encodes the source name (3C138), the configuration (A), and the frequency (C). Careful choice of filenames can save you a lot of other bookkeeping. Note how SEQ numbers are used to distinguish different versions of a file with the same name. The file name can be any valid string up to 12 characters in length.

4.2.1. Renaming data files

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

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

There are two verbs which can alter the catalog numbers assigned to files. RENUMBER moves a file to an empty, user-specified slot. RECAT simply compresses the catalog without changing the order of the entries in the catalog.

4.2.2. Speedy data file selection

Associated with each catalog entry is an identification number called the "catalog slot number". The CAT column at the left of the listing shows these catalog numbers. They can be used conveniently to initialize inputs for *AIPS* programs which read data sets from disk. Use:

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

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

> IN2D *n2* ; GET2N *ctn2* CR to set the second set.

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

4.2.3. Catalog entry status

Note that several catalog slots on disks 2 and 3 in our example catalog listing above do not have blank entries in the STAT column. This listing could have been taken while the user was running a CLEAN under MX on the sorted *uv* data set in slot 36, as this file is opened for READing. The CLEAN image file, ICLN in slot 28, and the scratch and UVWORK files are opened for WRITing. Procedures which attempt to read files which are opened for writing, or vice versa, will be rejected with appropriate error messages, so you should note any non-blank entries in the STAT column carefully. In some situations, mainly involving system crashes or abortion of tasks, files may be left in READ or WRIT status indefinitely. The file status may be reset in such cases by use of CLRSTAT CR following 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 file is corrupt and should be deleted using the verb ZAP following use of CLRSTAT.

Check that a data set you are about to use has a clear status before running any *AIPS* programs that will use it. (It is often possible to let two tasks read the same file at the same time, but this is not recommended.) Also note its ordinal number in the catalog, as this will be useful for GETN.

4.2.4. Other catalog listings

You can get a listing of the image header file on your terminal by following the GETNAME step above with

> IMHEAD CR for a detailed listing.

> QHEAD CR for a shorter listing.

QHEAD reports the position at the *numeric* center of the image while IMHEAD reports the position of the "reference pixel". The output of these verbs may also be printed via PRTMSG (at PRIORITY 2).

4.3. History files

Associated with each *uv* and image file is a "history" file giving important information about the processing done so far on the data in the file. Every *AIPS* task and verb, that alters the data or header, records the parameters used by it in this file. In general, each "card" in the file begins with the task or verb name and then gives one or more of the input parameter values actually 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 TRUE ⏎	to direct the display to your terminal.
> 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.

4.4. Saving and restoring inputs

All input parameters ("adverbs") are global throughout AIPS. When a parameter value is specified for any program, it will remain at that value in any other program which has an input of the same name until you change it. This global nature of the input parameters is useful in some cases, but can be inconvenient in others. Before running any task or verb check the input parameters carefully with:

> INP *name* ⏎ where *name* is the program name.

The parameters you have specified for *AIPS* at any given time may be saved on disk by typing:

> SAVE *aaaaa* ⏎ where *aaaaa* is any string of up to 12 characters.

> GET *aaaaa* ⏎ will restore these inputs at a later time.

These commands save or restore all the inputs and the rest of your AIPS "environment". For this reason, they must be the only commands on the input line. AIPS automatically saves the environment in an area called LASTEXIT whenever an EXIT or RESTART command is processed. A GET LASTEXIT is automatically executed whenever that user logs back in to the AIPS program. Thus, each user retains his own environment from one use of AIPS to the next. To obtain a null version of the adverb values and the rest of the environment, type:

> RESTORE 0 ⏎

There are three temporary areas for saving the user's AIPS environment as well. To save your inputs temporarily, type:

> STORE *n* ⏎ to save the inputs in area *n*, where *n* = 1, 2, or 3.

> RESTORE *n* ⏎ to recover the inputs previously stored in area *n*.

The input parameter 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* ⏎

This avoids, to some extent, the global nature of the adverb values in AIPS. Whenever a task is executed (by the verb GO), TPUT is run automatically. TGET will therefore produce the set of input parameters used for the last execution of the task, unless you deliberately overwrite them with a TPUT of your own.

4.5. The verb GO

GO is shown in examples throughout this *COOKBOOK*. What is not emphasized elsewhere, and is often overlooked by users, is the fact that GO has inputs just like other verbs. You may specify which task you want either with an immediate argument, *i.e.*, GO UVSRT ⏎, or with the parameter TASK, *i.e.*, TASK 'UVSRT' ; GO ⏎. GO has two other parameters, DOWAIT and VERSION, as well. The former is passed to the

task and instructs it to resume AIPS as soon as possible (DOWAIT FALSE $\subset_R$) or to resume AIPS only after completing its operations (DOWAIT TRUE $\subset_R$). The latter allows the task to return a meaningful error code to which AIPS may respond by aborting the current input line, procedure, FOR loop, etc. The verb WAIT *'taskname'* also forces AIPS to wait for a task to complete, but it cannot respond to some failure in that task. For example, the line:

> GO UVSRT ; WAIT UVSRT ; GO UVMAP $\subset_R$

may cause unwanted images to be generated by UVMAP if UVSRT fails for lack of disk space or some other reason. However, the line:

> DOWAIT TRUE ; GO UVSRT ; GO UVMAP $\subset_R$

will not attempt to execute UVMAP if UVSRT fails. Note that AIPS will not get hung up when a task aborts even if DOWAIT is true.

VERSION, the last input to GO, is used to specify which version of the program you wish to execute. For example, it allows you to use the TST or OLD versions of a task from the NEW version of AIPS. It also allows private versions of programs to be executed. Type HELP VERSION $\subset_R$ for details.

GO has another useful capability. Normally, in order to invoke a verb, you simply type its name, *e.g.*,:

> PRTMSG $\subset_R$ to print the message file contents.

However, if you type, instead:

> GO PRTMSG $\subset_R$

having forgotten that PRTMSG is a verb, then AIPS will actually execute:

> TPUT PRTMSG ; PRTMSG $\subset_R$ to save the current PRTMSG inputs and then print the contents of the message file.

You can recover those inputs at a later time with TGET PRTMSG $\subset_R$.

4.6. Monitoring the disk space

Disk space is generally at a premium and there are several ways to monitor the available amount of space. Within AIPS there is:

> FREE $\subset_R$ to list the total space available on each *AIPS* disk and other useful information.

and

> USER 32000 ; INDISK 0 $\subset_R$ to get all disks and users.

> GO DISKU $\subset_R$ to run the *AIPS* disk user task.

This will (eventually) list on the *AIPS* monitor the amount of data space in use by each user for all *AIPS* disks. Identify the worst disk hogs and apply appropriate peer pressure.

Sometimes the available disk space has been eaten up by *AIPS* scratch files which are no longer being used. Tasks which abort and other mysterious events can cause this situation to arise. To delete all your scratch files, except those for tasks which are still running, type:

> SPY $\subset_R$ to see which tasks are running.

> SCRD $\subset_R$ to delete the files.

Scratch-file destruction is now fairly safe, so SCRD is run automatically whenever EXIT, RESTART, or ABORT *taskname* are executed. Tasks MX, ASCAL, UVMAP, APCLN, and UVSRT often create rather large scratch files.

The verb TIMDEST will destroy all data sets which have not been used in some minimum time interval. In standard versions of *AIPS*, this time interval is 14 days. TIMDEST will also delete messages older than 3 days from all users' message files. The parameters of TIMDEST allow you to request less stringent cutoffs. Your local *AIPS* Manager may set other limits on the time ranges.

Section 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. To use these methods:

> EXIT CR to exit from AIPS, saving your AIPS inputs in the LASTEXIT area.

Then consult Appendix Z for instructions appropriate to your machine.

4.7. Moving and compressing files

There are two tasks which may be used to move files from one disk to another with options to reduce the size of the files. They are SUBIM, used on images, and UVCOP, used on *uv* data sets. SUBIM uses the parameters BLC and TRC to select a portion of the input image. If these parameters are defaulted (set to 0), the entire image is copied. Clean component and history extension files are copied as well, but plot and slice extensions are not. Similarly, UVCOP uses the parameter array BPARM to select a range of *uv*-sample times to copy. If BPARM is zero, all data are copied except for completely flagged records. If you have done extensive data editing, UVCOP may produce a rather smaller data set even when the whole time range is copied. Antenna and gain extension files are copied, but plot files are not.

4.8. Finding information in *AIPS*

Much of the documentation of *AIPS* can be displayed on the terminals by typing HELP *word* CR, where *word* can be the name of a 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. To have the HELP information printed on the line printer, enter EXPLAIN *word* CR instead. For the more important verbs and tasks, EXPLAIN will print extra information giving more detailed explanations, hints, cautions and examples to assist users of the program.

There are more general help files which list a variety of information. Some illustrate the use of features in *AIPS* (try HELP BATCHJOB CR for a description of the batch processor). Others list the currently working software (*e.g.*, HELP ANALYSIS CR gives a listing of software which analyzes images). These files should be consulted often. All of these general HELP files are listed in § 14 of this *COOKBOOK* and give a summary of the software which is currently available in *AIPS*. They are updated as new software is developed. HELP and EXPLAIN have other capabilities as well. Type:

> HELP CR

or

> HELP HELP CR to get a summary.

5. IMPROVING IMAGES — DECONVOLUTION, SELF-CAL, EDITING

This Section deals with the major programs which can be used to improve the quality of dirty images which are not noise-limited but which suffer from dynamic range problems. See Lectures 7–11 in the 1985 *NRAO-VLA Workshop on Synthesis Mapping* for general discussion and guidance about image defects and the strategies for reducing them.

5.1. Deconvolving images

The most widely used deconvolution method is CLEAN. Images made with UVMAP should be CLEANed using the task APCLN described in § 5.1.1. MX combines imaging and CLEANing functions in a single task. Its CLEAN option uses many of the same inputs as APCLN, plus some extra items described in detail in § 5.1.2. Sections 5.1.3 through 5.1.5 give some tips, which apply to both APCLN and MX, on monitoring the progress and success of a CLEAN and on restarting CLEANs.

Deconvolution can also be done by a variety of other methods, several of which are available in *AIPS*. The most popular of these alternatives is a maximum entropy-like algorithm implemented in the task VM. It is described in § 5.1.6.

5.1.1. APCLN

APCLN implements an array-processor CLEAN of the type devised by Barry Clark (*Astron. Astrophys.*, **89**, 355 (1980)). CLEAN components are found during "minor" iteration cycles by CLEANing the brightest parts of the residual image in the array processor with a "beam patch" of limited size. More precise CLEANing is achieved at the ends of "major" iteration cycles when the Fourier transform of the CLEAN components is computed, multiplied by the transform of the beam, transformed back to the image plane, and then subtracted from the dirty image. The rule for deciding when a major iteration should end in order to achieve a desired accuracy is complicated (see the Clark paper). APCLN allows the user to vary the rule somewhat to suit the requirements of the particular image and to control all the usual options associated with CLEANing.

Type EXPLAIN APCLN $\crarr$ to list useful guidelines on the line printer. Then enter:

> TASK 'APCLN' ; INP $\crarr$	to tell you what you have to specify.
> INDI *n1* ; GETN *ctn1* $\crarr$	to select the dirty image, where *n1* is the disk on whichit is stored and *ctn1* is its catalog number there.
> IN2D *n2* ; GET2N *ctn2* $\crarr$	to select the dirty beam, where *n2* and *ctn2* are its disk and catalog numbers.

Examples of other inputs to APCLN are:

> OUTN '3C138 A C' $\crarr$	to specify the CLEAN image name — default: same as INNAME.
> OUTS 0 $\crarr$	to create a new output file. If OUTSEQ $\neq$ 0, the specified value is used. OUTSEQ must be set to restart a CLEAN (§ 5.1.4).
> OUTC 'ICLN1' $\crarr$	to specify the output file class. It defaults to 'ICLN' for INCLASS IMAP, 'QCLN' for QMAP, 'UCLN' for UMAP, and 'VCLN' for VMAP.
> GAIN 0.2 $\crarr$	to set the loop gain parameter, defaults to 0.1. Values in the range 0.1 to 0.25 may be suitable for complicated sources, higher values are okay for simple, point-like sources.
> FLUX 0.003 $\crarr$	to stop CLEANing when peak of residual image falls to 3 mJy.

> NITER 500 $\subset_R$	to stop CLEANing when 500 components have been subtracted (default 100).
> BMAJ 1.25 $\subset_R$	to set the major-axis FWHM of restoring beam to 1.25 arcsec. BMAJ = 0 specifies that the beam is to be fitted. BMAJ < 0 means that the *residual* image is to be stored in the output file.
> BMIN 1.10 $\subset_R$	to set the minor-axis half-width of restoring beam to 1.10 arcsec.
> BPA 52.8 $\subset_R$	to set the position angle of beam axis to 52°48′ measured CCW from North (East from North).
> INVERS 1 $\subset_R$	to specify which version of the CLEAN components file(s) is desired. Default is 1.
> NBOXES 1 $\subset_R$	to set the number of boxes in which to search for CLEAN components. Must be ≤ 10, default is 1.
> BOX 156, 156, 356, 356 $\subset_R$	CLEANing box is 201 × 201 centered on pixel (256, 256). Default 0, 0, 0, 0 specifies inner quarter of image area. Fill BOX with more windows if NBOXES > 1. Note that APCLN cannot CLEAN properly over more than one-quarter of the image area.

The TV cursor can be used to set the CLEANing boxes if the TV display shows either the dirty image or a previous version of the CLEAN image. Type:

> NBOXES *n* $\subset_R$	to set the number of boxes, where $n \leq 10$.
> TVBOX $\subset_R$	to begin an interactive, graphical setting of the *n* boxes.

Position the TV cursor at the bottom left corner of the first CLEANing box and press a trackball button. Then position the cursor at the top right corner of the box and press Button B. Repeat for all desired boxes. This will fill the box array for APCLN. Note that the terminal will display some additional instructions. These may be followed to reset any of the previously set corners should you need to do so.

To continue the example parameters:

> DOTV TRUE $\subset_R$	to show residual images on TV in interactive AIPS and to provide an opportunity to terminate the CLEAN after each major cycle using trackball button D. The residual image is displayed on the TV after each major cycle and the CLEAN image is displayed at the end, if DOTV is set to TRUE (+1.0). Default = FALSE (−1.0) for no display. (See also § 5.1.3.)
> INP $\subset_R$	to review what you have specified.

Note that specifying INNAME and (usually) NITER may be enough to run the program if your dirty image and beam have the same image name (differing only by their classnames, *e.g.*, IMAP and IBEAM) and if you are content to use the system defaults for the CLEANing parameters. These defaults are reasonably chosen for CLEANing a complicated image.

The FACTOR parameter in APCLN can be used to speed up or to slow down the CLEANing process by increasing or decreasing the number of minor cycles in the major cycles. The default FACTOR 0 causes major cycles to be ended using Barry Clark's original criterion. Setting FACTOR in the range 0 to +1.0 will speed up the CLEAN, by up to 20% for FACTOR 1.0, at the risk of poorer representation of extended structure. Setting FACTOR in the range 0 to −1.0 will slow it down, but gives better representation of extended structure. For further details on the use of non-standard FACTORs, consult with Bill Cotton in Charlottesville.

When you're satisfied with the inputs listed by INP, then:

> GO $\subset_R$	to start the CLEAN.

5.1.2. MX

In its CLEANing mode, MX finds the individual components in "minor" iterations by the same methods as APCLN. However, MX then subtracts CLEAN components from the ungridded uv data using either a direct Fourier Transform (DFT) or a hybrid DFT/gridded Fast Fourier Transform (gridded-FFT) to compute the model visibility from the CLEAN components. The DFT is more accurate, but is often slower than the gridded-FFT (except in some multiple-field cases — EXPLAIN MX CR will print out some text containing, among many other things, details of timing as a function of data set size and number of CLEAN components). You can let MX choose which method to use as its CLEANing progresses by using the default (null) value of its input parameter CMETHOD; we strongly recommend use of this default unless you are sure which method to specify for your case (again, use EXPLAIN MX CR to print out some detailed advice on this).

CLEANing with subtraction from the ungridded data avoids aliasing of sidelobes into the field of view. In many cases, therefore, MX CLEANs more accurately than does APCLN over a much larger portion of the dirty image.

MX can also deconvolve components from up to 16 fields of view simultaneously (provided that they were imaged simultaneously using MX's imaging step), taking correct account of the w term at each field center. This is a vital advantage if you have many localized bright emission regions scattered throughout your primary beam; only the regions containing significant emission need to be imaged and cleaned, rather than the entire (mainly empty) area of sky encompassing them all. To take advantage of this property of MX to CLEAN high-resolution images on the basis of knowledge about the field derived from prior information (or from a low-resolution image made from the same data set), you will need to invoke the multifield options at MX's imaging step. An illustrative set of input parameters might be:

> NFIELD 3 CR	to specify that you want to image 3 fields.
> IMSIZE 128	to specify that the minimum image size for any of the fields should be 128 by 128.

> RASHIFT = -70, 0, 250 CR

> DECSHIFT = 120, 10, -400 CR

to specify that the first field should be centered on $(-70'', 120'')$, the second on $(0'', 10'')$ and the third on $(250'', -400'')$ relative to the original tangent point. You might then specify the CLEANing areas within each field by the parameter:

> FLDSIZ 32, 32, 246, 246, 20, 20 CR

These inputs would produce three images. The first would be 128 by 128 cells centered on $(-70'', 120'')$; it would be searched for CLEAN components over its center 32 by 32 cells. The second would be 256 by 256 cells centered on $(0'', 10'')$; it would be searched for CLEAN components over its center 246 by 246 cells, leaving a five-cell "guard band" around the CLEANing area to avoid interpreting the region most likely to be contaminated by effects of aliasing and the gridding convolution. The third would be 128 by 128 cells centered on $(250'', -400)$. It would be searched for CLEAN components over its center 20 by 20 cells. The beam would be 256 by 256 cells centered on $(0, 0)$.

Most of MX's CLEANing inputs are like those described under APCLN in § 5.1.1. The following are used exactly as in APCLN: OUTN, OUTS, OUTCL, GAIN, FLUX, NITER, BMAJ, BMIN, BPA, and FACTOR. NBOXES and BOX in MX work as in APCLN, but apply only to the first field defined by the imaging inputs. If NBOXES = 0, the CLEAN search area of field number 1 is determined from the first FLDSIZ window. If NBOXES> 0, FLDSIZ is used only to check the size of the image required for field 1. DOTV = -1 in MX specifies no residual display on TV channel 1 (as in APCLN) but DOTV = n with $1 \leq n \leq 16$ in MX specifies TV display of the residuals from field number n.

MX has the following inputs which differ substantively from those of APCLN:

> INDI *n* $^{C}_{R}$	to specify the number of the disk on which the original *uv* data set used to make the images resides.
> INNAM *'filename'* $^{C}_{R}$	to specify the filename of the original *uv* data set.
> INCL *'classname'* $^{C}_{R}$	to specify the classname of the original *uv* data set.
> INSEQ *seqn* $^{C}_{R}$	to specify the sequence number of the original *uv* data set.
> IN2DI *n* $^{C}_{R}$	to specify the number of the disk on which resides a *uv* work file in which MX stores the current *uv* data with the current list of CLEAN components subtracted. The file is created at the imaging step with default name the same as the input *uv* data set and default class .UVWORK. It is modified as MX's CLEAN progresses.
> IN2N *'filename'* $^{C}_{R}$	to specify the filename of the *uv* work file; default is INNAM.
> IN2CL *'classname'* $^{C}_{R}$	to specify the classname of the *uv* work file; default is UVWORK.
> IN2SEQ *sqn* $^{C}_{R}$	to specify its sequence number; default is highest.

The above inputs may of course be set using:

> INDI *n1* ; GETN *ctn1* $^{C}_{R}$

> IN2DI *n2* ; GET2N *ctn2* $^{C}_{R}$

as usual, where (*n1*, *ctn1*) and (*n2*, *ctn2*) select the disk numbers and catalog slot numbers of the appropriate files. Alternatively, if you have previously run MX with the same parameters, TASK 'MX' ; TGET $^{C}_{R}$ will restore them.

Note that the *uv* workfile will exist, and you should specify its parameters, even if you have only made a dirty image or images with MX previously. If you do not specify the workfile parameters, MX will repeat all steps of making the dirty image, which wastes time.

Note also that MX will remake the beam file whenever it is restarted, as the beam it uses for CLEANing need not have the same size as the image. (During CLEANing, it uses the smallest beam it thinks it can get away with). We therefore recommend that you delete or rename the beam file from any previous run of MX whenever you restart analysis of the same data set with MX, to prevent confusion later. (Renaming the full-size beam made when MX is used without CLEANing may be useful if you subsequently want to use it for another type of deconvolution, such as VM, which needs such a beam file).

To specify the default component subtraction method, type:

> CMETHOD ' ' $^{C}_{R}$	to allow MX to use DFT or gridded-FFT component subtraction at each major cycle, depending on which is faster. We strongly recommend use of this default unless you are very sure that you wish to force DFT or gridded-FFT subtraction at all iterations. 'DFT' forces DFT subtraction; 'GRID' forces gridded-FFT subtraction.

Before running MX, use:

> TASK 'MX' ; INP $^{C}_{R}$	to review your inputs (there are so many that it's easy to forget some).
> GO $^{C}_{R}$	to begin execution.

5.1.3. What to do while CLEAN is running

The *AIPS* monitor displays useful information about the progress of the CLEANing, and the TV itself displays the residual images after each major cycle (if DOTV was set to TRUE). You may change the image display parameters (*e.g.*, coloring, transfer function, zoom, etc.) using the display modification commands while the CLEAN is in progress to keep the residual image looking intelligible. When using the TV, APCLN and MX schedule 15-second pauses after each major cycle. The new residual image will be displayed on the TV monitor just before these pauses. Pressing trackball button D during the 15 seconds will end APCLN/MX in an orderly fashion at that point. Pressing any of buttons A, B, or C causes the execution to resume (as does waiting 15 seconds). The *AIPS* monitor prompts you for this action, which allows quasi-interactive CLEANing. Its use is encouraged when images are being CLEANed for the first time.

If you do not specify BMAJ and BMIN, a Gaussian CLEAN beam will be fitted to the central portion of the dirty beam. The results may not be desirable since the central portions of many dirty beams are not well represented by a single Gaussian and since the present fitting algorithm is not very elaborate. If you use the default, check that the CLEAN beam is a satisfactory representation of the dirty beam. Use task PRTIM on the central part of the dirty beam for this purpose.

When APCLN and MX terminate, a record of the CLEANed image is entered into your disk catalog and the image can then be displayed, contoured, etc. as described in §§6 and 7 below. The most important parameters of the CLEAN are logged in a "history" file that is cataloged as an extension of the CLEANed image file. This history file can be printed by:

> INDI *n* ; GETN *ctn* ⏎ — to select the CLEAN image, where *n* and *ctn* select its disk and catalog numbers.

> PRTHI ⏎

5.1.4. Restarting CLEANs

CLEANing can be restarted under control of the parameter BITER in the APCLN inputs or BCOMP in the MX inputs. BCOMP is an array of up to 16 values, one for each field imaged by MX. Set BITER / BCOMP equal to the number of components to be used from the previous CLEAN when CLEANing is restarted. When you are restarting a CLEAN, the OUTNAME and OUTSEQ parameters *must* be set explicitly to those of the previous CLEAN image since the CLEAN component list is associated with this image file (as its CC extension). When restarting a CLEAN with MX, you must also spcify IN2N, IN2CL and IN2SEQ explicitly to identify the *uv* workfile. An image can be re-convolved by setting NITER = BITER (in APCLN only) and specifying the desired (new) CLEAN beam.

Both APCLN and MX write over the CLEAN image file(s) as they proceed to clean deeper. Note that intermediate CLEAN images can be preserved if needed either by copying them with SUBIM or by writing them to tape with FITTP.

5.1.5. Listing CLEAN components

The list of CLEAN components associated with a CLEAN image can be printed by:

> INDI *n* ; GETN *ctn* ⏎ — to select the CLEAN image, where *n* and *ctn* select its disk and catalog numbers.

> BITER *n1* ; NITER *n2* ; XINC *n3* ⏎ — to list CLEAN components from *n1* to (*n1* + *n2* - 1) with increment *n3*.

> DOCRT TRUE ⏎ — to route the list to your terminal.

> DOCRT FALSE ⏎ — to route the list to the line printer.

> GO PRTCC ⏎ — to execute the task.

5.1.6. Alternatives to CLEAN — VM

The subject of image deconvolution has been widely studied and an enormous variety of methods proposed. Three of the alternatives to CLEAN have been implemented as *experimental* tasks in *AIPS*. These are algorithms due to Gerchberg and Saxton (APGS), van Cittert (APVC), and David Steer (STEER). Type EXPLAIN APGS ^{CR}, EXPLAIN APVC ^{CR}, or EXPLAIN STEER ^{CR} for further information on these tasks and, if needed, call Tim Cornwell at the VLA before using them. The most widely used and successful alternative to CLEAN has been the Maximum Entropy Method ("MEM") implemented in *AIPS* by the task VM. It requires a dirty image and beam, such as those produced by UVMAP or by MX with NCOMP set to 0, each twice the (linear) size of the region of interest. It deconvolves to produce an all-positive image which has as compressed a range of pixel values as the data allow. The final VM image is usually smooth, but provides "super-resolution" in regions of good signal-to-noise ratio.

There are three main reasons for preferring MEM over the CLEAN methods:

1. MEM can be much faster for images which have real signal in many pixels. "Many" seems to be $\geq 512^2$ or so.
2. MEM produces smoother reconstructions of extended emission than does CLEAN. The latter often yields images with many beam-sized lumps in low brightness, apparently smooth regions.
3. MEM allows introduction of *a priori* information about the image in the form of a "default" image.

Contrary to the received wisdom, MEM can be used for quantitative work on regions of good (> 10) signal-to-noise ratio, if the dirty image is convolved with a CLEAN beam prior to deconvolution. Use the *AIPS* task CONVL for this purpose. Note that MEM cannot be used on images which are not intrinsically positive, such as images of the Stokes Q, U, and V parameters. MEM can produce very nice deconvolutions, but requires careful control. We recommend studying the output of EXPLAIN VM ^{CR} before using the task. Lecture 7 of the 1985 *NRAO-VLA Workshop on Synthesis Mapping* also provides relevant information.

5.2. Self-calibration

The task ASCAL is a tool for obtaining images with high dynamic range when there is sufficient signal to noise in the *uv* data. It does this by comparing the input *uv* data set with the predictions of a source model in order to compute a set of antenna-based amplitude and phase corrections which would bring the data into better agreement with model, as a function of time throughout the observations. For an n-element array there are $(n-1)/2$ times more observations than unknown antenna gains at any time, so the process is well-determined when n is reasonably large. The input model may be either a point source (parameters set by BPARM(1), BPARM(2), BPARM(3) and BPARM(4) in ASCAL) or a set of CLEAN components derived from a previous image (parameters set by IN2DI, IN2NAME, IN2CLASS, IN2SEQ, INVER and NITER).

Do not use ASCAL unless your data are of sufficiently high signal to noise to warrant improvement. Ask yourself whether your externally-calibrated CLEAN images contain unCLEANable artifacts well above the noise, and whether your source meets the criteria for self-calibration given by Tim Cornwell in Lecture 9 of the 1985 *NRAO-VLA Workshop on Synthesis Mapping.* Note that if your images are limited by receiver noise, self-calibration may produce erroneous results.

5.2.1. Programs to run

If you decide to use ASCAL, a good sequence of steps is:

1. Use UVPLT to make a plot file showing the shape of the visibility function as a function of baseline length in the externally-calibrated data set. *N.B.*, for large data sets, use XINC to reduce the number of points plotted to no more than a few thousand; otherwise it will take too long to make and plot the plot file.
2. Use TKPL and the hard-copy switch on your TEK terminal, or PRTPL or QMSPL, to get hard copy of the above plot file.
3. If you can use a point-source model for the first iteration, *i.e.*, if a range of baselines sufficient to calibrate all antennas is dominated by a single component (flat visibility function well above the noise), go to step 6 directly.
4. If you must use a more complicated model, obtain a CLEAN component representation of it by making and CLEANing an image of the externally-calibrated data using either (UVMAP + APCLN) or MX. Note that you may want to use a higher loop GAIN in a CLEAN to be used as an input model for an early iteration of self-calibration than you would for final deconvolution of a very extended structure.
5. Use PRTCC to help you decide how many components from this CLEAN to include in the ASCAL model; when you have decided this, determine the appropriate *uv*-limits for the gain solution by referring to the hard copy of the visibility function you made at step 2.
6. Use UVSRT to sort the data from "XY" (pre-imaging) order back to "TB" order (unless you preserved the TB sort for self-calibration purposes).
7. Plan your ASCAL inputs; see the inputs list and the section on "Choosing ASCAL inputs" that follows below (§ 5.2.3).
8. Use ASCAL to calculate the gain corrections, to produce a new TB-sorted data set, and to catalog the gain corrections as an extension to the new *uv* data file.
9. Use GNPLT on the output data file, followed by TKPL, to review the gain corrections before proceeding further. (You may want to take hard copy of this output for future reference also — use the TEK hard-copy switch, or run PRTPL or QMSPL on the plot file extension of the new *uv* data file that was written by ASCAL.) In problem cases, GAPLT should be used to make more detailed plots of the gain file.
10. Ask whether the gain corrections were believable — were they smaller than at the previous iteration of ASCAL, if any? If not, is there a good reason why not — did you change input parameters such as the model, the type of solution, or the solution interval, in a way that may have forced larger corrections than before? Proceed only if you are reasonably sure you understand what is happening at this point — otherwise consult a local expert at your site.
11. If the corrections were believable, run UVSRT to produce a new version of the *uv* data in XY order for imaging. You should consider deleting one or more of the *uv* data sets from the previous iteration at this point to save disk space.
12. Go back to step 4 and repeat the whole process if your new CLEAN image is a significant improvement over the previous one (with comparable CLEANing parameters on both occasions). You may want to go back to step 1 and repeat the process from there if you have been using amplitude self-calibration and wish to check that your

amplitude calibration has not drifted significantly. If the new CLEAN image differs little from the previous one, do not continue on with further iterations of steps 5 through 11 unless you feel you can make an informed change to the ASCAL input parameters at step 7.

5.2.2. Inputs to ASCAL

The following input parameters are used by ASCAL:

> INDI *n1* ; GETN *ctn1* $\subset_R$	to select the 'TB' sorted *uv* data base.
> IN2D *n2* ; GET2N *ctn2* $\subset_R$	to select the CLEAN image to use.
> NCOMP = *n* $\subset_R$	to cut off the model at the n^{th} CLEAN component associated with this image. If $n < 0$, then use up to ABS(n) components, but stop at the first negative component.
> DOCAT 1 $\subset_R$	to catalog the gain file as an extension to the input *uv* data base. This is necessary to self-cal a spectral line data base and to display the gain solutions.
> APARM(1) = *x1* $\subset_R$	minimum baseline (in kilowavelengths) to be given full weight.
> APARM(2) = *x2* $\subset_R$	maximum baseline (in kilowavelengths) to be given full weight.
> APARM(3) = *x3* $\subset_R$	weight for baselines outside the APARM(1) → APARM(2) range.
> APARM(4) = *n4* $\subset_R$	reference antenna (choose a known good one for best results).
> APARM(7) = 1 $\subset_R$	to solve for amplitude and phase corrections.
> APARM(7) = 2 $\subset_R$	to solve for phase weighted by amplitude.
> APARM(7) = 3 $\subset_R$	to solve for phase ignoring amplitude.
> APARM(8) = 0 $\subset_R$	to solve for RR and LL separately.
> APARM(8) = 1 $\subset_R$	to average RR and LL correlators before solving.
> APARM(9) = *x9* $\subset_R$	to set the length of the solution interval (in minutes).
> APARM(10) = 0 $\subset_R$	to scale the gain corrections, when APARM(7) = 1, by the mean modulus of all gains to keep the flux density scale from drifting.
> APARM(10) = 1 $\subset_R$	to request no gain normalization.
> BPARM(1) = 0 $\subset_R$	to use CLEAN components as a model.
> BPARM(1) = 1 $\subset_R$	to use a single point source model instead.
> BPARM(2) = *y2* $\subset_R$	flux density of point source model (if BPARM(1) = 1).
> BPARM(3) = *y3* $\subset_R$	east offset of point source model in arcsec (if BPARM(1) = 1).
> BPARM(4) = *y4* $\subset_R$	north offset of point source model in arcsec (if BPARM(1) = 1).
> BPARM(7) = 0 $\subset_R$	to do no editing of data.
> BPARM(7) = 1 $\subset_R$	to flag "bad" data under control of BPARM(8) and BPARM(9).
> BPARM(8) = *y8* $\subset_R$	if BPARM(7) = 1, to flag full weight points that differ from the model by BPARM(8) times the rms residual.
> BPARM(9) = *y9* $\subset_R$	if BPARM(7) = 1, to calculate the antenna-IF running mean amplitude correction and flag an antenna-IF if that correction is more than BPARM(9) $\times\sigma$ away from the mean.

The parameter BPARM(5) is not used. Other parameters are defaulted sensibly — type EXPLAIN ASCAL $\subset_R$ for further information.

5.2.3. Choosing ASCAL inputs

In many cases, only a few ASCAL input parameters need be set, other than those selecting the *uv* data and the input model. The key parameters are NCOMP, APARM(1), APARM(9) and, if you are interested in polarization, APARM(4).

It pays to be conservative when using NCOMP to select the number of CLEAN components which will comprise the input source model. Setting NCOMP too high will fossilize errors from the earlier calibrations in the model for the next one; after this, you are stuck with them as long as you continue feeding ASCAL a model with as much CLEANed flux density. When calibrating I images, do not set NCOMP in ASCAL so high that any negative CLEAN components are included. The first few iterations of ASCAL should be phase-only calibration, since the tropospheric and ionospheric phase errors will almost always dominate amplitude errors due to the atmosphere or to system drifts. In these first iterations, it is prudent to be even more conservative, setting NCOMP so that the total CLEANed flux included in the model is between 50% and 80% of that at which the first negative CLEAN component appeared. If your field is dominated by a few very strong, small-diameter regions, it is a good idea to make the first iterations of ASCAL work on CLEAN components from these regions alone, restricting the range of baselines suitably via APARM(1).

It is always important to restrict the high-weight domain of the ASCAL solution to the part of the *uv* plane that is described well by the model. In the early stages of self-calibration, the trustworthy part of your CLEAN model will almost always contain less flux density than was measured in the visibility function at the shorter baselines. Another way of putting this is that the large-scale structure of the source will be poorly represented by the model. You should set APARM(1) so that the total flux density in the input model (the sum of the CLEAN components to the CLEAN iteration selected by NCOMP) exceeds the peak visibility amplitude in your data at a baseline of APARM(1) kilowavelengths (read this off a plot file output from UVPLT).

APARM(9) sets the length of the time interval, in minutes, over which the model and the data are averaged when computing the gain corrections. This must be *short* enough that the gain corrections can track the fluctuations produced by the atmosphere over the longer baselines with sufficient accuracy. It must be *long* enough that the variances of the computed gain corrections (which depend on the signal-to-noise ratios in the data over the *uv* range in which the model is being compared with the data) are acceptably small. These constraints vary from source to source, frequency to frequency, and (because of the "weather") from day to day. They may not in fact be reconcilable for weak sources, especially in the wider VLA configurations and/or at the higher frequencies. In many combinations of these circumstances, you may not be able to self-calibrate your data. See Lecture 9 in the 1985 *NRAO–VLA Workshop on Synthesis Mapping* for details of how to make this assessment.

APARM(4) selects the number of the reference antenna for the gain solutions. For total intensity continuum calibration, the choice of this ASCAL input is unimportant. It is always best, however, to choose a reference antenna that was stable and present in all data throughout the run, if only because this prevents propagation of noise or glitches in the reference antenna through the gain solutions (and plots of them) for the other antennas. For polarization work it is important to select an antenna for which both polarizations were always present, otherwise any polarization calibration which preceded ASCAL may be seriously compromised. For spectral line work, where ASCAL's companion program ASCOR may be used for interpolation of the antenna/IF gains, it is also preferable to select a stable antenna with a full duty cycle as the reference.

Note that ASCAL should almost always be run with APARM(7) set to its default (selecting phase-only calibration) for the first iteration or two. Consider turning on amplitude calibration by setting APARM(7) = 1 only when either the phase adjustments being made are generally small (*i.e.*, the worst cases being a few tens of degrees) or the new reCLEANed image is clearly dominated by amplitude errors — which will give symmetric Y-shaped patterns around strong point sources. In general, you will want to use the default

APARM(10) = 0 when using APARM(7) = 1, to restrict drifting of the flux-density scale during amplitude self-calibration.

Use of ASCAL to edit out bad data is a good idea during the later iterations at which the adjustments are becoming reasonably small (see above). The editing is turned off by default. To turn it on, set BPARM(7) = 1 and then ASCAL will flag those points used at full weight in the solution whose residuals are greater than BPARM(8) times the rms residual.

For further guidance and information on other ASCAL inputs, type EXPLAIN ASCAL $^{C}_{R}$ and/or read Lecture No. 9 in the 1985 *NRAO-VLA Workshop on Synthesis Mapping.*

5.3. Editing *uv* data

There are many programs which aid in the processing, display, and editing of *uv* data. A summary of this software is listed on your terminal by:

> HELP UVPR $^{C}_{R}$

and is also in § 14 of the *COOKBOOK*. In particular, there are facilities in ASCAL, CLIP, and CORER to flag *uv* data in *AIPS* based on deviations from specified norms. There is also a task, UVFLG, which allows flagging and unflagging by antenna-IF or by correlator. Type HELP ASCAL $^{C}_{R}$, HELP CLIP $^{C}_{R}$, or HELP UVFLG $^{C}_{R}$ for details. The task UVPLT plots various combinations of *uv* data — type HELP UVPLT $^{C}_{R}$ for details. The task UVFND is also recommended for printing out suspicious portions of the data base. Note that CLIP examines the data correlator by correlator, but UVFND normally converts the data to Stokes components (using the same criteria as UVMAP) before checking that the amplitudes are in range. To examine the correlators individually, use STOKES 'CORR' in UVFND.

CLIP is a very useful task for flagging data on the basis of its discrepancies from the visibility predicted by a set of CLEAN components. The task UVSUB can be used to subtract the Fourier transform of the CLEAN components from the visibility data. UVPLT enables the residuals to be displayed and CLIP can be used to flag abnormally high points. Cautious users may prefer to run UVFND to display such points before running the automatic CLIP task. Of course, after flagging, UVSUB should be run again to add the transform of the CLEAN components back into the data (using the input FACTOR = −1.0).

Another method for finding suspicious data is provided by the task FFT. Transform your image back into the *(u,v)* plane by running FFT and then display the results on the TV. Verbs like CURVALUE and IMPOS will help you find the *u* and *v* values for abnormally high cells. Then UVFND with OPCODE 'UVBX' will print the data surrounding these cells and UVFLG can be used to delete the bad data. This method is particularly effective on the residual images from CLEAN. (You can instruct APCLN and MX to put out a residual image by setting BMAJ less than 0.)

6. READING AND DISPLAYING IMAGES

In many cases, images will be generated outside of *AIPS* (*e.g.*, on the VLA "Pipeline" imaging system or on non-NRAO optical or radio telescopes). Such images are transported to *AIPS* by writing them on magnetic tape using the standard FITS format. A listing of the software associated with tape handling, reproduced in § 14, can be viewed on your terminal with HELP TAPU ⏎.

A listing of the software associated with TV displays can be found by listing the HELP files CURSOR, TVGEN, TVINTER, and TVCOLOR. These files are reproduced in § 14 of the *COOKBOOK*.

6.1. Loading an image from tape to disk

The task IMLOD is used to load images from tape to disk. It can read images in either the standard FITS format (which has one image per tape file) or in the CV-IBM image format (which may have more than one image per file). IMLOD assumes that the tape is positioned at the beginning of the image file. To move the tape forward or backward by a specified number of file marks, enter:

> INTAPE *n* ⏎ to specify the tape drive labeled *n*.

> NFILES *nf* ⏎ to specify the number of file marks to move the tape.

> AVFILE ⏎ to move the tape.

If $nf > 0$, AVFILE will advance the tape the specified number of file marks. If $nf = 0$, the tape is moved backward to the beginning of the current file. If $nf < 0$, the tape is moved backwards to the abs(nf) previous file. In all cases, the tape is left at the beginning of a file. With a CV-IBM format tape, the verb AVMAP may also be needed. Type HELP AVMAP ⏎ for details.

To make sure that the tape is positioned correctly, type:

> TPHEAD ⏎

This verb will display on the terminal information about the image header at which the tape is positioned. The tape position is not altered. Once the tape has been positioned at the desired image, enter:

> OUTNAME '*your-chosen-name*' ⏎ to specify the output disk file name in *AIPS*.

> GO IMLOD ⏎ to run the task.

The string *your-chosen-name* can be any (< 12-character) title that you want to use as the image name within *AIPS*. To cross-reference your bookkeeping on multiple computers, it is often useful to specify the name of the image file and its extension from a previous machine (*e.g.*, the VLA "Pipeline"). *AIPS* provides equivalents of the DEC-10 extensions, which it calls the image "Class". IMLOD allows you to specify this 6-character parameter, OUTCLASS '*abcdef*' ⏎, as well.

If OUTNAME is unspecified, it defaults to the "name" of the image read from the FITS header — either the name previously used in *AIPS* or the source name. If OUTCLASS is unspecified, it defaults to the Class previously used in *AIPS* or to a compound name (*e.g.*, IMAP, IBEM, QMAP, ICLN) which attempts to describe the image. You can change the *AIPS* image and class names later using RENAME (see § 4.2.1 of this *COOKBOOK*).

To load further images from the same tape in sequence:

> OUTNAME '*your-chosen-name*' ⏎ to set a new name.

> GO IMLOD ⏎ to run the task again.

To load *n* consecutive images from tape using the system default OUTNAME (the name from the FITS header):

> OUTNAME ' ' ⏎ to specify that the system default is wanted.

> TASK 'IMLOD' ⏎ to set task name for GO verb.

> FOR I = 1 TO *n* ; PRINT I ; WAIT ; GO ; END ↵ — to loop for *n* executions of IMLOD, waiting for the previous execution to finish before starting the next. Be careful with the value of *n* to avoid running off the end of the tape.

To rewind the tape before unloading:

> REWIND ↵

6.2. Loading an image from disk to the TV display

To select an image file for display you must assign a number of input parameters to the values carried by the chosen file. If you know the disk and catalog numbers of the file from an MCAT listing, you can initialize all of the name parameters together with the instruction:

> INDI *n* ; GETN *ctn* ↵ — where *n* and *ctn* select the required image.

Alternatively, you may specify the parameters individually.

The procedure TVALL incorporates most of the major display functions and will be described here. See § 14 for a complete listing of available display verbs.

You should use one of the following commands to specify the initial transfer function converting your image file intensities to display pixel intensities (the slopes and intercepts of the display transfer functions can be modified later, but you have an initial choice between linear, logarithmic, and negative linear displays and a choice of the range of intensities loaded):

> FUNC 'LN' ↵ — linear—this is the default.

> FUNC 'LG' ↵ — logarithmic.

> FUNC 'NE' ↵ — negative linear.

> FUNC 'NG' ↵ — inverse logarithmic.

> PIXRA *x1, x2* ↵ — to load only image intensities *x1* to *x2* in the units of the image. The default is to load the full range of intensities in the image.

Then

> TVALL ↵ — to load the selected image.

The image should appear on the TV screen in black-and-white. If you don't get what you expected, hit button D to get out of TVALL and review your inputs to the loading routine with:

> INP TVALL ↵

If *no* image was loaded, check which image channel (TVCHAN) is specified by your inputs. There are several image channels available on most TV display systems (*e.g.*, 4 on NRAO's I^2S systems). Try:

> TVON ↵ — to turn on channel TVCHAN.

If this still does not display your image, call the *AIPS* Manager for your site. If a labeled wedge does not appear on the display, you should also call your system Manager, as the display may need adjustment.

After your image has been displayed on the TV by TVALL, the trackball and its buttons (labeled A, B, C, and D) can be used to modify the display. Pressing button A will enable black-and-white and color-contour coding of the image intensities, successively. Adjustment of the cursor position on the TV (using the trackball) will vary the slope and intercept of the display transfer function. TVALL will superpose a calibrated horizontal wedge on the image. This should help you to choose the optimum cursor setting for the display.

The I^{2}S display can show images up to 512 by 512 in size. If the image is larger than about 424 pixels in the y-direction, portions of the labeling of the wedge (the units) will be omitted or superposed on top of the wedge (the tick numeric values). Other television systems will have comparable, but not necessarily identical, numeric limits. A useful technique for displaying large images is to load only alternate pixels. The command:

> TXINC 2 ; TYINC 2 C_R to load every other x and y pixel.

before TVALL would do this. Also use:

> TBLC = *n1, n2, n3, ...* C_R bottom left pixel to load.

> TTRC = *m1, m2, n3, ...* C_R top right pixel to load.

to limit the displayed field.

You will find that the TV display does not respond instantaneously to the changes in the transfer function requested by adjustment of the trackball, especially when the computer is being used heavily. It's best to turn the trackball slowly.

Pressing buttons B and C adjusts the zoom of the display: B to increase the magnification and C to decrease it. When these buttons are enabled, the cursor controls the position of the center of the zoomed field of view. Magnification factors of 1, 2, 4 and 8 are available on the I^{2}S.

Pressing button D will exit from the TVALL display modification routine, leaving your display parameters as they were when button D was pressed. Note that your terminal issues instructions when buttons are pressed, but that it is in the death-like grip of TVALL otherwise until you press button D to exit from it.

You may resume the image modification without reloading the image by typing:

> TVFIDDLE C_R

6.2.1. Alternative color coding

To obtain alternative color coding of the TV image type:

> TVPSEUDO C_R

and then follow the instructions listed on your terminal.

A rich zoo of color coding is available. First-time users should take a little time to experiment with the AIPS coloring options until they develop a feel for the effects of the trackball settings on the image appearance. The wedge displayed by TVALL will adjust to the alternative colorations selected with TVPSEUDO. Remember that the TV image does not respond instantaneously to trackball motions; move the ball slowly. Aficionados of certain Dutch color contouring schemes or of the old VLA *IMPS* "spectrum colors" will find these options available in TVPSEUDO using trackball button C.

6.2.2. Transfer functions

To fiddle the (black-and-white) transfer function of the display without reusing TVALL type:

> TVTRAN C_R

and then follow the instructions listed on the terminal.

When using TVTRAN, notice that the TV display does not respond instantaneously to modifications you make to the transfer function using the trackball. It's best to turn the trackball slowly. Also, the transfer function is displayed in the top right corner of the image only when the zoom is used at lowest magnification.

6.3. Reading out image data using the trackball/cursor

There are several facilities for reading out intensity and position information from displayed images using the TV cursor and the trackball:

> IMPOS CR_R	reads out the two coordinate values (*e.g.*, RA and Dec) from the cursor position when any trackball button is depressed.
> CURV CR_R	continuously reads out pixel coordinates and the pixel intensity in user-recognizable units at the position selected by the TV cursor. The pixel parameters are displayed in the upper left corner of the image.
> TVSTAT CR_R	determines the mean, rms, and peak of a set of polygonal regions in the image currently displayed on the TV. The regions are selected with the TV cursor. Type EXPLAIN TVSTAT CR_R for details.
> TVWIN CR_R	reads pixel coordinates from the next two cursor positions at which a trackball button is depressed. The TV graphics shows the current shape and position of the window. Button A allows you to switch to (re)setting the other corner while the other buttons exit after both corners have been set. TVWIN uses the pixel coordinates to set up the bottom left (BLC) and top right (TRC) corners of an image subsection, *e.g.*, for input to the contouring programs CNTR and PCNTR, to the mean/rms calculator IMEAN, and to many other tasks.
> SETXWIN (*dx,dy*) CR_R	reads pixel coordinate of the center of a *dx*-pixel by *dy*-pixel window and sets the adverbs BLC and TRC.
> TVBOX CR_R	works like TVWIN above except that it is used to set up pixel coordinates to define CLEANing areas for the *AIPS* CLEAN routine (APCLN). The adverb NBOXES must be entered before the TVBOX command and is the number of rectangular CLEANing boxes to be set with the TV cursor, trackball, and buttons.
> REBOX CR_R	allows revision using the TV of the CLEANing areas set previously with TVBOX.
> TVSLICE CR_R	works like TVWIN above to set BLC and TRC. Instead of a rectangle however, the display shows a diagonal line which is useful for setting the ends of slices.

Additional interactive TV functions are available. Type HELP TVINTER CR_R and HELP CURSOR CR_R for additional information. These help files are also reproduced in § 14.

7. TAKING HARD COPY OF IMAGES

A listing of available *AIPS* tasks for two-dimensional displays and hard copies of images can be obtained at your terminal by typing HELP PL2D. The listing is also given in § 14 of the *COOKBOOK*. Representative samples of the hard-copy displays are given in § 7.9 at the end of this section.

7.1. Ordinary contour plots (CNTR)

In addition to the usual image selection parameters, you may specify the following parameters to the contouring task CNTR:

> TASK 'CNTR' ; INP ⏎	to tell you what you may specify.

Examples of the command syntax (*not* necessarily a recommended combination):

> BLC 250, 230 ⏎	to set the bottom left corner of plot at 250,230.
> TRC 300, 330 ⏎	to set the top right corner of plot at 300,330 (in pixels with 1,1 at extreme bottom left of the image).
> CLEV 0 ; PLEV 1 ⏎	to get contour levels at 1% of the peak image value.
> PLEV 0 ; CLEV .003 ⏎	to get contour levels at 3 mJy.
> LEVS -1, 1, 2, 4, 6 ⏎	to get actual contours at -1, 1, 2, 4, and 6 times the basic level set by PLEV or CLEV. LEVS need not be integers, but very fine subdivisions cannot be represented accurately on the plot.

N.B., if you request more than one negative level with the LEVS input, you *must* use commas between the negative levels. Otherwise the minus sign(s) will be treated as subtraction symbols and the desired levels will be combined into a single negative level by the AIPS language processor. BLC and TRC can be initialized conveniently from the TV display using the cursor with the TVWIN instruction (see § 6.3). Check:

> INP ⏎	to review what you have specified.

When you're satisfied with the inputs, then:

> GO ⏎

generates a plot file as an extension to your image file, with the parameters you have just specified. Watch the *AIPS* monitor (which, on some systems, is your terminal) to see the progress of this task. If the number of blocks written in the plot file is over 200, the contour plot will be messy (unless the field is also large).

To look at the results, enter:

> GO TKPL ⏎	to display the plot file produced by CNTR on the Tektronix 4012 (graphics) terminal. The HARD COPY switch on the terminal can be used to send the TEK image to the copier (often a printer/plotter).
> ABORT TKPL ⏎	to stop an overly messy plot on the 4012.
> GO TVPL ⏎	to display the plot file on a TV graphics plane.
> GO PRTPL ⏎	to display the plot file to the printer/plotter.
> GO QMSPL ⏎	to display the plot file on a QMS laser printer.

Systems having other types of display devices may have other tasks to put the results on those devices. The above tasks will select the plot file with the largest version number if INVER 0. This is generally what is desired.

7.2. Contour plots with polarization vectors (PCNTR)

PCNTR plots polarization vectors on top of contours. You must first make a polarized-intensity image and a polarization position-angle image from the Q and U images (see §8.1.1 below). The inputs to PCNTR can be reviewed by:

> TASK 'PCNTR' ; INP ⏎

It is a long list, so you will need to use the page-scroll keys (see §2.2) to review the start of these inputs. Use:

> INDI 0 ; MCAT ⏎	to review the contents of your image catalog for image names and classes.
> INDI *n1* ; GETN *ctn1* ⏎	where *n1* and *ctn1* select the disk and catalog numbers of the image to be contoured.
> IN2DI *n2* ; GET2N *ctn2* ⏎	where *n2* and *ctn2* select the polarized intensity image.
> IN3DI *n3* ; GET3N *ctn3* ⏎	where *n3* and *ctn3* select the position-angle image.
> PCUT *nn* ⏎	to blank out vectors less than *nn* in the units of polarized intensity.
> ICUT *mm* ⏎	to blank out vectors at pixels where the total (image 1) intensity is less than *mm* in the units of image 1.
> FACTOR *xx* ⏎	to set the length of a vector of 1 (in units of total polarization) to *xx* cell widths.
> INP ⏎	to review your inputs and remind you of others. Most are similar to those in CNTR and sensibly defaulted.
> GO PCNTR ⏎	to generate the plot file, which can then be routed to output devices via TKPL, TVPL, QMSPL or PRTPL.

7.3. Plotting two images (GREYS)

The combination of laser printers and their *AIPS* driver task QMSPL makes the plots produced by GREYS particularly useful. GREYS creates a plot file of the grey-scale intensities in the first input image plane and, optionally, a contour representation of a second input image plane. A sample set of inputs could be:

> TASK 'GREYS' ; INP ⏎	to review the inputs.
> INDISK *n1* ; GETN *ctn1* ⏎	to select the grey-scale image.
> BLC 250 , 250 , 3 ⏎	to select the lower left corner and the plane in the first image.
> TRC 320 , 310 , 12 ⏎	to select the upper right corner in the first image and, with TRC(3), the plane in the second image.
> DOCONT TRUE ⏎	to specify that contours are to be drawn.
> IN2D *n2* ; GET2N *ctn2* ⏎	to select the contour image.
> PLEV 0 ; CLEV 0.005 ⏎	to select 5 mJy/beam contour increments.
> LEVS -3 , -1 , 1 3 10 30 100 ⏎	to plot contours at -15, -5, 5, 15, 50, 150, and 500 mJy/beam.
> DOWEDGE 2 ⏎	to plot a grey-scale step-wedge along the right-hand edge; 1 for along the bottom and 0 for no wedge.
> INP ⏎	to review the inputs.
> GO ⏎	to make the plot file.

When GREYS has finished, run QMSPL to view the plot file. Note that QMSPL has a variety of options which control the plotting and scaling of the grey-scale images without having to rerun GREYS.

7.4. Deleting unwanted plot files (EXTDEST)

Plot files generated by CNTR, PCNTR, GREYS and other plot tasks are archived in your disk catalog as PL extensions to the image file from which they were derived. Running CNTR a second time or some other plot task does not overwrite the previous plot file, but makes another with a higher "version" number.

You can review the parameters of the plot files associated with a given image file by typing:

> INDI *n* ; GETN *ctn* C_R	where *n* and *ctn* select the disk and catalog number of the desired image.
> INEXT 'PL' ; EXTLIST C_R	to select a listing of plot file contents.

Plot files (and other "extension files") are automatically deleted when an image is deleted by ZAP. However, large plot files should be deleted as soon as they are no longer needed:

> INP EXTDEST C_R	to review the inputs required.
> INDI *n* ; GETN *ctn* C_R	where *n* and *ctn* select the disk and catalog numbers of the image file.
> INEXT 'PL' ; INVERS *m* C_R	to set the type to PL (plot) and the version number to be deleted to *m*. $m = -1$ means all and $m = 0$ means the most recent (highest numbered).
> EXTDEST C_R	to do the deletion.
> INVERS 0 C_R	to reset the version number to its default — usually advisable.

7.5. SLICE files (profile plots)

You can generate a one-dimensional slice (profile) through any two-dimensional plane of an image file using the *AIPS* task SLICE. The output file is appended to the image file as an SL extension file. Slices are computed along lines in the two-dimensional image joining any valid pair of points selected by BLC and TRC. The set of software dealing with slice file analysis and display can be obtained on your terminal by typing HELP SL1D. The list is also given in § 14.

To generate a slice:

> TASK 'SLICE' ; INP C_R	reviews the inputs to SLICE.

Use INDISK and GETNAME to select the input image. The beginning (BLC) and ending (TRC) points for the slice can be specified conveniently using the TV cursor if the image to be sliced is first displayed on the TV with TVLOD or TVALL. To set these points with the TV, type:

> TVSLICE C_R

then set the TV cursor to the desired beginning point for the slice, press any trackball button, and repeat for the ending point for the slice. Note that, for slices, BLC need not be below or to the left of TRC. Finally:

> GO C_R	to generate the slice file.

When SLICE has terminated (watch the *AIPS* monitor), the file may be plotted on the Tektronix 4012 (graphics) terminal using the verb TKSLICE:

> INP TKSLICE C_R	to review what you can specify.
> INEXT 'SL' ; EXTL C_R	to find the intensity range and number of points in the interpolated slice.

The default scales will plot all slice points on a vertical scale from the image minimum to the image maximum. You can alter the part of the slice that is plotted and the vertical scale by specifying, for example:

> BDROP 100 ; EDROP 225 $\subset_R$	to drop 100 points from the beginning and 225 points from the end of the plotted portion of the slice.
> PIXRANGE -0.001 0.004 $\subset_R$	to set the range of the vertical axis to be −1 to 4 mJy/beam.

Then:

> TKSLICE $\subset_R$	to plot the slice on the TEK screen.

Note: several slices may be put on one TEK plot. Use TKASLICE $\subset_R$ for the additional ones.

Slice files may be converted into plot files by:

> GO SL2PL $\subset_R$

The resulting plot files may then be output by:

> GO QMSPL $\subset_R$	to display the plot file on the QMS laser printer.
> GO PRTPL $\subset_R$	to display the plot file on the printer/plotter.

to obtain a better-quality plot than can be obtained with the HARD COPY button on the graphics terminal. In addition:

> GO TKPL $\subset_R$	to display the plot file on the graphics terminal.
> GO TVPL $\subset_R$	to display the plot file on a TV graphics plane.

A one-dimensional fit of up to four Gaussian components to slice data can be calculated using the task SLFIT (see § 8.3.3). Relevant displays of the data, the fit, and the residuals are also available both as verbs (like TKSLICE) and as options in the task SL2PL. See HELP SL1D or the listing in § 14 for complete lists.

Slice files are archived in your disk catalog as SL extensions to the image file from which they were derived. Running SLICE again with new parameters does not overwrite the slice file, but makes another with a higher "version" number. To review and/or delete slice files, follow the instructions for EXTLIST and EXTDEST of plot files in § 7.4 above, but use INEXT 'SL' $\subset_R$ in place of INEXT 'PL' $\subset_R$.

7.6. Other one-dimensional plots

The "slices" described above may, of course, be specified to be single rows of your image (*i.e.*, BLC(2) = TRC(2)). There are several tasks, however, which plot rows more directly. The simplest of these is PLROW which makes a plot file of all selected rows in an image plane. Each row is plotted as a slice offset a bit from the previous row. Low intensities which are "obscured" by foreground (*i.e.*, lower row number) bright features are blanked to keep the plot readable. Example inputs would be:

> TASK 'PLROW' ; INP $\subset_R$	to review the inputs.
> INDISK *n* ; GETN *ctn* $\subset_R$	to select the image on disk *n* catalog slot *ctn*.
> BLC 100 ; TRC 300 $\subset_R$	to select the subimage from (100,100) to (300,300).
> YINC 3 $\subset_R$	to plot only every 3rd row.
> PIXRANGE -0.001 0.050 $\subset_R$	to clip intensities outside the range −1 to 50 mJy.
> OFFSET 0.002 $\subset_R$	to set the intensity scaling such that 2 mJy separates rows of equal intensity.
> INP $\subset_R$	to check the inputs.
> GO $\subset_R$	to run PLROW.
> GO QMSPL $\subset_R$	to display the plot file on the laser printer after PLROW has finished.

The plot files produced by PLROW are a simple, special case of those produced by PROFL. PROFL makes a plot file of a "wire-mesh" representation of an image plane complete with user-controlled viewing angles and correct perspective. Enter EXPLAIN PROFL CR for a full description. Both of these tasks are especially useful where the signal-to-noise ratio is high.

Two other row-plotting tasks, PLCUB and XPLOT, are designed primarily for spectral-line and other data "cubes" (see § 9). PLCUB makes one or more plot files showing the intensities in each selected row. The row subplots are positioned in a matrix in the coordinates of the 2nd and 3rd axes of the cube. XPLOT, unlike the tasks described previously, does not create plot files. Instead, it is an interactive task which plots selected rows on the graphics terminal. Rows are selected not only with the usual inputs BLC, TRC, YINC, and ZINC, but also with inputs designed to limit the rows displayed to those with peak intensities falling within the flux range of interest.

7.7. Dicomed copies of images

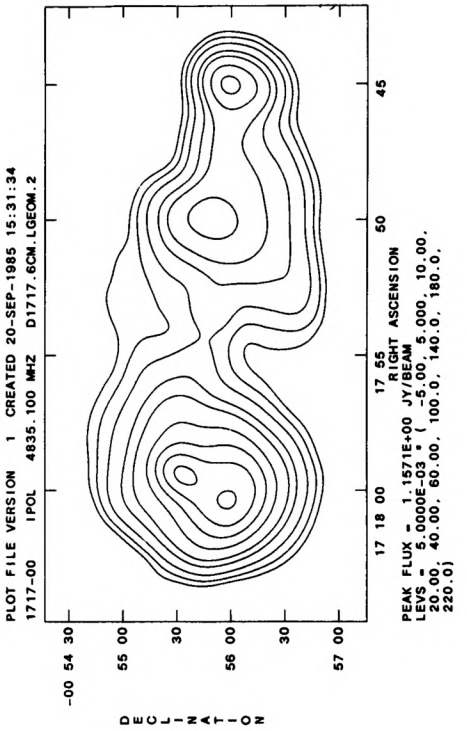
At the time of this edition of the *COOKBOOK*, there is no *AIPS*-standard method for recording images on the Dicomed film recorders. However, ad hoc methods exist both at the VLA site and in Charlottesville for doing this. At neither site are the image recorders directly connected to an *AIPS* computer by *AIPS* software. Thus the images must be transferred to a non-*AIPS* system. At the VLA site, Decnet is used for the transfer (see Appendix Z, § Z.1.9). In Charlottesville, a FITS tape is used (see Appendix Z, § Z.2.9).

7.8. Banana poundcake

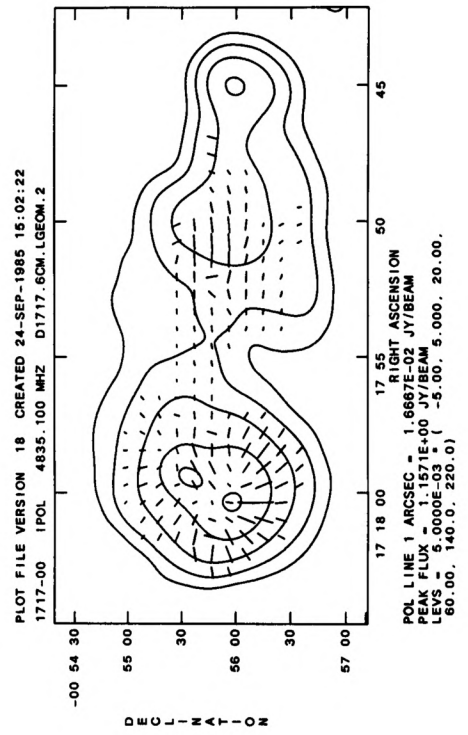
1. Mix in large bowl until blended:
   - $1\frac{1}{3}$ cups mashed **bananas** (4 medium)
   - 1 pkg. ($18\frac{1}{2}$ oz.) **yellow cake mix**
   - 1 pkg. ($3\frac{3}{4}$ oz.) instant **vanilla pudding mix**
   - $\frac{1}{3}$ cup **salad oil**
   - $\frac{1}{2}$ cup **water**
   - $\frac{1}{2}$ teaspoon **cinnamon**
   - $\frac{1}{2}$ teaspoon **nutmeg**
   - 4 **eggs** at room temperature
2. Beat at medium speed for 4 minutes.
3. Turn batter into greased and lightly floured 10-inch tube pan.
4. Bake in 350° oven for 1 hour or until cake tester inserted in cake comes out clean.
5. Cool in pan 10 minutes, then turn out onto rack and cool completely.
6. If desired, dust with confectioners sugar before serving.

Thanks to the United Fresh Fruit and Vegetable Association.

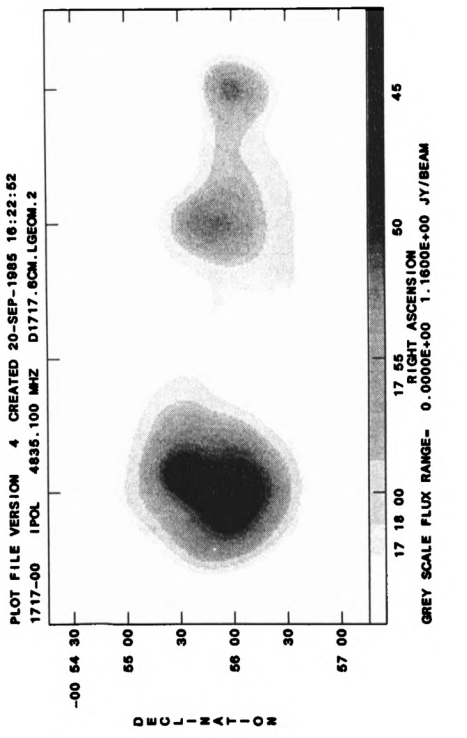
7.9. Sample displays



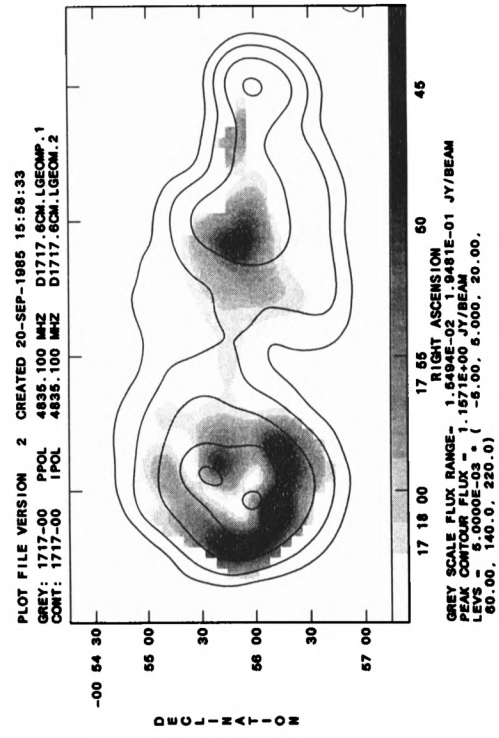
CNTR



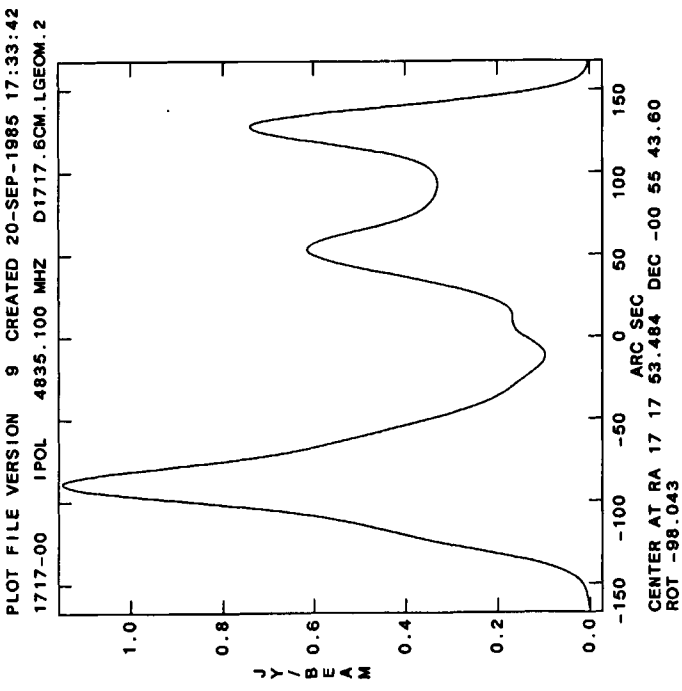
PCNTR



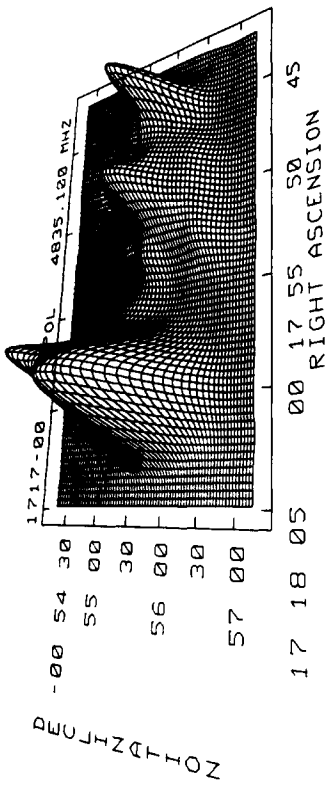
GREYS



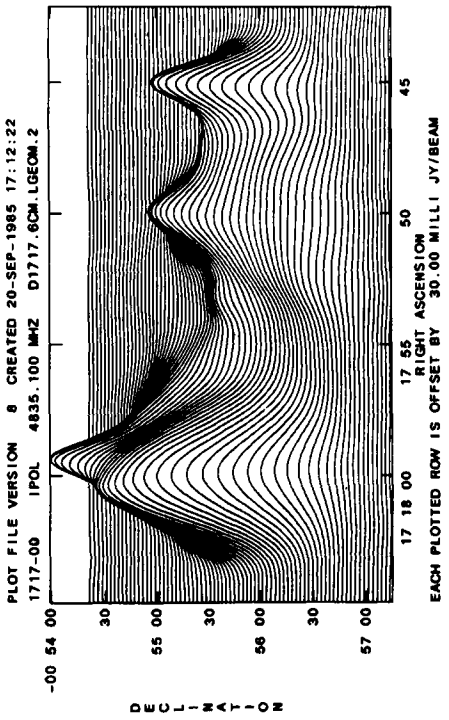
GREYS with contours



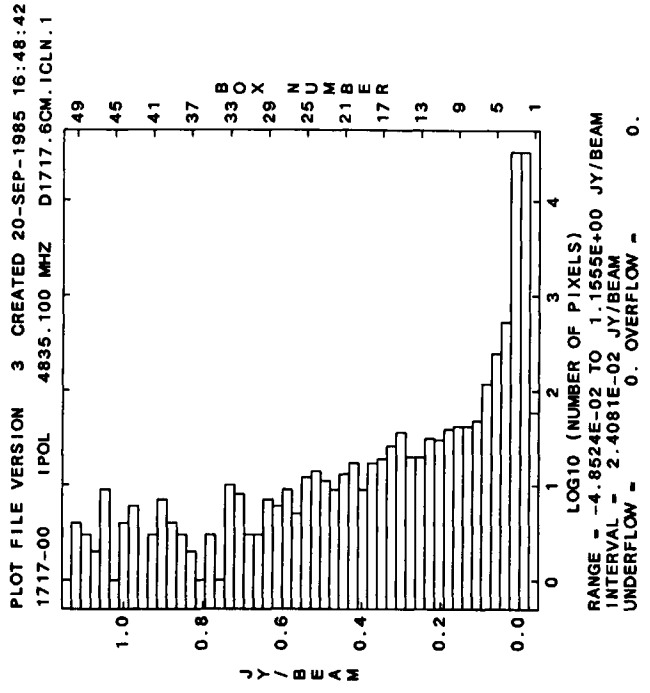
SL2PL



PROFL



PLROW



IMEAN

8. ANALYZING IMAGES

In order to obtain useful astronomical information from the data, software exists for the analysis of images, combining of images, estimating errors, *et al.* Only a few of the programs are described here; the others should be self-explanatory using the HELP and INPUTS files for the tasks listed in § 14. A complete list of software in *AIPS* for the analysis of images may also be obtained at your terminal by typing HELP ANALYSIS CR_R.

8.1. Combining images (COMB)

The task COMB is a general purpose program for combining two images, pixel by pixel, to obtain a third image. Many options are available and, as a first example, we illustrate inputs to derive the polarization intensity and angle from the Q and U Stokes parameter images.

8.1.1. Polarized intensity and position angle images

To compute a polarized intensity image, enter:

> TASK 'COMB' ; INP CR_R	to review the required inputs.
> INDI 0 ; MCAT CR_R	to help you find the catalog numbers of the Q and U images that you want to combine.
> INDI *n1* ; GETN *ctn1* CR_R	where *n1* and *ctn1* select the Q image.
> IN2D *n2* ; GET2N *ctn2* CR_R	where *n2* and *ctn2* select the U image.
> OUTN '*xxxxx*' CR_R	where *xxxxx* is your choice of outfile name for the polarized intensity image.
> OUTC '*ccc*' CR_R	where *ccc* is your choice of classname for the polarized intensity image, *e.g.*, PCLN.
> OPCODE 'POLI' CR_R	to select the $\sqrt{Q^2 + U^2}$ algorithm.
> GO CR_R	to compute the polarized intensity image.

The *AIPS* monitor should note TASK COMB BEGINS followed by a listing of the POL. INTENSITY algorithm. While it is running, you can prepare the inputs to make a polarization position angle image. Type:

> OUTN '*yyyyy*' CR_R	where *yyyyy* is your choice of outfile name for the polarization angle image.
> OUTC '*ddd*' CR_R	where *ddd* is your choice of class name for the polarization angle image, *e.g.*, PSIMAP, CHICLN.
> OPCODE 'POLA' CR_R	to select the $\frac{1}{2}\tan^{-1}\left(\frac{U}{Q}\right)$ algorithm.

Once COMB has finished, enter:

> GO CR_R	to compute the polarization angle image.

The *AIPS* monitor should note again TASK COMB BEGINS followed by a listing of the POL. ANGLE algorithm. Once both COMB tasks have terminated with COMB: ENDS SUCCESSFULLY messages on the message monitor, you should find the requested images in your catalog:

> MCAT CR_R	to list the images in your catalog.

8.1.2. Other image combination options

COMB may also be used to add or subtract images, to rescale them, and to compute spectral indices, optical depths, *et al.* Type:

> HELP COMB ⏎ to review the available options.

At the time of writing the options include:

'SUM '	Addition	$a_1 M_1 + a_2 M_2 + a_3$
'MULT'	Multiplication	$a_1 M_1 M_2 + a_2$
'DIV '	Division	$a_1 M_1 / M_2 + a_2$
'SPIX'	Spectral Index	$a_1 \ln(M_1/M_2) \,/\, \ln(\nu_1/\nu_2) + a_2$
'OPTD'	Opacity	$a_1 \ln(a_3 M_1/M_2 + a_4) + a_2$
'POLI'	RMS sum	$a_1\sqrt{M_1^2 + M_2^2} + a_2$
'POLA'	Arctangent	$a_1 \tan^{-1}(M_2/M_1) + a_2$
'REAL'	Real part	$a_1 M_1 \cos(a_2 M_2) + a_3$
'IMAG'	Imaginary part	$a_1 M_1 \sin(a_2 M_2) + a_3$
'CLIP'	Clipping	M_1 *except blanked where* $a_1 > M_2 > a_2$ *or* $a_2 > a_1 > M_2$ *or* $M_2 > a_2 > a_1$

where the a_i are user-adjustable parameters and M_1 and M_2 are the images selected by INNAME, *et al.* and by IN2NAME, *et al.*, respectively.

8.1.3. Considerations in image combination

For some applications of COMB, undefined pixel values may occur. For example, if the spectral index is being calculated and the intensity level on either image is negative, the index is undefined. In this case, the pixel value is given a number which is interpreted as undefined or "blanked." Blanking also arises naturally in operations of division, opacity, polarization angle, and clipping and, of course, the input images may themselves be blanked. In addition, the output image can be blanked whenever either M_1 <APARM(9) or M_2 <APARM(10). Using APARM(8) = 1 ⏎, the user may specify that all undefined pixels are to be assigned an apparently valid value of zero, rather than the "magic" undefined-pixel value.

A version of COMB, called CORMS, offers additional options. It can produce images which are estimates of the noise in the combination, if desired, rather than the combination itself. It offers the blanking options of COMB, but can also, if desired, blank on the basis of noise or signal-to-noise ratio. CORMS does not perform the 'CLIP', 'REAL' or 'IMAG' operations of COMB, but it does offer an additional rotation measure, 'RM', operation. Type HELP CORMS ⏎ for additional information.

When combining two or more images, COMB, PCNTR, *et al.* must decide which pixels in the 2nd image go with which pixels in the 1st image. The user input parameter DOALIGN controls this process. A value of 1 requires the two headers to be correct and sufficiently similar that an alignment by coordinate value is possible. A value of −2 tells the programs to ignore the headers and align by pixel number. Enter HELP DOALIGN ⏎ for details and intermediate options. In some cases, the images may have been created on different grids which are correctly described in the headers. The observations, for example, could have differed in the phase reference position or projective geometry used or the imaging could have been done with different axis increments. Such images should *not* be combined directly. Instead, the header of one should be used as a template for re-gridding the other. Task HGEOM provides this service with up to 7th-order polynomial interpolation. See § 8.4.1 and type EXPLAIN HGEOM ⏎ for more information.

8.2. Image statistics and flux integration (IMEAN, IMSTAT, TVSTAT, BLSUM)

The task IMEAN is used to determine the statistics of the image over a specified rectangular area. It derives the minimum and maximum value and location, the rms, the average value and, if the image has been CLEANed, an approximate flux density within the area. A typical run might be:

> TASK 'IMEAN' ; INP ⏎	to list the input parameters.
> INDI *n* ; GETN *ctn* ⏎	where *n* and *ctn* select the disk and catalog numbers of the relevant image file.
> BLC *n1, n2* ; TRC *m1, m2* ⏎	to set the window from (*n1,n2*) to (*m1,m2*) — or use TVWIN with the cursor on the TV.
> DOHIST TRUE ⏎	to make a plot file of the pixel histogram.
> PIXRANGE *x1, x2* ⏎	to set the range of the histogram from *x1* to *x2*.
> NBOXES *n* ⏎	to set the number of boxes in the histogram.
> GO ⏎	to run the task.

The statistics will appear on the *AIPS* monitor. For a hard copy type:

> PRTASK 'IMEAN' ; PRTMSG ⏎	with PRIO $\leq$ 5.

To see the histogram of the intensities, an example of which is shown in §7.9, type one of:

> GO TKPL ⏎	to display histogram on the Tektronix.
> GO PRTPL ⏎	to display histogram on the printer/plotter.
> GO QMSPL ⏎	to display histogram on the laser printer.

The verbs TVSTAT and IMSTAT provide similar functions to IMEAN without the histogram option. Both return their results as AIPS parameters PIXAVG (mean), PIXSTD (rms), PIXVAL (maximum), and PIXXY (pixel position of the maximum). IMSTAT uses the same file name, BLC, and TRC parameters as IMEAN. TVSTAT, however, works on the image plane currently displayed on the TV and is not limited to a single rectangular area. Instead, the TV cursor is used to mark one or more polygonal regions over which the function is to be performed. Type EXPLAIN TVSTAT ⏎ for a description of its operation.

The interactive task BLSUM employs a method similar to that of TVSTAT. The TV cursor is used to mark a region of interest in a "blotch" image. Then BLSUM finds the flux in that region not only in the blotch image but also in each plane (separately) of a second image. More than one region of interest may be done in any given execution of the task. In spectral-line problems, the blotch image is often the continuum or the line sum while the second image is the full "cube" (see §9.4) in almost any transposition. However, numerous continuum applications also exist (*e.g.*, polarization, comparison across frequency). Type EXPLAIN BLSUM ⏎ for a description of the operation.

8.3. Fitting of images

There are three programs which estimate the position and intensity of a component on a two-dimensional image. The simple and fast method is the verb MAXFIT. This fits a two-dimensional parabola to the maximum within a few pixels of an image position, and gives the peak and its position. The tasks IMFIT and JMFIT are similar and fit an image subsection with up to four Gaussian components with error estimates. In one dimension, the task SLFIT fits Gaussian components to slice data and the task XGAUS fits Gaussian components to each row of an image.

8.3.1. Parabolic fit to maximum (MAXFIT)

MAXFIT's speed makes it useful for simple regions. Type:

> HELP MAXFIT ⏎ to get a good explanation of the algorithm.

These should be self-explanatory. The IMSIZE parameter can be important in crowded fields. MAXFIT can be used conveniently by first displaying the image on the TV and then typing:

> IMXY ; MAXFIT ⏎

First the cursor will appear on the TV. Move it close to a maximum and hit any of the buttons behind the trackball. The fit will then appear on your terminal.

8.3.2. Two-dimensional Gaussian fitting (IMFIT)

A more sophisticated least-squares fit of an image is obtained with IMFIT, which fits an image with up to four Gaussian components and attempts to derive error estimates. A linear or curved, two-dimensional "baseline" may also be fitted. A sample set-up is as follows:

> TASK 'IMFIT' ; INP ⏎	to list the input parameters.
> INDI *n* ; GETN *ctn* ⏎	where *n* and *ctn* select the disk and catalog numbers of the required image.
> BLC *n1, n2* ; TRC *m1, m2* ⏎	to set the area to be fitted as (*n1,n2*) to (*m1,m2*) — or use TVWIN with the cursor on the TV.
> NGAUSS 2 ⏎	to set the number of components to be fitted to 2.
> CTYPE 1, 1 ⏎	to have both components be Gaussians.
> GMAX 0.34 0.20 ⏎	to give estimates of peak intensity in Jy.
> GPOS 200, 100, 210, 110 ⏎	to give estimates of the pixel locations of each component.
> GWID 6 4 20 6 4 20 ⏎	to give estimates of component sizes in pixels. In this case, each component has a FWHP of 6 by 4 pixels with the major axis at position angle 20 deg.
> DOWID FALSE ⏎	to hold all of the widths constant in the fitting process (if required).
> INP ⏎	to review inputs.
> GO ⏎	to run the task.

To save execution time, include as small an area as possible in the fit. In some cases, it is useful to hold some of the parameters constant, particularly when fitting a complex clump of emission with several components. The parameters can interact. Error estimates are given for each component. IMFIT will sometimes fail to converge in complicated regions. When this happens, you might try using the task JMFIT, which is very similar in function, but uses a different mathematical method to minimize the rms and to estimate the errors. Comparison of the results of IMFIT and JMFIT will sometimes be instructive. It is wise to treat the results of MAXFIT, IMFIT and JMFIT with considerable caution, particularly the estimates of the errors in the component widths after deconvolution.

Use RUN INPFIT ⏎ (see § 13.6) to obtain a procedure which will help to supply input parameters to IMFIT. This RUN file loads a procedure called INPFIT into AIPS. To invoke it, load the image which you want to fit onto the TV with TVALL and type:

> INPFIT (3) ⏎ to specify three components.

The procedure will prompt you to set the desired subimage window with the TV cursor (it uses verb TVWINDOW) and then to point the TV cursor at the peaks of each of the Gaussians, hitting any button when the cursor is correctly placed. The inputs GMAX, GPOS, BLC, and TRC are set in this way.

8.3.3. Gaussian fits to slices (SLFIT)

The task SLFIT fits Gaussian components to one-dimensional data in slice files (see § 7.5). Assuming that the usual GETNAME step has been done, a typical session would go like:

> INEXT 'SL'; EXTL $\crarr_R$	to list the parameters of the slice files.
> INVERS m $\crarr_R$	to select the m^{th} file for analysis.
> TKSLICE $\crarr_R$	to plot the slice on the graphics terminal.
> EDROP 840 ; BDROP 700 $\crarr_R$	to select a subsection to fit.
> TKSLICE $\crarr_R$	to replot just the subsection.
> NGAUS 2 $\crarr_R$	to fit 2 Gaussians.
> TKSET $\crarr_R$	

This verb will prompt you to POSITION CURSOR AT CENTER & HEIGHT OF GAUSSIAN COMP 1 and will turn on the Tektronix crosshairs. Move the crosshairs with the thumbwheels to the requested position and press any key *except* $\crarr_R$ on the Tektronix 4012 keyboard. The terminal then requests you to POSITION CURSOR AT HALFWIDTH OF GAUSSIAN COMP 1 and turns the crosshairs back on. Move the crosshairs until they are at the half-intensity point of the component and press any key except $\crarr_R$. Continue until all components have been entered. Then type:

> TKAGUESS $\crarr_R$	to plot the guess on top of the slice plot.

If everything looks ok, then:

> GO SLFIT $\crarr_R$	to run the task.

When the task gets an answer, the solution will be displayed on the *AIPS* monitor, recorded in the message file, and recorded in the slice file itself. To get a hard copy of the results:

> PRTASK 'SLFIT' ; PRTMSG $\crarr_R$	to print the message file.

and, to display the results on the graphics terminal, enter:

> TKSLICE $\crarr_R$	to replot the slice.
> TKAMODEL $\crarr_R$	to add the model results to the plot.
> TKARESID $\crarr_R$	to add the residuals (data – model) to the plot.

To get a higher quality plot of the results, an example of which is shown in § 7.9, type:

> DORES TRUE ; DOMOD TRUE $\crarr_R$	to request the model and the residuals.
> DOSLICE FALSE $\crarr_R$	to leave the slice data out of the plot.
> TASK 'SL2PL' ; GO ; WAIT $\crarr_R$	to make a plot file and wait for it to be complete.
> GO PRTPL $\crarr_R$	to display it on the printer/plotter.

8.3.4. Other one-dimensional Gaussian fits (XGAUS)

XGAUS is an interactive task which can fit up to four Gaussians and a linear baseline to each row of an image. It writes its results as a set of $n-1$ dimensional image files. Although XGAUS was designed for use primarily on transposed spectral-line cubes (see § 9.10), it has a wide variety of other applications. The interaction is optional and uses both the terminal and the graphics screen (Tektronix 4012 or TEK). The data, initial guess, model fit, and the residual for each row may be plotted on the TEK screen. If the number of Gaussians being fit is larger than one, you may choose for each row to enter a revised initial guess using the TEK crosshairs. This process is similar to that of TKSET described above (§ 8.3.3). This task has too many options to do them justice here. Enter EXPLAIN XGAUS $\crarr_R$ for details.

8.4. Image analysis

Image analysis is a very broad subject covering essentially all that *AIPS* does or would like to do plus specialized programs designed to analyze a user's particular image in the light of his favorite astrophysical theories. *AIPS* provides some general programs to perform geometric conversions, image filtering or enhancement, and model fitting and subtraction. These are the subjects of the following sections. Specialized programs for spectral-line and VLBI data reduction are described in §§ 9 and 10, respectively.

8.4.1. Geometric conversions

The units of the geometry of an image are described in its header by the coordinate reference values, reference pixels, axis increments, axis dimensions, and axis types. The types of coordinates (celestial, galactic, etc.) and the type of tangent-plane projection (SIN from the VLA, TAN from optical telescopes, ARC from Schmidt telescopes, NCP from the WSRT) are specified in the *AIPS* headers by character strings. See *AIPS* Memo No. 27 for details of these projections. A "geometric conversion" is an alteration of one or more of these geometry parameters while maintaining the correctness of both the header and the image data. The *AIPS* tasks which do this interpolate the data from the pixel positions in the input image to the desired pixel positions in the output image. Most of them require very large internal buffers and hence, are available only on virtual-memory computer systems.

The simplest geometric conversion is a regridding of the data with new axis increments and dimensions with no change in the type of projection or coordinates. Two tasks, GEOM and LGEOM, perform this basic function and also allow rotation of the image. One use of these tasks is to obtain smoother displays by regridding a subimage on a finer grid. To rotate and blow up the inner portion of a 512^2 image, enter:

> TASK 'LGEOM' ; INP $\circ_R$	to review the inputs.
> INDISK *n* ; GETN *ctn* $\circ_R$	to select the image.
> BLC 150 ; TRC 350 $\circ_R$	to select only the inner portion of the image area.
> IMSIZE 800 $\circ_R$	to get an 800^2 output image. This will allow the subimage to be blown up by a factor of 3 and rotated without having the corners "falling" off the edges of the output image.
> APARM 0 $\circ_R$	to reset all parameters to defaults.
> APARM(3) = 30 $\circ_R$	to rotate the image 30° CCW (East from North usually).
> APARM(4) = 3 $\circ_R$	to blow up the scale (axis increments) by a factor of 3.
> APARM(6) = 1 $\circ_R$	to use cubic polynomial interpolation.
> INP $\circ_R$	to check the inputs.
> GO $\circ_R$	to run the program.

LGEOM allows shifts of the image center, an additional scaling of the y axis relative to the x axis, and polynomial interpolations of up to 7^{th} order. Type EXPLAIN LGEOM $\circ_R$ for more information and advice. GEOM is a small-buffer version of LGEOM. As a result, it works on non-virtual-memory computers, but is very limited in the amount of rotation it can do on larger images.

A much more general geometric transformation is performed by HGEOM, which converts one image into the geometry of a second image. The type of projection, the axis increments, the rotation, and the coordinate reference values and locations of one image are converted to those of a second image. HGEOM should be used before comparing images (with COMB, GREYS, PCNTR, BLANK, TVBLINK, etc.) made with different geometries, *i.e.*, radio and optical images in different types of projection or VLA images taken with different phase reference positions. Use EXPLAIN HGEOM $\circ_R$ to obtain the details and useful advice.

A potentially very powerful transformation is performed by PGEOM. In its basic mode, it converts between rectangular and polar coordinates. An example of this operation is illustrated in § 8.4.4. However, PGEOM can also "de-project" elliptical objects to correct for their inclination and "unwrap" spiral objects. Type EXPLAIN PGEOM ↵ for information.

8.4.2. Filtering

For our purposes here, we can define "filtering" as applying an operator to an image in order to enhance some aspects of the image. The operators can be linear or nonlinear and do, in general, destroy some of the information content of the image. As a result, users should be cautious about summing fluxes or fitting models in filtered images. (Technically, these remarks can also be made about CLEAN and self-calibration.) However, filtered images may bring out important aspects of the data and often make excellent, if unfamiliar-looking, displays of particular aspects.

NINER produces an image by applying an operator to each cell of an image and its 8 nearest cells. The task offers three nonlinear operators which enhance edges (regions of high gradient in any direction). It also offers linear convolutions with a 3×3 kernel which can be provided by the user or chosen from a variety of built-in kernels. Among the latter are kernels to enhance point sources and kernels to measure gradients in any of 8 directions. The 'SOBL' edge-enhancement filter can bring out jets, wisps, and points in the data, while the gradient convolutions produce images which resemble a landscape viewed from above with illumination at some glancing angle (as when viewing the Moon). Both are very effective when displayed on the TV or by the GREYS / QMSPL combination (see § 8.4.4). Enter EXPLAIN NINER ↵ for additional information.

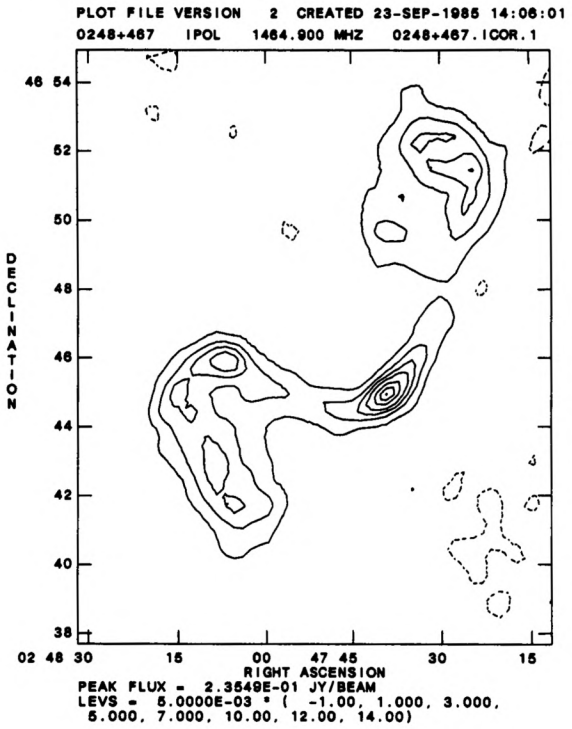
MWFLT, at present, applies any one of four nonlinear, low-pass filters to the input image. Each filter is applied in a user-specified window surrounding each input pixel. One of the operators is a "normalization" filter designed to reduce the dynamic range required for the image while bringing out weaker features. The others produce, at each pixel, the weighted sum of the input and the median, the "alpha-trimmed" mean, or the alpha-trimmed mode of the data in the window surrounding the pixel. These last filters can be turned into high-pass filters by subtracting the output of MWFLT from the input with COMB. Type EXPLAIN MWFLT ↵ for further information.

8.4.3. Modeling

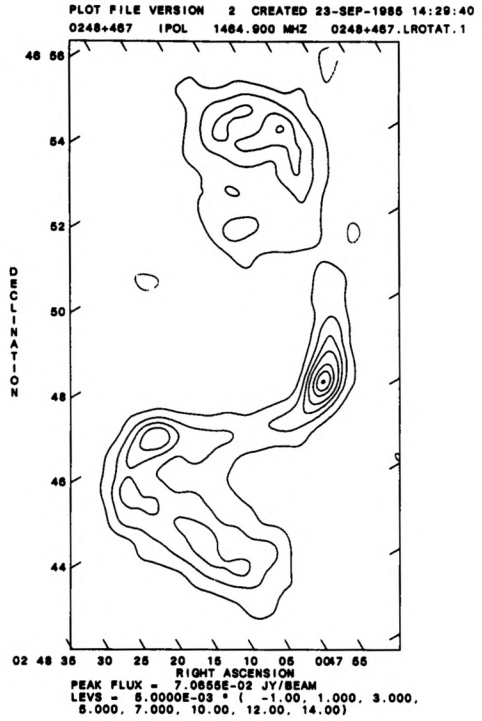
The addition of model data to an image or *uv* data set is often useful either to simplify later processing steps or to study processing steps using a "source" of known structure. For example, the removal of the response to an appropriate uniform disk from the *uv* data for a planet will leave CLEAN the task of deconvolving only the remaining fine-scale structure to which it is well suited. The removal of a few bright point sources of known position and strength may allow imaging with significant tapers in a numerically smaller field. The tasks IMMOD and UVMOD will add (or subtract) a point, Gaussian, disk, or rectangular source to the (scaled) input image or *uv* data, respectively. Both tasks can also add noise and both allow the original data to be replaced by the model. Type EXPLAIN IMMOD ; EXPLAIN UVMOD ↵ for details.

The task CCMOD will create a clean-components file representing the chosen Gaussian or disk model. CLEAN may then be "restarted" with the model as its initial set of components. The task UVFIT may be useful for fitting Gaussian or uniform-sphere models to small (< 2000 visibility) *uv* data sets.

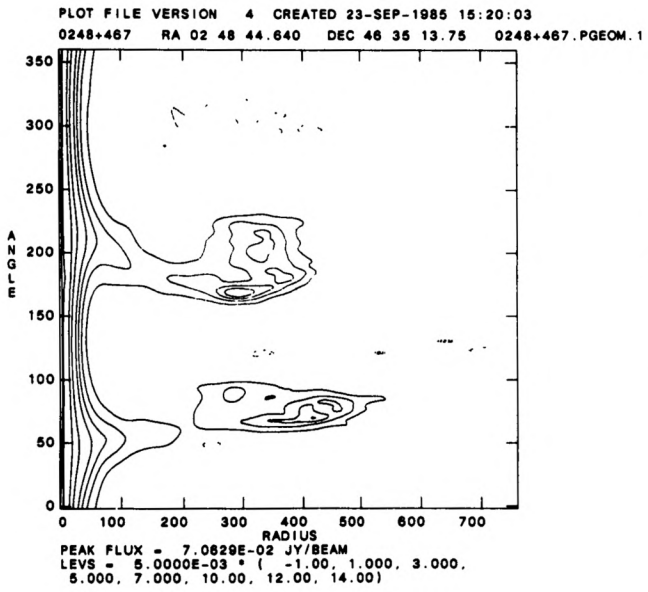
8.4.4. Examples



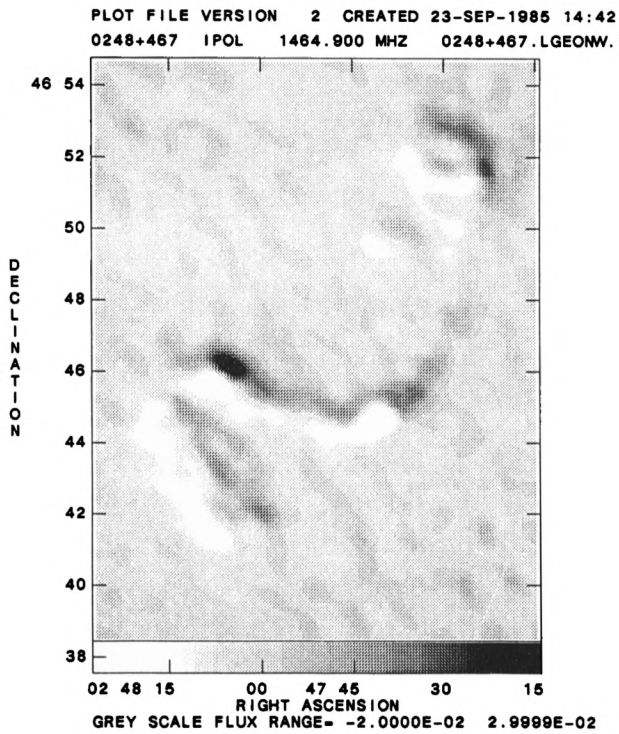
input image



LGEON with rotation and interpolation



PGEON



NINER: 'NW ' derivative

9. SPECTRAL-LINE SOFTWARE

Spectral-line software generally involves three-dimensional images, often called "cubes", in which the third dimension is frequency or velocity. Special programs are needed to build and transpose these cubes and to display them properly. Much of the continuum software will work on data cubes or on appropriate two-dimensional images from a data cube. Spectral-line *uv* data are read into *AIPS* from "*uv*" FITS tapes; if written from the VLA "Pipeline", these contain files of eight channels at a time. Spectral-line images can be read into *AIPS* from FITS tapes.

This section is organized as follows. First, a "flow-diagram" is given of how a typical spectral-line data set could be processed in *AIPS*. Second, input examples are given for each program. Finally, some more advanced profile analysis programs are discussed. Some aspects of the art of spectral-line imaging are discussed in the 1985 *NRAO/VLA Workshop on Synthesis Mapping*, in Lecture 12 on "Spectral-line Techniques".

9.1. Data reduction strategies

Starting from *uv* data,

1. Use UVLOD to load data from FITS tape (see § 9.2).
2. Use IMHEAD to check the headers to see which frequency channels are in the file.

or, starting from images,

1. Use IMLOD to load images from FITS tape and then go to step 11.

Do you need self-calibration? Decide on this by first processing the broad-band continuum channel or a line channel with a very strong signal:

3. Use UVCOP to split off a channel with a strong line, or use channel 0.
4. Use UVSRT to sort the data from TB to XY order.
5. Use (UVMAP + APCLN), or MX, to make an image and CLEAN it.
6. Use ASCAL to self-calibrate the TB-sorted *uv* data.
7. Repeat steps 4 + 5.

Now ask: did the image improve so much that the improvement would be noticeable even in the (line minus continuum) images? That is, would such dynamic range improvement of a (line minus continuum) image be above the noise? If the answer is "yes":

8. Use ASCOR to apply the same correction to all your channels.
9. Use UVSRT to sort the data to XY order.
10. Use UVMAP to make the images (§ 9.3). (You may want to use MX for wide field continuum imaging, or in the exceptional case that *no* continuum has to be subtracted *and* the line signal needs CLEANing).
11. Use MCUBE to put the images in a cube (§ 9.4).
12. Use ALTDEF to put velocity in the header (§ 9.5).
13. Use ALTSWTCH to display velocities rather than frequencies.

14. Use TVMOVIE to look at all your images (§ 9.6).
15. Use OFFZOOM and make hard copies of TV planes with sets of channel images (TVON 1 ; TVOFF 234 $^{O}_{R}$, TVOFF 1 ; TVON 2 $^{O}_{R}$, etc.)

Do you need to subtract a continuum (§ 9.7)? If so,

16. Use SUMIM to combine sets of adjacent channels.
17. Use COMB to combine mean continuum images at the outer ends of the band.
18. Use COMB again to subtract the continuum image from the cube.
19. Use TVMOVIE to look at the cube with continuum subtracted.
20. Use OFFZOOM and make hard copies of line images.

As an important check:

21. Use IMEAN to calculate the noise in some line-free channels.

Did you reach the theoretical noise? If not, *think*, and go back to steps 10 or 16. Do not proceed until you understand what went wrong.

Do you need to CLEAN the line signal?

22. Use SUBIM to split off the channels that need CLEANing (§ 9.9).
23. Use APCLN to CLEAN the images (if made by UVMAP), otherwise MX.
24. Use MCUBE to put the clean images back in the cube.

Once you are satisfied with the CLEANing:

25. Use TRANS to change the axis order in the cube to let you look at profiles (§ 9.10).

Now you can do a variety of things (§ 9.11):

26. Use COMB to compute optical depth.
27. Use BLSUM or PRFPL (VLA only) to make integral profiles.
28. Use SLICE and TKSLICE to look at single profiles.
29. Use PLCUB or XPLOT to look at many profiles.
30. Use XBASL to remove baselines.
31. Use XGAUS to fit gaussians.
32. Use BLANK to filter out noise in many different ways.
33. Use XSMTH to Hanning smooth.
34. Use CONVL to smooth spatially.
35. Use XMOM to calculate moments.
36. Use MOMNT to smooth, blank and calculate moments in one go.
37. Use GAL to fit a rotation curve.
38. Use TV3COLOR, TVHUEINT, TVSLV to have fun with TV displays of the data.
39. Use KONTR (VLA only) to plot all the channel images in one plot.

9.2. Reading in multichannel *uv* data

UVFITS tapes written from the VLA Pipeline system contain files with 8 frequency channels at a time. Other systems may produce files with other numbers of channels (any number can be handled in *AIPS*).

As a VLA-specific example, consider a spectral-line data set with 31 frequency channels. The UVFITS tape contains 5 files: the first, channel 0; the second channels 0 to 7; the third, channels 8 to 15, etc. Suppose that the source is N315. To read all 5 files:

> TASK 'UVLOD' ; INP $\subset_R$	to review the inputs.
> OUTCL ' ' ; OUTN ' ' $\subset_R$	to use default outname and outclass.
> POINTS 1 $\subset_R$	to make it different from zero.
> DOWAIT TRUE $\subset_R$	to have AIPS wait for each UVLOD to finish before starting the next one.
> FOR I = 1 : 5 ; PRINT I ; GO ; END $\subset_R$	to run UVLOD 5 times.

Now check the headers with IMHEAD. This gives a lot of information, but the line giving the frequency differentiates the files. For example, if you centered at 1400 MHz and used a total bandwidth of 3.125 MHz and a channel spacing of 97.656 kHz, you should see the following:

```
TYPE    PIXELS   COORD VALUE  AT PIXEL    COORD INCR   ROTAT
FREQ         1  1.4000000E+09     1.00 2.3400000E+06    0.00
FREQ         8  1.4000000E+09    17.00 9.7656000E+03    0.00
FREQ         8  1.4000000E+09     9.00 9.7656000E+03    0.00
```

for the first three files loaded. The listing shows the number of channels (PIXELS) there are in the file, the frequency at the reference pixel (which can be outside the file), and the channel spacing (COORD INCR).

9.3. Making spectral-line images

There are two programs you can use, UVMAP and MX, both of which are shown in examples given below. UVMAP can make images of all frequency channels in a file. It does the gridding once, for the first channel in the file. It produces one beam for the first channel. MX does the gridding separately for each channel and produces beams for all channels. It puts the channel images of 1 *uv* file into a cube. In most cases, you will want to subtract a continuum before you do any CLEANing, so we won't discuss the simultaneous imaging and cleaning option in MX.

The inputs in UVMAP that are spectral-line specific are, for example, to image channel 0 (one image and one beam):

> TASK 'UVMAP' ; INP $\subset_R$	to review the inputs.
> INNA 'N315' $\subset_R$	to select the source.
> INSEQ 1 $\subset_R$	to select the first file.
> STOKES ' ' $\subset_R$	to make a total intensity image.
> NMAPS 1 $\subset_R$	to make only 1 image.
> CHANNEL 1 $\subset_R$	to start at the first frequency in the file.
> OUTN 'N315C0' $\subset_R$	to give channel 0 a special name.
> GO UVMAP $\subset_R$	to run UVMAP.

To image channels 1 to 7 and make one beam (channel 1):

> INSEQ 2 $\subset_R$	to select the second file.
> STOKES 'LINE' $\subset_R$	to make more images at different frequencies.
> NMAPS 7 $\subset_R$	to make 7 images.
> CHANNEL 2 $\subset_R$	to start at the second frequency in the file.
> OUTN ' ' $\subset_R$	to use the default outname.
> GO UVMAP $\subset_R$	to run UVMAP.

To image channels 8 to 15 and make one beam (channel 8):

> INSEQ 3 $\subset_R$	to select the third file.
> NMAPS 8 $\subset_R$	to make 8 images.
> CHANNEL 1 $\subset_R$	to start at the first frequency in the file.
> GO UVMAP $\subset_R$	to run UVMAP.

As these illustrations show, the meaning of CHANNEL in UVMAP is the frequency pixel in the *uv* data file, at which one wants to start making a total of NMAPS images. The examples also use the global nature of AIPS' inputs; those, such as INNAME, which don't change, don't have to be entered for each execution of UVMAP.

To image channels 8 through 15 using MX:

> INNA 'N315' $\subset_R$	to select the source.
> INSEQ 3 $\subset_R$	to select the third file.
> STOKES 'I' $\subset_R$	to make intensity images.
> NMAPS 8 $\subset_R$	to make 8 images.
> CHANNEL 1 $\subset_R$	to start at the first frequency in the file.
> ECHAN 0 $\subset_R$	to grid each frequency separately.
> BCHAN 0 $\subset_R$	to specify that you're not restarting an image.
> GO MX $\subset_R$	to run MX.

This makes images and beams of channels 8 through 15.

9.4. Building the cube

The task MCUBE is used to put a series of 2- or 3-dimensional images at different frequencies into a 3-dimensional image ("cube") cataloged on disk. The cube can be transposed to create images with velocity on one axis and RA or DEC as the other axis of each plane. Many spectral-line functions make sense only on cubes. Once the images have been put into a cube, you can inspect all the data with one command, TVMOVIE. And, if you want to perform the same operation on many images, it is most convenient to do it on a cube instead.

To put 31 frequency channel images into one cube, type:

> TASK 'MCUBE' ; INP $\subset_R$	to review the inputs.
> INNA 'N315' $\subset_R$	to select the source.
> INCL 'IMAP' $\subset_R$	to select input image class for all images.
> INSEQ 1 ; IN2SE 31 ; IN3SE 1 $\subset_R$	to set the first and last image and the step in the task's loop over input sequence number.

> AXREF 1 ; AX2REF 31 C_R	to set the pixel coordinate of the first and last image on the third axis in the cube.
> NPOINTS 31 C_R	to set the total number of points on the third axis.
> OUTN ' ' C_R	to use the default OUTNAME.
> OUTCL 'LMVCUB' C_R	to specify, in OUTCLASS, the order of axes.
> GO C_R	to run MCUBE.

Later on, you might want to replace some of these images by CLEANed images. *E.g.*, assume that you have CLEANed channels 10 to 20. These images got the class ICLN. It is a good idea to give them sequence numbers that are the same as the channel numbers, thus 10 to 20. Putting them in the existing cube can then be done as follows:

> TASK 'MCUBE' ; INP C_R	to review the inputs.
> INNA 'N315' ; INCLA 'ICLN' C_R	to select the clean images.
> INSE 10 ; IN2SE 20 ; IN3SE 1 C_R	to set the first and last image and the step in the loop.
> OUTN 'N315' ; OUTCL 'LMVCUB' C_R	to select the existing cube.
> OUTSE 1 C_R	
> GO C_R	to run MCUBE.

9.5. Modifying the image header

On occasion you may feel the need to modify or add to the information in the image header. For example, to add an alternate velocity description for the frequency axis of a cube, type:

> INDISK *n* ; GETN *ctn* C_R	to select the image.
> AXTYPE 'OPTHEL' C_R	to specify optical-convention velocities relative to the Sun.
> AXREF 16 C_R	to specify the velocity reference pixel (channel 16 is the center of the band in our example).
> AXVAL 5.E6 C_R	the velocity at the reference pixel in m/s.
> RESTF 1420.4E6, 5752 C_R	to specify the line rest frequency in Hz (1420405752 Hz).
> ALTDEF C_R	to add the information to the header.
> ALTSW C_R	to switch between frequency and velocity information in labeling.

Observers may find the Galactic coordinates of their sources to be of interest. To switch the header between Celestial and Galactic coordinates, type:

> CELGAL C_R	to go to galactic coordinates.
> CELGAL C_R	to go back to celestial coordinates.

9.6. Displaying the cube

The easiest way to look at the cube is by using TVMOVIE. This verb loads sub-portions of planes of a cube into the TV memory, and displays them in sequence at a variable frame rate. Type:

> INDISK *n* ; GETN *ctn* C_R	to select the cube.
> INP TVMOVIE C_R	to review the inputs; use defaults for a start.

> TVMOVIE ↵ to load the images and start the movie.

Now follow the instructions on your screen on how to change the transfer function, change speed, etc. To restart the movie, type:

> REMOVIE ↵

Once you are done, type:

> OFFZOOM ↵ to show one of the memory planes with a number (16 in our example) of images.

To look at a different TV channel with the remaining images, type:

> TVON 2 ; TVOFF 1 ↵ depending on which one you were looking at.

Be aware that the order of planes is reversed in even numbered planes.

In general, if you want to use any of the other display programs, you must specify which plane in the cube you want to see. So, *e.g.*, if you want to do TVALL on channel 16, which is on pixel 16 at the third axis, you type:

> TBLC 0 0 16 ; TVALL ↵

Note that adverbs TBLC and TTRC are used for TV displays while BLC and TRC are used for tasks.

9.7. Subtracting the continuum

The continuum can be subtracted in a variety of ways. XBASL may offer the best method, but the standard is still to combine a group of line-free dirty channel images and subtract that from the cube. Let us assume that there is no line emission in channels 1 through 10 and 21 through 31. We first make two images of channels 1–10 and of 21–31 with SUMIM. Type:

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

> INNA 'N315' ; INCL 'IMAP' ↵ to select the images.

> INSE 1 ; IN2SE 10 ; IN3SE 1 ↵ to select the first and last image and the step in the loop.

> OUTN 'N315C1,10' ↵ to select an output file name.

> FACTOR 0 ↵ to use the default, which takes the average of the images.

> GO ↵ to run SUMIM.

To do the other set of images:

> INSE 21 ; IN2SE 31 ; IN3SE 1 ↵

> OUTNAME 'N315C2131' ↵

> GO ↵

With COMB, these images can be averaged by typing:

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

> INNA 'N315C1,10' ; INCL 'SUMIM' ↵ to select the first image.

> INSE 1 ↵

> IN2NA 'N315C2131' ; IN2CL 'SUMIM' ↵ to select the second image.

> IN2SE 1 ↵

> OPCODE 'SUM' ↵ to take APARM(1) × MAP_1 + APARM(2) × MAP_2.

> APARM 0.5, 0.5, 0 ↵ to average the images.

> OUTN 'N315CON' $\subset_R$ to select an output file name.
> GO $\subset_R$ to run COMB.

and then subtracted from the cube by entering:

> TASK 'COMB' ; INP $\subset_R$ to review the inputs.
> INNA 'N315' ; INCL 'LMVCUB' $\subset_R$ to select the cube.
> INSE 1 $\subset_R$
> IN2NA 'N315CON' ; IN2CL 'COMB' $\subset_R$ to select the continuum.
> IN2SE 1 $\subset_R$
> OUTN 'N315L-C' ; OUTCL 'LMVCUB' $\subset_R$ to give it an output name.
> OPCODE 'SUM' $\subset_R$
> APARM 1, -1 $\subset_R$ to take $MAP_1 - MAP_2$.
> GO $\subset_R$ to run COMB.

In general, this will work fine; however, in some cases, *e.g.*, if you have used very broad bandwidth or if you have a very strong continuum source, you might want to do the subtraction in the *uv*-plane. Use SUMIM and COMB to combine the beams of the line-free channel images. Use APCLN to CLEAN the mean continuum image with the mean continuum beam. Use UVSUB to subtract these clean components from all the *uv* data sets. Remake the channel images, form a mean residual continuum image, and subtract it from the channel images.

9.8. Converting from real to integer and back

The output from MCUBE or MX is in floating point, but some programs work only on integers (*e.g.*, COMB at present). CNVRT converts from floating point to integer and back; type:

> TASK 'CNVRT' ; INP $\subset_R$ to review the inputs.
> INDISK *n* ; GETN *ctn* $\subset_R$ to select the image.
> GO $\subset_R$ to run CNVRT.

9.9. Decomposing the cube into separate images

Some programs do not work on cubes, *e.g.*, APCLN. Suppose that you want to CLEAN channels 10 to 20 after you have subtracted the continuum. To separate the images from the cube, use task SUBIM; type:

> TASK 'SUBIM' ; INP $\subset_R$ to review the inputs.
> INNA 'N315L-C' ; INCL 'LMVCUB' ; INSE 1 $\subset_R$ to select the cube.
> OUTN 'N315L-C' ; OUTCL 'IMAP' $\subset_R$ to give it an output name and class.
> BLC 0 ; TRC 0 $\subset_R$ to select full planes.
> FOR J=10:20 ; BLC(3)=J ; TRC(3)=J ; PRIN J ; OUTSE J ; GO ; WAIT ; END

This runs the program 11 times, taking planes 10 to 20 and creating separate images for them. To put the CLEANed images back into the cube, see §9.4.

9.10. Transposing the cube

The task TRANS will transpose the cube. Typical inputs are:

> TASK 'TRANS' ; INP $\subset_R$	to review the inputs.
> INCLASS 'LMVCUB' $\subset_R$	to select the untransposed cube.
> TRANSC '312' $\subset_R$	to make new axis order 3,1,2 in terms of the old axis order (RA, DEC, VEL becomes VEL, RA, DEC).
> OUTCL 'VLMCUB' $\subset_R$	to give it an outclass reflecting the axis order.
> BLC 0 ; TRC 0 $\subset_R$	to transpose the whole cube.
> GO $\subset_R$	to run the program.

9.11. Further profile analysis

A wide variety of programs is available to do further analysis of the data.

> HELP CUBE $\subset_R$	to list them all on your terminal.

This help file is also listed in § 14.

By displaying the transposed cube you can inspect RA, VEL or DEC, VEL images. The cube can be rotated with LGEOM, *e.g.*, to align one of the axes with the major axis of a galaxy.

A single profile can be produced from these images with SLICE, then plotted using TKSLICE (see § 7.5). PLCUB, PLROW and XPLOT are convenient programs for displaying multiple profiles.

The task XBASL can be used to remove baselines. Be aware, however, that if you have made an error in the calibration, this has most likely caused slopes in amplitude *and* phase. Therefore, it is generally better to track down the error and correct it than to decide (arbitrarily) to take out slopes in amplitude.

Smoothing and blanking are important for almost all analysis programs. CONVL works on cubes and does a spatial smoothing (on LMV cubes). Using all the defaults in XSMTH performs a Hanning smoothing in velocity (on VLM cubes). Smoothing is not just useful for bringing out weak extended signals. Smoothed images can also assist in determining the boundaries of sources to set windows for subsequent spectral analysis. For example, the smoothed cube could be used to set the CLIP limits in task COMB to be applied to the unsmoothed cube.

The task BLANK offers a variety of algorithms for "blanking" out regions of bad data or source-free regions in spectral-line cubes. It has an interactive mode, which allows you to indicate with the cursor on the TV what are "good" regions. Set everything to default, use OPCODE 'TVCU' $\subset_R$ and type GO BLANK $\subset_R$. Then just follow the instructions, pushing button A to lay out the polygon and button D followed by $\subset_R$ to go to the next image. If marking "bad" regions is easier, set DOINV TRUE $\subset_R$ before running BLANK.

The blanked cubes can be used to calculate integral profiles with BLSUM (or PRFPL at the VLA) and to calculate moments 0 to 3 of the profiles with XMOM. Thus, the 0-moment image will be the integral under the profile (*e.g.*, total HI), the first moment is the velocity field, etc. Task MOMNT does the smoothing, blanking and calculating of moments all in one run. This is very easy to use, but can be dangerous since you don't see what is going on. RGBMP computes "integral" images another way — as three weighted sums representing the low, center, and high velocity parts of the cube. An interesting display results from:

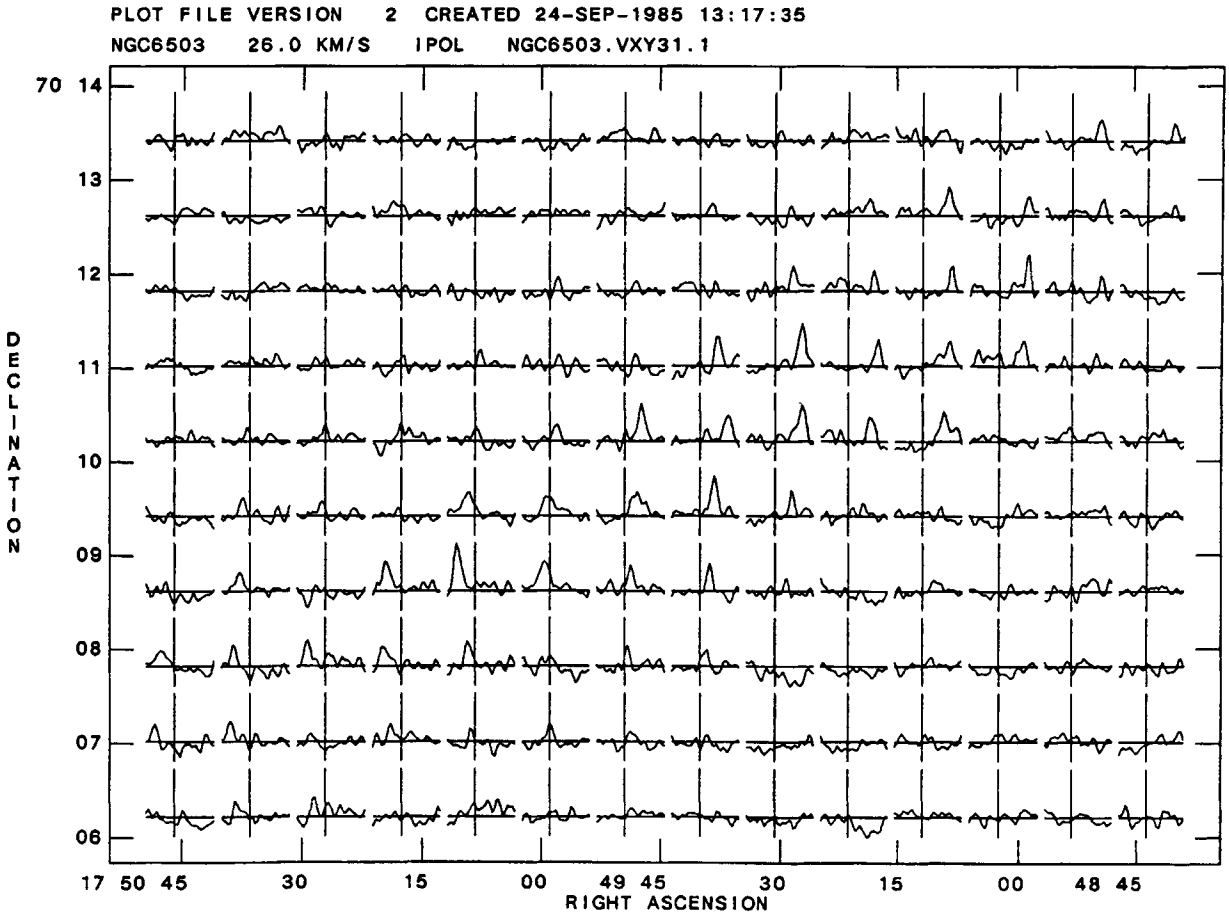
> TVINIT ; TBLC 0 ; TTRC 0	to initialize everything.
> INDI *n* ; GETN *ctn* $\subset_R$	to select the RGBMP output.
> TV3C ; FOR TVCH = 1 TO 3 ; TBLC(3) = TVCH ; TVLO ; END $\subset_R$	

If you prefer to fit Gaussians instead of calculating moments, the program XGAUS can be used. It is a good idea to use XPLOT first to look at (a sample of) the profiles, before you do any Gaussian fitting. In most cases, it is perhaps preferable to use the interactive mode, so that you can see what is happening, but be aware that it might be rather time-consuming. The experimental task NNLSQ performs a constrained, non-linear deconvolution of the spectra in a VLM cube.

GAL fits models of galaxy rotation to images of the predominant velocity (*e.g.*, the first moment images written by XMOM, XGAUS, or MOMNT).

At the VLA, you can plot all your channel images in one plot on the Zeta plotter. Star positions and beams can be included. All this is done with the task KONTR.

9.12. Sample display from PLCUB



PLCUB

10. REDUCING VLBI DATA IN *AIPS*

This section focuses on the three main differences between reducing VLA and VLBI data in *AIPS*. The special concerns for VLBI are: global fringe fitting, applying *a priori* amplitude calibrations, and imaging data sets that are sparsely sampled in the *uv* plane. The *AIPS* tasks that have been written primarily for VLBI have names that are prefixed with a V or VB. A list of programs of particular interest for VLBI may be found in § 14 or displayed on your terminal by typing HELP VLBI CR_R. Remember, the best and most complete information available on all *AIPS* verbs and tasks may be found in their EXPLAIN files.

10.1. Copying VLBI data into and out of *AIPS*

There are three portals into *AIPS* for VLBI data. They are:

1. UVLOD which reads FITS tapes.
2. VBCIT which reads Caltech VLBI Merge formatted data.
3. VBLIN which reads NRAO/SAO VLBI formatted data.

There are two portals out of *AIPS*. They are:

1. FITTP which writes a FITS tape.
2. TOVLB which writes a Caltech VLBI Merge format file.

10.1.1. VLBI data sets on a FITS tape.

To copy a *uv* data file from a FITS tape, run UVLOD:

> TASK 'UVLOD' ; INP CR_R	to review the inputs.
> INTAPE 1 CR_R	to specify which tape drive is used.
> OUTNA 'DA193VLB' CR_R	to set the output file name and class.
> OUTCL 'UVDATA' CR_R	
> GO CR_R	to run the program.

10.1.2. Data from the Caltech Mark II VLB Correlator

To copy a VAX-11 file containing data in Caltech Merge format, run VBCIT:

> TASK 'VBCIT'; INP CR_R	to review the inputs.
> INFILE 'UMA3:[VLB.ME]DATA.CIT' CR_R	to set the complete (VAX) input file name.
> OUTNA 'DA193VLB' CR_R	to set the output file name and class.
> OUTCL 'VBCIT' CR_R	
> NPOINTS 20 CR_R	to set the maximum number (in 1000's) of visibility records to be copied.
> APARM 0 CR_R	to reset the APARM options to the default values.
> GO CR_R	to run the task.

10.1.3. Data from the NRAO Mark II VLB Correlator

Generally, it is easier to copy all, or part, of your VLBI data tape onto disk first. The *AIPS* task, VBLIN, will run slowly, so it is more comfortable not to tie up a tape drive for several hours. Run the NRAO/SAO program EDIT or LINEEDIT or DOCSAMP to copy selected data from tape to disk. These programs run only in the VAX/VMS environment, and run outside of *AIPS*. Mount your VLBI tape with the foreign qualifier:

$ MOUNT/FOR MTA0: TAPE TAPE ↵

and run LINEEDIT. Answer the * prompt with these parameters:

* INFILE='IBMVAX' TRANS ↵

* OUTFILE='*output file name*' DATAHDR=24 / ↵

If you need to run LINEEDIT more than one time, the output files can be appended onto the same VAX output file by specifying the APPEND parameter after the * prompts.

Copy the VLBI data into *AIPS* via the task VBLIN, which transforms delay lags to frequency channels and applies corrections for high-order terms in the correlator model. Use your favorite text editor to create a text file (outside of *AIPS*) that contains a list of the names of all of the stations that participated in the VLBI run. Enter them one per line, left-justified, and spelled exactly as they were spelled when used by the correlator. Then, from inside AIPS, run VBLIN:

> TASK 'VBLIN' ; INP ↵	to review the inputs.
> INFILE 'UMA3:[VLB.ME]DATA.DAT' ↵	to set the complete input file name of the visibility data.
> IN2FIL 'UMA3:[VLB.ME]STNS.LIS' ↵	to specify the complete input file name of a list of all stations used in the VLBI observation (see above).
> OUTNA 'DA193VLB' ↵	to select the output file name and class.
> OUTCL 'VBLIN' ↵	
> SOURCE 'DA193' ↵	to specify the source whose data are to be copied.
> NPOINTS 20 ↵	to give the maximum number (in 1000's) of visibility records to be copied.
> BCHAN 4 ; ECHAN 10 ↵	to define the range of delay lags to be transformed into frequency channels. Generally, the NRAO/SAO formatted data contains 12 delay lags where zero residual delay is centered on channel 7. The present values cause 7 delay lags to be transformed into 4 single-sideband frequency channels.
> APARM 0, 0, 0, 1, 1, 1, 1, 265, 1, 0 ↵	to switch on the transform and various correction algorithms. APARM(8) is the reference day number. It should be the earliest day in your observing run. If you run VBLIN in several separate passes in order to load sub-array data into *AIPS*, use the same reference day number in each VBLIN pass. Otherwise, your sub-array data with different reference days cannot be merged.
> GO ↵	to run the program.

Warning: the Caltech VLBI package and the *AIPS* tasks which interact with it (VBCIT, VBLIN, and TOVLB) have not been written to *AIPS*' usual software portability standards. As a result, they run well on VAXes under VMS, but may not be available on other computers and operating systems.

10.2. Sorting, concatenating and merging *uv* data files.

The *AIPS* data files created by VBLIN and UVLOD will be in an arbitrary sort order. Use UVSRT to sort them into time-baseline order:

> TASK 'UVSRT' ; INP $\crarr$	to review the inputs.
> INDISK *n* ; GETN *ctn* $\crarr$	to select the input file.
> OUTNA INNA ; OUTCL 'TBSRT' $\crarr$	to specify the output file.
> GO $\crarr$	to make the sorted *uv* file.

You may use DBCON to concatenate several separate *uv* files. Both of the input files must have the same reference day number, and have identical antenna id numbers. That is, the antennas extension (AN) files with each input *uv* data file must be the same. You may list the contents of AN files using PRTAN. To run DBCON:

> TASK 'DBCON' ; INP $\crarr$	to review the inputs.
> INDISK *n1* ; GETN *ctn1* $\crarr$	to select the 1st input file.
> IN2DISK *n2* ; GET2N *ctn2* $\crarr$	to select the 2nd input file.
> OUTNA INNA ; OUTCL 'DBCON' $\crarr$	to specify the output file.
> DOARRAY 1 $\crarr$	to force DBCON not to mark the output data records as in separate sub-arrays. Both of the input files must have the same reference day and have identical antennas files.
> GO $\crarr$	to concatenate the two files.

VLBI correlators usually produce redundantly correlated data. Merge your UVSRT or DBCON output using UVAVG:

> TASK 'UVAVG' ; INP $\crarr$	to review the inputs.
> INDISK *n* ; GETN *ctn* $\crarr$	to specify the input file.
> OUTNA INNA ; OUTCL 'UVMRG' $\crarr$	to specify the output file.
> YINC 4.0 $\crarr$	to set the averaging interval of the input data records (in seconds).
> OPCODE 'MERG' $\crarr$	to direct the task to perform the merge operation.
> GO $\crarr$	to run the program.

10.3. Global fringe fitting

There are two ways to use global fringe fitting. The choice depends on the source strength. Be prepared to spend some time fiddling in order to understand how the process works. The first try probably should be on a subset, *i.e.*, a one-hour slice, of your experiment.

10.3.1. Standard global fringe fitting

This subsection describes fringe fitting on data from a source that can be detected easily on most baselines. This "standard" mode of global fringe fitting uses only data from the program source itself. The sequence of programs described below should yield good results.

Run VBFIT to find the delay and fringe rate residuals on a per station basis, but do not use VBFIT to apply these corrections to the visibility data.

> TASK 'VBFIT' ; INP ⏎	to review the inputs.
> INDISK *n* ; GETN *ctn* ⏎	to specify the input file.
> IN2NA ' ' ; IN2CL ' ' ⏎	to reset the input model file to the null file.
> OUTNA INNA ; OUTCL 'VBFIT' ⏎	to specify the output file.
> APARM 0 ; BPARM 0 ⏎	to reset the APARMs and BPARMs to the default values.
> APARM(1) 4 ⏎	to set the fringe searching and fitting interval (in minutes). It may be set safely to a time interval at least as long as the phase coherence time.
> APARM(5) 4 ⏎	to give the integration time of the input data records (in seconds).
> APARM(6) 750 ⏎	to specify the residual delay range, centered at 0, over which to search for fringes (in nanoseconds). Mark II VLBI observes through 2 MHz bandpasses and correlations are sampled in 250 nanosec intervals. The Nyquist delay range is *nchans* × 250 nanoseconds, where *nchans* is the number of frequency channels in the visibility records.
> APARM(7) 100 ⏎	to set the residual fringe rate range, centered at 0, over which VBFIT searches for fringes (in milliHz). The Nyquist fringe rate range is 1000.0 mHz / 2 × $T_{\rm avg}$, where $T_{\rm avg}$ is the data-record average interval (in seconds).
> APARM(8) 1 ⏎	to turn on a CRT print-out that shows the SNR's of each station for each fit interval.
> BPARM(2) = -1 ⏎	to tell VBFIT that the input data have already been divided by a model (which is equivalent to using a point source model) and to have only the delay and rate solutions determined.
> LEVS 1.5, 1.2, 2.3, 9.3, ... ⏎	to specify the antenna weights that are used to select a reference antenna, and to calculate the delay and fringe rate solutions. Each LEVS entry corresponds to one antenna in the same order as they appear in the antennas file. (Use PRTAN to list the antennas file.) Estimate the LEVS for each antenna by using the system temperatures in Janskys and the antenna sensitivities. That is, LEVS = 50.0 × $\sqrt{T_{\rm ant/Jy}/T_{\rm sys}}$.
> INP ⏎	to check the inputs.
> GO ⏎	to do the fit — finally.

Use PRTDR to list the delay and delay rate solutions.

Run VBCOR to smooth the delay/rate solutions obtained from VBFIT, and to apply them to the visibility data. Use the same input file as in the VBFIT step and type:

> TASK 'VBCOR' ; INP ⏎	to review the inputs.
> IN2DISK *n* ; GET2N *ctn2* ⏎	to get the VBFIT output file. VBCOR will use the delay and rate solutions in the DR extension file.
> OUTNA INNA ; OUTCL 'VBCOR' ⏎	to specify the output, corrected *uv* data file.
> BPARM(1) 0 ⏎	to average the frequency channels in the output records together.

> BPARM(2) 4.0 C_R	to give the averaging time of the input visibility records.
> TIMSMO 2.0 C_R	to set the smoothing time for the delay and delay rate solutions to 2.0 hours.
> GO C_R	to run the program.

The VBCOR output file is the fringe-fitted *uv* data set.

Before proceeding further, it is important to satisfy yourself that the VBFIT and VBCOR operations were successful. You should examine the delay and rate solutions found by VBFIT (the DR extension file in the output file), and the smoothed solutions in the DR extension file from VBCOR . Use PRTDR. The delays are listed in nanoseconds and the rates in milliHz. Entries of $-1.$ indicate that no solution was possible. The delays and rates should not vary wildly from one time interval to the next.

When you are satisfied with the preceding steps, run VSCAL to remove the station-dependent phase errors left over from the fringe fitting. The phase solutions may be done on a record-by-record basis, but it is faster to use as long a time interval as is consistent with the coherence time, *i.e.*, $< 1/2$ the coherence time.

> TASK 'VSCAL' ; INP C_R	to review the inputs.
> INDISK *n* ; GETN *ctn* C_R	to get the VBCOR output file as the VSCAL input file.
> IN2NA ' ' ; IN2CL ' ' C_R	to reset the model file to a null file.
> OUTNA INNA ; OUTCL 'VSCAL' C_R	to specify the output file.
> APARM 0 ; BPARM 0 C_R	to reset the APARMs and BPARMs to the default values.
> APARM(7) 2 C_R	to force a phase solution only.
> APARM(9) 0.0666 C_R	to set the solution time interval equal to the individual record average interval (in minutes). In this example, 4 seconds = 0.0666 minutes.
> BPARM(1) 1 C_R	to cause the input data to be divided by a point-source model.
> BPARM(2) 1.0 C_R	to give the flux density of the point source model.
> INP C_R	to check the inputs.
> GO C_R	to run the program.

Now you may time-average the fringe-fitted data set down to a reasonable size. Run UVAVG to average the VSCAL output:

> TASK 'UVAVG' ; INP C_R	to review the inputs.
> INDISK *n* ; GETN *ctn* C_R	to specify the VSCAL output file as the UVAVG input file.
> OUTNA INNA ; OUTCL 'UVAVG' C_R	to specify the output file.
> YINC 120.0 C_R	to set the time-averaging interval to 120 seconds.
> OPCODE 'AVG' C_R	to enable the averaging operation.
> GO C_R	to run UVAVG.

At this point, it is well worth spending time to examine your visibility data carefully. You may plot them with VBPLT or UVPLT, and list them with PRTUV. You may transfer a copy of the *uv* data out of *AIPS* and into a VAX-11 disk file in the Caltech VLB Merge format with TOVLB.

10.3.2. Global fringe fitting a very weak source

In order to fit fringes, VBFIT has to find fringes. If VBFIT has to search through a large residual delay by rate window, a weak signal will always be lost in the noise. Increasingly weaker fringes can be detected as the search window is made smaller. But, of course, the smaller search windows must be more nearly centered on the fringes. Using calibrator-source observations, the station-based delay and rate residuals can be measured as often as you observed the calibrator source. Then the calibrator-source delays and rates can be used to shift the weak source's fringes to the center of the delay-rate window, thereby allowing fringe searching through a smaller window.

This "fringe delay and rate referencing" and subsequent fringe fitting are accomplished by running VBFIT and VBCOR in the order specified below. For the sake of brevity, some of the more obvious input parameters are not listed. If you are confused, see § 10.3.1 for details on VBFIT and VBCOR.

First, run VBFIT on the calibrator source to measure the residual delay and rate offsets.

> INDISK *n* ; GETN *ctn* C_R	to specify the input file (the calibrator source data).
> APARM(1) 4 C_R	to set the fringe fitting interval (in minutes). It may be set to a time interval as long as the phase coherence time.
> APARM(5) 4 C_R	to give the integration time of the input data records (in seconds).
> APARM(6) 750 C_R	to specify the residual delay range over which to search for fringes (in nanoseconds).
> APARM(7) 100 C_R	to specify the residual fringe rate range over which VBFIT searches for fringes (in milliHz).
> BPARM(2) = -1 C_R	to solve for delay and rate corrections only.
> LEVS 1.5, 1.2, 2.3, 9.3, ... C_R	to set the antenna weights.
> GO VBFIT C_R	to run the program.

Use PRTDR to list the delay and delay rate solutions.

Now correct the program-source data with the VBFIT solutions from the calibration-source scans. Run VBCOR on the program-source data:

> INDISK *n1* ; GETN *ctn* C_R	to get the program-source file.
> IN2DISK *n2* ; GET2N *ctn2* C_R	to get the VBFIT output file. VBCOR will use the delay and rate solutions from the calibration-source scans.
> BPARM(1) 1 C_R	to block the averaging of the output frequency channels.
> BPARM(2) 4.0 C_R	to give the averaging time of the input visibility records.
> TIMSMO 2.0 C_R	to set the smoothing time for the delay and delay-rate solutions to 2.0 hours.
> GO VBCOR C_R	to run the program.

Run VBFIT on the corrected program-source data. Use a small fringe-search window. Estimate the dimensions of the window from the stability and consistency of the delay/rate solutions on the calibration source.

> INDISK *n* ; GETN *ctn* C_R	to select as the input file the output file from the preceding VBCOR step.
> APARM(1) 10 C_R	to set the fringe fitting solution interval to a moderately long time, 10 minutes.

> APARM(5) 4 $\subset_R$	to give the input data averaging interval.
> APARM(6) 50 $\subset_R$	to set the fringe search range in residual delay, 50 nanoseconds.
> APARM(7) 10 $\subset_R$	to set the fringe search range in residual fringe rate, 10 milliHz.
> BPARM(1) 0 $\subset_R$	to cause the output frequency channels to be averaged together.
> BPARM(2) = -1 $\subset_R$	to solve for delay and rate corrections only.
> LEVS 1.5, 1.2, 2.3, 9.3, ... $\subset_R$	to set the antenna weights.
> GO VBFIT $\subset_R$	to run the program.

Next run VBCOR on the corrected program source data to smooth and apply the delay and delay-rate solutions:

> INDISK *n1* ; GETN *ctn1* $\subset_R$	to get the output file from the previous VBCOR step. Remember that these are the program-source data that have been corrected by the delays and rates derived from the calibrator-source data.
> IN2DISK *n2* ; GET2N *ctn2* $\subset_R$	to get the output file from the preceding VBFIT stage. These are the delay and rate solutions from the corrected program-source data.
> BPARM(1) 0 $\subset_R$	to average the frequency channels in the output records together.
> TIMSMO 2.0 $\subset_R$	to set the smoothing time for the delay and delay-rate solutions to 2.0 hours.
> GO VBCOR $\subset_R$	to run the program.

The output from VBCOR is the weak source fringe fitted data set. You should examine carefully the delay and rate solutions from each stage of the above menu (use PRTDR). Fringe searching on a very weak source is a touchy business. You may need to experiment a bit for best results.

When you are satisfied with your fringe fitting results, you may calibrate the phases and average the data. Run VSCAL to remove the residual station-dependent phase errors before time averaging. The phase solutions should be on a time interval $< 1/2$ the coherence time.

> INDISK *n* ; GETN *ctn* $\subset_R$	to get the VBCOR output file as the VSCAL input file.
> APARM(7) 2 $\subset_R$	to force a phase solution only.
> APARM(9) 2 $\subset_R$	to set the solution time interval equal to the 2 minutes for a coherence time > 4 minutes.
> BPARM(1) 1 $\subset_R$	to divide the input data by a point-source model.
> BPARM(2) 1.0 $\subset_R$	to specify the flux density of the point source.
> GO VSCAL $\subset_R$	to run the program.

Run UVAVG to average the VSCAL output:

> INDISK *n* ; GETN *ctn* $\subset_R$	to get the VSCAL output file as the UVAVG input file.
> YINC 120 $\subset_R$	to set the time-averaging interval to 120 seconds.
> OPCODE 'AVG' $\subset_R$	to enable the averaging operation.
> GO UVAVG $\subset_R$	to run UVAVG.

Display the visibility data using VBPLT, UVPLT, and PRTUV.

10.4 Amplitude calibration

The visibility amplitudes in the fringe fitted data set are raw cross-correlation coefficients. These data are converted to correlated flux densities by using the task VBANT. The amplitude calibration is done in the following manner:

1. Create a text file which contains the system temperatures, antenna temperatures and gain curve descriptions from each station in the VLB array.
2. Using observations of one or more calibration sources, calibrate the flux density scales of each antenna in the VLB array.
3. Apply the calibration file that results from steps 1 and 2 to the program source(s) using VBANT.
4. (Optional). Use BCAL1 and BCAL2 to solve for and remove baseline-dependent gain errors. This requires very high quality calibration source observations (good SNR data and an accurate source model).

10.4.1. Creating the calibration text file

Using your favorite text editor, create the calibration text file required by VBANT. The text file contains the source flux densities, and the system temperatures, antenna temperatures and gain curves from each VLB station in the observing run. The format of the calibration file is described in the *EXPLAIN VBANT* CR. Please read it carefully.

Once the text file is completed, copy it into the local *AIPS* RUN file sub-directory, [AIPS.RUN]. The file name must be six or fewer characters and the qualifier must be *.nnn* (your hexadecimal user id). Run VBANT on any source with DOCAT=1. You may list the gain table created by VBANT using PRTGA or plot it with GAPLT. It is well worth the time and effort required to get the system temperatures entered correctly in the calibration file. Be certain to enter T_{sys}'s at the end of scans immediately before source changes.

10.4.2. Constant Gain Corrections

After you have entered your best *a priori* numbers in the calibration text file, run VBANT on your calibration source scans. Plot the resulting correlated flux densities against *uv* distance using UVPLT. Bad data should be obvious and may be edited using CLIP and UVFLG. With good models of the calibrator sources' structures, you can determine corrections to the flux density scales for each station in the VLB array. Most VLB calibrator sources can be adequately described by one or two Gaussian components.

UVFIT can be used to fit a source model and overall antenna gains. It can handle up to 2,000 visibilities only, so averaging with UVAVG may be required. The following example shows how to fit antenna gains and a single elliptical gaussian model of known position and flux:

> TASK 'UVFIT' ; INP CR	to review the inputs.
> INDISK *n* ; GETN *ctn* CR	to specify the input *uv* data set.
> OPCODE 'GAUS' ; NGAUS 1 CR	to specify 1 gaussian component.
> GMAX 1.2 ; DOMAX FALSE CR	to fix the flux at 1.2 Jy.
> GPOS 0 ; DOPOS FALSE CR	to hold the position fixed at the origin.
> GWIDTH 0.002 , 0.001 , 45. CR	to provide an initial guess of the gaussian widths as 2×1 mas at a position angle of 45°.

> DOWIDTH TRUE ⏎	to fit for size.
> GAINERR 1 ⏎	to fit for all antenna gains with initial guess = 1.
> NITER 50 ⏎	to limit the fitting to 50 iterations.
> IMSIZE 0.0005 , 0.01 ⏎	to limit sizes to be in the range 0.5 to 10 mas.
> DOCAT TRUE ; INVER 1 ⏎	to save the solution in a CC file of version number 1.
> INP ⏎	to check the inputs.
> GO ⏎	to run UVFIT.

The scale-factor corrections from UVFIT may be entered as multipliers in the VBANT calibration text file. For example, if the NRAO scale factor is 1.043, set the FT parameter on the NRAO TSYS card to FT=(1.043*1.043). Alternately, VBCAL may be used to correct data already processed through VBANT. After applying the antenna gain solution with VBCAL or VBANT, you may compare the data with the model using VBPLT. If you have a sufficient amount of data, it is a good idea to image your calibrator sources. Use VSCAL and MX.

10.4.3. VBANT for program sources

Run VBANT on your program source. Set DOCAT = 1 to save the GA file. It's useful to examine the GA file using GAPLT and PRTGA. You can plot the visibilities with VBPLT or UVPLT and edit the wild points with CLIP or UVFLG.

10.4.4. Baseline-dependent gain errors

VLBI data sets typically have non-closing gain errors of several percent in amplitude and a few degrees in phase. You may attempt to measure and remove these baseline-dependent errors if you have good enough data from several calibrator sources. The noise in the calibrator-source images should approach the theoretical limit. The signal-to-noise ratio in the visibility data must be at least 100:1 on baseline-averaged data. Use the following sequence of tasks to solve for and remove the non-closing errors.

Run UVSUB to divide the calibration source visibility data by your best model.

> TASK 'UVSUB' ; INP ⏎	to review the inputs.
> INDISK *n1* ; GETN *ctn1* ⏎	to select the corrected visibility data as the input file.
> IN2DISK *n2* ; GET2N *cnt2* ⏎	to select your best image as the input model file.
> CMETHOD 'DFT' ⏎	to calculate model visibilities using DFT.
> OPCODE 'DIV' ⏎	to divide by the model.
> GO ⏎	to run the program.

Next run BCAL1. This task averages each baseline in the UVSUB output over all times and writes an ASCII output file. The output file contains one entry (row) for each baseline. The columns contain the antenna numbers (columns 1 and 2), the real and imaginary amplitudes and weight for IF channel 1 (columns 3, 4 and 5), and the real and imaginary amplitudes and weight for IF channel 2 (columns 6, 7 and 8). You would expect BCAL1 results to be similar for different calibration sources.

Run BCAL2 on your calibrator and program source visibility data sets. The INFILE is, of course, the ASCII file created by BCAL1.

10.5. Self-calibration and imaging

To self-calibrate and make images, follow this sequence of tasks:

1. VSCAL self-calibrates the *uv* data. VSCAL is a special version of ASCAL written for VLBI.
2. UVSRT sorts the VSCAL output into XY order.
3. MX images and cleans. MX is generally preferable to UVMAP and APCLN for VLBI data.

10.5.1. VSCAL

Special considerations for running VSCAL on VLBI data are:

1. During the first several iterations of VSCAL – MX, solve for phase corrections only, *i.e.*, set APARM(7) = 0. Switch on the amplitude corrections only after you have converged to a fairly good image.
2. On the first VSCAL iteration, use a point source as a starting model. Set BPARM(2) to the zero-spacing flux density and BPARM(1) = 1.
3. When solving for the amplitude corrections, use the TIMSMO parameter, initially setting the smoothing times to several hours.
4. APARM(10) = 0 prevents the flux density scale from floating. If you are confident in your *a priori* gain calibrations in VBANT, keep APARM(10) = 0.
5. Sources having a significant amount of extended structure will be poorly sampled due to the lack of short VLBI baselines. The extended structure will cause erroneous solutions for the antennas on the short baselines. Use APARM(1), APARM(2), and APARM(3) to down-weight the short baselines in the VSCAL solutions. For example, for a typical 18-cm VLBI run, you might set APARM(1) = 4000, APARM(2) = 0 and APARM(3) = 0.1. Remember, during the first few VSCAL – MX iterations, your image will probably be dominated by a compact feature.

10.5.2. MX

A few special points about using MX for VLBI data are:

1. Print out and read the MX EXPLAIN file. MX is a powerful and complicated task. It shouldn't be used blindly.
2. It's useful to make a dirty beam image in order to measure the beam full-width at half maximum. Set CELLSIZE so that there are at least three cells across the narrowest part of the beam and run PRTIM to examine the central part of the beam image in detail. Use the dirty beam dimensions to set BMAJ, BMIN, and BPA; fitted values from MX are unreliable for VLBI data.
3. For sources that are nearly unresolved on the short baselines, set the ZEROSP parameters. You will have to experiment with the zero spacing weight, ZEROSP(5). A weight that is too high will cause a DC offset in the CLEANed image.
4. Examine your CLEAN components with PRTCC after each MX run. It's important to note when the negative components first appear.
5. For VLBI data sets, uniform weighting gives the best results. Set UVWTFN =' ' Because the visibility record weights can vary by over a factor of 1000, set the UVBOX parameter to a large number to smooth out the visibility weights.

11. TIDYING UP AND EXITING FROM *AIPS*

Before exiting from *AIPS*, you should do several things to tidy up various kinds of debris you may have left in the system.

11.1. Backups

While processing and particularly just before exiting from *AIPS*, please delete as many of your own data sets as possible. Images and *uv* data may be backed up on tape in FITS format using the task FITTP. *Uv* data may also be backed up on tape in Export format using the task UVEXP. FITTP can write more than one *AIPS* file on tape in a single execution. For example, to backup all sorted *uv* files (class UVSRT), type

> TASK 'FITTP' ; INP $\subset_R$	to review the inputs.
> DOALL TRUE $\subset_R$	to specify that all files with the allowed name parameters are to be written.
> INNA ' ' ; INSEQ 0 $\subset_R$	to allow any name and sequence number.
> INDISK 0 $\subset_R$	to allow any disk.
> INTYP 'UV' $\subset_R$	to restrict to *uv* files.
> INCLASS 'UVSRT' $\subset_R$	to restrict to class UVSRT files.
> INP $\subset_R$	to check the inputs.
> GO $\subset_R$	to write the tape.

Then, to write all 3C123 files on disk 2 after the sorted *uv* data files, for example, type:

> INTYP ' ' ; INCLASS ' ' $\subset_R$	to allow any class and type.
> INNA '3C123' $\subset_R$	to restrict things to 3C123 files.
> INDISK 2 $\subset_R$	to restrict to disk 2.
> WAIT ; GO $\subset_R$	to have AIPS wait for the FITTP execution started above to finish and then to run FITTP with the new inputs.

On some systems (*e.g.*, VAX/VMS), a system-dependent global backup procedure may be available. This procedure will allow you to write all of your files on tape quickly, verify the tape, and then, optionally, delete the files. Type EXPLAIN ABACKUP and EXPLAIN ARESTORE for information. Such procedures are very nice, but they produce a highly system-dependent tape. A tape written on one VAX may not be readable on another, for example. Nevertheless, ABACKUP should be used (where available) instead of leaving data on disk for prolonged periods.

If space is short, data sets which remain on the disk for more than a week or two will be ruthlessly deleted by designated *AIPS* gorillas. If you wish to leave data on disk for an extended period of time, please notify the System Manager and bring an appropriate "gift".

11.2. Deleting your data

Please delete redundant images and data as soon as possible to preserve disk space for other users. It is tempting to work on many sets of data at the same time, but this generally takes a lot of disk space and users should limit the amount of data resident on disk to that which will be processed during the session.

A data set and all extension files can be deleted by:

> IND *n* ; GETN *ctn* ⏎ — where *n* and *ctn* select the disk and catalog numbers of the data set to be deleted.

> ZAP ⏎ — to do the deletion.

To delete data in contiguous slots from *n* to *m* in a catalog, set the INDISK and use the loop:

> FOR I = *n* TO *m* ; GETN I ; ZAP ; END ⏎

For massive deletions — the kind we hope you will use when you leave an NRAO site — use:

> ALLDEST ⏎ — to destroy all data files which are consistent with the inputs to ALLDEST.

And to clear all your messages and compress your file, after using PRTMSG to print any you want to keep, use:

> PRNUM -1 ; PRTASK ' ' ; PRTIME 0 ⏎ to do all messages.

> CLRMSG ⏎ — to do the clear and compress.

DO NOT DELETE OTHER USERS' DATA OR MESSAGES WITHOUT THE EXPLICIT PERMISSION EITHER OF THE OTHER USER OR OF THE SYSTEM MANAGER. Old data and messages belonging to any user (including you) may be deleted by the verb TIMDEST. The definition of "old" is set by your local *AIPS* Manager, who must be consulted about the rules for invoking TIMDEST.

A list of the software associated with deleting various files can be obtained at your terminal by typing:

> HELP DELETE ⏎

This listing is also given in § 14.

11.3. Sending comments to the *AIPS* programmers

Comments, suggestions and bug reports about any facet of *AIPS* can be entered into a file by typing:

> GRIPE ⏎

while in AIPS. Simply follow the directions and record your comment. On a weekly basis, "gripes" are collected from all NRAO computers and an acknowledgement is sent to each individual "griper." More detailed replies to the gripes are generated by the *AIPS* programming staff and are mailed to each griper later. Normally, a full reply will be given at the end of the current *AIPS* release cycle (*i.e.*, quarterly), but significant modifications and additions to *AIPS*, if warranted in response to gripes, may take longer to implement. In such cases, follow-up replies are also generated and sent to the gripers. The gripes and responses will become public (at NRAO), so write them clearly and in good taste. Complete copies of the gripes and responses are kept in notebooks in the *AIPS* Caiges at Charlottesville and the VLA.

Gripes can be registered outside of AIPS at monitor level by typing (in VAX/VMS when logged in as AIPS):

$@GRIPE ⏎

and following directions. The program invoked by this procedure, GRIPR, implements the same code as AIPS, but its control structure is much simpler. Try:

> HELP ⏎

after the @GRIPE for some information about the program's capabilities. The command to invoke this program may be different at some installations, particularly non-VAX/VMS ones.

11.4. Exiting

To exit from AIPS type:

> EXIT ↵

Please clean up any papers, tapes, etc. in the area around your terminal before you go.

11.5.1. Bananes rôties

1. Preheat oven to 375°.
2. Place 6 (peeled) bananas in a baking dish.
3. Sprinkle bananas with juice of 1/2 lemon.
4. Pour 2 tablespoons melted butter and 2 tablespoons dark rum over the bananas. Sprinkle with 2 tablespoons brown sugar.
5. Place in oven for 10 minutes.
6. Pour on 2 more tablespoons melted butter and 2 more tablespoons dark rum and bake for 5 minutes more.
7. Serve at once, spooning some sauce over each banana.

11.5.2. Going bananas with bananas

1. Garnish a baked ham or ham steak with bananas.
2. Make a quick, rich desert with bananas and cream.
3. Bananas are perfect for lunch boxes. They come in their own wrapper, are easy to eat and mess-less.
4. Slice a banana in half lengthwise, brush with melted butter and bake it until tender; serve it as a "vegetable" with roasted meats or fish. Very Caribbean.
5. Don't forget old favorites like bananas sliced over cereal, diced in pancake batter, or buried midst the ice cream in a banana split.
6. Slice and stir-fry bananas with carrots, tomatoes and ground beef for a super-quick main dish.

11.5.3. Golden mousse

1. Combine 1 cup mashed ripe bananas, 2 tablespoons orange juice, 1/4 cup shredded coconut, 3 tablespoons brown sugar, a few grains salt, and 1/8 teaspoon grated orange rind.
2. Whip until stiff 1 cup heavy cream.
3. Fold whipped cream into fruit mixture and turn into freezing tray. Freeze rapidly without stirring until firm.

12. PANIC SECTION

DON'T PANIC !!!!

On all computer systems things go wrong due to user error, program error, or hardware failure. Unfortunately, *AIPS* is not immune to this. The section below reviews several general problem areas and their generalized solutions. Refer to Appendix Z for the details appropriate to your particular computer system. Type HELP PANIC CR_R to review recent information on bugs, disasters and misfortunes within *AIPS* that have known solutions. Some well-known possibilities follow.

12.1. My printout is semi-infinite!!

To abort a job which is currently printing out, do the following:

1. Turn off power to the printer to conserve paper.
2. Type SPY CR_R to see if the offending program is still executing. If so, ABORT *taskname* CR_R will kill it.
3. Get into monitor mode (prompt $ on most machines) either on your terminal or another one.
4. Abort the current print job. See Appendix Z for the details for your machine (*e.g.*, § Z.1.7.1).
5. Turn the power back on for the printer.

12.2. The TEK screen will not print hard copy!!

1. Find the device which makes the hard copies. Is it turned on?
2. Check to see if there is paper in the device.
3. Power it down, then power it up again. Try the HARD COPY button on the TEK again.
4. If it still fails, call for hardware repair.

12.3. My data catalog has vanished!!

1. Are you connected to the right *AIPS* computer, if your site has more than one?
2. Are you logged in to the correct *AIPS* version? NRAO *AIPS* systems, in particular, may have separate data areas for the TST and NEW / OLD releases.
3. Have you set INDISK *et al.* correctly before running CAT? Type INP CAT CR_R to check. Is USERID not set to 0 or your user number?
4. Is a non-system disk mounted? Type FREE CR_R or exit from AIPS, and have the operating system tell you which disks are currently running. See Appendix Z for details.

12.4. My program crashed for lack of disk!!

Read this *COOKBOOK*, § 4.6 and Appendix Z for your computer system.

12.5. My tape refuses to read/write!!

1. Is the tape correctly loaded in the drive and is the drive "on line" (check the ON LINE light)?
2. Have you set the density correctly? Some drives need the density to be set by a switch, others have software control. Some try to read the tape and sense the density automatically. Be aware that some drives do not set the density until you actually read or write the tape. Under these circumstances, the density indication on the drive can be misleading. If in doubt, consult your local *AIPS* Manager about the meaning of the tape density indicator lights on the drive you are using.
3. Did you actually mount the tape in software from the *AIPS* level with the MOUNT verb (or, on those systems requiring it, from the Monitor level)?
4. Have you specified the INTAPE or OUTTAPE number to correspond with the drive you mounted the tape on?
5. Are you using the correct program to read the tape? If you are unsure of the format of a tape, use the task PRTTP to diagnose it for you. It will recognise any format that *AIPS* is able to read.
6. Are you writing to a completely blank tape? This fails sometimes. Or are you writing to an old tape which is new to you? In both cases, try specifying DOEOT FALSE $^{CR}_{R}$ and then rerunning the tape-writing program. If this fails, refer to the notes in Appendix Z (*e.g.*, § Z.1.7.3) for suggestions.
7. Has the drive been cleaned recently? Do *not* attempt to clean a drive yourself. Using the wrong cleaning fluid or cleaning the wrong parts of a drive can do serious damage. If you have any doubts, use another drive.
8. Is your tape defective? Tapes can lose oxide or become stretched, creased, or dirty, all of which will cause problems. Try using another tape, if possible.

12.6. My terminal has hung itself up!!

If your terminal is "dead", *i.e.*, refuses to echo characters typed on the keyboard or otherwise to show signs of talking to your computer, you have a problem. There are numerous possible causes.

1. Are you executing a long verb, *e.g.*, TIMDEST, REWIND, AVFILE, INSTAT? If so, be patient.
2. Are you executing some interactive TV or TEK verb which is waiting for input from the cursor or buttons? If so, provide the input.
3. Have you started a task with DOWAIT set to TRUE (+1.0)? If so, wait for the task to finish. Most tasks report their progress on the message monitor (or your terminal).
4. Is AIPS waiting while a tape rewinds or skips files or is it waiting to open some disk file currently being used by one of your tasks? Be patient.

5. Have you stopped output to your terminal accidentally by hitting the appropriate NO SCRL or other XOFF control sequence? If so, hit the XON control sequence. (These are CTRL S and CTRL Q, respectively on VAX/VMS.)
6. Can you abort AIPS at your terminal using the appropriate system commands (*i.e.*, CTRL Y on VAX/VMS)?
7. Do other terminals connected to the computer appear to be "alive"? If so, login on one of them and inquire about the status of your AIPS program and tasks. It might be necessary to stop them from your new terminal and then log back in on your old terminal. If this doesn't bring your terminal back to life, report the problem to the *AIPS* Manager.
8. If all terminals appear dead, then your computer has probably "crashed". Report the problem to your *AIPS* or System Manager. If you feel you must reboot the system, do so *only* after checking that all current users and the System Manager (if available) agree that that action is required.
9. If even a reboot fails, report the problem to the System Manager or hardware experts and go do something else. UNDER NO CIRCUMSTANCES SHOULD YOU ATTEMPT TO REPAIR THE TERMINAL, TELEVISION, DISK, CPU, OR OTHER HARDWARE DEVICES. Such repairs must be performed by trained personnel.

12.7.1. Banana nut bread

1. Cream 1 cup **sugar** and 1/2 cup **margarine** together.
2. Add 2 **eggs**, 2 cups **flour**, 1/2 teaspoon **salt**, and 1 teaspoon **baking soda** and mix thoroughly.
3. Add 1 cup chopped **nuts** (walnuts or pecans), 3/4 cup mashed **bananas**, and, lastly, 4 teaspoons **sour milk** and mix well.
4. Put in greased loaf pan.
5. Bake in 350° oven for 1 hour.

12.7.2. Orange gingered bananas

1. Combine in a small saucepan 1/4 cup **orange juice** and 1/2 teaspoon **cornstarch**. Cook and stir over medium heat until boiling.
2. Add 1/4 cup **orange juice**, 1 1/2 teaspoons **honey**, and 1 1/2 teaspoons chopped **crystallized ginger** and cook, stirring, until thoroughly heated.
3. Place 2 peeled, green-tipped **bananas** in a shallow baking dish and cover with sauce.
4. Bake at 350° about 15 minutes or until the bananas are tender (but not soft), basting with the sauce several times.

13. *AIPS* FOR THE MORE SOPHISTICATED USER

The program AIPS uses a computer language called *POPS* to communicate with the user. This language has numerous capabilities which have been mostly hidden in the previous sections. However, to use the full power of the *AIPS* system, the user will need to know more. This section of the *COOKBOOK* attempts to provide additional detail. There is also a four-volume *AIPS* Manual, the first volume of which is devoted to the information most pertinent to users. A copy should be available in your *AIPS* Caige — ask your System Manager if it is not.

13.1. AIPS syntax

Some niceties of using the syntax of AIPS are:

1. More than one expression can be put on a line. These expressions must be separated by a semicolon (;). Exceptions are RESTORE, GET, RUN and a few other "pseudoverbs" which, with their arguments, must stand alone. For example, GET APMAP ; INDISK = 1 ^{CR} will ignore the INDISK = 1. When in doubt, see the HELP files for the pseudoverb to find the restrictions on its use.
2. As in numerous systems, recognized keywords in AIPS do not need to be typed in full. One must type only enough of the leading characters so that the symbol is not ambiguous. The first four characters usually suffice. This "minimum matching" has been exploited throughout this *COOKBOOK*.
3. The parameter variables in AIPS are called "adverbs." They are assigned values by the equals verb (=), *e.g.*, INTAPE = 2 ^{CR}. However, the equals sign may be replaced by a space in almost all cases. The exception arises when the variable on the left is a subscripted array element and the expression on the right involves a unary minus or other function reference (*e.g.*, APARM(3) = -1 ; APARM(4) = SIN(X) ^{CR}).
4. Array adverbs are set to a constant value by putting a single value on the right hand side of the equals sign, *e.g.*, CELLSIZE = 1.5 ^{CR}. A list of values may be put in the array by putting the list on the right hand side of the = sign separated by commas (,), *e.g.*, LEVS = -1, 1, 2, 3, 6, 9 ^{CR}. The commas may be replaced by spaces in most cases. An exception occurs if an element is negative or some other arithmetic expression. Thus, SHIFT = -19 -2 ^{CR} will produce SHIFT = -21, -21.
5. Adverbs can be used in arithmetical expressions or set equal to other adverbs, *e.g.*, OUTNAME = INNAME ; OUTSEQ = 2.5 * INSEQ + 3.
6. Both upper and lower case letters may be used in AIPS. However, adverb character string values are converted to upper case before being stored and used.

These shortcuts permit the following AIPS command sequence, for example:

```
> INNAME '3C138' CR
> INCLASS 'IMAP' CR
> INSEQ 0 CR
> BLC 200,200 CR
> TRC 300,300 CR
> OUTNAME '3C138' CR
```

```
> OUTSEQ 5 ⏎
> GO PRTIM ⏎
```

to be shortened to:

```
> Inn '3c138' ; INC 'Imap' ; blc 200 ; Trc 300 ⏎        (Note use of upper and lower case.)
> OUTN INN ; OUTS INS + 5 ⏎
> go prtl ⏎                                              (Task name can be in either case, too.)
```

18.2. Data-file names and formats

The physical name of the data file is generated internally, is dependent on the type of computer, and need not concern the user. The user selects an image by specifying the disk number, the type of image (`'MA'` for images, `'UV'` for *uv* data), the user identification number, and the three parts of the user-assigned image designation. The last are:

1. Name — A string of up to 12 legal characters.
2. Class — A string of up to 6 legal characters.
3. Seq — A number between 0 and 9999.

Each of these parts corresponds to separate input adverbs called INNAME, INCLASS, and INSEQ (and their variations). The image name can be chosen arbitrarily by the user and sensible choices of names can relieve a lot of bookkeeping. Many programs will choose a reasonable image name if not specified by the user.

There is a set of conventions for the name adverbs which is used throughout *AIPS*. For example, INNAME `' '` means accept any image "name" having the user-specified class and sequence. INSEQ 0 means accept any image having the specified name and class or, if only one image is to be used, accept that image with the specified name and class having the highest sequence number. OUTNAME `' '` means use the actual INNAME. OUTCLASS `' '` means use the task name except for those tasks which write more than one output image. OUTSEQ 0 means use a sequence number one higher than any currently on disk (which match the name and class being used). The name and class strings also support "wild-card" characters both for input and output. This construct is very powerful, especially in tasks, such as FITTP, which can be instructed to use *all* images matching the user-specified name parameters. Type HELP INNAME ⏎ and HELP OUTNAME ⏎ for the details.

Only the array data, in the form of 2-byte scaled integers or 4-byte floating-point numbers, are stored in the data file. The header information is stored in a separate file for each image or *uv* data set. Directory information is stored in a special file, called the Catalog File. Each disk has such a file and it contains directory information for all images and *uv* data sets for all users on the disk pack. On some systems, each user will have his own catalog file on each of the disks. The Modcomp in Charlottesville has the former arrangement, while the VAXes at the VLA and Charlottesville have the latter.

Extension files may be associated with any image data file. Each image can have (in principle) up to ten types of extension files and up to 255 "versions" of each type. These subordinate files contain additional information associated with the image and are designated by a two-letter type code. `'HI'` is a history file, `'CC'` is a CLEAN components file, `'PL'` is a plot file, `'AN'` is an antennas file, `'SL'` is a slice file. In *AIPS*, an extension file associated with an image is uniquely specified by the usual file-naming adverbs plus the extension file type (adverb INEXT) and the version number (adverb INVERS). The default convention for INVERS is reasonable — on input, zero means the highest (*i.e.*, most recent) version and, on output, the highest plus one.

Array elements in an image are designated by their pixel values. If $M(i,j,k,l,m,n,o)$ is a seven-dimensional array, the (1,1,1,1,1,1,1) pixel will be associated with the lower left hand corner of the image. The i^{th} (first) coordinate increments fastest and is associated with a column in each plane of the image. The j^{th} (second) coordinate is associated with a row in each plane. The other coordinates allow the image to be generalized to cover up to seven dimensions, *i.e.*, "cubes" and the like. The two adverbs BLC for bottom left corner and TRC for top right corner allow the user to specify the desired subarray in up to seven dimensions. Whenever a sub-image is taken from an image, the pixel designation of any image element will usually change.

13.3. AIPS language

AIPS contains a basic set of symbols and keywords which are needed for facility in the language, as well as the symbols needed by the application code. A list of the basic symbols is given in § 14 and may be listed on your terminal by typing:

```
> HELP POPSYM ⏎
```

Here are some simple examples of uses of the AIPS language:

`> TYPE (2 + 5 * 6) ⏎`	32 is written on the terminal.
`> TYPE 'X =', ATAN (1.0) ⏎`	X = 45 is written on the terminal.
`> TYPE 'MAPNAME ', INNAME, INCLASS, INSEQ ⏎`	MAPNAME 3C138 IMAP 1 is written on the terminal.

13.4. Processing loops

The do-loop capability in AIPS uses the pseudoverbs FOR, TO, and BY and allows repetitive operations. Such loops are intended for use in "procedures," but, if the loop can be typed fully on one input line, it will work outside the procedure mode. The following example shows how to delete a series of images with the same name and class and with consecutive sequence numbers 1 through 10:

`> INNA 'TEST' ; INCL 'IMAP' ⏎`	to set (fixed) name parts.
`> INDI 1 ⏎`	to set (fixed) disk number.
`> FOR INSEQ = 1 TO 10 ; ZAP ; END ⏎`	to delete the files.

FOR loops must be terminated with an END. The following example shows how to delete every other file in a catalog with 20 entries:

```
> FOR INSEQ = 1 TO 20 BY 2 ; GETN(I) ; ZAP ; END ⏎
```

13.5. Procedures

Procedure building is a method of combining keywords in AIPS in any convenient way to obtain useful constructs. For complicated sequences, it will be easier to prepare procedures in RUN files (§ 13.6) than to prepare them in interactive *AIPS*. A procedure construct is given a name, with or without arguments, and then can be treated as a verb in AIPS. For an example, we will build a procedure which will allow the user to load an image on the TV, set the cursor, and fit for the maximum intensity. The following should be typed on the terminal:

> PROC MFIT (I) $^{CR}_{R}$	to define procedure MFIT with one argument I. (I and J are two dummy adverbs which are already defined in AIPS.)
: GETNAME(I) $^{CR}_{R}$	(Notice the prompt symbol : . This means that we are in the procedure-building mode.)
: TVLOD ; IMXY ; MAXFIT $^{CR}_{R}$	to load the image, produce and read the cursor, and fit the maximum near the cursor position when a TV button is pressed.
: RETURN $^{CR}_{R}$	to designate a return point in the procedure — normally not required at the end of a procedure unless a value is to be left on the stack, *i.e.*, a function.
: FINISH $^{CR}_{R}$	to designate the end of the procedure-building mode and to get back into the normal (prompt >) mode.

>

When you are typing a procedure into AIPS, the code is compiled as it is typed. Most syntax errors are spotted immediately and the user is unceremoniously dumped out of procedure mode. However, all lines written before the detected error are kept and the procedure editor can be used to carry on.

Rather limited procedure editing capabilities are available. However, for making procedures which are longer than about five lines, the use of permanent storage files in the "RUN" area is recommended. This is discussed in § 13.6 below.

To list procedure MFIT, type:

> LIST MFIT $^{CR}_{R}$

This will produce the following:

```
1    PROC MFIT(I)
2    GETNAME(I)
3    TVLOD ; IMXY ; MAXFIT
4    RETURN
5    FINISH
```

The procedure is identical to that typed, with the addition of line numbers.

Procedure editing is done on a line by line basis. To edit line 2 in the above procedure, type:

> EDIT MFIT 2 $^{CR}_{R}$	to enter Procedure editing mode.
; GETNAME(I) ; TVLOD $^{CR}_{R}$	(Notice prompt symbol ; for procedure-editing mode.) This change replaces the old line 2 adding a TVLOD.
; IMXY ; MAXFIT $^{CR}_{R}$	to add a line between the changed line 2 and old line 3.
; GETNAME(I+1) $^{CR}_{R}$	to add yet another line after 2.
; ENDEDIT $^{CR}_{R}$	to terminate procedure editing.

> LIST MFIT $^{CR}_{R}$

The listing of the modified procedure will give:

```
1    PROC MFIT(I)
2    GETNAME(I) ; TVLOD
3    IMXY ; MAXFIT
4    GETNAME(I+1)
5    TVLOD ; IMXY; MAXFIT
6    RETURN
7    FINISH
```

To delete lines *n* through *m* from a procedure, type:

> ERASE *xxxxxxx n* : *m* CR where *xxxxxxx* is the name of the procedure.

To insert one or more lines between lines 3 and 4 of a procedure, type:

> EDIT *xxxxxxx* 3.5 CR

; *(Type additional lines as needed.)*

; ENDEDIT CR

Notice that the lines are renumbered after any EDIT or ERASE. Use LIST to determine the new line numbers.

The pseudoverb MODIFY allows the user to modify characters within a line of a procedure in order to correct that line or to change its meaning. The grammar is:

> MODIFY *proc-name line-number* where *proc-name* is the name of the procedure and *line-number* is the line number in the procedure as shown by LIST.

MODIFY begins by showing the existing line with a ? as a prefix. Then it prompts for input with a ? To keep the character of the original line immediately above the cursor, type a blank (space-bar). To delete that character, type a $ (dollar-sign). To replace that character, type the new character (to get a new blank character, type an @ sign). Insertions complicate things. To insert text prior to the character immediately above the cursor, type a \ followed by the desired text followed by another \. You may continue to MODIFY the remainder of the line, but you must remember that the current character position in the old line is to the left of the current cursor position by the number of inserted characters (including the 2 \'s). MODIFY will display the resulting line of code after you hit a carriage return (CR) and does not change the line number. Example:

```
> MODIFY  ED  2  CR
?TYPE 'THIS IS EDS PROC'
?              MY@\NEW\     @FOR@EXAMPLE'  CR
TYPE 'THIS IS MY NEW PROC FOR EXAMPLE'
> MODIFY  ED  2  CR
?TYPE 'THIS IS MY NEW PROC FOR EXAMPLE'
?                 $$$$    \EDURE,\  CR
TYPE 'THIS IS MY PROCEDURE, FOR EXAMPLE'
```

More information about procedure building and editing can be found by typing:

> HELP PROCEDUR CR

Procedure creation and editing uses up the limited memory of the *POPS* processor. When the memory is gone, the message BLEW CORE! will appear and you can do no more procedure writing without starting over (*i.e.*, RESTORE 0 CR). CORE CR will tell you how much memory is left.

The procedure MFIT can be executed by:

> MFIT(*n*) CR where *n* is the slot number of the appropriate image.

(It is assumed that the correct disk unit number has already been set.) This procedure can also be part of another procedure or put in a loop. For example:

> FOR I= 1 TO 10 BY 2; MFIT(I) ; END CR

will load the TV and fit the maximum for the first ten images on the appropriate disk.

All of the syntax available in AIPS is available for use inside procedures *except for certain pseudoverbs.* These "prohibited" pseudoverbs include SAVE, GET, STORE, RESTORE, PROCEDURE, EDIT, ENDEDIT, MODIFY, LIST, CORE, and SCRATCH. Others do not make much sense in procedures, including MSGKILL, DEBUG, and ABORTASK. Other pseudoverbs are, however, particularly useful in procedures. These include TGET, TPUT, and GO.

There are a number of verbs which are extremely useful in procedures. To set the image name adverbs to those visible on the TV, use TVNAME. When GETN accesses an empty slot, an error condition is raised and the procedure dies. To handle this error condition in your procedure, use EGETN *n* instead and test the adverb ERROR which will be "true" if the slot is empty. Some tasks require image-data dependent inputs. To help handle this in general procedures, the verb GETHEAD allows all header parameters to be fetched into adverbs. Type EXPLAIN GETHEAD ⏎ for details. There are numerous arithmetic functions, useful looping constructions, and powerful methods of building arithmetic, logical, and string expressions in *POPS*. Type HELP POPSYM ⏎ or see § 14 for details.

Once a procedure is written and edited, it can be stored in a SAVE file for later use. Procedures are lost whenever another GET file is obtained. Procedures can also be stored more permanently in RUN files which are described below. To obtain a listing of the names of all of the procedures which are currently in your AIPS environment, type:

> HELP PROCS ⏎

This will produce a listing of internal AIPS procedures as well as those produced by the user.

13.6. RUN files

The writing and editing of procedures more than about 10 lines in length is cumbersome in AIPS because of its primitive editor. Long procedures are best written in text files at monitor level in the computer where there are good editing facilities. These files can then be transferred to AIPS very simply using the RUN file facility.

Any set of commands which can be typed on the terminal can be stored in a text file. At the present time, the computer's own text editor must be used to generate this file so this part of the process is machine dependent. The text files are generally stored in an area designated by the word RUN. Thus, in VAX/VMS these text files are stored in a subdirectory called [AIPS.RUN] under the AIPS logon. On most systems, the file name must be of the form *Accccc.nnn*, where *A* is any alphabetic character, *ccccc* are any 0–7 alphanumeric characters, and *nnn* is your user number expressed in 3 hexadecimal characters with leading zeros. *AIPS* programmers also provide a few RUN files for general use in each release. These are stored under user number 1 and are automatically available to all users. See Appendix Z for some helpful information applying to NRAO *AIPS* systems (*e.g.*, § Z.1.8).

The first line of a RUN file is ignored by AIPS. You should type comments into your RUN files to remind you what they are doing. The first line, any line which begins with an * in column one, and all text following a $ sign (in any line) are comments which will not be executed by AIPS.

To use a procedure, type:

> RUN *xxxxxx* ⏎	to read and execute file *xxxxxx* in AIPS. The text will be listed as it is read.
>	continue AIPS in normal mode.

The file *xxxxxx.nnn* or, if it does not exist, *xxxxxx.001* will be executed by the above command. Note that minimum match also applies to RUN file names.

RUN files are a convenient place to store procedures which are used commonly in AIPS since the code can be accessed in AIPS simply by typing RUN *whatever* ⏎. RUN files are also useful for storing the commands associated with a convenient set of tasks, such as loading an image, sorting the data, and making and cleaning the image. All facilities in AIPS such as GET, SAVE, and TGET can be used in RUN files.

13.7. Batch jobs

AIPS has a batch processor which can be used to run jobs outside interactive *AIPS*. The job consists of a set of *AIPS* instructions which do not need user interaction. This excludes the TV, the Tektronix, and the tape-drive oriented tasks and verbs. The Array Processor can be used in batch although interactive users generally have priority on the AP. RUN files may be used in batch jobs and, since the batch editor facility is also primitive, they are particularly attractive for use in batch.

The instructions to be executed in the batch processor are prepared in a "workfile." The workfile can be made while in AIPS and detailed instructions are given by typing:

```
> HELP BATCHJOB ⏎
```

A simple example is given here:

`> BATQUE = 2 ; BATCLEAR ⏎`	to select queue 2 and clear its workfile. There are generally two queues and, if so, jobs which use the array processor cannot be run in queue 1.
`> BATCH ⏎`	to enter batch preparation mode.
`< TASK = 'UVSRT' ⏎`	(Notice < prompt. Begin typing as in AIPS.)
`< INN = '3C16' ; INCL = 'UVDATA' ⏎`	
`< INSEQ 1 ; OUTN INN ; OUTCL INCL ⏎`	
`< OUTSEQ = 0 ; SORT = 'XY' ⏎`	
`< GO ⏎`	(Batch AIPS always waits for a task to finish before continuing.)
`< RUN XXXXX ⏎`	(RUN files are good to use in a batch job.)
`< GO ⏎`	
`< ENDBATCH ⏎`	to leave batch preparation mode type in ENDBATCH spelled out in full.
`>`	(Resume normal interactive processing.)

To list a batch file, type:

```
> BATFLINE = 0 ; BATLIST ⏎
```

To edit line *n* in a batch file, type:

`> BATEDIT n ⏎`	
`< put text here ⏎`	to replace old line *n*.
`< some more text ⏎`	to insert more commands between old lines *n* and *n*+1.
`< ENDBATCH ⏎`	(spelled out in full.)
`>`	(Resume normal interactive processing.)

As with procedures, if *n* is an integer, the existing line *n* is overwritten with the line or lines typed before ENDBATCH. If *n* is not an integer, the new lines are simply inserted between lines *n* and *n*+1. BAMODIFY provides, for workfiles, the same functions as MODIFY does for procedures.

Finally, the workfile can be submitted to the batch processor by typing:

```
> SUBMIT ⏎
```

The instructions are sent to a checking program which checks that the input is free of obvious errors. All RUN files are expanded and checked. If Checker (the task AIPSC*m* where *m* is some hexadecimal number < F) approves, the job goes into the *AIPS* job queue, which is managed by QMNGR*n*. If you change your mind, the job can be removed from the queue and returned to the workfile with the verb UNQUE.

To submit a batch job outside of the AIPS interactive program, log in to the AIPS area and then type:

$ @BATER ↵

and generate the batch work file as shown above. It can be submitted without ever going into AIPS. Note that a different version of this command syntax may be required on your non-VAX/VMS computer.

Batch has several limitations. First, devices which require interactive use (TV device, Tektronix device, and the tape drives) cannot be used in batch. Also, batch uses a different set of TPUT and TGET files. Thus, a TGET in batch does not get the adverbs from your last interactive use of the specified task. However, the AIPS facilities GET and SAVE are particularly useful for batch. You can use interactive AIPS to set up and test set(s) of procedures and adverb values and SAVE them in named files. These files may then be recovered by batch for the routine processing of large sets of data. This is considerably more convenient than using the batch editor. Note, however, that SAVE / GET files may become obsolete with new *AIPS* releases.

At present, batch jobs are run after a short delay, on a first-come, first-served basis. However, batch jobs which use the array processor are forbidden in batch queue number 1 (if there is > 1 batch queue) and may be excluded at some times of day by your *AIPS* Manager in the other queues. After your job has been submitted successfully, type:

> QUEUES ↵ to list jobs in the queue.

Note the SUBMIT TIME for your job. It will not start before that time. The messages generated by your batch job will be printed automatically. They are kept in your message file, however, and can be reprinted or examined later via PRTMSG with PRNUMB set to the *AIPS* number of the batch queue.

13.8. Writing your own programs with *POPS*

Users may wish to write their own programs for *AIPS*. To generate an *AIPS*-standard FORTRAN program requires considerable additional information and labor (see HELP NEWTASK ↵ and the *Going AIPS* manuals). Fortunately, less elaborate programs may be constructed as procedures using existing verbs and tasks.

Consider a more complicated procedure as an example. (This example is presented as if it were typed into an interactive AIPS. However, users will probably prefer to prepare such complicated procedures in RUN files.) We wish to determine the average value and rms scatter at any pixel location in a set of *n* images. We shall demand that the *n* images all have the same INNAME and INCLASS with sequence numbers between 1 and *n*. The RENAME verb can be used to name the images appropriately. We could call this procedure:

```
AVGRMS (PIXXY, N, AVG, RMS)
```

where

PIXXY	is the pixel location in the images,
N	is the number of images,
AVG	is the average value at the pixel location, and
RMS	is the rms value at the pixel location.

The array adverb PIXXY is a standard AIPS adverb, but the variables N, AVG, and RMS are unknown to AIPS. These must be defined before we can write the procedure AVGRMS. This is done by a short dummy procedure which we will call DAVGRMS:

> PROC DAVGRMS ↵ to define dummy procedure.

: SCALAR N, AVG, RMS ↵ to define scalar adverbs.

: FINISH ↵ to exit from dummy procedure.

Now begin the procedure AVGRMS:

```
> PROC AVGRMS (PIXXY, N, AVG, RMS) ↵R        to enter procedure building mode.
:  SCALAR SUM, SUM2 ↵R                       to define more variables.
:  ARRAY VAL(20) ↵R                          to define an array.
:  RMS = 0 ; SUM = 0 ; SUM = 0 ↵R            to zero some variables.
:  FOR INSEQ =1 TO N ↵R                      to begin summing loop.
:      QIMVAL ↵R                             to get pixel value at PIXXY in image INNAME INCLASS INSEQ.
:      VAL(INSEQ) = PIXVAL ↵R                to save pixel value (placed in PIXVAL by IMVAL) in our array.
:      SUM = SUM + PIXVAL ↵R                 to sum for averaging.
:      SUM2 = SUM2 + PIXVAL * PIXVAL ↵R      to sum for rms.
:      END ↵R                                to mark end of FOR loop.
:  AVG = SUM / N ↵R                          to get average value.
:  IF N > 1.5 THEN ↵R                        to check if N > 1.
:      RMS = SQRT((SUM2 - N*AVG*AVG) / (N * (N-1))) ↵R   to calculate rms if N > 1.
:      END ↵R                                to mark end of IF clause.
:  TYPE 'AVG=',AVG,'RMS=',RMS,'AT PIXEL',PIXXY ↵R
:  TYPE ' # ',' VAL ',' ERROR ' ↵R           to print a heading.
:  FOR INSEQ = 1 TO N ↵R                     to begin another loop.
:      SUM = AVG - VAL(INSEQ) ↵R             to get residual.
:      TYPE INSEQ, VAL(INSEQ), SUM ↵R        to print data and residual.
:      END ↵R                                to mark end of FOR loop.
:  FINISH ↵R                                 to return to regular AIPS mode.
>
```

The above procedure could be run in the following way. First fill in the adverbs INNAME, INCLASS and PIXXY with the desired values. Then type:

```
> AVGRMS (PIXXY, n, AVG, RMS) ↵R
```

where *n* is the number of images to average. The calculation of the average and rms will take place and be written on the terminal and on the message file. This procedure could be used by another procedure. Suppose we wanted to determine the average and rms of the pixels within a rectangular area. If we set BLC and TRC in the usual way to define the rectangular boundary, then the procedure:

```
> PROC AVGARRAY (BLC, TRC) ↵R               to define new proc.
:  FOR I = BLC(1) TO TRC(1) ↵R               to loop over x-coordinate
:      FOR J = BLC(2) TO TRC(2) ↵R           to loop over y-coordinate.
:          PIXXY = I , J ↵R                  to set pixel coordinates for AVGRMS.
:          AVGRMS (PIXXY, N, AVG, RMS) ↵R
:          END ; END ↵R                      to end y loop, then x loop.
:  FINISH ↵R                                 to end the proc., RETURN not needed.
>
```

will calculate the average value and rms at this array of pixel locations. Please note that this is just an example. The verb IMSTAT performs this function much more efficiently.

14. CURRENT *AIPS* SOFTWARE

The complete list of software in *AIPS* is kept up-to-date in certain special HELP files. The list of all such special HELP files can be obtained on your terminal by typing:

> HELP CR_R

Among these is a very long alphabetical list of all tasks, verbs, pseudoverbs and AIPS built-in procedures. This may be scanned at the terminal by typing:

> HELP INDEX CR_R

Shorter lists of the software, grouped according to usage, are also given in special HELP files. These lists are presented below, followed by the long INDEX file. The name of the special HELP file is shown in bold type above each group. Each line within a group lists a task (T), verb (V), pseudoverb (PV and pV), or procedure (P) with a very brief description of its function. Pseudoverbs come in two flavors: those that act roughly like verbs (pV) and those that must be treated specially (PV), *i.e.*, that must appear alone on a line or only in certain contexts.

> HELP *name* CR_R

where *name* is one of these entries, will give more useful information about that task, verb, pseudoverb, or procedure. References to sections in the *AIPS COOKBOOK* are given to the right of each line below.

CURSOR

Type: Interactive use of cursor on TV and TEK

IMPOS	V	Find absolute position of cursor on TV	§5.3, 6.3
TKPOS	V	Find absolute position of cursor on TEK	§
IMXY	V	Find image pixel location of cursor on TV	§8.3.1
TKXY	V	Find image pixel location of cursor on TEK	§
IMXY;IMVAL	P	Find intensity at cursor location on TV	§
TKXY;IMVAL	P	Find intensity at cursor location on TEK	§
TVPOS	V	Find TV pixel location of cursor on TV	§
TVNAME	V	Display name of image under the cursor	§13.5
TKVAL	V	Find intensity from slice plot using cursor	§
CURBLINK	V	Blink cursor or return to steady cursor	§
CURVALUE	V	Display location and intensity under cursor	§5.3, 6.3
TVSTAT	V	Determine statistics in TV-selected areas of image	§6.3, 8.2
TVWINDOW	V	Set BLC and TRC using cursor, graphics display	§6.3
TVBOX	V	Set cleaning boxes with graphics display	§5.1.1, 6.3
REBOX	V	Reset cleaning boxes with graphics display	§6.3
TVSLICE	V	Set slice end points with graphics display	§6.3, 7.5
TKBOX(I)	P	Set I^{th} cleaning box using cursor on TEK	§
TKNBOX(I)	P	Set I cleaning boxes using cursor on TEK	§
TKWIN	P	Set BLC and TRC using cursor on TEK	§
SETXWIN(I,J)	P	Set rectangular box $I \times J$ around cursor location	§6.3

TVINTER

Type: Interactive use of the TV display

Loading of TV image

TVINIT	V	Initialise TV completely	§
TVRESET	P	Reset TV functions leaving image on	§
TVCLEAR	V	Clear TV channel	§
TVLOD	V	Load an image on a TV channel	§9.11, 13.5
TVALL	P	General purpose interactive TV display	§6.2, 9.6
TVMOVIE	V	Load planes of cube, run movie	§9.6
TVHLD	T	Load high precision image, do equalization	§
TVSLV	T	Display TVSLD output on TV	§9.1

Image transfer function (contrast) and wedges

TVTRANS	V	Modify transfer function for image	§6.2.2
OFFTRANS	V	Reset normal transfer function for image	§
IMWEDGE	V	Display image wedge from image min to image max	§
TVWEDGE	V	Display image wedge, range as loaded	§
IMERASE	V	Erase image (or wedge) from TV	§
TVALL	P	Provides contrast and wedges	§6.2, 9.6
TVFIDDLE	V	Same as TVALL except no TV load	§6.2
TVHUEINT	V	Use 2 images for hue/intensity with enhancement	§9.1
OFFHUINT	P	Clear the hue/intensity display	§
TVLUT	V	Set n-point linear transfer function	§
TVMLUT	V	Set many point linear transfer function	§
TVHXF	T	Interactive histogram equalization of image	§

Zooming

TVZOOM	V	Activate hardware zoom on TV	§
OFFZOOM	V	Deactivate hardware zoom on TV	§9.1, 9.6
TVALL	P	Provides zooming capability	§6.2, 9.6
TVFIDDLE	V	Same as TVALL except no TV load	§6.2

Scrolling

TVSCROL	V	Activate hardware scroll on TV	§
OFFSCROL	V	Deactivate hardware scroll on TV	§

Blinking

TVBLINK	V	Automatic blink between two channels	§
TVMBLINK	V	Manual blink between two channels	§
TVMOVIE	V	Load planes of subimage, run movie	§9.6
REMOVIE	V	Re-run preloaded movie	§9.6

Roaming—using several channels for one image

TVROAM	V	Load image on several channels, initiate scroll and zoom	§
SETROAM	V	Fill in roam parameters, then roam	§

REROAM	V	Same as TVROAM with no loading	§
OFFROAM	P	Initialize TV after roaming	§

Other displays

CURVALUE	V	Display pixel position and value under cursor	§5.3, 6.3
TVNAME	V	Print name of image under the cursor	§13.5
TVLABEL	V	Label TV image with coordinate axes	§
TVWLABEL	V	Label wedge with intensities	§
GRCLEAR	V	Clear graphics channel(s) of plots or labels	§

TVGEN

Type: General TV functions for loading and display

TVINIT	V	Initialize TV completely	§
TVCLEAR	V	Erase entire TV channel	§
TVLOD	V	Load image into TV channel	§9.11, 13.5
TVON	V	Turn on TV channel	§6.2, 9.1, 9.6
TVOFF	V	Turn off TV channel	§9.1, 9.6
TVRESET	P	Reset TV functions leaving image on	§
IMERASE	V	Erase TV image on portion of a TV channel	§
WEDERASE	V	Erase wedge image on portion of TV channel	§
GRCLEAR	V	Erase entire graphics channel	§
GRON	V	Turn on specified graphics channel	§
GROFF	V	Turn off specified graphics channel	§
TVALL	P	General purpose interactive TV display	§6.2, 9.6
TVFIDDLE	V	Same as TVALL with no TV load	§6.2
TVMOVIE	V	Load planes of cube, run movie	§9.6

TVCOLOR

Type: Using color on the TV

TVRESET	V	Reset TV functions leaving image on	§
TVPSEUDO	V	Enable several types of pseudo coloring	§6.2.1
OFFPSEUD	V	Disable pseudo coloring	§
TVALL	P	Load, color, contrast and zoom TV display	§6.2, 9.6
TVFIDDLE	V	Like TVALL except no load	§6.2
TV3COLOR	V	Produce false color display for 3 image planes	§9.1
TVHUEINT	V	Produce Hue-intensity display from 2 image planes	§9.1
OFFHUINT	P	Turn off the hue-intensity display	§
RGBMP	T	Create 3 planes for false color from data cube	§9.11
IMLHS	T	Produce Intensity/Hue/Saturation image from 3 images	§

GENERAL

Type: General *AIPS* utilities

HELP	pV	Provide information for verbs, tasks, adverb	§3.4.1, 4.8, all
EXPLAIN	pV	List help information on printer	§4.8, all
INPUTS	pV	Display current parameters for a task or verb	§4.8
INP	pV	Same as INPUTS except not written on message file	§4.4, all
DISKU	T	List by user all disk space used in *AIPS*	§4.6
FREESPAC	V	List total available disk space in *AIPS*	§4.6, 12.3
SAVE	pV	Save AIPS environment in a semi-permanent file	§4.4
GET	PV	Retrieve a SAVE file	§4.4, 13.1
TGET	pV	Get adverb values used in previous task execution	§4.4, 4.5
TPUT	pV	Save adverb values for later retrieval by TGET	§4.4, 4.5
STORE	pV	Save AIPS environment in a temporary file	§4.4
RESTORE	PV	Retrieve a STORE file	§4.4, 13.1
SGINDEX	V	List all SAVE / GET files	§
TGINDEX	V	List all TPUT / TGET entries	§
PRINT	V	Print the following keyword value(s)	§6.1, 9.9
TYPE	V	Print the following keyword value(s)	§13.3, 13.8
READ	V	Read values of the following keyword(s)	§
PRTMSG	V	Print the messages in the user's message file	§3.2, 4.1, 13.7
CLRMSG	V	Delete messages from the user's message file	§4.1, 11.2
MSGKIL	pV	Stop all messages from being saved	§
PRTHI	V	Print the history file of an image	§3.4.1, 3.4.2, 4.3, 5.1.1
RESTART	V	Restart AIPS	§
EXIT	V	Exit from AIPS	§4.6, 11.4
PASSWORD	V	Enter new *AIPS* password for login user	§
GO	pV	Starts a shed task and passes parameters to it	§4.5, all
ABORTASK	pV	Aborts specified task, removes scratch files	§7.1, 12.1
SPY	V	Inquire which tasks are active	§4.6, 12.1
WAITTASK	pV	Suspends AIPS until specified task is done	§4.5, 6.1, 9.9
PROC	PV	Begin building a procedure (See HELP PROC)	§13.5, 13.8
PROCEDUR	PV	Begin building a procedure (See HELP PROC)	§
RUN	PV	Execute a text file generated in the RUN area	§13.1, 13.6
BATCH	V	Begin building a batch file (See HELP BATCHJOB)	§13.7
SUBMIT	V	Submits specified batch queue into execution	§13.7
GRIPE	V	Enter a complaint/suggestion in the Gripe file	§11.3
GRINDEX	V	Index of all entries in the current Gripe file	§
GRLIST	V	List any one of the entries in the Gripe file	§
GRDROP	V	Drop one of your entries in the Gripe file	§

CATINFO

Type: Dealing with the image catalog

CATALOG	V	List all images in catalog consistent with inputs	§3.2, 4.2
MCAT	V	List all images in catalog	§3.4.1, 4.2
UCAT	V	List all *uv* data sets in catalog	§3.2, 4.2
PCAT	V	List all files in "half" of catalog	§
CLRSTAT	V	Clear WRIT and READ statuses from catalog	§4.2.3
IMHEADER	V	List the contents of an image header	§4.2.4, 9.2
QHEADER	V	List summary of image header	§4.2.4
TPHEAD	V	List image header from tape	§6.1
EGETNAME	V	Fill INNAME *et al.* from slot number of INDISK	§13.5
GETNAME	V	Fill INNAME *et al.* from slot number of INDISK	§4.2.2, all
GET2NAME	V	Fill IN2NAME *et al.* from slot number of IN2DISK	§4.2.2, all
GET3NAME	V	Fill IN3NAME *et al.* from slot number of IN3DISK	§4.2.2, 7.2
CLRNAME	V	Fill INNAME *et al.* with null values	§3.4.1, 4.2
CLR2NAME	V	Fill IN2NAME *et al.* with null values	§
CLR3NAME	V	Fill IN3NAME *et al.* with null values	§
TVNAME	V	Set INNAME *et al.* to that under TV cursor	§13.5
RECAT	V	Compress catalog file renumbering entries	§4.3
RENAME	V	Rename file	§4.3, 13.8
RENUMBER	V	Renumber a catalog entry	§4.3
RESCALE	V	Rescale file using header scale factor and offset	§
SGINDEX	V	List all SAVE / GET files	§
TGINDEX	V	List all TGET files	§
EXTLIST	V	List extension file information	§7.4, 7.5
CORFQ	T	Correct *uv* data header for incorrect frequency	§
AXDEFINE	V	Redefine axis values in header	§
CELGAL	V	Switch between Galactic and Celestial coords	§9.5
ADDBEAM	V	Insert clean beam values in header	§
GETHEAD	V	Fetch header value into adverb	§13.5
PUTHEAD	V	Put adverb value into header	§

PL2D

Type: Two-dimensional displays and plot files

CNTR	T	Contour diagram → plot extension file	§7.1, 7.9
GREYS	T	Greyscale image plus contour → plot file output	§7.3, 7.9
PCNTR	T	Contour plus polarization line vector → plot file output	§7.2, 7.9
PFPL1	T	Plot task paraform: x and y same as image	§
PLCUB	T	Plot rows of image on a 2-D grid	§7.6, 9.11, 9.12
PLROW	T	Plot rows of image w user-controlled spacing	§7.6, 7.9, 9.11
PROFL	T	Profile (perspective) diagram → plot file output	§7.6, 7.9

TKPL	T	Display plot file on the TEK	§5.2.1, 7.1
PRTPL	T	Display plot file on the line printer	§5.2.1, 7.1
QWKPL	T	Display plot file on the line printer (VAX only)	§
TVPL	T	Display plot file on the TV	§7.1
QMSPL	T	Display plot file on QMS laser printer	§5.2.1, 7.1, 7.6
EXTLIST	V	List extension files associated with an image	§7.4, 7.5
EXTDEST	V	Destroy extension files	§7.4, 7.5
TVLOD	V	Load an image onto specified TV channel(s)	§9.11, 13.5
TVALL	P	Load an image onto TV and interact with display	§6.2, 9.6
TVROAM	V	Load a large image onto several TV channels and roam	§
PRTIM	T	Print image intensities on the line printer	§5.1.3

SL1D

Type: One-dimensional displays and analysis

Create slice and display

SLICE	T	Store intensity along a line segment in extension file	§7.5, 9.11
SL2PL	T	Convert slice file to a plot file for general display devices	§7.5, 7.9, 8.3.3
TKSLICE	V	Initialize TEK and display slice file	§7.5, 8.3.3, 9.11
TKASLICE	V	Superpose slice file to existing display on TEK	§7.5
PLCUB	T	Plot rows of image on a 2-D grid	§7.6, 9.11, 9.12
PLROW	T	Plot rows of image w user-controlled spacing	§7.6, 7.9, 9.11
XPLOT	T	TEK plots of image rows	§7.6, 9.11

Fitting model to slices

XGAUS	T	Fit Gaussian components to rows of cube	§8.3.4, 9.11
SLFIT	T	Fit model guess to a slice file	§8.3.3
TKSET	V	Use TEK cursor to guess initial Gaussian model for slice	§8.3.3
TK1SET(I)	V	Use cursor on TEK to change guess of I^{th} component	§
TKGUESS	V	Initialize TEK and display guess	§
TKAGUESS	V	Superpose guess to existing plot on TEK	§8.3.3
TKMODEL	V	Initialize TEK and display fitted model	§
TKAMODEL	V	Superpose fitted model to existing display on TEK	§8.3.3
TKRESID	V	Initialize TEK and display fitted model-slice	§
TKARESID	V	Superpose fitted model-slice to existing display on TEK	§8.3.3

Utilities

EXTLIST	V	List extension files associated with an image	§7.4, 7.5
EXTDEST	V	Destroy extension files	§7.4, 7.5
PFPL2	T	Plot task paraform: x is a slice of image	§
PFPL3	T	Plot task paraform: x and y no relation to image	§

UVPR

Type: *uv* processing

Index, write and read *uv* data tapes

EXIND	T	List index from an EXPORT tape	§3.1
EXFND	T	List data from an EXPORT tape	§
EXPND	T	Copy EXPORT tape, one source/file	§
FILLR	T	Read VLA ModComp format *uv* data tapes	§
FITTP	T	Copy *uv* data from disk to *uv*-FITS tape	§11.1, 13.2
PRTTP	T	Print summary of tapes, any recognized format	§3.1
UVEXP	T	Copy from disk *uv* data to an EXPORT tape	§11.1
UVLOD	T	Copy *uv* data from EXPORT or *uv*-FITS tape to disk	§3.2, 9.2, 10.1.1
WSLOD	T	Load *uv* data from WSRT tape to *AIPS* disk format	§

uv data manipulation

DBCON	T	Concatenate *uv* data bases	§10.2
BLOAT	T	Convert data to correct spectral-line form	§
UVCOP	T	Copy part of a *uv* data base	§4.7, 9.1
UVAVG	T	Average or merge BT or TB sorted *uv* data	§10.2, 10.3
AVER	T	Average *uv* data in time	§
DESCM	T	Reduce size of a *uv* file	§
UVSRT	T	Sort *uv* data on disk	§3.3, 9.1, 10.2
UVFLG	T	Flag selected *uv* data	§5.3
CLIP	T	Flag *uv* data above a given intensity	§5.3
UVFIX	T	Recompute *u*, *v*, and *w* from antenna data	§

uv processing

APMAP	T	Make image(s) from EXPORT tape	§
UVMAP	T	Make image(s) from disk file	§3.4, 9.3
UVSUB	T	Subtract point components from *uv* data	§5.3, 9.7, 10.4.4
UVMOD	T	Rescale *uv* data while adding model component	§8.4.3
ASCAL	T	Self-cal *uv* data	§5.2, 9.1
ASCOR	T	Apply ASCAL results	§5.2.3, 9.1
BCAL1	T	Find baseline-based *uv* calibration	§10.4.4
BCAL2	T	Apply baseline-based *uv* calibration	§10.4.4
VBANT	T	Apply T sys and gain corrections for VLBI	§10.4
VBFIT	T	Global fringe fitting for VLBI	§10.3
VBCOR	T	Apply results of VBFIT	§10.3
VBCAL	T	Apply antenna scale factors to data	§
FFT	T	Make Fourier transform of an image	§5.3

Display of *uv* data

PRTUV	T	Print selected *uv* data	§10.3
UVFND	T	Print selected *uv* data	§5.3
PRTAN	T	Print antenna files	§
PRTGA	T	Print gain files	§
UVPLT	T	Plot *uv* data in a variety of ways → plot file	§3.4.4, 5.2.1, 5.3

VBPLT	T	Plot data vs model 1 baseline/plot	§10.3
		Miscellaneous *uv* processing	
CORER	T	Calculate *uv* correlator statistics	§5.3
CORFQ	T	Correct *uv* data header for incorrect frequency	§
FUDGE	T	Template task to work on *uv* data	§
UVFIL	T	Template task to create a *uv* data set	§
UVFIT	T	Fit models to small *uv* data sets	§8.4.3
UVSEN	T	Find sensitivity and rms sidelobes of data set	§

MAPETC

Type: Imaging, Cleaning and Self-calibration

		Imaging	
APMAP	T	Make an image directly from the EXPORT tape	§
UVMAP	T	Make image from sorted data on disk	§3.4, 9.3
MX	T	Multi-field and channel image and clean	§3.4, 5.1, 9.3, 10.5.2
FFT	T	Take the Fourier transform of an image	§5.3
UVDIS	T	Display the Fourier transform on the TV	§
GRIDR	T	Grid pseudo-*uv*, single-dish data into image	§
		Deconvolution	
APCLN	T	Clean an image using the Array Processor	§3.4.3, 5.1, 9.1
APGS	T	Deconvolution by Gerchberg-Saxton algorithm	§5.1.6
APVC	T	Image deconvolution by van Cittert iteration	§5.1.6
CCMOD	T	Prepare a model clean component file	§8.4.3
DCONV	T	Gaussian deconvolution of an image	§
IMMOD	T	Rescale image while adding model component	§8.4.3
PRTCC	T	Print selected parts of clean components file	§5.1.5, 5.2.1
STEER	T	Image deconvolution by David Steer clean method	§5.1.6
VM	T	Make an image using a maximum entropy-related method	§5.1.6
XBASL	T	Removes n^{th}-order baselines from image rows	§9.7, 9.11
		Self-calibration	
ASCAL	T	Self-calibrate a *uv* data set using a clean image	§5.2, 9.1
ASCOR	T	Apply ASCAL gain solutions to another file	§5.2.3, 9.1
GAPLT	T	Plot gain files antenna by antenna	§5.2.1
GNMRG	T	Merge two similar gain files	§
GNPLT	T	General plot routine for ASCAL gain solution	§5.2.1
PRTGA	T	Print gain files	§
VBCOR	T	Apply VBFIT results to other data	§10.3
VBFIT	T	Global fringe fitting for VLBI	§10.3
VSCAL	T	Self-calibrate a VLBI *uv* data set	§10.3, 10.5.1
		Miscellaneous	
TAFFY	T	Template task to work on image data	§
CANDY	T	Template task to make model images	§

ANALYSIS

Type: Image processing, analysis and combination

Image processing

CNVRT	T	Convert between I*2 and R*4 images	§9.8
CONVL	T	Convolve with a gaussian or another image	§9.11
GEOM	T	Stretch, rotate and displace an image	§8.4.1
HGEOM	T	Transform image to another image's geometry	§8.1.3, 8.4.1
LGEOM	T	Large image interpolation and rotation	§8.4.1, 8.4.4, 9.11
MWFLT	T	Image lowpass filtering: median, mode, average	§8.4.2
NINER	T	3 x 3 matrix operator on images	§8.4.2, 8.4.4
NTERP	T	Interpolate an image to a new grid	§
PBCOR	T	Correct an image for the primary beam attenuation	§
PGEOM	T	Convert image between polar and rectangular coords	§8.4.1, 8.4.4
SMOTH	T	Smooth an image of up to 7-dimensions	§
SUBIM	T	Create a new image from subsection of an image	§4.7, 9.9
TRANS	T	Transpose an image	§9.10
XSMTH	T	Smooth data along x axis only	§9.11
XTRAN	T	Optical plate solution and correction	§

Image analysis

BLSUM	T	Sums images over polygonal regions	§8.2, 9.1, 9.11
GAL	T	Fit model rotation curve to velocity image	§9.11
IMEAN	T	Intensity statistics, flux density and histogram	§7.9, 8.2, 9.1
IMFIT	T	Fit Gaussian components to a portion of an image	§8.3.2
IMSTAT	V	Intensity statistics into adverbs	§8.2, 13.8
IMVAL	V	Get image intensity at specified pixel	§
IRING	T	Finds radial intensity profile of projected image	§
JMFIT	T	Fit Gaussian components to a portion of an image	§8.3.2
MAXFIT	V	Determine intensity and position of a maximum	§8.3.1, 13.5
MOMFT	T	Fit moments to a portion of an image	§
MOMNT	T	Finds images of moments 0 to 3 from a data cube	§9.11
NNLSQ	T	Deconvolve spectral components	§9.11
QIMVAL	V	Get image intensity into adverb, no messages	§13.8
PATGN	T	Make images of test patterns	§
SLFIT	T	Fit Gaussian components to a 1-D slice	§8.3.3
TVSTAT	V	Determine statistics in TV-selected areas of image	§6.3, 8.2
UVFIT	T	Fit models to small uv data sets	§8.4.3
WARP	T	Fit model of warped galaxy	§
XGAUS	T	Fit Gaussian components to rows of cube	§8.3.4, 9.11
XMOM	T	Find moments along x axis	§9.11
XSUM	T	Produce n-1 dimensional image summing x axis	§

Image combination

BLANK	T	Delete regions from images	§9.11
COMB	T	Image combination with many algorithms	§8.1, 9.1, 9.7, 9.11
CORMS	T	Image combination with error analysis	§8.1.3

POLCO	T	Correct polarisation images for noise bias	§
RM	T	Rotation Measure and B field from set of images	§
SUMIM	T	Add together many images	§9.7

Handling table extension files

PRTAB	T	Print contents of general table file	§
TASRT	T	Sort contents of table file	§

TAPU

Type: Reading from, writing on, and positioning tapes

General tape handling

MOUNT	V	Logically mount a tape in software	§3.2, Z.1.5, Z.2.5
DISMOUNT	V	Rewind and logically dismount a tape	§3.2, Z.1.5, Z.2.5
AVFILE	V	Advance or backspace files on a tape	§6.1
AVMAP	V	Advance images on a tape	§6.1
REWIND	V	Rewind a tape	§6.1
AVEOT	V	Advance tape to end-of-information	§
PRTTP	T	Print summary of a tape; any recognised format	§3.1
AVTP	T	Reposition tape	§

Reading and writing FITS-format tape of images or *uv* data

TPHEAD	V	Give info on tape data at current position	§6.1
IMLOD	T	Copy an image to disk from FITS or IBM tape	§6.1, 9.1
UVLOD	T	Copy *uv* data to disk from FITS tape	§3.2, 9.2, 10.1.1
FITTP	T	Copy images or *uv* data to FITS-tape from disk	§11.1, 13.2

Reading and writing EXPORT-format tape of *uv* data

EXIND	T	List index from an EXPORT tape	§3.1
EXPND	T	Copy EXPORT tape to a tape with one source per file	§
UVLOD	T	Copy *uv* data to disk from an EXPORT tape	§3.2, 9.2, 10.1.1
UVEXP	T	Copy *uv* data to an EXPORT tape from disk	§11.1

Reading and writing other data formats

FILLR	T	Read VLA ModComp format *uv* data tape	§
IBMTP	T	Copy an image to an IBM-tape from disk	§
WSLOD	T	Read WSRT format *uv* data tape	§

Data analysis

EXFND	T	List *uv* data directly from an EXPORT tape	§
APMAP	T	Make image(s) directly from an EXPORT tape	§

CUBE

Type: Data cube (spectral line) processing

Note that most other programs support data cube processing in a general way.

ALTDEF	V	Define velocity relation to frequency axis	§9.5
ALTSWTCH	V	Switch between velocity and frequency in header	§9.5
AXDEFINE	V	Redefine axis values in the header	§
BLANK	V	Delete regions from images	§9.11
BLSUM	T	Sums images over polygonal regions	§8.2, 9.1, 9.11
CNVRT	T	Convert between I*2 and R*4 images	§9.8
COMB	T	Image combination including cube with a plane	§8.1, 9.1, 9.7, 9.11
DCONV	T	Gaussian deconvolution of an image	§
MCUBE	T	Build an n-dim image from set of (n-1) dim images	§9.4
MOMNT	T	Calculate 4 moment images from a cube	§9.11
NNLSQ	T	Deconvolve spectral components	§9.11
PLCUB	T	Plot rows of image on a 2-D grid	§7.6, 9.11, 9.12
REMOVIE	V	Re-run preloaded sequence of planes on TV	§9.6
RGBMP	T	Create a cube with three planes for TV3COLOR use	§9.11
SMOTH	T	Smooth an image, up to 7-dimensions	§
SUBIM	T	Create a new image from subsection of an image	§4.7, 9.9
SUMIM	T	Add together many images	§9.7
TRANS	T	Transpose an image	§9.10
TV3COLOR	T	False color TV display from three image planes	§9.1
TVCUB	T	Prepare cube for TVSLD 3-D "solid"	§
TVHUEINT	T	Hue-intensity TV display using two image planes	§9.1
TVMOVIE	V	Load planes to TV, display in sequence	§9.6
TVSLD	T	Prepare 3-D image from TVCUB "cube"	§
TVSLV	T	Display TVSLD output on TV	§9.1
XBASL	T	Removes n^{th}-order baselines from image rows	§9.7, 9.11
XGAUS	T	Fit Gaussian components to rows of cube	§8.3.4, 9.11
XMOM	T	Find moments along x axis	§9.11
XPLOT	T	TEK plots of image rows	§7.6, 9.11
XSMTH	T	Smooth data along x axis only	§9.11
XSUM	T	Produce n-1 dimensional image summing x axis	§

DELETE

Type: Deleting files

ZAP	V	Delete an image and all extension files	§3.3, 11.2, 13.4
ALLDEST	V	Delete all images consistent with the inputs	§11.2
TIMDEST	V	Delete all images older than time parameters	§4.6
SCRDEST	V	Delete a scratch file associated with a task	§4.6
EXTDEST	V	Delete an extension file	§7.4, 7.5
SAVDEST	V	Delete all SAVE / GET files	§
SGDESTR	pV	Delete specified SAVE / GET file	§
CLRMSG	V	Clear messages from user's message log file	§4.1, 11.2

VLBI

Type: Software dealing specifically with VLB data

AVER	T	Time averages data	§
CITCC	T	Writes CC file in *CIT* package format	§
FITTP	T	Write *uv*-FITS tape from *AIPS* *uv* data files	§11.1, 13.2
PRTDR	T	Prints contents of fringe fitting (DR) file	§10.3
STRIP	T	Antenna scaling for VLB	§
TOVLB	T	Converts *AIPS* *uv* data to *CIT* MERGE data set	§10.3, 10.4.2
UVAVG	T	Average or merge BT or TB sorted *uv* data	§10.2, 10.3
UVEXP	T	Write EXPORT tape from VLB and other disk data	§11.1
UVFIT	T	Fits models to small *uv* data sets	§8.4.3
UVLOD	T	Write disk data from normal and VLB EXPORT tapes	§3.2, 9.2, 10.1.1
VBANT	T	Apply tables of antenna gains for VLBI	§10.4
VBCAL	T	Multiplies amplitudes by antenna scale factors	§
VBCC	T	Writes CC file in *CIT* package format	§
VBCIT	T	Convert CIT Merge data to *AIPS*	§10.1.2
VBCOR	T	Applies the result of VBFIT to other data	§10.3
VBFIT	T	Global fringe fitting task	§10.3
VBLIN	T	Convert NRAO-SAO Decode data to *AIPS*	§10.1.3
VBMRG	T	Merges a VLBI data set including polarisation	§
VBPLT	T	Plot data vs model 1 baseline/plot	§10.3
VLBDR	T	Estimates antenna delays and rates from NRAO-FRINGE output and writes control table for FRINGE	§ §
VSCAL	T	Self-calibrate a VLBI *uv* data set	§10.3, 10.5.1

APTASKS

Type: Tasks which use the Array Processor

APCLN	T	Clean an image	§3.4.3, 5.1, 9.1
APGS	T	Deconvolution by Gerchberg-Saxton algorithm	§5.1.6
APMAP	T	Make an image directly from the EXPORT tape	§
APVC	T	Image deconvolution by van Cittert iteration	§5.1.6
ASCAL	T	Self-calibrate a *uv* data set using a clean image	§5.2, 9.1
CONVL	T	Convolve with a Gaussian or another image	§9.11
FFT	T	Take the Fourier transform of an image	§5.3
GRIDR	T	Grid pseudo-*uv*, single-dish data into image	§
MX	T	Multi-field and channel image and clean	§3.4, 5.1, 9.3, 10.5.2
NOBAT	T	Reserve array processor, otherwise dummy	§
NTERP	T	Interpolate an image with splines	§
STEER	T	Image deconvolution by D. Steer Clean	§5.1.6
UVDIS	T	Display FFT of image on TV	§
UVMAP	T	Make an image from sorted data on disk	§3.4, 9.3
UVSUB	T	Subtract point components from *uv* data	§5.3, 9.7, 10.4.4
VBFIT	T	Global fringe fitting for VLBI	§10.3
VM	T	Make an image using a maximum entropy-related method	§5.1.6
VSCAL	T	Self-calibrate a VLBI *uv* data set	§10.3, 10.5.1

POPSYM

Type: Symbols used in the POPS interpretive language

VERB	USE	COMMENTS	§
	— Arithmetic expressions —		
+	A + B	Add the expression A to B	§ 13.3, 13.8
-	A - B	Subtract the expression B from A	§ 13.1
*	A * B	Multiply the expression A by B	§ 13.3, 13.8
/	A / B	Divide the expression A by B	§ 13.8
**	A ** B	Calculate A to the power B	§
( )	(A+B)*C	Grouping expressions as desired	§ 13.8
=	A = B	Store the value of B into A	§ 13.1, 13.8
,	A = 3,5,4	Separator of elements in an array	§ 13.1
:	TO	Equivalent to the verb TO	§ 9.9, 13.4
;		Separator between POPS statements	§ 13.1, all
	— Logical expressions —		
>	A > B	A greater than B	§ 13.8
<	A < B	A less than B	§
=	A = B	A equal B (numeric or string)	§
>=	A >= B	A equal to or greater than B	§
<=	A <= B	A equal to or less than B	§
<>	A <> B	A not equal to B (numeric or string)	§
!	A ! B	A or B	§
&	A & B	A and B	§
^	^ A	not A	§
	— String expressions —		
!!	A !! B	String = string A followed by string B	§
SUBSTR	SUBSTR(A,*i*,*j*)	String = chars *i* through *j* of string A	§
LENGTH	LENGTH(A)	Position last non-blank in A	§
CHAR	CHAR(A)	Convert number A to string	§
VALUE	VALUE(A)	Convert string A to number	§
	— Looping constructions —		
FOR TO		FOR I = A TO B BY C	§ 6.1, 9.9, 13.4, 13.8
BY		*any valid set of* POPS *syntax*	§
END		END	§
WHILE		WHILE *any logical expression*	§
		any valid set of POPS *syntax*	§
END		END	§
IF		IF *any logical expression*	§ 13.8
THEN		THEN *any valid set of* POPS *syntax*	§
ELSE		ELSE *any valid set of* POPS *syntax*	§
END		END	§

— Built-in functions —

ATAN	X = ATAN(A)	Arctangent (result in degrees)	§13.3
ATAN2	X = ATAN2(B,A)	Arctangent (B/A) (result in degrees)	§
COS	X = COS(A)	Cosine (A in degrees)	§
SIN	X = SIN(A)	Sine (A in degrees)	§
TAN	X = TAN(A)	Tangent (A in degrees)	§
EXP	X = EXP(B)	Exponential	§
LN	X = LN(B)	Log base *e*	§
LOG	X = LOG(B)	Log base 10	§
SQRT	X = SQRT(B)	Square-root	§13.8
MAX	X = MAX(A,B)	Maximum	§
MIN	X = MIN(A,B)	Minimum	§
MODULUS	X = MODU(Q,U)	Root-square sum: $X = \sqrt{Q^2 + U^2}$	§
MOD	X = MOD(A,B)	A - floor (A/B) * B, *i.e.*, remainder of A/B	§
CEIL(A)	X = CEIL(A)	Lowest integer $\geq$ A	§
FLOOR(A)	X = FLOOR(A)	Highest integer $\leq$ A	§

— Procedure building verbs —

PROC	PV	Begin building a procedure	§13.5, 13.8
PROCEDUR	PV	Begin building a procedure	§
LIST	pV	List a procedure	§13.5
EDIT	PV	Edit a procedure	§13.5
ENDEDIT	PV	End editing a procedure	§13.5
ERASE	PV	Delete line(s) of a procedure	§13.5
MODIFY	PV	Modify a line in a procedure	§13.5
RETURN	V	Last statement in a procedure	§13.5
FINISH	PV	End procedure building	§13.5, 13.8

— Variable declarations —

SCALAR	pV	Declare scalars	§13.8
ARRAY	pV	Declare arrays	§13.8
STRING	pV	Declare strings	§

— Input/Output functions —

PRINT	V	Print the following keyword value(s)	§6.1, 9.9
TYPE	V	Print the following keyword value(s)	§13.3, 13.8
READ	V	Read value(s) from terminal after # prompt	§

— Other information —

CORE	pV	Amount of core left in *POPS*	§13.5
DEBUG	pV	Turns on compiler debug information	§
DUMP	V	Dump K array on terminal screen	§
SCRATCH	PV	Remove a procedure in *POPS*	§
$	PV	Makes rest of input line a comment	§

INDEX

Type: Alphabetical listing of all verbs (V), tasks (T), special pseudoverbs (PV), verb-like pseudoverbs (pV) and built-in procedures (P) currently in *AIPS*.

Use: All of the following keywords initiate action in *AIPS* by causing a program or set of programs to execute. A more complete description of each keyword can be obtained from the additional HELP files indicated in the last column below or from the HELP file associated with the keyword.

CCMOD	T	Model clean components	§8.4.3
CEIL	V	Return lowest integer ≥ argument	§
CELGAL	V	Switch between Galactic and Celestial coords	§9.5
CHAR	V	Convert a number to a string	§
CITCC	T	Write CC in *CIT*-VLB format	§
CLIP	T	Flag *uv* data above value	§5.3
CLRMSG	V	Delete messages from user's message file	§4.1, 11.2
CLRNAME	V	Set INNAME etc. to null	§3.4.1, 4.2
CLR2NAME	V	Set IN2NAME etc. to null	§
CLR3NAME	V	Set IN3NAME etc. to null	§
CLRSTAT	V	Clear catalog status	§4.2.3
CNTR	T	Produce contour plot file	§7.1, 7.9
CNVRT	T	Convert image between I*2 and R*4	§9.8
COMB	T	Combining two images	§8.1, 9.1, 9.7, 9.11
COMPRESS	PV	Not implemented	§
CONVL	T	Convolves two images	§9.11
COPY	T	Copies all files of catalog entry	§
CORER	T	*uv* correlator statistics	§5.3
CORE	pV	Available space in *POPS*	§
CORMS	T	COMB with error analysis	§8.1.3
COS	V	Cosine (arg in degrees)	§
CORFQ	T	Correct observing freq.	§
CURBLINK	V	Turn cursor blink on/off	§
CURVALUE	V	Display value under cursor	§5.3, 6.3
— D —			
DBCON	T	Concatenate two *uv* files	§10.2
DCONV	T	Deconvolution of Gaussian from image	§
DEBUG	pV	Turn on compiler debug	§
DESCM	T	Reduce size of *uv* file	§
DISKU	T	List by user all disk space used in *AIPS*	§4.6
DISMOUNT	V	Rewind and logically dismount a tape	§3.2, Z.1.5, Z.2.5
DUMP	V	Dump *POPS* K array	§
— E —			
EDIT	PV	Begin editing procedure	§13.5
EGETNAME	V	Insert image name by catalog number	§13.5
END	V	End of *POPS* loop	§6.1, 9.9, 13.4, 13.8
ENDEDIT	PV	End editing procedure	§13.5
ELSE	pV	*POPS* IF THEN ELSE	§
ERASE	PV	Delete line(s) of a procedure	§13.5
EXFND	T	List *uv* data from tape	§
EXIND	T	Index *uv* data from tape	§3.1
EXIT	V	Exit from AIPS	§4.6, 11.4
EXP	V	Exponentiation	§
EXPLAIN	pV	List help information on printer	§4.8, all
EXPND	T	Copy Export tape, 1 source/file	§
EXTDEST	V	Destroy extension file	§7.4, 7.5
EXTLIST	V	List extension files	§7.4, 7.5

— F —

— G —

— H —

— I —

IMVAL	V	Obtain intensity under cursor	§
IMWEDGE	V	Put wedge on TV (image min-max)	§
IMXY	V	Obtain cursor pixel position	§ 8.3.1
INP	pV	List inputs for verb or task	§ 4.4, all
INPUTS	pV	List inputs for verb or task	§ 4.8
IRING	T	Find radial intensity projected disk	§
— J —			
JMFIT	T	2-D Gaussian fitting of image	§ 8.3.2
JOBLIST	V	List batch jobs in queue	§
— L —			
LENGTH	V	Return number characters in string	§
LGEOM	T	Large image interpolation and rotation	§ 8.4.1, 8.4.4, 9.11
LIST	pV	List a procedure	§ 13.5
LN	V	Natural log	§
LOG	V	Base 10 log	§
— M —			
MAX	V	Maximum function	§
MAXFIT	V	Parabolic fit of image peak	§ 8.3.1, 13.5
MCAT	V	Index of images in catalog	§ 3.4.1, 4.2
MCUBE	T	Build a data cube	§ 9.4
MIN	V	Minimum function	§
MOD	V	Mod function (*i.e.*, remainder)	§
MODIFY	PV	Modify a line in a procedure	§ 13.5
MODULUS	V	Modulus ($\sqrt{A^2 + B^2}$)	§
MOMFT	T	Find moments of a component	§
MOMNT	T	Finds moments 0 to 3 of cube	§ 9.11
MOUNT	V	Logically mount a tape in software	§ 2.5, Z.1.5, Z.2.5
MSGKIL	pV	Stop messages from file	§
MWFLT	T	Image lowpass filtering: median, mode, average	§ 8.4.2
MX	T	Multi-field and channel image and clean	§ 3.4, 5.1, 9.3, 10.5.2
— N —			
NINER	T	3 x 3 matrix operator on images	§ 8.4.2, 8.4.4
NNLSQ	T	Deconvolve spectral components	§ 9.11
NOBAT	T	Reserve array processor, else dummy	§
NTERP	T	Image interpolation	§
— O —			
OFFHUINT	P	Turn off hue-intensity display	§
OFFPSEUD	V	Turn off TV pseudo color	§
OFFROAM	P	Turn off TV roam mode	§
OFFSCROL	V	Turn off TV scrolling	§
OFFTRANS	V	Turn off TV transfer function	§
OFFZOOM	V	Turn off TV zooming	§ 9.1, 9.6

— P —

PASSWORD	V	Set user's *AIPS* password	§
PATGN	T	Make images of test patterns	§
PBCOR	T	Correct for primary beam attenuation	§
PCAT	V	List all files in "half" of catalog	§
PCNTR	T	Contour and pol line vector	§7.2, 7.9
PFPL1	T	Plot paraform: x/y same as image	§
PFPL2	T	Plot paraform: x is slice of image	§
PFPL3	T	Plot paraform: x/y no relation to image	§
PGEOM	T	Convert image rectangular/polar coords	§8.4.1, 8.4.4
PLCUB	T	Plot rows of image on a 2-D grid	§7.6, 9.11, 9.12
PLROW	T	Plot rows of image w user spacing	§7.6, 7.9, 9.11
POLCO	T	Correct pol. image for noise bias	§
PRINT	V	Print the following value	§6.1, 9.9
PROC	PV	Define procedure	§13.5, 13.8
PROCEDUR	PV	Define procedure	§
PROFL	T	Produce profile plot file	§7.6, 7.9
PRTAB	T	Print table file contents	§
PRTAN	T	Print antenna file	§
PRTCC	T	Print clean components file	§5.1.5, 5.2.1
PRTDR	T	Print VLB fringe fit file	§10.3
PRTGA	T	Print gain ext. file	§
PRTHI	V	Print history file	§3.4.1, 3.4.2, 4.3, 5.1.1
PRTIM	T	Print digital display of image	§5.1.3
PRTMSG	V	Print message file	§3.2, 4.1, 13.7
PRTPL	T	Print plot file on printer	§5.2.1, 7.1
PRTSD	T	Print single-dish "*uv*" data file	§
PRTTP	T	Print summary of a tape	§3.1
PRTUV	T	Print selected *uv* data	§10.3
PUTHEAD	V	Put adverb value into header	§

— Q —

QHEADER	V	List summary of image header	§4.2.4
QIMVAL	V	Get image intensity with no messages	§13.8
QMSPL	T	Display plot file on QMS laser printer	§5.2.1, 7.1, 7.6
QUEUES	V	List batch job queue	§13.7
QWKPL	T	VAX: plot file to printer	§

— R —

READ	V	Read terminal inputs	§
REBOX	V	Reset cleaning boxes with graphics display	§6.3
RECAT	V	Compress catalog file	§4.3
REMOVIE	V	Rerun previously loaded TV movie	§9.6
RENAME	V	Rename a file	§4.3, 13.8
RENUMBER	V	Renumber a catalog entry	§4.3
REROAM	V	Reenter roam mode on TV	§
RESCALE	V	Rescale image values	§
RESTART	V	Restart AIPS from scratch	§
RESTORE	PV	Get STORE file of AIPS environment	§4.4, 13.1

RETURN	V	Last statement in procedure	§ 13.5
REWIND	V	Rewind a tape	§ 6.1
RGBMP	T	Create RED, GREEN, BLUE cube from cube	§ 9.11
RM	T	Determine rotation measure	§
RUN	PV	Execute text in RUN file	§ 13.1, 13.6

— S —

SAVDEST	V	Destroy a save file	§
SAVE	pV	Permanent storage of AIPS	§ 4.4
SCALAR	pV	Scalar declaration in *POPS*	§ 13.8
SCRATCH	PV	Remove procedures in *POPS*	§
SCRDEST	V	Scratch temporary AIPS files	§ 4.6
SELSD	T	Prepare single-dish "*uv*" data for GRIDR	§
SETROAM	V	Fill in ROAM parameters	§
SETXWIN	P	Fill BLC, TRC using cursor and width	§ 6.3
SGDESTR	pV	Destroy SAVE / GET files	§
SGINDEX	V	List SAVE / GET files	§
SIN	V	Sine (in degrees)	§
SL2PL	T	Convert slice file to plot file	§ 7.5, 7.9, 8.3.3
SLFIT	T	1-D Gaussian fit to slice file	§ 8.3.3
SLICE	T	Make a slice of an image	§ 7.5, 9.11
SMOTH	T	Smooth an image, up to 7 dimensions	§
SPY	V	Interrogation of system	§ 4.6, 12.1
SQRT	V	Square-root	§ 13.8
STEER	T	Image deconvolution by D. Steer Clean	§ 5.1.6
STRING	pV	Declare string in *POPS*	§
STRIP	T	Antenna scaling for VLB	§
STORE	pV	Temporary storage of AIPS	§ 4.4
SUBIM	T	Create a subimage	§ 4.7, 9.9
SUBMIT	V	Submit batch file to queue	§ 13.7
SUBSTR	V	Reference portion of character string	§
SUMIM	T	Sum or average many images	§ 9.7

— T —

T1VERB	V	Spare verb for development	§
T2VERB	V	Spare verb for development	§
T3VERB	V	Spare verb for development	§ Z.1.9
T4VERB	V	Spare verb for development	§
TAFFY	T	Template task to work on image data	§
TAN	V	Tangent (in degrees)	§
TASRT	T	Sort table file	§
THEN	pV	*POPS* IF THEN ELSE	§ 13.8
TGET	pV	Get most recent task set-up	§ 4.4, 4.5
TGINDEX	V	List all TGET files	§
TIMDEST	V	Destroy all files older than ...	§ 4.6
TO	V	*POPS* FOR TO BY loop	§ 6.1, 13.4, 13.8
TOVLB	T	Convert *AIPS* data to VLBI	§ 10.3, 10.4.2
TPHEAD	V	List image header from tape	§ 6.1
TPUT	pV	Save a task or verb set-up	§ 4.4, 4.5
TRANS	T	Transpose a data cube	§ 9.10
TYPE	V	Print the following value	§ 13.3, 13.8

— TK —

TKBOX	P	Fill any BOX using TEK cursor	§
TKGUESS	V	Initialize and display guess	§
TKAGUESS	V	Add guess to display	§8.3.3
TKMODEL	V	Initialize and display model	§
TKAMODEL	V	Add model fit to display	§8.3.3
TKNBOX	P	Fill BOXes using TEK cursor	§
TKPL	T	Display plot file on TEK	§5.2.1, 7.1
TKPOS	V	Obtain position under cursor	§
TKRESID	V	Initialize and display residual	§
TKARESID	V	Add fit residual to display	§8.3.3
TKSET	V	Use cursor to fill initial guess	§8.3.3
TK1SET	V	Modify model guess	§
TKSLICE	V	Initialize and display Slice	§7.5, 8.3.3, 9.11
TKASLICE	V	Add Slice to plot	§7.5
TKVAL	V	Obtain image value under cursor	§
TKWIN	P	Fill BLC, TRC with TEK cursor	§
TKXY	V	Get pixel position under cursor	§

— TV —

TV3COLOR	V	Make a false color display	§9.1
TVALL	P	General purpose TV display	§6.2, 9.6
TVBLINK	V	Blink TV between two images	§
TVBOX	V	Set BOX with cursor, graphics	§5.1.1, 6.3
TVCLEAR	V	Clear TV channels(s)	§
TVCUB	T	Prepare cube for TVSLD 3-D "solid"	§
TVFIDDLE	V	Same as TVALL with no load	§6.2
TVHLD	T	Load high precision image, do equalization	§
TVHUEINT	V	Hue-intensity display 2 images	§9.1
TVHXF	T	Interactive histogram equalization of image	§
TVINIT	V	Initialize TV	§
TVLABEL	V	Label image with coordinates	§
TVLOD	V	Load image on TV channel(s)	§9.11, 13.5
TVLUT	V	N-point transfer function	§
TVMBLINK	V	Blink TV between two images	§
TVMLUT	V	Many-point transfer function	§
TVMOVIE	V	Load planes of cube, run movie	§9.6
TVNAME	V	Print image name under cursor	§13.5
TVOFF	V	Turn off image channel(s)	§9.1, 9.6
TVON	V	Turn on image channel(s)	§6.2, 9.1, 9.6
TVPL	T	Display plot file on the TV	§7.1
TVPOS	V	Obtain position under cursor	§
TVPSEUDO	V	Initiate pseudo coloring	§6.2.1
TVRESET	P	Reset TV functions leaving image on	§
TVROAM	V	Load image channels and roam	§
TVSCROLL	V	Scroll an image if necessary	§
TVSLD	T	Prepare 3-D image from TVCUB "cube"	§
TVSLICE	V	Set slice end points with graphics display	§6.3, 7.5
TVSLV	T	Display TVSLD output on TV	§9.1
TVSTAT	V	Determine stats in TV-selected areas	§6.3, 8.2

TVTRANS	V	Change transfer function	§6.2.2
TVWEDGE	V	Put wedge on TV (as loaded)	§
TVWINDOW	V	Set BLC / TRC with graphics	§6.3
TVWLABEL	V	Label step wedge with intensities	§
TVZOOM	V	Turn on TV zoom capability	§

— U —

UNQUE	V	Unque waiting batch files	§13.7
UCAT	V	Index of *uv* files in catalog	§3.2, 4.2
UVAVG	T	Average or merge BT or TB sorted *uv* data	§10.2, 10.3
UVCOP	T	Copy selected *uv* data	§4.7, 9.1
UVDIS	T	Display FFT of image on TV	§
UVERR	T	Load incorrect *uv* FITS from $\leq$ 15May83	§
UVEXP	T	Write *uv* data on tape	§11.1
UVFIL	T	Template task to create a *uv* data set	§
UVFIX	T	Recompute *u*, *v*, and *w* from antenna data	§
UVFLG	T	Flag selected *uv* data	§5.3
UVFND	T	Print selected *uv* data	§5.3
UVFIT	T	Fit models to small *uv* data set	§8.4.3
UVLOD	T	Read *uv* data from tape	§3.2, 9.2, 10.1.1
UVMAP	T	Make image from sorted *uv* data	§3.4, 9.3
UVMOD	T	Rescale *uv* data while adding model component	§8.4.3
UVPLT	T	Generate *uv* plots	§3.4.4, 5.2.1, 5.3
UVSEN	T	Find sensitivity, sidelobes of data set	§
UVSRT	T	Sort *uv* data	§3.3, 9.1, 10.2
UVSUB	T	Subtract model from *uv* data	§5.3, 9.7, 10.4.4

— V —

VALUE	V	Return numeric value of string	§
VBANT	T	Apply tables of antenna gains for VLBI	§10.4
VBCAL	T	Apply VLB antenna factors	§
VBCC	T	Write CC files in *CIT* format	§
VBCIT	T	Convert CIT Merge data to *AIPS*	§10.1.2
VBCOR	T	Apply fringe fit result	§10.3
VBFIT	T	Global fringe fitting for VLB	§10.3
VBLIN	T	Convert NRAO-SAO Decode data to *AIPS*	§10.1.3
VBMRG	T	Merge VLB data	§
VBPLT	T	Plot data vs model 1 baseline/plot	§10.3
VLBDR	T	Fringe, delay and antenna fit	§
VM	T	Make image with maximum entropy method	§5.1.6
VSCAL	T	Self-calibrate a VLBI *uv* data set	§10.3, 10.5.1

— W —

WAITTASK	pV	Permits task sequencing	§4.5, 6.1, 9.9
WARP	T	Fit model of warped galaxy	§
WEDERASE	V	Erase wedge image in TV channel	§
WHILE	pV	*POPS* conditional expression	§
WSLOD	T	Load WSRT *uv*-data tape to disk	§

— X —

XBASL	T	Removes n^{th}-order baselines from rows	§9.7, 9.11
XGAUS	T	Fit Gaussian components to rows of cube	§8.3.4, 9.11
XMOM	T	Find moments along x axis	§9.11
XPLOT	T	TEK plots of image rows	§7.6, 9.11
XSMTH	T	Smooth data along x axis only	§9.11
XSUM	T	Produce n-1 dimensional image summing x axis	§
XTRAN	T	Optical plate solution and correction	§

— Z —

ZAP	V	Destroy an image	§3.3, 11.2, 13.4

— NON-ALPHA —

+	V	Addition or Positive	§13.3, 13.8
-	V	Subtraction or Negative	§13.1
*	V	Multiplication	§13.3, 13.8
/	V	Division	§13.8
**	V	Exponentiation	§
(	V	Begin grouping	§13.8
)	V	End grouping	§13.8
=	V	Store or logical equals	§13.1, 13.8
,	V	Separator of array elements	§13.1
:	V	Same as TO in loop	§9.9, 13.4
;	V	Separator of *POPS* statements	§13.1, all
>	V	Greater than	§13.8
<	V	Less than	§
>=	V	Greater or equal to	§
<=	V	Less than or equal to	§
<>	V	Not equal to	§
!	V	Logical or	§
&	V	Logical and	§
~	V	Logical not	§
!!	V	Concatenate strings	§
$	PV	Makes rest of input line a comment	§

G. Glossary

adverb — See *POPS symbols.*

AIPS monitor — a computer terminal (perhaps lacking a keyboard) whose CRT screen is used in AIPS solely for the display of information related to the progress of the execution of the AIPS tasks. (Except, at those AIPS sites without a terminal dedicated to this use, the AIPS user's interactive terminal is used for dual purposes—i.e., to serve as the AIPS monitor as well.) Many of the messages which the AIPS tasks write to the monitor also are recorded in the *message file* (*q.v.*).

allased response — in a radio interferometer map, a spurious feature due to a source—or to a sidelobe—that lies outside of the field of view. Consider the sampling of a visibility function V at the lattice points of a rectangular grid as multiplication of V by the comb-like distribution $R(u,v) = \sum_k \sum_l \delta(u - k\Delta u, v - l\Delta v)$. The Fourier transform $\widehat{RV}$ of RV is given by the convolution $\hat{R} * \hat{V}$. Since $\hat{R}$ is again a comb-like distribution, with peaks, or teeth, separated by $\frac{1}{\Delta u}$ in one direction and by $\frac{1}{\Delta v}$ in the perpendicular direction, $\widehat{RV}$ is periodic, and, about the position of each tooth in the comb, it looks like an infinite summation of rectangular pieces of $\hat{V}$, each of size $\frac{1}{\Delta u} \times \frac{1}{\Delta v}$, taken from all over the plane. Aliased responses can be suppressed very effectively, by judicious choice of the *gridding convolution function* (*q.v.*).

For a more complete discussion, see Dick Sramek's Lecture No. 5 in the *1985 Summer School Proceedings*. Also see VLA Scientific Memoranda Nos. 129 and 131.

allasing — in spectral analysis, error which is due to undersampling: one may wish to sample a signal that is known to be bandlimited, but whose bandwidth may not be known a priori. The Fourier transform of *Shannon's series* is periodic; aliasing error is of the form of an overlapping, or superposition, of these "replicated" spectra. See *Nyquist sampling rate* and *aliased response.*

ALU — (Arithmetic Logic Unit) an (optional) microcomputer CPU unit within the I^{2}S TV display device which allows simple arithmetic operations, such as sums, products, and convolutions, to be performed on the data recorded in the I^{2}S image planes. At present, AIPS makes little use of the ALU, since many of its features are unique to the I^{2}S display unit. See *I^{2}S.*

antenna file — in AIPS, an *extension file*, associated with a *u-v data file*, in which a list of the interferometer antenna positions is stored.

antenna/i.f. gain — Many of the systematic errors affecting radio interferometer measurements are multiplicative in the visibility amplitude and additive in the visibility phase, and are ascribable to individual antenna elements and their associated i.f./l.o. chains. For each antenna/i.f. these sources of error may be lumped together into a complex-valued function of time, $g(t)$, called the *antenna/i.f. gain.* Then, the visibility measurement obtained on the i–j baseline at time t is given by $\hat{V}(u_{ij}(t), v_{ij}(t)) = g_i(t)\overline{g}_j(t)V(u_{ij}(t), v_{ij}(t)) + \epsilon_{ij}(t)$, where V is the true source visibility and where the spatial frequency coordinates (u, v) have been parametrized by time. $g_i\overline{g}_j$ is the systematic "calibration error", and ϵ_{ij}, an additive error component, is assumed to be random and well-behaved. (Another type of systematic error, the *instrumental polarization* (*q.v.*), is not included in the g_k, and always must be corrected, by proper calibration, in order to interpret polarization data.)

Some of the most serious sources of error—including atmospheric attenuation, error arising from variations in the atmospheric path length, clock error, and error in the baseline determination—conform fairly well to this multiplicative model. This model relation is exploited heavily by the *self-calibration algorithm* (*q.v.*). Compare *antenna/i.f. phase*, and see *isoplanaticity assumption* and *correlator offset.*

antenna/i.f. phase — The *antenna/i.f. phase* for antenna k of an interferometer array is given by the argument (or phase) of the *antenna/i.f. gain* g_k: $\psi_k(t) = \arg g_k(t)$. Often in *self-calibration* one assumes that no amplitude errors are present and solves only for the ψ_k.

antenna residual delay — See *residual delay* and *global fringe fitting algorithm.*

antenna residual fringe rate — See *residual fringe rate* and *global fringe fitting algorithm.*

AP — See *array processor.*

AP–120B array processor — an *array processor* manufactured by Floating Point Systems, Inc., and used at a number of AIPS sites. Its floating-point word length is 38-bits. Typically it is equipped with a main data memory of 32–64 kilowords and a program source memory of 2048 words. With both a pipeline multiplier and a pipeline adder, and a memory cycle time of 167 ns., when programmed at top efficiency it can perform at an arithmetic rate of 12 million floating-point operations per second.

The AP–120B is no longer in production; this product has been superseded by the 5000 series product line. Though the AP–5000's are used at some AIPS sites, their advanced features are not used by AIPS—only those features which are shared by the older model are fully exploited by AIPS tasks.

array processor — a computer peripheral attachment which is capable of performing certain floating-point computations, especially vector and matrix operations, at high speed, and independently of the *host computer* central processing unit. Usually the high-speed performance is achieved by a technique known as pipelining. The basic arithmetic operations of addition and multiplication are performed in stages, by a so-called pipeline adder and a pipeline multiplier. These units operate just like an assembly line in a manufacturing plant. Some array processors (AP's) are constructed with multiple pipelines. Address computations are performed concurrently with the arithmetic operations, by a unit which is separate from the pipelines. The algorithms best-suited to an array processor implementation are those which can be structured so as to keep the pipelines filled a fair fraction of the time. Most AP's have their own high-speed data memory, but some are parasitic on the memory of the host computer. Portions of many AIPS tasks have been programmed for the Floating Systems, Inc. model *AP-120B array processor*, (*q.v.*). Also see *array processor microcode*, *Q-routine*, and *pseudo-array processor.*

array processor microcode — program source code written in the assembly language of an *array processor*, (*q.v.*). Array processor (AP) manufacturers usually provide an extensive library of utility subroutines that may be called from a high-level programming language, such as Fortran; however, some computationally-intensive algorithms cannot be easily or efficiently implemented using only these libraries. Portions of these algorithms must be written in microcode—a painstaking process. The assembly languages of different models of AP's differ considerably (as do the subroutine libraries, too, in fact) because of differences in the hardware architectures. Thus the AIPS programming group tries to avoid writing microcode. But portions of the AIPS *tasks* for

mapmaking, deconvolution, and self-calibration are written in AP microcode. Also see *Q-routine*.

associated file — In AIPS, any two or more files among a collection consisting of a *primary data file* and all of its *extension files* are termed *associated*.

auto re-boot — a boot initiated by the computer itself, of its own volition. See *boot*.

back-up — The act of copying the contents of a computer file to some permanent storage medium such as magnetic tape or punched cards, for the purpose of protecting against accidental loss or in order to liberate storage space (e.g., disk space), is termed *backing-up*. The new copy of the file is termed a *back-up* copy, or simply a *back-up*. See *scratch*.

bandwidth smearing — in a radio interferometer map, space-variance of the *point spread function* which is attributable to non-monochromaticity, or finite bandwidth. The point spread function—at a particular point in a map—taking into account bandwidth smearing, but ignoring other instrumental effects, is termed a *delay beam*. Bandwidth smearing is a radial effect: the delay beams become more elongated, in the radial direction from the interferometer *phase tracking center*, as their distance from the phase tracking center increases. The delay beams are easily calculable when all of the receivers in an array have identical, and known, i.f. passbands. E.g., with rectangular passbands of width $\Delta\nu$, and observations centered at a frequency ν_0, the measured visibility amplitude of a point source is proportional to $\frac{\sin\gamma}{\gamma}$ —where $\gamma \equiv \pi(ux + vy + wz)\frac{\Delta\nu}{\nu_0}$, (u, v, w) denotes the spatial frequency coordinates, measured in wavelengths at ν_0, and (x, y, z) denotes the direction cosines of the location of the point source, with respect to the phase tracking center. For more details, see A. R. Thompson's Lecture No. 2 and W. D. Cotton's Lecture No. 8 in the *1985 Summer School Proceedings* and see VLA Scientific Memo. No. 137.

Bandwidth smearing can, in principle, be eliminated (assuming that the bandpasses are known) by applying an image reconstruction algorithm which has a knowledge of the smearing mechanism; that is, by an algorithm which is more general than the usual deconvolution algorithms—see *image reconstruction*. The most common method for reducing bandwidth smearing is the technique of *bandwidth synthesis*, (*q.v.*).

bandwidth synthesis — a technique of radio interferometry which is intended to diminish the effect of *bandwidth smearing*. Bandwidth synthesis observing is very similar to spectral-line mode observing: the i.f. bandpasses are split up into a number of pieces, or channels, and the data in each channel are treated separately up until the mapping/deconvolution stage of processing. At that stage, the problem can be formulated as a system of simultaneous convolution equations: one has the system $g_1 = b_1 * f + \epsilon_1, \ldots, g_n = b_n * f + \epsilon_n$, where n is the number of frequency channels, g_k is the *dirty map* for channel k, b_k the *dirty beam* for that channel, f the unknown radio source brightness distribution (here assuming that f is not a function of frequency), and ϵ_k is noise (were it not for the noise, and for the fact that each deconvolution problem is ill-posed—in its own right—, there would be no reason to treat the equations simultaneously, or even to consider more than a single one of them). (For a description of a refinement to the bandwidth synthesis technique, for sources with spatially-varying spectral indices, see *broadband mapping technique*.) Note that all the b_k are identical, apart from a dilation factor; i.e., as the *u-v coverage* "shrinks", toward the low end of the observing band, the b_k dilate by the reciprocal of the *u-v* shrinkage factor.

The present state of software development does not allow solving the problem in quite the way it is formulated above. Rather, some mapping/deconvolution algorithm is applied separately to each of the channels, and the resulting maps are averaged.

baseline-time order — An ordered set of visibility measurements $\left\{ V_{ij}(t_k) \;\middle|\; 1 \le i < j \le n,\ k = 1, \ldots, l \right\}$ recorded with an n element interferometer at times t_k is said to be in *baseline-time order* if the ordering is such that all of the data for the 1–2 baseline, sorted by time, occur first, followed by the data for the 1–3 baseline, again sorted by time, etc., etc. (This canonical ordering by baseline is the order $V_{12}, V_{13}, \ldots, V_{1n}, V_{23}, \ldots, \ldots, V_{n-1,n}$.) Compare *time-baseline order*.

Baseline-time ordering of a *u-v data file* is convenient for purposes of data display.

batch editor — a *text editor* within the AIPS program which allows the user to prepare *batch jobs* (*q.v.*), to be run non-interactively.

batch job — AIPS may be run either interactively—allowing the user to make 'split-second' decisions—or in batch mode. In batch mode, the user first decides on a set strategy for reducing the data, and then, using the special AIPS *batch editor*, the user prepares a *text file*, containing those AIPS commands which are appropriate to the anticipated data reduction needs. The batch job is placed in a *batch queue*, and the job steps are executed by the *batch processor*, in a non-interactive mode.

batch processor — the server, or scheduler, for *batch jobs* (*q.v.*). The AIPS batch processor follows certain rules in scheduling: batch jobs requiring the use of an array processor (AP) often are scheduled to run only during nighttime hours; the processor serving one of the *batch queues* might refuse service, altogether, to a job requiring an AP; and batch jobs may be given lower priority than those AIPS tasks which are run interactively.

batch queue — a waiting line for *batch jobs*. The AIPS batch queue is a single-server queue—i.e., the server (the *batch processor*) initiates the execution of the jobs one after the other, rather than in parallel. However, AIPS can be configured with more than one batch queue, each with its own batch processor; this number varies according to site.

"battery-powered" CLEAN algorithm — a modified version of the Clark CLEAN algorithm, devised by Fred Schwab and Bill Cotton. At each major cycle of the algorithm, or perhaps less frequently, the residual map is computed not by convolving the current iterate with the dirty beam map, but rather by computing the visibility residuals, and then re-gridding and re-mapping. By this means, the edge effects are compensated, and hence one can search the full dirty map field of view for clean components. Simultaneously, instrumental effects (finite bandwidth and finite integration time) and sky curvature (the wz term) can be compensated for (i.e., the algorithm solves a more general equation than a convolution equation). See *Clark CLEAN algorithm*.

A "mosaicing" version of this algorithm is implemented in the AIPS task MX. The deconvolved image is defined over some number $1 \le n \le 16$ of rectangular patches. Within each patch, the data are corrected for sky curvature, by the correction appropriate to the center of the patch. Instrumental corrections are not included, at present.

beam — 1. in radio interferometry, the inverse Fourier transform (FT^{-1}) of the *u-v sampling distribution*, or FT^{-1} of a weighted *u-v* sampling distribution, possibly convolved

with a gridding convolution function—the idealized response to a point, or unresolved, radio source. 2. a numerical approximation to 1. 3. a digitized version of 2, sampled on a regular grid (usually regarded as a map or image). 4. ≈ *point spread function*, q.v. 5. (occasionally) as above, but taking into account instrumental effects, so that the beam depends on position in the sky. See *dirty map*.

Occasionally, any one of the above, other than 5, is termed the *synthesized beam*.

beam patch — in the Clark CLEAN algorithm, that portion of the central part of the beam which is used in the inner iterations, or the minor cycles. In the AIPS implementation, the beam patch size typically is set at 101 pixels × 101 pixels. See *Clark CLEAN algorithm*.

beam squint — In radio interferometry, direction dependent, or space-variant *instrumental polarization*, which is difficult to calibrate, can arise from *beam squint*. The beam squint effect, for the usual case of a pair of (nominally) orthogonally polarized feeds on each array element, is due to differences in their power patterns—in particular, to differences in the directions of their peak response.

blanked pixel — in a digital image, a pixel whose value is undefined. In computer storage of quantized digital images, some special numeric value is assigned to the blanked pixels, so that they may be recognized as undefined and given whatever special treatment is required. See *pixel*.

BLC — *bottom left corner* (of an image). See *m* × *n map*.

blink — See *TV blink*.

boot — A computer is restarted by means of a *boot*strapping procedure, whereby the operating system and the data management facilities are re-initialized in a succession of steps. This ritual, through which the computer gathers it wits, is termed the *boot*. A boot (≈ *re-boot*) is required after any system crash (e.g., after a power failure). Usually the sequence of steps required to accomplish the boot is posted in a notice located close to the system operating console, or on the CPU panel. On modern computers, such as the Vax, the boot procedure is highly automated. In fact, there may be an abbreviated boot procedure, termed a *quick boot*, to follow after a "soft" system crash. (On such systems, a quick boot should be attempted before resorting to a full boot.) Indeed, some systems (the Vax included) re-boot on their own initiative following a soft system crash—this is termed an *auto re-boot*.

BOT marker — (Beginning-Of-Tape marker) a short strip of metal foil attached near the front, or beginning, end of a computer magnetic tape. The tape drive uses the BOT marker in order to position the tape at its starting position.

bpi — (bits per inch) the basic unit of measurement used to specify the density at which information is recorded on a computer magnetic tape: the effective number of bits per inch per track. The standard recording densities are 800, 1600, and 6250 bpi. Modern computer tapes are nine-track tapes: eight recording tracks are used for the data, and the ninth track is used to record "parity bits" for error-checking. See *tape blocking efficiency*.

broadband mapping technique — a refinement of the radio interferometric method of *bandwidth synthesis* (*q.v.*), in which one solves simultaneously for the radio brightness distribution $f_{\nu_r}(x,y)$ at some reference frequency ν_r, and for the (spatially varying) spectral index $\alpha(x,y)$ across the observing band. Assuming that the observing band is split into frequency channels centered at $\nu_1,\ldots,\nu_n$, one solves the simultaneous system of convolution equations $g_1 = b_1 * f_1,\ldots,g_n = b_n * f_n$, where g_k is the *dirty map* from channel k, b_k the *dirty beam* from that channel, and where f_k is given by

$$f_k(x,y) = \left(\frac{\nu_k}{\nu_r}\right)^{\alpha(x,y)} f_{\nu_r}(x,y)\,.$$

All of the b_k are identical, apart from a dilation factor. Assuming that the frequency channels are narrow enough, one can expand the *u-v coverage* considerably, with immunity to the *bandwidth smearing* effect. Fractional bandwidths as large as 20–30% can be used, depending on the linearity of the spectral index variations.

This mapping technique is described by Tim Cornwell [Broadband mapping of sources with spatially varying spectral index, VLB Array Memo. No. 324, Feb. 1984]. Extensive modification of one of the standard deconvolution algorithms is required. The requisite modification of the *Högbom CLEAN algorithm* is in progress.

b-t order — See *baseline-time order*.

bug — an actual or a perceived programming error or program deficiency. The bug may be in the eye of the beholder since the program user may fancy an application similar to, but differing from, the one for which the program is intended. In AIPS there is a formal mechanism for reporting program bugs; see *gripe file* for a description.

byte — a unit of eight bits of computer storage.

carriage-return key — One of the most used keys on any computer terminal keyboard is the carriage-return key ($\hookleftarrow_R$). This is the button which ordinarily must be depressed when one has finished typing a command to the computer, in order for the computer to accept or acknowledge the command.

catalog entry — an entry within an AIPS *catalog file* ("CA" file) pertaining to a particular *primary data file*.

catalog file — In AIPS, each user has, for each disk on which he has data stored, his own *catalog file*, or "CA" file—a directory of all of his primary data files which reside on that disk. The AIPS *verb* CATALOG (as do its variants MCAT and UCAT) allows the user to see a summary listing of the contents of his catalog files. See *header record*.

catalog slot — in AIPS, a numbered space reserved in a *catalog file* for the insertion of a *catalog entry*.

cell-averaging — in radio interferometer mapping, gridding convolution which is achieved simply by averaging the visibility data which lie in each *u-v* grid cell. This is equivalent to use of a *gridding convolution function* equal to the *characteristic function* of the rectangle $\{|u| < \Delta u/2,\ |v| < \Delta v/2\}$, where Δu and Δv denote the grid spacing—i.e., it is equivalent to the use of a so-called *pillbox* function. The Fourier transform of the pillbox gridding convolution function is proportional to a separable product of two $\frac{\sin x}{x}$ functions; this function does not decay rapidly enough to yield very effective *aliasing* suppression. The zero-order *spheroidal functions* offer much better aliasing suppression, at somewhat increased computational expense (equivalent to averaging the data over a region 36 times larger, in the case of the default gridding convolution function used by the AIPS mapping tasks).

cellsize — in radio interferometer mapping, the size $\Delta u \times \Delta v$ of the *u-v* grid cells. Ordinarily, the visibility data are smoothed by an appropriate *gridding convolution function* and this convolution then is sampled at the coordinate locations of the centers of the grid cells. After appropriate weighting, the *discrete Fourier transform* yields the *dirty map*. Δu and Δv are chosen according to *Shannon's sampling theorem*: if the size of the dirty map is x radians by y radians, then $\Delta u = \frac{1}{x}$ wavelengths and $\Delta v = \frac{1}{y}$ wavelengths.

cereal bowl map defect — same as *negative bowl artifact*. See *zero-spacing flux*.

characteristic function — The characteristic function χ_A of a set $A \subset X$ is defined for all $x \in X$ by the formula

$$\chi_A(x) = \begin{cases} 1, & \text{if } x \in A, \\ 0, & \text{if } x \notin A. \end{cases}$$

(χ_A is also called the *indicator function* of A, and the notations c_A and 1_A commonly are used in lieu of χ_A.) Note that this usage of the term, which is standard in mathematical analysis, differs from its usage in probability and statistics, where it refers to the Fourier transform of a probability measure (i.e., to the FT of the distribution function of a random variable).

chromaticity — in visual perception, essentially the dominant wavelength and the purity of the spectral distribution of light, as perceived. *Hue* and *saturation* determine the chromaticity, which is independent of *intensity*. See *C.I.E. chromaticity diagram*.

C.I.E. chromaticity diagram — a two-dimensional diagram devised in 1931 by the Commission Internationale de l'Eclairage (International Commission on Illumination) to show the range of perceivable colors as a function of normalized chromaticity coordinates (x, y), under standardized viewing conditions. The color, for an additive mixture of monochromatic red, green, and blue (R, G, B denoting the intensities at 650, 520, and 380 nm. wavelengths) as perceived by a 'standard observer', is displayed in this diagram as a function of the normalized *chromaticity coordinates* $x = R/(R+G+B)$ and $y = G/(R+G+B)$.

Other chromaticity diagrams can be drawn for different choices of primary hues, for mixtures of nonmonochromatic light, or for 'nonstandard observers'. In digital imagery, such a diagram may be tailored to a particular color image display unit. See [G. S. Shostak, Color basics—a tutorial. In R. Albrecht and M. Capaccioli, I.A.U. Astronomical Image Processing Circular No. 9, Space Telescope Science Institute, Jan. 1983] and [G. Wyszecki and W. S. Stiles, *Color Science*, Wiley, New York, 1967], a comprehensive textbook on colorimetry.

Clark CLEAN algorithm — a modified version of the Högbom CLEAN algorithm, devised by Barry Clark in order to accomplish an efficient *array processor* implementation of CLEAN (see [B. G. Clark, An efficient implementation of the algorithm CLEAN, *Astron. Astrophys.*, **89** (1980) 377–378]). To operate on, say, an $n \times n$ map, the original CLEAN algorithm requires on the order of n^2 arithmetic operations at each iteration, and typically there may be hundreds or thousands of iterations. The Clark algorithm proceeds by operating not on the full residual map, but rather by picking out only the largest residual points, iterating on these for a while (during its *minor cycles* or inner iterations) and only occasionally (at the *major cycles*) computing the full $n \times n$ residual map, by means of the FFT algorithm. After each major cycle, it again picks out the largest residuals and goes into more minor cycles. And, for further economy, during these inner iterations the dirty beam is assumed to be identically zero outside of a relatively small box (termed the *beam patch*) which is centered about the origin. See *Högbom CLEAN algorithm*.

CLEAN — See *Högbom CLEAN algorithm*.

clean beam — in the Högbom CLEAN algorithm, an elliptical Gaussian function h with which the final iterate is convolved, in order to diminish any spurious high spatial frequency features—also termed *restoring beam*. h is specified by its major axis (usually the FWHM), its minor axis, and the position angle on the plane of the sky of its major axis. Usually these parameters are set by fitting to the central lobe of the dirty beam. See *Högbom CLEAN algorithm* and *super-resolution*.

clean box — a rectangular subregion of a *clean window* (*q.v.*).

clean component — in the Högbom CLEAN algorithm, a δ-function component which is added to the $(n-1)$st iterate in order to obtain the nth iterate. Its location is the location of the peak residual after the $(n-1)$st iteration, and its amplitude is a fraction μ (the *loop gain*) of the largest residual. See *Högbom CLEAN algorithm*.

The AIPS task implementing the (Clark) CLEAN algorithm stores a list of the clean components in an extension file which is termed a *components file*.

clean map — an approximate deconvolution of the *dirty beam* from the *dirty map*, derived by an application of the Högbom CLEAN algorithm or one of its derivatives. See *Högbom CLEAN algorithm*.

clean speed-up factor — in the *Clark CLEAN algorithm*, a number α in the range $[-1, 1]$ used in determining when to end a major cycle. Smaller α causes a larger number of major cycles to occur (at greater computational expense) but yields a result closer to that of the classical *Högbom CLEAN algorithm*.

clean window — in the Högbom CLEAN algorithm, the region A of the residual map which is searched in order to locate the *clean components* comprising the successive approximants to the radio source brightness distribution. In the AIPS implementation, A is a union of rectangles, called *clean boxes*, which may be specified by the user. When A is not explicitly specified, the algorithm searches over the central rectangular one-quarter area of the residual map. See *window clean* and *Högbom CLEAN algorithm*.

clipping — the discarding (i.e., the *flagging*) of visibility data whose amplitudes exceed some threshold value, or the discarding of visibility data whose differences from some tentative source model are too large in amplitude. The AIPS task CLIP is used for clipping. See *u-v data flag*.

closure amplitude — Assume that the visibility observation on the i–j baseline $(i < j)$ is given by $\tilde{V}_{ij} = g_i \bar{g}_j V_{ij}$, where V_{ij} is the true visibility and where g_i and g_j are the *antenna/i.f. gains* (ignore any additive error). Then, for certain combinations of (at least four) baselines, one may form ratios of observed visibilities (and their conjugates)—including each visibility only once—in such a manner that the g's cancel one another. For example, if $i < j < k < l$, then

$$\frac{\tilde{V}_{ij}\tilde{V}_{kl}}{\tilde{V}_{il}\tilde{V}_{jk}} = \frac{V_{ij}V_{kl}}{V_{il}V_{jk}}.$$

The modulus of such a ratio is termed a *closure amplitude* (and its argument, a *closure phase*).

Closure amplitude is called a "good observable", since, under the above assumptions, it is not sensitive to measurement error. The closure amplitude and closure phase relations are exploited in the *hybrid mapping algorithm* (*q.v.*). Also see *self-calibration algorithm*.

closure phase — Assume that the visibility observation on the i–j baseline $(i < j)$ is given by $\tilde{V}_{ij} = g_i \bar{g}_j V_{ij}$, where V_{ij} is the true visibility and where g_i and g_j are the *antenna/i.f. gains* (ignore any additive error). Then, for a combination of any three or more baselines forming a

closed loop, one may sum the visibility phases in such a manner that the *antenna/i.f. phases* ψ_k drop out. For example, if $i < j < k$, then $\arg\tilde{V}_{ij} + \arg\tilde{V}_{jk} - \arg\tilde{V}_{ik} = \arg V_{ij} + \psi_i - \psi_j + \arg V_{jk} + \psi_j - \psi_k - \arg V_{ik} - \psi_i + \psi_k$. Such a linear combination of observed visibility phases is termed a *closure phase*.

Closure phase is called a "good observable", since, under the above assumptions, it is not sensitive to measurement error. The closure phase relations are exploited in the *hybrid mapping algorithm* (*q.v.*). Also see *closure amplitude* and *self-calibration algorithm*.

color contour display — a color digital image display of a real-valued function f of two real variables (x, y), in which the color assignment (the *hue*) is a coarsely quantized function of $f(x, y)$. The visual effect of this type of *pseudo-color display*, in the case when f is continuous, is similar to the traditional sort of contour display. One sees curves along which f is constant, separated by swathes of constant hue—each hue corresponding to a distinct quantization level.

color triangle — Any three non-collinear points plotted on a chromaticity diagram determine a color triangle. Since the points are non-collinear, they correspond to basic, or *primary* hues. All of those colors on the chromaticity diagram which fall within the triangle determined by the three points may be produced by addition of the three hues. See *C.I.E. chromaticity diagram*.

compact support — See *support*.

components file — in AIPS, an extension file, associated with an image file containing a *clean map*, whose content is a list of the positions and amplitudes of the *clean components* included in that clean map, as determined by the CLEAN algorithm. The source model specified by this list of components often is used in *self-calibration*.

conjugate symmetry — that property which characterizes a *Hermitian function* (*q.v.*). Generally an assumption of conjugate symmetry is implicit whenever one speaks of the *u-v coverage* corresponding to some radio interferometric observation.

Conrac monitor — the CRT unit of the I^{2}S TV display device, in use at a number of AIPS installations. See *I^{2}S*.

convolution theorem — This theorem is well-known, but seldom is quoted in its distributional form: for two distributions, f and g, the Fourier transform of the convolution of f and g is given by $\widehat{f * g} = \hat{f}\hat{g}$, whenever one distribution is of *compact support* and the other is a "tempered" distribution. (Loosely speaking, a tempered distribution is one which does not increase too rapidly at infinity.) See [Y. Choquet-Bruhat, C. Dewitt-Morette, and M. Dillard-Bleick, *Analysis, Manifolds, and Physics*, North-Holland, New York, 1977, ch. VI].

One ought to be aware of this form of the theorem, since often one must deal with convolution of functions that are not of compact support—*dirty beams, principal solutions, invisible distributions*, etc.—whose Fourier transforms do not exist as ordinary functions, but only as distributions or generalized functions.

Convolution of distributions, itself, is defined, in general, whenever the support of either distribution is compact, or (in one dimension) when the supports of both distributions are limited on the same side. For distributions which are absolutely integrable ordinary functions, and whose Fourier transforms possess the same property, the compact support assumption is not required here, or above. Related fact: convolution is not always associative (i.e., $f*(g*h) \neq (f*g)*h$), in general), but it is associative provided that all the distributions, with the possible exception of one, are of compact support. See the above-cited reference.

convolving function — See *gridding convolution function*.

coordinate reference pixel — in an AIPS *image file*, a *"pixel"* whose coordinates are recorded in the image *header* together with the coordinate increments (i.e., the pixel coordinate separations) that allow the physical coordinates of all other pixels in the image to be computed. This "coordinate reference pixel" may not actually be present in the image: all that matters are its physical coordinates and its pixel coordinates (which too are recorded in the header—and which may, in fact, be fractional).

Often, in a radio map (and by default, when the standard AIPS mapmaking tasks are executed), the position of the coordinate reference pixel coincides with the map center and with the *visibility phase tracking center*. See *m* × *n map* and *pixel coordinates*.

correlator offset — One of the basic assumptions of much of the VLA calibration software (e.g., the *self-calibration algorithm*) is that the systematic errors in the visibility measurements are multiplicative errors that are ascribable to individual array elements and their associated i.f./l.o. chains, and that—at a given instant—each such antenna-based error has an identical effect on each visibility observation involving that antenna/i.f. combination. Systematic measurement errors which do not conform to this model are called *correlator offsets* or *non-closing errors*. See *antenna/i.f. gain*.

Correlator offsets can be the limiting factor in obtaining high dynamic range VLA maps. Some observers have reported fairly large multiplicative correlator offsets which vary slowly with time and which do not appear to vary with the *phase tracking center* or with source structure. From observations of an external calibrator, one may estimate, and compensate for, such offsets. This mechanism is provided in the AIPS tasks BCAL1 and BCAL2. See [R. C. Walker, Non-closing offsets on the VLA, VLA Scientific Memo. No. 152].

crash — the abrupt failure of a computer system or program. More specifically, a *system crash* is the abrupt failure of a computer—or of a computer's operating system—causing the computer to halt the execution of programs; and a *program crash* is the abrupt failure of a computer program resulting either from a flaw in the logic of the program itself, or from some peculiar interaction with the operating system, the storage management facility, another program, or the user—or from an act of God. A *hardware crash* (e.g., a *disk crash*) is a crash which results from the failure of the computer electronics or electro-mechanics, and a *software crash* is one which results from a flaw or an inadequacy in program logic, or in operating system program logic. A *soft crash* is a crash from which it is easy to recover—i.e., easy to restart the computer and resume work—, and a *hard crash* is the opposite.

crosshair — 1. a marker on the *TEK screen*, or *green screen*, which may be moved about through the use of thumbwheel knobs which are located on the terminal keyboard panel. The position of the crosshair may be sensed by the computer program, and thus the user may point out to the program features that are of interest in the graphical display on the CRT screen. 2. a marker with the same function as just described, but on a TV display device, and more likely controlled by a *trackball* than by thumbwheels. Same as *TV cursor*; and see *trackball*.

cube — See *data cube*.

cursor — 1. a marker on an interactive computer terminal indicating the position on the CRT screen where the next

character is to be typed. 2. *TV cursor*—on a TV display device, a marker whose manually controlled position may be sensed by the computer. See *crosshair*.

data cube — 1. in VLA spectral line data analysis, a three-dimensional map or "image" representing a function of three real variables—two spatial variables representative of position in the sky, and one variable related to frequency or velocity. 2. any n-dimensional *image*, $n \geq 3$.

Computer access of a multi-dimensional data array, residing in any standard type of storage medium such as disk or magnetic tape, is sequential, as if the data were one-dimensional. Spectral line data cubes are stored plane-by-plane, row-by-row, column-by-column. Permutation of the correspondence between plane, row, and column, and the coordinate axis numbering, is referred to as *transposition* of the data cube.

database — a computer filing system, or file structure system. For example, the AIPS database consists not only of the data themselves, but also of the directories and the cross-reference lists of all the AIPS data files (including extension files), the data format definitions, etc., as well as the rules and principles governing the use thereof.

data file — on a computer storage medium, such as disk or magnetic tape, the concrete, or physically present representation of a logically distinct grouping of data in a manner permitting repeated access by computer programs.

data flag — See *u-v data flag*.

DCP — (Dicomed Control Program) a program (which runs only on the Modcomp Classic Computer in Charlottesville) to control the Dicomed Image Recorder, for purposes of photographic display of image data recorded on magnetic tape. The DCP, not a part of AIPS, is described in [T. R. Cram and E. W. Greisen, *The NRAO Image Recording System*, NRAO Computer Division Internal Report No. 21, May 1975]. At present there is no path, other than through the DCP, for Charlottesville AIPS users to use the Dicomed unit. Data tapes to be read by the DCP must be written on the IBM computer in Charlottesville.

deconvolution — the numerical inversion of a convolution equation, either continuous or discrete, in one or several variables; i.e., the numerical solution (for f) of an equation of the form $f * g = h +$ noise, given g and given the right-hand side of the equation. Except in trivial cases, deconvolution is an ill-posed problem: In the absence of constraints or extra side-conditions, and in the case of noiseless data—assuming that some solution exists—there usually will exist many solutions. In the case of noisy data, there usually will exist no exact solution, but a multitude of approximate solutions. In the latter case, if one is not careful in the choice of a numerical method, the computed approximate solution is likely not to have a continuous dependence on the given data. The so-called *regularization method* (*q.v.*) (of which the *maximum entropy method* is a special case) is an effective tool for the deconvolution problem.

Discrete two-dimensional deconvolution is an everyday problem in radio interferometry, owing to the fact that—under certain simplifying assumptions—the so-called *dirty map* is the convolution of the *dirty beam* with the true celestial radio image. In addition to the maximum entropy method, the *Högbom CLEAN algorithm* is commonly applied to this problem. See Tim Cornwell's Lecture No. 7 in the *1985 Summer School Proceedings*.

delay — See *residual delay*.

delay beam — in radio interferometry, the *point spread function* or *beam*, taking into account *bandwidth smearing*, but ignoring other instrumental effects. See *bandwidth smearing*.

delay-rate file — in AIPS, an *extension file*, associated with a *u-v data file*, in which a table of the computed *antenna residual delays* and *antenna residual fringe rates* is stored. Ordinarily the delay-rate file is created by the AIPS program VBFIT operating on VLBI data. The file can be read by the program VBCOR, which applies the appropriate corrections to the data. See *global fringe fitting algorithm*.

DFT — an abbreviation for *discrete Fourier transform* and *direct Fourier transform* (*q.v.*). When used in disciplines other than radio astronomy, it usually signifies the former.

Dicomed Image Recorder (Model D47) — a computer-controlled image display device intended for photographic reproduction of digital images. The film is exposed by a cathode ray tube. The device is capable of 4096 pixel × 4096 pixel resolution and of both black-and-white and color reproduction. The digital exposure control and eight-bit pixel input allow 256 discrete exposure levels. The CRT has a single electron gun and a screen with a white phosphor; color reproduction is accomplished by means of multiple exposures, with the insertion of red, green, and blue filters. There is a Dicomed recorder at the NRAO in Charlottesville, and another at the VLA. See *DCP*.

direct Fourier transform — a term used imprecisely in radio astronomy to mean either: 1) a finite trigonometric sum, of the form

$$\sum_{j=0}^{n-1} a_j e^{2\pi i u_j x},$$

with a_j complex, where the (real) u_j are irregularly-spaced; 2) the brute-force evaluation of such a sum; or 3), the naïve, or brute-force evaluation (using $O(n^2)$ arithmetic operations) of the (n-point) *discrete Fourier transform*.

The direct Fourier transform, in senses 1) and 2) of the definition, arises in synthesis mapping applications because of the irregular distribution of the visibility measurements. Common practice is to use a *gridding convolution function* to interpolate the data onto a regularly-spaced lattice, so that, for computational economy, the *fast Fourier transform algorithm* may be used.

dirty beam — in radio interferometry, simply a *beam*, but computed with precisely the same operations as those used to compute some companion *dirty map* (i.e., with the same *u-v* coverage, the same manner of gridding convolution, the same *u-v* weight function and taper, etc.). In cleaning a dirty map, only the companion dirty beam should be used.

dirty map — 1. ignoring instrumental effects, the inverse Fourier transform (FT^{-1}) of the product of the visibility function V of the radio source and the (possibly *weighted* and/or *tapered*) *u-v sampling distribution* S; i.e., FT^{-1} of the *u-v measurement distribution*. 2. a discrete approximation to 1; in this case, the product SV is convolved with some function C, of *compact support*, and an inverse discrete Fourier transform of samples of $C * (SV)$ taken over a regular grid yields the *dirty map*. 3. as in 2, but corrected for the taper ($\check{C}$, the FT^{-1} of C) induced by the convolution. 4. any of the above, but now taking into account various instrumental effects (receiver noise, non-monochromaticity or finite bandwidth, finite integration time, sky curvature, etc.).

If it is assumed that $V \equiv 1$, then the map, or point source response, so obtained is termed the *beam* (*q.v.*). Also see *gridding convolution function*, *u-v taper function*, *u-v weight function*, *dirty beam*, and *principal solution*.

discrete Fourier transform — The (one-dimensional) discrete Fourier transform (DFT) $y_0, \ldots, y_{n-1}$ of a sequence of complex numbers $x_0, \ldots, x_{n-1}$ is given by the summation

$$y_k = \sum_{j=0}^{n-1} x_j e^{2\pi ijk/n} .$$

(The multi-dimensional generalization is straightforward). The x_j are given by the *inverse DFT* of the y_k:

$$x_j = \frac{1}{n} \sum_{k=0}^{n-1} y_k e^{-2\pi ijk/n} .$$

(Frequently the forward and inverse transforms are defined in the manner opposite to that given here, and the $\frac{1}{n}$ normalization factor sometimes is moved about.) The DFT arises most naturally in numerically approximating the Fourier coefficients $c_m = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-imx}\, dx$ of a 2π-periodic function f which is representable by the trigonometric series $\sum_{m=-\infty}^{\infty} c_m e^{imx}$. The *fast Fourier transform algorithm* (*q.v.*) can be used for efficient numerical evaluation of the DFT.

disk hog — a derogatory term, used to connote a computer user whose disk data files are excessively voluminous or numerous, therefore putting other computer users at a relative disadvantage. Unneeded data files should be *scratched*, or destroyed, in order to free up disk space. Large disk files which will not be needed for a time should be *backed-up* on magnetic tape and then deleted from disk.

dynamic range — a summary measure of image quality indicative of the ability to discern dim features when relatively stronger features are present—i.e., a measure of the ability to distinguish the dim features from artifacts of the *image reconstruction* procedure (in a radio map, from remnants of the sidelobes of stronger features) and from noise. The dynamic range achievable in a radio interferometer map is determined primarily by the uniformity of the *u-v coverage*, the density and extent of the coverage, the sensitivity of the array, and the quality of the calibration.

If the true radio source brightness distribution f is known, one can define the dynamic range of a reconstruction $\tilde{f}$ as, say, the ratio of the maximum value of $|f|$ to the r.m.s. difference between f and $\tilde{f}$. When f is unknown, as is usually the case, an empirical measure of the dynamic range is used—perhaps the ratio of the maximum value of $|\tilde{f}|$ to the r.m.s. level in an apparently empty region of the map, or the ratio of the strongest feature to the weakest "believable" feature—, but there is no widely-accepted definition.

What one might wish to call the "true" dynamic range of a radio map is a spatially-variant quantity. The ability to discern a dim feature depends on its proximity to brighter features, because there are relatively stronger sidelobe remnants near the bright features. The quality of a map (and perhaps the dynamic range—depending on how it is defined) deteriorates away from the *phase tracking center*, because of the inability of the image reconstruction algorithms to compensate for various instrumental effects (e.g., bad pointing, *bandwidth smearing*, etc.).

EDT — a sophisticated text editor (a *screen editor*) used on the Vaxes. It makes use of the "keypad" feature of the fancier terminals. EDT can be run only on certain model terminals: on the DEC (Digital Equipment Corp.) Models VT-52 and VT-100, and on terminals such as the Visual-50's and the Visual-100's which are capable of emulating the DEC terminals. See *text editor*.

EMACS — a sophisticated text editor used on the Vaxes, as well as on many computers which run under the UNIX operating system. (There is also a version for the IBM-PC.) EMACS is a *screen editor*, and the one which is favored by most among those in the AIPS programming group. On terminals with the "keypad" feature, the keypad keys can be programmed by the user to perform many useful editing tasks; however, EMACS can be run from other models of terminals, as well. EMACS provides two powerful and convenient features which most other text editors do not offer: the ability to temporarily exit from the editor and "return to monitor level," and the ability to initiate an interactive "job control session," or initiate *sub-tasks*, in an EMACS buffer. See *text editor*.

explain file — in AIPS, a *text file* containing a detailed explanation of a particular AIPS *task* or *verb*, often including hints, suggested applications, algorithmic details, and bibliographical references. Issuing the AIPS verb EXPLAIN causes the contents of an explain file to be printed on the terminal screen or on a line printer. Compare *help file*.

EXPORT format — a visibility data magnetic tape format for transport of VLA data from the DEC-10 computer or the on-line computer at the VLA.

EXPORT tape — a magnetic tape containing data recorded in the *EXPORT format*.

exp × sinc function — a useful gridding convolution function: same as the *Gaussian-tapered sinc function* (*q.v.*), except that the exponent of the argument to the exponential function may be other than two.

extension file — in AIPS, a data file containing data supplemental to those contained in a *primary data file* (either a *u-v data file* or an *image file*). Whenever a primary data file is deleted by the standard mechanism within AIPS for file destruction, all extension files associated with that primary data file also are destroyed. Extension files, however, may be deleted without deleting the the associated primary data file.

Extension files are grouped into categories of named types. Examples: *plot files, history files, slice files, gain files*, etc.

When an AIPS task creates a new primary data file from an old one, generally it attaches, to the new file, clones of any extension files associated with the old file that remain relevant to the new one.

false color display — In digital imagery, a *false color display* is one which is generated by using a number $n > 1$ of real-valued functions $f_1(x,y), \ldots, f_n(x,y)$ to control the proportions, at each *pixel* coordinate (x,y), of an additive mixture of three primary hues. In practical terms, the user of a digital display system supplies $f_1, \ldots, f_n$, and twists knobs that control the mapping $\mathbf{R}^n \rightarrow \mathbf{R}^3$ that sends the n pixel values at each (x,y) into the proper image *chromaticity* and *intensity*. Compare *pseudo-color display*.

A so-called *true color display* is obtained with $n = 3$ and with *transfer functions* chosen such that the color assignment corresponds in an approximate way to the actual coloration of a scene (as in a color photograph).

fast Fourier transform algorithm — a fast algorithm for the computation of the *discrete Fourier transform* (DFT) $y_0, \ldots, y_{n-1}$ of a sequence of n complex numbers $x_0, \ldots, x_{n-1}$,

$$y_k = \sum_{j=0}^{n-1} x_j e^{2\pi ijk/n} ,$$

typically requiring only $O(n \log n)$ arithmetic operations—or a multi-dimensional generalization thereof. By con-

trast, straightforward, or naïve evaluation of the DFT requires $O(n^2)$ operations. The fast Fourier transform algorithms (FFT's) which currently are the most popular are the Cooley–Tukey (1965) algorithms, for the case of n highly composite. For n a power of two, the (radix-2) Cooley–Tukey FFT requires about $2n \log_2 n$ real multiplications and $3n \log_2 n$ real additions. More generally, the Cooley–Tukey algorithms require a few times $n\sigma(n)$ complex arithmetic operations, where $\sigma(n)$ is the sum of the prime factors of n, counting their multiplicities. S. Winograd has produced FFT algorithms which are more efficient than those of Cooley and Tukey, typically requiring about the same number of additions, but only about 20% the number of multiplications. (Computation of the required complex exponentials—or sines and cosines—is not counted, since these generally are either pre-computed and stored in compact tables, or generated recursively.)

A further advantage of the FFT algorithms is their avoidance of round-off error, which can build up severely when the DFT is evaluated by brute-force. There are related, fast algorithms for the convolution of sequences of real numbers, for the discrete cosine transform, etc. Algorithmic details may be found in [H. J. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, Springer–Verlag, Berlin, 1982]. The computational complexity of the DFT is discussed by L. Auslander and R. Tolimieri [Is computing with the finite Fourier transform pure or applied mathematics, *Bull. (New Series) Amer. Math. Soc.*, **1** (1979) 847–897].

AIPS programs which use the FFT make use of the Cooley–Tukey algorithm. When an *array processor* is used to compute the large two-dimensional DFT's of data which reside on disk, as typically is required in synthesis mapping, the input/output time greatly exceeds the actual computation time.

FFT — See *fast Fourier transform algorithm*.

FITS format — (Flexible Image Transport System) a magnetic tape data format well-tailored for the transport of image data among observatories. The FITS format is recommended for bringing data into and out of AIPS. See [D. C. Wells, E. W. Greisen, and R. H. Harten, FITS: A flexible image transport system, *Astron. Astrophys. Suppl. Ser.*, **44** (1981) 363–370]. Also see *u-v FITS format* and *FITS tape*.

FITS tape — a magnetic tape containing data recorded in the *FITS format*. FITS format data blocks are 2880 bytes in length. The resultant *tape blocking efficiency* is 83%, 75%, and 61% at recording densities of 800, 1600, and 6250 bpi, respectively.

flagging — in AIPS, the act of discarding one or more visibility data points by setting a *u-v data flag* (*q.v.*). Compare *clipping*.

fringe rotator — in a correlating-type radio interferometer, a mechanism to introduce a time-varying phase shift into the local oscillator signal of a receiver, in order to reduce the frequency of the oscillations of the correlator output. Fringe rotation allows the correlator output (whose amplitude is proportional to visibility amplitude) to be sampled at a lower rate. The natural fringe frequency can be as high as 200 Hz on the VLA. The fringe rotation is chosen so that the fringe frequency for a point source located at the so-called *fringe stopping center* would be reduced to zero, or at least close to zero. Usually the fringe stopping center and the *delay tracking center* coincide; both then are called the *visibility phase tracking center*. For further details, see A. R. Thompson's Lecture No. 2 and L. R. D'Addario's Lecture No. 3 in the *1985 Summer School Proceedings*, and see R. M. Hjellming and J. Basart's Ch. 2 of the *Green Book*.

full-synthesis map — in earth-rotation aperture synthesis, with stationary interferometer elements, a *map* derived from an observation which is of such lengthy duration that the fullest possible *u-v coverage* is obtained (i.e., from an observation extending from "horizon to horizon"). Compare *snapshot*.

gain file — in AIPS, an *extension file*, associated with a *u-v data file*, in which a table of approximate *antenna/i.f. gains* (typically obtained by *self-calibration*) is stored.

Gaussian-tapered sinc function — A useful *gridding convolution function* (*q.v.*), of *support width* equal to the width $m\Delta u$ of m *u-v* grid cells, is given by the separable product of two Gaussian-tapered sinc functions, each of the form

$$C(u) = \begin{cases} \left(\frac{\pi u}{b\Delta u}\right)^{-1} e^{-\left(\frac{\pi u}{a\Delta u}\right)^2} \sin\frac{\pi u}{b\Delta u}, & |u| < \frac{m\Delta u}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

The choice $m = 6$, $a \simeq 2.52$, and $b \simeq 1.55$, yields what is, in a certain natural sense, an optimal gridding convolution function of this particular parametric form (see [F. R. Schwab, Optimal gridding, VLA Scientific Memo. No. 132]). Also see *spheroidal function*.

Gerchberg–Saxton algorithm — a simple iterative algorithm which, in the field of signal processing, is used for the extrapolation of band-limited signals—and, in image processing, for deconvolution. Assume that the Fourier transform $\hat{f}$ of an image f has been measured over a region B, and that f is known to be confined to a region A. Let χ_A denote the *characteristic function* of A and χ_B that of B. Denote the measured data by $\hat{g}_{\text{approx}}$—i.e., $\hat{g}_{\text{approx}} = \chi_B \hat{f} + \text{error}$. From the initial approximant f_0 ($f_0 \equiv 0$ may be used) a sequence f_n of successive approximants to f is obtained, via the formula

$$f_{n+1} = f_n + \mu\chi_A \cdot (\hat{g}_{\text{approx}} - \chi_B \hat{f}_n)^{\vee}.$$

Here, $^{\vee}$ denotes inverse Fourier transform, and μ is a fixed scalar, analogous to the *loop gain* parameter of the Högbom CLEAN algorithm.

To apply the algorithm in radio interferometry, one may identify χ_B with the *u-v sampling distribution* and think of A to be analogous to a *clean window*. Denoting the *dirty map* by g and the *dirty beam* by b, the iteration can be written as

$$f_{n+1} = f_n + \mu\chi_A \cdot (\hat{g} - \hat{b}\hat{f}_n)^{\vee} \quad (= f_n + \mu\chi_A \cdot (g - b * f_n)).$$

The Gerchberg–Saxton algorithm has been implemented by Tim Cornwell in an AIPS program named APGS. APGS includes an *ad hoc* nonnegativity constraint—at each iteration, any pixel value which would be driven negative is modified to become nonnegative. Convergence usually is sluggish.

Some algorithms which are very similar to the Gerchberg–Saxton algorithm are the Lent–Tuy algorithm, which is used in medical imaging, the Papoulis, or Papoulis–Youla algorithm, used in signal processing, and the so-called method of alternating orthogonal projections, used in image reconstruction. See [J. L. C. Sanz and T. S. Huang, Unified Hilbert space approach to iterative least-squares linear signal restoration, *J. Opt. Soc. Am.*, **73** (1983) 1455–1465] and references cited therein.

Gibbs' phenomenon — in the neighborhood of a discontinuity of a periodic function f, the overshoot and oscillation (or ringing) of the partial sums S_n of the Fourier series for f. In the vicinity of a simple jump discontinuity, S_n always overshoots the mark by about 9%, regardless how large n.

See [H. S. Carslaw, *Introduction to the Theory of Fourier's Series and Integrals*, Dover, New York, 1930, ch. IX].

In harmonic analysis, often the Fourier coefficients are multiplied by a weight function tending smoothly to zero at the boundaries of its *support*, in order to smooth out the discontinuities and thereby reduce the ringing in the synthesized spectrum. (This degrades the spectral resolution, however.) See *Hanning smoothing*. For a discussion of Gibbs' phenomenon in the context of VLA cross correlation analysis, see §IV-C of Larry D'Addario's Lecture No. 3 in the *1985 Summer School Proceedings*.

GIPSY — (Groningen Image Processing System) a data reduction system, similar in scope to AIPS, used in the Netherlands for analysis of Westerbork Synthesis Radio Telescope (WSRT) data.

global fringe fitting algorithm — an antenna-based algorithm (in the spirit of the *self-calibration algorithm*) for VLBI fringe search. For an n element array, the classical VLBI fringe fitting technique, a correlator-based method, requires the estimation of $n^2 - n$ parameters. The global fringe fitting method reduces this number to $3n - 3$. Expressing the *antenna/i.f. gain* for antenna k of the array as $g_k(t,\nu) = a_k e^{i\psi_k(t,\nu)}$ (here we include a frequency dependence) one has that the observed visibility on the i–j baseline, to first-order, is given by

$$\tilde{V}_{ij}(t,\nu) = a_i a_j V_{ij}(t_0,\nu_0) \times e^{\sqrt{-1}((\psi_i-\psi_j)(t_0,\nu_0)+(\tau_i-\tau_j)(t-t_0)+(\tau_i-\tau_j)(\nu-\nu_0))},$$

where V_{ij} is the true visibility, and where the τ_k are the *antenna residual fringe rates* and the τ_k the *antenna residual delays*.

Given a source model, one may solve for the $\psi_k(t_0,\nu_0)$, the τ_k, and the τ_k, using either a least-squares method or a Fourier transform method. Because of the overdeterminacy provided by a simultaneous solution for the parameters, this method allows proper delay and fringe rate compensation of data on baselines of too low signal-to-noise for the correlator-based method to work effectively. A full description of the method is given in [F. R. Schwab and W. D. Cotton, Global fringe search techniques for VLBI, *Astron. J.*, 88 (1983) 688–694]. This algorithm is implemented in the AIPS programs VBFIT and VBCOR.

graphics overlay plane — same as *graphics plane*.

graphics plane — a storage area within a TV display device, such as the I^2S, in which a full screen load of one-bit graphics information (labeling, plotting, axis lines, etc.) is stored. A typical I^2S unit is equipped with four graphics planes, each 512 pixels × 512 pixels in area. Compare *image plane*.

gray-scale display — a black-and-white display of a digitized *image*—typically either a photographic or a video display.

gray-scale memory plane — same as *image plane*.

Green Book — *An Introduction to the NRAO Very Large Array*, edited by R. M. Hjellming, NRAO, Socorro, NM—a useful reference on many of the technical aspects of the VLA.

green screen — same as *TEK screen*.

gridding convolution function — in radio interferometer mapmaking, a function C—usually supported on a square the width of, say, six u-v grid cells—with which the u-v measurement distribution is convolved. The purpose is twofold: 1) to interpolate and smooth the data, so that samples may be taken over the lattice points of a rectangular grid (in order that the fast Fourier transform algorithm may be applied) and 2) to reduce aliasing (the convolution in the u-v plane induces a taper in the map plane). See *aliased response*, *gridding correction function*, *cell-averaging*, *dirty map*, and *uniform weighting*.

With judicious choice of C, a high degree of aliasing suppression is possible. A high degree of suppression is desirable, even when there are no "confusing" radio sources very near the field of interest, because the effect is not only to reduce the spurious responses due to sources lying outside of the field of view, but also to reduce the response to sidelobes of the source of interest, which too are aliased into the map from outside the field of view. See *spheroidal function*.

gridding correction function — in radio interferometry, the reciprocal $1/\hat{C}$ of the Fourier transform (FT) of the *gridding convolution function* C. Since the map plane taper induced by the gridding convolution usually is very severe, the dirty map normally is corrected by pointwise division by the FT of the convolution function. Obviously C should be chosen such that $\hat{C}$ has no zeros within the region that is mapped. See *dirty map*.

gripe — in AIPS, an entry in the *gripe file* (*q.v.*).

gripe file — in AIPS, a disk file repository for formal reports of program *bugs*, and for formal complaints and suggestions of a more general nature. A mechanism by which the user may enter gripes into the gripe file is activated by the issuance of the AIPS verb *GRIPE*. The AIPS group provides prompt, written responses to all *gripes*.

Hanning smoothing function — in the analysis of power spectra, a weight function w by which the measured correlation function is multiplied, in order to reduce that oscillation (*Gibbs' phenomenon*) in the computed spectrum which is due to having sampled at only a finite number of lags. w, as a function of lag, is given by

$$w(\tau) = \begin{cases} \frac{1}{2}\left(1 + \cos\frac{\pi\tau}{\tau_{max}}\right), & |\tau| < \tau_{max}, \\ 0, & \text{otherwise}. \end{cases}$$

This is equivalent to convolving the discrete spectrum with the sequence $\{\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\}$.

Hanning smoothing sometimes is applied to the cross correlation measurements obtained in VLA spectral line observing, in order to reduce the effect of sharp bandpass filter cutoffs. It also is used frequently in radio astronomical autocorrelation spectroscopy. See *Gibbs' phenomenon*, and for more on smoothing see [R. B. Blackman and J. W. Tukey, *The Measurement of Power Spectra*, Dover, New York, 1958].

hard copy — computer output printed on paper (rather than, say, written on magnetic tape); e.g., a printed contour plot or gray scale display, or a listing of a catalog file.

hardware mount — the combined acts of installing a computer external storage module, such as a disk pack or a reel of magnetic tape, in some electro-mechanical unit (e.g., a disk drive or a tape drive) that provides computer access to this data storage medium, and placing that unit in readiness to be operated under computer control (e.g., positioning a magnetic tape at the *BOT marker*). Compare *software mount*.

header record — a distinguished record within a *data file*—generally the first record—which serves to define the contents of the other records in the file by supplying relevant parameters, units of measurement, etc.; also termed simply *header*.

In AIPS, however, the header record of each *primary data file* is stored apart from that file, in a file which is termed a "CB" file. And a directory, termed a *catalog file* (*q.v.*), or

"CA" file, of all of each user's primary data files on a given disk is stored on that disk. AIPS *extension file* headers are stored within the extension files themselves.

help file — in AIPS, a *text file*, whose contents may be displayed on the terminal screen of the interactive user, giving a brief explanation of a particular AIPS verb, adverb, pseudoverb, task, or miscellaneous general feature. Compare *explain file*.

Hermitian function — a complex-valued function, of one or more real variables, whose real part is an even function and whose imaginary part is odd. The Fourier transform (FT) of a real-valued function is Hermitian, and the inverse FT of a Hermitian function is real.

Since each of the radio brightness distributions $I(x,y)$, $Q(x,y)$, $U(x,y)$, and $V(x,y)$ representing *Stokes' parameters* is real-valued, *Stokes' visibility functions* have the property of *conjugate symmetry*: $V_I(-u,-v) = \overline{V}_I(u,v)$, $V_Q(-u,-v) = \overline{V}_Q(u,v)$, $V_U(-u,-v) = \overline{V}_U(u,v)$, and $V_V(-u,-v) = \overline{V}_V(u,v)$. (Here, $V_I = \hat{I}$, $V_Q = \hat{Q}$, etc., where $\hat{}$ denotes FT.)

history file — in AIPS, an *extension file* containing a summary of all, or most of the processing, by AIPS tasks, of the data recorded in all associated files.

Högbom CLEAN algorithm — a deconvolution algorithm devised by Jan Högbom for use in radio interferometry [J. A. Högbom, Aperture synthesis with a non-regular distribution of interferometer baselines, *Astron. Astrophys. Suppl. Ser.*, **15** (1974) 417–426]. Denote (the discrete representations of) the dirty map by g and the dirty beam by b. The algorithm iteratively constructs discrete approximants f_n to a solution f of the equation $b * f = g$, starting with an initial approximant $f_0 \equiv 0$. At the nth iteration, one searches for the peak in the residual map $g - b * f_{n-1}$. A δ-function component, centered at the location of the largest residual, and of amplitude μ (the *loop gain*) times the largest residual, is added to f_{n-1} to yield f_n. The search over the residual map is restricted to a region A termed the *clean window*. The iteration terminates with an approximate solution f_N either when N equals some iteration limit $N_{\max}$, or when the peak residual (in absolute value) or the r.m.s. residual decreases to some given level.

To diminish any spurious high spatial frequency features in the solution, f_N is convolved with a narrow elliptical Gaussian function h, termed the *clean beam*. Generally h is chosen by fitting to the central lobe of the dirty beam. Also, one generally adds the final residual map $g - b * f_N$ to the approximate solution $f_N * h$, in order to produce a final result, termed the *clean map*, with a realistic-appearing level of noise. See *super-resolution*.

host computer — In the parasitic relationship of a computer program or program package, such as AIPS, to the computer on which it runs, the latter is termed the *host computer*. Also, in the master–slave relationship of a computer to one of its peripheral devices, such as an array processor, the master may be termed the *host*.

hue — one of the three basic parameters (*hue*, *intensity*, and *saturation*) which may be used to describe the physical perception of the light that reaches one's eye. Hue, which is also termed *tint*, or simply *color*, refers to the dominant wavelength of the coloration, at a given location in an image or scene. The term also may be used to describe a multimodal color spectrum—e.g., one speaks of a purple hue. Different spectral distributions of light, of identical intensity and saturation, are capable of producing identical retinal responses; these unique responses comprise the set of perceptible hues.

Color matching tests have established that there are three basic types of human retinal receptors, whose peak responses are to red, green, and blue light. These are the three *primary hues* used in additive color mixing—e.g., in digital image display. They may be used to produce all, or virtually all, of the perceptible hues.

See *C.I.E. chromaticity diagram*.

hybrid mapping algorithm — an algorithm for calibration of radio interferometer data which is essentially equivalent to the *self-calibration algorithm* (*q.v.*) (used in VLA data reduction), except in that it makes explicit use of the *closure phase* and *closure amplitude* relations, rather than explicit use of the relation $\tilde{V}_{ij} = g_i \overline{g}_j V_{ij}$ relating observed visibility to the product of the true visibility and a pair of *antenna/i.f. gains*. Hybrid mapping, which is used extensively in VLBI data reduction, is described in [A. C. S. Readhead *et al.*, Mapping radio sources with uncalibrated visibility data, *Nature*, **285** (1980) 137–140].

Either algorithm (assuming that one cares to make some distinction) can be applied to data obtained with connected- (e.g., the VLA) and non-connected-element interferometers (e.g., VLBI arrays). Any differences in the results produced by the two algorithms would be attributable primarily to differences in the effective weighting of the data (in particular, early implementations of both algorithms discarded data which could have been used to obtain overdetermined solutions for the calibration parameters).

IIS — See *I^2S*.

image — in the context of AIPS, any finite-volume, linear, rectangular, or hyper-rectangular array of pixels; e.g., a digitized photograph, or a radio map. The term also is used (less technically) to refer to the *display* of data—e.g., a television picture of a radio map.

image catalog — in AIPS, a disk file containing data records describing the data stored on the TV display device *image planes*. These records are essentially identical in structure to the *header records* stored in the *catalog file*. The data in the image catalog furnish the information that is required for proper axis labeling, pixel value retrieval, etc.

image file — in AIPS, a *primary data file* whose content is an *image*.

image plane — a storage area within a TV display device, such as the *I^2S*, in which a full screen load of single word pixels is stored. A typical I^2S unit is equipped with four image planes, each 512 pixels × 512 pixels in area (each pixel is represented by eight bits). Often several image planes are used at one time—either for black-and-white or *pseudo-color* display of a large image, sections of which may occupy different image planes—or for *false color* or *true color* display of a smaller image, now using, say, three image planes—one to control each of the three electron guns (for red, green, or blue phosphor) in the TV display. Compare *graphics plane*.

image reconstruction — the attempted recovery of an *image* after it has undergone the distorting effects, the blurring, etc., produced by some physical measurement and recording device, such as a camera, a radio interferometer, or a tomography machine. The operation of many measurement devices can be adequately modeled by a linear Fredholm integral equation of the first kind. In the two-dimensional case, e.g., one assumes that the measurement $g(x,y)$ is related to the undistorted image $f(x,y)$ by the equation

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K(x,y,x',y') f(x',y')\, dx'\, dy' + \epsilon(x,y)\,.$$

(Often it is convenient to use the more compact, operator notation, $g = \mathrm{K}f + \epsilon$.) The kernel K of the equation is called the *point spread function*, (*q.v.*). Measurement error and the error arising from any simplifying assumptions are lumped together into the $\epsilon(x, y)$ term. Some particularly well-behaved measurement systems can be adequately modeled by a simple convolution equation, in which case K is given by $K(x, y, x', y') = h(x - x', y - y')$. This is the case, e.g., when the VLA is used to observe a small 'unconfused' radio source; then g may be identified with the *dirty map* and h with the *dirty beam*. Or when K, considered as a function of (x, y), is given at each (x', y') by the *delay beam* for that position, the equation models the *bandwidth smearing effect* (*q.v.*); as the bandwidth $\rightarrow 0$, the convolution model again becomes valid.

Except in trivial cases, solution of the Fredholm equation always is an ill-posed problem. Mild conditions on K and f (the classical 'Picard conditions'—see F. Smithies [*Integral Equations*, Cambridge Univ. Pr., London, 1958]) ensure the existence of (non-unique) solutions when $\epsilon \equiv 0$. But, because of the effect of measurement noise, one usually does not seek an exact solution, but rather an approximate solution—one which fits the data to within the measurement errors. Uniqueness and regularity of the computed approximate solution are obtained by imposing such constraints as known *support*, nonnegativity, and smoothness conditions. See *regularization method*. Also see H. C. Andrews and B. R. Hunt [*Digital Image Restoration*, Prentice–Hall, Englewood Cliffs, NJ, 1977] and *phaseless reconstruction*.

inputs file — in AIPS, a *text file*, whose contents may be displayed on the terminal screen of the interactive user, giving a summary of the *adverbs* relevant to a given *verb* or a given AIPS *task*.

instrumental polarization — any contamination of a polarization measurement by an instrument's response to an undesired polarization state. In radio interferometry, the instrumental polarization arises mainly from feed imperfections and from plumbing leaks between the feeds and the receiver front-ends. One tries to remove the instrumental polarization by applying corrections derived from observations of calibration sources whose polarization properties are known. Within AIPS, there is, at present, no facility for polarization calibration. The polarization calibration of VLA data normally takes place on the DEC–10 computer at the VLA. For more details, see Carl Bignell's Lecture No. 4 in the *1985 Summer School Proceedings*. See *beam squint*.

intensity — one of the three basic parameters (*hue*, *intensity*, and *saturation*) which may be used to describe the physical perception of color. Intensity is a measure of the energy of the spectral distribution, at a given point in an image or scene, weighted by the spectral response of the visual system. *Luminance* is the energy of the physical spectrum, but not weighted by the visual response. *Brightness* sometimes is used synonymously with either term.

See *C.I.E. chromaticity diagram*.

invisible distribution — in the context of radio interferometry, a function f (or a generalized function—or distribution) whose Fourier transform $\hat{f}$ vanishes everywhere that the interferometer pairs have sampled. This term was introduced by R. N. Bracewell and J. A. Roberts [Aerial smoothing in radio astronomy, *Austr. J. Phys.*, **7** (1954) 615–640]. Also see *principal solution*.

For an actual interferometer, there exist fewer physically plausible invisible distributions than for an idealized interferometer. This is because each visibility sample is not a point sample of $\hat{f}$, but rather some kind of local average. By the *Paley–Wiener theorem*, if $\hat{f}$ is nontrivial and vanishes in some open neighborhood, then f cannot be of *compact support*, and hence it may be considered implausible.

IPL — (Initial Program Load) same as *boot*.

isoplanaticity assumption — in the context of radio interferometry (the term is used too in optics), the assumption that over each element of an array all wavefronts arriving from different parts of the sky to which the interferometer pairs are sensitive are subject to identical atmospheric phase perturbations. A patch of sky over which the assumption is valid is referred to as an *isoplanatic patch*.

Approximate validity of the isoplanaticity assumption is a necessary condition for the success of calibration (*self-calibration*, in particular) of radio interferometer data (from an earth-based array) if one is to rely on a model incorporating time-varying *antenna/i.f. gains*, one per antenna, whose arguments (or phases) are to include the atmospheric phase corruption. However, see F. R. Schwab [Relaxing the isoplanatism assumption in self-calibration; applications to low-frequency radio interferometry, *Astron. J.*, **89** (1984) 1076–1081].

I^{2}S — (International Imaging Systems Models 70 and 75) a TV display device, capable of both black-and-white and color display, manufactured by the Stanford Technology Corporation. At an AIPS site typically it is equipped with four 512 pixel×512 pixel eight-bit *image planes*, four one-bit *graphics planes*, a *trackball*, and sometimes an *ALU*. The eight-bit pixel representation (in the image planes) allows the intensity of each of the three electron gun beams to be set at any of 256 discrete levels. (Actually, 1024 levels can be used, because of an extra two bits of capability provided in the *transfer function* tables and the internal arithmetic unit.) An I^{2}S is attached to three of the NRAO's computers on which the AIPS system runs (the VLA and Charlottesville Vaxes).

line editor — a *text editor* (*q.v.*) which allows the modification of single lines or records within a text file, but one which does not allow the simultaneous modification of more than one line. *SOS* and *SEDIT* are both line editors. *Screen editors* (*q.v.*) are more versatile than line editors.

lobe rotator — same as *fringe rotator*, (*q.v.*).

loop gain — in the Högbom CLEAN algorithm, the fraction μ of the largest residual which is used in determining the amplitude, or flux, of a *clean component*. Convergence can be achieved for μ in the range $(0, 2)$, but generally a small value, say $\mu = \frac{1}{10}$, is recommended, especially in dealing with extended sources. See *Högbom CLEAN algorithm*.

luminance — See *intensity*.

l_1 solution algorithm — See *self-calibration gain solution algorithm*.

l_2 solution algorithm — See *self-calibration gain solution algorithm*.

major cycle — In the *Clark CLEAN algorithm* (*q.v.*), a number of minor cycles, or inner iterations, followed by the computation by the FFT algorithm of the full residual map, comprise a major cycle.

map — an *image*, one or more of whose coordinate axes represents some spatial coordinate.

maximum entropy method — a *regularization method* (*q.v.*) for the numerical solution of ill-posed problems, given noisy data, in which the regularizing (or smoothing) term—which measures the roughness of the computed approximate solution $\tilde{f}$—is given by the negative of the Shannon entropy

of $\tilde{f}$, $-H(\tilde{f})$: in the continuous case, letting A denote the domain of definition of $\tilde{f}$, $H(\tilde{f}) \equiv -\int_A \tilde{f}(x) \log \tilde{f}(x)\, dx$, where $\tilde{f}$ has been normalized so that $\int_A \tilde{f}(x)\, dx = 1$ (and $0 \log 0 \equiv 0$); and in the discrete case, $H(\tilde{f}) \equiv -\sum \tilde{f}(x_i) \log \tilde{f}(x_i)$, where $\tilde{f}$ has been normalized so that $\sum \tilde{f}(x_i) = 1$. The underlying philosophy of the method, espoused early on by Jaynes ("Jaynes' method of prior estimation", [E. T. Jaynes, Prior probabilities, *IEEE Trans. Syst. Sci. Cyb.*, **SSC-4** (1968) 227–241]) and by J. P. Burg at a 1967 meeting of the Society of Exploration Geophysicists, is that one is being "maximally noncommittal" in regard to the insufficiency of the data if one maximizes the entropy, and thus minimizes the "information content", of $\tilde{f}$, subject to the constraint that $\tilde{f}$ should agree with the given data.

For one-dimensional discrete convolution equations, with noiseless, regularly-spaced data, there exists a closed-form solution—for other cases, iterative methods are used, as with other forms of the regularization method.

Use of the method in radio astronomy was encouraged by J. G. Ables in 1972 in public lectures, and it now is in common use in radio interferometry (cf. [S. F. Gull and G. J. Daniell, Image reconstruction from incomplete and noisy data, *Nature*, **272** (1978) 686–690]). Nonnegativity of the computed solution is a natural by-product of the method. For reconstruction of polarized brightness distributions in interferometry (Stokes' Q, U, and V), which, unlike the total intensity, may assume negative values, Ponsonby has derived an appropriate generalization of the method [J. E. B. Ponsonby, An entropy measure for partially polarized radiation ..., *Mon. Not. R. Astr. Soc.*, **163** (1973) 369–380]. See *Variational Method*.

MAX IV — the operating system of the Modcomp Classic computer which is used for AIPS reduction in Charlottesville. See *operating system*.

memory page — See *virtual memory page*.

memory paging — same as *virtual memory page swapping*.

memory thrashing — an excessive amount of *virtual memory page swapping* (*q.v.*) on a computer (such as the Vax) with a virtual memory operating system. A condition of memory thrashing is likely to occur whenever too many programs with large memory requirements are active (a single program with excessive memory requirements also can cause memory thrashing).

message file — in AIPS, a *text file* containing progress report messages generated during the execution of AIPS *tasks* and also containing a chronicle of the user's interaction (via *verb* commands) with AIPS. Each AIPS user is assigned a message file, the contents of which may be printed out, typed upon a terminal display screen, or emptied—at will—by invoking the appropriate verb command. See *AIPS monitor*.

message terminal — same as *AIPS monitor*.

minor cycle — in the *Clark CLEAN algorithm* (*q.v.*), an inner iteration, in which the peak residual over a subregion (the *clean window*) of the full residual map is found and is used to obtain the next successive iterate. Compare *major cycle*.

microcode — See *array processor microcode*.

monitor — See *AIPS monitor* or *Conrac monitor*.

$m \times n$ map — The convention adopted for AIPS is opposite the standard matrix algebra terminology: whereas an $m \times n$ matrix is comprised of m rows and n columns, an $m \times n$ *map* or *image* in AIPS has, in the usual display format, m pixels along the horizontal axis (usually termed the x-axis) and n pixels along the vertical axis (usually termed the y-axis). Moreover, pixels of a two-dimensional map in the usual display format are numbered from the bottom left-hand corner: the pixel location specified by the ordered pair (i, j) is in column number i and row number j, counting from the bottom left. In other than two-dimensional "images", the $(1, \ldots, 1)$ pixel is also said to be at the "bottom left corner" (BLC), just as in the two-dimensional case. See *data cube*, *pixel coordinates*, and *coordinate reference pixel*.

MX — See *"battery-powered" CLEAN algorithm*.

natural weighting — See *uniform weighting*.

negative bowl artifact — See *zero-spacing flux*.

non-closing offset — See *correlator offset*.

Nyquist sampling rate — the slowest rate of sampling which, according to the *Shannon sampling theorem* (*q.v.*), would allow a band-limited function $f(t)$ to be recovered via the *Shannon series*. If the smallest symmetric interval which contains the *support* of the Fourier transform of f is the interval $[-a, a]$, then the Nyquist sampling rate for f is $2a$; i.e., the interval between samples (the *sampling period*) must be less than the *reciprocal bandwidth* $1/2a$. The terms *oversampling* and *undersampling* refer to sampling at rates faster or slower than the Nyquist rate. The difference between f and the Shannon series formed from too coarsely spaced samples is called *aliasing*.

page — See *virtual memory page* and *terminal page*.

page swapping — See *virtual memory page swapping*.

Paley–Wiener theorem — The classical Paley–Wiener theorem says that a square-integrable complex-valued function $\hat{f}$, defined over the real line, can be extended off the real line as an entire function of exponential type $\leq 2\pi a$ if and only if $f(x) \equiv 0$ for $|x| > a$—i.e., iff $\hat{f}$ is band-limited to $[-a, a]$ (here $\hat{}$ denotes Fourier transform). (An everywhere-analytic function $g(z)$ is said to be of exponential type $\leq A$ if $\exists c$ such that, for all z, $|g(z)| \leq ce^{A|z|}$.) For a derivation, see H. Dym and H. P. McKean [*Fourier Series and Integrals*, Academic Press, 1972]. The *Shannon series* is a means of extending $\hat{f}$ to $\mathbf{C}$. The extension of the Paley–Wiener theorem to the case of generalized functions (to tempered distributions) is called the Paley–Wiener–Schwartz theorem.

The Fourier transform $\hat{f} : \mathbf{R}^n \to \mathbf{C}$ of a function f with support in a given n-dimensional convex compact set K can be analytically extended to all of $\mathbf{C}^n$. Growth properties on $\hat{f}$ which are sufficient in order for the converse to hold are given by K. T. Smith, D. C. Solomon, and S. L. Wagner [Practical and mathematical aspects of the problem of reconstructing objects from radiographs, *Bull. Amer. Math. Soc.*, **83** (1977) 1227–1270] (in addition to the classical version of the multidimensional Paley–Wiener theorem, for rectangular K, they give versions with tighter growth bounds, and for arbitrary convex K). Smith *et al.* use the Paley–Wiener theorems to establish indeterminacy theorems for tomographic reconstruction. Their results are also relevant to Fourier synthesis, because of the connection between the two-dimensional Fourier transform and the one-dimensional Radon transform. The Paley–Wiener theorems have also been used in establishing results on the problem of *phaseless reconstruction* (*q.v.*) and in proving the convergence of constrained *Gerchberg–Saxton*-type algorithms (see A. Lent and H. Tuy [An iterative method for the extrapolation of band-limited functions, J. Math. Anal. Appl., **83** (1981) 554–565]).

phaseless reconstruction — the reconstruction of an image f (see *image reconstruction*) from knowledge of (only)

the magnitude $|\hat{f}|$ of the Fourier transform of f (and usually from only partial knowledge of $|\hat{f}|$). Phaseless reconstruction has been considered for the NRAO's proposed millimeter wave interferometer array [T. J. Cornwell, Imaging of weak sources with compact arrays, NRAO Millimeter Array Memo. No. 12]. Recent results on phaseless reconstruction appear in the JOSA Feature Issue on Signal Recovery [*J. Opt. Soc. Am.*, **73** No. 11 (Nov. 1983)]. Also see the papers by J. R. Fienup and by R. H. T. Bates *et al.* in the *1983 Sydney Conference Proceedings*.

phase tracking center — same as *visibility phase tracking center*, (*q.v.*).

physical memory — core or semiconductor memory within a computer (as opposed to slower memory—virtual memory, disk storage, magnetic tape footage, etc.). A typical Vax is equipped with a physical memory 3–4 megabytes in size. The Modcomp Classic computer in Charlottesville has $\frac{1}{2}$ megabyte of physical memory.

pillbox — See *cell-averaging*.

pixel — (*pic*ture *el*ement) an element of a digitized image (or of a map). A pixel is characterized by its position in the image and by its numerical value. See *m* × *n map*, *coordinate reference pixel*, and *pixel coordinates*.

pixel coordinates — in an AIPS *image file*, the *pixels* are numbered consecutively, beginning with (1,...,1) at the bottom left corner (*BLC*) of the image. See *coordinate reference pixel* and *m* × *n map*.

plot file — an AIPS extension file containing plotting information, in the form of the commands which are necessary in order for a line drawing peripheral device, such as a Calcomp or other pen plotter, a green screen, or an electrostatic printer/plotter, to generate a plot.

point source response — same as *point spread function*.

points per beam — in a digitized radio map, the characteristic width, somehow defined, of the major lobe of the *beam* pattern, or *point spread function*, divided by the *pixel* separation. Ordinarily the number of points per beam is calculated by measuring the narrowest diameter of the 50% contour level of the major lobe of the beam. To avoid excessively severe discretization error, deconvolution algorithms such as the *Högbom CLEAN algorithm* and the *maximum entropy method* require, as a rule-of-thumb, at least three (and preferably 4–5) points per beam.

point spread function — (PSF) 1. the response of a system or an instrument to an impulsive, or point source, input. 2. in radio interferometry, the response of the instrument to a point, or unresolved, radio source—a fancy term for *beam*. Ignoring instrumental effects, such as finite bandwidth and finite integration time, the response does not depend upon the displacement of the source away from the *visibility phase tracking center*—hence the term *space-invariant PSF (SIVPSF)*, and the contrary term *space-variant PSF (SVPSF)*.

A so-called linear space invariant measurement system (i.e., a linear system with an SIVPSF) is equivalently described as a system which can be modeled by a convolution equation; a linear space-variant measurement system is modeled by a more general linear Fredholm integral equation of the first kind. See *image reconstruction*.

POPS — (People-Oriented Parsing System) the parser, or command interpreter, embedded within the AIPS program; that part of the AIPS program which attempts to interpret the user's commands (*POPS symbols*) and then initiate the appropriate reaction. POPS is used in other astronomical data reduction programs at the NRAO: in Condare, TPOWER/SPOWER, and the Tucson 12 m single-dish packages.

POPS procedure — See *POPS symbols*.

POPS symbols — The AIPS user's primary means of communicating his wishes to AIPS is by typing commands, termed *POPS symbols*, at the keyboard of a computer terminal. There are four classes of POPS symbols: *adverb*, *verb*, *pseudoverb*, and *procedure*. An *adverb* is a *symbol* representing the storage area for a datum or for data that are used to control the action of verbs, tasks, and procedures; that is to say that the adverb symbols are used to set *control parameters*. A *verb* is a symbol which causes POPS (or AIPS) to initiate some action after POPS has finished interpreting, or compiling, the command line typed at the computer terminal. A *pseudoverb* is a symbol which suspends, temporarily, the normal parsing of an input line and which causes some action to take place while the line is being compiled, and, possibly, after compilation. A *procedure* is a symbol representing a pre-compiled sequence of POPS symbols. Also see *task*.

primary beam correction — in radio interferometry, the multiplicative correction of a radio *map* by the reciprocal of an average of the power patterns of the array elements. Measurements of the primary beam parameters of the 25 m VLA elements are given by Peter Napier and Arnold Rots in the memorandum [VLA primary beam parameters, VLA Test Memo. No. 134, Feb. 1982]. There an average power pattern and its reciprocal are approximated by radial functions, polynomials in the distance from the pointing position. The AIPS task PBCOR is used to apply this correction to VLA maps. The appropriate correction at large distances from the pointing position is not well-determined, thus PBCOR "blanks" the map pixel values beyond a certain radius (see *blanked pixel*).

primary data file — in AIPS, either a *u-v data file*, containing measurements of the visibility function of a radio source, or an *image file*, containing a digitized image or a radio map. Compare *extension file*.

principal solution — in the context of radio interferometry, the inverse Fourier transform of the *u-v measurement distribution*; i.e., the *dirty map* (*q.v.*) in sense 1 of the definition. This term was introduced by R. N. Bracewell and J. A. Roberts [Aerial smoothing in radio astronomy, *Austr. J. Phys.*, **7** (1954) 615–640]. Except in the trivial case, the principal solution to the mapping problem in interferometry is a physically implausible solution, because the principal solution has not the property of *compact support*.

An *invisible distribution* (*q.v.*) added to the principal solution yields another solution—i.e., another brightness distribution which is consistent with the observations.

procedure — See *POPS symbols*.

prolate spheroidal wave function — an eigenfunction of the finite, or truncated, Fourier transform—more precisely, for given c, one of the countably many solutions of the integral equation

$$(*) \qquad \nu f(\eta) = \int_{-1}^{1} e^{ic\eta t} f(t)\, dt\,;$$

equivalently, a solution of the differential equation $(1-\eta^2)f'' - 2\eta f' + (b - c^2\eta^2)f = 0$; or, equivalently, a solution of the wave equation in a system of prolate spheroidal coordinates. The eigenfunction of (*) associated with the largest eigenvalue ν is termed the 0-order solution.

If we want a gridding convolution function C, of *support width* equal to the width of m grid cells, that is optimal in

the sense that its Fourier transform $\hat{C}$ has the property that the concentration ratio

$$\frac{\iint_{\text{map}} |\hat{C}(x,y)|^2\,dx\,dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |\hat{C}(x,y)|^2\,dx\,dy}$$

is maximized, then C is the separable product of two 0-order prolate spheroidal wave functions, with $c = \pi m/2$. See *gridding convolution function* and *spheroidal function*.

prompt character — a character (often the dollar sign "$" or the greater-than sign ">") which the computer program or the operating system prints on the terminal screen of the interactive user in order to prompt, or invite, a typed response from the user. The AIPS program's standard prompt character is the greater-than sign, and on the Vax and Modcomp systems at the NRAO the operating system's prompt character is the dollar sign. Thus, most commands (or *POPS symbols*) peculiar to AIPS must be typed on a line beginning with the >-character, and any command to the operating system, such as the command to mount a tape, must be typed on a line beginning with the $-character.

When operating in some lesser-used, special modes, AIPS employs other prompt characters: ":" for procedure building, ";" for procedure editing, "!" for entry of gripes, "<" for batch file preparation, and "#" for parameter reading.

Prussian helmet CLEAN algorithm — a modified version of the Högbom CLEAN algorithm, devised by Tim Cornwell. The idea is to drive the CLEAN algorithm toward an approximate solution f of minimal Euclidean norm—i.e., to find an f consistent with the data, confined to the clean window, comprised of a small number of point components, and such that $\iint_{\substack{\text{clean}\\ \text{window}}} [f(x,y)]^2\,dx\,dy$ is minimized. This is accomplished by adding a δ-function of amplitude ω, centered at the origin, to the dirty beam, and then just proceeding as normal with the CLEAN algorithm. Proper choice of ω depends on the distribution of measurement errors. See [T. J. Cornwell, A method of stabilizing the clean algorithm, *Astron. Astrophys.*, 121 (1983) 281–285]. A provision for this modification is incorporated in the AIPS tasks APCLN and MX. See *regularization method*.

pseudo-AP — See *pseudo-array processor*.

pseudo-array processor — in AIPS, the term which is applied to a collection of Fortran subroutines which may be used to emulate the operation of an FPS Model AP–120B array processor. At those AIPS sites which do not have an array processor, the AIPS tasks which normally would make use of an array processor use the pseudo-array processor subroutines instead. See *array processor*.

pseudo-color display — In digital imagery, a *pseudo-color* display is one which is derived from a single real-valued function $f(x,y)$ and a mapping $\mathbf{R}^1 \to \mathbf{R}^3$ that controls the *hue*, *intensity*, and *saturation*—or, equivalently, the proportions in an additive mixture of three primary hues—of the coloration at each *pixel* coordinate (x,y) of the display, according to the value of $f(x,y)$. A pseudo-color display might be used, for example, to represent measurements of the intensity of the radio continuum flux density of a source.

Compare *false color display* and see *color contour display*.

pseudo-continuum u-v data file — in VLA spectral line data reduction, a *u-v data file* containing the visibility measurements from a small number of spectral line channels, recorded in the same format as continuum visibility data. The purpose is to enable the use, for spectral line data analysis, of programs originally intended only to handle continuum data reduction.

pseudoverb — See *POPS symbols*.

PSF — See *point spread function*.

Q-routine — in AIPS, a primitive level subroutine designed to function on a particular manufacturer's production model of an *array processor*. A goal of the AIPS project is to construct libraries of Q-routines—one library appropriate to each model of array processor which might be used in conjunction with AIPS—with identical names, argument lists, and functionality. Existing Q-routines emulate the standard library of Floating Point Systems, Inc.'s, model *AP–120B array processor*.

quick boot — an abbreviated boot procedure. See *boot*.

RANCID — (Real (or Radio) Astronomical Numerical Computation and Imaging Device) the name by which the AIPS data reduction system formerly was known.

re-boot — Having booted once already, one *re-boots*. See *boot*.

regularisation method — in the numerical solution of ill-posed problems, given noisy data, a method in which the original problem is converted into a well-posed problem by requiring of the solution to the modified problem (which now is an approximate solution to the original problem) that it satisfy some smoothness constraint. The prototypical ill-posed problem has the form $Kf = g + \epsilon$, where K is a known linear integral operator (e.g., a convolution operator), where $g + \epsilon$, which is given, represents some noisy measurement, and where f is unknown. In the context of radio interferometry, one may take $g + \epsilon$ to be the *dirty map* and K to be the operator which convolves the "true" radio source brightness distribution f with the *dirty beam*. Now, denoting our approximate solution to the ill-posed problem by $\tilde{f}$, $\tilde{f}$ is found by minimizing the expression

$$(1-\lambda)\|g - K\tilde{f}\|^2 + \lambda S(\tilde{f}),$$

for some given choice of the *regularization parameter* λ, $0 < \lambda < 1$. $\|g - K\tilde{f}\|^2$ is the mean squared residual (occasionally some other measurement of the error is used), and $S(\tilde{f})$ is a measure of the roughness of the computed solution—say, some power of a norm or seminorm of $\tilde{f}$, or a similar quantity, such as the negative of the (Shannon) entropy of $\tilde{f}$.

Proper choice of λ must be based on statistical considerations which depend on the distribution of measurement errors; often, one chooses λ in order achieve an *a priori* reasonable value of the mean squared residual. The *maximum entropy method*, *Tikhonov regularization*, and the *Prussian helmet CLEAN algorithm* are special cases of the regularization method. Appropriate choice of S is discussed by J. Cullum [The effective choice of the smoothing norm in regularization, *Math. Comp.*, 33 (1979) 149–170], and the choice of S and λ, by a statistical method known as "cross validation", is described by G. Wahba [Practical approximate solutions to linear operator equations when the data are noisy, *SIAM J. Numer. Anal.*, 14 (1977) 651–677]. Often, some Sobolev norm is chosen for S.

Usually, in addition to the smoothness constraint, f is assumed to be of known, *compact support*. Other constraints, such as nonnegativity, may be included as well. In the case in which the data are exact—i.e., when $\epsilon = 0$, so that $g = Kf$—one may obtain the regularized solution corresponding to $\lambda = 0$ as the limit of regularized solutions $\tilde{f}_\lambda$ as $\lambda \to 0$. See *Variational Method*. Also see D. M. Titterington [General structure of regularization procedures in image reconstruction, *Astron. Astrophys.*, 144 (1985) 381–387].

regularisation parameter — in the *regularization method* (*q.v.*) for the solution of ill-posed problems, a smoothing parameter λ, $0 < \lambda < 1$, which controls the trade-off between an error term, measuring agreement of the computed solution $\tilde{f}$ with the given data, and a term $S(\tilde{f})$, which measures the roughness of $\tilde{f}$. I.e., λ controls the amount of "regularization". See *super-resolution*.

re-IPL — same as *re-boot*.

residual delay — Expressing the *antenna/i.f. phase*, ψ_k, for antenna k of a VLBI array as a function of frequency as well as of time, the residual delay on the i–j baseline at (t_0, ν_0) is given by $\tau_{ij} \equiv \left.\frac{\partial(\psi_i - \psi_j + \phi_{ij})}{\partial \nu}\right|_{(t_0,\nu_0)}$, where ϕ_{ij} denotes the visibility phase on the i–j baseline. (The partial w.r.t. t is called the *residual fringe rate*.) Usually the major contributor to residual delay is the difference in the station clock errors. The residual delay is a group delay, rather than a phase delay. It is termed residual because it is assumed that geometric effects have already been compensated for.

The "antenna components" of τ_{ij}, namely $\tau_k \equiv \left.\frac{\partial \psi_k}{\partial \nu}\right|_{(t_0,\nu_0)}$, are called the *antenna residual delays*. They are among the solution parameters of the global fringe fitting algorithm for VLBI. See *residual fringe rate* and *global fringe fitting algorithm*.

residual fringe rate — Expressing the *antenna/i.f. phase*, ψ_k, for antenna k of a VLBI array as a function of frequency as well as of time, the residual fringe rate on the i–j baseline at (t_0, ν_0) is given by $r_{ij} \equiv \left.\frac{\partial(\psi_i - \psi_j + \phi_{ij})}{\partial t}\right|_{(t_0,\nu_0)}$, where ϕ_{ij} denotes the visibility phase on the i–j baseline. (The partial w.r.t. ν is called the *residual delay*.) Usually the major contributor to residual fringe rate is the drift of the station clocks.

The "antenna components" of r_{ij}, namely $r_k \equiv \left.\frac{\partial \psi_k}{\partial t}\right|_{(t_0,\nu_0)}$, are called the *antenna residual fringe rates*. They are among the solution parameters of the global fringe fitting algorithm for VLBI. See *residual delay* and *global fringe fitting algorithm*.

resolution — See *spatial resolution*.

restoring beam — same as *clean beam*.

roam — See *TV roam*.

run file — in AIPS, a *text file* written by an AIPS user and containing a sequence of AIPS commands (*POPS symbols*). Run files are useful for the storage of strings of commands which one might wish to execute repeatedly (in particular, for the storage of lengthy *procedures*). The run files for all users at a particular AIPS installation are stored in a common area. These files ordinarily are created through use of one of the standard *text editors* of AIPS' host computer.

sampling theorem — See *Shannon sampling theorem*.

saturation — one of the three basic parameters (*hue*, *intensity*, and *saturation*) which may be used to describe the physical perception of color. Saturation is a measure of the (perceived) narrowness of the color spectrum, or the difference of the hue from a gray of the same intensity. Neutral gray—or a "white" spectrum—is termed 0% saturated, and a monochromatic spectrum is termed 100% saturated.

See *C.I.E. chromaticity diagram*.

scratch — 1. The act of deleting a data file—i.e., surrendering the storage medium space which that file occupies—is termed *scratching* the data file. Use of the term *delete* may be preferable, but *scratch* is more common among AIPS users. One who is about to delete a data file may wish first to create a back-up copy. See *back-up*. 2. an adjective meaning *temporary*, as in *scratch file*.

In AIPS a primary data file and all of its associated extension files can be deleted by means of the verb *ZAP*.

scratch file — a data file intended for temporary storage (esp., of data which represent intermediate results—i.e., *scratch*work). Many of the AIPS tasks use scratch files; the necessary scratch files are created and destroyed automatically by the tasks. However, when an AIPS task *crashes*, sometimes a scratch file remains.

screen editor — a *text editor* (*q.v.*) which, unlike a *line editor*, allows the simultaneous modification of more than one line or record within a text file. For example, a mechanism to facilitate alignment of margins often is incorporated by a screen editor. *EDT* and *EMACS* are both screen editors.

scroll — See *terminal scroll* and *TV scroll*.

SEDIT — an unsophisticated text editor (a *line editor*) available on the Modcomp computers. See *text editor*.

self-calibration algorithm — Many of the systematic errors affecting interferometer visibility measurements may be assumed to be multiplicative and ascribable to individual array elements. That is, in an n element array, the observations on the $n(n-1)/2$ baselines are afflicted by n sources of systematic error, the so-called *antenna/i.f. gains* $g_k(t)$. Given a rough estimate of the true source visibility, a model obtained, say, by mapping and cleaning roughly calibrated data, one may solve for the unknown gains—and it is not unreasonable to do so, because there are $(n-1)/2$ times more observations than antenna gains. The number of degrees of freedom can be held further in check by assuming that the $g_k(t)$ are slowly-varying or that they are of unit modulus (i.e., that no amplitude errors are present), or by designing an array with redundant spacings.

Having once solved for the unknown g_k, one may correct the data, make another map, and repeat the process. This iterative scheme, which yields successive approximations to the true radio source brightness distribution, is known as *self-calibration*. Self-calibration is essentially identical to the technique of *hybrid mapping*, which is widely used in VLBI. See *self-calibration gain solution algorithm*; also see Tim Cornwell's Lecture No. 9 in the *1985 Summer School Proceedings* and the review paper by T. J. Pearson and A. C. S. Readhead [Image formation by self-calibration in radio astronomy, *Ann. Rev. Astron. Astrophys.*, **22** (1984) 97–130].

self-calibration gain solution algorithm — In self-calibration, the unknown *antenna/i.f. gains* $g_k(t)$ may be approximated by minimizing a functional $S(g_1, \ldots, g_n)$ given by a weighted discrete l^p norm of the residuals:

$$S(\mathbf{g}) = \left(\sum_{1 \le i < j \le n} w_{ij} \left| \tilde{V}_{ij} - g_i \bar{g}_j V_{ij} \right|^p \right)^{1/p} .$$

Here $\tilde{V}_{ij}$ is the visibility measurement obtained on the i–j baseline (at a given instant), V_{ij} is the corresponding *model* visibility, and w_{ij} is a suitably chosen weight. Usually the g_k may be assumed not to vary too rapidly with time, so that one may minimize, instead, the functional

$$S(\mathbf{g}) = \left(\sum_{1 \le i < j \le n} w_{ij} \left| \left\langle \tilde{V}_{ij} / V_{ij} \right\rangle - g_i \bar{g}_j \right|^p \right)^{1/p} ,$$

where $\langle \tilde{V}_{ij}/V_{ij} \rangle$ is the time-average of the ratio of observed visibility to model visibility, over a time period during which the g_k may be assumed constant.

The AIPS implementation allows the choices $p = 1$ and $p = 2$. Choosing $p = 2$ yields the least-squares solution for g. When one chooses $p = 1$, so that a weighted sum of the moduli of the residuals is minimized, the computed gain solutions are less influenced by wild data points, but there is some loss of statistical efficiency—i.e., the least-squares solutions are superior when the distribution of measurement errors is well-behaved. (Probably the choice $p \simeq 1.2$ would offer a better compromise between efficiency and robustness). See [F. R. Schwab, Robust solution for antenna gains, VLA Scientific Memo. No. 136] for further details.

One may wish to solve only for the *antenna/i.f. phases* $\psi_k(t)$ rather than for the g_k if, for example, atmospheric phase corruption is believed to be the dominant source of systematic error. In this case, one minimizes

$$S(\Psi) = \left(\sum_{1 \le j < k \le n} w_{jk} \left| \tilde{V}_{jk} - e^{i(\psi_j - \psi_k)} V_{jk} \right|^p \right)^{1/p},$$

or the version thereof incorporating time-averages.

Cornwell and Wilkinson [A new method for making maps with unstable radio interferometers, *Mon. Not. R. Astr. Soc.*, **196** (1981) 1067–1086] suggest adding to S terms which arise by assuming prior distributions for the g_k; these "penalty terms" would be chosen so as to increase in magnitude as the solution parameter deviates from a prior mean which one might take, say, as the running mean of previous gain solutions. The widths of the prior distributions could be based on empirical knowledge of the behavior of the array elements. Such a modification can be useful when the array is composed of antenna elements of differing collecting area. This modification is used in order to constrain the moduli of the computed gains in one version of the AIPS task for self-calibration which is used primarily for VLBI data reduction (VSCAL).

Shannon sampling theorem — Suppose the complex-valued function f of the real variable t to be square-integrable, and assume that f is band-limited; i.e., that its Fourier transform $\hat{f}(x) = \int_{-\infty}^{\infty} f(t) e^{2\pi i x t}\, dt \equiv 0$ for $|x| > a$. Then f is completely determined by its values at the discrete set of sampling points $n/2a$, $n = 0, \pm 1, \pm 2, \ldots$, and f can be recovered via the *Shannon series* (also called the cardinal series)

$$f(t) = \sum_{n=-\infty}^{\infty} f\left(\frac{n}{2a}\right) \frac{\sin 2\pi a(t - n/2a)}{2\pi a(t - n/2a)}.$$

The series converges both uniformly and in the mean-square sense.

The Shannon series can be derived by expanding $\hat{f}$ in a Fourier series, and then applying Fourier inversion—or it can be derived from the classical Poisson summation formula. It is sometimes referred to as Whittaker's cardinal interpolation formula or the Whittaker–Shannon sampling series, having first been studied in detail by E. T. Whittaker in 1915 and later introduced into the literature of communications engineering by Shannon in 1949. By the *Paley–Wiener theorem*, since f is band-limited, it can be analytically extended from the real line to the full complex plane, as an entire function of slow growth. The Shannon series, which converges for complex as well as real t, is one means of doing so. Whittaker referred to the series as "a function of royal blood in the family of entire functions, whose distinguished properties separate it from its bourgeois brethren."

Suppose that $f(t)$ is "small" for $|t| > b$ (no nontrivial signal is both band-limited and time-limited). Then, assuming that b is integral, the number of terms in the Shannon series that really matter is $4ab$. This suggests that the space of "essentially band-limited" and "essentially time-limited" signals has dimension equal to the *time-bandwidth product* $4ab$. The precise sense in which this is so, together with a discussion of the *prolate spheroidal wave functions* (*q.v.*), which are relevant to the problem, is described by H. Dym and H. P. McKean [*Fourier Series and Integrals*, Academic Press, New York, 1972] and by David Slepian [Some comments on Fourier analysis, uncertainty and modeling, *SIAM Rev.*, **25** (1983) 379–393].

The multi-dimensional extension of the sampling theorem to rectangles implies that if an "unconfused" radio source $f(x, y)$ is confined to a small region of sky $|x| < x_0$, $|y| < y_0$ (radians), then it can be reconstructed unambiguously from a discrete set of visibility samples $\hat{f}(m\Delta u, n\Delta v)$, $m, n = 0, \pm 1, \pm 2, \ldots$, with $\Delta u = 1/2x_0$ and $\Delta v = 1/2y_0$ wavelengths. See *cellsize* and *Nyquist sampling rate*. Other useful extensions of the sampling theorem—for example, to various multi-dimensional sampling configurations (e.g., 2–D hexagonal sampling lattices), to the case of stochastically jittered sampling, to derivative sampling (e.g., in 1–D, f can be recovered from samples of f and its derivatives through order r taken at intervals $(r+1)\frac{n}{2a}$), etc.—and sampling theorems for functions whose transforms of other than Fourier type are of *compact support*—are described in survey articles by A. J. Jerri [The Shannon sampling theorem—its various extensions and applications: a tutorial review, *Proc. IEEE*, **65** (1977) 1565–1596] and J. R. Higgins [Five short stories about the cardinal series, *Bull. (New Ser.) Amer. Math. Soc.*, **12** (1985) 45–89].

Shannon series — See *Shannon sampling theorem*.

shed — See *sub-task*.

SIVPSF — See *point spread function*.

slice — a one-dimensional cut across an *image*. E.g., the slice of a two-dimensional image f which passes through (x_0, y_0) and has orientation angle ϕ is the *subimage* h given by $h(t) = f(x_0 + t\cos\phi, y_0 + t\sin\phi)$. In AIPS, a slice may be excised from an image by issuing the *verb* command SLICE. Since AIPS deals only with digitized images, the program must interpolate to obtain data along the cut, except when the slice is taken along a row or column of the image.

slice file — in AIPS, an *extension file*, associated with an *image file*, in which a digitized *slice* (*q.v.*), or one-dimensional subimage, of the primary image is stored. In order to display a slice, one may issue the *verb* command SL2PL, which causes AIPS to read the contents of a slice file and generate a *plot file*.

snapshot — in earth-rotation aperture synthesis interferometry, an observation which is of such short duration that Earth's motion does not significantly enhance the *u-v coverage*, or a *map* derived from such a brief observation. Compare *full-synthesis map*.

For a thorough discussion of the use of the VLA in snapshot mode, see §5 of A. H. Bridle's Lecture No. 16 in the *1985 Summer School Proceedings*.

software mount — a computer's reaction to the issuing of a command to it informing it that the *hardware mount* of some external storage module, such as a disk pack or a reel of magnetic tape, has occurred, and that the computer should open the channel of access to this module. See *hardware mount*.

sort order — the ordering of visibility measurements within a *u-v data file*. *Time-baseline order* is convenient for purposes of calibration, *baseline-time order* for data display, and so-called *x-y order* for gridding and subsequent mapping.

SOS — a text editor available on the NRAO's Vaxes, as well as on the PDP-11's and the DEC-10 at the VLA and on Vaxes operating under control of the VMS operating system. SOS (a *line editor*) is provided by the computer manufacturer. *N. B.*: SOS is nearly obsolete—it will not generally be provided on Vax/VMS operating systems with version numbers ≥ 4.0. See *text editor*.

source editor — same as *text editor*. (Formerly, computers were used mainly for numerical computations and text editors primarily for the editing of program source code—hence the name *source editor*).

spatial resolution — In digital image analysis, this term refers rather imprecisely to the minimum size of details which can be discerned. The spatial resolution is determined by three factors: the inherent indeterminacy of whatever *image reconstruction* problem underlies the method by which the image was produced (and the properties of the image reconstruction *algorithm* which produced the image); the measurement noise; and the *pixel* size—i.e., the size of the squares or the rectangles comprising the reconstruction matrix.

In radio interferometry, the inherent spatial resolution goes roughly in inverse proportion to the physical size scale D of the array (measured in wavelengths). For observations at a wavelength λ, the inherent spatial resolution, with a filled aperture, is essentially λ/D radians. However, with a synthesis array with large gaps in the *u-v coverage*, the effective resolution is somewhat coarser. Often, some measure of the spread of the central lobe of the *dirty beam* (say, the FWHM) is quoted as the spatial resolution. However, some reconstruction methods (e.g., the *regularization methods*) produce images in which the resolution of bright features may be much finer than that of dim features. This property of regularization methods may be viewed as either good or bad: S/N dependent spatial resolution complicates the interpretation of an image, but, on the other hand, one may gain additional contrast resolution—i.e., low surface-brightness features may become more readily discernible. An honest statement concerning the spatial resolution of an image must be based upon empirical knowledge of the reconstruction method that was used. See *super-resolution*.

spawn — See *sub-task*.

spheroidal function — an eigenfunction $\psi_{\alpha n}$ of a finite, weighted-kernel Fourier transform—more precisely, for given c and given $\alpha > -1$, one of the countably many solutions of the integral equation

$$(*) \qquad \nu f(\eta) = \int_{-1}^{1} e^{ic\eta t}(1-t^2)^{\alpha} f(t)\,dt\,;$$

equivalently, a solution of the differential equation $(1-\eta^2)f'' - 2(\alpha+1)\eta f' + (b - c^2\eta^2)f = 0$. The eigenfunction $\psi_{\alpha 0}$ of $(*)$ associated with the largest eigenvalue ν is termed the 0-order solution. The choice $\alpha = 0$ of weighting exponent yields the family $\{\psi_{0n} \mid n = 0, 1, 2, \ldots\}$ of prolate spheroidal wave functions.

Weighted 0-order spheroidal functions $(1-\eta^2)^{\alpha}\psi_{\alpha 0}$ are optimal gridding convolution functions in the same sense that the *prolate spheroidal wave functions* (*q.v.*) are optimal, except that now the weighted concentration ratio

$$\frac{\iint_{\text{map}} |\hat{C}(x,y)|^2 (1-(2x\Delta u)^2)^{\alpha}(1-(2y\Delta v)^2)^{\alpha}\,dx\,dy}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} |\hat{C}(x,y)|^2 |1-(2x\Delta u)^2|^{\alpha}|1-(2y\Delta v)^2|^{\alpha}\,dx\,dy}$$

is maximized (see the paper by F. R. Schwab in the *1983 Sydney Conference Proceedings*). The weighting exponent α is used to trade off the effectiveness of the aliasing suppression at the edge of the field of view, against that in the central region of the map. The choice $\alpha = 1$, with a *support width* of six *u-v* grid cells, yields an effective gridding convolution function, emphasizing aliasing suppression in the central region of the map; this function, ψ_{10}, with $c = 3\pi$, is the default function used in the AIPS mapping program. See *gridding convolution function*.

Stokes' parameters — the four coordinates relative to a particular basis for the representation of the polarization state of an electromagnetic wave propagating through space. Consider a wave propagating along the z-direction in a right-handed (x, y, z) Cartesian coordinate system. At a fixed point in space, let the instantaneous components of the electric field vector, in the x- and y-directions, be denoted by $E_x(t)$ and $E_y(t)$, respectively; and assume them to be stationary (in the weak sense, and square-integrable) stochastic processes. Form the matrix

$$S = \begin{pmatrix} \langle E_x(t)\overline{E}_x(t+\tau)\rangle\hat{} & \langle E_x(t)\overline{E}_y(t+\tau)\rangle\hat{} \\ \langle E_y(t)\overline{E}_x(t+\tau)\rangle\hat{} & \langle E_y(t)\overline{E}_y(t+\tau)\rangle\hat{} \end{pmatrix}.$$

Here, the bracketed expressions are expectation values, or correlation functions, in the lag variable τ, and $\hat{}$ denotes Fourier transform with respect to τ. Thus each element of S is a function of frequency ν. S is Hermitian (conjugate symmetric), owing to the stochasticity assumptions. The three Pauli spin matrices, together with the 2×2 identity matrix, form a basis for the algebra of 2×2 Hermitian matrices; i.e., each such matrix S can be represented in the form

$$S(\nu) = \sigma_1(\nu)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \sigma_2(\nu)\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + \sigma_3(\nu)\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + \sigma_4(\nu)\begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}.$$

The four (real) coefficients, $\sigma_1, \ldots, \sigma_4$, of the representation of S in this basis are called Stokes' parameters. They commonly are denoted by $I(\nu)$, $Q(\nu)$, $U(\nu)$, and $V(\nu)$, respectively. In other words,

$$S(\nu) = \begin{pmatrix} I(\nu)+Q(\nu) & U(\nu)+iV(\nu) \\ U(\nu)-iV(\nu) & I(\nu)-Q(\nu) \end{pmatrix},$$

with I, Q, U, and V real.

Stokes' parameter I measures the total intensity of the radiation field, Q and U the linearly polarized intensity, and V the circularly polarized intensity. I always is nonnegative. For a totally unpolarized wave, $Q = U = V = 0$; for a partially polarized wave, the ratio $\sqrt{Q^2+U^2+V^2}/I$ measures the total degree of polarization, $\sqrt{Q^2+U^2}/I$ the degree of linear polarization, and $\frac{1}{2}\arctan\frac{U}{Q}$ the orientation angle of the linearly polarized component. $Q + iU$ is called the complex linear polarization. The IAU and IEEE orientation/sign conventions have the z-axis directed toward the observer, the x-axis directed north, and a $+i$ in the argument of the exponential kernel of the FT. Positive V corresponds to right

circular polarization, and conversely. The polarization response of an interferometer can be described by forming the so-called cross-spectral density matrix, which is like the S above but is formed from measurements of the electric field taken at two points in space. For further details, including a description of the polarization response of an interferometer, for various feed configurations, see Carl Bignell's Lecture No. 6 in the *1982 Summer Workshop Proceedings*.

Stokes' visibility functions — Stokes' visibility functions, V_I, V_Q, V_U, and V_V, are the Fourier transforms (FT's) of the radio brightness (spatial) distributions of *Stokes' parameters*, $I(x,y)$, $Q(x,y)$, $U(x,y)$, and $V(x,y)$. (Here, $V_I = \hat{I}$, $V_Q = \hat{Q}$, etc., where $\hat{}$ denotes FT.)

For a radio interferometer with ideal circularly polarized feeds, the relations between Stokes' visibility functions and the visibilities, V_{RR}, V_{LL}, V_{RL}, and V_{LR}, obtained by correlating right circular response with right, left with left, etc., are $V_I = \frac{1}{2}(V_{RR} + V_{LL})$, $V_Q = \frac{1}{2}(V_{LR} + V_{RL})$, $V_U = \frac{i}{2}(V_{LR} - V_{RL})$, $V_V = \frac{1}{2}(V_{RR} - V_{LL})$. Note that each of Stokes' visibility functions is *Hermitian*. On the assumption that circular polarization is absent (i.e., that $V(x,y) \equiv 0$), V_{RR} is equal to V_{LL}, and both are Hermitian.

Components of the systematic errors affecting visibility measurements are i.f.-dependent; hence VLA *u-v* data files usually do not contain Stokes' visibilities, but rather V_{RR}, V_{LL}, V_{RL}, and V_{LR}—as these are what is required for calibration purposes. Stokes' visibility functions generally are constructed only within the mapping programs. (But the AIPS visibility data format is designed to accommodate either type of visibility function, and the mapmaking tasks are able to recognize the form of their input data and deal with them appropriately.)

subimage — in AIPS parlance, any linear, rectangular, or hyper-rectangular section of an *image*.

sub-task — a task, or computer program, whose execution is initiated by the action of another program. The act of initiating the execution of the sub-task is called *task shedding* or *task spawning*. See *task*.

super-resolution — The problem of image reconstruction in radio interferometry is one of finding an approximation to an unknown function f (generally assumed to be of *compact support*) from *partial knowledge* of its Fourier transform $\hat{f}$ — i.e., from a finite number of measurements of the visibility. Any of the techniques which are applied to the problem—the *Högbom CLEAN algorithm*, the *regularization method*, etc.—may be thought of as methods of smoothing, interpolating, and extrapolating the noisy measurements. *Super-resolution* is a term which refers to the extrapolation aspect: Cautious extrapolation yields an image whose *spatial resolution* is $\approx \lambda/D$, where D is the diameter of the largest centered region in the *u-v* plane which has been reasonably well sampled. Less cautious extrapolation yields super-resolution; spurious detail appears as caution is abandoned.

Super-resolution in a *clean map* is effected by choosing an artificially narrow *clean beam*. With regularization methods (in image reconstruction, and more generally), super-resolution comes about by choosing a small value of the *regularization parameter*. The spatial resolution achieved by a regularization method may be signal-to-noise dependent—bright features may be super-resolved, and dim ones not.

support — The closure of that subset of the domain of definition of a function f (or of a generalized function, or distribution) on which the function assumes a nonzero value is called the *support* of the function, and is denoted by $\mathrm{supp}(f)$. I.e., $\mathrm{supp}(f) = \overline{\{x \mid f(x) \neq 0\}}$.

For example, the support of the function $f(x) = x$ is the whole real line, even though $f(0) = 0$. And the support of

$$f(x,y) = \begin{cases} 1, & x^2 + y^2 < 1, \\ 0, & \text{otherwise}, \end{cases}$$

is the *closed* unit disk, $\{(x,y) \mid x^2 + y^2 \leq 1\}$.

In Euclidean space, a function f whose support is bounded—i.e., such that $f \equiv 0$ "far-out"—is said to be of *compact support*. The Fourier transform of a nontrivial function of compact support (such as a *u-v measurement distribution* or a *gridding convolution function*) cannot itself be of compact support; i.e., it has "sidelobes" extending to infinity.

support width — of a function whose *support* is a rectangle or a hyper-rectangle (e.g., the Fourier transform of a band-limited function), the linear measure of one of the edges of its *support*.

SVPSF — See *point spread function*.

synthesised beam — in radio interferometry, the *beam*—but always ignoring instrumental effects. Hence, the synthesized beam is fully determined by the *u-v sampling distribution*, the *u-v weight function*, the *u-v taper function*, and the *gridding convolution function*. See *beam*.

tape blocking efficiency — Data are stored on magnetic tape in units of *blocks*. An *inter-record gap*—essentially wasted space—separates one block from the next. The *tape blocking efficiency*, or the fraction of unwasted space, is the ratio

$$\frac{\frac{\text{block length}}{\text{recording density}}}{\frac{\text{block length}}{\text{recording density}} + \text{length of an inter-record gap}}.$$

The length of an inter-record gap is about $\frac{3}{4}$, $\frac{3}{5}$, and $\frac{3}{10}$ inch at recording densities of 800, 1600, and 6250 bpi, respectively.

taper — See *u-v taper function*.

task — used in two senses: 1) the execution of a computer program and 2) the program itself. Thus, if two computer users are (independently) running the same program at the same time, it may be said either that two tasks are running, or that two incarnations of the same task are in existence. A *sub-task* (*q.v.*) is a task whose execution is initiated by the action of another program. Many of the more complicated and the more specialized functions of AIPS are accomplished by the action of sub-tasks shed by the AIPS program. (Simpler functions are invoked by the issuance of *verb* commands—see *POPS symbols*.)

t-b order — See *time-baseline order*.

TEK screen — a cathode ray tube (CRT) terminal and display device appropriate for pictorial display of data, in the form of contour plots, graphs, etc., as well as for display of textual data. The Tektronix company's Model 4012 terminal (with a green P4 phosphor, hence the synonymous term *green screen*) is the canonical device of this type. The "make copy" button on this device can be used to produce a copy, on paper, of the image shown on the CRT screen. Each of the NRAO's AIPS data reduction computers is outfitted with a *TEK screen*.

TEK4012 — same as *TEK screen*.

Telex 6250 tape drive — a model of tape drive used on the VLA Vaxes, capable of operation at 1600 and 6250 bpi.

terminal page — Many modern computer terminals contain a semiconductor memory with a capacity of several CRT screen loads ($\approx$ 24 lines) of character data. A *terminal page* is a unit of one screen load of such data. Certain terminal keys

allow one to cause data which previously appeared on the CRT screen to reappear—this feature is called *terminal scroll* (*q.v.*). A typical terminal at the NRAO has three terminal pages of memory.

terminal scroll — that feature present on certain models of computer terminals which allows data which previously appeared on the CRT screen to be made to reappear. Often, depressing one key on the terminal will cause earlier information to reappear line-by-line (this is termed *line scroll*), while the action of another key will cause a whole earlier screen load to reappear (this is termed *page scroll*).

text editor — a computer program designed for the creation, manipulation, and modification of computer files containing textual data such as reports, documentation, alphanumeric command lines, and program source code. Generally, one or more text editors are supplied by the computer manufacturer. Three text editors are in widespread use on the Vax—*SOS*, *EMACS*, and *EDT*—, and *SEDIT* is used on Modcomp computers. See *line editor* and *screen editor*.

text file — a computer data file containing only textual data, as might be written by a *text editor* (*q.v.*). Programs such as the AIPS tasks sometimes write messages, especially progress report messages, into a text file—see *message file*.

thrashing — See *memory thrashing*.

time-baseline order — An ordered set of visibility measurements $\left\{ V_{ij}(t_k) \;\middle|\; 1 \le i < j \le n,\ k = 1, \ldots, l \right\}$ recorded with an n element interferometer at times $t_1 < t_2 < \cdots < t_l$ is said to be in *time-baseline order* if the ordering is such that all of the data obtained at time t_1, sorted into the canonical ordering by baseline, occur first, followed by the data obtained at time t_2, again ordered canonically, etc., etc. (The canonical ordering by baseline is the order $V_{12}, V_{13}, \ldots, V_{1n}, V_{23}, \ldots, \ldots, V_{n-1,n}$.) Compare *baseline-time order*.

Time-baseline ordering of a *u-v data file* is convenient for calibration purposes. The AIPS task for self-calibration requires that its input *u-v* data file be time-baseline ordered.

time smearing — in a radio interferometer map, the space-variant broadening of the *point spread function* (or *beam*) which is due to time averaging of the data. When, for example, the visibility data along a *u-v* track are averaged, with equal weight, over time intervals of width Δt sec., the visibility amplitude of a point source is reduced by a factor $\approx \frac{\sin\gamma}{\gamma}$ —where $\gamma \equiv \pi(u'x + v'y + w'z)\Delta t$, where the primes denote the time rate of change of the spatial frequency coordinates (u, v, w) along the track (wavelengths/sec.), and where (x, y, z) denotes the direction cosines of the location of the point source with respect to the *phase tracking center*. For further details, see A. R. Thompson's Lecture No. 2 and A. H. Bridle's Lecture No. 16 in the *1985 Summer School Proceedings*. Compare *bandwidth smearing*.

trackball — a spherical ball mechanism, about the size (10 cm., or so, in diameter) of a tennis ball, which may be oriented manually by the interactive user of a television display device such as the I^{2}S. The ball can be rotated about any axis, and its orientation, which is sensed by the computer, typically is used to control the enhancement or the coloration of the displayed data (i.e., to control the TV *transfer function(s)*), or to position the *TV cursor*, in order to point out to a program features in the displayed image which are of particular interest.

trackball button — On the unit which houses the trackball for the I^{2}S Model 70 TV display device are the four *trackball buttons*, labeled A, B, C, and D. These are switches that are used, in conjunction with the display routines, to exert additional control over the TV display. Occasionally these buttons are put to other use in AIPS, such as stopping the CLEAN deconvolution program.

transfer function — a transform which can be used to describe the output of a device (say, an electrical transducer) as a function of the input to the device. See *TV look-up table*.

TRC — *top right corner*, the corner of an image diagonally opposite the BLC. See *m* × *n map*.

true color display — a type of *false color display*, (*q.v.*).

TU77 tape drive — a model of tape drive used on the NRAO's Vaxes, capable of operation at 800 and 1600 bpi.

TU78 tape drive — a model of tape drive used on the VLA Vaxes, capable of operation at 1600 and 6250 bpi.

TV blink — a feature of a computer-controlled TV display device, such as the *I^{2}S*, intended to facilitate the comparison of a pair of images stored on two different *image planes*. The TV display is made to alternate between the two images. The AIPS implementation of blinking allows the user, by manipulating the *trackball*, to control the rate of alternation and the fraction of time that each image is displayed.

TV cursor — See *crosshair*.

TV image catalog — See *image catalog*.

TV look-up table — a memory within the control unit of a TV display device which is used for storage of the *transfer functions* controlling the intensity of the display, as a function of pixel value. Within AIPS, the transfer functions may be altered through the use of interactive verbs and manipulation of the *trackball*.

TV roam — a feature of a computer-controlled TV display device such as the *I^{2}S* which allows contiguous parts of a single large image, stored on more than one *image plane*, to be displayed as if the image were stored on a single, larger image plane. On the I^{2}S unit, the portion of the image to be displayed on the TV screen is selected by manipulation of the *trackball*. See *image plane*.

TV scroll — a feature of a computer-controlled TV display device such as the *I^{2}S* which allows the display of an image stored on a single *image plane* to be moved about the display screen. This feature, which also is called panning, commonly is used in combination with the *TV zoom* capability. On the I^{2}S unit, the scroll ordinarily is controlled by manipulation of the *trackball*. Compare *TV roam*.

TV zoom — a magnification feature of a computer-controlled TV display device such as the *I^{2}S*. On the I^{2}S, the three available magnification factors (which multiply the linear dimensions of the original display of the image by a factor of 2, 4, or 8) generally are selected by depressing one of the *trackball buttons*. Since the magnification is achieved by pixel replication (i.e., by piecewise linear interpolation)—rather than by a smooth interpolation—the visual impression may be somewhat displeasing. The entire magnified image may not fit on the TV screen, so zoom usually is used in combination with the *TV scroll* feature.

uniform weighting — A *dirty map* obtained by computing the inverse Fourier transform (FT) of a weighted *u-v measurement distribution* in which each visibility sample has been weighted in inverse proportion to the local density of the *u-v coverage* is said to have been computed using *uniform weighting*. When a radio map is computed via the fast Fourier transform algorithm, uniform weighting may be achieved by computing normalised discrete convolution summations $\sum_{i=1}^{N} C(u - u_i, v - v_i)\tilde{V}_i/N$, where (u, v) denotes the spatial

frequency coordinates of a given u-v grid cell, where C is an appropriately chosen *gridding convolution function*, and where the $\tilde{V}_i$ are the N visibility measurements obtained at positions (u_i, v_i) in some neighborhood of (u, v), the size of which is determined by the *support* of C. The uniform weighted map is given by the inverse discrete FT of data interpolated and smoothed in this manner, onto the lattice points of a rectangular grid. So-called *natural weighting* is achieved by using unnormalized convolution sums, rather than by dividing by N. The AIPS mapmaking tasks use a weighting scheme which is slightly more complicated than that described here.

Since the density of u-v coverage typically is greater in the inner regions of the u-v plane, a map computed using uniform weighting has finer *spatial resolution* than one computed with natural weighting. With natural weighting, low surface-brightness extended features may be more easily discernible than with uniform weighting. Essentially the same effect can be achieved with uniform weighting, when accompanied by use of a *u-v taper function*.

UNIX — a "universal" computer operating system developed at the Bell Telephone Laboratories. Its virtue is that program packages such as AIPS—once having been made to run under one UNIX-based operating system—ought to run on any other such system, even on a computer of different manufacture, with no alterations. Many Vaxes operate under UNIX, though not the NRAO's. However, UNIX emulators are run as secondary operating systems on the IBM-4341 and (experimentally) on the VAX-11/780 in Charlottesville. See *operating system*.

user-coded task — an AIPS *task* written by a user, rather than by a professional programmer or a member of the AIPS programming group. One of the design goals for AIPS, not yet fully realized, is that it should be relatively easy for a user who is not an experienced programmer to write an AIPS task suited to his own needs—i.e., that it should be fairly simple for him to make some sense of the AIPS database, and to get at his data and manipulate it as he sees fit. The AIPS task named FUDGE is intended to serve as a paradigm for user-coded tasks for manipulation of *u-v data files*; two other tasks, TAFFY and CANDY, are paradigms for *image file* manipulation. A useful reference is the manual by W. D. Cotton and a 'cast of AIPS' [*Going AIPS! A Programmers Guide to the NRAO Astronomical Image Processing System*, NRAO, Green Bank, WV, 1984].

The addition to AIPS of new *verbs*, and modification of the functioning of existing verbs, requires modifying the AIPS program itself; this is best left to the AIPS programming group.

u-v coverage — the *support* of the *u-v sampling distribution* (*q.v.*). Also see *conjugate symmetry*.

u-v data file — in AIPS, a *primary data file* designed to accommodate the measurements of the visibility function of a radio source.

u-v data flag — In an AIPS *u-v data file*, each visibility measurement is accompanied by a real-valued weight, which ordinarily is (positive and) proportional to the length of the integration period over which the measurement was obtained. A non-positive weight represents a u-v data flag, which signifies that the visibility measurement ought to be ignored. See *flagging* and *clipping*.

u-v FITS format — an extension of the *FITS format* (originally designed for the interchange of image data) to accommodate radio interferometer visibility data [E. W. Greisen and R. H. Harten, An extension of FITS for groups of small arrays of data, *Astron. Astrophys. Suppl. Ser.*, **44** (1981) 371–374]. See *FITS format*.

u-v measurement distribution — in radio interferometry, a linear combination of shifted Dirac δ-functions, one located at the position in the u-v plane of each visibility measurement, and each weighted by the visibility measurement obtained at that location. Denoting the u-v coverage by $\{(u_i, v_i)\}_{i=1}^n$, the visibility function by V, and the measured visibility by $\tilde{V}$, the (two-dimensional) u-v measurement distribution S is given by $S(u,v) = \sum_{i=1}^n \tilde{V}(u_i, v_i)\delta(u - u_i, v - v_i)$. Compare *u-v sampling distribution*.

This definition may be modified to incorporate two types of weight function, yielding a *weighted* and/or *tapered* measurement distribution—see *u-v taper function* and *u-v weight function*.

The visibility measurements $\{\tilde{V}(u_i, v_i)\}$ are not actual samples of V, but rather are error-corrupted samples of a function which represents some sort of *local average* of the visibility—this is a distinction which it is worthwhile to note, and then to ignore. Various systematic errors affecting the measurements may be corrected by proper calibration—see *antenna/i.f. gain* and *instrumental polarization*.

u-v sampling distribution — in radio interferometry, a linear combination of shifted Dirac δ-functions, one located at the position in the u-v plane of each visibility measurement. Sometimes termed *u-v transfer function*. See *beam*.

If $\{(u_i, v_i)\}_{i=1}^n$ (the *u-v coverage*) is the set of spatial frequency coordinates at which the source visibility has been sampled, then the (two-dimensional) u-v sampling distribution S is given by $S(u,v) = \sum_{i=1}^n \delta(u - u_i, v - v_i)$.

Occasionally the term *u-v sampling distribution* is used in the same sense as the term *u-v measurement distribution* (*q.v.*).

u-v taper function — an even, real-valued weight function (typically, an elliptical Gaussian), smooth and peaked at the origin, which may be incorporated into the definition of *u-v measurement distribution* or *u-v sampling distribution*, above, serving to control the spatial resolution of the radio map or the beam; i.e., to enhance the response to extended features in the radio source brightness distribution by giving relatively higher weight to the measurements at short u-v spacings. Compare *u-v weight function*.

u-v transfer function — same as *u-v sampling distribution*, but always explicitly incorporating any u-v weight function or u-v taper function.

u-v weight function — a real-valued function which may be incorporated in the definition, above, of *u-v measurement distribution* or *u-v sampling distribution*, serving to weight each measurement either according to an estimate of the statistical measurement error, or according to the local density of sampling, or both. Compare *u-v taper function* and see *uniform weighting*.

Varian printer — an electrostatic printer/plotter manufactured by the Varian Corp., and attached to the Modcomp Classic computer used to run AIPS in Charlottesville.

Variational Method — the name which applies to Tim Cornwell's AIPS implementation (in the program VM) of the *maximum entropy method*, to solve the image deconvolution problem $g = b * f$, where g and b are given, and f is unknown. The regularizing term $S(\tilde{f})$ (see *regularization method*), a function of the computed approximate solution $\tilde{f}$, is given by the negative of an entropy expression, of the form

$$H(\tilde{f}) = -\int_A \tilde{f}(x) \log \frac{\tilde{f}(x)}{h(x)}\, dx.$$

Here A denotes the (assumed known) support of f, and h is a prior estimate of f; when $h \equiv$ constant, this agrees with the standard formulation of the maximum entropy method. A weighted sum $\chi^2(\tilde{f}) + \lambda S(\tilde{f})$ of a χ^2 error term and S is minimized, and the regularization parameter λ is chosen so that the r.m.s. residual corresponding to the final iterate is approximately equal to an input value. For optical data the χ^2 term is taken as $\|g - b * \tilde{f}\|^2$, whereas for radio data the χ^2 term is evaluated in the visibility domain, where the measurement errors may more properly be assumed to be statistically independent. Also, $\int_A \tilde{f}$ is constrained to be near an estimate of the *zero-spacing flux* which is supplied by the user. The minimization is done using a Newton-type method, with a diagonal approximation to the Hessian of the objective function and intricate control of the steplength. In terms of execution speed, this method is competitive with the *Clark CLEAN algorithm*—at least in the case of large objects of complex structure observed with the VLA—and superior results usually are obtained for this class of objects. See [T. J. Cornwell, Deconvolution with a maximum entropy type algorithm, VLA Scientific Memo. No. 149].

verb — See *POPS symbols.*

Versatec printer — an electrostatic printer/plotter manufactured by the Versatec Corp., and used on the NRAO's AIPS computer systems.

virtual memory page — on a computer running under a virtual memory operating system, one unit of *virtual memory storage*. At a typical Vax installation, the size of a virtual memory page is 512 bytes.

virtual memory page swapping — on a computer running under a virtual memory operating system, the action (initiated automatically by the operating system) of reading new virtual memory pages into the physical memory, and storing on disk (i.e., in the *virtual memory*) the data which thus have been displaced. Each occurrence of the displacement of a memory page is referred to as a *page fault*. See *memory thrashing.*

virtual memory storage — computer storage—typically disk storage—in an area apart from the *physical memory* of a computer. Access to virtual memory storage is controlled by the operating system, in a way intended to give the programmer the illusion that a large amount of physical memory is present. Access to virtual memory may be much slower than access to physical memory, and the operating system may incur a significant amount of overhead in managing the virtual memory. See *memory thrashing.*

visibility phase tracking center — In a correlating-type radio interferometer usually the *fringe stopping center* and the *delay tracking center* coincide. When this is the case, both are referred to as the visibility phase tracking center.

VM — See *Variational Method.*

VMS — (Virtual Memory System) the operating system used on the NRAO's Vax computers. See *virtual memory storage* and *operating system.*

wedge — a legend, or scale—generally in the form of a bar graph with gradations in *intensity* and *chromaticity*—which may be displayed adjacent to a photographic or video display of a digitized *image*. The wedge is a visual representation of the *transfer function* that was used in generating the display. The wedge is either colored or gray, depending on whether the display is a *pseudo-color display* or a *gray-scale display.*

Note that a *false color display* would require more than one wedge (or a multi-tiered wedge) to display the several transfer functions, as well as an additional wedge to display the possible color mixtures.

window clean — an application of the *Högbom CLEAN algorithm*, with an explicit specification, by the user, of the clean window. Generally the user should specify a clean window whenever it is possible to make a reasonably valid and restrictive estimate of the *support* of the true radio source brightness distribution. At the termination of the algorithm, it is prudent to examine a display of the residual map for the presence of large residuals outside of the clean window; their presence could suggest that an inappropriate window was selected. See *clean window.*

working set size — on a computer running under a virtual memory operating system, the amount of *physical memory* allocated to a task. Any program memory requirement in excess of the working set size is relegated to *virtual memory storage*. At a typical Vax installation, the working set size is set at $\frac{1}{4}$ or $\frac{1}{2}$ megabyte.

x-y order — An ordered set of visibility samples $\{V(u_i, v_i, w_i)\}_{i=1}^n$ arranged according to descending absolute value of the spatial frequency coordinate u — i.e., with $|u_1| \geq |u_2| \geq \cdots \geq |u_n|$ — is said to be in *x-y order*.

x-y order is a convenient ordering for the operation of gridding convolution; hence the AIPS mapping tasks require that their input *u-v data files* be sorted accordingly. See *sort order.*

Y-routine — in AIPS, a subroutine designed to aid in the use of a specific model of TV display device, such as the *I²S* Model 70. AIPS requires a relatively small core of Y-routines implementing basic TV display functions; complicated display functions then are accomplished by combining these basic functions that are supposed to be common to many models of TV display device. At present there are approximately 25 Y-routines for use at those AIPS installations equipped with an I²S. Compare *Z-routine.*

zero-spacing flux — The visibility $V(u,v) \equiv \hat{f}(u,v)$ ($\hat{}$ denotes Fourier transform) of a source brightness distribution f in a neighborhood of $u = v = 0$ is inaccessible to an interferometer composed of elements of finite collecting area. The *zero-spacing flux* is equal to the total, or integrated flux density of the source—i.e., it is given by $V(0,0) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\, dx\, dy$. Because the hole in the *u-v coverage* in the neighborhood of the origin may be fairly large, image reconstruction methods, such as the *Högbom CLEAN algorithm*, may do a poor job, within this central region, of interpolating the measured data. This frequently is manifested by the appearance of a *negative bowl artifact*—a negative 'baseline' beneath the reconstruction of f—owing to the reconstruction method having underestimated the zero-spacing flux. The *Variational Method* for maximum entropy reconstruction requires that the user supply an estimate of $V(0,0)$. The CLEAN algorithm, too, may benefit if a datum at $u = v = 0$ is included when the *dirty map* is constructed.

A zero-spacing estimate can be derived from single-dish measurements. Providing a proper estimate is difficult, because of contamination of single-dish measurements by 'confusing sources.' The estimate ought to correspond to a telescope with the same primary beam response as the array elements; and it is not just a single datum $V(0,0)$ which is missing, but rather a region—so proper weighting of the zero-spacing information is tricky. See B. G. Clark's Lecture No. 10 in the *1982 Summer Workshop Proceedings.*

zoom — See *TV zoom.*

Z-routine — in AIPS, a subroutine—generally designed to perform some routine, often needed function—written for

a specific model of *host computer* or for a specific host computer operating system. The implementation of certain basic functions, especially those for file access and file management, generally is machine dependent and operating system dependent. The typical AIPS installation requires 50–100 Z-routines. Compare *Y-routine*.

1978 Groningen Conference Proceedings — *Image Formation from Coherence Functions in Astronomy. Proceedings of IAU Colloquium No. 49 held at Groningen, the Netherlands, August 10–12, 1978*, edited by C. van Schooneveld, D. Reidel, Dordrecht, Holland, 1979—contains many papers on aperture synthesis techniques, including some of the early papers on *hybrid mapping*.

1982 Summer Workshop Proceedings — *Synthesis Mapping. Proceedings of the NRAO–VLA Workshop held at Socorro, New Mexico, June 21–25, 1982*, edited by A. R. Thompson and L. R. D'Addario, NRAO, Green Bank, WV, 1982—a collection of the fifteen lectures which comprised this short course on aperture synthesis techniques—a useful introduction to VLA data reduction methods.

1983 Sydney Conference Proceedings — *Indirect Imaging: Measurement and Processing for Indirect Imaging. Proceedings of an International Symposium held in Sydney, Australia, August 30–September 2, 1983*, edited by J. A. Roberts, Cambridge Univ. Press, Cambridge, 1984—contains a number of interesting papers on aperture synthesis techniques.

1985 Summer School Proceedings — lecture notes from the second NRAO summer short course on radiointerferometric imaging (in preparation). This volume supersedes the *1982 Summer Workshop Proceedings*.

4012 — See *TEK screen*.

Appendix Z. SYSTEM-DEPENDENT *AIPS* TIPS

Although *AIPS* attempts to be system independent, some aspects of its use inevitably depend on the specific site. These vary from procedural matters (*e.g.*, location of sign-up sheets, terminals, and tape drives) to the hardware (*e.g.*, names and numbers of tape and disk drives, the parameters of television and array processor devices) to the peculiar (*e.g.*, the response of the computer to specific keys on the terminal, the presence of useful job control procedures). This appendix contains information specific to NRAO's individual *AIPS* installations. It is intended that non-NRAO installations replace this appendix with one describing their own procedures, perhaps using this version as a template.

Z.1. VLA Site VAXes

The VLA *AIPS* computers are two VAX 11/780s running the VMS operating system, each equipped with an I^2S display and a Floating-Point Systems AP-120B array processor. Each has three (logical) disks and two or three tape drives. The *AIPS* Caige (VAX room) is located at the East end of the Library-Office Building. There are two *AIPS* terminals for each computer. To the left as you enter the door are those for the older VAX known as VAX1 (or as AIPS:: to Decnet). To the right are the terminals for VAX3. Both I^2S devices are equipped with four grey-scale memory planes and four graphics overlay planes. A Tektronix 4012 is available for TK graphics outputs on each. A Versatec is used as a line printer and for plotting and hard copy of the TK screens on each. A QMS Lasergrafix printer-plotter is connected to VAX3, but is available from both VAXes via the task QMSPL. This printer produces plots of much higher quality than those produced on the Versatec.

Z.1.1. Signing up for *AIPS* time at the VLA

There is a sign-up sheet for *AIPS* on the notice board just to the left of the entrance to the *AIPS* Caige in the Library-Office Building. As *AIPS* is used by a majority of VLA observers, it is usually essential to sign up for your *AIPS* time in advance. To promote fair and efficient use of the system, there are strong restrictions on the amounts of time that any one user or user group may reserve — see the *AIPS* Caige notice board for details.

AIPS supports up to four simultaneous users on each computer at the VLA. The user signed up for AIPS1 has priority for the use of the displays, array processor, and tape drives. In many cases, however, the user signed up for AIPS2 may carry out similar processing tasks by verbal negotiation with the AIPS1 user over the use of these resources. The facilities available through AIPS3 and higher are very limited and most users will require either AIPS1 or AIPS2.

Z.1.2. Using the terminals at the VLA

As you attempt to correct mistyped characters, you will note that CTRL H does not delete on VAX/VMS terminals as it does on DEC-10 terminals. Use the RUB OUT key at the upper right of the keyboard instead of CTRL H.

During execution, paging can be halted by typing CTRL S and resumed by typing CTRL Q. To abort any execution, type CTRL Y or CTRL C. Using CTRL Y while in AIPS will unceremoniously eject the user to monitor level (prompt $). If CTRL Y was entered accidentally, AIPS may be resumed by typing CONT $_{R}$. If the abort was intentional, then the user will have to reinitialise AIPS with all the input parameters having been lost.

The Tektronix 4025 terminals have three pages of memory. To scroll upwards press the 1 key on the numeric pad to the right of the main keyboard. The 2 key on the numeric pad will scroll downwards.

Z.1.3. Logging in at the VLA

Go to the *AIPS* Caige at the East end of the Library-Office Building and find the terminal connected to the VAX for which you have signed up. Each terminal in the *AIPS* Caige is labeled to tell you which computer and which *AIPS* number it is for. Typing CR_R on the terminal will produce a prompt which will tell you the current state of that terminal. Other terminals at the VLA may be used to communicate with the VLA VAXes via the VLA Digital Data Switch. On these terminals, therefore, you may get the response:

```
WELCOME TO THE VLA DIGITAL DATA SWITCH
DDS PORT mmm
```

and a prompt for the desired computer. Enter:

```
VLA DDS:  AIPS  CR
```

to connect to VAX1 or

```
VLA DDS:  VAX3  CR
```

to connect to VAX3. You should then see the message:

```
CONNECTING PORT nnn (xxxx BAUD) TO PORT nnn (yyyy BAUD)
```

Hit CR_R until you get a **Username** prompt.

If the prompt symbol is > the terminal is already being used for AIPS. Check with other possible users before proceeding. If it's okay to use the system, type:

```
EXIT  CR
```

If the prompt is **$** (due to the initial CR_R or to following the step above), the computer is at system level, but not in use. Check nearby users to make sure and then type:

```
SHOW PROC  CR
```

to establish the current owner of the terminal. If it is not AIPS, type:

```
LO  CR
 CR
```

to log out the previous user and to begin the log in process. If the current owner of the terminal is AIPS, type one of:

```
AIPS OLD  CR
AIPS NEW  CR
AIPS TST  CR
```

The initial CR_R or the ones given after logging out or completing the DDS connection should give a prompt of **Username:**, which means that you can log in. Proceed as follows:

```
Username:   AIPS  CR
```

The message **Welcome to VAX/VMS Version ...** should appear, followed by some system messages. The login procedure should then produce the statement

```
To start up AIPS type AIPS OLD or AIPS NEW or AIPS TST
```

will appear. Type in your choice. The OLD version is likely to be relatively free of bugs, but the NEW version will contain the recent, tested improvements. The TST version is under active development and should be used by NRAO staff only. (Note that this choice affects only the version of the AIPS program itself. You may choose TST, NEW or OLD versions of the *AIPS* reduction programs at a later time [see §4.5].)

You should then see the messages:

```
Starting 15OCT85 version of AIPS
BEGIN THE ONE TRUE AIPS NUMBER n
```

where 15OCT85 identifies the release of AIPS you have selected and *n* is a number between 1 and 4. Check that this "*AIPS* number" is indeed the one for which you signed up. If not, you are using the wrong terminal: AIPS1 and AIPS2 have specific terminals, labeled as such, while all other *AIPS* numbers are available on a first-come, first-served basis. AIPS will now ask you for your VLA user number:

```
AIPS n:  ENTER USER ID NUMBER
? uuu ⏎
```

where *uuu* is your VLA user number. The AIPS prompt > should now appear.

Z.1.4. Hardware tape mount at the VLA

Go into the "Inner Sanctum" of the VAX room containing the VAXes themselves. Three drives are available on VAX1 and two on VAX3:

VAX1 tape 1	DEC TU78	1600 and 6250 bpi
VAX1 tape 2	DEC TU78	1600 and 6250 bpi
VAX1 tape 3	Telex	1600 and 6250 bpi
VAX3 tape 1	DEC TU77	800 and 1600 bpi
VAX3 tape 2	Telex	1600 and 6250 bpi

FITS and EXPORT tapes written on the DEC-10 are normally at 1600 bpi density, although a 6250-bpi drive may be installed by the time you read this *COOKBOOK*. FITS tapes written by the Pipeline system may be at either 1600 or 6250 bpi density.

Z.1.4.1. Mounting tapes on TU77s and TU78s

Open the door by pulling from the right hand edge. The upper spool is ready to receive or release a tape reel if a red band shows inside the spool hub. Mount your tape on this upper spool so that the protrusions in its protective rim engage the slots to the right hand side of the tape enclosure. Press the central lock in the hub until it snaps into position with no red band showing. Shut the door and press the white LOAD button at the left of the button row between the spools. The tape should thread itself and advance to the first file after which the BOT signal should light up. Press the ON LINE button — the ON LINE signal should light up. Do not open the door again until you have finished with the tape and it is completely unloaded.

If this mounting sequence fails repeatedly, consult a VAX advisor (see the list posted on the noticeboard beside the entrance to the main VAX room).

Z.1.4.2. Mounting tapes on Telex 6250s

Open the door by pulling from the left hand edge. Mount the tape on the upper spool so that the protrusions on its protective rim engage the slots on the upper side of the tape enclosure. There is no manual lock for the tape spool on this drive. Close the door, then press the white LOAD REWIND button, followed by the ON LINE button. The tape should load and go to the BOT point for the first tape file. If you have difficulty, consult a VAX advisor (see the list on the notice board beside the entrance to the main VAX room).

Z.1.5. Software tape mount at the VLA

When you have the tape mounted on the tape drive, VAX/VMS must also mount the tape in software. This should be done from within AIPS; type:

> INTAPE *n* ↵	to specify that your tape is mounted on the drive labeled *n*.
> DENSITY *m* ↵	to specify that the system is to write the tape at a density of *m* bpi, where *m* = 800, 1600, or 6250.
> MOUNT ↵	to mount the tape in software.

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*.

Please also remove the tape from the tape drive.

Z.1.6. Monitoring disk space at the VLA

See § 4.6 for a general discussion of the problem of disk space. Since disk space is often very tight, there are at the VLA a number of system utilities and procedures to assist the user in monitoring how much space is available and clearing additional space as required. These capabilities are available only at monitor (system job control) level. To use them:

> EXIT ↵	to exit from AIPS, saving your parameters in the LASTEXIT SAVE area.

The job control commands:

$ SHOW DEV MX ↵

$ SHOW DEV DU ↵

are similar to the AIPS verb FREESPAC and will list the vacant space on each of the disks. When the available space drops to less than about 30,000 blocks (or a lot more for large images) some programs will begin to fail. On both systems, disks DUA0, DUA1, and DUA2 together correspond to INDISK 1; MXA0 corresponds to INDISK 2; and MXA1 corresponds to INDISK 3.

On both systems, MXA1 may be used to hold private, dismountable disk packs. These can be assigned temporarily to users who expect to require large amounts of disk space. VLA staff will allocate and initialize a new disk for a user. The disk labels are based on the user's name; the default public packs are called VLAV2 on VAX1 and BEN on VAX3. It is the responsibility of the disk owner to mount and dismount his pack. If, by oversight, a private disk has been left mounted, then only an experienced disk user, who has been trained by VLA staff, should change the pack. A slot is provided on the sign-up sheets to reserve use of a private disk.

Sometimes there are scratch files which are no longer in use and can be deleted. To check this, exit to monitor level (prompt $) and type:

$ @SCRATCH ↵

to list all known scratch files and their creation times. If there are any old files listed, type:

$ @AJAX ↵

to delete all scratch files generated by the *AIPS* system which are no longer in use. This procedure is now safe; you should not be able to delete a scratch file which is currently being used by another *AIPS* user (or batch task). Type:

$ SHOW DEV MX ↵

$ SHOW DEV DU ↵

to find out if you have freed up enough disk space.

If destroying scratch files is not enough, you will need to find the worst disk space hogs and apply appropriate peer pressure. At monitor level, type:

$ @SPACE ↵

which spawns the process "SYSYPHUS" to find out who has how much disk space and when they last used it. After about 1-2 minutes a list of the disk usage of each user will appear on the line printer. You must then persuade one or more of these recalcitrants to relinquish some space. DO NOT delete files without the express permission of either the user or the System Manager.

Z.1.7. Solving problems at the VLA

Below are the details specific to the VLA VAX/VMS systems for handling some of the problems which may arise in *AIPS*.

Z.1.7.1. Stopping excess printout at the VLA

If you wish to abort a job which is currently printing out, do the following:

1. Turn the power off on the printer to conserve paper. For the laser printer, pull out the paper tray.
2. Get into monitor mode (prompt $) on the VAX which is producing the output. (Use EXIT ↵ to save your inputs and exit the program.)
3. Type STOP/ABORT SYS$PRINT ↵.
4. If you do not know which VAX is producing the output, type SHOW/QUE/ALL SYS$PRINT ↵ from monitor level on each VAX to find the one which has a job marked as "printing."
5. Type STOP/ABORT SYS$PRINT ↵ from this VAX.
6. For the laser printer, issue a STOP/ABORT QMS ↵ from monitor level on VAX3 regardless of which VAX initiated the print or plot.
7. Turn the power back on for the printer or replace the paper tray for the laser.

Z.1.7.2. Hard copy device

The Versatec printer/plotter is used as the hard copy device. Try the instructions in § 12.2. Paper for the Versatecs is kept on the metal cabinet in the room to the right just before entering the *AIPS* Caige.

Z.1.7.3. Recalcitrant blank tapes

If you have trouble writing to new, blank tape on one of the VLA VAXes and have checked that you have mounted the tape correctly on the drive specified by your OUTTAPE parameter, follow the procedure below.

1. Exit from AIPS and log out (LO ↵) which will dismount your tape.
2. Log in to the user name INITTAPE; there is no password.
3. Mount the tape on the desired tape drive, again.

4. The VAX will inform you what drives are available and what densities they support. Answer the questions about which drive the tape is on and what density you want.
5. The tape will be initialized and you will automatically be logged out of the INITTAPE account.
6. Log in to AIPS again, MOUNT the tape, and try again.
7. Be sure to specify DOEOT FALSE $\subset_R$ initially. Otherwise, the header written by INITTAPE will be left on the tape and cause problems later.

Note that writing to blank tape that has not been initialized in this manner can generate error messages suggesting gruesome hardware problems. Don't panic — consult a local expert first.

Z.1.7.4. Other problems at the VLA

Sometimes tasks under development or investigation are compiled and linked in a "debug" mode. If they have been left this way, they will wake up (after the GO command) and issue a prompt of DBG>. To run the program, type

DBG> GO $\subset_R$ to start it.

When it finishes, it may issue another prompt to which you reply:

DBG> EXIT $\subset_R$ to finish it.

Tell the System Manager so that the program may be relinked in normal mode.

Z.1.8. Editing RUN files at the VLA

RUN files are located in a directory called DISK$RES:[AIPS.RUN] which may also be referred to by the logical name RUNFIL:. RUN file names must begin with a letter and must have the three-letter extension *.nnn*, where *nnn* is your user number in hexadecimal with leading zeros. The files are edited from monitor level using either EDT or EMACS. For example, log in to the AIPS directory. From monitor level, type:

$ EDIT RUNFIL:MAPIT.13D $\subset_R$

to edit with EDT a file called MAPIT for user 317. For instructions on the editors consult manuals available from the System Manager.

Z.1.9. Making pictures on the VLA Dicomed

At the VLA site, exact copies of the I^2S screen can be transferred directly and fairly smoothly to 35-mm film. The 35-mm frames may contain a single screen image or a mosaic of up to six screen images; text captions may be added in a variety of colors and character sizes. The first step is to get the image that one wants recorded to look the way one wants it on the VAX I^2S screen using *AIPS*. Then use the T3VERB with an appropriate OUTNAME to record the screen onto a set of intermediate disk files. When you have saved all the images you want recorded, exit from AIPS and, at monitor level, type DICOMED $\subset_R$. This logs you into the CPU to which the Dicomed recorder is attached and sets a recording request program running. In dialogue style, you may specify all the 35-mm frames you want recorded. After reviewing a summary, you may delete or save the recording request and you will then be logged out and returned to the VAX. The request is saved in a file with a user-specified name (preferably your name) and will be picked up by the evening operator who will take care of the recording automatically. File names beginning with X will not be picked up by the operator. You should check with the operator to give him or her any special instructions, *e.g.*, to use black-and-white film. Processed film will be mailed to the requesters. This procedure works from both VLA VAXes and, in principle, could also be used from the Charlottesville and Tucson VAXes.

More detailed instructions are available in a labeled manual in the *AIPS* Caige or from Arnold Rots.

Z.2. Charlottesville VAX

The computer is a VAX 11/780 running the VMS operating system and equipped with an I^2S television display device and a Floating-Point Systems AP-120B array processor. It has three *AIPS* data disks and two tape drives. The *AIPS* Caige is located on the second floor in Room 215 of the Edgemont Road Office Building. The primary *AIPS* terminal is a Tektronix 4025 located at the back left (as you enter) next to the television monitor. Three other terminals (2 Visual 100s and a Visual 400) are located along the left side of the room. Additional VAX terminals are located in Room 214. The I^2S device is equipped with four grey-scale memory planes and four graphics overlay planes. A Tektronix 4012 is available in the *AIPS* Caige for TK graphics output. A Versatec in the *AIPS* Caige is used for printing and plotting. The QMS Lasergrafix laser printer in Room 214 produces plots of higher quality, however.

Z.2.1. Signing up for *AIPS* time on the CV VAX

There is a sign-up sheet for AIPS1 on the noticeboard just to the left of the entrance to the *AIPS* Caige, Room 215. If you wish to be certain of priority on the VAX, you should sign up for your *AIPS* time in advance. To promote fair and efficient use of the system, users are asked to restrict the amount of time that they reserve. Formal rules are not now in effect, but may be imposed should the need arise. The Charlottesville VAX also supports significant VLBI data reduction and most of the *AIPS* software development. These projects must receive their fair share of the computer resources.

The Charlottesville VAX allows up to six simultaneous *AIPS* users. The user signed up for AIPS1 has priority for the use of the displays, array processor, and tape drives, but is expected to be reasonable about sharing these resources with the programming staff and other users. Normally, the facilities available through the higher numbered AIPS*n* are sufficient for most users.

Z.2.2. Using the terminals on the CV VAX

To correct characters which you have typed, do not use the BACK SPACE key on VAX/VMS systems. Instead, on the Tektronix 4025, use the RUB OUT key at the upper right of the keyboard. On the Visual 100 and 400 terminals, press the DEL key as often as required.

During execution, paging can be halted by typing CTRL S and resumed by typing CTRL Q. To abort any execution, type CTRL Y or CTRL C. Using CTRL Y while in AIPS will unceremoniously eject the user to monitor level (prompt $). If CTRL Y was entered accidentally, AIPS may be resumed by typing CONT CR. If the abort was intentional, then the user will have to reinitialise AIPS with all the input parameters having been lost.

The Tektronix 4025 terminal has three pages of memory. To scroll upwards press the 1 key on the numeric pad to the right of the main keyboard. The 2 key on the numeric pad will scroll downwards. The Visual 400s also have three pages of memory. Hold the SHIFT key down and press the "up-arrow" key to scroll upwards and the "down-arrow" key to scroll back. Visual 100s remember only the current page.

Z.2.3. Logging in to *AIPS* on the CV VAX

Go to the *AIPS* Caige in Room 215 and find a terminal connected to the VAX. The Tektronix 4025, located next to the television monitor, is reserved for the user signed up for AIPS1. All other terminals are available on a first-come, first-served basis. We recommend the Visual 400s for use in *AIPS* since they have a multi-page memory. Typing CR_R on the terminal will produce a prompt which will tell you the current state of that terminal.

If the prompt symbol is > the terminal is already being used for AIPS. Check with other possible users before proceeding. If it's okay to use the system, type:

EXIT CR_R

If the prompt is $ (due to the initial CR_R or to following the step above), the computer is at system level, but not in use. Check nearby users to make sure and then type:

SHOW PROC CR_R

to establish the current owner of the terminal. If it is not AIPS, type:

LO CR_R

CR_R

to log out the previous user and to begin the log in process. If the current owner of the terminal is AIPS, type one of:

AIPS OLD CR_R

AIPS NEW CR_R

AIPS TST CR_R

Charlottesville is equipped with a "Digital Data Switch" (DDS), which allows a terminal to be connected to any one of several computers. The initial CR_R or the one given after logging out should produce the following messages from the DDS:

WELCOME TO THE NRAO CV DIGITAL DATA SWITCH

DDS PORT *mmm*

and a prompt for the desired computer. Enter:

CV DDS: CVAX CR_R

to connect to the CV VAX. You should then see the message:

CONNECTING PORT *nnn* (*xxxx* BAUD) TO PORT *nnn* (*yyyy* BAUD)

Hit CR_R until you get a Username prompt — once for *yyyy* = a numeric baud rate, twice for *yyyy* = 'AUTO'.

The initial CR_R or the ones given after logging out or completing the DDS connection should give a prompt of Username:, which means that you can log in. Proceed as follows:

Username: AIPS CR_R

The message Welcome to VAX/VMS Version ... should appear, followed by some system messages. (The Tektronix 4025 reserved for AIPS1 is not on the DDS and only the Username: prompt will appear.) The login procedure should then produce the statement

To start up AIPS type AIPS OLD or AIPS NEW or AIPS TST

will appear. Type in your choice. The OLD version is likely to be relatively free of bugs, but the NEW version will contain the recent, tested improvements. The TST version is under active development and should be used by NRAO staff only. (Note that this choice affects only the version of the AIPS program itself. You may choose TST, NEW or OLD versions of the *AIPS* reduction programs at a later time [see §4.5].)

You should then see the messages:

```
Starting 15OCT85 version of AIPS
BEGIN THE ONE TRUE AIPS NUMBER n
```

where 15OCT85 identifies the release of AIPS you have selected and *n* is a number between 1 and 6. AIPS will now ask you for your user number:

```
AIPS n:  ENTER USER ID NUMBER
? uuu ⏎
```

where *uuu* is your Charlottesville user number. The AIPS prompt > should now appear.

Z.2.4. Hardware tape mount on the CV VAX

The VAX has two tape drives. Drive number 1 is a TU77 drive located in the VAX (and IBM) machine room on the first floor (Room 112). The VAX is the computer all the way to the left (as you enter the room) next to the windows and the drive is adjacent to the CPU. Densities of 800 and 1600 bpi are supported. Tape drive number 2 is located in the *AIPS* Caige just behind the primary *AIPS* user's chair. It is a Systems Industries drive and supports 800, 1600, and 6250 bpi.

Z.2.4.1. Mounting tapes on the TU77

Open the door by pulling from the right hand edge. The upper spool is ready to receive or release a tape reel if a red band shows inside the spool hub. Mount your tape on this upper spool so that the protrusions in its protective rim engage the slots to the right hand side of the tape enclosure. Press the central lock in the hub until it snaps into position with no red band showing. Shut the door and press the white LOAD button at the left of the button row between the spools. The tape should thread itself and advance to the first file after which the BOT signal should light up. Press the ON LINE button — the ON LINE signal should light up. If this sequence fails repeatedly, consult a VAX advisor.

Z.2.4.2. Mounting tapes on the Systems Industries drive

Open the door by removing the metal "night door latch", if needed. The door will slide down. Pull out (toward you) the central lock in the hub. Mount the tape on the spool so that the protrusions on its protective rim engage the slots on the left side (slightly below center) of the tape enclosure. Then press the LOAD REWIND button, followed by the ON LINE button. The door will close and the tape should load and go to the BOT point for the first tape file. If you have difficulty, consult a VAX advisor.

Z.2.5. Software tape mount on the CV VAX

When you have the tape mounted on the tape drive, VAX/VMS must also mount the tape in software. This should be done within AIPS; type:

> INTAPE *n* ⏎	to specify that your tape is mounted on the drive labeled *n*.
> DENSITY *m* ⏎	to specify that the system is to write the tape at a density of *m* bpi, where *m* = 800, 1600, or 6250.
> MOUNT ⏎	to mount the tape in software.

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*.

Also, please remove the tape from the tape drive.

Z.2.6. Monitoring disk space on the CV VAX

See § 4.6 for a general discussion of the problem of disk space. The verb **TIMDEST** is useful, but we prefer that users ask the System Manager before invoking it. A number of disk packs are available for disk drive number 3. The packs are called **VISTOR** (for our guests from out of town), **VLBI1** (for VLBI *AIPS* visitors), **VLAV2, ALBERT, BRYAN, CALVIN, EDWARD** and **FRITZ**. Users with large disk space requirements should inquire about having one of these temporarily assigned to them and should force their output files to disk 3. Disk packs are changed *only* by the IBM/VLBI-processor operators. When you are the primary *AIPS* user, you may ask them to mount the pack assigned to you. At other times, you must negotiate with the other users of the VAX before asking the operators to change packs.

Since disk space is often very tight, there are on the Charlottesville VAX a number of system utilities and procedures to assist the user in monitoring how much space is available and clearing additional space as required. These capabilities are available only at monitor (system job control) level. To use them:

`> EXIT` ↵ to exit from **AIPS**, saving your parameters in the **LASTEXIT SAVE** area.

The job control command:

`$ FREE` ↵

is similar to the **AIPS** verb **FREESPAC** and will list the vacant space on each of the disks. When the available space drops to less than about 30,000 blocks (or a lot more for large data sets and images) some programs will begin to fail. On CVAX, the disk **UMA1** corresponds to **INDISK 1**; **DUA0, DUA1**, and **DUA2** together correspond to **INDISK 2**; and **UMA2** corresponds to **INDISK 3**. The other disks are used for the system and for VLBI, *AIPS* and private program development. The **FREE** commands in **AIPS** and job control are also useful for determining if your assigned disk pack is currently mounted.

Sometimes there are scratch files which are no longer in use and can be deleted. To check this, exit to monitor level (prompt **$**) and type:

`$ @SCRATCH` ↵

to list all known scratch files and their creation times. If there are any old files listed, type:

`$ @AJAX` ↵

to delete all scratch files generated by the *AIPS* system which are no longer in use. This procedure is now safe; you should not be able to delete a scratch file which is currently being used by another *AIPS* user (or batch task). Type:

`$ FREE` ↵

to find out if you have freed up enough disk space. Note that the **AIPS** command **TIMDEST** performs, among other things, the same function as **AJAX**.

If destroying scratch files is not enough, you will need to find the worst disk space hogs and apply appropriate peer pressure. At monitor level, type:

`$ @SPACE` ↵

which spawns the process **"SYSYPHUS"** to find out who has how much disk space and when they last used it. After about 1-2 minutes a list will appear on the line printer of the disk usage of each user. You must then persuade one or more of these recalcitrants to relinquish some space. DO NOT delete files without the express permission of either the user or the System Manager (other than with the **AIPS** verb **TIMDEST**).

Z.2.7. Solving problems on the CV VAX

Below are the details specific to the Charlottesville VAX/VMS system for handling some of the problems which may arise in *AIPS*.

Z.2.7.1. Stopping excess printout on the CV VAX

If you wish to abort a job which is currently printing out, do the following:

1. Turn off power to the printer, to conserve paper. For the laser printer, pull out the paper tray.
2. Get into monitor mode (prompt $). (Use EXIT CR_R to save your inputs and exit the program.)
3. Type STOP/ABORT SYS$PRINT CR_R. For the laser printer, substitute TXA0: for SYS$PRINT.
4. Turn the power back on for the printer or replace the paper tray for the laser.

Z.2.7.2. Hard copy device

The (ModComp) Varian printer/plotter is used as the hard copy device. Try the instructions in § 12.2. Paper for the Varian and the Versatec is kept in the ModComp computer room (Room 216) on the floor at the back right. Be sure to get the correct type for the printer you are filling.

Z.2.7.3. Recalcitrant blank tapes

If you have trouble writing to new, blank tape on the CV VAX and have checked that you have mounted the tape correctly on the drive specified by your OUTTAPE parameter, follow the procedure below.

1. Exit from AIPS and log out (LO CR_R) which will dismount your tape.
2. Log in to the user name INITTAPE; there is no password.
3. Mount the tape on the desired tape drive, again.
4. The VAX will inform you what drives are available and what densities they support. Answer the questions about which drive the tape is on and what density you want.
5. The tape will be initialized and you will automatically be logged out of the INITTAPE account.
6. Log in to AIPS again, MOUNT the tape, and try again.
7. Be sure to specify DOEOT FALSE CR_R initially. Otherwise, the header written by INITTAPE will be left on the tape and cause problems later.

Note that writing to blank tape that has not been initialized in this manner can generate error messages suggesting gruesome hardware problems. Don't panic — consult a local expert first.

Z.2.7.4. Other problems on the CV VAX

Sometimes tasks under development or investigation are compiled and linked in a "debug" mode. If they have been left this way, they will wake up (after the GO command) and issue a prompt of DBG>. To run the program, type

DBG> GO ↵ to start it.

When it finishes, it may issue another prompt to which you reply:

DBG> EXIT ↵ to finish it.

Tell the System Manager so that the program may be relinked in normal mode.

Z.2.8. Editing RUN files on the CV VAX

RUN files are located in a directory called UMA0:[AIPS.RUN] and must have the three-letter extension *.nnn* in their names, where *nnn* is your user number in hexadecimal with leading zeros. They are edited from monitor level using either EDT or EMACS. For example, log in to the AIPS directory. From monitor level, type:

$ EDIT RUNFIL:MAPIT.03D ↵

to edit with EDT a file called MAPIT for user 61. For instructions on the EDT editor consult the appropriate VAX/VMS Manuals. An abridged instruction manual for the EMACS editor is available from Don Wells or Gary Fickling.

Z.2.9. Making pictures on the Charlottesville Dicomed

In Charlottesville, a rather inelegant route is still required to get from *AIPS* to the Dicomed. FITS tapes must be written by FITTP, read in to the IBM via FITS2DEC, converted to a picture tape via DECPIX, and then brought back up to the ModComp computer. The ModComp program DCP interprets the picture tapes, recording images and labeling on the Dicomed. Consult Kerry Hilldrup or Eric Greisen for details.

Z.2.10. Orange baked bananas

1. Mix in a saucepan 1/2 cup firmly packed **brown sugar**, 1 tablespoon **cornstarch**, 1/8 teaspoon **cinnamon**, and a few grains **salt**.
2. Add gradually, blending in 3/4 cup boiling water.
3. Bring rapidly to boiling and cook about 5 minutes or until sauce is thickened, stirring constantly.
4. Remove from heat and blend in $1\frac{1}{2}$ teaspoons grated **orange peel**, 1/4 cup **orange juice**, 1 teaspoon **lemon juice**, and 2 tablespoons **butter**.
5. Peel and cut into halves lengthwise 6 **bananas** with all-yellow or green-tipped peel.
6. Arrange halves cut side down in baking dish and brush with about 2 tablespoons melted **butter**.
7. Sprinkle 1/2 teaspoon **salt** over bananas and then pour the orange sauce over bananas.
8. Bake at 375° for 10 to 20 minutes.

Z.3. Charlottesville ModComp

The computer is a ModComp Classic equipped with an I^2S television display device and a Floating-Point Systems AP-120B array processor. It has three disks and two tape drives. The *AIPS* Caige is located in Room 215 on the second floor of the Edgemont Road Office Building. There are four ModComp terminals: 2 Tektronix 4025 text terminals and two Tektronix 4012 graphics terminals. They are labeled to show their various functions. The primary *AIPS* text terminal is located at the back right of the room (as you enter) next to the television monitor. The 4012 to its right (labeled A02) is used by AIPS1 for its graphics output. The other 4025 is primarily an Operator terminal, but may be used for AIPS2. The other green screen (labeled TY1) is then used for its graphic output. The I^2S device is equipped with four grey-scale memory planes and four graphics overlay planes. A Varian is used for plotting and printing.

Z.3.1. Signing up for *AIPS* time on the ModComp

There is a sign-up sheet for *AIPS* on the noticeboard just to the left of the entrance to the *AIPS* Caige, Room 215. If you wish to be certain of the availability of the ModComp, it is advisable to sign up for your *AIPS* time in advance. To promote fair and efficient use of the system, users are asked to restrict the amount of time that they reserve. Formal rules are not now in effect, but may be imposed should the need arise.

AIPS supports up to two simultaneous interactive users, plus batch queues, on the ModComp. The user signed up for AIPS1 has priority for the use of the displays, array processor, and tape drives, but is expected to be reasonable about sharing these resources. Normally, the facilities available through AIPS2 are *not* sufficient for most users. The ModComp simply cannot handle the loads which even one experienced *AIPS* user can readily generate.

Z.3.2. Using the Tektronix terminals on the ModComp

To correct characters which you have typed, use the BACK SPACE key on the ModComp. The RUB OUT key (at the upper right of the keyboard) is interpreted as deleting the entire line.

During execution, paging *cannot* be halted, nor can the program be aborted (by normal means). Fortunately, the Tektronix 4025 terminal has three pages of memory. To scroll upwards press the 1 key on the numeric pad to the right of the main keyboard. The 2 key on the numeric pad will scroll downwards. If a program must be aborted, go to the Operator terminal (4025 labeled TY), press the CTRL key and hit the A key. (This signals your desire to talk to the OC [Operator Communications] program.) Then type:

/AIPS1/A $^{C}_{R}$

to abort, for example, the AIPS1 program.

Z.3.3. Logging in to *AIPS* on the ModComp

Go to the *AIPS* Caige in Room 215 and find a terminal connected to the ModComp. The Tektronix 4025, located next to the television monitor, is reserved for the user who has signed up for AIPS1. Typing $^{C}_{R}$ on the terminal will produce a prompt which will tell you the current state of that terminal.

If the prompt symbol is > the terminal is already being used for AIPS. Check with other possible users before proceeding. If it's OK to use the system, type:

EXIT $^{C}_{R}$

If the prompt is **$** (due to the initial ⏎ or to following the step above), the computer is at system level, but not in use. Check nearby users to make sure and then type:

$ JOB ⏎	to initialize the job control processor.
$ AIPS ⏎	to make an AIPS run.

No matter which terminal the above commands were entered upon, they will make AIPS1 begin on the terminal next to the television monitor or, if AIPS1 was already running, they will make AIPS2 begin on the Operator terminal. These commands will also terminate any jobs running on the green screen appropriate to the AIPS being started. If someone is using the relevant green screen, follow the instructions in the next paragraph.

On the ModComp, it is possible to detach all job control processors from a terminal. Thus, your initial ⏎ may produce no reaction at all. In this case, try some other terminal, particularly the Operator terminal. If a ⏎ still produces no reaction, try on the Operator terminal:

CTRL A	to get OC's attention.
//I ⏎	to see if the system is working at all.
/AIPS1/E,,LMV ⏎	to start AIPS1.

If the CTRL A produces no response, then the ModComp has probably halted. See the booting instructions in § Z.3.7.3.

You should then see the message:

```
BEGIN THE ONE TRUE AIPS NUMBER n
```

where *n* is 1 or 2. AIPS will now ask you for your user number:

```
AIPS: ENTER USER ID NUMBER
? uuu ⏎
```

where *uuu* is your Charlottesville user number. The AIPS prompt > should now appear.

Z.3.4. Hardware tape mount on the ModComp

The ModComp has two identical tape drives located in the *AIPS* Caige. They are labeled MT1 and MT2, corresponding to the adverb INTAPE values of 1 and 2, respectively. Densities of 800 and 1600 bpi are supported.

To mount a tape on one of these drives, open the door by pulling from the left hand edge. Push in the spot marked PRESS in the hub of the upper spool. Remove any protective rims from the tape and slide it over the upper spool. Then engage the tape lock by pressing on a spot opposite that marked PRESS in the hub of the upper spool. The tape must be threaded manually. Follow the black lines drawn on the unit around the various capstans and onto the take-up spool at the bottom. Wind the tape onto the take-up spool for several turns. Then close the door and press the white button marked LOAD (2nd from the top). The tape should advance to the BOT point. Press the ON LINE button. The PE/NRZI lighted button should be on for 1600 bpi tapes and off for 800 bpi tapes. Press the button, if needed, to get this state.

When you are done with a tape, press the RESET button and then the REWIND button. This will cause the tape to rewind to the BOT point. Press the REWIND button again to dismount the tape.

Z.3.5. Software tape mount on the ModComp

The ModComp is happy to take your word (or anyone else's) that the hardware-mounted tape is yours. Therefore, no software mount is required and ModComp users do not have to exit from AIPS to mount and dismount tapes. The AIPS verbs MOUNT and DISMOUNT simply cause the tape to rewind and an appropriate message to appear. However, there is some danger that another user might accidentally read or write your tape. Therefore, cautious users do not leave their tapes ON LINE longer than they have to and do not insert write rings unless they are about to write on their tapes.

Z.3.6. Monitoring disk space on the ModComp

See § 4.6 for a general discussion of the problem of disk space. The total available disk space may be displayed with the verb FREESPAC. The verb TIMDEST is useful, but it is preferred that users ask the System Manager before invoking it. On the ModComp, the *AIPS* catalog files are "public" — all users' images are cataloged in the same file. Type:

> USER = 32000 $\subset_R$	to include all users.
> INDISK = 0 $\subset_R$	to include all disks.
> MCAT $\subset_R$	to list all image files.
> UCAT $\subset_R$	to list all *uv* files.

The users who have large numbers of files, particularly *uv* files, are the users on whom you wish to apply pressure in order to obtain free disk space. Typing:

> GO DISKU $\subset_R$

with the above adverb values, provides more exact and detailed information, but takes a long time to run.

Sometimes there are scratch files which are no longer in use and can be deleted. One way to do this is to exit to monitor level (prompt $) and type:

$ JOB $\subset_R$	to initialise the job control processor.
$ ASS 5 LO 6 LO $\subset_R$	to assign the terminal.
$ EXEC AJAX,LMV $\subset_R$	to delete any scratch files.

AJAX does take a while to finish, so be patient. It will report any files it deletes. Do *not* run AJAX if any tasks are executing.

Z.3.7. Solving problems on the CV ModComp

Below are the details specific to the Charlottesville ModComp system for handling some of the problems which may arise in *AIPS*.

Z.3.7.1. Stopping excess printout on the ModComp

If you wish to abort a job which is currently printing out do the following:

1. Turn off the printer to conserve paper.
2. Make sure that the *AIPS* task doing the printing has stopped or ABORT it. (See § 12.1.)
3. Go to the Operator terminal and get OC's attention with a CTRL A. Type /S/A $\subset_R$ to stop the print out.
4. Turn the power back on for the printer.

Z.3.7.2. Hard copy device

The ModComp Varian printer/plotter is used as the hard copy device. Try the instructions in §12.2. Paper for the Varian and the Versatec is kept in the ModComp computer room (Room 216) on the floor at the back right. Be sure to get the correct type for the printer you are filling.

Z.3.7.3. Booting the ModComp system

The ModComp can get hung up fairly easily. The easiest cure, if all (any) current users agree, is to boot the system. Go to the ModComp machine room (Room 216). The CPU is at your left as you enter the door — it has flashing lights (like a *real* computer). Turn the RUN / HALT toggle switch to the HALT (upper) side. Make sure that the numbered switches are all in the horizontal position except switch 15 which should be up. Then, depress the lower side of the toggle switches labeled M CLEAR, FILL, and RUN in that order leaving the last down in the RUN position. The lights should do a lot of blinking. After a couple of minutes, you will be asked on the Operator terminal to provide the date and time. Please enter these accurately using a 24-hour clock. The system should then provide the $ prompt on the Operator terminal.

Z.3.7.4. Other problems on the ModComp

The ModComp has a wide bag of tricks to play on experienced and inexperienced users alike. We will add more hints in this section as they arise.

Inside the ModComp operating system lives a gremlin called TMP. For cognoscenti, TMP can perform wondrous tricks. However, TMP can take control of a terminal without warning (thinking some static electricity discharge called him from his bottle). When a terminal appears dead or TKPL asserts that it can't access the green screen now, one thing to try is to type, on the Operator terminal:

CTRL A	to get OC's attention.
/TMP/A ^R	to abort TMP.

This incantation may resurrect your terminal.

The ModComp reacts to devices which are not ready by issuing an error message to the Operator terminal and placing the device in a software "Off Line" status. To bring the device back on line, first mount the required tape, add the needed paper, or whatever. Then turn the power on and type on the Operator terminal:

CTRL A	to get OC's attention.
/ON LPP ^R	to turn on the line printer.
/ON MT1 ^R	to turn on tape drive 1.
/ON MT2 ^R	to turn on tape drive 2.
/ON SP ^R	to turn on the plotter part of the Varian.

The ModComp normally gives the AIPS program high priority, *AIPS* tasks medium priority, and other users low priority. Such an arrangement is not always needed. To change the priority of any program, type on the Operator terminal:

CTRL A	to get OC's attention.
/pgmname/CHA *pn* ^R	where *pgmname* is the full name of the program to be changed and *pn* is the priority it is to assume.

The normal values of *pn* are 128 (high), 200 (medium), and 255 (low). Since TPOWER users don't require much CPU time or real memory, it is friendly to change them to 128 or so. Since the ModComp is not a virtual memory machine, it can get clogged with too many equal priority tasks all rolling in and out and competing for the available resources. Changing AIPS1 to low priority often helps and, in bad situations, changing some of the tasks to low priority may also be needed.

There is a useful tape copy routine on the ModComp. Get to monitor level (prompt $) and type:

$ JOB CR	to initialize the job control processor.
$ ASS PL DQL CR	
$ ASS BI MT1 CR	the input tape drive name.
$ ASS BO MT2 CR	the output tape drive name.
$ EXEC TAPTAP,PL CR	to copy the first file.
$ WEOF BO CR	to write an EOF (if desired) on the output tape.

This may be repeated for as many files as needed. Note that the output files are concatenated if you do not type the WEOF BO CR command.

Z.3.8. Editing RUN files on the ModComp

RUN files are located in a special directory on the ModComp and may have any name of ≤ 8 characters. They are edited from monitor level using SEDIT. For instructions on this editor consult the MAX II/III/IV System Processors Manual. To get into SEDIT in the appropriate directory, type on any terminal at monitor level:

$ JOB CR	to initialize the job control processor.
$ RUN CR	to make appropriate assignments and start SEDIT.

A session in which you create a new file might go as:

RAN REC CR	to go to random mode.
D 1 1000 CR	to clear work area.
A CR	to start typing new lines.
any lines you want of POPS text	
.CAT *filename* CR	to catalog the new RUN file.
E CR	to exit from SEDIT.
$ JOB CR	to initialize the job control processor again.
$ AIPS CR	to resume AIPS.