6 DISPLAYING YOUR DATA

This chapter is concerned with the ways in which you may display your data. There are a number of tasks for generating "plot files" which contain graphics commands for the making of various displays of your *uv* and image data. All of these now offer a "preview" option to draw the plot directly on the *AIPS* TV, rather than putting the commands into a file. Once the files are created, a variety of tasks may be used to translate them into displays on various devices, such as the *AIPS* TV and graphics windows and PostScript printers and film recorders. There are also verbs to display and manipulate your images on the *AIPS* TV and a single task `TVCPS` to capture that display, if desired, into a PostScript file for printing, recording on film, or even including in your scientific papers.

Several indices of the *AIPS* software are relevant to this discussion. To generate current lists of *AIPS* functions on your workstation window (or terminal) use ABOUT HARDCOPY C_R, ABOUT INTERACT C_R, ABOUT PLOT C_R, ABOUT TV-APPL C_R, and ABOUT TV C_R. The `15JAN95` versions of these indices are reproduced in Chapter 13 of this *CookBook*.

6.1 Getting data into your *AIPS* catalog

By the time you reach this chapter, most of your data will probably already be loaded into your *AIPS* catalog either by reading an external tape or disk or by being generated by some *AIPS* task. Visibility data which are not presently on disk may be read by the *AIPS* tasks `FILLM`, `UVLOD` and `FITLD`; see § 4.1 and § 5.1 for details. Images that are generated by other imaging systems, (*e.g.*, images from non-NRAO radio telescopes or non-radio images) can be transported to *AIPS* by writing them out of the other imaging system on tape or disk in the standard FITS format. The tasks `IMLOD` and `FITLD` can then be used to read them into *AIPS*. These tasks are also used to read images saved with `FITTP` from previous *AIPS* sessions.

6.1.1 `IMLOD` and `FITLD` from tape

`IMLOD` and `FITLD` will position tapes for you using the `NFILES` adverb. It is safer to use `AIPS` verbs instead to position the tape and to check that positioning. First, mount your tape in hardware and software as described in § 3.9. To move the tape forward by *nf* file marks to position it at the first interesting image, enter:

> INTAPE *n* C_R	to specify the tape drive labeled *n*.
> NFILES *nf* C_R	to specify the number of file marks to move the tape.
> AVFILE C_R	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. Once you have moved part-way into a tape, you may use $nf < 0$ to move backwards to the abs(nf) previous file. In all cases, the tape is left at the beginning of a file. Task `PRTTP` is an invaluable aid to determine what is on your tape and where it is; see § 3.9.4 and § 5.1.1. If you happen to come across a CV-IBM format tape from some astronomical museum, the verb `AVMAP` may also be needed. Type HELP AVMAP C_R for details. You must use `IMLOD` not `FITLD` for such antiques.

To check that the tape is positioned where you expect, type:

> TPHEAD C_R

Your terminal will then list 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:

> OUTDI *n* C_R	to specify writing the image to your *AIPS* catalog on disk *n*.

> OUTNAME '*your-chosen-name*' C_R	to specify the output disk file name in *AIPS*; the default is the image name on tape if FITTP was used to write the image to tape.

The string *your-chosen-name* can be any ($\leq$ 12-character) title that you want to use as the image name within *AIPS* and should be specified for images from other image-processing software systems. FITLD also allows you to specify the 6-character image "class" parameter. Use OUTCLASS '*abcdef*' C_R, if you wish to change the class from that on your input tape as the image is read or if the image comes from a "foreign" system.

> OUTS -1 C_R	to keep the sequence number the same as that on tape; the default is the highest unique number for images with this name and class in your current *AIPS* catalog.
> NFILES 0 C_R	to have no further files skipped — **important** if you have just used AVFILE to position the tape!
> NCOUNT m C_R	to load m images consecutively starting with the image at the current tape position; default is $m = 1$. If you use this option, do *not* specify the OUTNAME unless you want the same name for all the new images in your catalog.
> GO FITLD C_R	or IMLOD, to run the task.

If OUTNAME is left unspecified, it defaults to the "name" of the image read from the FITS header — either the name previously used in earlier image processing or the source name. If OUTCLASS is unspecified, it defaults to the Class previously used in earlier image processing or to a compound name (*e.g.*, IMAP, IBEM, QMAP, ICLN) which attempts to describe the image. These defaults are frequently good ones when you are loading multiple consecutive images with NCOUNT > 0. You may of course change the *AIPS* image and class names later by using RENAME (see § 3.3.3 of this *CookBook*).

To load m consecutive further images from the same tape using the default OUTNAME (the names from their FITS header), skipping n from the sequence:

> OUTNAME ' '; OUTCL ' ' C_R	to ask for the system defaults.
> NFILES n C_R	to skip n file marks.
> NCOUNT m C_R	to specify loading m consecutive images after the skip.
> GO FITLD C_R	or IMLOD, to run the program.

To dismount the tape when FITLD is done:

> DISMOUNT C_R

6.1.2 IMLOD and FITLD from FITS-disk

FITLD and IMLOD can also read FITS-format images from external disk files into your *AIPS* catalog. This option is indicated by setting the adverb INFILE to a non-blank value. The control parameters are the same as described above for reading FITS tapes, except that INTAPE, NFILES and NCOUNT are ignored. Disk image files must therefore be read in only one at a time (per execution of IMLOD). INFILE is a string of up to 48 characters that must completely specify the disk, directory, and name of the input disk file to your computer's operating system. See § 3.10.3 for a discussion of FITS-disk files.

One "feature" of *AIPS* complicates this otherwise straightforward disk analog of FITS tape reading. AIPS translates all of your alphabetic inputs to upper case (this was demanded by users who otherwise became confused between upper and lower cases). So if your computer distinguishes upper and lower cases for disk, directory, or file names, you must do two things to prepare for this before running *AIPS*. First, you must restrict your external disk file names to upper-case characters and numbers. Second, you must set an upper-case "environment variable" or "logical" to point to the disk and directory where your FITS-disk images are stored before you run AIPS. You may need help from your System Manager when doing this for the first time. A common strategy on UNIX machines is to create an upper-case logical name after logging in but before starting up AIPS:

% setenv MYLOGICAL *myarea* C_R	if using C-shell, or
$ export MYLOGICAL=*myarea* C_R	if using korn, bourne, or bash shells,

where MYLOGICAL is an all-upper-case string of your choice and *myarea* is the full path name of the disk directory that contains your FITS-disk data. AIPS usually provides a public disk area known as FITS which you may use.

Then, once inside AIPS, tell FITLD or IMLOD:

> INFILE 'MYLOGICAL:IMAGE.DAT' C_R	to read in the FITS-disk file *myarea*:IMAGE.DAT.

6.2 Printer displays of your data

The most old fashioned way to look at your data — and the most exact — is simply to print it out and read the numbers. AIPS provides a variety of tasks and verbs to print visibility data, image data, tabular data, and miscellaneous other information. All of these tasks and verbs allow you to specify where the printed output goes using two adverbs, DOCRT and OUTPRINT. If DOCRT ≤ 0 and OUTPRINT is blank, then the output is placed in a temporary file and queued to the printer you selected when starting AIPS; see § 2.2.3. If DOCRT ≤ 0, a non-blank OUTPRINT specifies a text file into which the output is to be written; see § 3.10.1. The current output is appended to the file if it already exists. Thus, you can combine a number of printed outputs for later editing and/or printing. If DOCRT > 0, the output is directed to your workstation window or terminal. If DOCRT ≥ 72, its value is taken as the width of the display window. Since most print tasks display more information on wider windows, we recommend widening your window to 132 characters and specifying DOCRT = 132, which is the upper limit. The print routines will pause whenever the screen is full and offer you the choice of continuing or quitting. Thus, you can start what might be a very long print job, find out what you wanted to know after a few screens full, and quit without using up any trees.

6.2.1 Printing your visibility data

Before beginning calibration, it is a very good idea to make a summary list of the contents of your data set. LISTR with OPTYPE = 'SCAN' will list the contents of each scan in the data set. DTSUM also produces a listing summarizing the data set in either a condensed or full form.

The most basic display of your visibility data is provided by PRTUV which lists selected correlators in the order they occur in the data set:

> TASK 'PRTUV' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and data set to print.
> CHANNEL *c* ; BIF 1 C_R	to print starting with channel *c* from IF 1.
> BCOUNT *m*; XINC *i* C_R	to print every $i^{\rm th}$ visibility starting with the $m^{\rm th}$ visibility in the data set.
> DOCRT 132 ; GO C_R	to run the task with display on the terminal.

When you have seen enough, enter q C_R or Q C_R at the page-full prompt.

You may limit the sources, range of projected baselines, and times displayed and may select only one baseline or one antenna. PRTUV does not apply calibration or flagging tables. To get a similar display with all "standard" calibration, flagging, and data selection (optionally) applied, use the task UVPRT. LISTR also uses all of the calibration options to list the data in simple lists or in a display showing all the baselines at each time in a matrix form. `SHOUV` also makes list of calibrated visibility data with options to average all channels in each IF and to display closure rather than observed phases.

There are a number of tasks used to diagnose possible problems in your data and to print information about them. `UVFND` examines a data set for excess fluxes, excess apparent V-polarization, or simply any data with a specified fringe spacing and position angle or a specified range in u and v. As it does this, it also checks for bad antenna numbers, bad times, and (optionally) bad data weights. `CORER` examines a data set for excessive mean values and rms in each correlator (after subtracting a point source at the origin). `RFI` examines the rms fluctuations in the real and imaginary visibilities of each correlator looking for (and reporting) periods of apparent RF interference. `UVDIF` directly compares two data sets reporting any excess differences. It is useful for determining whether your latest operations (flagging, self-cal) have made a significant (or any) difference.

6.2.2 Printing your image data

The most basic display of an image is a print out of the numbers it contains. Such a display is provided by `PRTIM`:

> TASK 'PRTIM' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and image to print.
> NDIG 3 C_R	to use 3 digits, printing numbers between -99 and 999 with appropriate power of 10 scaling.
> FACTOR 10 C_R	to raise the default scaling by a factor of 10, overflowing regions of high values to see low valued regions better.
> BLC 0 ; TRC 0 C_R	to see the whole image.
> XINC 2 ; YINC 2 C_R	to see every other column and every other row.
> DOCRT FALSE ; GO C_R	to print the image on the selected printer.

Other imaging tasks which use the printer are `BLSUM` and `ISPEC`, which compute and print spectra by summing over regions of each plane in a data cube (see § 8.6), and `IMFIT`, `JMFIT`, and `SAD`, which fit one or more Gaussians to an image (see § 7.5). `IMTXT` writes an ASCII-formatted file containing an image.

6.2.3 Printing your table data

If you have any doubts about the contents of tables in *AIPS* it is best to resolve them by looking at the contents of the tables involved. `PRTAB` is a very general task which will print the contents of any *AIPS* table file. For example, to print flag table version 1:

> TASK 'PRTAB' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry to print.
> INEXT 'FG' ; INVERS 1 C_R	to select flag table version 1.
> BCOUNT 0 ; ECOUNT 0 ; XINC 1 C_R	to print everything.
> DOHMS TRUE C_R	to print times in sexagesimal notation.
> DOCRT 132 ; GO	to print the flag table on the terminal.

When you have seen enough, enter `q` C_R or `Q` C_R at the page-full prompt. For a table with a significant number of columns, `PRTAB` shows all rows for the first columns and then loops for the next set of columns. To see all columns for some rows, set a low `ECOUNT` value or be very patient.

Some of the tables have specialized printing programs. These include `PRTAN` for antenna tables, `PRTCC` for Clean component tables, `LISTR` with `OPTYPE = 'GAIN'` for calibration, solution, and system temperature tables, and **PRTGA** for gain tables of the type used by **ASCAL**. The verb **EXTLIST** will list information about various extension files, particularly plot files (see below), which may be printed with **PRTMSG**. **OFMLIST** is a verb to print the contents of an *AIPS* TV color table. Finally, task **TBDIF** will compare columns of two tables and print information about their differences.

6.2.4 Printing miscellaneous information

There is a variety of miscellaneous information which may also be sent to the printer in the same way. Verb **PRTMSG** prints selected contents of the *AIPS* message file; see § 3.2. Verb **PRTHI** prints selected lines from a history file; see § 3.4. Pseudoverb **ABOUT** prints lists of *AIPS* symbols by category while pseudoverbs **HELP** and **EXPLAIN** print information about a selected symbol; see § 3.8. Task **PRTTP** prints the contents of magnetic tape volumes; see § 5.1.1 and § 3.9.4. Task **PRTAC**, which may also be run in a stand-alone mode, prints information selected from the *AIPS* accounting file.

Task **TXPL** will attempt to represent an *AIPS* plot file (see below) on the printer. This will not work well for complicated plots, but, for simple plots, it may be the only way someone running over a slow telephone line can see his/her data in plot form.

6.3 Plotting your data

The basic concept in *AIPS*' plotting is to use some task to create and write a device-independent plot file as a PL extension file to a cataloged image or visibility data set and then to use some device-dependent task to interpret that file for the desired output device. Plot files are not overwritten by subsequent plot tasks. Instead they make new plot files with higher "version" numbers. The device-dependent tasks include **TVPL** (*AIPS* TV devices including **XAS**), **TKPL** (Tektronix graphics devices including *AIPS*' **TEKSRV** server), **TXPL** (line printers), and **LWPLA** (PostScript printer/plotters). Tasks called **PRTPL**, **QMSPL**, and **CANPL** support Versatec, QMS, and Canon printer/plotters if you still have one of these around. To plot on a PostScript printer/plotter:

> TASK 'LWPLA' ; INP C_R	to review the inputs.
> INDI *n* ; GETN *ctn* C_R	to select the disk and catalog entry to print.
> PLVER *n* C_R	to plot the $n^{\rm th}$ plot file.
> OUTFILE ' ' ; GO C_R	to do the plot immediately.

LWPLA offers the option to save the file for later plotting or inclusion as encapsulated PostScript in other documents. It also has options to control scaling, output paper size, width and darkness of lines, and transformation of grey-scale intensities.

All *AIPS* plot tasks now offer a "preview" option. If you set **DOTV = TRUE** when running any plot task, then the plot appears immediately on the *AIPS* TV display and no plot file is generated. This option allows you to make sure that the parameters of the plot are reasonable and lets you avoid making files and wasting paper for quick-look plots. Additional options allow you to control which graphics channel is used for the line drawing (**GRCHAN**) and to select pixel scaling of the plot at your specified location on the TV screen (**TVCORN**). These two options allow you to view more than one plot at a time on the TV, usually for purposes of comparison. Each graphics channel on the TV has a different color and a complementary color is used when two or more channels are on at the same point. This allows for a fairly detailed and effective comparison of plots, all of which may be captured with task **TVCPS** (see below). (Since the **DOTV** option can be fairly slow on complicated plots, you may prefer to use **TKPL** on plot files produced with **DOTV FALSE**.)

You can review the parameters of the plot files associated with a given image or visibility data set by typing:

> INDI *n* ; GETN *ctn* CR	to select the disk and catalog entry to print.
> INEXT 'PL' ; EXTLIST CR	to list summaries of the 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 CR	to review the inputs required.
> INEXT 'PL' ; INVERS *m* CR	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 CR	to do the deletion.
> INVERS 0 CR	to reset the version number to its default — usually advisable.

6.3.1 Plotting your visibility data

The most basic plot program for visibility data is called `UVPLT`. It allows you to select the x and y axes of the plot from real, imaginary, amplitude, and phase of the visibility, time, and projected baseline length, position angle, u, v, and w. It offers all of the usual calibration and data selection options and it plots the selected points individually or in a controlled number of bins along the x axis. For example, to plot calibrator phases as a function of time for all baselines to one antenna:

> TASK 'UVPLT' ; INP CR	to review the inputs.
> INDI *n* ; GETN *ctn* CR	to select the disk and catalog entry of the data set.
> BPARM = 11,2 CR	to plot time in hours on the x axis and phase in degrees on the y axis.
> SOURCES ' ' ; CALCODE '*' CR	to select all calibrator sources.
> XINC 4 CR	to plot only every fourth selected sample.
> ANTENNA 2,0 ; BASELINE 0 CR	to do all baselines with antenna 2.
> DOCRT = -1 ; GO CR	to make a plot file of these data.

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

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

There are several other tasks to plot your visibility data. `VPLOT` is a new and faster version of `VBPLT`, both of which plot all of the parameters offered by `UVPLT`, but one baseline at a time with multiple baselines per page (*i.e.*, per plot file) and multiple pages per execution. `CLPLT` is a similar task but restricted to plotting closure phases as a function of time. All three of these can also plot a source model (based on Clean components) as well as the observations. For spectral-line users, `POSSM` plots visibility spectra averaged over selected baselines and time intervals. For VLBI users (primarily), `FRPLT` plots visibilities versus time or, more importantly, the fringe rate spectrum. To examine the statistical distribution of your data, try `UVHGM` which plots histograms showing the number of samples or weights versus a wide variety of parameters.

Two plot programs actually convert visibility data to the image plane for plotting. Observers of point objects which might vary with time either intrinsically or by scintillation (*e.g.*, stars, masers) might wish to try `DFTPL`, which plots the Fourier transform of the data shifted to a selected position as a function of time. VLBI spectral-line observers may need to use `FRMAP`, which performs imaging via fringe-rate inversion and plots the loci of possible source positions.

6.3.2 Plotting your image data

Plot symbols (*e.g.*, plus signs) may be drawn on the plots produced by `CNTR`, `PCNTR`, `GREYS`, `KNTR` and several of the other tasks mentioned below. In these tasks, the parameter which controls the plotting is `STFACTOR`,

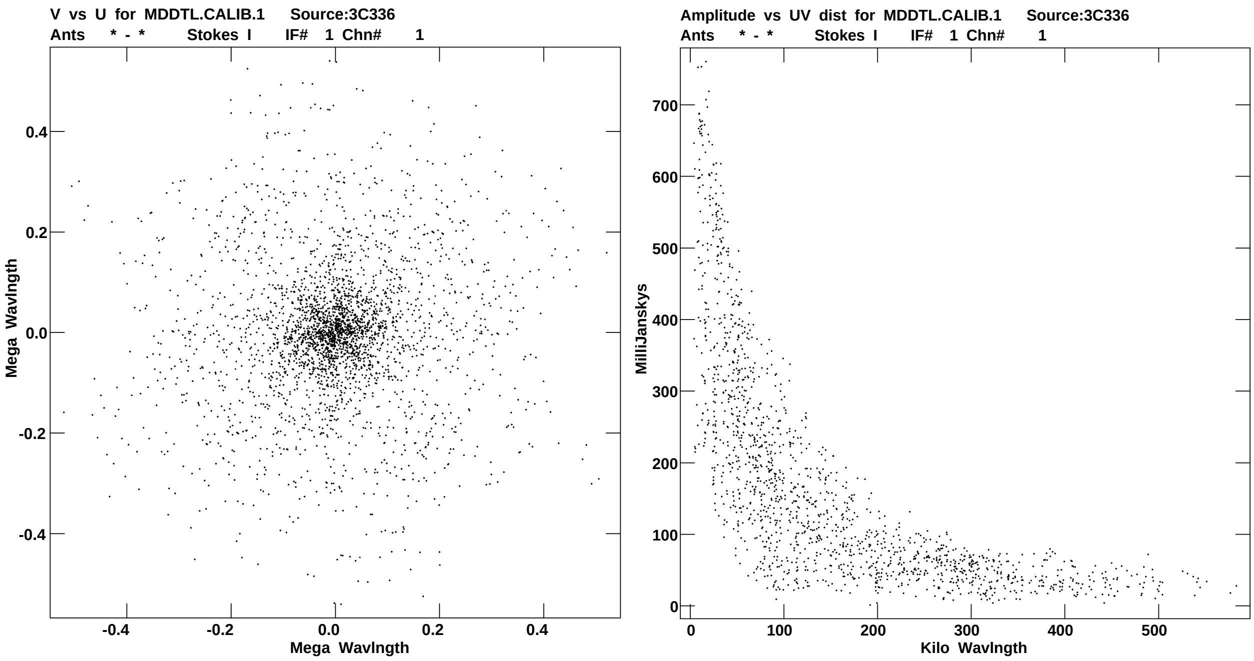


Figure 6.1: `UVPLT` displays of where the data were observed in the *uv* plane (left) and of the visibility amplitudes as a function of baseline length (right). The data on 3C336 were provided by Alan Bridle from observations made with the VLA on 6 December 1987.

a scale factor for the symbols. When using this option, there must be a table of "star" positions associated with the image being plotted. To create one, enter `EXPLAIN STARS` C_R to learn the format of the input data file and the parameters for the task. See also Appendix Z or your local equivalent for instructions on editing text files. A star file may also be created by `MF2ST` from a model fit file produced by task `SAD` (see § 7.5.5 and § 10.4.4).

All plots are drawn with labeled tick marks although these may be suppressed with the `LTYPE` parameter. For plots having significantly non-linear coordinate axes, *e.g.*, wide-field images, it is useful to draw a full, non-linear coordinate grid rather than just short lines at the edges of the plot. Tasks like `CNTR` and even the verb `TVLABEL` offer this option; enter `DOCIRC TRUE` C_R.

Example outputs of the following three tasks are given in Figure 6.3.2.2.

6.3.2.1 Contour and grey-scale plots

The most basic contouring drawing task is `CNTR`. In addition to the usual image selection parameters, you may specify:

> TASK 'CNTR' ; INP C_R	to tell you what you may specify.
> BLC 250 , 230 C_R	to set the bottom left corner of plot at 250, 230 (in pixels with 1,1 at extreme bottom left of the image).
> TRC 300, 330 C_R	to set the top right corner of plot at 300, 330.
> CLEV 0 ; PLEV 1 C_R	to get contour levels at 1% of the peak image value.
> PLEV 0 ; CLEV .003 C_R	to get contour levels at 3 mJy.

> LEVS -1, 1, 2, 4, 6 CR — 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 § 6.4.4). Then check:

> INP CR — to review what you have specified.

> GO CR — to run the task when you're satisfied with the inputs.

This 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 records used" in the plot file is over 200, the contour plot will be messy (unless the field is also large). In this case, check that you have not inadvertently set `PLEV` or `CLEV`, for example, to unrealistically low values. Printing a large, messy plot file on the printer can take a considerable length of time and will inconvenience other users. Consider plotting directly on the TV first (`DOTV = TRUE`) to check on your selection of contours.

`PCNTR` plots polarization vectors on top of contours. You may make a polarized-intensity image and a polarization position-angle image from the Q and U images (see § 7.1.2) or use the Q and U images themselves. Then:

> TASK 'PCNTR' ; INP CR — to review the input parameters.

> INDI *n1* ; GETN *ctn1* CR — where *n1* and *ctn1* select the disk and catalog numbers of the image to be contoured.

> IN2DI *n2* ; GET2N *ctn2* CR — where *n2* and *ctn2* select the Q or polarized intensity image.

> IN3DI *n3* ; GET3N *ctn3* CR — where *n3* and *ctn3* select the U or position-angle image.

> PCUT *nn* CR — to blank out vectors less than *nn* in the units of polarized intensity.

> ICUT *mm* CR — to blank out vectors at pixels where the total intensity (image 1) is less than *mm* in the units of image 1.

> FACTOR *xx* CR — to set the length of a vector of 1 (in units of total polarization) to *xx* cell widths.

> DOCONT 1 ; DOVECT 1 CR — to request both contours and vectors.

> INP CR — to review your inputs and remind you of others. Most are similar to those in `CNTR` and sensibly defaulted.

> GO CR — to generate the plot file, which can then be routed to output devices via `TKPL`, `TVPL`, `LWPLA` etc.

The lengths of the vectors are controlled by image 2 while the directions of them are controlled by image 3. Clearly this program can also be used for other combinations of images, so long as one of them represents an angle.

`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 CR — to review the inputs.

> INDISK *n1* ; GETN *ctn1* CR — to select the grey-scale image.

> BLC 250 , 250 , 3 CR — to select the lower left corner and the plane in the first image.

> TRC 320 , 310 , 12 CR — to select the upper right corner in the first image and, with `TRC(3)`, the plane in the second image.

> DOCONT TRUE CR — to specify that contours are to be drawn.

> IN2D *n2* ; GET2N *ctn2* CR — to select the contour image.

> PLEV 0 ; CLEV 0.005 CR — to select 5 mJy/beam contour increments.

> LEVS -3 , -1 , 1 3 10 30 100 C_R	to plot contours at -15, -5, 5, 15, 50, 150, and 500 mJy/beam.
> DOWEDGE 2 C_R	to plot a grey-scale step-wedge along the right-hand edge; 1 for along the top and 0 for no wedge.
> DOCRT FALSE ; GO C_R	to make the plot file.

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

There are two other contour drawing tasks which offer additional options. `KNTR` is able to draw multiple contour and/or grey-scale images in a single plot file, primarily to show multiple planes of a spectral-line cube; see § 8.5.4. It also has the option to draw an image of the Clean beam. `CCNTR` is virtually identical to `CNTR` except that it can draw extra symbols on the plot representing the locations and intensities of source model components found in `CC` (Clean component) or `MF` (model fit Gaussians from `SAD`, see § 7.5.5 and § 10.4.4).

6.3.2.2 Row tracing plots

There are a number of tasks which plot rows directly. Two of these are for use with single image planes while others are more intended for use with, *e.g.*, spectral-line data cubes transposed into velocity-ra-dec order. Of the former, `PLROW` is the simpler. It 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 C_R	to review the inputs.
> INDISK *n* ; GETN *ctn* C_R	to select the image on disk *n* catalog slot *ctn*.
> BLC 100 ; TRC 300 C_R	to select the subimage from (100, 100) to (300, 300).
> YINC 3 C_R	to plot only every 3rd row.
> PIXRANGE -0.001 0.050 C_R	to clip intensities outside the range −1 to 50 mJy.
> OFFSET 0.002 C_R	to set the intensity scaling such that 2 mJy separates rows of equal intensity.
> INP C_R	to check the inputs.
> GO C_R	to run `PLROW`.
> GO LWPLA C_R	to display the plot file on the laser printer after `PLROW` has finished.

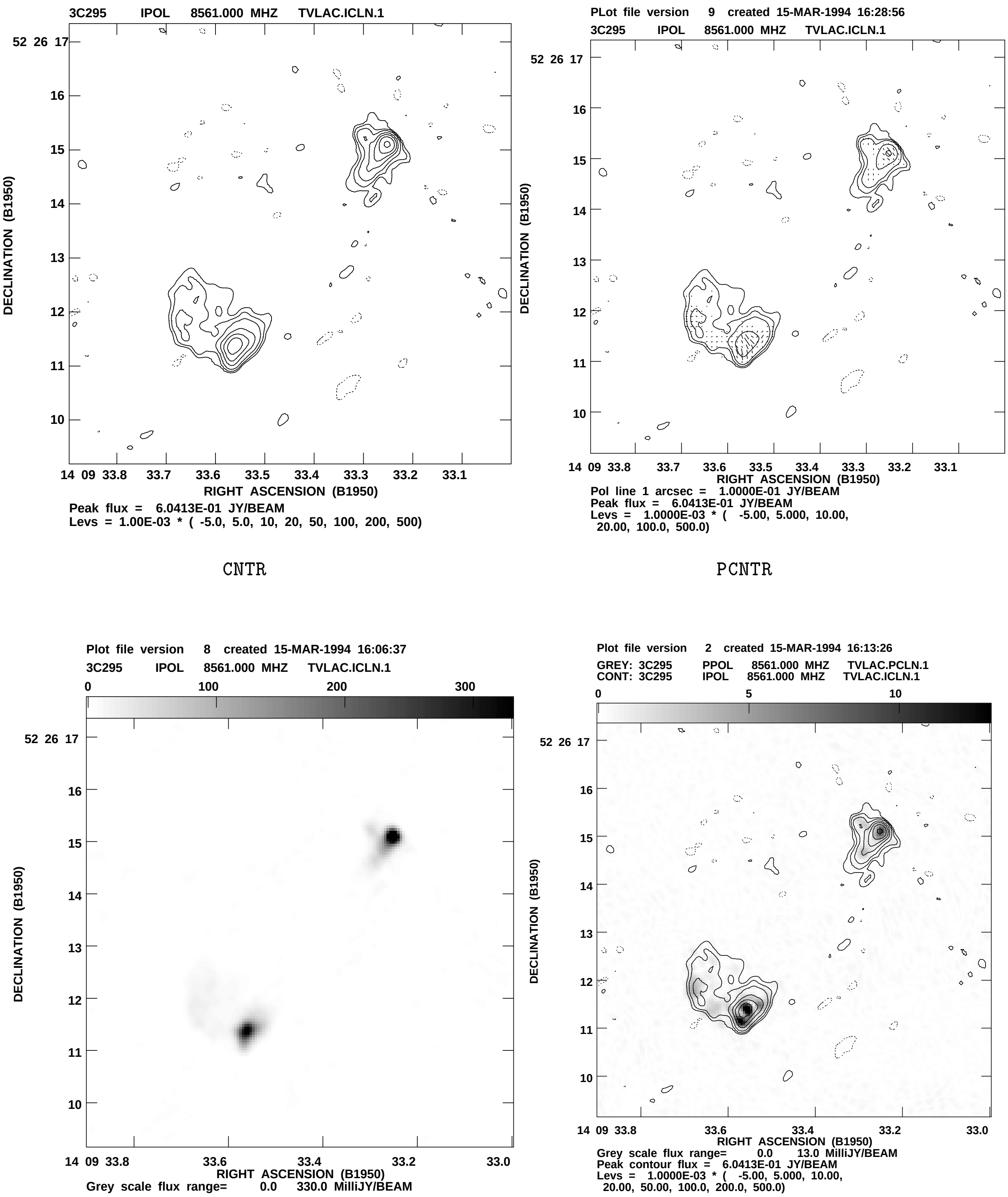
The plot files produced by `PLROW` are a simple, special case of those produced by `PROFL`. This task makes a plot file of a "wire-mesh" representation of an image plane complete with user-controlled viewing angles and correct perspective. Enter EXPLAIN PROFL C_R for a full description. Both of these tasks are especially useful where the signal-to-noise ratio is high and examples of them are given in Figure 6.3.3.

In Chapter 8 we discuss the computation and use of "slices," one-dimensional profiles interpolated along any line in an image plane. Once a slice has been computed, it may be plotted by `SL2PL` on the TV or into a device-independent plot file.

Two other row-plotting tasks, `PLCUB` and `ISPEC`, are designed primarily for spectral-line and other data "cubes" (see § 8.5.4 and § 8.6). `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. `ISPEC` averages rectangular areas in each plane of a cube and plots the resulting spectrum.

6.3.2.3 Miscellaneous image plots

`IMVIM` allows a variety of image comparisons by plotting the pixel values of one image against the pixel values



CNTR PCNTR

GREYS GREYS with contours

Figure 6.2: Contour, polarization, and grey-scale plots of an image

of another image. The special options include binning the values (and plotting symbols proportional to the number of samples in a bin) and shifting one of the images in x and/or y with respect to the other. The former reduces large scatter diagrams to more manageable sets of numbers while the latter allows cross-correlation functions to be developed.

IMEAN prints the mean, rms, and extrema over a user-specified window in an image. Beginning with the 15JAN95 release, it also prints the intensity at, and rms of, the noise peak in the histogram. Optionally, it plots histograms of image intensities over the user-specified window using a user-specified number of boxes over a user-specified range of intensities. An example of this is also shown in Figure 6.3.3.

6.3.3 Plotting your table data

TAPLT is a very general task to plot information from *AIPS* table extension files. It can plot a histogram of a function of the values in one or two columns of the table and it can plot a function of one or two columns against another function of another one or two columns. The latter can be averaged in bins or have every point plotted. At first blush, the inputs seem rather complicated, but the results may well justify some effort to understand them. For example, to plot Clean component fluxes as a function of radius from the image center:

> TASK 'TAPLT' ; INP C_R	to review the inputs.
> INDISK *n* ; GETN *ctn* C_R	to select the image on disk *n* catalog slot *ctn*.
> INEXT 'CC' ; XINC 1 C_R	to select every row in the Clean components file.
> BCOUNT 1 ; ECOUNT 0 C_R	to select all rows in the table.
> APARM = 2 , 1, 3 , 1 , 16 , 1 , 1 C_R	to plot the modulus of columns 2 and 3 on the x axis (*i.e.*, $\sqrt{\Delta_x^2 + \Delta_y^2}$) and column 1 on the y axis (*i.e.*, component flux).
> BPARM = 0 ; CPARM = 0 C_R	to use self-scaling of the plot and no scaling of the column values.
> DOTV TRUE ; GO	to plot the fluxes on the TV screen.

You may need to set the scaling with **BPARM** after seeing the preview plot in order, for example, to bring out the details in the low-level components.

There are a few tasks intended to make plotting specific kinds of tables rather easier. **SNPLT** plots calibration, solution, and system temperature tables for selected antennas as a function of time, antenna elevation, hour angle, or sidereal time. It will make several plots per page (one antenna per plot) and multiple pages if needed, or you can plot the most discrepant value over all antennas in a single plot. **POSSM** will plot bandpass tables when **APARM(8)** is set to 2. It can also do multiple plots per page with all the usual data selection adverbs. **GAPLT** plots the gain tables produced by **ASCAL**, the old self-calibration task. It too can do multiple plots per page and the like. **GNPLT** plots the extrema of the solutions from all antennas at each time, again as found by **ASCAL**.

6.3.4 Plotting miscellaneous information

There are a few other tasks which create plot files, but which do not fit into the categories above. **CONPL** is a task created for the 15JAN95 release which plots *AIPS* convolving functions (used by various *uv*-data gridding tasks such as **IMAGR**) and their Fourier transform, expected signal-to-noise ratio, or convolution with a user-specified Gaussian. **IRING** integrates an image in concentric annulli about the user-specified object center with specified major axis position angle and inclination. The results may be placed in a plot file for later display. **GAL** calculates the orientation and rotation curve parameters of a galaxy from an image of the predominant velocities. The observed rotation curve is plotted together with the fitted model curve.

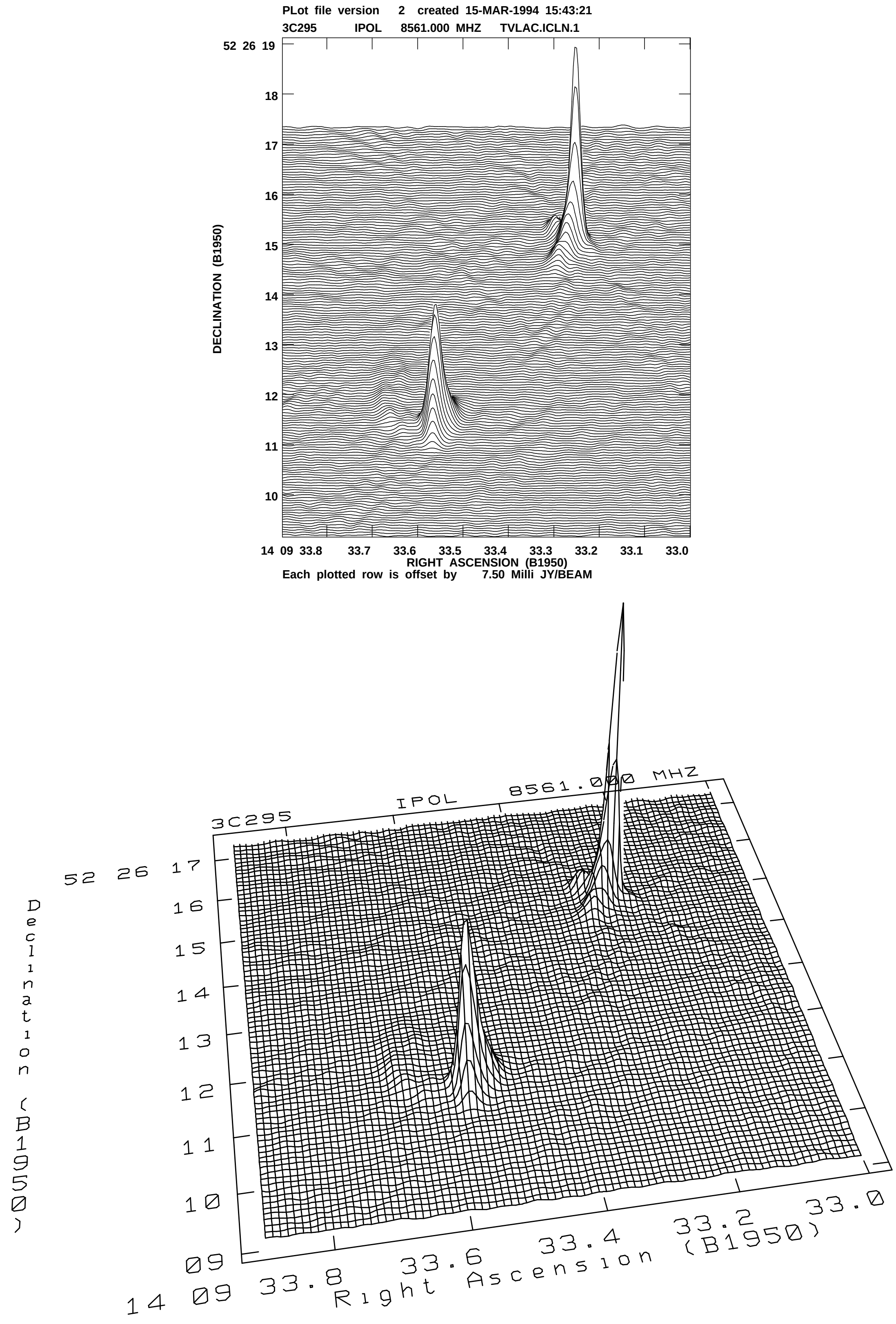
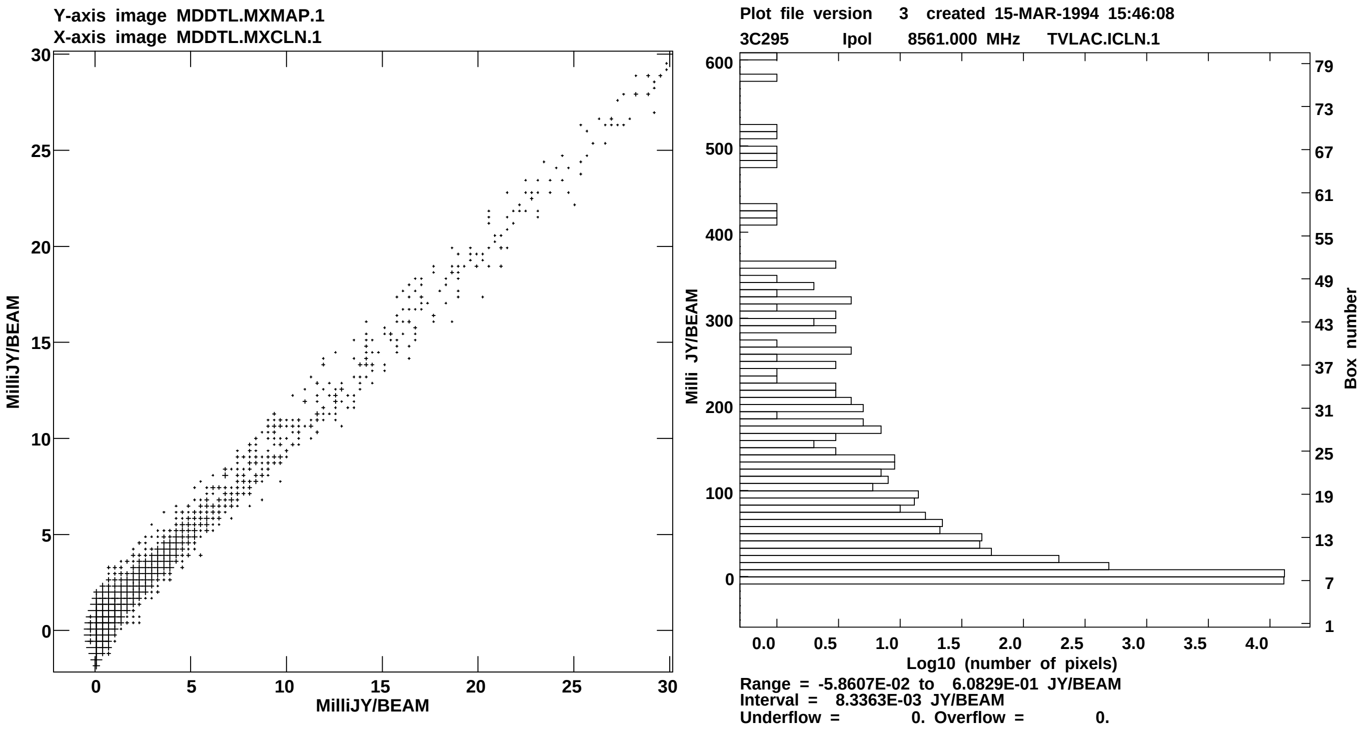
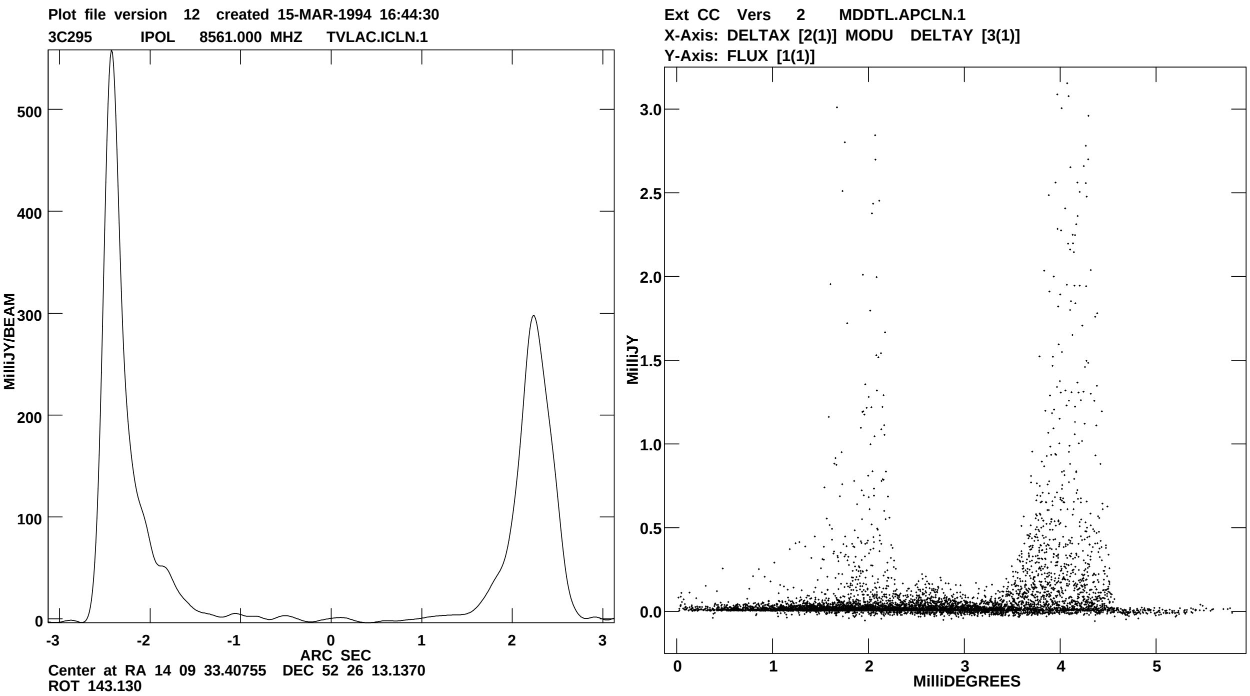


Figure 6.3: Row plots of an image with (bottom, PROFL) and without (top, PLROW) perspective



`IMVIM` (binned) `IMEAN`

Figure 6.4: Plots of statistical parameters of an image.



`SL2PL` `TAPLT`

Figure 6.5: Slice and table plots.

6.4 Interactive TV displays of your data

The *AIPS* TV display allows you to look at your image data in detail and to set parameters by pointing at interesting feature visible on the screen. Although *AIPS* will run on a variety of hardware display devices (*e.g.*, I^2S Models 70, 75 and IVAS and DeAnza), it is now used almost exclusively with the X-Windows TV-simulation program called `XAS`. See § 2.3.2 for information on using this basic *AIPS* tool. Information on TV verbs and tasks may be found by entering ABOUT TV C_R, ABOUT INTERACT, and ABOUT TV-APPL C_R, or by consulting the corresponding sections of Chapter 13 of this *CookBook*.

6.4.1 Loading an image to the TV

The simplest way to load an image from your catalog to the TV and then to manipulate the display is with the procedure called `TVALL`:

> INP TVALL C_R	to review the input parameters.
> INDI *n* ; GETN *ctn* C_R	to select the disk and image name parameters from the catalog.

Use one of the following commands to specify the initial transfer function that converts your image file intensities to display pixel intensities:

> FUNC 'LN' C_R	linear—this is the default.
> FUNC 'LG' C_R	logarithmic.
> FUNC 'NE' C_R	negative linear.
> FUNC 'NG' C_R	inverse logarithmic.
> PIXRA *x1, x2* C_R	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.

(The slopes and intercepts of the display transfer functions can be modified later, but the above options let you choose initially between linear, logarithmic, and negative linear displays and restrict the range of intensities that is loaded). Then:

> TVALL C_R	to load the selected image.

The image should appear on the TV screen in black-and-white. If you see a new image but it is not what you expected, hit button `D` to end `TVALL` and then review your inputs with:

> INP TVALL C_R

If no image appears, make sure that `AIPS` started up with your workstation assigned as the display; see § 2.2.3 for information on starting up `AIPS` and assigning the TV display.

After your image has been displayed on the TV by `TVALL`, your trackball with its buttons (which should be labeled `A`, `B`, `C`, and `D`) — or your workstation mouse with keyboard buttons `A`, `B`, `C`, and `D`) — can be used to modify the display transfer functions, coloring and zoom. Pressing button `A` alone enables black-and-white and color-contour coding of the image intensities, successively. Adjust the cursor position on the TV (using the trackball or mouse) to 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. Black-and-white displays are generally much more suitable than color for high-dynamic-range images, while color contouring may be used to accentuate interesting features. Note also that a much wider range of image-coloring options is available outside `TVALL` by invoking `TVPSEUDO` and `TVPHLAME`. 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 through 16 are available on most workstations. 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.

The size of the image which can be displayed depends on the size (in pixels) of your workstation screen or

TV display and some other parameters which can vary from site to site. A typical Sun workstation can display up to 1024 by 900 pixels. Then, if the image is larger than about 800 pixels or so 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). 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. A small image may be interpolated to fill the TV screen by setting TXINC = -1 ; TYINC = -1 C_R.

TVALL is a procedure that insures that the desired graphics and TV channels are on and cleared and the others off, then loads the image with verb **TVLOD**, loads a wedge with **TVWEDGE**, and labels the wedge with **TVWLABEL**. You do not have to use **TVALL**, which can be rather slow, simply to load a new image on the TV. Use **TVLOD** instead. **TVWEDGE** has a variety of options concerning the width of the wedge and its position; type HELP TVWED C_R for details.

6.4.2 Manipulating the TV display

There are a number of verbs which allow you to manipulate the display, including:

> TVINIT C_R	to initialize the entire TV and TV image catalog. This should be done when you first start using a workstation display since the catalog will remember things that are no longer there and since the current **XAS** can have different parameters than a previous one.
> TVON n C_R	to turn on TV grey channel n; you may have to turn off other channels to see n since **XAS** can only display one channel at a time.
> TVOFF m C_R	to turn off channel m, where m is "decimal coded," meaning that $m = 12$ refers to both channel 1 and channel 2.
> GRON n C_R	to turn on one or more of the 4 graphics channels, where n is decimal coded.
> GROFF m C_R	to turn off one or more graphics channels.
> TVCHAN n ; TVCLEAR C_R	to zero one or more TV channels.
> GRCHAN m ; GRCLEAR C_R	to zero one or more graphics channels.
> TVZOOM C_R	to set the zoom magnification and center interactively, follow instructions on the screen.
> OFFZOOM C_R	to reset the zoom and zoom center to null.
> TVPOS C_R	to read the TV cursor position, returning adverbs for use in procedures or other verbs.

6.4.3 Intensity and color transfer functions

The *AIPS* model of a TV postulates two intensity transfer functions, called the LUT and the OFM, which are basically multiplicative. In most circumstances, the LUT is used for black-and-white enhancements and the OFM for coloring, but both can be used for either. To manipulate the LUT interactively, while leaving the pseudo-coloring alone, use the `TVTRAN` verb. The cursor position controls the slope and intercept of the transfer function, buttons `A` and `B` switch a plot of the transfer function on and off, button `C` switches the sign of the slope, and button `D` (as always) exits. To turn the LUT back to normal, enter `OFFTRAN`.

A rich zoo of color coding is available with `TVPSEUDO`, which alters the OFM while leaving the LUT alone. Repeated hits on button `A` select a variety of color triangles, button `B` selects a circle on hue, and repeated hits on button `C` select a variety of color contours. First-time users should experiment with the *AIPS* coloring options until they develop an intuitive feel for the effects of cursor settings on the image appearance. The wedge displayed by `TVALL` adjusts to the alternative colorations selected with `TVPSEUDO`, and it is helpful to watch changes in both the wedge and the image. A flame-like coloring is available with `TVPHLAME`, or variations on the scheme with repeated hits on buttons `A` or `B`. In both these verbs the cursor position controls aspects of the coloring such as enhancements, richness, or cycles of hue. To turn off pseudo-coloring, enter `OFFPSEUD`.

A set of less well-known verbs is available to allow you to create, manipulate, and save desirable versions of the OFM table. `OFMSAVE` allows you to save a named OFM, `OFMDIR` lists all saved OFMs belonging to you or generally available from the *AIPS* distribution, `OFMGET` loads a name OFM, `OFMZAP` deletes a named OFM, and `OFMLIST` prints the current OFM. `OFMCONT` is an elaborate interactive verb which allows you to set the hue, intensity, and saturation of the OFM divided up into a number of color contours. Each of these contours can be a constant level or a step wedge. `OFMADJUS` is another elaborate interactive verb to alter pieces of the OFM, while `OFMTWEAK` is a simpler verb to stretch the OFM.

6.4.4 Setting parameters with the TV

One reason to load the image to the TV is to set adverbs for use by other verbs and tasks. Verbs which use the TV cursor to set adverbs include:

> TVNAME C_R	to set `INDISK`, `INNAME`, etc. to the name parameters of the image currently visible. If there is an ambiguity, you will be asked to move the cursor to the desired image and press a button.
> TVWIN C_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) C_R	reads pixel coordinate of the center of a dx-pixel by dy-pixel window and sets the adverbs `BLC` and `TRC`.
> TVBOX C_R	is similar to `TVWIN` above except that it is used to set up pixel coordinates to define rectangular *or circular* Cleaning areas for the *AIPS* Clean tasks. The adverbs `NBOXES` and `CLBOX` are set. The circular option appeared in the `15JUL95` release and is not supported by `APCLN` and `MX`.
> REBOX C_R	allows revision using the TV of the Cleaning areas set previously with `TVBOX`. Revises `NBOXES` too.
> FILEBOX C_R	is `REBOX` for boxes in the text file used with `IMAGR`; multiple fields and many more boxes are allowed.

> TVSLICE CR	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.

6.4.5 Reading image values from the TV

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

> IMPOS CR	reads out the two coordinate values (*e.g.*, RA and Dec) from the cursor position when any button is depressed.
> IMXY ; IMVAL CR	reads out the image intensity and the two coordinate values (*e.g.*, RA and Dec) from the cursor position and when any button is depressed.
> TVFLUX CR	reads out image intensities and coordinates whenever a TV button is pressed, looping until button D is pressed.
> TVMAXFIT CR	whenever a TV button is pressed, fits a quadratic function to the image to find the position and strength of an extremum, looping until button D is pressed.
> CURVAL CR	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	determines the mean, rms, extrema and integrated intensity (if appropriate) in user-defined "blotch" regions within the image currently displayed on the TV. The regions are irregular polygons selected with the TV cursor. Type EXPLAIN TVSTAT CR for details.

6.4.6 Labeling images on the TV

There are a number of facilities for labeling images on the TV including:

> TVLABEL CR	to draw standard axis labels around the visible image. You may control the type of labeling and whether coordinates are shown as a grid or short tick marks. If more than one image is visible, you will be asked to indicate which one you want with the cursor and any button.
> TVWLABEL CR	to draw axis labels around the visible intensity wedge.
> TVANOT CR	to draw a text string into a grey-scale channel or a graphics plane at a location specified via an adverb or via the TV cursor.
> TVLINE CR	to draw a straight line into a grey-scale channel or a graphics plane at locations specified in part or in whole via adverbs or via the TV cursor.
> COSTAR CR	to plot a "star" positions at a user-specified coordinate on the TV image.
> TVSTAR CR	to plot "star" positions from an ST file on top of the visible image; see § 6.3.2.
> GREAD CR	to read the current color of a specified graphics overlay channel into RGBCOLOR.

> GWRITE C_R	to change the color of a graphics overlay channel to that specified by RGBCOLOR. This may be done for aesthetic reasons or because the default colors may not show up well when captured by TVCPS and printed on a color printer or film recorder.

6.4.7 Comparing images on the TV

It is often useful to compare two images, *e.g.*, to decide whether one contains artifacts that are not present in another at the same frequency, or to look for frequency-dependent features at constant resolution. *AIPS* provides several tools for such image comparisons.

The first tool is a capability for loading multiple images to the same plane (or channel) of the TV device. The parameter TVCORN specifies where the bottom left corner of the image or image subsection will be positioned in the TV frame by TVLOD or TVALL. If TVCORN is left at zero, TVLOD and TVALL adjust it to center the displayed image. You may however use TVCORN to control loading successive images to different regions of the display with successive executions of TVLOD. For example, the following commands would load two 512 by 512 pixel images from slots 1 and 2 on disk 1 *side-by-side* on channel 1 of a 1024 by 900 TV display:

> INDI 1; GETN 1 C_R	to select the input disk and the first image.
> TVCH 1 C_R	to select TV channel 1 for the loading.
> TVCORN 1 193 ; TVLOD C_R	to load the first image.
> GETN 2 C_R	to select the second image.
> TVCORN 513 193; TVLOD C_R	to load the second beside the first.

You could then adjust the color coding, transfer function, etc. for both images simultaneously with TVFIDDLE or TVTRAN. You may load as many as 256 images to a single TV plane with this technique, which is therefore a powerful method for making "montages." The number of simultaneous images is limited mostly by your image sizes, the need to avoid overlaps (which are allowed if you want) — and your ability to do the arithmetic for appropriate TVCORN settings! You are also limited by the need for all the images in one plane to share that plane's transfer function. Judicious use of the PIXRANGE and FUNC inputs to TVLOD permits making useful montages of disparate images, however.

A second tool is the classic "blink" technique from optical astronomy. TVBLINK allows you to load images to two different planes of the TV memory and then to alternate the display rapidly between the two. The two images described above could be"blinked" against each other by the following command sequence:

> INDI 1; GETN 1 C_R	to select the input disk and first image.
> TVINIT; TVCORN 0	to clear the TV and restore the default positioning.
> TVCH 1; TVLOD C_R	to load the first image on plane 1.
> GETN 2 C_R	to select the second image.
> TVCH 2 ; TVLOD C_R	to load the second image on plane 2.
> TVCH 12 ; TVBLINK C_R	to blink planes 1 and 2.

The rate and duty cycle of the blinking, and the transfer functions applied to the planes, are controlled interactively with the TV cursor. Instructions for these operations appear on your terminal while TVBLINK is running.

In 15JUL95 and later releases, the task PLAYR provides a menu-driven method to enhance and blink two images and to develop and save TV color tables ("OFMs").

For data cubes (*e.g.*, frequency or time sequences of images), the verbs TVMOVIE and TVCUBE combine the two previous techniques. These are described in more detail in § 8.5.4. Both verbs load one or more image planes with as many planes from the cube as possible (and as requested). Then they display each frame in sequence with interactive controls over the frame rate in movie mode, the chosen frame in single-frame mode, and the brightness, contrast, and color of the displayed images. TVMOVIE makes a somewhat more efficient movie sequence, but TVCUBE makes a better montage by using a more normal arrangement of the image planes.

Certain real TV displays used to provide powerful tools to compare images using color as well as intensity to represent real information. Unfortunately, most workstations can display only 256 simultaneous colors (or even fewer), and XAS was constructed with that limit in mind. There are two tasks, also discussed in § 8.6, which attempt to recover much of this capability by trying to optimize color assignments over the limited range available. The first of these, TVHUI produces a composite display in which the intensity is set by one image, the hue is derived from another image, and the saturation is optionally derived from a third image. An interactive menu allows you to enhance each of the images individually, to select linear or logarithmic transfer functions for the intensity image, to select the subimage used during interactive enhancements, to repaint the full image, and to exit with or without writing out the final three-color image. A number of uses for this are obvious, including spectral-line moment images (velocity setting color, line width setting saturation), polarization images (polarization angle setting color in a circular scheme, polarization intensity setting saturation), and depolarization observations (color set by a two-frequency depolarization image).

TVHUI can be instructed to write out a three-color image cube containing one plane for red, one for green and one for blue. It does this using the transfer functions established interactively, but with full accuracy unlimited by the TV display. There is also an AIPS task called TVRGB which writes three-color cubes using weighted sums over a data cube. Three-color cubes also arise when digitizing color photographs of real scenes. The second task, TVRGB, can be used to display these three-color cubes or to generate a three-color display from any three AIPS image planes. Common examples of the latter are the superposition of radio continuum and/or line data on optical or X-ray images, and color-coding of effective temperatures or spectral indices from 3-channel continuum data. TVRGB can also be used to color-code different types of depolarization effects from multi-frequency polarimetry. Like TVHUI, TVRGB offers a simple menu to enhance each of the images individually or all together, to select the window specifying the subimage which is used during interactive enhancements, to repaint the full image on the TV, and to exit. TVRGB does not write an output image per se, but, as of the 15JAN96 release, it can be instructed to write out a full 24-bit color PostScript plot file to be sent to a color printer or film recorder. Its display (or any other TV display including that of TVHUI) can be captured and sent to a color printer or film recorder; see § 6.4.9 below.

6.4.8 Other functions using the TV

There are a number of tasks which use the TV to give the user real interactive input to the operation based on the images displayed by the task on the TV. These include BLANK (§ 8.6) to blank out non-signal portions of an image, BLSUM (§ 8.6) to sum images over irregular blotch regions printing out summed spectra, TVFLG (§ 4.4.3) to edit visibility data based on grey-scale displays of some function of the visibility with baseline on the x axis and time on the y axis, SPFLG (§ 8.1, § 10.2.2) to edit visibility data based on grey-scale displays of some function of the visibility with spectral channel on the x axis and time on the y axis, EDITR (§ 5.5.2) to edit visibility data based on plots of visibility versus time, 1–11 baselines at a time, EDITA (§ 4.4.2) to edit visibility data based on plots of system temperature (TY tables) or antenna gains (SN or CL tables), and SNEDT to edit SN and CL tables.

In 15JUL95 and later releases, the imaging tasks, IMAGR and SCMAP display the results of the computation at its current stage on the TV and provide a menu of interactive options to the user. The menu includes the usual display enhancements, the ability to choose among the images being computed, the ability to set Clean windows in those images, and the ability to end the computation at its current stage. The computation will resume when instructed via the menu or after a period of inactivity. A number of older iterative tasks use the TV to display the results of the computation so far and then prompt the user to hit button D within some number of seconds to stop the computation. Tasks that do this include APCLN, MX, SDCLN, VTESS, and several less significant tasks. UVMAP uses the TV simply to draw a picture indicating which cells are sampled in the *uv* plane. All of these tasks are described in Chapter 5.

6.4.9 Capturing the TV

Having done all the work to prepare the absolutely perfect display on your TV screen, it would be a good idea to capture it before someone, such as the local power company, does a TVINIT. See § Z.1.2.2 for a discussion of Unix tools to do this. We recommend, however, TVCPS to capture the image on your TV, optionally including graphics overlay channels whether or not they are on, and to write the result to an encapsulated PostScript file. This file can be printed immediately on a black-and-white or color printer or on any other device which understands PostScript, such as NRAO's film recorders. It can also be saved for later printing or inclusion in other documents. (See § Z.1.3 for a method by which *AIPS* users may have their pictures recorded on film for them using the Charlottesville film recorder.) TVCPS was used to make the picture on the title page of this *CookBook*, the picture of TVFLG's display in Chapter 4, and the picture of a right ascension - velocity - declination data cube in Chapter 9. TVCPS bases its picture on the current size of your TV display. If you are using a workstation with XAS, be sure to adjust the size of the display window to encompass all of your image plus a modest border. If you leave a large border, you will get a large border in your output. Your TVCPS session could look like:

> TASK 'TVCPS' ; INP C_R	to review the inputs.
> OUTFILE 'MYAREA:TV.PIX C_R	to save the output in a file called TV.PIX in an area defined by the logical MYAREA; see § 3.10.1.
> GRCHAN 1234 C_R	to include all graphics channels.
> OPCODE 'COL' C_R	to make a color picture.
> APARM 7.5 , 11 C_R	to set the output device size to 7.5 by 11 inches, appropriate to the film recorder.
> APARM(8) 1 C_R	to suppress labeling by user number.
> GO C_R	to run the task when the inputs are set.

TVCPS has an option to add a character string below the image with adverb REASON. If your image is too large to fit on the TV, you can instruct TVCPS to read the image from disk with DOTV = -2 C_R, using adverbs TBLC, TTRC, TXINC, and TYINC. When doing this, you should turn off the graphics display (GRCHAN 0 C_R) since it is not aligned properly. Some color printers and recorders have rather different transfer characteristics than the workstation screen. TVCPS offers the option to remove the "gamma correction" used for your workstation and to apply a different one appropriate to your color recorder. To see the effects of the gamma correction, try the verb GAMMASET in AIPS. You may need to use GWRITE to select better colors for the graphics overlay planes as well.

6.5 Graphics displays of your data

In the dim dark past, Tektronix invented some nice graphical display devices and an inconvenient but functional way to talk to them. This communication language became so imbedded in software that workstation vendors now provide X-Windows windows that understand it. *AIPS* also arose from this dim dark past and once upon a time talked to those lovely green screens. To retain the graphics capability, we now provide a TEKSRV server which will provide a Tektronix-like graphics screen on which certain *AIPS* tasks and verbs plot. When AIPS starts up on workstations, it brings up a window called TEKSRV. Leave this window in its iconic state; it is only a marker for the presence of the server. The first time you write to TEKSRV, it will create and open a window called TEKSRV (Tek) in which the plot is done. You can resize this window within some limits and the plot will automatically resize itself. When the workstation cursor is in the Tek window, it changes to a diagonal arrow pointer. When an *AIPS* task or verb tries to read from the Tek window, this pointer becomes a plus sign. You should position the pointer to the desired location *without* touching the keyboard or the mouse buttons. When the pointer is exactly where you want it, press any mouse button or any key (*except* `RETURN`) to return the pointer position to the program. Note that the functions using the graphics display are not quite as friendly as those on the TV. This is due to the inability to erase a piece of a plot without erasing all of it.

6.5.1 Plotting data and setting values with the graphics display

`TKPL` interprets *AIPS* plot files to the graphics window or device. Experienced *AIPS* users like it because it is much faster than the TV for complicated line drawings, *e.g.*, those produced by `UVPLT`, and because it is of higher resolution than many of the TV plots. The graphics screen can be used to read back data values and set adverbs, much like the TV:

> TKXY C_R	to read the graphics cursor position, setting adverb `PIXXY`; requires a contour or comparable image to be shown on the screen.
> TKXY ; IMVAL C_R	to read the graphics cursor position and return the image value and coordinates of the selected position.
> TKPOS C_R	to read the graphics cursor position and return the image coordinates of the selected position.
> TKWIN C_R	to read the graphics cursor position twice, first setting `BLC` and then setting `TRC`.
> TKBOX(i) C_R	to read the graphics cursor position twice, first setting the lower left and then the upper right of Clean box i.
> TKNBOXS(n) C_R	to set `NBOXES` to n and then set all n Clean boxes using the graphics cursor.

All of these verbs require a graphics display of a plot file produced by `CNTR`, `PCNTR`, `GREYS`, or `SL2PL`. The window procedures don't make much sense with a slice plot, but they will work.

6.5.2 Slice files and the graphics display

In Chapter 8 we discuss the computation and use of "slices," one-dimensional profiles interpolated along any line in an image plane. Once a slice has been computed, it may be plotted by `TKSLICE` on the graphics display. A second slice may be plotted on top of the first with `TKASLICE`. The graphics display is used to prepare initial guesses for `SLFIT`, which fits Gaussians to slices. The verbs involved are:

> TKVAL C_R	to return the flux level pointed at by the graphics cursor.
> NGAUS n ; TKSET C_R	to set the number of Gaussians to be fitted to n and then to prepare an initial guess at the parameters by pointing at the peaks and half width points on a graphics plot of the slice.
> TK1SET j C_R	to revise the initial guess for the $j^{\rm th}$ Gaussian.
> TKGUESS C_R	to plot the initial guess of the model on the graphics device, erasing any previous plot.
> TKAGUESS C_R	to add a plot of the initial guess of the model to the current slice plot on the graphics device.
> TKMODEL C_R	to plot the fit model on the graphics device, erasing any previous plot.
> TKAMODEL C_R	to add a plot of the fit model to the current slice plot on the graphics device.

> TKRESID C~R~ — to plot the data minus the fit model on the graphics device, erasing any previous plot.

> TKARESID C~R~ — to add a plot of the residuals (data minus model) to the current slice plot on the graphics device.

The units for the slice model parameters are fairly problematic, so we recommend using these graphical input and output functions. At least, they all have the same strange ideas.

6.5.3 Data analysis with the graphics display

There is a set of related tasks for analysis of data cubes transposed so that the first axis is the one on which baselines or Gaussians are to be fit. In the case of a spectral-line cube, the image would be transposed so that velocity is the first axis. It is a good idea to use `XPLOT` first to get an idea of what the profiles really look like. It uses a flux cutoff to determine which profiles to display and prompts you for permission to continue after each plot. `XBASL` is used to remove $n^{\rm th}$-order polynomial baselines from each spectrum. It has a batch mode of operation and an interactive mode which uses the graphics display to plot each spectrum and to accept guidance on which channels to use in determining the baselines. `XGAUS` is a similar task, with the rather harder job of fitting up to four Gaussians plus a linear baseline to each profile. In its interactive mode it plots each selected spectrum on the graphics device and accepts guidance on the baseline regions and the initial guesses for the Gaussians. `XGAUSS` writes images of the fit Gaussian parameters and `XBASL` can be asked to write images of the baseline parameters. Unfortunately, these tasks require you to do the full cube in a single execution, which is rather an endurance contest.

6.6 Additional recipe

6.6.1 Delightful banana cheesecake

1. Preheat oven to 350 deg.
2. Combine 1.5 cups crushed `cereal` (3 cups un-crushed Multi-Bran Chex suggested), 1/3 cup melted `margarine` or butter, and 1/4 cup packed `brown sugar`; mix well.
3. Press firmly onto bottom and sides of greased 9-inch pie plate. Bake 8–10 minutes, then cool completely.
4. Arrange 1.5 cups sliced `bananas` onto sides and bottom of cooled crust.
5. Combine 16 oz. softened light or regular `cream cheese`, 1.5 cups `powdered sugar`, and 3/4 teaspoon `vanilla extract`.
6. Mix well, then fold in 2 cups light or regular `whipped topping`. Pour over sliced bananas.
7. Cover and refrigerate for 4 hours or until set.
8. Garnish with 1/2 cup sliced `bananas`.

Thanks to Ralston Purina Company.