7 Analyzing Images

In order to obtain useful astronomical information from the data, software exists for the analysis of images, combining of images, estimating of errors, etc. Only a few of the programs are described in any detail here; the others should be self-explanatory using the HELP and INPUTS files for the tasks listed in Chapter 13. A complete list of software in $\mathcal{AIPS}$ for the analysis of images may also be obtained at your terminal by typing ABOUT ANALYSIS C_R.

7.1 Combining two 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 subtract a continuum image from a spectral line image cube.

7.1.1 Subtracting a continuum image from an image cube

A common method to obtain a spectral data cube containing only line signal without any continuum emission is to create a line-free continuum image *C*, and subtract it from the data cube *L*. For a more general discussion and alternative methods see § 8.3. COMB can be used to this purpose as follows:

> TASK 'COMB' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to help you find the catalog numbers of *C* and *L*.
> INDI *n1* ; GETN *ctn1* C_R	to select the *L* image cube from disk *n1* catalog slot *ctn1*.
> IN2D *n2* ; GET2N *ctn2* C_R	to select the *C* image from disk *n2* catalog slot *ctn2*.
> OUTN '*xxxxx*' C_R	to specify *xxxxx* for the name of the continuum-free image cube.
> OUTC '*ccc*' C_R	to specify *ccc* for the class of the continuum-free image cube, *e.g.*, LCUBE.
> OPCODE 'SUM' C_R	to select the addition algorithm.
> APARM 1, -1 C_R	to specify that we want $+1 \times L - 1 \times C$.
> GO C_R	to compute the continuum-free, line-only output cube.

Once COMB task has terminated with the message COMB: APPEARS TO END SUCCESSFULLY, you should find the requested image in your catalog:

> MCAT C_R	to list the images in your catalog.

7.1.2 Polarized intensity and position angle images

As a second example, we derive the polarization intensity and angle from the Q and U Stokes parameter images. To compute a polarized intensity image, enter:

> TASK 'COMB' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to find the catalog numbers of the Q and U images.
> INDI *n1* ; GETN *ctn1* C_R	to select the Q image from disk *n1* catalog slot *ctn1*.
> IN2D *n2* ; GET2N *ctn2* C_R	to select the U image from disk *n2* catalog slot *ctn2*.

> OUTN '*xxxxx*' C_R	to specify *xxxxx* for the name of the polarized intensity image.
> OUTC '*ccc*' C_R	to specify *ccc* for the class of the polarized intensity image, *e.g.*, PCLN.
> OPCODE 'POLC' C_R	to select the $\sqrt{\mathrm{Q}^2 + \mathrm{U}^2}$ algorithm with correction for noise.
> BPARM *ns1* , *ns2* C_R	to specify the noise levels of the 2 images.
> GO C_R	to compute the corrected, polarized intensity image.

AIPS will write the message 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*' C_R	to specify *yyyyy* for the name of the polarization angle image.
> OUTC '*ddd*' C_R	to specify *ddd* for the class of the polarization angle image, *e.g.*, PSIMAP, CHICLN.
> OPCODE 'POLA' C_R	to select the $\frac{1}{2}\tan^{-1}\left(\frac{\mathrm{U}}{\mathrm{Q}}\right)$ algorithm.

Once COMB has finished, enter:

> GO C_R	to compute the polarization angle image.

7.1.3 Other image combination options

COMB may also be used to re-scale images, and to compute spectral indices, optical depths, etc. Type:

> HELP COMB C_R	to review the available options.

The OPCODE options are:

'SUM '	Addition	$a_1M_1 + a_2M_2 + a_3$	
'SUMM'	Addition	$a_1M_1 + a_2M_2 + a_3$	except blanked pixels replaced with 0
'MEAN'	Average	$a_1M_1 + a_2M_2$	except M_j where M_i blanked
'MULT'	Multiplication	$a_1M_1M_2 + a_2$	
'DIV '	Division	$a_1M_1/M_2 + a_2$	
'SPIX'	Spectral Index	$a_1\ln(M_1/M_2)/\ \ln(\nu_1/\nu_2) + a_2$	where $M_1 > a_3$ and $M_2 > a_4$
'OPTD'	Opacity	$a_1\ln(a_3M_1/M_2 + a_4) + a_2$	where $M_1 > a_5$ and $M_2 > a_6$
'POLI'	RMS sum	$a_1\sqrt{M_1^2 + M_2^2} + a_2$	
'POLC'	RMS sum	$a_1C(M_1, M_2)\sqrt{M_1^2 + M_2^2} + a_2$	where C is a noise-based correction for Ricean bias
'POLA'	Arctangent	$a_1\tan^{-1}(M_2/M_1) + a_2$	where $\sqrt{M_1^2 + M_2^2} > a_3$
'REAL'	Real part	$a_1M_1\cos(a_2M_2) + a_3$	
'IMAG'	Imaginary part	$a_1M_1\sin(a_2M_2) + a_3$	
'RM '	Rotation measure	$a_1RM(M_1, M_2) + a_2$	
'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 — specified by APARM — and M_1 and M_2 are the images selected by INNAME, *etc.* and by IN2NAME, *etc.*, respectively. COMB may be instructed to write an image of the estimated noise in the combination in addition to the direct result of the combination. These noise images may be used as inputs to COMB and, *e.g.*,BLANK to control later computations. When combining numerous images together, set DOHIST=-2 to prevent history files from becoming much larger than the images.

CONVL can compute the cross-correlation of two images and report the location of the peak cross-correlation.

7.1.4 Considerations in image combination

COMB can use images of the uncertainties in the first two input images to control the computation of the output. The new task RMSD may be used to create an image of the rms in an image, computing the rms self-consistently (robust, histogram, and median absolute deviation methods available) in windows surrounding each pixel. Task FLATN can compute weight and noise images corresponding to mosaiced images.

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 (set BPARM(4) = 0) whenever either M_1 <APARM(9) or M_2 <APARM(10). Alternatively, blanking may be done on the basis of the estimated noise (set BPARM(4) = 1) or signal-to-noise ratio (set BPARM(4) = 2) in the combination. See HELP COMB C_R for a description of these options and certain limitations in their use. With APARM(8) = 1 C_R, 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. Alternatively, the task REMAG can be used to replace blanked pixels in the output image with a user-specified value.

When combining two or more images, COMB, PCNTR, *et al.* must decide which pixels in the 2^{nd} image go with which pixels in the 1^{st} 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 C_R 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. tasks HGEOM and OHGEO provide this service with good interpolation methods. See § 7.6.1 and type EXPLAIN HGEOM C_R or EXPLAIN OHGEO C_R for more information.

7.2 Combining more than two images (CMPLX, SUMIM, SPIXR, STACK)

The task CMPLX is new in 31DEC21. For pairs of images which represent a complex quantity (either real and imaginary or amplitude and phase), CMPLX will perform various complex mathematical operations on one such pair (conjugate or swap the type) or combining two such pairs (various sums, multiplications, and divisions).

The task SUMIM is used to sum or average any number of images. Since *AIPS* has only a limited number of adverbs of the kind INNAME, IN2NAME, etc., SUMIM requires that all input images have identical INNAME and INCLASS. The input images are then specified by INSEQ (the sequence number of the *first* input image), IN2SEQ (the sequence number of the *last* input image), and IN3SEQ (the increment in sequence number). All input images have to reside on the *same* disk.

> TASK 'SUMIM' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to help you find the catalog number of the first input image.
> INDI *n* ; GETN *ctn* C_R	to select the first input image from disk *n* catalog slot *ctn*.
> IN2SEQ *s* C_R	to specify the sequence number of the last image to be included.
> IN3SEQ 0	to specify the increment in sequence number (= 1).
> OUTN '*xxxxx*' C_R	to specify *xxxxx* for the name of the output image.

> OUTC '*ccc*' C_R	to specify *ccc* for the class of the output image.
> FACTOR *f* C_R	to specify the factor with which to multiply each image before adding. $f = 1$ leads to summation, $f = 0$ defaults to the inverse of the number of input images (average)
> GO C_R	to start `SUMIM`.

This is a very noisy way to make a line-sum image. For more serious work, use `BLANK` (§ 7.4) and `XMOM` (§ 8.6) instead.

A popular analysis technique has been developed in which images centered upon objects of some chosen kind are simply added up even though the objects' emission is less than the noise. The belief is that, if there is real emission just below detectability in individual images, the summed image will reduce the noise and reveal the weak emission. The task `STACK` attempts such an analysis doing a weighted mean or median of a matched set of images ignoring the coordinates.

The task `SPIXR` is intended to fit spectral indexes optionally including curvature to a cube of image planes. This cube is build with `FQUBE` to make an `FQID` axis out of irregularly spaced frequencies. The cube is transposed by `TRANS` to put the `FQID` or, if build with `MCUBE`, frequency axis first. Then `SPIXR` will do a least squares fit for spectral index. Again, the results will be noisy unless the initial images have been blanked and converted to similar spatial resolution. The `DPARM` and `CPARM` options in `SPIXR` may also be used to blank pixels with bad solutions. An interactive version of `SPIXR` called `TVSPX` has appeared in `31DEC24`. It allows you to interact with the outputs to attempt to get better solutions by omitting selected channels. In `31DEC25`, a similar task called `LISPX` fits spectral indices to spectra in text files rather than image files. Another new task `SPXMD` creates model image cubes with spectral indices. See *AIPS* Memo 126.[1] The task `SPFIX` may be used to construct a model image cube based on the output of `SPIXR`. In `31DEC24`, `UNSPX` may be used to correct the input cube (or, for example, a matching polarization cube) for the spectral index.

7.3 Image statistics and flux integration

The task `IMEAN` is used to determine the statistics of the image inside, or outside a specified rectangular or circular 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 C_R	to list the input parameters.
> INDI *n* ; GETN *ctn* C_R	to select the image file from disk *n* catalog slot *ctn*.
> BLC *n1*, *n2* ; TRC *m1*, *m2* C_R	to set the window from (*n1*,*n2*) to (*m1*,*m2*) — or use `TVWIN` with the cursor on the TV.
> DOHIST TRUE C_R	to make a plot file of the pixel histogram.
> PIXRANGE *x1*, *x2* C_R	to set the range of the histogram from *x1* to *x2*.
> NBOXES *n* C_R	to set the number of boxes in the histogram.
> DOALL 1 C_R	to include pure 0 pixels in the statistics, a `31DEC24` option. Otherwise they are ignored.
> GO C_R	to run the task.

A circular aperture may be specified with `BLC = -1,`$radius$; `TRC =` X_c, Y_c. `IMEAN` attempts to determine the true noise of the image by fitting the peak of the histogram and reports both that result and the one found by including all pixels within the window. `IMEAN` now gets the initial guess for the true noise from a robust computation over the input window and returns the adverbs `PIXSTD` and `PIXAVG` from the histogram fit to the `AIPS` program. `IMEAN` now fills an additional output adverb such that `TRIANGLE(`i`)` is the brightness level exceeded by only i per cent of the image.

[1]Greisen, E. W. 2025, "Spectral index fitting in *AIPS*," AIPS Memo 126, `http://www.aips.nrao.edu/aipsdoc.html`

The statistics will appear in the *AIPS* window. For a hard copy type:

> PRTASK 'IMEAN' ; PRTMSG C_R   with PRIO ≤ 5.

To see the histogram of the intensities, an example of which is shown in § 6.3.2.3, type one of:

> GO TKPL C_R   to display the histogram in the Tektronix window.

> GO LWPLA C_R   to display the histogram on a PostScript printer.

The verbs TVSTAT and IMSTAT provide similar functions to IMEAN without the histogram and true rms options. Both return their results as AIPS parameters PIXAVG (mean), PIXSTD (rms), PIXVAL (maximum), PIXXY (pixel position of the maximum), PIX2VAL (minimum), PIX2XY (pixel position of the minimum). IMSTAT uses the same file name, BLC, and TRC parameters as IMEAN including the circular aperture convention. It is useful to prepare the initial rms guess for that task although the PIXSTD it returns will often be a factor of several too large. 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 C_R 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" in almost any transposition. The spectrum obtained may be saved as a SLice file for further analysis and display. The spectra may be saved as true plot files as well as printer plots and the flux summation may be weighted by the values (or their squares) found in the blotch image. Numerous continuum applications also exist (*e.g.*, polarization, comparison across frequency). Type EXPLAIN BLSUM C_R for a description of the operation.

The verb IMDIST is used to measure the angular distance and position angle between two pixel positions in up to two images. The separation is returned as adverb DIST. Verb TVDIST allows you to select the two pixels interactively from the TV display.

The verb IMCENTER may be used to determine the intensity-weighted centroid of a rectangular or circular portion of an image. The verb returns adverbs PIXXY, COORDINA, and ERROR giving the pixel and physical coordinates of the centroid and an indicator of success or failure.

7.4 Blanking of images

In order to determine accurate flux values in images, or moments of velocity profiles, it is desirable to restrict the integrations to pixels that contain emission, or, in other words, to exclude pixels that contribute only noise. If this is not done, the inclusion of noisy pixels will increase the rms in the derived integrated value to an unacceptable extent. The task BLANK gives the user the opportunity to replace pixels containing pure noise with values that *AIPS* and its tasks interpret as *undefined*. The decision whether a certain pixel contains pure noise, or carries some emission, can be made subjectively (using the TV) or in a more objective fashion (see below for an example). In all cases, BLANK creates an output image which is a copy of the input image with some pixels replaced by undefined values, or — if the user specifies it — by the value zero.

The most straightforward use of BLANK is to apply a cutoff to the input image, e.g. let BLANK replace with an undefined value every pixel in the input image that lies below a specified. *e.g.*, 3σ noise level. This effectively removes almost all noisy pixels. The disadvantage is that this method also removes any *signal* below the 3σ noise level. Since a substantial fraction of the total flux may be "hidden" in pixels below 3σ, this method prevents an accurate total flux determination. Another straightforward use of BLANK is to remove all pixels outside a user-specified radius. This allows blanking regions for which the primary-beam corrections, and hence the noise levels, are large.

A better way to perform the blanking is one which is not based on the pixel values in the input image itself, but on those in a *second* input image. Typically this is a convolved (spatially and/or in velocity) version of the input image, which has a higher signal to noise for extended emission than the input image. In the example given here we have the input image I_1 of full spatial resolution, and a convolved version of this input image I_2 with a linear beam size roughly twice full resolution. Careful inspection of this second image has shown that there are no outlying noise peaks above *f* mJy/beam. `BLANK` is then run as follows:

> TASK 'BLANK' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to help you find the catalog numbers of I_1 and I_2.
> INDI *n1* ; GETN *ctn1* C_R	to select I_1 from disk *n1* catalog slot *ctn1*.
> IN2D *n2* ; GET2N *ctn2* C_R	to select I_2 from disk *n2* catalog slot *ctn2*.
> OUTN '*xxxxx*' C_R	to specify *xxxxx* for the name of the blanked output image.
> OUTC '*ccc*' C_R	to specify *ccc* for the class of the blanked output image.
> OPCODE 'IN2C' C_R	to specify that the blanking is performed using pixel values in a *second* input image.
> DPARM(3) $sim\ -f\ ,\ f$ C_R	to set `DPARM(3)` and `DPARM(4)` to specify that all pixels with fluxes in the second input image in the interval *(-f,f)* should be blanked.
> GO C_R	to compute the blanked output image.

The task `REMAG` can be used to replace blanked pixels by a value to be specified by the user.

The AIPS TV display may be used to do a more subjective blanking with this task. Set OPCODE 'TVCU' C_R to display the image, one plane at a time in any transposition. You will be prompted to set "blotch" regions (much like `TVSTAT` and `BLSUM`) to define the areas to be blanked. This is one method for having different regions of signal at different spectral channels. There are also four windowing methods for blanking spectral-line cubes which have been transposed to have the frequency axis be first. In these methods, a window (range of spectral channels) about the peak signal in each spectrum is retained.

The task `RMSD` may be used to write a version of the input image blanking pixels below N times the rms in the image, computing the rms self-consistently in windows surrounding each pixel.

7.5 Fitting of images

There are three tasks and two verbs which estimate the position and intensity of a component on a two-dimensional image. The simplest and fastest methods are the verb `IMCENTER` and `MAXFIT`. The latter 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. Tasks `SAD` and `TVSAD` attempt to automate the process of finding and fitting Gaussian components in an image. Additionally, 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.

7.5.1 Centroid fits (`IMCENTER`) and parabolic fit to maximum (`MAXFIT`)

You may determine a centroid for a region in an image with the verb `IMCENTER`. Set the name parameters for the desired image and then define the region with adverbs `BLC` and `TRC`, perhaps using `TVLOAD; TVWIN` C_R. Set `FLUX` if you wish to limit the computations to pixel values greater than `FLUX`. Then

> IMCENTER; IMVAL C$_R$	to find the value at the centroid.

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

> EXPLAIN MAXFIT C$_R$	to get a good explanation of the algorithm.

The inputs 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 C$_R$

First the cursor will appear on the TV. Move it close to a maximum, press the left mouse button, and hit button A, B, C, or D. The fit will appear in your AIPS window. Adverb values PIXXY, PIXVAL, COORDINA, and ERROR will be set appropriately. Adverb FSHIFT will also be set to suggest the shift needed to place the source directly on a pixel.

7.5.2 Two-dimensional Gaussian fitting (IMFIT, JMFIT)

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 C$_R$	to list the input parameters.
> INDI *n* ; GETN *ctn* C$_R$	to select the image from disk *n* catalog slot *ctn*.
> BLC *n1*, *n2* ; TRC *m1*, *m2* C$_R$	to set the area to be fitted as (*n1*,*n2*) to (*m1*,*m2*) — or use TVWIN with the cursor on the TV.
> NGAUSS 2 C$_R$	to set the number of components to be fitted to 2.
> CTYPE 1, 1 C$_R$	to have both components be Gaussians.
> GMAX 0.34 0.20 C$_R$	to give estimates of peak intensity in Jy.
> GPOS 200, 100, 210, 110 C$_R$	to give estimates of the pixel locations of each component.
> GWID 6 4 20 6 4 20 C$_R$	to give estimates of component sizes in pixels. In this case, each component has a FWHM of 6 by 4 pixels with the major axis at position angle 20 degrees.
> DOWID FALSE C$_R$	to hold all of the widths constant (if required).
> STVERS 0 C$_R$	to have the results places in a new "stars" extension file for later plotting.
> INP C$_R$	to review inputs.
> GO C$_R$	to run the task.

To improve accuracy, 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 based on the image rms in the image header or determined by robust means over the full input image plane. 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. Comparison of the results of IMFIT and JMFIT will sometimes be instructive. The tasks will correct the results for the effects of the primary beam and bandwidth smearing if you wish. These tasks return adverbs FMAX, FPOS, and FWIDTH with the answers to the fit, DOMAX, DOPOS, and DOWIDTH with the uncertainties in the fits, and FSHIFT with the shift needed to place the first component directly on a pixel. It is wise to treat the results of MAXFIT, IMFIT and JMFIT with caution. The estimates of the errors, in particular, are based on theory and on trials of deconvolution over a range of widths.

The verb MFITSET may be used to set many of the input adverbs for IMFIT and JMFIT from the image displayed on the TV. The verb sets the image name adverbs, then directs you to set the fitting window (like

TVWINDOW), and then to point at the peak of Gaussian component 1 (sets GMAX(i) and GPOS(*,i)), then its major axis half-power point (sets GWIDTH(1,i) and GWIDTH(3,i)), then its minor axis half-power point (sets GWIDTH(2,i)). Push buttons A, B, or C to set each of these points or push button D to stop the process and set NGAUSS. Instructions will appear on your terminal.

Alternatively, use RUN INPFIT C_R (see § 12.2.1) 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) C_R to specify three components. The procedure will prompt you to set the desired sub-image window with the TV cursor (it uses verb TVWINDOW) and then to point the TV cursor at the peaks of each of the Gaussians, click the left mouse button when the cursor is correctly placed, and push button A, B, C, or D. The inputs GMAX, GPOS, BLC, and TRC are set in this way.

7.5.3 Source recognition and fitting (SAD, TVSAD)

The task SAD (§ 10.4.4) attempts to find all sources in a sub-image whose peaks are brighter than a given level. It searches the sub-image specified by BLC and TRC for all points above this level and merges such points in contiguous "islands." For each island, initial estimates of the strength, size, and number of components are generated. Then the fitting algorithm used in JMFIT is called to determine the least square Gaussian fit. Solutions which fail to meet certain criteria can be retried as two components and, if they still fail, rejected. SAD is a task with many adverbs, a full description of which would be beyond the scope of this *CookBook*. Enter EXPLAIN SAD C_R for a full description of this task and its parameters. The effects of bandwidth smearing and the primary beam may be corrected. SAD produces a Model-Fit extension file which may be converted to a stars file (§ 6.3.2) with MF2ST. The MF file may be printed with MFPRT in formats suitable for STARS and in formats which may be used, with task BOXES, to prepare Clean boxes for input to the imaging tasks. SAD can now directly attach a "stars" extension file to the input image containing the fit Gaussians. Plot tasks can then overlay them on images.

A version of SAD called TVSAD has been written along with an *AIPS* Memo[2] was written describing its use in detail. Like SAD it finds source islands and makes an initial guess. At this point it displays an interpolated image of the area surrounding the island and marks it with the island boundary and the initial guesses (plotted as ellipses at the half-power point). You may choose from a TV menu to enhance the display, to change the island boundary, to change one of the initial guesses, to change the number of Gaussians guessed and set their values (like verb MFITSET), to have the fit attempted, or to reject this island and go on to the next. After the fit, a residual image is displayed with the island boundary and fit Gaussians marked. At this stage you may accept the fit, go back to the previous menu to try again, or reject this island and go on the next island. You may turn off the TV interaction and have the task run automatically. It will turn the TV interaction back on if it finds a fit that does not meet the acceptance criteria. Finally, TVSAD writes out the residual image, printer displays, and extension files exactly like SAD.

7.5.4 Gaussian fits to slices (SLFIT)

You can generate a one-dimensional slice (profile) through any plane (characterized by the first two coordinates) 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. A slice along the third axis, at a single pixel (not necessarily integers) in the first two axes, may also be prepared. Slices directly along any single axis may be saved without interpolation if desired. If the slice is directly along an FQID axis, SLICE will output a slice which

[2] Greisen, E. W. 2014, "TVSAD: interactive search and destroy" AIPS Memo 119, http://www.aips.nrao.edu/aipsdoc.html

is linearly sampled in frequency. Tasks ISPEC, BLSUM, and TVSPC may also save slice extension files. These are somewhat different in that they can represent a sum over an area in the sky as a function of frequency. Nonetheless, the slice file display and fitting software is of considerable use with these as well. The set of software dealing with slice file analysis and display can be obtained on your terminal by typing ABOUT ONED C_R. The list is also given in Chapter 13.

To generate a slice:

> TASK 'SLICE' ; INP C_R	to review 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:

> SETSLICE C_R

then set the TV cursor to the desired beginning point for the slice, press the left mouse 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.

Slice files may be output as ASCII text files using the OUTTEXT adverb. They contain detailed pixel and coordinate information as well as the interpolated slice values. They may also be printed later by SLPRT including any one of the models fit to it. 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 § 6.3 above, but use INEXT 'SL' C_R in place of INEXT 'PL' C_R.

When SLICE has terminated, the file may be plotted in the TV display on your workstation using:

> INP TVSLICE C_R	to review the inputs to verb TVSLICE.
> 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 slice minimum to the slice maximum. You can alter the part of the slice that is plotted and the vertical scale by specifying, for example:

> BDROP 100 ; EDROP 225 C_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 C_R	to set the range of the vertical axis to be -1 to 4 mJy/beam.
> TVSLICE C_R	to plot the slice in the TV window.

Note: several slices may be put on one TV plot. Use TVASLICE C_R for the additional ones. Multiple colors may be achieved by using different graphics channels (GRCHAN).

Slice files may be converted into plot files by:

> GO SL2PL C_R

The resulting plot files may then be output by:

> GO LWPLA C_R	to display the plot file on a PostScript printer.
> GO TKPL C_R	to display the plot file in the Tek window.
> GO TVPL C_R	to display the plot file on a TV graphics plane.

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

> INEXT 'SL'; EXTL C_R	to list the parameters of the slice files.
> INVERS m C_R	to select the $m^{\rm th}$ file for analysis.
> TVSLICE C_R	to plot the slice in the TV window.
> EDROP 840 ; BDROP 700 C_R	to select a subsection to fit.
> TVSLICE C_R	to re-plot just the subsection.

> ORDER 1 C_R	to fit a linear baseline, no baseline up to quadratic are allowed.
> NGAUSS 2 C_R	to fit 2 Gaussians.
> TVSET C_R	

This verb will prompt you to `POSITION CURSOR AT CENTER & HEIGHT OF GAUSSIAN COMP 1`. Move the cursor to the requested position and hit any button. Then you are asked to `POSITION CURSOR AT HALFWIDTH OF GAUSSIAN COMP 1`. Move the cursor to the half-intensity point of the component and click any button. Continue until all components have been entered. (Note: these operations are also available on the Tek device with verbs beginning with `TK`. We recommend the TV versions since cursor reading in X-Windows emulations of Tek devices appears to be unreliable.) Then type:

> TVAGUESS C_R	to plot the guess on top of the slice plot.

If everything looks ok, then:

> GO SLFIT C_R	to run the task.

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

> PRTASK 'SLFIT' ; PRTMSG C_R	to print the message file.

and, to display the results in the TV window, enter:

> TVSLICE C_R	to re-plot the slice.
> TVAMODEL C_R	to add the model results to the plot.
> TVACOMPS C_R	to add the individual components of the model to the plot.
> TVARESID C_R	to add the residuals (data – model) to the plot.

The TV plot may be easier to examine if you use different `GRCHAN` for the different types of plot. To get a higher quality plot of the results, an example of which is shown in § 6.3.2.1, type:

> DORES TRUE ; DOMOD 3 C_R	to request the residuals, the model, and the model components.
> DOSLICE FALSE C_R	to leave the slice data out of the plot.
> TASK 'SL2PL' ; GO ; WAIT C_R	to make a plot file and wait for it to be complete.

7.5.5 Spectral parameter fitting (TVSPC, XGAUS, ZEMAN, RMFIT, FARS)

Task `TVSPC` assists users in getting to know their data cubes. It uses a 2-dimensional image plane of some parameter that is useful in visualizing the full field. A moment-zero image is one obvious choice. The user selects a pixel on this image using the TV display and a spectrum at that coordinate is displayed. The task is highly interactive allowing a quick choice of an interesting pixel. The spectrum, or a portion of the spectrum, at that point may be fit by up to 4 Gaussians plus baseline and may be saved as a `SL`ice file. The corresponding spectrum from a second transposed cube may also be displayed, allowing for example the I and V or the Q and U polarizations to be displayed simultaneously. Optionally, a third spectral cube may be displayed with the spectral plane selected interactively from the spectral plots. See the associated AIPS Memo[3]. In earlier releases, use `XPLOT` with `DOTV 1` to familiarize yourself with your image cube.

A task called `TVIEW` offers interactive TV displays of an image cube in all 3 transpositions plus a reference image or spectrum. The task is similar to `TVSPC` but is designed solely to help the user become better acquainted with the data cube. The task is described in an AIPS Memo[4].

`XGAUS` is an interactive task which can fit up to eight Gaussians and a linear baseline to each row of an image. In `31DEC13`, it was redesigned so that it begins by setting up a table of the peak image intensities

[3]Greisen, E. W. 2019, "Exploring Image Cubes in AIPS," AIPS Memo 120 revised, `http://www.aips.nrao.edu/aipsdoc.html`
[4]Greisen, E. W. 2019, "Further Exploration of Image Cubes in AIPS," AIPS Memo 124, `http://www.aips.nrao.edu/aipsdoc.html`

and fit results, so that it may be re-started multiple times with the same table, and so that it provides multiple means to edit the results before it finally 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 § 8.5.2), it has a wide variety of other applications. The interaction is optional and uses the TV window on your workstation. The data, initial guess, model fit (total and components), and the residual for each row may be plotted on the TV 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 cursor in the TV window. This process is similar to that of TVSET described above (§ 7.5.4). If things go well, you may turn off the interactive model setting; the task will turn it back on if there is a bad fit at some pixel. After all pixels have been fit, the task enters an image and edit mode. Images are computed for all Gaussian parameters and their uncertainties. A menu is shown offering displays of the images and several methods to revisit the solutions for selected pixels. This includes swapping the solutions between components and re-doing a list of individual pixels or all pixels for which the solutions or uncertainties exceed specified ranges. When you are satisfied with the result, you may instruct the task to write out images of the Gaussian parameters and their uncertainties. The use of XGAUS as well as ZEMAN, RMFIT, and the plotting and modeling tasks (below) is discussed in detail in AIPS Memo 118.[5]

Task ZEMAN was written to fit for Zeeman splitting. The basic function fit to the V Stokes spectrum is a constant time the total intensity plus another constant times the derivative of the total intensity with frequency. The task is similar to XGAUS in that it first sets up a table of image intensities and fit results, then fits all pixels above specified intensities, and then offers the image display and edit modes. Unlike previous Zeeman programs (in other packages), ZEMAN also offers the option to use the results of XGAUS on the total intensity cube to fit the V Stokes parameter with a constant times the total intensity plus a separate constant for the derivative of each Gaussian component with frequency. This allows for multiple magnetic fields at the same pixel, each corresponding to a separate Gaussian spectral component.

Tasks AGAUS and ZAMAN have appeared to perform the same functions but on absorption-line spectral cubes. The fit is done to the observations, but the parameter that is Gaussian is the optical depth. These tasks and the essential mathematics are described in AIPS Memo 122.[6]

Task XG2PL plots the solution for individual spectra from XGAUS. The corresponding ZEMAN fit may also be plotted in a separate panel. Absorption line and optical depth models may also be plotted.

Polarization spectra are often fit for rotation measure. The old task RM reads a transposed cube of the polarization angle made with multiple COMBs, then FQUBE or MCUBE followed by TRANS. It fits the rotation measure and the intrinsic magnetic field direction using one of two possible methods to resolve lobe ambiguities.

The task FARS reads two similar cubes of Q and U polarization images and performs a "rotation-measure synthesis." The output of FARS is an image cube of the real part and another image cube of the imaginary part of the Fourier transform on the λ^2 axis, with or without one-dimensional Cleaning. Slice files of the complex RM transfer function ("dirty beam") are also written. Task AFARS reads these cubes to produce images of the maximum rotation measure and of the amplitude or phase at the maximum. Task RFARS applies the rotation measure image to the FARS input cubes to produce de-rotated cubes of Q and U. These may be SQASH'ed to make an image of the average un-rotated polarization. The RUN file/procedure DOFARS assists with the whole process, doing matrix transpositions, finding channel weights, running FARS, transposing the outputs, and cleaning up.

Rotation-measure synthesis remains an experimental procedure at present. To test the handling of spectra in Q and U, the task TARS was written. It reads a text file giving the spectrum of Q and U at some single coordinate and performs the RM synthesis using methods from FARS plus an additional experimental form of complex Clean. Up to 20 model components may be added to the Q/U spectral data at specified rotation

[5]Greisen, E. W. 2020, "Modeling Spectral Cubes in AIPS," AIPS Memo 118 revised, http://www.aips.nrao.edu/aipsdoc.html
[6]Greisen, E. W. 2020, "Modeling Absorption-line Cubes in AIPS," AIPS Memo 122, http://www.aips.nrao.edu/aipsdoc.html

measure, amplitude, and phase. Task QUXTR was written to extract a spectrum from and pair of Q and U polarization cubes for input to TARS. The text-file output of TARS may then be plotted with TARPL, which allows multiple RM spectra from the same TARS output file to plotted on top of each other. In 31DEC20, new task MTARS prepares model Q/U spectra with up to 9999 model components and FTARS performs an interactive fitting of the input to TARS using the TARS output as a guide. This is RMFIT (see below) for a single spectrum. TARPL can now also lot the input to TARS.

It has been found that rotation-measure synthesis is severely limited in its ability to separate multiple RM components. Therefore, another XGAUS-like task called RMFIT was written. It uses the FARS rotation-measure output cubes to provide initial guesses for fits to the Q and U polarization spectral cubes. Like XGAUS, it first builds a table of image intensities to hold the fit results and then displays the FARS spectrum at each pixel to take an initial guess with which to fit the Q and U spectra. They are displayed and, if the fit is accepted, the task goes on to the next pixel. RMFIT is restartable and has the same sort of image/edit stage in which images are displayed and the fits may be revisited by means similar to those in XGAUS. An option to fit a spectral index in Q and U is offered, but, in general, the input cubes should have been corrected by the total intensity spectral index and the channel-dependent primary beam before being used in RMFIT. An additional spectral index in Q and U may then indicate "thick" rotation measure screens rather than a true source spectral index. Other models of thickness may be fit instead. The task may be re-started repeatedly and output images written only when desired. Despite rather simple fitting methods, RMFIT has shown the ability to separate closely spaced components not separable by rotation measure synthesis. Task RM2PL makes plot files from the solutions for individual Q/U or Polarization/angle spectra.

7.6 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, VLBI, and single-dish data reduction are described in Chapter 8, Chapter 9, and Chapter 10, respectively. Chapter 11 of *Synthesis Imaging in Radio Astronomy*[7] covers the topic of image analysis in more detail.

7.6.1 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 § 6.1.2 and *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.

The simplest geometric conversion is a re-gridding of the data with new axis increments and dimensions with no change in the type of projection or coordinates. The task LGEOM performs this basic function and also allows rotation of the image. One use of this task is to obtain smoother displays by re-gridding a sub-image onto a finer grid. To rotate and blow up the inner portion of a 512^2 image, enter:

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

> TASK 'LGEOM' ; INP C_R	to review the inputs.
> INDISK *n* ; GETN *ctn* C_R	to select the image.
> BLC 150 ; TRC 350 C_R	to select only the inner portion of the image area.
> IMSIZE 800 C_R	to get an 800^2 output image. This will allow the sub-image 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 C_R	to reset all parameters to defaults.
> APARM(3) = 30 C_R	to rotate the image $30°$ counterclockwise (East from North usually).
> APARM(4) = 3 C_R	to blow up the scale (axis increments) by a factor of 3.
> APARM(6) = 1 C_R	to use cubic polynomial interpolation.
> INP C_R	to check the inputs.
> GO C_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. OGEOM is similar to LGEOM, but handles blanked pixels in a manner that does not increase the blanked area.

A much more general geometric transformation is performed by OHGEO and HGEOM, which convert 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. One of these tasks should be used before comparing images (with COMB, KNTR, 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 OHGEO C_R to obtain the details and useful advice. SKYVE re-grids images from the Digital Sky Survey (optical DSS) into coordinates recognized by AIPS.

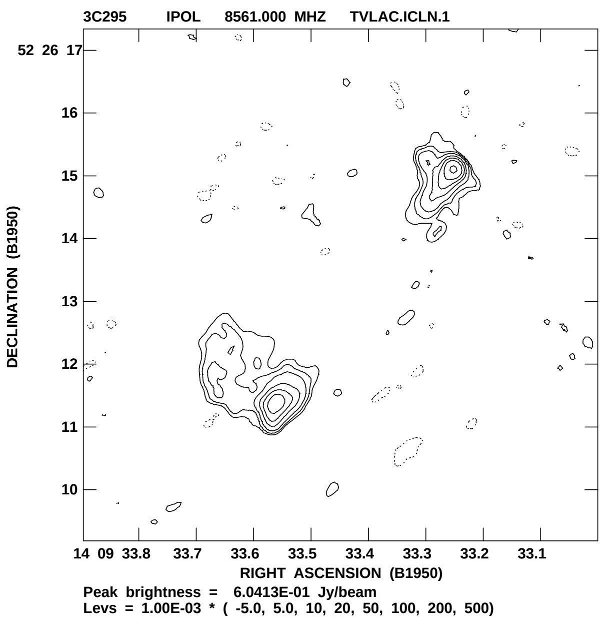
A potentially very powerful transformation is performed by PGEOM. In its basic mode, it converts between rectangular and polar coordinates. An example of this operation is illustrated in Figure 7.1. However, PGEOM can also "de-project" elliptical objects to correct for their inclination and "unwrap" spiral objects. Type EXPLAIN PGEOM C_R for information.

7.6.2 Mathematical operations on a single image

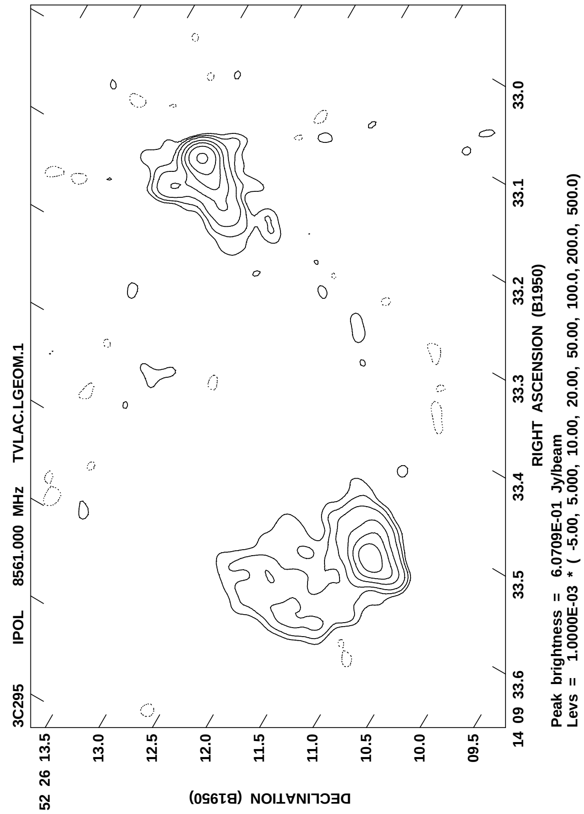
The task MATHS allows the user to do a mathematical operation on a single image on a pixel by pixel basis. Currently supported mathematical operators are: SIN, COS, TAN, ASIN, ACOS, ATAN, LOG, LOGN, ALOG, EXP, POLY, POWR, and MOD. An example of MATHS follows, in which the output image (OUT) is computed in terms of the natural logarithm of the input image (IN) as follows: $\texttt{OUT} = 4 + 2 \times (\log(3 \times \texttt{IN}) - 1)$

> TASK 'MATHS' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to help you find the catalog number of the input image.
> INDI *n* ; GETN *ctn* C_R	to specify the image, on disk *n* catalog slot *ctn* as the input.
> OUTN '*xxxxx*' C_R	to choose *xxxxx* as the name for the output image.
> OUTC '*ccc*' C_R	to choose *ccc* as the class for the output image.
> OPCODE 'LOGN' C_R	to specify the operation to be performed (a natural logarithm).
> CPARM 4 , 2 , 3 , -1 C_R	to specify the coefficients.
> GO C_R	to start MATHS.

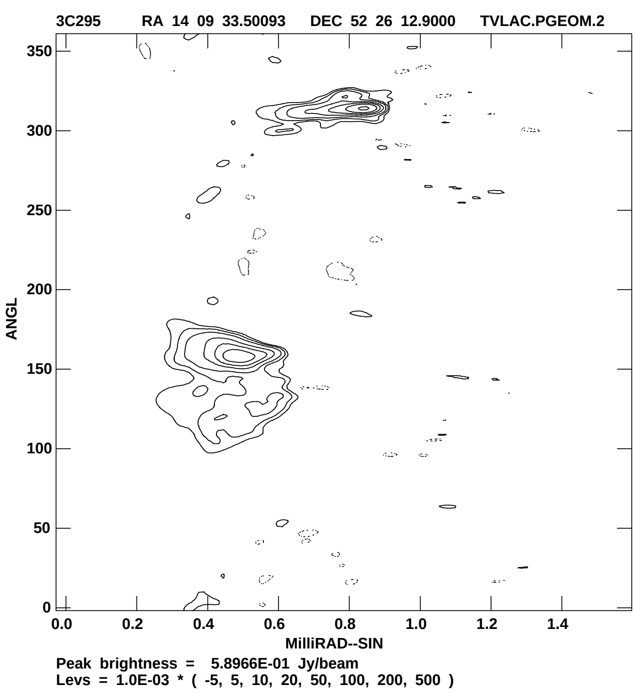
Undefined output pixels (in the current example, all pixels in the input image ≤ 0) are either blanked ($\texttt{CPARM(6)} \leq 0$) or put to zero ($\texttt{CPARM(6)} > 0$). Type EXPLAIN MATHS C_R for further information on the available operators and the meaning of CPARM for any particular operator.



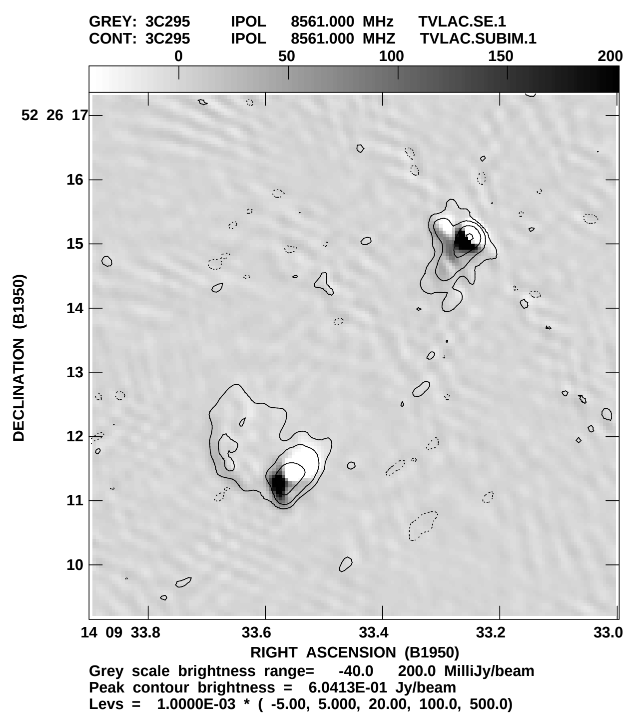
input image



`LGEOM` with rotation and interpolation



`PGEOM`



`NINER`: `'SE '` derivative

Figure 7.1: Geometric and other functions on an image.

7.6.3 Primary beam correction

PBCOR allows correction for the attenuation due to the shape of the primary beam. Its use is straightforward:

> TASK 'PBCOR' ; INP C_R	to review the required inputs.
> INDI 0 ; MCAT C_R	to help you find the catalog number of the input image
> INDI *n* ; GETN *ctn* C_R	to select the input image from disk *n* catalog slot *ctn*.
> OUTN '*xxxxx*' C_R	to specify *xxxxx* for the name of the output image.
> OUTC '*ccc*' C_R	to specify *ccc* for the class of the output image
> PBPARM 0 C_R	to use the VLA, EVLA, ATCA, GMRT, or MeerKAT beam parameters which are known for the particular receiver.
> COORDIN 0 C_R	to use the pointing position from the image header.
> GO C_R	to start PBCOR.

The default behavior requested above uses the position in the header as the pointing position and uses the empirically determined shape of the EVLA, VLA, MeerKAT, GMRT, or ATCA primary beam; PBCOR will scale the primary beam shape according to the frequency provided in the image header and use the parameters associated with the particular antenna feed. These defaults can be overridden by specifying particular values of COORDIN and PBPARM. They beam parameters are frequency dependent within each band, so PBCOR determines the beam value at the two tabulated frequencies nearest to the image frequency and then interpolates. In 31DEC21, Gaussian and polynomial beam patterns have been added as options specified with OPCODE and APARM. Beam parameters for MeerKAT and GMRT were added to 31DEC22 and 31DEC23, respectively, and are known to all tasks which correct for the primary beam.

The task SPCOR may be used to apply corrections both for primary beam and for spectral index. The latter are based on images of spectral index and spectral index curvature such as those used in IMAGR (§ 5.3.4.4). Such corrections may be significant in Faraday-rotation synthesis (§ 7.2, § 7.5.5).

An image of the primary beam may be generated with the task PATGN using OPCODE 'BEAM' C_R with other adverbs to give the antenna type, frequency, cell size, image size, and, optionally, the parameters of the beam shape.

7.6.4 Changing the resolution of an image

CCRES allows you to change the resolution of a Cleaned image by removing any Clean components in the image and then restoring Clean components with your choice of resolution. The task may also be used to remove Clean components to create a residual image or to restore Clean components to an existing residual image. CCRES allows you to smooth or hyper-resolve your image. Unlike RSTOR, CCRES re-scales the residual image to put it into units of Jy/beam for the new beam. This may be a superior way to image with a beam that does not replicate the central portion of the dirty beam. IMAGR leaves the residual image in units of Jy per dirty beam while restoring the Clean components in units of Jy per the Clean beam given in the header.

Note that CCRES only changes the Clean components, not the residual image resolution. Furthermore, CCRES does not take into account the varying resolutions of the many planes in an image cube. CONVL on the other hand, with OPCODE 'GAUS', will convolve both the Clean components and the residual to the requested resolution, taking into account the change in input resolution as a function of frequency.

7.6.5 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 output 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 KNTR / LWPLA combination (see Figure 7.1). Enter EXPLAIN NINER C_R for additional information.

MWFLT, at present, applies any one of six non-linear, 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. Two of the operators are a "min" and "max" within the window. When applied in succession, they produce a useful low-pass filtered image (Rudnick, L. 2002, PASP, 114, 427). Other operators 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 filters can be turned into high-pass filters by subtracting the output of MWFLT from the input with COMB. Type EXPLAIN MWFLT C_R for further information.

Histogram equalization provides another form of non-linear filtering. HISEQ converts the intensities of the full input image to make an output image with a nearly flat histogram. This magnifies small differences in the heavily occupied parts of the histogram (usually noise) and diminishes large differences in the less occupied parts (often real signal). TVHLD is an interactive task that loads an image to the TV with histogram equilization and then allows the intensity range and method of computing the histogram to be modified. It can write out at the end an equilized image in arbitrary (non-physical) units. AHIST does an "adaptive" histogram equalization on each pixel using a rectangular window centered on that pixel. This will magnify small differences in a more local sense, bringing out structures in smooth areas of different brightness. SHADW generates a shadowed image as if a landscape having elevation proportional to image value were illuminated by the Sun at a user-controlled angle. Although these tasks magnify noise, they are likely to elucidate real structures in large areas of nearly constant brightness.

7.7 Modeling in the image and uv planes

Special considerations for analysis and modeling of spectral-line image cubes are discussed in § 8.6. Models of galaxy rotation are fit by GAL to images of the predominant velocity (first moment images usually), while the entire data cube may be fit by task CUBIT.

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 MODIM, MODSP, UVMOD, MODAB, and SPMOD will add (or subtract) up to 9999 point, Gaussian, disk, rectangular, spherical, or exponential sources to the (scaled) input image or *uv* data, respectively. These tasks can also add noise and allow the original data to be replaced by the model. UVMOD can include a spectral index for the sources, SPMOD includes spectral lines for the sources, MODIM can include both a spectral index, a rotation measure, and even a rotation measure thickness for each component, and MODSP can include polarized spectral lines with spatially varying spectral structure. Type EXPLAIN MODIM ; EXPLAIN UVMOD ; EXPLAIN SPMOD ; EXPLAIN MODAB ; EXPLAIN MODSP C_R for details or see AIPS Memos 118[8] and 122[9]. For simple cases, the older task IMMOD may be easier to use than MODIM.

The task CCMOD will create a clean-components file representing a Gaussian, disk, or Gaussian times 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 *uv* data sets with a maximum of 2.5 million visibilities. Up to 60 source components may be fit along with up to 30 antenna gains. Provision for getting initial guesses for the specified spectral channel from a text file with guesses for many channels is now available as is a compact text file output option for the solutions. Task OMFIT is a more complicated and difficult to use task which fits source components in a *uv* data set along with antennas gains. It allows multiple different model types and even a full self-calibration. The results from OMFIT are often well worth the extra effort to obtain them.

AIPS contains tasks which create simulated data. PATGN will make images of various patterns. DTSIM generates *uv* data following instructions given primarily in a keyin-format text file. You may specify the antennas in detail or as the VLBA or VLA. Source parameters and models, frequency values and structure, data calibration errors of numerous types, scan structure and sequence, and more may be specified. UVCON also generates *uv* data from "scratch" using text files of antenna data and Tsys and efficiency data as well as images of a source model. The two tasks differ in their approach, making both of considerable interest.

7.7.1 Analysis in the uv plane

Several tasks now perform some analysis on *uv* data. The other new task, ELFIT, fits polynomials to table data as functions of elevation, zenith angle, hour angle, parallactic angle, or azimuth. It has been used, for example, to measure antenna spillover, plotting Psys versus zenith angle.

DFTIM is similar to the plot task DFTPL in that it Fourier transforms visibility data to a partuclar image pixel (clestial coordinate). DFTPL makes a plot versus time for a single group of frequencies. It can do various Stokes including polarization amplitude and position angle. DFTIM instead writes out an image at that coordinate with frequency and time as the axes. The frequencies and times in this "waterfall" image may be averaged as the image is created. ACIMG makes a similar waterfall plot for autocorrelation data, with the third axis being antenna number.

Statistical analysis of visibility data is performed by several tasks. UVRMS averages the selected data in SOLINT time intervals and then can plot the average values as a function of time and as a histogram. A variety of statistical parameters are computed and printed. SPRMS determines and plots the mean and standard deviation of the selected data as a function of spectral channel. Note that both tasks will combine data from multiple sources, baselines, etc. if so instructed. RIRMS computes statistics over the included visibility data and then prints a matrix over baselines of the real and imaginary parts separately. Histogram plots and plots over time may be generated. In 31DEC21, task VBRFI computes statistics for a data set consisting only of auto-correlation records. It writes out a text file containing the results and can make plots. PLRFI can make plots from one or more of these text files.

[8]Greisen, E. W. 2020, "Modeling Spectral Cubes in AIPS," AIPS Memo 118 revised, http://www.aips.nrao.edu/aipsdoc.html
[9]Greisen, E. W. 2020, "Modeling Absorption-line Cubes in AIPS," AIPS Memo 122, http://www.aips.nrao.edu/aipsdoc.html

7.8 Additional recipes

7.8.1 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.

7.8.2 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 $350\deg$ about 15 minutes or until the bananas are tender (but not soft), basting with the sauce several times.

7.8.3 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 cup 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 $9 \times 5 \times 3$-inch loaf pan and spread to corners.
8. Bake at 350° F 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.