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 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 help you find the catalog numbers of the Q and U images that you want to combine.
> 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 $\sqrt{Q^2+U^2}$ 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 $\frac{1}{2}\tan^{-1}\left(\frac{U}{Q}\right)$ 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 rescale 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	$a_1M_1 + a_2M_2 + a_3$
'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.

7.1.4 Considerations in image combination

For some applications of `COMB`, undefined pixel values may occur. For example, if the spectral index is being calculated and the intensity level on either image is negative, the index is undefined. In this case, the pixel value is given a number which is interpreted as undefined or "blanked." Blanking also arises naturally in operations of division, opacity, polarization angle, and clipping and, of course, the input images may themselves be blanked. In addition, the output image can be blanked (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 2nd image go with which pixels in the 1st image. The user input parameter `DOALIGN` controls this process. A value of 1 requires the two headers to be correct and sufficiently similar that an alignment by coordinate value is possible. A value of −2 tells the programs to ignore the headers and align by pixel number. Enter HELP DOALIGN 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. Task `HGEOM` provides this service with up to 7th-order polynomial interpolation. See § 7.6.1 and type EXPLAIN HGEOM C_R for more information.

7.2 Combining more than two images (`SUMIM`)

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.

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

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

> TASK 'IMEAN' ; INP 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.
> GO C_R	to run the task.

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. The adverbs PIXSTD gives IMEAN help in determining which values to use for the true noise fit.

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

> PRTASK 'IMEAN' ; PRTMSG C_R	with PRIO $\leq$ 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 TEK 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 and is useful to prepare the initial rms guess for that task. 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. However, numerous continuum applications also exist (*e.g.*, polarization, comparison across frequency). Type EXPLAIN BLSUM C_R for a description of the operation.

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 zero values.

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.

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 below 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 form 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) $-f$; DPARM(4) f C_R	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.

7.5 Fitting of images

There are three programs which estimate the position and intensity of a component on a two-dimensional image. The simplest and fastest method is the verb `MAXFIT`. This fits a two-dimensional parabola to the maximum within a few pixels of an image position, and gives the peak and its position. The tasks `IMFIT` and `JMFIT` are similar and fit an image subsection with up to four Gaussian components with error estimates. Task `SAD` attempts to automate the process of finding and fitting Gaussian components in an image. 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 Parabolic fit to maximum (MAXFIT)

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.

7.5.2 Two-dimensional Gaussian fitting (IMFIT)

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

> TASK 'IMFIT' ; INP 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 in the fitting process (if required).
> INP C_R	to review inputs.
> GO C_R	to run the task.

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

Use RUN INPFIT 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 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. 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:

> TVSLICE 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 OUTFILE adverb. 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 TEK window on your workstation using:

> INP TKSLICE C_R	to review the inputs to verb TKSLICE.
> 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.
> TKSLICE C_R	to plot the slice in the TEK window.

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

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.
> TKSLICE C_R	to plot the slice in the TEK window.
> EDROP 840 ; BDROP 700 C_R	to select a subsection to fit.
> TKSLICE C_R	to re-plot just the subsection.

> NGAUSS 2 C_R	to fit 2 Gaussians.
> TKSET 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 click the left mouse 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 the left mouse button. Continue until all components have been entered. Then type:

> TKAGUESS 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 TEK window, enter:

> TKSLICE C_R	to re-plot the slice.
> TKAMODEL C_R	to add the model results to the plot.
> TKARESID C_R	to add the residuals (data – model) to the plot.

To get a higher quality plot of the results, an example of which is shown in § 6.3.2.1, type:

> DORES TRUE ; DOMOD TRUE C_R	to request the model and the residuals.
> 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.

7.5.4 Other one-dimensional Gaussian fits (XGAUS)

`XGAUS` is an interactive task which can fit up to four Gaussians and a linear baseline to each row of an image. It writes its results as a set of $n-1$ dimensional image files. Although `XGAUS` was designed for use primarily on transposed spectral-line cubes (see § 8.5.2), it has a wide variety of other applications. The interaction is optional and uses the TEK window on your workstation. The data, initial guess, model fit, and the residual for each row may be plotted on the TEK screen. If the number of Gaussians being fit is larger than one, you may choose for each row to enter a revised initial guess using the cursor in the TEK window. This process is similar to that of `TKSET` described above (§ 7.5.3). This task has too many options to do them justice here. Enter EXPLAIN XGAUS C_R for details.

7.5.5 Source recognition and fitting (SAD)

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. `SAD` produces a Model-Fit extension file which may be converted to a stars file (§ 6.3.2) with `MF2ST` or printed (§ 10.4.4) with `MFPRT`.

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*[1]*Synthesis Imaging in Radio Astronomy*, A collection of Lectures form 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) 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 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:

> 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 deg 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. Type EXPLAIN LGEOM C~R~ for more information and advice.

A much more general geometric transformation is performed by `HGEOM`, which converts one image into the geometry of a second image. The type of projection, the axis increments, the rotation, and the coordinate reference values and locations of one image are converted to those of a second image. `HGEOM` should be used before comparing images (with `COMB`, `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

[1] *

reference positions. Use EXPLAIN HGEOM C_R to obtain the details and useful advice. SKYVE regrids 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 § 7.6.6. 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: OUT $= 4 + 2 \times (\log(3 \times$ 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 (CPARM(6) ≤ 0) or put to zero (CPARM(6) > 0). Type EXPLAIN MATHS C_R for further information on the available operators and the meaning of CPARM for any particular operator.

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
> DPARM 0 C_R	to use the header pointing position and VLA beam parameters.
> GO C_R	to start PBCOR.

The default behavior of DPARM is to use the position in the header as pointing position, and use the empirically determined shape of the VLA primary beam; PBCOR will scale the primary beam shape according to the frequency provided in the image header. All these defaults can be overridden by specifying particular values of DPARM.

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

7.6.4 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 § 7.6.6). Enter EXPLAIN NINER C_R for additional information.

`MWFLT`, at present, applies any one of four 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. The others produce, at each pixel, the weighted sum of the input and the median, the "alpha-trimmed" mean, or the alpha-trimmed mode of the data in the window surrounding the pixel. These last filters can be turned into high-pass filters by subtracting the output of `MWFLT` from the input with `COMB`. Type EXPLAIN MWFLT 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). `AHIST` does an "adaptive" histogram equalization on each pixel using a rectangualr window centered on that pixel. This will magnify small differences in a more local sense, bringing out structures in smooth areas of different brightness. Although both tasks magnify noise, they are likely to ellucidate real structures in large areas of nearly constant brightness.

7.6.5 Modeling

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

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

7.6.6 Examples

Figure(s) omitted to shorten PostScript file.

input image `LGEOM` with rotation and interpolation

Figure(s) omitted to shorten PostScript file.

`PGEOM` `NINER`: `'SE '` derivative

Figure 7.1: Geometric and other functions on an image.