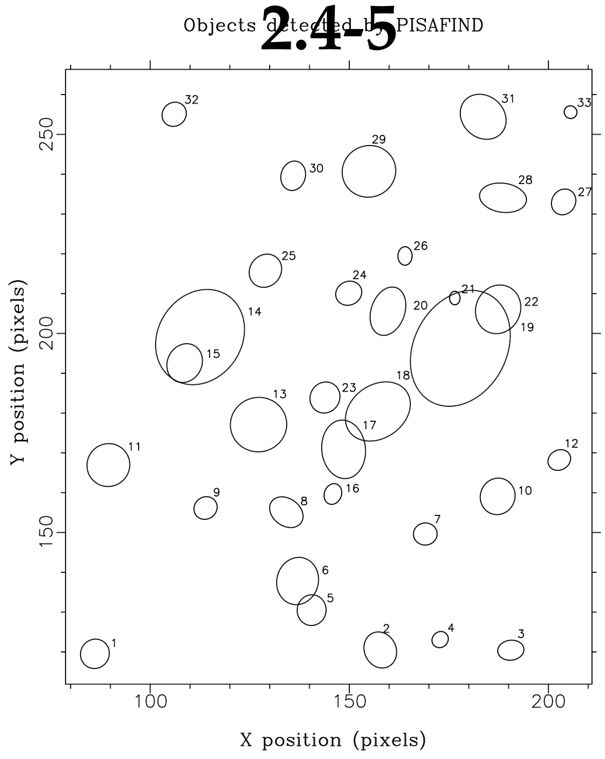
SUN/109.11

Starlink Project
Starlink User Note 109.11

Peter W. Draper and Nicholas Eaton

23 October 2002

PISA
Position Intensity and Shape Analysis
2.4-5



Contents

1 Introduction

The acronym PISA stands for Position, Intensity and Shape Analysis, and is the group name for a package of routines that deal with the location and parameterisation of objects on an image frame.

The core of this package is the routine PISAFIND which performs image analysis on a 2-dimensional data frame. The program searches the data array for objects that have a minimum number of connected pixels above a given threshold and extracts the image parameters (position, intensity, shape) for each object. The image parameters can be determined using thresholding techniques or an analytical stellar profile can be used to fit the objects. In crowded regions deblending of overlapping sources can be performed.

PISAFIND is based on the APM IMAGES routine. The algorithms are the same as in IMAGES but the interface has been adapted to Starlink, so that for instance the input data is now expected to be an NDF (see [1] which also describes other features of the Starlink Software Environment). The APM object finding and analysis package IMAGES was originally written by Mike Irwin at the University of Cambridge to analyse output from the Automated Photographic Measuring system. The background to the APM image analysis package is given in [2].

In addition to its object detection capability PISA can also perform simple object classification. The classification is performed by the simple thresholding of an additionally generated 'peakedness' measure, or by the multivariate analysis of a set of intensity corrected variables.

2 Classified listing of routines

The package routines (other than PISAFIND) are:

PISAPLOT plots the results of PISAFIND as a series of ellipses on a graphics device, it allows the ellipses to be overlaid onto displayed images.

PISAFIT fits the PISA analytic stellar profile to a list of objects.

PISAGEN generates analytic objects and places them in an NDF data array.

PISAPEAK modifies the PISAFIND parameterisations so as to produce variables which are nearly intensity independent. The variables are further modified so that they have values in proportion to those of an equivalent stellar profile.

PISAKNN performs non-parametric (distribution free) multivariate discriminant analysis on the results from PISAPEAK.

PISACUT is a utility routine for separating any file into two given a threshold value for a particular variable.

PISAMATCH is a utility program for matching indices between files. It included primarily to match classification indices against the PISAFIND parameterisations.

PISA2CAT converts any of the PISA results files into catalogues that can be used by the CURSA[5] or CATPAC[8] application packages.

PISA2ARD converts the results file from the PISAFIND routine into an ARD description of (possibly scaled) ellipses – one for each detected object.

PISAGREY plots an NDF data array as a greyscale.

3 Running the PISA software

PISA is available as part of the Starlink Software Collection and can be used from ICL and the C-shell. The package is always initialised by the command

```
# pisa
```

after performing the required Starlink initialisations. Note that in this document the # sign is used as a generic prompt and should not be typed.

When using PISA from the C-shell care needs to be taken with special characters, some of which may be required by the parameter system. In these cases (such as quoted strings `""`, and vector braces `[]`) the characters must be protected from interpretation by the use of the escape character \ or by use of single quotes.

3.1 Getting Help

On-line help is available using the

```
# pisahelp
```

command from the C-shell. From ICL use the command

```
ICL> help pisa
```

Help can also be obtained at any prompt by specifying ? or ??. Program execution can be halted at a prompt by returning the abort command (!!).

As an alternative to these approaches this document can be viewed on-line using a hypertext browser using the command:

```
# showme sun109
```

4 Object detection and parameterisation – PISAFIND

PISAFIND performs image analysis on a 2-dimensional data frame. It has two basic modes of operation. The first is isophotal analysis in which pixels with data values above a given threshold are examined for connectivity and combined into objects. This type of analysis should be used on images that contain a heterogeneous collection of objects such as a mixture of stars and galaxies.

The second mode of operation is profile fitting in which an analytical stellar profile is fitted to the objects found by a preliminary isophotal analysis. This latter option should only be used in situations where all the images on a data frame have the same shape, such as in a star cluster. This type of analysis is also performed by the DAOPHOT [3] package which specializes in analysing crowded stellar fields. The determination of the best fit model parameters is performed by PISAFIT.

In crowded regions deblending of overlapping sources can be performed. The isophotal analysis does this by examining each object at a number of higher isophotes to see if the object splits into more than one component. The profile fitting does this by modelling the data with the analytic profiles and seeing if the fit can be improved by decreasing or increasing the number of objects, as well as altering their position and brightness. In both cases the intensity of the combined image is partitioned between the components to ensure conservation of flux.

If the objects have extensive wings to their intensity profiles then the simple isophotal analysis can underestimate the total intensity in an object; two options for estimating the total intensity are available. The first uses a circular aperture of specified radius within which you know the total light to be contained. The second uses an automatic curve of growth analysis in which the intensity within elliptical apertures of increasing size is measured until a maximum is reached. This is similar to a Kron-style analysis.

For the profile fitting case the intensity of an object is obtained by integrating under the analytic profile using the relevant parameters (the actual functions are described in appendix B). The analytic profile is made up of three components. The core of the profile is a Gaussian but below a given level an exponential function takes over. The two functions are under-pinned by a Lorentzian function which is summed over all regions. If the results of the analysis are to be compared to other frames then the magnitudes from the profile fitting have to be related to total magnitudes by analysing at least one object on the frame by both methods. The rest of the measurements can then be suitably scaled to give total magnitudes.

It is always a good idea to run the isophotal analysis on a frame before running any of the other options to check that there are no peculiarities, and in the case of deblending to check that the objects do not fragment too much. If the data is significantly oversampled (the point-spread function covers many pixels) then the data should be binned into a smaller array. There will be no significant loss of accuracy, but a great improvement in execution time.

A full list of the PISAFIND parameters is given in appendix A which should be consulted before running the application.

4.1 Pre-processing of the image

PISAFIND expects there to be no background variation in the image array. Any such variations should be removed in advance with a suitable application, such as SURFIT in KAPPA [6]. The frame should also be clean of any defects and bad pixels.

4.2 Restriction on input data

Currently PISAFIND works with INTEGER data in the range 0 to 32766. An array of real data in this range will be accepted by the routine but the following message will appear:

```
Input NDF is of type _REAL - this application can only process at
_WORD precision; significance may be lost
```

If the array contains data outside this range the program will abort with the following message:

```
! The input NDF contains "bad" pixels or values outside the range of
  _WORD, these cannot be correctly handled by this application.
```

A previous restriction on the size of the input data has been removed, however, other limitations still apply. The maximum length of the *first* dimension of any input data is 10240 pixels. Objects cannot fragment into more than 200 pieces.

4.3 Content of the results files

The results of the parameterisation analysis are written into two files.

The first file name is defined by the RESULTS parameter. There are eleven columns in the output file containing the following information :-

Column	Name	Description
1	INDEX	Index number of object.
2	XPOS	X position of object in pixels.
3	YPOS	Y position of object in pixels.
4	INTENSITY	Integrated intensity of object.
5	NPIX	Number of pixels above threshold.
6	PEAK	Peak intensity of object in one pixel.
7	ELLIPT	Ellipticity of object.
8	ANGLE	Orientation of object, anti-clockwise from y-axis.
9	SXX	Second moment of data in x.
10	SYY	Second moment of data in y.
11	SXY	Cross moment of data in x and y.

The SXX, SYY and SXY moments are defined as:

$$SXX = \frac{\sum x^2 I_i}{\sum I_i},\ SYY = \frac{\sum y^2 I_i}{\sum I_i},\ SXY = \frac{\sum xy I_i}{\sum I_i}$$

Where x and y are offsets from the centre of the object (determined by the centroid) and I_i is the intensity in a pixel, corrected for the background contribution.

The ellipticity is defined by the equation:

$$ell = \frac{a - b}{a}$$

Where a and b are the semimajor and semiminor axes:

$$a^2 = 2\,(SXX + SYY) + 2\,\sqrt{(SXX - SYY)^2 + 4\,SXY^2}$$

$$b^2 = 2\,(SXX + SYY) - 2\,\sqrt{(SXX - SYY)^2 + 4\,SXY^2}$$

Note that these are intensity weighted rms-like distances. If you want to calculate values that go out to say the detection isophote then you should use the geometric formulae:

$$a = \sqrt{\frac{NPIX}{(\pi\,(1 - ELLIPT))}}$$

$$b = a\,(1 - ELLIPT)$$

The results of the areal thresholding analysis are written to the file whose name is given by the SIZES parameter. There are nine columns in the output file, the last seven contain the number of pixels within the intensity thresholds : -

$$I_i = I_t * 2^{(i+2)},\ i = 2, 8$$

where I_t is the threshold intensity and I_i is the object intensity above the threshold :-

Column	Name	Description
1	INDEX	Index number of object.
2	A1	Number of object pixels within threshold.
3	A2	Number of object pixels within i=2 threshold.
4	A3	Number of object pixels within i=3 threshold.
5	A4	Number of object pixels within i=4 threshold.
6	A5	Number of object pixels within i=5 threshold.
7	A6	Number of object pixels within i=6 threshold.
8	A7	Number of object pixels within i=7 threshold.
9	A8	Number of object pixels within i=8 threshold.

The RESULTS output of PISAFIND can be used as input to the aperture photometry program PHOTOM [4]. This could be exploited to automate the photometry of standard stars on CCD frames.

5 Visual inspection – PISAPLOT

PISAPLOT plots the results of the PISAFIND analysis as a series of ellipses which reflect the size and shape of the objects as defined by their RESULTS parameters. The ellipses can be annotated with their index number to provide a cross-reference to the object list. The plot may be overlaid on an existing image for a direct comparison of the results.

The program uses AGI [7] the graphics database so that suitable tasks, such as the KAPPA [6] routine CURSOR, can be used to get positional information. PISAPLOT has many parameters which control such things as the line thickness, line colour, the upper and lower bounds of the plot axis, so that the output can be modified for presentation and comparison purposes.

Perhaps the most likely use of PISAPLOT is to overlay a results file over a previously displayed image. This is achieved using the OVERLAY parameter

```
# pisaplot overlay
```

If the plotting device has an overlay plane then the present contents of it may be cleared using the CLEAR parameter

```
# pisaplot overlay clear
```

The colours of the plotted ellipses are controlled using the PALNUM parameter. If the graphics device allows colour then different pen numbers may be used for successive plots using the OVERLAY option. So for instance a group of objects separated from the main results by some criterion (being stellar) may be plotted in one colour and the others in another colour. The pen colours are set by PGPLOT and are:

Pen Number	Colour
0	background colour
1	foreground colour
2	red
3	green
4	blue
5	cyan
6	magenta
7	yellow
8	orange

and so on up to pen 16.

After PISAPLOT has been run these colours can be superseded by using the KAPPA palette facilities PALDEF and PALENTRY, but note that any subsequent runs of PISAPLOT will reinstate the PGPLOT default colours so using the KAPPA facilities should be delayed until all different sets of objects have been displayed. The KAPPA palette pen numbers correspond to PALNUM values (hence the parameter name).

6 Determining a profile model – PISAFIT

PISAFIT fits the radially symmetric mixed Gaussian - Exponential - Lorentzian function (appendix B) as used by PISAFIND in its profile fitting mode. It fits objects whose *accurate* positions are given in a formatted list. The objects should be stars which are well separated on the frame and which are not saturated. The list of positions may contain the usual PISA format of identifier (an integer) followed by the X and Y positions, or may just contain the X and Y positions as returned by the KAPPA [6] CENTROID routine.

The PISA profile fitting function is described by three separate parameters; the gaussian sigma (GSIGM); the cross over point (CROSS), as a fraction of the peak intensity, from the gaussian core to an exponential wing; and the fractional mix (COMIX) of a Lorentzian to these two functions at each point. The parameters resultant from the functional fit are stored as the global values, PISA_GSIGM, PISA_CROSS and PISA_COMIX (global values are stored in the global parameter file ($HOME/adam/GLOBAL.sdf or $ADAM_USER/GLOBAL.sdf, if you've set the $ADAM_USER environment variable). These will be accessed automatically by PISAFIND, PISAGEN and PISAPEAK.

The fit produced by this routine is displayed on a graphics device, together with the residuals (derived from the variance of data contributing to each point), so that the quality of fit can be assessed. Extra data, displayed with a different point type, is shown beyond the fitting radius. After the plot and fitting statistics are shown, you are prompted about your satisfaction with the fit, if you are not happy then it is possible to re-do the fit using a different radius. This cycle can be repeated until you are satisfied. Experience shows that the first fit is rarely the best.

Different constraints can be placed on the ranges of values that the three fit parameters can take. Type APM sets up the range bounds so that a minimisation as performed by the original APM version of this routine is done. Type USER allows you to specify the ranges within which the fit parameters can vary. Type NONE frees the minimisation routine to fit the function as it can, the only restriction being that the parameters must have values greater than zero. The fit can be done using a weighted scheme (this uses the residuals displayed in the plot).

PISAFIT uses the graphics database AGI [7], so plots can be made within predefined pictures on the chosen graphics device and values can read directly using a suitable cursor program. AGI pictures can be defined by the program PICDEF in KAPPA [6].

7 Model data generation — PISAGEN & ADDNOISE

PISAGEN generates model data using the PISA profile fitting function (appendix B). The objects are formed using the positions and integrated intensities output by PISAFIND in its profile fitting mode (or any other data of the same form for the first four columns):-

- Object number
- X position
- Y position
- Integrated intensity.

Extra parameters not available in the other PISA routines allow PISAGEN output objects to be elliptical. The image output from PISAGEN can have noise added of either gaussian form or of a pseudo-poissonian type. The noise levels are recorded in the variance component of the output NDF. The resultant model image can then be subtracted from the original data and the difference frame may be inspected, for missed objects, the goodness of the profile fit etc.

PISAGEN can also perform the generation of model data for test purposes, the program accepts integrated intensities and can generate very accurate model objects by integrating the flux within each pixel. It is also possible to generate mixed frame data, i.e. data with objects of differing model parameters (and ellipticities). This is done by running PISAGEN multiply with the different parameters (with a zero background, except for one frame) and then combining the frames by addition. Noise can be added to this 'mixed' frame (which could for instance mimic a frame of stars and elliptical objects with extended wings (galaxies! although not $R^{1/4}$ ones)) by an ancillary program ADDNOISE, which is available with the PISA package.

8 Using the PISA parameters — PISAPEAK, PISAKNN, PISA2CAT & PISA2ARD

Other packages exist which can be used to perform selections, transform values, and generally analyse and categorise the PISAFIND parameterisations. With sufficient effort they may be used to classify objects. Catalogue manipulations can be done using the CURSA [5] and CATPAC[8] packages. More general plotting packages also be used to investigate the various results; PONGO [12] and SM [13] both plot data points and allow mathematical transformations of columns. Using CLUSTAN you could investigate the natural clustering of objects in parameter space (with the aim of classification in mind).

8.1 PISA data in CURSA and CATPAC

Any of the PISA output files (from PISAFIND and PISAPEAK) can be converted for use by the CURSA [5] and the CATPAC [8] packages. The easiest way to convert a PISA results file into a catalogue is to use the

```
    # pisa2cat
```

routine. CURSA and CATPAC can be used to perform simple manipulations — operations such as, sorting and subsetting according to various criteria.

8.2 Getting the PISA output files into CLUSTAN

It is possible to get PISA format data into CLUSTAN and do analyses based on the 'natural' clustering of the RESULTS. The analyses are highly dependent on the similarity measure (the distance in the parameter space) and consequently the clustering which is found is more often than not due to the measure rather than the physical attributes of the objects. In runs of CLUSTAN on the test NDF 'FRAME' in the PISA directory seem to select strongly by angle and the sign of the cross moment (SXY), and very weakly in others (ellipticity for one) which one might expect to bear more relevance to the real 'clustering'. This problem probably requires careful selection of the variables to use and may well require the production of new hybrid variables (things like intensity/peak) as performed by PISAPEAK.

A description of the entry of PISA data into CLUSTAN is given below and an analysis using this may be attempted, however, NO significance should be ascribed to the results. If you really want to use this method there is no substitute to a proper understanding.

A file called `clustan.dat` is found in $PISA_DIR, take a copy of it. It shows how to get data into CLUSTAN, to run it in 'batch' mode simply change the file names to those of your PISAFIND data files and type.

```
    # clustan < clustan.dat
```

8.3 Object classification

It should, in principle, be possible to classify objects using the PISAFIND parameterisations of a *single* frame, but, this will only be possible provided that the apparent morphology of subject objects allow it (faint noise-limited objects are unlikely to be distinguishable from stars, and you shouldn't be too disappointed when they are not).

However, given a set of distinguishable objects a classification can be performed. Before such a task can be undertaken it is first necessary to remove the intensity dependency of the parameters. This allows the 'shape' for a class of objects to remain reasonably constant. The quality 'shape', in this context, really refers to a multivariate function. Usually the only objects on any single frame which have a constant shape are the stars. So the approach adopted in PISA is to 'normalize' the PISAFIND parameterisations so that they are referenced to an ideal star. This is the first stage of classification using PISA.

8.4 Transforming PISA parameters to intensity invariant form

PISAPEAK transforms the PISAFIND RESULTS parameterisations so that the variables are intensity invariant, assuming a stellar profile. The output from PISAPEAK is intended for use in star-galaxy separation, either by applying direct cuts in variable values (PISACUT) or by discrimination analysis routines such as PISAKNN. A fit to stars on a frame can be performed by PISAFIT, which should be used prior to PISAPEAK. The output from PISAPEAK is a list of four parameters:

- A 'peakedness' measure.
- The total intensity to peak intensity ratio, normalised to the analytical function.
- The unmodified ellipticity.
- The absolute value of the intensity weighted cross moment (SXY).

The peakedness measure is the ratio of the semi-major axis of the detected object (at the detection threshold) to the radius of the analytic function which has the same peak value (averaged with the central 9 pixels). Thus it specifies how more extended the object is than a star with the same peak value. This should select strongly between galaxies and stars. Indeed very good separation can be performed by using cuts of this value.

The model profile is used to scale the object peakedness and intensity peak ratio to values around one. This only works well if the PISA profiling function is a good fit to the stars on your frame. If it is not a good fit or you cannot determine the model parameters then inaccurate values can be used, the only criterion being that the GSIGM value is about the FWHM seeing of your data (this is important for resampling the peak of the object). The results file will now contain values 'normalised' to this 'imaginary' object. The stars will still form a group of objects with similar values, although they may not be as tightly clustered as with a good fit.

8.5 Distribution free classification

PISAKNN use the results of PISAPEAK to discriminate objects into two classes. PISAKNN uses KNN (k nearest neighbours) distribution-free multivariate discrimination to classify objects into two classes. The classes are seeded by supplying two files which contain the indices of objects typical to the class in question (> 5, approximately equal numbers of each). Each object then propagates its class to the other objects on the basis of which class of the 2*k nearest neighbours (in the parameter space of the PISAPEAK results) of each of the unclassified objects is most common. This procedure is iterated until all objects are assigned and have a stable class or until a maximum number of iterations is exceeded. The results of the discrimination are written to two output files, one for each class.

KNN relies on good seed statistics as propagation is essentially linear (but remember that this is in a multivariate sense). If the seed subjects do not reasonably span the whole of the object parameter space improper incursion can occur, leading to misclassification. The classification in boundary areas between the objects will depend on the size of the nearest neighbour count, larger values will help the investigation of the 'fuzzy' areas. If a very small value is chosen then this will act almost as a thresholding (but of all the parameters not just one).

KNN has the advantage over classical discriminant analysis in that it does not relies on the classes of objects having multinomial distributions. The assumption of the normality of the objects contributing to each class relies all classes having a random spread across a particular part of parameter space. It is unlikely that this requirement can be met for small galaxy populations, although this may work well for large statistical samples.

The ellipticity is included in the analysis, however, this may not always help selection. If some smallish round galaxies are present using this variable will increase the weight of selecting them as stars. In this case it may be profitable to switch off the ellipticity. Ellipticity can be used for other purposes, say if you want a complete sample of stars, all stars will have ellipticities below a given threshold and can be selected thus. Further refinement can then be applied to the list by thresholding in peakedness to remove objects with large wings.

8.6 PISA2ARD

The PISA2ARD application converts the results file output from PISAFIND into an 'ARD description'. This description consists of a series of ellipses – one for each detected object. The ARD description can be used by suitable applications to remove or analyse the areas within the ellipses. So for instance to remove all the detected objects from a frame, convert the results file into an ARD description (with the ellipses possibly scaled by some small amount to remove outlying parts) and use an ARD masking routine, such as KAPPA ARDMASK. ARD is also used by the packages ESP [10] and CCDPACK [11].

9 List manipulation utilities — PISACUT & PISAMATCH

PISACUT separates a file of variables into two. The separation is performed by thresholding the values in a single column. The routine works on any formatted data files, applying a threshold to one of the variables. File entries with the selected variable value above or below the threshold are written out to separate files. It is intended for use in separating any PISA package results files.

PISAMATCH matches the first column of values (the object indices) of two input files. When a match is located between the indices the complete entry in the second file is copied to the output file. The routine is primarily intended for matching the indices output from PISACUT or PISAKNN to those of the original PISAFIND data file, thus the new classes can have their properties analysed, or can be displayed using PISAPLOT.

10 Examples of running the PISA software

This section shows examples of how the PISA package may be used. Most applications have more capabilities than are shown here. Refer to the appendix with the complete descriptions if the sort of task which you want to perform is not shown here.

The examples as shown are intended to be 'platform independent'. No items specific to running PISA from the C-shell or ICL are shown. All the application parameters are identical; the main differences are due to special character interpretation. In ICL the continuation character is '~' and in the C-shell '\'. No continuation character is shown in the examples. When using the C-shell double quotes around "" strings will be stripped as will vector braces [], so it is necessary to protect them by using the escape character or by using single quotes around the complete parameter value i.e. use '"a parameter string"' or \"a parameter string\" not "a parameter string", and '[1,2,3,4]' or \[1,2,3,4\] not [1,2,3,4]

10.1 Isophotal analysis with deblending

This example performs isophotal analysis with deblending of overlapped images on a frame containing a mixture of stars and galaxies. The results are then plotted on a hardcopy device.

```
# pisafind
IN - NDF containing input image > frame
 Analysing whole image
MINPIX - Minimum pixel size for images (typically 4-16) > 6
METHOD - Intensity analysis ( 0=Isophotal, 1=Total, 2=Profile ) > 0
  Estimated background level =   492.2
  Background standard deviation =     7.4
BACKGROUND - Background (global sky) value /492.17/ >
THRESH - Threshold for analysis (data units) > 11.5

  Total number of positive images = 141
  The results have been written to pisafind.dat
      and pisasize.dat

# pisaplot
RESULTS - File of PISAFIND parameterised data /@pisafind/ >
DEVICE - Name of graphics device > postscript_p
```

10.2 Plotting overlaid ellipses

Using PISAPLOT to overlay ellipses is achieved using the OVERLAY and CLEAR parameters. The image over which the plot is to be overlaid should have been displayed using the KAPPA DISPLAY or a routine which uses AGI to store its graphics information (such as PISAGREY).

```
# display in=frame mode=per percentiles=[1,99] device=xw
# pisaplot results=pisafind.dat overlay device=xw
```

If a device with a suitable overlay is used then the overlay plane may be erased prior to the plotting of the ellipses using the CLEAR parameter.

```
# pisaplot results=pisafind.dat overlay clear device=xov
```

The colour of the ellipses is controlled using the PALNUM parameter. So for instance if one had two lists of results which are separated into stars and galaxies for instance (or one could have PISAFIND results files separated by intensity, ellipticity etc.)

```
# pisaplot results=stars.dat overlay palnum=3
# pisaplot results=galaxies.dat overlay palnum=4
```

This sequence of commands would result in the data in file `stars.dat` being overlaid and coloured green. The data in file `galaxies.dat` would also be overlaid on the same image and coloured blue. The colours of these could then be altered using the KAPPA palette facilities for pens 3 and 4.

10.3 Performing profile fitting analyses

To perform a profile fit a list of the accurate positions of good well separated stars are required. One way to get such a list is too use the KAPPA CENTROID routine to select objects from a displayed image.

```
# centroid coout=stars.acc mode=cursor mark
```

The output file `stars.acc` is of a format suitable for input to PISAFIT. Running PISAFIT allows the determination of the best fit to the stars.

```
# pisafit
IN - NDF containing input image /@frame/ >
 Analysing whole image
 Estimated background level =   492.2
 Background standard deviation =     7.4
BACKGROUND - Background (global sky) value /492.2/ >
POSITIONS - File containing star positions / / > stars.acc
MINMODE - Minimisation bounds mode /'APM'/ >
RADIUS - Limiting Radius > 10
 Number of points in curve fit = 28

 May be having problems with the minimisation, check plot.
 RMS of fit                 = 2.0545775E-02
 Gaussian Sigma      (GSIGM)=  2.17
 % Cross Over        (CROSS)= 30
 Mixture Coefficient (COMIX)= 0.087
AGAIN - Do fit again /TRUE/ > f
```

The parameters controlling other various options are detailed in the routine description appendix. Usually the default options (as shown) are all that are necessary.

PISAFIT writes the model parameterisations into the application's parameter file. These values are then automatically picked up by any other PISA application which requires them. So for instance running PISAFIND in its profile fitting mode is straight-forward after running PISAFIT.

```
# pisafind
IN - NDF containing input image /@frame/ >
 Analysing whole image
MINPIX - Minimum pixel size for images (typically 4-16) / / > 4
METHOD - Intensity analysis ( 0=Isophotal, 1=Total, 2=Profile ) /0/ > 2
GSIGM - Gaussian sigma (pixels) /2.17146/ >
CROSS - Crossover point % of peak /30/ >
COMIX - Mixture coefficient /8.7035753E-02/ >
UPLIM - Upper intensity limit to use in analysis (saturation) /32767/ >
IFULL - Do you want full surface modelling /FALSE/ >
 Estimated background level =   492.2
 Background standard deviation =     7.4
BACKGROUND - Background (global sky) value /492.17/ >
THRESH - Threshold for analysis (data units) /20/ >

 Total number of positive images = 119
 The results have been written to pisafind.dat
     and pisasize.dat
```

10.4 Fancy plots

PISAPLOT allows the use of PGPLOT text escape sequences. It also has parameters controlling the size of the annotations, the presence of annotations, the number of tick mark and the thickness of the lines. So for instance the plot on the front page of this document was produced using the commands.

```
 # pisafind frame(80:210,115:260) reset accept
 # pisaplot device=pscript_p pltitl="\fr Objects located by PISAFIND"
           abslab="\fr X position (pixels)"
           ordlab="\fr Y position (pixels)" thick=2 annoscale=2.5
```

Hence the labels were written using a Roman font.

10.5 Simple object classification

PISA contains routines which when used co-operatively can separate objects into lists of stars and 'other' objects. If the objects have been located using PISAFIND and a suitable profile fit has been made using PISAFIT, then using PISAPEAK produces a set of 'intensity invariant' measures. Basically what we're looking for here are some transformations of the original PISAFIND RESULTS measurements which are not different simply because the objects are of different apparent brightnesses. The transformations chosen try to have the same values for a star of any brightness. Try the following sequence of commands.

```
# pisapeak in=frame finddata=pisafind.dat results=pisapeak.dat
# pisacut input=pisapeak.dat column=2 thresh=0.85 lower=stars.indices
         higher=gals.indices
# pisamatch one=stars.indices two=pisafind.dat out=stars.dat
# pisamatch one=gals1.indices two=pisafind.dat out=gals.dat
```

This example transforms the PISAFIND RESULTS file using a model stellar profile fit, writing the results to `pisapeak.dat`. PISACUT is then used to separate the PISAPEAK results, thresholding the data using a cut of 0.85[1] in the peakedness measure (this strongly selects for stars). The indices of the objects in file `stars.indices` and `gals.indices` are then matched against those of the original PISAFIND RESULTS file, giving two files which contain the PISAFIND RESULTS for the separated objects.

[1]The value 0.85 is decided basically by trial and error. Ideally objects with peakedness measure less than 1 would always be stars, however, the actual value depends strongly on the quality of the profile fit.

10.6 Comprehensive example

A more comprehensive example of PISA usage `‘pisa_demo’` can be found in the pisa directory ($PISA_DIR). From the C-shell `pisa_demo` is defined as a command so just run it as any of the applications. From ICL the `pisa_demo` procedure needs to be loaded before running, this is achieved just by typing the procedure name `pisa_demo` followed by the invocation `pisa_demo ‘device_name’`.

11 Acknowledgements

Thanks go to Mike Irwin and Steve Maddox for explaining some of the secrets of the APM IMAGES package, and to Malcolm Currie for supplying the original version of PISAPLOT.

12 References

1 — SG/4: ADAM - The Starlink Software Environment.

2 — Irwin, M.J., 1985, Automatic analysis of crowded fields, Mon.Not.R.astr.Soc., **214**, 575-604.

3 — SUN/42: DAOPHOT - Stellar Photometry Package.

4 — SUN/45: PHOTOM - An aperture photometry routine.

5 — SUN/190: CURSA - Catalogue and Table Manipulations Applications

6 — SUN/95: KAPPA - Kernel Application Package.

7 — SUN/48: AGI - Applications Graphics Interface Programmers Guide.

8 — SUN/120: CATPAC - Catalogue Applications Package.

9 — SUN/15: PGPLOT - Graphics Subroutine Library.

10 — SUN/180: ESP - The extended surface photometry package.

11 — SUN/139: CCDPACK - CCD data reduction package.

12 — SUN/137: PONGO - A Set of Applications for Interactive Data Plotting.

13 — MUD/159/160: SM - interactive graphics package.

A Full routine descriptions

ADDNOISE
Adds noise to model data

Description:

ADDNOISE, adds noise to model data. The noise can be either poissonian or gaussian. If gaussian a constant noise level, described by a given standard deviation, is introduced. If poissonian the data is scaled by a factor to change data values into counts. The counts are then used as estimates of the mean value in that pixel and noise is added on this basis. Note that the poissonian noise is pseudo gaussian and so the count levels in the frame need to be greater than 10.

Each time this program is started a different set of random numbers should be generated.

Usage:

$$\texttt{ADDNOISE IN NOISE OUT} \quad \underbrace{\left\{\begin{array}{l}\texttt{ADU=?}\\ \texttt{SIGMA=?}\end{array}\right.}_{\texttt{noise}}$$

Parameters:

IN = NDF (Read)
The input NDF containing the data to which noise needs to be added.

NOISE = LITERAL (Read)
The noise type to introduce into the data. Either Gaussian or Poissonian, which may be abbreviated to G and P. [P]

ADU = REAL (Read)
The scaling factor to convert the data values in the input NDF to counts for which Poisson statistics are assumed valid. [1.0]

SIGMA = REAL (Read)
The standard deviation of the gaussian noise. [1.0]

OUT = NDF (Write)
The output NDF to contain the data with noise added.

Examples:

```
ADDNOISE IN=MODEL NOISE=G SIGMA=20 OUT=MODEL_WITH_NOISE
```

This adds gaussian noise to the input NDF model. The noise has a standard deviation of 20 units.

```
ADDNOISE IN=MODEL NOISE=P ADU=10 OUT=MODEL_WITH_NOISE
```

This adds poissonian noise to the input NDF model. The data values are scaled by a factor of 10 before the noise is calculated.

PISA2ARD
Creates an ARD description of detected objects

Description:

This routine converts the RESULTS output from PISAFIND into an ARD description. The ARD description consists of a series of ellipses, one for each detected object, which can be used to identify the area of the images on a frame.

The scale of the ellipses can be changed allowing the effective areas to be modified.

Usage:

```
PISA2ARD RESULTS ARDFILE SCALE
```

Parameters:

ARDFILE = LITERAL (Read)

The name of a text file to contain the ARD description. [PISA2ARD.DAT]

RESULTS = LITERAL (Read)

The name of the file containing the RESULTS from a run of the PISAFIND application. [PISAFIND.DAT]

SCALE = _REAL (Read)

The scale factor to apply to the ellipse major and minor axes. This value must be in the range 0.01 to 100.0. [1.0]

Examples:

```
PISA2ARD PISAFIND.DAT PISA2ARD.DAT 1.0
```

In this example an ARD description is written to file PISA2ARD.DAT. The ellipses cover exactly the same area as the ellipses fitted to the detected objects.

```
PISA2ARD PISAFIND.DAT PISA2ARD.DAT 2.0
```

In this example an ARD description is written to file PISA2ARD.DAT. The ellipses have major and minor axes which are twice as long as those which fitted the detected objects. This gives four times as much area.

Notes:

- The input to the parameter RESULTS must have the same format as the output from the PISAFIND parameter RESULTS.

PISA2CAT
Converts a PISA formatted data file into a catalogue

Description:

PISA2CAT converts PISAFIND and PISAPEAK results files into catalogues. The output catalogues can be used by CURSA or CATPAC applications.

Usage:

```
pisa2cat datatype data cat
```

Parameters:

DATATYPE = LITERAL (Read)
The type of PISA data. Either FIND, SIZE or PEAK. FIND indicates that the input data is a results file produced by PISAFIND (RESULTS parameter) and that it contains the main object parameterisations (ie. position, total intensity, peak intensity etc.). SIZE indicates that the input file is a results file from PISAFIND, but which contains the pixel sums within different intensity thresholds (from the SIZES parameter). Finally PEAK indicates that the input is a results file from the PISAPEAK routine. [FIND]

DATA = FILENAME (Read)
Name of the file containing the PISA results to be converted into a catalogue. [PISAFIND.DAT]

CAT = FILENAME (Write)
Name of the output catalogue. [FIND]

Examples:

```
pisa2cat find pisafind.dat find
```

This example converts the results file containing the object parameterisations from PISAFIND to FITS table format. It writes the results to the file find.FIT

```
pisa2cat peak pisapeak.dat peak.sdf
```

This example converts the output from a run of PISAPEAK into an HDS catalogue that can be used with the CATPAC applications.

Notes:

- The output format of the catalogue can be manipulated by changing the file extension. Without a file extension a binary FITS table is produced. If you add a ".sdf" extension then an HDS catalogue that can be used by CATPAC will be created.

Column_names :

The column names used in the output catalogues are.

- DATATYPE = FIND
  INDEX XPOS YPOS INTENSITY NPIX PEAK ELLIPT ANGLE SXX SYY SXY
- DATATYPE = SIZE
  INDEX A1 A2 A3 A4 A5 A6 A7 A8
- DATATYPE = PEAK
  INDEX RRATIO IRATIO ELLIP ABSSXY

PISACUT
Separates a formatted file into two parts

Description:

This routine works on any formatted data files, applying a threshold to one of the variables. The variable is specified by a column number. File entries with variable value above and below the threshold are written out to separate files.

Usage:

```
PISACUT INPUT COLUMN THRESH LOWER HIGHER
```

Parameters:

INPUT = FILENAME (Read)
Name of a file containing the data to be separated by applying a threshold value to the variable in the specified column.

COLUMN = _INTEGER (Read)
The column which contains the data which is to be thresholded. [1]

THRESH = _REAL (Read)
The threshold value for variable in the given column. Entries with this variable value above the threshold will be entered to file HIGHER. Entries with a value less than or equal to this variable value will be entered into file LOWER. [0.0]

LOWER = FILENAME (Write)
Name of a file to contain the entries with selected variable value less than equal to the threshold.

HIGHER = FILENAME (Write)
Name of a file to contain the entries with selected variable value greater than the threshold.

Examples:

```
PISACUT INPUT=PISAPEAK.DAT COLUMN=2 THRESH=1.5 LOWER=STARS HIGHER=GALS
```

This separates the results from a run of the PISAPEAK program, whose results are stored in file PISAPEAK.DAT, into two different files STARS and GALS. The variable selected is the one found in column 2 (the object peakedness). Entries, in file PISAPEAK, with the variable value in column 2 greater than 1.5 are written into file GALS, those with selected variable value less than equal to 1.5 are written to file STARS.

PISAFIND
Locate and parameterise objects on an image frame

Description:

PISAFIND performs image analysis on a 2-dimensional data frame. The program searches the data array for objects that have a minimum number of connected pixels above a given threshold and extracts the image parameters (position, intensity, shape) for each object. The image parameters can be determined using thresholding techniques or an analytical stellar profile can be used to fit the objects. In crowded regions deblending of overlapping sources can be performed.

It is important the frame supplied to the program is clean of all defects and has a flat background. The data values should be in the range 0 to 32766.

Parameters:

BACKGROUND = _REAL (Read)

The actual background value to be used. The intensities used in any analysis are background subtracted using this value. [Dynamic]

COMIX = _REAL (Read)

The mixture coefficient, as a fraction of the Gaussian peak, of a Lorentzian function use to model the wings of the stellar profile. At each point in the analytical profile the Lorentzian function is added to the Gaussian/exponential core scaled by the mixture coefficient. The profile is used to fit the data and the intensity is obtained by integrating under the profile. [Global]

CROSS = _REAL (Read)

The crossover point, as a percentage of the Gaussian peak, where an exponential fall-off in the analytical stellar profile takes over from the Gaussian core. The exponential function is joined on smoothly to the Gaussian. The profile is used to fit the data and the intensity is obtained by integrating under the profile. [Global]

DEBLEND = _LOGICAL (Read)

Separate overlapping images or not.

If this is true then all objects which are connected at the threshold level are examined to see if they comprise more than one object at a higher level. If this is so the signal in the object is apportioned between the components and the parameters calculated accordingly.

If this is false the image parameters are calculated for those objects which are connected at the threshold level. [TRUE]

GSIGM = _REAL (Read)

The standard deviation, in pixels, of the Gaussian core of the analytical stellar profile. The profile is used to fit the data and the intensity is obtained by integrating under the profile. [Dynamic]

IBREF = _LOGICAL (Read)

Perform background refinement or not if full surface modelling has been requested.

If this is true then during the full surface modelling procedure the background is allowed to vary, over a blend of objects, in addition to the object's position and intensity, to best fit the data.

If this is false then the background is held constant during the full surface modelling procedure. [FALSE]

ICIRC = _LOGICAL (Read)

The total intensity is to be estimated from a circular aperture of fixed size, or from an asymptotic curve of growth analysis.

If this is true a fixed sized circular aperture is used. If this is false the total intensity is estimated from the curve of growth using elliptical apertures that match the shape of the objects. [FALSE]

IFULL = _LOGICAL (Read)

Perform full surface modelling or not when analysing profile intensities.

If this is true the program examines a difference map of the raw data minus the fitted profiles to see if any other objects are present. The fitting procedure is repeated until a stable solution is reached. This option is very time consuming.

If this is false the program uses the images found from an isophotal analysis as a basis for the profile fitting, and does not examine the difference map for additional objects. Some objects may be merged together if they are too close, or if the fit is better in a least squares sense. [FALSE]

IMNEG = _LOGICAL (Read)

Search for negative going images or not.

If this is true then negative going images (those below the background level) will be searched for, as well as the usual positive going images. The same threshold and minimum connectivity is used for both directions. If the negative going images are assumed to be due to random fluctuations in the background then the number of negative images will indicate the proportion of false detections that may be present amongst the positive going images. Only isophotal analysis is applied to the negative images. Negative going images are indicated in the results file with minus signs in the first field.

If this is false then only positive going images are sought. [FALSE]

IN = NDF (Read)

An NDF data structure containing the 2-dimensional image on which the image analysis will be performed.

ISMOO = _LOGICAL (Read)

Smooth the data before searching for objects with the isophotal analysis or not.

If this is true then the data is smoothed with a 3 x 3 Hanning filter before the isophotal analysis is performed. This is useful if the deblending is fragmenting the data too much. The object parameters are calculated using the unsmoothed data.

If this is false the raw data is used for the object searches. [FALSE]

METHOD = _INTEGER (Read)

The type of intensity analysis to be performed on the objects. This is an integer code which can be one of the following choices :-

0 — Isophotal intensities. The intensity above the threshold is summed for each object.

1 — Total intensities. For each object the total intensity is estimated. This is done by one of two methods: A circular aperture of fixed size is specified and the intensity is summed within it. Otherwise the program automatically estimates the total light within an object using an asymptotic curve of growth analysis with elliptical apertures that match the shape of the object.

2 — Profile intensities. This uses an analytical stellar profile to fit the individual objects. All pixels above the threshold are used in the fit. The fit is made in the least-squares sense and the intensity is obtained from the integrated profile.

MINPIX = _INTEGER (Read)

Minimum number of connected pixels an object needs to have to qualify for further analysis. Only those pixels with data values above the specified threshold are considered.

PRALL = _LOGICAL (Read)

Analyse the whole array or select a subset of the data.

If this is true the whole data array is analysed. If this is false the pixel coordinates defining a subset of the data to be analysed is requested. [TRUE]

RCIRC = _REAL (Read)
The radius in pixels of the circular aperture to be used for the total intensity analysis.

RESULTS = FILENAME (Write)
The name of the file to receive the results of the analysis. [PISAFIND.DAT]

SIZES = FILENAME (Write)
The name of the file to receive the areal sum results. Note that -1. indicates that no measurement has been made either because the object is part of a blend, or because the profiling analysis has been requested. [PISASIZE.DAT]

THRESH = _REAL (Read)
The threshold above which a pixel is considered as a potential member of a larger object. An object has to contain a minimum number of connected pixels above this threshold to be accepted. The threshold is defined in data units above the background level.

UPLIM = _INTEGER (Read)
The upper limit, in data units, for a pixel to be included in the profile fitting procedure. If the stellar profiles are saturated then this can be used to constrain the profile fitting to the unsaturated pixels. [Dynamic]

XPIXS = _INTEGER (Read)
Initial and final pixel coordinates of the subset of the data array to be analysed. [Dynamic]

YPIXS = _INTEGER (Read)
Initial and final pixel coordinates of the subset of the data array to be analysed. [Dynamic]

Examples:

```
PISAFIND ARP199
```

Performs image analysis on the 2-dimensional NDF data structure ARP199.

```
PISAFIND PRALL=F
```

Requests that a sub-section of the input data array is analysed.

```
PISAFIND ISMOO=T
```

Smooths the data with a Hanning filter before searching for objects with the isophotal analysis.

Notes:

- The data input to PISAFIND can be any size (within system limits), however, the length of the first dimension is restricted to less than 10241 pixels.
- The output in the RESULTS file contains the following information:

Column	Name	Description
1	INDEX	Index number of object.
2	XPOS	X position of object in pixels.
3	YPOS	Y position of object in pixels.
4	INTENSITY	Integrated intensity of object.
5	NPIX	Number of pixels above threshold.
6	PEAK	Peak intensity of object in one pixel.
7	ELLIPT	Ellipticity of object.
8	ANGLE	Orientation of object, anti-clockwise from y-axis.
9	SXX	Second moment of data in x.
10	SYY	Second moment of data in y.
11	SXY	Cross moment of data in x and y.

- The output in the RESULTS file contains the following information:

Column	Name	Description
1	INDEX	Index number of object.
2	A1	Number of object pixels within threshold.
3	A2	Number of object pixels within i=2 threshold.
4	A3	Number of object pixels within i=3 threshold.
5	A4	Number of object pixels within i=4 threshold.
6	A5	Number of object pixels within i=5 threshold.
7	A6	Number of object pixels within i=6 threshold.
8	A7	Number of object pixels within i=7 threshold.
9	A8	Number of object pixels within i=8 threshold.

The SIZES thresholds are determined by the equation.

$$I_i = I_t + 2^{(i+2)},\ i = 2,8$$

where I_t is the threshold intensity.

PISAFIT
Fits a mixed Gaussian - Exponential - Lorentzian profile to STELLAR images

Description:

PISAFIT fits a radially symmetric mixed Gaussian - Exponential - Lorentzian function to STELLAR type objects. The function is described by three separate parameters, the gaussian sigma, the cross over point, as a fraction of the peak intensity, from the gaussian core to an exponential wing and the fractional mix of a Lorentzian to these two functions at each point. The parameters describing the resultant functional fit are stored in the GLOBAL file and can be subsequently accessed by PISAFIND if the profile fitting option is chosen, or by PISAGEN for generating a model data frame of the detected objects. The fit produced by this routine is displayed on a graphics device, together with the residuals (derived from the variance of data contributing to each point), so that the quality of fit can be assessed. The user is prompted as to their satisfaction with the displayed fit, and can re-do the fit, out to a specified radius. This cycle can repeat until the user indicates that he is satisfied with the fit. Experience shows that the first fit is rarely the best.

The fit parameters can be controlled, within certain limitations, by the user. Three different forms of minimisation parameter bounding are available. Type APM sets up the bounds so that a minimisation as used in the original APM specification of this routine can be performed. Type USER allows the user to directly specify the ranges within which the fitting parameters can vary, and type NONE freely allows the minimisation routine to fit the function with the only restriction that the values are greater than zero. Finally the user can specify that the fit is done using a weighted scheme, using the shown residuals.

Usage:

```
PISAFIT IN POSITIONS [DEVICE] [LINEW] [MINMODE] AGAIN=?  RADIUS=?
```

Parameters:

AGAIN = _LOGICAL (Read)
Controls whether another refinement of the fit takes place or not. [TRUE]

BACKGROUND = _REAL (Read)
The background (sky) value. Used across whole frame. [Dynamic]

COMIX = _REAL (Write)
The value found for the mixture ratio between the Lorentzian and the other functions, on exit from the program.

COMIXRANGE = _REAL (Read)
The range of values between which COMIX is allowed to vary during the minimisation. Only used if MINMODE = USER. [0,1]

CROSS = _REAL (Write)
The value found for the percentage cross over point, from the gaussian core to the exponential wings, on exit from the program.

CROSSRANGE = _REAL (Read)
The range of values between which CROSS is allowed to vary during the minimisation. Only used if MINMODE = USER. [0,100]

DEVICE = DEVICE (Read)
A character string specifying a valid device name.

GSIGM = _REAL (Write)
The value found for the gaussian sigma on exit from the program.

GSIGMRANGE = _REAL (Read)
The range of values between which GSIGM is allowed to vary during the minimisation. Only used if MINMODE = USER. [0.5,5,5]

IN = NDF (Read)
The Input NDF containing the objects to be fitted.

LINEW = _INTEGER (Read)
The relative width of the lines plotted (greater than equal to 1). [1]

MINMODE = LITERAL (Read)
String defining which type of bounds are to be applied to the minimisation parameters. The allowed returns are any string beginning with the characters `'A'`,`'U'` or `'N'`. These represent `'A'`PM (default) `'U'`SER or `'N'`ONE. If return is APM then the bounds for the minimisation are as in the original APM version of this routine (0.5 to 5.5 for GSIGM, 0.0 to 1.0 for CROSS and 0.0 to approximately 0.12 for COMIX). If the return selects USER then the user will be prompted for the limitations to be applied during the minimisation. If the returns selects NONE then the parameters will be allowed to vary to any value greater than 0.0. [APM]

POSITIONS = FILENAME (Read)
ASCII file containing the positions of the objects to be fitted. The file may be of the type produced by PISAFIND or just of list of either

Object-number X-position Y-position etc.,

or

X-position Y-position.

[PISAFIND.DAT]

PRALL = _LOGICAL (Read)
Analyse the whole array (to derive the background value) or select a subset of the data.

If this is true the whole data array is analysed. If this is false the pixel coordinates defining a subset of the data to be analysed are requested. [TRUE]

RADIUS = _REAL (Read)
The radius out to which the fit will be attempted. This should be at least a few stellar radii. [10.0]

WEIGHTED = _LOGICAL
Set to true if a weighted fit is to be attempted. The weighting is based on the distribution of errors of values contributing to a bin in the mean profile. Note that under certain conditions weighting can seem to bias fit unfairly to the central bins (usually seen as a very poor fit to the outer values). If this situation occurs then an unweighted fit is the best option. [FALSE]

XPIXS = _INTEGER (Read)
Initial and final pixel coordinates of the subset of the data array to be analysed. [Dynamic]

YPIXS = _INTEGER (Read)
Initial and final pixel coordinates of the subset of the data array to be analysed. [Dynamic]

Examples:

```
PISAFIT FRAME FRAME.STARS_ACC CANON_PORTRAIT 3 NONE AGAIN=F RADIUS=15
```

Performs the radial fit on the stars contained in the list FRAME.STARS_ACC. It directs the graphical output to a laser printer whose lines are printed at three times normal density. The bounds on the minimisation are freed to be any value greater than 0. The interactive, recursive, fitting loop is disabled by setting AGAIN=F and the fit is done out to a radius of 15 pixels.

```
PISAFIT FRAME FRAME.STARS_ACC
```

Performs the production of the radial profile. Enters into an interactive session allowing the user to modify the radius of the data used in the fit.

Notes:

The potential quality of the fit is very dependent on the objects chosen to produce the radial profile. The best results should be obtained from a range of unsaturated, well separated objects. The list of objects used for the fit can be passed directly from PISAFIND or by a list of object x,y positions produced by the user, say by using the KAPPA routine CENTROID. Note that any input lists of the latter type should be of the form object_number, X_position, Y_position in the first three columns or X_position, Y_position) only. This means that output from many KAPPA routines will require editing.

Timing :

The timing for the initial processing (i.e. radial profile production) is proportional to the number of objects given. The latter processing (the fitting loop) is dependent only on the number of points in the fit .

Functional Forms :

The basic functional forms of the fitting equations are described in appendix B.

Implementation Status:

The present status of the program has 4 main drawbacks.

- The data must lie within the range of signed word reasonably approximated by integers.
- There is no bad pixel handling.
- The input arrays cannot be any larger than 10240 pixels in any axis.
- A maximum of 1000 input objects is allowed.

PISAGEN
Generates a frame of model objects

Description:

PISAGEN generates objects using the PISA profile model function. Input data are a list which should have been generated by PISAFIND in the profiling mode, or be of the same general form, for at least the first four columns (i.e. index, X position, Y position and integrated intensity). An optional input frame can be given, this is used to determine the size of the output frame. If this is not given then the input list is scanned to generate a suitable default size for the output frame. The output objects from PISAGEN can have the flux within each pixel integrated to give a improved estimate of the actual flux.

The PISA profiling function model parameters can be read from GLOBAL parameters as set up by the program PISAFIT. Additional parameters allow the output stars to be elliptical. The model may be generated on a background value and can be saturated. Noise can be added to the data to simulate real data for test purposes.

Usage:

```
PISAGEN INPUT POSITIONS OUTPUT GSIGM CROSS COMIX NOISE
     ⎧ ADU=?
     ⎨
     ⎩ SIGMA=?
  method
```

Parameters:

ADU = REAL (Read)
The scaling factor to convert the data output values into counts for which Poisson statistics are required. This is the analogue to data units conversion factor for unmodified CCD frames. [1.0]

ANGLE = _REAL (Read)
The positional angle of the major axis of the stellar ellipses measured in degrees anti-clockwise from the y-axis (ie. as per PISAFIND). [0.0]

BACKGROUND = _REAL (Read)
The background value for the output frame. [0.0]

COMIX = _REAL (Read)
The mixture coefficient, as a fraction, which defines how the Lorentzian function, at each point in the analytical profile is added to the Gaussian or Exponential. [Global]

CROSS = _REAL (Read)
The crossover point, as a percentage of the Gaussian peak, where an exponential fall-off in the analytical stellar profile takes over from the Gaussian core. The exponential function is joined on smoothly to the Gaussian. [Global]

ELLIP = _REAL (Read)
The ellipticity of the output objects. [0.0]

GSIGM = _REAL (Read)
The standard deviation, in pixels, of the Gaussian core of the analytical stellar profile. [Global]

INPUT = NDF (Read)
An NDF containing a data array which is the same size as the required output data array.

NOISE = LITERAL (Read)
The noise contribution to be added to output data. Can be Poisson, Gaussian or None. [NONE]

NSIGMA = _REAL (Read)
The number of gaussian sigma to generate the output objects to. [10.0]

OUTPUT = NDF (Write)
The output NDF to contain the model objects.

POSITIONS = FILENAME (Read)
Input list of star positions and intensities (see description). [PISAFIND.DAT]

SAT = _REAL (Read)
The saturation value for output stars. [32767]

SCALE = _INTEGER (Read)
if scale is greater than one then each output object pixel value is determined by the integration of the function values at scale∗scale equidistant parts within the pixel. This simulates an integration of the flux within each pixel. Using this option increases the accuracy of the output pixel intensities, this may be important in areas with large intensity gradients. [Dynamic]

SIGMA = REAL (Read)
The standard deviation of the Gaussian noise contribution. [1.0]

TITLE = LITERAL (Read)
Title for the output frame. [Output from PISAGEN]

XHIGH = _INTEGER (Read)
If an input frame is not given then this value defines the upper limit of the output data array X-axis. [Dynamic]

XLOW = _INTEGER (Read)
If an input frame is not given then this value defines the origin of the output data array X-axis. [Dynamic]

YHIGH = _INTEGER (Read)
If an input frame is not given then this value defines the upper limit of the output data array Y-axis. [Dynamic]

YLOW = _INTEGER (Read)
If an input frame is not given then this value defines the origin of the output data array Y-axis. [Dynamic]

Examples:

```
PISAGEN INPUT=FRAME POSITIONS=FRAME.STARS_ACC OUTPUT=STARS BACKGROUND=100.0
SAT=32767 NOISE=NONE
```

Generates an output NDF STARS of the same size as FRAME. The objects are placed at the positions defined in FRAME.STARS_ACC. The model parameterisations are accessed as the global parameters PISA_GSIGM, PISA_CROSS and PISA_COMIX, these values were probably generated by PISAFIT.

Implementation Status:

This program will not handle bad pixels.

PISAGREY
Plots an NDF as a greyscale

Description:

This routine displays a greyscale representation of the data component of an NDF.

Parameters:

ABSLAB = LITERAL (Read)
Label for the plot abscissa, may include PGPLOT escape sequences. This parameter is only used when the axes option is selected. [X]

AXES = _LOGICAL (Read)
True if annotated axes are to be drawn around the displayed image. This parameter is ignored in the overlay mode, since there is no guarantee that the axes would lie entirely within the current picture. [TRUE]

DEVICE = DEVICE (Read)
The name of the graphics device on which to plot the map of images found. If the overlay mode is required it is recommended that the image be displayed in KAPPA on an image display's base plane, then run this application using the device's overlay plane. [Current graphics device]

MAJTIC(2) = _REAL (Read)
The parameter controlling the numbers of major tick marks for the x and y axes. A negative value for an axis makes the graphics package decide an appropriate value. This parameter is only used when the axes option is selected. [-1.,-1.]

MINTIC(2) = _INTEGER (Read)
The number of minor tick marks between each major tick mark for the x and y axes. A negative value forces the graphics package to compute appropriate values. This parameter is only used when the axes option is selected. [-1.,-1.]

ORDLAB = LITERAL (Read)
Label for the plot ordinate, may include PGPLOT escape sequences. This parameter is only used when the axes option is selected. [Y]

OUTTIC = _LOGICAL (Read)
True if the axis tick marks are to appear on the outside of the axes instead of inside. This parameter is only used when the axes option is selected. [TRUE]

PLTITL = CHAR (Read)
The title of the plot, may include PGPLOT escape sequences. Up to about 40 characters can be accommodated. This parameter is only used when axes option is selected. [PISAGREY]

XPIXS = _REAL (Read)
Initial and final pixel coordinates of plot in x-direction.

YPIXS = _REAL (Read)
Initial and final pixel coordinates of plot in y-direction.

PISAKNN
Uses the results of PISAPEAK to discriminate objects into two classes

Description:

PISAKNN uses KNN (k nearest neighbours) distribution-free multivariate discrimination to classify objects into two classes. The classes are seeded by supplying two files which contain the indices of objects typical to the class in question (>5, approximately equal numbers of each). Each object then propagates its class to the other objects on the basis of which class of the 2∗k nearest neighbours (in the parameter space of the PISAPEAK results) of each of the unclassified objects is most common. This procedure is iterated until all objects are assigned and have a stable class or until a maximum number of iterations is exceeded. The results of the discrimination are written into two output files, one for each class.

Usage:

```
PISAKNN PEAKDATA SEED1 SEED2 K CLASS1 CLASS2 NITER
```

Parameters:

CLASS1 = FILENAME (Write)
Name of a file to contain the indices of the objects selected for membership of class 1. [CLASS1.DAT]

CLASS2 = FILENAME (Write)
Name of a file to contain the indices of the objects selected for membership of class 2. [CLASS2.DAT]

ELLIP = _LOGICAL (Read)
If 'true' then the ellipticities are used in the analysis. If 'false' then they are excluded. Using ellipticities may increase the weighting of some (small) round galaxies as stars. [TRUE]

K = _INTEGER (Read)
The number of nearest neighbours about the current values which are to be used in classifying an object. The class used is the most frequently encountered in this range of objects. If classes 1 and 2 are equally frequent then the object classification is not changed. [1]

NITER = _INTEGER (Read)
The maximum number of iterations allowed to classify and reclassify objects. [10]

PEAKDATA = FILENAME (Read)
Name of a file containing the results of the PISAPEAK parameter transformation. This file must contain at least five columns which have the values:

- object index
- radius ratio
- intensity-peak ratio
- ellipticity
- absolute value of intensity weighted cross moment

in that order. [PISAPEAK.DAT]

SEED1 = FILENAME (Read)
Name of a file containing the indices of the objects to seed class1. The file can contain any number of columns but must have the object indices in column one. [SEED1.DAT]

SEED2 = FILENAME (Read)
Name of a file containing the indices of the objects to seed class2. The file can contain any number of columns but must have the object indices in column one. [SEED2.DAT]

Examples:

```
PISAKNN PISAPEAK S1 S2 3 C1 C2 5
```

This performs a KNN analysis on file PISAPEAK, using the indices in files S1 and S2 as seeds for classes 1 and 2 respectively. The new classifications are assigned using the nearest 6 neighbours (2K). The maximum number of iterations allowed is 5. After the maximum number of iterations is exceeded or the classifications become stable the indices of the class 1 objects are written to file C1 and class 2 to C2.

Notes:

- The seed objects are always returned in their initial classes.
- The maximum number of objects allowed in any input file is 10000

PISAMATCH
Matches the indices in one file against those in a second file

Description:

This routine matches the first column of values (the object indices) of the two input files. When a match is located between the indices the complete entry in the second file is copied to the output file. It is intended primarily to match a list of object indices (such as those produced by PISAKNN or possibly a PISACUT list of PISAPEAK results).

Usage:

```
PISAMATCH ONE TWO OUT
```

Parameters:

ONE = FILENAME (Read)
Name of the file containing the indices to match to those of file TWO.

TWO = FILENAME (Read)
Name of the file containing the entries whose indices are to be matched to those of file ONE. Entries with a matched index will be written to file OUT.

OUT = FILENAME (Write)
Name of a file to contain the entries from file TWO which have the same indices as those specified in file ONE. [PISAMATCH.DAT]

Examples:

```
PISAMATCH CLASS1 PISAFIND STARS
```

This matches the indices in file CLASS1 to those of file PISAFIND. The PISAFIND entries with indices found in file CLASS1 are written to file STARS.

Notes:

- The maximum number of entries allowed in file ONE is 10000. No restriction applies to file TWO.
- The indices in both files must be in the first column.

PISAPEAK
Transforms the PISAFIND parameterisations so that the variables are intensity invariant

Description:

PISAPEAK reads in the object parameterisations file produced by PISAFIND. Using the PISA profiling function it then transforms certain of the variables into values which are intensity invariant for a radial stellar profile. Two of the output variables may have values of around 1.0 for objects which are well represented by the given PISA profiling parameters. The variables produced are the ratio of the semi-major axis of the object fit to the radius of an object whose peak intensity is the same, but whose size is determined by the PISA profiling function (a peakedness measure). The ratio of the integrated intensity to the peak intensity and the model object ratio. The ellipticity (unmodified) and the absolute value of the intensity weighted cross moment (ABS(SXY)).

The output from PISAPEAK can be used in star-galaxy separation, either by applying direct cuts in variable values or by discrimination analysis routines such as PISAKNN.

Usage:

```
PISAPEAK IN FINDDATA RESULTS GSIGM CROSS COMIX BACKGROUND THRESH
```

Parameters:

BACKGROUND = _REAL (Read)
The frame background value (sky) as used by PISAFIND. [Global]

COMIX = _REAL (Read)
The mixture coefficient, as a fraction of the Gaussian peak, of a Lorentzian function used to model the wings of the stellar profile. At each point in the analytical profile the Lorentzian function is added to the Gaussian/exponential core scaled by the mixture coefficient. [Global]

CROSS = _REAL (Read)
The crossover point, as a percentage of the Gaussian peak, where an exponential fall-off in the analytical stellar profile takes over from the Gaussian core. The exponential function is joined on smoothly to the Gaussian. [Global]

FINDDATA = FILENAME (Read)
Name of the file containing the PISAFIND parameterisations. [PISAFIND.DAT]

GSIGM = _REAL (Read)
The standard deviation, in pixels, of the Gaussian core of the analytical stellar profile. [Global]

IN = NDF (Read)
The NDF containing the objects which have been parameterised by PISAFIND.

RESULTS = FILENAME (Write)
Name of a file to contain the PISAPEAK results. [PISAPEAK.DAT]

THRESH = _REAL (Read)
The detection threshold as used by PISAFIND. [Global]

Examples:

```
PISAPEAK IN=FRAME FINDDATA=PISAFIND.DAT RESULTS=PISAPEAK.DAT BACKGROUND=255
THRESH=12
```

This example shows PISAPEAK using the objects found in NDF FRAME with parameters stored in file PISAFIND.DAT. The results are stored in file PISAPEAK.DAT. Note that the model parameters GSIGM CROSS and COMIX are defaulted to those which have been stored in GLOBAL. These values were probably written by the routine PISAFIT which uses a list of stars to fit the PISA profiling function. The background value and threshold are those used by PISAFIND when detecting and parameterising the objects.

Notes:

- The model profile is used to scale the object values to those around unity. This only works well if the PISA profiling function is a good fit to the stars on your frame. If it is not a good fit or you cannot determine the model parameters then inaccurate values can be used, the only criterion being that the GSIGM value is about the FWHM seeing of your data. The results file will now contain values 'normalised' to this 'imaginary' object. The stars will still form a group of objects with similar values, although they will not be as tightly clustered as with a good fit.

PISAPLOT
Plots an ellipse map of the objects found by PISAFIND

Description:

This application shows the positions, shapes and sizes of the objects located by PISAFIND via a plot of ellipses, one per object. Optional annotation of the ellipses provide cross-referencing with the object list. The plot may be overlaid on an existing image for a direct comparison. In this case the plot appears within the last DATA picture in the graphics database, otherwise it is situated within the current picture.

Control of the plot allows labelled axes whose extent may be defined and the selection of the colour of the ellipses

Parameters:

ABSLAB = LITERAL (Read)
Label for the plot abscissa, may include PGPLOT escape sequences. This parameter is only used when the axes option is selected. [X]

ANNOTA = _LOGICAL (Read)
If true the ellipses are annotated with the object identification number to the top right. [TRUE]

ANNOSCALE = REAL (Read)
The scale height of the annotations. This value is a multiple of the normal text height. [1.0]

AXES = _LOGICAL (Read)
True if annotated axes are to be drawn around the displayed image. This parameter is ignored in the overlay mode, since there is no guarantee that the axes would lie entirely within the current picture. [TRUE]

CLEAR = _LOGICAL (Read)
True if the device is to be cleared first when in overlay mode. Useful if displaying different classes of objects successively. This flag acts as a switch retaining its last value. [FALSE]

DEFAXES = _LOGICAL (Read)
If set true then prompting for the axis bounds will occur, otherwise the program generated defaults will be used. [FALSE]

DEVICE = DEVICE (Read)
The name of the graphics device on which to plot the map of images found. If the overlay mode is required it is recommended that the image be displayed in KAPPA on an image display's base plane, then run this application using the device's overlay plane. [Current graphics device]

MAJTIC(2) = _REAL (Read)
The parameter controlling the numbers of major tick marks for the x and y axes. A negative value for an axis makes the graphics package decide an appropriate value. This parameter is only used when the axes option is selected. [-1.,-1.]

MINTIC(2) = _INTEGER (Read)
The number of minor tick marks between each major tick mark for the x and y axes. A negative value forces the graphics package to compute appropriate values. This parameter is only used when the axes option is selected. [-1.,-1.]

ORDLAB = LITERAL (Read)
Label for the plot ordinate, may include PGPLOT escape sequences. This parameter is only used when the axes option is selected. [Y]

OUTTIC = _LOGICAL (Read)
: True if the axis tick marks are to appear on the outside of the axes instead of inside. This parameter is only used when the axes option is selected. [TRUE]

OVERLAY = _LOGICAL (Read)
: True if the plot is to be overlaid on the last DATA picture in the graphics database. This is used to compare a displayed 2-d image with the objects detected. This flag acts as a switch retaining its last value. [FALSE]

PALNUM = _INTEGER (Read)
: PISAPLOT allows the user to specify which pen number to use when plotting. Thus different classifications of objects can be identified on the same plot using different colours. The colours associated with these pens are the default PGPLOT pens (see the PGPLOT manual for a complete description). These are:

0	background colour
1	foreground colour
2	red
3	green
4	blue
5	cyan
6	magenta
7	yellow
8	orange

and so on up to pen 16 (up to the number available on the current graphics device). After PISAPLOT has been run these colours can be superseded by using the KAPPA palette facilities PALDEF and PALENTRY, but note that any subsequent runs of PISAPLOT will reinstate the PGPLOT default colours so using the KAPPA facilities should be delayed until all object classifications have been displayed. The KAPPA palette pen numbers correspond to PALNUM values (hence the parameter name). [3]

PLTITL = CHAR (Read)
: The title of the plot, may include PGPLOT escape sequences. Up to about 40 characters can be accommodated. This parameter is only used when axes option is selected. [Images Detected]

RESULTS = FILENAME (Read)
: The ASCII file produced by the PISAFIND application containing the parameterised data for the various objects it detected. [PISAFIND.DAT]

THICK = _REAL (Read)
: The thickness of the lines in the plot, where 1.0 is the normal thickness. It should be between 0.5 and 5. This feature is only available on some devices. This parameter is only used when axes option is selected. [1.0]

XPIXS = _REAL (Read)
: Initial and final pixel coordinates of plot in x-direction. [Dynamic]

YPIXS = _REAL (Read)
: Initial and final pixel coordinates of plot in y-direction. [Dynamic]

Examples:

```
PISAPLOT OVERLAY
```

Using the overlay parameter allows ellipses to be plotted over a previously displayed image.

```
PISAPLOT OVERLAY CLEAR
```

Including the clear parameters clears a plot before overlaying. This is useful when displaying the overlaid ellipses in an overlay.

```
PISAPLOT RESULTS=STARS.DAT PALNUM=3 OVERLAY
```

Using this combination of parameters would use a green pen to plot the ellipse overlaying them on the displayed picture. Using sequences of this example with difference results files and palnums would result in a display which had different colours for each results file displayed over the current picture.

Notes:

- The following pictures are stored in the graphics database: a FRAME encompassing annotated axes and the plot itself, provided the overlay option is not selected; a DATA picture for the plot itself. The latter uses the standard Starlink co-ordinate system.

B The PISA profiling function

The PISA profiling function is made up of three functions, a Gaussian, an exponential and a Lorentzian. Inside a radius R_c fixed proportions of the Gaussian and Lorentzian functions are used, outside of R_c the exponential replaces the Gaussian and is added to the continuing Lorentzian. The exponential is joined smoothly to the Gaussian. Inside of R_c the function takes the form:-

$$\frac{1}{\pi\sigma^2(1+(\frac{\tau}{2\ln(\frac{1}{\tau})}))}\left(\frac{Q}{(1+\frac{r^2}{\sigma^2\ln(2)})}+(1-Q)\exp(\frac{-r^2}{\sigma^2})\right)$$

and outside of R_c it takes the form:-

$$\frac{1}{\pi\sigma^2(1+(\frac{\tau}{2\ln(\frac{1}{\tau})}))}\left(\frac{Q}{(1+\frac{r^2}{\sigma^2\ln(2)})}\frac{+(1-Q)\exp(\frac{-2r}{\sigma}\sqrt{\ln(\frac{1}{\tau})})}{\tau}\right)$$

where:-

$\tau =$ the fraction of the peak intensity at which to change from the gaussian to an exponential function (CROSS/100),

$\sigma =$ the gaussian function sigma (GSIGM),

$Q =$ the fraction of the Lorentzian function to add to the gaussian or exponential function at each point (COMIX).

The radius at which the exponential replaces the gaussian is:-

$$R_c = \sigma\sqrt{\ln(\frac{1}{\tau})}$$

The actual function is that which when multiplied by the **integrated** intensity gives the intensity at the given radius. Basically the functional forms from which the above equations are derived are:-

Gaussian — $\exp(\frac{-r^2}{\sigma^2})$

Lorentzian — $1/(1+\frac{r^2}{\sigma^2})$

Exponential — $\exp(-(\frac{r-R_c}{\sigma}))$.

C Limitations

The PISA package has various in-built limitations (buffer sizes etc.), This section details them; if they cause real problems then contact the supporter of this package.

- PISAFIND
  - The number of fragments per image (i.e. connected pixel groups) is limited to 200. Hence blends of objects cannot fragment into more than 200 pieces.
  - The number of pixels per image allowed during surface modelling is 10000.
  - The length of the first side of input image cannot be more than 10240 pixels.
  - The input data is processed using an HDS type of _WORD so the input data range must lie within the bounds -32768 to +32767, although PISAFIND itself only processes positive data up to the value 32766. This limitation is fundamental.
  - There are no other known restrictions. Input NDFs are now accessed via file mapping. There is no restriction on the number of detected objects.
- PISAFIT
  - As in PISAFIND the length of first side of the input image has to be less than 10240 pixels.
  - The input data is processed using an HDS type of _WORD so the input data range must lie within the bounds -32768 to +32767.
- PISAMATCH
  - The maximum number of input records is 10000.
- PISAKNN
  - The maximum number of input records is 10000.

D Changes in this release

- The ADDNOISE command has been changed so that a different sequence of random numbers is used each time it is started.